



**UNIVERSIDAD DE CASTILLA-LA MANCHA**  
**ESCUELA SUPERIOR DE INFORMÁTICA**

**GRADO EN INGENIERÍA INFORMÁTICA**

**Tecnologías de la Información**

**TRABAJO FIN DE GRADO**

Monitorización y visualización de hábitos y comportamiento a partir del uso de dispositivos móviles

Elena Díaz del Campo González-Gallego

Julio, 2020





**UNIVERSIDAD DE CASTILLA-LA MANCHA  
ESCUELA SUPERIOR DE INFORMÁTICA**

**Tecnologías y Sistemas de Información**

**Tecnologías de la Información**

**TRABAJO FIN DE GRADO**

**Monitorización y visualización de hábitos y  
comportamiento a partir del uso de dispositivos móviles**

Autor: Elena Díaz del Campo González-Gallego

Tutor: Ramón Hervás Lucas

Co-Tutor: Luis Cabañero Gómez

Julio, 2020

Monitorización y visualización de hábitos y comportamiento a partir del uso de dispositivos móviles  
© Elena Díaz del Campo González-Gallego, 2020

Este documento se distribuye con licencia CC BY-NC-SA 4.0. El texto completo de la licencia puede obtenerse en <https://creativecommons.org/licenses/by-nc-sa/4.0/>.

La copia y distribución de esta obra está permitida en todo el mundo, sin regalías y por cualquier medio, siempre que esta nota sea preservada. Se concede permiso para copiar y distribuir traducciones de este libro desde el español original a otro idioma, siempre que la traducción sea aprobada por el autor del libro y tanto el aviso de copyright como esta nota de permiso, sean preservados en todas las copias.

Este texto ha sido preparado con la plantilla L<sup>A</sup>T<sub>E</sub>X de TFG para la UCLM publicada por Jesús Salido en GitHub<sup>1</sup> y Overleaf<sup>2</sup> como parte del curso «L<sup>A</sup>T<sub>E</sub>X esencial para preparación de TFG, Tesis y otros documentos académicos» impartido en la Escuela Superior de Informática de la Universidad de Castilla-La Mancha.



---

<sup>1</sup>[https://github.com/JesusSalido/TFG\\_ESI\\_UCLM](https://github.com/JesusSalido/TFG_ESI_UCLM)

<sup>2</sup><https://www.overleaf.com/latex/templates/plantilla-de-tfg-escuela-superior-de-informatica-uclm/phjgscmfqtsw>

**TRIBUNAL:**

Presidente: \_\_\_\_\_

Vocal: \_\_\_\_\_

Secretario: \_\_\_\_\_

**FECHA DE DEFENSA:** \_\_\_\_\_

**CALIFICACIÓN:** \_\_\_\_\_

PRESIDENTE

VOCAL

SECRETARIO

Fdo.:

Fdo.:

Fdo.:



*A ti Raúl,  
por tu amor, paciencia, apoyo y comprensión*



## **Resumen**

El impacto de las Tecnologías de la Información y la Comunicación (TIC) en nuestra sociedad aporta numerosas ventajas que permiten mejorar la vida diaria de las personas. Adicionalmente, el uso de las TICs en nuestra vida cotidiana sirve para caracterizar nuestro comportamiento, hábitos y habilidades.

En el presente Trabajo de Fin de Grado (TFG) se propone monitorizar y caracterizar acciones y eventos realizados en un dispositivo móvil, mediante el desarrollo de una aplicación que contiene varios servicios proporcionados por Android, con el objetivo de caracterizar la interacción con el dispositivo que realizamos en nuestra vida cotidiana. La información monitorizada ha sido almacenada en bruto y visualizada de forma que, tanto el usuario como agentes externos, puedan analizar los hábitos y comportamientos. Esta aplicación, denominada BIPapp, es parte de un proyecto de investigación que pretende diagnosticar de forma temprana el deterioro cognitivo leve analizando cambios en el comportamiento. La aplicación, en cualquier caso, tiene múltiples usos. Por ejemplo, podría ser usada también para detectar adicciones tecnológicas o malos hábitos en el uso de las nuevas tecnologías.

En este proyecto se ha hecho uso de los servicios de accesibilidad de Android para analizar las interacciones y eventos en el sistema, almacenar la información en bases de datos remotas y aplicar técnicas de visualización en forma de gráficos para mostrar los distintos aspectos de la vida cotidiana, filtrados por tiempo y tipo. Además, y de forma innovadora, se recoge también la actividad física realizada (distinguiendo entre en reposo, andando o corriendo) de forma que se puede correlacionar el uso del móvil con la actividad.



## **Abstract**

The impact of Information and Communication Technologies (ICT) on our society brings many advantages that allow us to improve people's daily lives. Additionally, the use of ICTs in our daily lives serves to characterize our behavior, habits and skills.

In this End of Degree Project (TFG) we propose to monitor and characterize actions and events performed on a mobile device, through the development of an application that contains several services provided by Android, with the aim of characterizing the interaction with the device that we perform in our daily lives. The monitored information has been stored in raw and visualized so that both the user and external agents can analyze the habits and behaviors. This application, called BIPapp, is part of a research project that aims to diagnose early mild cognitive impairment by analyzing changes in behavior. The application, in any case, has multiple uses. For example, it could also be used to detect technological addictions or bad habits in the use of new technologies.

In this project we have made use of Android's accessibility services to analyze interactions and events in the system, store the information in remote databases and apply visualization techniques in the form of graphs to show the different aspects of daily life, filtered by time and type. In addition, and in an innovative way, the physical activity performed is also collected (distinguishing between resting, walking or running) so that the use of the mobile phone can be correlated with the activity.



# AGRADECIMIENTOS

---

“Hay sueños que al comienzo nos parecen IMPOSIBLES, luego IMPROBABLES y si nos comprometemos seriamente, se vuelven INEVITABLES” Mahatma Gandhi.

La finalización de este proyecto pone fin a una etapa muy importante de mi vida, ya que he conseguido cumplir uno de mis deseos desde hace muchos años. Han sido cuatro años de gran esfuerzo personal, entrega, satisfacción y dedicación, pero como dice el dicho “Es de bien nacidos, ser agradecidos”, y no podría olvidarme de todas las personas que han estado a mi alrededor, aguantando mis flaquezas, pero también compartiendo mis éxitos.

En primer lugar, dar las gracias a mis dos tutores, Ramón Hervás y Luis Cabañero, por su confianza en mí para realizar este proyecto. Gracias por vuestra orientación, implicación y apoyo en el desarrollo de este TFG.

A mis padres, a los que siempre les estaré eternamente agradecida, porque sin ellos no habría conseguido llegar hasta aquí. Por todo el apoyo y confianza que habéis depositado en mí, por sufrir conmigo en los peores momentos de agobio. Sois para mí ejemplo de constancia, dedicación, fortaleza e inspiración diaria.

A mi hermano, el mayor regalo que me dieron mis padres y que sé, que desde el silencio, me ha apoyado y me ha hecho saber que soy muy fuerte y que nada puede conmigo.

A ti, Raúl, por ser sin duda la persona que más me ha acompañado y apoyado desde que empecé la carrera. Solo tú sabes lo que me ha costado llegar hasta aquí y el tiempo que he sacrificado en mi vida.

A mi compañera Sara, por todos los momentos que hemos pasado tanto buenos como malos y por esas tardes de agobio en la ALU que nos han hecho más fuertes.

A mis amigos, familiares, compañeros de piso y todos aquellos que me habéis acompañado en esta etapa.

