



UNIVERSIDAD
CATÓLICA
DE CUENCA

UNIVERSIDAD CATÓLICA DE CUENCA

Comunidad Educativa al Servicio del Pueblo

**FACULTAD DE INFORMÁTICA, CIENCIAS DE LA
COMPUTACIÓN E INNOVACIÓN TECNOLÓGICA**

CARRERA DE SOFTWARE

**Prototipo de sistema para el registro de notas y asistencia basado en
microservicios con integración SIGA-Moodle en el Instituto Superior
Tecnológico del Austro**

**PROYECTO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL
TÍTULO DE INGENIERO EN SOFTWARE**

AUTOR: NELSON EMMANUEL BRITO ORDÓÑEZ

DIRECTOR: ING. MILTON ALFREDO CAMPOVERDE MOLINA, PHD.

CUENCA - ECUADOR

2026

DIOS, PATRIA, CULTURA Y DESARROLLO



UNIVERSIDAD CATÓLICA DE CUENCA

Comunidad Educativa al Servicio del Pueblo

**FACULTAD DE INFORMÁTICA, CIENCIAS DE LA
COMPUTACIÓN**

E INNOVACIÓN TECNOLÓGICA

CARRERA DE SOFTWARE

Prototipo de sistema para el registro de notas y asistencia basado en
microservicios con integración SIGA-Moodle en el Instituto Superior
Tecnológico del Austro

**PROYECTO DE TITULACIÓN PREVIO A LA OBTENCIÓN
DEL TÍTULO DE INGENIERO DE SOFTWARE**

AUTOR: NELSON EMMANUEL BRITO ORDÓÑEZ

DIRECTOR: ING. MILTON ALFREDO CAMPOVERDE MOLINA, PHD.

CUENCA - ECUADOR

2026

DIOS, PATRIA, CULTURA Y DESARROLLO



Universidad
Católica
de Cuenca

DECLARATORIA DE AUTORÍA Y RESPONSABILIDAD

Declaratoria de Autoría y Responsabilidad

Nelson Emmanuel Brito Ordóñez, portador de la cédula de ciudadanía N.º 0107828261, declaro ser el autor de la obra: **“Prototipo de sistema para el registro de notas y asistencia basado en microservicios con integración SIGA-Moodle en el Instituto Superior Tecnológico del Austro”**, sobre la cual me hago responsable de las opiniones, versiones e ideas expresadas. Declaro que la misma ha sido elaborada respetando los derechos de propiedad intelectual de terceros y eximo a la Universidad Católica de Cuenca de cualquier reclamación que pudiera existir al respecto. Declaro finalmente que mi obra ha sido realizada cumpliendo con todos los requisitos legales, éticos y bioéticos de investigación, que la misma no incumple con la normativa nacional e internacional en el área específica de investigación, sobre la que también me responsabilizo y eximo a la Universidad Católica de Cuenca de toda reclamación al respecto.

Cuenca, 5 de Agosto del 2026

F.: Emmanuel Brito.....

Nelson Emmanuel Brito Ordóñez

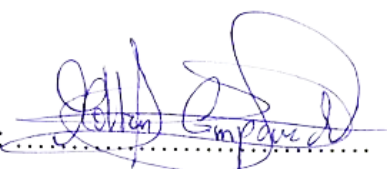
C.I. 0107828261



Universidad
Católica
de Cuenca

CERTIFICADO DEL TUTOR

Yo, Milton Alfredo Campoverde Molina, certifico que el presente trabajo de investigación, con el título **“Prototipo de sistema para el registro de notas y asistencia basado en microservicios con integración SIGA-Moodle en el Instituto Superior Tecnológico del Austro”**, fue desarrollado por Nelson Emmanuel Brito Ordóñez, con número de cédula 0107828261, bajo mi supervisión.

F. 

Ing. Milton Alfredo Campoverde Molina, PhD.

C.I. 0103902532

Dedicatoria

Este trabajo va dedicado como un reconocimiento al cariño, apoyo y comprensión en primer lugar de mis padres, Nelson Efrén Brito Alemán y Claudia Verónica Ordóñez Vázquez quienes siempre estuvieron a mi lado, con su apoyo incondicional, consejos y ejemplo que fue mi fuerza y mi motivación para seguir adelante y cumplir la meta anhelada. A cada miembro de mi familia quienes estuvieron cerca mientras este proyecto tomaba forma, con paciencia y sin dudar nunca del resultado. A todos y cada uno de los docentes que me formaron a lo largo de la carrera. A ustedes dedico con mucho cariño este logro.

Agradecimiento

Agradezco en primer lugar a Dios, por darme la fortaleza, el entendimiento, la capacidad, y la constancia para no decaer y seguir siempre adelante. A mi familia, en especial a mi padre quien me dio la fuerza, perseverancia, resiliencia, y consejos. A mi madre, por su infinito amor y dedicación, por estar siempre a mi lado con palabras de aliento y ánimo que fueron el pilar fundamental para conseguir este logro. A mis abuelos, por su constante preocupación y motivación que fueron muy importantes para mi formación como persona y en el futuro, un profesional honesto y responsable. Con gran cariño agradezco a mi tía Priscila Fernández y a mi tío Juan José Ordóñez, por su gran ayuda en mi vida universitaria, y constante seguimiento durante este proceso. A mi tío Alfredo Brito por su bondad y generosidad. A mi tutor, Ing. Milton Campoverde Molina, PhD., por su guía y su disposición en cada etapa del desarrollo de este trabajo. Al Instituto Superior Tecnológico del Austro, por abrirme sus puertas y permitirme trabajar con información real de su proceso académico, y en particular a las autoridades, coordinadores, docentes y personal de TIC que participaron en las entrevistas que dieron sustento a este trabajo, por su tiempo y su franqueza al describir el proceso que hoy busca mejorarse. A la Universidad Católica de Cuenca, a la Facultad de Informática, Ciencias de la Computación e Innovación Tecnológica, y al Centro de Ingeniería de Software, por la formación recibida durante toda la carrera. A mis amigos y compañeros, por el ánimo compartido en los momentos más difíciles del camino. A todos y cada uno, muchísimas gracias y que Dios y la vida les retribuya toda su bondad.

Resumen

Esta investigación tiene como objetivo desarrollar un prototipo de software funcional del Módulo de Registro de Notas y Asistencia (MRNA). El problema radica en que el proceso de registro de notas y asistencia se realiza de forma manual en el Instituto Superior Tecnológico del Austro (ISTA). Lo cual, obliga a los docentes a calificar en la plataforma Moodle, trasladar los datos a hojas de cálculo y volver a digitarlos en el sistema nacional SIGA. Esto los llevaba entre seis y ocho horas por cierre de parcial y un alto riesgo de errores de transcripción. El desarrollo del sistema web se llevó a cabo utilizando la metodología Kanban. El sistema web tiene una arquitectura de seis microservicios sobre Spring Boot y Java, una interfaz en *React* y una base de datos PostgreSQL. Además, se incorporó un adaptador dual para la integración del SIGA, considerando que este sistema no tiene una interfaz pública. En la validación de usabilidad se obtuvo una valoración de excelente (87,5/100) aplicado el cuestionario *System Usability Scale*. La evaluación de usabilidad se realizó con un docente experto. La prueba técnica se llevó a cabo con el registro de las notas de un parcial en el sistema web. Esto se realizó en pocos segundos frente al tiempo invertido del proceso manual. En conclusión, el sistema web permite el registro y la consolidación de calificaciones y asistencia de los estudiantes del ISTA.

Palabras clave: *microservicios, integración de sistemas, Moodle, SIGA, registro de calificaciones.*

Abstract

This research aims to develop a functional software prototype of the Grade and Attendance Recording Module (GARM). The problem lies in the fact that the process of recording grades and attendance is carried out manually at the Instituto Superior Tecnológico del Austro (ISTA by its Spanish acronym). This requires professors to enter grades into the Moodle platform, transfer the data to spreadsheets, and then re-enter it into the national academic system (SIGA by its Spanish acronym). This process takes between six and eight hours at the end of each grading period and involves a high risk of transcription errors. The web system was developed using the Kanban methodology. The system has a six-microservice architecture built with Spring Boot and Java, a React-based user interface, and a PostgreSQL database. In addition, a dual adapter was implemented to integrate with SIGA, considering that this system does not provide a public interface. In the usability validation, the system received an excellent rating (87.5/100) using the System Usability Scale (SUS) questionnaire. The usability evaluation was conducted with an expert instructor. The technical test consisted of recording the grades for one grading period in the web system. This process was completed in a few seconds compared with the time required for the manual process. In conclusion, the web system enables the recording and consolidation of student grades and attendance at ISTA.

Keywords: *microservices, systems integration, Moodle, SIGA, grade recording.*

Índice General

Introducción	1
Capítulo 1. Fundamentación Teórica	2
1.1. Introducción del capítulo	2
1.2. Conceptos relacionados	2
1.2.1. Sistemas de gestión académica en la educación superior ecuatoriana.....	2
1.2.2. Plataformas LMS: Moodle como caso institucional.....	3
1.2.3. Sistema Integrado de Gestión Académica (SIGA-SENESCYT).....	4
1.2.4. Arquitectura de microservicios: principios, patrones y beneficios	5
1.2.5. Integración de sistemas: APIs REST, sincronización bidireccional y patrones de adaptador.....	7
1.2.6. Bases de datos relacionales y PostgreSQL	9
1.2.7. Autenticación moderna: JWT, refresh tokens y Spring Security.....	10
1.2.8. Metodologías ágiles para proyectos de un solo desarrollador: Kanban.....	11
1.2.9. Calidad de software y testing automatizado	13
1.2.10. Frameworks del stack: Spring Boot, React y TailwindCSS	14
1.3. Estado del arte / Trabajos relacionados	15
1.4. Marco normativo institucional	17
1.5. Síntesis y aporte del capítulo	19
Capítulo 2. Diagnóstico Situacional	21
2.1. Introducción del capítulo	21
2.2. Contexto institucional	21
2.3. Planteamiento del problema.....	23
2.4. Justificación	26
2.4.1. Importancia teórica	26
2.4.2. Importancia práctica.....	26
2.4.3. Resultados esperados	27

2.4.4. Factibilidad	28
2.5. Objetivos	28
2.5.1. Objetivo general.....	28
2.5.2. Objetivos específicos	28
2.6. Preguntas de investigación.....	29
2.7. Alcance del prototipo	30
2.8. Análisis del proceso actual.....	33
2.9. Levantamiento y análisis de requerimientos.....	36
2.9.1. Stakeholders entrevistados.....	36
2.9.2. Catálogo de requerimientos funcionales	36
2.9.3. Matriz de trazabilidad RF ↔ entrevistas ↔ historias de usuario	37
2.9.4. Catálogo de requerimientos no funcionales	38
2.10. Metodología de la investigación	40
2.10.1. Tipo y nivel de investigación.....	40
2.10.2. Enfoque y diseño metodológico.....	41
2.10.3. Población y muestra.....	41
2.10.4. Instrumentos y validación	42
2.10.5. Medición de variables del estudio.....	43
2.11. Metodología de desarrollo de software.....	44
2.12. Síntesis del capítulo	46
Capítulo 3. Desarrollo y Validación de la Propuesta.....	48
3.1. Introducción del capítulo	48
3.2. Visión general de la solución MRNA.....	48
3.3. Arquitectura del sistema	50
3.3.1. Vista lógica de microservicios	50
3.3.2. Vista de despliegue	53
3.3.3. Patrones arquitectónicos aplicados	55

3.4. Stack tecnológico	57
3.5. Diseño de la base de datos	60
3.5.1. Modelo entidad-relación	60
3.5.2. Diccionario de datos	62
3.5.3. Estrategia de migraciones con Flyway	64
3.6. Diseño de los microservicios	66
3.6.1. auth-service	66
3.6.2. academic-service	67
3.6.3. sync-service	67
3.6.4. report-service	68
3.6.5. notification-service	68
3.6.6. api-gateway	69
3.7. Diseño de la API REST	69
3.8. Diseño de la interfaz de usuario	71
3.8.1. Sistema de diseño	71
3.8.2. Pantallas del prototipo por rol	72
3.8.3. Flujos de usuario clave	74
3.9. Estrategia de seguridad	76
3.10. Estrategia de integración	77
3.10.1. Integración con Moodle local	77
3.10.2. Adaptador SIGA dual	79
3.11. Cumplimiento de Requerimientos No Funcionales en la arquitectura	81
3.12. Aplicación de la metodología Kanban	83
3.13. Implementación del prototipo	86
3.14. Validación funcional del prototipo	87
3.14.1. Estrategia de validación	87
3.14.2. Plan de pruebas por requerimiento funcional	88

3.14.3. Pruebas unitarias	90
3.14.4. Pruebas de integración entre microservicios.....	91
3.14.5. Pruebas end-to-end de los flujos críticos	92
3.15. Síntesis del capítulo	92
Capítulo 4. Resultados y Discusión	94
4.1. Introducción del capítulo	94
4.2. Validación de usabilidad.....	94
4.2.1. Protocolo de validación.....	94
4.2.2. Población participante.....	95
4.2.3. Resultados	95
4.3. Resultados de desempeño y comparativa proceso manual vs. MRNA.....	98
4.4. Cumplimiento de los Requerimientos No Funcionales	101
4.4.1. RNF de seguridad	101
4.4.2. RNF de fiabilidad y disponibilidad.....	102
4.4.3. RNF de mantenibilidad	103
4.4.4. RNF de portabilidad.....	103
4.4.5. RNF de usabilidad.....	104
4.5. Resultados de la metodología Kanban.....	106
4.5.1. Métricas obtenidas	106
4.5.2. Análisis cuantitativo del flujo	107
4.6. Discusión de resultados.....	109
4.7. Limitaciones del estudio	111
4.8. Síntesis del capítulo	113
Capítulo 5. Conclusiones y Recomendaciones	114
5.1. Conclusiones por objetivo específico	114
5.2. Aportes del trabajo.....	115
5.3. Recomendaciones	116

5.4. Trabajo futuro	117
Referencias Bibliográficas	119

Índice de Tablas

<i>Tabla 1. Comparativa de antecedentes frente a MRNA. Nota. Elaboración propia, 2026.</i>	<i>16</i>
<i>Tabla 2. Matriz MoSCoW y estado de avance del alcance del prototipo MRNA. Nota. Elaboración propia — Consolidado de Requerimientos Funcionales e Historias de Usuario, 2026.....</i>	<i>32</i>
<i>Tabla 3. Problemáticas identificadas en el proceso actual de registro de notas y asistencia del ISTA. Nota. Elaboración propia con base en entrevistas a docentes, coordinadores y autoridades del Instituto, 2025.</i>	<i>35</i>
<i>Tabla 4. Matriz de trazabilidad de los requerimientos funcionales core del MRNA. Nota. Elaboración propia con base en DOC-06 y entrevistas a stakeholders, 2025.....</i>	<i>37</i>
<i>Tabla 5. Catálogo de requerimientos no funcionales del MRNA. Nota. Elaboración propia con base en DOC-08 — Levantamiento de Requerimientos No Funcionales, 2026.....</i>	<i>39</i>
<i>Tabla 6. Población y muestra del estudio. Nota. Elaboración propia, 2026.</i>	<i>42</i>
<i>Tabla 7. Medición de variables del estudio. Nota. Elaboración propia, 2026.....</i>	<i>43</i>
<i>Tabla 8. Microservicios del prototipo MRNA, responsabilidades y puertos de desarrollo. Nota. Elaboración propia, 2026.....</i>	<i>51</i>
<i>Tabla 9. Tecnologías seleccionadas para el desarrollo del prototipo MRNA: versión, capa y función por componente. Nota. Elaboración propia — Archivos de configuración del repositorio tesis-mrna (pom.xml, package.json, docker-compose.yml), 2026.....</i>	<i>58</i>
<i>Tabla 10. Diccionario de datos de la entidad nota. esquema PostgreSQL del repositorio tesis-mrna. Nota. Elaboración propia — esquema PostgreSQL del repositorio tesis-mrna de la tabla nota, 2026</i>	<i>63</i>
<i>Tabla 11. Secuencia de migraciones Flyway del esquema MRNA. Nota. Elaboración propia — repositorio tesis-mrna, 2026.....</i>	<i>64</i>
<i>Tabla 12. Códigos de estado HTTP devueltos por la API REST del MRNA. Nota. Elaboración propia — repositorio tesis-mrna, 2026.....</i>	<i>71</i>
<i>Tabla 13. Comparativa de las tres alternativas evaluadas para la integración con SIGA. Nota. Elaboración propia, 2026.....</i>	<i>80</i>
<i>Tabla 14. Mapeo de requerimientos no funcionales a sus decisiones arquitectónicas. Nota. Elaboración propia, 2026.....</i>	<i>81</i>
<i>Tabla 15. Columnas del tablero Kanban del proyecto MRNA y su propósito. Nota. Elaboración propia con base en ADR-004, 2026.....</i>	<i>84</i>

<i>Tabla 16. Dimensiones de la estrategia de validación del prototipo MRNA. Nota. Elaboración propia, 2026.</i>	88
<i>Tabla 17. Plan de pruebas funcionales por requerimiento del prototipo MRNA. Nota. Elaboración propia, 2026.</i>	89
<i>Tabla 18. Cobertura de pruebas unitarias por microservicio. Nota. Elaboración propia a partir de los reportes JaCoCo del proyecto, 2026.</i>	91
<i>Tabla 19. Resultados de las tareas guiadas de usabilidad del prototipo MRNA. Nota. Elaboración propia a partir de la sesión de validación con el Docente de Desarrollo de Software, 2026.</i>	96
<i>Tabla 20. Métricas de desempeño del prototipo MRNA bajo carga. Nota. Elaboración propia a partir de la prueba de carga con k6, 2026.</i>	99
<i>Tabla 21. Comparativa entre el proceso manual y el prototipo MRNA. Nota. Elaboración propia; los tiempos manuales provienen del diagnóstico del apartado 2.8, 2026.</i>	100
<i>Tabla 22. Cumplimiento de los requerimientos no funcionales del prototipo MRNA, por aspecto. Nota. Elaboración propia, 2026.</i>	104
<i>Tabla 23. Métricas de flujo Kanban del proyecto MRNA por cadencia. Nota. Elaboración propia a partir de la exportación del tablero "MRNA - Tesis Kanban" en GitHub Projects (corte al cierre de la Cadencia 5), 2026.</i>	106

Índice de Ilustraciones

<i>Figura 1. Flujo actual del proceso de registro de notas y asistencia en el ISTA. Nota. Elaboración propia con asistencia de inteligencia artificial, con base en entrevistas a docentes y coordinadores del Instituto, 2025.</i>	<i>25</i>
<i>Figura 2. Tablero Kanban del proyecto MRNA configurado en GitHub Projects para la gestión del desarrollo. Nota. Captura propia, 2026.</i>	<i>45</i>
<i>Figura 3. Diagrama de contexto del sistema MRNA (C4 nivel 1). Nota. Elaboración propia con asistencia de inteligencia artificial, 2026.</i>	<i>50</i>
<i>Figura 4. Vista lógica de la arquitectura de microservicios del MRNA (C4 nivel 2 – contenedores). Nota. Elaboración propia con asistencia de inteligencia artificial, 2026.</i>	<i>53</i>
<i>Figura 5. Diagrama de despliegue del entorno de desarrollo MRNA (C4 nivel 3). Nota. Elaboración propia con asistencia de inteligencia artificial, 2026.</i>	<i>55</i>
<i>Figura 6. Diagrama entidad-relación del esquema de base de datos MRNA agrupado por módulos funcionales. Nota. Elaboración propia con asistencia de inteligencia artificial, 2026.</i>	<i>62</i>
<i>Figura 7. Pantalla de notas por paralelo del prototipo MRNA. Nota. Captura del prototipo, elaboración propia, 2026.</i>	<i>73</i>
<i>Figura 8. Pantalla de descarga de reportes (roles coordinador, administrador y directivo), con la opción de exportación hacia el SIGA (exclusiva de coordinador y administrador). Nota. Captura del prototipo, elaboración propia, 2026.</i>	<i>73</i>
<i>Figura 9. Pantalla de carga masiva de docentes (rol administrador). Nota. Captura del prototipo, elaboración propia, 2026.</i>	<i>74</i>
<i>Figura 10. Pantalla de importación SIGA, mecanismo legado conservado por continuidad con la validación de usabilidad (roles coordinador y administrador). Nota. Captura del prototipo, elaboración propia, 2026.</i>	<i>74</i>
<i>Figura 11. Flujos clave de interacción entre los roles docente, coordinador/administrador y el sistema MRNA. Elaboración propia con asistencia de inteligencia artificial.</i>	<i>76</i>
<i>Figura 12. Secuencia de sincronización Moodle → MRNA. Elaboración propia con asistencia de inteligencia artificial.</i>	<i>79</i>
<i>Figura 13. Diagrama del adaptador dual SIGA con dos modos detrás de una misma interfaz interna. Elaboración propia con asistencia de inteligencia artificial.</i>	<i>81</i>
<i>Figura 14. Captura del tablero Kanban al cierre de la Cadencia 3. Captura del prototipo, elaboración propia, 2026.</i>	<i>86</i>

<i>Figura 15. Puntuación por ítem del cuestionario SUS aplicado al prototipo MRNA (contribución normalizada 0–4). Nota. Elaboración propia, 2026.....</i>	<i>97</i>
<i>Figura 16. Histograma de latencias (p50, p95, p99) por endpoint del prototipo MRNA. Nota. Elaboración propia, 2026.....</i>	<i>100</i>
<i>Figura 17. Diagrama de Flujo Acumulado (CFD) del proyecto MRNA, de la Cadencia 1 a la Cadencia 5. Nota. Elaboración propia a partir del tablero en GitHub Projects, 2026.....</i>	<i>107</i>

Lista de acrónimos y siglas

La siguiente tabla enumera los acrónimos y siglas utilizados a lo largo del documento, ordenados alfabéticamente.

Sigla	Significado
ADR	Architecture Decision Record
API	Application Programming Interface
APA	American Psychological Association
BPMN	Business Process Model and Notation
CFD	Cumulative Flow Diagram
CRUD	Create, Read, Update, Delete
DDL	Data Definition Language
DTO	Data Transfer Object
E-R	Entidad-Relación
HU	Historia de Usuario
ISTA	Instituto Superior Tecnológico del Austro
JPA	Java Persistence API
JWT	JSON Web Token
LMS	Learning Management System
LOES	Ley Orgánica de Educación Superior
MoSCoW	Must / Should / Could / Won't have
MRNA	Módulo de Registro de Notas y Asistencia
RBAC	Role-Based Access Control
REST	Representational State Transfer
RF	Requerimiento Funcional
RNF	Requerimiento No Funcional
SaaS	Software as a Service
SENESCYT	Secretaría de Educación Superior, Ciencia, Tecnología e Innovación
SIGA	Sistema Integrado de Gestión Académica
SUS	System Usability Scale
TIC	Tecnologías de la Información y la Comunicación
UCACUE	Universidad Católica de Cuenca

Sigla	Significado
UI	User Interface
UX	User Experience
WIP	Work In Progress

Introducción

La transformación digital de la educación superior ha cambiado la forma en que las instituciones gestionan su información académica. Las plataformas de aprendizaje en línea, los sistemas internos de gestión y las plataformas nacionales de registro conviven hoy en la mayoría de institutos y universidades, pero rara vez lo hacen de manera integrada. Esa fragmentación, en la que cada sistema guarda su propia versión de los mismos datos, obliga al personal docente y administrativo a mover información de una plataforma a otra de forma manual, con el costo de tiempo y de errores que ello implica. El Instituto Superior Tecnológico del Austro es un caso concreto de esa situación. Sus docentes califican en el aula virtual Moodle, consolidan esas notas en hojas de cálculo Excel para las juntas académicas y, por último, las vuelven a digitar en el SIGA, el sistema nacional de la SENESCYT. Esta triple digitación de la misma información, además de repetir un trabajo que podría automatizarse, consume una parte considerable del tiempo de docentes y coordinadores en cada cierre de parcial y abre la puerta a errores de transcripción que después son difíciles de rastrear.

Frente a ese problema, este proyecto de titulación desarrolló un prototipo funcional de sistema web, el Módulo de Registro de Notas y Asistencia (MRNA), que automatiza el registro y la consolidación de notas entre Moodle y el MRNA, genera el archivo listo para la carga en el SIGA y elimina la digitación repetida. El prototipo se construyó con una arquitectura de microservicios sobre Spring Boot y Java para el backend, React para la interfaz y PostgreSQL como base de datos, y su desarrollo se organizó con la metodología Kanban, que permitió medir el flujo de trabajo a lo largo del proyecto. La integración con el SIGA se resolvió mediante un adaptador dual, dado que el sistema nacional no expone una interfaz pública.

El documento se organiza en cinco capítulos. El primero presenta la fundamentación teórica sobre la que se apoya la propuesta. El segundo describe el diagnóstico de la situación actual del Instituto y el levantamiento de requerimientos. El tercero detalla el desarrollo y la validación de la propuesta, desde la arquitectura hasta las pruebas del prototipo. El cuarto expone los resultados obtenidos y su discusión. El quinto reúne las conclusiones por cada objetivo y plantea las recomendaciones y el trabajo futuro.

Capítulo 1. Fundamentación Teórica

1.1. Introducción del capítulo

El presente capítulo desarrolla el sustento conceptual y normativo sobre el que se apoya el prototipo MRNA. Para cumplir ese propósito, el capítulo se organiza en cuatro secciones complementarias. La primera expone el marco teórico, que comprende los fundamentos de la arquitectura de microservicios, los mecanismos de integración entre sistemas mediante API REST, las tecnologías del stack seleccionado (Spring Boot, React, PostgreSQL) y las herramientas de autenticación, documentación, gestión de proyectos y pruebas de software empleadas en el desarrollo. La segunda sección presenta el estado del arte, donde se analizan investigaciones recientes de alcance comparable que permiten identificar la brecha que el prototipo busca cerrar en el contexto de la educación superior ecuatoriana. La tercera sección describe el marco legal y normativo que regula el registro académico en los institutos superiores del país, con especial atención a las obligaciones que imponen la Ley Orgánica de Educación Superior y los instrumentos del Consejo de Educación Superior en relación con el Sistema SIGA. El capítulo cierra con una síntesis que consolida los aportes teóricos y conecta el fundamento construido con la propuesta de solución desarrollada en los capítulos siguientes.

1.2. Conceptos relacionados

1.2.1. Sistemas de gestión académica en la educación superior ecuatoriana

Un sistema de gestión académica es, en términos generales, una plataforma de software que centraliza los procesos administrativos asociados a la vida académica de una institución: matrícula de estudiantes, registro de calificaciones, control de asistencia, gestión de horarios y elaboración de reportes institucionales. La definición, sin embargo, varía bastante según el tipo de institución y el marco regulatorio en el que opera. En el contexto de la educación superior ecuatoriana, un SGA no puede comprenderse sin referencia al entorno normativo que lo condiciona, porque son precisamente las exigencias legales las que determinan qué información debe gestionar el sistema, con qué frecuencia debe actualizarse y ante quién debe rendir cuenta [1].

En Ecuador, el Consejo de Educación Superior y la SENESCYT establecen los lineamientos que toda institución de educación superior pública debe seguir en la gestión

de sus registros académicos. La Ley Orgánica de Educación Superior impone, en sus artículos 93 y 96, la obligación de garantizar la calidad académica y el acceso transparente de los estudiantes a sus registros de calificaciones, lo que implica que los sistemas utilizados para ese fin deben ser seguros, auditables y accesibles [2]. El Reglamento de Régimen Académico del CES, en su artículo 67, va más allá y establece criterios específicos sobre las categorías de evaluación, los plazos de cierre de período y las condiciones para reportar resultados académicos [3]. Estos artículos no son simples referencias contextuales: constituyen el marco de requisitos funcionales implícitos que cualquier sistema de gestión académica debe satisfacer si quiere operar dentro de la legalidad en el sistema ecuatoriano.

Para los institutos superiores tecnológicos públicos, ese marco se vuelve todavía más restrictivo, porque la SENESCYT no sólo regula, sino que administra directamente el sistema de registro oficial: el SIGA. Todos los institutos públicos del país están obligados a reportar su información académica a través de esa plataforma, que actúa como repositorio centralizado desde el que la SENESCYT ejerce su función de control y supervisión del sistema [4]. El ISTA no es una excepción a esta regla. En ese sentido, hablar de gestión académica en el Instituto implica hablar de dos sistemas que coexisten con lógicas distintas: el Moodle institucional, orientado al proceso pedagógico, y el SIGA, orientado al reporte regulatorio. La tensión entre ambos es, en buena medida, la razón de ser del prototipo MRNA.

1.2.2. Plataformas LMS: Moodle como caso institucional

Una plataforma de gestión del aprendizaje, conocida por sus siglas en inglés como LMS (Learning Management System), es un sistema de software diseñado para administrar, distribuir y evaluar contenido educativo en entornos digitales. En el contexto de la educación superior, los LMS cumplen un rol que va más allá del aula virtual: integran la gestión de actividades académicas, el registro de calificaciones, el control de asistencia y la comunicación entre docentes y estudiantes dentro de un mismo entorno. Un estudio publicado en 2024 comparó cuarenta y cinco plataformas LMS para educación superior aplicando seis criterios (interoperabilidad, accesibilidad, herramientas de productividad, herramientas de comunicación, herramientas de aprendizaje y estándares de seguridad). En esa comparación, Moodle quedó en el segundo lugar con una puntuación de 9,25 sobre 10, apenas detrás de la plataforma Paradiso, que obtuvo 9,50 [5]. Esta puntuación corresponde a los criterios concretos que definieron los propios autores del estudio. No

es una medición universal de calidad, sin embargo sí confirma que Moodle está entre las plataformas mejor evaluadas de ese conjunto, lo que respalda su uso extendido como plataforma de código abierto en instituciones de educación superior, incluyendo el sistema ecuatoriano.

En el ISTA, Moodle opera bajo el nombre institucional de E-Virtual y es administrado por el área de TIC de la institución. Para este proyecto se utiliza una instancia local de Moodle 4.1.14+ desplegada en Docker, que replica las condiciones de la plataforma institucional sin requerir acceso a las credenciales del servidor de producción. Esta decisión responde a la necesidad de desarrollar y probar los mecanismos de integración en un entorno controlado, y está documentada en el ADR-003 del proyecto. La elección de Moodle como plataforma fuente del MRNA no es una preferencia técnica: es la plataforma que el ISTA ya usa, y cualquier prototipo que no se integre con ella carecería de pertinencia institucional real.

El componente técnico que hace posible esa integración es el protocolo de servicios web que Moodle expone de forma nativa. A través de su API REST, Moodle permite que sistemas externos consulten y extraigan datos académicos (calificaciones, asistencia, usuarios matriculados, actividades) mediante tokens de autenticación y llamadas HTTP estándar [6]. Esta capacidad es la que el sync-service del MRNA utiliza para sincronizar automáticamente la información desde Moodle hacia la base de datos central del sistema, eliminando la necesidad de que el docente exporte manualmente los datos. El flujo de sincronización construido sobre esta API es, técnicamente, la pieza que hace viable la propuesta de automatización que justifica todo el proyecto.

1.2.3. Sistema Integrado de Gestión Académica (SIGA-SENESCYT)

El Sistema Integrado de Gestión Académica, conocido por su acrónimo SIGA, es la plataforma oficial de registro académico administrada por la Secretaría de Educación Superior, Ciencia, Tecnología e Innovación del Ecuador. Su uso no es opcional: todos los institutos superiores tecnológicos públicos del país están obligados a reportar en el SIGA los datos de matrícula, calificaciones y asistencia de sus estudiantes, en los formatos y plazos que SENESCYT define [4]. En la práctica, esto convierte al SIGA en el repositorio oficial del sistema nacional de educación superior técnico-tecnológica pública, y cualquier gestión académica que un instituto realice en sus propias plataformas internas debe terminar, necesariamente, reflejada en ese sistema.

Lo que el SIGA no ofrece, y que resulta determinante para el diseño del MRNA, es un mecanismo de integración programática con sistemas externos. No existe una API pública documentada que permita a aplicaciones de terceros consultar o escribir datos en el SIGA mediante llamadas automatizadas. Esta limitación fue confirmada directamente por el responsable de TIC del Instituto, quien señaló que no hay ningún canal de integración formal disponible para el ISTA y que la única forma de cargar información al sistema es a través de la interfaz web o mediante la importación manual de archivos (Responsable de TIC, comunicación personal, 2025). Este testimonio describe la experiencia concreta del ISTA con el SIGA, no una auditoría oficial del sistema completo a nivel nacional. El presente trabajo toma esta restricción como un hecho confirmado para el caso del ISTA, que es el alcance real de este proyecto, y no la presenta como una condición demostrada para todos los institutos del país, aunque la naturaleza centralizada del SIGA hace razonable suponer que la situación puede repetirse en otras instituciones similares. Para el alcance del ISTA, esta restricción descarta cualquier solución basada en integración directa en tiempo real.

Esta realidad tiene una consecuencia arquitectónica directa sobre el prototipo. El componente del MRNA responsable de la interfaz con el SIGA (el sync-service) no puede operar mediante conexión directa al sistema nacional, sino que debe implementar lo que en el proyecto se denomina un adaptador dual: por un lado, genera archivos en formato XLSX con la estructura que el SIGA acepta para su importación manual, permitiendo que el coordinador descargue el archivo y lo cargue al sistema sin necesidad de reingreso de datos; por otro, expone una interfaz de simulación que replica el comportamiento esperado de una API para efectos de demostración del prototipo. Esta decisión, documentada en el ADR-003, no es una limitación del diseño sino la respuesta técnica más honesta a las restricciones del entorno institucional real en el que el sistema va a operar.

1.2.4. Arquitectura de microservicios: principios, patrones y beneficios

La arquitectura de microservicios es un estilo de diseño de software en el que una aplicación se estructura como un conjunto de servicios pequeños e independientes, cada uno ejecutable de forma autónoma, responsable de una capacidad de negocio específica y comunicable con los demás a través de interfaces bien definidas, generalmente APIs REST [7]. A diferencia de los enfoques tradicionales en los que toda la lógica del sistema convive dentro de un único proceso desplegable, los microservicios distribuyen esa lógica

entre unidades que pueden desarrollarse, probarse, desplegarse y escalarse de manera independiente. Esta independencia no es sólo técnica: implica que cada servicio puede evolucionar a su propio ritmo sin obligar a redespigar el sistema completo, lo que reduce el riesgo asociado a cada cambio y facilita el mantenimiento a largo plazo.

Para el MRNA, esta arquitectura se materializa en seis servicios independientes que se comunican a través de un API Gateway centralizado: auth-service, academic-service, sync-service, report-service, notification-service y el propio api-gateway. Cada uno implementa patrones de diseño reconocidos en la arquitectura de software. En su forma ideal, el patrón Database per Service plantea que cada microservicio tenga su propia base de datos independiente. En el prototipo MRNA, en cambio, los seis servicios comparten una misma base de datos PostgreSQL, como regla general, cada servicio trabaja solo con las tablas de su propio dominio, con una excepción documentada, el sync-service, que por la naturaleza de su función de sincronización lee y escribe directamente sobre el esquema de notas e inscripciones que pertenece a academic-service (se detalla en la sección 3.6.3). Esta decisión, documentada en el ADR-001, simplifica la instalación y el mantenimiento del prototipo, separar físicamente las bases de datos y cerrar esa excepción se plantea como mejora futura (sección 5.4). El patrón API Gateway concentra en un único punto de entrada todas las peticiones externas, simplificando la autenticación, el enrutamiento y el control de acceso sobre los cuatro roles institucionales del sistema. El patrón Adapter se aplica en el sync-service para encapsular las particularidades del protocolo de Moodle y del formato de archivo del SIGA detrás de interfaces uniformes, de modo que el resto del sistema no depende de los detalles específicos de esas plataformas externas.

Se eligió esta estructura para el MRNA tras compararla cuidadosamente con otras opciones. Una revisión sistemática de 35 estudios sobre la transición de sistemas monolíticos a microservicios identifica que los monolitos clásicos, aunque más simples de desarrollar en etapas iniciales, concentran toda la lógica en un único proceso que se vuelve progresivamente difícil de mantener, escalar y modificar de forma incremental [8]. El monolito modular mejora esa situación al separar la lógica en módulos internos, pero sigue compartiendo un proceso de despliegue único que limita la flexibilidad operativa. Los microservicios, por el contrario, permiten que cada capacidad del sistema evolucione de manera independiente, algo especialmente relevante en un contexto como el del ISTA, donde la integración con plataformas externas heterogéneas (Moodle y SIGA) exige que cada canal de integración pueda modificarse o reemplazarse sin afectar al núcleo de gestión académica.

1.2.5. Integración de sistemas: APIs REST, sincronización bidireccional y patrones de adaptador

Para entender cómo se comunican las distintas piezas del MRNA conviene partir de una idea sencilla, la del modelo cliente-servidor. En este modelo hay un programa que pide algo (el cliente) y otro programa que lo atiende (el servidor), y los dos se conectan a través de una red sin necesidad de conocer cómo funciona el otro por dentro. En el MRNA esto se ve todos los días. La pantalla que usa el docente, hecha en React, es el cliente, y el conjunto de microservicios que responde a esa pantalla es el servidor. El mismo patrón se repite cuando el sync-service le pide información a Moodle, ahí es el MRNA el que actúa como cliente frente a Moodle.

Para que un cliente y un servidor puedan entenderse hace falta un lenguaje común, y ese lenguaje es lo que se conoce como API (interfaz de programación de aplicaciones). El MRNA usa el estilo REST para construir sus APIs. Con REST, cada cosa que el sistema puede hacer se representa como un recurso con su propia dirección web (por ejemplo `/api/academic/notas`), y para trabajar con ese recurso se usan palabras estándar de internet como GET para consultar información, POST para crear algo nuevo, y PUT o PATCH para modificar lo que ya existe. La ventaja de trabajar así es que el cliente no necesita saber cómo está armado el servidor por dentro, solo necesita conocer esas direcciones y esas palabras. Un estudio con más de cien participantes de la industria y la universidad probó que seguir buenas prácticas al diseñar una API REST hace una diferencia real, once de doce reglas de buen diseño evaluadas hicieron que los desarrolladores entendieran mejor y más rápido cómo usar esa API [9]. Ese hallazgo es la razón por la que cuidar el diseño de la API propia del MRNA, que se explica con más detalle en la sección 3.7, no es un detalle menor sino una decisión que afecta directamente qué tan fácil es trabajar con el sistema.

La integración de sistemas es el proceso mediante el cual dos o más aplicaciones de software independientes intercambian datos y coordinan sus operaciones a través de interfaces definidas, sin que ninguna de ellas deba conocer los detalles internos de la otra. En el contexto de los sistemas de información académicos, esta integración es especialmente relevante porque las instituciones de educación superior operan habitualmente con varias plataformas heterogéneas que gestionan distintos aspectos del proceso educativo y que, sin un mecanismo de conexión, obligan al personal a trasladar datos manualmente entre ellas [10]. El caso del ISTA es un ejemplo concreto de ese

problema: Moodle gestiona el proceso pedagógico, el SIGA centraliza el reporte regulatorio, y entre ambos no existe canal automatizado alguno.

El mecanismo de integración que el MRNA adopta para conectarse con Moodle es la API REST (Representational State Transfer). A través de este protocolo, Moodle expone un conjunto de funciones que permiten a sistemas externos autenticarse mediante tokens y realizar llamadas HTTP para consultar o extraer datos académicos en formato JSON [6]. La arquitectura REST es especialmente adecuada para este tipo de integración porque es sin estado, lo que significa que cada llamada es autónoma y no depende del historial de interacciones previas, simplificando la lógica de sincronización. El sync-service del MRNA consume esta API en ciclos programados para extraer calificaciones y registros de asistencia desde Moodle y consolidarlos en la base de datos central del sistema, evitando la intervención manual del docente en ese traslado.

La sincronización entre sistemas plantea, sin embargo, un problema que va más allá de la simple lectura de datos: el manejo de conflictos cuando la misma información puede haber sido modificada en más de un sistema entre dos ciclos de sincronización. Para el MRNA, la estrategia adoptada define a Moodle como la fuente de verdad para los datos de calificaciones generados en actividades virtuales, de modo que en caso de discrepancia el sistema privilegia el dato de Moodle sobre cualquier versión previa almacenada localmente. Esta política de resolución es consistente con lo que [11] documentaron en un caso de integración entre Moodle y un sistema de información académico mediante RESTful API: la definición explícita de la fuente primaria de cada tipo de dato es la medida más efectiva para evitar inconsistencias en sincronizaciones periódicas.

Para el caso del SIGA, donde la integración directa vía API no es posible, el MRNA implementa el patrón Adapter. Este patrón, descrito por [7] como uno de los mecanismos centrales de integración en arquitecturas de microservicios, consiste en encapsular la lógica específica de comunicación con un sistema externo detrás de una interfaz uniforme, de modo que el resto del sistema interactúa con esa interfaz sin depender de los detalles del protocolo subyacente. En la práctica, el adaptador del MRNA para el SIGA traduce los datos académicos consolidados al formato de archivo XLSX que el sistema nacional admite para su importación manual, aislando al núcleo del sistema de cualquier cambio futuro en ese formato.

1.2.6. Bases de datos relacionales y PostgreSQL

La elección del motor de base de datos para un sistema de gestión académica no es una decisión aleatoria. Los datos que maneja el MRNA (calificaciones, registros de asistencia, períodos académicos, usuarios institucionales) tienen un nivel que impone requisitos claros sobre cómo deben almacenarse y modificarse. Los sistemas gestores de bases de datos relacionales garantizan las propiedades ACID en cada transacción: atomicidad, consistencia, aislamiento y durabilidad. Esto significa que una operación de registro de notas o cierre de período se ejecuta completa o no se ejecuta en absoluto, que el estado de la base de datos siempre es coherente, y que los datos confirmados persisten incluso ante fallos del sistema [12]. Las bases de datos NoSQL no son mejores ni peores en términos generales, ofrecen otro tipo de compromiso. Ganan en flexibilidad para guardar datos que no siguen una estructura fija y en la facilidad para repartir la carga entre varios servidores, pero varias de ellas relajan a propósito alguna de las garantías ACID para lograr eso, sobre todo cuando muchas escrituras ocurren al mismo tiempo [12]. Existen motores NoSQL más recientes que sí ofrecen transacciones completas, así que la elección no depende del nombre de la categoría sino del motor concreto y de lo que cada proyecto necesita. Para el MRNA, donde una nota mal registrada o un cierre de período incompleto tiene consecuencias académicas directas sobre un estudiante, esa garantía transaccional completa fue el criterio que más pesó al elegir un motor relacional.

PostgreSQL 15 es el motor relacional seleccionado para el MRNA, y su elección responde a un conjunto de características técnicas que van más allá del cumplimiento básico de ACID. El soporte nativo de triggers permite definir lógica de negocio directamente en la base de datos que se dispara automáticamente ante eventos de inserción, actualización o eliminación, sin depender de que la capa de aplicación lo gestione en cada caso. Las vistas materializadas permiten precalcular y almacenar físicamente el resultado de consultas complejas (como el promedio ponderado de calificaciones por estudiante y asignatura), reduciendo significativamente el tiempo de respuesta en operaciones de lectura intensiva. El modelo de control de concurrencia multiversión (MVCC) garantiza que las lecturas no bloqueen las escrituras y viceversa, lo que resulta especialmente relevante en períodos de cierre académico cuando docentes y coordinadores acceden simultáneamente al sistema [13]. A estas capacidades se suman los esquemas para separar lógicamente los datos de cada microservicio dentro de una misma instancia, y los procedimientos almacenados para encapsular lógica académica compleja como el cálculo de promedios de período.

Esto se traduce en triggers para el registro automático de auditoría sobre cada modificación de nota, vistas materializadas para los reportes de rendimiento académico del dashboard, y procedimientos almacenados para el cálculo de promedios y la validación de condiciones de junta académica. Esta decisión de diseño (concentrar parte de la lógica en la base de datos en lugar de distribuirla únicamente en los microservicios) implica un acoplamiento mayor entre el motor y la aplicación, pero garantiza que las reglas de negocio críticas se apliquen independientemente del servicio que realice la operación.

Un tema relacionado con la elección del motor de base de datos es cómo se controla su evolución en el tiempo. A medida que un sistema crece, la estructura de sus tablas va cambiando. Se agregan columnas nuevas o se corrigen errores de diseño que solo aparecen una vez que el sistema está en uso. Hacer esos cambios a mano, ejecutando comandos SQL directamente sobre cada entorno, es una fuente común de errores porque no queda un registro confiable de qué cambió, cuándo cambió y en qué orden. La práctica de versionar el esquema de una base de datos con herramientas como Flyway resuelve ese problema. Cada cambio se guarda como un archivo numerado y con fecha, y la herramienta se encarga de aplicarlos siempre en el mismo orden, sea en la máquina del desarrollador, en el entorno de pruebas o en producción. Una revisión reciente de la literatura sobre evolución de esquemas de bases de datos confirma que llevar un control ordenado y trazable de estos cambios reduce errores de despliegue y facilita reproducir el mismo estado de la base de datos en distintos entornos, algo especialmente relevante cuando el sistema se integra a un flujo de integración continua [14]. El MRNA aplica exactamente ese enfoque con Flyway, como se detalla en la sección 3.5.3.

1.2.7. Autenticación moderna: JWT, refresh tokens y Spring Security

El JSON Web Token (JWT) es un mecanismo de autenticación basado en tokens que permite transmitir información de identidad y permisos entre dos partes de forma compacta, segura y sin necesidad de mantener estado en el servidor. Su estructura se compone de tres segmentos codificados en Base64URL y separados por puntos: el encabezado, que indica el tipo de token y el algoritmo de firma; el payload, que contiene los claims o afirmaciones sobre el usuario y sus roles; y la firma, que garantiza la integridad del token y permite detectar cualquier alteración en tránsito. La adopción de JWT en aplicaciones web ha crecido sostenidamente porque desacopla la validación de identidad del estado del servidor: cualquier servicio que conozca la clave de verificación

puede autenticar al usuario directamente, sin consultar una base de datos central en cada petición [15]. En arquitecturas de microservicios, donde múltiples servicios independientes deben verificar la identidad del solicitante, esta propiedad es particularmente valiosa.

El flujo de autenticación adoptado en el MRNA combina dos tipos de token con responsabilidades distintas. El access token tiene una vida útil corta y es el que se adjunta a cada petición HTTP para identificar al usuario y sus roles ante los microservicios. El refresh token, de mayor duración, se utiliza exclusivamente para obtener un nuevo access token cuando el anterior expira, sin que el usuario tenga que autenticarse de nuevo con sus credenciales. Esta separación reduce la ventana de exposición en caso de que un access token sea interceptado: su corta vigencia limita el daño potencial sin afectar la experiencia del usuario. Las implicaciones de seguridad de este esquema son concretas: el refresh token debe almacenarse en cookies HTTP-only para impedir su acceso desde JavaScript, los access tokens expirados no pueden revocarse retroactivamente salvo que se mantenga una lista negra en base de datos, y los algoritmos de firma asimétrica son preferibles en entornos distribuidos donde múltiples servicios validan los tokens sin compartir secretos [16].

En el MRNA, la capa de autenticación y autorización está implementada en el auth-service usando Spring Security 6, el framework de seguridad estándar del ecosistema Spring Boot. Spring Security gestiona el filtrado de peticiones entrantes, la validación de tokens JWT, la extracción de los claims de rol y la aplicación del modelo de control de acceso basado en roles (RBAC) que define los permisos de los cuatro roles institucionales del sistema: docente, coordinador, administrador y directivo. El API Gateway actúa como punto de entrada único y delega en el auth-service la validación del token antes de enrutar cada petición al microservicio correspondiente, garantizando que ningún servicio interno necesite implementar su propia lógica de autenticación.

1.2.8. Metodologías ágiles para proyectos de un solo desarrollador: Kanban

Kanban es un método de gestión del flujo de trabajo cuya aplicación al desarrollo de software fue sistematizada por David J. Anderson, quien estableció sus principios fundamentales en torno a la visualización del trabajo, la limitación del trabajo en curso (WIP) y la gestión activa del flujo como mecanismo principal de mejora continua. A diferencia de los marcos ágiles basados en iteraciones fijas (como Scrum, que organiza el trabajo en sprints cerrados con ceremonias predefinidas de planificación, revisión y

retrospectiva), Kanban no obliga a detener y reiniciar el trabajo en bloques: las tareas entran al tablero y avanzan por sus columnas según la capacidad disponible, no según el calendario de una iteración. Esto no excluye los puntos de sincronización temporal: el propio método contempla, entre sus prácticas generales, un conjunto de cadencias periódicas (reuniones y revisiones a intervalos regulares) que conviven con el flujo continuo sin convertirlo en una sucesión de compromisos cerrados [17]. El proyecto MRNA sigue esa misma lógica, organiza su trabajo en cadencias bisemanales que sirven como puntos de planificación y revisión, pero sin las ceremonias fijas ni el compromiso cerrado de alcance que distingue a un sprint de Scrum. En un proyecto de un solo desarrollador con plazo fijo, la sobrecarga ceremonial de Scrum representa una carga operativa mayor, mientras que Kanban permite concentrar el esfuerzo disponible directamente en el desarrollo.

La justificación de Kanban como metodología del proyecto MRNA responde a tres factores concretos. El primero es la naturaleza unipersonal del desarrollo: Scrum fue diseñado para equipos de entre tres y nueve personas, y varias de sus ceremonias (la planificación de sprint, la revisión y la retrospectiva) pierden sentido cuando no hay un equipo que coordinar. El segundo es la naturaleza del plazo, con la fecha de cierre de la versión evaluada fijada para julio de 2026. El tercero es la naturaleza del trabajo mismo: los requerimientos del MRNA combinan tareas de configuración de infraestructura, desarrollo de microservicios, integración con plataformas externas y documentación académica, una mezcla heterogénea que encaja mejor con un sistema de pull continuo. Una revisión sistemática de Kanban en el desarrollo de software confirma que el método es especialmente efectivo en contextos de trabajo individual o equipos pequeños donde la prioridad es mantener el flujo estable y reducir los tiempos de entrega [18].

Las métricas que Kanban genera de forma natural son también las que el Capítulo 4 utiliza para validar el proceso de desarrollo del MRNA. El lead time mide el tiempo total desde que una tarea entra al backlog hasta que se completa, y es el indicador más directo del ritmo de entrega del sistema. El cycle time mide únicamente el tiempo activo de trabajo sobre una tarea, excluyendo el tiempo de espera en cola, y permite identificar cuellos de botella específicos en el flujo. El diagrama de flujo acumulado (CFD) visualiza la evolución del WIP a lo largo del tiempo y revela patrones de bloqueo o acumulación que no son visibles en las métricas individuales. Todas estas métricas se recolectan automáticamente a través del tablero configurado en GitHub Projects, sin requerir esfuerzo manual adicional por parte del desarrollador.

1.2.9. Calidad de software y testing automatizado

La calidad del software en sistemas basados en microservicios adquiere una complejidad específica derivada de su naturaleza distribuida. A diferencia de las aplicaciones monolíticas, donde la validación puede concentrarse en un único proceso, los sistemas de microservicios exigen verificar tanto el comportamiento interno de cada servicio como los contratos de comunicación entre ellos. Los autores de [19], en un estudio de mapeo sistemático que analizó 93 investigaciones primarias sobre métodos de prueba para microservicios, identificaron nueve categorías de estrategias de verificación y concluyeron que la principal brecha en el campo es la escasez de herramientas dedicadas a la generación automática de pruebas para las interacciones entre servicios; hallazgo que refuerza la necesidad de definir una estrategia de pruebas estructurada desde la fase de diseño de cualquier prototipo de esta naturaleza.

La organización más extendida para esa estrategia es la pirámide de pruebas, que agrupa los tipos de verificación en tres niveles según su cobertura y costo de ejecución. En la base se ubican las pruebas unitarias, orientadas a validar funciones o clases de forma aislada mediante el reemplazo de sus dependencias externas por dobles de prueba (mocks o stubs); en el nivel intermedio, las pruebas de integración verifican la colaboración real entre módulos y la coherencia con la base de datos; y en el vértice, las pruebas de sistema o extremo a extremo (E2E) recorren flujos completos a través de todos los componentes desplegados simultáneamente. Los autores de [20] demuestran que, en arquitecturas de microservicios, las métricas de cobertura E2E y de API resultan indispensables porque las pruebas unitarias aisladas son incapaces de detectar los fallos que emergen únicamente en las interfaces de comunicación entre servicios.

En el prototipo MRNA esta estrategia se implementa con herramientas seleccionadas para cada nivel y para cada capa tecnológica del stack. En el backend desarrollado con Spring Boot 3.2+, JUnit 5 actúa como motor de ejecución de pruebas para Java y Mockito permite sustituir repositorios de datos y clientes HTTP, de modo que la lógica de negocio de cada microservicio puede validarse de manera aislada sin necesidad de infraestructura activa. En el frontend construido con React 19 y Vite, Vitest asume la función equivalente al integrarse de forma nativa con el entorno de construcción del proyecto, mientras que Testing Library complementa la validación al simular interacciones reales de usuario sobre los componentes sin depender de un navegador real. La cobertura que generan estas herramientas se articula directamente con el Capítulo 3, donde cada requerimiento

funcional implementado cuenta con al menos un caso de prueba documentado que certifica su comportamiento esperado bajo condiciones controladas.

1.2.10. Frameworks del stack: Spring Boot, React y TailwindCSS

Ya se explicó qué es un cliente, qué es un servidor y cómo se comunican mediante una API REST (sección 1.2.5). Falta explicar con qué herramientas concretas se construyó cada lado de esa comunicación en el MRNA.

Del lado del servidor, el MRNA se construyó con Spring Boot, un framework para programar en Java que facilita crear servicios web sin tener que escribir desde cero todo el código repetitivo que normalmente hace falta, como la conexión a la base de datos, el manejo de las peticiones web o la seguridad. Spring Boot se encarga de buena parte de esa configuración de forma automática, lo que le deja al desarrollador más tiempo para programar la lógica propia del sistema en lugar de resolver detalles técnicos que se repiten en cualquier proyecto. Una comparación reciente entre varios de los frameworks más usados para construir aplicaciones web, entre ellos Spring Boot, React y Angular, probó de forma experimental que Spring Boot es uno de los que mejor responde cuando el sistema recibe muchas peticiones al mismo tiempo, aunque también es uno de los que toma más tiempo aprender a usar bien frente a otras opciones más simples [21]. Para el MRNA, que necesita responder bien incluso cuando varios docentes registran notas al mismo tiempo, ese resultado respalda la elección hecha en la sección 3.4.

Del lado del cliente, la interfaz del MRNA se construyó con React, una librería para armar pantallas web a partir de piezas más pequeñas llamadas componentes, por ejemplo, una tabla de notas o un botón de guardar. Cada componente guarda su propia información y se puede reutilizar en distintas pantallas, en lugar de escribir el mismo código una y otra vez para cada rol del sistema. La misma comparación mencionada arriba también evaluó React y lo ubicó entre las opciones que responden mejor y más rápido a lo que hace el usuario en pantalla [21], algo que le sirve al MRNA porque varias de sus pantallas, como el ingreso de notas por paralelo de la sección 3.8.2, necesitan actualizarse al instante cada vez que el docente cambia un valor.

Sobre esos mismos componentes de React se aplica TailwindCSS, la herramienta que define cómo se ven las pantallas del sistema. En lugar de escribir una hoja de estilos distinta para cada pantalla, TailwindCSS ofrece un conjunto de clases ya preparadas para cosas como el espaciado, el color o el tamaño del texto, y esas clases se aplican directamente sobre cada elemento de la pantalla. Esto permite construir pantallas nuevas

más rápido y mantener una apariencia consistente entre las pantallas de los cuatro roles del sistema, sin tener que definir un estilo propio desde cero para cada una.

1.3. Estado del arte / Trabajos relacionados

Con el fin de ubicar el aporte del prototipo MRNA en el panorama investigativo reciente, se revisó literatura de los últimos cinco años centrada en dos ejes: la integración entre plataformas de gestión del aprendizaje y sistemas de información académica institucional, y el uso de microservicios en entornos educativos de nivel superior. Se identificaron cinco trabajos directamente comparables cuyo análisis permite delimitar el alcance original de la presente propuesta.

La búsqueda de estos antecedentes se hizo en buscadores y repositorios académicos de acceso abierto, principalmente Google Scholar, Redalyc y SciELO, dado que buena parte de los estudios finalmente incluidos están publicados en revistas alojadas en esos repositorios. Se usaron combinaciones de términos en español e inglés relacionados con los dos ejes ya mencionados, entre ellos integración Moodle sistema académico, LMS academic information system integration, y microservicios educación superior. Se aplicó como criterio de inclusión que el estudio describiera un caso real o un prototipo de integración entre una plataforma de aprendizaje y un sistema de información académica, o el uso de microservicios en un contexto educativo, publicado en los últimos cinco años en español o en inglés. Se excluyeron notas divulgativas sin proceso de revisión por pares y estudios centrados en niveles educativos distintos al superior.

Los autores de [10] evaluaron, mediante un diseño mixto de corte transversal, la percepción de los usuarios sobre el Sistema de Información para la Educación Digital (SIDigital) de la Universidad de Guanajuato, una plataforma institucional que centraliza la información generada por la operación de la educación digital entre distintas dependencias de esa universidad. El estudio identificó áreas de mejora concretas sobre esa plataforma ya en funcionamiento, pero se limita a describir la percepción de los usuarios sobre un sistema centralizado propio de la institución. No aborda la integración entre dos plataformas separadas ni el escenario, propio del ISTA, en el que el sistema de destino carece de una API pública.

Los autores de [22] propusieron un modelo de integración entre un sistema de información académica y Moodle mediante servicios web REST con base de datos externa compartida, implementado en una universidad indonesia. La solución consigue

sincronización en tiempo real de matrículas y resultados de evaluación; sin embargo, emplea una arquitectura orientada a servicios tradicional (SOA) que no permite el despliegue ni el escalado independiente de cada función del sistema.

