UNIVERSIDAD CATÓLICA DE CUENCA
SEDE AZOGUES

UNIDAD ACADÉMICA DE TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIÓN

INGENIERÍA EN SISTEMAS

TEMA:

SISTEMA DE MICROSERVICIOS WEB PARA PLATAFORMAS IoT

TRABAJO DE TITULACIÓNPRESENTADO EN CONFORMIDAD CON LOS REQUISITOS ESTABLECIDOS PARA LA OBTENCIÓN DEL TÍTULO DE

INGENIERO EN SISTEMAS

AUTOR:

Raúl Rigoberto Rivera Calle

DIRECTOR:

Ing. Andrés Sebastián Quevedo Sacoto MSc.

AZOGUES – ECUADOR
2020

AZOGUES, MARZO DE 2020

APROBACION DEL TUTOR

En calidad de tutor del trabajo de grado, presentado por el Sr. Raúl Rigoberto Rivera Calle para optar por el título de INGENIERO EN SISTEMAS, doy fe que dicho trabajo reúne los requisitos y méritos suficientes para ser sometido a presentación pública y evaluación por parte del jurado examinador que se designe.

En la ciudad de Azogues, a los 28 días del mes de febrero de 2020.

_____

ING. ANDRÉS SEBASTIÁN QUEVEDO SACOTO MSc.

C.I. 0301826434

CESIÓN DE DERECHOS DE AUTOR

Yo, Raúl Rigoberto Rivera Calle con documento de identidad 0302450069 manifiesto mi voluntad y cedo a la Universidad Católica de Cuenca la titularidad sobre los derechos patrimoniales en virtud de que soy el autor de trabajo de titulación: **"SISTEMA DE MICROSERVICIOS WEB PARA PLATAFORMAS IoT"**, mismo que ha sido desarrollado para optar por el título de: Ingeniero de Sistemas, en la Universidad Católica de Cuenca, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En aplicación a lo determinado en la Ley de Propiedad Intelectual, en mi condición de autor me reservo los derechos morales de la obra antes citada. En concordancia suscribo este documento en el momento que hago entrega del trabajo final en formato impreso y digital a la Biblioteca de la Universidad Católica de Cuenca.

Azogues, marzo 2020

_____

Raúl Rigoberto Rivera Calle

C.I. 0302450069

CERTIFICADO DE AUTORÍA

El presente trabajo investigativo de proyecto profesional de grado previo a la obtención del título de Ingeniero en Sistemas, cuyo tema es "Sistema de microservicios web para plataformas IoT", corresponde al trabajo de investigación del autor, además certifico que he cumplido con todas las observaciones realizadas por el tribunal evaluador, por lo que las ideas, opiniones vertidas en el presente, son de exclusiva responsabilidad del autor.

Estudiante:

_____

Raúl Rigoberto Rivera Calle

C.I. 0302450069

DEDICATORIA

Mi tesis va dedicado con mucho amor y cariño a mis padres: Arturo Rivera y Piedad Calle, quienes con mucho esfuerzo y paciencia me han sabido guiar por el camino del bien y haberme permitido llegar a ser la persona que soy en la actualidad.

A mis hermanos Juan Rivera y Rosa Rivera, por la paciencia que han sabido tener conmigo, por haber estado siempre pendientes en cada momento, brindándome en cada momento su apoyo y sus consejos que siempre los tendré presentes.

A mis grandes amigos: Belenchis Encalada, Xavier Matute y a mi tio que siempre ha estado conmigo aconsejándome y brindándome todo su apoyo: Homero Rivera. Que por cuestiones de la vida, Dios quiso que se nos adelantaran y hoy yo sé que ellos me acompañan, ellos son quienes me guían y me brindan su apoyo para conseguir mis objetivos.

A cada uno de mis compañeros por haber compartido grandes momentos y recuerdos que quedaran impregnados para siempre y por haber sido parte de este sueño.

Gracias a todos.

AGRADECIMIENTO

En primer lugar, agradezco a Dios por haberme acompañado y guiado en los momentos más difíciles brindándome fuerza y sabiduría para conseguir mi objetivo.

Quiero también agradecer a mi tutor de trabajo de titulación por la ayuda brindada, su orientación, conocimiento y sus valiosas sugerencias para permitirme llegar a mi objetivo planteado. De igual manera, mi agradecimiento a Cristian Guillen, por su conocimiento y gran apoyo que me supo brindar en cada momento.

Agradezco también a todos los catedráticos de la Universidad Católica y todo el personal que integran la misma por haberme brindado siempre su apoyo incondicional cuando más los necesitaba.

Por último y no menos importante, agradecer a mi familia por haberme apoyado siempre en las buenas y en las malas, aconsejándome siempre por el camino del bien y además por la paciencia que siempre me han sabido brindar.

RESUMEN

SISTEMA DE MICROSERVICIOS WEB PARA PLATAFORMAS IoT

Autor: Raúl Rigoberto Rivera Calle

Tutor: Ing. Andrés Sebastián Quevedo Sacoto MSc.

El presente proyecto se desarrolla con la finalidad de crear una infraestructura de microservicios compatibles con arquitecturas para Internet de las Cosas (IoT), la infraestructura de microservicios propuesta permiten la comunicación directa y en tiempo real con los usuarios que consumen los servicios "Páginas Web, teléfonos inteligentes y otros dispositivos IoT". En el primer capítulo se realiza el análisis de la problemática, antecedente, objetivo y las contribuciones que se brindará en el presente trabajo de titulación. En el segundo capítulo se visualiza el marco teórico por medio del cual se realizará un estudio para elegir las mejores herramientas para obtener un software de calidad. En el tercer capítulo se efectúa el desarrollo del actual sistema a través de la metodología Scrum y finalmente en el cuarto y último capítulo se presentan las conclusiones y recomendaciones del presente trabajo de titulación.

Palabras clave: Microservicios, IoT, Scrum.

ABSTRACT

WEB MICROSERVICES SYSTEM FOR IoT PLATFORMS

This research work is developed to creating a micro service infrastructure compatible with architectures for Intenert of Things (IoT), the proposed micro services infrastructure allows a direct and real-time communication with users who use the services “Web Page, smartphones and other IoT devices.”

In the first chapter, it was done an analysis about the ptoblem, antecedent, objective and the contributions. In the second chapter, it is visualized the theoretical framework throughout a study, it allowed to choose the best tools to get a quality software. In the third chapter, it was carried out a development of the current system through the Scrum methodology and finally, in the fourth and last chapter, it was presented the conclusions and recommendations of this research work.

Keywords: Microservices, IoT, Scrum.

1.1 Índice

1.2Índice de Tablas

1.3 Índice de Figuras

TÍTULO DEL TRABAJO DE GRADUACIÓN

SISTEMA DE MICROSERVICIOS WEB PARA PLATAFORMAS IoT

CAPÍTULO 1

1.1Introducción

El término "microservicio" fue discutido en un taller de arquitectos de software por Venecia en mayo de 2011 con el objeto de describir lo que los participantes vieron como un estilo arquitectónico común que recientemente era explorado por algunos de ellos. En mayo de 2012, se decidió "microservicios" como el nombre más apropiado (Fowler, 2014).

En la actualidad, a nivel empresarial tanto en el sector público como en el privado se desarrolla software para cubrir las necesidades de automatización de procesos internos, siguiendo las tendencias impuestas por la plataforma, lenguaje de programación o por la experiencia del área de desarrollo, lo cual deviene en la implantación de sistemas de construcción tradicional o monolítico (Maya, E., y López, 2017).

La implementación de microservicios con IoT admite la adopción de un proceso de entrega continua de alta velocidad con el objeto de crear soluciones confiables y escalables de Software como Servicio que están diseñadas y construidas mediante la utilización de la arquitectura de microservicio que pueden ser administradas por medio de la automatización. Las aplicaciones SaaS (Software as a Service, es un prototipo de asignación de software en el que tanto el software como los datos utilizados son agrupados e incrustados en un exclusivo servidor externo a la institución) son productos de software que están disponibles

24x7, pueden funcionar en cualquier dispositivo, escalan elásticamente, además son resistentes al cambio (Familiar & Barnes, 2015).

El desarrollo de este trabajo de tesis está compuesto por cuatro capítulos, los cuales se presentan brevemente a continuación:

- ✓ **Capítulo 1**: Este capítulo detalla la fundamentación teórica del problema a resolver en este trabajo de investigación. Primero se realiza la descripción de los antecedentes de la problemática, la descripción del problema, luego se realiza el planteamiento de los objetivos que ayudarán a encaminar el desarrollo del presente trabajo de titulación, así como también se puede visualir el estado del arte en referencia a trabajos realizados anteriormente y finalmente las contribuciones de este trabajo de investigación. En el presente capitulo también se encontrará detallado el marco teórico.
- ✓ **Capítulo 2:** Hace referencia al desarrollo y aplicación de la metodología elegida.
- ✓ **Capítulo 3:** Este capítulo se basa en sí, en el desarrollo del sistema de microservicios para plataformas IoT.
- ✓ **Capítulo 4:** En este capítulo, se presentan las conclusiones y recomendaciones que se puedan hacer respecto al presente sistema.

1.2Antecedentes

La gran acogida que ha tenido en los últimos tiempos la industria del software a nivel económico se apoya en el patrocinio de la operatividad y estabilidad que brinda a otros grupos industriales considerables de la economía nacional. En el Ecuador el sector de la industria del software ha evolucionado de manera significativa, siendo los programas informáticos un bien adquirido de manera obligatoria por las empresas para mejorar su productividad (Antonio ESPINOZA Mina & del Pilar GALLEGOS Barzola, 2017).

En la actualidad a través del manejo de los microservicios es posible crear grandes proyectos un claro ejemplo es la implementación de la tecnología de microservicios con websockets la cual nos permite interactuar en tiempo real con nuestros usuarios.

En el Ecuador, en todos los sectores tanto público como privado se desarrolla programas informáticos (software) que permiten satisfacer las necesidades a través de la automatización de procesos, utilizando las tendencias de desarrollo que han impuesto la plataforma, el lenguaje de programación o simplemente por la experiencia del área de desarrollo, permitiendo la implementación de sistemas de construcción tradicional (Hinojosa & Daniel, n.d.).

Las atractivas prestaciones tecnológicas que ofrecen los microservicios han logrado posesionarse en los ojos de las tecnologías del Internet de aquellas cosas ya que ha evolucionado de

manera rápida permitiendo que IoT deje de ser una simple visión de futuro sino una realidad. Esta tecnología ha alcanzado la fama debido a que todas las aplicaciones y posibilidades que nos permiten mejorar tanto la vida cotidiana, así como los entornos empresariales, dónde ya se está implementado desde hace algún tiempo (Gracia, 2018).

Microservicios es una arquitectura que tiene varias ventajas como: la complejidad bajo control, su implementación es independiente, posees mayor cantidad de opciones para la pila de tecnología y la tolerancia a fallas, permitiendo mayor facilidad en el desarrollo de aplicaciones IoT a gran escala (Sun, Li, & Memon, 2017).

Por lo tanto, es de mucha importancia tener en cuenta que un sistema IoT es capaz de involucrar múltiples implementaciones técnicas de elementos clave como contenedores y protocolos de notificación de eventos de manera simultánea (Lu, Huang, Walenstein, & Medhi, 2017).

En la actualidad, el desarrollo de microservicios con IoT están mejorando de gran forma, brindando diversos frutos, rentabilidad, rendimiento y es muy penoso tener que expresar, que en la actualidad en la ciudad de azogues se carece de un sistema de microservicios para plataformas IoT. Por lo cual se plantea el desarrollo del presente trabajo de tesis que brindará una conexión activa entre el cliente y los microservicios.

1.3 Descripción del Problema

Hoy en día, existe una gran variedad de tecnologías y aplicaciones que están enlazadas de una u otra forma en todas las actividades que realizan en la vida cotidiana. Una de las tecnologías más reconocidas en la actualidad son las plataformas del Internet de las cosas (IoT), estas requieren conectarse a sistemas que permitan gestionar la información que es producida por los mismos, este vínculo se puede ejecutar de diferentes maneras, por ejemplo arquitecturas cliente servidor tradicionales, monolíticas, con infraestructura de microservicios, o con almacenamiento estático en archivos planos. Tradicionalmente los dispositivos IoT almacenan la información en archivos planos.

Las plataformas IoT manejan una comunicación muy variada por lo que provocan muchos problemas entre dispositivos o plataformas. También, en vista de que sostiene un bajo poder de computación en estos dispositivos, es muy común encontrar embotellamientos en la comunicación de estos dispositivos. Por lo cual, si no se cuenta con la implementación de microservicios para estas plataformas vamos a mantener constantes problemas de interoperabilidad y el escalado de los dispositivos, ya que son los microservicios los cuales permiten ser desarrollados en diferentes áreas de programación y usar distinta tecnología de almacenamiento.

1.4 Objetivos

1.4.1 Objetivo General

Crear un prototipo de un sistema de microservicios web para el empleo de plataformas IoT, implementando tecnologías actuales de construcción de microservicios web, para brindar una mayor facilidad de interacción entre los sistemas y los usuarios.

1.4.2 Objetivos Específicos

- ✓ Analizar estudios realizados para la implementación de sistemas de microservicios con plataformas IoT para detallar la revisión de la problemática.
- ✓ Detallar el levantamiento de la información mediante el análisis de estudios realizados para el desarrollo del sistema.
- ✓ Desarrollar un sistema de información a través de la metodología Scrum, con la ayuda de cada una de las herramientas y frameworks de desarrollo que brindaran su aporte para crear los microservicios para plataformas IoT, los cuales permitirán obtener interacción entre los microservicios y los usuarios.
- ✓ Determinar conclusiones y recomendaciones a través de invenciones detectadas para la culminación del trabajo de titulación.

1.5 Estado del Arte.

En lo concerniente al presente desarrollo se han encontrado una gran variedad de trabajos realizados tanto a nivel nacional como internacional, los cuales especificaremos a continuación:

Claus Djernæs Nielsen (2015) en la Universidad de Aarhus de Dinamarca, realiza la evaluación de diferentes tácticas de disponibilidad y mantenibilidad para el desarrollo de un prototipo con arquitectura de microservicios, en aquel procedimiento, culmina con la comprobación de las particularidades de los microservicios enunciadas por Fowler y Lewis (2014) en cuatro criterios principales que definen a un microservicio (Hinojosa & Daniel, 2017).

- ✓ **Enfoque en las capacidades de negocio.** El proceso de desarrollo de software se vuelve más flexible con esta arquitectura, debido a que, cada microservicio está conformado por su propia lógica de negocios, base de datos, interfaz de usuario y funcionalidad de comunicación con otros servicios.
- ✓ **Independencia de los servicios autónomos.** Cada servicio está conformado por su propia lógica de negocio que se desarrolla por separado. Permitiendo de esta manera cambiar y extender gradualmente con nuevas características sin perturbar el resto del sistema. La autonomía permite flexibilidad en la selección de la tecnología más adecuada para un servicio específico.

- ✓ **Gestión descentralizada de datos.** Todos los sistemas sostienen una base datos pequeña y simple, que se pueden observar por separado, estos datos están desacoplados, por lo que se necesita mucha gestión y pruebas de manera que garantice que el sistema no se actualice o elimine datos en un servicio sin antes actualizar o eliminar los datos correspondientes en otros servicios que contienen los mismos datos o conjunto de referencias.
- ✓ **Tolerancia a fallos.** Estos sistemas están conformados por múltiples unidades pequeñas que pueden fallar; dichas unidades están acopladas de tal modo que pueden ser restauradas de manera automática, dando como resultado un sistema más estable y robusto.

Gran cantidad de investigadores proporcionan soluciones de IoT a través de la arquitectura de microservicios. Por ejemplo, Vresk y Tomislav exhiben una arquitectura justificada en microservicios que se orienta en conectarse con dispositivos heterogéneos, donde el sistema se limita al aspecto del modelo de datos. Krylovskiy y col. analiza cómo aplicar la arquitectura de microservicios para diseñar una plataforma Smart City IoT. Bak et al presenta tres microservicios en la nube: microservicio de activación contextual, microservicio de visualización y detección de anomalías y microservicio de causa raíz, para acelerar y facilitar el desarrollo de aplicaciones basadas en contexto y ubicación. Sin embargo, los sistemas de IoT de microservicio que se ha mencionados solo consideran una aplicación específica. Por lo tanto, es necesario diseñar un marco más genérico y abierto en el sistema IoT (Sun et al., 2017).