*Elena Díaz del Campo González-Gallego  
Ciudad Real, 2020*



# ÍNDICE GENERAL

---

<b>Resumen</b>	<b>IX</b>
<b>Abstract</b>	<b>XI</b>
<b>Agradecimientos</b>	<b>XIII</b>
<b>Índice de figuras</b>	<b>XIX</b>
<b>Índice de tablas</b>	<b>XXI</b>
<b>Índice de listados</b>	<b>XXIII</b>
<b>Lista de acrónimos</b>	<b>XXV</b>
<b>Glosario de términos</b>	<b>XXVII</b>
<b>1. Introducción</b>	<b>1</b>
1.1. Contexto . . . . .	1
1.2. Motivación . . . . .	2
1.3. Solución propuesta . . . . .	2
1.4. Estructura del documento . . . . .	5
<b>2. Objetivos</b>	<b>7</b>
2.1. Objetivo general . . . . .	7
2.2. Objetivos específicos . . . . .	7
2.3. Objetivos teóricos y prácticos . . . . .	7
2.4. Medios tecnológicos utilizados . . . . .	8
2.4.1. Medios <i>software</i> . . . . .	8
2.4.2. Medios <i>hardware</i> . . . . .	10
2.4.3. Lenguajes de programación . . . . .	10
2.4.4. Servicios externos y almacenamiento remoto . . . . .	11
2.4.5. Otras herramientas . . . . .	12
<b>3. Antecedentes</b>	<b>13</b>
3.1. Interacción Persona - Dispositivo Móvil . . . . .	13
3.2. Monitorización con teléfonos móviles . . . . .	15
3.2.1. Actividad física . . . . .	15
3.2.2. Interacción con el dispositivo . . . . .	16
3.2.3. Conclusiones . . . . .	19
<b>4. Metodología</b>	<b>21</b>
4.1. Metodología de trabajo . . . . .	21

4.1.1. Manifiesto ágil . . . . .	21
4.1.2. Kanban . . . . .	22
4.2. Aplicación de la metodología Kanban al proyecto . . . . .	25
4.3. Planificación e iteraciones . . . . .	26
<b>5. Resultados</b> . . . . .	<b>33</b>
5.1. Visión general . . . . .	33
5.2. Iteraciones . . . . .	34
5.2.1. Iteración inicial . . . . .	34
5.2.2. Iteración 1 . . . . .	38
5.2.3. Iteración 2 . . . . .	40
5.2.4. Iteración 3 . . . . .	48
5.2.5. Iteración 4 . . . . .	54
5.2.6. Iteración 5 . . . . .	56
5.2.7. Iteración 6 . . . . .	61
5.2.8. Iteración 7 . . . . .	63
5.2.9. Iteración 8 . . . . .	64
5.2.10. Iteración 9 . . . . .	67
5.2.11. Iteración 10 . . . . .	69
5.3. Evaluación . . . . .	72
5.3.1. Errores detectados y solucionados en la fase de evaluación . . . . .	72
5.3.2. Análisis de hábitos y comportamientos . . . . .	73
<b>6. Conclusiones</b> . . . . .	<b>77</b>
6.1. Objetivos alcanzados . . . . .	77
6.2. Competencias obtenidas . . . . .	78
6.3. Trabajo futuro . . . . .	79
6.4. Opinión personal . . . . .	80
<b>A. Diagramas de casos de uso y prototipos iniciales</b> . . . . .	<b>83</b>
A.1. Diagrama de casos de uso . . . . .	83
A.1.1. Diagrama de casos de uso de la gestión de usuarios . . . . .	83
A.1.2. Diagrama de casos de uso de la monitorización de eventos . . . . .	85
A.1.3. Diagrama de casos de uso de la visualización de datos . . . . .	86
A.2. Prototipos iniciales . . . . .	87
A.2.1. Prototipos aplicación móvil . . . . .	87
A.2.2. Prototipos aplicación web . . . . .	88
<b>B. Eventos monitorizados y métricas obtenidas</b> . . . . .	<b>89</b>
<b>C. Configuración servicio de Accesibilidad y API REST Slim</b> . . . . .	<b>93</b>
C.1. Configuración servicio de accesibilidad . . . . .	93
C.2. Configuración API REST con Slim . . . . .	94
<b>D. Características principales librerías de gráficos JS</b> . . . . .	<b>97</b>
<b>E. Esquema base de datos</b> . . . . .	<b>99</b>
<b>F. Generación de informes y posicionamiento</b> . . . . .	<b>101</b>
<b>G. Aplicaciones existentes en el mercado</b> . . . . .	<b>105</b>
G.1. Actividad física . . . . .	105
G.2. Interacción con el dispositivo . . . . .	107

**H. Documentación API REST** 111

**Bibliografía** 125



# ÍNDICE DE FIGURAS

---

1.1. Esquema de la propuesta del Trabajo de Fin de Grado . . . . .	3
1.2. Esquema de eventos monitorizados en el sistema . . . . .	4
4.1. Fases de la metodología Kanban . . . . .	23
4.2. Planificación de los objetivos entre 04/02/20 - 30/06/2020 . . . . .	26
4.3. Diagrama de Gantt . . . . .	27
5.1. Esquema del Trabajo de Fin de Grado . . . . .	34
5.2. Tablero Kanban estado inicial . . . . .	35
5.3. Tablero Kanban iteración 0 . . . . .	38
5.4. Interfaz del login y registro inicial . . . . .	40
5.5. Tablero Kanban iteración 1 . . . . .	40
5.6. Funcionamiento del servicio de Accesibilidad . . . . .	41
5.7. Estructura tabla primerPlano . . . . .	44
5.8. Estructura de las tablas botones y bloqueo . . . . .	46
5.9. Tablero Kanban iteración 2 . . . . .	47
5.10. Interfaz del login y registro inicial . . . . .	49
5.11. Ejemplo de usuario registrado en la base de datos . . . . .	51
5.12. Inicio de sesión y registro definitivo de la app móvil . . . . .	51
5.13. Pulsación de botones físicos . . . . .	53
5.14. Número de llamadas y duración llamadas . . . . .	54
5.15. Tablero Kanban iteración 3 . . . . .	54
5.16. Ejemplo de registro de letras escritas . . . . .	55
5.17. Gráficos letras, símbolos y emoticonos enviados . . . . .	56
5.18. Tablero Kanban iteración 4 . . . . .	56
5.19. Estructura tabla detectaActividades . . . . .	58
5.20. Menú de opciones de la aplicación móvil . . . . .	60
5.21. Tablero Kanban iteración 5 . . . . .	60
5.22. Estructura tabla appsInstaladas . . . . .	61
5.23. Tiempo de uso de las aplicaciones más usadas . . . . .	62
5.24. Tablero Kanban iteración 6 . . . . .	62
5.25. Gráficos métricas WhatsApp . . . . .	63
5.26. Tablero Kanban iteración 7 . . . . .	64
5.27. Gráficos métricas Google Fotos . . . . .	67
5.28. Tablero Kanban iteración 8 . . . . .	67
5.29. Gráficos métricas RRSS y correo . . . . .	68
5.30. Gráficos métricas WhatsApp . . . . .	69
5.31. Tablero Kanban iteración 9 . . . . .	69
5.32. Opción resetear información . . . . .	70
5.33. Internacionalización de la aplicación móvil y web . . . . .	71

5.34. Tablero Kanban iteración 10 . . . . .	72
5.35. Comparativa del tiempo de uso de las aplicaciones más usadas por los usuarios . . . . .	74
5.36. Comparativa del número de conversaciones abiertas y notificaciones recibidas en WhatsApp de cada usuario . . . . .	74
5.37. Comparativa del tiempo de visualización de vídeos e imágenes y reproducción de audios en WhatsApp . . . . .	75
5.38. Comparativa del porcentaje de letras, símbolos y emoticonos enviados . . . . .	75
5.39. Comparativa tiempo de uso WhatsApp y el tiempo que se ha usado mientras el primer usuario está andando . . . . .	76
5.40. Comparativa tiempo de uso WhatsApp y el tiempo que se ha usado mientras el segundo usuario está andando . . . . .	76
5.41. Comparativa tiempo de uso WhatsApp y el tiempo que se ha usado mientras el tercer usuario está andando . . . . .	76
A.1. Diagrama de casos de uso gestión de usuarios . . . . .	84
A.2. Diagrama de casos de uso monitorización de eventos . . . . .	85
A.3. Diagrama de casos de uso visualización de datos . . . . .	86
A.4. Bocetos aplicación móvil . . . . .	87
A.5. Bocetos aplicación web ordenador . . . . .	88
A.6. Bocetos aplicación web móvil . . . . .	88
E.1. Esquema relacional de la base de datos . . . . .	99
F.1. Página de error 404 ordenador . . . . .	101
F.2. Página de error 404 móvil . . . . .	101
F.3. Indexación de la app web en Google Search Console . . . . .	102
F.4. Ejemplo de informe PDF . . . . .	103
G.1. Aplicación Adidas Running by Runtastic . . . . .	105
G.2. Aplicación RunKeeper . . . . .	106
G.3. Aplicación Google Fit . . . . .	106
G.4. Aplicación seguimiento de pasos - podómetro . . . . .	107
G.5. Aplicación analizador de uso . . . . .	107
G.6. Aplicación YourHour . . . . .	108
G.7. Aplicación Apps Usage . . . . .	108
G.8. Aplicación Smartphonoholic . . . . .	109
G.9. Aplicación AntiSocial . . . . .	109
G.10. Aplicación MyAddictometer . . . . .	110
G.11. Aplicación StayFree . . . . .	110