Los autores de [11] demostraron que la conexión de un sistema académico institucional con los módulos de calificaciones de Moodle a través de una API REST propia reduce errores y tiempos de proceso frente al registro manual. La propuesta cubre exclusivamente el flujo de notas de un único servicio y no incorpora control de asistencia ni perfiles de acceso diferenciados por rol, elementos que el prototipo MRNA atiende de forma integrada.

Los autores de [23] presentaron un prototipo de plataforma de e-learning construida sobre microservicios con API Gateway, validando la viabilidad técnica de esta arquitectura en entornos educativos y reportando mejoras de rendimiento respecto a la implementación monolítica de referencia. El sistema se limita a funciones de entrega de contenidos de aprendizaje y no contempla la integración con sistemas externos de registro académico oficial ni con una plataforma LMS preexistente.

Los autores de [24] realizaron una revisión de literatura sobre el rol de los microservicios en la mejora de la calidad de Moodle, concluyendo que esta arquitectura resuelve problemas de escalabilidad, mantenimiento y capacidad de respuesta que afectan a las instalaciones monolíticas del LMS. Al tratarse de un estudio teórico, la propuesta no incluye un sistema implementado ni métricas de validación que permitan evaluar el comportamiento en producción.

Tabla 1. Comparativa de antecedentes frente a MRNA. Nota. Elaboración propia, 2026.

Autor / Año	Objetivo principal	Arquitectura	Integración LMS	Gestión asistencia	Contexto normativo	Limitación principal
Magdaleno Horta et al. (2024)	Percepción de usuarios sobre un sistema institucional de educación digital (SIDigital)	Centralizada	No	No	México / Universidad de Guanajuato	Evalúa percepción de usuarios de un sistema ya centralizado, no aborda integración entre plataformas ni ausencia de API
Sahara, Abdullah & Saputra (2023)	Integración AIS–LMS con servicios web REST y BD externa	SOA	Sí	No	Indonesia / sistema local	Sin despliegue independiente por función; sin adaptador para sistemas cerrados

Prasetyo & Saintika (2021)	Transferencia de notas SIA → Moodle por API REST propia	REST monolítica	Sí	No	Indonesia / sistema local	Un solo servicio de notas; sin control de asistencia ni perfiles de acceso
Kurniawan & Dzikri (2024)	Prototipo e-learning sobre microservicios con API Gateway	Microservicios	Parcial	No	Indonesia / plataforma propia	Solo entrega de contenidos; sin integración con registro académico externo
Muhdiantini et al. (2024)	Revisión de literatura sobre microservicios para mejorar Moodle	Teórica	Teórica	No	No especificado	Sin sistema implementado ni validación funcional medida
MRNA — propuesta actual (2026)	Integración Moodle ↔ SIGA con notas, asistencia y 4 roles	6 microservicios	Sí	Sí	Ecuador / SENESCYT –SIGA	Prototipo funcional; adaptador SIGA dual por ausencia de API pública

El conjunto de antecedentes revisado deja en evidencia un espacio no cubierto que el prototipo MRNA aborda directamente. Ninguno de los trabajos anteriores enfrenta simultáneamente la integración con un sistema nacional sin API pública, la gestión conjunta de notas y asistencia, el soporte a múltiples roles institucionales y el despliegue sobre microservicios dentro del marco normativo de la educación superior ecuatoriana. Esa combinación de condiciones configura el aporte diferenciado de MRNA respecto a los sistemas documentados en la literatura.

1.4. Marco normativo institucional

Para el proyecto MRNA, el marco normativo es el eje que explica por qué el proceso que se intenta automatizar existe bajo qué reglas funciona y qué parámetros debe respetar cualquier solución tecnológica que lo reemplace. Tres niveles de regulación convergen sobre la gestión académica del ISTA, y entenderlos resulta indispensable para delimitar correctamente el alcance del prototipo.

Nivel nacional. En el nivel más alto de la jerarquía normativa ecuatoriana, la Ley Orgánica de Educación Superior establece las condiciones generales bajo las cuales opera el sistema de educación superior en el país. Su artículo 93 define los criterios de calidad que deben orientar la gestión de toda institución de educación superior, incluyendo la

sistematización y el acceso oportuno a la información académica como componente indispensable de ese estándar [2]. El artículo 96, por su parte, regula la garantía de los derechos de los estudiantes, entre los que figura el acceso transparente a sus registros de calificaciones, derecho que sólo puede ejercerse si esos registros están debidamente integrados y son trazables. Esta doble exigencia de calidad y accesibilidad es precisamente la que el flujo manual Moodle-Excel-SIGA no puede cumplir de manera consistente, dado que descansa en la disciplina individual de cada docente para garantizar que los datos ingresados en un sistema lleguen intactos al siguiente.

El Reglamento de Régimen Académico del Consejo de Educación Superior (RPC-SE-08-No.023-2022) desciende al plano operativo. Su artículo 67 especifica los criterios que deben regir el registro y el reporte de la actividad académica de los estudiantes, definiendo las categorías de evaluación, los plazos de cierre de período y las condiciones generales para la conducción de las juntas académicas [3]. Para el MRNA, ese artículo respalda dos funcionalidades de forma directa, el bloqueo de registros al cierre del período académico (RF-012) y la validación previa a la carga de notas al SIGA (RF-010). El período de gracia de cuarenta y ocho horas que RF-015 formaliza no proviene de ese artículo, sino de una práctica interna del ISTA descrita durante las entrevistas con coordinadores, en las que se explicó que tras la junta académica el docente dispone de ese plazo para modificar notas antes del bloqueo automático. El reglamento del CES exige que exista un procedimiento formal de juntas académicas y de cierre de período, pero deja la duración exacta de ese margen a criterio de cada institución, y el ISTA la fija en cuarenta y ocho horas.

Nivel sectorial. La SENESCYT, como entidad rectora del sistema de educación superior ecuatoriano, administra el SIGA como plataforma de registro obligatorio para todos los institutos superiores tecnológicos públicos del país [4]. A diferencia de las universidades, que cuentan con mayor autonomía para gestionar sus propios sistemas de información, los institutos superiores tecnológicos públicos están obligados a reportar al SIGA en formatos y plazos definidos centralmente. Esta exigencia es la que genera la presión en el flujo académico del ISTA: no basta con registrar correctamente en Moodle; los datos deben llegar al SIGA en los formatos que el sistema central admite, y esa transferencia hoy se hace de manera manual.

Lo que el SIGA no hace, y que este trabajo documenta como una restricción arquitectónica de primera importancia, es exponer una API pública que permita a sistemas externos conectarse directamente para consultar o escribir datos. Esta limitación fue

confirmada de forma explícita por el responsable de TIC del Instituto, quien señaló que no existe ningún mecanismo de integración formal disponible para el ISTA (Responsable de TIC, comunicación personal, 2025). Esta confirmación corresponde al caso concreto del ISTA, como se explica en la sección 1.2.3, y no se toma aquí como una auditoría del sistema completo a nivel nacional. Como consecuencia, el adaptador diseñado para el MRNA adopta un modo dual: generación de archivos en formato XLSX compatibles con la carga manual al SIGA, que es el mecanismo que efectivamente opera hoy, y una interfaz reservada para una futura API, documentada pero no conectada a ningún flujo ejecutable del prototipo. Esta decisión no es una limitación del diseño sino una respuesta directa a la restricción normativa y técnica del entorno institucional real.

Nivel institucional. En el plano interno del ISTA, el respaldo formal al proyecto MRNA proviene del Plan Estratégico de Desarrollo Institucional 2021-2026. Este documento traza la hoja de ruta estratégica de la institución hasta el cierre del período en curso y contempla entre sus objetivos el fortalecimiento de las capacidades tecnológicas institucionales y la innovación en los procesos académicos orientada a la mejora continua de la calidad educativa [25]. La alineación entre esas líneas estratégicas y el objeto del prototipo no es forzada: el MRNA responde directamente a la necesidad de tecnificar el proceso de registro académico que el propio Instituto identifica como prioritario en su planificación.

La convergencia entre los tres niveles normativos (la exigencia de calidad y trazabilidad impuesta por la LOES, la obligación de reporte al SIGA establecida por SENESCYT y el compromiso de modernización que el ISTA asumió en su propio plan estratégico) es lo que da al prototipo su sustento institucional completo. El MRNA no nace de una inquietud académica desconectada del contexto real: responde a un problema documentado, respaldado por normativa vigente y alineado con la planificación estratégica de la institución que lo necesita.

1.5. Síntesis y aporte del capítulo

A lo largo de este capítulo se construyó el andamiaje teórico, investigativo y normativo que sostiene el prototipo MRNA. El marco teórico estableció que la arquitectura de microservicios ofrece las propiedades de independencia de despliegue, escalabilidad selectiva y resiliencia que un sistema de integración académica como el propuesto requiere para operar de forma estable frente a cargas variables y actualizaciones

frecuentes. Se fundamentó la elección del API Gateway como punto de entrada unificado, de la API REST de Moodle como canal de extracción de datos desde el entorno virtual, y de JWT como mecanismo de autenticación sin estado adecuado para entornos distribuidos. Se justificó asimismo la selección del stack tecnológico (Spring Boot, React, PostgreSQL) en función de su madurez, su compatibilidad con los patrones de microservicios y el soporte activo de sus comunidades. La metodología Kanban se presentó como el marco de gestión que mejor se adapta a un equipo unipersonal con entregas continuas y plazos institucionales fijos, mientras que las herramientas de prueba seleccionadas (JUnit 5, Mockito, Vitest y Testing Library) proveen la cobertura necesaria para validar cada capa del sistema.

El estado del arte evidenció que las propuestas existentes abordan de manera parcial el problema de integración entre plataformas de aprendizaje y sistemas de registro académico. La mayoría opera sobre arquitecturas centralizadas, cubre únicamente el flujo de calificaciones y no contempla escenarios donde el sistema de destino carezca de API pública. Ese conjunto de condiciones, propio del contexto ecuatoriano con SIGA como sistema nacional cerrado, define la brecha que el prototipo MRNA busca cerrar. El marco normativo, por su parte, confirmó que automatizar el registro de notas y asistencia responde a obligaciones legales vigentes que el ISTA debe cumplir ante entidades como el SENESCYT, y no únicamente a una mejora de eficiencia operativa.

Con estos fundamentos establecidos, el Capítulo 2 aborda el diagnóstico situacional del Instituto Superior Tecnológico del Austro, describiendo el estado actual de sus procesos académicos, las entrevistas realizadas a actores clave y la problemática concreta que motiva el desarrollo del prototipo.

Capítulo 2. Diagnóstico Situacional

2.1. Introducción del capítulo

Este segundo capítulo establece la base situacional del trabajo. Antes de proponer una arquitectura técnica conviene entender contra qué problema se trabaja, en qué institución, con qué actores, bajo qué marco normativo y con qué evidencia previa. Esa es la función del capítulo. Empieza con una caracterización del Instituto Superior Tecnológico del Austro, donde se sitúa el proyecto, y describe el flujo manual de registro de notas y asistencia que actualmente se utiliza en su gestión académica. A partir de esa descripción se formaliza el problema, se justifican las razones para abordarlo y se enuncian los objetivos del prototipo, en los términos literales con que fueron aprobados en el anteproyecto. Las secciones siguientes cubren el alcance del trabajo, los requerimientos funcionales y no funcionales que orientan el desarrollo, y cierran con la metodología de la investigación y la metodología de desarrollo de software adoptadas para la construcción del prototipo. Al terminar este capítulo, el lector debería contar con un panorama suficiente del qué, el para quién, el porqué y el cómo del proyecto, antes de entrar en el diseño detallado de la propuesta en el Capítulo 3.

2.2. Contexto institucional

El proyecto MRNA tiene como cliente institucional al Instituto Superior Tecnológico del Austro, en adelante el ISTA o el Instituto, una institución pública de educación superior con sede en la ciudad de Azogues, provincia del Cañar, Ecuador. Su oferta formativa se concentra en carreras técnico-tecnológicas de tercer nivel, modalidad que en el sistema educativo ecuatoriano está orientada a formar profesionales con énfasis en competencias prácticas y vinculación directa con el sector productivo. Como instituto público, su gestión académica está supeditada a las directrices de la Secretaría de Educación Superior, Ciencia, Tecnología e Innovación (SENESCYT) [4], entidad rectora que administra el Sistema Integrado de Gestión Académica (SIGA), de uso obligatorio para todos los institutos superiores tecnológicos públicos del país.

Para entender cómo opera el Instituto en el día a día y qué actores participan en la gestión académica de notas y asistencia, se levantó evidencia primaria a través de ocho entrevistas semiestructuradas con autoridades, coordinadores académicos, docentes y personal de tecnologías de la información del ISTA. El resultado de ese levantamiento permite

mapear los roles institucionales que intervienen en el proceso. En el nivel directivo se ubica el vicerrectorado, responsable de la planificación estratégica institucional y de la articulación con SENESCYT. En el nivel intermedio se encuentran las coordinaciones académicas de las distintas carreras, a cargo del Coordinador de la carrera de Desarrollo de Software, el Coordinador 1 y el Coordinador 2, quienes supervisan el cumplimiento del Plan de Enseñanza-Aprendizaje (PEA), conducen las juntas académicas de cierre de parcial y consolidan la información que se reporta al SIGA. En el nivel operativo se sitúa el cuerpo docente, del que se entrevistó a la Docente 1 y a la Docente 2, junto con el Docente 3, responsables directos del registro de calificaciones por componente y del control de asistencia de sus paralelos. Finalmente, el área de Tecnologías de la Información y Comunicación, representada por el responsable de TIC del Instituto, administra el Moodle institucional (conocido internamente como E-Virtual) y constituye el contacto técnico para cualquier integración con sistemas externos. El Instituto cuenta actualmente con cuarenta y nueve docentes activos y con una coordinación académica unificada que articula las distintas carreras de su oferta formativa (Coordinación Académica del ISTA, comunicación personal, 2026).

De estas entrevistas emerge un retrato bastante consistente del Instituto. Se trata de una organización en pleno proceso de digitalización, donde conviven plataformas formales como Moodle y SIGA con prácticas informales basadas en hojas de cálculo, correos electrónicos y carpetas compartidas en Google Drive. El vicerrector fue explícito al señalar que muchos procesos de gestión académica "se gestionan de manera artesanal", que existen problemas de duplicación de esfuerzos cuando los docentes deben cargar la misma información en Moodle, Excel, Drive y SIGA, y que persiste una "débil cultura digital" que dificulta la adopción homogénea de las herramientas institucionales (Vicerrector del ISTA, comunicación personal, 2025). Esta caracterización es particularmente relevante para el diseño del prototipo MRNA, porque define una restricción clara: cualquier solución tecnológica que se proponga debe ser intuitiva, debe minimizar la curva de aprendizaje y debe convivir con prácticas heredadas, no reemplazarlas de un día para otro.

El tratamiento de la información recogida en estas entrevistas siguió un criterio de protección básica de identidad. Los participantes supieron en todo momento que la conversación formaba parte del levantamiento de este proyecto de titulación, y su participación fue voluntaria, aunque no se firmó un documento formal de consentimiento informado. Siguiendo la orientación del director del proyecto, el presente trabajo no

identifica a ningún entrevistado por su nombre completo, únicamente por el cargo o rol institucional que ocupa, lo que reduce el riesgo de exposición individual sin restarle trazabilidad a la fuente de cada afirmación. Las entrevistas quedaron registradas en audio en los casos en que fue posible grabar, y en todos los casos en formato de texto a partir de las respuestas documentadas durante la sesión. Ambos registros se conservan únicamente para fines académicos de este trabajo y no se comparten fuera de los anexos que forman parte del documento de tesis.

A esta realidad operativa se superpone un marco normativo exigente que el Instituto está obligado a cumplir. La Ley Orgánica de Educación Superior establece, en sus artículos 93 y 96, que las instituciones de educación superior deben garantizar la calidad académica y la accesibilidad de los registros estudiantiles [2]. El Reglamento de Régimen Académico del Consejo de Educación Superior, en el artículo 67, regula los criterios específicos para la calificación, el registro y el reporte de la actividad académica de los estudiantes [3]. En el plano institucional, el Plan Estratégico de Desarrollo Institucional 2021-2026 del ISTA, aprobado mediante Resolución 084-OCS-09-ISTAUSTRO-2024, contempla entre sus objetivos estratégicos el fortalecimiento de las capacidades tecnológicas institucionales y la promoción de la innovación en los procesos académicos para mejorar continuamente la calidad educativa [25]. Este último documento es el que da el respaldo institucional formal al desarrollo del MRNA, pues alinea el prototipo con la línea estratégica que la propia institución se ha trazado para el período en curso.

Con esta caracterización del cliente institucional, de sus actores y del marco normativo que lo gobierna, queda planteado el escenario sobre el que se va a delimitar el problema en la siguiente sección.

2.3. Planteamiento del problema

La gestión académica en instituciones de educación superior enfrenta retos significativos asociados a la fragmentación de sistemas de información y a la dependencia de procesos manuales para el registro de calificaciones y asistencia. La revisión sistemática de [26] sobre transformación digital en instituciones de educación superior identifica que la falta de integración entre plataformas educativas genera duplicación de datos, inconsistencias en los registros y una carga administrativa considerable para el personal docente. Este patrón se repite en distintos países y contextos, y ha llevado a posicionar la integración

de sistemas como un eje estratégico de modernización académica frente a la creciente complejidad de los procesos institucionales [1].

En el caso ecuatoriano, los institutos superiores tecnológicos públicos operan bajo un marco normativo específico que añade exigencia al panorama anterior. La Ley Orgánica de Educación Superior establece la obligación de garantizar la calidad y la consistencia de los registros académicos [2], mientras que el Reglamento de Régimen Académico del Consejo de Educación Superior regula los criterios de calificación y reporte de la actividad académica de los estudiantes [3]. Como consecuencia, los institutos están obligados a reportar al Sistema Integrado de Gestión Académica administrado desde Quito por SENESCYT [4], al tiempo que mantienen plataformas de aprendizaje virtual como Moodle para gestionar las actividades académicas en línea. La coexistencia de ambos sistemas, sin un canal de integración formal entre ellos, define el escenario problemático sobre el que se construye este trabajo.

En el ISTA, el registro de notas y asistencia se realiza actualmente sobre un flujo manual que obliga a docentes y coordinadores a duplicar esfuerzos en al menos tres puntos del proceso. Los docentes ingresan inicialmente las calificaciones por componente (docencia, práctica, autónoma y evaluación) en la plataforma Moodle institucional. A continuación, trasladan esos datos manualmente a una plantilla de Excel proporcionada por la coordinación académica, donde se calculan promedios y se consolidan los registros para discutirlos en las juntas académicas de cierre de parcial. Finalmente, los datos consolidados se cargan al SIGA por parte del coordinador. Este ciclo de triple digitación fue confirmado de forma explícita por el Coordinador 1, quien describió que el proceso "se realiza hasta tres veces" y estimó que un docente a tiempo completo, con un promedio de cinco asignaturas a su cargo, dedica aproximadamente dos horas por asignatura sólo al traspaso de información entre plataformas (Coordinador 1, comunicación personal, 2025). El Coordinador de Desarrollo de Software reforzó la misma observación al señalar que la falta de integración entre Moodle y la plantilla de Excel "obliga a los docentes a duplicar esfuerzos" y vuelve el proceso ineficiente y propenso a errores (Coordinador de Desarrollo de Software, comunicación personal, 2025). La Figura 1 sintetiza este flujo y permite visualizar los puntos críticos sobre los que se acumula el problema. Este patrón de duplicación de esfuerzo entre plataformas que no se comunican entre sí no es exclusivo del ISTA, es justamente el mismo problema que documenta la literatura revisada en la sección 1.3 sobre integración de sistemas académicos [23].

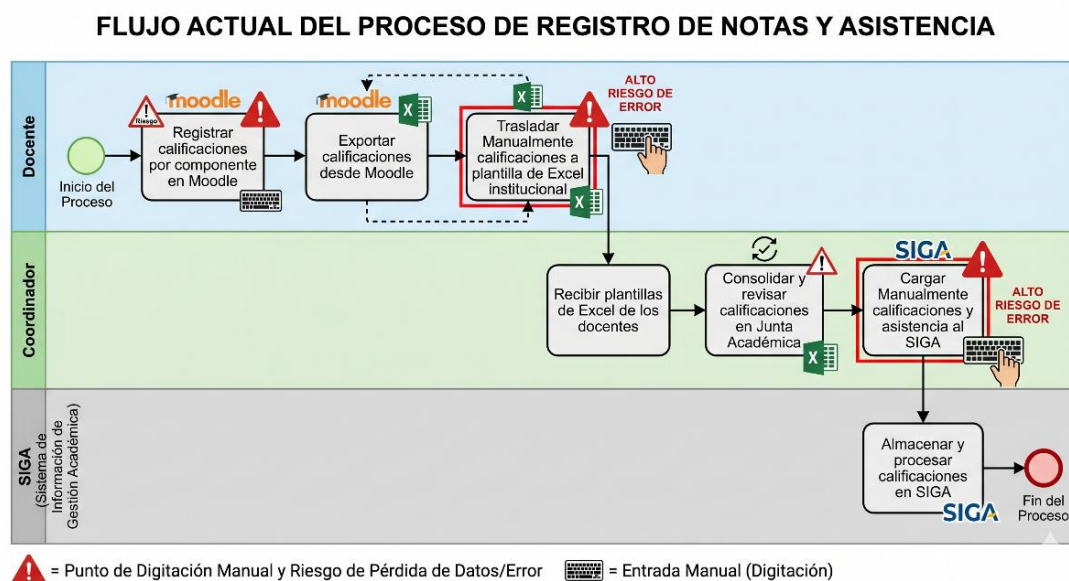


Figura 1. Flujo actual del proceso de registro de notas y asistencia en el ISTA. Nota. Elaboración propia con asistencia de inteligencia artificial, con base en entrevistas a docentes y coordinadores del Instituto, 2025.

La problemática, sin embargo, no se agota en la duplicación de digitación. Las entrevistas con docentes y autoridades del Instituto revelan riesgos adicionales que afectan la integridad y la trazabilidad de la información académica. Se identifican casos recurrentes de pérdida de archivos Excel por la ausencia de un respaldo centralizado, generación de versiones inconsistentes de un mismo documento y carencia de mecanismos de control de acceso que prevengan modificaciones no autorizadas. El vicerrector destacó además la inexistencia de un sistema institucional de alertas tempranas que permita identificar oportunamente a estudiantes con bajo rendimiento académico o ausentismo significativo, lo que limita la capacidad de la coordinación para intervenir antes del cierre del período (Vicerrector del ISTA, comunicación personal, 2025). A esto se suma la dependencia de canales informales como WhatsApp para la comunicación entre docentes y coordinadores, una práctica que el propio vicerrector caracteriza como un obstáculo para formalizar la trazabilidad institucional. Estos riesgos de integridad y trazabilidad de la información coinciden con lo que la misma revisión describe como una consecuencia habitual cuando varias plataformas conviven sin integración formal entre ellas [23].

El impacto agregado de esta situación se observa en tres dimensiones que conviene explicitar. La primera es la calidad académica, comprometida por la fragilidad de un proceso de consolidación que descansa en la disciplina manual de cada docente y de cada coordinador. La segunda es la carga administrativa sobre el cuerpo docente, que dedica

horas semanales a tareas repetitivas de transcripción que podrían orientarse a tutorías, seguimiento estudiantil o mejora pedagógica. La tercera, y posiblemente la más crítica desde la perspectiva institucional, es la limitada capacidad del Instituto para generar información oportuna y confiable sobre el rendimiento de sus estudiantes, lo cual restringe su posibilidad de tomar decisiones estratégicas basadas en datos. Sobre esta problemática se construye la justificación del proyecto, que se desarrolla en la sección siguiente.

2.4. Justificación

La pertinencia del proyecto MRNA se sustenta en cuatro frentes complementarios que se desarrollan a continuación, siguiendo la estructura prevista en el esquema oficial de titulación de la Carrera de Software.

2.4.1. Importancia teórica

Desde el punto de vista del conocimiento, el proyecto aporta evidencia empírica sobre la aplicabilidad de la arquitectura de microservicios y sus patrones asociados en un dominio poco explorado en la literatura, la gestión académica de los institutos superiores tecnológicos públicos del Ecuador. La revisión de [23] sobre transformación digital en educación superior confirma que la integración entre plataformas educativas tiende a documentarse para universidades, pero rara vez para el subsistema técnico-tecnológico, en particular bajo las restricciones que imponen sistemas centralizados como el SIGA. Ese mismo cuerpo de literatura sitúa la integración de sistemas como uno de los ejes de la modernización académica, algo que el presente proyecto confirma con un caso de aplicación concreto. El presente trabajo, al construir un prototipo funcional documentado en este contexto, contribuye con un caso de aplicación concreto que puede servir de referencia para futuros estudios sobre integración entre sistemas de información estudiantil y plataformas LMS en realidades institucionales con limitaciones similares a las del ISTA.

2.4.2. Importancia práctica

En el plano operativo, el MRNA se justifica por el beneficio directo que genera para el ISTA. La eliminación del ciclo de triple digitación entre Moodle, Excel y SIGA es la más visible: el Coordinador 1 estimó que un docente a tiempo completo dedica aproximadamente dos horas por asignatura sólo al traspaso manual de calificaciones

(Coordinador 1, comunicación personal, 2025), tiempo que con un sistema automatizado puede redirigirse a tutorías, retroalimentación pedagógica o seguimiento individual de estudiantes. A esto se suma la incorporación de alertas automáticas por bajo rendimiento y ausentismo, una capacidad cuya ausencia el vicerrector identificó como una de las debilidades centrales de la gestión académica actual (Vicerrector del ISTA, comunicación personal, 2025). El prototipo también introduce trazabilidad mediante auditoría sobre cada modificación de notas y asistencia, requisito que el Coordinador de Desarrollo de Software asocia directamente con la posibilidad de tomar decisiones académicas sustentadas en datos verificables (Coordinador de Desarrollo de Software, comunicación personal, 2025).

2.4.3. Resultados esperados

Los entregables previstos del proyecto se concentraron en seis bloques verificables, cuyo grado de cumplimiento real se reporta de forma desglosada en la tabla de la sección 2.7 y se sustenta con evidencia en el Capítulo 4. En primer lugar, un prototipo funcional orientado a cubrir los catorce requerimientos funcionales clasificados como Must y Should, junto con hasta cuatro requerimientos Could-have si el ritmo del desarrollo lo permitía; al cierre de la Cadencia 5, nueve de esos catorce quedaron completamente implementados, cuatro se completaron de forma parcial y uno permanece en fase de diseño. En segundo lugar, una arquitectura de seis microservicios desplegados sobre Docker Compose que, en esta versión del prototipo, comparten una única instancia de base de datos PostgreSQL, la independencia entre servicios opera a nivel de código y de responsabilidad, todavía no a nivel de almacenamiento, una decisión que se explica en la sección 3.3 y se retoma como línea de trabajo futuro en el Capítulo 5. En tercer lugar, la base de datos PostgreSQL operativa con triggers, vistas materializadas y procedimientos almacenados. Como cuarto entregable se contempla la documentación técnica del sistema, que incluye la especificación OpenAPI generada con SpringDoc. El quinto bloque corresponde a las métricas Kanban recolectadas durante todo el desarrollo, las cuales sustentan el análisis cuantitativo del Capítulo 4. Y como cierre, una validación empírica que compara el proceso manual actual con el flujo automatizado del prototipo, con datos cuantitativos sobre tiempos de ejecución, tasas de error y percepción de usabilidad.

2.4.4. Factibilidad

El proyecto es factible en cuatro dimensiones que conviene explicitar. En lo técnico, el stack tecnológico seleccionado (Spring Boot, React, PostgreSQL, Docker) es de código abierto, ampliamente documentado y consistente con el perfil curricular de la Carrera de Software de la Universidad Católica de Cuenca, lo que descarta riesgo de incapacidad técnica. En lo operativo, el respaldo institucional está formalizado a través de la aprobación del anteproyecto y se cuenta con acceso confirmado a stakeholders del ISTA, evidenciado por las entrevistas ya levantadas durante la fase de análisis. En lo económico, el proyecto no involucra costos de licencia ni infraestructura adicional, dado que todo el desarrollo se ejecuta sobre el equipo personal del autor con herramientas gratuitas o de uso académico. Y en lo legal, el prototipo se mantiene dentro del marco de la Ley Orgánica de Educación Superior [2], del Reglamento de Régimen Académico [3] y de la autorización institucional documentada en el anteproyecto, con especial atención al manejo responsable de datos personales de estudiantes y docentes durante la operación del prototipo.

2.5. Objetivos

Los objetivos del proyecto MRNA son los aprobados en el anteproyecto firmado por el director y se reproducen a continuación de forma literal. Ninguna conclusión del Capítulo 5 se redactará sin contrastarse contra esta formulación.

2.5.1. Objetivo general

Desarrollar un prototipo de sistema web para el registro de notas y asistencia, mediante una arquitectura de microservicios, que permita reemplazar el proceso manual basado en hojas de cálculo Excel, para mejorar la eficiencia de la gestión académica.

2.5.2. Objetivos específicos

1. Analizar y documentar los requerimientos funcionales y no funcionales del sistema mediante entrevistas con stakeholders, análisis del proceso actual en Excel y evaluación de los mecanismos de integración disponibles con Moodle y con el SIGA, definiendo el alcance del prototipo funcional.
2. Diseñar la arquitectura de microservicios del sistema, especificando la estructura de la base de datos PostgreSQL, los servicios independientes necesarios para

soportar las funcionalidades de registro académico, generación de reportes y notificaciones, así como los mecanismos de integración con Moodle mediante API REST y con el SIGA

3. Desarrollar el prototipo funcional del sistema construyendo los microservicios core (gestión de usuarios, registro de notas y asistencia, sincronización Moodle, generación de reportes), validando cada servicio mediante pruebas funcionales.
4. Validar el prototipo desarrollado mediante pruebas de funcionalidad, accesibilidad y usabilidad con casos de uso reales, obteniendo métricas de desempeño, tiempos de respuesta y tasa de errores para evaluar el cumplimiento de los requerimientos funcionales y documentar resultados cuantitativos, con el fin de obtener datos estadísticos de cuánto mejoraría el proceso con la solución planteada.

2.6. Preguntas de investigación

En coherencia con los objetivos planteados, este proyecto de titulación busca responder dos preguntas de investigación que articulan los aspectos centrales de la propuesta y orientan la recolección de evidencia que se desarrolla en los capítulos posteriores. La primera se concentra en el aporte funcional del prototipo frente al proceso manual actual; la segunda, en las decisiones arquitectónicas que hacen viable la integración con las plataformas externas del Instituto.

PI-1. ¿En qué medida un prototipo basado en una arquitectura de microservicios permite automatizar el flujo de registro de notas y asistencia, y reducir el tiempo administrativo que demanda el proceso manual actual del ISTA?

PI-2. ¿Qué decisiones arquitectónicas hacen posible una integración funcional con la plataforma Moodle institucional y, al mismo tiempo, dejan preparada la viabilidad de una integración futura con el SIGA, considerando que este sistema actualmente no expone una API pública para terceros?

Las dos preguntas se responden de manera explícita en la sección 4.6 del Capítulo 4, sobre la base de las métricas funcionales, de desempeño y de usabilidad recolectadas durante la fase de validación del prototipo, así como del análisis del adaptador SIGA dual implementado en el sync-service.

2.7. Alcance del prototipo

El alcance del prototipo MRNA queda delimitado por los hallazgos del levantamiento de requerimientos y por la priorización aprobada en el anteproyecto. Esta sección distingue de manera explícita lo que entra al alcance del proyecto de titulación, lo que queda fuera de él y la justificación de las exclusiones.

Lo que entra en el alcance. Forman parte del alcance del prototipo dieciocho requerimientos funcionales implementables, distribuidos según la priorización MoSCoW en catorce de prioridad alta (siete clasificados como Must-have y siete como Should-have) y cuatro de prioridad opcional clasificados como Could-have, que se abordarán únicamente si la velocidad del flujo de trabajo lo permite. Estos requerimientos se materializan en una arquitectura de seis microservicios independientes (api-gateway, auth-service, academic-service, sync-service, report-service y notification-service) desplegados sobre Docker Compose, una base de datos PostgreSQL con aproximadamente treinta y seis tablas y lógica embebida mediante triggers, vistas materializadas y procedimientos almacenados, una integración funcional con una instancia local de Moodle 4.1.14 o superior, un adaptador dual para SIGA con dos modos de operación (archivo XLSX, con una interfaz de API reservada para una integración futura) y un frontend en React con los flujos de interacción correspondientes a los roles de docente, coordinador, administrador y directivo del Instituto.

Estado de avance real. De los catorce requerimientos de prioridad alta, nueve quedaron completamente implementados y verificados al cierre de la Cadencia 5, la carga masiva hacia el SIGA (RF-001), la sincronización con Moodle (RF-002, RF-003), la gestión de roles y permisos (RF-005), el filtrado avanzado (RF-007), la validación pre-carga (RF-010), el bloqueo por período cerrado (RF-012), la carga masiva de usuarios (RF-013) y la auditoría de notas y asistencia (RF-018). Cuatro se completaron de forma parcial: la generación de reportes (RF-008) resuelve los formatos CSV y Excel, pero la exportación a PDF permanece en diseño, las notificaciones automáticas por bajo rendimiento (RF-006) cuentan con el canal de correo operativo, pero la detección se ejecuta bajo demanda y no de forma automática, las notificaciones programadas (RF-014) comparten esa misma limitación, sin un mecanismo de agendamiento implementado, y la junta académica con período de gracia de 48 horas (RF-015) tiene resuelto el modelo de datos que sostiene la ventana de modificación, pero no la lógica de negocio que aplica el bloqueo y desbloqueo de edición. Un requerimiento permanece pendiente y es el dashboard con histograma de

rendimiento (RF-004), sin avance de código ni en el backend ni en la interfaz. En conjunto, el prototipo alcanza al cierre de la Cadencia 5 un cumplimiento completo en nueve de los catorce requerimientos de prioridad alta (64%), parcial en cuatro (29%) y pendiente en uno (7%); la Tabla 2 detalla esta clasificación requerimiento por requerimiento.

Lo que queda fuera del alcance. El prototipo deja por fuera, de manera explícita, cinco requerimientos funcionales que se documentan como trabajo futuro: la firma electrónica de reportes mediante Quipux, el módulo de ayuda interactiva, el portal de noticias académicas, el chat integrado entre docentes y estudiantes, y la sincronización con calendarios externos como Google Calendar u Outlook. Quedan también fuera del alcance el despliegue del sistema en infraestructura institucional productiva, la migración de los datos históricos académicos del ISTA al nuevo modelo, la capacitación formal de docentes y coordinadores, y cualquier servicio de mantenimiento o soporte técnico posterior a la fecha de entrega del proyecto de titulación.

Justificación de las exclusiones. Las cinco funcionalidades diferidas no se descartan por ausencia de pertinencia, sino por restricciones técnicas y de alcance documentadas en el catálogo consolidado de requerimientos. La firma electrónica vía Quipux requiere integración con sistemas documentales del gobierno ecuatoriano y entidades certificadoras externas que exceden la viabilidad del prototipo. El módulo de ayuda interactiva no resuelve un problema central del negocio académico y ninguna entrevista lo identificó como necesidad crítica. El portal de noticias y el chat docente-estudiante no fueron validados como prioridades en el levantamiento, y en el caso del chat demandarían infraestructura de comunicación en tiempo real con WebSockets que excede la capacidad del prototipo. Finalmente, la sincronización con calendarios externos depende de APIs de terceros sin control institucional. Las cuatro exclusiones operativas restantes (despliegue productivo, migración de datos, capacitación y mantenimiento) se inscriben en una fase de implementación posterior al proyecto de titulación, donde concurren actores institucionales que no participan del proyecto de prototipado. La Tabla 2 sintetiza la clasificación MoSCoW completa de los veintitrés requerimientos levantados durante el diagnóstico con las entrevistas institucionales, cuyo proceso de levantamiento se detalla en la sección 2.9, distinguiendo los implementables de los diferidos.

Tabla 2. Matriz MoSCoW y estado de avance del alcance del prototipo MRNA. Nota. Elaboración propia — Consolidado de Requerimientos Funcionales e Historias de Usuario, 2026.

Código	Nombre del Requerimiento	MoSCoW	Alcance	Estado de avance (cierre Cadencia 5)
RF-001	Carga masiva de datos consolidados desde el MRNA hacia el SIGA	Must	Implementar	Implementado
RF-002	Sincronización de datos desde Moodle hacia el MRNA	Must	Implementar	Implementado
RF-003	Interfaz para recibir datos de Moodle hacia el MRNA	Must	Implementar	Implementado
RF-004	Dashboard con histograma visual de rendimiento	Should	Implementar	Pendiente
RF-005	Gestión de roles y permisos	Must	Implementar	Implementado
RF-006	Notificaciones automáticas por bajo rendimiento	Should	Implementar	Parcial
RF-007	Filtrado y búsqueda avanzada	Should	Implementar	Implementado
RF-008	Generación de reportes (PDF/Excel/CSV)	Must	Implementar	Parcial
RF-009	Firma electrónica de reportes (Quipux)	Should	Diferido	No aplica
RF-010	Validación automática de errores antes de carga masiva	Must	Implementar	Implementado
RF-011	Módulo de ayuda interactiva	Could	Diferido	No aplica
RF-012	Acceso restringido por período activo	Must	Implementar	Implementado
RF-013	Carga masiva de usuarios	Should	Implementar	Implementado
RF-014	Notificaciones programadas	Should	Implementar	Parcial
RF-015	Junta académica con gracia de 48 horas	Should	Implementar	Parcial
RF-016	Portal de noticias académicas	Could	Diferido	No aplica
RF-017	Chat integrado docente-estudiante	Could	Diferido	No aplica
RF-018	Auditoría de notas y asistencia	Should	Implementar	Implementado
RF-019	Historial de cambios para notas	Could	Opcional	No aplica
RF-020	Gestión de feriados y eventos especiales	Could	Opcional	No aplica

Código	Nombre del Requerimiento	MoSCoW	Alcance	Estado de avance (cierre Cadencia 5)
RF-021	Evaluación continua (tareas, quizzes)	Could	Opcional	No aplica
RF-022	Calendario académico	Could	Opcional	No aplica
RF-023	Integración con Google Calendar/Outlook	Could	Diferido	No aplica

2.8. Análisis del proceso actual

La sección 2.3 introdujo el problema en términos generales y la Figura 1 mostró el flujo a vista general. Esta sección lo detalla etapa por etapa, con base en la plantilla institucional de notas que el ISTA distribuye a sus docentes y en las entrevistas a tres profesores activos del Instituto. El análisis permite identificar puntos concretos donde se acumulan las problemáticas que se sintetizan en la Tabla 3 al cierre de la sección.

Etapas 1: Registro inicial en Moodle. El primer paso del proceso es el registro de las calificaciones por componente en la plataforma Moodle institucional, denominada internamente E-Virtual. La Docente 1 explicó que en esta etapa se evalúan los componentes prácticos, autónomos y de docencia para las tres etapas del periodo (primer parcial, segundo parcial y recuperación) y que es allí donde efectivamente conviven los estudiantes con sus actividades, foros, tareas y cuestionarios (Docente 1, comunicación personal, 2025). Sin embargo, esta etapa ya muestra fricciones tempranas. El Docente 3 señaló que algunos foros y actividades de Moodle no admiten calificaciones con decimales, lo que obliga a redondeos manuales que distorsionan la puntuación final del estudiante (Docente 3, comunicación personal, 2025).

Etapas 2: Traslado a la plantilla institucional de Excel. La segunda etapa es la consolidación en una plantilla Excel estandarizada que la coordinación académica del ISTA distribuye a cada docente al inicio del periodo. La plantilla examinada en este trabajo está estructurada en una hoja maestra con el distributivo docente, cinco hojas por asignatura denominadas M1 a M5, y tres hojas adicionales para las juntas académicas de Parcial 1, Parcial 2 y Recuperación. Las hojas M concentran los datos por componente y calculan los promedios con fórmulas pre-cargadas, mientras que las hojas de Junta Académica recogen los indicadores que se discuten formalmente: promedio del parcial, estudiantes con rendimiento inferior al 70%, estudiantes con asistencia inferior al 75%,

distribución de niveles de desempeño (Excelente, Muy bueno, Bueno, Aprobado, Reprobado) y avance porcentual del Plan de Enseñanza-Aprendizaje. La Docente de Desarrollo de Software describió en sus propias palabras el costo operativo de esta etapa: descargar las notas desde Moodle, abrir la plantilla, calcular los promedios y transcribir manualmente los datos al Excel institucional (Docente de Desarrollo de Software, comunicación personal, 2025).

Etapa 3: Carga al SIGA. La tercera etapa cierra el ciclo con la carga manual de los datos consolidados al Sistema Integrado de Gestión Académica, responsabilidad que recae en el coordinador de carrera tras la junta académica. El responsable de TIC del Instituto confirmó en su entrevista que el SIGA no expone una API pública para institutos superiores tecnológicos, por lo que la carga se realiza directamente sobre la interfaz web del sistema, registro por registro (Responsable de TIC, comunicación personal, 2025). En este punto, la misma información ha viajado desde Moodle hasta SIGA atravesando la plantilla Excel, sin que ninguno de los pasos intermedios cuente con un mecanismo de validación cruzada o de respaldo automático.

Tiempo administrativo agregado y problemáticas derivadas. Cuando se suma el costo de las tres etapas a lo largo del semestre, el resultado es significativo. La Docente 1 reportó dedicar aproximadamente 26 horas semanales a tareas administrativas de gestión académica (Docente 1, comunicación personal, 2025). El Docente 3 reportó entre 1 y 1,5 horas en semanas regulares y entre 6 y 8 horas en semanas de cierre de parcial (Docente 3, comunicación personal, 2025). La Docente de Desarrollo de Software reportó entre 15 y 20 minutos diarios solo para registrar la asistencia en SIGA (Docente de Desarrollo de Software, comunicación personal, 2025). Más allá de los tiempos, el levantamiento de campo permitió identificar un conjunto de problemáticas recurrentes que compromete la integridad de la información, la trazabilidad de los cambios y la capacidad institucional de respuesta oportuna. La Tabla 3 sintetiza esas problemáticas y la fuente primaria de cada una.

Estas tres cifras miden cosas distintas y no son intercambiables entre sí, la de la Docente 1 agrega el conjunto de tareas administrativas de una semana típica, la del Docente 3 distingue entre semanas regulares y semanas de cierre de parcial, y la de la Docente 2 se limita al registro diario de asistencia en el SIGA. La comparación de desempeño del Capítulo 4 retoma específicamente la estimación de seis a ocho horas del Docente 3 para el cierre de un parcial completo, por ser la que corresponde en alcance a la tarea que el prototipo automatiza y mide.

Tabla 3. Problemáticas identificadas en el proceso actual de registro de notas y asistencia del ISTA. Nota. Elaboración propia con base en entrevistas a docentes, coordinadores y autoridades del Instituto, 2025.

#	Dolor identificado	Evidencia primaria
1	Triple digitación de calificaciones entre Moodle, Excel y SIGA	Docente 3, Coordinador 1
2	Carga administrativa elevada, que en algunos casos alcanza 26 horas semanales por docente	Docente 1
3	Cálculo manual de promedios sin validación automática, propenso a errores	Docente 3, Docente de Desarrollo de Software
4	Bloqueo del SIGA por registro tardío de asistencia, con trámite de desbloqueo ante coordinación	Docente de Desarrollo de Software, Docente 1
5	Inconsistencia entre las listas de estudiantes en Moodle, Excel y SIGA	Docente de Desarrollo de Software
6	Pérdida de archivos Excel por la ausencia de un respaldo centralizado	Vicerrector del ISTA
7	Ausencia de mecanismos de auditoría sobre cambios en notas y asistencia	Coordinador de Desarrollo de Software
8	Inexistencia de alertas tempranas por bajo rendimiento académico o ausentismo	Vicerrector del ISTA
9	Limitaciones de Moodle para registrar calificaciones decimales en algunos componentes	Docente 3
10	Uso de canales informales como WhatsApp para comunicación académica formal	Vicerrector del ISTA

El mapa que resulta de este análisis define con precisión los frentes sobre los que el prototipo MRNA debe actuar. La automatización del flujo descrito en la Figura 1 y la mitigación de las problemáticas listadas en la Tabla 3 constituyen los criterios concretos contra los que se contrastará la validación funcional en el Capítulo 3 y la de usabilidad y desempeño en el Capítulo 4.

2.9. Levantamiento y análisis de requerimientos

El levantamiento de requerimientos del prototipo MRNA combinó dos fuentes complementarias: ocho entrevistas semiestructuradas con actores institucionales del ISTA, ya descritas en la sección 2.2, y el análisis documental de los artefactos que sostienen el proceso actual, en particular la plantilla institucional de notas comentada en la sección anterior. Esta sección presenta el resultado de ambas fuentes en cuatro bloques: la descripción de los stakeholders consultados, el catálogo completo de requerimientos funcionales obtenidos del levantamiento, la matriz de trazabilidad que vincula cada requerimiento con su fuente primaria y el catálogo de requerimientos no funcionales construido por análisis derivativo.

2.9.1. Stakeholders entrevistados

La selección de los stakeholders entrevistados no fue aleatoria. Siguió un muestreo intencional o de conveniencia, un enfoque habitual en la investigación cualitativa cuando el objetivo no es generalizar estadísticamente sino capturar en profundidad la experiencia de quienes conocen de primera mano el proceso que se estudia. Bajo ese criterio, se buscó cubrir los tres niveles de participación que intervienen en la gestión académica del Instituto, el nivel directivo, encargado de la planificación estratégica institucional, el nivel intermedio, responsable de la coordinación académica de las carreras, y el nivel operativo, que reúne a docentes y al área técnica de soporte. La selección priorizó además a personas con experiencia directa sobre el flujo manual descrito en la sección 2.8, lo que permitió contrastar la perspectiva de quien diseña el proceso con la de quien lo ejecuta cotidianamente. Esta combinación de niveles y de experiencia práctica es lo que le da validez de contenido al levantamiento, no la cantidad de personas entrevistadas.

2.9.2. Catálogo de requerimientos funcionales

A partir de las entrevistas y del análisis documental se identificaron veintitrés requerimientos funcionales que cubren los frentes principales de la gestión académica de notas y asistencia: integración con SIGA y Moodle, registro y consolidación de calificaciones, control de acceso y seguridad, generación de reportes, notificaciones, validación de datos, gestión de períodos académicos y auditoría. La distribución de los requerimientos refleja la naturaleza compuesta del problema. Once requerimientos se concentran en el flujo central de notas y asistencia, cuatro se refieren a las funciones de

integración entre plataformas, tres atienden la gestión de usuarios y permisos, dos se orientan a la generación y firma de reportes, y los tres restantes cubren funciones complementarias como ayuda interactiva, calendario académico y portal de comunicación. El catálogo completo, con la denominación de cada requerimiento y su prioridad MoSCoW, fue presentado en la Tabla 2 de la sección 2.7. La descripción detallada en formato Como/Quiero/Para de cada requerimiento, junto con sus criterios de aceptación expresados en notación Given-When-Then, se conserva en el Anexo C de este trabajo.

2.9.3. Matriz de trazabilidad RF ↔ entrevistas ↔ historias de usuario

La trazabilidad bidireccional entre cada requerimiento funcional y su fuente primaria es un criterio de calidad del levantamiento, ya que permite justificar cualquier decisión de alcance frente a evidencia documentada y prevenir requerimientos sin soporte. Para los catorce requerimientos clasificados como Must-have y Should-have del prototipo, la Tabla 4 mapea el código del requerimiento, la entrevista o entrevistas en las que se sustenta su origen y la historia de usuario que lo descompone funcionalmente.

Tabla 4. Matriz de trazabilidad de los requerimientos funcionales core del MRNA. Nota. Elaboración propia con base en DOC-06 y entrevistas a stakeholders, 2025.

Código RF	Origen primario (entrevistas)	Historia de usuario
RF-001	Responsable de TIC, Coordinador de Desarrollo de Software	HU-001
RF-002	Docente 1, Docente de Desarrollo de Software, Docente 3, Coordinador de Desarrollo de Software, Coordinador 1	HU-002
RF-003	Docente 1, Docente 3, Coordinador de Desarrollo de Software	HU-003
RF-004	Vicerrector, Coordinador de Desarrollo de Software	HU-004
RF-005	Vicerrector, Coordinador de Desarrollo de Software, Coordinador 2	HU-005
RF-006	Vicerrector, Coordinador 1	HU-006
RF-007	Vicerrector, Coordinador de Desarrollo de Software	HU-007
RF-008	Coordinador de Desarrollo de Software, Coordinador 1, Vicerrector	HU-008
RF-010	Responsable de TIC, Coordinador de Desarrollo de Software	HU-009

Código RF	Origen primario (entrevistas)	Historia de usuario
RF-012	Docente de Desarrollo de Software, Docente 1, Coordinador 2	HU-010
RF-013	Responsable de TIC, Coordinador 2	HU-011
RF-014	Vicerrector, Coordinador 2	HU-012
RF-015	Coordinador 2	HU-013
RF-018	Coordinador de Desarrollo de Software, Docente 1	HU-014

2.9.4. Catálogo de requerimientos no funcionales

A diferencia de los veintitrés requerimientos funcionales, levantados directamente con stakeholders, los requerimientos no funcionales del prototipo se construyeron mediante análisis derivativo, sin una ronda de entrevistas propia. Conviene aclarar el orden real en que aparecieron sus fuentes, porque no todas llegaron al mismo tiempo. Primero estuvieron el anteproyecto aprobado, el marco normativo nacional aplicable a la educación superior y las declaraciones implícitas en las ocho entrevistas a stakeholders ya consolidadas en la sección 2.9.1. Esas tres fuentes existían antes de tomar cualquier decisión de arquitectura, y fueron las que orientaron decisiones tempranas del proyecto como la elección de microservicios o del entorno local de Moodle, documentadas en los Architecture Decision Records. Después, ya con esas decisiones tomadas, se formalizó el catálogo completo de requerimientos no funcionales, y en ese momento se sumaron dos fuentes adicionales, las restricciones esenciales al catálogo de requerimientos funcionales y los propios Architecture Decision Records, que en esta segunda etapa no funcionan como origen del requerimiento sino como consecuencia técnica que había que documentar formalmente. Por ejemplo, haber elegido microservicios con base de datos compartida obliga a declarar un requerimiento de rendimiento que no existía como preocupación explícita antes de esa decisión. No se trata entonces de haber inventado requerimientos para justificar decisiones ya tomadas, sino de un proceso en dos momentos, unas fuentes anteceden y motivan las decisiones de diseño, y otras las suceden y documentan sus implicaciones. Ese patrón es reconocido en la literatura de ingeniería de requerimientos como propio de la naturaleza emergente de los requerimientos no funcionales.

El catálogo resultante reúne dieciséis requerimientos no funcionales distribuidos en siete categorías de la norma ISO/IEC 25010 sobre calidad de software: seguridad, rendimiento, fiabilidad, disponibilidad, mantenibilidad, portabilidad y usabilidad. Cada requerimiento cuenta con un código único, una descripción operativa y un criterio cuantitativo de aceptación. La Tabla 5 resume el catálogo; el detalle completo, se reproduce en el Anexo N. La sección 3.11 del Capítulo 3 retoma este catálogo y mapea cada requerimiento con la decisión arquitectónica concreta que lo materializa, y la sección 4.4 del Capítulo 4 reporta los resultados de la validación cuantitativa contra los criterios de aceptación de cada uno.

Tabla 5. Catálogo de requerimientos no funcionales del MRNA. Nota. Elaboración propia con base en DOC-08 — Levantamiento de Requerimientos No Funcionales, 2026.

Código	Categoría	Descripción breve	Criterio cuantitativo
RNF-01	Seguridad	Autenticación obligatoria con JWT	Endpoints sin token devuelven HTTP 401
RNF-02	Seguridad	Control de acceso basado en roles	Roles no autorizados devuelven HTTP 403
RNF-03	Seguridad	Hashing de contraseñas y manejo seguro de credenciales	Cero contraseñas en texto plano
RNF-04	Seguridad	Auditoría de cambios en notas y asistencia	100% de modificaciones registradas en bitácora
RNF-05	Rendimiento	Latencia aceptable en endpoints CRUD	p95 menor a 2 s con 50 usuarios concurrentes
RNF-06	Rendimiento	Throughput sostenido en horarios pico	100 peticiones/s sostenidas en el api-gateway
RNF-07	Rendimiento	Generación rápida de reportes consolidados	Reporte completo de período en menos de 30 s
RNF-08	Fiabilidad	Manejo controlado de fallos de servicios externos	Cero excepciones no controladas visibles
RNF-09	Fiabilidad	Tasa baja de errores en operación normal	Errores HTTP 5xx menores al 1% del total

Código	Categoría	Descripción breve	Criterio cuantitativo
RNF-10	Disponibilidad	Reproducibilidad del entorno con un solo comando	docker compose up levanta todo en menos de 5 min
RNF-11	Mantenibilidad	Cobertura mínima de pruebas unitarias	Cobertura $\geq 60\%$ en services y controllers
RNF-12	Mantenibilidad	Convenciones de código y baja deuda técnica	Cero violaciones críticas; complejidad < 10
RNF-13	Mantenibilidad	Documentación navegable de endpoints REST	100% de endpoints documentados en OpenAPI
RNF-14	Portabilidad	Ejecución en distintos sistemas operativos	Verificado en Windows, Linux y macOS
RNF-15	Usabilidad	Percepción aceptable por docentes y coordinadores	Puntuación SUS global ≥ 68 sobre 100
RNF-16	Usabilidad	Accesibilidad web mínima	Cumplimiento WCAG 2.1 niveles A y AA

2.10. Metodología de la investigación

Esta sección describe el marco metodológico que sustenta el proyecto de titulación. Cubre el tipo y nivel de la investigación, el enfoque y diseño adoptados, la población y la muestra sobre la que se aplican los instrumentos, los instrumentos de campo previstos junto con su proceso de validación, y la operacionalización de las variables sobre las que se construirán las mediciones del Capítulo 4.

2.10.1. Tipo y nivel de investigación

El presente trabajo se inscribe como una investigación aplicada, en la medida en que se orienta a la solución de un problema institucional concreto del ISTA mediante el desarrollo de un prototipo funcional que reemplaza un proceso manual existente. Su nivel es descriptivo y comparativo. Descriptivo en la fase de diagnóstico situacional, donde se caracteriza el flujo actual del registro de notas y asistencia y se identifican sus puntos críticos. Comparativo en la fase de validación, donde se contrastan los indicadores del proceso manual contra los del flujo automatizado por el prototipo, sin que ese contraste

sea un experimento en sentido estricto, porque no hay asignación aleatoria de participantes, no existe un grupo de control independiente del grupo intervenido, y la condición previa del proceso manual se toma de estimaciones declaradas por los docentes entrevistados, no de una medición directa bajo las mismas condiciones que la prueba del prototipo, esa diferencia de método ya se explica con más detalle en la sección 2.10.2. A este carácter se suma un componente tecnológico marcado, asociado al desarrollo del software con el stack definido en el ADR-001 del proyecto.

2.10.2. Enfoque y diseño metodológico

El enfoque del trabajo es mixto. La fase de levantamiento de requerimientos se sostiene sobre técnicas cualitativas, en particular las ocho entrevistas semiestructuradas con stakeholders del ISTA descritas en la sección 2.9. La fase de validación, en cambio, se sostiene sobre técnicas cuantitativas, ya que recolecta indicadores numéricos sobre tiempos de ejecución, tasas de error, latencia de operaciones críticas y percepción de usabilidad. El diseño metodológico combina dos fuentes de naturaleza distinta que conviene no confundir, primero, la línea base del proceso manual proviene de estimaciones declaradas por los docentes entrevistados en la sección 2.8, no de una medición cronometrada directa, mientras que el desempeño del prototipo se mide con una prueba técnica reproducible (sección 4.3). Por tanto, no se trata de un experimento controlado con pretest y postest bajo condiciones idénticas, sino de un contraste entre una estimación cualitativa del proceso actual y una medición cuantitativa del prototipo bajo las condiciones descritas en el Capítulo 4, esa diferencia de método se declara explícitamente al interpretar los resultados comparativos de la sección 4.3.

2.10.3. Población y muestra

La aplicación de los instrumentos previstos requiere identificar tres grupos poblacionales distintos. El primero corresponde a los stakeholders entrevistados durante el levantamiento de requerimientos, conformado por ocho actores institucionales que representan los tres niveles de participación en la gestión académica. Para este grupo se trabajó con la población total, dado su tamaño reducido. El segundo grupo corresponde a los docentes activos del ISTA considerados para la fase de validación de usabilidad del prototipo. Dado el carácter exploratorio de las pruebas y la disponibilidad limitada del cuerpo docente para sesiones individuales, se planificó una muestra de tres docentes. En la práctica, las restricciones de tiempo del cronograma institucional impidieron coordinar

sesiones adicionales antes del cierre de documentación de la Cadencia 5, por lo que la validación de usabilidad se ejecutó finalmente con un único evaluador, el Docente de Desarrollo de Software. La sección 4.2.2 detalla el perfil de ese evaluador y las razones de su selección, y la sección 4.7 recoge esta desviación como limitación del estudio. El tercer grupo corresponde a la coordinación académica, para la que se previó al menos un participante en la validación funcional del flujo de consolidación, sesión que finalmente no pudo concretarse antes del cierre de documentación, tal como se recoge en la sección 4.7. La Tabla 6 sintetiza esta información.

Tabla 6. Población y muestra del estudio. Nota. Elaboración propia, 2026.

Grupo	Descripción	Tamaño total	Muestra y técnica
Stakeholders del levantamiento	Vicerrector, coordinadores, docentes y responsable de TIC del ISTA	8	8 (población total entrevistada)
Docentes para validación de usabilidad	Docentes activos del ISTA con experiencia en el flujo manual de notas y asistencia	49	3 docentes planificados por conveniencia; 1 docente participó efectivamente (ver 2.10.3 y 4.2.2)
Coordinación académica para validación funcional	Coordinadores académicos del ISTA	1	1 coordinador

Las mediciones de tiempo, tasa de errores y latencia no requieren una muestra de personas, sino de operaciones realizadas sobre el prototipo. Para esos indicadores el universo lo constituye el conjunto de transacciones ejecutadas durante las pruebas funcionales y de desempeño del prototipo, tomadas en su totalidad sin muestreo.

