FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

IDENTIFICACIÓN Y SEGUIMIENTO DE LA ACTIVIDAD FÍSICA EN PERSONAS DE LA TERCERA EDAD, MEDIANTE EL USO DE TELÉFONOS MÓVILES

AUTOR

Roberto Alejandro Armas Carrera

AÑO

2018

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

IDENTIFICACIÓN Y SEGUIMIENTO DE LA ACTIVIDAD FÍSICA EN PERSONAS DE LA TERCERA EDAD, MEDIANTE EL USO DE TELÉFONOS MÓVILES

Trabajo de Titulación presentado en conformidad con los requisitos establecidos para optar por el título de Ingeniero en Sistemas de Computación e Informática.

Profesor guía

Mgt. Diego Patricio Buenaño Fernández

Autor

Roberto Alejandro Armas Carrera

Año

2018

DECLARACIÓN DEL PROFESOR GUÍA

“Declaro haber dirigido este trabajo, IDENTIFICACIÓN Y SEGUIMIENTO DE LA ACTIVIDAD FÍSICA EN PERSONAS DE LA TERCERA EDAD, MEDIANTE EL USO DE TELÉFONOS MÓVILES, a través de reuniones periódicas con el estudiante ROBERTO ALEJANDRO ARMAS CARRERA, en el semestre 2018-2, orientando sus conocimientos y competencias para un eficiente desarrollo del tema escogido y dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación.”

_____

Diego Patricio Buenaño Fernández

Magíster en Gerencia Empresarial Mención en Gerencia de Proyectos

C.I: 1708709652

DECLARACIÓN DEL PROFESOR CORRECTOR

“Declaro haber revisado este trabajo, IDENTIFICACIÓN Y SEGUIMIENTO DE LA ACTIVIDAD FÍSICA EN PERSONAS DE LA TERCERA EDAD, MEDIANTE EL USO DE TELÉFONOS MÓVILES, de ROBERTO ALEJANDRO ARMAS CARRERA, en el semestre 2018-2, dando cumplimiento a todas las disposiciones vigentes que regulan los Trabajos de Titulación.”

_____

Bernarda Cecibel Sandoval Romo

Máster en Ciencias de la Computación

C.I: 1709974453

DECLARACIÓN DE AUTORÍA DEL ESTUDIANTE

“Declaro que este trabajo es original, de mi autoría, que se han citado las fuentes correspondientes y que en su ejecución se respetaron las disposiciones legales que protegen los derechos de autor vigentes.”

_____

Roberto Alejandro Armas Carrera

C.I: 1725656563

AGRADECIMIENTO

Agradezco a mi abuelita Fanny y mi mamá por el apoyo incondicional durante toda mi carrera, a mi novia Anne Jácome por su aporte en el estudio de factibilidad de mi tesis, a Nicolás Rizzo por su tiempo en la captura de datos mediante la aplicación móvil y a mi profesor guía Diego Buenaño por la dirección para realizar este documento.

DEDICATORIA

Dedico el esfuerzo de este trabajo de titulación a mi abuelita ya que su sueño está realizado al verme graduar. Esta tesis también va dedicada a las personas que quieran cambiar el mundo desarrollando soluciones para el cuidado de las personas mayores.

RESUMEN

El riesgo de mortalidad a nivel mundial es la inmovilidad física, siendo esta su principal causa según la OMS u Organización Mundial de la Salud. Las personas mayores sufren de complicaciones severas y accidentes por la debilidad que tienen en articulaciones y músculos. Es por este hecho que el presente proyecto busca desarrollar una interacción directa entre fisioterapeutas y pacientes usando un sistema basado en web para el monitoreo remoto de la actividad física de personas mayores. El sistema vuelve fácil el acceso a la información de la actividad del paciente y los reportes gráficos a los fisioterapeutas. La actividad física es recolectada por medio de la información que generan los acelerómetros y sensores Global Positioning System (GPS) de los teléfonos inteligentes, los cuales se han tornado indispensables en nuestra sociedad. La aplicación móvil fue diseñada y desarrollada para recolectar y procesar información que será almacenada y reportada por un sistema web. Se aplican técnicas de inteligencia artificial para la predicción de la actividad física, para que el sistema sea escalable se aplica una arquitectura n capas. El proyecto es una solución asequible por cualquier persona que brinda servicio a grupos de población en riesgo como personas mayores en países en desarrollo como Ecuador, en busca de mejorar la salud y calidad de vida. A través de los gobiernos locales, se podrá difundir con mayor facilidad la aplicación, pero no serán los únicos clientes. La aplicación provee un servicio de salud que cubra efectivamente a una gran porción de la población promoviendo la democratización de la salud a bajo costo.

ABSTRACT

The risk of mortality worldwide is physical immobility, this being the main cause according to the WHO (World Health Organization). Older people suffer from severe complications and accidents due to the weakness they have in joints and muscles. It is for this reason that project presented seeks to develop a direct interaction between physicians and patients using a web-based system for the remote monitoring of the physical activity of elderly people. The system makes easy to access patient's physical activity data and the graphic reports to physicians. The physical activity is collected through information that is generated by accelerometers and Global Positioning System (GPS) sensors from smartphones, which are becoming prevalent in our society. A mobile application was designed and developed in order to collect and process the data which is stored and reported by a web-system. It uses artificial intelligence techniques to obtain scalability, it applies n layer architecture. The project is an affordable solution by anyone to service groups of population at risk such as the older people in developing countries like Ecuador, in order to improve their health and quality of life. Thought local governments, the information will be broadcasted easily the application, but this will not be the only clients. The application provides a health service that will cover effectively as many people as possible promoting health democratization at low cost.

ÍNDICE

ÍNDICE DE FIGURAS

1. Introducción

Desde la antigüedad, los seres humanos eran nómadas. Por ello, la caza y la recolección de frutos eran los principales motivos para realizar actividad física para sobrevivir. Sin embargo, a partir del desarrollo de la tecnología agrícola e industrial hizo que las personas se vuelvan sedentarias, dejando de lado la necesidad de seguir movilizándose en búsqueda de comida. Hoy en día, el desarrollo de tecnología y de las grandes ciudades ha provocado que el 60% de las personas ya no realice actividad física suficiente que le permita gozar de los beneficios que brinda el ejercicio para cuidar de su salud. (Ministerio de Educación, 2017).

Según la Organización Mundial de la Salud (OMS), una de las principales causas de riesgo de mortalidad a nivel global lo constituye la inactividad física. Aproximadamente el 81% de los adolescentes de edad escolar (11 a 17 años) no realiza suficiente actividad física, según datos de 2010. Si las proyecciones se mantienen, la tasa de mortalidad puede ser aún mayor. En 2014, la prevalencia de hipertensión en la Región de las Américas fue de 18,7%. Además, el 15% de la población de más de 18 años tenía diabetes (OMS, 2017).

Una de las principales causas de accidentes por complicaciones severas en personas mayores es la debilidad que tienen en los músculos y articulaciones. Esto provoca caídas que pueden ser mortales para adultos mayores.

El ejercicio ayuda a que las articulaciones de las personas mayores tengan capacidad de movimiento, se refuercen músculos y ligamentos, aumente la capacidad respiratoria y cardíaca. Además, tardará más tiempo en aparecer la fatiga. También ayuda a prevenir o mejorar la evolución de enfermedades como la diabetes, obesidad, osteoporosis, hipertensión arterial, cardiopatía isquémica, entre otros (OMS, 2016).

La agenda para 2030 de la Organización de Naciones Unidas (ONU) apunta al desarrollo sostenible. Este incluye a que la salud sostenible sea uno de sus objetivos estratégicos para poder dar soporte a sus objetivos macro. Estos objetivos buscan promover la innovación y el uso de aplicaciones asequibles

para la salud digital (*eHealth*), telemedicina (*mHealth*) y el aprendizaje en línea (*e-learning*).

Los sistemas de vida asistida ("*Assisted living systems*") pueden ayudar a apoyar a las personas adultas a llevar un registro de sus actividades diarias, de manera que puedan ser monitoreados por sus familiares o médico de cabecera, ayudando de esta forma a mantener una vida sana y segura mientras viven de manera independiente. Estos sistemas deben resolver cuestiones prácticas como la aceptación del usuario y la facilidad de uso para ayudar verdaderamente a las personas mayores (Chernbumroongab, Cangc, Atkinsa, & Yud, 2013).

1.1. Alcance

El alcance de este trabajo de titulación es elaborar un sistema web de monitoreo de actividad física para personas de la tercera edad a través del uso de dispositivos móviles para poder dar seguimiento al paciente y así verificar que ha cumplido con los objetivos que el médico le ha recomendado hacer hasta su siguiente visita. Este sistema cumplirá con los estándares de desarrollo, buenas prácticas y patrones de diseño que permitirán alcanzar un grado de madurez necesario para poder entrar al mercado de la salud, generando valor a los controles de los pacientes.

Para poder cumplir con lo anteriormente mencionado se desarrollará una aplicación móvil que sea capaz de detectar la actividad física que está realizando una persona aplicando técnicas de inteligencia artificial. Para este proceso se efectuará la captura y procesamiento de datos de los sensores como el acelerómetro, giroscopio y GPS. Luego, se almacenarán los datos en un repositorio. Posteriormente el médico podrá acceder a la interfaz web para conocer las estadísticas de actividad física que ha generado el paciente durante el transcurso de un tiempo establecido. Las actividades que se tomarán en cuenta son correr y caminar.

Para cumplir con la propuesta planteada se aplicarán los conocimientos adquiridos en las materias de programación orientada a objetos, desarrollo de

software, ingeniería de software, integración de sistemas, base de datos e ingeniería web, seminario de sistemas.

1.2. Justificación

La inactividad física es uno de los principales causantes para el padecimiento de enfermedades no transmisibles (ENT), por lo cual el proyecto aporta a brindar una herramienta de seguimiento que permita mitigar una de las causas de muerte más altas en los últimos años. Dado que 4 de 5 muertes son relacionadas a las ENT (OMS, 2013).

Con este proyecto también estaríamos apuntando a uno de los objetivos de salud sostenible de la Agenda 2030 de la ONU. Uno de ellos menciona “Desarrollar capacidades para la generación, la transferencia y el uso del conocimiento y la tecnología en materia de salud, promoviendo la investigación e innovación”.

Mediante este proyecto se propone dar una solución al seguimiento de la actividad física de las personas adultas para que puedan conocer acerca de cuánto ejercicio deben realizar y cuánto han realizado.

En el Ecuador y en países en desarrollo, no existe apego a seguir las instrucciones de los doctores. Esto puede ser también un factor de riesgo que se podría mitigar mediante un sistema de monitoreo de actividad física.

Los dispositivos móviles se han convertido en una herramienta indispensable en la vida de los seres humanos gracias al desarrollo de las nuevas tecnologías microelectrónicas se han hecho mucho más asequibles, fáciles de usar y brindan muchos servicios que ahora son básicos como el GPS o acceso a Internet de alta velocidad 4G (Álvarez, 2013).

1.3. Objetivo general

- Desarrollar una aplicación para la lectura de los datos generados por los acelerómetros y sistemas de posicionamiento en dispositivos móviles para dar seguimiento de actividad física de personas adultos mayores.

1.4. Objetivos específicos

- Desarrollar un esquema general para la lectura remota de variables de aceleración recopilada por acelerómetros en dispositivos móviles.
- Seleccionar un mecanismo de clasificación para la identificación de la actividad física cotidiana que realizan las personas.
- Desplegar una aplicación web que permita el almacenamiento y visualización de los resultados de actividad física.

1.5. Metodología a utilizar

En el presente proyecto se utilizará los tres métodos de investigación: inductivo, exploratorio y experimental. El método experimental se refiere a la construcción de la aplicación móvil y la aplicación web que ayudará a detectar y dar seguimiento a la actividad física que realiza un paciente, durante el periodo de tratamiento. Asimismo, se utilizará el método inductivo para la extracción de características de los sensores, así como para determinar los criterios para el filtrado de los datos obtenidos.

Finalmente, se utilizará el método exploratorio con el fin de conocer las funcionalidades que tiene el Software Development Kit (SDK) de Google Fit, referente a la detección de actividad física, además se analizará la precisión de los datos recolectados.

Para el desarrollo del software de administración se aplicará metodología ágil Scrum. Es uno de los marcos de referencia más populares para manejo de proyectos a través de una metodología ágil. Scrum tiene unas connotaciones únicas por su compromiso con iteraciones breves de trabajo. Estas iteraciones de longitud fija llamadas *sprint* aportan un marco para lanzar software con regularidad (Atlassian, 2018).

2. Marco teórico

2.1. Sistemas *eHealth*

eHealth o salud digital se refiere a aplicar las Tecnologías de Información y Comunicación (TIC) en todos los niveles: administrativo, diagnóstico, prevención, seguimiento y tratamiento (CODDii, 2017). Esto permite que existan nuevas formas de trabajo para mejorar el cuidado de la salud localmente, regionalmente e internacionalmente usando las TIC (Vorrink, 2016).

Según Vorrink, los adultos mayores de una comunidad son la porción más grande de consumidores en cuidados de la salud en países desarrollados. Estos grupos se incrementarán dramáticamente entre los años 2000 y 2050, los sistemas *eHealth* pueden ayudar a combatir los potenciales riesgos de enfermedades de este grupo de población.

De acuerdo a la Figura 1, el presente proyecto esta clasificada en telemedicina en la subcategoría de monitoreo.

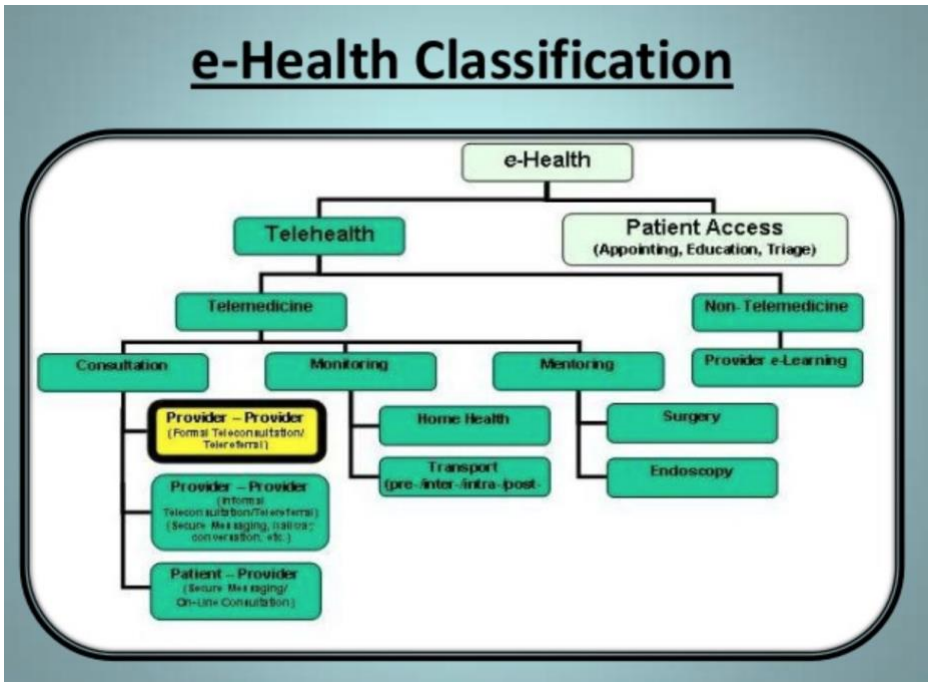

Figura 1. Clasificación eHealth.

Adaptado de Krumar Deepak, s.f.

2.2. Metodologías ágiles de desarrollo

2.2.1. Kanban

Kanban viene de dos palabras en kanji (*kan* = visual) y (*ban* = tarjeta o tablero) es un concepto de producción *Just in Time* (JIT). Kanban comenzó como un sistema de producción altamente efectivo y eficiente en el sistema de producción de Toyota para soportar un control descentralizado por demanda.

La optimización de procesos empíricos y continuos es uno de los pilares en la cual se basa Kanban, permite una respuesta más rápida que Scrum y es útil para soporte técnico. En Kanban no existen las iteraciones por tiempo, sino que esta herramienta se centra en controlar el flujo de trabajo en progreso (Wordpress, 2011)**.**

Los principios de Kanban son los siguientes:

- Eliminar los desperdicios de recursos.
- Determinar el flujo de valor.
- Remover las demoras del flujo de trabajo.
- Controlar el progreso de las tareas, si se demoran aumentar recursos libres para terminar la tarea.
- Evitar que los recursos tengan varias tareas asignadas, un recurso tendrá una sola tarea asignada.
- Busca entregar valor lo más rápido posible, descompone el producto en Minimal Marketable Features.
- Visualizar el flujo de trabajo mediante el Kanban-Board.

2.2.2. Scrum

Scrum es una metodología ágil que puede ser usada para desarrollo de software. Es uno de los marcos más populares para implementar una metodología ágil. Esta metodología tiene como compromiso trabajar iteraciones breves de trabajo (Atlassian, 2018).

Como se muestra en la Figura 2, las iteraciones que se manejan se denominan *sprints*, cada *sprint* tiene una duración de 1 a 2 semanas. Para todo proyecto, se

hace una lista de tareas (*Product Backlog*). El ciclo empieza con la planificación de las tareas del sprint (*Sprint Backlog*). Existirá una reunión diaria (*Daily Scrum Meeting*) para finalizar con un producto funcional entregable (*Product Increment*). Scrum sugiere cuatro protocolos:

- ***Product Backlog* (**Planificación de *sprints*)**:** reunión del equipo en la cual se especifica el alcance que debe ser incluido.
- **Daily Scrum Meeting (**Reunión rápida diaria)**:** reunión máxima de 15 minutos en la cual se hacen cuatro preguntas: ¿Qué he hecho desde la última reunión?, ¿Qué voy a hacer? y ¿qué impedimentos tengo o voy a tener para cumplir con los compromisos de la iteración? (Proyectos Agiles, 2018)
- **Demostración del *sprint*:** reunión participativa en la cual se muestra el avance que se ha obtenido.
- **Retrospectiva del *sprint*:** reunión en la cual el equipo revisa lo que ha ido bien y mal para poder mejorar en la siguiente iteración.

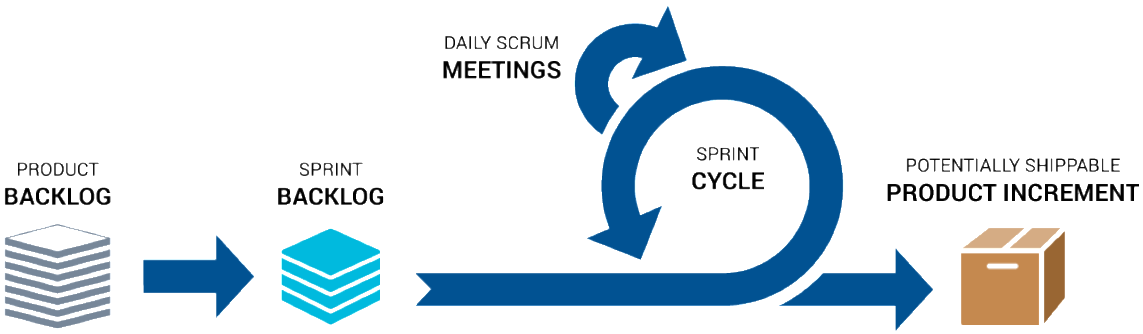

Figura 2. Ciclo sprint

Adaptado de Stigasoft, 2018

2.3. Programación Orientada a Objetos

Programación Orientada a Objetos (POO, por sus siglas en inglés Object Oriented Programming) es un paradigma de la programación que puede aplicarse con los lenguajes de programación. Los lenguajes de programación que se usan tradicionalmente como Java, PHP, Swift, Python, C#, entre otros, adoptan este paradigma.