# ÍNDICE DE TABLAS

---

3.1. Tabla comparativa de aplicaciones que monitorizan la actividad física . . . . .	16
3.2. Tabla comparativa de aplicaciones de monitorización de eventos del dispositivo . . . . .	20
5.1. Requisitos funcionales. . . . .	35
5.2. Requisitos no funcionales . . . . .	36
5.3. Tabla comparativa librerías gráficos JavaScript . . . . .	53
5.4. Métricas de letras, símbolos y emoticonos representadas . . . . .	55
5.5. Métricas de eventos realizados en WhatsApp para ser representados . . . . .	63
5.6. Métricas de eventos realizados en Instagram . . . . .	65
5.7. Métricas de eventos realizados en Facebook . . . . .	65
5.8. Métricas de eventos realizados en Twitter . . . . .	65
5.9. Métricas de eventos realizados en Linkedin . . . . .	65
5.10. Métricas de eventos realizados en Gmail . . . . .	66
5.11. Métricas de eventos realizados en Outlook . . . . .	66
5.12. Métricas de eventos realizados en Google Fotos para ser representados . . . . .	66
A.1. Especificación del primer caso de uso . . . . .	83
A.2. Especificación del segundo caso de uso . . . . .	85
A.3. Especificación del tercer caso de uso . . . . .	86
B.1. Tabla de eventos y métricas recogidas de cada aplicación . . . . .	89



# ÍNDICE DE LISTADOS

---

5.1.	Mantener la sesión iniciada del usuario . . . . .	39
5.2.	Cerrar sesión con SharedPreferences . . . . .	39
5.3.	Declaración del servicio de accesibilidad . . . . .	42
5.4.	Configuración del recurso XML . . . . .	43
5.5.	Detección de las actividades en primer plano . . . . .	44
5.6.	Envío de información a la base de datos . . . . .	45
5.7.	Pulsación de teclas subir y bajar volumen . . . . .	45
5.8.	Detección bloqueo y desbloqueo pantalla . . . . .	46
5.9.	Obtención del identificador del usuario . . . . .	47
5.10.	Configuración Chrome Custom Tabs . . . . .	50
5.11.	Inicio de sesión aplicación web . . . . .	52
5.12.	Petición <i>API REST</i> en función del usuario . . . . .	52
5.13.	Detección de cambios en el sensor acelerómetro . . . . .	59
C.1.	Configuración del servicio de accesibilidad . . . . .	93
C.2.	Configuración del recurso XML . . . . .	93
C.3.	Configuración <i>API REST Slim</i> . . . . .	94
C.4.	Conexión BBDD . . . . .	94
C.5.	Ejemplo de petición HTTP . . . . .	95
F.1.	Posicionamiento de la aplicación web . . . . .	102
F.2.	Generación de informes PDF . . . . .	103



# LISTA DE ACRÓNIMOS

---

- API** Application Programming Interface - Interfaz de Programación de Aplicaciones. 3, 4, 6, 7, 9-11, 26, 29-31, 33, 34, 36, 48-53, 55, 57, 58, 61, 63, 66, 68, 78, 79, 92, 94
- REST** REpresentational State Transfer - Transferencia de representación de estado. 3, 6, 7, 9-11, 26, 29-31, 33, 34, 36, 48, 50-53, 61, 63, 68, 78, 79, 94
- URL** Uniform Resource Locator - Localizador Uniforme de Recursos. 50, 52, 79, 111-123
- TFG** Trabajo de Fin de Grado. 1, 8, 10, 12, 25, 26, 36, 41, 72
- SO** Sistema Operativo. 9, 10, 14, 36
- SQL** Structured Query Language - Lenguaje de Consulta Estructurada. 9, 12, 49, 98
- IDE** Integrated Development Environment - Entorno de desarrollo integrado. 9, 10
- JSON** JavaScript Object Notation - Notación de objeto de JavaScript. 11, 49, 52
- HTML** HyperText Markup Language - Lenguajes de marcas de hipertexto. 9, 11, 29, 71, 97, 98
- PHP** Hypertext Pre-Processor - Lenguaje de Programación Interpretado. 9, 11, 29, 32, 38, 39, 44, 48, 49, 70
- CSS** Cascading Style Sheets - Hoja de Estilos en Cascada. 9, 11, 29, 97
- MVC** Modelo Vista Controlador. 37
- M4S** Mobile-computing-based Multitasking for Mild cognitive impairment Monitoring and early Screening. 1, 2
- BIPapp** App to monitor Behaviour & Interaction Patterns. 1, 2, 19, 30, 33, 73
- CASE** Computer Aided Software Engineering - Ingeniería de *software* asistida por ordenador. 9
- XML** eXtensible Markup Language - Lenguaje de Etiquetado Extensible. 42, 43, 52, 93
- MAC** Media Access Control - Dirección física. 47
- IMEI** International Mobile Equipment Identity - Identidad internacional de equipo móvil. 47
- HTTP** Hypertext Transfer Protocol - Protocolo de transferencia de hipertextos. 10, 44, 48, 49, 79, 95, 111-123
- SEO** Search Engine Optimization - Optimización en Motores de Búsqueda. 71
- AJAX** Asynchronous JavaScript and XML. 29, 52, 53
- App** Aplicación. 1, 16, 18, 20, 31, 37, 51, 65, 66, 68, 73, 74, 89-92, 107-109
- RRSS** Redes Sociales. 20, 31, 68, 87
- BBDD** Base de Datos. 9, 38, 47, 89, 94

**IPO** Interacción Persona Ordenador. 13

**TICs** Tecnologías de la Información y la Comunicación. 15, 19

# GLOSARIO DE TÉRMINOS

---

- Front-end** Parte de la aplicación que interactúa con el usuario y que está en el lado del cliente. 3, 9, 11, 48, 50, 51, 79
- Back-end** Parte que procesa la entrada desde el *front-end* y que está en el lado del servidor. 11, 48
- Endpoint** Son los distintos puntos de comunicación que permiten realizar diferentes operaciones en una API. 6, 49, 53, 61, 63, 68, 94, 95, 111
- Hosting** Servicio en línea que te permite publicar un sitio o aplicación web en Internet. 3, 11, 37
- Framework** Entorno de trabajo que engloba un conjunto de estándares y herramientas para la realización de una tarea. 11, 17, 58, 94, 97
- Microframework** Entorno de trabajo minimalista con menor número de funcionalidades que un *framework*, que engloba un conjunto de estándares y herramientas para la realización de una tarea. 7, 48
- Sprints** Intervalo previamente definido durante el cual se crea un incremento del producto terminado utilizable y que puede ser entregado. 21
- Feedback** Retroalimentación enviada al usuario para que compruebe que sus acciones han producido ciertos resultados. 24-26, 34, 79
- AJAX** Procesamiento de solicitudes al servidor de forma asíncrona. 29, 52, 53
- Local Storage** Almacenamiento local de información en el navegador hasta que se limpian los datos del navegador. 50-52
- Cookies** Información enviada por un sitio web y almacenada en el navegador del usuario. 50
- Layout** Contenedor de una o más vistas que incluye varios elementos según el diseño de una aplicación. 35
- Log** API para enviar la salida del registro. 9, 36, 44
- Widget** Parte esencial de la personalización de la pantalla de inicio. 50
- Rootear** Proceso que permite a los usuarios de cualquier dispositivo Android, obtener control privilegiado, en el cuál se pueden modificar ciertas funciones que vienen por defecto en los terminales Android. 36
- Activity/Actividad** Una Activity representa una interfaz de usuario en Android. 33, 37-39, 45, 56, 57, 59, 61, 69, 73, 76, 84, 87



---

## CAPÍTULO 1

# INTRODUCCIÓN

---

Este primer capítulo tiene la finalidad de presentar una breve introducción sobre el contexto en el que se enmarca este proyecto, la motivación que ha provocado realizarlo y la solución propuesta para abordarlo. La implementación de esta solución se encuentra alojada en Github<sup>1</sup>, que ha permitido registrar un control de versiones. Por último, se enumerarán los distintos capítulos y anexos que componen la estructura principal de esta memoria.

### 1.1. CONTEXTO

Los teléfonos móviles se han convertido actualmente en un elemento imprescindible que ofrece muchas prestaciones para facilitar el día a día a los jóvenes, adultos y personas de avanzada edad. De hecho, realizamos la mayoría de las actividades diarias apoyándonos en los dispositivos móviles, desde nuestras relaciones sociales, nuestros hábitos de consumo y ocio, hasta nuestro trabajo. Es necesario realizar un estudio sobre el uso que hacen los usuarios de los dispositivos móviles, analizando en detalle no solo qué aplicaciones y servicios se utilizan, sino también cuándo se usan y sobre todo cómo el usuario hace uso de estas aplicaciones.

