Facultad de Ingeniería
Escuela de Ingeniería Civil en Informática

DESARROLLO DE UNA PLATAFORMA DE MONITOREO DE CAÍDAS EN DOMICILIO PARA ADULTOS MAYORES

Por

Sebastián Cristopher Andrés Murray Toledo

Trabajo realizado para optar al Título de
INGENIERO EN INFORMÁTICA
Prof. Guía: Carla Taramasco T.
Abril 2020

Certifico que he leído este documento y que, en mi opinión, es adecuado en ámbito y calidad como trabajo para optar al título de Ingeniero en Informática.

Carla Taramasco T. Profesor Guía

Certifico que he leído este documento y que, en mi opinión, es adecuado en ámbito y calidad como trabajo para optar al título de Ingeniero en Informática.

Marta Barría Profesor Informante

Certifico que he leído este documento y que, en mi opinión, es adecuado en ámbito y calidad como trabajo para optar al título de Ingeniero en Informática.

Cristian Carrión Profesor Informante

Aprobado por la Escuela de Ingeniería Civil en Informática, UNIVERSIDAD DE VALPARAÍSO.

Resumen

Las caídas son un gran problema para la población conformada por adultos mayores. La ayuda inmediata podría reducir el riesgo de varias complicaciones posteriores, incluyendo la muerte. Además, la existencia de herramientas que ayuden a detectar caídas no están orientadas a los adultos mayores o bien tienen un alto costo de adquisición. Los estudios indican que para detectar caídas se pueden utilizar sensores, como el acelerómetro, incluidos dentro de dispositivos vestibles. Con estos dispositivos se pueden desarrollar sistemas que puedan ser de ayuda con la problemática de las caídas que existe para los adultos mayores.

Por esto surge la necesidad de desarrollar un sistema que detecte caídas, utilizando un dispositivo vestible de bajo costo y desarrollando un aplicación móvil que pueda permitir enviar alertas a los cuidadores de los adultos mayores cuando ellos tengan una caída.

El sistema desarrollado obtiene los datos enviados desde un dispositivo vestible (que cuenta con un acelerómetro) y luego a través de la implementación de un algoritmo, que utiliza el método de umbrales, se analizan estos datos y se valida si existe o no una caída. Luego en una aplicación móvil el cuidador podrá ver un histórico de las caídas de los adultos mayores y también recibir notificaciones de las cuando se detecte una caída.

Según las pruebas realizadas el algoritmo alcanzo una sensibilidad del 93,75 % y especificidad del 100 %.

Con este trabajo se busca disminuir el tiempo que existe entre el momento que un adulto mayor tiene una caída y la llegada de ayuda.

Palabras claves: detección de caídas, adulto mayor, dispositivos vestibles, sensores, acelerómetro, umbrales.

Agradecimientos

Desde que entré a la universidad he tenido momentos llenos de emociones, para bien y para mal. Fueron años muy diversos en los que comencé siendo una persona que a pesar de estar siempre interesada en aprender se distraía mucho con cualquier otra cosa. Ese fue uno de los principales motivos por los cuales pasé por mis primera dificultades en la universidad, darle prioridades a otras cosas que a cumplir con los estudios. Pese a ello, agradezco enormemente esa experiencia ya que me permitió encontrar uno de mis primeros puntos de inflexiones en mi vida, lo cual me ayudo bastante a convertirme en lo que soy ahora.

Comenzaré agradeciendo todos mis seres queridos, los cuales siempre estuvieron presentes para ofrecerme ayuda en los momentos que la necesitaba, a mi familia, la cual siempre me apoyo y se preocuparon, a su manera, cuando necesitaba alguna ayuda. en especial a mi abuela que es como mi segunda mamá, la cual nunca ha dejado de preocuparse por mí, lo que agradezco bastante.

Agradezco a la Escuela de Ingeniería Civil Informática, por haberme entregado las bases para tener el conocimiento que tengo ahora, y que de una u otra forma me apoyaron. A los profesores, por lograr hacer que uno tenga esa confianza que se formó con ellos, de llegar a poder a comunicarnos de igual a igual, cosa que no se ve en cualquier otra universidad. A la secretaría, por la paciencia y el aguante que tuvo todas las veces que iba a preguntarle sobre las dudas que tenía y por ayudarme a solucionarlas, especialmente durante el proceso del trabajo de título. En fin a todas las personas y docentes que me ayudaron de alguna manera durante estos años.

A todos mis amigos, que de alguna u otra forma me apoyaron para que terminara este proceso y que no lo dejara a medias, en el ultimo tiempo destaco bastante la ayuda de varios, lo que me motivaron a terminar todo esto. También, a todas las grandes amistades que conocí en la escuela, de los cuales algunos han llegado a convertirse en muy grandes amigos. Todos esos momentos que viví con ellos no se me olvidaran jamás, sin ellos todos estos años, definitivamente, no hubiesen sido lo mismo. Todas esas experiencias dentro de la universidad y fuera de la universidad me ayudaron a ser lo que soy hoy.

En fin, fueron diversos años y agradezco haber vivido todos ellos, me ayudaron a crecer como persona y como profesional, muchas gracias.

Índice general

Resumen III

Agradecimientos IV

1. Introducción 1

2. Marco Conceptual y Estado del Arte 4
2.1. Marco Conceptual 4
2.1.1. Dispositivos Vestibles 4
2.1.2. Sensores 5
2.1.3. Acelerómetro 5
2.1.4. Etapas en una caída 6
2.2. Estado del Arte 7
2.2.1. Trabajos realizados 7

3. Definición del Problema y Análisis 14
3.1. Formulación del Problema 14
3.2. Solución Propuesta 15
3.2.1. Importancia del trabajo 16
3.3. Objetivos 17
3.3.1. Objetivo General 17
3.3.2. Objetivos Específicos 17
3.4. Metodología 18
3.5. Especificación de Requerimientos 19
3.5.1. Requerimientos Funcionales 19
3.5.2. Requerimientos No Funcionales 19
3.6. Funcionalidades del Sistema 20
3.6.1. Diagramas de Casos de Uso 20
3.6.2. Casos de Uso 21
3.6.3. Casos de Uso expandidos 21
3.6.4. Diagramas de Secuencia 23

3.6.5. Diagramas de Estado 24
3.6.6. Modelo Conceptual 24

4. Diseño **26**
4.1. Diseño Arquitectónico 26
4.1.1. Tecnologías utilizadas 26
4.1.2. Flujo de datos 27
4.2. Diseño Lógico 28
4.2.1. Diagrama de despliegue 28
4.2.2. Diagrama de componentes 28
4.2.3. Diagrama de paquetes 29
4.2.4. Diagrama de clases 29
4.3. Diseño de Datos 30
4.3.1. Diagrama Entidad Relación 31
4.3.2. Diagrama Relacional 32
4.3.3. Diccionario de Datos 32
4.4. Diseño del Algoritmo 33
4.5. Diseño de Interfaz 36
4.5.1. Arquitectura de la Información 36
4.5.2. Interfaces Gráficas 36
4.6. Diseño de Pruebas 38
4.6.1. Pruebas Unitarias 38
4.6.2. Pruebas de Integración 38
4.6.3. Pruebas de sistema 40
4.6.4. Pruebas con Usuarios 41
4.6.4.1. Prueba de satisfacción de la aplicación 41
4.6.4.2. Pruebas de Rendimiento 41
4.6.5. Pruebas de Aceptación 43

5. Implementación **45**
5.1. Hardware utilizado 45
5.2. Software utilizado 46
5.3. Lenguajes de programación 46
5.4. Estrategia de implementación 46
5.5. Interfaces 47

6. Pruebas y análisis de resultados **50**
6.1. Pruebas Unitarias 50
6.1.1. Resultados pruebas unitarias 52
6.2. Pruebas de Integración 52
6.2.1. Resultados pruebas de integración 53

6.3. Pruebas de Sistema . . . 53
6.3.1. Resultados pruebas de sistema . . . 54
6.4. Pruebas con Usuarios . . . 55
6.4.1. Prueba de satisfacción de la aplicación . . . 55
6.4.1.1. Resultados de la satisfacción de la aplicación . . . 55
6.4.2. Pruebas de rendimiento . . . 55
6.4.2.1. Resultados pruebas de rendimiento . . . 57
6.5. Pruebas de Aceptación . . . 57
6.5.1. Resultados pruebas de aceptación . . . 58

7. Implantación **59**
7.1. Requerimientos . . . 59
7.2. Preparación de Ambiente . . . 60
7.2.1. Servidor . . . 60
7.2.2. Raspberry Pi . . . 61
7.2.3. Aplicación móvil . . . 65

8. Conclusiones **67**
8.1. Trabajo a futuro . . . 68

A. Manual de Usuario **69**
A.1. Aplicación móvil . . . 69
A.2. Detector de caídas . . . 78

B. Documentación de desarrollador **83**
B.1. Preparación del ambiente en raspberry pi . . . 83
B.2. Implementación de notificaciones push . . . 84
B.3. API . . . 85
B.4. Contacto para el desarrollador . . . 87

Bibliografía **88**

Índice de tablas

2.1. Tabla comparativa sobre el estado del arte. 13

3.1. Requerimientos Funcionales 19
3.2. Requerimientos No Funcionales 19
3.3. Funcionalidades del Sistema 20
3.4. Descripción casos de uso. 21
3.5. Caso de uso expandidos: Detectar caída. 22
3.6. Caso de uso expandidos: Ver registro de caídas. 22

4.1. Diccionario de Datos 32
4.2. Pruebas Unitarias 38
4.3. Pruebas de Integración 40
4.4. Pruebas de Sistema 40
4.5. Pruebas con usuarios: Satisfacción de la aplicación 41
4.6. Satisfacción de la aplicación: Niveles de apreciación 41
4.7. Pruebas de Rendimiento: Especificidad 42
4.8. Pruebas de Rendimiento: Sensibilidad 43
4.9. Pruebas de Aceptación 44

6.1. Prueba Unitaria 01 - Transmisión de datos 50
6.2. Prueba Unitaria 02 - Análisis de Umbrales 51
6.3. Prueba Unitaria 03 - Detectar una caída 51
6.4. Prueba Unitaria 04 - Ver registro de caída 51
6.5. Prueba Unitaria 05 - Recibir notificación 52
6.6. Resultado Pruebas unitarias 52
6.7. Pruebas de integración 53
6.8. Resultado Pruebas de integración 53
6.9. Pruebas de Sistema 54
6.10. Resultado Pruebas de sistema 54
6.11. Satisfacción de la aplicación: Niveles de apreciación 55
6.12. Satisfacción de la aplicación 55
6.13. Pruebas de Rendimiento: Grupo de usuarios 56

6.14. Pruebas de Rendimiento: Resultado especificidad 56
6.15. Pruebas de Rendimiento: Resultado sensibilidad 57
6.16. Pruebas de Aceptación: Validación Profesora Carla Taramasco T. 58
6.17. Resultado Pruebas de aceptación 58

7.1. Características del servidor 60
7.2. Características de la Raspberry Pi 60
7.3. Requisitos aplicación móvil 60

Índice de figuras

2.1. Arquitectura del sistema [1]. 8
2.2. Situación de uso de la herramienta [1]. 8
2.3. En esta pantalla se pregunta al usuario si está bien o no [2]. 9
2.4. Si responde No, en esta pantalla se le preguntara qué hacer [2]. 9
2.5. Arquitectura general del sistema [3]. 10
2.6. Arquitectura básica del sistema [4]. 11
2.7. Esquema de detección combinada de acelerómetro y giroscopio [5]. 12

3.1. Diagrama de alto nivel de la solución 16
3.2. Diagrama Metodología 18
3.3. Diagrama de casos de uso. 20
3.4. Diagrama de secuencia: Detectar caída. 23
3.5. Diagrama de secuencia: Ver registro de caídas. 23
3.6. Diagrama de estados: Detectar caída. 24
3.7. Diagrama de estados: Ver registro de caídas. 24
3.8. Modelo Conceptual. 25

4.1. Diagrama arquitectónico 27
4.2. Diagrama de despliegue 28
4.3. Diagrama de componentes 29
4.4. Diagrama de paquetes 29
4.5. Diagrama de clases 30
4.6. Diagrama modelo entidad relación 31
4.7. Diagrama modelo relacional 32
4.8. Pseudocódigo del algoritmo detección de caídas 34
4.9. Diagrama algoritmo detección de caídas 35
4.10. Arquitectura de la Información 36
4.11. Interfaz iniciar sesión 37
4.12. Interfaz de registro 37
4.13. Interfaz de adultos 37
4.14. Interfaz de caídas 37

4.15. Pruebas de Integración 39

5.1. Interfaz Registro. 47
5.2. Registro fallido. 47
5.3. Interfaz ingreso. 48
5.4. Ingreso fallido. 48
5.5. Lista con adultos. 49
5.6. Aviso al agregar adulto. 49
5.7. Ingreso de adulto fallido. 49
5.8. Lista de caídas. 49

7.1. Acceso al servidor con el usuario creado 60
7.2. Iniciación de sesión tmux 61
7.3. Ejecución del web server en tmux 61
7.4. Cerrar sesión tmux manteniendo la ejecución del web server 61
7.5. Configuraciones iniciales en Raspbian 62
7.6. Actualización de Raspbian 62
7.7. Comprobación de instalación python3 y pip3 62
7.8. Descarga e instalación de compilación ARM de pylsl 63
7.9. Instalación programa detector de caídas 63
7.10. Identificación de Faros 180° 64
7.11. Faros 180° pareado con Raspberry Pi 64
7.12. Ejecución de programa implantado en Raspberry Pi 65
7.13. Implantación en Raspberry 65
7.14. Build y firma de la Apk 66
7.15. Aplicación en la plataforma de Google Play 66

A.1. Búsqueda de aplicación en Google Play 70
A.2. Instalación de aplicación en Google Play 70
A.3. Icono de la aplicación en el dispositivo 71
A.4. Pantalla inicio de sesión 72
A.5. Pantalla de registro 73
A.6. Acceso a administración de adultos mayores 74
A.7. Agregar un adulto mayor 75
A.8. Aviso de que ha agregado a un adulto mayor 75
A.9. Registro de caídas de sus adultos mayores 76
A.10. Notificaciones cuando un adulto mayor tiene una caída 76
A.11. Eliminar un adulto mayor 77
A.12. Esquema conexión raspberry pi 78
A.13. Símbolos e indicadores Faros 180° 79
A.14. Parear Faros 180° con raspberry pi 80

A.15. Acceder a consola en raspberry pi 80
A.16. Escanear Faros 180° 81
A.17. Detección de caídas 82

Capítulo 1

Introducción

Los avances en medicina permiten que las personas vivan más años en comparación con las generaciones anteriores; en todo el mundo había 901 millones de personas de 60 años o más en el año 2015, un aumento del 48 % respecto de los 607 millones de personas que había de esa misma edad en todo el mundo en el año 2000. Para el año 2030, se proyecta que en el mundo la cantidad de personas de 60 años o más crezca en un 56 %, hasta alcanzar 1.400 millones; y para el año 2050, se proyecta que la población mundial de personas mayores a 60 años sea más del doble del tamaño de 2015, esto es casi 2.100 millones [6]. Esto ha llevado a la necesidad de desarrollar aplicaciones inteligentes para el cuidado de la salud de los adultos mayores generándoles una gran oportunidad con el fin de que ellos puedan tener una vida diaria independiente y segura.

El monitoreo de las actividades diarias puede mejorar el bienestar y detectar enfermedades graves con el tiempo. Si un adulto mayor se encuentra con algún problema de salud es de mucha importancia que sea monitorizado por un sistema que pueda identificar cambios en su actividad diaria, especialmente para las personas mayores, que viven solas y sin ningún tipo de apoyo.

Las caídas son uno de los problemas de salud que afectan a muchos adultos mayores en el mundo. Según un informe de la Organización Mundial de la Salud (ONU), alrededor del 28-35 % de las personas mayores de 65 años y 32-42 % de las personas mayores de 70 años experimentan caídas cada año [7]. Sin ayuda inmediata, los adultos mayores pueden sufrir dolor, angustia emocional o incluso desarrollar otras complicaciones, como neumonía, deshidratación e hipotermia, por lo tanto la ayuda después de una caída es de mucha importancia, ya que podría reducir el riesgo de complicaciones, muertes y aumentar considerablemente la probabilidad de volver a tener una vida independiente [8].

Actualmente, existen dispositivos inteligentes (smartwatch, smartphones, etc.) equipados con sensores (acelerómetros, giroscopios, etc.) que, conectados a través de bluetooth (u otros mecanismos de transmisión de datos) con otros dispositivos, sirven para poder procesar capturas de datos de los movimientos que realizan los usuarios con el fin de

alertar caídas [9]. Además, se toma en cuenta que muchos de estos dispositivos como los smartwatchs cuentan con un SDK (Software Development Kit) abierto para poder trabajar con ellos.

Grandes empresas ya han desarrollado dispositivos que puedan implementar estas ayudas, por ejemplo, el "Apple Watch Series 4" [10] incluye dentro de todas sus características funciones como frecuencia cardiaca, detección de caídas y emergencia SOS, lo que lo hace una potente tecnología dentro del ámbito, pero debido al costo y uso que se le da dentro del mercado, lleva a que estos dispositivos no estén orientados a los adultos mayores.

Por otro lado, una de las características importantes de este tipo de desarrollo es disminuir el tiempo entre la caída de un adulto mayor y la llegada de ayuda, por lo que enviar una alerta cada vez que exista este evento sería lo ideal para lograrlo.

Debido a este contexto se propone desarrollar un sistema que detecte caídas, utilizando un vestible, y una aplicación móvil que de acuerdo al adulto mayor que se está monitorizando pueda recibir notificaciones cada vez que el adulto mayor tenga una caída.

Este documento se estructura de la siguiente manera:

- **Capítulo 2:** Trata acerca del marco conceptual donde se definen conceptos utilizados en este trabajo y del estado del arte donde se muestra el estudio de los trabajos ya realizados en el tema, lo que sirvió para obtener una base científica como pilar para el desarrollo de este trabajo.
- **Capítulo 3:** Se define el problema y situación actual. En base a esto, se propuso una solución, planteando objetivos y la metodología utilizada. También se especifican requerimientos y funcionalidades del sistema incluyendo respectivos diagramas para los casos de uso planteados.
- **Capítulo 4:** Se exponen los diseños de la solución tanto de arquitectura e interfaces como también el lógico, de datos, el diseño del algoritmo usado para detectar las caídas y el diseño del plan de pruebas.
- **Capítulo 5:** Se detalla la implementación de la solución. Es aquí donde son presentadas las tecnologías seleccionadas y usadas en el desarrollo de la solución, por otra parte se muestran las interfaces con su respectiva funcionalidad.
- **Capítulo 6:** Presenta los resultados de las pruebas realizadas en el sistema.

- **Capítulo 7:** Se muestra como fue el proceso de implantación de todo lo que fue implementado anteriormente. Además, se presenta la documentación necesaria para un usuario y un desarrollador que desee utilizar el sistema.
- **Capítulo 8:** Se exponen las conclusiones de este trabajo de título, indicando los resultados obtenidos, sus limitantes, los trabajos a futuros y opiniones sobre el tema que se abordo.
- **Bibliografía:** Se encuentran todas las referencias utilizadas en este trabajo.

Capítulo 2

Marco Conceptual y Estado del Arte

Considerando que este trabajo trata sobre el uso de dispositivos vestibles para detectar caídas en adultos mayores, es conveniente familiarizarse con algunos términos básicos que tienen que ver con todo este tema. En la sección del marco conceptual se presentan conceptos que permitirán tener una mejor comprensión acerca de este trabajo de título, abordando y explicando conceptos claves como que son los dispositivos vestibles, que es un sensor y de que manera se puede detectar una caída. También, en la sección del estado del arte se darán a conocer métodos y técnicas de detecciones de caídas utilizados en investigaciones, como también las aplicaciones que se realizaron en ellas, lo que ayudo para posteriormente implementar una solución basada en lo investigado en esta sección.

2.1. Marco Conceptual

2.1.1. Dispositivos Vestibles

El término dispositivo vestible, también conocido como Wearables, se refiere a las tecnologías que pueden incorporarse a prendas de vestir o a accesorios y usarse en el cuerpo [11]. Estos dispositivos pueden llegar a realizar (no siempre de igual manera) algunas de las funciones que tienen los teléfonos móviles y computadoras portátiles. Los vestibles tienden a ser una tecnología más sofisticada que otras ya que hoy en día pueden proporcionar funciones sensoriales que no se ven normalmente en dispositivos móviles [11].

En general, los dispositivos vestibles tienen un tipo de comunicación que permite al usuario acceder a la información en tiempo real o a datos ya guardados en él de forma local. De manera que si el dispositivo esta siendo portado en el cuerpo, este cumplirá con crear un acceso constante a la información obtenida, logrando así un método conveniente y portátil a la hora de analizar información.

Las implicaciones y los usos que tienen los dispositivos vestibles son de gran alcance y pueden influir en los campos de la medicina, la condición física, el envejecimiento, etc. El objetivo de las tecnologías portátiles en cada uno de estos campos será incorporar sin problemas la electrónica y las computadoras portátiles y en la vida diaria de las personas [11].

2.1.2. Sensores

Básicamente, un sensor es un dispositivo electrónico que detecta y responde a algún tipo de entrada del entorno físico. La entrada puede ser de luz, calor, movimiento, humedad, presión, etc. La salida es generalmente una señal legible para los humanos en la misma ubicación del sensor o transmitida electrónicamente a través de una red para su posterior lectura [12].

Los sensores de movimiento inercial (un sensor inercial es aquel que utiliza el principio de la inercia para lograr sus mediciones [13]) son los sensores más comúnmente utilizados para evaluar el movimiento corporal. Su tamaño pequeño les permite incluirse en dispositivos vestibles. Estos pueden detectar el balanceo postural al medir la aceleración lineal, la velocidad angular y la dirección de los movimientos del cuerpo.

Entre las investigaciones que se verán en la sección 2.2 están aquellas que para poder detectar caídas utilizan un dispositivo vestible conformado principalmente por sensores inerciales, siendo los mas utilizados los acelerómetros, giroscopios y magnetómetros.

2.1.3. Acelerómetro

El funcionamiento de un acelerómetro consiste en la comparación del movimiento entre una masa suspendida dentro de este y una masa fijada al objeto en movimiento. La masa fija se mueve con el objeto en movimiento, pero la respuesta de la masa suspendida se retrasa debido a su inercia o resistencia al cambio. La diferencia de movimiento entre estas dos masas es dependiente de la aceleración, por lo tanto, se puede medir la aceleración del objeto en movimiento [14]. Si el acelerómetro tiene tres ejes (X, Y y Z), quiere decir que tiene una sola masa estará suspendida y tres masas independientes, fijadas cada una a su respectivo eje. De esta manera se puede medir la aceleración individual de cada eje.