2.10.4. Instrumentos y validación

El trabajo prevé dos instrumentos de campo. El primero es el guion de entrevista semiestructurada que se utilizó durante la fase de levantamiento, conformado por entre nueve y doce preguntas abiertas adaptadas al rol del entrevistado, con espacio para profundizar en temas emergentes. La versión formalizada del guion se incluye en el Anexo A del trabajo. El segundo instrumento es el protocolo de validación de usabilidad

del prototipo, que se aplica en la fase de validación del Capítulo 4. Este protocolo combina aproximadamente ocho tareas guiadas que recorren los flujos críticos del sistema, junto con el cuestionario estandarizado System Usability Scale en su versión de diez preguntas con escala de uno a cinco.

El proceso de validación de ambos instrumentos no quedó documentado de forma formal, tal como se había previsto inicialmente. El guion de entrevista no fue revisado por escrito por el director del proyecto antes de aplicarlo. El acompañamiento recibido durante esa etapa fue de tipo verbal y general. El protocolo de validación de usabilidad sí recibió comentarios informales antes de aplicarse, de parte del Docente de Desarrollo de Software. Esa es la misma persona que después actuó como el único evaluador de la fase de validación descrita en la sección 4.2.2, y esos comentarios tampoco quedaron registrados en ningún documento. Esta falta de documentación en la validación de instrumentos se reconoce como una limitación del estudio, y queda como mejora concreta para un trabajo futuro que replique o extienda esta investigación, formalizar por escrito el juicio de expertos.

2.10.5. Medición de variables del estudio

Esta subsección define en términos prácticos cómo se va a medir cada variable del estudio durante la fase de validación. La idea es que cuando en el Capítulo 4 se reporten resultados como "el tiempo administrativo se redujo en X%", el lector sepa exactamente qué se midió, cómo se midió y con qué herramienta. La variable independiente es la implementación del prototipo MRNA en sí misma, entendida como la presencia o ausencia del sistema funcional. Sobre ella se levantan cuatro variables dependientes que reflejan distintas dimensiones del impacto del prototipo: el tiempo que un docente dedica al registro de notas y asistencia, la cantidad de errores que aparecen al consolidar las calificaciones, la percepción de usabilidad reportada por los usuarios y el tiempo de respuesta del sistema. La Tabla 7 muestra la forma en que cada variable se medirá.

Tabla 7. Medición de variables del estudio. Nota. Elaboración propia, 2026.

Variable	Cómo se mide	Con qué herramienta
Implementación del prototipo MRNA	Presencia o ausencia del sistema funcional en el ISTA	Demostración funcional del prototipo

Variable	Cómo se mide	Con qué herramienta
Tiempo administrativo por docente	Minutos que tarda un docente en registrar las notas y la asistencia de una asignatura	Cronómetro aplicado durante las pruebas con docentes
Tasa de errores en consolidación	Porcentaje de errores detectados sobre el total de registros consolidados	Auditoría manual de los registros del prototipo
Percepción de usabilidad	Puntaje obtenido en el cuestionario SUS, en escala de 0 a 100	Cuestionario SUS aplicado a los docentes participantes
Latencia de operaciones críticas	Tiempo de respuesta del sistema en milisegundos	Pruebas de carga con Apache JMeter o k6

2.11. Metodología de desarrollo de software

Para la construcción del prototipo MRNA se adoptó Kanban como metodología de gestión del trabajo, en sustitución de las opciones más tradicionales como Scrum o Extreme Programming. La decisión fue formalizada en el cuarto Architecture Decision Record del proyecto y se sustenta en tres condiciones específicas del contexto. La primera es que el desarrollo lo realiza un solo estudiante, lo que vuelve innecesarias las ceremonias y los roles diferenciados de Scrum como Product Owner, Scrum Master y equipo de desarrollo. La segunda es que el cronograma tiene fechas inamovibles, con la versión evaluada cerrada el 31 de julio de 2026 y sustentación entre el 1 y el 11 de septiembre aproximadamente, lo que demanda una metodología capaz de absorber imprevistos sin comprometer compromisos cerrados. La tercera es que los bloqueos externos son frecuentes en proyectos de titulación, ya sea por retroalimentación pendiente del tutor, trámites administrativos o limitaciones de acceso a recursos institucionales. Como sintetizan [18] en su revisión sistemática sobre Kanban aplicado al desarrollo de software, esta metodología ofrece flexibilidad para incorporar cambios, permite visualizar el progreso de forma inmediata y limita la sobrecarga de tareas simultáneas mediante el control del trabajo en progreso, lo cual resulta particularmente útil cuando un único desarrollador asume la totalidad de la implementación. De aquí en adelante, este documento usa cadencia para nombrar cada ciclo bisemanal de planificación y revisión del proyecto, y reserva sprint para los elementos que llevan ese nombre de forma literal en la herramienta (la columna y el campo "Sprint" de GitHub Projects) o para la práctica propia de Scrum.

El tablero del proyecto se implementa en GitHub Projects sobre el repositorio privado tesis-mrna y se estructura en seis columnas que reflejan los estados del flujo de trabajo. La columna Backlog reúne todas las tareas identificadas y priorizadas, pero sin fecha de inicio definida. La columna de Sprint Actual, (nombre que trae por defecto la plantilla de GitHub Projects) contiene las tareas seleccionadas para la cadencia de dos semanas en curso. La columna En Desarrollo agrupa las tareas activamente en construcción y aplica un límite de trabajo en progreso de tres tarjetas simultáneas. La columna En Pruebas recoge las tareas completadas que están en proceso de verificación, también con límite de tres tarjetas. La columna Terminado registra las tareas verificadas y cerradas, y es la fuente principal de las métricas del proyecto. Finalmente, la columna Bloqueado visibiliza las tareas que no pueden avanzar por causa externa, y obliga a registrar la razón del bloqueo para alimentar el análisis posterior. Como argumentan [17], la visualización mediante tableros Kanban facilita la identificación de cuellos de botella y mantiene la trazabilidad del trabajo en proyectos de software de pequeña y mediana escala. La Figura 2 muestra una captura del tablero del proyecto MRNA al inicio del desarrollo, con la configuración descrita y las primeras tarjetas de gestión inicial.

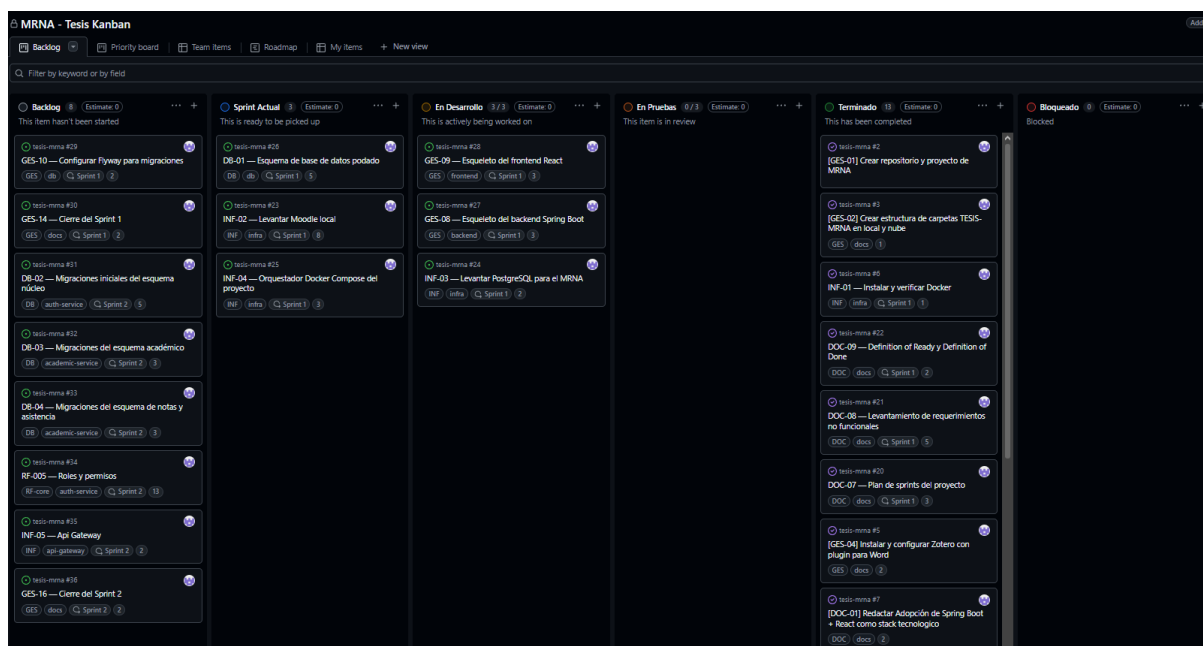


Figura 2. Tablero Kanban del proyecto MRNA configurado en GitHub Projects para la gestión del desarrollo. Nota. Captura propia, 2026.

A lo largo del ciclo de desarrollo, hasta el cierre de la cadencia 5 que constituye el corte de documentación de este trabajo, se recolectaron cinco métricas que sustentan el análisis cuantitativo del Capítulo 4. El lead time mide el tiempo total desde que una tarea entra al

Backlog hasta que se marca como Terminada, y refleja el tiempo real de entrega de cada funcionalidad. El cycle time mide el tiempo desde que una tarea entra a En Desarrollo hasta su finalización, y permite evaluar la eficiencia del proceso de construcción sin contar la espera previa. El throughput contabiliza el número de tareas terminadas por ciclo de dos semanas, lo cual permite comparar la productividad entre ciclos. El WIP efectivo registra el número promedio de tareas simultáneas en cada columna a lo largo del tiempo, y verifica el cumplimiento del límite de tres tarjetas. Finalmente, el Diagrama de Flujo Acumulado muestra la evolución acumulada del tablero y se incluye en la sección 4.5 del Capítulo 4 como evidencia visual del comportamiento del flujo.

Las políticas explícitas de paso entre columnas se formalizan en el documento DOC-09 del proyecto. La Definition of Ready establece siete criterios que una tarjeta debe cumplir para pasar de Sprint Actual a En Desarrollo, entre ellos contar con un código asignado según la convención del proyecto, tener entre tres y cinco criterios de aceptación verificables y respetar el límite WIP de la columna destino. La Definition of Done establece siete criterios para que una tarjeta pase de En Pruebas a Terminado, incluyendo la integración del código a la rama principal mediante Pull Request, la ejecución exitosa de pruebas unitarias y de integración, y la actualización de la documentación correspondiente. Estas políticas evitan la subjetividad en las transiciones y aseguran que las métricas recolectadas se sostengan sobre criterios homogéneos a lo largo de todo el desarrollo.

2.12. Síntesis del capítulo

Este capítulo dejó establecido el escenario completo sobre el que se construirá la propuesta de solución. Se caracterizó al ISTA como cliente institucional y se documentaron los actores que intervienen en su gestión académica de notas y asistencia. Se planteó formalmente el problema (la triple digitación entre Moodle, Excel y SIGA, junto con las problemáticas asociadas al proceso manual) y se justificó la pertinencia del proyecto desde sus dimensiones teórica, práctica y de factibilidad. Se establecieron los objetivos en los términos literales del anteproyecto aprobado, las preguntas de investigación que orientan el trabajo, el alcance del prototipo con sus dieciocho requerimientos funcionales implementables y los dieciséis requerimientos no funcionales asociados, y se describieron las metodologías de investigación y de desarrollo de software que sustentan el proceso. El siguiente capítulo desarrolla la propuesta técnica que da

respuesta al problema diagnosticado, comenzando por la arquitectura general del sistema y avanzando hacia el diseño detallado de cada uno de sus componentes.

Capítulo 3. Desarrollo y Validación de la Propuesta

3.1. Introducción del capítulo

Este tercer capítulo pasa del diagnóstico a la propuesta: documenta las decisiones de diseño y los resultados de implementación del prototipo MRNA. Arranca con una visión de alto nivel que permite entender el sistema antes de entrar en detalles, y desde ahí avanza hacia la arquitectura de microservicios adoptada, el stack tecnológico seleccionado y el diseño de la base de datos con su lógica embebida. Se describen los seis servicios que componen el sistema y la API que los conecta, seguidos por el diseño de la interfaz de usuario y la estrategia de seguridad implementada. Se dedica también una sección a la integración con Moodle y con el SIGA, que representaba uno de los retos técnicos centrales del proyecto. El capítulo cierra mostrando cómo cada requerimiento no funcional se tradujo en una decisión arquitectónica concreta, cómo se gestionó el desarrollo mediante Kanban y cuál es el estado del prototipo al momento de escritura.

3.2. Visión general de la solución MRNA

El capítulo anterior documentó el proceso que el ISTA sigue hoy para registrar calificaciones y asistencia, el docente ingresa los datos en el aula virtual Moodle, los exporta a una plantilla Excel, calcula los promedios por componente y entrega los resultados al coordinador, quien a su vez los carga en el SIGA. El prototipo MRNA existe para eliminar ese recorrido manual. Su propósito es ofrecer un espacio único donde las notas y la asistencia se registren una sola vez, se sincronicen automáticamente con Moodle y, al momento de reportar al SIGA, generen el archivo listo para su carga. La carga en el SIGA en sí sigue siendo manual, porque ese sistema no ofrece otra vía, lo que el MRNA elimina es la reconstrucción de los datos desde cero y la digitación repetida, no el paso final de subir el archivo.

La forma más sencilla de entender el sistema es verlo como un puente. De un lado está Moodle, la plataforma que el ISTA utiliza para sus aulas virtuales, conocida internamente como E-Virtual. Del otro está el SIGA, el sistema nacional que administra la SENESCYT y al que el Instituto está obligado a reportar su información académica al cierre de cada período. El MRNA se conecta con Moodle para importar automáticamente las calificaciones que los docentes ya registraron en sus cursos, las procesa, las almacena y, cuando llega el momento de reportar al SIGA, genera el archivo en el formato exacto que

ese sistema requiere. El coordinador lo descarga y lo sube, sin tener que reconstruir los datos desde cero.

Cuatro tipos de usuarios interactúan con el sistema. Los docentes son quienes lo utilizan con mayor frecuencia en el día a día ya que registran asistencia directamente en el MRNA, revisan las calificaciones importadas desde Moodle, las ajustan si es necesario y confirman el cierre del parcial cuando el período lo permite. Los coordinadores académicos acceden a una vista más amplia, consultan el rendimiento de los paralelos bajo su cargo y generan desde el mismo sistema los reportes que necesitan para las juntas académicas y para la carga al SIGA, la detección de bajo rendimiento y ausentismo, prevista en el diseño del sistema (RF-006), se ejecuta hoy bajo demanda y no de forma automática, tal como se detalla en la sección 3.6.5. El administrador del sistema gestiona las cuentas de usuario, configura la conexión con Moodle y revisa los registros de sincronización para detectar cualquier inconveniente. Los directivos, por su parte, tienen previsto en el diseño un acceso a indicadores agregados (promedios por carrera, tasas de asistencia, estado de los períodos académicos) sin necesidad de solicitar información a las coordinaciones, esa vista quedó planteada pero no se construyó dentro del alcance del prototipo, tal como se documenta en la sección 3.8.2, y hoy el rol directivo conserva acceso al reporte general de notas.

El sistema no pretende reemplazar ni a Moodle ni al SIGA. Moodle sigue siendo la plataforma pedagógica del Instituto y el SIGA continúa siendo el repositorio oficial ante la SENESCYT. Lo que el MRNA hace es reducir el trabajo manual que hoy ocurre en el espacio entre los dos. La Figura 3 ilustra esta relación desde afuera: el sistema como una caja que recibe información de Moodle, la pone a disposición de cuatro tipos de usuarios a través de una interfaz única, y genera las salidas que se cargan al SIGA.



Figura 3. Diagrama de contexto del sistema MRNA (C4 nivel 1). Nota. Elaboración propia con asistencia de inteligencia artificial, 2026.

3.3. Arquitectura del sistema

3.3.1. Vista lógica de microservicios

Cuando se definió cómo estructurar el MRNA, la arquitectura de microservicios ya venía fijada por el anteproyecto aprobado, así que la comparación que sigue no evalúa si esa elección fue acertada, sino qué costo concreto tiene frente a las alternativas que un proyecto de estas características habría podido considerar. Frente a un monolito, que reuniría la autenticación, el registro de notas, la sincronización con Moodle, los reportes y las notificaciones en un mismo proceso desplegable, la ventaja medible para el MRNA no es que un monolito se caiga entero ante cualquier error (un monolito con buena separación interna de módulos puede aislar fallas igual de bien) [8], sino que enlaza el ciclo de despliegue de toda la lógica académica al de los dos componentes con más probabilidad de cambiar durante el proyecto (el adaptador de Moodle y el de SIGA). Si cambiara la versión de Moodle o si SIGA llegara a habilitar una API propia, un monolito habría obligado a redespregar el sistema completo, con microservicios, ese ajuste queda contenido en el sync-service.

También se evaluó una arquitectura orientada a servicios (SOA). El bus central de mensajería (ESB) es la forma con la que suele asociarse a SOA en la práctica, pero no es un requisito que la arquitectura exija por definición, es posible construir un sistema SOA con integración directa entre servicios, sin ese componente central [27]. Para el MRNA la razón para descartar SOA no fue esa, sino que su forma habitual agrupa los servicios de negocio bajo una capa de integración compartida, un acoplamiento que no encaja con la meta del proyecto de que Moodle y SIGA se traten como canales aislados, reemplazables sin afectar al resto del sistema. Se descartó igualmente el enfoque serverless, que habría eliminado la gestión de infraestructura propia, pero habría hecho al sistema dependiente de servicios en la nube con costo operativo, algo incompatible con el entorno local de desarrollo que se definió.

La decisión adoptada, documentada en ADR-002, fue organizar el MRNA en seis microservicios independientes, cada uno con una única responsabilidad. Esta elección encuentra respaldo en la literatura: El autor en [7] sostiene que los sistemas compuestos por servicios con límites bien definidos son más fáciles de mantener y de hacer evolucionar de manera independiente. En la misma línea, los autores de [8] confirman, a partir de una revisión sistemática de treinta y cinco estudios, que la descomposición en microservicios mejora la escalabilidad y el aislamiento de fallos frente a la arquitectura monolítica tradicional. En el ámbito educativo, los autores de [23] validaron un enfoque similar al combinar microservicios con una puerta de enlace en un sistema de aprendizaje electrónico, lo que refuerza la pertinencia de esta arquitectura para el contexto del MRNA. La Tabla 8 presenta los seis servicios del sistema junto con la responsabilidad que cumple cada uno y el puerto que expone en el entorno de desarrollo.

Tabla 8. Microservicios del prototipo MRNA, responsabilidades y puertos de desarrollo. Nota. Elaboración propia, 2026.

Servicio	Nombre descriptivo	Responsabilidad principal	Puerto
<i>api-gateway</i>	Puerta de entrada	Punto único de acceso; valida token JWT y enruta peticiones a los servicios internos	8080
<i>auth-service</i>	Autenticación y usuarios	Gestión de usuarios, roles y permisos; emisión y rotación de tokens JWT; ejecución de migraciones de base de datos	8081
<i>academic-service</i>	Núcleo académico	Registro de notas y asistencia; gestión de períodos académicos, carreras, asignaturas y paralelos	8082

<i>sync-service</i>	Sincronización e integración	Importación de calificaciones desde Moodle; generación de archivos y consulta al adaptador SIGA	8083
<i>report-service</i>	Generación de reportes	Producción de reportes consolidados en formato PDF, Excel y CSV	8084
<i>notification-service</i>	Notificaciones y alertas	Envío de alertas automáticas por correo y gestión de recordatorios programados	8085

El api-gateway es el único punto de entrada al sistema: toda petición que llega, independientemente del tipo de usuario que la emita, pasa primero por él. Allí se verifica que el usuario esté autenticado y la solicitud se redirige al servicio que corresponde. Ningún cliente externo puede invocar directamente a un microservicio sin pasar por este punto. El auth-service concentra todo lo relacionado con identidades: gestiona el registro de usuarios, el inicio de sesión, la asignación de roles y la emisión de los tokens que permiten acceder al sistema. Es además el único servicio que ejecuta las migraciones de base de datos al arrancar, lo que lo convierte en la pieza que debe estar activa antes que cualquier otra. El academic-service es el núcleo del sistema: administra notas, asistencia, períodos académicos, carreras, asignaturas y paralelos; toda escritura y consulta de información académica pasa por él. El sync-service se encarga de la comunicación con los sistemas externos: importa las calificaciones desde la instancia local de Moodle y genera los archivos para el SIGA, operando como intermediario entre el MRNA y los dos sistemas con los que se integra. El report-service produce los reportes consolidados en formato PDF, Excel y CSV que los coordinadores utilizan en juntas académicas y para la carga al SIGA. Por último, el notification-service gestiona las alertas automáticas por correo cuando un estudiante presenta bajo rendimiento o ausentismo, y los recordatorios que el administrador puede programar con anticipación.

La regla general de comunicación entre estos servicios es que ninguno invoca a otro sin pasar por el api-gateway como punto de control, y que, en el plano de los datos, cada uno trabaja solo con las tablas de su propio dominio. Esta restricción, establecida en ADR-002, es la que permite que cada servicio pueda actualizarse o corregirse sin generar dependencias ocultas con el resto del sistema. Tiene una excepción documentada, y es que, el sync-service, por su función de sincronización masiva de calificaciones, lee y escribe directamente sobre el esquema de notas e inscripciones de academic-service en lugar de pasar por su API (sección 3.6.3). Es la única ruptura de esta regla en el sistema, aceptada como decisión de costo-beneficio para el prototipo y no como un descuido de diseño.

La Figura 4 muestra cómo se relacionan los seis servicios entre sí, con la base de datos compartida, con el frontend y con los sistemas externos, siguiendo el modelo C4 en su segundo nivel de detalle.

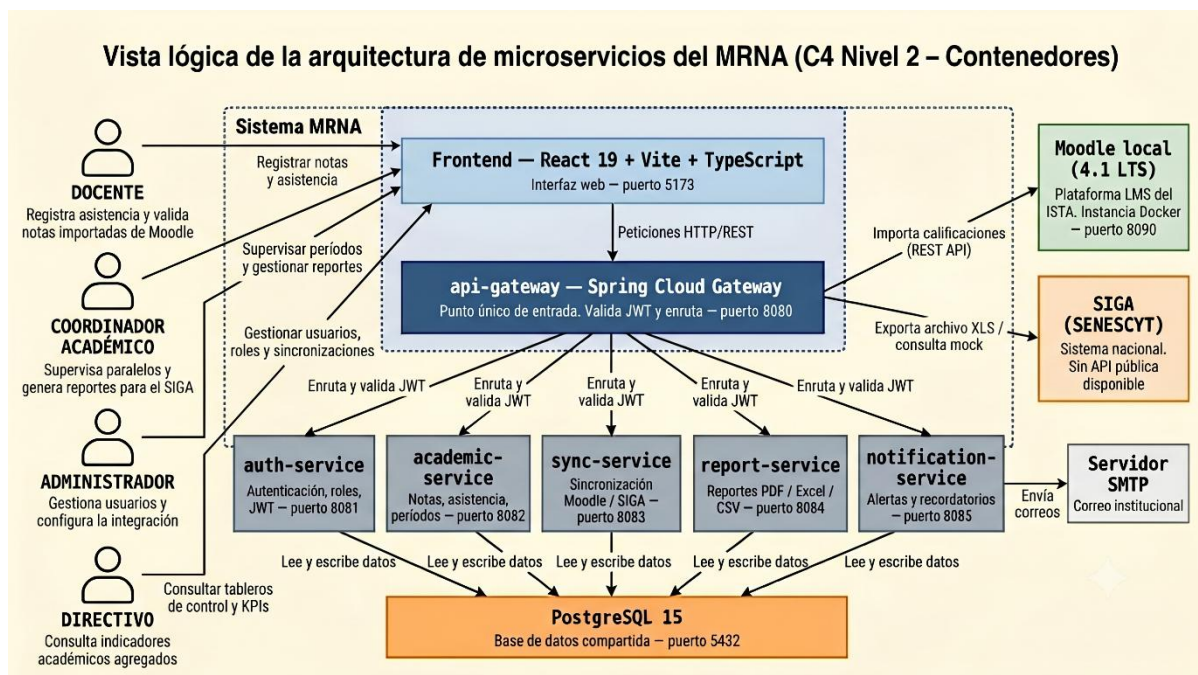


Figura 4. Vista lógica de la arquitectura de microservicios del MRNA (C4 nivel 2 – contenedores). Nota. Elaboración propia con asistencia de inteligencia artificial, 2026.

3.3.2. Vista de despliegue

El entorno de desarrollo del MRNA funciona de forma completamente local en la máquina del desarrollador, sin depender de servidores externos ni de infraestructura en la nube durante la fase de prototipado. Esto se logra mediante Docker Compose, una herramienta que permite definir todos los contenedores del sistema en un único archivo de configuración y levantarlos de forma coordinada con un solo comando. Al ejecutarse, Docker crea automáticamente una red interna privada en la que todos los contenedores se comunican entre sí usando el nombre del servicio como dirección, sin necesidad de conocer direcciones IP fijas ni configurar puertos internos manualmente.

El sistema completo se compone de diez contenedores que se pueden agrupar en tres categorías según su rol. La infraestructura de datos incluye la instancia de PostgreSQL 15, que actúa como base de datos principal del MRNA, y el par formado por Moodle 4.1 con su propia instancia de MariaDB 10.6. Este último par representa la instancia local del aula virtual que simula el entorno institucional, conforme a lo definido en ADR-003 ante la imposibilidad de acceder al Moodle institucional directamente. Los dos motores de

base de datos son independientes entre sí: PostgreSQL almacena toda la información del MRNA, mientras que MariaDB es exclusivamente interna al contenedor de Moodle y no expone ningún puerto al sistema anfitrión. La persistencia de datos en ambos casos se garantiza a través de volúmenes Docker: `postgres_data` para PostgreSQL, y `moodle_data` junto con `moodle_db_data` para Moodle y su base de datos respectivamente. Estos volúmenes aseguran que la información no se pierda cuando los contenedores se detienen o reinician.

El segundo grupo lo forman los cinco microservicios del dominio de negocio (`auth-service`, `academic-service`, `sync-service`, `report-service` y `notification-service`) más el `api-gateway` que los expone como una fachada unificada. El orden de arranque no es arbitrario: `auth-service` debe estar completamente disponible antes que los demás microservicios porque es quien ejecuta las migraciones de base de datos a través de Flyway y quien valida la identidad en cada petición. Los cuatro microservicios restantes declaran una dependencia hacia `auth-service`, y el `api-gateway` espera a que los cinco estén en funcionamiento antes de comenzar a enrutar peticiones. Esta cadena de dependencias, declarada explícitamente en el archivo de configuración, evita errores de arranque por condiciones de carrera entre contenedores.

En términos de autenticación, el recorrido de una petición sigue siempre la misma secuencia. Cuando una persona inicia sesión, el `api-gateway` redirige esa solicitud al `auth-service`, que valida las credenciales contra la base de datos y, si son correctas, emite el token de acceso y el token de actualización descritos en la sección 3.9. A partir de ese momento, cada petición que el frontend hace hacia cualquier microservicio pasa primero por el `api-gateway`, que exige el token de acceso en la cabecera de la solicitud y lo valida antes de reenviar la petición al microservicio correspondiente. Ningún microservicio de dominio necesita implementar su propia lógica de validación de token. Esa responsabilidad queda concentrada en el `api-gateway`, lo que evita repetir el mismo código de verificación seis veces y asegura que una petición sin token válido nunca llegue a tocar la lógica de negocio.

El tercer elemento es el frontend de React, cuyo servidor de desarrollo Vite levanta en el puerto 5173 y se conecta al `api-gateway` para todas las operaciones de datos.

En cuanto a la exposición de puertos hacia la máquina anfitriona, el diseño es selectivo. El `api-gateway` queda accesible en el puerto 8080, que es el único punto de entrada al sistema desde el navegador en escenarios de producción; el frontend en el 5173; y Moodle en el 8090 para configuración y pruebas de la integración. PostgreSQL expone el puerto

5432 para facilitar el trabajo con herramientas de administración de base de datos durante el desarrollo. Los cinco microservicios también exponen sus puertos (8081 a 8085) en el entorno de desarrollo para permitir pruebas unitarias e inspección directa, aunque en un despliegue productivo únicamente el gateway debería ser accesible desde el exterior.

La Figura 5 muestra la distribución completa de contenedores, volúmenes y puertos que conforman el entorno de desarrollo local del MRNA.

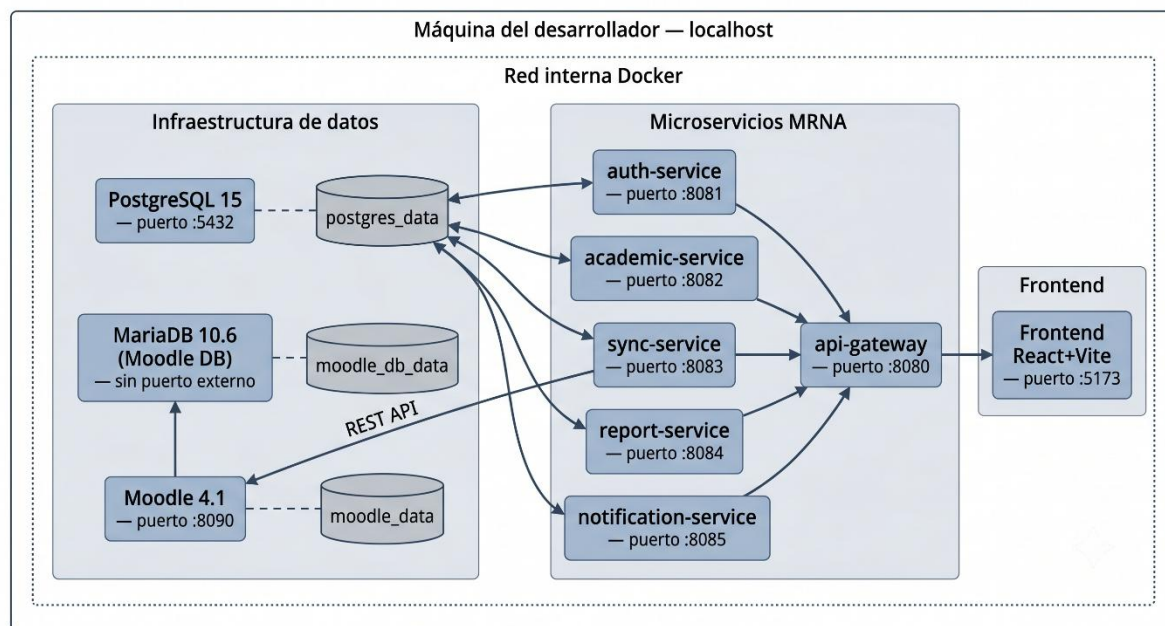


Figura 5. Diagrama de despliegue del entorno de desarrollo MRNA (C4 nivel 3). Nota.

Elaboración propia con asistencia de inteligencia artificial, 2026.

3.3.3. Patrones arquitectónicos aplicados

La elección de patrones de diseño en un sistema de microservicios no es un detalle secundario: define cómo cada componente se relaciona con los demás, cómo fluyen los datos entre capas y qué tan fácil resulta modificar una parte del sistema sin romper el resto [8]. En el contexto del MRNA, los patrones no se eligieron por convención sino porque cada uno resolvía un problema concreto identificado durante el diseño de la arquitectura. A continuación, se describen los cinco patrones que estructuran el prototipo. El primero es el patrón API Gateway, que establece un único punto de entrada al sistema. Todas las peticiones que el frontend realiza al backend pasan obligatoriamente por el api-gateway en el puerto 8080, que se encarga de autenticar el token JWT presente en la cabecera, verificar que la ruta solicitada corresponde a un microservicio registrado y redirigir la petición al destino correcto. Esta centralización tiene dos beneficios prácticos:

el cliente no necesita conocer las direcciones internas ni los puertos de cada microservicio, y los mecanismos transversales como la validación de sesión se definen en un único lugar en lugar de repetirse en cada servicio.

El segundo patrón es el repositorio, aplicado mediante Spring Data JPA en cada microservicio. Las interfaces de repositorio abstraen completamente la lógica de consulta a la base de datos: el código de negocio solicita datos a través de métodos con nombre descriptivo (como `findByCursoIdAndEstudianteId`) sin necesidad de escribir SQL directamente. Esta separación permite cambiar detalles de la capa de persistencia, como la forma en que se construye una consulta o el índice que usa, sin tocar la lógica que la consume.

Complementando al patrón repositorio, el patrón DTO (Data Transfer Object) regula qué datos fluyen hacia el exterior de cada servicio. Las entidades de dominio nunca se exponen directamente a través de la API; en su lugar, se construyen objetos de transferencia específicos para cada operación, que incluyen solo los campos que el cliente necesita ver. Esto desacopla el modelo interno de persistencia del contrato externo de la API, lo que permite que ambos evolucionen de forma independiente.

El cuarto patrón es el adaptador, cuya aplicación más relevante en el MRNA es la integración con SIGA. Dado que SIGA no dispone de una API pública accesible, según lo confirmado durante el levantamiento de información con el responsable de TIC del ISTA, se diseñó un adaptador con dos modos: modo archivo, que genera un fichero XLSX compatible con el formato de carga manual del SIGA y es el que efectivamente opera en el prototipo, y un modo de API reservado para una integración futura, documentado como valor de configuración, pero no conectado hoy a ningún flujo ejecutable. El adaptador encapsula toda la lógica de este límite externo en un solo componente; si en el futuro SIGA llegara a exponer un servicio web, únicamente habría que actualizar ese módulo sin modificar el resto del sistema. La estructura exacta de columnas del archivo se reconstruyó a partir de la práctica institucional y de la plantilla de uso interno del ISTA, no de una especificación oficial publicada por el SIGA, ya que es de carácter privado.

El quinto patrón es el interceptor, utilizado en el mecanismo de auditoría. Un interceptor HTTP captura cada petición entrante antes de que llegue al controlador de negocio, extrae la identidad del usuario autenticado y su dirección IP, y los propaga a la sesión de base de datos mediante variables de sesión de PostgreSQL. Los disparadores definidos a nivel de base de datos leen esas variables al momento de registrar cualquier cambio en notas o asistencia. La separación entre el interceptor en capa de aplicación y los triggers en capa

de datos constituye una división de responsabilidades clara: la aplicación aporta el contexto del usuario, y la base de datos garantiza que ese contexto quede registrado de forma íntegra independientemente del servicio que realizó la operación.

3.4. Stack tecnológico

La selección de tecnologías para el MRNA respondió a tres criterios aplicados de forma consistente a lo largo de todo el proceso de diseño: compatibilidad entre componentes, horizonte de soporte garantizado y curva de aprendizaje razonable para un proyecto desarrollado individualmente bajo restricciones de tiempo. La elección del stack en un sistema de microservicios tiene implicaciones directas sobre la mantenibilidad y la cohesión del código a largo plazo [8], por lo que se priorizaron herramientas con documentación madura y ecosistemas activos antes que las más novedosas del mercado. La Tabla 9 resume el stack completo.

El núcleo del backend descansa en Java 17 LTS junto con Spring Boot 3.2. Java 17 es una versión con soporte extendido garantizado hasta 2029, lo que evita que el prototipo quede técnicamente obsoleto en el corto plazo; además, incorpora características del lenguaje que reducen la verbosidad del código sin romper la compatibilidad con el ecosistema existente. Spring Boot 3.2 se eligió porque su modelo de autoconfiguración permite levantar microservicios funcionales con mínima configuración manual, y su integración nativa con Spring Security 6 cubre la autenticación y la autorización basadas en JWT sin requerir librerías externas adicionales.

La estructura del proyecto se organiza como un build multi-módulo de Maven, donde cada microservicio es un módulo independiente bajo un POM padre que centraliza las versiones de dependencias y permite compilar el sistema completo desde un único punto. Para el acceso a datos, Spring Data JPA con Hibernate abstrae las operaciones sobre la base de datos: los repositorios expresan su intención a través del nombre del método sin necesidad de escribir SQL en la capa de negocio. Las migraciones de esquema se gestionan con Flyway, centralizado en el auth-service según lo descrito en la sección 3.3.1. Complementa la capa backend SpringDoc OpenAPI, que genera de forma automática la documentación interactiva de cada endpoint en Swagger UI durante el desarrollo, sin configuración adicional.

El frontend se construyó con React 19.2.5, TypeScript y Vite. React 19 es la versión estable más reciente de la librería al momento del desarrollo, e introduce mejoras en el

manejo de formularios y transiciones que simplifican escenarios como la captura de notas en tiempo real. TypeScript agrega tipado estático sobre JavaScript, lo que hace explícitos los contratos de datos entre componentes y reduce la clase de errores que solo aparecen en ejecución. Vite actúa como servidor de desarrollo y herramienta de empaquetado con tiempos de recarga casi instantáneos, lo que acelera el ciclo de iteración visual durante la construcción de la interfaz.

Para el sistema visual se eligió TailwindCSS en combinación con shadcn/ui, una colección de componentes accesibles construidos sobre Radix UI que permite levantar interfaces consistentes sin definir estilos desde cero. La gestión del estado se divide en dos responsabilidades con herramientas distintas: TanStack Query administra el estado del servidor, incluyendo el caché de peticiones, la invalidación automática y los reintentos en caso de error de red; Zustand gestiona el estado global del cliente, como la sesión activa o las preferencias de filtro del usuario.

En la capa de datos, PostgreSQL 15 es el motor relacional del sistema. Su elección se apoya en el soporte nativo del tipo jsonb, utilizado en el registro de auditoría para almacenar los valores anteriores y posteriores a cada cambio, y en la robustez de su motor de triggers, sobre el que descansa el mecanismo de trazabilidad documentado en ADR-006. El entorno de desarrollo completo se orquesta mediante Docker Compose, ya descrito en la sección 3.3.2. Para la integración continua, GitHub Actions ejecuta la compilación y el conjunto de pruebas automatizadas ante cada push a la rama principal, garantizando que ningún cambio se integre sin haber pasado por la verificación automática. Las pruebas del backend se implementan con JUnit 5 y Mockito, y las del frontend con Vitest junto a Testing Library, una librería orientada a verificar comportamiento desde la perspectiva del usuario antes que detalles de implementación interna.

Tabla 9. Tecnologías seleccionadas para el desarrollo del prototipo MRNA: versión, capa y función por componente. Nota. Elaboración propia — Archivos de configuración del repositorio tesis-mrna (pom.xml, package.json, docker-compose.yml), 2026.

Tecnología	Versión	Capa	Función principal en el MRNA
Java	17 LTS	Backend	Lenguaje base; soporte extendido hasta 2029
Spring Boot	3.2.5	Backend	Framework de arranque y autoconfiguración de microservicios

Spring Security	gestionado por Spring Boot 3.2.5	Backend	Autenticación y autorización mediante JWT
JJWT	0.12.6	Backend	Generación y validación de tokens JWT
Spring Data JPA / Hibernate	gestionado por Spring Boot 3.2.5	Backend	Abstracción del acceso a datos y ORM
Flyway	gestionado por Spring Boot 3.2.5	Backend	Control de versiones del esquema de base de datos
SpringDoc OpenAPI	2.5.0	Backend	Generación automática de documentación Swagger UI
Apache POI	5.2.5	Backend	Generación y lectura de archivos Excel (.xlsx)
Maven	3.9	Backend	Gestión de dependencias y build multi-módulo
Spring Cloud Gateway	2023.0.1	Backend	Enrutamiento en el api-gateway
React	19.2.5	Frontend	Construcción de la interfaz de usuario
TypeScript	6.0.2	Frontend	Tipado estático sobre JavaScript
Vite	8.0.10	Frontend	Servidor de desarrollo y empaquetado
TailwindCSS	4.2.4	Frontend	Framework de estilos utilitarios
shadcn/ui	4.6.0	Frontend	Componentes UI accesibles sobre Radix UI
TanStack Query	5.100.9	Frontend	Gestión de estado del servidor y caché de peticiones
Zustand	5.0.12	Frontend	Estado global del cliente
react-router-dom	7.14.2	Frontend	Enrutamiento de la SPA
PostgreSQL	15	Base de datos	Motor relacional con soporte nativo de jsonb y triggers
Docker Compose	V2	Infraestructura	Orquestación del entorno local de desarrollo
GitHub Actions	ubuntu-latest (runner CI)	CI/CD	Integración continua y ejecución automática de pruebas
JUnit 5 + Mockito	gestionado por Spring Boot 3.2.5	Testing	Pruebas unitarias e integración del backend
Vitest + Testing Library		Testing	Pruebas unitarias del frontend orientadas a comportamiento

3.5. Diseño de la base de datos

3.5.1. Modelo entidad-relación

El esquema de base de datos del MRNA contiene treinta y seis tablas organizadas en siete grupos funcionales, todos persistidos en una instancia compartida de PostgreSQL 15. Esta decisión de base de datos compartida en el prototipo está documentada en ADR-001: simplifica la infraestructura de desarrollo sin comprometer la separación lógica entre microservicios, que respetan sus propios dominios de datos sin acceder directamente a las tablas del resto.

El primer grupo, identidad y soporte, agrupa las doce tablas que sostienen el núcleo de autenticación y configuración del sistema: pais, persona, usuario, rol, permiso, rol_permiso, usuario_rol, token_recuperacion, refresh_token, configuracion_sistema, auditoria y notificacion. Este grupo es propiedad del auth-service y constituye la base sobre la que los demás microservicios validan identidad y permisos. La tabla refresh_token, incorporada en la migración V2, almacena los tokens de refresco del mecanismo JWT según lo definido en ADR-002, con un único token activo por usuario revocado en cada nuevo inicio de sesión.

El segundo grupo contiene las siete tablas de estructura académica: periodo_academico, carrera, asignatura, paralelo, parcial, tipo_evaluacion y asignacion_docente. Estas tablas modelan la jerarquía institucional del ISTA - carrera - asignatura - paralelo y el calendario evaluativo que rige cuándo se pueden registrar notas y cuándo se cierran los parciales. La tabla tipo_evaluacion almacena los cuatro componentes de calificación establecidos por la normativa institucional: Docencia, Práctica, Autónoma y Evaluación, con ponderaciones de 20 %, 30 %, 30 % y 20 % respectivamente, valores que fueron corregidos en la migración V7 para alinearse al esquema real del ISTA.

El tercer grupo gestiona la trayectoria del estudiante a través de cuatro tablas: tipo_beca, beca_estudiante, matricula e inscripcion_asignatura. La tabla inscripcion_asignatura es la llave de acceso al núcleo del sistema: cada registro de nota y cada registro de asistencia referencia una inscripción concreta, lo que garantiza que ningún dato académico quede huérfano de un estudiante y un paralelo identificados.

El cuarto grupo, núcleo funcional del MRNA, reúne dos tablas, nota y registro_asistencia. La tabla nota registra la calificación obtenida por un estudiante en una inscripción, un parcial y un componente de evaluación, con el campo moodle_activity_id para mantener la trazabilidad con la actividad de Moodle que originó el dato. La tabla registro_asistencia

registra la presencia o ausencia del estudiante en cada sesión, con soporte para estados de tardanza y justificación.

El quinto grupo cubre la integración con sistemas externos mediante cuatro tablas: `configuracion_moodle`, `log_sincronizacion_moodle`, `proceso_importacion_siga` y `error_importacion_siga`. La primera asocia cada paralelo con su curso equivalente en Moodle; la segunda registra cada operación de sincronización con su resultado. Las dos últimas conservan en su nombre la lógica con la que se pensaron originalmente, pero hoy documentan el sentido contrario: `proceso_importacion_siga` registra cada exportación de notas consolidadas hacia el SIGA en modo archivo, y `error_importacion_siga` registra los errores de validación detectados fila por fila antes de entregar ese archivo al coordinador. La actualización del nombre de ambas tablas para que coincida con lo que efectivamente registran queda como una tarea pendiente sobre el esquema, no cubierta en el alcance de este prototipo.

El sexto grupo, servicios derivados, reúne cuatro tablas, `alerta_academica`, `reporte_generado`, `notificacion_programada` y `log_notificacion_programada`. Estas tablas soportan las funcionalidades de alerta por bajo rendimiento (RF-006), generación de reportes en múltiples formatos (RF-008) y notificaciones periódicas programadas (RF-014).

El séptimo grupo, gestión académica complementaria, reúne las tres tablas restantes del esquema, encargadas de procesos auxiliares al registro de calificaciones: `junta_academica`, `evento_calendario` y `nota_recuperacion`. La tabla `junta_academica` modela la junta de calificaciones y, sobre todo, la ventana posterior al cierre del parcial: guarda la fecha de la junta y la fecha límite hasta la cual todavía se admiten correcciones de notas (RF-015), además del acta y las decisiones acordadas. La tabla `evento_calendario` registra los feriados y los hitos del calendario institucional (fecha de inicio y de fin de cada evento, si ocupa el día completo y si afecta al cálculo de asistencia), de modo que el sistema no contabilice como falta una fecha en la que no hubo clase (RF-020 y RF-022). La tercera, `nota_recuperacion`, almacena la calificación del examen de recuperación como un registro aparte, vinculado uno a uno con la inscripción del estudiante, en lugar de tratarla como un quinto componente ponderado; esa separación es la que habilita la regla institucional de recuperación (el examen reemplaza parcialmente al promedio base cuando el estudiante queda dentro de los umbrales establecidos) y mantiene limpia la ponderación de los cuatro componentes ordinarios (RF-003). Con este

grupo, el conteo de tablas del esquema cierra en las treinta y seis anunciadas al inicio del apartado.

La Figura 6 presenta el diagrama entidad-relación del esquema agrupado por módulos funcionales.

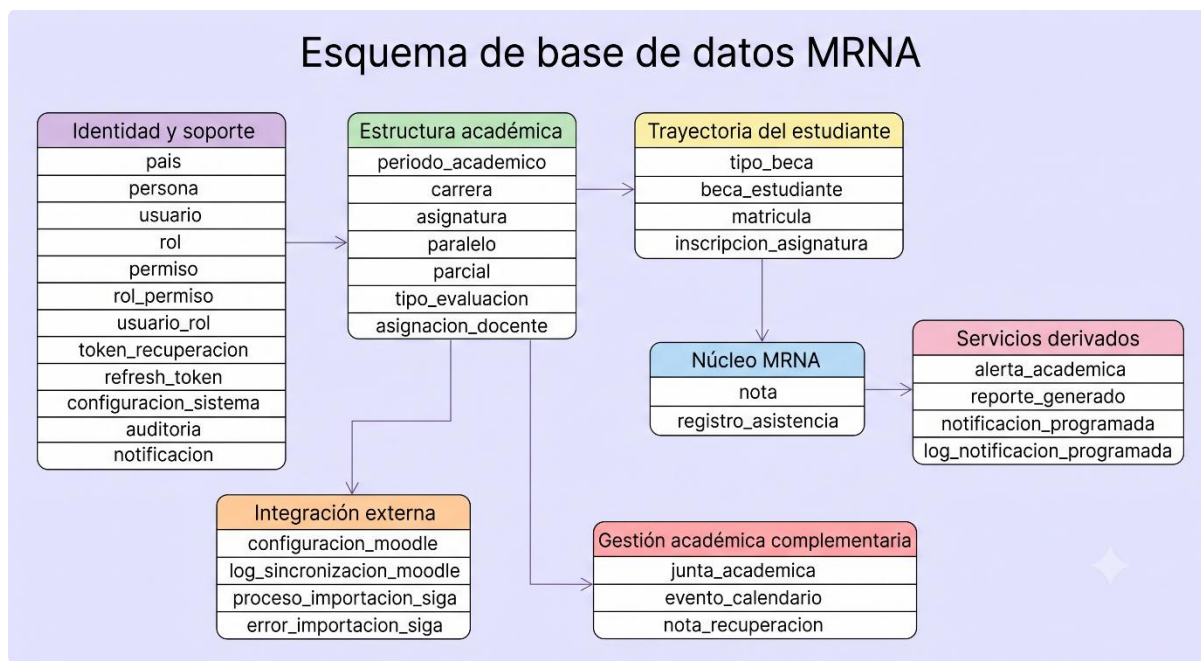


Figura 6. Diagrama entidad-relación del esquema de base de datos MRNA agrupado por módulos funcionales. Nota. Elaboración propia con asistencia de inteligencia artificial, 2026.

3.5.2. Diccionario de datos

El diccionario de datos completo del esquema MRNA, con el nombre de cada columna de las 36 tablas, su tipo de dato, sus restricciones y una descripción de su propósito, se documenta en su totalidad en el Anexo F, dada su extensión. En el cuerpo del capítulo se presenta como ejemplo representativo el diccionario de la entidad nota, por ser la tabla núcleo del sistema y la que concentra el mayor número de reglas de integridad del esquema.

La Tabla 10 detalla la estructura de la tabla nota. Cada calificación queda anclada a una inscripción, un parcial y un componente de evaluación mediante tres llaves foráneas, y un índice único sobre esa misma combinación garantiza que un estudiante no pueda tener más de una nota por componente dentro de un mismo parcial.

Tabla 10. Diccionario de datos de la entidad nota. esquema PostgreSQL del repositorio tesis-mrna. Nota. Elaboración propia — esquema PostgreSQL del repositorio tesis-mrna de la tabla nota, 2026

Columna	Tipo de dato	Restricción	Descripción
id	uuid	PK, DEFAULT uuid_generate_v4()	Identificador único de la calificación.
inscripcion_id	uuid	NOT NULL, FK → inscripcion_asignatura.id	Inscripción del estudiante a la que pertenece la nota.
parcial_id	uuid	NOT NULL, FK → parcial.id	Parcial académico al que corresponde la calificación.
tipo_evaluacion_id	uuid	NOT NULL, FK → tipo_evaluacion.id	Componente de evaluación calificado (Docencia, Práctica, Autónomo o Evaluación).
valor	decimal(5,2)	NOT NULL	Calificación obtenida por el estudiante.
sobre	decimal(5,2)	DEFAULT 10	Escala sobre la que se registra el valor.
moodle_activity_id	int	—	Identificador de la actividad en Moodle que originó el dato; permite sincronización idempotente (RF-002).
observaciones	text	—	Comentario opcional del docente sobre la calificación.
estado	EstadoNota (enum)	DEFAULT 'BORRADOR'	Estado del registro dentro del flujo de validación de la nota.
fuelle	FuenteRegistr o (enum)	DEFAULT 'MANUAL'	Origen del dato: registro manual o sincronización automática desde Moodle.
es_recuperacion	boolean	DEFAULT false	Indica si la nota corresponde a un proceso de recuperación.
usuario_registro	uuid	FK → usuario.id	Usuario que registró o modificó la calificación.
fecha_registro	timestamp	DEFAULT now()	Momento en que se registró la calificación.
created_at / updated_at	timestamp	DEFAULT now()	Marcas de auditoría temporal estándar del esquema.

Adicionalmente, la combinación (inscripcion_id, parcial_id, tipo_evaluacion_id) está protegida por un índice único (idx_nota_unico), lo que impide que el sistema acepte dos calificaciones del mismo componente para el mismo estudiante en el mismo parcial.

3.5.3. Estrategia de migraciones con Flyway

El esquema de la base de datos no se crea de una sola vez ni a mano: se construye con migraciones, que son archivos pequeños con instrucciones en SQL, numerados y aplicados en orden, que la herramienta Flyway ejecuta de forma automática cuando el sistema arranca. La ventaja es práctica: cualquier persona puede levantar el proyecto desde cero y obtener exactamente la misma base de datos, paso por paso, sin depender de ajustes manuales.

Para que dos servicios no intenten modificar la base al mismo tiempo, solo el auth-service tiene permiso de aplicar migraciones; los demás arrancan con esa función desactivada. De esa manera existe un único punto de control sobre la estructura de datos, tal como establece el ADR-001. Se eligió Flyway por su sencillez, las migraciones se escriben en SQL plano (sin capas extra de configuración) y los archivos se nombran de forma directa, de V1 hasta V13.

La Tabla 11 resume las trece migraciones aplicadas hasta cerrar el prototipo. La secuencia refleja algo normal en un proyecto real: el esquema no quedó perfecto desde el primer intento, sino que se fue corrigiendo conforme avanzaba el desarrollo. Por ejemplo, una migración corrigió el peso de los componentes de evaluación (que al inicio quedaron en partes iguales y debían ser 20/30/30/20), otra incorporó el estado "CERRADO" de un período que no estaba contemplado al principio, las migraciones V10 a V12 resolvieron ajustes que aparecieron al conectar el sistema con Moodle, y la última (V13) incorporó el umbral configurable que el sistema usa para detectar el bajo rendimiento académico. Cada corrección se registró como una migración nueva, dejando una historia clara de cómo evolucionó la base.

Tabla 11. Secuencia de migraciones Flyway del esquema MRNA. Nota. Elaboración propia — repositorio tesis-mrna, 2026.

Migración	Contenido	Trazabilidad
V1__schema_mrna_init.sql	Esquema completo inicial: 34 tablas, extensión uuid-osp, seeds de roles y tipos de evaluación	ADR-005
V2__auth_service_setup.sql	Tabla refresh_token y seed de usuario administrador de desarrollo	ADR-002, Cadencia 2

V3__varchar_to_enum_implicit_casts.sql	Casts implícitos varchar→ENUM para los tipos de auth-service	Cadencia 2
V4__academic_seed_data.sql	Datos de prueba del dominio académico (período, carrera, asignatura, paralelo, parcial)	DB-03, DB-04
V5__academic_enum_implicit_casts.sql	Casts implícitos varchar→ENUM para los tipos de academic-service	RF-002, RF-003, RF-010, RF-012, RF-018
V6__auditoria_triggers.sql	Función y triggers de auditoría sobre nota y registro_asistencia; bloqueo de UPDATE/DELETE sobre la bitácora	ADR-006, RF-018
V7__ponderaciones_institucionales.sql	Corrección de ponderaciones de tipo_evaluacion a 20/30/30/20	RF-003
V8__periodo_estado_varchar.sql	periodo_academico.estado de ENUM a varchar con CHECK (ABIERTO/CERRADO)	RF-012
V9__nota_recuperacion.sql	Tabla nota_recuperacion para la regla institucional de recuperación	RF-003
V10__log_datos_response_text.sql	Cambio de tipo de la columna datos_response de jsonb a texto en log_sincronizacion_moodle	RF-002, HU-002-03
V11__tipo_evaluacion_inscripcion_estado_varchar.sql	Conversión de estado de ENUM a varchar con restricción CHECK en tipo_evaluacion e inscripcion_asignatura	RF-002, HU-002-03
V12__parcial_estado_varchar.sql	Conversión de estado de ENUM a varchar con restricción CHECK en parcial	RF-002, HU-002-03
V13__configuracion_umbral_nota.sql	Seed del umbral de bajo rendimiento (nota.umbral.bajo_rendimiento = 7) en configuracion_sistema, leído por el notification-service para generar alertas	ADR-005, RF-006

Una decisión importante fue no deshacer nunca una migración ya aplicada. Cuando algo debía corregirse, no se editaba ni se revertía el cambio anterior, sino que se creaba una migración nueva que lo arreglaba (como pasó con los ajustes de pesos y de estados ya mencionados). Este criterio, que es además la recomendación oficial de Flyway [28], conserva intacto el historial de cómo se construyó la base de datos y evita sorpresas al

momento de desplegar. En la práctica, cuando alguien levanta el entorno desde cero con un solo comando, Flyway aplica las trece migraciones en orden, de la primera a la última, y deja la base lista para usar. Esta forma de trabajar coincide con lo que la literatura sobre evolución de esquemas identifica como buena práctica, mantener siempre hacia adelante el historial de cambios en lugar de reescribirlo [14].

3.6. Diseño de los microservicios

Cada uno de los seis servicios del MRNA se diseñó con una responsabilidad única, su propio recorte del modelo de datos y un conjunto de endpoints REST que expone a través del api-gateway, conforme a ADR-002. Los límites establecidos obedecen a una justificación técnica específica y no a una decisión aleatoria, ya sea el tipo de dato que administra, el patrón de comunicación que requiere (síncrono frente a asíncrono) o la superficie de integración externa que expone, como se precisa al cierre de cada subsección. Las siguientes subsecciones presentan a cada servicio bajo la misma estructura para facilitar la comparación entre ellos, los endpoints completos de cada uno se listan en el Anexo J.

3.6.1. *auth-service*

Responsabilidad: autenticación, gestión de usuarios, control de acceso basado en roles (RBAC), emisión y renovación de tokens JWT, y recuperación de contraseña.

Endpoints clave: POST /api/auth/login, POST /api/auth/refresh, POST /api/auth/logout, POST /api/auth/recuperar-contrasena, POST /api/auth/reset-password, GET/POST /api/auth/usuarios, y la gestión de roles por usuario bajo /api/admin/usuarios/{usuarioId}/roles.

Modelo de datos: persona, usuario, rol, permiso, rol_permiso, usuario_rol, token_recuperacion, refresh_token.

Dependencias: ninguna hacia otros microservicios, es la base de la pirámide y el único que ejecuta las migraciones Flyway al arrancar. Se aisló como servicio independiente porque la autenticación y el control de acceso son transversales a los otros cinco, y centralizarlos evita duplicar la validación de credenciales y de roles en cada servicio de dominio, y permite reforzar la política de seguridad sin tocar el resto del sistema.

3.6.2. *academic-service*

Responsabilidad: núcleo académico del sistema, que gestiona periodos académicos, carreras, asignaturas, paralelos, asignaciones docentes, matrículas, inscripciones, notas, asistencia y recuperación de notas.

Endpoints clave: GET /api/academic/periodos, PATCH /api/academic/periodos/{id}/cerrar, GET/POST /api/academic/carreras, GET/POST /api/academic/asignaturas, GET/POST/PUT /api/academic/notas, GET/PUT /api/academic/asistencias, PUT /api/academic/recuperaciones/inscripciones/{id}.