Por su parte (Borcin, 2017) en su trabajo de investigación, realiza un análisis de los atributos de calidad de la arquitectura de microservicios profundizando en los retos de su implementación (Hinojosa & Daniel, 2017).

- ✓ **Interfaces y comunicación.** Todo microservicio debe abarcar alguna interfaz, ya que para ciertas tecnologías no suministradas con una determinación, puede mostrar arduas dificultades incluso REST que es beneficiado de la arquitectura de microservicio para la comunicación no posee una interfaz bien definida.
- ✓ **Transacciones y bloqueos.** Es necesario un conjunto completamente nuevo de operaciones para una transacción que comprueba la comunicación de microservicios. Por ello es necesario que ningún microservicio utilice flujos cerrados o recursos bloqueados.
- ✓ **Programación coreográfica.** Permite la creación de sistemas libres de bloqueos, sin errores de comunicación, con la programación coreográfica.
- ✓ **Puntos de entrada.** La seguridad es un problema clave en la arquitectura de microservicios y los sistemas distribuidos en general, por ello en aplicaciones monolíticas se puede asegurar fácilmente la seguridad integra del sistema, ya que toda la relación pasa mediante los puntos de entrada que posee cada módulo que lo compone.
- ✓ **Red compleja.** Una amplia red de comunicación puede estar evidentemente comprometido a ataques, ya que es una tarea muy ardua manejar cuando el sistema consta de extensivos microservicios

y esos servicios envían miles de mensajes. El monitoreo de esta complicada red es también muy difícil.

Finalmente se puede definir que la implementación de microservicios con IoT, ofrece un gran servicio como: la interconexión de cualquier objeto o producto con cualquier por medio de la red. Permitiendo tener comunicación y cooperación, capacidad de direccionamiento, identificación, localización y sobre todo velocidad en el análisis de los datos.

Por otra parte cabe recalcar que no todo es factible, lo que conlleva una serie de consecuencias e inconvenientes como son: la compatibilidad, complejidad, privacidad y sobre todo la seguridad que de una u otra forma estaría propenso a ser hackeado y tomado la información para ser mal utilizada.

1.6 Contribuciones.

El presente documento aportará de gran manera a la sociedad mediante la implementación de un sistema de Microservicios web para plataformas IoT, el cual nos permitirá interactuar directamente con las personas que tengan acceso a la página web y que dispongan de la aplicación móvil, la misma que contará con grandes herramientas de desarrollo de software y tecnologías actuales.

CAPÍTULO 2
1.4 MARCO TEORICO

En el presente capítulo se determina cada uno de los mecanismos y tecnologías que se usaran para el desarrollo del proyecto, así como también se especificará las partes más importantes que de una u otra manera interactuaran con los microservicios, toda esta información recopilada será de gran ayuda para que los lectores tenga un mayor entendimiento de cada una de las áreas tales como: Plataformas IoT, placas de desarrollo de hardware, Microservicios, Web services, Desarrollo de aplicaciones móviles y uno de las áreas más importantes como es el almacenamiento en la Base de datos, metodología de desarrollo.

2.1 Plataformas IOT

Las plataformas IoT también conocidas como nivel alto porque se aplican en grandes proyectos de Internet de las cosas, lo que nos brinda una mayor eficiencia y por ende un mayor costo en el mercado. Como dicen los desarrolladores de Azure “Microsoft Azure le da la libertad de crear, administrar e implementar aplicaciones en una tremenda red mundial con sus herramientas y marcos favoritos”, lo que nos afirma que

estas plataformas son las mejores para realizar grandes proyectos (MARIN, 2018).

A continuación, se puede observar en la ilustración 1, los elementos básicos para las plataformas IoT.

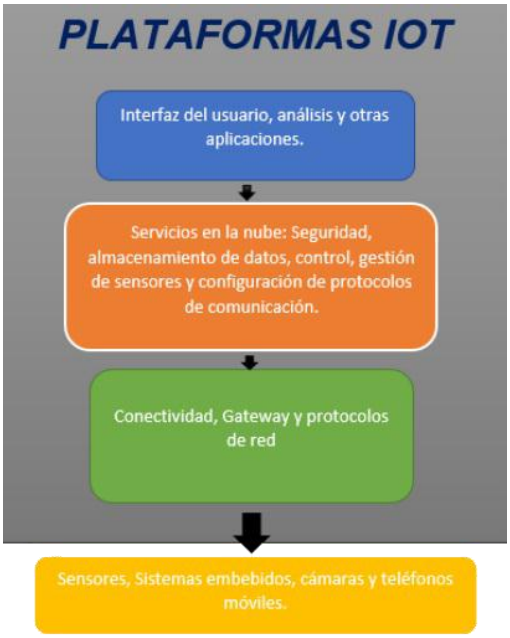


Ilustración 1: Plataforma IoT en un sistema de Internet de las cosas.

Fuente 1: http://190.131.241.186/bitstream/handle/10823/1215/Documento_Trabajo_Grado.pdf?sequence=1&isAllowed=y

2.2 Placas de desarrollo de hardware

Hoy en día existen diferentes tipos de hardware que permiten el desarrollo de aplicaciones IoT, entre las más destacadas tenemos (PCDuino, BeagleBone, Raspberry Pi, Arduino), a continuación en la tabla 1 se podrá observar detalladamente el desenvolvimiento de cada una de las placas en un cuadro de ponderación con sus respectivas características.

	Sistema Operativo Compatible con Windows	**Gran espacio de almacenamiento**	**Precio económico**	**Total**
PCDuino		X		1
BeagleBone		X		1
Raspberry Pi		X	X	2
Arduino	X		X	2

Tabla 1: Placas de desarrollo

Luego de haber revisado detalladamente cada una de las placas dentro de un cuadro de ponderación de acuerdo a sus características, se ha elegido arduino debido a la facilidad y bajo costo que nos brinda, la misma que aportará de gran manera en el desarrollo del proyecto.

2.3 Arduino

Arduino es una herramienta que permite que los ordenadores puedan controlar a través de un ordenador personal, esta plataforma es de código abierto, se basa en una placa con un sencillo microcontrolador y un ambiente de desarrollo para desarrollar programas para la placa (Arduino, 2012).

Se puede emplear Arduino para la producción de objetos interactivos, leyendo datos de diversos mandos y sensores, además de controlar miles de tipos de luces, motores y actuadores físicos. Los proyectos realizados con Arduino pueden ser autónomos o comunicarse con un programa (software) que se ejecute en un ordenador. La placa es muy fácil de montarla o se puede adquirir en el mercado ya lista para ser usada, su software de desarrollo es abierto y se lo consigue en la web gratuito descargando de su página oficial. A continuación se puede observar en la ilustración 2 una placa de arduino UNO (Arduino, 2012).

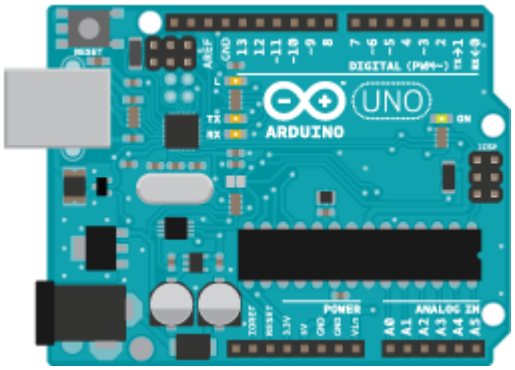


Ilustración 2: Arduino

2.4 ¿Cómo se originó el Arduino?

Arduino fue diseñado por el Instituto de Diseño Interactivo de Ivrea en el 2005 debido a la necesidad de contar con un dispositivo que sea fácil de utilizar en las aulas a un bajo costo. La idea original fue, construir una placa para uso interno de la escuela (Electronics, 2017).

El instituto se vio obligado a cerrar sus puertas en el mismo año 2005, debido a la perspectiva de perder durante el proceso todo el proyecto

Arduino, por lo que decidieron liberarlo y abrirlo al público para que todo el mundo pudiese participar en la evolución del proyecto, proponer mejoras y sugerencias (Electronics, 2017).

2.4.1 ¿Porque usar Arduino?

Arduino es libre y extensible: esto significa que cualquiera que desee ampliar y mejorar el diseño hardware de las placas como el entorno de desarrollo, puede hacerlo sin ningún problema. Esto permite que exista un poderoso ecosistema de placas electrónicas no oficiales para diferentes propósitos y de librerías de software de tercero, que pueden adaptarse mejor a nuestras necesidades (Electronics, 2017).

Arduino tiene una gran comunidad: Debido a su gran alcance hay una enorme comunidad trabajando con esta plataforma, genernado gran cantidad de documentación muy extensa, la cual abarca casi cualquier necesidad (Electronics, 2017).

Su ambiente de programación es multiplataforma: Se puede alojar y efectuar en sistemas operativos Windows, Mac OS y Linux (Electronics, 2017).

Lenguaje de programación de fácil compresión: El lenguaje de programación en que es desarrollado es C++ que es de fácil compresión pues permite una entrada sencilla a nuevos programadores, además posee una gran capacidad ya que los programadores mas expertos tienen la posiblidad de exprimir todo el potencial de su lenguaje y adaptarlo a cualquier situación (Electronics, 2017).

Bajo costo: La placa Arduino estándar posee un costo aproximado de $17, inclusive uno mismo la podría fabricar (una gran ventaja del hardware libre), mediante el cual el precio de la placa seria mucho menor (Electronics, 2017).

Re-usabilidad y versatilidad: Es re-utilizable, es decir, una vez terminado el proyecto es posible desmontar sus componentes externos a la placa y comenzar un nuevo proyecto, de igual forma todos los pines del microcontrolador son accesibles a través de conectores hembra, por lo que permite aprovechar de todas las bondades del microcontrolador con un riesgo muy bajo de hacer una conexión errónea (Electronics, 2017).

2.5 Microservicios

Según (López & Maya, n.d.) Es un planteamiento para el tratamiento de una aplicación singular que ofrece un conglomerado de pequeños servicios, cada uno de los cuales se efectúan en su propio proceso y mecanismos dinámicos de comunicación, a menudo un recurso de una interfaz de programación de aplicaciones (API) sobre protocolo de transferencia de hipertexto (HTTP). Estos servicios están desarrollados alrededor de las capacidades del negocio y con independencia de despliegue e implementación totalmente automatizada. Existe un mínimo de gestión centralizada de estos servicios, los que pueden estar escritos en lenguajes de programación diferentes y utilizar diferentes tecnologías de almacenamiento de datos.

(López & Maya, n.d.) Afirma que el término micro servicios posee un estilo arquitectural, fue acuñado por Martin Fowler en un taller de arquitectos de software como una descripción del nuevo campo que los participantes estaban explorando.

Esta arquitectura da origen al desarrollo de aplicaciones que están conformadas por unidades independientes, autónomas, modulares y auto-contenidas, diferenciándose de esta manera de su forma monolítica (López & Maya, n.d.).

Existen muchos tipos de frameworks que permiten la construcción de Microservicios pero entre los más destacados tenemos: Spark, Jersey, Payara Micro, Spring boot, AWS, entre otras. A continuación, se puede observar en la tabla 2, un cuadro de ponderación de los microservicios con sus respectivas características.

2.6 Análisis de microservicios

	Código abierto	**Soporte para la automatización con Maven y Gradle.**	**Configuración sugerida para iniciar rápidamente con un proyecto**	**Total**
Spark	X	X		2
Jersey	X	X		2
Payara micro	X	X		2
Spring boot	X	X	X	3
Aws	X	X		2

Tabla 2: Análisis de microservicios

Después de haber detallado el cuadro de ponderación con cada una de las características se pudo concluir que por su gran variedad de características que nos brinda se elegirá spring boot como el microservicios para el desarrollo del presente proyecto.

2.6.1 Spring boot

Spring boot es un framework cuyo propósito es simplificar el desarrollo de aplicaciones basado en Spring Core, además es de código abierto para la plataforma java.

Es intuitivo por tener valores predeterminados, y permite al usuario desarrollar de forma fácil y sencilla una aplicación mediante el uso de dichos valores.

Es personalizable, debido a que es muy fácil toda su configuración inicial así también en el ciclo de desarrollo.

(Perry, 2017) Afirma que Spring Boot nos permite mayor facilidad al instante de crear aplicaciones basadas en Spring, autónomas y del nivel de producción que únicamente se ejecutan.

2.6.2 Como funciona Spring boot.

Para la creación de un proyecto con Spring boot existen dos opciones: a) mediante el IDE a utilizar; a través de la página de Spring boot ingresando a SPRING INITIALIZR. Como se puede observar a continuación en la ilustración 3.

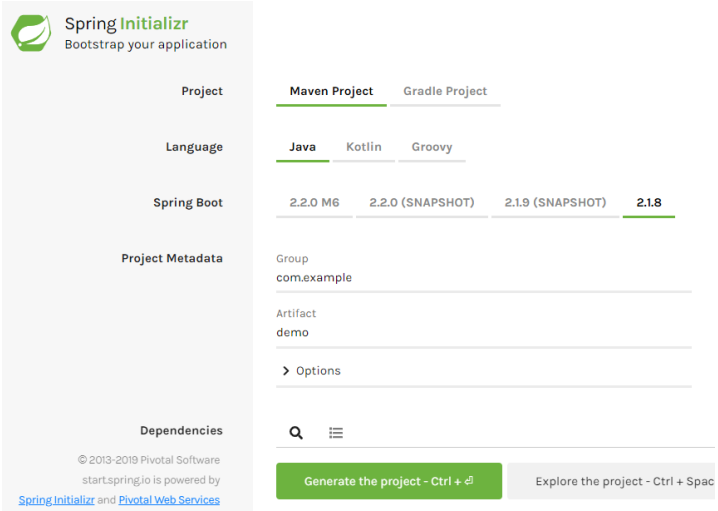


Ilustración 3: Spring Initializr

Como se puede apreciar en la ilustración es muy fácil crear una aplicación a través de Spring Initializr, basta con seleccionar las características, el lenguaje y las dependencias necesarias para el proyecto. Una vez configurado estas opciones se procede a presionar el botón "Generate project" e iniciara la descarga como un archivo .rar, luego descomprimimos el archivo e importar a través del IDE que se esté utilizando.

La segunda opción para crear una aplicación es a través de un IDE el cual facilita el manejo de proyectos con spring boot, siendo el más recomendable es SpringToolSuite.

Para la cual realizamos el siguiente proceso: nos dirigirnos al menú principal, seleccionar la opción File, New, Spring Starter Project. Una vez seleccionada esa opción se desplegará un menú en donde se seleccionará las características para nuestro proyecto, una vez realizado este proceso finalizar y el proyecto quedará lista para comenzar a trabajar.

2.7 Websockets

Los websockets admiten una comunicación interactiva y bidireccional entre el navegador del usuario y el servidor.

Con los WebSockets es necesario crear la conexión una vez, por lo que es llamada una conexión persistente. Una vez realizada la conexión se puede enviar datos en ambas direcciones de manera rápida y eficiente. Con estos webSockets se puede construir sitios web de forma rápida, además es muy fácil de entender y razonar cuando estamos escribiendo código (Formativa, 2016).

Los WebSockets no deben ser utilizados en todos los sitios web, pues en web simple e informativa realmente no es necesario. WebSockets son interactivos, es por esto que en este proyecto se requiere mantener esa comunicación debido a que el índice de radiación UV tendrá siempre una variación siendo imprescindible la visualización de los datos en tiempo real (Formativa, 2016).

2.8 HTTP

HTTP de HyperText Transfer Protocol (Protocolo de transferencia de hipertexto) es la forma más utilizada para el intercambio de información en la web, mediante éste se transfieren las páginas web a un ordenador (Masadelante, 2019).

El protocolo de transferencia es el método a través del cual se transfiere información entre los servidores y los clientes (por ejemplo los navegadores) (Masadelante, 2019).

Existe una manera segura de transferencia de información que se llama https, que utiliza cualquier método de cifrado mismo que debe ser entendido por el servidor y el cliente (Masadelante, 2019).