Las aceleraciones según el sistema internacional de unidades [15] se miden en metros por segundos al cuadrado $\frac{m}{s^2}$. También se pueden medir en gravedades g , donde

cada gravedad representa la aceleración que experimenta un cuerpo debido a la fuerza gravitacional de la tierra, por definición $g = 9{,}82\frac{m}{s^2}$.

2.1.4. Etapas en una caída

Para poder definir un algoritmo que detecte si existe o no una caída es necesario definir las etapas que el cuerpo experimenta al tener una. Estas etapas vendrían siendo tres: caída libre, impacto y finalmente reposo. Cada una de las etapas generaría un valor de aceleración diferente, siendo mas bajo durante la caída libre, mas alto al momento del impacto y un valor casi estático durante el estado de reposo [16].

Las etapas se caracterizan por:

1. Caída Libre: Esta etapa hace referencia al momento en que inicia una caída, donde el cuerpo se dirige hacia el suelo experimentando una aceleración vertical similar a la gravedad. En esta etapa la suma vectorial de las aceleraciones disminuye aproximándose a 0 g.

2. Impacto: Luego de la caída libre, el cuerpo choca con el suelo u otra plataforma, dando como respuesta un elevación en la suma de los tres ejes debido a la elevada desaceleración del cuerpo.

3. Reposo: Tras una caída, el cuerpo se mantiene en reposo durante un tiempo. Este tiempo puede ser muy elevado si el paciente se encuentra inconsciente. Además, queda en una posición diferente a la inicial lo que provoca un cambio en la aceleración de cada eje, especialmente el vertical y el horizontal.

2.2. Estado del Arte

Los sistemas de detección de caídas basados en vestibles utilizan los datos obtenidos de los sensores, en su mayoría acelerómetros, que se encuentran incorporados en los dispositivos que el usuario estaría portando. Esta sección explorará la investigación actual sobre sistemas de detecciones de caídas basados en dispositivos vestibles.

2.2.1. Trabajos realizados

Las aplicaciones que utilizan dispositivos vestibles o inteligentes para detectar caídas, detectan las actividades mediante la utilización de sensores MEMS (Micro Electro Mechanical System) de bajo costo como acelerómetro, giroscopio, brújula, magnetómetro, luz de proximidad y presión, permitiendo aplicar diferentes algoritmos para aumentar la precisión de los datos requeridos [17].

En [1] se propuso un sistema de detección y rescate de accidentes por caídas para adultos mayores. Este se basa en utilizar el acelerómetro triaxial y una brújula incorporados en un dispositivo inteligente para la detección y las redes de tercera generación (3G) para el rescate. Como se puede ver en la Figura 2.1, se puede observar la arquitectura del sistema propuesto que se compone principalmente de tres bloques: el detector de caídas basado en dispositivos inteligentes, el centro de coordinación, y el centro de rescate. Además, como se aprecia en la Figura 2.2, el sistema puede localizar al usuario y comunicarse con el centro de coordinación dentro del área siempre que la red 3G esté disponible. También, se evaluaron diferentes actividades que incluyen un evento de caída, como; correr, caminar, sentarse, etc. El rendimiento del sistema puede alcanzar hasta un 92 % de sensibilidad[1] y un 99.75 % de especificidad[2].

[1] La proporción de positivos reales que se identifican correctamente como tales, en este caso, cuando existe una caída y el sistema identifica que ésta existió.

[2] La proporción de negativos reales que se identifican correctamente como tales, en este caso, cuando no existe una caída y el sistema identifica que ésta no existió.

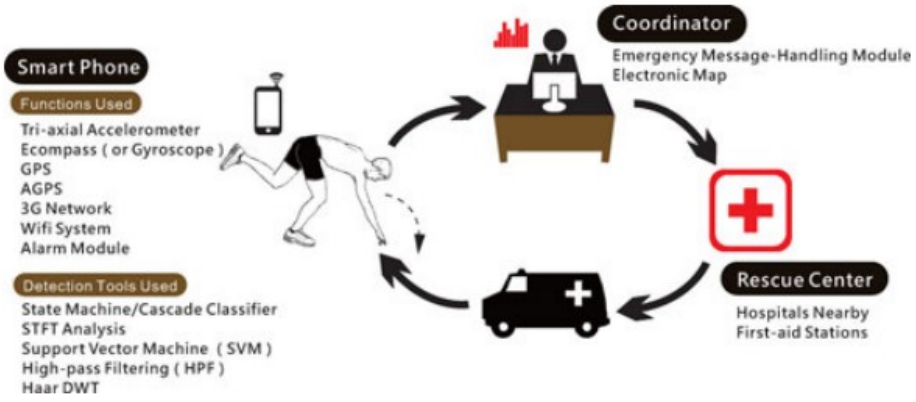

Figura 2.1: Arquitectura del sistema [1].

Figura 2.2: Situación de uso de la herramienta [1].

En [2], se diseñó una aplicación de detección de caídas con un smartphone la cual puede ejecutarse en cualquier dispositivo Android que tenga un acelerómetro y capacidad para realizar llamadas y enviar SMS. Se detecta una caída utilizando un solo umbral y para limitar las falsas alarmas, se aplicó una pantalla adicional (ver Figura 2.3) para preguntar si el usuario está bien o no después de detectar una caída (ver Figura 2.4). Midieron el rendimiento de la detección en términos de falso negativo[3] y falso positivo[4]. En general,

[3]El falso negativo ocurre cuando se produce una caída, pero el dispositivo no emite una alarma.
[4]El falso positivo ocurre cuando el dispositivo emite una alarma pero no ocurre.

cuanto más bajos sean los falsos negativos y positivos el rendimiento será mejor. El experimento lo realizaron cinco personas, cada una de ellas cinco veces. Los resultados muestran que el sistema ha perdido 9 de 25 caídas, y hay 7 de 25 tiempos de salto que se identificaron como caídas.

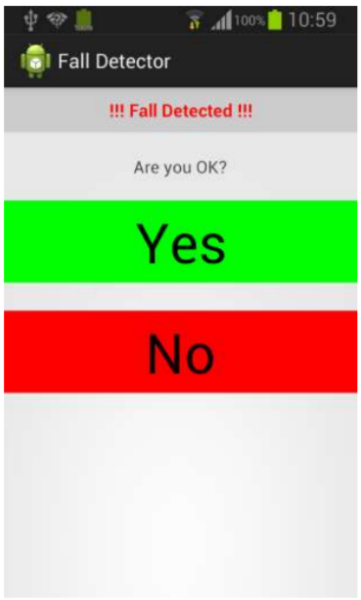

Figura 2.3: En esta pantalla se pregunta al usuario si está bien o no [2].

Figura 2.4: Si responde No, en esta pantalla se le preguntara qué hacer [2].

Por otro lado, continuando con los dispositivos inteligentes, también existen aplicaciones que utilizan smartwatches para poder detectar caídas [3, 18]. En [3] se presenta una aplicación enfocada para adultos mayores y personas con discapacidad que utiliza el reloj Pebble SmartWatch junto con un dispositivo Android y permite la captura, transmisión, almacenamiento y procesamiento eficientes de dichos datos de movimiento. En la Figura 2.5 se puede observar la arquitectura del sistema, en el cual se aprecia como la aplicación del smartwatch establece una conexión con el dispositivo Android, si tiene éxito, transmite los datos capturados mediante tecnología Bluetooth (BLE). Con respecto a la detección de caídas, se utilizó un algoritmo basado en un árbol de decisiones que determina una caída. Durante los experimentos, diez personas sanas estaban equipadas con el reloj inteligente Pebble y un smartphone que ejecutaba la aplicación. Estas últimas, realizaron una serie de tareas pre-etiquetadas como leves, moderadas o intensas y durmieron en sus rutinas de la vida diaria. La sensibilidad del algoritmo del árbol de decisiones fue de un 96.87 % y su

especificidad fue de un 99.30 % logrando una alta tasa de efectividad.

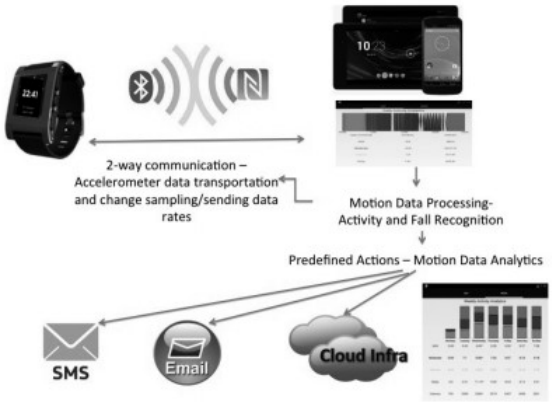

Figura 2.5: Arquitectura general del sistema [3].

En [18] se propone un sistema intuitivo y simple que detecta automáticamente cuando la persona se cae, utilizando el acelerómetro triaxial dentro de un smartwatch. Este sistema tiene varias funcionalidades, que incluyen: detección automática de caídas y un botón de SOS, y solo requiere solo una tarjeta SIM para funcionar. El usuario usa el smartwatch, en caso de tener una caída, se le notifica a un cuidador. La notificación se transmite llamando a un número predefinido y también notificando a un portal web a través de SMS. El portal web proporciona soporte y permite la comunicación con el smartwatch para la recepción y visualización de notificaciones enviadas. La evaluación del algoritmo de detección de caídas mostró que detecta todas las caídas y minimiza los falsos positivos, logrando un 85 % de precisión.

También en [4], utilizando un smartphone y un smartwatch (ambos provistos de un acelerómetro incorporado y un giroscopio), se evalúa y propone un sistema de detección de caídas que se beneficia de la detección realizada por los dos dispositivos inteligentes (ver Figura 2.6). Aquí, una aplicación específica en cada dispositivo rastrea y analiza permanentemente los movimientos del usuario. Se implementaron diversos algoritmos de detección de caídas en las aplicaciones desarrolladas de Android para discriminar las caídas de las actividades de la vida diaria del paciente. Lo que tiene este sistema es que solo supone que se ha producido una caída si se detecta de forma simultánea e independiente por los dos dispositivos. Los resultados obtenidos muestran que el uso conjunto de los dos dispositivos de detección aumenta la capacidad del sistema para evitar falsas alarmas mientras se mantiene la efectividad de las decisiones de detección.

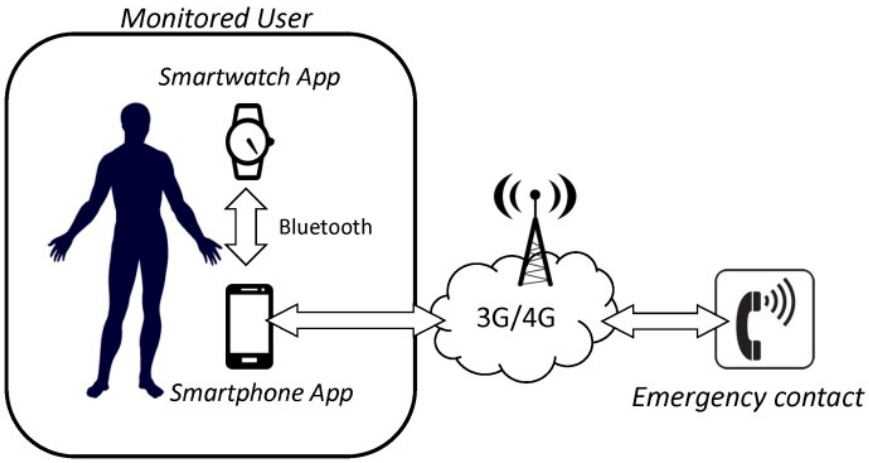

Figura 2.6: Arquitectura básica del sistema [4].

En [19] propusieron tres algoritmos de detección de caídas basados en datos proporcionados por un solo acelerómetro. Para detectar una caída, se tienen en cuenta los cambios de aceleración de dos o más de las fases que se producen durante una caída accidental, incluidas las fases de Inicio, Impacto, Consecuencias y Postura [20]. En el primer algoritmo se detecta una caída si se produce un posible impacto y después de eso se tiene una postura horizontal. Mientras que el segundo y tercer algoritmo detectan las fases de caída libre, impacto y reposo. Se obtuvo un resultado de 97.5 % de sensibilidad y un 100 % de especificidad.

En [21], determinaron los umbrales para los cambios rápidos en la señal de aceleración y la aceleración vertical para los algoritmos de detección de caídas utilizando datos recopilados por un solo acelerómetro. Se detecta una caída comparando uno de los parámetros anteriores con sus umbrales definidos y verificando la postura después de caer. Los resultados muestran que los algoritmos pueden alcanzar una alta sensibilidad y especificidad de hasta el 100 %. Sin embargo, los algoritmos utilizan tanto la comparación de umbrales como la detección de postura del cuerpo al momento de ser evaluados.

Los autores en [5] presentaron un algoritmo de detección de caídas basado en un sistema de sensor portátil compuesto por un acelerómetro de triaxial y un giroscopio. Los umbrales críticos para el vector de aceleración de suma total y la velocidad angular se utilizan para detectar una caída con sensibilidad y especificidad óptimas. En la Figura 2.7 se puede ver el algoritmo utilizado para la detección de caídas, en este se calcula y compara la aceleración con el valor menor de la aceleración en el umbral, si este es menor se comienza a medir el valor de la aceleración y la velocidad angular a partir de 0.5 segundos. Cuando los valores del acelerómetro y del giroscopio sean superiores a los

valores mayores del umbral de aceleración y velocidad angular se detectara una caída, en caso contrario el algoritmo volverá a empezar. Los experimentos se realizaron en 27 sujetos jóvenes y 9 sujetos de mediana edad con los sensores colocados en el pecho. Se realizaron movimientos como pararse, caminar, sentarse, pararse, caminar y correr y 4 caídas diferentes (hacia adelante, hacia atrás, a la derecha y a la izquierda). Los resultados mostraron que el algoritmo propuesto puede detectar caídas con una sensibilidad del 96.3 % y una especificidad del 96.2 %.

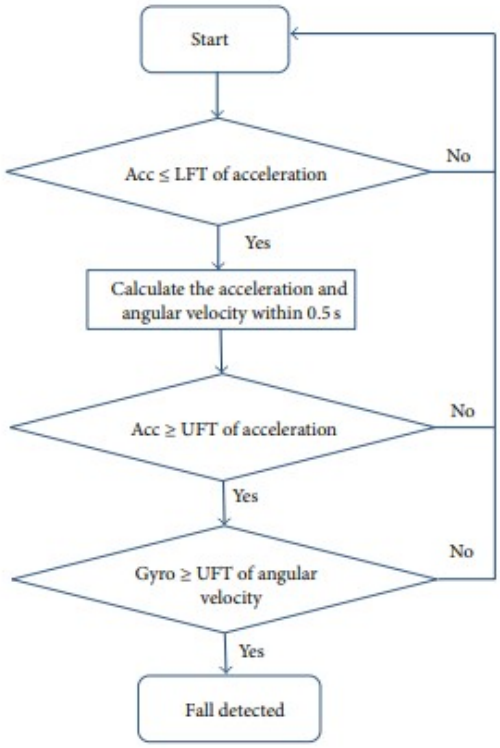

Figura 2.7: Esquema de detección combinada de acelerómetro y giroscopio [5].

En [22] se propone un algoritmo de detección de caídas que combina un método de umbral simple y el "Modelo oculto de Márkov (HMM)" utilizando la aceleración de 3 ejes. Para aplicar el algoritmo de detección de caídas propuesto, se diseñó y produjo un dispositivo de detección de caídas portátil. Se introdujeron varios parámetros de la aceleración de 3 ejes aplicadas a un método de umbral simple. Las posibles caídas se eligen a través del umbral y se aplican a dos tipos de HMM para distinguir entre una caída y una ADL. Se compararon y analizaron los resultados utilizando el umbral simple, HMM y la combinación del método simple y HMM. La combinación del método de umbral simple y HMM

redujo la complejidad del hardware, por lo tanto, el algoritmo propuesto mostró una mayor precisión teniendo como resultados 99.69 % de especificidad y 99.17 % de sensibilidad.

En [23] se tiene como objetivo detectar caídas basadas en rápidos cambios de aceleración utilizando el enfoque basado en umbrales y utilizando un solo acelerómetro. Se propuso un algoritmo de detección de caídas basado en el cambio de aceleración. Este observa y detecta el rápido cambio de la aceleración en el eje vertical y el valor promedio del vector de magnitud de la señal de la aceleración para diferenciar las caídas de las ADL. Los resultados iniciales demuestran que nuestro algoritmo logró un 100 % de sensibilidad, un 95.65 % de especificidad y un 96.35 % de precisión cuando se probó con un total de 44 caídas intencionales y 230 ADL en 32 conjuntos de datos.

En conclusión, los algoritmos basados en umbrales son fáciles de implementar y tienen un trabajo computacional mínimo, por lo tanto, son los preferidos en estos sistemas. En comparación los algoritmos basados en aprendizaje automático son más sofisticados, sin embargo, requieren altas habilidades matemáticas. Con respecto a los resultados, la mayoría de los trabajos mantienen un porcentaje de especificidad y sensibilidad arriba de un 85 % y aunque algunos trabajos no especifiquen (N.E.) los valores, sirvieron para tener una idea de hacia donde tener el enfoque.

	[1]	**[2]**	**[3]**	**[18]**	**[4]**	**[19]**	**[21]**	**[5]**	**[22]**	**[23]**
Utiliza acelerómetro	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Utiliza un dispositivo inteligente	✓	✓	✓	✓	✓	✗	✗	✗	✗	✗
Usa algoritmo de umbrales	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓
Especificidad	99.75	N.E.	99.30	70	85.8	100	N.E.	96.2	99.69	96.35
Sensibilidad	92	N.E.	96.87	90	85.8	97.5	N.E.	96.3	99.17	100

Tabla 2.1: Tabla comparativa sobre el estado del arte.

Capítulo 3

Definición del Problema y Análisis

En este capítulo se dará a conocer la problemática que aborda el tema de las caídas en los adultos mayores, junto con la solución propuesta. También se mencionara el objetivo general, los específicos, la metodología utilizada en este trabajo, y un análisis en términos de requerimientos y funcionalidades del sistema.

3.1. Formulación del Problema

Con los avances en la medicina las nuevas generaciones se consideran más longevas con respecto a las anteriores [6]. Debido a esto, la población adulta está más propensa a sufrir lesiones físicas, que pueden afectar de manera considerable su salud, puesto que no cuentan con un sistema que los ayude y que pueda informar a sus familiares y/o cuidadores en caso de algún problema o accidente. La caída es un riesgo que tienen los adultos mayores ya que si ellos se encuentran solos en algún sitio probablemente no tengan una manera de avisar sobre este incidente lo que provocaría que el adulto mayor pueda llegar a pasar una buena cantidad de tiempo sin recibir la ayuda necesaria. Sin la ayuda inmediata el daño que puede generar una caída en los adultos mayores es considerable tomando en cuenta el daño físico como recibir graves fracturas por el impacto de una caída, el daño emocional ya que al no poder ser autovalentes en sus actividades diarias puede que ellos se consideren una carga para su entorno e incluso se pueden llegar a desarrollar otras complicaciones como hipotermia e deshidratación, etc. [8]. Todo esto conlleva a que el desarrollo de tecnologías pueda ser utilizada para poder monitorizar el estado de salud, por ejemplo, de los adultos mayores.

Por otro lado, existen artículos que hablan sobre el reconocimiento de actividades y detecciones de caídas que utilizan la grabación y el procesamiento de sonidos para detectar situaciones de caídas, como también sensores de video vigilancia que detectan patrones inusuales en la trayectoria de movimiento del paciente brindando información sobre la

actividad del sujeto dentro de un entorno asistido [24, 25, 26]. Sin embargo, la mayoría de las personas no se sienten cómodas siendo monitoreadas por cámaras en su hogar ya que pueden suponer que actúan como una amenaza para su privacidad y por lo tanto consideran no utilizarlas. Por lo tanto, un método más preferible para recopilar información sobre la actividad del paciente es el uso de dispositivos que integran sensores como acelerómetros, giroscopios y sensores de contacto, ya que estos proporcionan un enfoque no invasivo para ellos lo cual habilita que puedan utilizarlas sin tener que pensar en que su privacidad será afectada.

Aplicaciones que cumplen estas funciones existen y cada vez van saliendo nuevas y más innovadoras en el mercado, pero no están a disposición de todos ya que incluyen un elevado costo de adquisición y además no son orientadas a un público de adultos mayores, como lo es el "Apple Watch Series 4" [10] que incluye entre todas sus características funciones que pueden detectar caídas [27].

3.2. Solución Propuesta

Se propone desarrollar un sistema de bajo costo utilizando un dispositivo vestible (con sensores inerciales, específicamente un acelerómetro) que, con ayuda de un algoritmo que implemente el método de umbrales (presentado en la sección 4.4) pueda detectar caídas. Además, desarrollar una aplicación que permita gestionar a los adultos mayores que porten el dispositivo vestible y que también se quiera tener constancia de sus caídas, recibiendo alertas de estas con la finalidad de disminuir el tiempo que transcurre entre una caída y la llegada de ayuda.

Utilizando los sensores que vienen incorporados en dispositivos vestibles, para este caso un acelerómetro, se pueden capturar los datos provenientes del sensor que servirán para utilizarlos en un algoritmo y así poder detectar una caída. En base a la revisión en el estado del arte, se decidió aplicar el método basado en umbrales, creando 3 etapas (caída libre, impacto y reposo) para posteriormente poder tener una mejor precisión al detectar una caída. También, se puede crear un historial basado en las detecciones de caídas, que servirá para poder tener una noción de como se encuentra el paciente dentro de un periodo de tiempo. Además, el uso de dispositivos vestibles es ideal para poder realizar esta labor ya que estos dispositivos son considerados como no invasivos, logrando que la privacidad de los adultos mayores no se vea afectada.

Como se puede ver en la Figura 3.1 se utiliza un dispositivo vestible que captura los datos transmitidos hacia el sistema por un adulto mayor, los que al ser analizados y pasados por cada etapa del umbral (caída libre, impacto y reposo), se detecta si existe una

caída o no. Al ser detectada esta caída, se registra en una base de datos con el fin de crear un histórico de las caídas que el adulto mayor ha tenido y también se envía una alerta hacia un cuidador, indicando que el adulto mayor acaba de tenerla.