Como breve historia, antes de la existencia de la programación orientada a objetos, la programación procedural permitía que se ejecuten los programas de manera lineal y existan saltos a bloques de código (llamados funciones) que procesaban los datos en base con parámetros y retornaban al punto de llamada. El inconveniente estaba en que, cuando el programa va creciendo y volviéndose más complejo, es difícil recordar las funciones que fueron declaradas para un proceso en específico. Así, los programadores debían memorizar en qué lugar del código tenían declaradas las funciones que fueron utilizadas para dicho proceso y tener que volver a revisar en un largo tiempo se volvía una tarea difícil.

Para que un lenguaje sea considerado POO debe dar soporte a los siguientes conceptos: clase, objeto, abstracción, encapsulación, herencia, polimorfismo. (Moreno, 2000).

2.3.1. Clases y objetos

Las clases son modelos de representaciones de entidades del mundo real, mientras que los objetos son las instancias de las clases que están declaradas en un programa. Los objetos pueden relacionarse con otros objetos e incluso puede ser compuestos de más objetos.

Los objetos de una misma clase mantienen el mismo comportamiento (métodos) pero difieren en las características (atributos). (The Orange Grove, 2008).

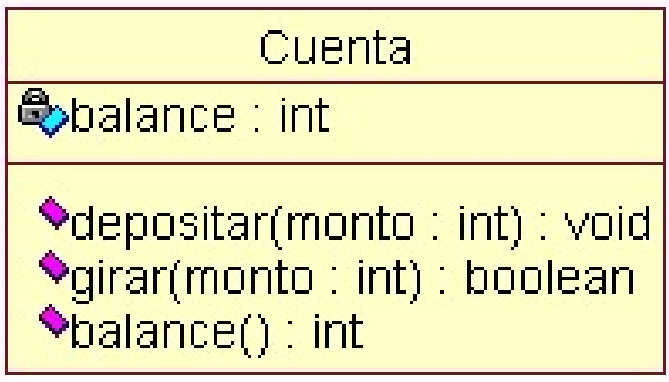

Figura 3. Representación UML de una clase

Adaptado de Universidad de Chile, 2018.

Como se observa en la Figura 3, una clase cuenta con dos componentes:

- **Atributo:** Define las características (variables) de los objetos.
  Ejemplo: color, balance.
- **Métodos:** Define las acciones (funciones) que poseerá el objeto
  Ejemplo: saltar, jugar, depositar, calcular balance.

2.3.2. Encapsulación y abstracción

La encapsulación permite crear componentes como “cajas negras”, en donde está cubierta la funcionalidad mediante niveles de acceso: público, privado, protegido. Así, solo expondrá el comportamiento y las características que vayan de acuerdo con su creación.

La abstracción es el acto de representar las características de su comportamiento sin incluir detalles o implementación. La abstracción es útil en patrones de diseño (Ddegjust, 2018).

2.3.3. Herencia

Mediante la herencia, los objetos de una clase pueden adquirir las propiedades de otras clases. Debido a la herencia se puede hacer una clasificación jerárquica, como se observa en la Figura 4, en la que se puede clasificar, por ejemplo, un proveedor como persona, y podrá así compartir la característica de tener un nombre con otra persona como un empleado.

En POO, el concepto de herencia provee la idea de la reusabilidad, esto quiere decir que se pueden crear clases y extender de una clase padre y compartirán las mismas características sin volver a escribir código (Ddegjust, 2018).

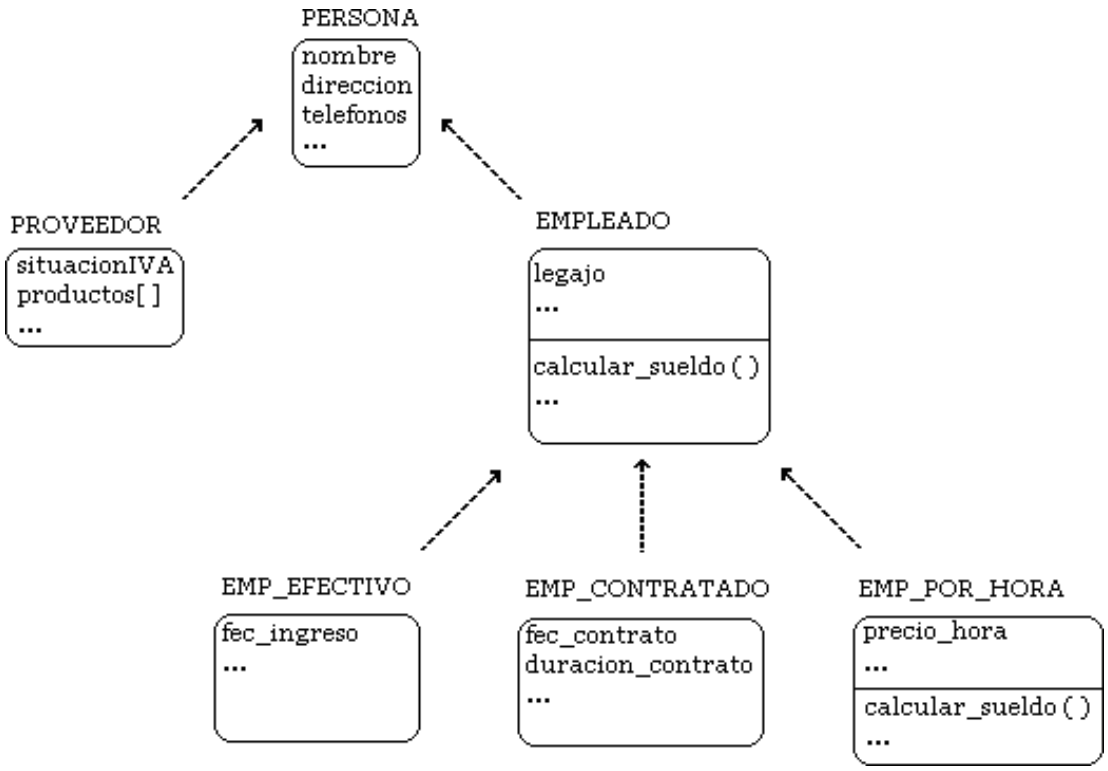

Figura 4. Representación herencia UML

Adaptado de Microsoft, 2018

2.3.4. Polimorfismo

El polimorfismo permite tener diferentes estructuras internas utilizando las mismas interfaces externas. El polimorfismo es muy importante cuando se manejan herencias, de esta manera una clase hija puede ser tratada como su padre (Ddegjust, 2018).

```
Vehicle aVehicle = new Bicycle();
```

Figura 5. Polimorfismo

2.4. Patrones de diseño

Un patrón de diseño en el ámbito del software es la descripción de un problema y la esencia de la solución recopilan sabiduría y experiencias acumuladas a un problema común que se puede presentar en un desarrollo de software. Esta

implementación puede tener un costo mayor al inicio, pero a largo plazo va a representar bajar costos de desarrollo.

Freeman da un ejemplo: “Los patrones de diseño son como pólizas de seguros para el desarrollo de software. Las pólizas de seguro funcionan negociando un pequeño costo ahora para evitar la posibilidad de perder un gran costo más adelante”. (Freeman, 2015).

Para el presente proyecto de titulación se propone la utilización de algunos patrones de diseño brindando así soluciones técnicas y aplicando buenas prácticas de programación, los patrones de diseño que serán usados en la solución de esta tesis son: Patrón Singleton, Builder, MVC.

2.4.1. Singleton

El patrón Singleton (Tabla 1) es parte del grupo de patrones de creación, este patrón se asegura que un solo objeto de un determinado tipo exista en una aplicación (Freeman, 2015).

Tabla 1

Patrón Singleton

Pregunta	**Respuesta**
¿Qué es?	El patrón Singleton se asegura que exista una sola instancia de objeto en una aplicación.
¿Cuáles son los beneficios?	El patrón Singleton puede ser usado para manejar objetos que representen recursos del mundo real o encapsular un recurso compartido.
¿Cuándo se debe usar?	El patrón Singleton es usado para crear futuros objetos que no necesiten incrementar el número de recursos disponibles del mundo real o para

	consolidar una actividad como un registro.
¿Cuándo se debe evitar usarlo?	El patrón Singleton se debe evitar usar cuando no se tengan varios componentes que deseen acceder a los recursos compartidos.
¿Cómo saber si se ha implementado de forma correcta?	El patrón ha sido implementado de forma correcta si existe una y solo una instancia del tipo dado y que esta no pueda ser clonada, copiada o creada una nueva instancia.

El patrón Singleton brinda solución a la problemática de poder tener un objeto que no se duplique en toda la aplicación. Como se observa en Figura 6, se llaman varias veces al componente y hace referencia al mismo objeto.

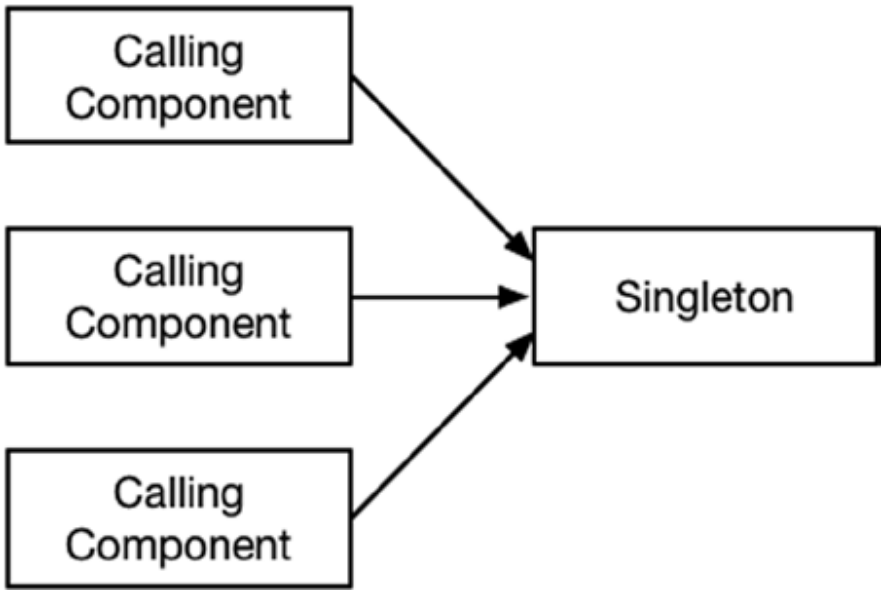

Figura 6. El patrón Singleton

Adaptado de Freeman, 2015

2.4.2. Builder

El patrón Builder (Tabla 2) es usado para separar la configuración de la creación del objeto. El componente puede ser creado con una configuración por defecto, y así solo cambiarlo de acuerdo con las necesidades para enviarlo a crear (Freeman, 2015).

Tabla 2

Patrón Builder

Pregunta	**Respuesta**
¿Qué es?	El patrón Builder pone la lógica y valores por defecto requeridos para crear un objeto dentro de una clase constructora.
¿Cuáles son los beneficios?	El patrón hace fácil la modificación de configuraciones de una clase, cuando se desee modificar algunos parámetros y no tener que estar declarando los valores de uno en uno sino solo modificar el que se desea.
¿Cuándo se debe usar?	Se debe usar este patrón cuando se requiera un proceso de configuración compleja para crear un objeto y no se desee crearlo con todos los valores por defecto.
¿Cuándo se debe evitar usarlo?	Evitar usar este patrón cuando los objetos que se creen tengan siempre diferente configuración.
¿Cómo saber si se ha implementado de forma correcta?	Crear un objeto modificando algunos valores por defecto.

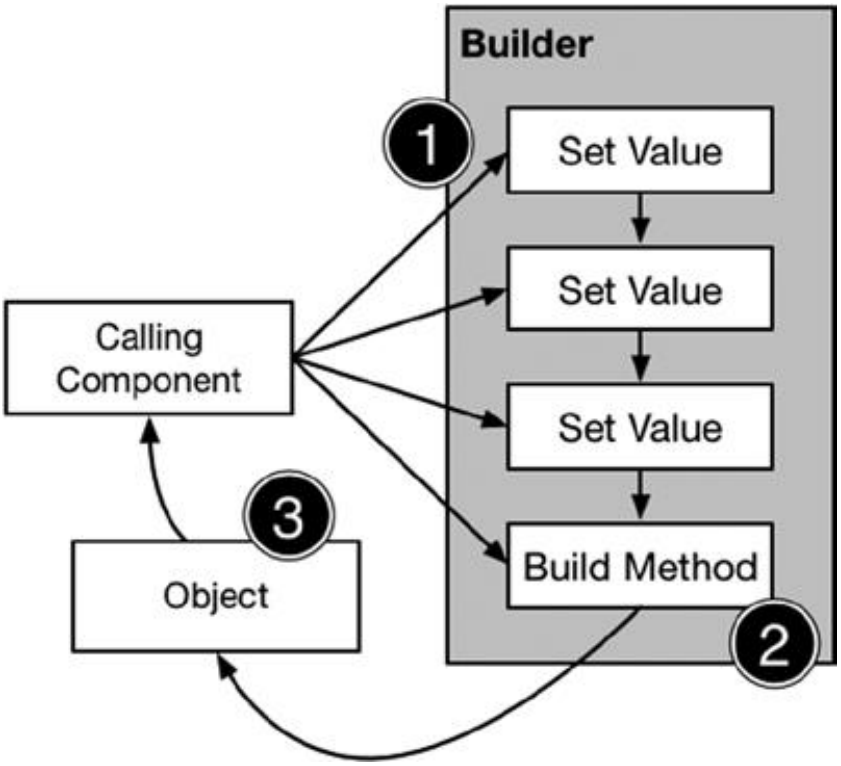

Figura 7. Patrón Builder

Adaptado de Freeman, 2015

Como se muestra en la Figura 7, se pueden llamar a configurar diferentes valores y al final llamar al método de construcción que devolverá el objeto con la configuración de atributos acorde a la modificación que se realizó en la clase constructora.

2.4.3. Modelo Vista Controlador

El patrón MVC (Modelo, Vista, Controlador) Tabla 3 se ha vuelto común en los recientes años, muchos de los desarrollos modernos de software implementan este patrón para reducir la complejidad y aumentar la mantenibilidad de los programas.

Tabla 3

Patrón MVC

Pregunta	**Respuesta**
¿Qué es?	El patrón MVC añade una estructura a toda la aplicación
¿Cuáles son los beneficios?	Las secciones de la aplicación pueden ser desarrolladas, probadas y mantenidas fácilmente.
¿Cuándo se debe usar?	En cualquier proyecto complejo.
¿Cuándo se debe evitar usarlo?	La planeación y la infraestructura requerida no se justifica para un proyecto de corta vida o muy simple.
¿Cómo saber si se ha implementado de forma correcta?	Se debe verificar que la vista esté separada de los datos y de la lógica de negocio, la lógica de negocio debe ser colocada en el controlador, mientras que la estructura de los datos, en el modelo.

El corazón del MVC es la separación de responsabilidades, esto significa que cada sección de la aplicación va a estar apartada de otra para hacer fácil desarrollarlo, probarlo y mantenerlo. El patrón MVC (Figura 8) establece cuatro secciones o capas que se deberían mantener en una aplicación (Freeman, 2015):

- Modelo: El modelo contiene los datos de la aplicación.
- Vista: La vista muestra los datos del modelo al usuario y provee una interfaz donde interactúa el sistema y el usuario.
- Controlador: El controlador responde a las peticiones del usuario y es el responsable de actualizar el modelo y reflejar los cambios en la vista.

- *Cross-cutting:* Todo el resto que no pueda estar en ninguna de las tres capas anteriores, ya que no todo puede ajustarse a estas secciones mencionadas anteriormente.

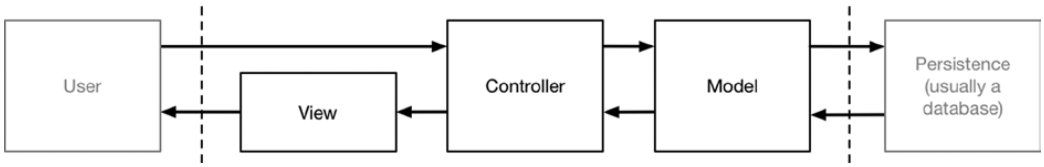

Figura 8. Patrón MVC

Adaptado de Freeman, 2015

2.5. Arquitectura N-Capas

La arquitectura N capas es un estilo arquitectural que se basa en la distribución de las responsabilidades para proporcionar alta cohesión y bajo acoplamiento. La arquitectura N capas permite que el sistema se vuelva distribuido. En la Figura 9, el propósito de esta arquitectura es dividir los servicios, base de datos, lógica de negocio, presentación, entre otros.

Las principales ventajas de utilizar esta arquitectura es la escalabilidad y la mantenibilidad (Moquillaza Henríquez, Vega Huerta, & Guerra Grados Luis, 2010).

Las capas se encargan de la distribución lógica mientras que los niveles tratan la distribución física. Al estructurar una aplicación se deben considerar los siguientes aspectos (De la Torre Llorente, Zorilla Castro, Calvarro Nelson, & Ramos Barroso, 2010):

- Es bueno localizar los cambios de un componente específico dentro de la solución.
- Separar las responsabilidades entre componentes, aumentar la escalabilidad, flexibilidad y mantenibilidad.
- Ciertos componentes pueden ser reutilizados en varios módulos o incluso en diferentes aplicaciones.
- Los componentes deben tener alta cohesión.

- Los componentes deben ser independientes y no existirá mucha dependencia entre cada uno. Esto ayuda a tener bajo acoplamiento.
- Cada capa debe contener sus propias pruebas unitarias con el fin de mantener la calidad y estabilidad de sus componentes.

La Figura 9 muestra un ejemplo de arquitectura N capas en donde se añaden dos capas transversales componentes comunes y entidades de negocio.

Se añaden varias capas como: Presentación, Servicios, Lógica de negocio, Acceso a datos y Base de Datos.

Figura 9. Arquitectura n capas

Adaptado de Molina, 2016

Para un diseño de una arquitectura N capas es importante seguir algunos principios como S.O.L.I.D. (de la Torre Llorente et al., 2010, pp. 27-28)

2.5.1. Principios S.O.L.I.D.

Los principios S.O.L.I.D. son un acrónimo de *Single Responsibility Principle, Open Close Principle, Liskov Substitution Principle, Interface Segregation Principle y Dependency Injection Principle.*

- *Single Responsibility Principle* (Principio de responsabilidad única): Una clase debe poseer una sola responsabilidad. Una clase debe tener una sola razón para la cual fue creada.
- *Open Close Principle* (Principio abierto cerrado): Una clase debe estar abierta para extender, pero cerrada para modificar.
- *Liskov Substitution Principle* (Principio de sustitución de Liskov): Los subtipos deben ser sustituibles por sus tipos base. Para ello las clases pueden ser abstractas en donde pueda implementarse más adelante su comportamiento.
- *Interface Segregation Principle* (Principio de segregación de interfaces): Los implementadores de clases no deben ser obligados a implementar métodos que no se va a utilizar. Las interfaces deben estar segregadas, granuladas. Si algunos métodos no son obligatorios, se puede pensar en crear otra interfaz en donde se agreguen estos métodos e implementar solo donde sea necesario.
- *Dependency Injection Principle* (Principio de inversión de dependencias): Las abstracciones no deben depender de los detalles, los detalles deben depender de las abstracciones, es por ello por lo que es importante implementar ID en proyectos. Las clases no deben tener dependencias directas con las clases sino con las interfaces.