2.9 Web Services

Los web services están constituidos por una serie de protocolos y estándares que tienen como finalidad el intercambio de información entre sistemas. Diferentes servicios de software en diferentes lenguajes de programación, y realizadas en cualquier soporte, además son capaces de usar los servicios web para el intercambio de información a través de internet (Culturación, 2019).

Se puede expresar claramente que un web services es una función que son utilizados por diferentes equipos o servicios; con tan solo enviar indicadores al servidor (sitio en donde se encuentra incrustado el web services) y éste contestará a la petición (Culturación, 2019).

Los Web Services permiten que diferentes aplicaciones de distinto origen se comuniquen entre sí sin la necesidad de escribir programas costosos, esto se debe a que la comunicación se realiza mediante XML. Los Web Services no están adheridos a ningún sistema operativo o Lenguaje de Programación. Para una mejor comprensión se exponen algunos ejemplos, un programa escrito en Java puede conversar con otro escrito en Pearl; Aplicaciones Windows puede conversar con aplicaciones Unix. Además los Web Services no requieren utilizar browsers (Explorer) ni el lenguaje de especificación HTML (Saffirio Mario, 2006a).

Existen varias características que permiten la utilización de los servicios web en las aplicaciones como.

- Ayudan en la interoperabilidad entre servicios de software de forma independiente de sus dominios o de las estructuras sobre las que se instalen (Culturación, 2019).
- Los servicios Web impulsan los modelos y actas que se basan en relatos, simplificando el ingreso al contenido siendo más fácil comprender su funcionamiento (Culturación, 2019).

- Al respaldarse con HTTP, los servicios Web usan los sistemas de convicción firewall sin la obligación de reformar las reglas de filtrado (Culturación, 2019).
- Aceptan que los servicios y software de diversas empresas situadas en distintos sitios geográficos se combinen con facilidad con el objeto de proveer servicios integrados (Culturación, 2019).
- Admiten la interoperabilidad entre diversos sistemas de diferentes productores a través de reglamentos común y libre. Las determinaciones son negociadas por una regulación abierta, la W3C, por lo que no existe secretos por agrado de productores concretos garantizando de esta manera la plena interoperabilidad entre aplicaciones (Culturación, 2019).

Los Web Services son establecidos a través de una gran variedad de tecnologías que trabajan de forma coordinada con los nuevos modelos, de esta manera brinda mayor seguridad y operatividad, haciendo uso compuesto de diversos Web Services, libre de sus proveedores (Saffirio Mario, 2006b).

Los modelos que están empleando los Web Services son los siguientes:

XML: Extensible Markup Language. desarrollado por W3C que permite a los programadores crear sus propios tags, de esta manera se puede preparar descripciones, validaciones, concesiones y comentarios de información entre sistemas y organizaciones (Saffirio Mario, 2006b).

SOAP: Compendio de Simple Object Access Protocol, construido por XML, es un reglamento de correo que se utiliza para codificar datos de los mandatos de los Web Services además sirven para contestar los mensajes antes de enviarlos a través la red. Es independiente de cualquier sistema operativo y pueden ser transportados por otros protocolos de la Internet, como ser: SMTP, MIME y HTTP (Saffirio Mario, 2006b).

WSDL: Extracto de Web Services Description Language, lenguaje establecido en XML, es usado para determinar los Web Service como compilaciones de punto de trato y son competentes en intercambiar notas. El WSDL es la parte global de UDDI y parte del registro global de XML, o sea; es un modelo de uso (Saffirio Mario, 2006b).

UDDI: Extracto de Universal Description, Discovery and Integration. Es una convención de la web que permite a las compañías difundir sus sitios, para que otras empresas conozcan y utilicen los Web Services que publican, tienen un funcionamiento parecido a las páginas amarillas (Saffirio Mario, 2006b).

2.10 Desarrollo de aplicaciones Móviles

Una aplicación (también llamada app) es un programa informático que ha sido desarrollado para facilitar una tarea en un dispositivo informático. Cabe resaltar que, aunque todas las aplicaciones son programas, no todos los programas son aplicaciones. Existe gran variedad de software en el mercado, pero sólo es una aplicación aquel que ha sido creado con un fin determinado, para realizar tareas específicas. No se consideraría una aplicación, por ejemplo, un sistema operativo, ni una suite, ya que tienen un propósito es general (Robertho Artica, Astengo, 2014).

Las aplicaciones surgen debido a una necesidad específica de los usuarios, para realizar tareas que el programador ha detectado cierta necesidad. Pero las aplicaciones también pueden responder a obligaciones lúdicas, también de laborales. Se suele decir que para cada problema hay una solución, del mismo modo en el campo de la informática, para cada problema hay una aplicación (Robertho Artica, Astengo, 2014).

2.10.1 Frameworks móviles

Existen muchos framework móviles que permiten crear aplicaciones haciendo el uso de HTML, CSS, JavaScript.

Para la creación de una aplicación hay gran variedad de programas o lenguajes que se deben tener en cuenta, entre los framework móviles más conocidos y utilizados tenemos: React Native, Framework 7, Phone Gap, Ionic.

Se ha realizado un cuadro de ponderacion de los frameworks móviles más conocidos destacando sus virtudes y cadencias según sus caracteristicas que se reflejan en la sisguiente tabla 3, las mismas que puedes verse a continuación.

	Librerías Plugins	**Comunidad fuerte**	**Integración de node**	**Fácil aprendizaje**	**Docume-ntación**	**CLI**	**Eficiencia**	**Total**
React native	X	X	X	X		X	X	6
Framework 7				X	X		X	3
Phone gap	X			X	X			3
Ionic	X	X	X	X	X	X		6

Tabla 3: Framework móviles

Después de haber especificado en el cuadro de ponderación cada una de sus virtudes y defectos se ha llegado a la conclusión de elegir Ionic para el desarrollo del proyecto por sus grandes características que son de gran aporte al momento del desarrollo.

2.10.1.1 Ionic

Es uno de los frameworks más famosos usados para desarrollar aplicaciones híbridas. En sus inicios Ionic usaba el framework AngujarJS y PhoneGap en plataformas móviles. Pero, en sus últimas versiones Ionic 4 ha incorporando nuevos frameworks Front-End para poder desarrollar. En la actualidad admite su desarrollo con los más conocidos como: Angular, React, Vue.JS.

Con esta información que se ha recopilado se puede observar la ventaja y variedad de características que nos ofrece el framework Ionic con respecto a los demás frameworks, uno de los más claros ejemplos son sus plugins que nos facilitan de gran manera para el desarrollo de este proyecto, permitiéndonos incluso soporte a un gigantesco framework como es AngularJS.

2.11 Almacenamiento

Para almacenar y procesar grandes cantidades de información, las alternativas de bases de datos relacionales podrían no satisfacer todas las necesidades requeridas (McCreary et al.,2014). Es ahí donde las bases de datos NoSQL entran en acción debido a la flexibilidad que proporcionan al momento de almacenar y gestionar los datos, siendo esto muy importante en el ámbito de IoT (Paulista, 2015).

2.11.1 Bases de Datos NoSQL

Con la aparición de la web 2.0 surge el término NoSQL que hasta la actualidad únicamente publicaban contenido en la red ciertas compañías que disponían de un portal, sin embargo con el surgimiento de servicios como Facebook, Youtube, o Twitter, cualquiera tenía la facultad de subir información, ocasionando un aumento bárbaro de los datos (Whitepaper, 2014).

Es ahí donde surgen las principales dificultades del manejo de todos esos datos guardados en bases de datos relacionales. Desde un inicio, para remediar aquellos dilemas de accesibilidad, las compañías debían usar una mayor cantidad de máquinas que luego se dieron cuenta de que esto no solucionaba el problema, y era muy costosa. Había otra resolución mediante la fundación de servicios destinados a un deterioro

especial que con el transcurso del lapso han dado oportunidad a soluciones robustas, manifestándose así la actividad NoSQL (Whitepaper, 2014).

2.11.2 Bases de datos NoSQL en microservicios

Una diferencia muy importante entre las bases de datos de NoSQL y las bases de datos relacionales tradicionales, es el hecho de que NoSQL es una forma de almacenamiento no estructurado.

Esto quiere decir que NoSQL carece de una estructura de tabla fija como las que se encuentran en las bases de datos relacionales.

2.11.3 Análisis de Base de datos NoSQL

Luego de haber revisado las ventajas del manejo de las bases de datos NoSQL se puede conocer que existe una gran variedad base de datos NoSQL entre las más conocidas y las más destacadas tenemos (Cassandra, Redis, CouchDB, MongoDB) las mismas que se especificaran a continuación en la tabla 4 mediante un cuadro de ponderación de acuerdo a sus caracteristicas.

	De código abierto	**Escalabilidad**	**Replicación**	**Total**
Cassandra	X	x		2
Redis	x		x	2
Couchdb	x	x	x	3
Mongodb	x	x	x	3

Tabla 4: Análisis de Base de Datos noSQL

Una vez detallado en el cuadro de ponderación cada una de las ventajas que brindan cada una de las bases de datos NoSQL, se ha conocido que las bases de datos más destacadas son CouchDB y Mongodb por lo que se ha tomado la decisión de elegir mongodb para la realización del presente proyecto.

2.11.3.1 Mongodb.

Mongodb es una base de datos multiplataforma libre, orientada a documentos, que se añaden datos o información deseados en documentos en lugar de que se almacene en registros como lo hacen en otras bases de datos.

Dentro de las bases de datos NoSQL, Mongodb. Posee un concepto distinto al de las bases de datos relacionales, pues se está transformando en una alternativa muy interesante (Rubenfa, 2014).

Pero cuando uno se inicia en Mongodb se puede sentir perdido. No existen tablas, ni tampoco registros y ni SQL. A pesar de eso, Mongodb es una digna candidata para almacenar los datos de nuestras aplicaciones (Rubenfa, 2014).

2.11.3.2 ¿Cómo funciona mongodb?

Mongodb está escrito en C++, a pesar que las consultas se realizan transportando objetos JSON como parámetro. Es algo demasiado deductivo, ya que los propios documentos se almacenan en BSON (Rubenfa, 2014).

Mongodb está constituida por una consola construida sobre Javascript, desde donde se ejecutan los diferentes comandos, y las consultas se realizan utilizando ese lenguaje. Además de las funciones de mongodb, se puede usar diversas funciones propias de JavaSciprt. En la consola también es posible definir variables, funciones o utilizar bucles (Rubenfa, 2014).

2.11.3.3 ¿Dónde se puede utilizar Mongo?

A pesar de que se dice que las bases de datos NoSQL tienen un ámbito de aplicación reducido, MongoDB puede ser usado en muchos de los proyectos que se desarrollan actualmente (Rubenfa, 2014).

A pesar, de que las colecciones de MongoDB no necesitan definir un esquema, es indispensable diseñar nuestra aplicación para seguir uno. Por lo que se debe tomar en cuenta si es necesario normalizar los datos, denormalizarlos o usar una aproximación híbrida. Todas estas decisiones pueden afectar al rendimiento de la aplicación. Es decir, el esquema lo definen las consultas que se realicen con mayor frecuencia (Rubenfa, 2014).

MongoDB es totalmente beneficiono en ambientes que requieren escalabilidad, ya que establecen opciones de replicación y sharding, muy sencillos de formar, por lo que es probable adquirir un sistema que escale horizontalmente sin problemas (Rubenfa, 2014).

2.12 Metodología para el desarrollo de software

Al desarrollar un software se nos puede presentar uno que otro inconveniente y para desarrollar un sistema de calidad y que sea satisfactorio para el cliente es sumamente necesario el uso de una metodología de acuerdo al tipo de proyecto.

A continuación en la tabla 5 se puede observar las principales diferencias entre metodologías agiles y tradicionales.

Metodología Ágil	Metodología Tradicional
Fundamentadas en heurísticas originarias de destrezas de realización de código	Basadas en normas provenientes de modelos continuos por el ambiente de desarrollo
Principalmente dispuestos para alteraciones en el transcurso del proyecto	Cierta resistencia a los cambios
Instruido internamente (por el equipo)	Instruido externamente
Desarrollo menos inspeccionado, con pocos principios.	Proceso mucho más revisado, con abundantes políticas/normas

Tabla 5: Metodologías para el desarrollo de software

Se ha seleccionado la metodología ágil ya que esta nos permite que el cliente sea parte del desarrollo, así como también podemos decir que esta metodología no permite disminuir costos y facilitar flexibilidad a los proyectos de software.

2.12.1 Metodologías Ágiles

El método Ágil es un proceso mediante el cual al equipo da respuestas rápidas e impredecibles a las valoraciones que reciben sobre su proyecto. Este método crea oportunidades de evaluar la dirección de un proyecto durante el ciclo de desarrollo. Los equipos calculan el proyecto en reuniones moderadas, llamadas sprints o iteraciones (Beedle, 2017).

A continuación en la tabla 6, se puede conocer la principal diferencia entre XP y Scrum.

XP	SCRUM
Los Equipos de XP normalmente trabajan con iteraciones muy cortas (1 o 2 semanas).	Los equipos de Scrum no permiten cambios en sus sprints.
Los equipos de XP trabajan en un orden estricto	Scrum no recomienda ninguna buena práctica de desarrollo. XP si lo hace.

Tabla 6: Metodologías ágiles

2.12.1.1 Scrum

Scrum es un proceso mediante el cual se puede aplicar de forma regular un conjunto de buenas prácticas para trabajar de forma colaborativa, es decir un trabajo en equipo, para obtener un buen resultado. Estos hábitos se protegen mutuamente y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos (Scrum, 2019).

2.12.1.2 Equipo Scrum

El equipo Scrum está formado por 3 roles.

- **El Product Owner o Dueño del producto:**

El Product Owner es la persona que se encarga de verificar que el equipo aporte valor al negocio. Interpreta las partes atraídas internas y externas, por lo que debe entender y patrocinar las necesidades de todos los usuarios en el negocio, así como también las necesidades y el funcionamiento del Equipo Scrum.

- **El Scrum Master:**

(Bass, 2014) Afirma que el scrum master actúa como facilitador para los equipos de desarrollo de software, fomentando la adherencia a las prácticas ágiles y eliminando los impedimentos para los miembros del equipo. Pero en grandes proyectos, los maestros scrum a menudo trabajan en equipos distribuidos geográficamente. Los maestros de Scrum utilizan la planificación de sprints con el fin de evitar tareas de desarrollo que se superponen a los límites del equipo, coordinan el estado y el esfuerzo entre los equipos e integran bases de códigos.

- **Equipo de desarrollo**

Los elementos del Equipo de Desarrollo, son los expertos de cada sector, que efectúa la entrega los elementos del Backlog, y de la gestión de sus propios esfuerzos (Miguel, n.d.).

Un equipo de desarrollo debe ser multifuncional: es decir debe tener capacidad para realizar cada uno de los elementos del Backlog de Producto de la A la Z. Además debe auto-organizarse, ser responsable y capaz de encontrar su propio camino en lugar de recibir órdenes, y al respecto deberá alinearse con el propósito del proyecto (Miguel, n.d.).

1.4.1 Artefactos de un Scrum

Los artefactos de Scrum permiten al equipo encargado del proyecto a tener la misma visión del proyecto.

a) Backlog de Producto / Product Backlog.

Es un balance que mantiene diferentes clases de tareas que sea necesario realizar en la manufactura: requisitos, casos de uso, labores y sujeciones. Es la primordial causa de información sobre el resultado en Scrum, una lista, en cualquier formato, que contiene todos los requerimientos que necesitamos implementar en el producto (Roche, 2020).

b) Backlog del Sprint/Sprint Backlog.

Es un registro que posee distintas clases de tareas que se deba realizar en el beneficio: requisitos, casos de uso, labores y sujeciones (Roche, 2020).

Es la primordial causa de aclaración sobre el trabajo en Scrum, una lista, en cualquier formato, que contiene todos los requerimientos que necesitamos implementar en el producto (Roche, 2020).

c) Incremento.

Si Scrum debiera ser transformado a una sola entidad, consistiría en proporcionar una porción de software realizado en cada Sprint. Un Incremento es la consecuencia del Sprint, es el monto total de todas las tareas realizadas, casos de uso, historias de usuario y cualquier componente que se haya realizado durante el Sprint y que se pondrá a orden del beneficiario final en estructura de software, brindando un atractivo de negocio al resultado que se está desarrollando (Roche, 2020).