Modelo de datos: periodo_academico, carrera, asignatura, paralelo, parcial, tipo_evaluacion, asignacion_docente, matricula, inscripcion_asignatura, nota, registro_asistencia, nota_recuperacion, auditoria (lectura de historial).

Dependencias: auth-service, para la validación de identidad y rol en cada petición. Concentra el núcleo académico porque notas, asistencia, matrículas e inscripciones comparten reglas de negocio e integridad referencial estrechamente relacionadas (una nota, por ejemplo, no puede existir sin una inscripción activa), separarlas en varios servicios habría obligado a coordinar transacciones distribuidas para operaciones que hoy resuelve una sola transacción de base de datos.

3.6.3. *sync-service*

Responsabilidad: integración con sistemas externos que comprende la sincronización de calificaciones desde la instancia local de Moodle y adaptador hacia SIGA en modo archivo, con una vía de API reservada para el futuro, conforme a ADR-003.

Endpoints clave: POST /api/sync/moodle/sincronizar, GET /api/sync/moodle/logs, GET /api/sync/siga/exportar (generación del archivo de notas consolidadas hacia el SIGA, flujo oficial del adaptador en modo archivo), y POST /api/sync/siga/validar junto con POST /api/sync/siga/importar, conservados como mecanismo legado de importación en sentido inverso.

Modelo de datos: configuracion_moodle, log_sincronizacion_moodle, proceso_importacion_siga, error_importacion_siga.

Dependencias: auth-service, para la validación de identidad y rol. A diferencia de los demás servicios de dominio, el sync-service no pasa por academic-service para escribir las notas que importa desde Moodle, lee y escribe directamente sobre las mismas tablas de notas e inscripciones que administra academic-service, porque ambos comparten la instancia de PostgreSQL bajo ADR-001. Es la única excepción real al principio de

aislamiento de datos entre servicios descrito en la sección 3.3.1, y conviene explicar por qué se aceptó: encadenar la sincronización masiva de calificaciones a través de una llamada al api-gateway y de ahí a academic-service habría añadido una capa de red y de reintentos para una operación por lotes que, al ejecutarse dentro de la misma base de datos, se resuelve en una sola transacción. El costo de esa decisión es un acoplamiento real a nivel de esquema, un cambio en la estructura de la tabla de notas obliga a revisar también el sync-service, algo que un patrón Database per Service estricto habría evitado. Se documenta aquí como una decisión de diseño consciente, no como un descuido, y queda registrada como línea de mejora futura en la sección 5.4.

3.6.4. report-service

Responsabilidad: generación de reportes consolidados para juntas académicas y carga al SIGA, conforme a RF-008.

Endpoints clave: GET /api/reports/consolidado, GET /api/reports/status.

Modelo de datos: no posee tablas propias; consulta el esquema académico de solo lectura para construir los reportes.

Dependencias: auth-service. Los formatos CSV y Excel están operativos, la exportación a PDF forma parte del diseño del servicio y se contempla como ampliación. Se separó del academic-service porque la generación de reportes consolidados tiene un patrón de carga distinto (lecturas masivas y ocasionales, frente a escrituras frecuentes y puntuales de notas), lo que permite escalarlo de forma independiente sin afectar el rendimiento del registro de calificaciones.

3.6.5. notification-service

Responsabilidad: envío de notificaciones por correo electrónico, con alertas por bajo rendimiento o ausentismo (RF-006) y recordatorios programados (RF-014).

Endpoints clave: POST /api/notifications/test-email.

Modelo de datos: notificacion.

Dependencias: auth-service. El servicio expone el canal de envío por correo, la detección automática de bajo rendimiento (RF-006) y la programación de recordatorios (RF-014) forman parte de su diseño funcional. Se aisló porque el envío de notificaciones es un proceso asíncrono y tolerante a fallos (reintentos, colas, dependencia de un proveedor de correo externo) que no debe bloquear ni acoplar su disponibilidad a la del flujo síncrono de registro académico.

3.6.6. *api-gateway*

Responsabilidad: punto de entrada único al sistema, con el enrutamiento hacia los cinco microservicios de dominio, validación del token JWT antes de reenviar cualquier petición, y configuración centralizada de CORS para el frontend.

Endpoints clave: no expone lógica de negocio propia; redirige cada ruta según prefijo (/api/auth/**, /api/academic/**, /api/sync/**, /api/reports/**, /api/notifications/**) hacia el microservicio correspondiente.

Modelo de datos: ninguno; es un componente sin estado.

Dependencias: los cinco microservicios de dominio, espera a que todos estén disponibles antes de comenzar a enrutar peticiones. No expone lógica de dominio porque su función es exclusivamente de enrutamiento y validación de token, y mantenerlo así lo vuelve reemplazable (por ejemplo, por un gateway administrado en la nube) sin que ese cambio afecte a ningún otro servicio.

3.7. Diseño de la API REST

El diseño de la API sigue un conjunto de convenciones aplicadas de manera uniforme en los cinco microservicios de dominio. Los recursos se nombran en plural (/notas, /asistencias, /usuarios), los verbos HTTP expresan la operación de forma semántica (GET para consulta, POST para creación, PUT/PATCH para actualización total o parcial) y cada respuesta usa el código de estado que corresponde a su resultado real, sin reservar un único código genérico para distintos escenarios. El listado de notas, al ser la consulta de mayor volumen del sistema (RF-007, filtros multidimensión), se enlaza con el mecanismo estándar de Spring Data (parámetros page, size y sort), esta decisión se ajusta al tamaño real de los datos de un instituto (cientos de estudiantes por período, no millones de registros), donde la paginación por desplazamiento no representa un problema de rendimiento. La ruta de cada recurso no incluye un segmento de versión (/api/academic/notas, no /api/v1/notas); el versionado de la API queda fuera del alcance del prototipo y se documenta como trabajo futuro, ya que ningún cliente externo distinto al frontend propio consume hoy estos endpoints.

Como ejemplo representativo del diseño completo, el siguiente fragmento documenta POST /api/academic/notas, el endpoint que registra o actualiza una calificación (operación upsert sobre inscripción-parcial-tipo de evaluación). El cuerpo de la petición

exige los tres identificadores de la clave de negocio y el valor de la nota, validado en rango de 0 a 10 con un máximo de dos decimales:

POST /api/academic/notas

Authorization: Bearer <jwt>

```
{
  "inscripcionId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "parcialId": "9b2e1c40-...",
  "tipoEvaluacionId": "a1d4e8f2-...",
  "valor": 8.55,
  "observaciones": "Recuperación de prueba escrita"
}
```

La respuesta exitosa devuelve 201 Created si la nota se crea por primera vez, o 200 OK si ya existía y se actualizó, con la representación completa del registro:

```
{
  "id": "c7a1...",
  "inscripcionId": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "parcialId": "9b2e1c40-...",
  "parcialNombre": "Primer parcial",
  "tipoEvaluacionId": "a1d4e8f2-...",
  "tipoEvaluacionNombre": "Práctica",
  "valor": 8.55,
  "sobre": 10.00,
  "estado": "REGISTRADA",
  "fuente": "MANUAL",
  "esRecuperacion": false,
  "fechaRegistro": "2026-06-19T10:32:00"
}
```

Si el período académico del parcial ya está cerrado (RF-012), el servicio no permite la escritura: responde 422 Unprocessable Entity y, en paralelo, registra el intento bloqueado en la tabla de auditoría. La Tabla 12 resume los códigos de estado HTTP que efectivamente devuelve la API del MRNA, con el escenario que los origina.

Tabla 12. Códigos de estado HTTP devueltos por la API REST del MRNA. Nota. Elaboración propia — repositorio tesis-mrna, 2026.

Código	Significado	Escenario típico
200	OK	Consulta exitosa o actualización de un recurso existente
201	Created	Creación de un recurso nuevo
400	Bad Request	Violación de una regla de validación de entrada (@Valid)
401	Unauthorized	Token JWT ausente, inválido o expirado
403	Forbidden	Usuario autenticado sin el rol requerido para la operación
404	Not Found	El recurso referenciado (nota, parcial, inscripción) no existe
422	Unprocessable Entity	Regla de negocio que bloquea la operación (ej. período cerrado)
500	Internal Server Error	Error no contemplado, capturado por el manejador global

Cada microservicio centraliza este mapeo en un `GlobalExceptionHandler` propio (`@RestControllerAdvice`), de modo que ningún controlador necesita construir manualmente la respuesta de error.

3.8. Diseño de la interfaz de usuario

3.8.1. Sistema de diseño

El frontend del MRNA construye su interfaz sobre TailwindCSS [29] y shadcn/ui [30], la misma que se fijó para el stack del proyecto. Por un lado, TailwindCSS aplica un enfoque *utility-first*: en lugar de escribir hojas de estilo separadas por componente, las clases de utilidad (espaciado, color, tipografía) se aplican directamente sobre el marcado y siguen una escala predefinida, lo que reduce el tiempo de desarrollo de interfaces y evita que cada pantalla termine con su propia variación de espaciado o tamaño de fuente. Por otro lado, shadcn/ui no es una librería de componentes que se instala como dependencia cerrada, sino un conjunto de componentes accesibles cuyo código se copia al proyecto y se ajusta libremente; esto permite mantener consistencia visual entre las pantallas de los cuatro roles del sistema (docente, coordinador, administrador, directivo) sin sacrificar la capacidad de adaptar cada componente a las reglas de negocio del MRNA.

El proyecto no cuenta con un manual de identidad visual institucional del ISTA aplicable a interfaces web, por lo que se adoptó una paleta neutral profesional: escala de grises en el espacio de color, con variantes para fondo, texto, bordes y estados (*primary*, *secondary*, *muted*, *destructive*, entre otros).

3.8.2. Pantallas del prototipo por rol

El prototipo MRNA no concentra el trabajo en un solo tablero, sino que lo reparte en pantallas independientes, cada una pensada para una tarea puntual y para el rol que la lleva a cabo. A continuación, se presentan las cuatro pantallas centrales del sistema, capturadas directamente del prototipo en funcionamiento, que cubren el recorrido principal de notas y reportes entre Moodle, el MRNA y el SIGA.

El docente trabaja sobre la vista de notas por paralelo (como se muestra en la Figura 7). Ahí elige un paralelo y un parcial, y el sistema despliega las calificaciones de cada estudiante separadas por los cuatro componentes de evaluación (docencia, práctica, autónoma y evaluación), junto con el promedio ponderado al final de cada fila. Es la pantalla que reemplaza la revisión que hoy el docente hace por separado entre Moodle y la plantilla de Excel.

El coordinador y el administrador acceden a la pantalla de descarga de reportes (Figura 8), donde seleccionan período, carrera y paralelo. Desde ahí pueden descargar el reporte consolidado de notas en Excel o CSV, pensado para uso interno del Instituto, y, en un bloque aparte dentro de la misma pantalla, generar el archivo de exportación hacia el SIGA: un XLSX con la nota final ya consolidada de cada estudiante, en el formato exacto que el sistema nacional exige para su carga manual. Esta segunda opción está disponible solo para coordinador y administrador; el directivo conserva acceso al reporte general, pero no a la exportación hacia el SIGA.

La incorporación de cuentas docentes ocurre en la pantalla de carga masiva (Figura 9), de uso exclusivo del administrador. A partir de un archivo de Excel con la plantilla del MRNA se registran los docentes de una sola vez, en lugar de cargarlos uno por uno, lo que agiliza la preparación de cada período.

El sistema conserva, además, la pantalla de importación (Figura 10) como mecanismo secundario, y este permite cargar un archivo con la estructura del SIGA para importar notas hacia el MRNA, validando su contenido antes de aceptarlo. Se mantiene visible solo para coordinador y administrador.

Las vistas de indicadores agregados para el nivel directivo, el panel de asistencia y la administración avanzada de usuarios quedaron planteadas en el diseño, pero no se construyeron dentro del alcance del prototipo; se documentan como trabajo futuro en la sección 5.4.

Notas por paralelo
Selecciona un periodo y un paralelo para ver las notas del parcial activo

Período académico: Marzo-Agosto 2026 | Paralelo: Desarrollo de Aplicaciones Web -- A | Parcial activo: Segundo Parcial (A)

29 estudiantes - Segundo Parcial

ESTUDIANTE	MATRICULA	DOCENCIA	PRÁCTICA	AUTÓNOMA	EVALUACIÓN	PROMEDIO
Brto Paola	M-DEMO-007	8.86	9.46	8.67	8.40	8.89
Castro Gabriela	M-DEMO-013	7.75	6.61	6.69	6.99	6.94
Chuva Diego	M-DEMO-006	8.87	7.97	8.02	7.72	8.12
Cárdenas Tatiana	M-DEMO-021	9.04	9.59	9.20	10.00	9.45
Flores Christian	M-DEMO-014	7.92	7.32	7.47	7.14	7.45
Guamán Bryan	M-DEMO-012	8.75	8.83	9.56	9.33	9.13
Gómez José	M-DEMO-002	4.99	6.02	4.97	6.16	5.53
Lema Kevin	M-DEMO-010	8.22	7.89	8.67	7.71	8.09

Figura 7. Pantalla de notas por paralelo del prototipo MRNA. Nota. Captura del prototipo, elaboración propia, 2026.

Descargar reporte
Reporte consolidado de notas y asistencia por paralelo

Período académico *: Marzo-Agosto 2026

Carrera *: Tecnología Superior en Desarrollo de Software

Ciclo: Nivel 2

Paralelo *: B -- Fundamentos de Programación

Formato:
☒ Excel (.xlsx) ☐ CSV (.csv)

[Descargar reporte](#)

Exportar a SIGA
 Genera el archivo XLS con la nota final consolidada de cada estudiante del periodo, carrera y paralelo seleccionados arriba, listo para cargar manualmente en SIGA.

[Exportar a SIGA](#)

Figura 8. Pantalla de descarga de reportes (roles coordinador, administrador y directivo), con la opción de exportación hacia el SIGA (exclusiva de coordinador y administrador). Nota. Captura del prototipo, elaboración propia, 2026.

Carga masiva de usuarios

Importa docentes desde un archivo Excel (.xlsx) con la plantilla MRNA.

Formato esperado del archivo (columnas en orden):
 A: Cedula | B: Nombre | C: Apellido | D: Correo institucional | E: Tipo (DOCENTE)

Ningún archivo seleccionado

Figura 9. Pantalla de carga masiva de docentes (rol administrador). Nota. Captura del prototipo, elaboración propia, 2026.

MRNA
 ISTA - Sistema Academico

Dashboard
 CATALOGOS
 Períodos
 Carreras
 Asignaturas
 Paralelos
 ACADEMICO
 Notas por paralelo
 Notas (filtros)
 Asistencia
 Carga masiva docentes
 Usuarios
 Descargar reporte
 LEGADO
 Importar SIGA (legado)

admin@ista.edu.ec
 ADMIN

Validación e importación SIGA

Cargue el archivo XLS exportado desde SIGA. El proceso tiene dos pasos: primero se valida la estructura y luego se importan las notas de los estudiantes al sistema.

1. Validar archivo
 Revisa columnas y datos (cédulas, notas, tipos de registro) y muestra los errores.
 No guarda nada en el sistema.

2. Importar definitivamente
 Guarda en el sistema las notas de los estudiantes (filas tipo NOTA) con fuente SIGA. Solo se habilita si la validación no tuvo errores.

Arrastra el archivo aquí o haz clic para seleccionar
 Formatos aceptados: xls, xlsx

Figura 10. Pantalla de importación SIGA, mecanismo legado conservado por continuidad con la validación de usabilidad (roles coordinador y administrador). Nota. Captura del prototipo, elaboración propia, 2026.

3.8.3. Flujos de usuario clave

Esta sección describe tres flujos que ilustran cómo se mueve la información entre Moodle, el MRNA y SIGA. Se seleccionaron estos tres porque corresponden directamente a las tres etapas del proceso manual descritas en la sección 2.8 (registro en Moodle, consolidación y carga al SIGA), y son los que un docente o un coordinador ejecutan con mayor frecuencia en la operación diaria del sistema. El primero es el ingreso o corrección

de una nota dentro de un parcial. La mayoría de las notas llegan al sistema por sincronización automática desde Moodle (RF-002), así que el punto de partida real del docente no es escribir un número en una casilla, sino abrir la vista de notas por paralelo (RF-003) y revisar lo que ya se sincronizó. Si encuentra un valor que falta o que está mal, corrige el registro a través del mismo endpoint que usa la sincronización (POST /api/academic/notas): el backend valida el rango de 0 a 10, verifica que el período siga abierto y, si está cerrado, bloquea la operación (RF-012) y registra el intento en la auditoría en vez de dejarlo pasar silenciosamente.

El segundo flujo es la sincronización de calificaciones desde Moodle. El sync-service expone un endpoint (POST /api/sync/moodle/sincronizar) que toma el token de servicio web de la instancia local de Moodle, recorre los cursos mapeados a paralelos y trae las calificaciones registradas por cada componente de evaluación. Cada corrida queda en un registro consultable (GET /api/sync/moodle/logs), lo que permite rastrear cuándo se sincronizó cada paralelo y si hubo errores de mapeo

El tercer flujo cierra el ciclo hacia el SIGA. Desde la pantalla de reportes, el coordinador o el administrador elige período, carrera y paralelo y, en el bloque de exportación, solicita el archivo para el SIGA. El sync-service genera un XLSX con la nota final de cada estudiante, en el formato que el sistema exige para su carga, y el usuario lo descarga. Como el SIGA no expone una API pública para esta institución, ese archivo se sube de forma manual a la interfaz del sistema nacional, siguiendo la estrategia de adaptador dual. La pantalla de importación se conserva únicamente como mecanismo secundario para cargar notas en sentido inverso y no forma parte de este flujo oficial.

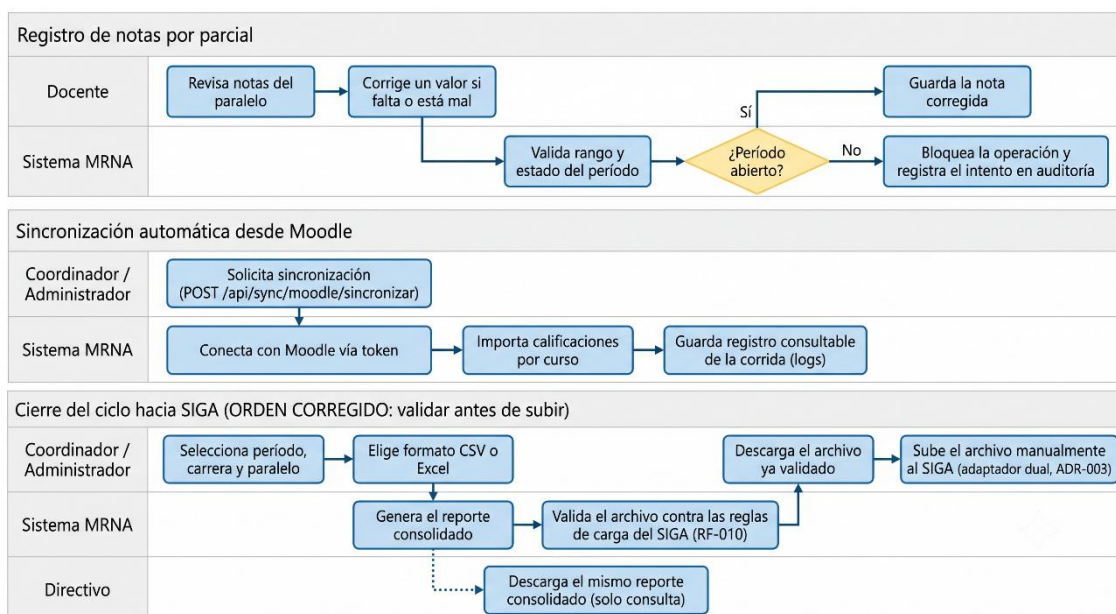


Figura 11. Flujos clave de interacción entre los roles docente, coordinador/administrador y el sistema MRNA. Elaboración propia con asistencia de inteligencia artificial.

3.9. Estrategia de seguridad

La estrategia de seguridad del MRNA se organiza en cuatro componentes complementarios, el control de acceso al sistema, los permisos según el rol del usuario ya autenticado, la protección de la información sensible y el registro de auditoría sobre lo ocurrido.

Para iniciar sesión, cada usuario introduce su correo y contraseña. El sistema nunca almacena la clave real, sino que la transforma mediante una función matemática no invertible llamada hash. A diferencia de un cifrado, que se puede deshacer con una llave, el hash es de una sola vía. Específicamente, el MRNA utiliza el algoritmo BCrypt configurado en un nivel de seguridad alto que procesa la contraseña 4,096 veces continuas para volverla ultra resistente a ataques informáticos. Además, el algoritmo añade automáticamente un código aleatorio único a cada contraseña para que los resultados nunca sean iguales. El dato final se guarda en la columna `password_hash`, impidiendo que nadie pueda reconstruir la contraseña original, incluso si tiene acceso total a la base de datos. Cuando el inicio de sesión es correcto, el sistema entrega dos tipos de "llave" digital. La primera es un token de acceso, que dura poco tiempo y es el que se envía en cada acción dentro del sistema para demostrar quién la está haciendo. La segunda es un token de actualización, más duradero, cuyo único propósito es pedir una llave de acceso nueva cuando la anterior caduca, sin obligar a la persona a escribir su contraseña de nuevo cada pocos minutos. Cada vez que alguien inicia sesión, el token de actualización anterior queda invalidado, así que solo puede existir una sesión activa por persona a la vez.

Una vez dentro, lo que cada persona puede ver o hacer depende de su rol, y el sistema reconoce cuatro: administrador, coordinador, docente y directivo. El administrador es el único perfil con control total: además de consultar, es quien crea y edita los catálogos del sistema (períodos, carreras, asignaturas y paralelos), registra a los docentes de forma masiva, gestiona las cuentas de usuario y tiene acceso tanto a la exportación hacia el SIGA como al mecanismo de importación. El coordinador trabaja sobre la información académica del día a día: consulta catálogos, notas y asistencia, genera el archivo de exportación hacia el SIGA y descarga los reportes consolidados, pero no altera la estructura de catálogos ni administra usuarios. El docente se mueve en un ámbito más acotado, centrado en la consulta de catálogos, notas y asistencia, sin acceso a la

exportación hacia el SIGA, a los reportes ni a la gestión de usuarios. El directivo tiene un perfil orientado a la visualización: consulta esa misma información académica y puede descargar los reportes generales, sin acceso a la exportación hacia el SIGA ni al mecanismo legado, pensado más para el seguimiento que para la operación. La edición de los catálogos quedó reservada solo al administrador, de modo que la responsabilidad sobre la estructura académica se concentra en un único perfil y no se dispersa. Y esta separación no es apenas una cuestión de qué botones aparecen en pantalla: el propio backend comprueba el rol antes de ejecutar cualquier acción, así que, aunque alguien intentara llamar directamente a una función que no le corresponde, el sistema la rechazaría igual.

La información que viaja entre el navegador y el servidor, y la que se guarda en la base de datos, tiene su propia protección. Los datos que entran a través de un formulario se validan antes de procesarse, para evitar que un campo mal llenado o intencionalmente malicioso llegue a ejecutarse como código. El token que usa el sistema para conectarse con Moodle tampoco se guarda en la base de datos: vive únicamente como variable de entorno del servidor, así que, aunque alguien accediera a las tablas, no encontraría esa credencial ahí.

Por último, el sistema bloquea operaciones indebidas y además las recuerda. Cada cambio sobre una nota o un registro de asistencia queda escrito automáticamente en una bitácora de auditoría, sin que el código de la aplicación tenga que pedirlo de forma explícita: es la propia base de datos la que lo hace en el momento exacto en que ocurre el cambio, lo que evita que alguien edite un registro y la auditoría correspondiente quede desactualizada o se olvide de registrarse. Si alguien intenta modificar una nota en un período que ya cerró, esa acción bloqueada también se guarda, porque saber qué se intentó hacer es tan importante como saber qué se hizo.

3.10. Estrategia de integración

3.10.1. Integración con Moodle local

El MRNA no se conecta con la instancia de Moodle del ISTA, sino con una instancia local levantada en Docker para el desarrollo del prototipo. La razón no es una preferencia de diseño: la institución no entrega credenciales de administrador sobre su Moodle real, y sin ese acceso no hay forma de crear el token de servicio web que la integración necesita (Responsable de TIC, comunicación personal, 2025). Esta restricción institucional es una

limitación real del estudio, y no apenas una decisión técnica. Moodle expone la misma API de servicios web sin importar dónde esté instalado [6], lo que permite validar el mecanismo de integración, pero la instancia local no reproduce el volumen de datos, la configuración ni las particularidades que tendría el Moodle institucional en operación real. Los resultados de sincronización que se reportan en el Capítulo 4 deben leerse con esa salvedad.

La sincronización parte de una tabla de configuración que asocia cada paralelo del MRNA con un curso de Moodle y guarda la URL base y el token de acceso correspondientes. Cuando se ejecuta una sincronización, el sync-service llama primero a la función `core_enrol_get_enrolled_users` para traer la lista de estudiantes matriculados en el curso, y después a `core_grades_get_grades` para obtener las calificaciones registradas en cada componente de evaluación. Ese token no viaja como un encabezado de autorización: Moodle lo espera como parámetro de consulta (`wstoken`) en la misma URL de la petición, una particularidad de su API que conviene tener clara porque rompe el patrón Bearer que usa el resto de los microservicios del MRNA.

Esa particularidad tiene una implicación de seguridad que conviene explicitar. En el entorno de desarrollo, la conexión hacia la instancia local de Moodle se hace por HTTP simple sobre localhost, no por HTTPS, porque ambos servicios corren en la misma máquina y no exponen ese tráfico fuera de ella. Un despliegue en producción sí necesitaría forzar HTTPS en ese canal. Un token que viaja por HTTP sin cifrar en la URL queda expuesto a cualquiera que intercepte el tráfico de red. Sobre el riesgo de que el token quede escrito en un registro de log, el sistema no aplica hoy una redacción deliberada del parámetro `wstoken`. Tampoco lo expone en la práctica, porque el cliente HTTP no tiene habilitado el nivel de log que imprimiría la URL completa, y la tabla que registra cada sincronización guarda solo el resultado y los mensajes de error de cada corrida, nunca la URL ni el token usados. Esta protección es un efecto de la configuración actual, no una política de redacción explícita, y queda como mejora concreta implementar un filtro que enmascare cualquier parámetro sensible antes de escribir un log. En cuanto al almacenamiento, el token vive únicamente como variable de entorno del servidor, como ya se explica en la sección 3.9, y no se persiste en la base de datos ni en el código fuente. Es, sin embargo, un valor fijo, sin ningún mecanismo de rotación periódica, lo que queda como otra mejora pendiente para un entorno de producción real.

Una vez que llega la respuesta de Moodle, el sync-service cruza cada calificación con el componente de evaluación correspondiente (docencia, práctica, autónoma, evaluación) y

con la inscripción del estudiante en ese paralelo, y recién ahí escribe en la base de datos del academic-service. Si un curso trae un estudiante que el MRNA no tiene matriculado, o una calificación que no corresponde a ningún componente reconocido, ese registro se descarta y queda anotado como error de la corrida, en lugar de detener toda la sincronización por un solo dato mal mapeado. Cada corrida, exitosa o con errores, deja un registro consultable con la fecha, el paralelo afectado y el resultado, lo que permite revisar después si una sincronización trajo menos notas de las esperadas.

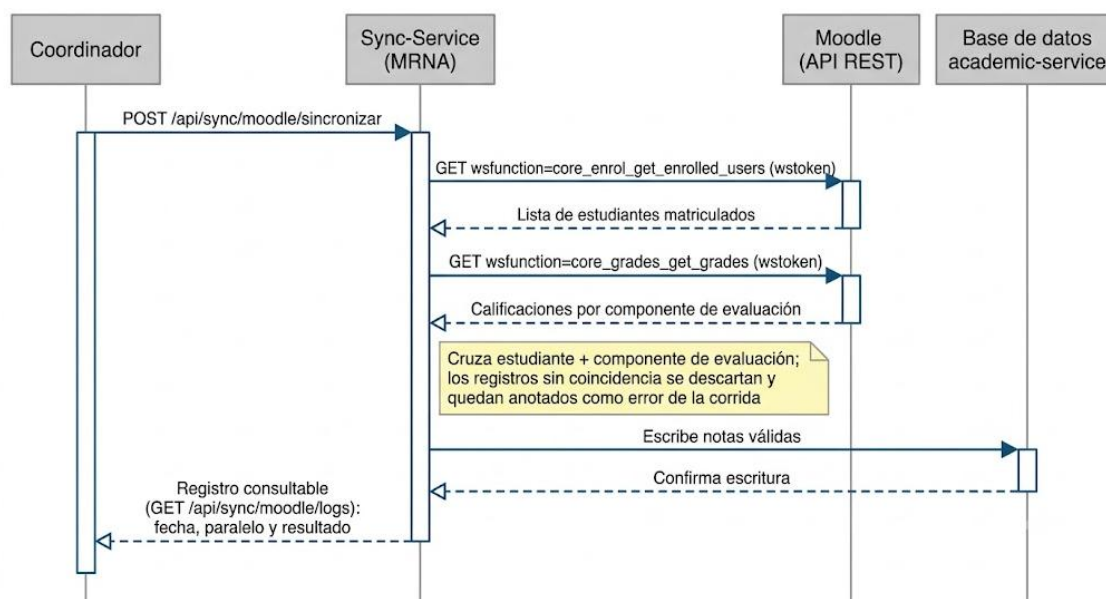


Figura 12. Secuencia de sincronización Moodle → MRNA. Elaboración propia con asistencia de inteligencia artificial.

3.10.2. Adaptador SIGA dual

Un punto que requiere explicación directa es por qué el prototipo no se integra directamente con el SIGA. La razón no es una limitación de diseño. Es una restricción real de la institución. Para el ISTA, el SIGA no expone una API pública ni una vía de integración programática hacia terceros (Responsable de TIC, comunicación personal, 2025), como se documenta en la sección 1.2.3. El registro de notas y asistencia hacia el sistema nacional depende, para el caso del ISTA, de la carga manual de archivos a través de su propia interfaz web.

Frente a esa restricción, se evaluaron tres caminos. El primero, acceso directo a la base de datos o a una API del SIGA, se descartó de entrada porque esa vía no existe y no depende del MRNA habilitarla. El segundo, el modo archivo, es el que opera en el prototipo. El sync-service, a través de su endpoint de exportación, toma las notas ya

consolidadas que obtiene del academic-service y genera un archivo con el formato y las columnas que el SIGA espera para su carga manual. Como esos datos provienen del propio sistema y ya cumplieron las validaciones de rango y formato en el momento del registro, el archivo se entrega al coordinador listo para subir, de modo que los errores se evitan dentro del MRNA y no aparecen después de intentar la carga. El tercero, el modo API, mantiene la misma interfaz interna del adaptador preparada para conectarse a un servicio real si la institución llega a obtenerlo en el futuro, sin que el resto del sistema tenga que cambiar.

Hoy el prototipo opera únicamente en modo archivo. El modo API existe como una interfaz reservada en el código (un valor de configuración documentado), pero no está conectada a ningún endpoint real ni a una simulación ejecutable, su propósito es dejar prevista la extensión futura, no demostrarla en este prototipo.

Tabla 13. Comparativa de las tres alternativas evaluadas para la integración con SIGA. Nota. Elaboración propia, 2026.

Alternativa	Descripción	Decisión
Acceso Directo	Conexión directa a la base de datos o a una API del SIGA	Descartada: el SIGA no expone ninguna de las dos
Modo archivo	Generación y validación de un archivo de carga manual con el formato del SIGA	Adoptada como vía principal del prototipo
Modo API	Adaptador preparado para una integración programática futura	Adoptada como vía complementaria, a la espera de que el SIGA habilite un canal real

Esta solución de dos modos detrás de una sola interfaz interna es la que documenta ADR-003. El resto del sistema le pide al adaptador enviar el reporte al SIGA sin saber si por dentro eso significa generar un archivo o llamar a un servicio. Si la institución llega a habilitar una API real, la implementación del modo API es el único componente que debe conectarse a ese servicio, sin modificar el resto del sistema, porque la interfaz interna del adaptador permanece igual.

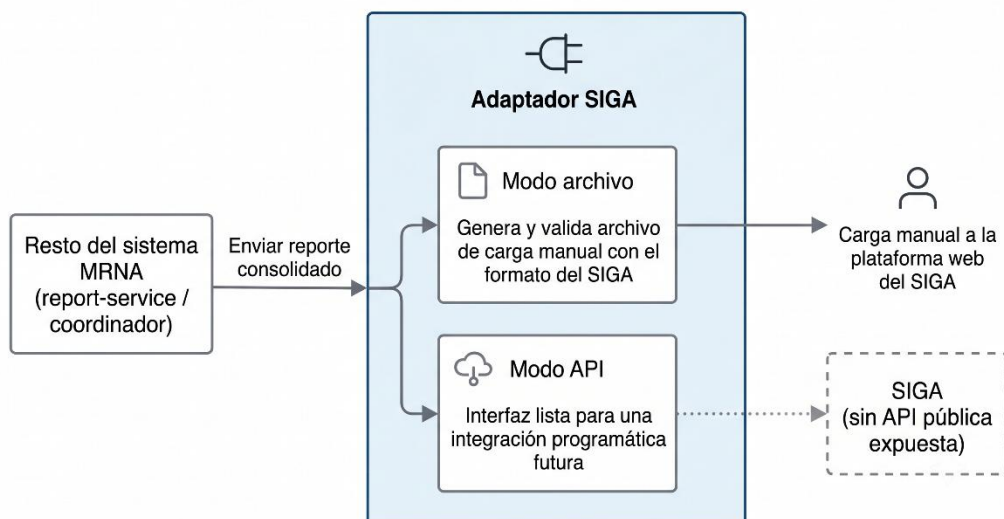


Figura 13. Diagrama del adaptador dual SIGA con dos modos detrás de una misma interfaz interna. Elaboración propia con asistencia de inteligencia artificial.

3.11. Cumplimiento de Requerimientos No Funcionales en la arquitectura

El catálogo de requerimientos no funcionales presentado en la sección 2.9.4 reunió dieciséis condiciones de calidad que el MRNA debía cumplir, además de hacer lo que tenía que hacer. Esta sección cierra ese compromiso: por cada uno de esos dieciséis puntos, se explica con qué decisión de arquitectura, de las ya descritas en este capítulo, se resuelve, y en qué parte del Capítulo 4 se comprueba que efectivamente se cumple. Para entenderlo, no basta con decir que el sistema "es seguro". Hay que decir exactamente qué hace seguro al sistema (en este caso, que cada acción exige una sesión válida y que cada rol solo puede tocar lo que le corresponde).

Tabla 14. Mapeo de requerimientos no funcionales a sus decisiones arquitectónicas. Nota. Elaboración propia, 2026.

Código	Lo que exige	Cómo se resuelve en el MRNA	Dónde se explica	Dónde se comprueba
RNF-01	Nadie entra sin iniciar sesión	Cada acción exige un token de acceso vigente	3.9	4.4: una petición sin token es rechazada
RNF-02	Cada rol ve y hace solo lo suyo	El backend revisa el rol antes de ejecutar cualquier acción	3.9	4.4: una petición con el rol equivocado es rechazada
RNF-03	Las contraseñas no	Hash irreversible de contraseñas	3.9	4.4: revisión directa de la base de datos

	se guardan al descubierto			
RNF-04	Todo cambio en notas o asistencia queda registrado	Bitácora automática a nivel de base de datos	3.9	4.4: revisión de la bitácora tras un lote de pruebas
RNF-05	El sistema responde rápido en el uso diario	Microservicios separados por responsabilidad, cada uno liviano	3.6, 3.7	4.4: medición de tiempo de respuesta con varios usuarios a la vez
RNF-06	El sistema aguanta el uso simultáneo de varias personas	Un punto de entrada único (api-gateway) que distribuye la carga	3.3	4.4: prueba de carga sostenida
RNF-07	El reporte para el SIGA se genera sin demoras	Generación delegada a un servicio dedicado solo a reportes	3.6	4.4: medición del tiempo de generación de un reporte completo
RNF-08	Una falla de Moodle o del archivo SIGA no debe tumbar el sistema	Los errores de sincronización o de validación se capturan y se registran, sin detener el resto del proceso	3.10	4.4: prueba con datos defectuosos a propósito
RNF-09	Pocos errores inesperados durante el uso normal	Manejo centralizado de errores con respuestas claras (Tabla 12)	3.7	4.4: revisión de registros durante la validación
RNF-10	El entorno se levanta sin complicaciones	Todo el sistema arranca con un solo comando de Docker	3.3	4.4: prueba de instalación cronometrada
RNF-11	El código tiene un mínimo de pruebas que lo respalden	Pruebas unitarias sobre servicios y controladores	3.14	4.4: reporte de cobertura
RNF-12	El código es fácil de mantener a futuro	Convenciones de nombrado y organización consistentes entre los seis microservicios	3.6	4.4: revisión de calidad de código

RNF-13	Cualquier desarrollador puede entender la API sin preguntar	Documentación automática de cada endpoint	3.7	4.4: verificación de cobertura de la documentación
RNF-14	El sistema no depende de un solo sistema operativo	Todo corre dentro de contenedores Docker	3.3	4.4: prueba en Windows, Linux y macOS
RNF-15	Los usuarios reales lo encuentran fácil de usar	Interfaz consistente y pensada por rol	3.8	4.2, 4.3: resultados de la encuesta SUS
RNF-16	La interfaz es usable también para personas con alguna limitación	Componentes de interfaz contruidos sobre una base accesible	3.8.1	4.4: revisión contra los criterios básicos de accesibilidad web

Hay tres puntos de calidad que un prototipo de un solo desarrollador no puede cubrir igual que un sistema en producción real: que el sistema siga funcionando si un servidor se cae, que la carga se reparta automáticamente entre varias máquinas, y que existan paneles que avisen solos cuando algo anda mal. Estos tres quedan reconocidos como trabajo pendiente para una eventual puesta en producción, y se retoman como recomendación en la sección 5.4.

3.12. Aplicación de la metodología Kanban

La decisión de adoptar Kanban en lugar de Scrum, justificada en el ADR-004, se tradujo en un tablero concreto en GitHub Projects, bajo el nombre "MRNA - Tesis Kanban". Conviene separar dos planos antes de entrar en detalle, y es que, Kanban es la metodología, con los principios de visualización del flujo, límite de trabajo en curso y gestión activa explicados en la sección 2.11, mientras que, GitHub Projects es únicamente la herramienta donde esos principios se implementan mediante columnas, campos y automatizaciones concretas. Lo que sigue describe la configuración de la herramienta, y con ella, cómo esa configuración pone en práctica los principios de la metodología, qué

reglas rigen el paso de una tarjeta de un extremo a otro, desde que una idea entra al Backlog hasta que se da por terminada.

El tablero se organiza en seis columnas, cada una con un propósito específico dentro del flujo de trabajo.

Tabla 15. Columnas del tablero Kanban del proyecto MRNA y su propósito. Nota. Elaboración propia con base en ADR-004, 2026.

Columna	Propósito
Backlog	Reúne todas las tareas identificadas y priorizadas, sin fecha de inicio asignada.
Sprint Actual	Tareas seleccionadas para el ciclo de dos semanas en curso, listas para tomarse.
En Desarrollo (máx. 3)	Tareas activamente en construcción. El límite de tres obliga a cerrar antes de abrir una nueva.
En Pruebas (máx. 3)	Tareas ya construidas, a la espera de verificación funcional.
Terminado	Tareas verificadas y cerradas.
Bloqueado	Tareas detenidas por una causa externa al desarrollador, con la razón registrada en el propio issue.

Cada tarjeta lleva, además del título y la descripción, cuatro campos personalizados que permiten filtrar y medir el trabajo, estos son Etiqueta, Microservicio, Estimación y Sprint (este último conserva el nombre que trae por defecto la plantilla de GitHub Projects, aunque en el texto de este documento nos referimos a la cadencia bisemanal). La Etiqueta identifica el tipo de trabajo detrás de la tarjeta, GES para tareas de gestión del propio proyecto (cierres de cadencia, configuración del tablero), DOC para documentación, INF para infraestructura, DB para cambios de base de datos y RF para un requerimiento funcional. La Estimación usa una escala Fibonacci ajustada (1, 2, 3, 5, 8 y 13 puntos), donde los saltos entre números reflejan un cambio real de dificultad y no una medición exacta de horas. Una tarjeta de 1 punto es trabajo de menos de medio día con pasos conocidos, mientras que una de 13 puntos es, en la práctica, lo que este documento llama tarjeta-épica, una tarjeta que agrupa un requerimiento funcional completo y que debe partirse en piezas más pequeñas antes de tocarla.

Esa partición es justamente la regla que ordena las tarjetas de tipo RF. Cada requerimiento funcional nace como una tarjeta-épica padre, y sus historias de usuario se registran como tarjetas-historia hijas, que a su vez se descomponen en sub-tareas técnicas puntuales. Las tarjetas-historia no se crean para todo el catálogo de requerimientos, se crean al arrancar la cadencia en la que efectivamente se va a trabajar esa épica, de modo que el tablero refleja trabajo real y no una planificación especulativa a varios meses. La misma lógica de trazabilidad se extiende al código, cada tarjeta que implica cambios en el repositorio se trabaja en una rama con el formato `feature/<código>-<descripción-corta>` (por ejemplo, `feature/RF-005-roles-permisos`), de manera que cualquier commit puede rastrearse hasta la tarjeta que lo originó.

El catálogo de tarjetas trabajadas a lo largo del proyecto, con su etiqueta, estimación, cadencia de ejecución y demás campos, se documenta en el Anexo I, junto con la exportación del tablero al corte de la documentación.

El paso de una tarjeta entre columnas no es una decisión arbitraria, responde a dos políticas explícitas, documentadas en el DOC-09, que existen precisamente para que las métricas de flujo del Capítulo 4 se apoyen en criterios verificables y no en una sensación subjetiva de "esto ya está". La Definition of Ready determina cuándo una tarjeta puede salir de Sprint Actual y entrar a En Desarrollo. Exige que tenga un código y un título claros, un contexto breve de por qué existe, entre tres y cinco criterios de aceptación verificables sin pasos técnicos de por medio, los cuatro campos personalizados completos, y que cualquier tarjeta de la que dependa ya esté en Terminado. Si la tarjeta es una tarjeta-épica, sus tarjetas-historia hijas deben existir como sub-issues antes de moverla. La Definition of Done determina cuándo una tarjeta puede pasar de En Pruebas a Terminado. El código debe estar fusionado a la rama principal mediante un Pull Request, las pruebas asociadas deben pasar sin errores, y si la tarjeta es una tarjeta-épica, las tarjetas-historia deben estar a su vez cerradas. Al tratarse de un proyecto de un solo desarrollador, no existe un segundo revisor que apruebe el Pull Request desde un rol distinto al del propio autor. La aprobación es una autorrevisión. El autor del prototipo confirma contra el checklist que el código cumple los criterios de aceptación de la tarjeta antes de fusionarlo, apoyado en que la integración continua de GitHub Actions haya pasado en verde como control automático adicional que no depende de su propio criterio. El Pull Request cumple entonces una función de trazabilidad y de punto de control estructurado, no de revisión por pares independiente, una condición inherente a un proyecto desarrollado por una sola persona. Una tarjeta que falla en la verificación no se

descarta, regresa a En Desarrollo y la causa del retroceso queda anotada, dato que también alimenta el análisis del Capítulo 4.

Toda esta disciplina de columnas, etiquetas y políticas de paso no es un fin en sí mismo: existe para producir datos de flujo confiables. El propio tablero, junto con la exportación periódica de su historial, es la fuente de las métricas de lead time, cycle time, throughput, WIP efectivo y diagrama de flujo acumulado que se presentan y analizan con cifras concretas en la sección 4.5.

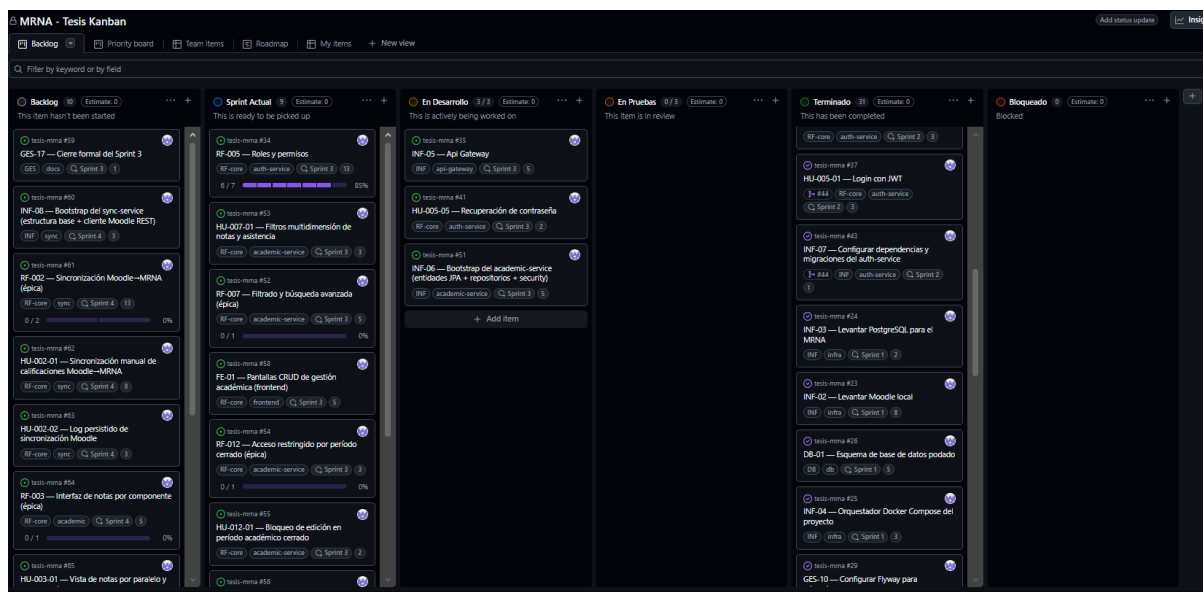


Figura 14. Captura del tablero Kanban al cierre de la Cadencia 3. Captura del prototipo, elaboración propia, 2026.

3.13. Implementación del prototipo

Todo lo descrito hasta aquí en este capítulo (la arquitectura, la seguridad, las integraciones con Moodle y SIGA) se implementó en el prototipo funcional, distribuido entre los seis microservicios del backend y la aplicación del frontend. Esta sección explica, en términos generales, cómo se tradujo ese diseño en código.

El control de quién puede hacer qué, explicado antes como una regla de seguridad, se aplica de la misma manera en cada acción concreta del sistema. Por ejemplo, cuando alguien intenta registrar una calificación, el backend revisa primero si esa persona tiene el rol adecuado (docente, coordinador o administrador) antes de permitir que la nota se guarde. Si el rol no corresponde, la petición se rechaza de inmediato, sin importar lo que se haya mostrado en la pantalla.

La nota de recuperación es un buen ejemplo de cómo una regla institucional se convirtió en una decisión de diseño concreta. En lugar de tratarla como un quinto componente más, mezclado con Docencia, Práctica, Autónoma y Evaluación, se modeló como un registro propio, vinculado de forma única a la inscripción de cada estudiante en su asignatura. Esto refleja con fidelidad cómo funciona la recuperación en la práctica: no es una nota más del parcial, es un mecanismo aparte que se aplica solo cuando el promedio del estudiante cae dentro del rango que da derecho a recuperar.

La validación de los archivos en formato SIGA se implementó como una revisión fila por fila, antes de entregarlos o antes de aceptarlos según el sentido del flujo: se comprueba que cada dato tenga el formato esperado (una cédula con sus diez dígitos, una nota dentro del rango de cero a diez, un porcentaje de asistencia entre cero y cien, entre otros) y, si algo no cumple, se informa exactamente en qué fila y en qué campo está el problema, en lugar de simplemente rechazar el archivo completo sin explicación. Esta misma lógica de validación la reutiliza el sync-service en sus dos mecanismos: en la exportación oficial hacia el SIGA, revisa el archivo generado antes de ponerlo a disposición del coordinador o del administrador y en el mecanismo legado, revisa el archivo que se sube antes de importar sus notas hacia el MRNA.

En el frontend, la pantalla donde el docente consulta las notas de un paralelo se construyó pensando en que la información se carga por pasos: primero se elige el período académico, lo que habilita la lista de paralelos disponibles; al elegir un paralelo, recién entonces se cargan las notas correspondientes. Esto evita peticiones innecesarias al servidor y asegura que el docente nunca vea una tabla vacía o con datos de un paralelo distinto al que seleccionó.

3.14. Validación funcional del prototipo

Antes de presentar los resultados de validación en el Capítulo 4, esta sección documenta cómo se diseñó y ejecutó esa validación: qué se probó, con qué herramientas y con qué alcance real sobre el prototipo construido hasta este punto.

3.14.1. Estrategia de validación

Validar un sistema de microservicios no se reduce a comprobar que cada endpoint responde. El prototipo MRNA se valida desde cinco ángulos complementarios, cada uno respondiendo una pregunta distinta sobre la calidad del software construido.

El primer ángulo es funcional: confirma que cada requerimiento (RF) hace, en la práctica, lo que su historia de usuario describe. El segundo es de calidad no funcional, apoyado en el catálogo de dieciséis RNF presentado en el apartado 2.9.4 y en el modelo de calidad de ISO/IEC 25010, y revisa atributos como seguridad, mantenibilidad o portabilidad que no se notan en una sola pantalla, sino en el comportamiento del sistema a lo largo del tiempo. El tercero es de usabilidad, medido con el cuestionario System Usability Scale (SUS), aplicado en esta fase a un único evaluador experto del ISTA, el Docente de Desarrollo de Software, cuyo perfil y las tareas que ejecutó se detallan en la sección 4.2. El cuarto es de desempeño, que observa los tiempos de respuesta del sistema bajo condiciones de uso simuladas. El quinto, menos técnico, pero igual de importante para esta tesis, es metodológico, confirma que el proceso Kanban descrito en el ADR-004 se siguió de verdad, con evidencia de flujo (lead time, cycle time, throughput).

Tabla 16. Dimensiones de la estrategia de validación del prototipo MRNA. Nota. Elaboración propia, 2026.

Dimensión	Qué confirma	Cómo se mide
Funcional (RF)	Que el sistema cumple lo descrito en cada requerimiento	Evidencia de prueba automatizada y trazabilidad RF↔prueba
Calidad no funcional (RNF)	Atributos de calidad según ISO/IEC 25010	Verificación por atributo según el método definido en el DOC-08
Usabilidad	Percepción real de uso por parte de docentes y coordinación	Cuestionario SUS
Desempeño	Tiempos de respuesta del sistema	Pruebas de carga sobre los flujos más usados
Metodología (Kanban)	Que el proceso de desarrollo siguió lo planificado	Métricas de flujo: lead time, cycle time, throughput

Las tres últimas dimensiones (usabilidad, desempeño y metodología) se miden con datos que solo existen al cierre del desarrollo, por lo que sus resultados se presentan en el Capítulo 4.

3.14.2. Plan de pruebas por requerimiento funcional

La validación funcional requiere definir primero qué se va a comprobar y cómo se determinará que cada requerimiento quedó bien resuelto. Este plan recorre los catorce requerimientos funcionales centrales del prototipo y, para cada uno, fija el escenario principal que lo pone a prueba, el criterio concreto con el que se da por aprobado y el nivel de prueba que lo respalda. La fuente de cada fila es la historia de usuario asociada

en el catálogo de requerimientos (DOC-06), de modo que la validación se mantiene atada a lo que el usuario pidió y no a una interpretación posterior.

El nivel de prueba sigue la lógica de la pirámide de pruebas: la lógica de negocio aislada (cálculos, validaciones, reglas) se cubre con pruebas unitarias, la colaboración entre un servicio y la base de datos real se cubre con pruebas de integración sobre contenedores temporales, y los recorridos completos que más se repiten en el día a día se cubren con pruebas de extremo a extremo. Un mismo requerimiento puede aparecer en más de un nivel cuando su complejidad lo necesita.

Tabla 17. Plan de pruebas funcionales por requerimiento del prototipo MRNA. Nota.

Elaboración propia, 2026.

RF	HU	Escenario de prueba principal	Criterio de aceptación verificado	Nivel de prueba
RF-001	HU-001	Generar el archivo de exportación de notas consolidadas para un período, carrera y paralelo, en el formato que exige el SIGA.	El archivo contiene la nota final correcta de cada estudiante del paralelo, incluida la recuperación cuando aplica, y el sistema detiene la generación si algún dato no cumple el formato esperado.	Integración y E2E
RF-002	HU-002	Sincronizar un curso de Moodle y contrastar sus calificaciones contra el aula.	Las notas se reflejan en el MRNA sin duplicados y una segunda ejecución no altera los datos (idempotencia).	Integración y E2E
RF-003	HU-003	Registrar las cuatro componentes de un parcial y revisar el promedio.	El promedio aplica la ponderación 20/30/30/20 y queda asociado al estudiante y parcial correctos.	Unitaria y E2E
RF-004	HU-004	Generar el histograma de rendimiento de un paralelo.	Los rangos de nota agrupan bien a los estudiantes y los totales coinciden con el número de matriculados.	Unitaria
RF-005	HU-005	Acceder a un recurso protegido con y sin el rol requerido.	El acceso se concede solo al rol autorizado y se rechaza en caso contrario; el token expira según lo configurado.	Unitaria e Integración
RF-006	HU-006	Procesar un parcial con estudiantes por debajo del umbral configurado.	Se genera una alerta por cada estudiante bajo el umbral (nota.umbral.bajo_rendimiento) y ninguna para quienes lo superan.	Unitaria

RF-007	HU-007	Aplicar filtros combinados (período, carrera, paralelo, parcial) sobre las notas.	El resultado contiene únicamente los registros que cumplen todos los filtros aplicados.	Unitaria e Integración
RF-008	HU-008	Descargar el reporte consolidado de un paralelo en los formatos disponibles.	El archivo se genera en el formato elegido y contiene las columnas y los totales esperados.	Unitaria y frontend
RF-010	HU-009	Cargar un archivo con errores de estructura y de contenido.	El sistema reporta cada error y bloquea la importación mientras quede alguno.	Unitaria
RF-012	HU-010	Intentar modificar una nota en un período cerrado.	El sistema rechaza la modificación y conserva el valor original.	Integración y E2E
RF-013	HU-011	Importar un archivo de docentes con la plantilla del MRNA.	Se crean las cuentas válidas, se omiten las duplicadas y se informa el resultado de la operación.	Integración
RF-014	HU-012	Ejecutar la tarea programada de notificaciones en la fecha configurada.	La notificación se envía una sola vez por evento y queda registrada.	Unitaria
RF-015	HU-013	Editar una nota dentro y fuera de la ventana de 48 horas posterior al cierre.	La edición se permite dentro de las 48 horas y se rechaza una vez vencida la ventana.	Unitaria
RF-018	HU-014	Crear, modificar y eliminar una nota y revisar la bitácora de auditoría.	Cada operación queda registrada con usuario, fecha y valores; la bitácora no admite UPDATE ni DELETE.	Integración

Las tres subsecciones siguientes detallan la ejecución de este plan en cada nivel: primero las pruebas unitarias sobre la lógica de negocio, luego las pruebas de integración que conectan cada servicio con una base de datos real y, por último, los flujos de extremo a extremo que recorren los recorridos críticos del sistema completo.

3.14.3. Pruebas unitarias

Una prueba unitaria aísla una sola pieza de lógica de negocio (por ejemplo, el cálculo de la nota final con recuperación, o la validación de una cédula en un archivo de carga) y verifica que produce el resultado esperado, sin necesidad de levantar una base de datos

real ni los demás microservicios. Esta independencia las hace rápidas y permite ejecutarlas con frecuencia durante el desarrollo.

El prototipo cuenta hoy con 150 pruebas unitarias distribuidas entre los seis microservicios del backend, además de un conjunto de pruebas sobre el flujo de descarga de reportes en el frontend. La herramienta JaCoCo mide automáticamente, en cada compilación, qué porcentaje del código realmente se ejecuta durante esas pruebas.

Tabla 18. Cobertura de pruebas unitarias por microservicio. Nota. Elaboración propia a partir de los reportes JaCoCo del proyecto, 2026.

Microservicio	Líneas cubiertas	Ramas de decisión cubiertas	Pruebas unitarias
auth-service	61.7 %	12.3 %	35
academic-service	53.4 %	43.8 %	48
sync-service	46.7 %	58.9 %	34
report-service	61.5 %	50.5 %	7
notification-service	42.6 %	29.6 %	11
api-gateway	90.2 %	36.8 %	15

La cobertura de líneas se concentra, como es esperable en esta etapa, en los caminos principales de cada servicio, el cálculo de notas, la validación de archivos SIGA, la sincronización con Moodle y la emisión de tokens de autenticación. La cobertura de ramas de decisión mide cuántos de los caminos alternativos dentro de una misma función ya están probados, incluyendo casos de error menos frecuentes, y es naturalmente más baja en esta etapa. Varios de esos caminos alternativos corresponden a validaciones secundarias que, al cierre de la Cadencia 5, todavía no están cubiertas por pruebas. Ampliar esa cobertura queda documentado como trabajo pendiente en el Capítulo 5.

3.14.4. Pruebas de integración entre microservicios

Las pruebas unitarias prueban piezas aisladas. Las pruebas de integración, en cambio, prueban que esas piezas funcionan correctamente cuando se conectan con una base de datos real. Para esto, el prototipo usa Testcontainers, una herramienta que levanta automáticamente una instancia real de PostgreSQL exclusivamente para la duración de la prueba, y la destruye al terminar. Esto da más confianza que probar contra una base de datos simulada, porque corre contra el mismo motor que se usaría en producción. Sin embargo, esto no equivale a validar el sistema en un entorno de producción real. La prueba corre en un contenedor aislado, con datos controlados para el escenario de prueba

y sin la carga concurrente, la configuración de red ni el volumen de información de un despliegue real. Lo que esta prueba confirma es el comportamiento correcto de la lógica frente a una base de datos del mismo motor, no el comportamiento del sistema completo bajo condiciones de producción.