El desarrollo de este TFG se enmarca en el proyecto del programa nacional Retos de Investigación denominado M4S (*Mobile-computing-based Multitasking for Mild cognitive impairment Monitoring and early Screening*), que surge con el objetivo de poder estudiar el impacto cognitivo que produce el uso de los dispositivos móviles y la interacción con estos. BIPapp (*App to monitor Behaviour & Interaction Patterns*) es el nombre del sistema desarrollado en este TFG, y que funciona en segundo plano para monitorear todas las acciones y eventos realizados por el usuario con el dispositivo, asegurando la privacidad y la ética en la recogida de los datos.

Se pretende recoger eventos del sistema, como por ejemplo, bloqueo/desbloqueo del dispositivo móvil, eventos más comunes de las aplicaciones más utilizadas como WhatsApp, redes sociales o aplicaciones de correo electrónico, detección de la actividad física del usuario y tiempo de uso del teléfono y de las aplicaciones más utilizadas.

Este TFG contribuye a las tareas futuras del proyecto donde se desea contribuir al diagnóstico temprano del deterioro cognitivo leve, ya que estos datos proporcionan a los expertos información muy valiosa obtenida directamente del usuario, sobre su estado funcional y cognitivo.

---

<sup>1</sup>Repositorio del TFG: <https://github.com/elenadiaz-tfg/BIPapp>

## 1.2. MOTIVACIÓN

El deterioro cognitivo leve (DCL) es un estado intermedio entre el deterioro cognitivo debido al envejecimiento y el deterioro de la demencia, que puede implicar problemas con la memoria, el lenguaje, la atención o la capacidad para razonar. En algunos casos el deterioro cognitivo leve puede aumentar el riesgo de padecer demencia en el futuro, pero en otros casos algunas personas nunca empeoran e incluso mejoran con el paso de los años.

Según una investigación reciente [15], expone que las personas interactúan de diferentes formas con sus dispositivos móviles. Mediante la tecnología *Cupertino* monitorizaron 113 teléfonos de adultos de entre 60 y 75 años y, 31 de ellos presentaron un diagnóstico de deterioro de sus facultades mentales. Cada estudio abarcaba todas las acciones realizadas por los usuarios desde que desbloqueaban su dispositivo y hasta que los volvían a bloquear. Alrededor del 80 % del tiempo del estudio, el sistema fue capaz de diferenciar usuarios con problemas cognitivos y qué, el contexto, la frecuencia, el tiempo de uso y la combinación de aplicaciones en estas sesiones guarda relación con el estado cognitivo de los participantes.

Como demuestra el estudio anterior, existen problemas cada vez más frecuentes que provocan demencia digital o en otras palabras, un deterioro cognitivo leve. Es necesario realizar estudios en profundidad sobre las diferentes formas de interacción que presentan este tipo de usuarios con los dispositivos móviles y el uso que hacen de ellos. Basándonos en esta premisa, una de las hipótesis que presenta el proyecto M4S es la posibilidad de apoyar en investigaciones para realizar un diagnóstico temprano de deterioro cognitivo leve identificando cambios en la forma de utilizar el móvil.

Para poder llegar a demostrar dicha hipótesis es necesario realizar un análisis en profundidad de las distintas formas de interacción de los usuarios con los dispositivos móviles y del uso que hacen de ellos obteniendo información directa del usuario. De esta necesidad, surge la idea principal y motivación de este Trabajo de Fin de Grado, donde se propone el desarrollo de un sistema que sea capaz de monitorizar distintos eventos del usuario provenientes de eventos y acciones que surgen de la interacción del usuario con el dispositivo móvil a lo largo del tiempo y que serán almacenados en bruto en una base de datos para después procesar la información relevante y representarla de forma gráfica. Además, brindar esta herramienta a los investigadores y expertos para que puedan analizar tanto los datos en bruto almacenados como visualizar convenientemente los datos recolectados y obtener un conjunto de métricas significativas, para que puedan realizar un análisis del uso de los móviles, mejorar la interacción persona-móvil y que pueda servir para realizar evaluaciones exhaustivas que puedan ayudar a detectar síntomas de deterioro cognitivo leve en etapas tempranas y poder realizar un diagnóstico completo del usuario.

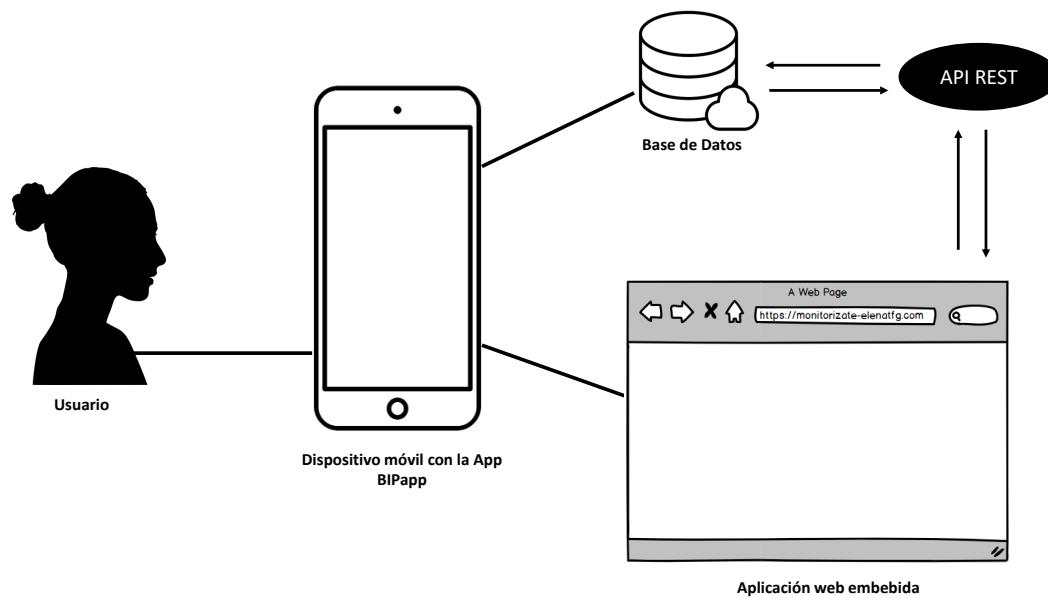
## 1.3. SOLUCIÓN PROPUESTA

Teniendo en cuenta todos los problemas presentados anteriormente, se pretende desarrollar una aplicación Android denominada BIPapp, que permita monitorizar y supervisar los comportamientos e interacción con el dispositivo móvil.

Esta aplicación incorpora servicios proporcionados por Android que el usuario tiene que activar manualmente. Esta información se almacena y se procesa para poder representarse en forma de gráficos y tablas, acotados por tiempo y tipo.

En la Figura 1.1, se muestra un esquema formado por los principales componentes del sistema:

- **Usuario.** En este proyecto el usuario es tan importante como la implementación desarrollada, ya que dependiendo de su interacción con el móvil se obtendrán más o menos datos.
- **Aplicación móvil.** La aplicación Android es el núcleo principal del sistema donde el usuario realizará distintas acciones sobre la interfaz que permitirá la monitorización de eventos.
- **Aplicación web embebida.** Es el componente esencial para el usuario donde podrá visualizar los eventos realizados con su dispositivo dentro de la aplicación o en un navegador externo.
- **API REST.** Arquitectura *software* que sirve como interfaz común entre distintos sistemas para acceder y manipular la información de la base de datos.
- **Base de datos en la nube.** Un *hosting* alojado en la nube que ofrece la creación de una base de datos remota que será accedida mediante una *API REST* que se alojará en el servidor de archivos. Además, el servidor contiene el *front-end* de la aplicación.