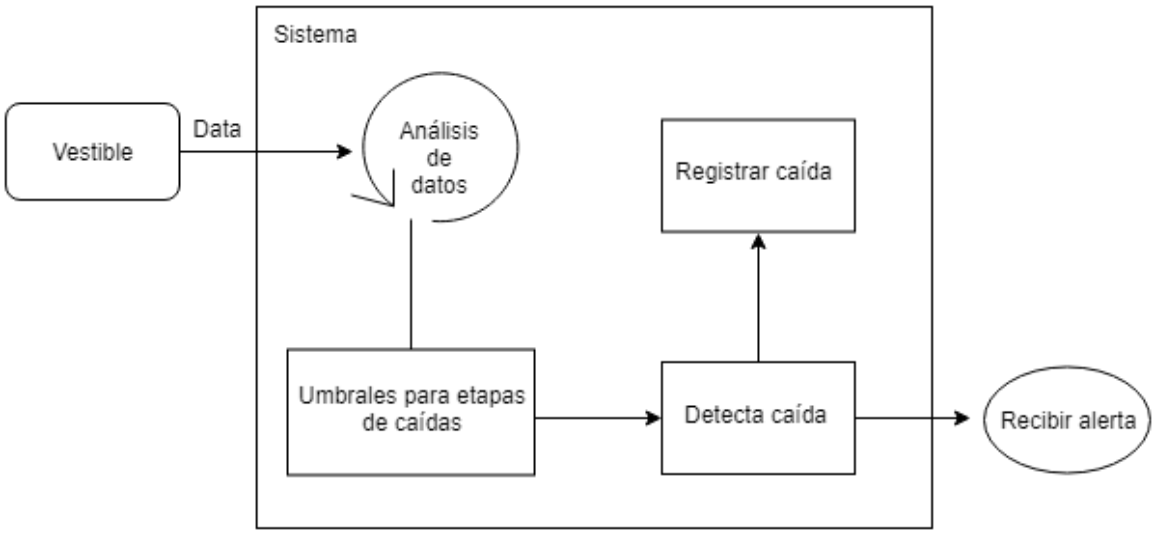

Figura 3.1: Diagrama de alto nivel de la solución

Por estos motivos, este trabajo de título lleva a cabo el desarrollo de una aplicación móvil que permita detectar caídas y generar una alerta a un cuidador con el fin de disminuir el tiempo que transcurre entre la caída y la ayuda del adulto mayor.

3.2.1. Importancia del trabajo

El presente trabajo tiene una importancia relevante ya que, como se mencionó anteriormente, si un adulto mayor tiene una caída puede repercutir en graves daños físicos y emocionales por lo que el poder detectarlas ayudará a disminuir el tiempo de auxilio hacia ellos y prevenir que pasen a tener mayores complicaciones. Además, el que los cuidadores puedan recibir una alerta cada vez que exista una caída permitirá que los adultos mayores tengan una mayor libertad a la hora de hacer sus actividades de la vida diaria ya no tendrán la necesidad de estar siempre al lado de su cuidador. Por otro lado, desarrollar un sistema que use dispositivos vestibles de bajo costo es un gran aporte para todos aquellos que no tienen los recursos para adquirir uno desarrollado por grandes empresas de la tecnología.

3.3. Objetivos

A continuación, se darán a conocer los objetivos que enmarcan este trabajo de título el cual se alcanzará mediante los objetivos específicos.

3.3.1. Objetivo General

Desarrollar una sistema para detectar caídas en adultos mayores, usando un dispositivo vestible.

3.3.2. Objetivos Específicos

- Implementar un algoritmo basado en el método de umbrales, para poder detectar las caídas, utilizando los datos recibidos desde el dispositivo vestible.
- Implementar una aplicación para los cuidadores que permita recibir alertas y visualizar el histórico de las caídas de los adultos que se quieren monitorizar .
- Evaluar la sensibilidad y especificidad del algoritmo desarrollado.

3.4. Metodología

El desarrollo de este trabajo de título se llevara a cabo utilizando una metodología con enfoque incremental [28]. Esta se compone del modelo en cascada, el cual ordena cada etapa de forma rigurosa, lo que ayuda a cumplir con los plazos y actividades definidas de este trabajo, y del desarrollo evolutivo, que busca exponer las implementaciones iniciales para así ir refinando a través de iteraciones las actividades que requieran mayor seguridad para llegar a un sistema mas adecuado.

En la Figura 3.2, se presenta el diagrama correspondiente a la metodología de este trabajo.

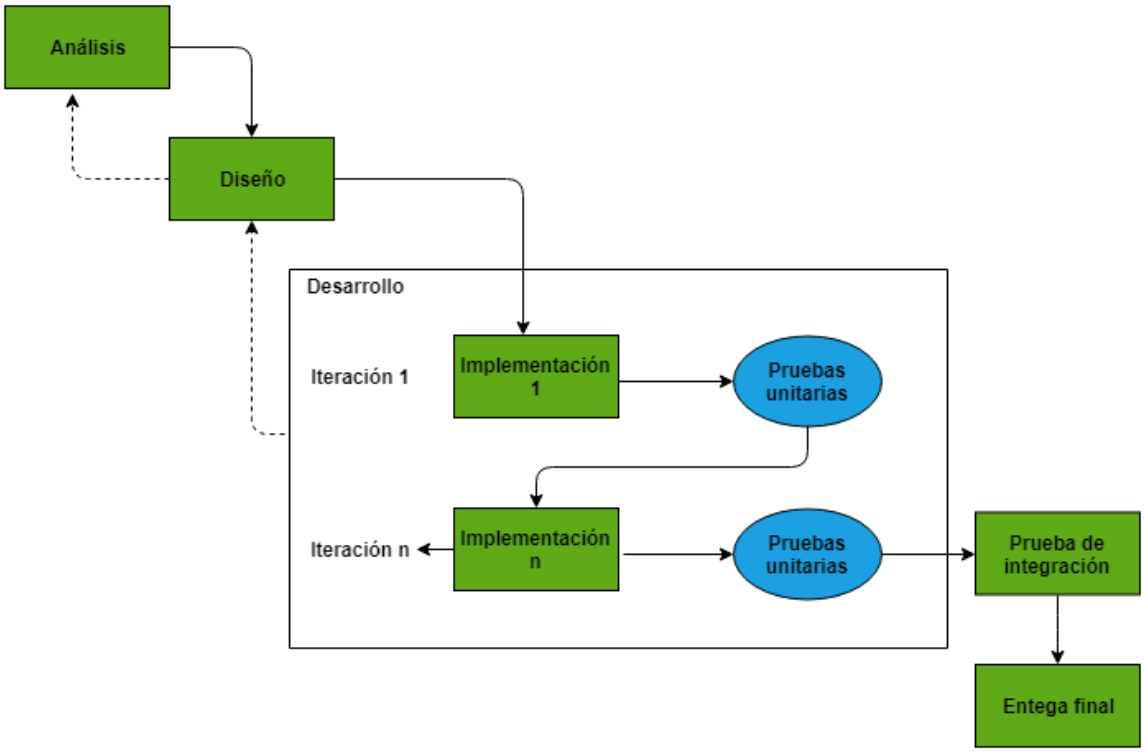

Figura 3.2: Diagrama Metodología

3.5. Especificación de Requerimientos

En esta sección se presentan los requerimientos que requerirá el sistema. Estos se dividen en:

- Requerimientos funcionales: son las funciones básicas que debe proporcionar el sistema.
- Requerimientos no funcionales: son las propiedades no funcionales, como la disponibilidad, el rendimiento, la seguridad, etc.

3.5.1. Requerimientos Funcionales

Los requerimientos funcionales se detallan en la Tabla 3.1.

ID	Requerimiento Funcional	Prioridad
RF01	El programa puede capturar datos del acelerómetro.	Alta
RF02	El programa tiene umbrales fijos para detectar cada etapa de una caída.	Alta
RF03	El programa permite detectar caídas.	Alta
RF04	El cuidador puede registrar a los adultos que quiere monitorear en la app móvil.	Alta
RF05	El cuidador puede ver registro de caídas por fecha de sus adultos monitoreados en la app móvil.	Alta
RF06	El cuidador puede recibir alertas de las caídas en la app móvil.	Alta

Tabla 3.1: Requerimientos Funcionales

3.5.2. Requerimientos No Funcionales

Los requerimientos no funcionales se detallan en la Tabla 3.2.

ID	Requerimiento No Funcional	Prioridad
RNF01	La aplicación móvil es sencilla de utilizar para los cuidadores.	Media
RNF02	El sistema está disponible siempre que se requiera.	Alta

Tabla 3.2: Requerimientos No Funcionales

3.6. Funcionalidades del Sistema

En la Tabla 3.3 se describen las principales funciones del sistema.

ID	Descripción	Prioridad
FN01	Registrarse.	Media
FN02	Iniciar sesión.	Media
FN03	Agregar adulto.	Media
FN04	Eliminar adulto.	Media
FN05	Ver registro de caídas de los adultos agregados.	Alta
FN06	Recibir alerta cuando se detecta una caída.	Alta

Tabla 3.3: Funcionalidades del Sistema

3.6.1. Diagramas de Casos de Uso

En la Figura 3.3 se presenta el diagrama de casos de uso, que tiene los siguientes actores:

- Adulto Mayor: Usuario que porta el dispositivo vestible, mientras este enviara los datos de su actividad.
- Cuidador: Usuario que recibe la alerta de una caída si es que el sistema llega a identificar alguna. También podrá acceder a ver un registro de caídas del adulto mayor.

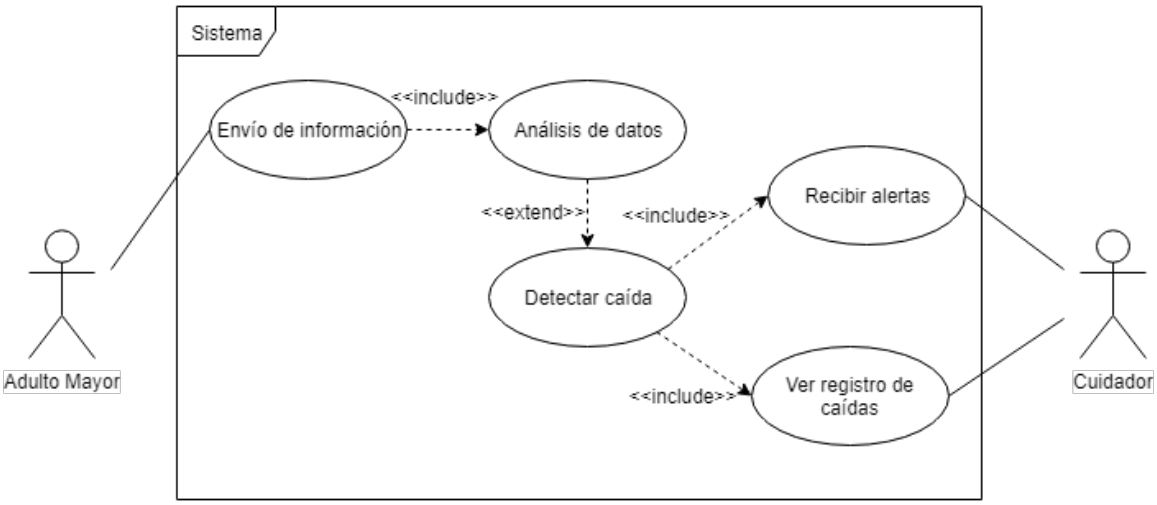

Figura 3.3: Diagrama de casos de uso.

3.6.2. Casos de Uso

En la Tabla 3.4 se detallan los casos de uso mostrados en la figura 3.3.

ID	Caso de Uso	Descripción
CU01	Envío de información	Este caso de uso representa la acción del usuario al utilizar el dispositivo y enviar información del acelerómetro.
CU02	Análisis de datos	Se analizarán los datos a través de un algoritmo basado en umbrales para detectar si se presenta una caída o no.
CU03	Detectar caída	Si el análisis de datos cumple con los parámetros, el sistema detectara una caída.
CU04	Ver registro de caídas	Muestra el registro de caídas detectadas por el sistema.
CU05	Recibir alertas	El usuario recibirá una alerta si el sistema ha detectado una caída.

Tabla 3.4: Descripción casos de uso.

3.6.3. Casos de Uso expandidos

Los casos de uso expandidos explican el proceso de integración entre el sistema en sí y quien lo usa (usuario). A continuación se presentan los principales casos de uso expandidos:

Caso de uso:	Detectar caída
ID	CUE01
Caso de uso relacionado	CU01 - CU02 - CU03 - CU05
Actor	Usuario
Tipo	Primario
Precondición	El dispositivo debe estar enviando información
Descripción	El usuario enviará información para así, analizar los datos y poder detectar una caída. Si existe una caída el usuario recibirá una alerta.
Curso normal de eventos	
Actor	**Sistema**
1. El usuario porta el dispositivo.	
	2. El dispositivo estará enviando información.
	3. Se analizarán los datos y si cumplen los parámetros, se detectara una caída.
	4. Se detecta una caída y se enviará una alerta.
5. Recibirá alerta.	
Curso alternativos de eventos	
	4. No se detecta una caída.
5. No recibirá alerta.	

Tabla 3.5: Caso de uso expandidos: Detectar caída.

Caso de uso:	Ver registro de caídas
ID	CUE02
Caso de uso relacionado	CU04
Actor	Usuario
Tipo	Primario
Precondición	Debe tener la aplicación abierta.
Descripción	Muestra registros de las caídas.
Curso normal de eventos	
Actor	**Sistema**
1. Ve registro de caídas	
	2. Muestra registro de caídas.

Tabla 3.6: Caso de uso expandidos: Ver registro de caídas.

3.6.4. Diagramas de Secuencia

A continuación se presentan los diagramas de secuencia.

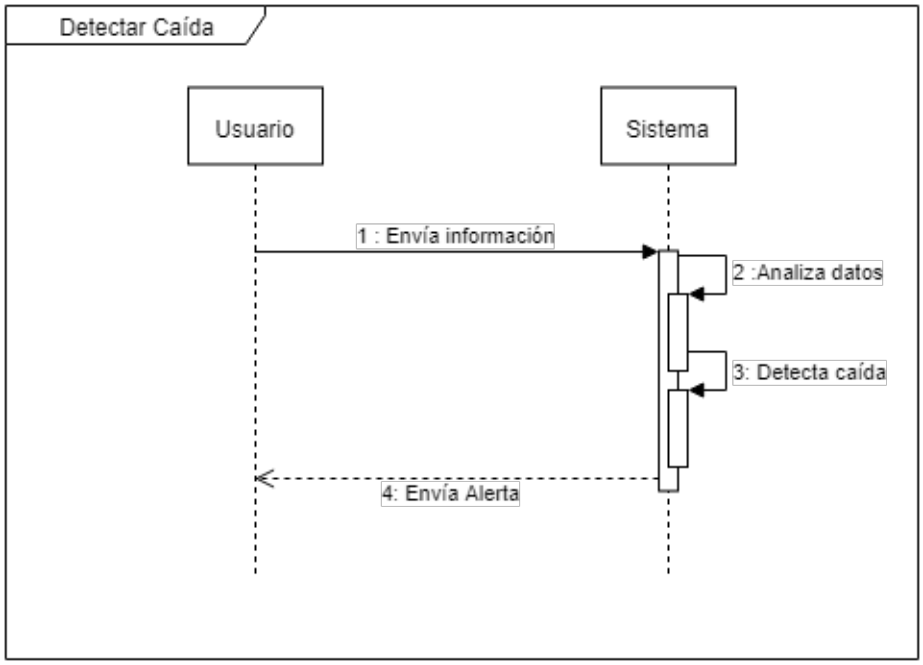

Figura 3.4: Diagrama de secuencia: Detectar caída.

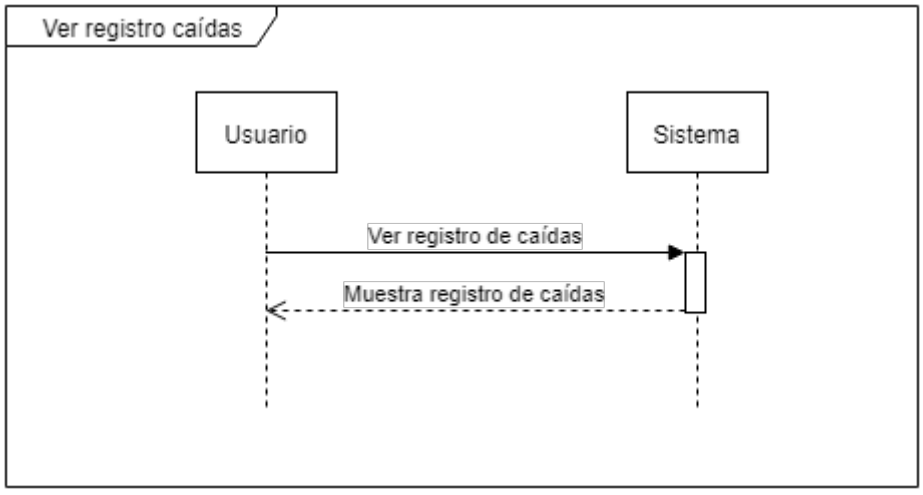

Figura 3.5: Diagrama de secuencia: Ver registro de caídas.

3.6.5. Diagramas de Estado

A continuación se presentan los diagramas de estado.

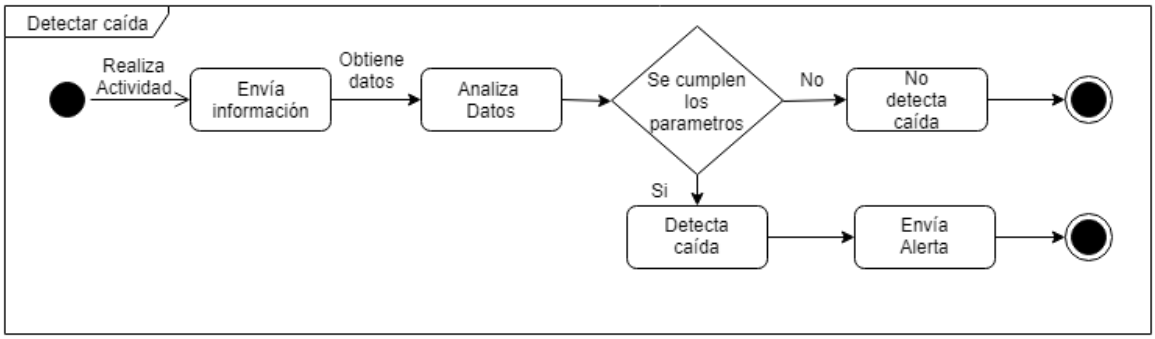

Figura 3.6: Diagrama de estados: Detectar caída.

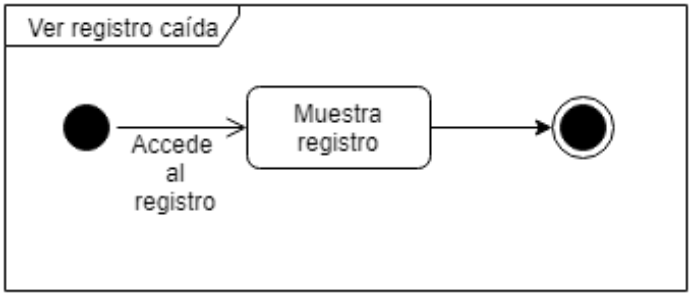

Figura 3.7: Diagrama de estados: Ver registro de caídas.

3.6.6. Modelo Conceptual

El modelo conceptual entrega una representación visual a través de un diagrama en donde se apreciaran como interactúan las diversas entidades del sistema.

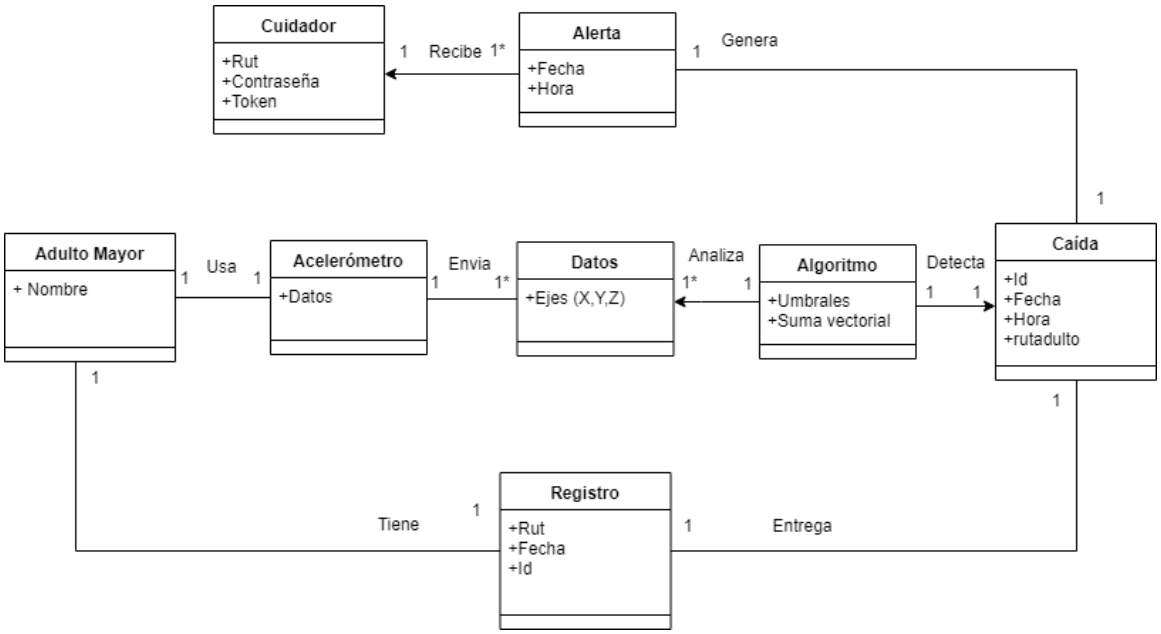

Figura 3.8: Modelo Conceptual.

Capítulo 4

Diseño

En este capítulo se documenta todo el proceso de diseño de la aplicación a desarrollar. Este capítulo se divide en las siguientes secciones: arquitectura, diseño lógico, diseño de datos, interfaces de usuario y pruebas.

4.1. Diseño Arquitectónico

En esta sección se describe el diseño de arquitectura de la aplicación explicando las tecnologías utilizadas.

4.1.1. Tecnologías utilizadas

El propósito de esta aplicación es utilizar un dispositivo vestible, que tenga acelerómetro, para poder detectar caídas en un adulto mayor. Debido a esto se decidió utilizar el vestible faros 180° [29] ya que su tamaño permite que un adulto mayor pueda portarlo a la altura de su cadera lo que ayudaría a tener una mejor evaluación de las distintas aceleraciones cuando exista una una caída, además, cuenta con un acelerómetro como sensor interno que servirá realizar el cálculo y así detectar las caídas. Para poder obtener los datos de este dispositivo se dispuso de un código en python [30] que facilita la conexión con el faros 180°, a este código se se le incluirá un algoritmo que sea capaz de detectar caídas utilizando el método de umbrales (ver Sección 4.4). Por otro lado, la aplicación móvil es solamente para ver un registro de datos, recibir alertas y administrar los adultos de los cuales se quiere obtener notificaciones cuando tengan caídas, por lo que se escogió el framework Ionic [31] para llevarla a cabo ya que facilita el uso de tecnologías web pero en este caso enfocadas en tecnologías móviles.