El prototipo cuenta con seis escenarios de integración, ejecutados automáticamente en cada cambio subido al repositorio a través del flujo de integración continua configurado en GitHub Actions. Estos escenarios verifican, en conjunto, diecisiete casos concretos: que los disparadores (triggers) de auditoría en PostgreSQL registran correctamente cada modificación o eliminación de una nota, que las restricciones de no duplicidad sobre el correo institucional impiden registrar dos veces a la misma persona durante una carga masiva, que las migraciones de base de datos se aplican sin conflictos, que el flujo completo de sincronización con Moodle trae y guarda correctamente la información del aula virtual, que el mapeo entre cédula, matrícula e inscripción asigna a cada estudiante sincronizado la inscripción que le corresponde, y que un período académico abierto o cerrado restringe las operaciones de registro de notas tal como exige el RF-012.

3.14.5. Pruebas end-to-end de los flujos críticos

Además de las pruebas unitarias y de integración, el prototipo se valida siguiendo tres flujos completos de uso real, de principio a fin, que atraviesan la interfaz, el microservicio correspondiente y la base de datos.

El primer flujo cubre el ciclo de un docente: iniciar sesión, registrar las notas de sus estudiantes en un parcial y comprobar que, una vez que el período académico se cierra, el sistema impide nuevas modificaciones. El segundo flujo cubre la sincronización con Moodle: traer la información de un aula virtual hacia el MRNA y confirmar que cada estudiante importado queda correctamente vinculado a su matrícula real. El tercer flujo cubre la salida hacia SIGA: generar, a partir de las notas ya registradas, el archivo consolidado en formato XLSX que el coordinador puede descargar para la carga manual en el sistema nacional.

3.15. Síntesis del capítulo

Este capítulo documentó cómo se construyó el prototipo del MRNA, partiendo del catálogo de requerimientos y las decisiones arquitectónicas planteadas en capítulos anteriores. Se describió la organización del repositorio y la separación de los seis

microservicios, el modelo de datos resultante de depurar el esquema original hasta las tablas realmente necesarias para el alcance de esta tesis, los mecanismos de seguridad y control de acceso por rol, y la manera concreta en que el prototipo resuelve el problema central planteado en el diagnóstico: automatizar el registro y la consolidación de notas entre Moodle y el MRNA, y resolver la carga hacia el SIGA mediante un adaptador que genera el archivo listo para su carga manual, sin depender de un acceso que el ISTA no tiene. También se documentó el proceso de trabajo bajo la metodología Kanban adoptada para este desarrollo, con su tablero, sus políticas de paso entre columnas y la disciplina de estimación que sostiene las métricas de flujo.

La parte final del capítulo estableció la estrategia con la que se valida que todo lo anterior funciona: pruebas unitarias sobre la lógica de negocio de cada microservicio, pruebas de integración contra una base de datos real, y pruebas de extremo a extremo sobre los flujos que un usuario del ISTA recorrería en la práctica. Esta estrategia entrega evidencia técnica del comportamiento del prototipo, pero no agota la validación: queda pendiente medir cómo perciben el sistema sus usuarios reales, qué tan rápido responde bajo carga, en qué medida cumple los dieciséis requerimientos no funcionales levantados en el diagnóstico, y si el proceso Kanban realmente se comportó como se planificó en términos de tiempos de entrega.

Esas cuatro dimensiones (usabilidad, desempeño, cumplimiento de RNF y métricas de metodología) son precisamente el contenido del Capítulo 4, que toma como insumo todo lo construido y probado aquí para reportar resultados concretos y discutirlos frente a los objetivos planteados al inicio de esta tesis.

Capítulo 4. Resultados y Discusión

4.1. Introducción del capítulo

El capítulo anterior cerró señalando lo que separaba el prototipo ya construido de su validación: qué tan utilizable resulta para quien lo va a operar todos los días, qué tan rápido responde bajo una carga simulada, en qué medida cumple los requerimientos no funcionales planteados desde el diagnóstico, y si el proceso Kanban adoptado para este desarrollo se comportó tal como se anticipó en el plan de cadencias. Este capítulo retoma esas cuatro preguntas y las responde con evidencia. La sección 4.2 presenta la validación de usabilidad realizada mediante el cuestionario SUS. La sección 4.3 reporta el desempeño del prototipo y lo compara contra el proceso manual diagnosticado en el Capítulo 2. La sección 4.4 verifica el cumplimiento de los requerimientos no funcionales de seguridad, fiabilidad, mantenibilidad y portabilidad. La sección 4.5 recoge las métricas de flujo obtenidas en el tablero Kanban a lo largo de las cinco cadencias que cubre el corte de documentación, de los siete planificados en total. El capítulo cierra discutiendo estos resultados frente a los objetivos de la tesis (4.6), reconociendo las limitaciones del estudio (4.7) y sintetizando los hallazgos (4.8).

4.2. Validación de usabilidad

4.2.1. *Protocolo de validación*

La validación de usabilidad del prototipo combina dos instrumentos complementarios: un conjunto de tareas guiadas sobre el sistema real y un cuestionario estandarizado, el System Usability Scale (SUS), tal como lo describe el protocolo documentado en el Anexo M.

Las tareas guiadas reproducen el uso que un docente, un coordinador o un administrador del ISTA haría del sistema. El guion completo de las ocho tareas se presenta más adelante, junto con sus resultados, en la tabla de la sección 4.2.3. Cada tarea corresponde a una función realmente disponible en el prototipo, de modo que la evaluación se realiza sobre el sistema en funcionamiento y no sobre una maqueta.

Una vez terminadas las tareas, el evaluador responde el cuestionario SUS, compuesto por diez afirmaciones, 1 (totalmente en desacuerdo) a 5 (totalmente de acuerdo). El puntaje se calcula restando uno a la respuesta de cada ítem impar y restando la respuesta de cada

ítem par a cinco; la suma de los diez valores se multiplica por 2,5 para obtener un resultado entre 0 y 100. Sobre esa escala, un puntaje superior a 68 se interpreta como un nivel de usabilidad aceptable y uno superior a 80 como excelente, criterio de interpretación consistente con el aplicado en evaluaciones recientes de sistemas de gestión académica en educación superior [31]. El protocolo cierra con una sección de retroalimentación cualitativa abierta, en la que el evaluador compara el prototipo contra el proceso manual vigente (Moodle – Excel - SIGA) y señala ventajas, funciones faltantes y posibles puntos de confusión.

4.2.2. Población participante

A diferencia de una validación de usabilidad basada en una muestra estadística de usuarios finales, este protocolo se apoya en la evaluación de un único experto: el Docente de Desarrollo de Software del Instituto Superior Tecnológico del Austro, el mismo cargo consultado durante el diagnóstico del Capítulo 2 para caracterizar el proceso manual de registro entre Moodle, Excel y SIGA.

La selección de este perfil responde a dos criterios. Por un lado, su conocimiento previo del proceso manual que el MRNA busca remplazar le permite comparar el flujo automatizado contra una línea base que ya conoce de primera mano, en lugar de evaluar el sistema de forma aislada y sin un punto de referencia. Por otro lado, su formación en desarrollo de software le permite señalar con vocabulario técnico problemas de consistencia, navegación o estructura de la interfaz que un usuario sin ese trasfondo podría percibir, pero no necesariamente describir con precisión durante la retroalimentación cualitativa.

Este diseño constituye, en sentido estricto, un estudio exploratorio de caso único, sus resultados documentan la percepción de ese evaluador y no permiten caracterizar la usabilidad percibida por la población general de docentes del ISTA, ni por los otros tres roles institucionales (coordinador, administrador y directivo), que no fueron incluidos en esta ronda de validación.

4.2.3. Resultados

Los resultados de la validación de usabilidad se organizan en dos planos: el desempeño del evaluador al ejecutar las tareas guiadas y la puntuación obtenida en el cuestionario SUS. Ambos se recogieron en una sesión con el Docente de Desarrollo de Software, quien

operó el prototipo con el rol de administrador (el perfil con acceso completo al menú), de acuerdo con el protocolo descrito en 4.2.1.

En cuanto a las tareas guiadas, el evaluador completó las ocho propuestas, lo que representa una tasa de finalización del 100% para ese único evaluador (Tabla 19). El recorrido abarcó el inicio de sesión y la navegación del menú, la consulta de notas y del promedio ponderado de un estudiante, la consulta y justificación de asistencia, el filtrado de calificaciones por período, carrera o asignatura, la creación de un registro de catálogo, la carga masiva de docentes desde un archivo, la validación de un archivo antes de enviarlo y la descarga del reporte consolidado en formato Excel. Ninguna tarea quedó incompleta ni requirió que se interviniera para resolverla. Durante la sesión, el evaluador registró además un conjunto de observaciones que conviene analizar: algunas, como incorporar un filtrado en forma de árbol por carrera y parcial o mostrar junto a la nota si el estudiante aprueba o pierde, son extensiones que no formaban parte del alcance original, otras coinciden con funciones que la sección 2.7 ya reconoce como pendientes o parciales al cierre de la Cadencia 5. Ambos tipos de observación se retoman en la discusión (4.6) y en el trabajo futuro del Capítulo 5, distinguiendo entre una ampliación real y una función ya identificada como incompleta.

Tabla 19. Resultados de las tareas guiadas de usabilidad del prototipo MRNA. Nota. Elaboración propia a partir de la sesión de validación con el Docente de Desarrollo de Software, 2026.

N°	Tarea	Resultado
1	Inicio de sesión y navegación del menú	Completada
2	Consulta de notas y promedio ponderado de un estudiante	Completada
3	Consulta de asistencia y justificación de una falta	Completada
4	Filtrado de notas por periodo, carrera o asignatura	Completada
5	Creación o edición de un registro de catálogo	Completada
6	Carga masiva de docentes desde archivo Excel	Completada
7	Validación de un archivo antes de enviarlo al SIGA	Completada
8	Generación y descarga del reporte consolidado (Excel)	Completada

La puntuación SUS alcanzó 87,5 sobre 100. El cálculo sigue el procedimiento estándar del instrumento [31] en los ítems impares, redactados en sentido positivo, se resta uno a la respuesta, y en los pares, redactados en sentido negativo, se resta la respuesta a cinco; la suma de las diez contribuciones (35 puntos) se multiplica por 2,5. La Figura 15 muestra el aporte de cada ítem ya normalizado en la escala de 0 a 4, donde un valor más alto siempre indica una mejor valoración. El patrón es homogéneo: los diez ítems se ubican entre 3 y 4 puntos, sin ninguna caída que delate un punto débil de la interfaz. Frente a los umbrales de referencia del propio instrumento (68 como promedio esperado y 80 como frontera de una valoración excelente), el resultado de 87,5 queda con holgura en la banda superior, lo que ubica la usabilidad percibida del prototipo en el rango de excelente [31]. El detalle de cada una de las diez respuestas y el cálculo paso a paso se puede revisar en el Anexo M.

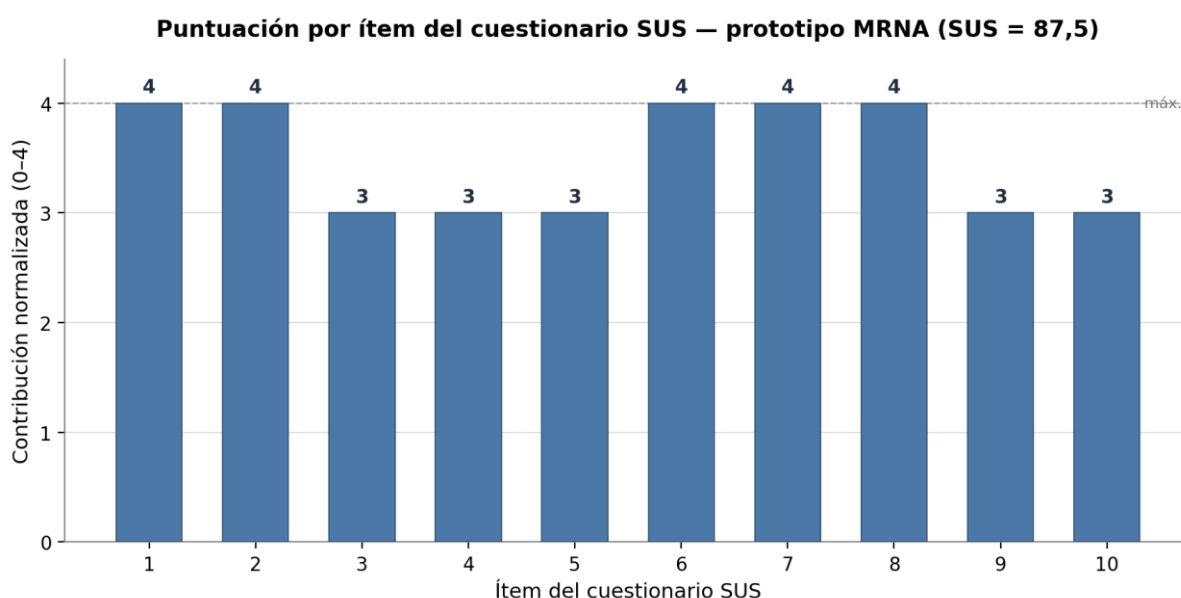


Figura 15. Puntuación por ítem del cuestionario SUS aplicado al prototipo MRNA (contribución normalizada 0–4). Nota. Elaboración propia, 2026.

Más allá del número, la retroalimentación cualitativa ayuda a interpretar de dónde viene esa valoración, y se analiza aquí junto con el resultado cuantitativo del SUS, no por separado. Al comparar el prototipo con el flujo manual que hoy sostiene en el Excel institucional, el evaluador destacó la velocidad para consultar la información y el hecho de que los datos quedan mejor organizados. Entre las funciones que consideró más útiles señaló la importación de datos y la generación de reportes con filtros, justamente las que hoy le consumen más tiempo en el proceso manual. No identificó ningún paso confuso o propenso a error durante las tareas ejecutadas. Sus observaciones señalaron también

funciones ausentes como la gestión de un docente con más de un rol (un docente que además ejerce como coordinador de carrera), el manejo de las mallas vigentes por carrera, reporte de calificaciones desglosadas por periodo y una organización de la información por período, carrera, nivel y asignatura. Las tres primeras son extensiones que no formaban parte del alcance original del prototipo y se documentan como trabajo futuro en el Capítulo 5. La última coincide con el requerimiento de indicadores agregados (RF-004), que la sección 2.7 ya identifica como pendiente al cierre de la Cadencia 5, aquí queda confirmada, desde la perspectiva de un usuario técnico, la misma brecha ya reportada en el alcance.

En conjunto, la validación respalda que el prototipo es utilizable por el perfil que lo operará en la práctica. La tasa de finalización del 100% confirma que las tareas centrales del flujo de notas y asistencia se completan sin asistencia, y la puntuación de 87,5 sitúa la experiencia en el extremo alto de la escala. Este resultado es el que sostiene el cumplimiento del requerimiento no funcional de usabilidad que se retoma en 4.4, y alimenta la discusión sobre en qué medida la propuesta responde a la problemática planteada (4.6).

4.3. Resultados de desempeño y comparativa proceso manual vs. MRNA

Esta sección reporta el comportamiento del prototipo bajo carga y lo compara contra el proceso manual diagnosticado en el Capítulo 2. El desempeño se midió con una prueba de carga reproducible (herramienta k6), y la comparación de tiempos con un cronometraje del flujo real del sistema, sobre un conjunto de 25 estudiantes del Segundo Parcial de la asignatura Desarrollo de Aplicaciones Web (paralelo A).

La prueba de carga sometió al sistema a 25 usuarios virtuales concurrentes durante un minuto en régimen estable (con 20 segundos de calentamiento y 20 de enfriamiento), completando 624 iteraciones en una única ejecución, sin repeticiones adicionales para promediar varianza, a un ritmo sostenido de 18,40 peticiones por segundo. La prueba se ejecutó en el mismo entorno descrito en la sección 4.4 (procesador AMD Ryzen 7 5800H, 13,9 GB de RAM, Windows 11 Home, Docker Engine 29.4.3), con el cliente de carga y el api-gateway corriendo en el mismo equipo (localhost), sin red física de por medio ni su latencia asociada. La base de datos, poblada mediante el script de siembra del entorno de pruebas, contenía un único paralelo activo, con 25 estudiantes matriculados e inscritos (25 registros de matrícula y 25 de inscripción), 100 registros de notas (25 estudiantes por

los cuatro componentes de evaluación) y 400 registros de asistencia (25 estudiantes por 16 sesiones). Bajo esas condiciones, los tiempos de respuesta se mantuvieron bajos y la tasa de error fue nula en todos los puntos medidos (Tabla 20). Las operaciones de lectura, que son las más frecuentes en el uso diario (consultar períodos y notas), respondieron en menos de diez milisegundos en el percentil 95 (p95). El inicio de sesión es deliberadamente el punto más lento (267 ms), porque el sistema calcula el hash de la contraseña con un algoritmo intencionalmente costoso para dificultar ataques de fuerza bruta, aun así, se mantiene muy por debajo de un umbral aceptable para una autenticación. Conviene precisar que estas condiciones (25 usuarios concurrentes, una sola ejecución, sin tráfico de red real) quedan por debajo de los criterios formales de RNF-05 (p95 menor a 2s con 50 usuarios concurrentes) y RNF-06 (100 peticiones por segundo sostenidas), definidos en la sección 2.9.4, los resultados obtenidos son favorables dentro de la condición efectivamente probada, pero no permiten afirmar el cumplimiento de esos dos criterios en su formulación original, tal como se recoge en la sección 4.4.

Conviene precisar que estas condiciones (25 usuarios concurrentes, una sola ejecución) quedan por debajo de los criterios formales de RNF-05 (p95 menor a 2s con 50 usuarios concurrentes) y RNF-06 (100 peticiones por segundo sostenidas), definidos en la sección 2.9.4, los resultados obtenidos son favorables dentro de la condición efectivamente probada, pero no permiten afirmar el cumplimiento de esos dos criterios en su formulación original, tal como se recoge en la sección 4.4.

Tabla 20. Métricas de desempeño del prototipo MRNA bajo carga. Nota. Elaboración propia a partir de la prueba de carga con k6, 2026.

Endpoint	p50 (ms)	p95 (ms)	p99 (ms)	Tasa de error
POST /api/auth/login	243	267	275	0,00%
GET /api/academic/periodos	4	6	8	0,00%
GET /api/academic/notas	7	8	9	0,00%

A nivel de cada microservicio, el tiempo de respuesta promedio confirma el mismo patrón: el academic-service (la consulta académica más usada) respondió en torno a 5,5 ms, mientras que el auth-service promedió 236 ms por el costo del cálculo del hash en el inicio de sesión.

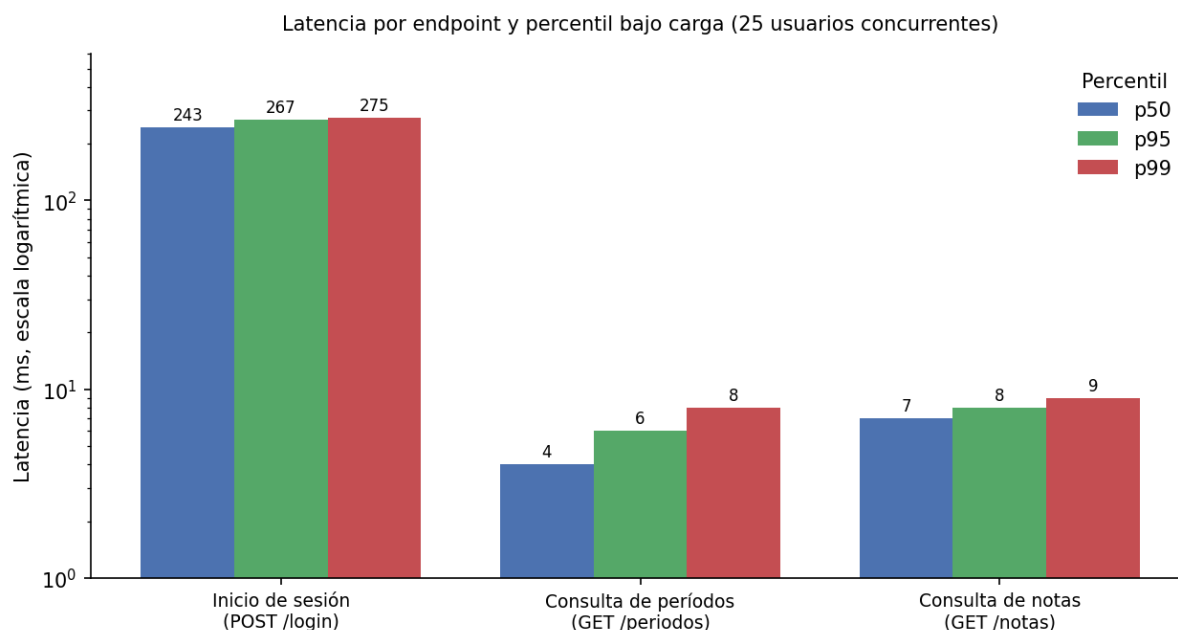


Figura 16. Histograma de latencias (p50, p95, p99) por endpoint del prototipo MRNA. Nota. Elaboración propia, 2026.

En cuanto a la comparación con el proceso manual, la diferencia observada es amplia, aunque debe leerse como un contraste entre magnitudes de distinta naturaleza y no como una medición controlada de ambos procesos bajo condiciones idénticas (Tabla 21; ver la precisión metodológica al final de esta sección). El registro de las notas de un parcial completo, que según la estimación de un docente ocupa entre seis y ocho horas por cierre, se compara aquí con el procesamiento técnico de cien calificaciones (25 estudiantes por cuatro componentes) que el MRNA completa en 2,9 segundos. La generación del reporte consolidado, que de forma manual implica cálculos en una hoja de Excel, el sistema la resuelve en 1,0 segundo entregando el archivo listo para descargar. Bajo esta comparación de magnitudes, la reducción potencial estimada supera el 99% en ambos casos, cifra que describe la distancia entre una estimación declarada y una medición técnica. A esto se suma la validación, en donde el proceso manual acumula entre tres y cinco errores por docente y parcial según el diagnóstico, el sistema rechaza automáticamente cualquier nota fuera de rango o incompleta durante la prueba técnica ejecutada.

Tabla 21. Comparativa entre el proceso manual y el prototipo MRNA. Nota. Elaboración propia; los tiempos manuales provienen del diagnóstico del apartado 2.8, 2026.

Métrica	Proceso Manual (estimado, entrevistas)	Prototipo MRNA (medido, prueba técnica)	Reducción potencial estimada
---------	---	--	------------------------------------

Tiempo de registro (~25 estudiantes, un parcial)	6–8 h (360–480 min)	2,9 s	>99,9%
Tiempo de consolidación del reporte	cálculo manual en Excel	1,0 s	>99,9%
Errores de validación por parcial	3–5 (estimado)	0 detectados en la prueba	Eliminación de errores de rango/formato en la prueba

Conviene precisar el alcance de esta comparación para no sobreinterpretarla, el MRNA no expone un formulario para registrar nota por nota: las calificaciones ingresan desde Moodle por sincronización o desde archivos del SIGA, así que el tiempo de 2,9 segundos corresponde al procesamiento que ejecuta el sincronizador, no a una digitación dentro del sistema. Además, los tiempos del proceso manual provienen de las entrevistas del diagnóstico (apartado 2.8) y no de una medición directa con cronómetro, por lo que la comparación debe leerse como una estimación de la magnitud del ahorro y no como una medición controlada de ambos procesos en idénticas condiciones. Aun con ello, la diferencia es lo bastante amplia como para sostener que la automatización elimina la carga de digitación repetida y el reproceso de errores que caracterizan al flujo actual.

4.4. Cumplimiento de los Requerimientos No Funcionales

Esta sección verifica en qué medida el prototipo cumple los requerimientos no funcionales planteados desde el diagnóstico. Los relacionados con el desempeño puro (latencia, rendimiento y consumo de recursos) se reportan en la sección 4.3, aquí se documentan los de seguridad, fiabilidad, mantenibilidad, portabilidad y usabilidad.

4.4.1. RNF de seguridad

El sistema protege el acceso de los usuarios mediante un mecanismo de sesiones temporales: cada inicio de sesión genera una credencial de acceso válida por veinticuatro horas y una credencial de renovación válida por treinta días, de modo que el usuario no debe escribir su contraseña constantemente, pero tampoco queda con una sesión abierta de forma indefinida. Cada vez que alguien vuelve a iniciar sesión, la credencial de renovación anterior queda invalidada, lo que reduce el riesgo de que una sesión robada siga siendo útil. Las contraseñas no se guardan en texto legible, sino mediante un algoritmo de hash de un solo sentido, y antes de permitir cualquier operación sensible

(registrar notas, consultar la auditoría, administrar carreras) el sistema verifica que la persona tenga el rol adecuado: docente, coordinador o administrador. Además, todas las consultas a la base de datos están construidas de forma que un usuario no puede insertar comandos maliciosos a través de los formularios, lo que previene uno de los ataques más comunes contra los sistemas web.

Estas medidas se revisaron de forma documental contra el listado de riesgos del OWASP Top 10:2021 [32], el catálogo de referencia más usado para evaluar la seguridad de aplicaciones web. La verificación de roles antes de cada operación sensible corresponde a la categoría A01 (control de acceso roto), el hash de contraseñas y la expiración y rotación de credenciales, a la A02 (fallos criptográficos); el uso de consultas parametrizadas, a la A03 (inyección); y el manejo de sesiones temporales con revocación del token de renovación, a la A07 (fallos de identificación y autenticación). Esta revisión se hizo de forma manual, comparando cada control ya implementado contra la descripción oficial de cada categoría, sin un escaneo automatizado de vulnerabilidades sobre el sistema en ejecución; queda como limitación reconocida que las demás categorías del listado (por ejemplo, fallos de configuración de seguridad o de registro y monitoreo) no se revisaron con el mismo nivel de detalle, por quedar fuera del alcance definido para este prototipo.

4.4.2. RNF de fiabilidad y disponibilidad

La fiabilidad del prototipo se evaluó desde dos ángulos. El primero es su comportamiento bajo carga sostenida: durante la prueba descrita en la sección 4.3 (25 usuarios concurrentes, 624 iteraciones en un minuto de régimen estable) el sistema mantuvo una tasa de error del 0% en todos los puntos medidos, sin peticiones rechazadas ni caídas. El segundo es su tolerancia ante la indisponibilidad de un servicio externo: cuando el Moodle local no responde (por un tiempo de espera agotado, un token inválido o el servicio caído), el sync-service devuelve un código de estado controlado (503) en lugar de propagar el fallo, y el resto del sistema sigue operando, porque las calificaciones residen en la base de datos propia del MRNA y no dependen de que Moodle esté disponible en ese momento. En cuanto a la disponibilidad, todo el entorno se levanta de forma reproducible mediante Docker Compose, lo que permite reiniciar o reconstruir el sistema completo de manera consistente.

Estos resultados son favorables, pero provienen de una prueba puntual de un minuto y de la ejecución de un único escenario de caída del servicio Moodle, no equivalen a una

observación sostenida en el tiempo, por lo que no permiten afirmar fiabilidad ni disponibilidad en el sentido estricto de esos atributos de calidad, que exigirían un período de operación más largo, idealmente en un entorno de producción.

4.4.3. RNF de mantenibilidad

Para que el sistema pueda corregirse y ampliarse sin romper lo que ya funciona, el prototipo se respalda con un conjunto de 150 pruebas unitarias sobre la lógica de negocio (cálculo de notas, validaciones académicas, emisión de credenciales de acceso) y seis escenarios de pruebas de integración que verifican el comportamiento conjunto de los módulos contra una base de datos real. Tanto las pruebas unitarias como las de integración se ejecutan de forma automática en cada cambio subido al repositorio, mediante el flujo de integración continua configurado en GitHub Actions, lo que evita que un error pase desapercibido. La cobertura se concentra en la lógica de negocio, que es la parte más expuesta a cambios. La medición formal de la complejidad del código y de la deuda técnica acumulada (por ejemplo, mediante una herramienta como SonarQube) no se incorporó dentro del alcance de este prototipo y se plantea como trabajo futuro en el Capítulo 5.

4.4.4. RNF de portabilidad

El sistema completo (la base de datos, el entorno de Moodle y los seis módulos que conforman el backend, junto con la interfaz web) está empaquetado en contenedores Docker para instalarse con un solo comando, sin necesidad de configurar cada pieza por separado, en principio, ese empaquetado no depende del sistema operativo de la máquina donde se instale, aunque la verificación efectiva de esta sección se realizó únicamente sobre Windows, sin pruebas ejecutadas en Linux ni macOS.

La medición de este proceso se realizó partiendo de un estado absolutamente limpio (sin imágenes, sin volúmenes y sin caché de construcción), con Docker reiniciado a su estado de fábrica para simular las condiciones de una instalación nueva. El levantamiento completo, desde que se ejecutó el comando hasta que todos los servicios respondieron con estado saludable, tomó 210 segundos (3 minutos y 30 segundos) en el equipo de desarrollo (procesador AMD Ryzen 7 5800H, 13,9 GB de RAM, Windows 11 Home, Docker Engine 29.4.3). Ese tiempo incluye la descarga de todas las imágenes base desde los repositorios públicos y la compilación completa de los seis microservicios; en instalaciones posteriores, cuando las imágenes ya estén descargadas, el tiempo se reduce

de forma considerable. Cabe aclarar que la medición se realizó en el mismo equipo de desarrollo (no en una máquina físicamente distinta), por lo que factores como la velocidad de conexión a Internet y las características del hardware influyen directamente en el resultado, el dato debe leerse como un orden de magnitud representativo, no como un valor absoluto independiente del entorno. La portabilidad hacia Linux y macOS, prevista por el uso de contenedores estándar, queda como verificación pendiente para un trabajo posterior.

4.4.5. RNF de usabilidad

El requerimiento de usabilidad (RNF-15) pedía que la interfaz fuera percibida como aceptable o mejor por quienes hoy gestionan las notas y la asistencia. Su cumplimiento se sustenta en la validación con el cuestionario SUS descrita en la sección 4.2. El prototipo obtuvo una puntuación de 87,5 sobre 100, por encima del umbral de 80 que marca una valoración excelente, y el evaluador completó las ocho tareas guiadas sin asistencia. Con ese resultado el RNF-15 se da por cumplido.

El requerimiento de accesibilidad (RNF-16) exigía el cumplimiento de los niveles A y AA de las Pautas de Accesibilidad para el Contenido Web (WCAG) 2.1. Sobre este punto no se ejecutó una auditoría formal, automatizada ni manual, que permita afirmar su cumplimiento; las decisiones de diseño de la interfaz se tomaron con criterios generales de usabilidad, pero sin una verificación específica contra las pautas WCAG. El RNF-16 se mantiene, por tanto, como pendiente, y su verificación queda documentada como trabajo futuro en el Capítulo 5.

Para cerrar la revisión de los requerimientos no funcionales, la Tabla 22 resume el veredicto de los dieciséis, contrastando cada criterio de aceptación con el resultado observado durante la validación.

Tabla 22. Cumplimiento de los requerimientos no funcionales del prototipo MRNA, por aspecto.

Nota. Elaboración propia, 2026.

Aspecto	Qué se buscaba	Qué se logró	Resultado
Seguridad (RNF-01 a 04)	Que solo entren usuarios identificados, que cada rol vea únicamente lo que le corresponde, que las contraseñas no se guarden legibles y que todo cambio de notas quede registrado	Acceso protegido por credenciales temporales, permisos por rol verificados en el servidor, contraseñas protegidas y una bitácora	Cumple

		automática de cada modificación	
Rendimiento (RNF-05 a 07)	Que consultar información y generar reportes sea rápido, incluso con varios usuarios a la vez	El reporte consolidado se genera en un segundo (RNF-07); la prueba de carga se ejecutó con 25 usuarios concurrentes y alcanzó 18,4 peticiones por segundo, por debajo de los 50 usuarios y las 100 peticiones/s que exigen RNF-05 y RNF-06	Cumple parcialmente (RNF-07 cumple; RNF-05 y RNF-06 no verificados bajo la condición definida)
Fiabilidad (RNF-08 y 09)	Que el sistema no se caiga ni muestre errores confusos, aunque falle un servicio externo como Moodle	Cero errores durante una prueba puntual de un minuto y una respuesta controlada ante un único escenario de caída de Moodle	Cumple parcialmente (evidencia favorable puntual, no concluyente sobre fiabilidad sostenida)
Disponibilidad (RNF-10)	Que se pueda instalar en cualquier computador con un solo paso	El sistema completo se levanta desde cero en tres minutos y medio con un único comando	Cumple
Mantenibilidad (RNF-11 a 13)	Que el código esté probado, documentado y sea fácil de mantener y ampliar	150 pruebas automáticas y documentación de la interfaz generada sola; la cobertura de pruebas todavía no alcanza el mínimo en todos los módulos	Cumple parcialmente
Portabilidad (RNF-14)	Que funcione en distintos sistemas operativos	Empaquetado en contenedores estándar; verificado únicamente en Windows, sin pruebas en Linux ni macOS	No verificado en Linux/macOS
Usabilidad (RNF-15)	Que resulte fácil de usar para quien gestiona las notas	Puntuación de usabilidad de 87,5 sobre 100 (excelente), con las ocho tareas guiadas completadas	Cumple

Accesibilidad (RNF-16)	Cumplimiento de los niveles A y AA de las pautas WCAG 2.1	No se ejecutó una auditoría de accesibilidad	Pendiente
-------------------------------	---	--	------------------

4.5. Resultados de la metodología Kanban

4.5.1. Métricas obtenidas

El seguimiento del proyecto se llevó a cabo en un tablero Kanban en GitHub Projects, del que se exportó el historial completo de tarjetas para calcular las métricas de flujo. El corte se fijó en el cierre de la Cadencia 5, que coincide con la fecha de cierre de la versión evaluada. Hasta ese punto, el tablero acumuló 79 tarjetas terminadas, de las cuales 70 pertenecen a alguna de las cinco cadencias y las 9 restantes corresponden a la configuración inicial del entorno, previa al arranque del desarrollo por cadencias.

La Tabla 23 reúne las tres métricas de flujo por cadencia. El throughput (la cantidad de tarjetas que llegaron a Terminado) fue de 12 en la primera cadencia, bajó a 10 en la segunda y luego subió hasta 17 y 20 en la tercera y la cuarta, para cerrar en 11 en la quinta. Los puntos entregados, que ponderan cada tarjeta por su tamaño en la escala Fibonacci del tablero, muestran un patrón distinto al del conteo simple, la tercera y la cuarta cadencia entregaron el mismo total, 73 puntos cada una, pese a que la cuarta registra tres tarjetas más. El lead time promedio (el tiempo que transcurre desde que una tarjeta entra al tablero hasta que se cierra) estuvo en alrededor de entre 6 y 14 días según la cadencia.

Tabla 23. Métricas de flujo Kanban del proyecto MRNA por cadencia. Nota. Elaboración propia a partir de la exportación del tablero "MRNA - Tesis Kanban" en GitHub Projects (corte al cierre de la Cadencia 5), 2026.

Cadencia	Tarjetas terminadas (throughput)	Puntos entregados	Lead time promedio (días)
Cadencia 1	12	39	9,0
Cadencia 2	10	28	14,2
Cadencia 3	17	73	8,7
Cadencia 4	20	73	6,5
Cadencia 5	11	45	8,1
Total	70 (+9 de setup = 79)	258	8,2 global (mediana 5,6)

Las tarjetas cerradas tuvieron un lead time promedio de 8 días y una mediana de 5 días, dentro de un rango que va desde las que se resolvieron el mismo día hasta la más larga,

de 27 días. La distancia entre el promedio y la mediana indica que la mayoría de las tarjetas se cerró con rapidez y que unas pocas tarjetas largas estiran el promedio hacia arriba. En cuanto al trabajo en curso, el tablero llegó a tener hasta 21 tarjetas simultáneas repartidas entre sus seis columnas, una cifra que a primera vista podría leerse como un incumplimiento del límite WIP de 3, pero no lo es, ya que, en Kanban, el límite de trabajo en curso se aplica únicamente a las columnas que representan transformación activa, en este tablero, En Desarrollo y En Pruebas. Las columnas restantes (Backlog, Sprint Actual, Terminado y Bloqueado) agrupan trabajo pendiente de iniciar o ya cerrado, no trabajo en curso, y por eso no llevan tope. Contadas solo las dos columnas de ejecución, el proyecto nunca superó las 3 tarjetas simultáneas por columna a lo largo de todo su desarrollo.

La Figura 17 presenta el Diagrama de Flujo Acumulado (CFD) del proyecto. En él, la banda verde representa las tarjetas ya terminadas que se acumulan con el tiempo, la banda azul agrupa el trabajo aún en curso o pendiente, y las marcas verticales señalan el inicio de cada cadencia. La forma de esas bandas es la base de la lectura del flujo que se desarrolla en la siguiente subsección.

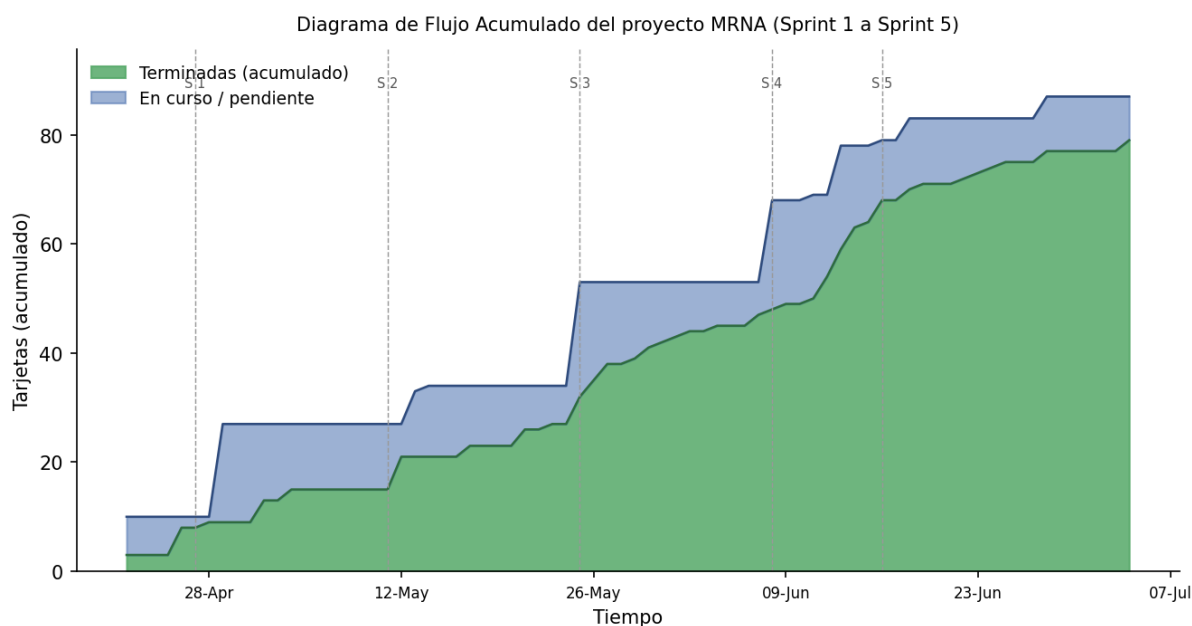


Figura 17. Diagrama de Flujo Acumulado (CFD) del proyecto MRNA, de la Cadencia 1 a la Cadencia 5. Nota. Elaboración propia a partir del tablero en GitHub Projects, 2026.

4.5.2. Análisis cuantitativo del flujo

La lectura de las métricas revela un flujo de trabajo que se estabiliza a medida que el proyecto avanza. Por número de tarjetas, el throughput dibuja una curva ascendente durante la mayor parte del desarrollo, parte de 12 en la primera cadencia, tiene una ligera

caída en la segunda (10) y crece hasta un máximo de 20 en la cuarta. Ese conteo, sin embargo, no basta para hablar de productividad, porque el tablero no exige tarjetas de tamaño uniforme (una tarjeta de 1 punto y una de 13 en la escala Fibonacci cuentan igual en ese número). Al ponderar por los puntos entregados (Tabla 23), la lectura se precisa, la tercera cadencia entregó 73 puntos con solo 17 tarjetas, el mismo total que la cuarta con 20, lo que indica que ambas tuvieron un rendimiento equivalente pese a que la cuarta parece ganar si solo se cuentan tarjetas. El descenso de la quinta cadencia (11 tarjetas, 45 puntos) no responde a una pérdida de ritmo, coincide con el tramo dedicado al trabajo de frontend y a tareas de cierre, donde las tarjetas individuales son más grandes y, en consecuencia, menos numerosas.

El comportamiento del lead time acompaña esa evolución. El valor más alto se registra en la segunda cadencia (14,2 días), a partir de ahí el indicador desciende hasta su mejor registro en la cuarta cadencia (6,5 días) y se mantiene en un nivel sano en la quinta (8,1 días). Descontado ese pico de la segunda cadencia, el tiempo que una tarjeta tardaba en atravesar el tablero tendió a acortarse conforme avanzó el proyecto.

La distribución global refuerza esa lectura. Con un promedio de 8,2 días frente a una mediana de 5,6, la mayoría de las tarjetas se cerró en menos de una semana, y solo unos pocos casos aislados elevan el promedio. Esa asimetría es habitual en un proyecto llevado por una sola persona, donde algunas tarjetas quedan a la espera de una decisión, de una dependencia externa o de la redacción de su documentación asociada, mientras el resto avanza con fluidez.

El Diagrama de Flujo Acumulado (Figura 17) confirma que no hubo cuellos de botella crónicos. La banda verde de tarjetas terminadas crece de manera continua, y la banda azul de trabajo en curso se ensancha cuando se planifica un lote de tarjetas y vuelve a estrecharse a medida que se completan, sin acumularse de forma indefinida. Este patrón es el que la literatura sobre Kanban asocia a un flujo saludable: al respetar los límites de trabajo en curso de las columnas de ejecución (tres tarjetas en En Desarrollo y tres en En Pruebas), el sistema evita saturarse y mantiene acotado el tiempo de entrega [18]. La columna de pruebas, que suele ser un punto de congestión frecuente en esta clase de tableros, no llegó a saturarse en ningún momento del proyecto [17].

Por último, conviene precisar el alcance temporal de estas cifras, cubren hasta el cierre de la Cadencia 5, que es el corte de documentación de la tesis. Las cadencias posteriores del cronograma (se tiene contemplado hasta una séptima) quedan fuera de esta medición,

de modo que las métricas describen el flujo del proyecto hasta el punto en que se congeló su documentación, no hasta el final absoluto del desarrollo.

4.6. Discusión de resultados

El objetivo general de este trabajo era construir un prototipo, basado en microservicios, capaz de reemplazar el registro manual de notas y asistencia del ISTA y mejorar su eficiencia. Los resultados presentados a lo largo del capítulo permiten sostener que ese objetivo se considera alcanzado dentro de las condiciones evaluadas, y conviene revisarlo en base a los cuatro objetivos específicos que lo articulaban. El primero, levantar y documentar los requerimientos, se cumplió en el diagnóstico del Capítulo 2, donde se catalogaron veintitrés requerimientos funcionales (dieciocho de ellos implementables) y dieciséis no funcionales. El segundo, diseñar la arquitectura, se materializó en los seis microservicios, el api-gateway y el esquema de treinta y seis tablas descritos en el Capítulo 3. El tercero, desarrollar el prototipo funcional, se concretó en los servicios centrales del flujo de notas, asistencia y gestión de usuarios docentes, respaldados por 150 pruebas unitarias y pruebas de integración automatizadas, el grado de avance de cada requerimiento, con las funciones que llegaron completas y las que quedaron parciales o en diseño al cierre de la Cadencia 5, se detalla en la sección 2.7. El cuarto, validar el prototipo, es el que ocupa este capítulo: se midieron el desempeño, los tiempos y la tasa de errores, y se evaluó la usabilidad con el cuestionario SUS.

El hallazgo más favorable es la reducción del tiempo administrativo. El proceso manual actual, según la estimación de entrevistas con docentes, recogida en el diagnóstico del apartado 2.8, consume entre seis y ocho horas por la triple digitación entre Moodle, Excel y el SIGA. En el prototipo, el procesamiento de las notas de un parcial completo tarda pocos segundos (2,9s) y la generación de un reporte consolidado, un segundo, lo que representa una reducción potencial superior al 99%. Aunque ya se mencionó que los tiempos manuales provienen de estimaciones de las entrevistas y no de una cronometría directa (una excepción que ya se declaró en la sección 4.3), la diferencia entre ambos es lo bastante amplia como para reducir considerablemente la incertidumbre, dentro de las condiciones de prueba descritas.

Estos datos responden directamente a la primera pregunta de investigación, sobre en qué medida un prototipo de microservicios permite automatizar el flujo y reducir el tiempo administrativo. La respuesta es que lo consigue en un grado alto. El flujo que antes exigía

transcribir manualmente las mismas calificaciones en tres sistemas distintos queda reducido a una sincronización desde Moodle y una carga validada hacia el SIGA. La automatización acorta el tiempo, y además, elimina la fuente principal de errores del proceso actual, que es la reescritura repetida de los mismos datos.

La segunda pregunta apuntaba a las decisiones arquitectónicas que hacen posible la integración con Moodle y dejan preparada la futura integración con el SIGA. Tres decisiones resultaron determinantes. La primera es la integración con Moodle a través de su API REST con una sincronización que no varía, anclada en el identificador de actividad de Moodle que guarda cada nota, de modo que reprocesar una sincronización no duplica registros. La segunda, y la más importante frente a la restricción documentada en el Capítulo 1, es el adaptador dual para el SIGA, como el sistema nacional no ofrece una API pública, el prototipo trabaja hoy en modo archivo (genera los formatos que se cargan manualmente) y deja reservado, a nivel de configuración, un modo de API para una futura integración directa; ese modo no está conectado a ningún flujo ejecutable todavía, de manera que el día en que la SENESCYT habilite un punto de integración real, el trabajo pendiente sería implementarlo y conectarlo, no apenas sustituirlo. La tercera es la separación en servicios independientes tras el api-gateway, que permite evolucionar cada integración de forma aislada. En conjunto, estas decisiones responden que sí es posible una integración funcional con Moodle y que el prototipo deja prevista, aunque no construida, la vía para una integración futura con el SIGA.

Situados frente a los trabajos relacionados revisados en el Capítulo 1, los resultados coinciden con la evidencia previa en lo esencial. Al igual que en las experiencias de integración de Moodle con sistemas institucionales mediante su API REST [10], [11], [22], el prototipo confirma que trasladar automáticamente las calificaciones desde el aula virtual suprime la doble digitación y reduce tanto los errores como los tiempos. Y en la misma línea que los estudios sobre microservicios en entornos educativos [23], [24], la arquitectura demostró ser viable y estable para este dominio, con tiempos de respuesta bajos y sin errores bajo carga.

La diferencia, y el aporte del MRNA, aparece justo donde esos antecedentes no llegan. Los trabajos de integración citados asumen un sistema institucional accesible, ya sea porque exponen su propia API o porque comparten una base de datos a la cual se puede acceder, mientras que el escenario del ISTA es distinto, porque el sistema de destino es una plataforma nacional que no expone ninguna interfaz de integración, y es precisamente esa restricción la que el adaptador dual resuelve. Además, mientras esas propuestas se

concentran en el flujo de calificaciones de un solo módulo, el MRNA gestiona de forma conjunta las notas y la asistencia, y lo hace bajo un esquema de cuatro roles institucionales con permisos diferenciados. El espacio que el Capítulo 1 identificó como no cubierto por la literatura (la integración con un sistema nacional sin API pública, la gestión conjunta de notas y asistencia, y el soporte a múltiples roles) es el que este prototipo ocupa.

En el plano práctico, el ISTA dispone de un prototipo que elimina la triple digitación que hoy recae sobre docentes y coordinadores, deja trazabilidad automática de cada cambio de nota o asistencia mediante la auditoría por triggers, y fue percibido como de usabilidad excelente por el perfil experto que lo evaluó. Esto lo posiciona como una base sólida para una eventual implementación productiva, con las salvedades ya señaladas en la sección 4.7: la validación se apoyó en un único evaluador experto y se ejecutó sobre una instancia local de Moodle, por lo que el paso a producción exigirá una prueba con usuarios reales y con la plataforma institucional. Con estas consideraciones, la evidencia reunida sostiene que la propuesta responde a la problemática planteada. El MRNA automatiza un proceso manual costoso y, al mismo tiempo, respeta las restricciones normativas y técnicas propias del Instituto, documentadas a partir de las entrevistas realizadas al inicio de este proyecto.

4.7. Limitaciones del estudio

Los resultados presentados en este capítulo deben leerse dentro del alcance que esta tesis se propuso desde el anteproyecto, y no como una validación en condiciones de producción. El MRNA es un prototipo funcional construido para demostrar la viabilidad técnica de automatizar el registro y la consolidación de notas entre Moodle y el MRNA, y de generar el archivo listo para la carga manual en el SIGA, no un sistema desplegado en el ISTA con usuarios reales operando sobre datos académicos vigentes. Esa distinción condiciona la lectura de cada uno de los apartados anteriores.

La validación se ejecutó sobre una instancia local de Moodle desplegada en Docker, porque no existe acceso ni credenciales al Moodle institucional del ISTA. Los resultados de sincronización reportados en este capítulo reflejan, entonces, el comportamiento del adaptador frente a una instancia de prueba con la misma versión y estructura, pero no descartan que la plataforma institucional, con su propia carga de datos y configuración, introduzca particularidades no cubiertas aquí. De manera análoga, el módulo de salida hacia SIGA opera hoy únicamente en modo archivo, porque SIGA no expone una API

pública (Responsable de TIC, comunicación personal, 2025); el modo API queda reservado en el diseño del adaptador para una integración futura, pero no está implementado ni conectado a ningún flujo ejecutable en este prototipo. Lo que este capítulo reporta sobre esa integración corresponde, entonces, a la generación y estructura del archivo de carga, no a una transacción confirmada contra el sistema nacional real ni a una simulación operativa del modo API.

La validación de usabilidad presentada en 4.2 se apoya en un solo evaluador experto, el Docente de Desarrollo de Software, y no en una muestra de usuarios finales del ISTA. El puntaje SUS resultante describe entonces la valoración de un perfil técnico familiarizado con sistemas de gestión académica, y no necesariamente la experiencia de un docente con menor afinidad tecnológica enfrentando el sistema por primera vez. De forma similar, las métricas de flujo Kanban reportadas en 4.5 provienen de un proceso ejecutado por un solo desarrollador a lo largo de las cinco cadencias que cubre el corte de documentación, por lo que no recogen la variabilidad de coordinación, bloqueos cruzados ni reparto de carga que aparecería en un equipo de desarrollo real.

De la misma manera, la validación funcional del flujo de consolidación prevista con la coordinación académica (Tabla 6) no llegó a ejecutarse antes del corte de documentación, por las mismas restricciones de agenda que limitaron la fase de usabilidad. La verificación de ese flujo se apoyó, por tanto, en las pruebas técnicas y automatizadas descritas en las secciones 4.3 y 4.4, y la participación de un integrante de coordinación queda como tarea pendiente para una extensión del estudio.

Finalmente, de los veintitrés requerimientos funcionales levantados durante el diagnóstico, cinco quedaron fuera de este alcance y se documentan como trabajo futuro en el Capítulo 5: la firma electrónica vía Quipux, la ayuda interactiva dentro del sistema, el portal de noticias, el chat docente-estudiante y la sincronización con calendarios externos como Google Calendar u Outlook. Su ausencia no compromete el objetivo central de la tesis, centrado en la automatización del registro y la consolidación de notas y asistencia entre Moodle y el MRNA, y en la generación del archivo listo para la carga manual en el SIGA, pero delimita lo que el prototipo evaluado en este capítulo puede y no puede demostrar.

El prototipo documentado y evaluado en esta tesis corresponde a su estado al cierre de la Cadencia 5 (5 de julio de 2026), que constituye el corte de documentación del trabajo. El desarrollo del software continúa más allá de ese corte, por lo que algunas funcionalidades que aquí se registran como trabajo futuro (por ejemplo, el tablero de indicadores para el

rol directivo) pueden encontrarse ya implementadas al momento de la defensa de este proyecto, sin que ello altere los resultados reportados, obtenidos sobre el estado documentado.

4.8. Síntesis del capítulo

Este capítulo reunió la evidencia que respalda la propuesta. La validación de usabilidad situó al prototipo en el rango de excelente, con una puntuación SUS de 87,5 y las ocho tareas guiadas completadas sin asistencia. La medición de desempeño mostró que el sistema procesa las notas de un parcial en segundos y genera reportes consolidados en un segundo, frente a las seis a ocho horas del proceso manual, con una tasa de error casi nula. La revisión de los requerimientos no funcionales confirmó el cumplimiento de la mayoría y señaló con transparencia los que quedaron a medias, como la cobertura de pruebas en algunos servicios y la accesibilidad. Las métricas del tablero Kanban, por su parte, describieron un flujo de trabajo saludable, sin cuellos de botella crónicos, a lo largo de las cadencias que se valoraron.

En conjunto, estos resultados sostienen que el prototipo resuelve la problemática que motivó el trabajo y responden a las preguntas de investigación planteadas en el diagnóstico. El capítulo siguiente recoge esas evidencias para formular las conclusiones por cada objetivo específico, exponer los aportes del trabajo y plantear las recomendaciones y las líneas de trabajo futuro que abre el prototipo.

Capítulo 5. Conclusiones y Recomendaciones

5.1. Conclusiones por objetivo específico

Las conclusiones se presentan siguiendo el orden de los cuatro objetivos específicos que guiaron el trabajo.

El primer objetivo planteaba levantar y documentar los requerimientos del sistema. Para cumplirlo se caracterizó el proceso manual vigente y esa realidad se tradujo en un catálogo de veintitrés requerimientos funcionales y dieciséis no funcionales, con su matriz de trazabilidad hacia las entrevistas y las historias de usuario. Del mismo levantamiento salió un hallazgo que terminó condicionando todo el diseño posterior, y es que, el SIGA no ofrece una vía pública de integración. El diagnóstico de esto, fue lo que definió las restricciones reales sobre las que se construyó la solución.

En cuanto al segundo objetivo, que consistía en diseñar la arquitectura del sistema, se especificó una solución de seis microservicios coordinados por un api-gateway y sostenidos por un esquema de treinta y seis tablas en PostgreSQL. El diseño incorporó decisiones pensadas para las condiciones concretas del Instituto, como el adaptador dual para el SIGA y la sincronización única con Moodle. El alcance técnico y los límites de cada servicio se justifican detalladamente en la sección 3.6 (identidad, núcleo académico, integración externa, reportes, notificaciones y enrutamiento), en lugar de hacer una división al azar. Cabe aclarar que este trabajo no comparó formalmente el costo o la complejidad de esta propuesta frente a otras más sencillas. Sin embargo, con los datos actuales podemos afirmar que el número de servicios se ajusta a las necesidades reales del proyecto, evitando partir el sistema en pedazos innecesarios.

El tercer objetivo buscaba desarrollar el prototipo funcional. Se construyeron los servicios centrales del flujo de notas y asistencia, la sincronización con Moodle, la gestión de usuarios docentes y la generación de reportes, todo respaldado por 150 pruebas unitarias y pruebas de integración que se ejecutan de forma automática ante cada cambio subido al repositorio. Con ello se confirmó que el diseño era realizable y que el núcleo del prototipo funciona de extremo a extremo, desde la entrada de las calificaciones hasta su reporte consolidado en Excel o CSV. Quedan fuera de ese núcleo, con distinto grado de avance al cierre de la Cadencia 5, el dashboard de indicadores (pendiente) y la exportación de reportes en PDF junto con las notificaciones automáticas y programadas (parciales), tal como se detalla en la sección 2.7.

Sobre el cuarto objetivo, que apuntaba a validar el prototipo, la evaluación combinó la medición del desempeño con la valoración de la usabilidad. En el desempeño, el sistema procesó las notas de un parcial en 2,9 segundos y generó reportes consolidados en pocos segundos, con una tasa de error del 0% bajo carga (sección 4.3), frente a las seis a ocho horas del proceso manual. En la usabilidad, el prototipo obtuvo una puntuación SUS de 87,5 sobre 100, dentro del rango de excelente, con las ocho tareas guiadas completadas por el evaluador (sección 4.2.3). Conviene leer este segundo resultado dentro de su alcance real: proviene de un único evaluador experto, el Docente de Desarrollo de Software, y describe su percepción individual del sistema, no la de una muestra representativa de los cuatro roles institucionales que en algún momento usarían el prototipo. La accesibilidad, prevista también en este objetivo, no contó con una auditoría formal contra las pautas WCAG, tal como se detalla en la sección 4.4, y queda documentada como trabajo pendiente. Los datos disponibles muestran una reducción potencial del tiempo administrativo superior al 99% bajo las condiciones de prueba descritas en la sección 4.3, y confirman que el prototipo resultó apto para el perfil que lo evaluó

5.2. Aportes del trabajo

Los aportes de este trabajo se aprecian en tres planos.

El primero es el aporte técnico. El proyecto demuestra, con un sistema funcional y no solo con argumentos teóricos, que la arquitectura de microservicios es adecuada para la gestión de notas y asistencia del Instituto. El punto más valioso es cómo se resolvió una restricción real, el sistema nacional al que hay que reportar, el SIGA, no permite que otros programas se conecten a él de forma directa. En lugar de detener el proyecto, esa restricción se resolvió con una solución que hoy entrega los archivos para cargarlos manualmente y que, si se implementa y conecta el modo de API que el diseño ya reserva para ese escenario, podrá operar de forma automática cuando el SIGA lo habilite. Esta forma de resolver la restricción puede servirle a cualquier otro instituto del país que enfrente la misma limitación.