**Figura 1.1:** Esquema de la propuesta del Trabajo de Fin de Grado

Los eventos y acciones concretos monitorizados y/o visualizados se presentan en la Tabla B.1 del Anexo B. No obstante, en la Figura 1.2, se muestran esquemáticamente representados en categorías los distintos eventos que el sistema monitorizará y almacenará en bases de datos. Estos eventos son producidos por el propio sistema (como es el caso de las notificaciones) o de la interacción del usuario con el dispositivo móvil (como es el caso de los eventos del teclado o de los botones físicos). Las distintas categorías que se pueden diferenciar son:

- **Sistema.** Son eventos que se producen en el propio sistema y no en aplicaciones concretas, donde se puede diferenciar la detección de las aplicaciones y servicios ejecutados en primer plano, notificaciones recibidas de cualquier aplicación o servicio del sistema, aplicaciones instaladas en el dispositivo y eventos de instalar o desinstalar aplicaciones, llamadas realizadas en el dispositivo o apertura de la cámara.
- **Botones físicos.** Se corresponden con eventos producidos por el usuario debido a la pulsación de los botones *hardware* como subir y bajar volumen o bloquear y desbloquear el dispositivo.
- **Teclado.** Son eventos que se producen al escribir letras (caracteres de la A-Z), símbolos (números del 0-9 y caracteres especiales) u emoticonos y que permiten calcular la frecuencia de pulsación de estos elementos.
- **Sensores y Google APIs.** Detección del número de pasos dados por el usuario y de la actividad física del usuario, que va a permitir relacionarlo con el tipo de evento detectado en el dispositivo en un momento determinado.
- **WhatsApp.** Acciones más comunes realizadas en la aplicación WhatsApp como la visualización de imágenes o vídeos, o las llamadas y videollamadas realizadas.
- **Outlook y Gmail.** Eventos producidos en las aplicaciones de correo electrónico Outlook y Gmail.
- **Redes Sociales.** Son eventos que se producen en las redes sociales más utilizadas en el mundo como son Instagram, Facebook y Twitter junto con la red empresarial más utilizada en España, como es LinkedIn<sup>2</sup>.
- **Google Fotos.** Eventos visualización de vídeos e imágenes, producidos en la aplicación de galería común presente en todos los dispositivos móviles Android denominada Google Fotos.

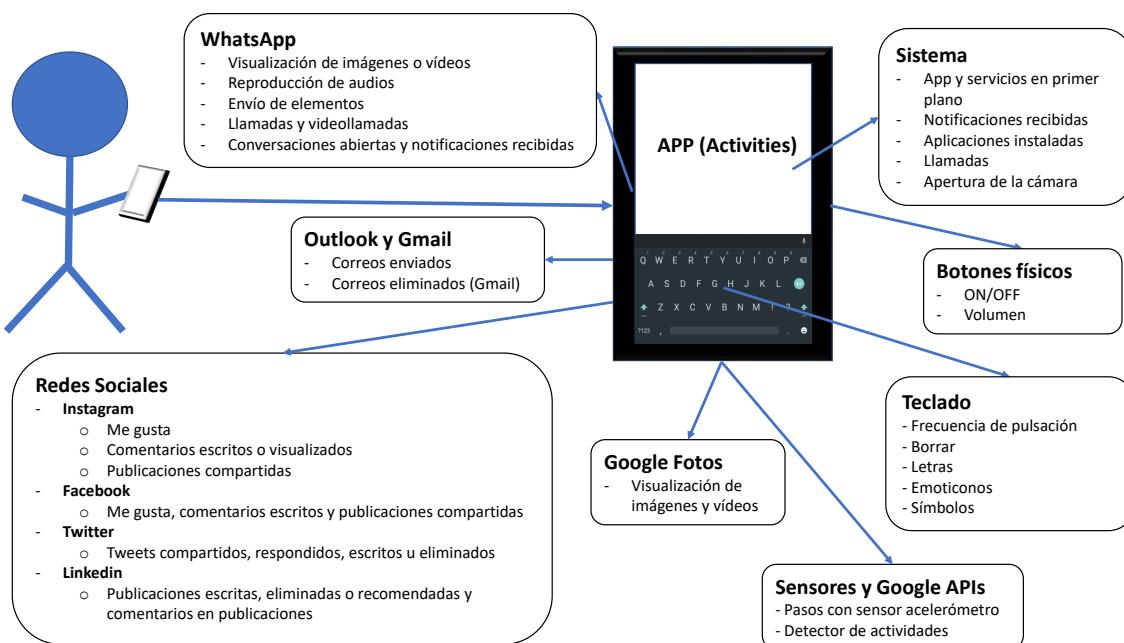


Figura 1.2: Esquema de eventos monitorizados en el sistema

<sup>2</sup><https://mkparadise.com/redes-sociales-mas-utilizadas>

## 1.4. ESTRUCTURA DEL DOCUMENTO

La memoria de este proyecto se estructura en los siguientes capítulos principales:

### **Capítulo 1. Introducción**

Es el capítulo inicial en el que nos encontramos, en el cuál, se hace una breve introducción sobre el contexto y la motivación de este proyecto, y que conllevan a la elaboración de una propuesta para solucionar los problemas presentados.

### **Capítulo 2. Objetivos y herramientas**

Se enumeran y se explican los objetivos de este proyecto que se desean alcanzar, divididos en general, específicos, teóricos y prácticos. Por último, se nombran todas las herramientas *hardware* y *software* que se han utilizado.

### **Capítulo 3. Antecedentes**

En este capítulo se analizan distintos estudios de investigación sobre el uso que hacen los usuarios de los dispositivos móviles en la actualidad y distintas aplicaciones que nos permiten monitorizar estas acciones físicas y eventos.

### **Capítulo 4. Metodología**

Se explica la metodología de trabajo que ha guiado el desarrollo de este proyecto y la planificación de la realización de cada iteración para cumplir con los plazos establecidos.

### **Capítulo 5. Resultados**

En este capítulo se muestran todos los resultados que se han obtenido en cada iteración, siguiendo la metodología de desarrollo mencionada en el capítulo anterior. En primer lugar, se muestra el resultado final y sucesivamente se explican todos los pasos realizados y problemas encontrados hasta alcanzar ese resultado. Por último, se explican las pruebas realizadas para comprobar su correcto funcionamiento.

### **Capítulo 6. Conclusiones**

En el último capítulo se justifica cómo se han cumplido todos los objetivos que se propusieron en el Capítulo 2. y las distintas propuestas que se podrían llevar a cabo en un futuro de este proyecto realizado.

## Anexos

- **Anexo A.** Muestra los diagramas de casos de uso del sistema desarrollado y el diseño de los prototipos iniciales.
- **Anexo B.** Presenta una tabla que incluye todos los eventos monitorizados y métricas obtenidas que se explicarán en cada iteración.
- **Anexo C.** Expone la configuración de los permisos para monitorizar eventos en el servicio de accesibilidad y los pasos de creación y configuración de la *API REST* con Slim.
- **Anexo D.** En este anexo se enumeran las distintas características estudiadas de las cinco librerías JavaScript más populares para la elaboración de gráficos.
- **Anexo E.** Expone el esquema del conjunto de tablas que componen la base de datos del sistema.
- **Anexo F.** Muestra el posicionamiento de la aplicación web, junto con el diseño de la página de error 404 y la generación de informes.
- **Anexo G.** Se enumeran las distintas aplicaciones existentes en el mercado que permiten monitorizar la actividad física del sistema y eventos producidos de la interacción con los dispositivos móviles.
- **Anexo H.** Por último, se muestra la documentación de la *API REST* con los distintos *endpoints* implementados para obtener las métricas deseadas.

---

## CAPÍTULO 2

# OBJETIVOS

---

En este capítulo se va a mencionar el objetivo general de este proyecto y se va a desglosar en objetivos más específicos que van a permitir llevar a cabo el objetivo principal. Además, se enumerarán unos objetivos prácticos y teóricos necesarios para poder llegar a alcanzar los objetivos específicos.

### 2.1. OBJETIVO GENERAL

El objetivo principal de este Trabajo de Fin de Grado es caracterizar el comportamiento y hábitos de uso del dispositivo móvil mediante el desarrollo de una aplicación que haga uso de servicios Android y monitorice acciones y eventos, tanto interactivos como del sistema, relacionándolos con la actividad física realizada en cada momento.

### 2.2. OBJETIVOS ESPECÍFICOS

A su vez, el objetivo general se puede descomponer en varios objetivos específicos:

- Captura de acciones y eventos, tanto interactivos, como del sistema, en dispositivos Android.
- Almacenamiento de la información en bases de datos.
- Diseño e implementación de técnicas de visualización de información que muestre los hábitos y comportamientos.

### 2.3. OBJETIVOS TEÓRICOS Y PRÁCTICOS

A continuación, se explican algunos objetivos teóricos y prácticos que se han tenido que realizar para completar este proyecto. Los objetivos teóricos son necesarios para documentarse antes de desarrollar y los prácticos son objetivos funcionales que se han tenido que seguir para realizar el sistema.

- **Estudio y aprendizaje de las tecnologías utilizadas.** Antes de empezar a implementar una funcionalidad, es necesario realizar un estudio de las distintas tecnologías y herramientas que se van a utilizar. Para el desarrollo de este proyecto, es fundamental investigar sobre los servicios en segundo plano y los servicios de accesibilidad proporcionados por Android. Por otro lado, se tiene que realizar un estudio de *microframeworks* para la realización de la *API REST*.