- Faros 180°: Dispositivo vestible que tiene un acelerómetro.
- Python: Lenguaje de programación con el que se desarrolla el algoritmo para la obtención de datos y la detección de caídas.
- Base de datos relacional: Utilizada para poder almacenar la información de usuarios y sus caídas para su posterior registro.
- Ionic framework: Framework con el que desarrollará la aplicación híbrida.

4.1.2. Flujo de datos

En la Figura 4.1 se puede ver un diagrama arquitectónico de alto nivel que representa el flujo de datos que sigue la aplicación. En primer lugar el vestible se conecta por bluetooth a una raspberry. Dentro de esta se ejecutará el código que realiza la conexión del faros 180°. Este código contará con la implementación de un algoritmo que utiliza el método de umbrales (ver Sección 4.4) detectando cada etapa de una caída. Cuando se cumplan todas las etapas se indica que se detecto una caída y así luego son registradas en una base de datos. El app server podrá comunicarse con la base de datos para luego poder renderizar las entradas en la aplicación y también a enviar los datos para así poder enviar la alerta en forma de notificación push.

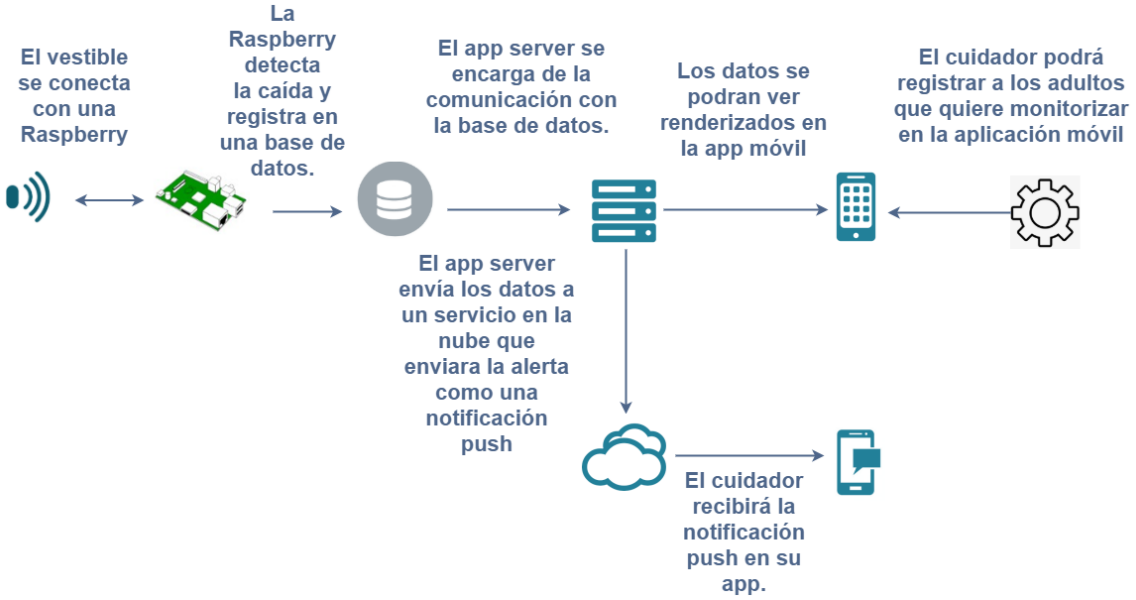

Figura 4.1: Diagrama arquitectónico

4.2. Diseño Lógico

4.2.1. Diagrama de despliegue

Los diagramas de despliegue son utilizados para facilitar la comunicación entre el hardware y el software permitiendo entregar una perspectiva global de la visualización de la implementación de un sistema de software. En la Figura 4.2 se presenta el diagrama de despliegue.

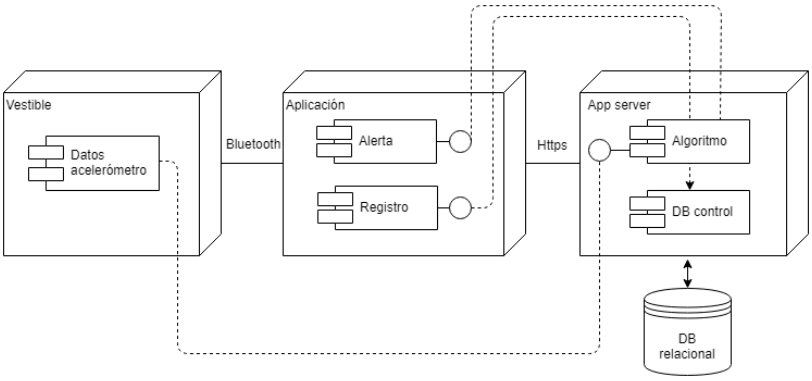

Figura 4.2: Diagrama de despliegue

4.2.2. Diagrama de componentes

Los componentes de la aplicación y sus relaciones se muestran en la Figura 4.3. A continuación se hará una descripción de cada uno de ellos.

- Datos acelerómetro: Son los datos que envía el acelerómetro a la aplicación, indispensables para detectar una caída.
- Algoritmo: Corresponde al código que utilizando el método de umbrales (ver Sección 4.4) irá detectando cada etapa de una caída hasta que esta se cumpla.
- DB control: Se refiere al control de datos como, almacenamiento y recuperación, de la base de datos.
- Registro: Corresponde al el registro de una caída, enviando los datos a la aplicación para así poder mostrarlos.
- Alerta: Corresponde a la notificación que se enviará cuando se detecte una caída.

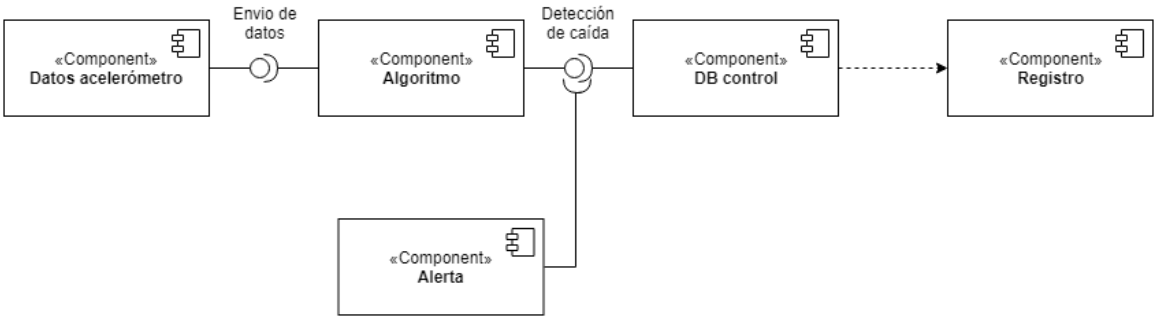

Figura 4.3: Diagrama de componentes

4.2.3. Diagrama de paquetes

Los diagramas de paquetes muestran los paquetes y las dependencias entre ellos. Estos ayudan a proporcionar agrupaciones de alto nivel de un sistema para que sea fácil visualizar cómo un paquete contiene elementos relacionados, así como la forma en que los diferentes paquetes dependen entre sí. En la Figura 4.4 se muestra el diagrama de paquetes.

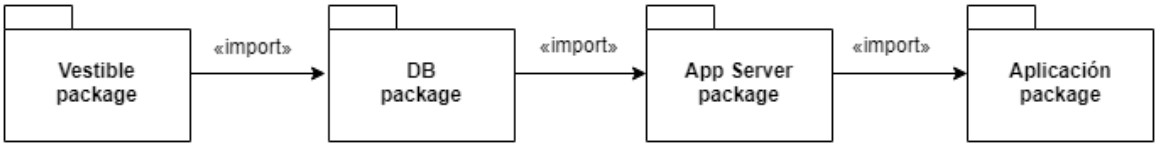

Figura 4.4: Diagrama de paquetes

4.2.4. Diagrama de clases

En la Figura 4.5 se muestra el diagrama de clases conformado por las siguientes clases:

- Cuidador: Tiene rut, contraseña y un token que permitirá recibir notificaciones. Puede ver registro de las caídas.
- Adulto Mayor: Tiene rut y envía datos.
- Registro: Contiene la id y fecha y la hora de la caída registrada al ser detectada.
- Alerta: Contiene la fecha y la hora de la caída detectada.
- Detección de caídas: Tiene los siguientes atributos como los ejes del acelerómetro, la suma vectorial usada para así compararlas con los distintos umbrales que detectan la caída. También obtiene los datos del acelerómetro y puede enviar la alerta y el registro.

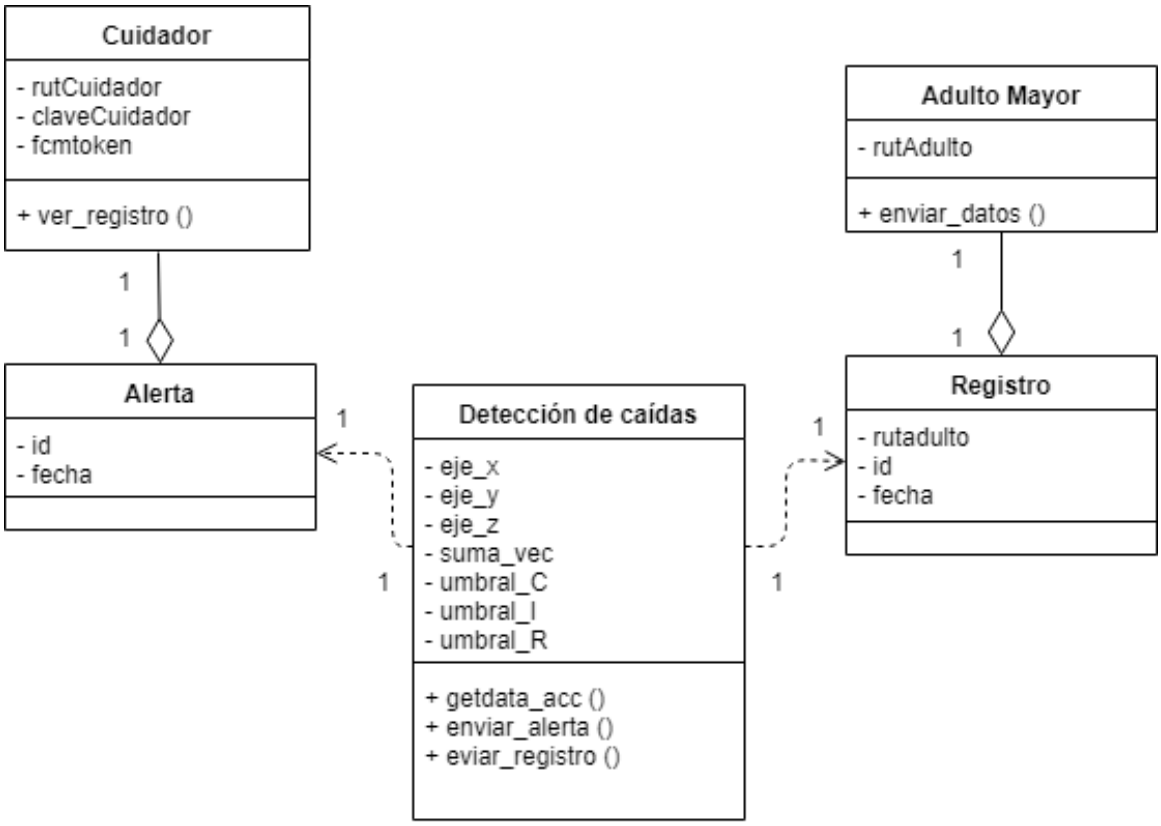

Figura 4.5: Diagrama de clases

4.3. Diseño de Datos

Como se ha comentado antes, se ha decidido almacenar los datos en una base de datos relacional. En la Figura 4.6 se muestra el modelo entidad-relación, en la Figura 4.7 relacional y en la Tabla 4.1 el diccionario de datos.

4.3.1. Diagrama Entidad Relación

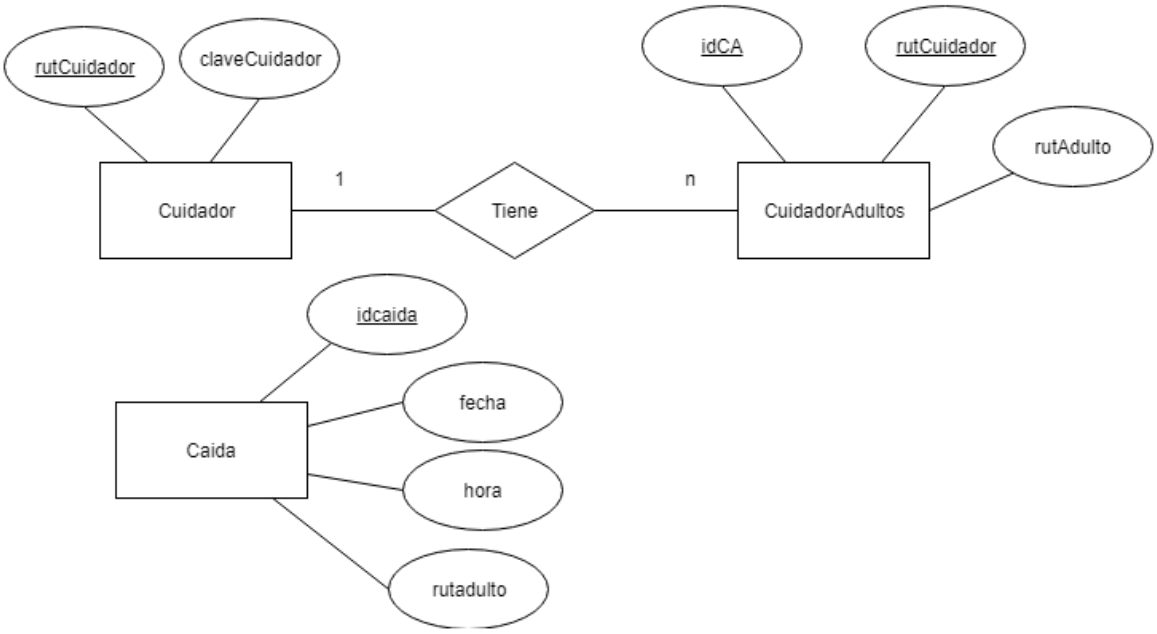

Figura 4.6: Diagrama modelo entidad relación

4.3.2. Diagrama Relacional

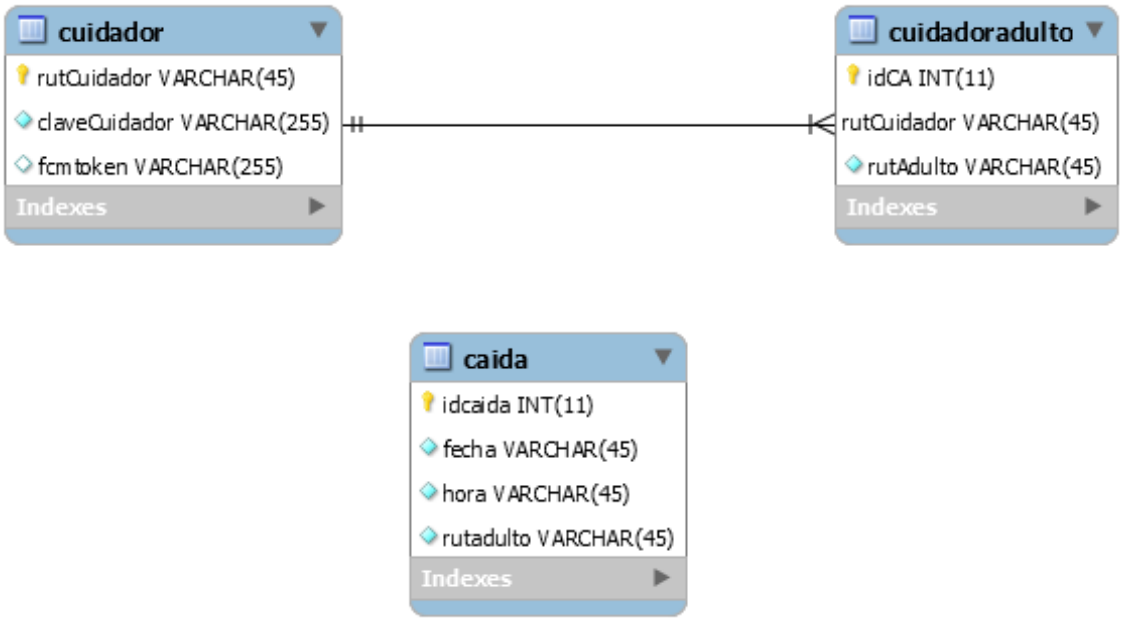

Figura 4.7: Diagrama modelo relacional

4.3.3. Diccionario de Datos

Entidad	Atributo	Tipo	Tamaño	Descripción
cuidador	- rutCuidador (PK) - claveCuidador - fcmtoken	- VARCHAR - VARCHAR - VARCHAR	- 45 - 255 - 255	- Identificador de Cuidador. - Clave de cuidador. -Token para notificaciones.
cuidadoradulto	- idCA (PK) - rutCuidador - rutAdulto	- INT - VARCHAR - VARCHAR	- 11 - 45 - 45	- Identificador. - Rut del Cuidador. - Rut del Adulto mayor.
caida	- idcaida (PK) - fecha - hora - rutadulto	- INT - VARCHAR - VARCHAR - VARCHAR	- 11 - 45 - 45 - 45	- Identificador de caída. - Fecha de la caída. - Hora de la caída. - Rut del Adulto mayor.

Tabla 4.1: Diccionario de Datos

4.4. Diseño del Algoritmo

El algoritmo diseñado en este trabajo sigue un conjunto muy específico de instrucciones para poder detectar una caída. Para empezar, se definieron los umbrales para las 3 fases de una caída; caída libre, impacto, reposo. A estos se les definió valores con respecto a las aceleraciones y desaceleraciones que tendría alguien al tener una caída.

El valor del UMBRAL_C corresponde a la fase de la caída libre, en la cual se produce una aceleración del eje x similar a la gravedad, por lo que se definió un valor de $0{,}6g$ (recordar: $g = 9{,}82\frac{m}{s^2}$) como un valor adecuado para detectarla.

El UMBRAL_I corresponde a la fase del impacto en donde se suman los 3 vectores del acelerómetro (eje x + eje y + eje z) y la suma debería superar ese umbral para demostrar que hubo una desaceleración del cuerpo al haber impactado, por lo que el valor que se definió para el umbral fue de $1{,}5g$, se vera si se alcanza ese valor durante un TIEMPO_CI de 2 segundos, si no es alcanzado se procederá a preguntar nuevamente por la fase anterior.

Si el UMBRAL_I fue alcanzado se esperara un TIEMPO_IR de 2 segundos para poder preguntar por el umbral de reposo. El umbral de reposo es mas bien un rango en el cual el eje x debe permanecer durante un TIEMPO_R de 2 segundos dentro de un RANGO_R de $0{,}5g$ y $-0{,}5$, si se llegase a salir de ese rango no se detectaría como caída, en el caso contrario de que pasen los 2 segundos y el eje x se quedo en ese rango durante ese tiempo, se detectaría una caída.

A continuación, en la Figura 4.9 se presenta el pseudocódigo y en la Figura 4.9 el diagrama de flujo del algoritmo de detección de caídas.

1. INICIO
2. Si [EjeX == UMBRAL_C] Entonces
   avanza a paso 3;
   SiNo
   vuelve a INICIO;
   FinSi
3. Inicializa TIEMPO = 0;
4. Inicializa SUMA = EjeX + EjeY + EjeZ;
5. Si [TIEMPO == TIEMPO_CI] Entonces
   vuelve a paso 1;
   SiNo
   avanza a paso 6;
   FinSi
6. Si [SUMA == UMBRAL_I] Entonces
   avanza a paso 7;
7. Inicializa TIEMPO = 0;
8. Si [TIEMPO == TIEMPO_IR] Entonces
   avanza paso 9;
   SiNo
   avanza a paso 8;
   FinSi
7. Inicializa TIEMPO = 0;
8. Si [TIEMPO == TIEMPO_IR] Entonces
   avanza paso 9;
   SiNo
   avanza a paso 8;
   FinSi
9. Inicializa TIEMPO = 0;
10. Si [TIEMPO == TIEMPO_R] Entonces
    avanza a paso 12;
    SiNo
    avanza a paso 11;
    FinSi
11. Si [EjeX € RANGO_R] Entonces
    vuelve a paso 10;
    SiNo
    vuelve a paso 1;
    FinSi
12. SE HA DETECTADO QUE EXISTE UNA CAÍDA
13. Vuelve a paso 1;

Figura 4.8: Pseudocódigo del algoritmo detección de caídas

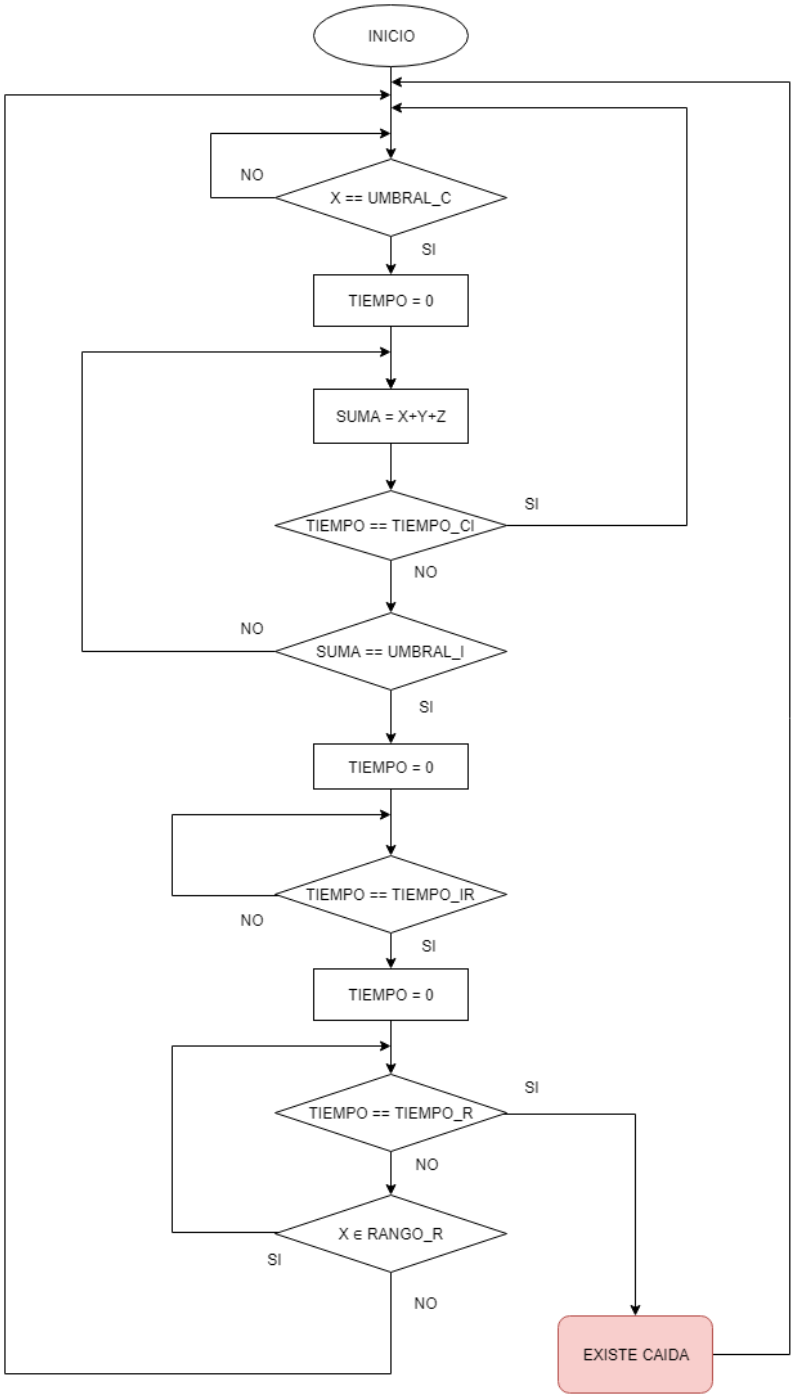

Figura 4.9: Diagrama algoritmo detección de caídas

4.5. Diseño de Interfaz

4.5.1. Arquitectura de la Información

La arquitectura de la información se refiere a la estructura de la organización del sitio, especialmente en cómo las diferentes páginas del sitio están relacionadas entre sí. Estas ayudan a orientar a los usuarios facilitando al máximo los procesos de comprensión y asimilación de la información así como las tareas que ejecutarán dentro del sistema. En la Figura 4.10 se muestra la estructura del sistema.