El segundo es el aporte metodológico, es decir, la manera de organizar el trabajo. El proyecto deja documentado, de principio a fin, cómo una sola persona gestionó un proyecto de titulación con un método de gestión ordenado y visual (Kanban) y, sobre todo, cómo lo midió con datos en lugar de con la simple impresión de ir avanzando:

cuántas tareas se cerraban en cada tramo, cuánto tardaba cada una y cuánto trabajo se llevaba a la vez (sección 4.5). Ese registro sirve de ejemplo replicable para futuros estudiantes que quieran llevar su proyecto con orden y con evidencia de su propio avance. El tercero es el aporte para la institución. El Instituto recibe una herramienta que ya funciona y que ataca directamente el problema que originó el trabajo (tener que escribir tres veces las mismas notas en sistemas distintos). Además de ahorrar ese tiempo, el sistema guarda por sí solo un registro de cada cambio en las notas y la asistencia, lo que da claridad sobre quién modificó qué y cuándo, y fue calificado como muy fácil de usar por la persona experta que lo evaluó. Con esto, el ISTA no parte de cero, sino que tiene una base concreta sobre la cual decidir si se lo lleva a un uso real y cotidiano.

5.3. Recomendaciones

Las recomendaciones se dirigen a dos destinatarios: el Instituto y los futuros estudiantes de la carrera.

Al Instituto Superior Tecnológico del Austro, si decide llevar el prototipo a un uso real, se le recomienda avanzar de forma gradual. Conviene empezar por una prueba con usuarios reales, con docentes y coordinadores usando el sistema durante un período acotado antes de un despliegue general, para recoger ajustes a partir del uso cotidiano. En paralelo, es aconsejable reforzar la infraestructura donde funcionaría el sistema (un servidor institucional con respaldos periódicos de la información) y definir con claridad quién administra las cuentas y los permisos. También resulta recomendable acompañar la puesta en marcha con una capacitación breve a los distintos roles del sistema. Antes de esos pasos conviene atender primero las mejoras que el propio evaluador señaló durante la validación, como la gestión de una persona con más de un rol y la organización de la información por período, carrera y nivel.

A los futuros estudiantes de la carrera se les recomiendan dos cosas que este trabajo dejó en claro. La primera es dedicar tiempo real al diagnóstico antes de empezar a programar, entender el proceso que se quiere mejorar y sus restricciones (en este caso, la imposibilidad de conectarse al sistema nacional) evita rehacer el trabajo más adelante. La segunda es conducir el proyecto con un método que permita medir el avance con datos y no con la sensación de estar adelantado, porque así se detecta a tiempo si algo se está estancando y se sostiene un ritmo parejo hasta el final.

5.4. Trabajo futuro

El prototipo que esta tesis documenta corresponde a su estado en el corte de documentación (el cierre de la Cadencia 5), pero el desarrollo del software no se detuvo ahí. Por eso conviene ordenar el trabajo futuro en dos horizontes, el primero, lo que se completa en las cadencias finales del propio desarrollo, que estaba pendiente al momento del corte, y después, las extensiones que van más allá del alcance que este trabajo se propuso.

En el plan contemplado del prototipo que ya se venían construyendo cuando se congeló la documentación y que se terminan en el cierre del desarrollo: la carga masiva completa del adaptador SIGA, el tablero de indicadores para el nivel directivo, la ventana de gracia de cuarenta y ocho horas para los ajustes de la junta académica, las notificaciones programadas y las de ausentismo, y el reporte consolidado en PDF con formato institucional. A ellas se suman las mejoras que surgieron directamente de la validación con el evaluador experto: la gestión de las mallas curriculares por carrera, la asignación de varios roles a un mismo usuario (porque una persona puede ejercer, por ejemplo, como docente y como coordinador a la vez) y la recuperación de contraseña desde la propia interfaz. Que estas mejoras nazcan de la retroalimentación del experto confirma el valor de haber validado el sistema con quien conoce el proceso real.

Más allá del prototipo quedan cinco requerimientos que se levantaron durante el diagnóstico pero se dejaron fuera del alcance por no ser críticos para el flujo central de notas y asistencia: la firma electrónica a través de Quipux, la ayuda interactiva dentro del sistema, el portal de noticias, el chat entre docente y estudiante, y la sincronización con calendarios externos como Google Calendar u Outlook. Su ausencia no compromete el objetivo central de la tesis.

La línea más importante para llevar el sistema a un uso real es la integración efectiva con el SIGA. Hoy se resuelve únicamente en modo archivo, precisamente porque el sistema nacional no expone una interfaz pública, el adaptador deja reservado un modo de API para ese escenario, pero implementarlo y conectarlo queda como trabajo futuro, no como algo ya construido. El día en que la SENESCYT habilite una API oficial, esa carga podría volverse directa y en línea, sin el paso intermedio del archivo. En paralelo, un despliegue en producción necesitaría resolver la orquestación de los seis contenedores del sistema fuera de Docker Compose, que hoy alcanza para el entorno de desarrollo, pero no ofrece alta disponibilidad ni escalado automático. Kubernetes es una alternativa posible para ese

escenario, aunque su adopción solo se justifica si el volumen real de usuarios y la infraestructura disponible en el ISTA lo requieren. Este trabajo no midió esa demanda ni comparó el costo de operar Kubernetes frente a alternativas más simples, como un segundo servidor con Docker Compose replicado. Por eso queda planteado como una opción a evaluar, no como una recomendación cerrada, y esa evaluación debería considerar cuántos usuarios concurrentes tendría el sistema en producción, con qué infraestructura cuenta el Instituto y qué esfuerzo de mantenimiento puede asumir el equipo técnico del ISTA. Sumar herramientas de observabilidad (como Prometheus y Grafana) sí es recomendable en cualquier escenario de producción, independientemente de si se adopta Kubernetes u otra forma de orquestación, porque permiten vigilar el sistema mientras opera con usuarios y datos reales.

En el contexto de la arquitectura, la decisión de que todos los servicios compartan una sola base de datos fue razonable para un prototipo, pero un sistema en producción se beneficiaría de migrar poco a poco hacia una base de datos por servicio, de modo que cada microservicio sea realmente independiente de los demás. Sobre esa base más madura, los datos académicos que el sistema acumula (calificaciones por componente, asistencia, historial de matrícula y alertas ya generadas por bajo rendimiento) podrían sustentar un modelo de predicción temprana de estudiantes en riesgo de perder una materia. Desarrollar ese modelo exigiría definir con precisión las variables de entrada, validar su desempeño contra casos históricos reales antes de usarlo en producción, y establecer salvaguardas éticas claras. Entre ellas, que la predicción funcione como una alerta que un docente o coordinador revisa y decide cómo atender, no como una decisión automática que etiquete a un estudiante sin intervención humana, y que el acceso a esos datos sensibles quede restringido a los roles que efectivamente lo necesiten.

Por último, la propia validación puede crecer. El resultado reportado en este trabajo proviene de un único evaluador experto, una evaluación futura podría incorporar a docentes con distintos niveles de familiaridad con la tecnología y a los demás roles del sistema, y ejecutarse sobre datos reales una vez el sistema esté desplegado, para con ello reforzar todavía más las conclusiones sobre su usabilidad.

Referencias Bibliográficas

- [1] Continental University, «Sistema de gestión académica: clave para la educación moderna». octubre de 2025. [En línea]. Disponible en: <https://surl.lt/qsjksm>
- [2] Asamblea Nacional del Ecuador, «Ley Orgánica de Educación Superior», Quito, Ecuador, ene. 2024. Accedido: 29 de abril de 2026. [En línea]. Disponible en: <https://surl.li/weyvrv>
- [3] Consejo de Educación Superior, «Reglamento de Régimen Académico», Quito, Ecuador, RPC-SE-08-No.023-2022, sep. 2022. [En línea]. Disponible en: <https://surl.li/rwrjlb>
- [4] Secretaría de Educación Superior, Ciencia, Tecnología e Innovación, «Sistema Integrado de Gestión Académica (SIGA)». Gobierno del Ecuador, Quito, Ecuador, 2025. Accedido: 29 de junio de 2026. [En línea]. Disponible en: <http://siga.institutos.gob.ec>
- [5] L. Sanchez, J. Penarreta, y X. Soria Poma, «Learning management systems for higher education: a brief comparison», vol. 3, Discover Education, 2024. doi: 10.1007/s44217-024-00143-5.
- [6] Moodle, «Using web services». 2024.
- [7] S. Newman, *Building Microservices: Designing Fine-Grained Systems*, 2.^a. O'Reilly Media, 2021.
- [8] Y. M. Abgaz, A. McCarren, P. Elger, D. Solan, y N. Lapuz, «Decomposition of monolith applications into microservices architectures: A systematic review», *IEEE Trans. Softw. Eng.*, vol. 49, n.º 8, pp. 4213-4242, 2023, doi: 10.1109/TSE.2023.3287297.
- [9] J. Bogner, S. Kotstein, y T. Pfaff, «Do RESTful API design rules have an impact on the understandability of Web APIs? A web-based experiment with API descriptions», *Empir. Softw. Eng.*, vol. 28, n.º 6, p. 132, nov. 2023, doi: 10.1007/s10664-023-10367-y.
- [10] L. Magdaleno Horta, R. J. Muñoz Mujica, y J. Martínez Patiño, «Sistema de información para la educación digital: percepción de usuarios en la Universidad de Guanajuato», *Acta Univ.*, vol. 34, 2024, doi: 10.15174/au.2024.4007.
- [11] A. Prasetyo y Y. Saintika, «Integration between Moodle and Academic Information System using Restful API for Online Learning», vol. 7, n.º 2, pp. 358-367, 2021, doi: 10.26555/jiteki.v7i2.21816.

- [12] W. Chingo Esquivel y G. López Sevilla, «Paralelismos entre bases de datos relacionales y no relacionales (un enfoque en seguridad)», n.º 10, pp. C1-16, 2021, doi: 10.32870/recibe.v10i2.189.
- [13] PostgreSQL Global Development Group, «PostgreSQL 15 Documentation». 2023. [En línea]. Disponible en: <https://www.postgresql.org/docs/15/>
- [14] Z. Brahmia, F. Grandi, y B. Oliboni, «A Literature Review on Schema Evolution in Databases», *Comput. Open*, vol. 02, p. 2430001, ene. 2024, doi: 10.1142/S2972370124300012.
- [15] A. Bucko, K. Vishi, B. Krasniqi, y B. Rexha, «Enhancing JWT authentication and authorization in web applications based on user behavior history», *Computers*, vol. 12, n.º 4, 2023, doi: 10.3390/computers12040078.
- [16] A. Venčkauskas, D. Kukta, S. Grigaliūnas, y R. Brūzgienė, «Enhancing microservices security with token-based access control method», *Sensors*, vol. 23, n.º 6, 2023, doi: 10.3390/s23063363.
- [17] A. R. León Yacelga y M. A. Checa Cabrera, «Uso de tableros Kanban como apoyo para el desarrollo de las metodologías ágiles», *Univ. Soc.*, vol. 14, n.º S2, pp. 208-214, 2022.
- [18] J. P. Jiménez Gonzales, E. A. López Pazmiño, D. A. Chávez Marcillo, y D. W. Cevallos Álava, «Kanban: una metodología ágil para la gestión eficiente del flujo de trabajo en el desarrollo de software, una revisión sistemática», *Rev. Ingenio Glob.*, vol. 4, n.º 2, may 2024, [En línea]. Disponible en: <https://editorialinnova.com/index.php/rig/article/view/68>
- [19] M. Hui, L. Wang, H. Li, R. Yang, y Y. Song, «Unveiling the microservices testing methods, challenges, solutions, and solutions gaps: A systematic mapping study», *J. Syst. Softw.*, vol. 220, 2025, doi: 10.1016/j.jss.2024.112232.
- [20] A. S. Abdelfattah, T. Cerny, J. Yero, E. Song, y D. Taibi, «Test Coverage in Microservice Systems: An Automated Approach to E2E and API Test Coverage Metrics», *Electronics*, vol. 13, p. 1913, 2024, doi: 10.3390/electronics13101913.
- [21] C. Castillo y O. Castellanos, «Empirical Evaluation of MVC Frameworks: Performance, Scalability, Developer Experience and Learning Curvex», en *Advanced Research in Technologies, Information, Innovation and Sustainability (ARTIIS 2025)*, en Communications in Computer and Information Science, vol. 2792. Springer, Cham, 2026, pp. 343-358. doi: 10.1007/978-3-032-16851-1_24.

- [22] R. Sahara, S. Abdullah, y M. I. Saputra, «Integration model of academic information systems and learning management systems with REST web services using external databases», vol. 47, 2023, [En línea]. Disponible en: <https://jios.foi.hr/index.php/jios/article/view/1912>
- [23] D. E. Kurniawan y A. Dzikri, «Development of a prototype for microservices architecture and API gateway integration in an eLearning platform», *EAI*, 2024. doi: 10.4108/eai.7-11-2023.2342930.
- [24] C. Muhdiantini, S. Z. Amril, D. Prayoga, A. Nurdin, y M. Lubis, «Optimizing learning environments: A literature review on microservices for Moodle quality improvement», *JIPi J. Ilm. Penelit. Dan Pembelajaran Inform.*, vol. 9, pp. 1285-1291, 2024, doi: 10.29100/jipi.v9i3.5263.
- [25] Instituto Superior Tecnológico del Austro, «Plan Estratégico de Desarrollo Institucional 2021-2026», Azogues, Ecuador, Resolución 084-OCS-09-ISTAUSTRO-2024, sep. 2024. [En línea]. Disponible en: <https://surl.li/mnlhkt>
- [26] L. M. Benavides, J. A. Tamayo Arias, M. D. Arango Serna, J. W. Branch Bedoya, y D. Burgos, «Digital Transformation in Higher Education Institutions: A Systematic Literature Review», *Sensors*, vol. 20, n.º 11, p. 3291, jun. 2020.
- [27] S. Weerasinghe y I. Perera, «An exploratory evaluation of replacing ESB with microservices in service oriented architecture», en *2021 International Research Conference on Smart Computing and Systems Engineering (SCSE)*, Kelaniya, Sri Lanka: IEEE, sep. 2021. doi: 10.1109/SCSE53661.2021.9568289.
- [28] Redgate, «Implementing a roll back strategy», Redgate Flyway Documentation. Accedido: 19 de junio de 2026. [En línea]. Disponible en: <https://documentation.red-gate.com/fd/implementing-a-roll-back-strategy-138347142.html>
- [29] Tailwind Labs, «Tailwind CSS Documentation», Tailwind CSS. Accedido: 19 de junio de 2026. [En línea]. Disponible en: <https://tailwindcss.com/docs>
- [30] shadcn, «shadcn/ui Documentation», shadcn/ui. Accedido: 19 de junio de 2026. [En línea]. Disponible en: <https://ui.shadcn.com/docs>
- [31] Y. Irawan, F. B. Suryani, S. Wanabuliandari, y S. Muzid, «System Usability Scale (SUS) Model in Evaluating Internal Quality Audit Systems for Accreditation Process Optimization», *J. Appl. Inform. Comput.*, vol. 9, n.º 2, pp. 511-516, mar. 2025, doi: 10.30871/jaic.v9i2.9210.
- [32] OWASP Foundation, «OWASP Top 10:2021», OWASP Foundation. Accedido: 29 de junio de 2026. [En línea]. Disponible en: <https://owasp.org/Top10/2021/>

ANEXOS

Anexo A. Instrumentos de Investigación de Campo

Guion de entrevistas estructuradas para cada rol del ISTA

- **Formulario para docentes**

1. ¿Cómo es su proceso actual para registrar calificaciones y asistencia?
2. ¿Cuánto tiempo dedica semanalmente a estas tareas administrativas?
3. ¿Qué aspectos del proceso actual encuentra más problemáticos o tediosos?
4. ¿Qué funcionalidades facilitarían significativamente su trabajo?
5. ¿Qué reportes o análisis necesita generar con frecuencia?
6. ¿Cómo le gustaría que funcionara el sistema para simplificar su trabajo?
7. ¿Qué experiencia tiene utilizando Moodle para la gestión de calificaciones?
8. ¿Qué experiencia tiene utilizando SIGA para registro de asistencia?
9. ¿Qué formato de presentación de información preferiría para monitorear el rendimiento de sus estudiantes?
10. ¿Qué notificaciones automáticas serían útiles en su labor docente?
11. ¿Cómo le gustaría poder comunicar a los estudiantes sus avances académicos?
12. ¿Qué otros procesos relacionados con la evaluación académica le gustaría automatizar?

- **Formulario para personal de TI**

1. ¿Cuál es la arquitectura actual de los sistemas SIGA y Moodle?
2. ¿Qué versiones específicas de software se están utilizando?
3. ¿Qué APIs o interfaces están disponibles para la integración?
4. ¿Cuáles son las principales limitaciones técnicas para la integración?
5. ¿Qué consideraciones de seguridad y acceso a datos deben tenerse en cuenta?
6. ¿Qué infraestructura tecnológica soportará el sistema integrado?
7. ¿Cómo se gestionan actualmente las bases de datos de ambos sistemas?
8. ¿Existe documentación técnica disponible de los sistemas actuales?
9. ¿Qué procesos de respaldo y recuperación están implementados?
10. ¿Qué recomendaciones técnicas tiene para la implementación?
11. ¿Cuáles son los picos de demanda o carga en los sistemas actuales?
12. ¿Qué consideraciones de escalabilidad deben tenerse en cuenta?

- **Formulario para coordinadores**

1. ¿Cómo se administran actualmente las calificaciones y asistencia en su coordinación?
2. ¿Qué tiempo aproximado dedican usted y su equipo a gestionar estos procesos manualmente?
3. ¿Qué informes o consultas específicas sobre rendimiento académico realiza con frecuencia?
4. ¿Qué problemas o limitaciones encuentra con los sistemas actuales (SIGA y Moodle)?
5. ¿Qué información necesita para monitorear el avance académico de los estudiantes?
6. ¿Qué requisitos específicos de su carrera deberían considerarse en el sistema?
7. ¿Qué datos necesita visualizar para la toma de decisiones a nivel de coordinación?
8. ¿Cómo realiza actualmente el seguimiento de estudiantes en riesgo académico?
9. ¿Qué procesos de su coordinación son prioritarios para automatizar?
10. ¿Qué información estadística utiliza para evaluaciones internas o procesos de acreditación?

- **Formulario para directivos**

1. ¿Cuáles son los principales desafíos que enfrenta actualmente la institución en la gestión de información académica?
2. ¿Qué tipos de informes o dashboards necesita para la toma de decisiones estratégicas?
3. ¿Qué indicadores académicos considera cruciales para monitorear el desempeño institucional?
4. ¿Qué procesos actuales relacionados con calificaciones y asistencia consumen más recursos o generan más problemas?
5. ¿Qué expectativas tiene sobre la integración de SIGA y Moodle?
6. ¿Qué requisitos de cumplimiento normativo o reportes para organismos externos deben considerarse en el sistema?
7. ¿Cómo le gustaría poder visualizar la información consolidada de rendimiento académico?
8. ¿Qué alertas o notificaciones automáticas serían útiles para su gestión?
9. ¿Qué tipos de análisis comparativos entre carreras, periodos o cohortes necesita realizar?

10. Desde su perspectiva, ¿cuáles deberían ser las prioridades de implementación del sistema?

Anexo B. Transcripción de Entrevistas a Stakeholders

- Ocho entrevistas realizadas a stakeholders durante el levantamiento de requerimientos, reproducidas como resumen estructurado por pregunta, tal como fueron redactadas al momento del levantamiento (año 2025):

E.1 — Vicerrector del ISTA.

1. ¿Cuáles son los principales desafíos que enfrenta actualmente la institución en la gestión de información académica?

Los desafíos más relevantes incluyen:

- **Falta de automatización:** Muchos procesos aún se gestionan de manera artesanal, usando herramientas como Google Drive, hojas de cálculo en Excel y correos electrónicos. Esta práctica consume tiempo, requiere esfuerzo duplicado y es propensa a errores.
- **Ausencia de una plataforma institucional integrada:** Actualmente, no existe un sistema único que integre todos los procesos académicos y administrativos. Esto genera dispersión de la información y dificulta la toma de decisiones basada en datos.
- **Duplicación de esfuerzos:** Los docentes deben cargar la misma información en múltiples formatos y plataformas (por ejemplo: Moodle, Excel, Drive, y SIGA), lo que genera carga administrativa excesiva.
- **Falta de alertas y seguimiento automatizado:** No existen mecanismos automáticos para notificar a los actores institucionales sobre tareas pendientes, problemas de cumplimiento o seguimiento a estudiantes con bajo rendimiento.
- **Problemas de infraestructura tecnológica:** Se mencionan limitaciones relacionadas con los servidores y la capacidad institucional para albergar un sistema robusto.
- **Débil cultura digital:** Hay una necesidad de fortalecer la cultura de uso de plataformas digitales entre docentes y demás actores. La familiarización con herramientas como Moodle o SIGA no es homogénea.
- **Roles poco definidos:** La falta de claridad sobre quién debe realizar qué tareas y cuándo dificulta la eficiencia en los procesos y la rendición de cuentas.

2. ¿Qué tipos de informes o dashboards necesita para la toma de decisiones estratégicas?

La institución requiere una reportería integral que contemple:

- **Indicadores de cumplimiento docente:** Nivel de avance del PEA, cumplimiento de actividades planificadas, evidencia cargada, calidad de los reactivos o pruebas, etc.
- **Indicadores de asistencia y participación estudiantil.**

- **Análisis de resultados académicos:** Promedios, notas más bajas, comparativas por corte académico, carrera, materia o periodo.
- **Alertas de seguimiento:** Notificaciones sobre estudiantes con bajo rendimiento o inasistencia, que puedan derivarse automáticamente a departamentos como bienestar estudiantil.
- **Datos sociodemográficos:** Estadísticas de género, quintiles socioeconómicos, fichas estudiantiles, etc.
- **Procesos de titulación y prácticas preprofesionales:** Información sobre cumplimiento de requisitos, seguimiento a criterios establecidos, etc.
- **Análisis comparativo entre carreras o periodos académicos:** Por ejemplo, niveles de deserción, matriculación y eficiencia terminal.
- **Visualizaciones claras y amigables:** Tablas, gráficos de barras y pastel, porcentajes y reportes personalizables según el rol (docente, coordinador, vicerrector, etc.).

3. ¿Qué indicadores académicos considera cruciales para monitorear el desempeño institucional?

La institución se basa en el modelo de evaluación externa del CASES:

- **Docencia:** Evaluación del desempeño docente, actividades académicas, cumplimiento del PEA, gestión de portafolios, evidencia cargada.
- **Investigación:** Participación docente en artículos científicos, nivel de indexación, citaciones, etc.
- **Vinculación con la sociedad:** Planificación y ejecución de proyectos, seguimiento a resultados.
- **Gestión académica:** Coordinación entre departamentos, cumplimiento de procesos, trazabilidad de acciones.
- **Infraestructura y accesibilidad:** Indicadores de seguridad, accesibilidad y condiciones de los espacios.
- **Estudiantes:** Desempeño académico, pertinencia del perfil de egreso, cumplimiento de prácticas preprofesionales, etc.

El sistema ideal permitiría adaptar y cargar directamente estos modelos y criterios para dar seguimiento automatizado.

4. ¿Qué procesos actuales relacionados con calificaciones y asistencia consumen más recursos o generan más problemas?

Los más problemáticos son:

- **Carga y reporte de calificaciones:** Los docentes deben pasar manualmente las notas desde Moodle a una matriz en Excel para luego subirlas al SIGA. Esto implica un proceso repetitivo y desgastante.

- **Registro de evidencias:** Los docentes deben cargar portafolios y documentos en múltiples lugares (Moodle, Drive, SIGA), lo que multiplica el trabajo.
- **Generación de informes:** Los docentes tienen que elaborar gráficos, cálculos de promedios, informes firmados, etc., lo que consume mucho tiempo y energía.

5. *¿Cómo le gustaría poder visualizar la información sobre el rendimiento académico?*

Se prefieren:

- **Tablas y gráficos dinámicos:** Que permitan filtrar por carrera, materia, periodo, etc.
- **Gráficos de pastel y barras:** Para análisis visuales rápidos.
- **Porcentajes e indicadores de alerta:** Que marquen visualmente qué estudiantes o asignaturas necesitan atención.
- **Flujo de acciones derivadas:** Que permita derivar automáticamente un caso a bienestar, talento humano, coordinación de carrera, etc.
- **Visualización personalizada según el rol:** Para docentes, coordinadores, vicerrectores y administrativos.

6. *¿Qué alertas o notificaciones automáticas serían útiles para su gestión?*

Algunas alertas sugeridas incluyen:

- **Faltantes en documentación:** Firma incompleta, fechas incorrectas, evidencias no cargadas.
- **Rendimiento académico bajo:** Alertas por baja nota, poca asistencia, tareas no entregadas.
- **Pasos a seguir:** Recordatorios para derivar casos, atender estudiantes, completar registros.
- **Integración con correo institucional:** Notificaciones formales y trazables (no vía WhatsApp).

7. *¿Qué tipos de análisis comparativos entre carreras, periodos o cortes necesitan realizar?*

Se requieren análisis como:

- Comparación de tasas de deserción por corte o carrera.
- Comparación de matriculación entre periodos (por ejemplo, 24-2 vs 25-1).
- Seguimiento a niveles de titulación y cumplimiento de requisitos.
- Evolución del cumplimiento académico entre cohortes.
- Comparativas de rendimiento entre asignaturas, paralelos y modalidades.

8. *¿Cuáles deberían ser las prioridades de implementación del sistema?*

La prioridad número uno es automatizar procesos para reducir la carga administrativa del docente. Esto permitirá:

- Dedicar más tiempo a tutorías, prácticas, seguimiento a estudiantes.
- Mejorar la calidad educativa al eliminar el trabajo redundante.
- Establecer una cultura de gestión de datos y análisis.

También se prioriza que:

- El sistema sea amigable, intuitivo y accesible para todos los roles.
- Se fomente el uso institucional formal (por ejemplo, dejar el uso de WhatsApp y usar correos y notificaciones del sistema).

9. ¿Cuál es su expectativa sobre la integración del SIGA y Moodle?

Se espera que:

- La información fluya de manera bidireccional y automatizada entre SIGA y Moodle.
- El docente no tenga que duplicar tareas al pasar información entre plataformas.
- El sistema sea intuitivo y útil incluso para quienes no están familiarizados con Moodle o SIGA.
- Exista manuales y capacitación para que los docentes puedan adaptarse fácilmente.
- Se formalice la comunicación (abandono de WhatsApp) y se usen medios institucionales.

E.2 — Coordinador de Desarrollo de Software.

1. ¿Cómo se administran actualmente las calificaciones y asistencia en su coordinación?

Actualmente, la asistencia se gestiona a través de la plataforma SIGA, donde los docentes registran la asistencia diariamente. Los estudiantes tienen un plazo de 24 horas para registrar su asistencia. Si no lo hacen, se bloquea el acceso y deben realizar una solicitud en la misma plataforma para que se habilite nuevamente el sistema.

Las calificaciones se administran por medio de la plataforma Moodle. Cada docente es responsable de registrar las notas de sus asignaturas en los tres componentes: prácticas, autónomos y docencia. Luego, se consolidan estas calificaciones en una matriz de Excel institucional diseñada por el Instituto, desde donde se generan los informes finales para ser discutidos en las juntas académicas del primer y segundo parcial. Esta misma matriz

incluye un espacio para registrar notas de recuperación si algún estudiante necesita mejorar su rendimiento.

2. ¿Qué tiempo aproximado dedican usted y su equipo a gestionar estos procesos manualmente?

La gestión es bastante demandante debido a que representa un doble trabajo. Primero, los docentes suben las notas en Moodle, y luego deben trasladarlas manualmente a la matriz de Excel en línea. Esto implica una inversión de tiempo considerable que podría evitarse si existiera una integración entre ambas plataformas, permitiendo el ingreso de notas en un solo paso.

3. ¿Qué informes o consultas específicas sobre rendimiento académico realiza con frecuencia?

De manera frecuente, los docentes realizan análisis individuales del rendimiento académico de sus estudiantes. Además, en los espacios de tutoría se identifican estudiantes que necesitan refuerzo. También se realizan evaluaciones colectivas en las juntas académicas del primer parcial y del final, donde se analiza el desempeño general de los estudiantes y se discuten posibles acciones de mejora.

4. ¿Qué problemas o limitaciones encuentra con los sistemas actuales (SIGA y Moodle)?

El principal problema es la falta de integración entre Moodle y la matriz de Excel utilizada para los informes finales. Esto obliga a los docentes a duplicar esfuerzos, registrando las notas en Moodle y luego transfiriéndolas manualmente al Excel. Este proceso es ineficiente y propenso a errores. Además, el sistema no permite una visualización consolidada inmediata de las notas y evidencias.

5. ¿Qué información necesita para monitorear el avance académico de los estudiantes?

Se requiere información consolidada y periódica (por ejemplo, mensual o por unidad) sobre el cumplimiento de actividades, calificaciones parciales y rendimiento general. Esto permite identificar a tiempo a los estudiantes con dificultades y tomar acciones correctivas antes de que finalice el periodo académico.

6. ¿Qué requisitos específicos de su carrera deberían considerarse en el sistema?

En la carrera se generan proyectos y evidencias pesadas (por su tamaño y contenido) que deben ser subidas y evaluadas. Por tanto, el sistema debe permitir el almacenamiento, visualización y descarga de archivos grandes y de distinta naturaleza. Además, debe contemplar la ejecución de proyectos como evidencia de aprendizaje, lo cual no siempre es fácil de evidenciar con plataformas convencionales.

7. ¿Qué datos necesita visualizar para la toma de decisiones a nivel de coordinación?

La coordinación necesita hacer seguimiento principalmente al cumplimiento de los docentes, específicamente si están cumpliendo con el Plan de Enseñanza y Aprendizaje (PEA), incluyendo las actividades autónomas y sus evidencias. Se requiere visualizar si las actividades planificadas fueron ejecutadas y respaldadas adecuadamente.

8. ¿Cómo realiza actualmente el seguimiento de estudiantes en riesgo académico?

Este seguimiento se realiza en las juntas académicas, donde se identifican estudiantes con bajo rendimiento. El primer paso es un análisis conjunto entre el docente y el tutor para entender las causas. Si se detecta que el problema es de tipo social o personal, se deriva el caso a la unidad de bienestar institucional para un seguimiento especializado. En casos justificados, se pueden aceptar trabajos atrasados o ajustes según corresponda.

9. ¿Qué procesos de su coordinación son prioritarios para automatizar?

Uno de los procesos más demandantes es el seguimiento de evidencias académicas de los docentes. También es prioritario automatizar la gestión del distributivo docente, que actualmente se lleva en una hoja de Excel y requiere mejoras en su presentación y control. Automatizar estos procesos permitiría ahorrar tiempo y aumentar la eficiencia en la coordinación.

10. ¿Qué información estadística utiliza para evaluaciones internas o procesos de acreditación?

Se utiliza la información del proceso llamado "seguimiento Pujico", que incluye cinco etapas. Este proceso permite evaluar si los docentes cumplieron con la planificación,

estrategias de práctica, vinculación curricular y si presentan todas las evidencias de las actividades realizadas. Esta información es clave para las evaluaciones internas y procesos de acreditación institucional.

E.3 — Coordinador 1.

1. ¿Cómo se administran actualmente las calificaciones y asistencia en su coordinación?

Actualmente, las calificaciones se manejan de manera manual y redundante. Los docentes ingresan inicialmente las notas en Moodle. Luego, deben transferir esta información a una plantilla de Excel proporcionada por la cohesión académica o el vicerrectorado. Esta plantilla calcula automáticamente los promedios, pero representa un doble trabajo. Finalmente, los datos consolidados se suben al sistema SIGA. Es decir, se realiza el proceso hasta tres veces. En cuanto a la asistencia, no está completamente integrada y no se refleja directamente en la nota final. Tampoco se incluyen claramente los componentes de docencia práctica y autónoma.

2. ¿Qué tiempo aproximado dedican usted y su equipo a gestionar estos procesos manualmente?

El tiempo varía según la cantidad de materias asignadas a cada docente. Un docente a tiempo completo, con un promedio de cinco materias, puede dedicar aproximadamente dos horas por materia solo para el traspaso de notas entre plataformas y sistemas. Esto implica un uso considerable del tiempo docente en tareas administrativas repetitivas.

3. ¿Qué informes o consultas específicas sobre rendimiento académico realiza con frecuencia?

Se realizan informes por unidad o por resultados de aprendizaje. Las evaluaciones se enfocan principalmente en los exámenes, donde cada pregunta está relacionada con uno o más resultados de aprendizaje establecidos en el PEA (Plan de Estudios). Sin embargo, estas consultas suelen hacerse únicamente en momentos específicos del ciclo académico, como en la mitad o al final del ciclo, y no de forma continua o semanal.

4. ¿Qué problemas o limitaciones encuentra con los sistemas actuales (SIGA y Moodle)?

Los principales problemas identificados incluyen:

- Duplicación de trabajo al ingresar calificaciones en múltiples sistemas.
- Falta de integración de componentes de evaluación como docencia práctica y autónoma.
- Limitaciones en el seguimiento del progreso del estudiante, ya que las notas se consolidan al final del ciclo y no permiten un monitoreo en tiempo real.
- Inexistencia de herramientas que permitan el seguimiento semanal o por unidad.
- La plataforma SIGA no refleja adecuadamente todos los requisitos para continuar con el plan de estudios, como la vinculación, prácticas e inglés.

5. ¿Qué información necesita para monitorear el avance académico de los estudiantes?

Sería ideal contar con una herramienta que permita visualizar el avance de los estudiantes cada semana o cada unidad. Actualmente, la información disponible es limitada y solo accesible al final de cada interciclo o del periodo académico. Se requiere monitorear aspectos como: calificaciones por unidad, nivel de cumplimiento de resultados de aprendizaje, y faltas o asistencias acumuladas.

6. ¿Qué requisitos específicos de su carrera deberían considerarse en el sistema?

Es crucial que el sistema tome en cuenta:

- Las actividades prácticas en talleres, que implican riesgos si no se controlan aspectos como el comportamiento, la seguridad y la limpieza del área.
- Evaluaciones basadas en el comportamiento dentro del taller.
- Seguimiento y registro de requisitos como prácticas, inglés y vinculación como condición para la matrícula o avance al siguiente ciclo.

7. ¿Qué datos necesita visualizar para la toma de decisiones a nivel de coordinación?

Se necesita:

- Información clara sobre el avance académico por unidad o parcial.
- Reportes de asistencia y faltas acumuladas.
- Identificación temprana de estudiantes con bajo rendimiento o posibles deserciones.

- Indicadores sobre cumplimiento de requisitos extracurriculares como prácticas o vinculación.

8. *¿Cómo realiza actualmente el seguimiento de estudiantes en riesgo académico?*

El seguimiento es limitado. No hay un sistema institucionalizado de alertas o monitoreo continuo. En ocasiones, los reportes provienen de los tutores o del área de bienestar estudiantil, pero no existe un reporte constante ni sistemático. La coordinación actúa solo cuando bienestar reporta un caso específico, lo que dificulta un seguimiento proactivo.

9. *¿Qué procesos de su coordinación son prioritarios para automatizar?*

Es prioritario automatizar:

- El ingreso y consolidación de calificaciones en una sola plataforma integrada.
- El control de requisitos como prácticas, vinculación e inglés, para evitar omisiones que impidan la matrícula.
- La generación automática de reportes de rendimiento, asistencia y cumplimiento de requisitos académicos y administrativos.

10. *¿Qué información estadística utiliza para evaluaciones internas o procesos de acreditación?*

Se utiliza la información de los colectivos de cátedra, que recopilan datos sobre evaluaciones, resultados de aprendizaje y retroalimentación de los estudiantes hacia los docentes. Esta información permite ajustar contenidos y metodologías al inicio de cada ciclo académico y contribuye a los procesos de acreditación institucional.

E.4 — Coordinador 2

1. *Sistema de Calificaciones y Ponderación*

Funcionalidades principales:

- Cuadro de ponderación configurable y editable
- Rangos de calificación personalizables:

Excelente: 9.5 - 10

Muy bueno: 8.5 - 9.5

Bueno: 7.5 - 8.5

Insuficiente: < 7

- Cálculo automático de porcentajes de estudiantes por rango
- Validación de campos antes de carga masiva
- Alertas por campos vacíos o datos incorrectos

Recomendaciones adicionales:

- Implementar historial de cambios en ponderaciones y Comparativas históricas de rendimiento
- Backup automático antes de modificaciones
- Plantillas predefinidas por facultad/carrera
- Exportación a múltiples formatos (Excel, PDF, CSV)

2. Gestión de Notas por Etapas

Etapas identificadas:

- Parcial 1 (mitad del período)
- Parcial 2 (final del período)
- Recuperación

Funcionalidades por etapa:

- Carga masiva de notas desde archivo
- Generación automática de informes
- Integración con calendario académico
- Notificaciones automáticas según fechas

Recomendaciones adicionales:

- Implementar evaluación continua (tareas, quizzes).
- Sistema de rúbricas personalizables
- Análisis estadístico por materia y docente
- Comparativas históricas de rendimiento

3. Sistema de Permisos y Roles

Roles identificados:

- **Administrador:** acceso completo, configuración de ponderaciones
- **Docente:** carga de notas, consultas de sus materias
- **Coordinador:** supervisión y reportes
- **Estudiante:** consulta de notas

Recomendaciones adicionales:

- Implementar autenticación de dos factores
- Logs de auditoría detallados
- Permisos granulares por módulo
- Sistema de delegación temporal de permisos

4. Calendario Académico Integrado

Funcionalidades:

- Configuración de fechas límite por proceso
- Automatización de notificaciones
- Bloqueo automático después de fechas límite
- Habilitación especial por administrador

Recomendaciones adicionales:

- Integración con calendarios externos (Google, Outlook)
- Recordatorios escalonados (7 días, 3 días, 1 día antes)
- Visualización de cronograma por docente y estudiante
- Gestión de días feriados y eventos especiales

5. Sistema de Juntas Académicas

Funcionalidades actuales:

- Generación de notas para junta
- Período de modificación post-junta (48 horas)
- Casos especiales y justificaciones
- Finalización definitiva de notas

Recomendaciones adicionales:

- Actas digitales de juntas académicas
- Sistema de votación electrónica para aprobación de notas en el cierre de cada parcial
- Seguimiento de casos especiales
- Workflow de aprobación para modificaciones excepcionales

6. Integración con SIGA

Funcionalidades:

- Formato automático para exportación
- Preparación de datos para carga en SIGA
- Validaciones pre-exportación

Recomendaciones adicionales:

- API de integración directa (si SIGA lo permite)
- Sincronización bidireccional de datos Moodle y M.R.N.A
- Validación cruzada de información
- Log de transferencias exitosas/fallidas

Módulos Adicionales Recomendados

7. Módulo de Reportes y Analytics

- Dashboard ejecutivo con KPIs
- Reportes de rendimiento académico
- Análisis predictivo de deserción
- Comparativas inter-semestrales
- Reportes por docente, materia y programa

8. Módulo de Comunicaciones

- Sistema de notificaciones push
- Envío de emails automáticos
- SMS para casos urgentes
- Portal de noticias académicas
- Chat integrado docente-estudiante

9. Módulo de Asistencia

- Control biométrico o por código QR
- Integración con sistema de notas
- Alertas por inasistencias
- Reportes de ausentismo
- Justificaciones digitales

10. Firma Electrónica

Funcionalidades mencionadas:

- Generación automática de reportes firmados
- Validación legal de documentos
- Descarga de archivos con firma

Recomendaciones adicionales:

- Certificados digitales por usuario
- Timestamping para validez legal
- Integración con entidades certificadoras
- Backup seguro de documentos firmados

11. Módulo de Distributivo Docente

Funcionalidades sugeridas:

- Asignación automática de materias
- Gestión de carga horaria
- Conflictos de horarios
- Optimización de recursos

El Sistema de 48 Horas Post-Junta

¿Qué dice la transcripción?

- Después de realizar la junta académica, hay un período de 48 horas donde el docente puede hacer modificaciones a las notas
- Durante la junta se analizan casos especiales (estudiantes enfermos, situaciones justificadas, etc.)
- La junta puede decidir dar oportunidades adicionales (exámenes de recuperación)
- Pasadas las 48 horas, cualquier cambio requiere autorización del administrador

Cómo debería funcionar en el sistema:

Durante la Junta

- El sistema genera las notas para presentar en junta
- Se registran los casos especiales discutidos
- Se aprueban modificaciones o exámenes adicionales

Período de Gracia (48 horas)

Junta Académica [48 HORAS DE MODIFICACIÓN LIBRE] Bloqueo Automático

Funcionalidades del sistema durante estas 48 horas:

- El docente puede modificar notas libremente
- Puede incluir estudiantes que tomaron exámenes especiales
- Puede ajustar calificaciones según lo acordado en junta
- El sistema muestra un contador regresivo de tiempo restante

Después de 48 Horas

- Bloqueo automático de modificaciones
- Solo el administrador puede habilitar cambios
- Se requiere justificación documentada
- Se genera log de auditoría para cualquier modificación tardía

Esto aplica para las 3 etapas mencionadas:

- Junta de Parcial 1 (mitad del período)

- Junta de Parcial 2 (final del período)
- Junta de Recuperación

Recomendación técnica:

El sistema debería enviar notificaciones automáticas:

- 24 horas restantes: "Recordatorio: quedan 24 horas para modificaciones"
- 6 horas restantes: "Última oportunidad para modificar notas"
- Al vencimiento: "Período de modificación cerrado. Contacte al administrador para cambios"

E.5 — Docente 1.

1. ¿Cómo es su proceso actual para registrar calificaciones y asistencia?

El proceso de registro de calificaciones se realiza mediante la evaluación de los componentes prácticos, autónomos y de docencia. Se califican las etapas de primer y segundo parcial, así como la recuperación. Inicialmente, las notas se ingresan en Moodle, desde donde se descarga una matriz de calificaciones. Esta información se transfiere a una matriz en Excel que consolida todas las carreras. A partir de ahí, se obtiene la nota final y se transfiere al sistema oficial en los diferentes cortes (primer parcial, segundo parcial, recuperación). En cuanto a la asistencia, se utiliza el sistema SIGA, donde el registro debe hacerse diariamente.

2. ¿Cuánto tiempo dedica semanalmente a estas tareas administrativas?

Se dedica aproximadamente 26 horas semanales a las tareas administrativas, lo cual incluye el manejo de calificaciones, asistencia y otros procesos relacionados.

3. ¿Qué aspectos del proceso actual encuentra más problemáticos o tediosos?

Los aspectos más problemáticos incluyen la extensión innecesaria de algunos procesos, la complejidad en la generación de informes, la carga repetitiva de datos entre sistemas (por ejemplo, de Moodle a Excel y luego al sistema institucional), y la necesidad de solicitar autorizaciones cuando no se registran datos dentro del tiempo establecido.

4. ¿Qué funcionalidades facilitarían significativamente su trabajo?

Automatización de procesos, especialmente en el traspaso de datos entre plataformas. También sería útil un sistema que reduzca la redundancia en la carga de información y permita una gestión más eficiente y directa.

5. ¿Qué reportes o análisis necesita generar con frecuencia?

Frecuentemente se generan reportes de notas y reportes de asistencia. Ocasionalmente también se requieren reportes de rendimiento académico para evaluar el progreso de los estudiantes.

6. ¿Cómo le gustaría que funcionara el sistema para simplificar su trabajo?

El sistema ideal debería ser claro, directo y rápido. Debería permitir una gestión ágil de las calificaciones, asistencia, cronogramas y demás tareas administrativas sin necesidad de realizar pasos redundantes o depender de múltiples plataformas.

7. ¿Qué experiencia tiene utilizando Moodle para la gestión de calificaciones?

La experiencia con Moodle ha sido muy buena. Es una plataforma amigable, interactiva y funcional, que permite una adecuada interacción con los estudiantes y ofrece herramientas útiles para la evaluación y seguimiento académico.

8. ¿Qué experiencia tiene utilizando SIGA para registro de asistencia?

La experiencia con SIGA es positiva siempre que se realice el registro diariamente. Sin embargo, si no se registra en el tiempo estipulado, se debe solicitar autorización al coordinador de carrera para poder hacerlo posteriormente, lo cual genera inconvenientes adicionales.

9. ¿Qué formato de presentación de información preferiría para monitorear el rendimiento de sus estudiantes?

Se prefiere el formato en Excel, ya que permite una visualización clara de los datos y facilita el análisis del rendimiento académico de los estudiantes.

10. ¿Qué notificaciones automáticas serían útiles en su labor docente?

Serían útiles recordatorios automáticos sobre reuniones, actividades pendientes y otras tareas académicas. Un calendario integrado que refleje las actividades programadas sería ideal.

11. ¿Cómo le gustaría poder comunicar a los estudiantes sus avances académicos?

A través de plataformas multimedia o institucionales, como Moodle o SIGA, donde los estudiantes puedan revisar directamente sus notas y observaciones sobre su desempeño.

12. ¿Qué otros procesos relacionados con la evaluación académica le gustaría automatizar?

Sería conveniente automatizar procesos relacionados con la evaluación académica que involucren a talento humano, la documentación institucional y aspectos vinculados al área de inglés, para integrar toda la información relevante en un solo sistema.

E.6 — Docente 2.

1. ¿Cómo es su proceso actual para registrar calificaciones y asistencia?

El registro de calificaciones y asistencia se realiza a través del sistema SIGA. Para la asistencia, se ingresa al módulo correspondiente dentro del sistema, se revisa el distributivo de clases, se selecciona la materia y se marca la asistencia diaria de los estudiantes. Para las calificaciones, se utiliza el módulo de "notas" dentro del mismo sistema, donde se ingresan los promedios calculados previamente. Los datos de calificaciones también se extraen desde Moodle (plataforma virtual), donde se registran las actividades, y luego se pasan manualmente al SIGA.

2. ¿Cuánto tiempo dedica semanalmente a estas tareas administrativas?

La asistencia se registra diariamente y toma entre 15 a 20 minutos por clase, ya que implica verificar la presencia de los estudiantes, justificar ausencias y relacionarlas con los temas impartidos. El tiempo para ingresar calificaciones varía, pero requiere un trabajo adicional al tener que calcular promedios desde Moodle, descargar plantillas y luego transcribir los datos al SIGA.

3. ¿Qué aspectos del proceso actual encuentra más problemáticos o tediosos?

Uno de los principales problemas es que el sistema SIGA bloquea la opción de registrar asistencia del día anterior si no se hace en el tiempo establecido, lo que obliga a realizar solicitudes de desbloqueo. Además, el proceso de trasladar notas desde Moodle hacia SIGA y luego hacia reportes en Excel es repetitivo y propenso a errores, lo que lo vuelve tedioso.

4. ¿Qué funcionalidades facilitarían significativamente su trabajo?

Sería útil que el sistema fuera más dinámico y que existiera una integración automática entre Moodle y SIGA. Esto permitiría que las notas ya registradas en Moodle (tareas, cuestionarios, actividades prácticas) se sincronizaran directamente con el sistema institucional sin tener que calcular promedios y transcribirlos manualmente. También facilitaría contar con una sola lista oficial de estudiantes en todos los sistemas para evitar tener que ordenar y comparar manualmente.

5. ¿Qué reportes o análisis necesita generar con frecuencia?

Se requiere generar reportes de asistencia, calificaciones y rendimiento académico. Estos reportes son necesarios tanto para el seguimiento individual de los estudiantes como para las juntas académicas y procesos administrativos internos.

6. ¿Cómo le gustaría que funcionara el sistema para simplificar su trabajo?

Le gustaría un sistema más ágil y fácil de usar, que permita obtener información rápidamente sin tener que seguir muchos pasos. Además, se valoraría una mayor interconexión entre plataformas (por ejemplo, que Moodle y SIGA compartan automáticamente los datos).

7. ¿Qué experiencia tiene utilizando Moodle para la gestión de calificaciones?

Tiene un buen dominio de Moodle. Considera que es una plataforma amigable para registrar actividades, tareas, prácticas, cuestionarios, y asignar calificaciones. Maneja con fluidez el ingreso de actividades y la evaluación de estudiantes dentro de esta plataforma.

8. ¿Qué experiencia tiene utilizando SIGA para registro de asistencia?

También tiene experiencia utilizando SIGA, aunque menciona que el sistema tiene ciclos y reglas que requieren adaptación. Con el tiempo ha aprendido a manejarlo eficientemente, aunque no es tan flexible como se desearía.

9. ¿Qué formato de presentación de información preferiría para monitorear el rendimiento de sus estudiantes?

Prefiere un formato de tabla que contenga toda la información relevante por estudiante y que además incluya un resumen estadístico que permita ver de forma rápida el rendimiento general y específico.

10. ¿Qué notificaciones automáticas serían útiles en su labor docente?

Serían útiles las notificaciones que alerten cuando no se ha cargado la asistencia o falta calificar alguna actividad. Estas alertas ayudarían a no olvidar tareas administrativas pendientes.

11. ¿Cómo le gustaría poder comunicar a los estudiantes sus avances académicos?

Preferiría que los estudiantes reciban notificaciones automáticas cuando se califica una actividad. Esto permitiría que los estudiantes estén al tanto de sus notas en tiempo real, sin necesidad de buscarlas manualmente.

12. ¿Qué otros procesos relacionados con la evaluación académica le gustaría automatizar?

Le gustaría automatizar la gestión de listas de estudiantes. Actualmente, se deben comparar las listas del SIGA, Moodle y otros documentos, lo que consume tiempo. Idealmente, las listas oficiales deberían integrarse automáticamente en todas las plataformas, manteniendo el mismo orden y estructura.

E.7 — Docente 3.

1. ¿Cómo es su proceso actual para registrar calificaciones y asistencia?

El proceso se realiza en tres plataformas distintas. Primero, se registran las calificaciones en Moodle, el aula virtual. Luego, se trasladan a una matriz de Excel en Google Drive,

que se utiliza para las juntas académicas. Finalmente, una vez validadas, se cargan las notas en el sistema SIGA. En cuanto a la asistencia, se registra diariamente en SIGA, pero para las juntas académicas se requiere también transformar esa información en un formato porcentual que se incluye en la matriz de Excel.

2. ¿Cuánto tiempo dedica semanalmente a estas tareas administrativas?

Durante la mayoría del semestre, se dedica entre una hora y hora y media por semana. Sin embargo, en las semanas finales de cada parcial el trabajo aumenta significativamente, destinando entre 6 a 8 horas semanales para cumplir con las cargas de notas y asistencias en las tres plataformas.

3. ¿Qué aspectos del proceso actual encuentra más problemáticos o tediosos?

Los aspectos más tediosos son tener que subir las calificaciones manualmente en tres plataformas distintas y ajustar los promedios y ponderaciones. Además, hay componentes del sistema como las actividades que no permiten decimales (por ejemplo, algunos foros en Moodle) que dificultan reflejar correctamente las calificaciones finales. También, el ingreso manual de asistencia en SIGA es un punto que consume tiempo y esfuerzo, ya que no permite control cruzado desde otras plataformas.

4. ¿Qué funcionalidades facilitarían significativamente su trabajo?

Facilitaría mucho poder registrar las calificaciones una sola vez y que el sistema replique esa información automáticamente en las otras dos plataformas (Moodle y SIGA). Además, sería útil que el sistema garantice el cálculo correcto de promedios, considerando adecuadamente los decimales, para evitar errores que puedan perjudicar a los estudiantes.

5. ¿Qué reportes o análisis necesita generar con frecuencia?

Se generan reportes de seguimiento del rendimiento académico de los estudiantes, especialmente al finalizar cada parcial. También se elaboran reportes de asistencia que se extraen del SIGA. Ambos deben ser presentados durante las juntas académicas y cargados en un portafolio electrónico docente cada periodo académico.

6. ¿Cómo le gustaría que funcionara el sistema para simplificar su trabajo?

El sistema ideal sería amigable y práctico, permitiendo ingresar directamente las notas promediadas por componente (docencia, práctica autónoma, evaluación parcial) y que calcule automáticamente el promedio del parcial. Esta información debería replicarse automáticamente en Moodle y SIGA o extraerse directamente desde la matriz de Excel.

7. ¿Qué experiencia tiene utilizando Moodle para la gestión de calificaciones?

La experiencia no ha sido del todo buena. Resulta complicado configurar correctamente los pesos de cada componente, y esto suele causar errores en los promedios finales. Además, algunas actividades como foros no permiten calificaciones con decimales, lo que complica la obtención de una nota exacta al final del parcial.

8. ¿Qué experiencia tiene utilizando SIGA para registro de asistencia?

El uso de SIGA para la asistencia es práctico y sencillo en general. El principal inconveniente es que no permite control desde otras plataformas y todo debe ingresarse manualmente, estudiante por estudiante.

9. ¿Qué formato de presentación de información preferiría para monitorear el rendimiento de sus estudiantes?

Aunque no se menciona explícitamente en la entrevista, se puede inferir que se preferiría un formato visual y consolidado, donde se muestre el promedio general de cada estudiante, junto con sus notas por componente y su porcentaje de asistencia, idealmente todo en una sola vista.

10. ¿Qué notificaciones automáticas serían útiles en su labor docente?

Sería útil recibir alertas que recuerden los plazos importantes, como la fecha de cierre de parciales y la necesidad de cargar calificaciones. Aunque Moodle permite programar algunas alertas, sería ideal tener notificaciones integradas también para otras tareas administrativas relacionadas con la evaluación.

11. ¿Cómo le gustaría poder comunicar a los estudiantes sus avances académicos?

Aunque no se detalla en la entrevista, dada la preocupación por los promedios y la precisión de las calificaciones, se puede deducir que preferiría que los estudiantes tengan

acceso a una vista clara y precisa de sus calificaciones acumuladas y promedios, de forma automática, confiable y sin discrepancias entre plataformas.

12. ¿Qué otros procesos relacionados con la evaluación académica le gustaría automatizar?

Le gustaría automatizar la gestión documental de prácticas preprofesionales y actividades de vinculación. Actualmente estos procesos generan mucho papeleo físico (informes con varios anexos) y sería ideal que todo se pueda hacer de forma digital, incluyendo la validación y entrega.

E.8 — Responsable de TIC - entrevista crítica, sustenta el ADR-003 (SIGA sin API pública).

1. ¿Cuál es la arquitectura actual de los sistemas SIGA y Moodle?

SIGA:

Frontend: Tecnologías: HTML, CSS, JavaScript

Backend Tecnología base: JAVA

MOODLE:

Frontend: Tecnologías: HTML, CSS, JavaScript (jQuery).

Backend Tecnología base: PHP (orientado a objetos)

2. ¿Qué versiones específicas de software se están utilizando?

SIGA:

No se dispone de esa información ya que este sistema está bajo la administración de los Institutos Tecnológicos de Educación Superior a Nivel Nacional.

Moodle:

Moodle 4.1.14+ (Build: 20241011)

PHP Version 8.1.28

MySQL 8.0.36

3. ¿Qué APIs o interfaces están disponibles para la integración?

Con lo antes indicado no se tiene conocimiento sobre APIs que estén disponibles para su uso y tampoco se ha informado al respecto por parte de TIC de Senacyt Quito sobre SIGA. Respecto a Moodle, plataforma administrada de la institución el servicio API REST, que es propio de la plataforma pero no está activa al momento.

4. ¿Cuáles son las principales limitaciones técnicas para la integración?

SIGA es un sistema que dispone de un sistema Académico, Sistema Administrativo y Sistema Documental de la Senacyt para los institutos tecnológicos de educación superior, es un sistema informático que está a cargo de la Senacyt con sede en Quito, sistema del cual el IST del Austro no cuenta con el aval para su administración, por lo que su integración con el mismo dependería de la autorización de algún ente superior de esta entidad, y toda gestión que esto conlleve en la parte técnica.

El sistema Moodle opera dentro de los servidores de un proveedor del cual la institución mantiene un convenio vigente más el help desk respectivo y monitoreo de enlaces e infraestructura, por lo que su integración sería viable en ese sentido. De ser necesario se puede gestionar actualizar la versión de moodle actualmente manejamos una versión (LTS) a una versión superior o la que el proveedor del servidor de este servicio recomiende por temas de seguridad de la plataforma. Las limitaciones van a depender principalmente de las recomendaciones que brinde el proveedor sobre la mejor opción para integrarse con otro sistema informático en relación a las seguridades ya que sobre sus servidores se maneja esta plataforma. Sin embargo, a criterio de la Unidad de TIC se podría hacer uso de API REST de Moodle con la respectiva documentación API. Cualquier requerimiento se tiene que gestionar con el proveedor del servidor a través de TIC del Instituto previa Autorización de la Máxima Autoridad, sin embargo, esta plataforma es de uso de la entidad por lo que de ser requerido en etapa de pruebas iniciales se puede utilizar un historial de backup que se genera de la plataforma previa autorización de Rectorado, mismos que mantenemos disponibles de periodos anteriores. En la pregunta 6 se agrega un punto importante y adicional sobre la implementación en el servidor que corresponda.

5. ¿Qué consideraciones de seguridad y acceso a datos deben tenerse en cuenta?

- Primero hacer uso de HTTPS en ambas plataformas para evitar ataques de riesgo (moodle ya dispone), como uso de tokens de acceso generados por moodle para consumir los servicios web sin que sean expuestos.
- Crear roles específicos y restringidos en moodle para la cuenta que hará uso por parte del otro sistema informático, no usando cuentas tipo administrador para este fin.
- Validar la información que se intercambia en los sistemas informáticos.
- Evitar SQL INJECTION a través del uso del servicio web a aplicarse.
- Nunca acceder directamente a la base de datos de Moodle desde el otro sistema sin usar una capa intermedia segura, como un API o un servicio autenticado.
- Encriptar información sensible (como contraseñas, tokens, identificadores únicos).
- Realizar la integración en un entorno de pruebas antes de pasar a producción.
- De ser factible montar la máquina virtual del sistema informático a desarrollarse dentro de los servidores del mismo proveedor para minimizar los riesgos, por las seguridades y servicios de soporte adicional que el proveedor de los servidores brinda actualmente.
- Realiza copias de seguridad regulares antes y después de cambios importantes o integraciones nuevas así como también hacer un control de monitoreo de acceso, que permita hacer un control adecuado de la seguridad y del acceso de los usuarios.

6. ¿Qué infraestructura tecnológica soportará el sistema integrado?