2.6. Object Relation Mapping

Object Relation Mapping (ORM) quiere decir mapear los objetos a tablas en una base de datos relacional. En el mapeo se utiliza la *metadata* para describir las relaciones entre los atributos de los objetos y las columnas de las tablas. Existen varios ORM en el mercado para distintos lenguajes de programación, entre los más conocidos están: Hibernate, Eloquent, Toplink. Estas herramientas ofrecen operaciones básicas en la base de datos como: crear, actualizar, insertar y borrar (CRUD). (World Colleges Information, 2010).

En varias arquitecturas, los ORM son utilizados en la capa de persistencia. Ya que es la capa para los objetos que se manipulan en lo más cercano a la base de datos.

2.7. Protocolo de comunicación websockets

El protocolo *websocket* es un protocolo para aplicaciones web que generan contenido en tiempo real. Comenzó como parte del estándar HyperText Markup Language (HTML) 5 y ahora se mantiene como estándar separado de la Internet Engineering Task Force (IETF) y la World Wide Web Consortium (W3C). Web socket es un remplazo a las técnicas de Comet. (SEDICI, 2011).

Comet es una técnica por la cual se establece una conexión con el servidor mediante una petición Hypertext Transfer Protocol (HTTP) prolongada, al ser una petición prolongada se debe configurar el servidor para que su tiempo de espera (*timeout*) sea muy largo y no finalice la conexión (SIGTE, 2008). La desventaja principal es que por cada usuario el servidor toma recursos de Central Proccesing Unit (CPU) y consume mucha memoria del servidor, si se manejan muchas peticiones puede llegar a tomar algo de tiempo. El protocolo *websockets* es un verdadero protocolo en tiempo real, en cuestión de recursos, no crea un loop por lo que el consumo de CPU y recursos de memoria es mínimo. (Sheiko, 2012)

Web Socket Brinda un marco estandarizado para aplicaciones, además de ofrecer una conexión full-duplex, las metas principales de este protocolo son (SEDICI, 2011):

- Permitir la comunicación entre el cliente y servidor en cada momento.
- Conexión TCP en dos direcciones.
- Reducir el envió de encabezados HTTP al momento de transmitir información.

Para su correcto funcionamiento, tal como se puede observar en la Figura 10, se deben cumplir con los siguientes aspectos:

- Establecer una conexión mediante un *handshake* bien definido.

- El protocolo debe proveer un método para cerrar la conexión cuando el cliente lo requiera.
- El protocolo debe soportar pérdidas de conexión y cierres abruptos de una conexión por parte del usuario.
- Se deben poder enviar datos binarios como en texto plano codificado en utf-8.

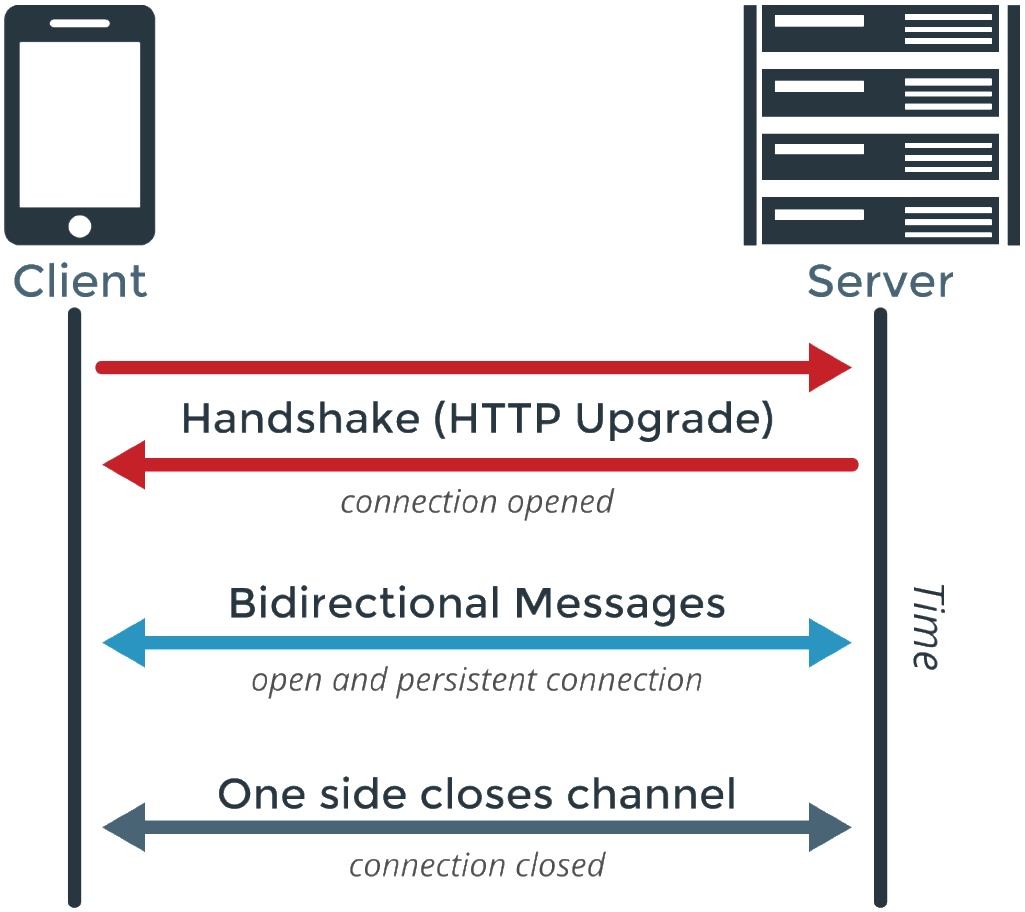

Figura 10. Protocolo websocket

Adaptado de PubNub, 2018

2.8. Framework

Un *framework* o también llamado marco de trabajo en el ámbito de desarrollo de software, se puede referir como una estructura de software compuesta de componentes que se pueden personalizar e intercambiar para el desarrollo de un sistema informático. Provee un conjunto de buenas prácticas y de

herramientas que facilitarán al programador el desarrollo del producto que se va a realizar, gracias a la reutilización de componentes de software.
En el mercado la mayoría de frameworks web ofrecen una capa de controladores, de modelos y de vistas, estos frameworks se basan el patrón MVC. (Universidad de Sevilla, 2018).

2.9. Herramientas de desarrollo

Durante el desarrollo de esta tesis, se requerirán la implementación y uso de varias herramientas tecnológicas que se detallarán a continuación.

2.9.1. Node JS

Node JS es un entorno de ejecución para JavaScript que es construido con el motor JavaScript V8 de Chrome. Node JS utiliza un modelo de operaciones de entradas y salidas sin bloqueo (Node.js, 2018).
Además, está orientado a eventos asíncronos, lo que lo hace eficiente y liviano. Es perfecto para aplicaciones en tiempo real, está diseñado para construir aplicaciones en red escalables.
Basado en Node JS, Express es una infraestructura de aplicaciones web, proporciona un conjunto sólido de características web y móviles. Express ofrece las características de un framework de aplicación web, tales como el ruteo, middlewares, manejo de errores, integración con base de datos. (Express, 2014)

2.9.2. PHP

PHP es el acrónimo de *Hypertext Preprocessor,* es un lenguaje de código abierto. PHP es de los lenguajes más popular especialmente para los desarrolladores web y que puede escribir HTML para sitios web dinámicos.
PHP es un lenguaje de programación que no es altamente tipiado, esto quiere decir que los tipos de datos son inferidos y no necesitan ser declarados al momento de declarar una variable.
Según PHP Group, “Lo mejor de usar PHP es su simplicidad para el principiante, pero a la vez ofrece muchas características avanzadas para los programadores profesionales”. (PHP Group, 2018).

2.9.3. Laravel

Laravel framework es un marco de trabajo de PHP y es uno de los más populares de los últimos años. Gracias a la contribución de Taylor Otwell, PHP ha tomado seriedad en cuanto a aplicaciones empresariales.

Laravel framework posee una documentación consistente, el framework ofrece características útiles como las migraciones, posee un ORM, posee un renderizador de vistas (Blade), métodos de autenticación web y Application Programming Interface (API), ofrece un inyector de dependencias e inversión de control, por lo que se puede aplicar un estilo arquitectural n capas. (Dzone, 2018).

2.9.4. Google Fit SDK

Google Fit SDK para Android es parte de los servicios de Google Play Services, es soportado desde la versión Android 2.3 (API 19), Google Fit se encarga de recolectar información de actividad física y salud de sus usuarios. (Developers, 2015).

Google Fit provee soporte para sensores en teléfonos inteligentes y en dispositivos Bluetooth de baja energía, como se observa en la Figura 11, se pueden tener dos tipos de clientes: Dispositivos móviles y Clientes web.

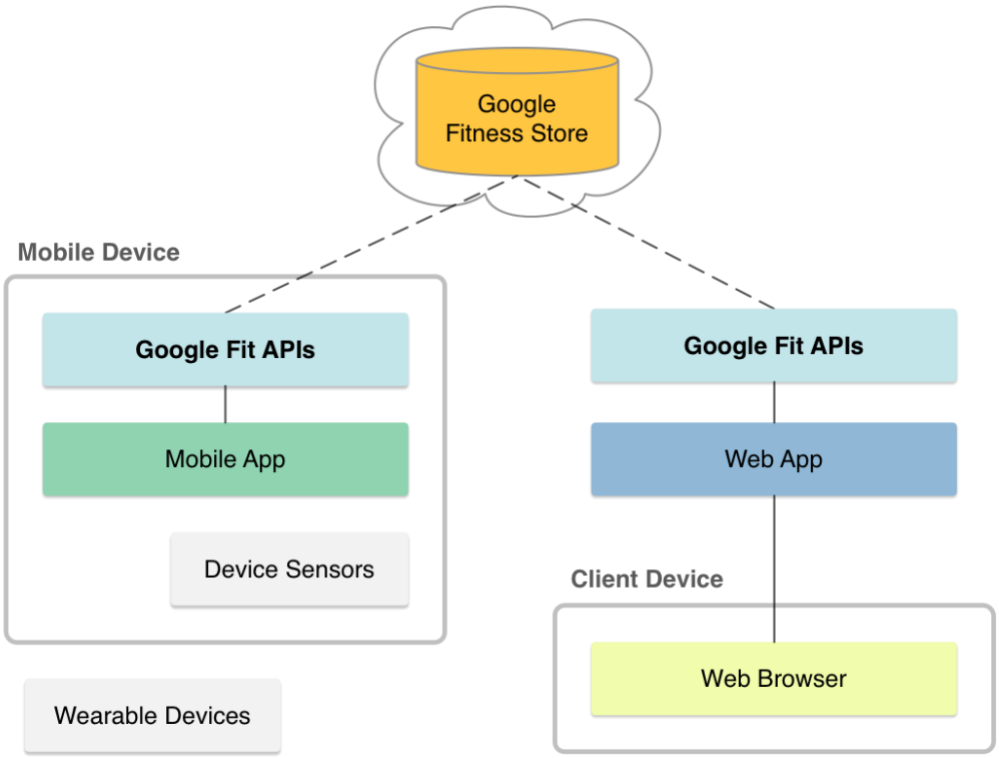

Figura 11. Plataforma Google Fit

Adaptado de Google, 2018

Los teléfonos inteligentes en combinación con Google Fit mejoran la precisión y procesamiento de la información de actividad física para que un médico profesional pueda acceder a la información o por la persona mayor (Armas, Buenaño, Rybarczyk, & Gonzáles, 2018) .

2.9.5. Android Studio

Android Studio es el IDE (Entrono Integrado de Desarrollo) oficial de Google para desarrollar aplicaciones nativas al sistema operativo Android. El IDE ofrece características de depuración, herramientas de gestión de rendimiento, un sistema de compilación que permiten la creación de aplicaciones de alta calidad. (Google, 2017).

Las principales características que ofrece son:

- Instant Run: Es una herramienta que evita volver a compilar la aplicación para poder observar los cambios realizados.

- Emulador: Dispositivo no físico que permite probar en un emulador la aplicación que se está realizando.
- Editor de código inteligente: Ofrece herramientas de refactorización de código, análisis y sugerencias.

2.9.6. Java para Android

Java es un lenguaje de programación para desarrollo de aplicaciones multiplataforma, sin embargo, también se lo utiliza para el desarrollo de aplicaciones nativas para el sistema operativo Android. Java es un lenguaje de propósito general que fue desarrollado por Sun Microsystems que después fue adquirido por Oracle (ATC, 2017).

Como se muestra en la Figura 12, los componentes necesarios para poder desarrollar aplicaciones para teléfonos Android son:

- Java JRE: Java Runtime Enviroment es el conjunto de librerías que permite que se pueda ejecutar un código de Java en cualquier máquina, en el caso de esta tesis será un teléfono celular con sistema operativo Android.
- Java JDK: Java Develpment Kit contiene las herramientas de desarrollo para Java, como el depurador.
- Sistema operativo: Es el sistema operativo de la maquina que va a tener instalado el JRE.
- IDE: Es el ambiente integrado de desarrollo, para este caso Android Studio.
- Código fuente: Código fuente que será desarrollado como producto de esta tesis.

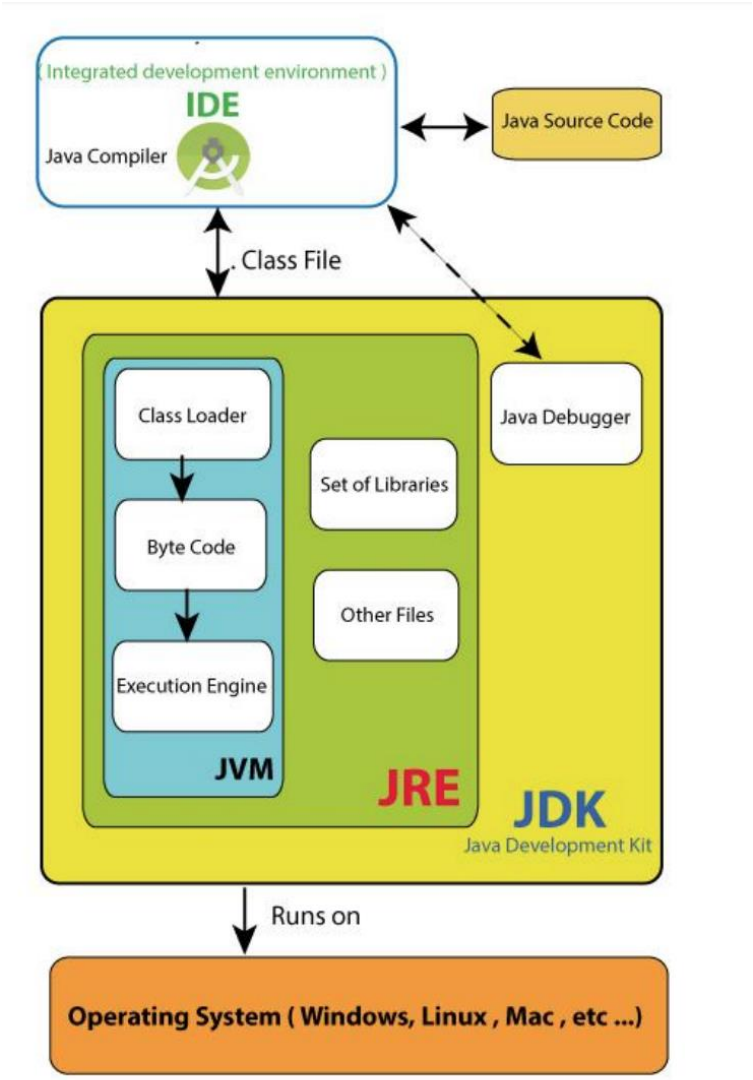

Figura 12. Flujo de trabajo de Java

Adaptado de Android ATC, 2018

3. Análisis y diseño

3.1. Análisis y comparación de herramientas existentes

Existen aplicaciones de podómetro en el mercado, estas aplicaciones se encargan de contar los pasos que la persona ha dado en el transcurso de un periodo de tiempo. En la tienda de Google Play se pueden encontrar aplicaciones como:

3.1.1. Contador de pasos - podómetro, contador de calorías

Es una aplicación móvil que utiliza los sensores integrados para hacer seguimiento de los pasos. No cuenta con una manera de compartir la información con otras personas, por lo que es una aplicación de uso personal y no se usa para dar seguimiento de terceros. Cuenta con informes gráficos (Figura 13) de calorías, pasos, tiempo y distancia (Leap Fitness Group, 2018).

Figura 13. Contador de pasos

Adaptado de Google Play, 2018

3.1.2. Runastic

Muestra estadísticas de actividad física como correr y caminar. Se pueden compartir los logros en redes sociales o dentro de la aplicación, pero no es un sistema de monitoreo de actividad física para integrarlo a sistemas de salud electrónicos

Runastic muestra gráficos estadísticos (Figura 14) en donde se puede observar los días que se ha realizado mayor actividad física.

Figura 14. Runastic estadísticas

Adaptado de Google Play, 2018

Como se observa en Figura 15, Runastic guarda todos los puntos para poder después graficar la ruta que siguió la persona, además de mostrar la distancia, duración y calorías quemadas, datos como velocidad media y ritmo medio. (Runastic, 2018).

Figura 15. Runastic mapa

Adaptado de Google Play, 2018

3.1.3. Podómetro y Entrenador de Peso

Podómetro y Entrenador de Peso (Figura 16) es una aplicación que mide las calorías, la distancia y el tiempo, además de establecer un objetivo y mostrar el reporte de cuánto falta para llegar al objetivo deseado (Pacer Heath, 2018).

Figura 16. Pantalla Podómetro y entrenador de peso

Adaptado de Google Play, 2018

Al igual que Runastic, en Podómetro y Entrenador de Peso (Figura 17) se muestran los pasos recorridos en función de los días de la semana para tener noción qué día se realizó mayor actividad y qué día menor actividad.

Figura 17. Pantalla análisis de datos de podómetro y entrenador de peso

Adaptado de Google Play, 2018

3.1.4. Análisis

Como conclusión, las herramientas investigadas dan un seguimiento de pasos, cálculos de calorías, tiempo invertido en realizar la actividad, pero no es un sistema *eHealth*, en donde se pueda integrar la información recolectada por el paciente para uso y análisis de un médico tratante. Por esta razón, el presente proyecto de tesis busca brindar una integración con sistemas médicos, permitiendo almacenar en un repositorio de datos para brindar un canal de interacción y seguimiento directo entre médico y paciente.

3.2. Requisitos funcionales y no funcionales de la solución

Para la extracción de requisitos funcionales y no funcionales, se aplicó la técnica llamada Brainstorm, que consiste en listar las funcionalidades que tendrá el sistema.

Para el proyecto se requiere:

- El sistema debe captar la actividad física que realiza el paciente.

- El sistema debe almacenar los datos en tiempo real.
- El sistema permitirá al médico gestionar la información del paciente.
- El sistema deberá autenticar al paciente de acuerdo a su registro previo.
- El sistema deberá mostrar en gráficos el tiempo y distancia de la actividad del paciente.