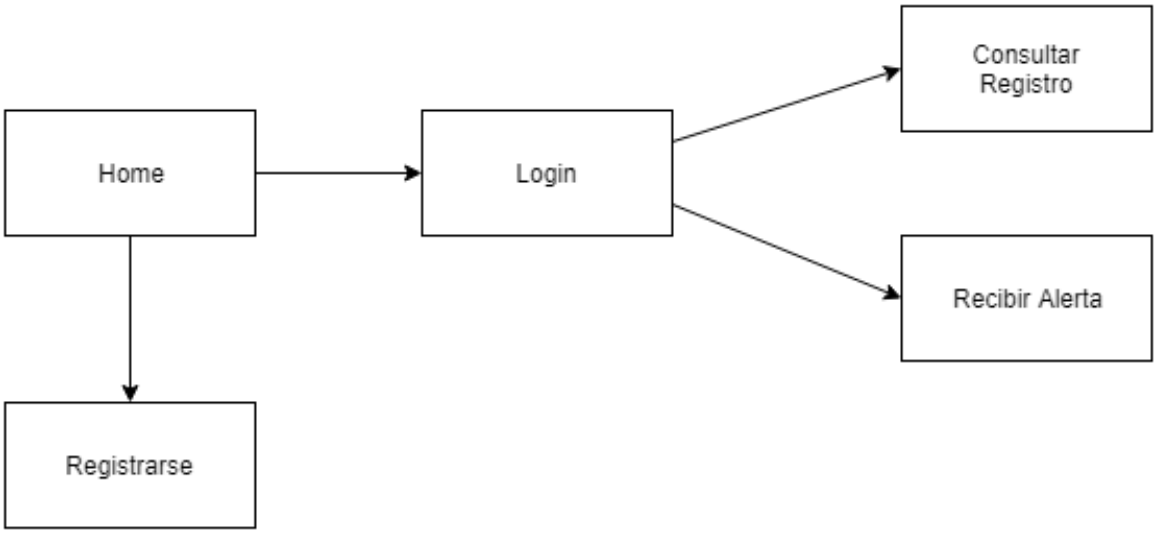

Figura 4.10: Arquitectura de la Información

4.5.2. Interfaces Gráficas

A continuación, se presentan los diseños de las principales pantallas para las interfaces de usuario.

- Interfaz iniciar sesión: Aquí el cuidador ingresa para que pueda registrar adultos a su nombre y poder ver las caídas que esos adultos han tenido. En la Figura 4.11 se puede observar el diseño de está interfaz.
- Interfaz de registro usuario: Aquí el cuidador debe registrarse para poder luego entrar al sistema. En la Figura 4.12 se puede observar el diseño de está interfaz.
- Interfaz de adultos: Aquí el cuidador una vez haya ingresado, podrá agregar y eliminar adultos a su lista de adultos, para luego poder ver las caídas de ellos en la interfaz de caídas. En la Figura 4.13 se puede observar el diseño de está interfaz.

- Interfaz de caídas: Una vez el cuidador ingrese los adultos que quiere monitorear, en esta interfaz podrá ver el registro de caídas de cada uno de ellos, mostrando el rut del adulto que tuvo una caída además de la fecha y hora de la caída. En la Figura 4.14 se puede observar el diseño de está interfaz.

Figura 4.11: Interfaz iniciar sesión

Figura 4.12: Interfaz de registro

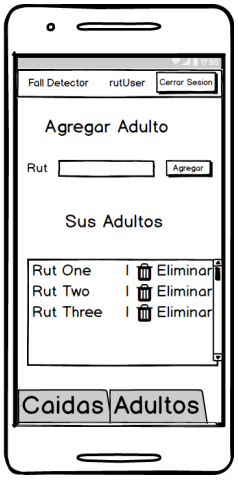

Figura 4.13: Interfaz de adultos

Figura 4.14: Interfaz de caídas

4.6. Diseño de Pruebas

4.6.1. Pruebas Unitarias

Para los casos de uso más relevantes del sistema se realizarán pruebas unitarias, las cuales son una parte esencial del proceso de pruebas. Se utilizará el método de caja negra, en el cual se prueban valores de entrada para así posteriormente evaluar los valores de salida. De esta manera, se identificarán y corregirán los errores de implementación de una forma más efectiva y sencilla. En la Tabla 4.2 se muestra el formato a utilizar.

Id de Prueba - Nombre de la Prueba	
Caso de uso asociado	
Descripción	
Casos de prueba	
Entrada válida	
Salida esperada	
Resultado	Correcto/Incorrecto

Tabla 4.2: Pruebas Unitarias

Se dará por aprobada las pruebas unitarias cuando los resultados de esta prueba correspondan al 100 % de resultados correctos.

4.6.2. Pruebas de Integración

Para las pruebas de integración se escogió utilizar la técnica "Bottom Up" que consiste en empezar la integración y pruebas por los módulos que están en los niveles inferiores de abstracción e integrar incrementalmente los niveles superiores. Como se aprecia en la Figura 4.15, se irán probando los módulos de forma gradual, uniéndose a través de sus entradas y salidas. A continuación se describen los procesos de cada nivel.

- Nivel A: Se probará enviar los datos del acelerómetro. Esto no requiere una salida de algún otro proceso.
- Nivel B: Se probará obtener los datos obtenidos de la salida de lo mencionado en el nivel A.
- Nivel C: Se hará la prueba de detectar la caída utilizando el algoritmo de detección. Para esto se requieren los datos enviados en el nivel A y que se obtengan exitosamente como se describe en el nivel B.

- Nivel D: En este nivel se harán las pruebas para almacenar la caída detectada en la base de datos.
- Nivel E: En este nivel se hará la prueba de obtener y ver el registro de caídas en la aplicación.
- Nivel F: En este nivel se harán las pruebas de recibir una alerta posterior a detectar la caída.
- Nivel G: Se prueba el sistema por completo.

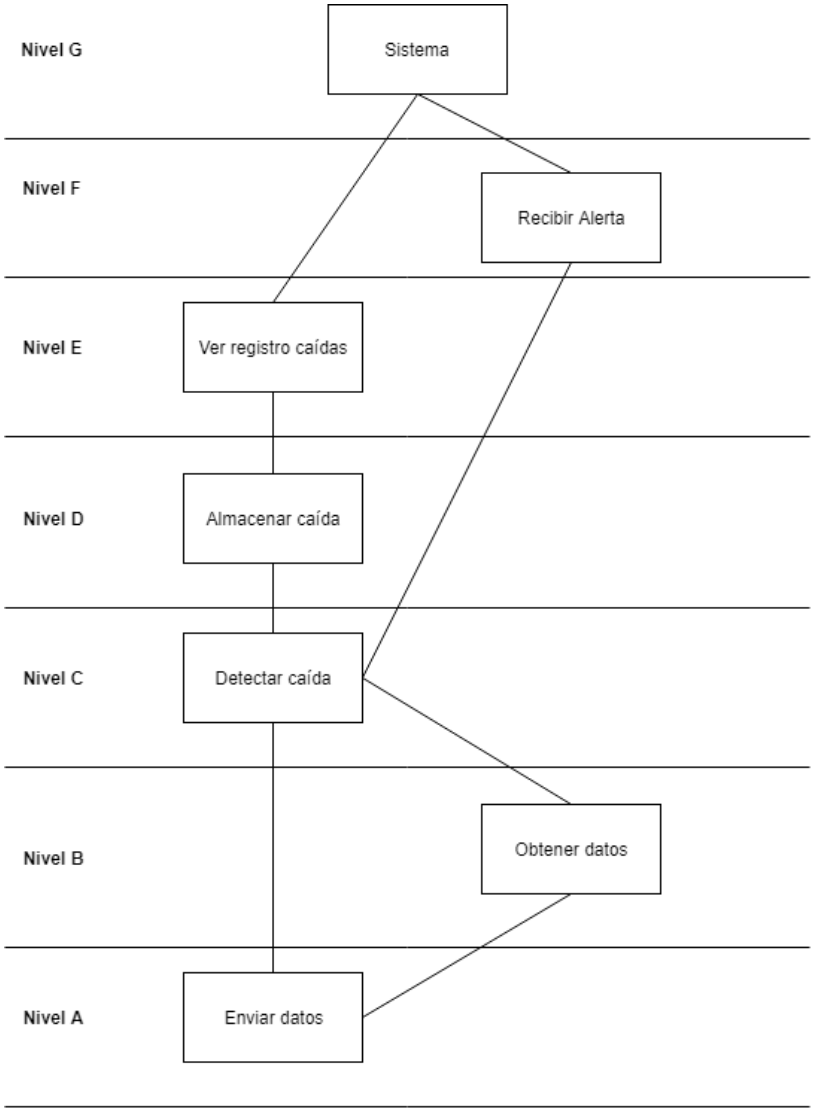

Figura 4.15: Pruebas de Integración

En la Tabla 4.3 se verán reflejados los resultados de cada módulo para una final evaluación de la prueba de integración.

Id Prueba	Id Módulo	Nombre Módulo	Dependencia	Estado de integración
				Correcto/Incorrecto

Tabla 4.3: Pruebas de Integración

Se dará por aprobada la prueba de integración cuando el resultado de esta prueba corresponda al 100 % de estados de integración correctos.

4.6.3. Pruebas de sistema

Se ha diseñado un plan de pruebas de sistema, donde se analizara el nivel de cumplimiento de los requerimientos funcionales y no funcionales. El ”nivel de cumplimiento se clasificará en los siguientes estados:

- **Incompleto:** El requerimiento no ha sido implementado en su totalidad con o sin errores.
- **Errores constantes:** Está implementado pero tiene errores frecuentemente o de alto impacto en el sistema.
- **Errores esporádicos:** Está implementado pero tiene errores escasamente o que no son de gran importancia.
- **Completo:** El requerimiento ha sido implementado en su totalidad y sin errores.

En la Tabla 4.4 se muestra el formato de esta prueba.

Id Requerimiento	Descripción Requerimiento	Tipo Requerimiento	Nivel de cumplimiento

Tabla 4.4: Pruebas de Sistema

Se dará por aceptada la prueba de sistema con el cumplimiento del 100 % de los requerimientos definidos en esta prueba.

4.6.4. Pruebas con Usuarios

4.6.4.1. Prueba de satisfacción de la aplicación

Con el fin de obtener datos confiables y una correcta evaluación de la aplicación se utilizaran las observaciones de los usuarios, en este caso los cuidadores de los adultos mayores, para poder evaluar sus principales funcionalidades. Para ello se consideraron 5 participantes a los cuales se les pedirá usar la aplicación y posteriormente llenar la Tabla 4.5 con niveles de apreciación por cada funcionalidad.

Funcionalidad	**Usuario #**	**Promedio funcionalidad**
Registrarse		
Iniciar sesión		
Agregar adulto		
Eliminar adulto		
Ver registro de caídas de los adultos agregados		
Recibir alerta cuando se detecta una caída		
Promedio total		

Tabla 4.5: Pruebas con usuarios: Satisfacción de la aplicación

Los niveles de apreciación se definirán en la tabla 4.6.

Valor	1	2	3	4
Descripción	Muy insatisfecho	Insatisfecho	Satisfecho	Muy satisfecho

Tabla 4.6: Satisfacción de la aplicación: Niveles de apreciación

Para dar por aprobada la satisfacción de la aplicación se necesitará que el promedio total de los valores indicados por los usuarios a cada funcionalidad corresponda a un nivel igual o mayor a satisfecho.

4.6.4.2. Pruebas de Rendimiento

Para las detecciones de caídas, la especificidad y la sensibilidad se definen [32] como:

- Especificidad: La proporción de negativos reales que se identifican correctamente como tales, en este caso, cuando no existe una caída y el sistema identifica que ésta no existió.
- Sensibilidad: La proporción de positivos reales que se identifican correctamente como tales, en este caso, cuando existe una caída y el sistema identifica que ésta existió.

Con el fin de poder saber si el sistema detecta caídas correctamente, se aplicarán pruebas de sensibilidad y especificidad para el sistema de detección de caídas.

El sistema tendrá 4 posibles resultados:

- Verdadero Positivo (VP): Ocurre una caída y el sistema la detecta.
- Falso Positivo (FP): No ocurre una caída y el sistema la detecta.
- Verdadero Negativo (VN): No Ocurre una caída y el sistema no la detecta.
- Falso Negativo (FN): Ocurre una caída y el sistema no la detecta.

De esos 4 posibles resultados, se tendrán los criterios de evaluación[19]:

$$Especificidad = \frac{VN}{VN + FP} \times 100 \tag{4.1}$$

$$Sensibilidad = \frac{VP}{VP + FN} \times 100 \tag{4.2}$$

Para obtener los resultados de estos criterios se han diseñado 2 tablas. La Tabla 4.7 servirá para realizar las pruebas de especificidad, la cual incluye actividades de la vida diaria, las cuales serian: sentarse/levantarse de una silla, acostarse/levantarse de la cama y recoger un objeto del suelo. Para poder evaluar la sensibilidad se diseño la Tabla 4.8 que incluye tipos de caídas, las cuales serian: caídas hacia la derecha, izquierda, adelante y atrás. Para estas pruebas se consideraran 6 usuarios, 2 adultos jóvenes y 4 adultos mayores. Para la prueba de especificidad se evaluaran a los 6 usuarios y para la prueba de sensibilidad se evaluaran a los 4 adultos jóvenes del grupo.

Pruebas especificidad			
Actividad	**Numero de pruebas**	**Usuario #**	
		FP	**VN**
Sentarse/levantarse de una silla			
Acostarse/levantarse de la cama			
Recoger un objeto del suelo			

Tabla 4.7: Pruebas de Rendimiento: Especificidad

Pruebas sensibilidad			
Caída	**Numero de pruebas**	**Usuario #**	
		VP	**FN**
Caída hacia atrás			
Caída hacia adelante			
Caída hacia la derecha			
Caída hacia la izquierda			

Tabla 4.8: Pruebas de Rendimiento: Sensibilidad

El resultado que den los criterios de evaluación puede que no sean estadísticamente viables, sin embargo ofrecerán una información general sobre el comportamiento del sistema ante los posibles resultados. De esta manera, se podrán realizar la mayor cantidad de correcciones para que el sistema funcione de la mejor manera.

De este modo la prueba de rendimiento se dará por aprobada basándonos en los valores indicados en el estudio que muestra la Tabla 2.1 en la sección Estado del Arte, siendo así los valores de aprobación para la especificidad y sensibilidad mayores o iguales a 85 %.

4.6.5. Pruebas de Aceptación

Las pruebas de aceptación tienen como propósito demostrar al usuario el cumplimiento de los requerimientos funcionales de la aplicación, donde por cada uno de ellos se generan pruebas de aceptación, en las cuales se define un escenario (secuencia de pasos) de ejecución o uso de la aplicación desde la perspectiva del cliente. Para este tipo de pruebas se especificó una tabla con los requerimientos iniciales para obtener los resultados macro de cada uno de ellos con algunas observaciones. En la Tabla 4.9 se presenta el diseño de la prueba de aceptación.

Requerimiento	Sí cumple	Cumple parcialmente	No cumple	Observaciones
El programa puede capturar datos del acelerómetro.				
El programa tiene umbrales fijos para detectar cada etapa de una caída.				
El programa permite detectar caídas.				
El cuidador puede registrar a los adultos que quiere monitorear en la app móvil.				
El cuidador puede ver registro de caídas por fecha de sus adultos monitoreados en la app móvil.				
El cuidador puede recibir alertas de las caídas en la app móvil.				

Tabla 4.9: Pruebas de Aceptación

Se dará por aprobada la prueba de aceptación con el 100 % de cumplimiento de los requerimientos definidos en la prueba.

Capítulo 5

Implementación

En este capítulo se dará a conocer el hardware y software utilizado para llevar a cabo la implementación de este trabajo, incluyendo los lenguajes de programación y estrategias de implantación. Además, también se mostraran las interfaces que se realizaron al implementar la aplicación.

5.1. Hardware utilizado

Para la implementación se utilizó un computador portátil con las siguientes características.

Marca y Modelo : Asus FX504GE-E4367T

Procesador : Intel Core i5 8300H

Memoria Ram : 8 GB

Almacenamiento : SSD 250GB PCIe NVMe M.2 y HDD 1 TB

Sistema Operativo : Windows 10 64 bits

Graficos : NVIDIA GeForce GTX 1050 Ti

Además, se utilizó el hardware Faros 180° [29] que corresponde al dispositivo vestible que se utiliza para enviar los datos desde el acelerómetro. Por otro lado, el algoritmo que detecta caídas (implementado en el computador portátil) se almacena en una una Raspberry Pi 3 la cual se utiliza como 'totem' para que pueda ser conectado mediante bluetooth al Faros 180°.

5.2. Software utilizado

Para la transmisión de datos y el algoritmo de detección de caídas, se utilizó el software de código abierto 'faros-streamer-2' [33] el cual permite la comunicación con el hardware Faros 180° y de esa manera poder ejecutar el algoritmo que utiliza el método de umbrales (ver Sección 4.4) para detectar caídas, integrándose con el software 'faros-streamer-2'.

Para almacenar las caídas y todo lo relacionado con el sistema se utilizó el motor de base de datos MySQL 5.6 [34], el cual es un sistema de gestión de bases de datos relacionales que utiliza el lenguaje de consulta estructurado SQL para acceder y transferir los datos y comandos tales como seleccionar, actualizar, eliminar e insertar para su administración.

Para la aplicación, por lado del servidor se utilizo ExpressJS [35], el cual es un framework realizado en NodeJS [36] muy ligero y flexible que proporciona un conjunto muy robusto de facilidades para crear servidores web y recibir peticiones HTTP, por lo que permite desarrollar API REST de forma muy rápida. Por el lado del cliente se utilizo Ionic [31] el cual es un framework que facilita la creación de aplicaciones móviles en diferentes plataformas que pueden instalarse en teléfonos con Android e iOS. Para las notificaciones se utilizó la plataforma Firebase de google [37] que es la encargada de gestionar el servicio de notificaciones push que utiliza en el sistema.

5.3. Lenguajes de programación

Para la creación del algoritmo se utilizó Python [30] el cual nos ofrece una amplia librería de funciones, proporcionando un desarrollo rápido y sencillo. Además, el software 'faros-streamer-2' se encuentra desarrollado en ese lenguaje por lo que se decidió utilizarlo.

Para la creación de la aplicación se utiliza Javascript, el cual es un lenguaje de programación ligero e interpretado por la mayoría de los navegadores. Sirve principalmente para mejorar la gestión de la interfaz cliente/servidor. También se utilizo HTML5 que es un lenguaje de programación que permite el desarrollo de las interfaces web en conjunto con CSS y SCSS que son lenguajes utilizados para describir y mejorar la presentación de los documentos HTML.

5.4. Estrategia de implementación

La estrategia para la implementación se fue definiendo de acuerdo a como se iba a implementar la transmisión de datos desde el dispositivo a la base de datos. En un principio, se pensaba utilizar un smartphone que se conectara al vestible y que así mandara constantemente los datos al servidor, pero fue descartado debido a que generaría un alto costo en el envió de datos. Por ello se decidió evaluar las caídas en forma local y así una vez detectada se procedería a enviar los datos.

Por otro lado, los patrones estructurales utilizados en la implementación fueron Modelo-Vista-Controlador (ya que la aplicación está realizada en Ionic, el cual sigue ese patrón de diseño al desarrollar una aplicación). Además, se puede incluir Facade [38] dado que la aplicación solamente seria interfaz simple para un subsistema complejo, por lo seria un fachada para la visualización de los datos que provienen desde la base de datos y la lógica que se encuentra en el servidor.

5.5. Interfaces

En esta sección se describirán las interfaces que se implementaron para la aplicación móvil. Se hacer un recorrido de cada interfaz mostrando que se puede hacer y que restricciones se generan en cada una de ellas.

Interfaz para Registrarse al sistema: En esa interfaz el cuidador podrá registrarse en el sistema, en la Figura 5.1 podemos ver como nos pide registrarnos con nuestro Rut y una contraseña, luego al registrarse correctamente sera redirigido a la interfaz de ingresar al sistema, de otra manera si se intenta registrar con un Rut que ya existe se mostrará una aviso de que el usuario ya existe como se puede ver en la Figura 5.2.

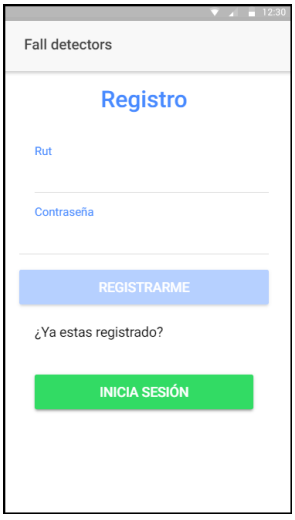

Figura 5.1: Interfaz Registro.

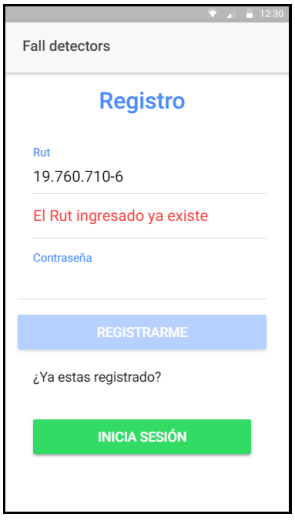

Figura 5.2: Registro fallido.