2.12.1.3 Proceso Scrum

En este proceso un proyecto se ejecuta por medio de ciclos cortos temporales con una duración fija (iteraciones que generalmente duran 2 semanas, pero en algunos equipos son de 3 y 4 semanas, que es el límite máximo de feedback de producto real y reflexión). Cada iteración debe entregar un resultado completo, un aumento de producto final que sea susceptible de ser

entregado con un mínimo de esfuerzo al cliente cuando lo solicite. A continuación, se puede observar en la ilustración 4, cómo se manejan los procesos en scrum.

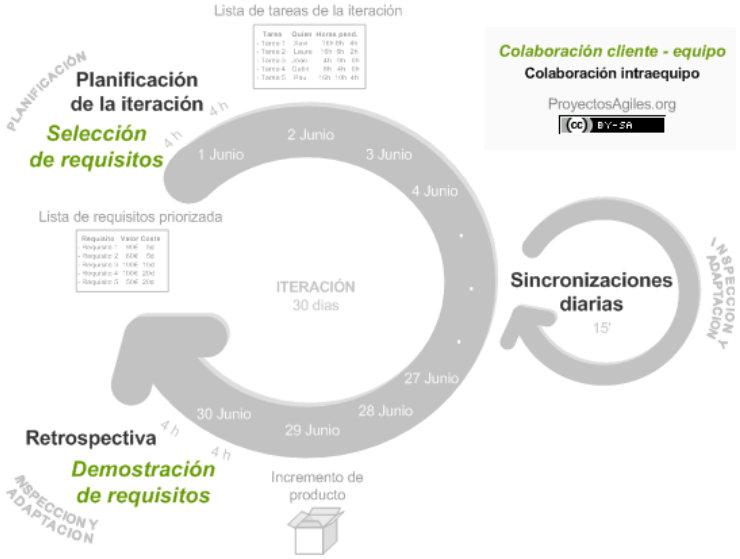


Ilustración 4: Proceso Scrum

Fuente 2: Fuente 3: Tomado de(https://www.scrum.org/resources/blog/que-es-scrum)

a) Planificación del Spring

En esta fase se define el Product Backlog. Esta planificación se divide en dos secciones.

- **Selección de requisitos.**

El beneficiario muestra al personal una lista de requerimientos preferenciados del producto. El grupo interroga al beneficiario sobre sus dudas y elige los requisitos que prevé más importantes para completar la iteración, de manera que puedan ser entregados si el cliente lo solicita.

- **Planificación de la iteración.**

El grupo crea una nómina de labores indispensables para la elaboración de los requisitos seleccionados. La estimación de esfuerzo se realiza en conjunto y los todos miembros del equipo se auto asignan las tareas, se auto-organizan para trabajar incluso en parejas con el fin de brindar conocimiento o para resolver juntos objetivos especialmente complejos.

b) Ejecución del Sprint

El equipo realiza una reunión de sincronización diariamente por un lapso de 15 minutos, generalmente delante de un tablero físico o pizarra. El equipo inspecciona el trabajo de los demás para luego hacer las adaptaciones respectivas que permitan cumplir con la previsión de objetivos a mostrar al final de la iteración. En la reunión cada integrante del equipo debe responder a tres preguntas:

¿Qué se ha realizado desde la última reunión?

¿Qué se va a realizar a partir de ese instante?

¿Cuáles son los impedimentos que nos dificultan cumplir los objetivos?

c) Revisión del Sprint o Reunión Demo

El equipo muestra al cliente los requisitos terminados durante la iteración, en forma de incremento de producto elaborado que será entregado con el mínimo esfuerzo. En función de los resultados mostrados y de los cambios que haya habido en el contexto del proyecto, el beneficiario ejecuta las adecuaciones necesarias de forma imparcial, ya desde la primera iteración, replanificando el proyecto.

CAPÍTULO 3

3.1 Escenario

Para realizar el presente trabajo se ha utilizado la metodología de desarrollo de software denominado Scrum, la cual brinda la facilidad de dividir en iteraciones de corta duración llamadas Sprints. Una de los beneficios de esta metodología es que permite la selección de los requerimientos de una lista denominada como historia de usuarios.

Cabe recalcar que en el presente trabajo de tesis se realizará un prototipo de medición de índice UV mediante la implementación de microservicios para plataformas IoT.

3.2 Implementación de la metodología

Para iniciar la implementación es sumamente necesario especificar el Product Backlog, en esta sección se encontraran enlistados los requerimientos de los usuarios, conocidos como historias de usuarios.

Cada historia tiene:

- ✓ Código
- ✓ Nombre de la historia
- ✓ Comprobar
- ✓ Estimación
- ✓ Importancia
- ✓ Comentario

A continuación en la tabla 7 se puede observar los requerimientos del proyecto:

Tabla 7: Product Backlog

Product Backlog						
Id	**Nombre**	**Comprobar**	**Sprint**	**Importancia**	**Estimación**	**Comentario**
1	Obtención de requerimientos e instalación de las herramientas	Conocer uno a uno cuales son los requerimientos e instalar las herramientas que se utilizarán.	1	5	2	Instalación de herramientas para la ejecución del presente trabajo
2	Creación de microservicio en Java con Spring Boot	Entrar a postman; hacer un get y un post con los métodos creados en spring boot	2	4	4	Se podrá realizar consultas y posteos a través de postman para comprobar su funcionalidad
3	Consumir el microservicio spring boot desde Arduino	Realizar un posteo a través de arduino, consumiendo los métodos creados en spring boot	3	3	4	Posteo desde arduino y visualización en base de datos mongodb
4	Desarrollo de página web y aplicación móvil con Ionic	Podremos visualizar el último dato que fue posteado desde arduino y almacenado en la base de datos mongodb.	4	2	4	Visualización del último dato almacenado en mongodb a través de Ionic

5	Microservicio con websockets	Una vez implementado los websockets se podrá visualizar en tiempo real cuando un nuevo dato es almacenado en la base de datos.	5	1	4	Visualización en tiempo real cuando un nuevo valor es ingresado a la base de datos.

3.3 Desarrollo del sistema

Una vez definido cada uno de los requerimientos en el product backlog se iniciará con el desarrollo del sistema de microservicios web para plataformas IoT.

Se iniciará con el primer proceso definido dentro del product backlog en la Tabla 6. Dentro de cada requerimiento se encuentra especificado la estimación de esfuerzo y además la importancia que requiere para cumplir con el objetivo. En este proceso se encuentra definido los siguientes detalles para la importancia y la estimación.

- **Importancia:** rangos entre 1 y 5
- **Estimación:** rangos entre 2 y 4

Con el product backlog y cada una de las tareas que deben realizarse, se detallará cada una de las iteraciones.

3.4 Estructura de la programación

En la tabla 8 se especifica las normas de la estructura de la programación, la misma que en un futuro nos facilitará de gran manera si se necesita implementar nuevos requerimientos u objetivos.

ESTRUCTURA DE LOS PAQUETES	
Arquitectura	**Contenido**
API	Ejecutable
Modelo	Aquí se encuentran las clases
Repositorio	JPA
Recursos	Aquí se encuentran los métodos

Tabla 8: Estructura de los paquetes

3.5 Despliegue del Sprint 1

En el Sprint 1 se detallará cada uno de los requerimientos de los usuarios y se realizará la instalación de las herramientas que son necesarias para el desarrollo del presente trabajo.

3.5.1 Desarrollo del Sprint 1

Para el desarrollo del sprint 1 se ha definido la duración de este sprint para 2 semanas, es decir 40 horas.

3.5.2 Sprint Backlog 1

En la tabla 9 se definen las tareas que deben ser realizadas para completar la historia de usuario, las mismas que se encuentran detalladas en la tabla.

Tabla 9: Spring Backlog 1

Sprint Backlog 1						
Id	**Nombre**	**Comprobar**	**Sprint**	**Importancia**	**Estimación**	**Comentario**
1	Obtención de requerimientos e instalación de las herramientas	Conocer uno a uno cuales son los requerimientos e instalar las herramientas que se utilizarán.	1	5	2	Instalación de herramientas para la ejecución del presente trabajo

A continuación en la tabla 10 se puede visualizar la pila de tareas del sprint 1.

Tabla 10: Pila de tareas del sprint 1

Elemento información	**Sprint** 1	**Fecha Inicio** 14-06-2019	**Duración** 2 semanas(40hrs)	Actualización	Días	14-18 jun	21-25 jun	28/jun-02/jul
Delegado/Autor: Raúl Rivera								
H. Usu	**Tarea**			**Estado**	h	20	20	20
1	Instalación de mongodb			Completado		X		
1	Instalación de MongoDB Compass			Completado		X		
1	Instalación de SpringToolSuite4			Completado		X		
1	Instalación de Arduino			Completado			X	
1	Instalación de Visual Studio Code			Completado			X	
1	Instalación de NodeJs			Completado				X
1	Instalación de Postman			Completado				X

3.5.3 Burn Down Chart 1

En la figura 7 se puede observar la gráfica del avance del proyecto al finalizar el sprint 1.

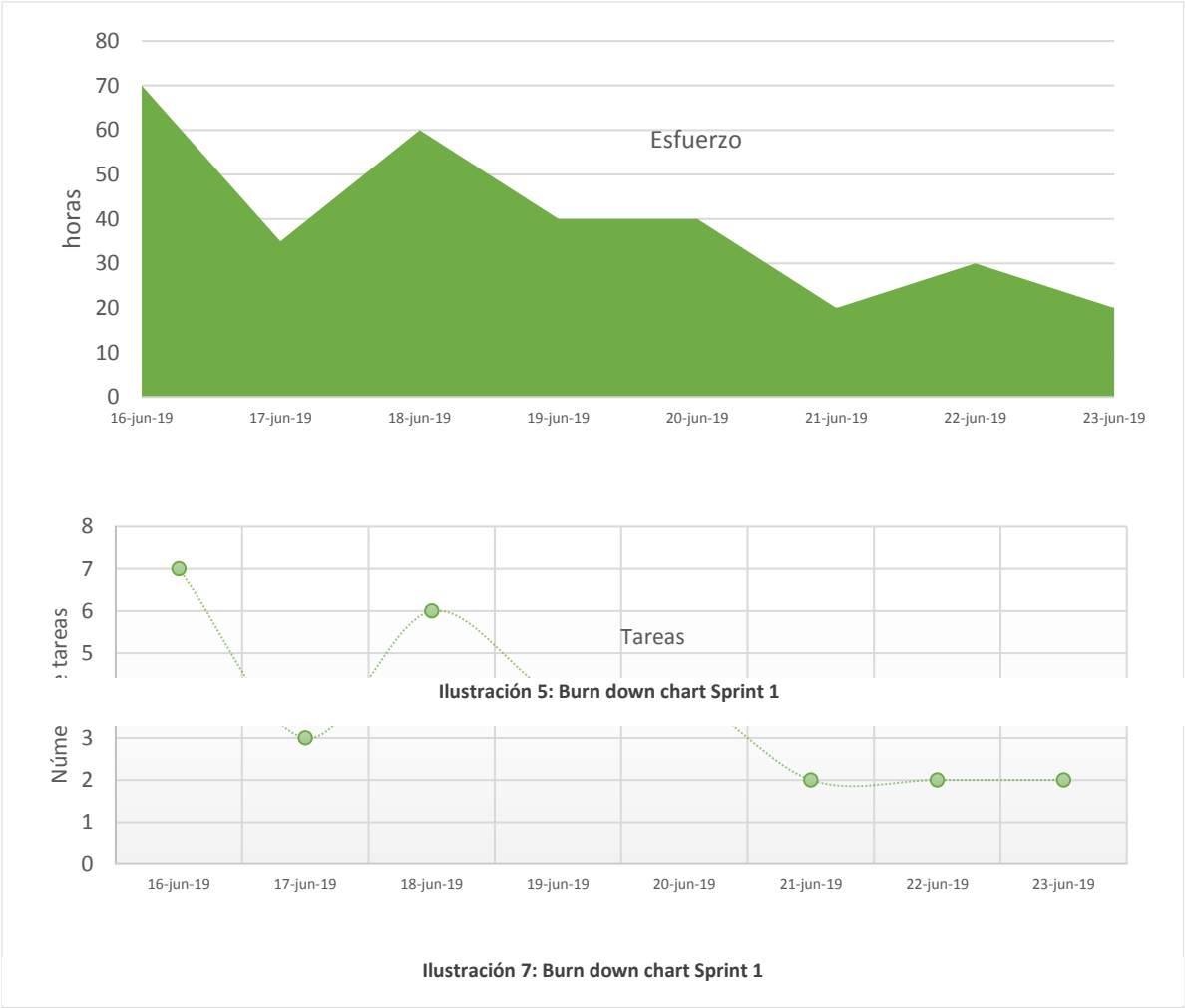


Ilustración 5: Burn down chart Sprint 1

Ilustración 7: Burn down chart Sprint 1

Ilustración 6: Burn down chart Sprint 1

3.5.4 Justificación del Sprint 1

En esta sección se evidencia los resultados del sprint 1, en las siguientes ilustraciones de los requerimientos presentados.En la ilustración 8 se puede observar la instalación de Mongodb.

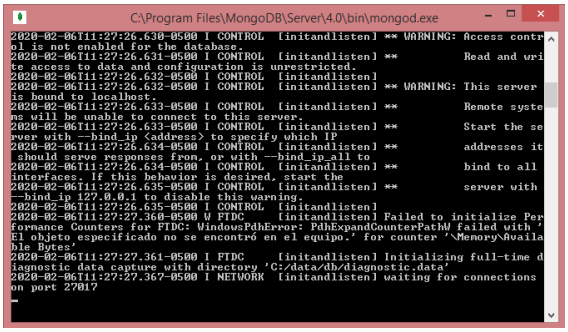


Ilustración 8: MongoDB

En la ilustración 9 se puede visualizar la instalación de mongodb Compass.

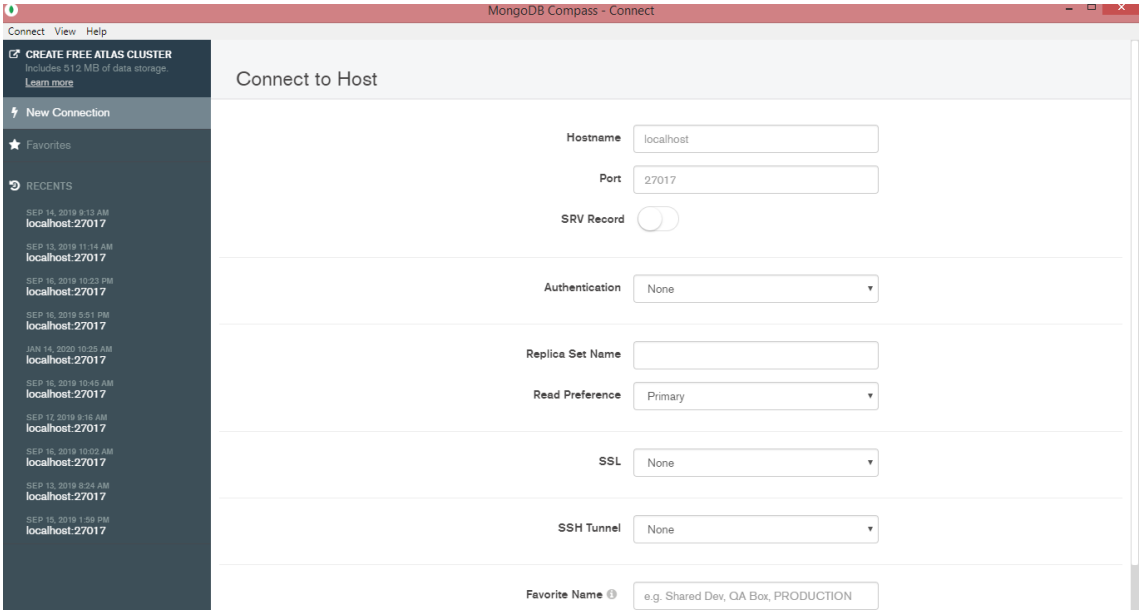


Ilustración 9: MongoDBCompass

De igual manera en la ilustración 10 se puede observar la instalación de Spring Tool suite.