A continuación, se procedió con la clasificación y priorización de los requisitos en base a la solución que cubre los objetivos planteados. Con estos objetivos se procedió al diagrama de los casos de uso, que dan una noción en diseño a lo que el sistema debe manejar. Posteriormente se realiza la especificación de cada caso de uso. Para este proyecto, el diagrama de caso de uso (Figura 18) se lo representa siguiendo la metodología Scrum, los casos de uso (historias de usuario) tienen una duración de 2 semanas, se ha distribuido un caso de uso por iteración.

El alcance del proyecto está orientado al paciente, como requerimiento el sistema no consta el registro de médicos, en este caso lo deberá realizar el administrador de la base de datos, sin embargo, en mejoras futuras se puede incluir esta característica.

3.2.1. Diagrama de casos de uso

Una vez recolectados los requerimientos, la Figura 18 muestra el diagrama de casos de uso que ayudará a tener una noción de los requerimientos del sistema.

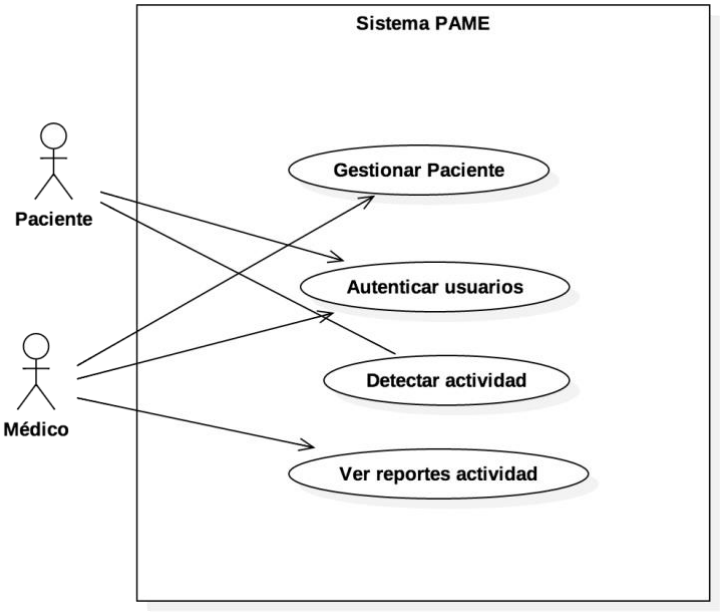

Figura 18. Diagrama de casos de uso

3.2.2. Requisitos funcionales

Los requisitos funcionales se especifican en la Tabla 4, Tabla 5 ,Tabla 6, Tabla 7, Tabla 8 y Tabla 9.

Gestionar Paciente

Tabla 4

RF1.1. Registrar Paciente

RF1.1. Registrar paciente	
Función	Formulario de registro para que el paciente pueda utilizar la aplicación móvil en su celular.
Descripción	El medico ingresa el nombre, el correo electrónico e identificación para que el paciente pueda registrarse en el sistema.

Entradas	Datos personales: nombre, cédula, correo, fecha de nacimiento, peso y objetivo diario.
Fuentes	Teclado.
Salidas	Se despliega un mensaje confirmando que se ha registrado al paciente en el sistema.
Proceso	Se llena un formulario con los datos del paciente, si el correo existe se tendrá que mostrar una alerta de que el correo ya está registrado al igual que la identificación, cuando se registre exitosamente se deberá enviar un correo para que el paciente pueda activar su cuenta.
Restricciones	El correo y la identificación deberán ser únicos.
Precondiciones	Ninguna.
Pos condiciones	Activar cuenta del paciente.

Gestionar Paciente

Tabla 5

RF1.2. Activar cuenta del paciente

RF1.2. Activar cuenta del paciente.	
Función	Verificar el correo del paciente, además del registro de su contraseña y activación de su cuenta.
Descripción	El paciente abre el enlace que recibe en su bandeja de entrada, ingresa una contraseña y se activa su cuenta.

Entradas	Datos personales: contraseña.
Fuentes	Teclado o teclado de pantalla táctil.
Salidas	Se despliega un mensaje confirmando que se ha activado la cuenta.
Proceso	Se llena un formulario con la contraseña para ingresar a la aplicación móvil, se modifican los datos de perfil (peso) si el paciente lo desea y se muestra un mensaje de activación de cuenta.
Restricciones	El peso no deben ser valores negativos. El peso debe ser expresado en Kilogramos.
Precondiciones	Registrar paciente.
Pos condiciones	Autenticar paciente.

Autenticar usuarios

Tabla 6

RF2.1 Autenticar paciente

RF2.1. Autenticar paciente	
Función	Autenticar y autorizar al paciente con sus credenciales.
Descripción	El paciente ingresa su correo y su contraseña para ingresar a la aplicación móvil.
Entradas	Datos personales: correo y contraseña.
Fuentes	Teclado de pantalla táctil.

Salidas	Se muestra la pantalla de inicio.
Proceso	Se llena el formulario con usuario y contraseña.
Restricciones	Las credenciales deben ser válidas.
Precondiciones	Activar cuenta de paciente.
Pos condiciones	Registrar actividad.

Autenticar usuarios

Tabla 7

RF2.2 Autenticar médico

RF2.2. Autenticar médico	
Función	Autenticar y autorizar al médico con sus credenciales.
Descripción	El médico ingresa su correo y su contraseña para ingresar al portal web.
Entradas	Datos personales: correo y contraseña.
Fuentes	Teclado de pantalla táctil.
Salidas	Se muestra la pantalla de inicio.
Proceso	Se llena el formulario con usuario y contraseña.
Restricciones	Las credenciales deben ser válidas.
Precondiciones	Ninguno.
Pos condiciones	Registrar paciente.

Detectar actividad.

Tabla 8

RF3.1 Registrar actividad

RF3.1. Registrar actividad	
Función	Guardar los datos emitidos por los sensores del teléfono para enviarlos al servidor.
Descripción	El paciente presiona el botón de empezar y la aplicación móvil trasmitirá la información al servidor.
Entradas	Botón de empezar
Fuentes	Pantalla táctil.
Salidas	Se muestra el tiempo de actividad.
Proceso	El paciente ingresa a la aplicación y presiona el botón de empezar, los datos recolectados de los sensores se envían al servidor para ser almacenado.
Restricciones	Debe existir conexión a Internet.
Precondiciones	Activar paciente.
Pos condiciones	Ver reportes de actividad.

Ver reportes actividad.

Tabla 9

RF4.1 Ver reporte de actividad

RF4.1. Ver reporte de actividad	
Función	Se muestra un reporte de cantidad de tiempo corriendo y cantidad de tiempo caminando.
Descripción	El medico consulta la actividad del paciente que desee.
Entradas	Paciente que se desea consultar, rango de fecha.
Fuentes	Teclado
Salidas	Se muestra un gráfico de cantidad de tiempo (eje y) y días (eje x).
Proceso	Se escoge el paciente que se desea consultar y el rango de fecha del reporte, el sistema agrupa la cantidad de ejercicio por actividad y por días.
Restricciones	Deben existir datos para poder mostrar un reporte.
Precondiciones	Registrar actividad.
Pos condiciones	Ninguna.

3.2.3. Requisitos no funcionales

Los requisitos no funcionales se especifican en la

Tabla 10, Tabla 11 y Tabla 12.

Tabla 10

RNF01 - Disponibilidad

Identificación del requerimiento:	RNF01
Nombre del requerimiento:	Disponibilidad
Características:	El sistema deberá ser tolerante a fallos.
Descripción del requerimiento:	El sistema debe contar con una alta disponibilidad, debe contar al menos con una disponibilidad del 99%. Esto quiere decir que debe estar disponible las 24 horas del día, en los 7 días de la semana.
Prioridad del requerimiento: Alta	

Para poder mitigar el riesgo de la disponibilidad, se deberá implementar un clúster, para que, de esta manera, si un nodo falla el otro se inicie de forma inmediata.

Tabla 11

RNF02 - Seguridad

Identificación del requerimiento:	RNF02
Nombre del requerimiento:	Seguridad
Características:	Privacidad
Descripción del requerimiento:	La información que maneje la aplicación deberá ser cuidadosamente manejada, debido a que son datos privados de cada paciente, solo podrá ser observada por su médico asignado.
Prioridad del requerimiento: Alta	

Hoy en día, los riesgos de seguridad y privacidad son requisitos no funcionales que se deben tomar en cuenta para evitar cualquier sanción por violación de derechos de privacidad de las personas.

Tabla 12

RNF03 - Mantenibilidad

Identificación del requerimiento:	RNF03
Nombre del requerimiento:	Mantenibilidad

Características:	El software debe aplicar buenas prácticas y una arquitectura establecida.
Descripción del requerimiento:	El sistema deberá seguir estándares y definir una arquitectura. Además de aplicar buenas prácticas de programación.
Prioridad del requerimiento: Alta	

Se sugiere la utilización de un marco de trabajo para la programación de la administración y la utilización de una arquitectura como puede ser la N capas, esto ayudará a cumplir el requerimiento no funcional de mantenibilidad.

3.3. Diseño arquitectónico de la solución

Para cumplir los requerimientos no funcionales se propone el siguiente despliegue (Figura 19). Este diseño cubrirá el requerimiento de disponibilidad y escalabilidad. Se proponen los siguientes nodos:

- Android *application*: La aplicación que usará el paciente y recolectará la información de actividad física.
    - Activity Detection Service: Es el componente que recolecta la información de los sensores e integra con la detección de Google Fit. El componente se encarga de armar la trama que será enviada al servidor. Se realizaron 50 pruebas en las cuales Google Fit mostro una fidelidad del 94% para mejorar la precisión del SDK se procedió a mejorar la clasificación en base al parámetro de la velocidad, si el SDK detecta que la persona esta caminando cuando la velocidad muestra que está corriendo, se tomará como resultado la opción de correr en lugar de caminar, obteniendo 98% de fidelidad.

    - Network Service: Se encarga de implementar eficientemente la conexión por websocket y que requerirá la aplicación para el envió de los datos.
- Node JS *app*: es la aplicación de servidor en tiempo real que provee un protocolo de comunicación *websocket* por la cual la aplicación móvil se conectará.
    - *Websocket*: es el componente que implementa la conexión por *websockets* y se encarga de guardar los datos que son recibidos a la base de datos.
    - API: provee las interfaces de programación de aplicaciones para autenticación de usuarios para la aplicación móvil a través del protocolo http.
- DB: es el nodo de base de datos, este servidor contendrá un motor de base de datos Microsoft SQL Server.
- *Backend*: es la aplicación web o administración en donde el médico podrá acceder para conocer la actividad física de sus pacientes.
    - Web: provee la interfaz web que será accedida por el médico.
    - API: conjunto de interfaces de programación de aplicaciones, las cuales serán expuestas para uso del componente web e integración con otros sistemas.

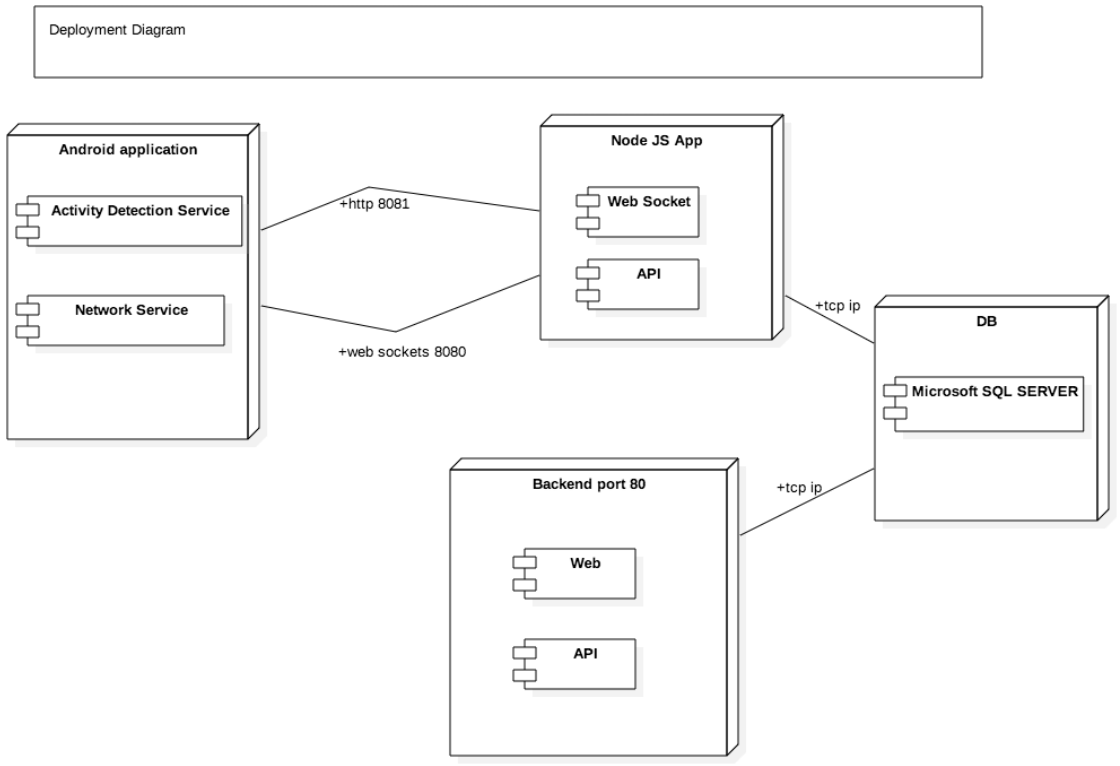

Figura 19. Diagrama de despliegue.

3.4. Diagrama de arquitectura de la aplicación web

El diagrama de arquitectura del nodo *backend* (Figura 20) cumple con el estilo arquitectónico N capas, consta de varios componentes para su arquitectura:

- Angular: Es la capa que se encarga de mostrar los datos mediante Javascript usando el *framework* Angular.
- Blade: Es la capa que se encarga de mostrar los datos que se renderiza en el lado del servidor, en el patrón MVC equivale a la vista.
- Controladores: Son los que procesan las peticiones del cliente, esta capa es la encargada de usar los repositorios para el manejo de la lógica de negocio, en el patrón MVC equivale al controlador.
- API: Es la capa encargada de procesar las peticiones que provienen de las llamadas *http*.

- Repositorios: Son los encargados de almacenar los datos mediante los modelos en la base de datos, además de implementar la lógica de negocio.
- Modelos: Los modelos son las representaciones en clase de las entidades de la base de datos.
- Base de datos: Esta capa es la que almacena los datos.
- *Framework* Laravel 5: Es la capa transversal de la arquitectura N capas propuesta, esta capa transversal contiene elementos comunes. Esta capa provee la comunicación entre las demás capas, provee inyección de dependencias e inversor de control.

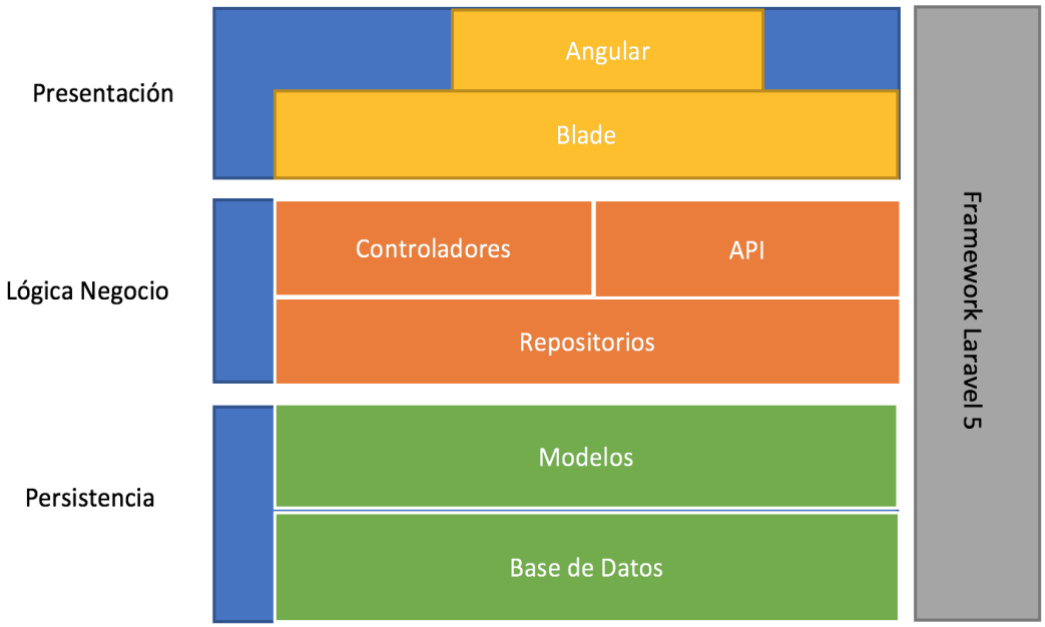

Figura 20. Diagrama de arquitectura.

3.5. Diagrama de clases de la aplicación móvil

Los datos que manejará la aplicación móvil serán estructurados de acuerdo con el diagrama de clases (Figura 21), esta se diseñó aplicando los conceptos de programación orientada a objetos.

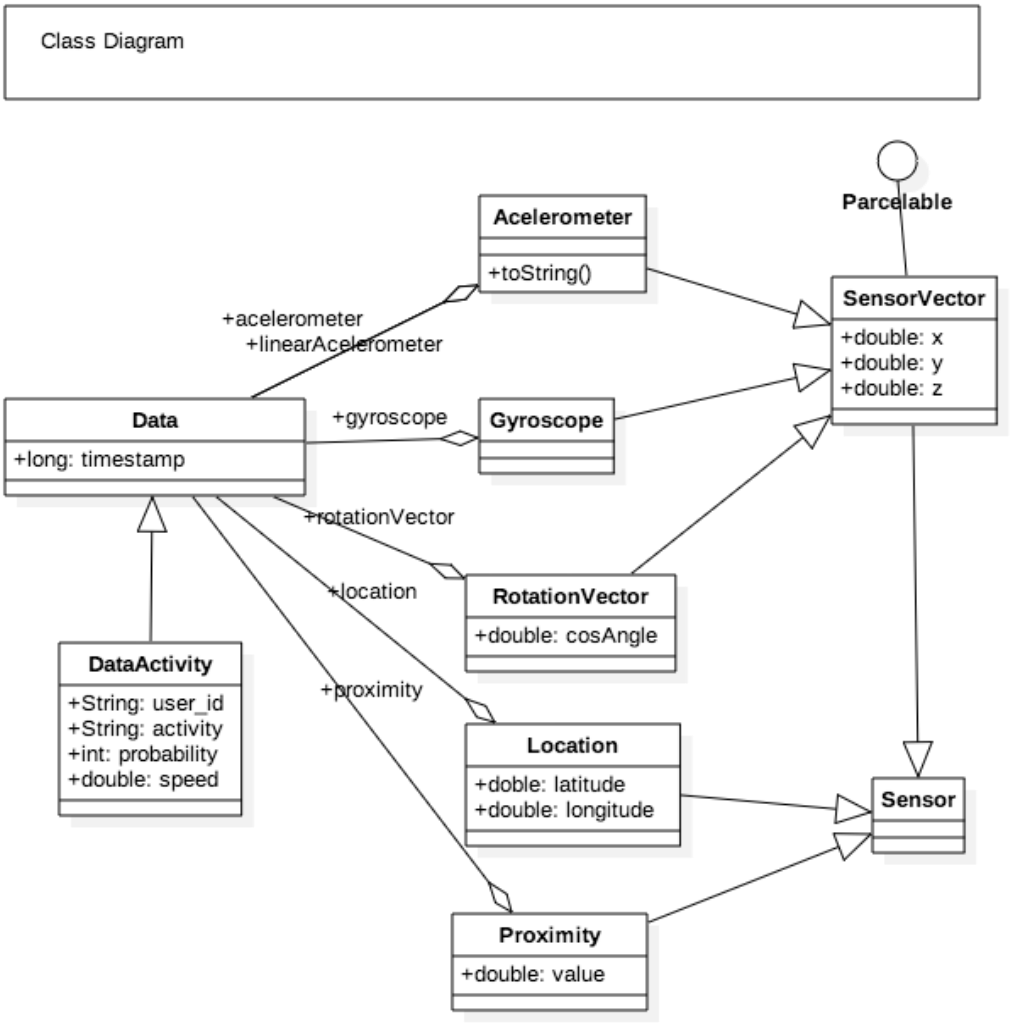

Figura 21. Diagrama de clases

3.6. Diagrama de Datos

Los datos que van a ser persistidos se han diseñado de acuerdo con el análisis de requerimientos. La Figura 22 muestra la relación de los datos.