Interfaz para Ingresar al sistema: En esa interfaz el cuidador se podrá autenticar en el sistema. En la Figura 5.3 podemos ver la interfaz esperando por ingresar con nuestras credenciales, si la autenticación es correcta podremos ingresar, de otra manera si la autenticación falla se mostrará un mensaje indicando que el Rut o la contraseña con incorrectos como se puede ver en la Figura 5.4.

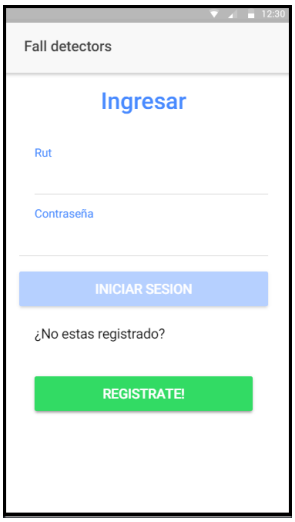

Figura 5.3: Interfaz ingreso.

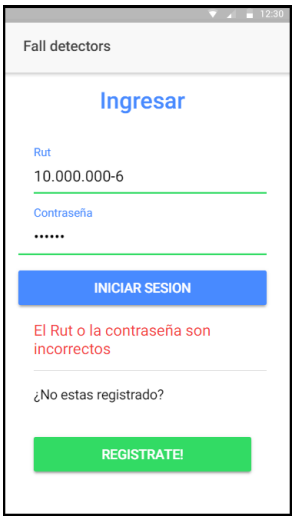

Figura 5.4: Ingreso fallido.

Interfaz para Agregar y Eliminar Adultos: Como se puede ver en la Figura 5.5 en esta interfaz el cuidador podrá agregar y eliminar a que adultos podrán consultar su el registro de caídas. Una vez se agregue un adulto a su lista le saldrá un mensaje, como se ve en la Figura 5.6, indicando que ha agregado al adulto y será redirigido a la interfaz de registro de caídas. Si se intenta ingresar un adulto que ya se encuentra en su lista, saldrá un mensaje avisando que ese adulto ya esta ingresado como indica la Figura 5.7.

Interfaz para Ver el Registro de caídas: En esta interfaz el cuidador podrá ver una lista que contendrá las caídas de todos los adultos mostrando su Rut junto con la fecha y hora del suceso como se puede apreciar en al Figura 5.8.

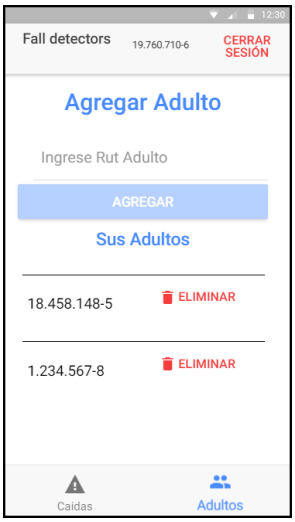

Figura 5.5: Lista con adultos.

Figura 5.6: Aviso al agregar adulto.

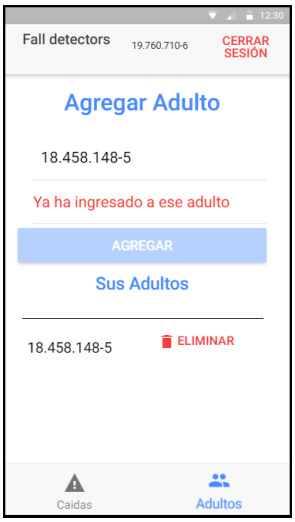

Figura 5.7: Ingreso de adulto fallido.

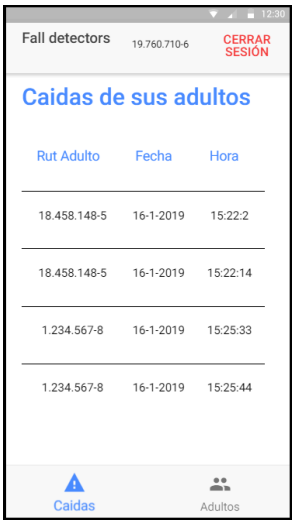

Figura 5.8: Lista de caídas.

Capítulo 6

Pruebas y análisis de resultados

En este capítulo se exponen los resultados de las pruebas realizadas al sistema. Como se menciono en la sección 4.6 se llevaron a cabo pruebas unitarias, pruebas de integración, pruebas de sistema, pruebas con usuarios y pruebas de aceptación. A continuación, se dará a conocer la ejecución de las pruebas y sus resultados:

6.1. Pruebas Unitarias

En las pruebas unitarias, se realizan tareas para analizar pequeñas partes del sistema con el objetivo de asegurar que cada parte que conforma la totalidad de éste tenga el funcionamiento correcto. Para este caso se utilizó la prueba de caja negra debido a la naturaleza de la implementación.

En las siguientes tablas, se presentan las pruebas unitarias realizadas:

PU01 - Transmisión de datos	
Caso de uso asociado	CU01 - Envió de información
Descripción	Ejecutar el programa para que comience a recibir los datos que envía el acelerómetro.
Casos de prueba	
Entrada válida	Dispositivo emparejado y transmisión de datos iniciada.
Salida esperada	Imprimir en consola los datos transmitidos desde el acelerómetro en tiempo real, verificando que estos cambien cuando el dispositivo se mueva.
Resultado	Correcto

Tabla 6.1: Prueba Unitaria 01 - Transmisión de datos

PU02 - Análisis de umbrales	
Caso de uso asociado	CU02 - Análisis de datos
Descripción	El algoritmo detectara los umbrales para la detección de caídas
Casos de prueba	
Entrada válida	Dispositivo emparejado y transmisión de datos iniciada.
Salida esperada	Imprimir en consola cuando los datos transmitidos se presenten en cada umbral.
Resultado	Correcto

Tabla 6.2: Prueba Unitaria 02 - Análisis de Umbrales

PU03 - Detectar una caída	
Caso de uso asociado	CU03 - Detectar caída
Descripción	El algoritmo detectara cuando existe y no existe una caída.
Casos de prueba	
Entrada válida	Transmisión de datos iniciada y se realiza una caída.
Salida esperada	Imprimir en consola cuando exista una caída, en caso contrario imprimir que no existe caída.
Resultado	Correcto

Tabla 6.3: Prueba Unitaria 03 - Detectar una caída

PU04 - Ver registro de caída	
Caso de uso asociado	CU04 - Ver registro de caídas
Descripción	En la aplicación móvil ingresar el rut del adulto de quien se quiere ver la caída registrada.
Casos de prueba	
Entrada válida	Caída detectada y ya almacenada en la base de datos.
Salida esperada	Ingresar el rut del adulto que tuvo una caída y visualizarla en la aplicación.
Resultado	Correcto

Tabla 6.4: Prueba Unitaria 04 - Ver registro de caída

PU05 - Recibir notificación	
Caso de uso asociado	CU05 - Recibir alertas
Descripción	En la aplicación móvil, luego de detectar una caída e ingresar el rut del adulto a la lista de adultos monitorizados, recibir una notificación alertando sobre la caída.
Casos de prueba	
Entrada válida	Caída detectada.
Salida esperada	Recibir en la aplicación una notificación indicando que existió una caída.
Resultado	Correcto

Tabla 6.5: Prueba Unitaria 05 - Recibir notificación

6.1.1. Resultados pruebas unitarias

Se realizaron 5 pruebas unitarias que corresponden a los casos de uso que forman parte del sistema. De las 5 pruebas unitarias realizadas se obtuvo un resultado exitoso del 100 %. Por lo tanto, al tener un 100 % las pruebas unitarias se encuentran aprobadas. A continuación en la Tabla 6.6 se puede ver el resultado de las pruebas unitarias.

Porcentaje sin errores	Porcentaje con errores	Número de pruebas
100 %	0 %	5

Tabla 6.6: Resultado Pruebas unitarias

6.2. Pruebas de Integración

Para las pruebas de integración, se utilizó la técnica Botton-Up, la cual consiste en ir probando integraciones de los módulos del sistema con sus dependencias de forma gradual, desde el nivel más bajo, hasta llegar al sistema completo. En la Tabla 6.7 se puede ver el resultado de las pruebas de integración.

Id Prueba	Id Módulo	Nombre Módulo	Dependencia	Estado de integración
PI01	MA01	Enviar datos	-	Correcto
PI02	MB01	Obtener datos	MA01	Correcto
PI03	MC01	Detectar caída	MA01, MB01	Correcto
PI04	MD01	Almacenar caída	MC01	Correcto
PI05	ME01	Ver registro caídas	MD01	Correcto
PI06	MF01	Recibir Alerta	MC01	Correcto
PI07	MG01	Sistema	ME01, MF01	Correcto

Tabla 6.7: Pruebas de integración

6.2.1. Resultados pruebas de integración

De las 7 pruebas de integración realizadas, se obtuvo un resultado exitoso del 100 % el cual corresponde a la integración de los módulos que forman parte del sistema detector de caídas y de la aplicación móvil, incluyendo también el sistema completo. Por lo tanto, al tener un 100 % la prueba de integración se encuentra aprobada. En la Tabla 6.8 se puede ver el resultado de las pruebas de integración.

Porcentaje sin errores	Porcentaje con errores	Número de pruebas
100 %	0 %	7

Tabla 6.8: Resultado Pruebas de integración

6.3. Pruebas de Sistema

Las pruebas de sistema, se basan principalmente en medir el cumplimiento de los requerimientos, tanto funcionales como no funcionales. En la Tabla 6.9 se muestran los resultados de cada uno de ellos.

Id Requerimiento	Descripción Requerimiento	Tipo Requerimiento	Nivel de cumplimiento
RF01	El programa puede capturar datos del acelerómetro.	Funcional	Completo
RF02	El programa tiene umbrales fijos para detectar cada etapa de una caída.	Funcional	Completo
RF03	El programa permite detectar caídas.	Funcional	Completo
RF04	El cuidador puede registrar a los adultos que quiere monitorear en la app móvil.	Funcional	Completo
RF05	El cuidador puede ver registro de caídas por fecha de sus adultos monitoreados en la app móvil.	Funcional	Completo
RF06	El cuidador puede recibir alertas de las caídas en la app móvil.	Funcional	Completo
RNF01	La aplicación móvil es sencilla de utilizar para los cuidadores.	No funcional	Completo
RNF02	El sistema esta disponible siempre que se requiera.	No funcional	Completo

Tabla 6.9: Pruebas de Sistema

6.3.1. Resultados pruebas de sistema

Como se puede observar en la Tabla 6.10 se cumple el 100 % de cumplimiento de los resultados de los requerimientos tanto funcionales como no funcionales, por lo tanto se da por aprobada las pruebas de sistema.

Resultados correctos	Resultados incorrectos	Total
100 %	0 %	8

Tabla 6.10: Resultado Pruebas de sistema

6.4. Pruebas con Usuarios

6.4.1. Prueba de satisfacción de la aplicación

Para obtener una correcta evaluación de la aplicación, se utilizaran observaciones de 5 usuarios teniendo una perspectiva de cuidador de los adultos mayores para realizar la prueba. Los niveles de apreciación en la Tabla 6.11 son los valores con los que se evaluó cada funcionalidad de la aplicación. En la Tabla 6.12 se encuentran los resultados de esta evaluación.

Valor	1	2	3	4
Descripción	Muy insatisfecho	Insatisfecho	Satisfecho	Muy satisfecho

Tabla 6.11: Satisfacción de la aplicación: Niveles de apreciación

Funcionalidad	**Usuario 1**	**Usuario 2**	**Usuario 3**	**Usuario 4**	**Usuario 5**	**Promedio funcionalidad**
Registrarse	3	3	4	4	4	3,6
Iniciar sesión	4	3	4	4	4	3,8
Agregar adulto	4	4	4	4	4	4
Eliminar adulto	4	4	4	4	4	4
Ver registro de caídas de los adultos agregados	4	4	4	4	4	4
Recibir alerta cuando se detecta una caída	4	4	4	4	4	4
Promedio total	3,83333333	3,66666667	4	4	4	3,9

Tabla 6.12: Satisfacción de la aplicación

6.4.1.1. Resultados de la satisfacción de la aplicación

Como se ve en la Tabla 6.12, el promedio total de las funcionalidades de los usuarios tiene un valor de 3,9, teniendo así un valor mayor a satisfecho, por lo tanto la satisfacción de la aplicación esta aprobada.

6.4.2. Pruebas de rendimiento

Para poder obtener los resultados de los criterios de evaluación del rendimiento del sistema de detección de caídas explicados en la sección 4.6.4.2 se procedió a tener un grupo

de 6 usuarios; 4 adultos jóvenes y 2 adultos mayores, descritos en la Tabla 6.13 .

Usuario #	**Género**	**Edad**
Usuario 1	Masculino	81
Usuario 2	Femenino	73
Usuario 3	Masculino	23
Usuario 4	Femenino	24
Usuario 5	Masculino	25
Usuario 6	Masculino	25

Tabla 6.13: Pruebas de Rendimiento: Grupo de usuarios

Como se puede ver en la Tabla 6.14 las actividades que servían para evaluar la especificidad (sentarse/levantarse de una silla, acostarse/levantarse de la cama, recoger un objeto del suelo) se realizaron 10 veces cada una para cada usuario, dando como resultado verdadero negativo en su totalidad.

Pruebas especificidad													
Actividad	**Numero de pruebas**	**Usuario 1**		**Usuario 2**		**Usuario 3**		**Usuario 4**		**Usuario 5**		**Usuario 6**	
		FP	**VN**	**FP**	**VN**	**FP**	**VN**	**FP**	**VN**	**FP**	**VN**	**FP**	**VN**
Sentarse/ levantarse de una silla	60	0	10	0	10	0	10	0	10	0	10	0	10
Acostarse/ levantarse de la cama	60	0	10	0	10	0	10	0	10	0	10	0	10
Recoger un objeto del suelo	60	0	10	0	10	0	10	0	10	0	10	0	10

Tabla 6.14: Pruebas de Rendimiento: Resultado especificidad

Por otro lado en la Tabla 6.15, las simulaciones de caídas que servían para evaluar la sensibilidad (caídas hacia la derecha, izquierda, atrás y adelante) se realizaron 10 veces cada una para cada usuario, dando como resultado verdadero positivo en casi la totalidad de las simulaciones. En los casos que el resultado fue falso negativo, se deduce que puede haber sido porque el impacto que tuvieron al realizar la simulación fue muy suave por lo tanto el algoritmo no pudo superar el umbral del impacto establecido llegando así a no detectar la caída cuando efectivamente si se había simulado una.

Pruebas sensibilidad									
Caída	**Numero de pruebas**	**Usuario 3**		**Usuario 4**		**Usuario 5**		**Usuario 6**	
		VP	**FN**	**VP**	**FN**	**VP**	**FN**	**VP**	**FN**
Caída hacia atrás	40	10	0	9	1	10	0	10	0
Caída hacia adelante	40	9	1	8	2	8	2	8	2
Caída hacia la derecha	40	10	0	10	0	8	2	10	0
Caída hacia la izquierda	40	10	0	10	0	10	0	10	0

Tabla 6.15: Pruebas de Rendimiento: Resultado sensibilidad

6.4.2.1. Resultados pruebas de rendimiento

Como se puede ver en la Tabla 6.14, de un total de 180 pruebas de actividades de la vida diaria realizadas se obtuvieron 180 verdaderos negativos y 0 falsos positivos, por lo que si aplicamos los resultados a la ecuación de especificidad 6.1 nos daría un 100 % de especificidad que corresponde a cuando el sistema logra identificar cuando una caída no existe. En la tabla 6.15 de un total de 160 simulaciones de caídas realizadas se obtuvieron 150 verdaderos positivos y 10 falsos negativos, por lo que si aplicamos los resultados a la ecuación de sensibilidad 6.2 nos da un 100 % de sensibilidad que corresponde a la capacidad del sistema para identificar que una caída existe.

$$Especificidad = \frac{180}{180 + 0} \times 100 = 100\,\% \tag{6.1}$$

$$Sensibilidad = \frac{150}{150 + 10} \times 100 = 93,75\,\% \tag{6.2}$$

Por lo tanto, la prueba de rendimiento se encuentra aprobada ya que el porcentaje de los resultados de especificidad y sensibilidad superan al 85 % respectivamente.

6.5. Pruebas de Aceptación

Las pruebas de aceptación tienen como propósito demostrar al usuario el cumplimiento de los requerimientos funcionales de la aplicación. Se realizó una validación experta en base a los requerimientos funcionales del sistema. La prueba fue realizada por la Prof. Carla Taramasco de la Escuela de Ingeniería Civil Informática de la Universidad de Valparaíso. En la Tabla 6.16 se presenta la prueba de aceptación de validación realizada.

Requerimiento	Sí cumple	Cumple parcialmente	No cumple	Observaciones
El programa puede capturar datos del acelerómetro.	✓			
El programa tiene umbrales fijos para detectar cada etapa de una caída.	✓			
El programa permite detectar caídas.	✓			
El cuidador puede registrar a los adultos que quiere monitorear en la app móvil.	✓			
El cuidador puede ver registro de caídas por fecha de sus adultos monitoreados en la app móvil.	✓			
El cuidador puede recibir alertas de las caídas en la app móvil.	✓			

Tabla 6.16: Pruebas de Aceptación: Validación Profesora Carla Taramasco T.

6.5.1. Resultados pruebas de aceptación

Como se puedo apreciar en la Tabla 6.17 los requerimientos que definen la prueba de aceptación se cumplen en su totalidad por lo que esta validación indica que el sistema funciona como se esperaba. Por lo tanto al tener un 100 % de cumplimiento esta prueba se da por aceptada.

Sí cumple	Cumple parcialmente	No cumple	Total
100 %	0 %	0 %	6

Tabla 6.17: Resultado Pruebas de aceptación

Capítulo 7

Implantación

La implantación es una etapa en el desarrollo de software que consta del proceso de instalación y configuración del software desarrollado, en el ambiente donde la aplicación estará en producción. Para este trabajo de título, la implantación se dividió en tres partes: implantación del web server, el cual se realizó en un servidor cuya propiedad es de la profesora guía Carla Taramasco, implantación del programa detector de caídas, el cual se realizó en una Raspberry Pi y la generación de la apk de la aplicación móvil que posteriormente se debe implantar en la plataforma Google Play [39] pudiendo ser distribuida para todo quien desee utilizarla.

En los apéndices A y B se deja la documentación respectiva de este trabajo de título, siendo el manual de usuario y la documentación de desarrollador, respectivamente.

7.1. Requerimientos

En la Tabla 7.1 se pueden ver las características del hardware utilizado para el web server. Además, se debe tener en cuenta que el servidor tenga habilitado el puerto 8000 y que estén instaladas las siguientes herramientas, con al menos las versiones indicadas (las versiones indicadas son con las cuales el servidor contaba al momento de implantar el web server).

- **npm:** 6.9.0
- **nodejs:** 8.10.0
- **mysql:** 5.7.26
- **Puerto habilitado:** 8000

	Servidor
CPU	Intel(R) Xeon(R) CPU E3-1220 v3 @ 3.10GHz
RAM	8 GB
S.O.	Ubuntu 18.04.2 LTS

Tabla 7.1: Características del servidor

En la Tabla 7.2 se pueden ver las características de la Raspberry Pi en la cual se implanto el programa detector de caídas.

	Raspberry Pi
CPU	ARMv7 Processor rev 4 (v7l)
RAM	1 GB
S.O.	Raspbian GNU/Linux 9.9 (stretch)

Tabla 7.2: Características de la Raspberry Pi

En la Tabla 7.3 se pueden ver los requisitos que se necesitan para poder utilizar la aplicación móvil.

	Aplicación móvil
S.O.	Android 4.4 y versiones posteriores

Tabla 7.3: Requisitos aplicación móvil

7.2. Preparación de Ambiente

7.2.1. Servidor

Para comenzar se pidió al administrador del servidor que cree un usuario para tener acceso en él.

```
murray@DESKTOP-IV755SI:~$ ssh sebastian@200.14.71.41
sebastian@200.14.71.41's password:
```

Figura 7.1: Acceso al servidor con el usuario creado

Una vez teniendo acceso en el servidor se comprobó si se encontraban instalas las herramientas nombradas en la sección 7.1.

Con las herramientas ya instaladas, el administrador debió crear un usuario en mysql con permisos, con el cual se procedió a insertar la base de datos, utilizada en la implementación, dentro del servidor.

Una vez teniendo la base de datos en producción se insertó el proyecto que contiene el web server, instalando todas las sus dependencias y luego se comprobó su ejecución de forma local.

Para poder tener en producción el web server, se utilizó el programa **tmux** [40] el cual permite ejecutar programas en segundo plano dentro del servidor sin mantener nuestra sesión abierta.

Figura 7.2: Iniciación de sesión tmux

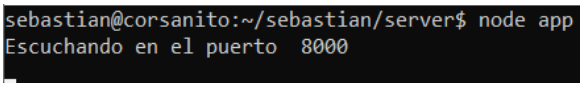

Figura 7.3: Ejecución del web server en tmux

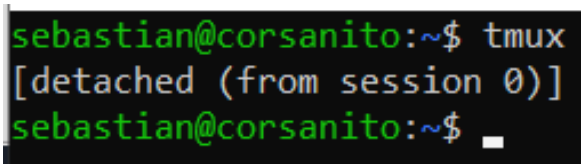

Figura 7.4: Cerrar sesión tmux manteniendo la ejecución del web server

Con esto la implantación del web server se ha completado.

7.2.2. Raspberry Pi

Para la implantación del programa detector de caídas, se descargo la última versión de Raspbian para poder instalarlo como sistema operativo de la Raspberry Pi. Se hicieron configuraciones iniciales y se descargaron e instalaron todas sus actualizaciones.

Figura 7.5: Configuraciones iniciales en Raspbian

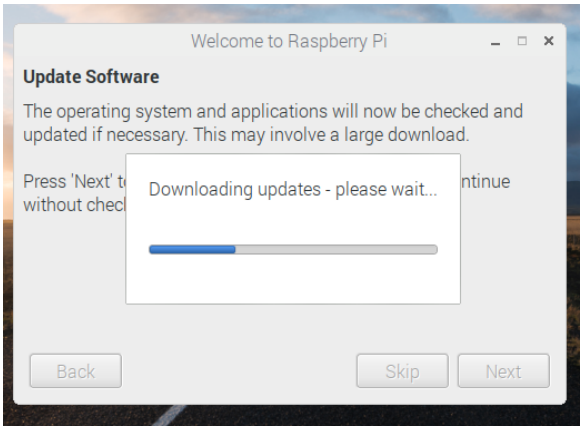

Figura 7.6: Actualización de Raspbian

Se debe comprobar que python3 y pip3 estén instalados, en este caso la versión de Raspbian instalada ya venia con python3 y pip3.