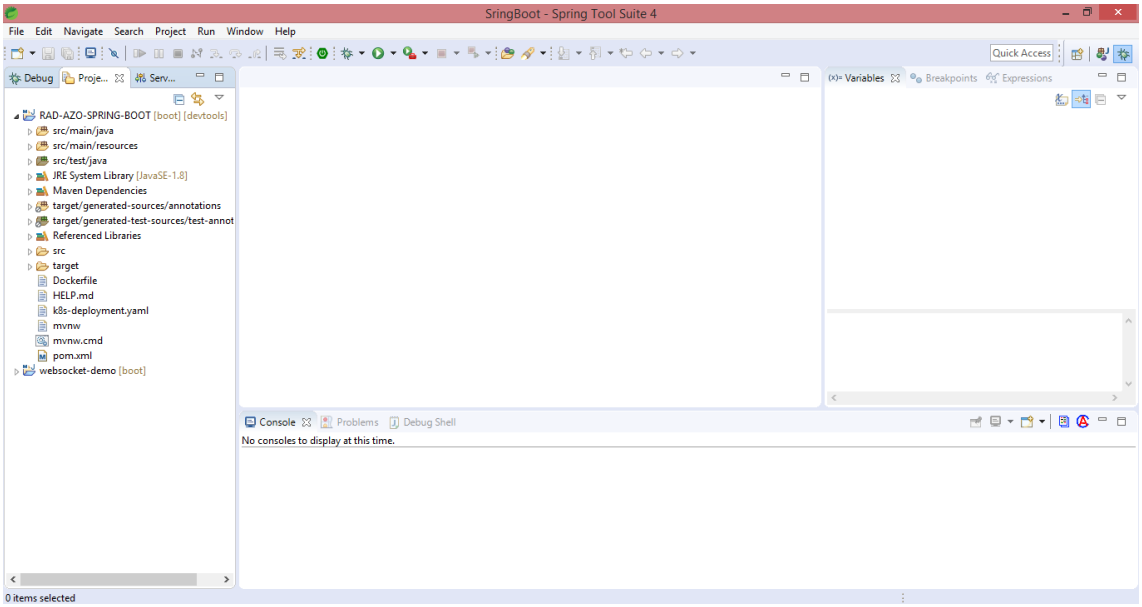


Ilustración 10: Spring Tool Suite 4

En la ilustración 11 se puede observar la ventana de trabajo arduino.

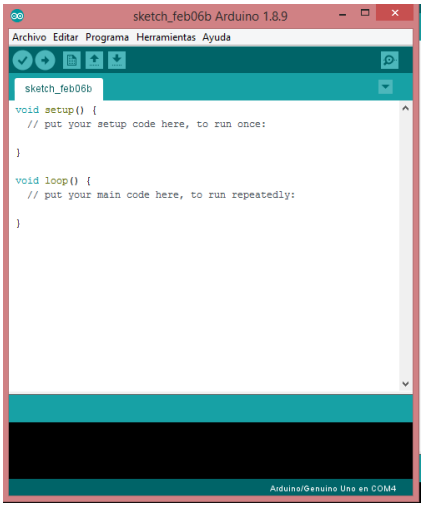


Ilustración 11: Arduino

Así también en la ilustración 12 se observa la instalación de Visual Studio Code.

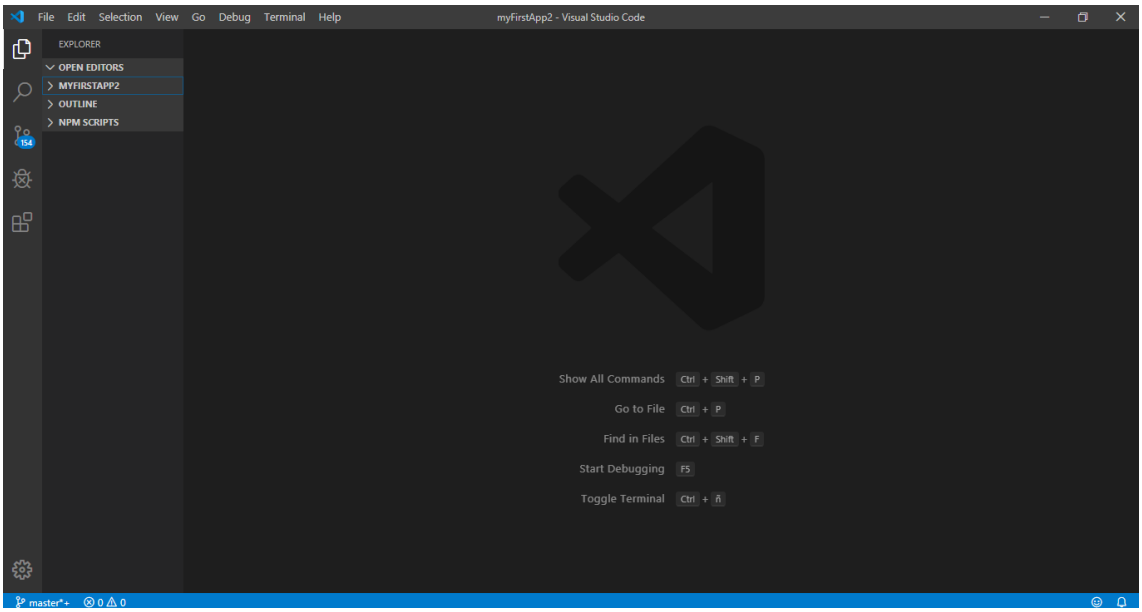


Ilustración 12: Visual Studio Code

En la ilustración 13 se observa la instalación de node js.

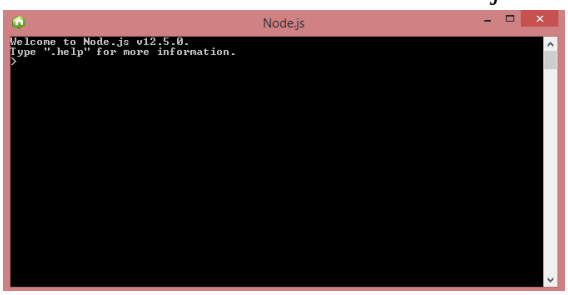


Ilustración 13: Node js

En lo referente a la instalación de las herramientas tenemos la instalación de postman que se puede visualizar en la ilustración 14.

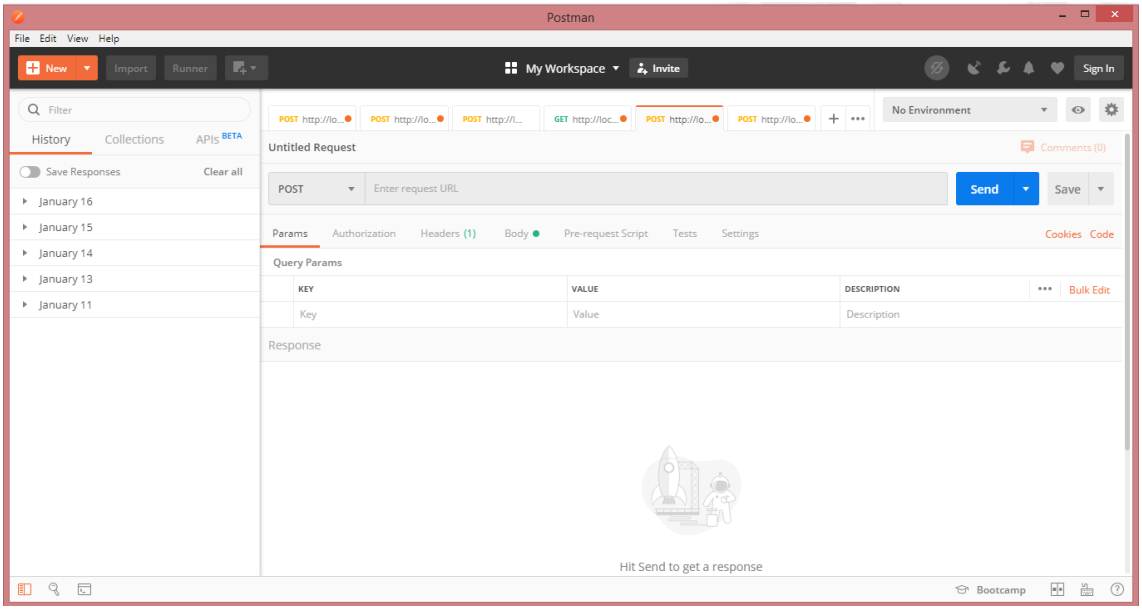


Ilustración 14: Postman

3.6 Despliegue Sprint 2

En el spring 2 se detalla la creación del primer microservicio que se realizará en java mediante Spring Boot. El microservicio que permitirá crear los métodos para postear los datos en MongoDB, así como también mediante los métodos se podrá realizar consultas.

3.6.1 Desarrollo del Sprint 2

Para el desarrollo del sprint 2 se ha definido la duración de este sprint para 4 semanas, es decir 80 horas.

3.6.2 Sprint Backlog 2

A continuación en la tabla 11 se puede observar el Spring Backlog 2.

Tabla 11: Spring Backlog 2

Sprint Backlog 2						
Id	**Nombre**	**Comprobar**	**Sprint**	**Importancia**	**Estimación**	**Comentario**
2	Creación de microservicio en Java con Spring Boot	Entrar a postman; hacer un get y un post con los métodos creados en spring boot	2	4	4	Se podrá realizar consultas y posteos a través de postman para comprobar su funcionalidad

También se puede visualizar la pila de tareas del Sprint 2 en la tabla 12.

Tabla 12: Pila de tareas del sprint 2

Elemento información	**Sprint** 2	**Fecha Inicio** 08-07-2019	**Duración** 4 semanas(80hrs)	Actualización		Días	08-12 jul	15-19 jul
Delegado/Autor: Raúl Rivera								
H. Usu	**Tarea**				**Estado**	h	20	20
2	Creación del API				Completado		X	
2	Creación del modelo				Completado		X	
2	Creación del repositorio				Completado			X
2	Creación de los recursos				Completado			X

3.6.3 Burn Down Chart 2

En la figura 15 se puede visualizar la gráfica del estado del proyecto al finalizar el sprint 2.

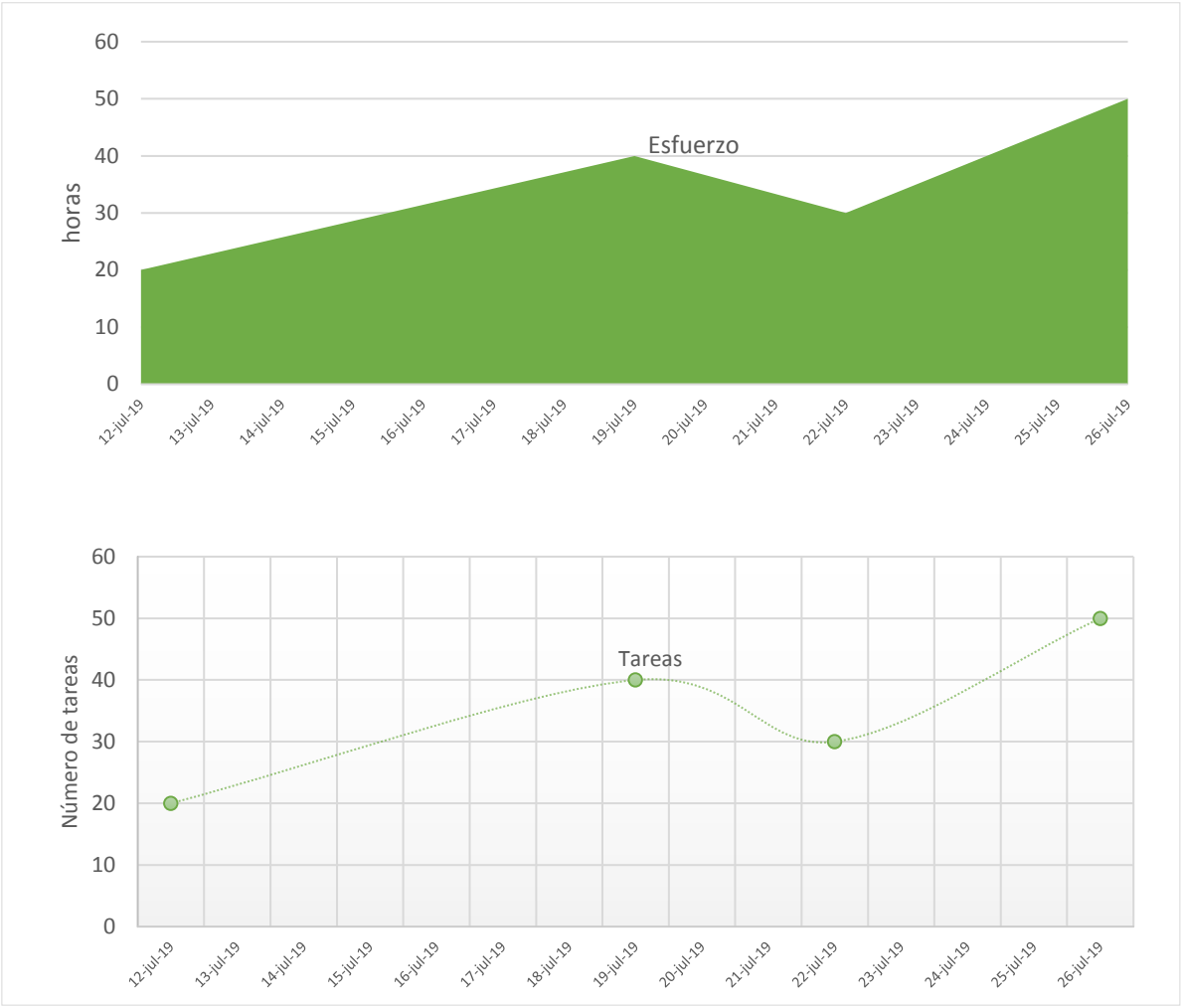


Ilustración 16: Burn down chart Sprint 2

En la ilustración 17 se puede visualizar la examinar la creación del modelado del proyecto.

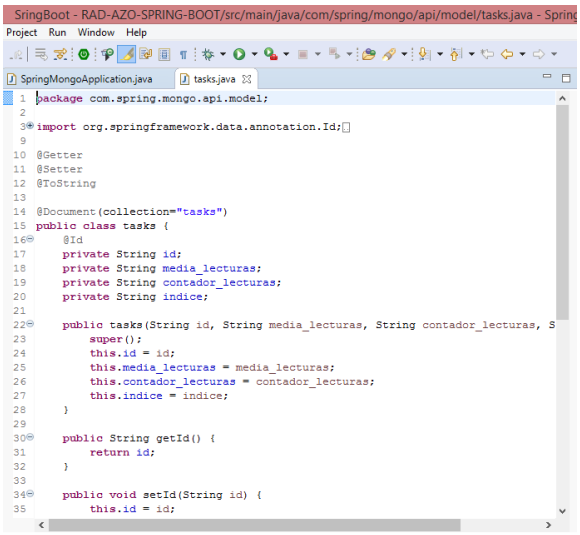


Ilustración 17: Modelo

También se puede observar el repository del presente trabajo en la ilustración 18.

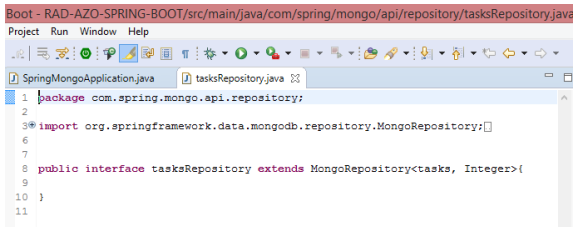


Ilustración 18: Repository

En la ilustración 19 se puede conocer los recursos para consumir los web sockets.

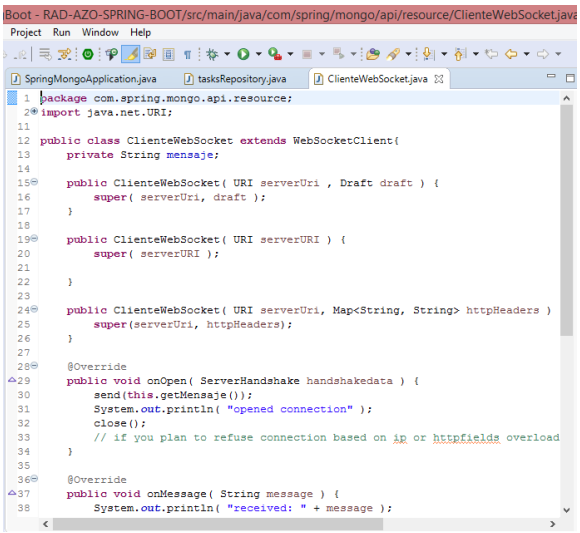


Ilustración 19: Resource cliente web socket

Y por último se puede observar en la ilustración 20 el controlador del presente trabajo.