- **Realizar un estudio de aplicaciones similares en el mercado y de artículos de investigación.** Una de las tareas iniciales, es la búsqueda de aplicaciones similares en el mercado que ofrezcan parcialmente las funcionalidades de este proyecto y artículos de investigación sobre este tema.
- **Estudio en profundidad de las metodologías de desarrollo software.** Realizar un estudio de las diferentes metodologías de desarrollo de *software* estudiadas en la asignatura Ingeniería del *Software II* y escoger la que más se ajuste al desarrollo de este TFG.
- **Diseñar una aplicación móvil para dispositivos Android que sea sencilla y usable.** Aprendizaje de las guías de usabilidad, accesibilidad y diseño estudiadas en Interacción-Persona-Ordenador para la creación de una aplicación móvil que sea usada por cualquier colectivo de personas.
- **Desarrollar una aplicación web responsive que sea sencilla e intuitiva.** Estudio de lenguajes de programación web que permiten realizar una aplicación web que se adapte a cualquier tipo de dispositivo y al mayor número de versiones del navegador mediante los fundamentos aprendidos en Tecnologías y Sistemas Web.
- **Permitir la visualización de los datos de una forma atractiva.** Realizar un estudio de las mejores librerías JavaScript para la realización de gráficos e implementar la que aporte más ventajas y que requiera una menor curva de aprendizaje debido a la duración de este proyecto.
- **Mejorar el posicionamiento de la aplicación web.** Aplicación de los fundamentos aprendidos en Comercio electrónico para mejorar el posicionamiento de la aplicación web y que incentive a los usuarios a descargarse y utilizar la aplicación.
- **Permitir la internacionalización de la aplicación móvil y web.** Ofrecer la opción de poder usar el sistema en dos idiomas: inglés y español, con el objetivo de obtener una gran variedad de usuarios.

## 2.4. MEDIOS TECNOLÓGICOS UTILIZADOS

En esta sección, se describen los distintos medios *software*, *hardware* y lenguajes de programación que han sido utilizados en el desarrollo de este TFG.

### 2.4.1. Medios *software*

- **Balsamiq Mockups.** Es una herramienta rápida para la creación de la estructura de interfaz de usuario. Es de baja fidelidad y reproduce la acción de dibujar en un papel, pero en este caso haciendo uso de una computadora y de una serie de componentes. Se ha utilizado para diseñar la estructura y el contenido, pero evitando los colores y detalles que se realizarían en etapas posteriores. Debido a que los bocetos se realizan en las fases iniciales, podrán sufrir

modificaciones de diseño, ya que habrá que adaptarse a las normas de estilos y componentes de Android.

- **Visual Paradigm Online.** Es una herramienta Ingeniería de *Software* Asistida por Computación (CASE). Proporciona numerosas ayudas para el desarrollo de programas informáticos, desde las fases iniciales de planificación, pasando por las fases de análisis y diseño, hasta la generación del código fuente de los programas y documentación a partir de estos diagramas. Fue creada con el propósito de soportar el ciclo de vida completo del proceso de desarrollo de *software* a partir de la creación de una gran variedad de diagramas como pueden ser: de clases, de diseño, de análisis, de secuencia etc. Se ha utilizado a lo largo del proyecto para la creación de diagramas de casos de uso.
- **Android Studio.** Es un Entorno de Desarrollo Integrado (IDE) oficial para el desarrollo de aplicaciones para el Sistema Operativo (SO) Android, basado en IntelliJ IDEA. Android ofrece numerosas funciones que aumenta la productividad en las aplicaciones Android desarrolladas. Se ha utilizado en este proyecto para la creación de la aplicación, haciendo uso de herramientas como el depurador, Logcat o el emulador de dispositivos que se indica a continuación.
- **Android Virtual Device (AVD).** Es un emulador utilizado para configurar dispositivos virtuales que permiten la depuración de la aplicación y visualizar la interfaz de nuestra aplicación en pantallas de dispositivos con distintas dimensiones.
- **GitHub.** Es un repositorio utilizado para la gestión de proyectos y para controlar las versiones de código. Permite realizar un seguimiento del trabajo y trabajar en colaboración junto con otras personas. Se ha utilizado para almacenar y organizar todo el proyecto y para controlar y gestionar los cambios.
- **GitHub Desktop.** Es una herramienta que simplifica el flujo de trabajo con GitHub, usando una interfaz visual en lugar de comandos de texto en la línea de comandos de Git. Se ha utilizado para revisar los cambios y subirlos al repositorio de GitHub, de una forma rápida y eficaz.
- **Sublime Text 3.** Es un editor de código multiplataforma, ligero y simple para programar en un gran número de lenguajes como CSS, HTML, PHP, JavaScript, SQL o Java, entre otros. Permite la opción de autoguardado y autocompletar, entre otras funcionalidades. Se ha utilizado para la creación de los *scripts* en PHP para conectarse a la BBDD desde la aplicación Android y guardar todos los datos recogidos. También se ha utilizado para la creación del *front-end* de la aplicación web y para la edición de la *API REST*.
- **TeXstudio.** Es una herramienta de escritura para crear documentos en LaTeX. Su objetivo principal es ayudar y facilitar a escribir los documentos con la incorporación de un visor integrado, resaltado de sintaxis, corrector ortográfico etc. Se ha utilizado para la creación de esta memoria, gracias a la plantilla proporcionada por el profesor de esta escuela Jesús Salido,

donde se define la estructura y el formato especificado en la normativa.

- **Postman.** Nació como una extensión del navegador Google Chrome que nos permitía crear peticiones sobre APIs para poderlas probar de forma sencilla. Actualmente, se ha convertido en una plataforma de desarrollo de APIs utilizada principalmente para crear y enviar peticiones HTTP a servicios *REST* mediante un interfaz gráfico que permite guardarlas para probarlas en cualquier momento. En este proyecto se ha usado para realizar distintas peticiones a la *API REST* y comprobar su funcionamiento.
- **Firefox Developer Edition.** Firefox Developer Edition es un navegador web creado por Firefox para ayudar explícitamente a los desarrolladores web. Permite inspeccionar las páginas, probar estilos y visualizar la página en diseño responsive de una forma avanzada. Esta herramienta ha sido utilizada para poder realizar el diseño de la aplicación web y realizar pruebas mediante la consola web y el depurador de JavaScript integrado.

#### 2.4.2. Medios hardware

- **Ordenador portátil MSI Prestige 15 A10SC** con procesador Intel Core i7 10TH Generación, 16 GB de memoria RAM y sistema operativo Windows 10 Pro hasta 64 bits. Ha sido utilizado desde el principio del TFG y en él se encuentran las distintas herramientas utilizadas para el desarrollo de este proyecto.
- **Smartphone BQ Aquaris X** con un procesador Qualcomm Snapdragon 626 hasta 2.20 Mhz de velocidad, 3 GB de memoria RAM, sistema operativo Android y versión 8.1.0. Este teléfono móvil se ha utilizado para realizar pruebas del funcionamiento de la aplicación, detección, filtrado y recogida de los datos que se han guardado en la base de datos. El resto de teléfonos móviles que se mencionan a continuación tienen los mismos propósitos que éste. La única diferencia que presentan es la versión del SO y el fabricante, en los que se ha estudiado la compatibilidad para la recogida de datos.
- **Smartphone Oppo RX17 Pro** con un procesador Qualcomm SDM710 de 8 núcleos y una frecuencia máxima de 2.2 GHz, 6 GB de memoria RAM, sistema operativo Android y versión 9.
- **Smartphone Xiaomi Mi Note 10** con procesador Qualcomm Snapdragon 730G, con una frecuencia máxima de hasta 2.2 GHz, frecuencia máxima de renderizado de 700 MHz, 6 GB de memoria RAM, sistema operativo Android 9.0.

#### 2.4.3. Lenguajes de programación

- **Java.** En Android Studio existen varios lenguajes para desarrollar las aplicaciones, los lenguajes principales son Java, Kotlin y C++. Se ha seleccionado el lenguaje Java por ser el más usado en el IDE Android Studio para realizar aplicaciones, estar más familiarizada con él a lo largo de todo el grado y haber realizado otras aplicaciones anteriores con este lenguaje en Android Studio.