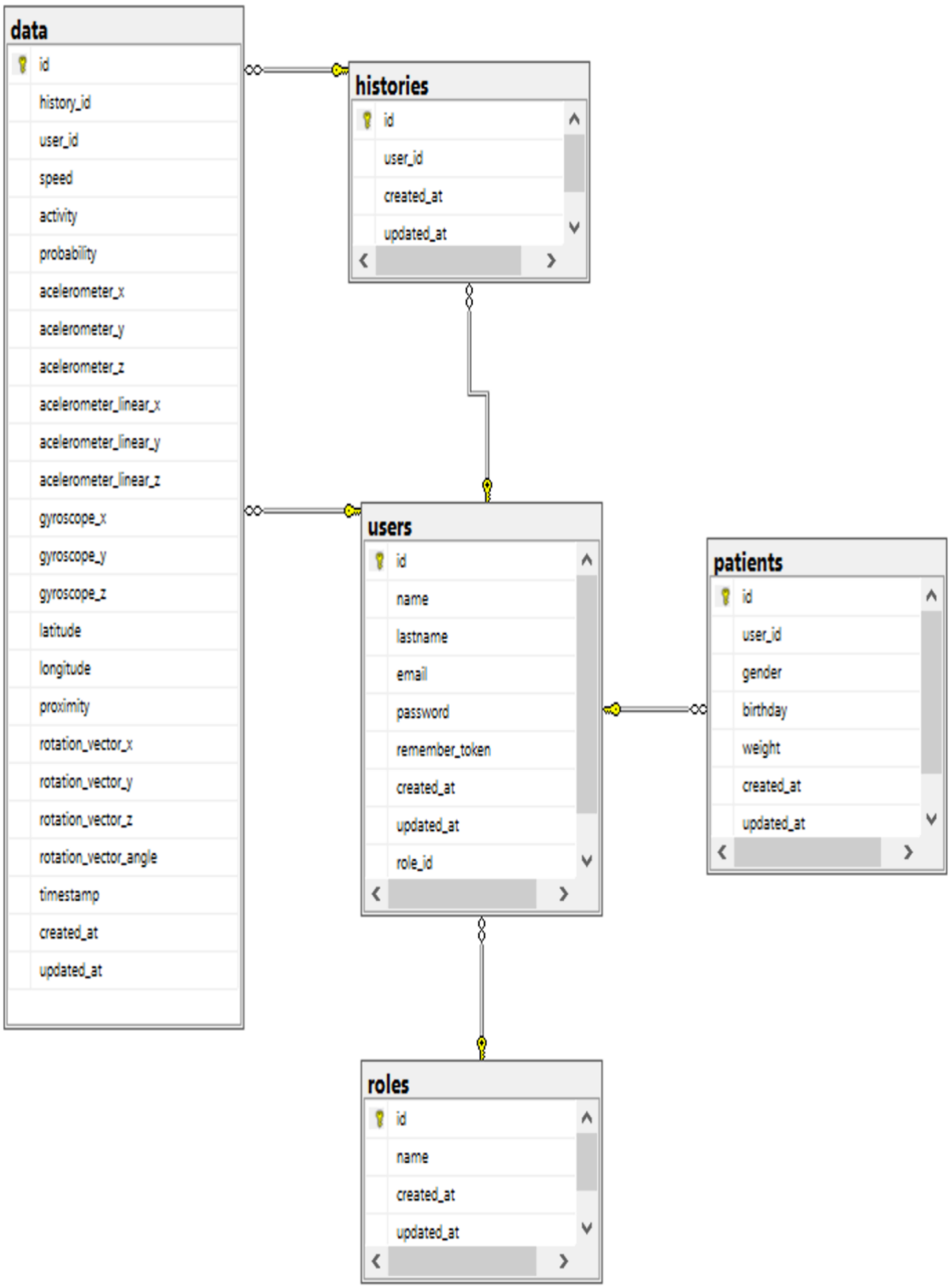

Figura 22. Diagrama de Datos

A continuación, se detalla el diccionario de datos del sistema en las siguientes tablas: Tabla 13, Tabla 14, Tabla 15, Tabla 16, Tabla 17.

Tabla 13

Diccionario de datos de usuarios

Users					
Calve	**Nombre**	**Tipo**	**Tamaño**	**Nulo**	**Descripción**
PK	id	Int	11	No	Clave primaria, identificación de registro
	name	Nvarchar	255	No	Nombre del usuario.
	lastname	Nvarchar	255	Sí	Apellido del usuario
	email	Nvarchar	255	No	Email por el cual el usuario podrá ingresar al sistema. También deberá ser un registro único.
	password	Nvarchar	255	No	Contraseña por el cual el usuario podrá acceder al sistema.
	remember_token	Nvarchar	255	No	Se guardará el token, en caso de persistir la sesión.

FK	role_id	Int	11	No	Clave foránea de la tabla de roles.
	created_at	Datetime		Sí	Tiempo en el cual la información se ingreso en la base de datos.
	updated_at	Datetime		Sí	Tiempo en el cual la información se actualizó en la base de datos.

Tabla 14

Diccionario de datos de roles

Roles					
Calve	**Nombre**	**Tipo**	**Tamaño**	**Nulo**	**Descripción**
PK	id	Int	11	No	Clave primaria, identificación de registro
	name	Nvarchar	255	No	Nombre del rol.
	created_at	Datetime		Sí	Tiempo en el cual la información se ingreso en la base de datos.

	updated_at	Datetime		Sí	Tiempo en el cual la información se actualizó en la base de datos.

Tabla 15

Diccionario de datos de pacientes

Patients					
Calve	**Nombre**	**Tipo**	**Tamaño**	**Nulo**	**Descripción**
PK	id	Int	11	No	Clave primaria, identificación de registro
FK	user_id	Int	11	No	Clave foránea que asocia la información de un paciente con un usuario del sistema.
	gender	Tinyint		No	Masculino = 1, Femenino = 2
	birthday	Date		No	Fecha de cumpleaños del paciente.
	weight	Float		No	Peso del paciente
	daily_goal	Int		No	Muestra la meta que va a tener el

					paciente expresado en minutos.
	created_at	Datetime		Sí	Tiempo en el cual la información se ingresó en la base de datos.
	updated_at	Datetime		Sí	Tiempo en el cual la información se actualizó en la base de datos.

Tabla 16

.Diccionario de datos de data

Data					
Calve	**Nombre**	**Tipo**	**Tamaño**	**Nulo**	**Descripción**
PK	id	Int	11	No	Clave primaria, identificación de registro
FK	user_id	Int	11	No	Clave foránea de la relación con la tabla de usuarios
	speed	Float		Sí	Contiene la velocidad en la cual se está desplazando

					en el tiempo de la muestra.
	activity	Nvarchar	255	Sí	Nombre de la actividad física que se está realizando.
	probability	Smallint		Sí	Probabilidad de la actividad que se está realizando
	aceleromenter_x	Float		Sí	Eje x del sensor acelerómetro.
	aceleromenter_y	Float		Sí	Eje y del sensor acelerómetro
	aceleromenter_z	Float		Sí	Eje z del sensor acelerómetro
	acelerometer_linear_x	Float		Sí	Eje x del sensor acelerómetro linear
	acelerometer_linear_y	Float		Sí	Eje y del sensor acelerómetro linear
	acelerometer_linear_z	Float		Sí	Eje z del sensor acelerómetro linear

	giroscope_x	Float		Sí	Eje x del sensor giroscopio
	giroscope_y	Float		Sí	Eje y del sensor giroscopio
	giroscope_z	Float		Sí	Eje z del sensor giroscopio
	latitude	Float		Sí	Valor de latitud del GPS
	longitude	Float		Sí	Valor de longitud del GPS
	proximity	Float		Sí	Valor del sensor de la proximidad.
	rotation_vector_x	Float		Sí	Valor del vector de rotación x
	rotation_vector_y	Float		Sí	Valor del vector de rotación y
	rotation_vector_z	Float		Sí	Valor del vector de rotación z
	rotation_vector_angle	Float		Sí	Valor del ángulo de rotación

	timestamp	Datetime		Sí	Tiempo en el cual los sensores del dispositivo registraron la información
	created_at	Datetime		Sí	Tiempo en el cual la información se ingresó en la base de datos.
	updated_at	Datetime		Sí	Tiempo en el cual la información se actualizó en la base de datos.

Tabla 17

Tabla de historia

Histories					
Calve	**Nombre**	**Tipo**	**Tamaño**	**Nulo**	**Descripción**
PK	id	Int	11	No	Clave primaria, identificación de registro
FK	user_id	Int	11	No	Clave foránea que asocia la información de un paciente

					con un usuario del sistema.
	created_at	Datetime		Sí	Tiempo en el cual la información se ingresó en la base de datos.
	updated_at	Datetime		Sí	Tiempo en el cual la información se actualizó en la base de datos.

4. Implementación

4.1. Configurar el ambiente de desarrollo

Para la configuración del ambiente de desarrollo se requiere de la instalación de algunas herramientas:

- Composer (https://getcomposer.org/download/)
- NodeJs (https://nodejs.org/es/)
- PHP 7.0 en adelante
- Postgresql o SQL Server 2012 en adelante.
- Android Studio 3.1 (https://developer.android.com/studio)
- JDK 8 en adelante.

Para verificar que se ha instalado Composer exitosamente, se ejecuta en el terminal Composer, como se observa en Figura 23.

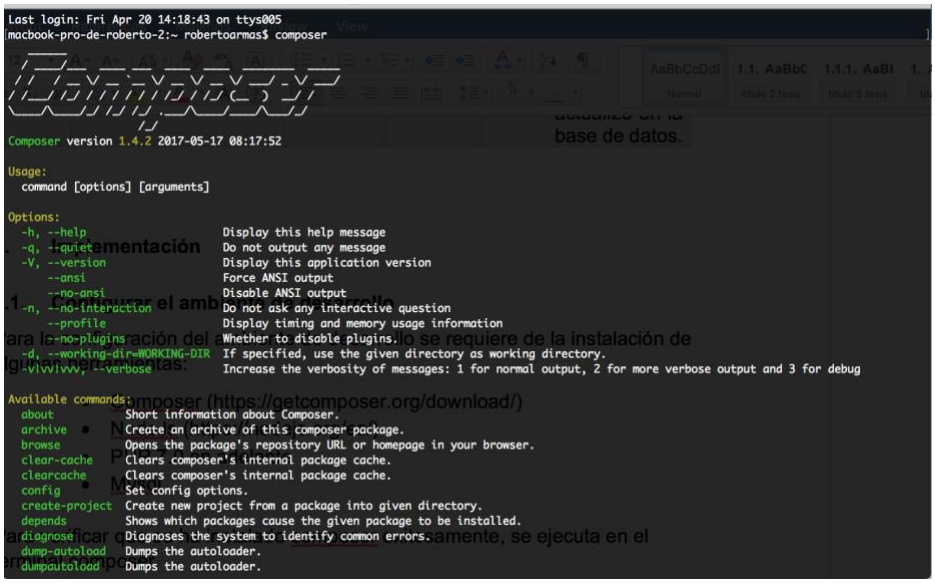

Figura 23. Verificación instalación Composer

Para verificar si PHP (Figura 24) está instalado en el sistema se ejecuta un comando *php –version* y se mostrará la versión que se instaló.

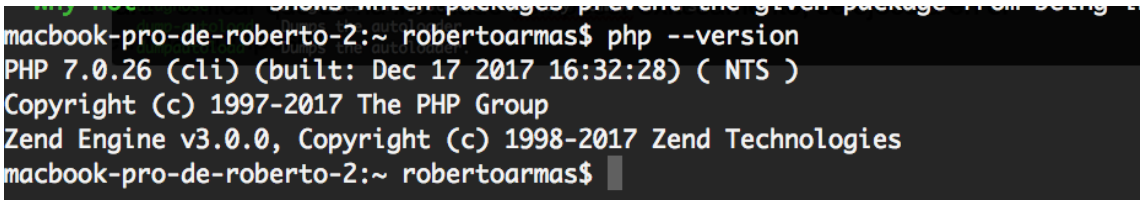

Figura 24. Verificación instalación PHP

Para verificar la instalación de Postgres (Figura 25) se ejecuta el comando *postgres –version* y se mostrará la versión de Postgres está instalado en el sistema.

Figura 25. Verificación de instalación de Postgres

Android Studio (Figura 26) se utilizará para el desarrollo de la aplicación móvil Android, una vez instalado se podrá crear el proyecto que se desea desarrollar.

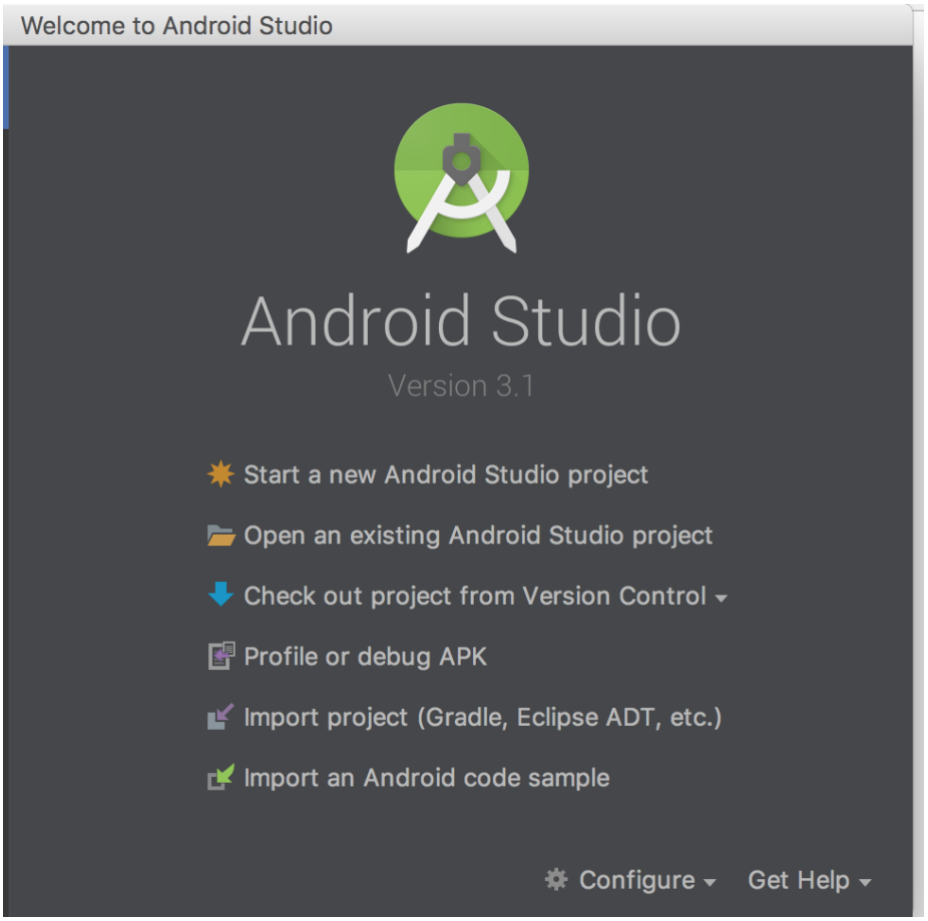

Figura 26. Pantalla inicial Android Studio.

4.2. Configurar el ambiente de producción

Para configurar el ambiente de producción se debe instalar:

- Windows Server 2012 en adelante.
- Microsoft SQL Server 2014 en adelante.
- PHP 7.0 en adelante.
- NodeJS

Se requiere de la instalación de Windows Server 2012 en adelante, como se muestra en la Figura 27, en la información del sistema se puede comprobar la instalación del sistema operativo.

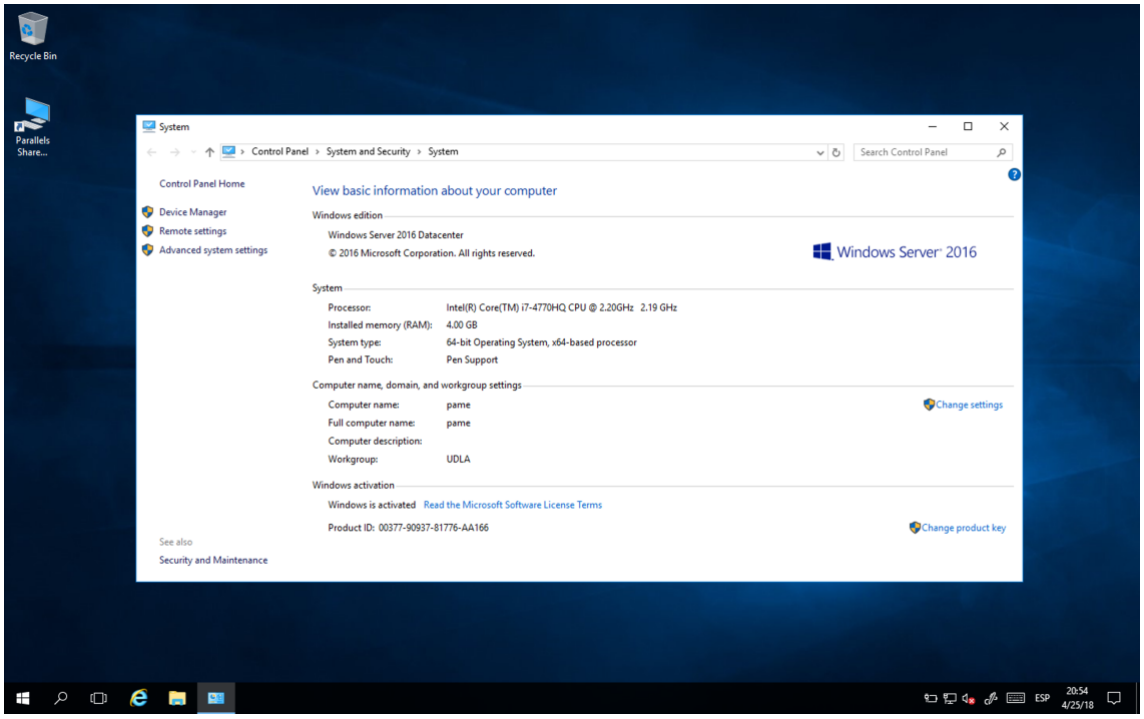

Figura 27. Windows Server 2016.