Opción 1 - Servidor Proveedor Convenio con el Instituto:

8 Procesadores Virtuales, Almacenamiento: 400 GB, memoria ram: 16GB son sus capacidades máximas actuales las mismas que están copadas al 100% con máquinas virtuales en uso de otros sistemas informáticos institucionales. Por lo que para hacer uso de este servicio se debe tramitar la ampliación del convenio actual para disponer de mayores recursos. Esto se gestionaría por parte del área de TIC, Rectorado, áreas pertinentes y posteriormente levantar la máquina virtual y acceso al servidor que al menos podría contener lo siguiente: 2CPUs, 100GB de Almacenamiento, memoria Ram 4GB como mínimo o lo que recomiende al menos el estudio realizado. En caso de aprobarse la ampliación al convenio se generará una máquina virtual para su gestión con la herramienta de control aaPANEL única disponible al ser open source y la máquina virtual disponible con los accesos al servidor y/o panel de control.

Opción 2 - Servidor Proveedor Senecyt:

Ubuntu Server, 2 Cpus, Almacenamiento: 100 GB, Memoria 4GB mínimo al menos o lo recomendado en el estudio realizado. Se generará una máquina virtual para su gestión con la herramienta de control aaPANEL, única disponible al ser open source. Estos recursos se gestionaría por parte del área de TIC con Senacyt sobre todo en el aspecto de recursos

de servidor, ips, permisos respectivos para levantar la máquina virtual y disponer del acceso al servidor y/o panel de control.

Otros componentes de infraestructura:

- Computadores y dispositivos de usuarios: Sistema Operativo Windows 10, Windows 11, Linux Mint, Linux Ubuntu, Mac.
- Red: local (LAN) cableada e inalámbrica para servicio de internet e impresión de documentos, misma que no dispone de servidores físicos conectados.
- Router: Principal configurado para red LAN.
- Datacenter: Servicio en la Nube 100%.
- Seguridad: Al menos certificado de seguridad SSL Let's Encrypt
- Soporte y mantenimiento: Servidor de proveedor Opción 1 y servidor de proveedor de Senacyt Opción 2. Por parte de TIC de la entidad se recomienda el servicio de proveedor Opción 1 por temas de seguridad, soporte, helpdesk oportuno, monitoreo constante de enlaces e infraestructura, seguridad y uso de servidores donde se dispone de la plataforma moodle actualmente.

7. ¿Cómo se gestionan actualmente las bases de datos de ambos sistemas?

Del sistema SIGA se conoce que mantiene un contrato vigente con CNT en relación a la infraestructura tecnológica de datacenter virtual, sistema de respaldos y monitoreo para la Senacyt e institutos de educación superior pero como se indicó no está bajo nuestro control esta plataforma. Del Sistema Moodle TIC gestionamos y realizamos copias (backups) regulares de todo el sistema al finalizar cada periodo académico. Cada docente puede hacer copias de sus cursos que maneja en la plataforma Moodle cuando sea necesario, para lo cual tienen conocimiento.

8. ¿Existe documentación técnica disponible de los sistemas actuales?

Del sistema SIGA:

Se dispone una página del manual de usuario completo de la plataforma:
<http://manualsigainstitutos.gob.ec/>

Del sistema Moodle:

Lo concerniente a lo indicado en consultas previas o posteriores y manual de usuario de uso de la plataforma del instituto que se compartirá junto a este archivo.

9. ¿Qué procesos de respaldo y recuperación están implementados?

Respecto a SIGA, los respaldos como se indicó está a cargo de CNT mediante un contrato vigente con la SENACYT. De Moodle respaldos regulares completo de la plataforma al finalizar cada periodo académico.

10. ¿Qué recomendaciones técnicas tiene para la implementación?

Se recomienda considerar la normativa del Código Ingenius para el desarrollo e implementación de plataformas informáticas para entidades del sector público. De ser posible del estudio realizado apearse a una infraestructura tecnológica de software similar a la plataforma con la cual se va a integrar haciendo uso de un framework o marco de trabajo que brinde una seguridad destacada y conexión a Moodle que permita por ejemplo una tokenización segura, autenticación de usuarios y control de permisos.

11. ¿Cuáles son los picos de demanda o carga en los sistemas actuales?

A inicios, intermedio y fin de cada periodo académico. En la instancia inicial de cada periodo se realiza altas masivas de estudiantes que ingresan a primer ciclo, la creación y restauración de cursos, cambios de rol y matrículas, gestión de cursos por parte de los docentes, por lo que en ese lapso de tiempo existirá una alta carga en la base de datos y sistema de archivos. Al finalizar la parte intermedia del ciclo denominada Parcial I y al finalizar el ciclo Parcial II por la carga masiva de trabajos de los estudiantes, la generación de informes y calificaciones en esta parte intermedia y final por parte de los docentes. La baja carga se da una vez que los docentes concluyen y hacen la entrega de notas del parcial II y no exista asistencia a clases por parte de los estudiantes.

12. ¿Qué consideraciones de escalabilidad deben tenerse en cuenta?

- Que el nuevo sistema informático se comuniquen con Moodle por medio de los servicios web (API) y no de manera directa a sus tablas.
- Realizar un sistema informático con uso de balanceo de carga recomendable. Sin embargo considere el número de usuarios máximo que podría hacer del nuevo sistema informático, considerando nuevas carreras que se puedan crear, futuros usuarios del sistema.
- Que se optimicen las consultas que se realicen al hacer integraciones.

- Considerar las mejores prácticas de seguridad en el desarrollo de sistemas informáticos y su integración.
- Separar el almacenamiento de archivos de la data de la BD.
- Que se considere un espacio de almacenamiento adecuado, cpu, ram y disco para la implementación del sistema informático.

En definitiva, una integración entre Moodle y otro sistema informático debe funcionar el día de su implementación, y además, estar preparada para crecer mañana sin colapsar.

Anexo C. Catálogo Completo de Requerimientos Funcionales

Catálogo completo de los 23 requerimientos funcionales levantados durante el diagnóstico (18 implementables + 5 diferidos), con su descripción detallada y su clasificación MoSCoW.

Código	Nombre	Descripción	MoSCoW	Estado
RF-001	Carga masiva de datos consolidados desde el MRNA hacia el SIGA	El módulo MRNA debe generar y poner a disposición del coordinador o del administrador el archivo con las notas finales ya consolidadas de un período, carrera y paralelo, en el formato exacto que exige el SIGA, listo para su carga manual en el sistema nacional.	Must	Implementar
RF-002	Sincronización de datos desde el Moodle hacia el MRNA	El módulo MRNA debe reflejar los cambios que se realicen sobre el Moodle para el registro de notas (crear nota y actualizar nota), incluyendo la validación cruzada de información entre Moodle y MRNA.	Must	Implementar
RF-003	Interfaz para recibir los datos del Moodle hacia el MRNA	El módulo debe incluir una interfaz con una plantilla que enliste todos los campos necesarios para recibir las notas desde el Moodle al final de cada parcial; las notas corresponden a Docencia, Práctica, Autónoma, Evaluación y Recuperación.	Must	Implementar
RF-004	Dashboard con histograma visual de rendimiento	Crear una interfaz que muestre un histograma con el detalle de cada curso o estudiante, para visualizar el estado de aprendizaje y facilitar la toma de decisiones al Instituto.	Should	Implementar
RF-005	Gestión de roles y permisos	El sistema debe permitir la gestión de roles y permisos diferenciados para los cuatro perfiles institucionales: Administrador, Coordinador, Docente y Directivo.	Must	Implementar
RF-006	Notificaciones automáticas por bajo rendimiento	El sistema debe generar y enviar notificaciones automáticas cuando se detecten casos de bajo rendimiento académico (nota menor a 7/10) o ausentismo crítico (asistencia menor al 70%), mediante correo y dentro del dashboard.	Should	Implementar
RF-007	Filtrado y búsqueda avanzada	El dashboard debe permitir filtrar por carrera, materia, paralelo y período académico, y buscar estudiantes específicos por nombre o cédula.	Should	Implementar
RF-008	Generación de reportes (CSV/Excel/PDF)	El módulo debe generar un archivo de notas consolidado por ciclo, notas por materia, documento de junta académica y reportes de rendimiento académico.	Must	Implementar
RF-009	Firma electrónica de reportes (Quipux)	Los reportes de notas deben firmarse mediante firma electrónica para garantizar su validación.	Should	Diferido

RF-010	Validación automática de errores antes de la carga masiva	El sistema debe detectar errores previos a la carga (campos vacíos, formato incorrecto, IDs inválidos) y mostrar mensajes específicos que permitan corregir sin reiniciar la carga completa.	Must	Implementar
RF-011	Módulo de ayuda interactiva	El sistema debe incluir un apartado de ayuda con tutoriales interactivos, preguntas frecuentes y un chat de soporte para guiar a docentes y administradores.	Could	Diferido
RF-012	Acceso restringido por periodo activo	El sistema debe controlar la edición de notas y asistencia según el estado del período académico (abierto/cerrado); una vez cerrado, solo lectura, salvo permisos especiales.	Must	Implementar
RF-013	Carga masiva de usuarios desde archivo	El administrador podrá subir archivos CSV o Excel para registrar docentes de forma masiva, con validación de duplicados y de estructura de datos.	Should	Implementar
RF-014	Notificaciones programadas	El sistema debe permitir configurar recordatorios automáticos recurrentes de carga de notas, cierre de parcial o asistencia faltante, según el calendario académico.	Should	Implementar
RF-015	Junta académica con gracia de 48 horas	Generación de notas para junta académica con un período de modificación posterior de 48 horas para casos justificados; vencida la ventana, el reporte final ya no admite cambios.	Should	Implementar
RF-016	Portal de noticias académicas	El sistema debe incluir un portal de noticias integrado, con categorización automática, suscripciones y notificaciones personalizadas para la comunidad educativa.	Could	Diferido
RF-017	Chat integrado docente-estudiante	El sistema debe incorporar un módulo de chat en tiempo real entre docentes y estudiantes, con historial organizado por materia.	Could	Diferido
RF-018	Auditoría de notas y asistencia	El sistema debe auditar toda modificación de notas o asistencia, registrando fecha y hora (timestamp) de cada cambio.	Should	Implementar
RF-019	Historial de cambios para notas y asistencia	Implementar un historial de cambios sobre ponderaciones y comparativas históricas de rendimiento.	Could	Opcional
RF-020	Gestión de feriados y eventos especiales	Permitir la gestión de días no laborables, incluyendo feriados y eventos especiales de la institución.	Could	Opcional
RF-021	Evaluación continua (tareas, quizzes)	Módulo de evaluación continua para crear, gestionar y calificar actividades académicas (tareas, quizzes, proyectos, participaciones), con seguimiento y retroalimentación.	Could	Opcional
RF-022	Calendario académico	Módulo de calendario académico que sincroniza eventos institucionales, plazos y horarios en una vista unificada.	Could	Opcional

RF-023	Integración con calendarios externos (Google, Outlook)	Integración bidireccional con Google Calendar y Microsoft Outlook para sincronizar eventos académicos y detectar conflictos de horario.	Could	Diferido
--------	--	---	-------	----------

Anexo D. Historias de Usuario Completas

La siguiente tabla consolida las 18 HU con sus datos principales.

ID HU	RF	Título	MoSCoW	Pts	Microservicio
HU-001	RF-001	Carga masiva de notas MRNA→SIGA	Must	8 pts	sync-service
HU-002	RF-002	Sincronización Moodle→MRNA	Must	13 pts	sync-service
HU-003	RF-003	Interfaz notas por componente	Must	5 pts	academic-service
HU-004	RF-004	Dashboard histograma rendimiento	Should	8 pts	academic-service
HU-005	RF-005	Gestión roles y permisos	Must	5 pts	auth-service
HU-006	RF-006	Notificaciones bajo rendimiento	Should	5 pts	notification-service
HU-007	RF-007	Filtrado y búsqueda avanzada	Should	3 pts	academic-service
HU-008	RF-008	Reportes PDF/Excel/CSV	Must	8 pts	report-service
HU-009	RF-010	Validación pre-carga archivos	Must	3 pts	sync-service
HU-010	RF-012	Bloqueo período cerrado	Must	2 pts	academic-service
HU-011	RF-013	Carga masiva usuarios	Should	5 pts	auth-service
HU-012	RF-014	Notificaciones programadas	Should	5 pts	notification-service
HU-013	RF-015	Período de gracia 48h	Should	3 pts	academic-service
HU-014	RF-018	Auditoría notas y asistencia	Should	3 pts	academic-service
HU-015	RF-019	Historial versiones nota	Could	3 pts	academic-service
HU-016	RF-020	Feriados y eventos especiales	Could	3 pts	academic-service
HU-017	RF-021	Evaluación continua	Could	5 pts	academic-service
HU-018	RF-022	Calendario académico	Could	3 pts	academic-service

Total puntos core (14 HU): 77 pts | Total puntos opcionales (4 HU): 14 pts | Total global: 91 pts

RF Core – 14 Historias de Usuario (Must + Should)

Estas 14 historias son obligatorias para el prototipo

HU-001 — RF-001 Carga masiva de notas MRNA→SIGA	
RF Asociado	RF-001 Carga masiva de notas MRNA→SIGA
Prioridad MoSCoW	Must 8 pts
Estimación	8 puntos de historia
Microservicio	sync-service

Como...	coordinador o administrador académico
Quiero...	generar y descargar el archivo con las notas consolidadas de un período, carrera y paralelo, en el formato exacto que exige el SIGA
Para...	cargarlo manualmente en el sistema nacional sin tener que reconstruir la información desde cero
Given (Dado que...)	Estoy autenticado como coordinador o administrador. Las notas del período, la carrera y el paralelo que necesito reportar ya están registradas y consolidadas en el MRNA.
When (Cuando...)	Selecciono el período, la carrera y el paralelo, y solicito la generación del archivo de exportación hacia el SIGA.
Then (Entonces...)	El sistema genera el archivo con las notas finales consolidadas del paralelo en el formato del SIGA y lo pone disponible para su descarga. Si alguna nota no cumple el formato esperado, el sistema lo señala antes de entregar el archivo, en lugar de generar un archivo con datos inconsistentes.

HU-002 — RF-002 Sincronización Moodle→MRNA	
RF Asociado	RF-002 Sincronización Moodle→MRNA
Prioridad MoSCoW	Must 13 pts
Estimación	13 puntos de historia
Microservicio	sync-service
Como...	coordinador o administrador
Quiero...	que las calificaciones registradas en Moodle se sincronicen hacia el MRNA clasificadas por componente (Docencia, Práctica, Autónoma, Evaluación)
Para...	eliminar la doble digitación manual que actualmente genera inconsistencias en los registros
Given (Dado que...)	El sync-service está configurado con el token de acceso de la instancia Moodle. Existe un período académico activo en el sistema.
When (Cuando...)	Ejecuto la sincronización desde el MRNA, de forma manual o programada.
Then (Entonces...)	Las notas por componente de cada docente y paralelo se actualizan en el MRNA.

	El sistema registra un log de sincronización con fecha, número de registros procesados y errores encontrados.
--	---

HU-003 — RF-003 Interfaz de notas por componente	
RF Asociado	RF-003 Interfaz de notas por componente
Prioridad MoSCoW	Must 5 pts
Estimación	5 puntos de historia
Microservicio	academic-service
Como...	docente
Quiero...	ver en el MRNA una interfaz que liste las notas de mis estudiantes por componente (Docencia, Práctica, Autónoma, Evaluación, Recuperación)
Para...	revisar y confirmar la información antes de la consolidación del parcial
Given (Dado que...)	Estoy autenticado como docente. Tengo asignaturas y paralelos asignados en el período académico activo.
When (Cuando...)	Accedo al módulo de registro de notas y selecciono un paralelo.
Then (Entonces...)	Veo una tabla con todos los estudiantes del paralelo con columnas para cada componente. El promedio parcial se calcula automáticamente. Los campos son de solo lectura hasta confirmar la sincronización.

HU-004 — RF-004 Dashboard histograma de rendimiento	
RF Asociado	RF-004 Dashboard histograma de rendimiento
Prioridad MoSCoW	Should 8 pts
Estimación	8 puntos de historia
Microservicio	academic-service
Como...	coordinador o directivo
Quiero...	acceder a un dashboard con histogramas de rendimiento académico filtrable por carrera, materia y paralelo

Para...	identificar grupos o estudiantes en riesgo y facilitar la toma de decisiones en junta académica
Given (Dado que...)	Estoy autenticado como coordinador o directivo. Existe al menos un período académico con notas registradas.
When (Cuando...)	Accedo al dashboard de rendimiento.
Then (Entonces...)	Veo histogramas con la distribución de calificaciones del grupo. Se muestra el porcentaje de aprobados y reprobados. Puedo filtrar por carrera, materia, paralelo y período sin recargar la página.

HU-005 — RF-005 | Gestión de roles y permisos

RF Asociado	RF-005 Gestión de roles y permisos
Prioridad MoSCoW	Must 13 pts
Estimación	13 puntos de historia
Microservicio	auth-service
Como...	administrador del sistema
Quiero...	crear y asignar roles diferenciados (Administrador, Docente, Coordinador, Directivo) con permisos específicos por módulo
Para...	controlar qué puede ver y hacer cada usuario en el sistema según su función institucional
Given (Dado que...)	Estoy autenticado como administrador.
When (Cuando...)	Accedo al módulo de gestión de usuarios, creo o modifico un rol y lo asigno a un usuario.
Then (Entonces...)	Los permisos del usuario cambian de inmediato. Si el usuario tiene sesión activa, los cambios se aplican en su próxima petición sin necesidad de cerrar sesión.

HU-006 — RF-006 | Notificaciones por bajo rendimiento

RF Asociado	RF-006 Notificaciones por bajo rendimiento
Prioridad MoSCoW	Should 5 pts

Estimación	5 puntos de historia
Microservicio	notification-service
Como...	coordinador
Quiero...	recibir notificaciones automáticas cuando un estudiante tenga nota inferior a 7/10 o ausentismo superior al 30%
Para...	intervenir oportunamente antes de que finalice el parcial y reducir el riesgo de reprobación
Given (Dado que...)	Hay un período académico activo. Existen notas o registros de asistencia cargados en el sistema.
When (Cuando...)	El sistema detecta que un estudiante supera el umbral configurado (nota < 7 o ausentismo > 30%).
Then (Entonces...)	El notification-service envía un correo electrónico al coordinador del paralelo afectado. El correo incluye datos del estudiante, asignatura e indicador que generó la alerta. El evento queda registrado en el log de notificaciones.

HU-007 — RF-007 Filtrado y búsqueda avanzada	
RF Asociado	RF-007 Filtrado y búsqueda avanzada
Prioridad MoSCoW	Should 3 pts
Estimación	3 puntos de historia
Microservicio	academic-service
Como...	coordinador o docente
Quiero...	filtrar los registros de notas y asistencia por carrera, materia, paralelo, período o nombre de estudiante
Para...	encontrar rápidamente la información requerida sin revisar listas completas
Given (Dado que...)	Estoy autenticado con rol de docente o coordinador. Accedo a cualquier listado de notas o asistencia.
When (Cuando...)	Selecciono uno o más criterios de filtro y los aplico.

Then (Entonces...)	La tabla se actualiza mostrando solo los registros que coinciden con los criterios. Puedo combinar múltiples filtros simultáneamente. Puedo limpiar todos los filtros con un solo botón.
---------------------------	---

HU-008 — RF-008 Generación de reportes PDF/Excel/CSV	
RF Asociado	RF-008 Generación de reportes PDF/Excel/CSV
Prioridad MoSCoW	Must 8 pts
Estimación	8 puntos de historia
Microservicio	report-service
Como...	coordinador
Quiero...	generar reportes consolidados de notas y asistencia en formato PDF, Excel o CSV
Para...	presentarlos en juntas académicas y utilizarlos en la carga a SIGA
Given (Dado que...)	Estoy autenticado como coordinador. Selecciono un período, carrera y paralelo específicos.
When (Cuando...)	Solicito la generación de un reporte en el formato deseado.
Then (Entonces...)	El report-service genera y entrega el archivo para descarga en menos de 10 segundos. El archivo tiene formato institucional correcto y datos consolidados del período seleccionado. El nombre del archivo incluye la fecha de generación.

HU-009 — RF-010 Validación pre-carga de archivos	
RF Asociado	RF-010 Validación pre-carga de archivos
Prioridad MoSCoW	Must 3 pts
Estimación	3 puntos de historia
Microservicio	sync-service
Como...	coordinador

Quiero...	que el sistema valide el archivo que subo antes de realizar cualquier importación
Para...	detectar errores de formato o datos faltantes sin modificar la base de datos si el archivo no es válido
Given (Dado que...)	Subo un archivo XLS al módulo de importación masiva.
When (Cuando...)	El sistema ejecuta la validación previa automáticamente.
Then (Entonces...)	Si hay errores, se muestra un listado con número de fila, campo con problema y descripción del error. No se realiza ningún cambio en la base de datos hasta que el archivo sea válido. Si el archivo es válido, se habilita el botón de confirmación de carga.

HU-010 — RF-012 Bloqueo por período cerrado	
RF Asociado	RF-012 Bloqueo por período cerrado
Prioridad MoSCoW	Must 2 pts
Estimación	2 puntos de historia
Microservicio	academic-service
Como...	sistema
Quiero...	bloquear automáticamente la edición de notas y asistencia cuando un período académico es marcado como cerrado
Para...	garantizar la integridad de los datos históricos y prevenir modificaciones no autorizadas
Given (Dado que...)	Un coordinador o administrador ha cerrado formalmente un período académico.
When (Cuando...)	Cualquier usuario intenta editar notas o asistencia correspondientes a ese período.
Then (Entonces...)	El sistema rechaza la operación con un mensaje claro indicando que el período está cerrado. El intento queda registrado en el log de auditoría con usuario, fecha y hora.

HU-011 — RF-013 Carga masiva de usuarios	
RF Asociado	RF-013 Carga masiva de usuarios
Prioridad MoSCoW	Should 5 pts
Estimación	5 puntos de historia
Microservicio	auth-service
Como...	administrador
Quiero...	importar masivamente usuarios desde un archivo Excel con la plantilla estándar del MRNA
Para...	registrar a todos los participantes de un período nuevo sin necesidad de ingreso individual
Given (Dado que...)	Tengo un archivo Excel con la plantilla de usuarios del MRNA. Estoy autenticado como administrador.
When (Cuando...)	Subo el archivo y confirmo la importación.
Then (Entonces...)	El sistema crea las cuentas de usuario y les asigna el rol correspondiente (docente o estudiante). Se muestra un reporte con registros creados, actualizados y errores por fila.

HU-012 — RF-014 Notificaciones programadas	
RF Asociado	RF-014 Notificaciones programadas
Prioridad MoSCoW	Should 5 pts
Estimación	5 puntos de historia
Microservicio	notification-service
Como...	administrador
Quiero...	configurar notificaciones automáticas con fecha, hora y destinatarios específicos
Para...	recordar a los docentes fechas importantes como el cierre de calificaciones sin hacerlo manualmente
Given (Dado que...)	Estoy autenticado como administrador. Accedo al módulo de notificaciones programadas.

When (Cuando...)	Configuro una notificación con mensaje, destinatarios (por rol o usuario) y fecha/hora de envío, y la guardo.
Then (Entonces...)	El notification-service encola la notificación y la envía automáticamente en el momento configurado. El envío queda registrado en el log con estado de entrega.

HU-013 — RF-015 Período de gracia 48 horas	
RF Asociado	RF-015 Período de gracia 48 horas
Prioridad MoSCoW	Should 3 pts
Estimación	3 puntos de historia
Microservicio	academic-service
Como...	coordinador
Quiero...	que el sistema permita modificar notas durante las 48 horas posteriores al cierre oficial del parcial
Para...	corregir errores evidentes sin necesidad de iniciar un proceso administrativo complejo
Given (Dado que...)	Un período académico ha sido cerrado. No han transcurrido más de 48 horas desde el cierre.
When (Cuando...)	Un docente o coordinador modifica una nota dentro de este período de gracia.
Then (Entonces...)	El sistema permite la modificación. La registra en el log de auditoría con marcador de 'período de gracia'. El log incluye usuario, fecha/hora y los valores anterior y nuevo de la nota.

HU-014 — RF-018 Auditoría de cambios	
RF Asociado	RF-018 Auditoría de cambios
Prioridad MoSCoW	Should 3 pts
Estimación	3 puntos de historia
Microservicio	academic-service

Como...	administrador o coordinador
Quiero...	que el sistema registre automáticamente cada cambio en notas o asistencia con información de quién lo hizo, cuándo y qué cambió
Para...	rastrear modificaciones y detectar inconsistencias en los registros académicos
Given (Dado que...)	Cualquier usuario autenticado modifica una nota o un registro de asistencia.
When (Cuando...)	El cambio se confirma y guarda en el sistema.
Then (Entonces...)	Se crea automáticamente un registro de auditoría con: usuario responsable, fecha y hora, campo modificado, valor anterior, valor nuevo e IP de origen. Este registro es visible para administradores y coordinadores en el módulo de historial.

RF Opcionales — 4 Historias de Usuario (Could Have)

Estas 4 historias se abordan únicamente si el margen de tiempo de la Cadencia lo permite.

No son bloqueantes

HU-015 — RF-019 Historial de versiones de nota (Opcional)	
RF Asociado	RF-019 Historial de versiones de nota (Opcional)
Prioridad MoSCoW	Could 3 pts
Estimación	3 puntos de historia
Microservicio	academic-service
Como...	coordinador
Quiero...	ver el historial completo de cambios de una nota específica a lo largo del tiempo
Para...	comparar versiones y entender el proceso de calificación de un estudiante
Given (Dado que...)	Estoy autenticado como coordinador. Accedo al detalle de la nota de un estudiante.
When (Cuando...)	Selecciono la opción 'ver historial'.
Then (Entonces...)	Se muestra una línea de tiempo con todos los cambios realizados a esa nota.

	Cada cambio incluye: usuario, fecha, valor anterior y valor nuevo.
--	--

HU-016 — RF-020 Feriados y eventos especiales (Opcional)	
RF Asociado	RF-020 Feriados y eventos especiales (Opcional)
Prioridad MoSCoW	Could 3 pts
Estimación	3 puntos de historia
Microservicio	academic-service
Como...	administrador
Quiero...	registrar feriados y eventos especiales en el sistema
Para...	que el cálculo de asistencia los excluya automáticamente sin afectar el porcentaje de los estudiantes
Given (Dado que...)	Estoy autenticado como administrador.
When (Cuando...)	Registro una fecha como feriado o evento especial con su descripción.
Then (Entonces...)	Esa fecha no se contabiliza en el cálculo de asistencia de ningún paralelo. Los docentes ven un indicador visual señalando el día como no lectivo en la vista de asistencia.

HU-017 — RF-021 Evaluación continua (Opcional)	
RF Asociado	RF-021 Evaluación continua (Opcional)
Prioridad MoSCoW	Could 5 pts
Estimación	5 puntos de historia
Microservicio	academic-service
Como...	docente
Quiero...	registrar calificaciones de actividades de evaluación continua (tareas, quizzes) directamente en el MRNA
Para...	tener un historial del desempeño de cada estudiante más allá de los parciales formales
Given (Dado que...)	Estoy autenticado como docente. Tengo paralelos asignados en el período activo.

When (Cuando...)	Registro una actividad con su nombre, fecha y calificación por estudiante.
Then (Entonces...)	La calificación queda registrada en el sistema. Es visible en el perfil del estudiante. Contribuye al componente de docencia configurado según las reglas del período.

HU-018 — RF-022 Calendario académico (Opcional)	
RF Asociado	RF-022 Calendario académico (Opcional)
Prioridad MoSCoW	Could 3 pts
Estimación	3 puntos de historia
Microservicio	academic-service
Como...	usuario autenticado con cualquier rol
Quiero...	ver un calendario con los eventos importantes del período académico activo
Para...	planificar actividades con anticipación y tener visibilidad del ciclo académico completo
Given (Dado que...)	Estoy autenticado en el sistema con cualquier rol.
When (Cuando...)	Accedo al módulo de calendario.
Then (Entonces...)	Veo una vista mensual con los eventos del período activo diferenciados por tipo (parcial, junta académica, feriado). Puedo navegar entre meses. Los feriados coinciden con los registrados por el administrador.

Anexo E. Matriz de Trazabilidad

Entrevista → RF → HU → Caso de Prueba → Resultado, para los catorce requerimientos funcionales del prototipo (Must-have y Should-have), construida a partir de la Tabla 4 (2.9.3) y la Tabla 17 (3.14.2) del cuerpo del documento.

RF	Origen (entrevistas)	HU	Caso de prueba	Resultado
RF-001	Responsable de TIC, Coordinador de Desarrollo de Software	HU-001	Generar el archivo de exportación de notas consolidadas de un período, carrera y paralelo, en el formato del SIGA	Cumple (3.14)
RF-002	Docente 1, Docente de Desarrollo de Software, Docente 3, Coordinador de Desarrollo de Software, Coordinador 1	HU-002	Sincronizar un curso de Moodle y contrastar sus calificaciones	Cumple (3.14)
RF-003	Docente 1, Docente 3, Coordinador de Desarrollo de Software	HU-003	Registrar los cuatro componentes de un parcial y revisar el promedio	Cumple (3.14)
RF-004	Vicerrector, Coordinador de Desarrollo de Software	HU-004	Generar el histograma de rendimiento de un paralelo	Cumple (3.14)
RF-005	Vicerrector, Coordinador de Desarrollo de Software, Coordinador 2	HU-005	Acceder a un recurso protegido con y sin el rol requerido	Cumple (3.14)
RF-006	Vicerrector, Coordinador 1	HU-006	Procesar un parcial con estudiantes bajo el umbral configurado	Cumple (3.14)
RF-007	Vicerrector, Coordinador de Desarrollo de Software	HU-007	Aplicar filtros combinados sobre las notas	Cumple (3.14)
RF-008	Coordinador de Desarrollo de Software, Coordinador 1, Vicerrector	HU-008	Descargar el reporte consolidado en los formatos disponibles	Cumple (3.14)
RF-010	Responsable de TIC, Coordinador de Desarrollo de Software	HU-009	Cargar un archivo con errores de estructura y contenido	Cumple (3.14)

RF-012	Docente de Desarrollo de Software, Docente 1, Coordinador 2	HU-010	Intentar modificar una nota en un período cerrado	Cumple (3.14)
RF-013	Responsable de TIC, Coordinador 2	HU-011	Importar un archivo de docentes con la plantilla del MRNA	Cumple (3.14)
RF-014	Vicerrector, Coordinador 2	HU-012	Ejecutar la notificación programada en la fecha configurada	Cumple (3.14)
RF-015	Coordinador 2	HU-013	Editar una nota dentro y fuera de la ventana de 48 horas	Cumple (3.14)
RF-018	Coordinador de Desarrollo de Software, Docente 1	HU-014	Crear, modificar y eliminar una nota y revisar la bitácora	Cumple (3.14)

Los cuatro requerimientos opcionales (RF-019 a RF-022) y los cinco diferidos (RF-009, RF-011, RF-016, RF-017, RF-023) no se incorporan a esta matriz individual porque, a diferencia de los catorce anteriores, no se derivaron de una entrevista puntual sino del análisis del catálogo consolidado de requerimientos, con el mismo criterio metodológico aplicado al catálogo de requerimientos no funcionales (2.9.4). Su seguimiento, alcance y condición de implementación se documentan en la sección 5.4

Anexo F. Diccionario de Datos

El esquema del prototipo MRNA consta de 36 tablas en PostgreSQL 15, gestionadas mediante migraciones versionadas de Flyway (V1 a V13,). La Tabla 10 del capítulo 3 (3.5.2) ya presenta, como ejemplo representativo, el diccionario columna por columna de la tabla núcleo nota; ese mismo formato y esas mismas convenciones se aplican a las 35 tablas restantes. El diagrama entidad-relación completo se presenta en el Anexo G.

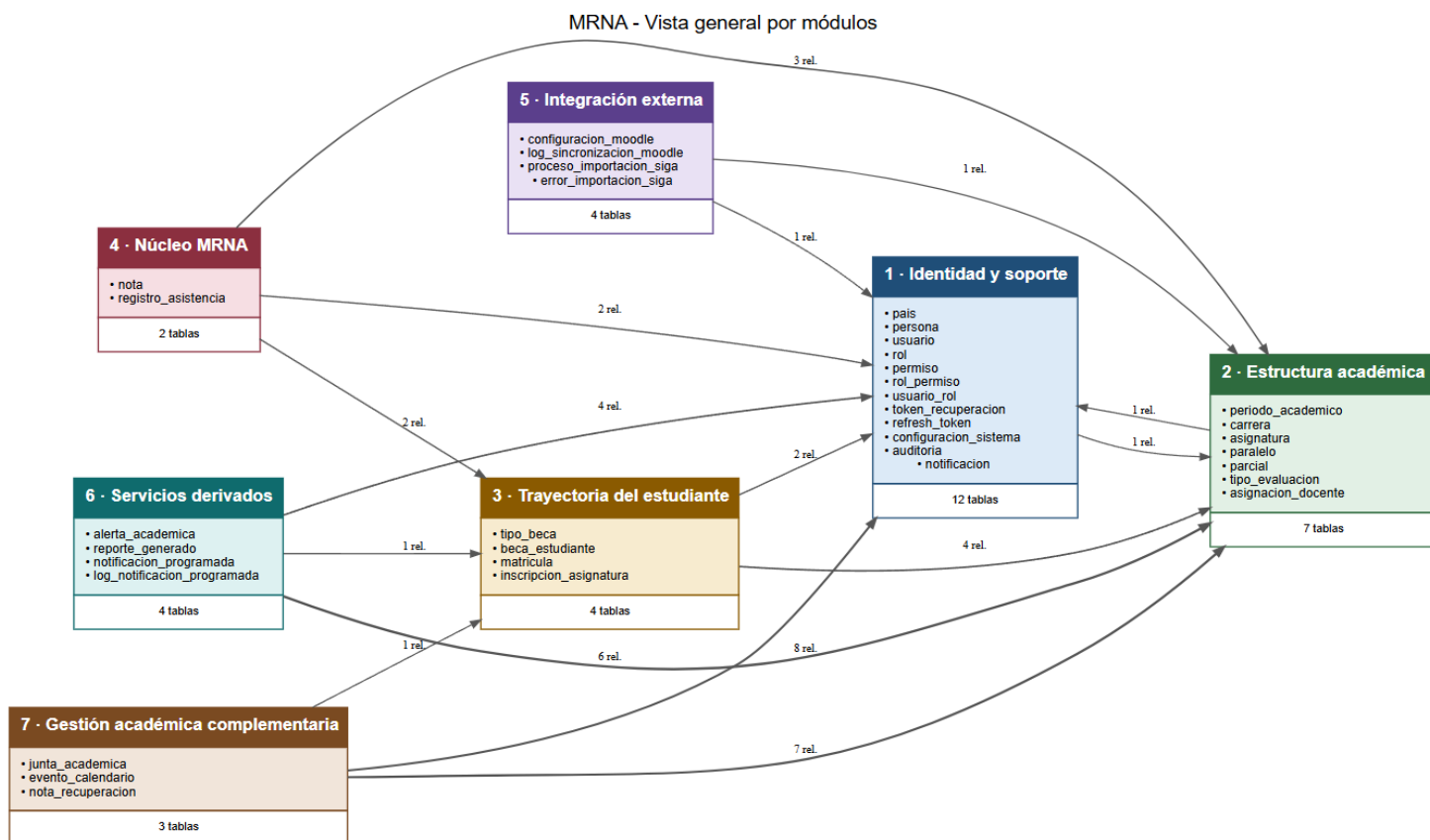
Grupo	Tabla	Propósito	# Campos	Llaves foráneas principales
1. Identidad y soporte	pais	Catálogo de países, usado como nacionalidad.	6	—
1. Identidad y soporte	persona	Datos personales únicos de cada individuo del sistema.	23	pais
1. Identidad y soporte	usuario	Cuenta de acceso con autenticación de dos factores.	19	persona
1. Identidad y soporte	rol	Roles del sistema (Administrador, Docente, Coordinador, Directivo).	10	usuario (auditoría)
1. Identidad y soporte	permiso	Permisos granulares por módulo, acción y recurso.	11	—
1. Identidad y soporte	rol_permiso	Asociación permiso-rol (RBAC).	5	rol, permiso
1. Identidad y soporte	usuario_rol	Asignación temporal de rol a usuario, con vigencia.	11	usuario, rol, periodo_academico
1. Identidad y soporte	token_recuperacion	Tokens de recuperación de contraseña.	9	usuario
1. Identidad y soporte	refresh_token	Refresh tokens del esquema JWT (ADR-002).	9	usuario
1. Identidad y soporte	configuracion_sistema	Parámetros globales (umbrales, frecuencias).	10	usuario (auditoría)
1. Identidad y soporte	auditoria	Bitácora de operaciones críticas por triggers PostgreSQL.	11	usuario
1. Identidad y soporte	notificacion	Notificaciones a usuarios por correo o push.	14	usuario
2. Estructura académica	periodo_academico	Período académico con fecha de cierre de notas.	10	—
2. Estructura académica	carrera	Carreras del ISTA.	8	—
2. Estructura académica	asignatura	Asignaturas por carrera.	9	carrera

2. Estructura académica	paralelo	Paralelo (curso) por asignatura y período; llave de sincronización con Moodle.	9	asignatura, periodo_academico
2. Estructura académica	parcial	Parciales del período, con fecha límite de registro de notas.	11	periodo_academico
2. Estructura académica	tipo_evaluacion	Componentes de evaluación de Moodle (DOC/PRA/AUT/EVA).	9	—
2. Estructura académica	asignacion_docente	Docente responsable de un paralelo.	12	persona, paralelo, periodo_academico, asignatura
3. Trayectoria del estudiante	tipo_beca	Catálogo de tipos de beca.	8	—
3. Trayectoria del estudiante	beca_estudiante	Beca asignada a un estudiante por período.	13	persona, periodo_academico, tipo_beca
3. Trayectoria del estudiante	matricula	Matrícula del estudiante por carrera y período.	11	persona, carrera, periodo_academico, beca_estudiante
3. Trayectoria del estudiante	inscripcion_asignatura	Inscripción del estudiante en un paralelo; llave de nota y asistencia.	7	matricula, paralelo
4. Núcleo MRNA	nota	Calificación por inscripción, parcial y componente. Ver Tabla 10 en §3.5.2 para el detalle columna por columna.	15	inscripcion_asignatura, parcial, tipo_evaluacion, usuario
4. Núcleo MRNA	registro_asistencia	Registro de asistencia por sesión.	13	inscripcion_asignatura, parcial, usuario
5. Integración externa	configuracion_moodle	Configuración de sincronización por paralelo.	10	paralelo
5. Integración externa	log_sincronizacion_moodle	Bitácora de sincronización con Moodle.	11	configuracion_moodle
5. Integración externa	proceso_importacion_siga	Carga masiva desde archivo (XLS de SIGA o usuarios).	13	usuario
5. Integración externa	error_importacion_siga	Errores detectados en la validación pre-carga.	9	proceso_importacion_siga
6. Servicios derivados	alerta_academica	Alerta por bajo rendimiento o ausentismo.	13	inscripcion_asignatura, parcial
6. Servicios derivados	reporte_generado	Reportes generados en PDF, Excel o CSV.	16	periodo_academico, carrera, paralelo, parcial, usuario

6. Servicios derivados	notificacion_programada	Notificaciones periódicas configurables.	29	rol, carrera, periodo_academico, parcial, usuario
6. Servicios derivados	log_notificacion_programada	Bitácora de ejecución de notificaciones programadas.	10	notificacion_programada
7. Gestión académica complementaria	junta_academica	Junta académica con cierre de modificaciones a +48h de gracia.	20	parcial, periodo_academico, carrera, paralelo, usuario
7. Gestión académica complementaria	evento_calendario	Calendario académico: feriados y eventos.	24	periodo_academico, carrera, paralelo, usuario
7. Gestión académica complementaria	nota_recuperacion	Nota de examen de recuperación por inscripción (relación 1:1).	7	inscripcion_asignatura, usuario

Anexo G. Diagrama Entidad-Relación





Anexo H. Architecture Decision Records (ADRs)

Se presenta una síntesis de cada uno de los seis ADR producidos durante el proyecto, con su contexto, decisión adoptada, alternativas descartadas, justificación central y consecuencias principales. Los documentos completos permanecen en el repositorio del proyecto como fuente de trazabilidad primaria.

ADR-001 — Adopción del Stack Tecnológico

Contexto: elegir un conjunto de herramientas seguro, con soporte a largo plazo, sin costo de licencia y compatible con Docker, para un proyecto académico con fecha de entrega fija.

Decisión: Spring Boot 3.2+ / Java 17 LTS en backend, React 19 + TypeScript + Vite en frontend, PostgreSQL 15, JWT + Spring Security para autenticación, Flyway para migraciones, TailwindCSS + shadcn/ui, Docker Compose y GitHub Actions.

Alternativas descartadas: Node.js + Angular (se prefirió Spring Security por recomendación explícita del responsable de TIC para el manejo de roles y tokens); Django + Vue.js (Python no forma parte del perfil curricular de la carrera); .NET 8 + Vue.js (sin ventaja técnica real, mayor curva de aprendizaje en el entorno académico ecuatoriano).

Justificación clave: Java 17 LTS tiene soporte garantizado hasta 2029; todo el stack es de código abierto, sin costo de licencia; el conjunto completo empaqueta en contenedores Docker para un entorno reproducible.

Ventajas: ecosistema bien documentado; TypeScript detecta errores en compilación; como riesgo controlado, si la institución exige a futuro un stack distinto para producción, el prototipo está explícitamente fuera de ese alcance.

ADR-002 — Adopción de Arquitectura de Microservicios

Contexto: construir autenticación, notas/asistencia, sincronización con Moodle/SIGA, reportes y notificaciones dentro de una sola aplicación generaría una pieza difícil de mantener, donde un fallo en un módulo (ej. reportes) podría tumbar el registro de notas.

Decisión: arquitectura de 6 microservicios independientes (api-gateway, auth-service, academic-service, sync-service, report-service, notification-service), desplegados como contenedores Docker orquestados con Docker Compose.

Alternativas descartadas: monolito (un fallo tumba todo el sistema; además el anteproyecto aprobado ya define microservicios como propuesta central); serverless (requiere infraestructura en la nube con costo, y el prototipo debe funcionar localmente con Docker).

Justificación clave: independencia entre módulos (el sync-service puede evolucionar sin tocar report-service ni academic-service); preparación para cuando SIGA exponga API pública, bastaría actualizar solo el sync-service.

Consecuencias: cada servicio se actualiza sin afectar a los demás; si sync-service falla, el resto del sistema sigue operando; como punto en contra, la configuración inicial es más compleja (6 contenedores en vez de uno), mitigada con Docker Compose.

ADR-003 — Estrategia de Integración: Moodle Local + Adaptador SIGA

Contexto: la API REST de Moodle existe pero está desactivada en el servidor institucional y no hay credenciales de administrador; SIGA es administrado por SENESCYT desde Quito, sin API pública ni acceso para terceros. Ambas limitaciones fueron confirmadas directamente por el responsable de TIC.

Decisión: instancia local de Moodle 4.1.14+ en Docker (con API REST activada y tokens propios para el prototipo) + adaptador para SIGA: modo archivo (genera el XLSX con el formato exacto que el coordinador sube manualmente al portal, es el que opera en el

prototipo) y modo API reservado para una integración futura (no conectado a ningún flujo ejecutable todavía).

Justificación clave: es la única estrategia técnicamente viable dado el contexto real del Instituto; no simula accesos que no existen; el modo archivo resuelve hoy la triple digitación manual (Moodle, Excel, SIGA) sin requerir cambios en los sistemas externos; si SENESCYT habilita una API pública, el adaptador ya delimita dónde debe implementarse esa integración, sin necesidad de rediseñar el resto del sistema.

Consecuencias: desarrollo autónomo sin depender de credenciales institucionales; la integración directa real queda fuera del alcance del prototipo (documentada como trabajo futuro en el capítulo 5).

ADR-004 — Adopción de Kanban como Metodología de Gestión

Contexto: proyecto de titulación individual, con fechas inamovibles.

Decisión: Kanban con cadencias de revisión, WIP máximo de 3 tareas en desarrollo y 3 en pruebas, y columna de Bloqueado con la razón documentada para cualquier tarea detenida por causa externa.

Por qué no Scrum: Scrum exige mínimo 3 roles diferenciados (este proyecto tiene uno); sus sprints cerrados no toleran bien los bloqueos institucionales; sus ceremonias fijas (daily, planning, review, retro) generarían sobrecarga sin equipo que coordinar.

Métricas de seguimiento: Lead Time, Cycle Time, Throughput, WIP efectivo y CFD, todas medibles con un solo desarrollador.

Ventajas: visibilidad inmediata del estado real del proyecto sin esperar cierre de sprint; el límite WIP=3 evita dispersión.

ADR-005 — Esquema Definitivo de Base de Datos del MRNA

Contexto: mientras se definía el alcance final del proyecto, se fueron generando varios borradores del diseño de la base de datos, cada uno reflejando una versión distinta de lo que el sistema iba a cubrir. Una vez que quedó claro exactamente qué requerimientos sí se iban a implementar, era necesario dejar fijada una sola versión oficial de la base de datos, la que realmente se iba a construir, antes de empezar a programar sobre ella.

Decisión: se define una única versión definitiva del esquema de base de datos, que contiene todas las tablas necesarias para soportar los requerimientos confirmados del proyecto, construida sobre PostgreSQL y gestionada con una herramienta que lleva un

historial ordenado de cada cambio que se le hace a la estructura con el tiempo, de modo que nada se modifica "a mano" ni se pierde el rastro de qué cambió y cuándo.

Alternativas descartadas: rediseñar la base de datos completamente desde cero, en vez de partir de lo ya avanzado. Se descartó porque ya existía un diseño sólido que cubría gran parte de lo necesario, y rehacerlo hubiera consumido tiempo valioso sin mejorar realmente el resultado.

Ventajas: el sistema cuenta con una base de datos completa y coherente que respalda todo lo que el proyecto necesita, con un registro claro de cómo se llegó a esa versión final. Como parte normal del proceso, la base de datos siguió creciendo de forma controlada más adelante, sumando algunas piezas puntuales que se identificaron ya durante el desarrollo (por ejemplo, para reforzar la seguridad de acceso y para cumplir una regla académica sobre exámenes de recuperación), siempre agregando, nunca reescribiendo lo que ya funcionaba.

ADR-006 — Mecanismo de Auditoría de Notas y Asistencia: Triggers PostgreSQL

Contexto: el sistema debe dejar un registro de quién modificó una nota o un registro de asistencia, cuándo, y cuál era el valor antes y después del cambio. Además, el tutor pidió que esta trazabilidad no dependiera únicamente del código de la aplicación, sino que estuviera garantizada directamente por la base de datos, así aunque alguien intentara alterar un dato, el registro del cambio quedaría igual. La implementación inicial solo cubría los intentos de edición que el sistema rechazaba (por ejemplo, cuando el período ya estaba cerrado), pero no capturaba el detalle de los cambios que sí se llegaban a guardar.

Decisión: se adopta un enfoque combinado. Cuando una nota o una asistencia se modifica o elimina de forma efectiva, es la propia base de datos la que registra automáticamente el cambio, de forma que ninguna operación pasa desapercibida, sin importar desde dónde se haya hecho. El componente que ya existía en la aplicación se conserva para seguir cubriendo los intentos que el sistema bloquea antes de guardarlos.

Alternativas descartadas: dejar la auditoría solo en manos de la aplicación (no cumplía con lo que el tutor pidió explícitamente, y no garantizaba el registro si alguien modificaba los datos por otra vía); dejarla solo en manos de la base de datos (funciona bien para los cambios que sí se completan, pero no alcanza a registrar los intentos que el sistema rechaza antes de guardarlos, porque ahí no llega a haber un cambio real que capturar).

Justificación clave: garantiza una bitácora que nadie puede alterar desde la aplicación, ni siquiera por error o de forma intencional; es una evidencia concreta y demostrable que el sistema protege la integridad de las calificaciones a nivel estructural y no de forma superficial.

Ventaja: se logra trazabilidad completa de cada cambio (quién, cuándo, y qué valores antes/después).

Anexo I. Tablero Kanban: Métricas y Capturas

El proyecto se gestionó en el tablero "MRNA - Tesis Kanban" sobre GitHub Projects, organizado en seis columnas (Backlog, Sprint Actual, En Desarrollo, En Pruebas, Terminado y Bloqueado), con límites de trabajo en curso de tres tarjetas en En Desarrollo y tres en En Pruebas. Cada tarjeta se clasificó con su etiqueta (requerimiento funcional, infraestructura, gestión, documentación o base de datos), su estimación en escala Fibonacci, el sprint en que se ejecutó y su estado.

Hasta el corte de documentación (cierre de la Cadencia 5) se trabajaron 87 tarjetas. La tabla que se presenta a continuación resume, por cadencia, las tarjetas terminadas, los puntos de estimación entregados y el lead time promedio, que son las métricas de flujo que sustentan la sección 4.5.

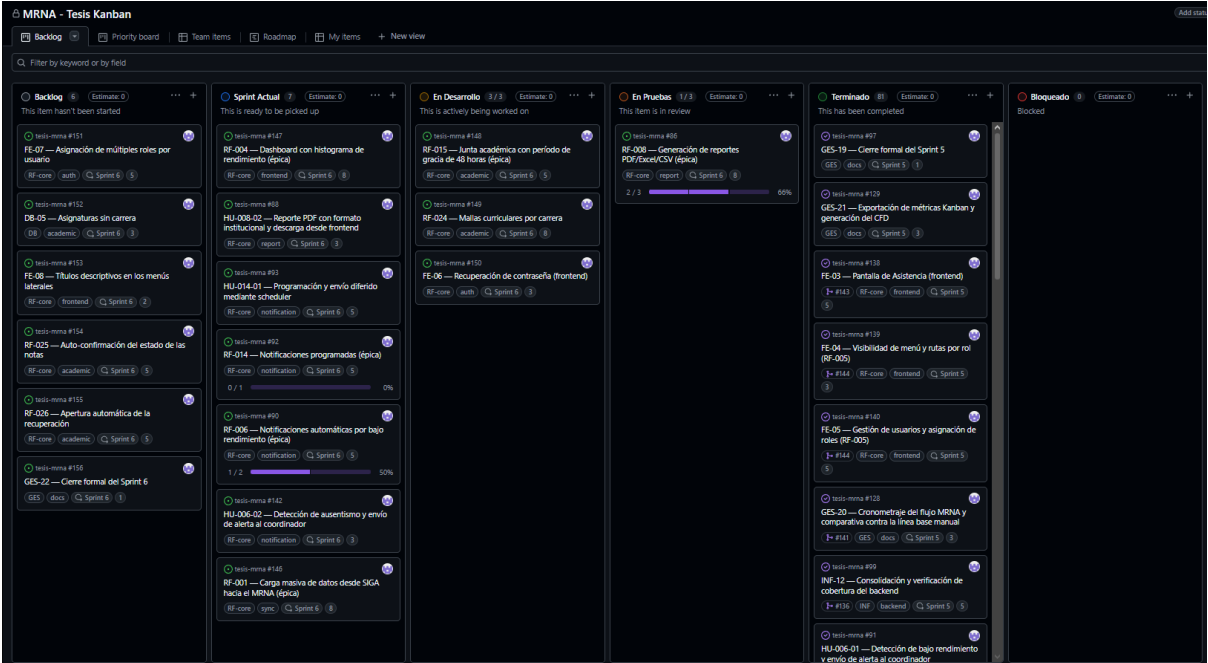
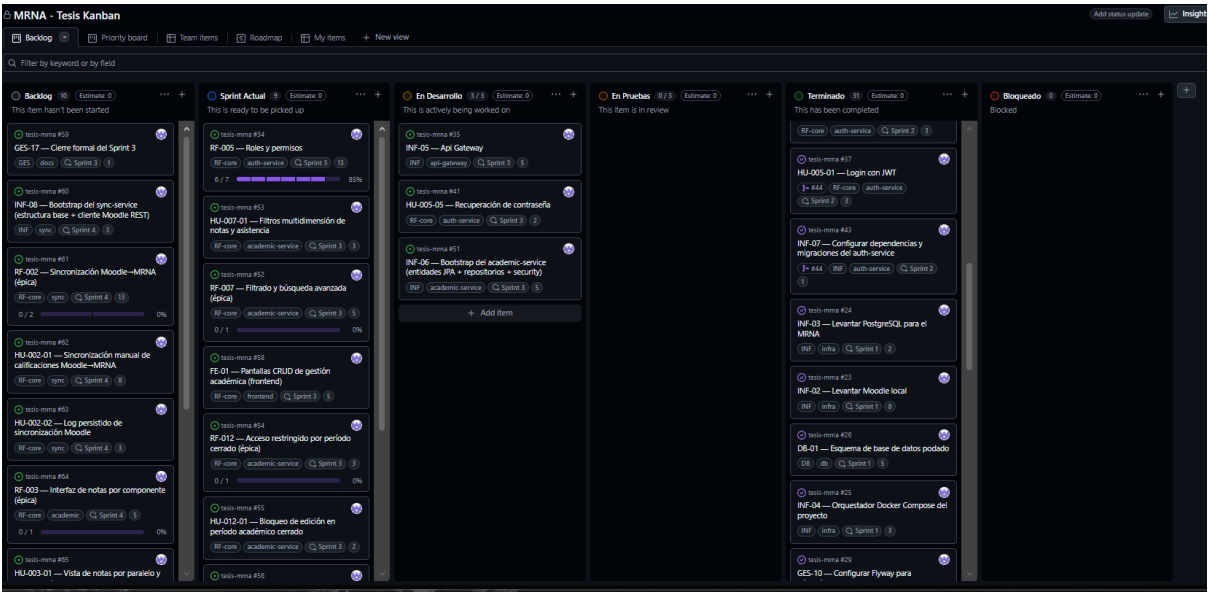
Cadencia	Tarjetas Terminadas	Puntos Entregados
Cadencia 1	12	39
Cadencia 2	10	28
Cadencia 3	17	73
Cadencia 4	20	73
Cadencia 5	11	45

A las 70 tarjetas anteriores se suman 9 tarjetas de configuración inicial sin cadencia asignada, para un total de 79 tarjetas terminadas. El lead time global (tiempo de entrega de una tarjeta desde el punto en el que se solicita) fue de 8,2 días y el WIP máximo observado de 21 tarjetas. El comportamiento acumulado del flujo se visualiza en el Diagrama de Flujo Acumulado presentado como Figura 17 en la sección 4.5.

La tabla siguiente muestra la distribución de las tarjetas por etiqueta.

Etiqueta	Tarjetas
Requerimientos funcionales (RF-core)	41
Infraestructura (INF)	17
Gestión (GES)	12
Documentación (DOC)	10
Base de Datos (BD)	6

Las imágenes que se muestran a continuación son capturas del tablero en dos momentos del proyecto, que evidencian el flujo de trabajo real sobre la metodología Kanban



Anexo J. API mediante Swagger

Cada uno de los servicios que conforman el backend del sistema genera automáticamente, al momento de encenderse, una especie de manual interactivo de todo lo que ese servicio puede hacer: qué información se puede consultar, qué se puede crear, qué se puede modificar. Esta herramienta se llama Swagger, y permite probar cada una de esas operaciones directamente desde el navegador, sin necesidad de escribir código ni usar herramientas adicionales, es la manera en la que, durante el desarrollo, se verificó que cada parte del sistema respondiera correctamente antes de conectarla con la interfaz visual (primero se verifico el backend y después su integración con el frontend).

Al levantar el sistema en el entorno de desarrollo, cada servicio abre su propia dirección web donde queda disponible este manual interactivo:

Servicio	Direccion para probarlo (levantado de manera local)
Autenticación y usuarios	http://localhost:8081/swagger-ui.html
Núcleo académico (notas, asistencia, periodos)	http://localhost:8082/swagger-ui.html
Sincronización con Moodle y SIGA	http://localhost:8083/swagger-ui.html
Generación de reportes	http://localhost:8084/swagger-ui.html
Notificaciones	http://localhost:8085/swagger-ui.html

Nota: estas direcciones solo funcionan mientras el sistema está corriendo en una computadora local, como corresponde a un prototipo académico ya que no son direcciones públicas accesibles.

Anexo K. Manual de Instalación del Prototipo

1. Requisitos previos

Docker Desktop instalado y corriendo, Git, y los siguientes puertos libres en la máquina local: 5432, 8080–8085, 8090, 5173.

2. Clonado del repositorio

- `git clone https://github.com/emmanuelbrito2003/tesis-mrna.git`
- `cd tesis-mrna`

3. Variables de entorno

Copiar la plantilla y completar los valores:

- `cp .env.example .env`

El archivo `.env` agrupa cinco bloques de configuración: credenciales de PostgreSQL (base de datos compartida MRNA), credenciales de MariaDB (base de datos exclusiva de Moodle), credenciales y token de la instancia local de Moodle, credenciales SMTP para envío de correo (recuperación de contraseña y notificaciones), y la URL base de la API para el frontend. El archivo `.env` nunca se sube al repositorio, ya que está excluido por `.gitignore`.

4. Levantar el sistema

El arranque es en dos pasos, porque Moodle necesita inicializarse antes que el resto:

- *# Paso 1 — infraestructura base (PostgreSQL + Moodle)*
- `docker-compose up postgres moodle-db moodle -d`

Moodle tarda entre 5 y 10 minutos en inicializar la primera vez. Verificar con `docker-compose logs -f moodle` hasta ver el mensaje "Apache started".

- *# Paso 2 — el resto de servicios, una vez Moodle esté listo*
- `docker-compose up -d`

auth-service ejecuta automáticamente las migraciones Flyway (V1 a V13) al arrancar, creando el esquema completo de `mrna_db`.

5. Verificación

- `docker-compose ps`

Servicio	Puerto	Estado esperado
mrna-postgres	5432	healthy
mrna-moodle-db	—	healthy
mrna-moodle	8090	running
mrna-auth	8081	running
mrna-academic	8082	running
mrna-sync	8083	running
mrna-report	8084	running

mrna-notification	8085	running
mrna-gateway	8080	running
mrna-frontend	5173	running

Los contenedores de PostgreSQL y de la base de datos de Moodle tienen verificación automática de salud (healthcheck) configurada en Docker; deben marcar healthy antes de levantar el resto del stack.