- **PHP.** El lenguaje de programación PHP se utiliza generalmente en el lado del *back-end* o servidor, ya que se puede usar en distintos sistemas operativos y en la mayoría de los servidores web. Procesa la información y realiza acciones que le permite enlazarse con otros lenguajes como HTML o Java. Se ha utilizado en este proyecto para la creación de distintos *scripts* para conectar la aplicación con la base de datos proporcionada por el *hosting* remoto y para la creación de la *API REST* con el *framework* Slim<sup>1</sup> que realiza las distintas conexiones a esa base de datos y accede a los datos solicitados. Estas peticiones devolverán los datos en formato JSON que posteriormente serán recibidos por la aplicación web y representados en gráficos con JavaScript.
- **HTML.** HTML es un lenguaje de marcado que se utiliza para la elaboración de páginas web, permite definir la estructura y la definición del contenido estático de la página. Se ha utilizado en este proyecto para crear la estructura de la aplicación web, incluyendo el menú, las distintas secciones, el contenido estático y los diferentes elementos *canvas* para mostrar los gráficos con los datos recogidos.
- **JavaScript.** JavaScript es un lenguaje de programación que permite realizar acciones complejas en una página web, es decir, mostrar información actualizada en tiempo real. Es el lenguaje principal utilizado para extraer los datos de los JSON devueltos por las peticiones realizadas a la *API REST* y la representación de esos datos en los gráficos, haciendo uso de una librería específica de JavaScript para hacer gráficos denominada *Chart.js*<sup>2</sup>.
- **jQuery.** jQuery es una librería de JavaScript que permite simplificar las acciones realizadas con JavaScript y permite agregar eventos interactivos a un sitio web, como por ejemplo la realización del menú de navegación que se ha llevado a cabo en la aplicación web.
- **CSS.** CSS es un lenguaje de estilos que se usa para diseñar o estilizar los elementos creados con el lenguaje HTML. En otras palabras, HTML constituye la base del sitio y CSS añade un estilo a esa página.

#### 2.4.4. Servicios externos y almacenamiento remoto

- **Hostinger.** Para realizar la conexión con la base de datos remota desde la aplicación, se necesita crear un servicio web para alojar los *scripts* y para publicar una aplicación web es necesario un servidor para poder almacenar los archivos. Para ello, se ha usado un servidor de la página oficial de Hostinger. Este hosting está formado por un cPanel que permite crear bases de datos MySQL y que pueden ser manejadas y administradas por una herramienta escrita en PHP en forma de interfaz web denominada *phpMyAdmin*<sup>3</sup>. En *phpMyAdmin* se han creado las distintas tablas necesarias para almacenar los datos recogidos. Por otro lado, ofrece un servidor para almacenar archivos, donde se han almacenado los diferentes *scripts*, la parte *front-end* de la aplicación web y la *API REST*.

---

<sup>1</sup><http://www.slimframework.com/>

<sup>2</sup><https://www.chartjs.org/>

<sup>3</sup><https://www.phpmyadmin.net/>

#### 2.4.5. Otras herramientas

- **Gimp y PowerPoint.** Se han utilizado estas herramientas para la generación de esquemas utilizados en la memoria y para la creación de los distintos iconos de la aplicación.
- **Microsoft Teams.** Es una herramienta utilizada para realizar las distintas reuniones con el cliente y el equipo del proyecto. En este proyecto el cliente se corresponde con los directores de este TFG. También se ha usado para alojar documentos o información en una wiki colaborativa por todos los miembros del grupo.
- **MySQL Workbench.** Es una herramienta visual para realizar el diseño de bases de datos o consultas SQL a las mismas y que ha sido utilizada para realizar el modelo de la base de datos con las distintas tablas que la componen.
- **Monday.** Monday<sup>4</sup> es una plataforma de gestión del trabajo basada en la nube que permite gestionar el trabajo en equipo y los proyectos desarrollados. Además, incluye más características y múltiples vistas de trabajo que la herramienta Trello<sup>5</sup>. Se ha usado en este proyecto para realizar los tableros Kanban con las distintas fases por las que han pasado las tareas implementadas.

---

<sup>4</sup><https://monday.com/lang/es/lp/mb/dpm-kan>

<sup>5</sup><https://trello.com/b/C4Awm5lK/kanban-board>

---

## CAPÍTULO 3

# ANTECEDENTES

---

Antes de empezar a desarrollar un proyecto es importante recopilar información y resultados de otros desarrollos e investigaciones relacionadas con el tema que se ha seleccionado, con el objetivo de no reinventar la rueda y determinar un enfoque que marque la diferencia respecto a los aspectos que ya se han investigado. En este capítulo, se pretende hacer una breve introducción sobre la interacción de las personas con los dispositivos móviles y realizar un análisis de artículos de investigación y herramientas que permitan monitorizar actividades físicas y eventos de interacciones con el dispositivo.

### 3.1. INTERACCIÓN PERSONA - DISPOSITIVO MÓVIL

La disciplina de la Interacción Persona-Ordenador (IPO) surgió alrededor del año 1969 [22, 36], aunque los cimientos se produjeron antes con la creación de los ratones, mapas de bits o metáforas de escritorio y punteros para *clicar*. Conocida en inglés como *Human-Computer-Interaction* (HCI), es una ciencia que estudia la interacción de las personas con la tecnología con el objetivo de mejorar la usabilidad y la experiencia del usuario. El uso de los *smartphones* ha incrementado a lo largo del tiempo y debido a esto, hay numerosas investigaciones que se centran en la interacción de las personas con los móviles y en un diseño centrado en los usuarios.

Hoober [21] llevó a cabo una investigación sobre cómo las personas sostienen los dispositivos móviles y dependiendo de las circunstancias, cómo interactúan con ellos. Realizaron alrededor de 1.300 observaciones de personas que utilizaban los dispositivos móviles en la calle, cafeterías, autobuses o aeropuertos y observaron tres formas de sujetar el móvil, tocar las pantallas y los botones: (1) Con una mano, (2) Con ambos manos, (3) Con dos manos, una para sujetar el dispositivo y otra para interactuar con él. Obtuvieron que un 49 % de los resultados de la muestra, los usuarios sujetaban el dispositivo con una mano, pero que constantemente estos usuarios realizaban otros gestos con la otra mano o cambiaban al uso de las dos manos para escribir. A partir de este estudio sobre cómo interactúan los usuarios con los dispositivos, obtuvieron datos respecto a si el uso era activo o pasivo, es decir, si realizaban algún movimiento táctil o en cambio visualizaban vídeos, escuchaban música o realizaban llamadas telefónicas. También obtuvieron información de la postura corporal del usuario al interactuar con el dispositivo móvil, diferenciando entre caminando, de pie o sentados.

Karam y Shraefel realizaron en 2005 una amplia taxonomía [24] que aborda una diversidad de interacciones gestuales, debido a que las taxonomías existentes eran demasiado abstractas y ambiguas.

En este proyecto no solo estudiaron la interacción de los usuarios con los móviles sino con cualquier dispositivo en general. Respecto a la interacción con los *smartphones*, obtuvieron información de la cámara, la superficie táctil y los sensores integrados como el acelerómetro o el GPS.

Existen otras investigaciones relacionadas de la interacción con las pantallas táctiles. Wroblewski [41] propuso una guía de referencia de gestos táctiles para los diseñadores y desarrolladores de *software* que se esfuerzan en la creación de interfaces táctiles. Es necesario realizar un estudio de la interacción de los usuarios con los dispositivos móviles, para solucionar errores y mejorar la experiencia de su uso. Sin embargo, los fabricantes de teléfonos consideran que todos los usuarios son similares y que se pueden diferenciar pocos tipos de usuarios. En el artículo [43], realizaron un estudio sobre el uso que hacen de una aplicación 106.762 usuarios en un mes y descubrieron 382 tipos de usuarios en función de su comportamiento en el uso de la aplicación.

Para analizar las interacciones con los teléfonos móviles se puede recoger información proporcionado por el SO, los sensores que tiene integrados, botones *hardware* y las aplicaciones instaladas. En otra investigación reciente realizada por la Universidad de Castilla-La Mancha [9], elaboraron una taxonomía para clasificar las tareas más comunes producidas por las interacciones con los teléfonos inteligentes a nivel cognitivo. Acuñaron la taxonomía como HuSBIT-10 (*Human-Smartphone Basic Interactions Taxonomy for 10-second tasks*), en la que propusieron tareas rápidas y simples, con una duración menor de 10 segundos. En primer lugar, identificaron cuatro tipos de interacciones relacionadas con los sentidos humanos y que son esenciales para analizar la carga cognitiva y el procesamiento de la información. Estos tipos de interacción eran: tocar, mirar, hablar y escuchar, y cada uno de estos tipos se podían clasificar en dos categorías: activa y pasiva, dependiendo de si el usuario interacciona directamente con el dispositivo o lo está utilizando sin interactuar con él. Definieron el término AMPEC-10, para agrupar cinco tipos de tareas que un usuario podría realizar con el dispositivo en tareas cortas de un tiempo máximo de 10 segundos, haciendo uso de los cuatro tipos de interacción mencionados anteriormente. Como su nombre indica, AMPEC-10 agrupa tareas según los siguientes tipos:

**(A) Automatizadas.** Son tareas que no suponen ningún esfuerzo para el usuario porque se realizan de forma automática o inconscientemente.