Para la verificación de la instalación de NodeJS en el ambiente de producción se ejecuta en la línea de comandos *node –version* como se observa en Figura 28.

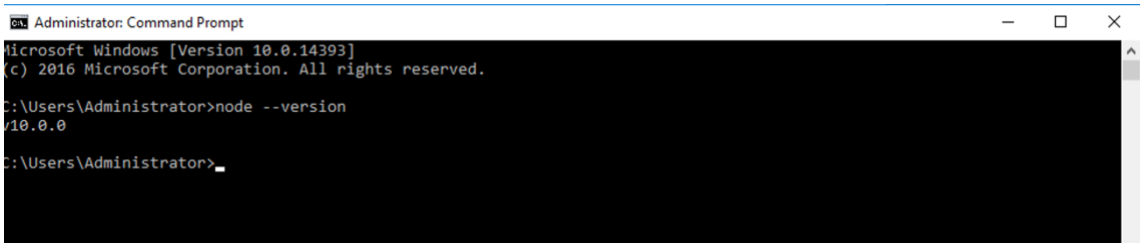

Figura 28. Verificación de instalación NodeJS en Windows Server.

Para el ambiente de producción se debe instar Microsoft SQL Server, como se observa en la Figura 29. Se puede observar que el servicio de base de datos se encuentra levantado.

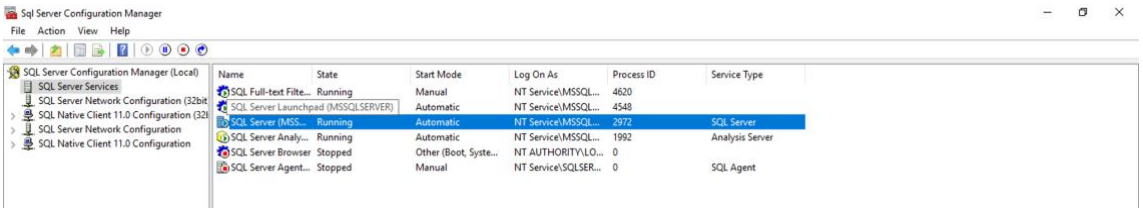

Figura 29. Microsoft SQL Server corriendo.

Se debe instalar la versión de PHP más reciente en el ambiente de producción para verificar la versión se ejecuta en la línea de comandos *php –version* como se observa en la Figura 30.

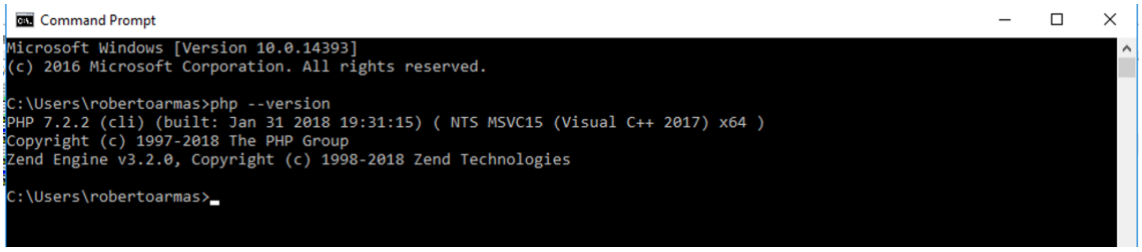

Figura 30. Verificación de instalación de PHP en Windows Server.

4.3. Descripción de componentes de aplicación web

RF1.1. Registrar paciente

En la Figura 31 se muestra la pantalla diseñada para el ingreso de información del paciente que será llenada por el médico que trata al paciente.

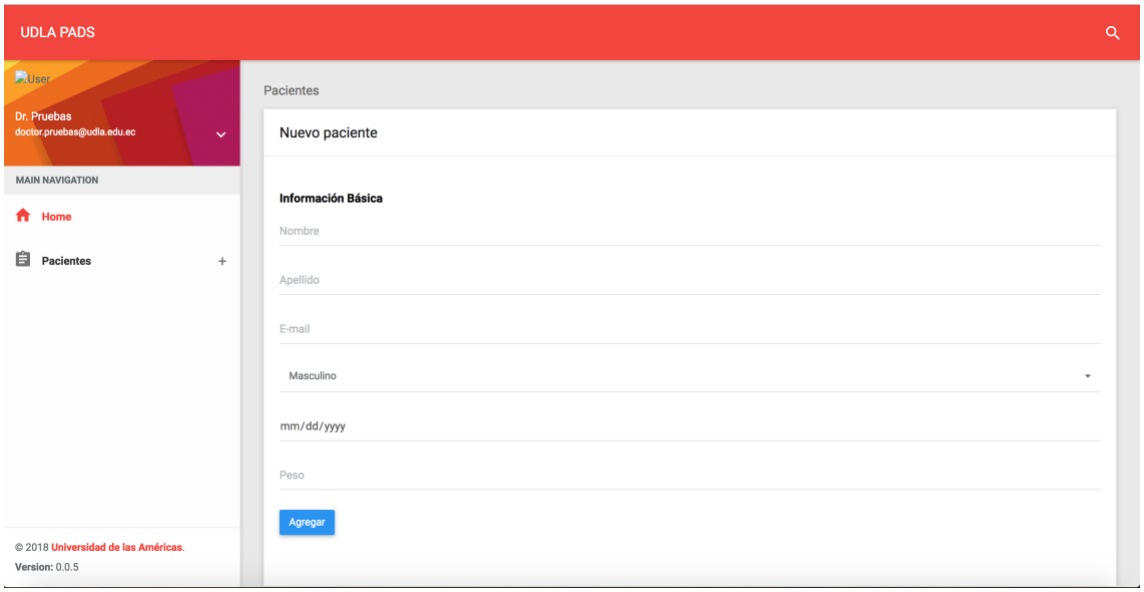

Figura 31. *Pantalla de registrar paciente.*

En Figura 32 se muestra un ejemplo como se deben llenar los campos con la información para dar de alta al paciente en el sistema. Los campos que contiene el formulario son los siguientes:

- Nombre: Se ingresará el nombre del paciente.
- Apellido: Se ingresará el apellido del paciente.
- E-mail: Se ingresará la dirección de correo electrónico del paciente, posterior servirá para enviar el email con la contraseña temporal con la cual el paciente podrá ingresar al sistema.
- Género: Se indicará el género del paciente (masculino, femenino).
- Fecha de nacimiento: Indicará el día, mes y año de nacimiento del paciente.
- Peso: Se registra el peso actual del paciente en kilogramos.
- Objetivo diario: Objetivo diario en minutos, el cual se propone realizar el paciente la actividad física.

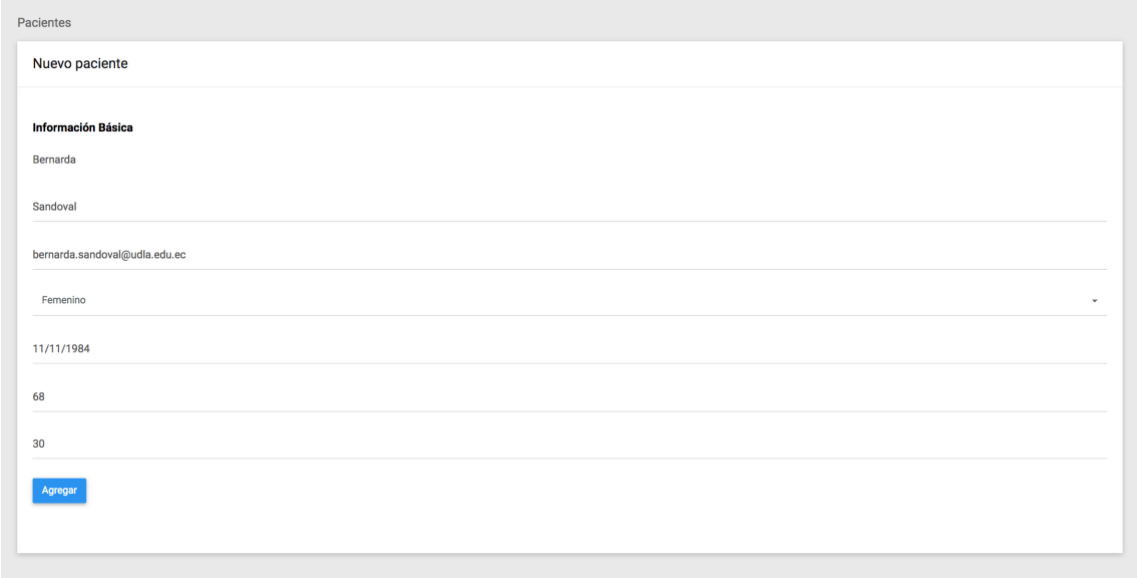

Figura 32. *Ejemplo de registro paciente.*

Una vez que se llenaron correctamente los datos en los campos del formulario al presionar el botón de “Agregar”, una alerta (Figura 33) se mostrará indicando que el paciente se ha creado exitosamente.

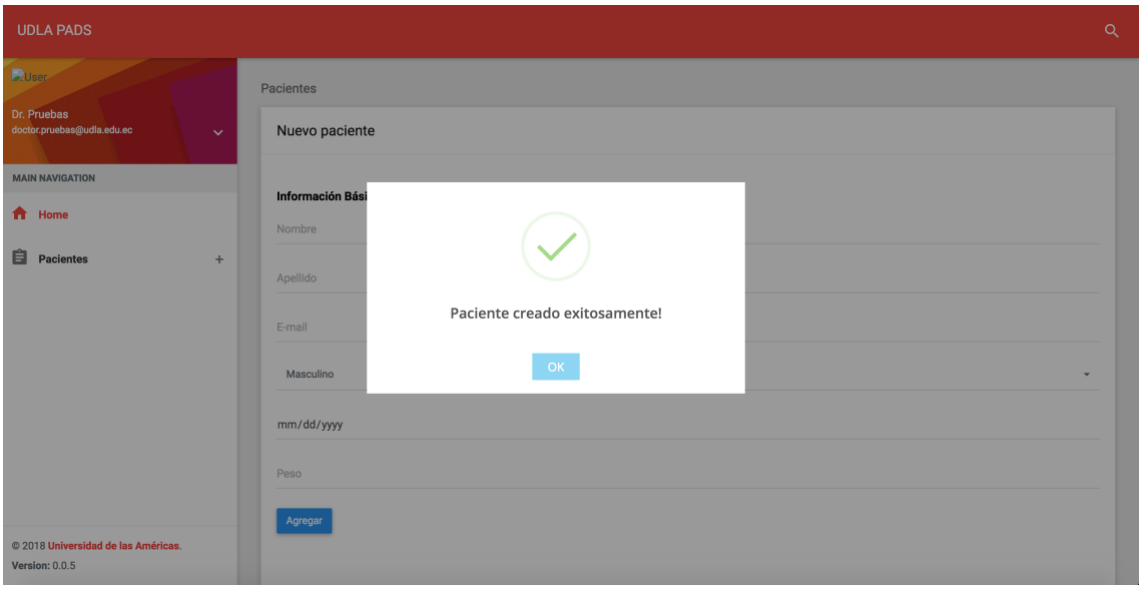

Figura 33. Confirmación de registro de paciente.

Cuando se dé de alta al paciente en el sistema, se enviará un e-mail (Figura 34) a la dirección de correo electrónico que se indicó en el formulario de registro. El

mail contendrá la contraseña temporal con el cual el paciente podrá ingresar a la aplicación móvil.

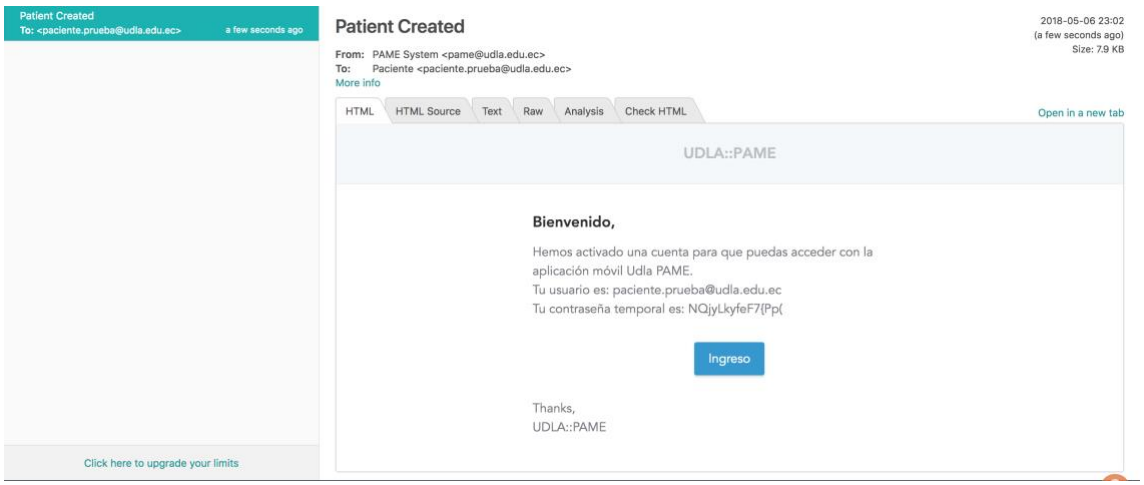

Figura 34. Email de paciente con contraseña temporal.

RF2.2. Autenticar médico

En la Figura 35 se muestra la pantalla de ingreso del médico para poder acceder al sistema. La pantalla de ingreso consta de dos campos:

- Email: es el email por el cual el se dio de alta al médico.
- Contraseña: contraseña por la cual el médico ingresa al sistema.

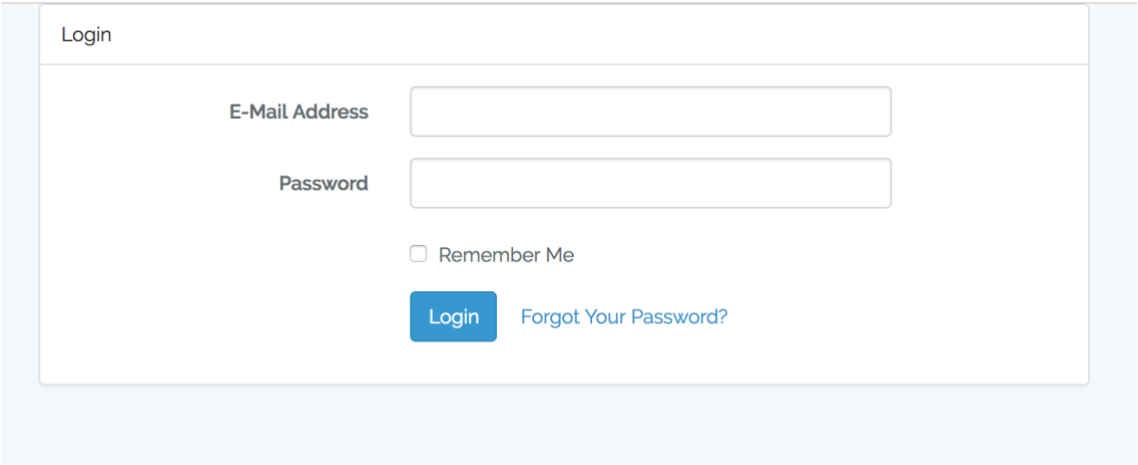

Figura 35. Pantalla de autenticación médico.

RF4.1. Ver reporte de actividad

Para mostrar la distancia recorrida en kilómetros se utiliza la vista de la Figura 36. Muestra una línea de color magenta en la cual el eje Y representa la distancia en km y el eje X los días en la cual se realizó la actividad física. Así el médico tendrá una percepción de la distancia recorrida durante el mes (opción por defecto).

Figura 36. *Reporte de distancia.*

En la Figura 37, se muestra el reporte del tiempo siendo el Eje X de los días del mes y el eje Y del tiempo en minutos. Además de mostrar dos barras por cada

día, la barra de color azul representa al estado de caminar y la barra de color magenta representa el estado de correr. Además de mostrar la línea roja de objetivo diario.

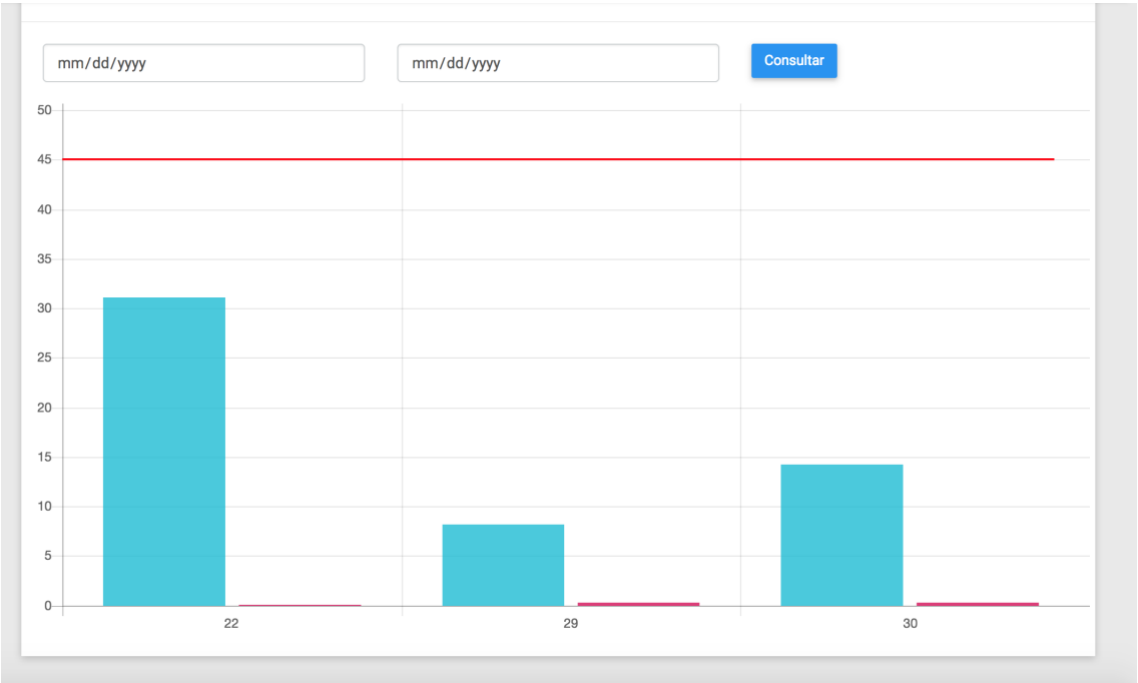

Figura 37. *Reporte de tiempo.*

En la Figura 38, se muestra el mapa de ruta por la cual se ha caminado o corrido. Esto es útil para dar seguimiento en caso de emergencia a pacientes de la tercera edad o como herramienta para su seguridad.