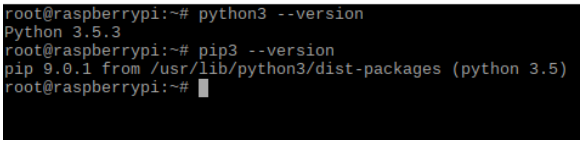

Figura 7.7: Comprobación de instalación python3 y pip3

Como requisito se debe instalar la librería de bluetooth para poder posteriormente poder instalar la librería pybluez usada en el programa detector de caídas. Además, para

que la librería pylsl (labstreaminglayer) funcione en una Rapsberry Pi se debe utilizar una compilacion de pylsl para arquitecturas ARM.

```
root@raspberrypi:/home/pi/Desktop# wget https://github.com/labstreaminglayer/liblsl/releases/download/1.12/liblsl-1.12.0-Linux-ARM7.deb
--2019-05-16 22:35:46--  https://github.com/labstreaminglayer/liblsl/releases/download/1.12/liblsl-1.12.0-Linux-ARM7.deb
Resolviendo github.com (github.com)... 192.30.253.113
Conectando con github.com (github.com)[192.30.253.113]:443... conectado.
Petición HTTP enviada, esperando respuesta... 301 Moved Permanently
Localización: https://github.com/sccn/liblsl/releases/download/1.12/liblsl-1.12.0-Linux-ARM7.deb [siguiendo]
--2019-05-16 22:35:47--  https://github.com/sccn/liblsl/releases/download/1.12/liblsl-1.12.0-Linux-ARM7.deb
Reutilizando la conexión con github.com:443.
Petición HTTP enviada, esperando respuesta... 302 Found
Localización: https://github-production-release-asset-2e65be.s3.amazonaws.com/123265865/4d2ea080-a268-11e8-9899-bcaecfa701f8?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Cred
ential=AKIAIWNJYAX4CSVEH53A%2F20190517%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20190517T023547Z&X-Amz-Expires=300&X-Amz-Signature=3227feaae0c6145300772bea1567b7ffc91
59dbfae2031ed8c66dbd6bff7a9f3&X-Amz-SignedHeaders=host&actor_id=0&response-content-disposition=attachment%3B%20filename%3Dliblsl-1.12.0-Linux-ARM7.deb&response-content-
type=application%2Foctet-stream [siguiendo]
--2019-05-16 22:35:47--  https://github-production-release-asset-2e65be.s3.amazonaws.com/123265865/4d2ea080-a268-11e8-9899-bcaecfa701f8?X-Amz-Algorithm=AWS4-HMAC-SHA256
&X-Amz-Credential=AKIAIWNJYAX4CSVEH53A%2F20190517%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20190517T023547Z&X-Amz-Expires=300&X-Amz-Signature=3227feaae0c6145300772bea
1567b7ffc9159dbfae2031ed8c66dbd6bff7a9f3&X-Amz-SignedHeaders=host&actor_id=0&response-content-disposition=attachment%3B%20filename%3Dliblsl-1.12.0-Linux-ARM7.deb&respon
se-content-type=application%2Foctet-stream
Resolviendo github-production-release-asset-2e65be.s3.amazonaws.com (github-production-release-asset-2e65be.s3.amazonaws.com)... 52.216.86.75
Conectando con github-production-release-asset-2e65be.s3.amazonaws.com (github-production-release-asset-2e65be.s3.amazonaws.com)[52.216.86.75]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 449988 (439K) [application/octet-stream]
Grabando a: “liblsl-1.12.0-Linux-ARM7.deb”

liblsl-1.12.0-Linux-ARM7.deb             100%[=======================================================================================>] 439,44K   585KB/s    in 0,8s

2019-05-16 22:35:49 (585 KB/s) - “liblsl-1.12.0-Linux-ARM7.deb” guardado [449988/449988]

root@raspberrypi:/home/pi/Desktop# apt-get install ./liblsl-1.12.0-Linux-ARM7.deb
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Nota, seleccionando «lsl-liblsl» en lugar de «./liblsl-1.12.0-Linux-ARM7.deb»
Se instalarán los siguientes paquetes NUEVOS:
  lsl-liblsl
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 0 B/450 kB de archivos.
Se utilizarán 2.178 kB de espacio de disco adicional después de esta operación.
Des:1 /home/pi/Desktop/liblsl-1.12.0-Linux-ARM7.deb lsl-liblsl armhf 1.12.0 [450 kB]
Seleccionando el paquete lsl-liblsl previamente no seleccionado.
(Leyendo la base de datos ... 133202 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../liblsl-1.12.0-Linux-ARM7.deb ...
Desempaquetando lsl-liblsl (1.12.0) ...
Configurando lsl-liblsl (1.12.0) ...
root@raspberrypi:/home/pi/Desktop#
```

Figura 7.8: Descarga e instalación de compilación ARM de pylsl

Una vez teniendo todo configurado se entró al directorio del proyecto que contiene el programa detector de caídas para poder instalarlo de forma global en el sistema.

```
Installed /usr/local/lib/python3.5/dist-packages/PyBluez-0.22-py3.5-linux-armv7l.egg
Searching for crc16>=0.1.1
Reading https://pypi.python.org/simple/crc16/
Downloading https://files.pythonhosted.org/packages/a6/e0/70a44c4385f2b33df82e518005aae16b5c1feaf082c73c0acebe3426fc0a/crc16-0.1.1.tar.gz#sha256=c1f86aa0390f4baf07d2631
b16b979580eae1d9a973a826ce45353a22ee8d396
Best match: crc16 0.1.1
Processing crc16-0.1.1.tar.gz
Writing /tmp/easy_install-gddc73es/crc16-0.1.1/setup.cfg
Running crc16-0.1.1/setup.py -q bdist_egg --dist-dir /tmp/easy_install-gddc73es/crc16-0.1.1/egg-dist-tmp-t25ea0s0
zip_safe flag not set; analyzing archive contents...
crc16.__pycache__.crc16.cpython-35: module references __file__
creating /usr/local/lib/python3.5/dist-packages/crc16-0.1.1-py3.5-linux-armv7l.egg
Extracting crc16-0.1.1-py3.5-linux-armv7l.egg to /usr/local/lib/python3.5/dist-packages
Adding crc16 0.1.1 to easy-install.pth file

Installed /usr/local/lib/python3.5/dist-packages/crc16-0.1.1-py3.5-linux-armv7l.egg
Searching for construct==2.5.2
Reading https://pypi.python.org/simple/construct/
Downloading https://files.pythonhosted.org/packages/fd/5a/e3105c79b3b0bbf7a80dc8218d7416df6551f9f2fe389c9ce6690a621c00/construct-2.5.2.tar.gz#sha256=665b6271eeadf15219c
726b180c8d7a841d026784d72ca3dad90a20aae009020
Best match: construct 2.5.2
Processing construct-2.5.2.tar.gz
Writing /tmp/easy_install-s1attekd/construct-2.5.2/setup.cfg
Running construct-2.5.2/setup.py -q bdist_egg --dist-dir /tmp/easy_install-s1attekd/construct-2.5.2/egg-dist-tmp-l9ek9egf
zip_safe flag not set; analyzing archive contents...
construct.__pycache__.debug.cpython-35: module MAY be using inspect.stack
creating /usr/local/lib/python3.5/dist-packages/construct-2.5.2-py3.5.egg
Extracting construct-2.5.2-py3.5.egg to /usr/local/lib/python3.5/dist-packages
Adding construct 2.5.2 to easy-install.pth file

Installed /usr/local/lib/python3.5/dist-packages/construct-2.5.2-py3.5.egg
Searching for pylsl==1.12.2
Best match: pylsl 1.12.2
Processing pylsl-1.12.2-py3.5.egg
pylsl 1.12.2 is already the active version in easy-install.pth

Using /usr/local/lib/python3.5/dist-packages/pylsl-1.12.2-py3.5.egg
Searching for six==1.12.0
Best match: six 1.12.0
Adding six 1.12.0 to easy-install.pth file

Using /usr/lib/python3/dist-packages
Finished processing dependencies for faros-streamer==1.0.0
```

Figura 7.9: Instalación programa detector de caídas

Ya teniendo el programa instalado, se identifico el dispositivo Faros 180° y luego se

pareo con la Raspberry Pi mediante bluetooth.

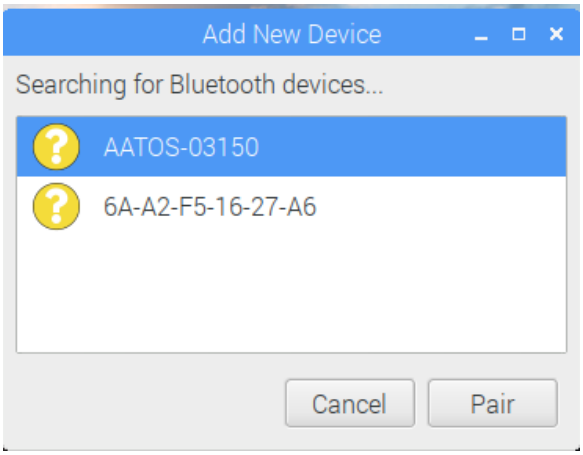

Figura 7.10: Identificación de Faros 180°

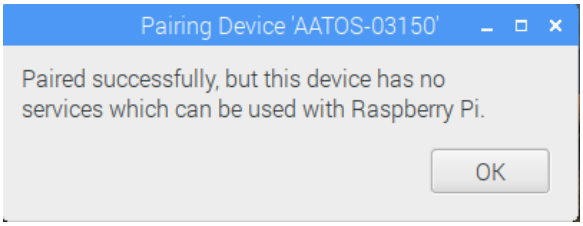

Figura 7.11: Faros 180° pareado con Raspberry Pi

Una vez que se tiene todo configurado e instalado se ejecuta el programa para comprobar su funcionamiento, completando así la implantación de éste.

```
root@raspberrypi:/home/pi/Desktop/faros fall detector# faros --scan
Scanning for available devices.
Found the following devices:
AATOS-03150     EC:FE:7E:12:CB:E0

No se ha dado ningún nombre o dirección MAC. No puede continuar.

Escriba faros --help para mostrar información de uso.

root@raspberrypi:/home/pi/Desktop/faros fall detector# faros --rut-adulto 18.458.148-5 --mac EC:FE:7E:12:CB:E0 --stream
Usando dispositivo con dirección MAC: EC:FE:7E:12:CB:E0
Conexión establecida.

Se detectaran caidas para el Rut: 18.458.148-5
Transmisión de datos. Escriba 'q' y luego 'Enter' para salir
 > caida libre 422
caida libre 576
caida libre 11
caida libre 5
impacto 7485
NO HUBO CAIDA

caida libre 357
caida libre 336
caida libre -6
qcaida libre -9


Transmisión detenida.

root@raspberrypi:/home/pi/Desktop/faros fall detector#
```

Figura 7.12: Ejecución de programa implantado en Raspberry Pi

Figura 7.13: Implantación en Raspberry

7.2.3. Aplicación móvil

Para poder implantar la aplicación móvil en la plataforma de Google Play, primero se debió generar una apk firmada. Para ello se utilizo el SDK Android Studio [41] en donde se realizo un build del proyecto para posteriormente poder firmar y generar la apk.

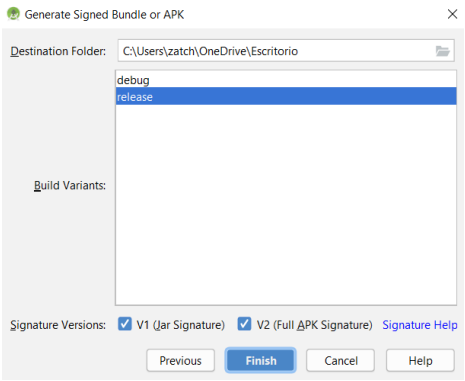

Figura 7.14: Build y firma de la Apk

Teniendo la apk creada se creó un documento html, que se encuentra en el directorio del web server, el cual tiene la política de privacidad de la aplicación que se implantara en la plataforma de Google Play. La política de privacidad es muy importante ya que la plataforma puede deshabilitar nuestra aplicación si no la tenemos al momento de subirla. Ya implantada la aplicación se verificó que ésta se encontrara en la plataforma de Google Play, finalizando así el proceso de implantación.

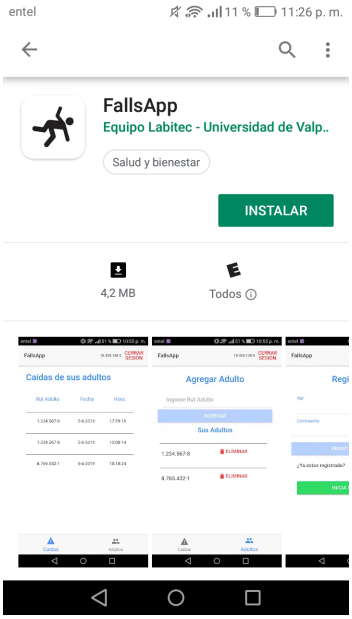

Figura 7.15: Aplicación en la plataforma de Google Play

Capítulo 8

Conclusiones

Las caídas son un grave riesgo que tienen los adultos mayores, ya que estas pueden llegar a generarles daños físicos como lo son las fracturas y también daño emocional al sentirse que son una carga para su entorno. Además, al no contar con herramientas que puedan detectar estas caídas, generan el temor de tener todas estas complicaciones y no poder sentirse seguros al realizar cualquier actividad en su vida diaria.

Se realizó un sistema que pueda detectar estas caídas en los adultos mayores, utilizando un dispositivo vestible, que cuenta con un sensor de movimiento (en este caso el acelerómetro), el cual se conecta a través de bluetooth a una raspberry pi y envía los registros de cuando ocurre una caída a una base de datos. Para poder detectar estas caídas se implemento un algoritmo basado en el método de umbrales, el cual detecta cada fase que experimenta una caída y decide si esta efectivamente es una o no. Por otro lado se realizo una aplicación móvil enfocada en los cuidadores o familiares de los adultos mayores que quieran registrarlos y monitorizar si tuviesen alguna caída, enviándoles una notificación en caso de tenerlas.

El haber creado este sistema favorece enormemente a los adultos mayores y a su entorno, ya que, se puede ver disminuido el tiempo en que ellos recibirán ayuda sin que pasen a tener a mayores complicaciones. Además el saber que un cuidador recibirá una alerta a la hora de que el sistema detecte una caída, permitirá que los adultos mayores puedan tener mas libertad al realizar sus actividades de la vida diaria, sin tener que estar siempre a su lado. También que este sistema use dispositivos vestibles de bajo costo ayuda a que pueda ser adquirible para todos aquellos que requieran utilizarlo.

Los resultados del sistema desarrollado fueron exitosos ya que se logró validar cada prueba de forma satisfactoria. Se logró la integración del sistema completo desde el envió de datos, luego detectar si es o no una caída, registrarla en la base de datos y notificar y mostrar un histórico al cuidador sobre esta en la aplicación móvil. Además, el algoritmo se implemento de forma eficiente ya que es capaz de diferenciar entre lo que era una caída y una actividad de la vida diaria.

Si bien el trabajo de realizo de forma exitosa, en un principio la principal limitante de este trabajo fue obtener un dispositivo vestible que pudiese transmitir datos y que además contara con algún sensor como lo es el acelerómetro. Sin embargo, al encontrar el Faros 180° y el software faros-streamer-2, se pudo realizar la transmisión de datos de forma correcta, lo que dio paso a poder crear el algoritmo que detecta caídas, que era lo mas importante de todo el sistema. Otra limitante fue que al realizar las pruebas con usuarios para poder obtener el rendimiento, específicamente para obtener los resultados de sensibilidad del algoritmo, estas no se podían realizar con adultos mayores reales ya que se pondría en peligro su salud al realizar estas pruebas, por lo que se tuvo que buscar voluntarios que estuviesen dispuestos a simular una caída.

8.1. Trabajo a futuro

Como trabajo a futuro, el algoritmo de detección podría ser implementado en un machine learning que aprenda de las caídas, reemplazando el método de los umbrales. Además poder utilizar otro dispositivo vestible que pueda enviar la ubicación u otra información adicional seria muy útil para los cuidadores. También incrementar la muestra para las pruebas de rendimiento, especialmente para calcular la sensibilidad y especificidad del algoritmo, ya que al tener una muestra mas grande aumentan las posibilidades de obtener un porcentaje mas preciso. No olvidar además que todo programa es escalable, este sistema no puede ser la excepción.

Para finalizar, tomando en cuenta el alcance que tiene todo el tema de las detecciones de caídas en los adultos mayores con todas las complicaciones que estas producen, lo ideal sería que este trabajo de título sirva como base para crear sistemas similares que puedan seguir ayudando en esta problemática o bien tomar lo propuesto y mejorar lo mas que se pueda este sistema.

Apéndice A

Manual de Usuario

El siguiente apéndice, tiene la finalidad de enseñar el uso del programa detector de caídas, agregando también el funcionamiento del dispositivo Faros 180° con el objetivo de poder registrar las caídas del adulto mayor que lo porte. Además se explicara como instalar y usar la aplicación móvil en cualquier dispositivo móvil que posea Android.

A.1. Aplicación móvil

1. **Instalación**
   En la primera instancia se debe ingresar a la Play Store de Google Play que esta disponible en todos los dispositivos con Sistema Operativo Android. Una vez que entramos, dentro de la barra de búsqueda ingresamos el nombre de la aplicación desarrollada ”FallsApp”, como se muestra en la Figura A.1.

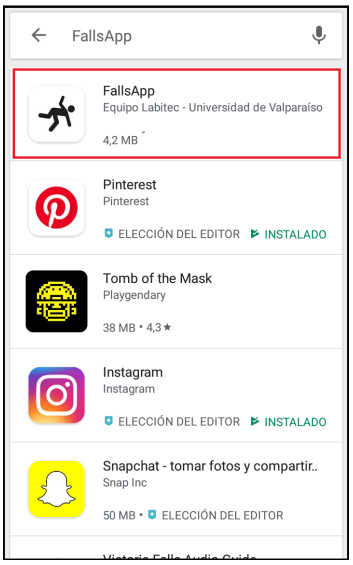

Figura A.1: Búsqueda de aplicación en Google Play

Una vez que estemos en la página de la aplicación, se debe presionar el botón Instalar como se muestra en la Figura A.2.

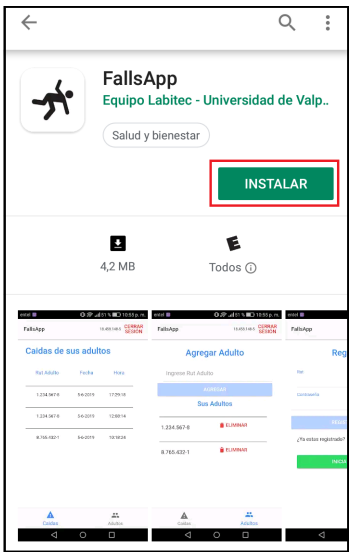

Figura A.2: Instalación de aplicación en Google Play

Una vez instalada ya solo resta abrir la aplicación.

2. **Aplicación**
   Una vez instalada la aplicación, hay que buscar su icono que se encuentra en la sección de aplicaciones del dispositivo como se puede ver en la Figura A.3.

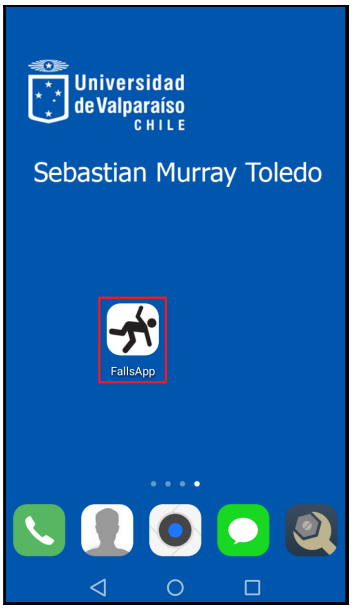

Figura A.3: Icono de la aplicación en el dispositivo

Una vez abierta la aplicación se tendrá acceso a la pantalla de inicio de sesión en donde usted podrá ingresar para poder administrar las caídas de sus adultos mayores. En caso de que usted este accediendo por primera vez sin haberse registrado, en la misma pantalla de inicio de sesión se puede acceder a la pantalla de registro.

Figura A.4: Pantalla inicio de sesión

Como se puede apreciar en la Figura A.4 usted podrá realizar las siguientes acciones en esta pantalla:

- 1: Iniciar sesión: Luego de ingresar su rut y contraseña podrá iniciar sesión.
- 2: Registrarse: Accederá a la pantalla de registro, para registrarse y posteriormente volver a iniciar sesión.

En caso de haber accedido a la pantalla de registro, usted deberá proporcionar sus datos para iniciar sesión.

Figura A.5: Pantalla de registro

Como se puede apreciar en la Figura A.5 usted podrá realizar las siguientes acciones en esta pantalla:

- 1: Registrase: Luego de proporcionar sus datos (rut y contraseña) podrá registrarse.
- 2: Registrarse: Accederá a la pantalla de inicio de sesión.

Cuando usted inicie sesión accederá a la administración de sus adultos mayores.

Figura A.6: Acceso a administración de adultos mayores

Como se ve en la Figura A.6 usted podrá:

- 1: Cerrar sesión: Cerrara sesión para luego poder acceder con otra cuenta o la misma.
- 2: Acceder a adultos: Accederá a la pantalla para administrar sus adultos mayores.
- 3: Acceder a caídas: Accederá a la pantalla para ver registro de caídas.

En la pantalla de adultos, usted podrá agregar el rut del adulto mayor que usted quiera administrar (ver Figura A.7). Cuando lo haya agregado, le saldrá un mensaje indicando que se ha agregado ese adulto y que ahora usted podrá ver el registro de sus caídas, como también recibir las notificaciones de cuando ese adulto mayor tenga caídas (ver Figura A.8).

Figura A.7: Agregar un adulto mayor

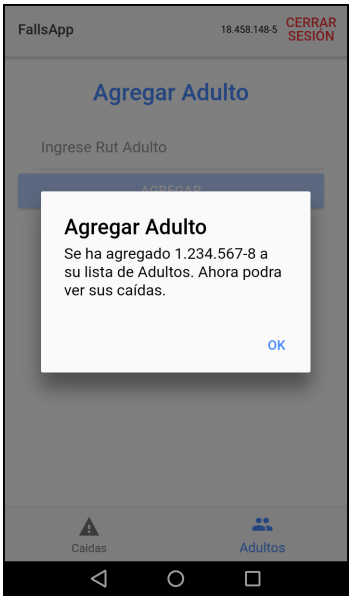

Figura A.8: Aviso de que ha agregado a un adulto mayor

Cuando usted agrega un adulto podrá, en la pantalla de caídas, podrá ver el registro de las caídas de su adulto mayor (ver Figura A.9). También podrá comenzar a recibir

notificaciones cada vez que los adultos mayores que administra tengan una caída (ver Figura A.10).