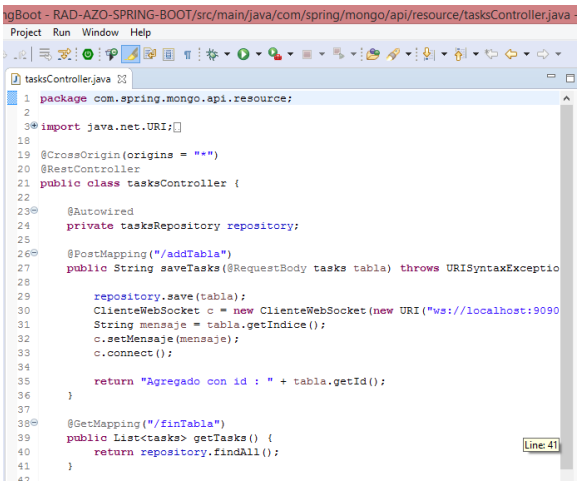


Ilustración 20: Resource controlador

3.6.5 Producto entregable del Sprint 2

Como resultado de las tareas realizadas en el presente Sprint se puede observar en la ilustración 21 un posteo desde la herramienta postman consumiendo los recursos creados en el microservicio.

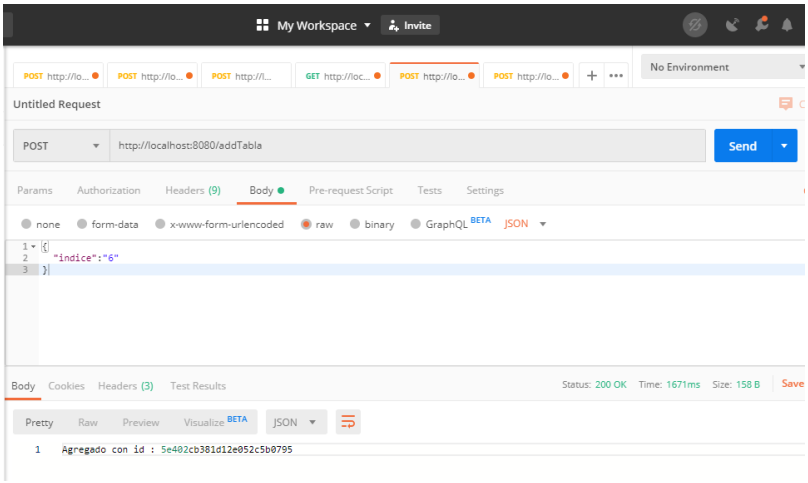


Ilustración 21: Posteo desde postman

De igual manera en la ilustración 22 se puede visualizar una consulta a través del método get a través de postman.

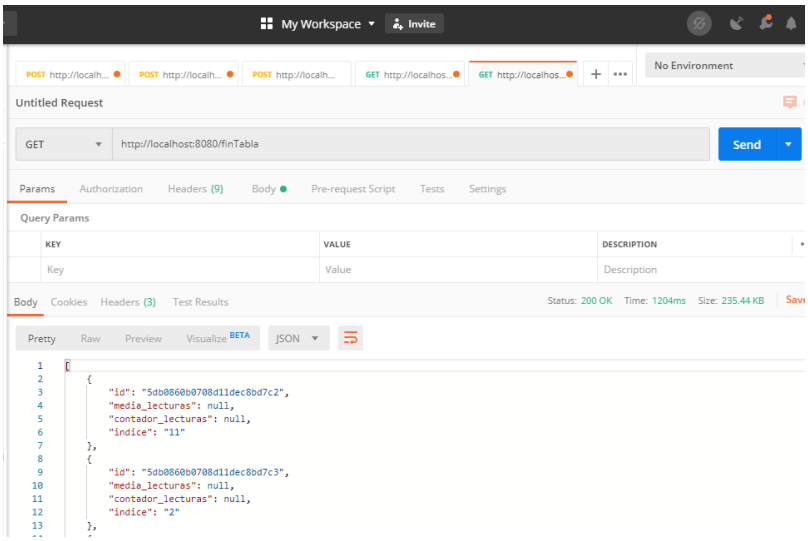


Ilustración 22: Get desde postman

3.7 Despliegue Sprint 3

En el actual spring se observa la creación del microservicio que permitirá realizar la obtención de los datos mediante un sensor, una vez obtenido el dato se realizará el posteo en la base de datos mediante la conexión con los métodos creados en el microservicio spring boot.

3.7.1 Desarrollo del Sprint 3

Para el desarrollo de este sprint se ha definido la duración de este sprint para 4 semanas, es decir 80 horas.

3.7.2 Sprint Backlog 3

A continuación en la ilustración 23 se puede observar el Sprint Backlog 3.

Ilustración 23: Sprint Backlog 3

Sprint Backlog 3						
Id	**Nombre**	**Comprobar**	**Sprint**	**Importancia**	**Estimación**	**Comentario**

3	Creación de microservicio en Arduino	Realizar un posteo desde arduino y comprobar en mongodb si efectivamente se posteó el dato.	3	3	4	Posteo desde arduino y visualización en base de datos mongodb

En la ilustración 24 se puede examinar cada una de las tareas a realizarse en el Sprint 3.

Ilustración 24: Pila de tareas del sprint 3

Elemento información	**Sprint** 3	**Fecha Inicio** 08-07-2019	**Duración** 4 semanas(80hrs)	Actualización	Días	26-30 Ago	02 – 06 Ago	09 – 13 Ago
Delegado/Autor: Raúl Rivera								
H. Usu	**Tarea**			**Estado**	h	20	20	20
3	Obtención de datos mediante sensor			Completado		X		
3	Posteo de datos en mongodb mediante arduino			Completado			X	

3.7.3 Burn Down Chart 3

En la figura 25 se puede observar la gráfica del avance del proyecto al finalizar el sprint 3.

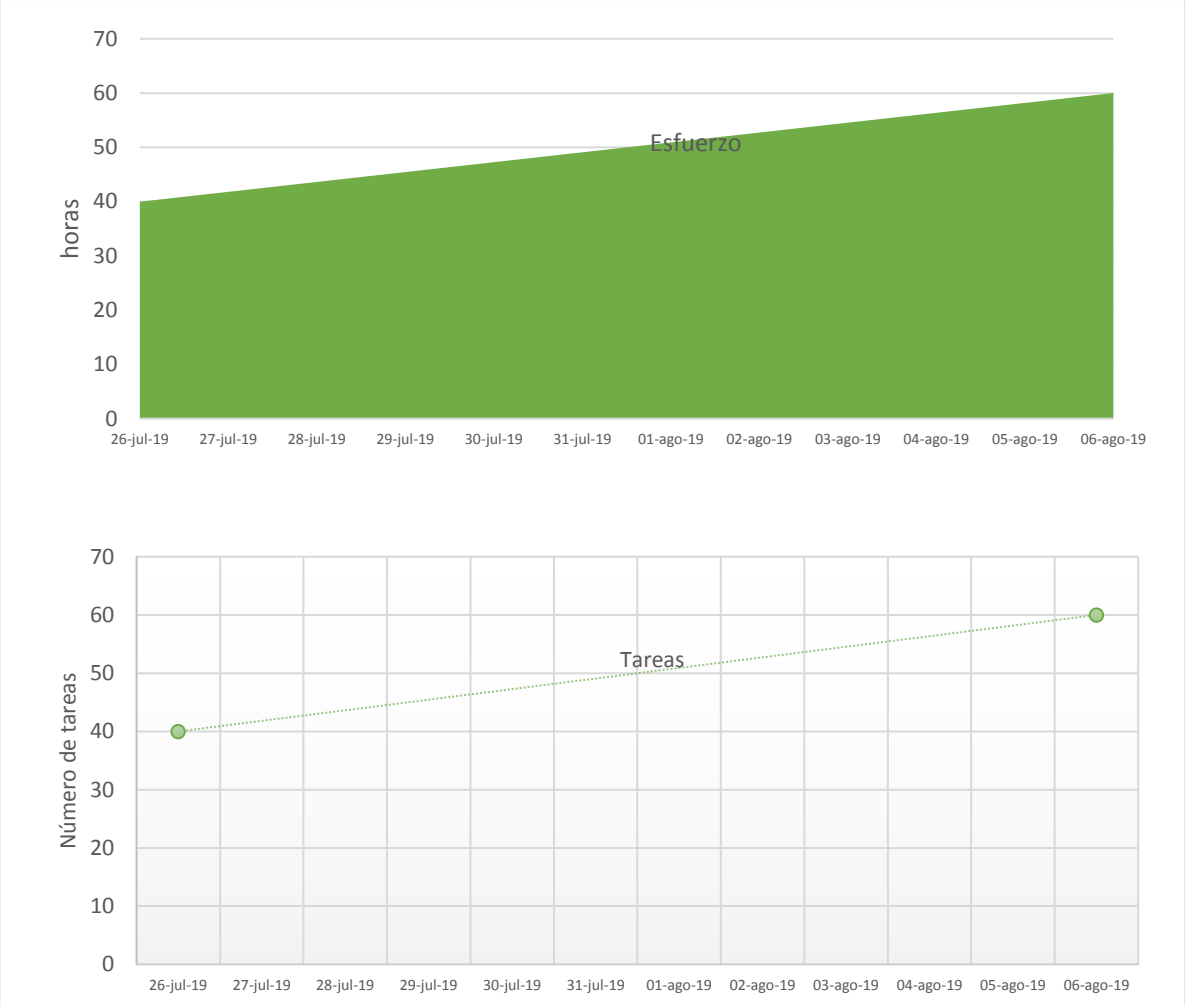


Ilustración 25: Burn down chart Sprint 3

En la graficas 26 se puede visualizar la programación para la obtención de datos desde arduino mediante un sensor UV.

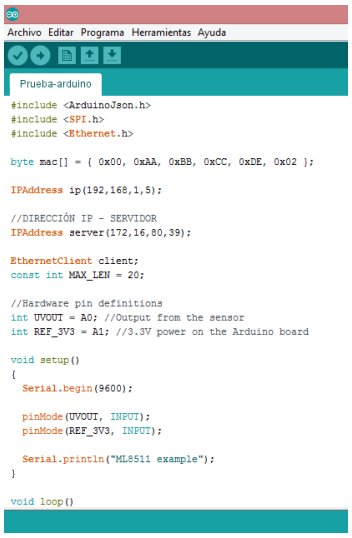


Ilustración 26: Obtención de datos en arduino

Una vez que se han obtenido los datos se realiza el posteo de datos consumiendo el microservicio creado anteriormente como se puede visualizar en la ilustración 27.

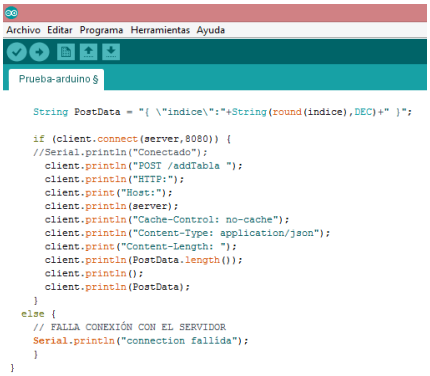


Ilustración 27: Posteo desde arduino

3.7.5 Producto entregable del Sprint 3

A continuación en la ilustración 28 se puede visualizar los resultados al obtener los datos en arduino.

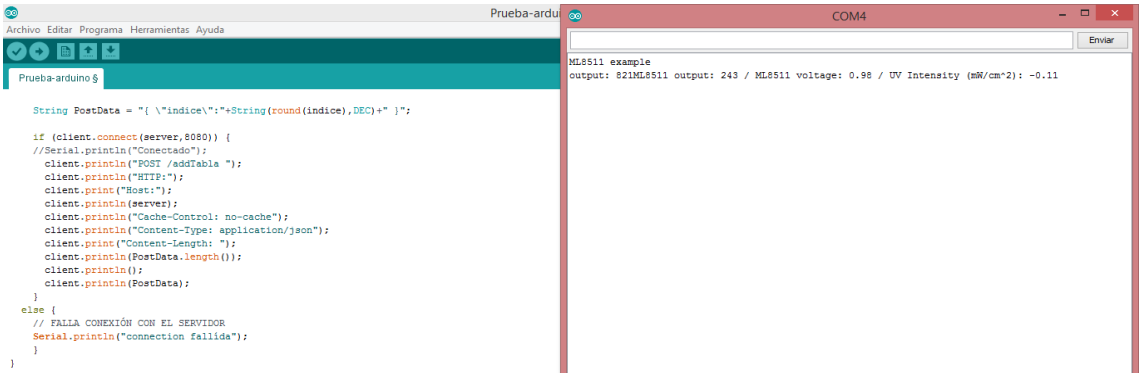


Ilustración 28: Obtencion y posteo de datos

3.8 Despliegue Sprint 4

En el presente sprint se presentará el desarrollo de la página web y aplicación móvil a través del framework de desarrollo Ionic. En el cual se visualizará el último dato ingresado en mongodb.

3.8.1 Desarrollo del Sprint 4

Para desarrollar el sprint 4 se ha determinado un tiempo de duración de 4 semanas, es decir 80 horas.

3.8.2 Sprint Backlog 4

En la ilustración 29 se puede visualizar el Sprint Backlog 4.

Ilustración 29: Sprint Backlog 4

Sprint Backlog 4						
Id	**Nombre**	**Comprobar**	**Sprint**	**Importancia**	**Estimación**	**Comentario**
4	Desarrollo de página web y aplicación móvil con Ionic	Podremos visualizar el último dato que fue posteado desde arduino y almacenado en la base de datos mongodb.	4	2	4	Visualización del último dato almacenado en mongodb a través de Ionic

También en la ilustración 30 se puede observar cada una de las tareas que se realizaran en el Sprint 4.

Ilustración 30: Pila de tareas del sprint 4

Elemento información	**Sprint** 4	**Fecha Inicio** 19-08-2019	**Duración** 4 semanas(80hrs)	Actualización	Días	19-30 Ago	02 – 13 Sept
Delegado/Autor: Raúl Rivera							
H. Usu	**Tarea**			**Estado**	h	20	20
4	Crear una nueva app en Ionic			Completado		X	
4	Visualizar último dato en app móvil			Completado			X

3.8.3 Burn Down Chart 4

En la figura 31 se puede observar la gráfica del avance del proyecto al finalizar el sprint 3.