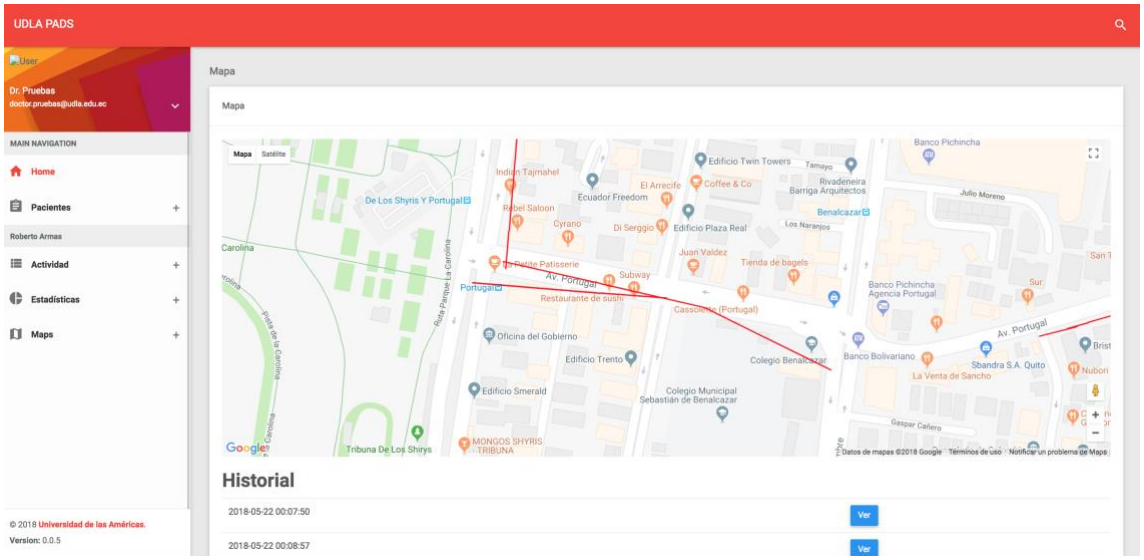

Figura 38. *Mapa de ruta*.

4.4. Descripción de componentes de aplicación móvil

RF2.1. Autenticar paciente

Para la implementación del requisito funcional se diseñó una pantalla de ingreso al sistema, como se observa en Figura 39. Esto es lo que verá el paciente que desee ingresar a la aplicación. Consta de dos campos:

- Usuario: En este campo se llena el correo que el médico registró en el sistema.
- Contraseña: En este campo se llena la contraseña temporal / cambiada, que pertenece al usuario que se desea autenticar.

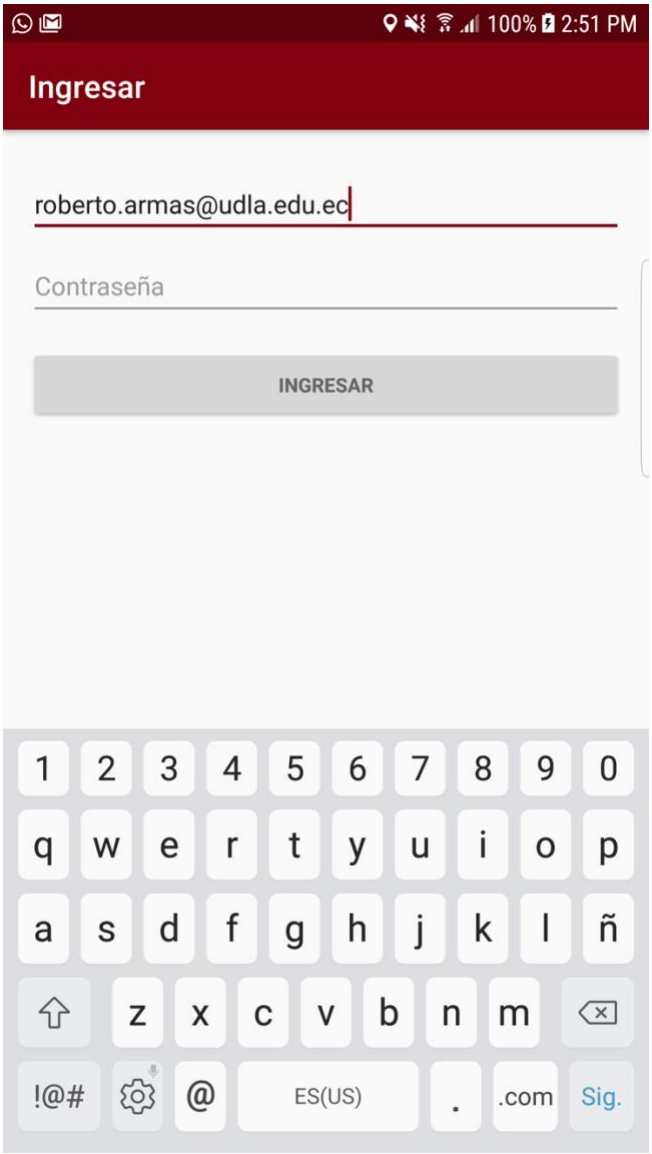

Figura 39. Pantalla de ingreso en aplicación móvil.

RF3.1. Registrar actividad

Para registrar los datos de actividad física del rol paciente se diseñó una pantalla, como se observa en la Figura 40, que muestra la información importante como: velocidad, distancia, actividad, probabilidad y tiempo. De esta manera el paciente puede verificar que la información que se este generando sea correcta.

La pantalla presenta además un botón de iniciar / parar, este botón se encargará de iniciar o detener la transmisión de los datos al servidor.

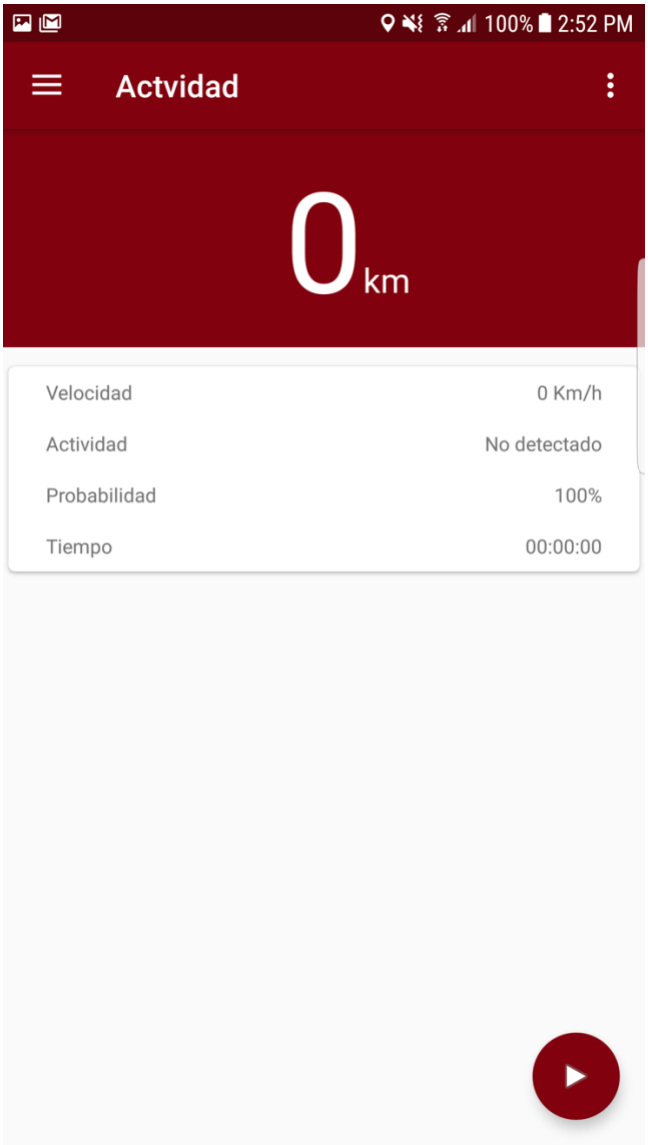

Figura 40. Pantalla de registro de actividad.

4.5. Implementación de ORM

Para el proyecto se desarrolló un ORM. Este componente se desarrollo e implementó con la intención de que los objetos que se manipulan en NodeJS App (Figura 19) puedan ser almacenados en la base de datos Microsoft SQL Server y Postgres SQL.

El ORM utilizado consta de los siguientes componentes:

Model: Clase modelo que representa a una entidad de la base de datos.

Adapter: Es el adaptador utilizado para comunicar con un motor de base de datos específico, en el caso del proyecto será con PGSQL o MSSQL Server.

Builder: Es la clase que construirá la consultas a la base de datos como un límite, unión, donde.

El ORM permite que se facilite la mantenibilidad del software que se desarrolla, además de proveer independencia de base de datos. Debido a que el proyecto se enfoca a ser una solución comercializable a distintas entidades, es importante mantener una independencia en base de datos, el sector público puede requerir el uso de Software Libre como es PGSQL, mientras el sector privado requiera utilizar una base de datos propietaria como es Microsoft SQL Server.

En el proyecto se realizó las respectivas pruebas con MySQL pero la base de datos no soporta este tipo de aplicación. Con 1500 registros en la vista de patient_data el servidor mostró un error de time-out (Figura 41). Por lo cual se demuestra que las consultas sobre vistas no son óptimas en MySQL.

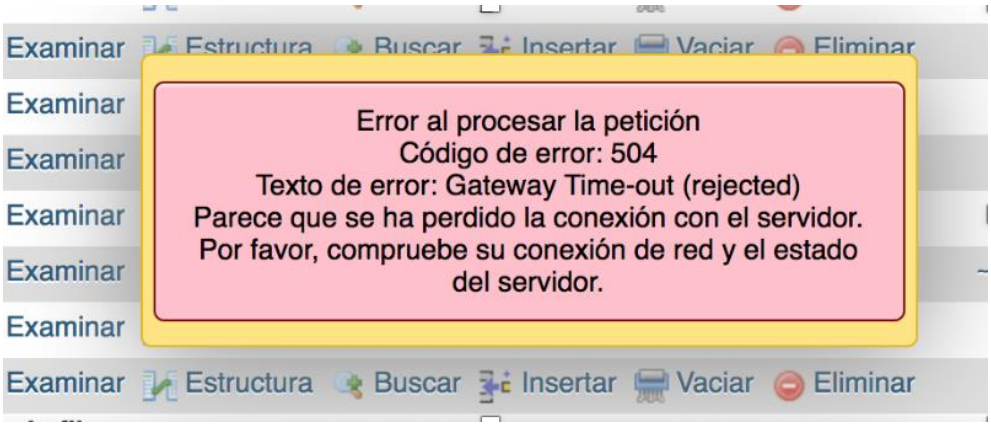

Figura 41. MySQL time-out.

5. Análisis de resultados

En la (Figura 42) se muestran los datos recuperados desde el teléfono móvil a través de la aplicación Android. Adicionalmente, se muestra la actividad física y se registran los valores que los sensores están capturando. De esta manera, con

toda la información disponible en la base de datos se podrá llegar a recuperar información para procesarla a futuro.

id	history_id	user_id	speed	activity	probability	acelerometer_x	acelerometer_y	acelerometer_z	acelerometer_linear_
4	1	23	1.800000000000000000	A PIE	71	1.049999952316284200	2.440000057220459000	9.659999847412110000	0.22999998927116394
5	1	23	0.000000000000000000	A PIE	71	-0.879999995231628400	2.849999904632568400	9.569999694824219000	-0.63999998569488530
6	1	23	0.000000000000000000	A PIE	71	-0.879999995231628400	2.849999904632568400	9.569999694824219000	-0.63999998569488530
7	1	23	2.700000000000000000	A PIE	71	-0.810000002384185800	2.059999942779541000	9.719999313354492000	1.75999999046325680
8	1	23	2.700000000000000000	A PIE	71	-0.810000002384185800	2.059999942779541000	9.719999313354492000	1.76999998092651370
9	1	23	0.000000000000000000	A PIE	71	-1.139999985694885300	2.750000000000000000	9.229999542236328000	-0.26999998092651367
10	1	23	0.000000000000000000	A PIE	71	-1.139999985694885300	2.750000000000000000	9.229999542236328000	-0.07999999821186066
11	1	23	1.800000000000000000	A PIE	87	1.599999904632568400	3.159999847412109400	5.730000019073486000	0.71999996900558470
12	1	23	5.400000000000000000	A PIE	87	7.359999656677246000	1.110000014305114700	12.659999847412110000	1.95999991893768300
13	1	23	5.400000000000000000	A PIE	87	7.359999656677246000	1.110000014305114700	12.659999847412110000	1.95999991893768300
14	1	23	6.300000000000000000	A PIE	87	3.949999809265136700	-1.860000014305114700	3.619999885559082000	-3.50999999046325700
15	1	23	6.300000000000000000	A PIE	87	3.949999809265136700	-1.860000014305114700	3.619999885559082000	-4.11000013351440400
16	1	23	4.500000000000000000	A PIE	87	6.369999885559082000	-2.519999980926513700	2.059999942779541000	0.21999999880790710
17	1	23	4.500000000000000000	A PIE	87	17.609998703002930000	-2.829999923706054700	6.679999828338623000	7.87999963760376000
18	1	23	4.500000000000000000	A PIE	87	17.609998703002930000	-2.829999923706054700	6.679999828338623000	7.07999992370605500
19	1	23	4.500000000000000000	A PIE	99	0.550000011920929000	1.939999938011169400	4.079999923706055000	-0.70999997854232790
20	1	23	5.400000000000000000	A PIE	99	1.870000004768371600	1.689999938011169400	13.409999847412110000	0.68999999761581430
21	1	23	4.500000000000000000	A PIE	99	0.919999957084655800	0.669999957084655800	9.699999809265137000	0.58999997377395630
22	1	23	4.500000000000000000	A PIE	99	1.519999980926513700	-0.370000004768371600	7.179999828338623000	0.28999999165534980
23	1	23	4.500000000000000000	A PIE	99	1.519999980926513700	-0.370000004768371600	7.179999828338623000	0.28999999165534980
24	1	23	3.600000000000000000	A PIE	99	2.470000028610229500	0.829999983310699600	7.299999713897705000	1.24000000953674320
25	1	23	3.600000000000000000	A PIE	99	2.470000028610229500	0.829999983310699600	7.299999713897705000	-0.69999998807907100

Figura 42. Captura de datos.

Los datos almacenados se despliegan mediante un reporte de actividad física que generó el paciente de pruebas como se observa en Figura 43, las barras de color azul muestran el tiempo en minutos por día. Estas pruebas se realizaron el mes de marzo del año 2018. De igual manera se hicieron pruebas simultáneas (dos usuarios al mismo periodo de tiempo) como se observa el 5 de marzo del 2018 mediante la Figura 43 y Figura 44.

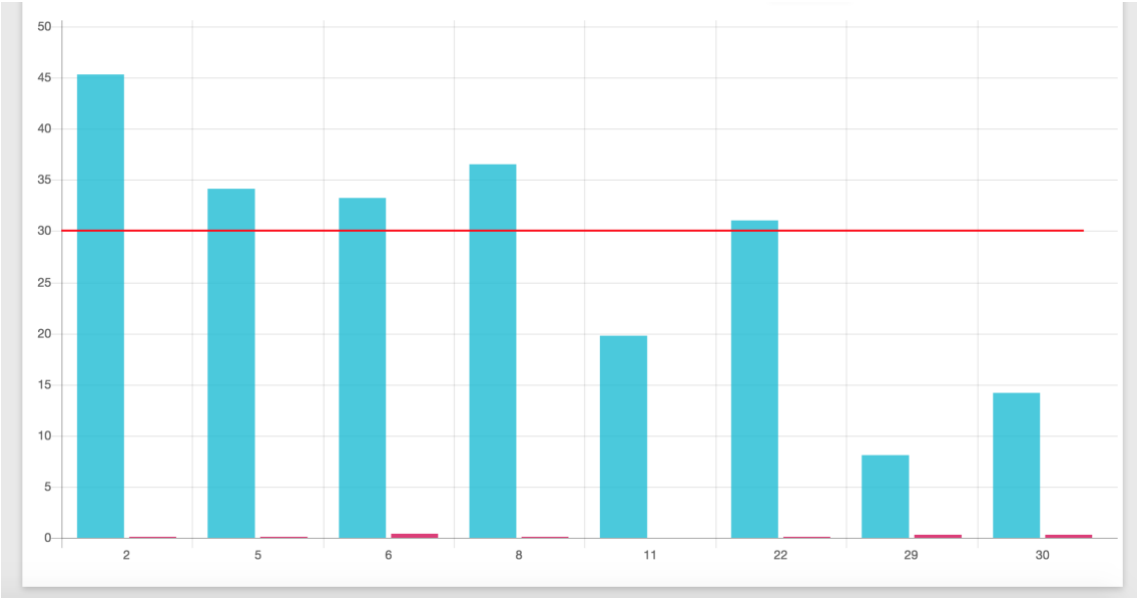

Figura 43. Reporte de tiempo de actividad física.

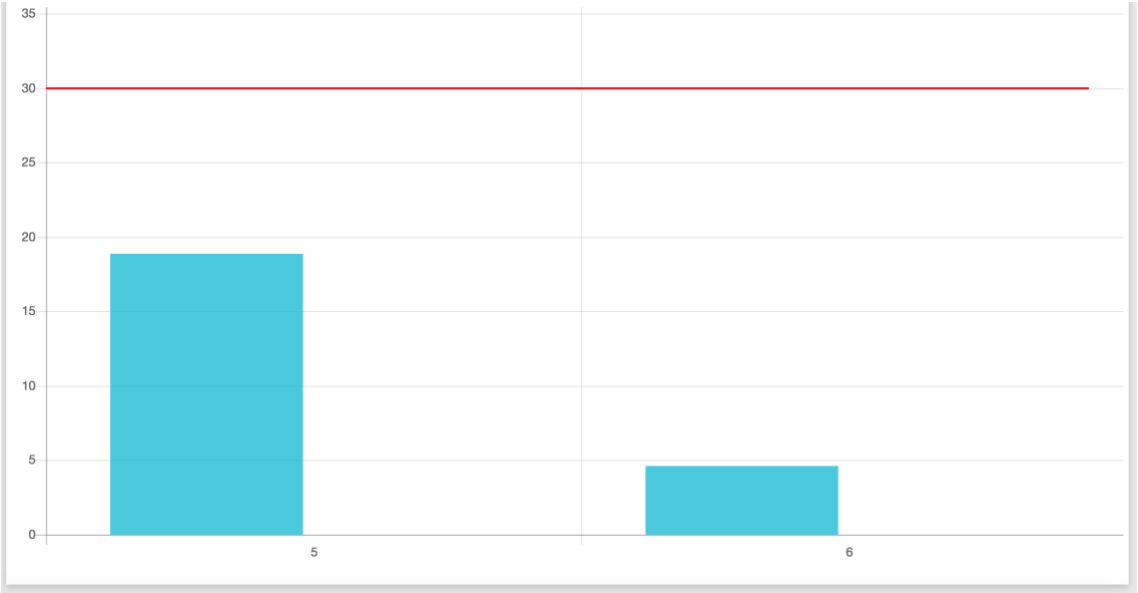

Figura 44. Reporte tiempo paciente 2.

Este historial de la actividad física es capaz de ser graficada mediante un conjunto de líneas poligonales que representan la ruta (Figura 45) por la cual el paciente se movió.