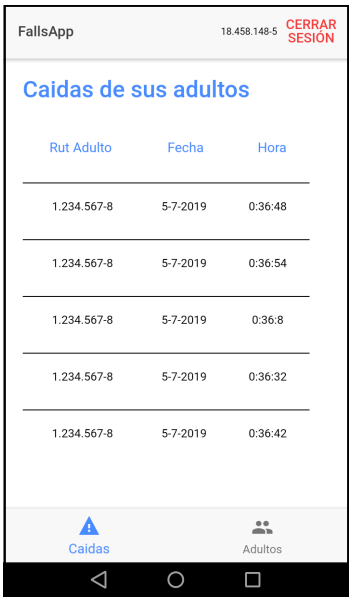

Figura A.9: Registro de caídas de sus adultos mayores

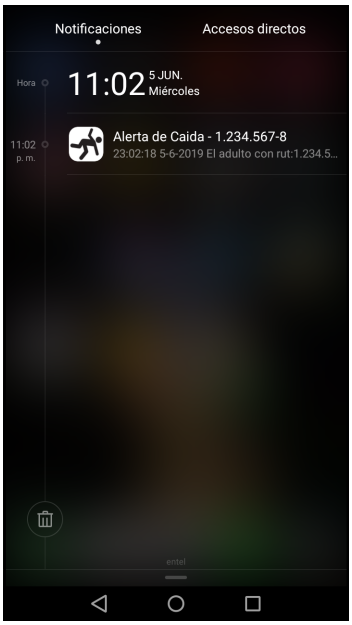

Figura A.10: Notificaciones cuando un adulto mayor tiene una caída

Si usted quiere dejar de ver el registro y de recibir notificaciones de las caídas de algún adulto mayor en específico, en la pantalla de adultos puede presionar el botón eliminar al lado del rut del adulto mayor como se puede ver en la Figura A.11. Al presionar el botón le aparece un aviso preguntándole si esta seguro que desea eliminarlo. Si presiona el botón ”si”, se eliminará todo registro de caídas y además dejará de obtener las notificaciones de las caídas asociadas ese rut. Si presiona ”no”simplemente se cancela la acción de eliminar a ese adulto mayor.

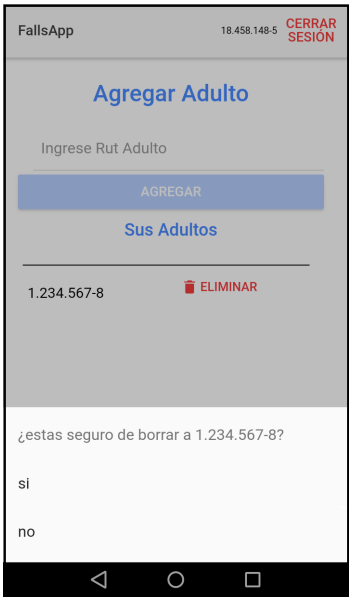

Figura A.11: Eliminar un adulto mayor

A.2. Detector de caídas

1. **Requisitos**
   Los requisitos de hardware que son necesarios para poder utilizar el programa detector de caídas son:

   - Un mouse y un teclado, cada uno con conexión usb.
   - 1 monitor o pantalla de televisión, con conexión hdmi.
   - 1 cable hdmi.
   - 1 raspberry pi, con su cargador, que cuente con el programa detector de caídas instalado previamente.
   - 1 dispositivo Faros 180°.

   Principalmente el mouse y el teclado serán utilizados como los periféricos para poder moverse dentro del sistema operativo de las raspberry pi, y así acceder al detector de caídas. El cable hdmi y el monitor serán utilizados para visualizar la interfaz gráfica de la raspberry pi. El dispositivo Faros 180° es el encargado de enviar los datos del acelerómetro, mediante conexión bluetooth, a el programa dentro de la raspberry pi, en el cual se encuentra el algoritmo que detectara las caídas según los datos recibidos. En la Figura A.12 se puede ver un esquema general de como se debe tener toda la conexión.

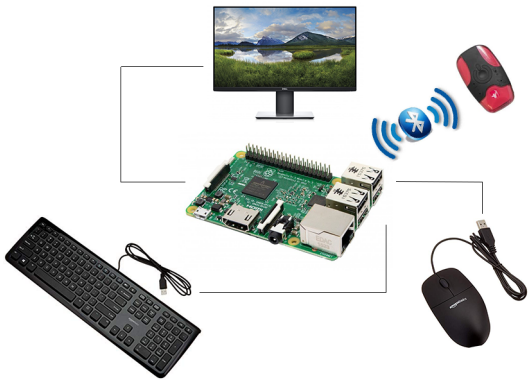

Figura A.12: Esquema conexión raspberry pi

2. **Faros 180°**
   El programa detector de caídas utiliza el dispositivo Faros 180°. A continuación, se darán a conocer algunas de las funcionalidades básicas que se deben saber sobre este dispositivo para utilizarlas en el programa detector de caídas. Si usted desea tener un conocimiento mas avanzado sobre él, puede revisar el **manual oficial de Faros 180°** [42].

- **Comportamiento de Faros 180°** En la Figura A.13 se puede apreciar los símbolos e indicadores del dispositivo que son utilizados.

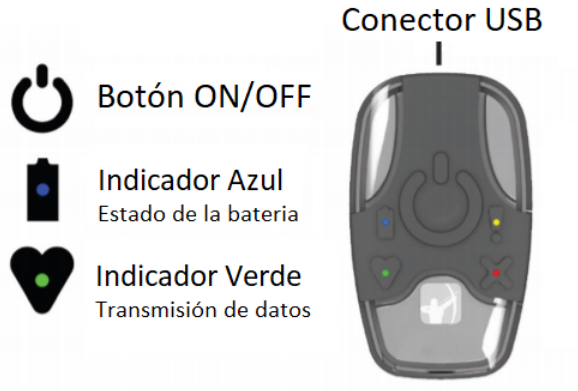

Figura A.13: Símbolos e indicadores Faros 180°

El indicador verde nos muestra el estado del dispositivo en cuanto a si esta encendido o apagado mientras que el indicador azul nos muestra el estado de la batería.

Para prender el dispositivo bastara con apretar el botón central (Boton ON/OFF) y el indicador verde comenzara a parpadear, eso significa que el dispositivo está listo para transmitir datos.

Para apagar el dispositivo se tendrá que mantener el botón central presionado por mas de 5 segundos y el indicador verde dejara de parpadear.

Cuando el dispositivo está cargándose el indicador azul estará parpadeando.

Cuando el indicador azul se mantenga iluminado significa que la carga esta completa.

- **Parear Faros 180° con Raspberry Pi**
  Para parear el dispositivo usted deberá primero prenderlo, luego en la raspberry pi deberá ir a la sección de bluetooth y presionar Add Device. Una vez lo encuentre presiona pair y el dispositivo quedara pareado tal como se ve en la Figura A.14.

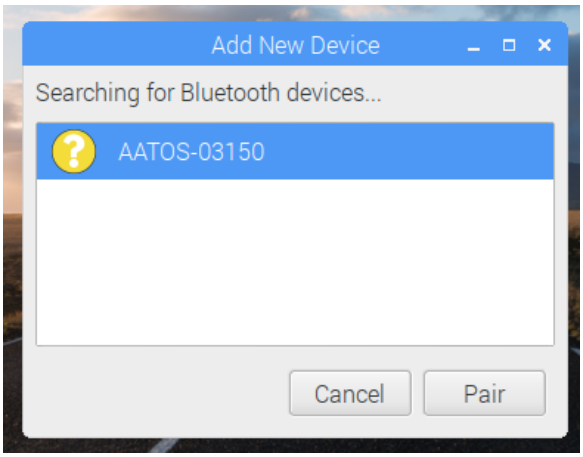

Figura A.14: Parear Faros 180° con raspberry pi

3. **Programa detector de caídas**
   El programa detector de caídas es básicamente un programa de consola, el cual se comunica con el servidor cada vez que se detecta una caída. En al Figura A.15 puede ver como acceder a la consola en raspberry pi, presionando el icono en la esquina superior izquierda de la pantalla.

Figura A.15: Acceder a consola en raspberry pi

El programa detector de caídas funciona escribiendo comandos en la consola el cual permite que el programa acceda a distintas funciones. A continuación, se listaran las funciones que tiene el programa detector de caídas:

- **Buscar dispositivo disponible**
  Para poder buscar los dispositivos que la raspberry pi tiene pareados con el fin de tener conocimiento de su MAC de bluetooth, se utiliza el comando **faros –scan**. Como se ve en la Figura A.16 este comando escanea los dispositivos y muestra en pantalla el nombre y su MAC. Generalmente los dispositivos Faros 180° siempre tienen un nombre “AATOS-XXXXX” y MAC “AA:BB:CC:11:22:33”.

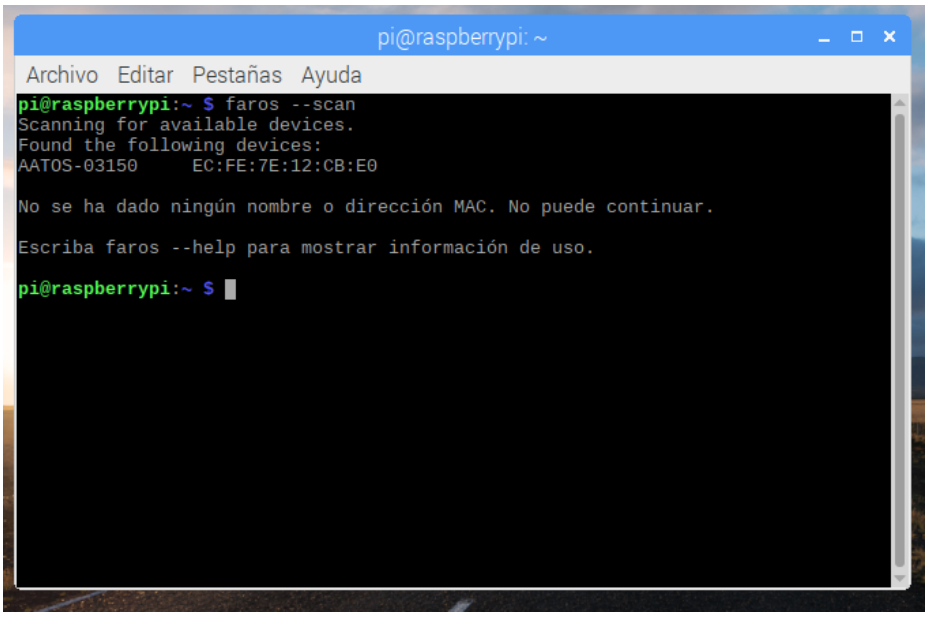

Figura A.16: Escanear Faros 180°

- **Parpadear las luces del dispositivo**
  En caso de que quiere comprobar que efectivamente la MAC del dispositivo escaneado es efectivamente la de su dispositivo usted puede utilizar el comando **faros –blink –mac AA:BB:CC:11:22:33**. Este comando hará que su dispositivo, según la MAC escrita, prenda todas las luces lo cual podrá indicarle que efectivamente ese es el dispositivo que usted quiere utilizar.
- **Detección de caídas**
  Para comenzar a detectar caídas utilizara el comando **faros –rut-adulto 1.234.567-5 –mac AA:BB:CC:11:22:33 –stream**. Este comando pide como parámetros el rut del adulto mayor que usted quiere monitorizar, como también la MAC del dispositivo que el adulto mayor va a portar, la opción **–stream** indica que el dispositivo comenzara a transmitir datos. Como se puede ver en la Figura A.17 al escribir ese comando se han comenzado a monitorizar las caídas para el rut 18.458.148-5 y para el dispositivo con MAC: EC:FE:7E:12:CB:E0.

```
pi@raspberrypi:~ $ faros --rut-adulto 18.458.148-5 --mac EC:FE:7E:12:CB:E0 --stream
Usando dispositivo con dirección MAC: EC:FE:7E:12:CB:E0
Conexión establecida.

('Se detectaran caidas para el Rut:', '18.458.148-5')
Transmisión de datos. Escriba 'q' y luego 'Enter' para salir
 > ('caida libre', -125)
('caida libre', -15)
('caida libre', 174)
('caida libre', 177)
('caida libre', 159)
```

Figura A.17: Detección de caídas

Apéndice B

Documentación de desarrollador

En este apéndice se presenta la documentación orientada para que un desarrollador pueda tener conocimiento de la implementación general de este sistema. Se especifica como preparar el ambiente en la raspberry pi para así hacer uso del programa detector de caídas, la implementación de las notificaciones push, la api para poder conectar el cliente con el servidor y se agrega además un medio de contacto para poder resolver cualquier interrogante.

B.1. Preparación del ambiente en raspberry pi

En una raspberry pi actualizada debe instalar la librería de bluetooth ya que la librería pybluez depende de ella. También tiene que instalar la librería pylsl (labstreaminglayer), pero debe tener en cuenta que para que esta funcione en una rapsberry se debe utilizar una compilación de pylsl para arquitectura ARM.

Esta se puede descargar utilizando el comando **wget** desde:

wget https://github.com/labstreaminglayer/liblsl/releases/download/1.12/liblsl-1.12.0-Linux-ARM7.deb.

Luego, instalarla con el comando: **apt-get install ./liblsl-1.12.0-Linux-ARM7.deb**.

Ya tiendo todos los requisitos instalados, debe entrar al directorio del proyecto en donde se encuentra el programa detector de caídas y con el comando **python3 setup.py install** se instalara. Tenga en cuenta que si usted realiza una modificación en el código fuente del programa, deberá instalarlo nuevamente, ya que se instala de manera global en la raspberry pi.

B.2. Implementación de notificaciones push

Para realizar las notificaciones push se utilizo Cloud Messaging de Firebase de Google.

- **Cliente**
  Por el lado del cliente se debe crear un proyecto en la consola de firebase de Google y luego registrar una nueva app.

  El nombre del paquete de Android debe ser le mismo que se encuentra en el archivo **config.xml**.

  Luego debe descargar el archivo **google-services.json** y copiarlo en el directorio raíz de el proyecto.

  Posteriormente se tiene que implementar la generación de tokens para que, según la lógica que usted desee aplicar, se guarden en la base de datos para cada usuario que inicia sesión.

- **Servidor**

  Por el lado del servidor, se debe ir a configuración del proyecto en la consola de firebase de Google y luego ir a cuentas de servicio.

  Debe **generar una nueva clave privada** y dejarla en el directorio del proyecto.

  Para poder realizar las conexiones con el servicio de Cloud Messaging, Firebase provee un **"Fragmento de configuración de Admin SDK"** el cual se implementa en el código fuente del servidor indicando la ruta de la clave privada que descargo.

B.3. API

Para generar una comunicación entre el cliente y el servidor se listaran las rutas de la API para tener un mayor conocimiento de estas al momento de implementarlas.

- **Seleccionar adultos**

  Método: GET
  Ruta: /adultos/:rut/

- **Seleccionar todos los adultos**

  Método: GET
  Ruta: /adultos/

- **Insertar adulto**

  Método: POST
  Ruta: /adultos/

- **Borrar adulto**

  Método: DELETE
  Ruta: /adultos/:id/

- **Registrar cuidador**

  Método: POST
  Ruta: /registro/

- **Seleccionar cuidadores**

  Método: GET
  Ruta: /registro/

- **Seleccionar cuidador**

  Método: GET
  Ruta: /registro/:id/

- **Login**

  Método: POST
  Ruta: /auth/login/

- **Guardar token FCM**

  Método: PUT
  Ruta: /fcmtoken/

- **Insertar caída**

  Método: GET
  Ruta: /insertarcaida/

- **Notificación push**

  Método: GET
  Ruta: /notificaciones/:adulto

- **Política de privacidad**

  Método: GET
  Ruta: /privacy_policy/

Estas rutas se deben implementar tanto en el cliente como en el código fuente del programa detector de caídas (en el caso de insertar las caídas).

Además, las rutas necesitan de una autorización **'Content-Type': 'application/json'**, **'Authorization':'Bearer '+ token**.

De momento, las siguiente rutas no necesitan de un autorización para acceder a ellas:

- **Login**
- **Registrar cuidador**
- **Seleccionar cuidadores**
- **Seleccionar cuidador**
- **Notificación push**
- **Insertar caída**
- **Política de privacidad**

B.4. Contacto para el desarrollador

En caso de cualquier duda con respecto al desarrollo de este sistema, los datos con que usted puede contactar al desarrollador son los siguientes:

Nombre: Sebastian Murray Toledo

Correo: sebastian.murray@alumnos.uv.cl

Bibliografía

[1] L.-J. Kau and C.-S. Chen, "A smart phone-based pocket fall accident detection, positioning, and rescue system," *IEEE journal of biomedical and health informatics*, vol. 19, no. 1, pp. 44–56, 2015.

[2] K. Yildirim, G. Ucar, T. Keskin, and A. Kavak, "Fall detection using smartphone-based application," *International Journal of Applied Mathematics, Electronics and Computers*, vol. 4, no. 4, pp. 140–144, 2016.

[3] I. Maglogiannis, C. Ioannou, and P. Tsanakas, "Fall detection and activity identification using wearable and hand-held devices," *Integrated Computer-Aided Engineering*, vol. 23, no. 2, pp. 161–172, 2016.

[4] E. Casilari and M. A. Oviedo-Jiménez, "Automatic fall detection system based on the combined use of a smartphone and a smartwatch," *PloS one*, vol. 10, no. 11, p. e0140929, 2015.

[5] Q. T. Huynh, U. D. Nguyen, L. B. Irazabal, N. Ghassemian, and B. Q. Tran, "Optimization of an accelerometer and gyroscope-based fall detection algorithm," *Journal of Sensors*, vol. 2015, 2015.

[6] D. of Economic and S. A. in Population Division of the United Nations., "World population ageing," https://www.un.org/en/development/desa/population/publications/pdf/ageing/WPA2015_Report.pdf, 2015.

[7] W. H. Organization, "Who global report on falls prevention in older age," https://apps.who.int/iris/bitstream/handle/10665/43811/9789241563536_eng.pdf, 2008.

[8] M. E. Tinetti, "Predictors and prognosis of inability to get up after falls among elderly persons," *JAMA: The Journal of the American Medical Association*, vol. 269, no. 1, p. 65, Jan. 1993.

[9] A. Rocha, A. Martins, J. C. F. Junior, M. N. K. Boulos, M. E. Vicente, R. Feld, P. van de Ven, J. Nelson, A. Bourke, G. ÓLaighin *et al.*, "Innovations in health care

services: The caalyx system," *International journal of medical informatics*, vol. 82, no. 11, pp. e307–e320, 2013.

[10] Apple Chile. Apple watch series 4. [Online]. Available: https://www.apple.com/cl/apple-watch-series-4/

[11] "Wearable technology and wearable devices: Everything you need to know," http://www.wearabledevices.com/what-is-a-wearable-device/.

[12] "What is sensor? - definition from whatis.com," https://whatis.techtarget.com/definition/sensor.

[13] "Inertial sensors - dr. kostas alexis," http://www.kostasalexis.com/inertial-sensors.html.

[14] "Analog devices : Mems technology - basic mems terminology," http://www.gmsystems.com/uploads/3/1/4/3/3143302/glossary_terms.pdf.

[15] "Guide for the use of the international system of units (si)," https://physics.nist.gov/cuu/pdf/sp811.pdf.

[16] "Detecting human falls with a 3-axis digital accelerometer," http://www.t-es-t.hu/download/analog/ad43-3.pdf.

[17] P. Vallabh, R. Malekian, N. Ye, and D. C. Bogatinoska, "Fall detection using machine learning algorithms." pp. 1–9, 2016.

[18] H. Gjoreski, J. Bizjak, and M. Gams, "Using smartwatch as telecare and fall detection device," pp. 242–245, 2016.

[19] M. Kangas, I. Vikman, J. Wiklander, P. Lindgren, L. Nyberg, and T. Jämsä, "Sensitivity and specificity of fall detection in people aged 40 years and over," *Gait & posture*, vol. 29, no. 4, pp. 571–574, 2009.

[20] P. Pierleoni, A. Belli, L. Palma, M. Pellegrini, L. Pernini, and S. Valenti, "A high reliability wearable device for elderly fall detection," *IEEE Sensors Journal*, vol. 15, no. 8, pp. 4544–4553, 2015.

[21] M. Kangas, A. Konttila, I. Winblad, and T. Jamsa, "Determination of simple thresholds for accelerometry-based parameters for fall detection," pp. 1367–1370, 2007.

[22] D. Lim, C. Park, N. H. Kim, S.-H. Kim, and Y. S. Yu, "Fall-detection algorithm using 3-axis acceleration: combination with simple threshold and hidden markov model," *Journal of Applied Mathematics*, vol. 2014, 2014.

[23] H. Nguyen, F. Mirza, M. A. Naeem, and M. M. Baig, "Detecting falls using a wearable accelerometer motion sensor," pp. 422–431, 2017.

[24] J. J. Villacorta, M. I. Jiménez, L. d. Val, and A. Izquierdo, "A configurable sensor network applied to ambient assisted living," *Sensors*, vol. 11, no. 11, pp. 10 724–10 737, 2011.

[25] M. N. Alkhomsan, M. A. Hossain, S. M. M. Rahman, and M. Masud, "Situation awareness in ambient assisted living for smart healthcare," *IEEE Access*, vol. 5, pp. 20 716–20 725, 2017.

[26] M. A. Quintana-Suárez, D. Sánchez-Rodríguez, I. Alonso-González, and J. B. Alonso-Hernández, "A low cost wireless acoustic sensor for ambient assisted living systems," *Applied Sciences*, vol. 7, no. 9, p. 877, 2017.

[27] "Use fall detection with apple watch series 4 - apple support," https://support.apple.com/en-us/HT208944.

[28] I. Sommerville, *Ingeniería del software*. Pearson Educación, 2005.

[29] "emotion mobile faros sensor for ecg, hrv. high resolution, small and lightweight." http://ecg.biomation.com/faros.htm.

[30] "Welcome to python.org," https://www.python.org/.

[31] "Build amazing native apps and progressive web apps with ionic framework and angular," https://ionicframework.com/.

[32] J. M. B. Douglas G Altman, "Diagnostic tests 1: sensitivity and specificity," https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2540489/pdf/bmj00444-0038.pdf.

[33] "bwrc/faros-streamer-2: Stream data from a mega electronics ltd. faros device using the lab streaming layer," https://github.com/bwrc/faros-streamer-2.

[34] "Mysql," https://www.mysql.com/.

[35] "Express - node.js web application framework," http://expressjs.com/.

[36] "Node.js," https://nodejs.org/es/.

[37] "Firebase," https://firebase.google.com/.

[38] "The gof design patterns reference," http://w3sdesign.com/GoF_Design_Patterns_Reference0100.pdf.

[39] "Google play," https://developer.android.com/distribute/console?hl=es.

[40] “Tmux: esteroides para tu terminal,” https://hipertextual.com/archivo/2014/09/tmux/.

[41] “Android studio,” https://developer.android.com/studio.

[42] “800778 emotion faros series manual,” https://ecgcloud.co.uk/software/800778-2.3.0%20eMotion%20Faros%20Series%20Manual.pdf.