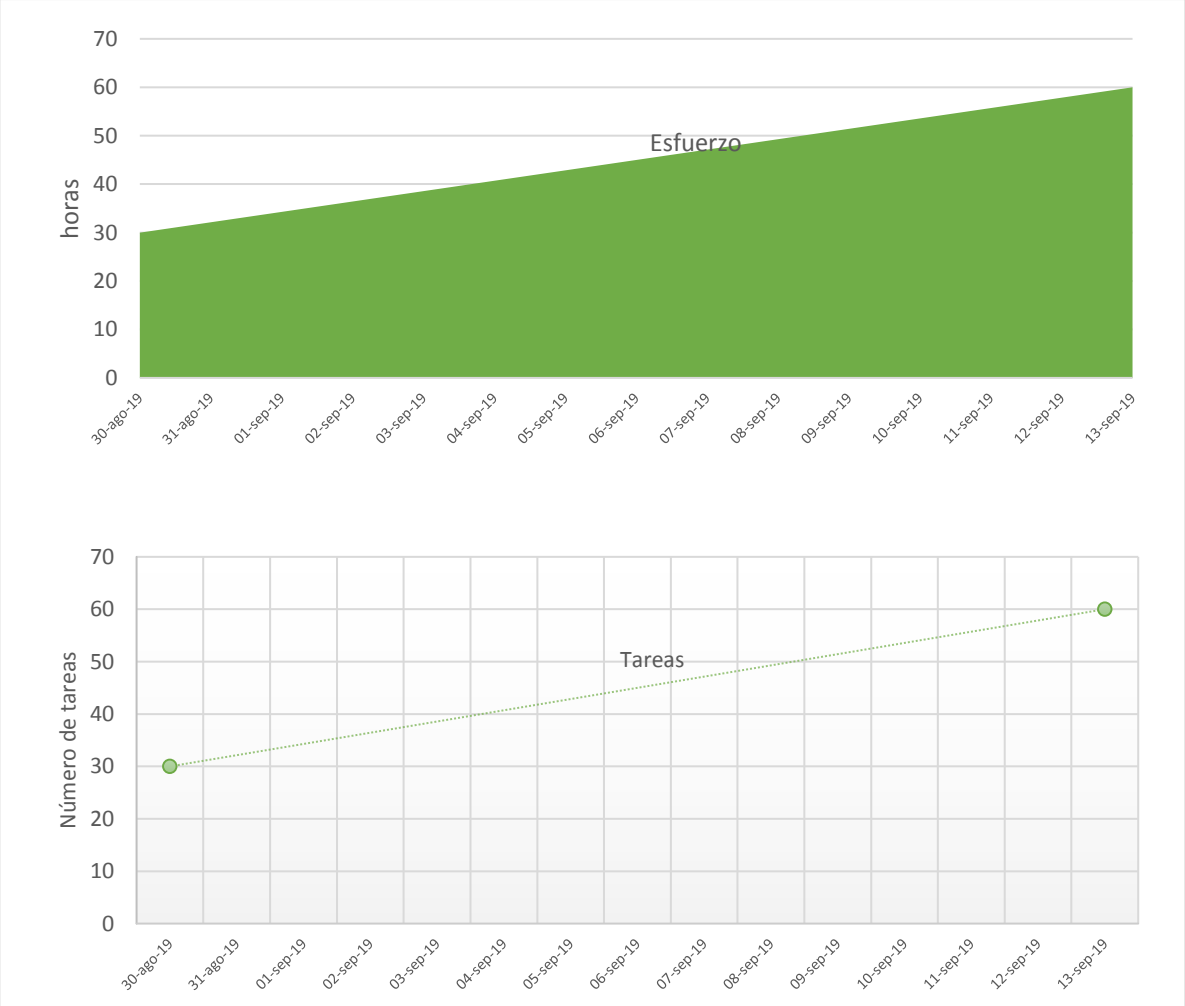
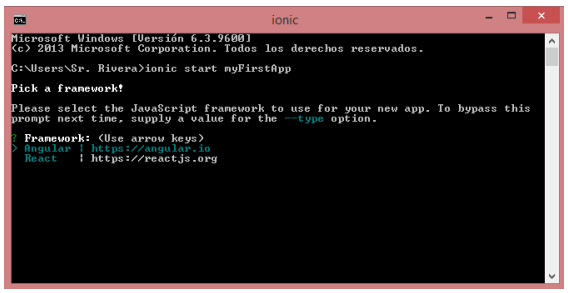


Ilustración 31: Burn down chart Sprint 4

3.8.4 Justificación del Sprint 4

En la ilustración 32 se puede observar la creación de una app con Ionic.

Ilustración 32: Creando app con Ionic



En la ilustración 33 se puede conocer la programación utilizada en la presente app realizada con Ionic.

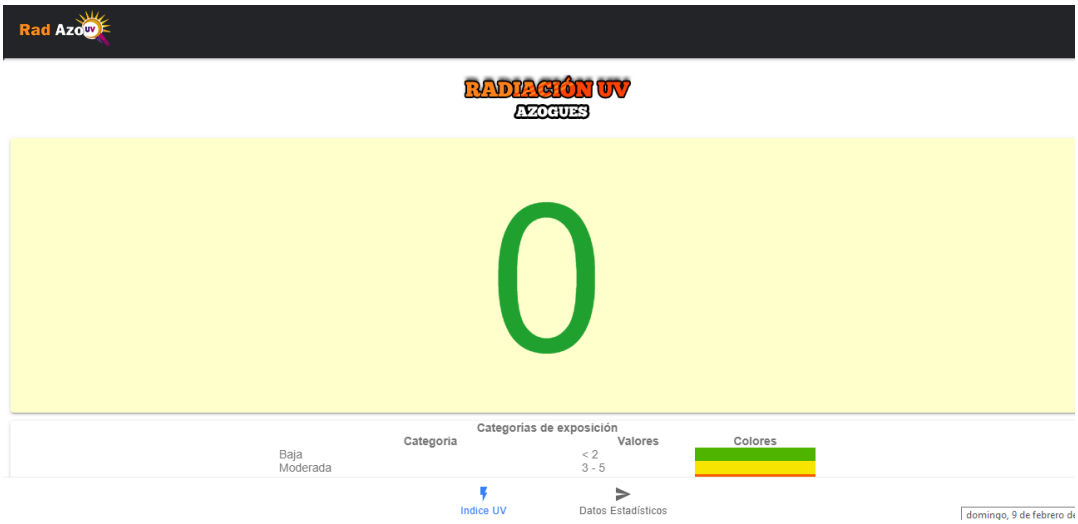
Ilustración 33: Visualización del último dato

```
<ion-header>
  <ion-toolbar color="dark">
    <ion-title>
      <img src="../../assets/rad.png"/>
    </ion-title>
  </ion-toolbar>
</ion-header>

<ion-content img src="../assets/slider_background.jpg" alt="">
 <h1 align="center"><img src="../../assets/logo.png"/></h1>
    <ion-card color="danger" class="welcome-card">

      <h1 align="center" class="estado"
        [ngClass]="{'baja': lastLectura <= 2 && lastLectura >0,
                    'moderada': lastLectura >=3 && lastLectura <=5,
                    'alta': lastLectura >=6 && lastLectura <=7,
                    'MuyAlta': lastLectura >=8 && lastLectura <=10,
                    'Extremadamente': lastLectura >= 11}">{{lastLectura}}</h1>

    </ion-card>
    <ion-card class="welcome-card">
     <div class="hero_side_text">
                    <div class="v_slider_image">
                        <table class="table table-striped table-bordered table-hover table-condensed" style="width: 50%" align="center">
                            <tr class="active">
                                <th class="text-center" colspan="3">Categorías de exposición</th>
                            </tr>
                            <tr class="active">
                                <th class="text-center">Categoria</th>
                                <th class="text-center">Valores</th>
                                <th class="text-center">Colores</th>
                            </tr>
                            <tr class="text-center">
                                <td>Baja</td>
                                <td>< 2</td>
```

3.8.5 Producto entregable del Sprint 4

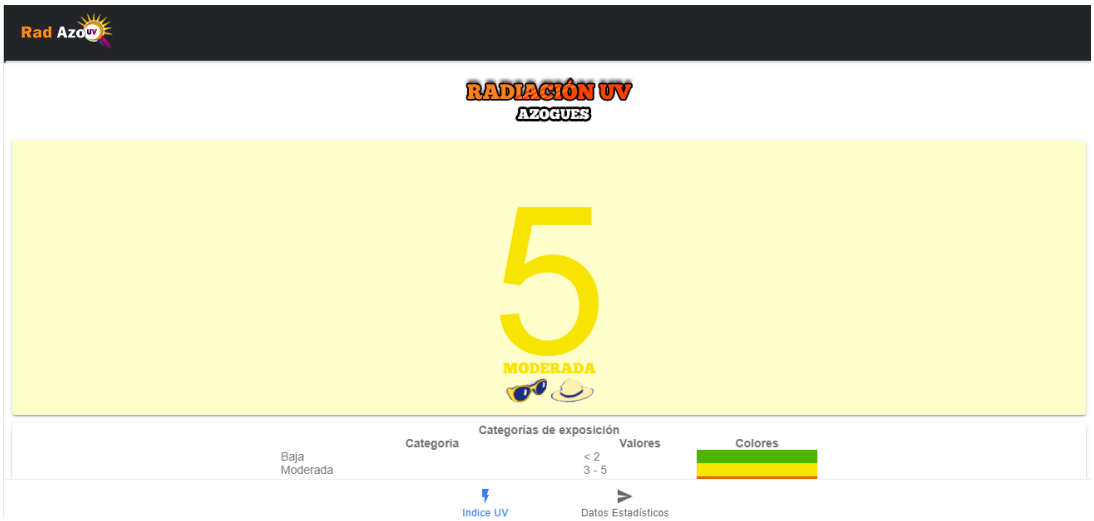
Una vez creada la aplicación, la interfaz se podrá observar de la siguiente manera como se presente en la ilustración 34.

Ilustración 34: App Ionic



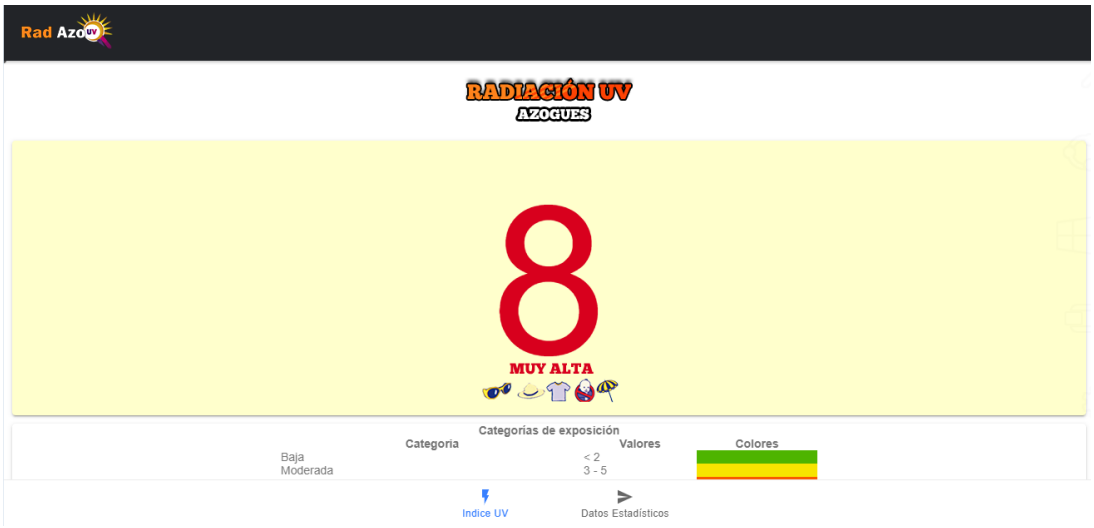
En la ilustración 35 se puede observar el último dato ingresado en la base de datos en este caso es el número 5.

Ilustración 35: Visualizando los datos 1



De igual manera en la ilustración 36 se ha realizado el posteo de otro número para verificar efectivamente que se está visualizando el último número ingresado.

Ilustración 36: Visualizando los datos 2



3.9 Despliegue Sprint 5

En el presente spring se detalla la creación del microservicio que permitirá obtener comunicación en tiempo real entre el microservicio y el usuario, mediante la implementación de la tecnología websockets.

3.9.1 Desarrollo del Sprint 5

Para realizar el quinto y último sprint se ha determinado un tiempo de 4 semanas, equivalente a 80 horas.

3.9.2 Sprint Backlog 5

En la ilustración 37 se presenta el Sprint Backlog 5.

Ilustración 37: Sprint Backlog 5

Sprint Backlog 5						
Id	**Nombre**	**Comprobar**	**Sprint**	**Importancia**	**Estimación**	**Comentario**
5	Microservicio con websockets	Una vez implementado los websockets se podrá visualizar en tiempo real cuando un nuevo dato es almacenado en la base de datos.	5	1	4	Visualización en tiempo real cuando un nuevo valor es ingresado a la base de datos.

Mientras tanto a continuación en la ilustración 5 se puede visualizar cada una de las tareas a realizarse en el Sprint 5.

Ilustración 38: Pila de tareas del sprint 5

Elemento información	**Sprint** 5	**Fecha Inicio** 16-09-2019	**Duración** 4 semanas(80hrs)	Actualización	Días	16-27 Sept	01 – 11 Sept
Delegado/Autor: Raúl Rivera							
H. Usu	**Tarea**			**Estado**	h	20	20
5	Websocket Handler			Completado		X	
5	Websocket Configuration			Completado			X
5	Websocket Demo Aplication						X

3.9.3 Burn Down Chart 5

En la figura 40 se puede visualizar el avance del proyecto al terminar el presente sprint.

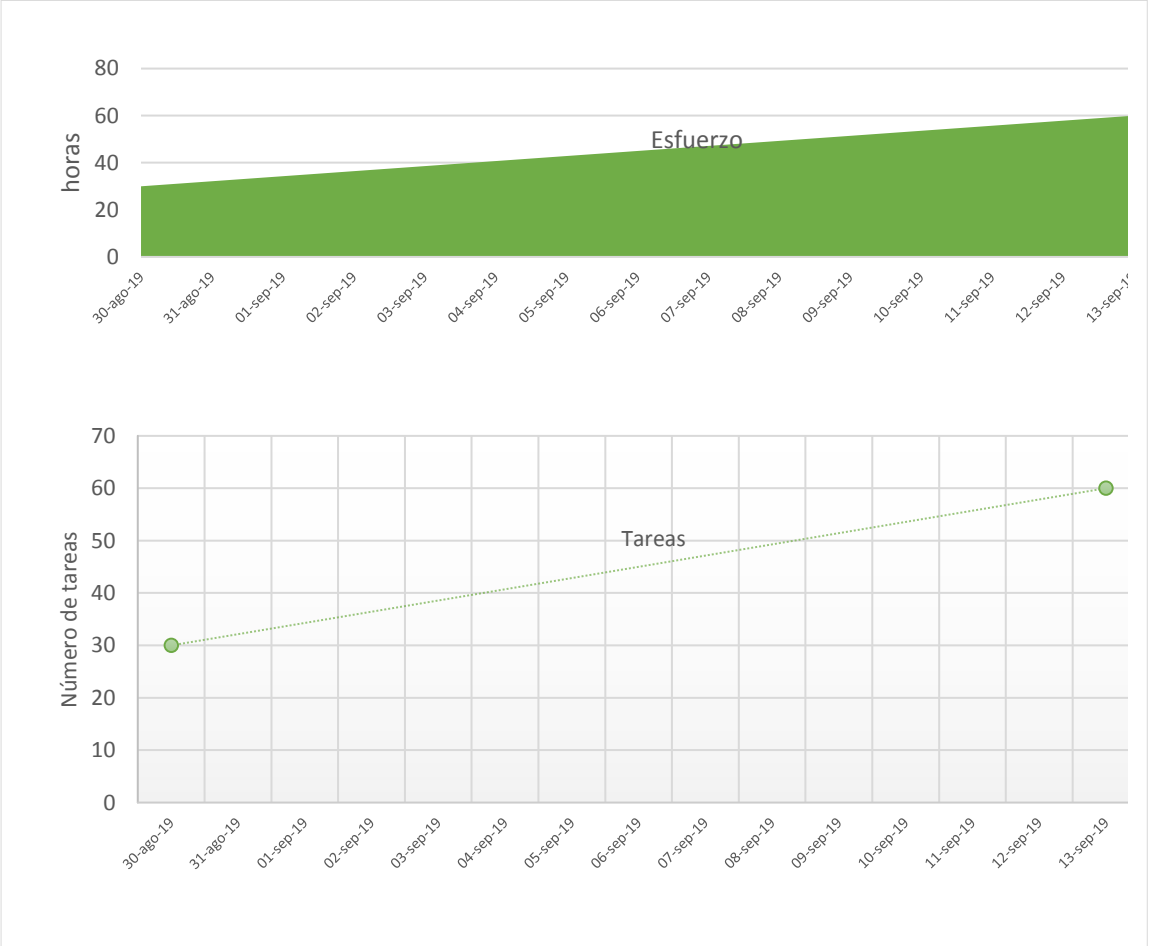
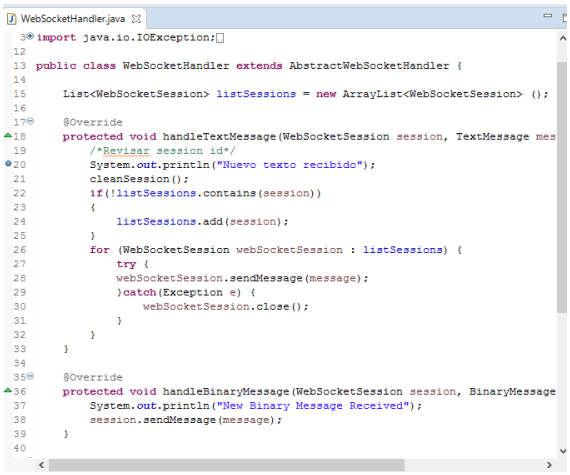


Ilustración 39: Burn down chart Sprint 5

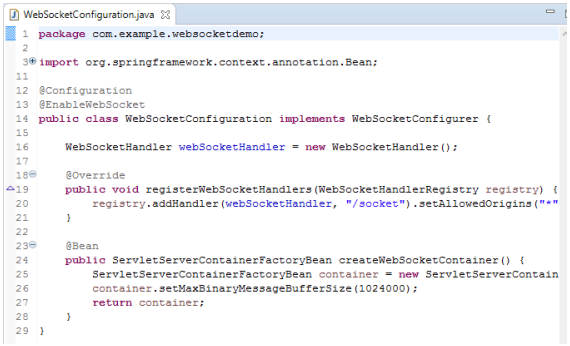
En la ilustración 41 se puede observar el controlador del presente microservicio.