Configuración adicional de un solo uso, para habilitar la API REST de Moodle: entrar a <http://localhost:8090> como admin, ir a Administración del sitio → Plugins → Servicios web, activar servicios web y protocolo REST, crear un servicio personalizado con las funciones `core_course_get_courses`, `core_grades_get_grades` y `core_enrol_get_enrolled_users`, generar el token y copiarlo a MOODLE_TOKEN en el `.env`.

6. Limpieza

- *`docker-compose down` # detiene los contenedores, conserva los datos*
- *`docker-compose down -v` # detiene y borra todos los datos (reinicio completo)*

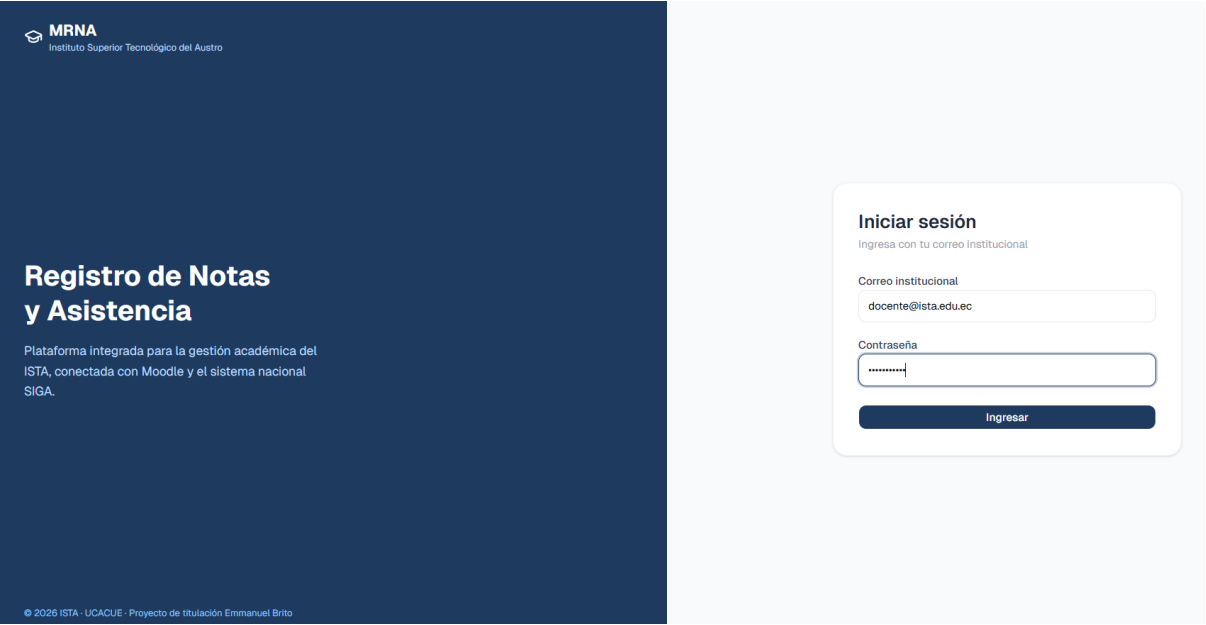
7. Resolución de problemas comunes

Problema	Causa probable	Solución
Puerto ocupado al levantar un contenedor	Otro proceso ya usa ese puerto (5432, 8080-8085, 8090 o 5173)	Liberar el puerto o detener el servicio que lo ocupa antes de <code>docker-compose up</code>
auth-service falla al arrancar por migración Flyway	Base de datos no lista todavía, o migración aplicada parcialmente en un intento anterior	Verificar que <code>mrna-postgres</code> esté healthy antes de levantar el resto; si el esquema quedó a medias, usar <code>docker-compose down -v</code> y volver a levantar desde cero
Moodle no responde tras varios minutos	Primera inicialización aún en curso	Esperar el mensaje "Apache started" en <code>docker-compose logs -f moodle</code> antes de continuar
Token de Moodle inválido en <code>sync-service</code>	MOODLE_TOKEN no configurado o generado antes de crear el servicio de Web Services	Regenerar el token siguiendo el paso de configuración de Moodle y actualizar el <code>.env</code>

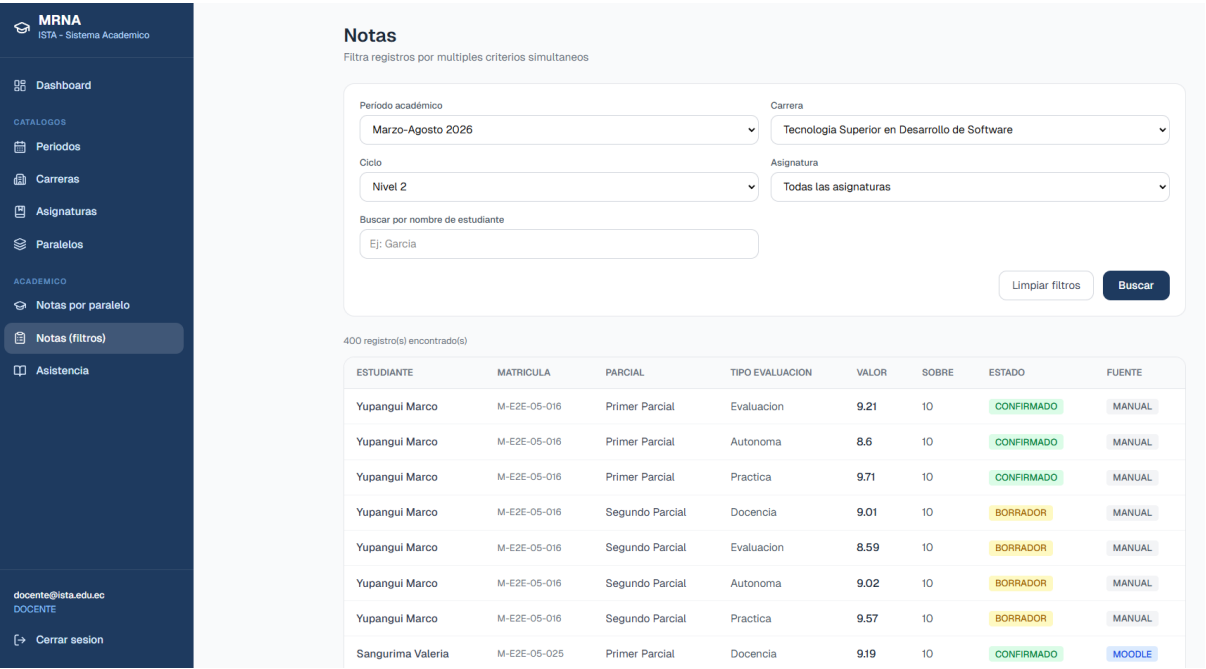
Anexo L. Manual de Usuario por Rol

- Manual del Docente

Login general al sistema.



Pantalla de consulta de notas del docente, con filtros combinables por carrera, ciclo, asignatura y estudiante.



Vista de notas por paralelo: una fila por estudiante con sus componentes de evaluación (Docencia, Práctica, Autónoma, Evaluación) y el promedio del parcial.

MRNA

ISTA - Sistema Académico

Dashboard

CATALOGOS

Periodos

Carreras

Asignaturas

Paralelos

ACADEMICO

Notas por paralelo

Notas (filtros)

Asistencia

docente@ista.edu.ec

DOCENTE

Cerrar sesion

Notas por paralelo

Selecciona periodo y filtra por carrera, ciclo y paralelo; el parcial puede ser el activo o uno anterior

Periodo académico

Marzo-Agosto 2026

Carrera

Tecnología Superior en Desarrollo de Software

Ciclo

Todos los ciclos

Paralelo

Fundamentos de Programación — A

Parcial

Parcial activo (por defecto)

Carrera: Tecnología Superior en Desarrollo de Software

Asignatura: Fundamentos de Programación (A)

Ciclo: Nivel 2

Docente: Sanmartín Víctor Eduardo

25 estudiante(s) - Segundo Parcial

ESTUDIANTE	MATRÍCULA	DOCENCIA	PRÁCTICA	AUTÓNOMA	EVALUACIÓN	PROMEDIO	RECUPERACIÓN
Brito Paola	M-E2E-05-007	6.41	6.41	6.53	5.62	6.29	—
Castro Gabriela	M-E2E-05-013	6.65	7.52	6.59	6.81	6.93	—
Chuva Diego	M-E2E-05-006	5.16	5.79	5.30	5.41	5.44	—
Cárdenas Tatiana	M-E2E-05-021	9.57	9.63	9.93	9.16	9.61	—
Flores Christian	M-E2E-05-014	7.31	7.38	7.51	7.78	7.49	—
Guamán Bryan	M-E2E-05-012	9.98	9.80	9.16	8.79	9.44	—
Gómez José	M-E2E-05-002	5.40	4.70	4.41	4.43	4.70	8.84
Lema Kevin	M-E2E-05-010	7.95	8.15	7.61	8.20	7.96	—
López Ana	M-E2E-05-003	3.26	2.33	3.25	3.40	3.01	5.49

Pantalla de asistencia con filtros por paralelo y parcial, y resumen estadístico de presencia del grupo.

MRNA

ISTA - Sistema Académico

Dashboard

CATALOGOS

Periodos

Carreras

Asignaturas

Paralelos

ACADEMICO

Notas por paralelo

Notas (filtros)

Asistencia

docente@ista.edu.ec

DOCENTE

Cerrar sesion

Asistencia

Consulta y edición de registros de asistencia

Periodo académico

Marzo-Agosto 2026

Carrera

Tecnología Superior en Desarrollo de Software

Ciclo

Nivel 2

Paralelo

Todos los paralelos

Parcial

Todos los parciales

Estado

Todos los estados

Fecha desde

dd/mm/aaaa

Fecha hasta

dd/mm/aaaa

Limpiar filtros

Buscar

Total registros

4800

% Asistencia

95%

4548 asistieron (pres.+tard.+just.)

Ausencias

5%

252 registros

Tardanzas / Justificados

600

600 tard. - 0 just.

4800 registro(s) encontrado(s)

ESTUDIANTE	MATRÍCULA	FECHA	PARCIAL	ESTADO	TARDANZA (MIN)	JUSTIFICACIÓN	ACCIONES
Castro Gabriela	M-E2E-11-013	2026-03-10	Primer Parcial	PRESENTE	0	—	Editar
Castro Gabriela	M-E2E-11-013	2026-03-17	Primer Parcial	PRESENTE	0	—	Editar
Castro Gabriela	M-E2E-11-013	2026-03-24	Primer Parcial	PRESENTE	0	—	Editar

Modal de edición de un registro de asistencia individual, con justificación para tardanzas o inasistencias.

MRNA

ISTA - Sistema Academico

Dashboard

CATALOGOS

- Periodos
- Carreras
- Asignaturas
- Paralelos

ACADEMICO

- Notas por paralelo
- Notas (filtros)
- Asistencia**

4800

4800 registro(s) encontrado(s)

95%
 4548 asistieron (pres.+tard.+just.)

5%
 252 registros

ESTUDIANTE	MATRÍCULA	FECHA	PARCIAL	ESTADO	TARDANZA
Pérez María	M-E2E-01-001	2026-03-10	Primer Parcial	AUSENTE	0
Pérez María	M-E2E-01-001	2026-04-28	Primer Parcial	AUSENTE	0

Editar asistencia
✕

Pérez María

M-E2E-01-001 · 2026-04-28 — Primer Parcial

Estado

TARDANZA ▾

Minutos de tardanza

10

Justificación

Describe la razón...

Cancelar

Guardar

MRNA
ISTA - Sistema Academico

Dashboard

CATALOGOS

Periodos

Carreras

Asignaturas

Paralelos

ACADEMICO

Notas por paralelo

Notas (filtros)

Asistencia

Importar notas SIGA

Descargar reporte

coordinador@ista.edu.ec
COORDINADOR

Cerrar sesion

Validación e importación SIGA

Cargue el archivo XLS exportado desde SIGA. El proceso tiene dos pasos: primero se valida la estructura y luego se importan las notas de los estudiantes al sistema.

1. Validar archivo
Revisa columnas y datos (cédulas, notas, tipos de registro) y muestra los errores.
No guarda nada en el sistema.

2. Importar definitivamente
Guarda en el sistema las notas de los estudiantes (filas tipo NOTA) con fuente SIGA. Solo se habilita si la validación no tuvo errores.

↑

sig_a_import_demo_valido.xlsx

Validar archivo

Importar definitivamente

El archivo es válido y puede importarse

15
Total de filas

5
Estudiantes

0
Docentes

5
Notas

5
Asistencia

Resultado de validar un archivo SIGA con errores: detalle fila por fila de las inconsistencias encontradas.

MRNA
ISTA - Sistema Academico

Dashboard

CATALOGOS

Periodos

Carreras

Asignaturas

Paralelos

ACADEMICO

Notas por paralelo

Notas (filtros)

Asistencia

Importar notas SIGA

Descargar reporte

coordinador@ista.edu.ec
COORDINADOR

Validación e importación SIGA

Cargue el archivo XLS exportado desde SIGA. El proceso tiene dos pasos: primero se valida la estructura y luego se importan las notas de los estudiantes al sistema.

1. Validar archivo
Revisa columnas y datos (cédulas, notas, tipos de registro) y muestra los errores.
No guarda nada en el sistema.

2. Importar definitivamente
Guarda en el sistema las notas de los estudiantes (filas tipo NOTA) con fuente SIGA. Solo se habilita si la validación no tuvo errores.

↑

sig_a_import_demo_con_errores.xlsx

Validar archivo

Importar definitivamente

Se encontraron 5 errores en el archivo

Fila	Columna	Descripción del error
2	CEDULA	La cédula debe tener exactamente 10 dígitos numéricos. Valor recibido: '123'
2	NOTA	La nota es obligatoria para registros de tipo NOTA
3	NOMBRES	El campo NOMBRES es obligatorio
4	NOTA	La nota debe estar entre 0.00 y 10.00. Valor recibido: 15
5	TIPO_REGISTRO	Valor inválido: 'XYZ'. Valores aceptados: ESTUDIANTE, DOCENTE, NOTA, ASISTENCIA

Confirmación de importación SIGA: notas procesadas, creadas y actualizadas en el sistema

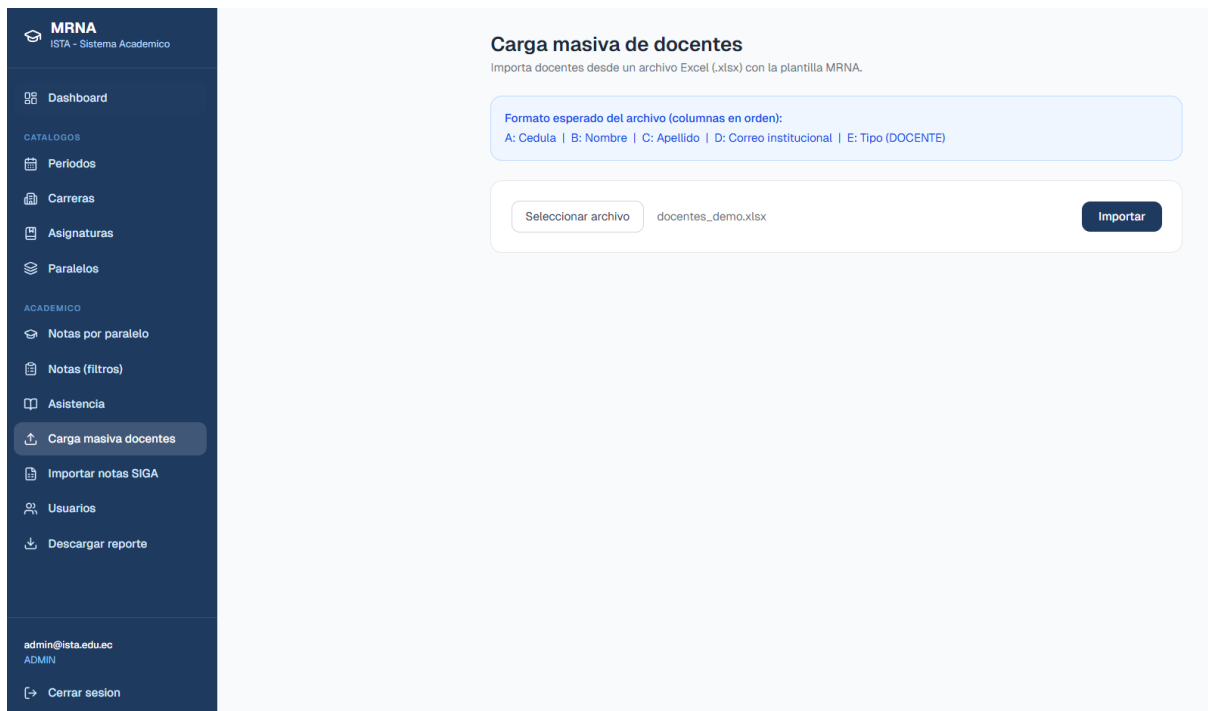
The screenshot shows the 'Importación SIGA' confirmation page. On the left is a dark blue sidebar with the 'ACADEMICO' header and a menu containing: 'Notas por paralelo', 'Notas (filtros)', 'Asistencia', 'Importar notas SIGA' (highlighted), and 'Descargar reporte'. At the bottom of the sidebar, it shows the user 'coordinador@ista.edu.ec' with the role 'COORDINADOR' and a 'Cerrar sesión' link. The main content area has a light blue header with two buttons: 'Validar archivo' and 'Importar definitivamente'. Below this, a green checkmark indicates 'Importación completada'. A summary row shows four green boxes: 'Notas procesadas' (0), 'Creadas' (0), 'Actualizadas' (0), and 'Filas omitidas' (15). A yellow box contains a warning: '5 advertencia(s)' followed by a list of five rows (3, 6, 9, 12, 15) where the 'paralelo' was not found for the 'Marzo-Agosto 2026' period in the 'B' group. At the bottom, a small text note says: 'Para verificar, ve a Notas (filtros) y busca registros con fuente SIGA.'

Pantalla de descarga de reportes, con la opción de exportación del archivo consolidado hacia el SIGA.

The screenshot shows the 'Descargar reporte' screen. It has a header with a report icon and the title 'Descargar reporte', with a subtitle 'Reporte consolidado de notas y asistencia por paralelo'. Below this are four dropdown menus for selection: 'Periodo academico *' (Marzo-Agosto 2026), 'Carrera *' (Tecnología Superior en Desarrollo de Software), 'Ciclo' (Nivel 2), and 'Paralelo *' (B — Fundamentos de Programación). Below the dropdowns are radio buttons for 'Formato': 'Excel (.xlsx)' (selected) and 'CSV (.csv)'. A dark blue button with a download icon and the text 'Descargar reporte' is positioned below the format options. At the bottom, there is a section titled 'Exportar a SIGA' with a description: 'Genera el archivo XLS con la nota final consolidada de cada estudiante del periodo, carrera y paralelo seleccionados arriba, listo para cargar manualmente en SIGA.' Below this description is a button with a document icon and the text 'Exportar a SIGA'.

- Manual del Administrador

Pantalla de carga masiva de docentes: formato de archivo esperado y selección del Excel a importar.



Modal de confirmación previo a la carga masiva, con las reglas de deduplicación por cédula.



Resultado de la carga masiva: total de filas procesadas, usuarios creados y actualizados.

Carga masiva de docentes

Importa docentes desde un archivo Excel (.xlsx) con la plantilla MRNA.

Formato esperado del archivo (columnas en orden):

A: Cedula | B: Nombre | C: Apellido | D: Correo institucional | E: Tipo (DOCENTE)

Seleccionar archivo

docentes_demo.xlsx

Importar

5

Total filas

5

Usuarios creados

0

Usuarios actualizados

Todas las filas procesadas sin errores.

Listado general de usuarios del sistema, con sus roles activos y estado de la cuenta.

MRNA

ISTA - Sistema Academico

Dashboard

CATALOGOS

Periodos

Carreras

Asignaturas

Paralelos

ACADEMICO

Notas por paralelo

Notas (filtros)

Asistencia

Carga masiva docentes

Importar notas SIGA

Usuarios

Descargar reporte

admin@ista.edu.ec

ADMIN

Cerrar sesion

Usuarios

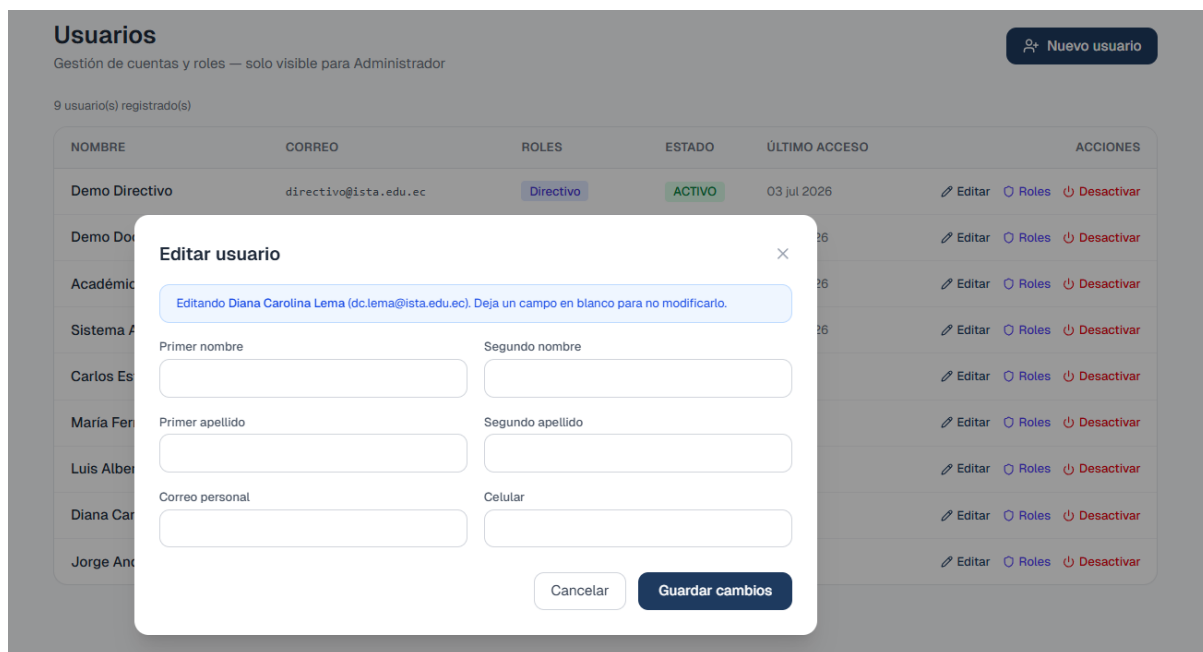
Gestión de cuentas y roles — solo visible para Administrador

Nuevo usuario

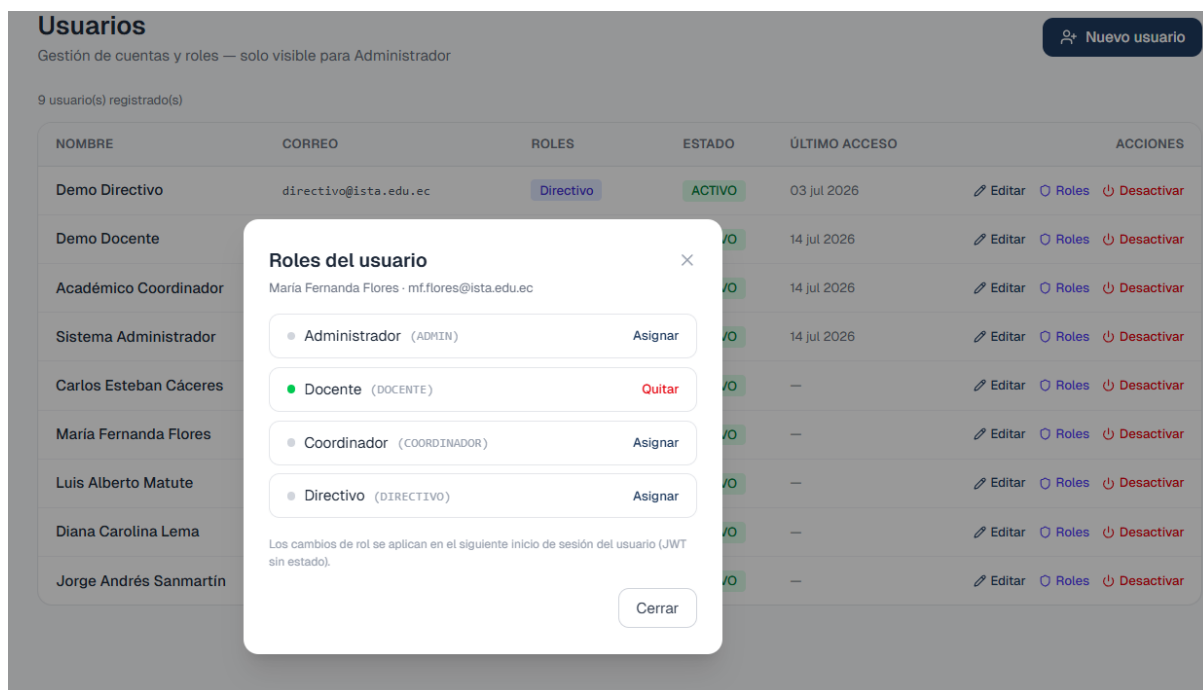
9 usuario(s) registrado(s)

NOMBRE	CORREO	ROLES	ESTADO	ÚLTIMO ACCESO	ACCIONES
Demo Directivo	directivo@ista.edu.ec	Directivo	ACTIVO	03 jul 2026	Editar Roles Desactivar
Demo Docente	docente@ista.edu.ec	Docente	ACTIVO	14 jul 2026	Editar Roles Desactivar
Académico Coordinador	coordinador@ista.edu.ec	Coordinador	ACTIVO	14 jul 2026	Editar Roles Desactivar
Sistema Administrador	admin@ista.edu.ec	Administrador	ACTIVO	14 jul 2026	Editar Roles Desactivar
Carlos Esteban Cáceres	esteban.caceres@ista.edu.ec	Docente	ACTIVO	—	Editar Roles Desactivar
Maria Fernanda Flores	mf.flores@ista.edu.ec	Docente	ACTIVO	—	Editar Roles Desactivar
Luis Alberto Matute	la.matute@ista.edu.ec	Docente	ACTIVO	—	Editar Roles Desactivar
Diana Carolina Lema	dc.lema@ista.edu.ec	Docente	ACTIVO	—	Editar Roles Desactivar
Jorge Andrés Sanmartín	ja.sanmartin@ista.edu.ec	Docente	ACTIVO	—	Editar Roles Desactivar

Modal de edición de datos personales de un usuario existente



Modal de gestión de roles: asignación o remoción de roles del sistema para un usuario.



Para las categorías de “CATALOGOS”, el rol de Administrador tiene la potestad de agregar campos a cualquiera de los 4 módulos (acción que los otros roles solo pueden visualizar)

MRNA

ISTA - Sistema Académico

Dashboard

CATALOGOS

Períodos

Carreras

Asignaturas

Paralelos

ACADEMICO

Notas por paralelo

Notas (filtros)

Asistencia

Carga masiva docentes

Importar notas SIGA

Usuarios

Descargar reporte

admin@ista.edu.ec

ADMIN

Cerrar sesión

Asignaturas

Materias por carrera

Todas las carreras

CÓDIGO	NOMBRE	CARRERA	CRÉDITOS	NIVEL	ESTADO	
ISW-301	Desarrollo de Aplicaciones Web	Tecnología Superior en Desarrollo de Software	4	3	ACTIVO	Editar
ISW-302	Bases de Datos Avanzadas	Tecnología Superior en Desarrollo de Software	3	3	ACTIVO	Editar
DS-AC-B-OF-101	Ofimática	Tecnología Superior en Desarrollo de Software	3	1	ACTIVO	Editar
DS-AC-B-MAT-102	Matemática Discreta	Tecnología Superior en Desarrollo de Software	4	1	ACTIVO	Editar
DS-AC-P-FP-103	Fundamentos de Programación	Tecnología Superior en Desarrollo de Software	4	2	ACTIVO	Editar
DS-AC-P-ADS-104	Análisis y Diseño de Sistemas	Tecnología Superior en Desarrollo de Software	3	3	ACTIVO	Editar
DS-AC-P-BD-105	Base de Datos	Tecnología Superior en Desarrollo de Software	4	3	ACTIVO	Editar

Nueva asignatura

- Manual del Directivo

Dashboard general del directivo

MRNA

ISTA - Sistema Académico

Dashboard

CATALOGOS

Períodos

Carreras

Asignaturas

Paralelos

ACADEMICO

Notas por paralelo

Notas (filtros)

Asistencia

Descargar reporta

directivo@ista.edu.ec

DIRECTIVO

Cerrar sesión

Dashboard

Bienvenido al Sistema MRNA del ISTA.

Promedio institucional

7.8/10

% Asistencia global

91%

Estudiantes en riesgo

34

Paralelos activos

12

Promedio de notas por Carrera

Ing. de Software	7.8
Biotechnología	6.7
Administración	5.6

Tendencia de Asistencia por Parcial

Paralelos con mayor tasa de reprobación

Asignatura	Paralelo	% Reprobación
Ing. de Software	3-1	8.0%
Biotechnología	3-2	9.2%
Administración	3-4	8.0%

Pantalla de descarga de reportes, accesible también para el rol Directivo.

MRNA

ISTA - Sistema Académico

Dashboard

CATALOGOS

Periodos

Carreras

Asignaturas

Paralelos

ACADEMICO

Notas por paralelo

Notas (filtros)

Asistencia

Descargar reporte

directivo@ista.edu.ec

DIRECTIVO

Cerrar sesion



Descargar reporte

Reporte consolidado de notas y asistencia por paralelo

Periodo academico *

Marzo-Agosto 2026

Carrera *

Tecnologia Superior en Desarrollo de Software

Ciclo

Todos los ciclos

Paralelo *

Selecciona un paralelo

Formato

☒ Excel (.xlsx) ☐ CSV (.csv)

 Descargar reporte

Anexo M. Resultados de Prueba SUS

Datos crudos de la prueba de usabilidad aplicada al evaluador experto (docente del área de Desarrollo de Software del ISTA, 13 años de experiencia docente, gestiona actualmente las notas en el Excel institucional).

Respuesta registrada en cada ítem del cuestionario SUS

(Teniendo en cuenta que se utilizó una escala del 1 al 5, en donde 1 es totalmente en desacuerdo y 5 totalmente de acuerdo)

N. º	Afirmación	Respuesta (1-5)
1	Creo que me gustaría usar este sistema con frecuencia.	5
2	Encuentro el sistema innecesariamente complejo.	1
3	Creo que el sistema es fácil de usar.	4
4	Creo que necesitaría el apoyo de un técnico para poder usar este sistema.	2
5	Encuentro que las diversas funciones del sistema están bien integradas.	4
6	Creo que el sistema es demasiado inconsistente.	1
7	Imagino que la mayoría de la gente aprendería a usar este sistema rápidamente.	5
8	Encuentro el sistema muy incómodo de usar.	1
9	Me siento muy seguro usando el sistema.	4
10	Necesité aprender muchas cosas antes de poder manejar el sistema.	2

Cálculo paso a paso del puntaje SUS

Ítems impares (1, 3, 5, 7, 9) — puntaje = respuesta – 1:

$$(5-1) + (4-1) + (4-1) + (5-1) + (4-1) = 4 + 3 + 3 + 4 + 3 = 17$$

Ítems pares (2, 4, 6, 8, 10) — puntaje = 5 – respuesta:

$$(5-1) + (5-2) + (5-1) + (5-1) + (5-2) = 4 + 3 + 4 + 4 + 3 = 18$$

$$\text{Suma total: } 17 + 18 = 35$$

$$\text{Puntaje SUS: } 35 \times 2,5 = 87,5 / 100$$

Según estudios en donde se aplican cuestionarios SUS, un puntaje superior a 80 se considera excelente, y el resultado obtenido supera ese umbral.

Observaciones registradas durante las ocho tareas guiadas del cuestionario

N.º	Tarea	¿Completada?	Observación del encuestado
1	Iniciar sesión y navegar el menú principal.	Sí	Detectó necesidad de recuperar contraseña.
2	Consultar las notas de un paralelo y ubicar el promedio ponderado de un estudiante.	Sí	Sugirió seleccionar carrera y organizar por parciales.
3	Consultar la asistencia de un paralelo y justificar una falta.	Sí	Observación sobre el encabezado de la tabla de datos.
4	Filtrar las notas por período, carrera o asignatura, o buscar por nombre de estudiante.	Sí	Sugirió encabezados adicionales (recuperación, aprueba/reprueba).
5	Crear o editar un registro de catálogo (paralelo o asignatura).	Sí	Sugirió poder configurar mallas por carrera.
6	Cargar masivamente docentes desde un archivo Excel.	Sí	Sugirió presentar alerta y evitar usuarios duplicados.
7	Validar un archivo antes de enviarlo al SIGA y observar cómo el sistema detecta y bloquea errores.	Sí	Sugirió títulos más claros para las acciones que realiza el módulo.
8	Generar y descargar el reporte consolidado de un paralelo en formato Excel.	Sí	Observación sobre el encabezado del reporte.

Retroalimentación cualitativa adicional del evaluador:

- *Ventajas frente al proceso manual (Moodle → Excel → SIGA):* "Velocidad en la consulta de información" y "Datos mejor organizados".
- *Funciones más útiles:* "Importación de datos y reportes con filtros".
- *¿Hubo algún paso confuso o propenso a error?:* "No."
- *¿Es viable su adopción en el Instituto?:* "Es viable su implementación en el momento que el Instituto cuente con la infraestructura tecnológica necesaria para su funcionamiento."

- *¿Qué le ahorraría más tiempo comparado con el Excel actual?:* "La carga de datos consolidados, la generación de reportes, tiempo de respuesta en la consulta de notas."
- *¿Qué fue lo más claro y lo más confuso de la interfaz?:* "Todo estaba claro y bien organizado."

Mejoras propuestas por el evaluador

1. Gestión de múltiples roles por usuario: "La gestión de doble rol para usuarios, pues un docente puede ejercer el rol de Coordinador de Carrera."
2. Mallas curriculares por carrera: "Gestión de roles, manejo de mallas vigentes por carrera."
3. Organización de la información por período, carrera, ciclo y consulta por parcial: "Consulta de notas por Parcial I y Parcial II"; y en comentarios finales: "Incluir la organización de información por período académico, por carrera, por ciclo o nivel; asignaturas presentando el nombre del profesor a cargo."

Estas tres mejoras son consistentes con lo ya resumido en la sección 4.2.3 y retomado como trabajo futuro en el 5.4.

- *Se adjuntan a continuación capturas del documento físico completo del cuestionario SUS completo*

PROTOCOLO DE VALIDACIÓN DE USABILIDAD DEL PROTOTIPO MRNA

Instrumento de evaluación experta — Anexo D / DOC-11

Proyecto	Sistema de Registro de Notas y Asistencia (MRNA) — Instituto Superior Tecnológico del Austro
Autor	Nelson Emmanuel Brito Ordóñez — Ingeniería de Software, UCACUE
Cargo del evaluador	Docente
Años de experiencia docente	13 años
¿Gestiona hoy las notas en el Excel institucional? (Sí/No)	Sí
Fecha de aplicación	

1. Presentación

El presente instrumento forma parte de la validación del prototipo MRNA, un sistema de registro de notas y asistencia con arquitectura de microservicios que automatiza el flujo de calificaciones entre Moodle y el SIGA, eliminando la triple digitación manual que hoy realiza el Instituto. La sesión consiste en que usted, como evaluador experto, realice un conjunto de tareas guiadas sobre el sistema real, responda un cuestionario estandarizado de usabilidad (System Usability Scale, SUS) y aporte observaciones cualitativas. La participación es voluntaria y dura aproximadamente entre 30 y 45 minutos.

2. Consentimiento informado

Declaro que participo de forma voluntaria en la evaluación del prototipo MRNA. Entiendo que la información que proporcione se utilizará únicamente con fines académicos, dentro del trabajo de titulación, y que mis datos personales se tratarán de forma anonimizada (se reportará mi cargo, no mi nombre).

Cargo / del evaluador:

Firma: 

Fecha: ___ / ___ / 2026

3. Guion de tareas guiadas

Realice cada tarea en el sistema. El facilitador registra si la completó y cualquier observación.

N°	Tarea	¿La completó? (Sí/No)	Observaciones
1	Iniciar sesión en el sistema y navegar el menú principal.	(No) Si	Necesidad de. Recuperar Cont.
2	Consultar las notas de un paralelo y ubicar el promedio ponderado de un estudiante.	Si	Seleccionar Carrera, por Parciales
3	Consultar la asistencia de un paralelo y	Si	Datos encabezado (x)

	justificar una falta de un estudiante.		
4	Filtrar las notas por periodo, carrera o asignatura, o buscar por nombre de estudiante.	Si	Encabezado Recuperación Aprobado, Pide Configurar Matrus por carrera.
5	Crear o editar un registro de catálogo (por ejemplo, un paralelo o una asignatura).	Si	Presentar Alerta Evitar Usuario Duplicado.
6	Cargar masivamente docentes desde un archivo Excel.	Si	Titulos de las acciones que realiza el módulo.
7	Validar un archivo antes de enviarlo al SIGA y observar cómo el sistema detecta y bloquea errores.	Si	(Encabezado X)
8	Generar y descargar el reporte consolidado de un paralelo en formato Excel.	Si	

4. Cuestionario de usabilidad (SUS)

Marque su grado de acuerdo con cada afirmación. Escala: 1 = Totalmente en desacuerdo, 2 = En desacuerdo, 3 = Neutral, 4 = De acuerdo, 5 = Totalmente de acuerdo.

N°	Afirmación	1	2	3	4	5
1	Creo que me gustaría usar este sistema con frecuencia.					✓
2	Encuentro el sistema innecesariamente complejo.	✓				
3	Creo que el sistema es fácil de usar.				✓	
4	Creo que necesitaría el apoyo de un técnico para poder usar este sistema.		✓			
5	Encuentro que las diversas funciones del sistema están bien integradas.				✓	
6	Creo que el sistema es demasiado inconsistente.	✓				
7	Imagino que la mayoría de la gente aprendería a usar este sistema rápidamente.					✓
8	Encuentro el sistema muy incómodo de usar.	✓				
9	Me siento muy seguro usando el sistema.				✓	
10	Necesité aprender muchas cosas antes de poder manejar el sistema.		✓			

5. Retroalimentación cualitativa

Comparado con el proceso manual actual (Moodle → Excel → SIGA), ¿qué ventajas o desventajas observa?

Velocidad en la consulta de información.

Datos mejor organizados

¿Qué funciones le resultaron más útiles?

Importación de Datos y Reportes con filtros

¿Qué funciones cree que faltan o mejoraría?

Gestión de Roles, Manejo de Mallas vigentes por carrera.
Consulta de Notas por parcial I y Parcial II

¿Hubo algún paso confuso o propenso a error?

No.

¿Considera viable su adopción en el Instituto? ¿Bajo qué condiciones?

Es viable su implementación en el momento que el instituto cuente con la infraestructura tecnológica necesaria para su funcionamiento.

Otros comentarios:

Incluir la organización de información por Período Académico, por Carrera, por ciclo o Nivel Asignaturas presentando el nombre del profesor a cargo

6. Preguntas complementarias al evaluador experto

Estas preguntas las responde el evaluador junto con el facilitador al cierre de la sesión.

¿Considera suficiente su evaluación como experto, o recomienda incluir a otros evaluadores? ¿A quiénes? (idealmente quienes vivan el flujo de notas, como el coordinador).

Comparado con el Excel actual, ¿qué es lo que más tiempo le ahorraría el sistema?

La carga de datos consolidados, la generación de reportes, tiempo de respuesta en la consulta de notas.

¿Qué fue lo más claro y lo más confuso de la interfaz?

Todo estuvo claro y bien organizado.

¿Qué le agregaría al sistema antes de usarlo en producción?

La gestión de doble rol para usuarios o múltiples roles pues un docente puede ejercer el rol de Coordinador de Carrera.

7. Cálculo e interpretación del SUS (uso del investigador)

Cálculo: para los ítems impares (1, 3, 5, 7, 9), el puntaje es (respuesta - 1). Para los ítems pares (2, 4, 6, 8, 10), el puntaje es (5 - respuesta). Se suman los diez puntajes y el total se multiplica por 2,5, obteniendo un valor entre 0 y 100.

Interpretación (Bangor, Kortum & Miller, 2009): un puntaje superior a 68 se considera por encima del promedio (aceptable); superior a 80 se considera excelente. Este valor se reporta en la sección 4.2 de la tesis junto con la retroalimentación cualitativa del evaluador experto.

Anexo N. Catálogo Completo de Requerimientos No Funcionales

Código	Categoría	Descripción breve	Criterio resumido
RNF-01	Seguridad	Autenticación obligatoria con JWT	Endpoints sin token devuelven HTTP 401
RNF-02	Seguridad	Control de acceso basado en roles	Roles no autorizados devuelven HTTP 403
RNF-03	Seguridad	Hashing de contraseñas y manejo seguro de credenciales	Cero contraseñas en texto plano
RNF-04	Seguridad	Auditoría de cambios en notas y asistencia	100% de modificaciones registradas en bitácora
RNF-05	Rendimiento	Latencia aceptable en endpoints CRUD	p95 menor a 2 s con 50 usuarios concurrentes
RNF-06	Rendimiento	Throughput sostenido en horarios pico	100 peticiones/s sostenidas en el api-gateway
RNF-07	Rendimiento	Generación rápida de reportes consolidados	Reporte completo de período en menos de 30s
RNF-08	Fiabilidad	Manejo controlado de fallos de servicios externos	Cero excepciones no controladas visibles
RNF-09	Fiabilidad	Tasa baja de errores en operación normal	Errores HTTP 5xx menores al 1% del total
RNF-10	Disponibilidad	Reproducibilidad del entorno con un solo comando	docker compose up levanta todo en menos de 5 min
RNF-11	Mantenibilidad	Cobertura mínima de pruebas unitarias	Cobertura $\geq 60\%$ en services y controllers
RNF-12	Mantenibilidad	Convenciones de código y baja deuda técnica	Cero violaciones críticas, complejidad menor a 10
RNF-13	Mantenibilidad	Documentación navegable de endpoints REST	100% de endpoints documentados en OpenAPI

RNF-14	Portabilidad	Independencia de plataforma	Verificado en Windows, Linux y macOS
RNF-15	Usabilidad	Percepción aceptable por docentes y coordinadores	Puntuación SUS global ≥ 68 sobre 100
RNF-16	Usabilidad	Accesibilidad web mínima	Cumplimiento WCAG 2.1 niveles A y AA

A continuación, la ficha individual de cada uno de los 16 RNF.

RNF-01 — Autenticación	
Categoría	Seguridad
Descripción	El sistema debe autenticar a todos los usuarios mediante credenciales personales (cédula y contraseña) antes de permitir acceso a cualquier funcionalidad distinta del login y la recuperación de contraseña.
Criterio de aceptación	Cero endpoints protegidos accesibles sin token JWT válido. El 100% de las peticiones a endpoints protegidos sin token deben devolver código HTTP 401.
Origen derivativo	Derivado del ADR-001 (autenticación con JWT) y del marco normativo COESC sobre protección de datos personales.
RF afectados	RF-005 directamente; transversalmente todos los demás RF que requieren usuario autenticado.
Mecanismo arquitectónico	Spring Security 6 con filtro JWT global validado en el api-gateway. PasswordEncoder BCrypt con strength 12. Tokens con tiempo de vida limitado (15 minutos) y refresh tokens con rotación.
Método de validación	Pruebas automatizadas que verifican rechazo de peticiones sin token y con token inválido. OWASP ZAP scan de los endpoints expuestos por el api-gateway. Reporte en sección 4.6.1 del Capítulo 4.
RNF-02 — Autorización	
Categoría	Seguridad
Descripción	El sistema debe aplicar control de acceso basado en roles (RBAC), donde cada rol tiene un conjunto definido de permisos sobre endpoints y recursos del sistema.
Criterio de aceptación	El 100% de los endpoints protegidos rechazan peticiones cuyo rol no tiene el permiso correspondiente, devolviendo código HTTP 403.

Origen derivativo	Derivado del RF-005 (Roles y permisos) y de las entrevistas con coordinadores que requieren separación de responsabilidades por rol.
RF afectados	RF-005, RF-007, RF-008, RF-013, RF-018 (todos los que tienen restricciones por rol).
Mecanismo arquitectónico	Anotaciones <code>@PreAuthorize</code> en los controllers de Spring Boot. Tabla de permisos persistida en BD con relaciones rol-permiso. Validación en el api-gateway antes de enrutar.
Método de validación	Pruebas de integración que recorren la matriz rol $\times$ endpoint. Reporte de cumplimiento por endpoint en sección 4.6.1.
RNF-03 — Protección de datos	
Categoría	Seguridad
Descripción	Las contraseñas de usuario nunca deben almacenarse en texto plano ni ser recuperables por reversión. Las claves y tokens sensibles no deben aparecer en código fuente ni logs.
Criterio de aceptación	Cero contraseñas en texto plano en BD. Cero credenciales en código fuente versionado (verificable por auditoría del repositorio).
Origen derivativo	Derivado de OWASP Top 10 (categoría Cryptographic Failures) y del marco normativo nacional sobre protección de datos personales.
RF afectados	RF-005 directamente.
Mecanismo arquitectónico	Hashing BCrypt para contraseñas. Variables de entorno fuera del repositorio para credenciales. Recuperación de contraseña por reset (email con token de un solo uso), no por envío de contraseña actual.
Método de validación	Inspección manual de la BD. Auditoría del repositorio mediante git-secrets. Verificación documentada en sección 4.6.1.
RNF-04 — Auditoría	
Categoría	Seguridad
Descripción	Todas las modificaciones sobre notas y asistencia deben quedar registradas con autor, fecha, valor anterior y valor nuevo. La bitácora de auditoría es de solo lectura.
Criterio de aceptación	El 100% de las operaciones de actualización o eliminación de notas y asistencia generan un registro en la tabla de auditoría.

Origen derivativo	Derivado del RF-018 (Auditoría) y de las entrevistas con coordinadores que destacaron la falta de trazabilidad en el proceso manual actual.
RF afectados	RF-003, RF-018, RF-019.
Mecanismo arquitectónico	Triggers en PostgreSQL sobre las tablas nota y registro_asistencia. Tabla auditoría con esquema inmutable (sin permisos de UPDATE ni DELETE para el usuario de aplicación).
Método de validación	Pruebas que ejecutan modificaciones sobre notas y verifican el registro correspondiente en auditoría. Reporte en sección 4.6.1.
RNF-05 — Latencia	
Categoría	Rendimiento
Descripción	Los endpoints CRUD principales deben responder en tiempos aceptables bajo carga normal, sin afectar la productividad del usuario docente o coordinador.
Criterio de aceptación	Latencia p95 menor a 2 segundos para endpoints CRUD bajo carga de hasta 50 usuarios concurrentes. Latencia p99 menor a 5 segundos.
Origen derivativo	Derivado del análisis del proceso manual actual (entrevistas con docentes que tardan minutos por nota) y de las recomendaciones generales de UX para sistemas web.
RF afectados	RF-003, RF-007, RF-008.
Mecanismo arquitectónico	Índices en columnas de búsqueda frecuente. Paginación cursor-based en listados. Connection pooling con HikariCP. Cache local de catálogos estables (carreras, paralelos del período).
Método de validación	Pruebas de carga con Apache JMeter o k6. Reporte cuantitativo en sección 4.5 (validación de desempeño) del Capítulo 4.
RNF-06 — Throughput	
Categoría	Rendimiento
Descripción	El sistema debe sostener un throughput suficiente para soportar el uso simultáneo del Instituto en horarios pico (cierre de parcial, junta académica).
Criterio de aceptación	Throughput sostenido mínimo de 100 peticiones por segundo en el api-gateway sin degradación de latencia.

Origen derivativo	Derivado del tamaño aproximado del Instituto (estimación de 20 a 30 docentes simultáneos durante cierre de parcial) y de la entrevista con el coordinador sobre uso simultáneo.
RF afectados	Transversal a todos los RF.
Mecanismo arquitectónico	Microservicios independientes (escalabilidad horizontal en el futuro). Operaciones costosas asincrónicas (sincronización Moodle, generación de reportes pesados). Connection pool dimensionado correctamente.
Método de validación	Pruebas de carga sostenida con JMeter o k6 durante 10 minutos. Reporte en sección 4.5.
RNF-07 — Generación de reportes	
Categoría	Rendimiento
Descripción	Los reportes consolidados deben generarse en tiempos razonables incluso con volúmenes de datos del Instituto completo.
Criterio de aceptación	Reporte consolidado de un período académico completo (estimado 500 estudiantes) generado en menos de 30 segundos.
Origen derivativo	Derivado del RF-008 (Reportes) y del proceso manual actual donde la consolidación toma horas.
RF afectados	RF-008, RF-004.
Mecanismo arquitectónico	Vistas materializadas en PostgreSQL para agregaciones frecuentes. Generación asíncrona con notificación al usuario al completarse. Streaming de archivos grandes.
Método de validación	Pruebas con datos de prueba a escala real. Comparativa con tiempo del proceso manual. Reporte en sección 4.5.
RNF-08 — Manejo de errores	
Categoría	Fiabilidad
Descripción	El sistema debe manejar fallos del servicio externo (Moodle) y errores durante la generación del archivo de exportación hacia SIGA, sin propagar errores no controlados al usuario final.
Criterio de aceptación	Cero excepciones no controladas visibles al usuario. El 100% de los errores devuelven una respuesta HTTP estructurada con mensaje legible.
Origen derivativo	Derivado del ADR-003 (estrategia Moodle local más adaptador SIGA dual) que explícitamente prevé fallos del lado de Moodle.

RF afectados	RF-001, RF-002, RF-010.
Mecanismo arquitectónico	GlobalExceptionHandler en cada microservicio. Timeouts explícitos en llamadas HTTP a Moodle. Manejo de errores durante la generación del archivo de exportación hacia SIGA, por ejemplo si falla la consulta a academic-service. Logs estructurados.
Método de validación	Pruebas que apagan deliberadamente Moodle local y verifican comportamiento del sync-service. Reporte en sección 4.6.2.
RNF-09 — Tasa de errores	
Categoría	Fiabilidad
Descripción	El sistema debe mantener una tasa de errores baja durante operación normal, evidenciando estabilidad para una eventual implementación productiva.
Criterio de aceptación	Tasa de errores HTTP 5xx menor al 1% sobre el total de peticiones durante pruebas de carga sostenida.
Origen derivativo	Derivado de buenas prácticas generales de servicios web y de la naturaleza académica del proyecto, donde un sistema inestable sería inaceptable.
RF afectados	Transversal a todos los RF.
Mecanismo arquitectónico	Validación exhaustiva de inputs. Transacciones bien delimitadas en operaciones de escritura. Tests de integración con cobertura de casos de error.
Método de validación	Bench de 10 minutos con JMeter midiendo tasa de errores. Reporte en sección 4.6.2 del Capítulo 4.
RNF-10 — Levantamiento del entorno	
Categoría	Disponibilidad
Descripción	El entorno de desarrollo completo del prototipo debe ser reproducible sin pasos manuales en cualquier máquina con Docker instalado.
Criterio de aceptación	Comando docker compose up -d levanta el sistema completo en menos de 5 minutos en una máquina con recursos suficientes, sin pasos manuales adicionales más allá de copiar el archivo de variables de entorno.
Origen derivativo	Derivado del ADR-001 (Docker Compose como infraestructura de desarrollo) y de buenas prácticas de portabilidad.
RF afectados	Transversal a todos los RF (es un requisito de infraestructura).

Mecanismo arquitectónico	docker-compose.yml único en la raíz del proyecto. Healthchecks configurados. Migraciones Flyway que se ejecutan en el primer arranque. Datos semilla cargados automáticamente.
Método de validación	Demostración del despliegue en una segunda máquina en sección 4.6.4. Tiempo de levantamiento medido y reportado.
RNF-11 — Cobertura de pruebas	
Categoría	Mantenibilidad
Descripción	El código de la lógica de negocio de los microservicios debe estar respaldado por pruebas unitarias que garanticen su evolución segura.
Criterio de aceptación	Cobertura mínima del 60% de líneas en la lógica de negocio (services y controllers) por microservicio.
Origen derivativo	Derivado de buenas prácticas generales de ingeniería de software y de la rúbrica del tutor que evalúa rigor en el desarrollo.
RF afectados	Transversal a todos los RF.
Mecanismo arquitectónico	JUnit 5 más Mockito en backend. Vitest más React Testing Library en frontend. Pipeline de CI que ejecuta las pruebas en cada Pull Request.
Método de validación	Reporte JaCoCo por microservicio. Reporte c8 para frontend. Tabla con cobertura por componente en sección 4.3.2.
RNF-12 — Calidad de código	
Categoría	Mantenibilidad
Descripción	El código del sistema debe seguir convenciones consistentes y minimizar deuda técnica acumulada durante el desarrollo del prototipo.
Criterio de aceptación	Cero violaciones críticas en SonarLint o herramienta equivalente. Complejidad ciclomática promedio menor a 10 por método.
Origen derivativo	Derivado de buenas prácticas generales de ingeniería de software y de la naturaleza académica del proyecto, que será revisado por pares.
RF afectados	Transversal a todos los RF.
Mecanismo arquitectónico	ESLint más Prettier en frontend. Checkstyle o SpotBugs configurables en backend. Convenciones de naming documentadas. Code reviews en cada Pull Request (auto-review aceptable para single developer).
Método de validación	Reporte SonarLint o equivalente al cierre del proyecto. Métricas de calidad reportadas en sección 4.6.3.

RNF-13 — Documentación de API	
Categoría	Mantenibilidad
Descripción	Todos los endpoints REST del sistema deben estar documentados de forma navegable y actualizada en sincronía con el código.
Criterio de aceptación	El 100% de los endpoints expuestos están documentados en OpenAPI con descripción, parámetros y respuestas. Swagger UI accesible en cada microservicio.
Origen derivativo	Derivado del ADR-001 (SpringDoc OpenAPI como estándar de documentación) y de buenas prácticas de microservicios.
RF afectados	Transversal a todos los RF.
Mecanismo arquitectónico	SpringDoc OpenAPI con anotaciones en controllers. Generación automática del esquema OpenAPI en build. Swagger UI expuesto en /swagger-ui.html de cada microservicio.
Método de validación	Inspección manual de Swagger UI por microservicio.
RNF-14 — Independencia de plataforma	
Categoría	Portabilidad
Descripción	El sistema debe ejecutarse correctamente en sistemas operativos distintos (Windows, Linux, macOS) sin modificación del código fuente.
Criterio de aceptación	El sistema arranca correctamente con Docker Desktop en Windows con WSL2, Docker en Linux nativo y Docker Desktop en macOS.
Origen derivativo	Derivado del ADR-001 (uso de Docker como contenedorización) y de la potencial heterogeneidad de las máquinas del Instituto.
RF afectados	Transversal a todos los RF.
Mecanismo arquitectónico	Imágenes Docker oficiales basadas en Linux. Variables de entorno para configuraciones específicas del sistema. Sin paths absolutos hardcoded.
Método de validación	Demostración en al menos dos sistemas operativos distintos. Reporte en sección 4.6.4 del Capítulo 4.
RNF-15 — Percepción del usuario	
Categoría	Usabilidad
Descripción	La interfaz del sistema debe ser percibida como aceptable o mejor por los usuarios docentes y coordinadores que la utilicen.

Criterio de aceptación	Puntuación SUS (System Usability Scale) global mayor o igual a 68 puntos sobre 100. Puntuación por subgrupo (rol) mayor o igual a 65.
Origen derivativo	Derivado de buenas prácticas de UX y del estándar SUS (Brooke, 1996). Las entrevistas con docentes confirmaron la frustración con sistemas actuales poco usables.
RF afectados	Transversal al frontend de todos los RF.
Mecanismo arquitectónico	Sistema de diseño consistente con TailwindCSS más shadcn/ui. Estados claros de carga, error y vacío. Mensajes de error accionables. Feedback inmediato en operaciones críticas.
Método de validación	Pruebas de usabilidad guiadas con docentes y coordinadores aplicando el cuestionario SUS estándar. Reporte detallado en sección 4.4 del Capítulo 4.
RNF-16 — Accesibilidad	
Categoría	Usabilidad
Descripción	El frontend del sistema debe cumplir con los estándares mínimos de accesibilidad para no excluir usuarios con discapacidades visuales o motoras.
Criterio de aceptación	Cumplimiento de los criterios A y AA de WCAG 2.1 verificable con herramientas automáticas (axe, Lighthouse).
Origen derivativo	Derivado del estándar WCAG 2.1 y del marco normativo nacional ecuatoriano sobre accesibilidad de servicios públicos.
RF afectados	Transversal al frontend de todos los RF.
Mecanismo arquitectónico	Componentes shadcn/ui que ya cumplen accesibilidad por defecto. Etiquetas semánticas correctas. Contraste de color verificado. Soporte de navegación por teclado.
Método de validación	Auditoría con axe DevTools y Lighthouse. Reporte de cumplimiento en sección 4.6.5 del Capítulo 4.

Nelson Emmanuel Brito Ordóñez, portador de la cédula de ciudadanía N.º 0107828261, en calidad de autor y titular de los derechos patrimoniales del proyecto de titulación “Prototipo de sistema para el registro de notas y asistencia basado en microservicios con integración SIGA-Moodle en el Instituto Superior Tecnológico del Austro”, de conformidad con lo establecido en el artículo 114 del Código Orgánico de la Economía Social de los Conocimientos, Creatividad e Innovación, reconozco a favor de la Universidad Católica de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos y no comerciales. Autorizo además a la Universidad Católica de Cuenca, para que realice la publicación de este proyecto de titulación en el Repositorio Institucional, de conformidad con lo dispuesto en el artículo 144 de la Ley Orgánica de Educación Superior.

Cuenca, 5 de Agosto del 2026

F.: 

Nelson Emmanuel Brito Ordóñez

C.I. 0107828261