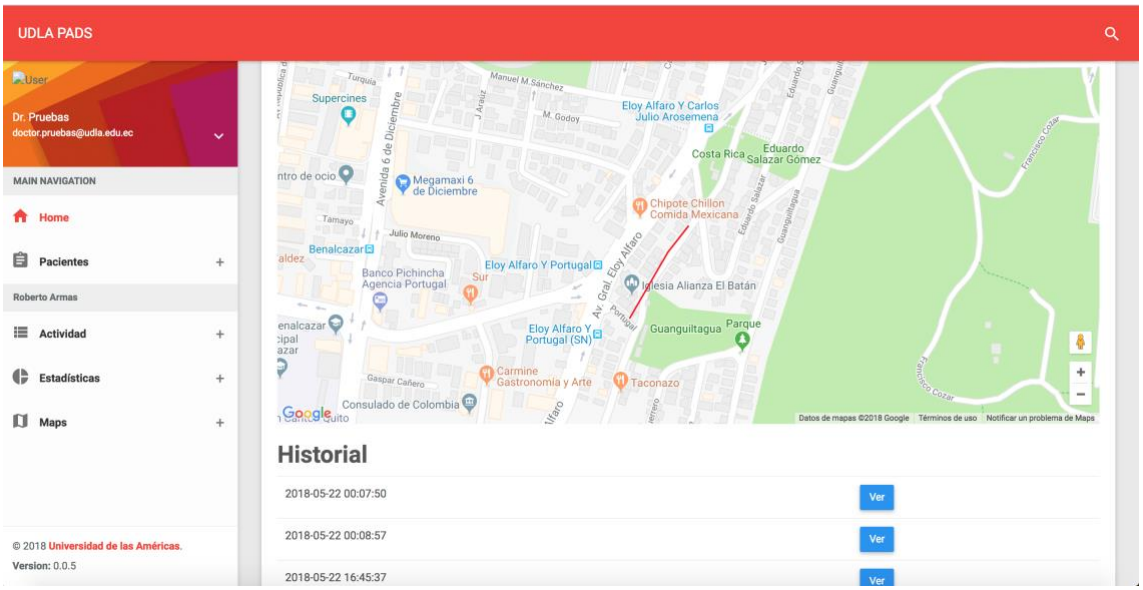

Figura 45. Gráfico de ruta.

6. Análisis de costos

Para evaluar la factibilidad económica del proyecto, se realizó un análisis financiero basándose en los conceptos adquiridos en la materia de ingeniería económica. Para ello se realiza un análisis de la tasa interna de retorno (TIR) y Valor Actual Neto (VAN) con el flujo de caja en una proyección de 5 años (Tabla 18), obteniendo una TIR de 146.55%, para poderlo comparar como una inversión en póliza con un banco, el banco ofrece una TIR de 3 – 5%. Por lo tanto, realizar una inversión en el presente proyecto resulta atractivo.

Para el proyecto se analizó la adquisición de un préstamo de $103,602.40 al banco para poder financiar el proyecto a una tasa del 16% que serán pagados en 4 años.

En la Tabla 18 se muestra que los gastos en sueldos y salarios es de 95 mil anuales. El valor representa al salario de un Chief Executive Officer (CEO), un contador, un ingeniero en sistemas, tres recursos en soporte, tres encargados de vender y un ejecutivo de cuentas. Para el proyecto se buscó una oficina por el sector de La Carolina con un valor mensual del $700 dólares americanos. Se plantea fomentar la investigación y desarrollo por lo cuál se contempla un presupuesto de 10 mil dólares en capacitaciones al personal.

Tabla 18

Análisis de Flujo de operaciones a 5 años

	PROYECTO					
	FLUJO DE OPERACIONES CON PROYECCIÓN A 5 AÑOS					
	0	**1**	**2**	**3**	**4**	**5**
Ventas		$ 147,852.00	$ 377,622.00	$ 773,226.00	$ 1,168,830.00	$ 1,564,434.00
Publicidad		$ 100.00	$ 100.00	$ 100.00	$ 100.00	$ 100.00
Total Costos de Ventas		$ 13,274.04	$ 15,928.85	$ 19,114.62	$ 22,937.54	$ 27,525.05
Utilidad Bruta		**$ 134,477.96**	**$ 361,593.15**	**$ 754,011.38**	**$ 1,145,792.46**	**$ 1,536,808.95**
Sueldos y Salarios		$ 95,094.86	$ 98,898.66	$ 102,854.60	$ 106,968.79	$ 111,247.54
Servicios Profesionales		$ 6,000.00	$ 6,000.00	$ 6,000.00	$ 6,000.00	$ 6,000.00
Servicios Básicos (Agua, Luz, Fono, Inte.)		$ 10,200.00	$ 10,200.00	$ 10,200.00	$ 10,200.00	$ 10,200.00
Servicios Bancarios		$ -	$ -	$ -	$ -	$ -
Capacitaciones		$ 10,200.00	$ 10,200.00	$ 10,200.00	$ 10,200.00	$ 10,200.00
Suministros		$ 400.00	$ 400.00	$ 400.00	$ 400.00	$ 400.00
Otros Aprovisionamientos		$ 1,200.00	$ 1,200.00	$ 1,200.00	$ 1,200.00	$ 1,200.00
Total Gastos Operativos		**$ 123,094.86**	**$ 126,898.66**	**$ 130,854.60**	**$ 134,968.79**	**$ 139,247.54**
Gastos Depreciación		$ 1,263.80	$ 1,390.18	$ 1,529.20	$ 1,682.12	$ 1,850.33
Gastos Amortización		$ -	$ -	$ -	$ -	$ -
Gastos Financieros		$39,068.20	$ 39,018.67	$ 38,960.54	$ 19,455.69	$ -
Utilidad antes de Impuestos		$ (28,948.90)	$ 194,285.64	$ 582,667.04	$ 989,685.86	$ 1,395,711.08
Impuestos (36 %)		$ (10,421.60)	$ 69,942.83	$ 209,760.13	$ 356,286.91	$ 502,455.99
Utilidad Neta		**$ (18,527.30)**	**$ 124,342.81**	**$ 372,906.91**	**$ 633,398.95**	**$ 893,255.09**
FLUJO DE CAJA		-17,263.50	125,732.99	374,436.10	635,081.07	895,105.42
UTILIDAD EN EFECTIVO		-17,263.50	125,732.99	374,436.10	635,081.07	895,105.42
CAPITAL DE TRABAJO	0.00	0.00	0.00	0.00	0.00	0.00
INVERSION	103,602.40					
FINANCIAMIENTO		0.00	0.00	0.00	0.00	0.00
VALOR RESIDUAL						3,454,446.80
VALOR RESIDUAL capital trabajo						0.00
FLUJO DE CAJA NETO	-103,602.40	-17,263.50	125,732.99	374,436.10	635,081.07	4,349,552.22
		-13,810.80	**80,469.12**	**191,711.29**	**260,129.21**	**1,425,261.27**
VAN	**1,840,157.68**					
TIR	**146.55%**					
VAN	1,840,157.68					

Dado los siguientes costos de operación:

Los costos de operación anuales (Tabla 19) cuentan con un servicio en la nube de Azure de Microsoft, con la finalidad de minimizar los costos en infraestructura.

Tabla 19

Costos de operación

Anuales	**$ 13,274.04**
Windows Server Azure Cloud 8 Core RAM 32GB 1TB	$13,175.04
Certificado Organizacional iOS	$ 99.00
Inicial	**$ 3,742.00**
SQL Server	$ 3,717.00
Mysql	$ -
Certificado Google Play	$ 25.00
Gastos Año 1	**$ 17,016.04**

Se espera en el año 1 llegar a 1500 clientes, estos clientes podrán ser los médicos traumatólogos, fisioterapistas y afines a tratamientos relacionados a actividad física. El plan incluirá una subscripción mensual de USD 9.99 por los servicios que brinda el sistema. Ellos tendrán un número ilimitado de pacientes.

7. Conclusiones y Recomendaciones

7.1. Conclusiones

Se observó que al implementar un protocolo en tiempo real (websocket) se optimiza el consumo de datos en los dispositivos móviles, mientras que al enviar las cabeceras en la trama de cada petición se tiene un consumo mayor. Este protocolo provee un método de envió para la lectura remota de variables de los sensores que es recuperada por los teléfonos inteligentes.

La creación de un portal web a través del cual el médico tratante puede llevar un control de la actividad física que realizan varios pacientes, constituye un valor agregado respecto a otras aplicaciones en el mercado, las mismas que se limitan a mostrar la información únicamente para el usuario. Los reportes se pueden personalizar de acuerdo con los requerimientos de los médicos tratantes.

Se concluye que el proyecto puede ser una iniciativa de plataforma de eHealth en donde se provee una interacción directa entre paciente y médico.

El sistema tiene la versatilidad de utilizar dos motores de bases de datos, con esto se puede decir que el sistema se alinea con las políticas de uso de software libre en Ecuador. Por otro lado, si un cliente privado pone como exigencia el uso de un motor de base de datos propietario, el sistema también está diseñado para ello.

En el desarrollo del proyecto se identificó que no todos los Data Base Managment System (DBMS) existentes en el mercado se pueden utilizar para el tipo de aplicación propuesta. Se realizaron las pruebas con MySQL, mostrando resultados negativos. Se comprobó que MySQL no está diseñado para este tipo de aplicación. Esto se debe a que el gestor de base de datos no soporta consultas complejas para los reportes.

Se han cumplido los objetivos establecidos ya que la aplicación realiza la identificación de la actividad que el paciente está realizando y se da un seguimiento para conocer si ha cumplido el objetivo o no lo ha logrado.

Los resultados del proyecto realizado se presentaron en el congreso International Conference on eDemocracy & eGoverment 2018 (ICEDEG) que se encuentra publicado en IEEE Xplore: https://ieeexplore.ieee.org/document/8372320/.

Se concluye que, con el trabajo realizado, se ha mejorado la precisión de Google Fit en las actividades de corriendo y caminando de 94% a 98%, según las pruebas realizadas. Google Fit ayuda a clasificar la información que se va recolectando para la generación de los reportes. La mejora en cuestión de precisión puede ser importante en tomas de datos que requieren de un seguimiento de un largo periodo de tiempo. Google Fit detecta también actividades de ir en bici o en vehículo por lo que esta mejora solo se centra en actividades de caminar y correr.

7.2. Recomendaciones

En el proyecto se recomienda el uso de una base de datos libre como es Postgres, ya que se comprobó que minimiza los costos de operación. El sistema está alineado a los requerimientos del Estado ecuatoriano con el uso de software libre. Si existiese el caso de usar una base de datos propietaria, el sistema soporta Microsoft SQL Server.

Se recomienda al lector, si desea replicar o continuar el proyecto utilizar el reconocimiento de Google Fit para las aplicaciones que involucren actividad física. Las pruebas realizadas al usar con el SDK demostraron tener efectividad y precisión en las actividades que se probaron: caminar y correr.

Para futuras versiones se propone el envío de la información al servidor, haciendo resúmenes cada cierto periodo, con la finalidad de disminuir la cantidad de datos enviados al servidor que no se están usando. Sin embargo, para proyectos de investigación donde se requiera realizar análisis estadísticos detallados, se recomienda mantener la forma de envío que se encuentra actualmente en el sistema, ya que los datos brindados ayudarán a realizar estudios y estadísticas de los movimientos del paciente.

Durante las pruebas se realizaron pruebas simultaneas, con todos los nodos en un solo servidor con dos usuarios, es recomendable probar con más usuarios con el fin de determinar la concurrencia que soporta la aplicación del sistema.

REFERENCIAS

Armas, R., Buenaño, D., Rybarczyk, Y., & Gonzáles, M. (2018). PAME : Physical Activity Monitoring for the Elderly. In *2018 International Conference on eDemocracy & eGovernment (ICEDEG)* (pp. 361–365). Ambato: IEEE. https://doi.org/10.1109/ICEDEG.2018.8372320

ATC, A. (2017). *Java Fundamentals for Android™ Development*. Android ATC. Recuperado el 15 de Abril del 2018 de www.androidatc.com

Atlassian. (2018). Scrum | Una breve introducción | Atlassian - El orientador ágil. Recuperado el 31 de Marzo del 2018 de https://es.atlassian.com/agile/scrum

Chernbumroongab, S., Cangc, S., Atkinsa, A., & Yud, H. (2013). Elderly activities recognition and classification for applications in assisted living. *ELSEIVER*, 1662–1674. https://doi.org/https://doi.org/10.1016/j.eswa.2012.09.004

CODDii. (2017). CONFERENCIA DE DIRECTORES Y DECANOS DE INGENIERÍA INFORMÁTICA eHealth (tecnología y medicina). *Coddinforme*, 8. Recuperado el 15 de Abril del 2018 de http://coddii.org/wp-content/uploads/2017/01/Informe-e-Health-2.pdf

Ddegjust. (2018). OPP with C++. Recuperado el 15 de Abril del 2018 de http://www.ddegjust.ac.in/studymaterial/mca-3/ms-17.pdf

De la Torre Llorente, C., Zorilla Castro, U., Calvarro Nelson, J., & Ramos Barroso, M. Á. (2010). *Arquitectura Macro de N-Capas*. *Krasis Press 2014*. https://doi.org/10.1007/s13398-014-0173-7.2

Developers, G. (2015). Android APIs | Google Fit | Google Developers. Recuperado el 2 de Abril del 2018 de https://developers.google.com/fit/android/

Dzone. (2018). Reasons Why Laravel Is the Best PHP Framework in 2018 - DZone Web Dev. el 15 de Abril del 2018 de https://dzone.com/articles/reasons-why-laravel-is-the-best-php-framework-in-2

Express. (2014). Express - Infraestructura de aplicaciones web Node.js. Recuperado el 2 de Abril del 2018 de http://expressjs.com/es/

Freeman, A. (2015). *Pro Design Patterns in Swift*. New York. https://doi.org/10.1007/978-1-4842-0394-1

Google. (2017). Funciones de Android Studio | Android Studio. Recuperado el 3 de Abril del 2018 de https://developer.android.com/studio/features.html?hl=es-419

Leap Fitness Group. (2018). Contador de pasos -Podómetro, contador de calorías - Aplicaciones en Google Play. Recuperado el 2 de Mayo del 2018 de https://play.google.com/store/apps/details?id=pedometer.steptracker.calorieburner.stepcounter&hl=es

Ministerio de Educación, C. y D. (2017). Historia y tecnología, 1–50. Recuperado el 2 de Abril del 2018 de http://recursostic.educacion.es/secundaria/edad/4esotecnologia/quincena1/pdf/4quincena1.pdf

Moquillaza Henríquez, S. D., Vega Huerta, H., & Guerra Grados Luis. (2010). Programación en N capas. *Revista de Investigación de Sistemas e Informática*. Recuperado el 15 de Abril del 2018 de http://sisbib.unmsm.edu.pe/BibVirtual/Publicaciones/risi/2010_n2/v7n2/a07v7n2.pdf

Moreno, F. (2000). Introducción a la OPP. *Grupo EIDOS*, 8–16. Recuperado el 15 de Abril del 2018 de https://kataix.umag.cl/~ruribe/Utilidades/Introduccion a la Programacion Orientada a Objetos.pdf

Node.js. (2018). About | Node.js. Recuperado el 2 de Abril del 2018 de https://nodejs.org/en/about/

OMS. (2013). Inactividad física: un problema de salud pública mundial. Recuperado el 16 de Abril del 2018 de http://www.who.int/dietphysicalactivity/factsheet_inactivity/es/

OMS. (2016). OMS / Actividad física. *Oms*. World Health Organization. Recuperado el 15 de Abril del 2018 de http://www.who.int/mediacentre/factsheets/fs385/es/

OMS. (2017). Agenda de Salud Sostenible para las Américas 2018-2030: Un llamado a la acción para la saluld y el bienestar en la Región de las Américas. In *160.ª Sesión del Comité Ejecutivo de la OMS para las Américas; del 26 al 30 de junio del 2017 (documento CE160/14, Rev. 1)* (p. 48). Recuperado el 15 de Abril del 2018 de http://iris.paho.org/xmlui/bitstream/handle/123456789/34194/CE160-14-s.pdf?sequence=2&isAllowed=y

Pacer Heath. (2018). Podómetro y Entrenador de Peso - Aplicaciones en Google Play. Recuperado el 16 de Mayo del 2018 de https://play.google.com/store/apps/details?id=cc.pacer.androidapp

PHP Group. (2018). PHP: ¿Qué es PHP? - Manual. Recuperado el 2 de Abril del 2018 de http://php.net/manual/es/intro-whatis.php

Proyectos Agiles. (2018). Reunión diaria de sincronización del equipo (Scrum daily meeting) – Proyectos Ágiles. Recuperado el 21 de Marzo del 2018 de https://proyectosagiles.org/reunion-diaria-de-sincronizacion-scrum-daily-meeting/

Runastic. (2018). Runtastic: entrenador personal de correr y caminar - Aplicaciones en Google Play. Recuperado el 2 de Mayo del 2018 de https://play.google.com/store/apps/details?id=com.runtastic.android&hl=es

SEDICI. (2011). Websocket: Comparación de performance e implementación de aplicaciones web. Recuperado el 15 de Abril del 2018 de http://sedici.unlp.edu.ar/bitstream/handle/10915/47014/Documento_completo__.pdf?sequence=1

Sheiko, D. (2012). WebSockets vs Server-Sent Events vs Long-polling. Recuperado el 11 de Junio del 2018 de http://dsheiko.com/weblog/websockets-vs-sse-vs-long-polling/

SIGTE. (2008). Uso de Comet (Reverse AJAX) en los SIG. Prototipo de SIG colaborativo. In *II Jornadas de SIG Libre* (p. 6). Recuperado el 15 de Abril del 2018 de http://www.sigte.udg.edu/jornadassiglibre2008/uploads/file/Comunicaciones/2.pdf

The Orange Grove. (2008). Principles of Object-Oriented Programming, 5–9. Recuperado el 15 de Abril del 2018 de https://florida.theorangegrove.org/og/file/33bf62f3-8ad1-7dde-e23f-6f17aca953c7/1/ooprogramming.pdf

Universidad de Sevilla. (2018). ¿Qué es un framework web?. Recuperado el 15 de Abril del 2018 de http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf

Vorrink, S. (2016). *eHealth to stimulate physical activity in patients with chronic obstructive pulmonary disease*. Recuperado el 15 de Abril del 2018 de http://dspace.library.uu.nl/handle/1874/331131

Wordpress. (2011). Kanban. Recuperado el 15 de Abril del 2018 de https://articulosit.files.wordpress.com/2011/11/kanban.pdf

World Colleges Information. (2010). ORM – Object Relational Mapping, 1–30. Recuperado el 15 de Abril del 2018 de http://cdn.worldcolleges.info/sites/default/files/Orm.pdf

ANEXOS

ABREVIATURAS

API: Application Programming Interface.

CEO: Chief Executive Officer.

CPU: Central Processing Unit.

CRUD: Create Update Read Delete.

DBMS: Data Base Management System.

ENT: Enfermedades no transmisibles.

GPS: Global Positioning System.

HTML: HyperText Markup Language.

HTTP: Hypertext Transfer Protocol.

ICEDEG: International Conference on eDemocracy & eGoverment.

IDE: Integrated Development Environment.

IEEE: Institute of Electrical and Electronics Engineers.

IETF: Internet Engineering Task Force.

JDK: Java Development Kit.

JIT: Just In Time.

JRE: Java Runtime Enviroment.

MVC: Modelo, Vista, Controlador.

OMS: Organización Mundial de la Salud.

ONU: Organización de las Naciones Unidas.

ORM: Object-Relational Mapping.

POO: Programación Orientada a Objetos.

SDK: Software Development Kit.

TIR: Tasa Interana de Retorno.

TIC: Tecnologías de la Información y Comunicación.

VAN: Valor Actual Neto.

W3C: World Wide Web Consortium.