Ilustración 40: Websocket Handler



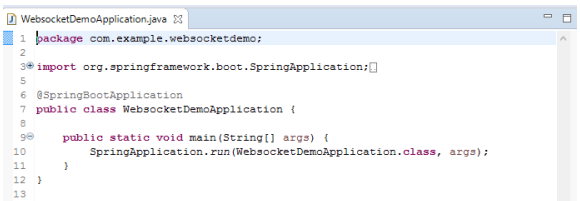
A continuación se puede observar la configuración de los websockets en la ilustración 42.

Ilustración 41: Websocket Configuration



Y por último tenemos el ejecutable como se puede observar a continuación en la ilustración 43.

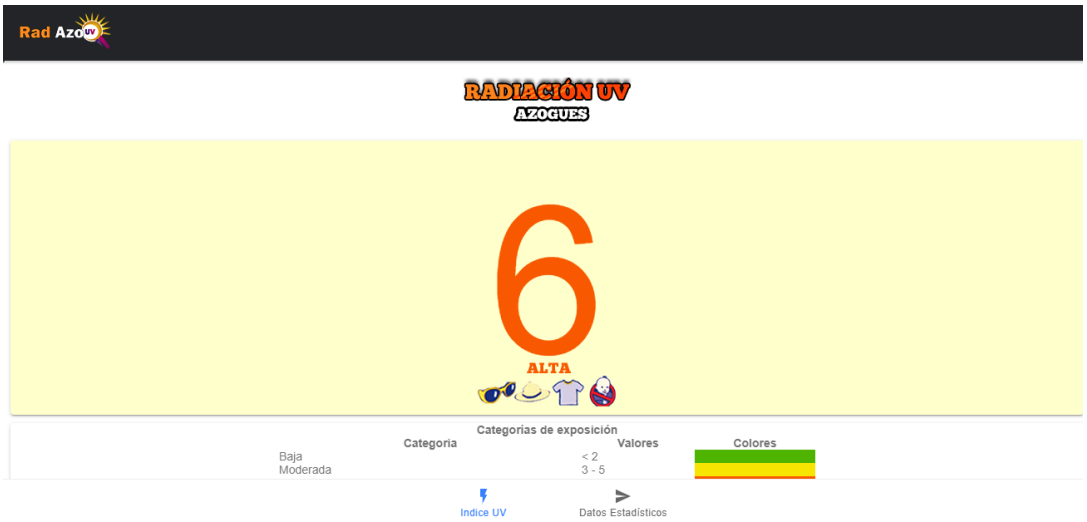
Ilustración 42: Websocket Demo Aplication



3.9.5 Producto entregable del Sprint 5

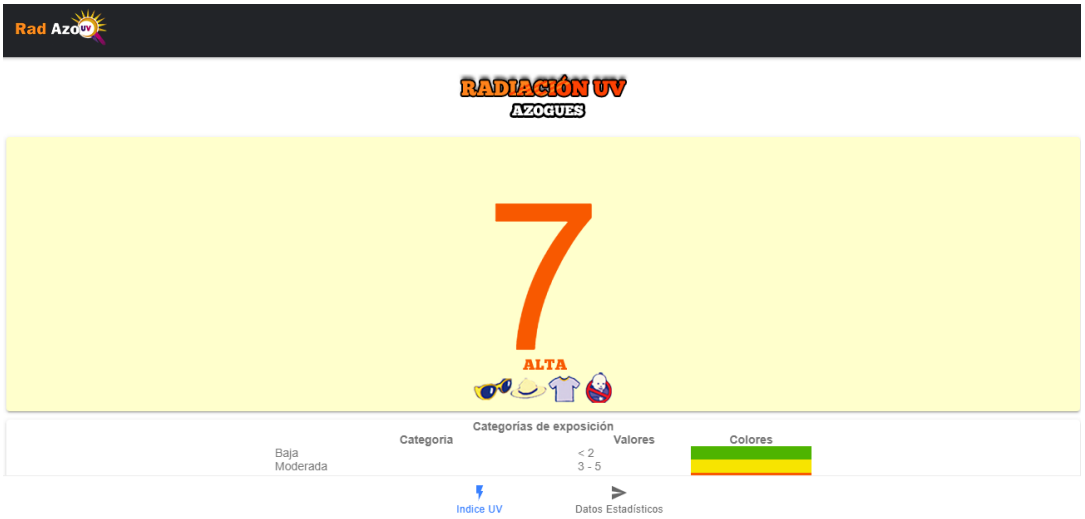
En la ilustración 44 se puede visualizar el último número ingresado en la base de datos.

Ilustración 43: Dato actual



Y una vez implementado el microservicio con websockets se puede visualizar en la ilustración 45, que el número variará automáticamente y en tiempo real mediante la arquitectura de los microservicios con websockets.

Ilustración 44: Cambiando automáticamente con websockets



CAPÍTULO 4

1.3 Conclusiones

- ✓ Se construyó un sistema de microservicios para plataformas IoT debido a la carencia del mismo.
- ✓ A través del marco teórico se pudo realizar un análisis para que la construcción del presente trabajo de titulación cuente con las mejores herramientas de trabajo.
- ✓ Se construyó un prototipo de una aplicación móvil que permita visualizar cual es el estado del índice de radiación UV.

1.4 Recomendaciones

- ✓ Se recomienda realizar la migración de la aplicación a la nube, para de esta manera poder tener acceso externo.
- ✓ Una vez realizado la migración a la nube, se recomienda publicar el apk en Play Store.
- ✓ Se recomienda la implementación de las presentes tecnologías para la obtención de datos importantes como: humedad, temperatura, viento, entre otros.

Bibliografía

Antonio ESPINOZA Mina, M., & del Pilar GALLEGOS Barzola, D. (2017). La industria del software en Ecuador: evolución y situación actual The software industry in Ecuador: evolution and current situation. In *Pág* (Vol. 38).

Arduino. (2012). *Practica 1:Comenzando con Arduino 1.1 Objetivos. Hardware y cable USB* (p. 43). p. 43. Retrieved from www.arduino.cc/en/.

Bass, J. M. (2014). Scrum master activities: Process tailoring in large enterprise projects. *Proceedings - 2014 IEEE 9th International Conference on Global Software Engineering, ICGSE 2014*, 6–15. https://doi.org/10.1109/ICGSE.2014.24

Beedle, M. (2017). *¿ Qué es la Metodología Ágil ?* 1–5. Retrieved from https://luis-goncalves.com/es/que-es-la-metodologia-agil/

Borcin, T. (2017). *Service activity monitoring for SilverWare*.

Culturación. (2019). ¿Qué es y para qué sirve un web service? - Culturación. Retrieved September 19, 2019, from https://culturacion.com/que-es-y-para-que-sirve-un-web-service/

Electronics, M. (2017). ¿Que es Arduino? ~ Arduino.cl - Plataforma Open Source para el desarrollo de prototipos electrónicos. Retrieved September 19, 2019, from http://arduino.cl/que-es-arduino/

Familiar, B., & Barnes, J. A. (2015). Microservices, IoT, and Azure. In *Microservices, IoT, and Azure*. https://doi.org/10.1007/978-1-4842-1275-2

Formativa, A. (2016). ¿Qué son WebSockets? Retrieved January 17, 2020, from https://blog.aulaformativa.com/que-son-websockets/

Fowler, M. (2014). Microservicios. Retrieved January 29, 2020, from https://martinfowler.com/articles/microservices.html

Goikolea, M. (2014, May 19). ¿Qué es un sistema SaaS? Definición y ventajas. Retrieved February 7, 2020, from Blog de IEBSchool website: http://comunidad.iebschool.com/iebs/software-de-gestion/que-es-saas-definicion-ventajas/

Gracia, M. (2018). ¿Qué es IoT (Internet Of Things)? | Deloitte España. Retrieved February 8, 2020, from https://www2.deloitte.com/es/es/pages/technology/articles/IoT-internet-of-things.html

Hinojosa, L., & Daniel, J. (n.d.). *(No Title)*.

Hinojosa, L., & Daniel, J. (2017). *(No Title)*. Ibarra.

López, D., & Maya, E. (n.d.). *Arquitectura de Software basada en Microservicios para Desarrollo de Aplicaciones Web*. Retrieved from https://documentos.redclara.net/bitstream/10786/1277/1/93 Arquitectura de Software basada en Microservicios para Desarrollo de Aplicaciones Web.pdf

Lu, D., Huang, D., Walenstein, A., & Medhi, D. (2017). A Secure Microservice Framework for IoT. *Proceedings - 11th IEEE International Symposium on Service-Oriented System Engineering, SOSE 2017*, 9–18. https://doi.org/10.1109/SOSE.2017.27

MARIN, J. J. M. C. N. M. (2018). *EXPLORACIÓN DE LAS PLATAFORMAS IoT EN EL MERCADO PARA FOMENTAR EL CONOCIMIENTO, BUEN USO Y EFECTIVIDAD DE LOS DISPOSITIVOS IoT CREADOS EN LA FACULTAD DE INGENIERIA Y CIENCIAS BASICAS DE LA INSTITUCIÓN UNIVERSITARIA POLITÉCNICO GRANCOLOMBIANO.*

Masadelante. (2019). ¿Qué significa http? - Definición y explicación del término http. Retrieved January 17, 2020, from https://www.masadelante.com/faqs/que-significa-http

Maya, E., y López, D. (2017). Arquitectura de Software basada en Microservicios para Desarrollo de Aplicaciones Web. Retrieved January 29, 2020, from Septima Conferencia de Directores de

Tecnologia de Informacion 2017 website: http://dspace.redclara.net:8080/bitstream/10786/1277/1/93 Arquitectura de Software basada en Microservicios para Desarrollo de Aplicaciones Web.pdf

Miguel. (n.d.). El Equipo de Desarrollo en SCRUM - Management Plaza. Retrieved February 4, 2020, from https://managementplaza.es/blog/el-equipo-de-desarrollo-en-scrum/

Paulista. (2015). *Cloud computing con herramientas open-source para Internet de las cosas*.

Perry, J. S. (2017). Aspectos básicos de Spring Boot. Retrieved April 13, 2019, from https://www.ibm.com/developerworks/ssa/library/j-spring-boot-basics-perry/index.html

Robertho Artica, Astengo, L. H. P. (2014). *UNIVERSIDAD NACIONAL DE LA AMAZONÍA PERUANA "DESARROLLO DE APLICACIONES MÓVILES" INFORME PRÁCTICO DE SUFICIENCIA PROFESIONAL PARA OPTAR EL TÍTULO PROFESIONAL DE: INGENIERO DE SISTEMAS E INFORMÁTICA*.

Roche, J. (2020). Artefactos Scrum: las 3 herramientas clave de gestión. Retrieved January 28, 2020, from https://www2.deloitte.com/es/es/pages/technology/articles/artefactos-scrum.html

Rubenfa. (2014). MongoDB. Qué es, cómo funciona y cuándo podemos usarlo (o no). Retrieved September 19, 2019, from https://www.genbeta.com/desarrollo/mongodb-que-es-como-funciona-y-cuando-podemos-usarlo-o-no

Saffirio Mario. (2006a). ¿Qué son los Web Services? – Tecnologías de la Información y Procesos de Negocios (BPM). Retrieved April 13, 2019, from https://msaffirio.wordpress.com/2006/02/05/¿que-son-los-web-services/

Saffirio Mario. (2006b). ¿Qué son los Web Services? – Tecnologías de la Información y Procesos de Negocios (BPM). Retrieved September 19, 2019, from https://msaffirio.wordpress.com/2006/02/05/¿que-son-los-web-services/

Scrum. (2019). Qué es SCRUM – Proyectos Ágiles. Retrieved January 21, 2020, from https://proyectosagiles.org/que-es-scrum/

Sun, L., Li, Y., & Memon, R. A. (2017). An open IoT framework based on microservices architecture. *China Communications*, *14*(2), 154–162. https://doi.org/10.1109/CC.2017.7868163

Whitepaper. (2014). *WHITEPAPER: BBDD NO SQL acenswhitepaper*.

ANEXOS

TESIS RAUL RIVERA V3

INFORME DE ORIGINALIDAD

2%	2%	0%	1%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	www.masterindustria40.com Fuente de Internet	1%
2	rincondelbibliotecario.blogspot.mx Fuente de Internet	<1%
3	Submitted to Fundacion Universitaria Juan de Castellanos Trabajo del estudiante	<1%
4	Submitted to UNIV DE LAS AMERICAS Trabajo del estudiante	<1%
5	repositorio.espe.edu.ec Fuente de Internet	<1%
6	upcommons.upc.edu Fuente de Internet	<1%
7	www.pmr.safisiol.org.ar Fuente de Internet	<1%

PERMISO DEL AUTOR DE TESIS PARA SUBIR AL REPOSITORIO INSTITUCIONAL

Yo, RAUL RIGOBERTO RIVERA CALLE, portador/a de la cédula de ciudadanía Nro.,0302450069. En calidad de autor/a y titular de los derechos patrimoniales del trabajo de titulación **"SISTEMA DE MICROSERVICIOSWEB PARA PLATAFORMA IoT "** de conformidad a lo establecido en el artículo 114 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, Así mismo; autorizo a la Universidad para que realice la publicación de éste trabajo de titulación en Repositorio Institucional de conformidad a lo dispuesto en el artículo 144 de la Ley Orgánica de Educación Superior.

Azogues, 05 de marzo de 2020

F: .....

RAUL RIGOBERTO RIVERA CALLE

0302450069

Universidad Católica de Cuenca

EL BIBLIOTECARIO DE LA SEDE AZOGUES

Que: **RIVERA CALLE RAÚL RIGOBERTO,** con cédula de ciudadanía Nro. **0302450069** de la Carrera de **Ingeniería en Sistemas.**

No adeuda libros, a esta fecha: **3 de marzo de 2020**

Eco. Fabián Rodríguez Herrera

BIBLIOTECARIO

Biblioteca Universitaria
MONS. "FROILAN POZO QUEVEDO"

CENTRO DE IDIOMAS

ABSTRACT

WEB MICROSERVICES SYSTEM FOR IoT PLATFORMS

Author: Raúl Rigoberto Rivera Calle

Tutor: Ing. Andrés Sebastián Quevedo Sacoto MSc.

This research work is developed to creating a micro services infrastructure compatible with architectures for Internet of Things (IoT), the proposed micro services infrastructure allows a direct and real-time communication with users who use the services "Web Pages, smartphones and other IoT devices."

In the first chapter, it was done an analysis about the problem, antecedent, objective and the contributions. In the second chapter, it is visualized the theoretical framework throughout a study, it allowed to choose the best tools to get a quality software. In the third chapter, it was carried out a development of the current system through the Scrum methodology and finally, in the fourth and last chapter, it was presented the conclusions and recommendations of this research work.

KEYWORDS: MICROSERVICES, IOT, SCRUM.

Azogues, 20 de febrero del 2020

EL CENTRO DE IDIOMAS DE LA UNIVERSIDAD CATÓLICA DE CUENCA, CERTIFICA QUE EL DOCUMENTO QUE ANTECEDE FUE TRADUCIDO POR PERSONAL DEL CENTRO PARA LO CUAL DOY FE Y SUSCRIBO

Abg. Liliana Urgilés Amoroso, Esp.
COORDINADORA CENTRO DE IDIOMAS AZOGUES