**(M) Psicomotriz.** Este tipo incluye tareas que requieren una interacción rápida con el dispositivo.

**(P) Producción.** Tareas que requieren la creación de contenidos básicos y para las que se necesita habilidades creativas para producir nuevos contenidos.

**(E) Exploración.** Tareas para las que se necesita analizar un conjunto de datos para obtener información concreta.

**(C) Consumo.** Tareas relacionadas con el consumo de contenido.

Realizando una primera exploración obtuvieron que las interacciones más comunes entre el usuario y el dispositivo móvil son las de tacto y mirada respecto a las de habla y el oído que son menos frecuentes. El objetivo de HusBIT-10 es proporcionar una base para clasificar las tareas del AMPEC-10 en términos de planificación y carga cognitiva, así como poder ser reutilizable en otros experimentos.

## 3.2. MONITORIZACIÓN CON TELÉFONOS MÓVILES

En esta sección se enumerarán distintos artículos de investigación y aplicaciones recopiladas, relacionadas con la monitorización de la actividad física de los usuarios y de la interacción con los dispositivos móviles.

### 3.2.1. Actividad física

A continuación, se detallarán artículos de investigación que relacionan el uso de las TICs con la actividad física y aplicaciones existentes en el mercado que permiten monitorizar la actividad física.

#### Artículos de investigación

Existen distintos artículos de investigación que relacionan el uso de las TICs con la actividad física realizada por los usuarios de los dispositivos móviles. Persisten en la importancia que presenta la monitorización de estos aspectos que pueden provocar adicciones a los móviles o cambios de humor y que es posible que afecten a la salud de la población.

En este estudio [10] analizaron la relación entre la frecuencia de la actividad física con los patrones de agresividad, impulsividad, internet y videojuegos. En esta investigación participaron 254 adolescentes provenientes de Colombia con un rango de edad de entre 12 y 17 años. Para ello, aplicaron la Escala de Impulsividad Barratt para niños, el cuestionario de agresividad de Buss & Perry (AQ) y el cuestionario CAGE para Detección de Problemas de Internet y Videojuegos en adolescentes. Los resultados obtenidos afirman que existe una relación entre la actividad física que realizan los usuarios con la variable cognitiva de impulsividad y el uso de internet.

En el año 2015 realizaron una Tesis [35] en la Institución Educativa Secundaria Mariano Melgar de Ayaviri sobre el uso de las TICs y su relación con la actividad física. Se basaron en una muestra probabilística proveniente de 195 estudiantes comprendidos del 1er al 5to año de educación secundaria basándose en la observación y como instrumentos los cuestionarios de cada variable. Los resultados afirman que existe una débil correlación del uso de las TICs con la variable de la actividad física realizada por estos estudiantes, con un correlación de Pearson de 0.233.

#### Aplicaciones existentes en el mercado

Hay diversos artículos que transmiten a los usuarios la necesidad de llevar un control de la actividad física con el fin de tener un mayor conocimiento sobre las reacciones de su cuerpo mientras practican deporte. La tecnología ha avanzado de tal forma que se ofrecen un gran número de aplicaciones que permiten detectar la actividad física por el usuario y otros datos como el número de pasos dados. Es necesario conocer este abanico de opciones y de alternativas con el propósito de integrarlo o implementarlo en este proyecto para permitir detectar qué actividad física está realizando el usuario mientras interacciona con el dispositivo móvil. En la Tabla 3.1 se muestra una comparación de algunas de las aplicaciones existentes en el mercado que permiten detectar las actividades físicas y contar el número de pasos dados. En el Anexo G se detalla el objetivo y características principales de cada aplicación.

**Tabla 3.1:** Tabla comparativa de aplicaciones que monitorizan la actividad física

Característica \ App	Adidas Running by Runtastic	RunKeeper	Google Fit	Seguimiento de pasos - podómetro
<b>Sistema Operativo</b>	Android e iOS	Android e iOS	Android e iOS	Android
<b>Mostrar estadísticas en gráficos</b>	Sí	No	Sí	No
<b>Registrar ruta/ Seguimiento</b>	Sí	Sí	Sí	Sí
<b>Sincronización relojes</b>	Sí, en marcas Polar y Garmin	No	Sí	No
<b>Nº de pasos</b>	No	No	Sí	Sí
<b>Distancia recorrida</b>	Sí	Sí	Sí	Sí
<b>Tiempo registrado</b>	Sí	Sí	Sí	Sí
<b>Cálculo de calorías</b>	Sí	Sí	Sí	Sí
<b>Desafíos/Records</b>	Sí	Sí	Sí	Sí

### 3.2.2. Interacción con el dispositivo

A continuación, se detallarán artículos de investigación que estudian la interacción de las personas con los dispositivos móviles y aplicaciones existentes en el mercado que permiten monitorizar eventos producidos de esas interacciones.

#### Artículos de investigación

Los dispositivos móviles se han convertido en el objeto principal de nuestro día a día. Pero, según aumenta el uso de los *smartphones*, crecen las preocupaciones por el impacto negativo que supone este uso excesivo. Se han realizado muchas investigaciones para evaluar esas adicciones a los dispositivos móviles y cómo afectan a nuestra vida y lo más importante, a nuestra salud. A continuación, se explicarán algunas investigaciones realizadas, centrándonos en qué y cómo se investigó y qué resultados se obtuvieron.

En este estudio [32], se propuso una metodología de análisis de datos basadas en reglas asociativas que permitían descubrir de forma automática los hábitos de los usuarios a partir del uso que se hace de los dispositivos móviles. Cada sesión comenzaba cuando se encendía la pantalla y terminaba cuando se apagaba o se bloqueaba el dispositivo. Para cada sesión recogían información contextual como el periodo (día de trabajo o vacaciones), una franja horaria de 2 horas, la ubicación del usuario y la actividad del usuario. El resultado de ese conjunto de datos se utilizaba para asociarlo a una serie de reglas o hábitos a través de un algoritmo, denominado *A priori*. Encontraron tres categorías principales de hábitos basadas en:

- **Hábitos de contexto**, que indican una fuerte relación entre el contexto y el uso de las aplicaciones móviles. Por ejemplo, en un día de trabajo entre las 10-12, la aplicación más usada fue Facebook.
- **Hábitos de uso de aplicaciones que están fuertemente relacionadas**, ya que el uso de una estimula el uso de otra. Como ocurre con WhatsApp, Twitter y Facebook.

**■ Hábitos en los que según un contexto dado, estimula el uso de una o más aplicaciones.**

Por ejemplo, durante el trabajo en un periodo entre las 2 y las 4 PM, se estimula el uso de Chrome e Instagram.

Esta metodología fue evaluada en aproximadamente 130.000 sesiones del uso del móvil recogidas de 35 usuarios. Analizando los resultados, obtuvieron que algunos usuarios no presentaban ningún hábito y otros demostraron hasta 100 hábitos complejos con una gran variedad de aplicaciones utilizadas, llegando a la conclusión de que el uso de los *smartphones* se puede caracterizar por distintos tipos de hábitos complejos que varían en función de los usuarios y de las aplicaciones. Por último, desarrollaron una aplicación móvil que sigue la metodología que propusieron para ayudar a los usuarios a cambiar sus hábitos del uso de los dispositivos móviles.

En otro estudio [40] examinaron cuento tiempo debe dedicarse para realizar una medición del uso de los teléfonos móviles y poder inferir de manera fiable los patrones generales de su uso y las conductas repetitivas. Primero hay que tener en cuenta si el usuario hace un uso diario y estable y después saber el número de datos que se necesitan para inferir las acciones comunes realizadas por los usuarios. En este estudio se recogieron datos de 27 estudiantes y personal administrativo, técnico y académico de la Universidad de Lincoln, con una edad media de 22.52 años. Los datos fueron recogidos a través de una aplicación Android que integraba el *framework* de sensores FUNF [6]. Detectaron acciones como el bloqueo o desbloqueo de la pantalla y otras actividades como llamadas, juegos o música. De estos datos, obtuvieron el número total de horas de uso y la frecuencia de uso al día. El estudio duró 13 días y calcularon el número total de uso para cada semana y cada día, llegando a la conclusión de que el promedio de tiempo de uso diario era similar entre un día de un fin de semana y un día de la semana y que únicamente era necesario recopilar los datos una semana para determinar el uso y los comportamientos de la semana siguiente.

En este otro estudio [28] propusieron un sistema que permitía la detección multimodal de estados de depresión a partir de la interacción con los teléfonos móviles y que pudiera ser utilizada para anticiparse a estados de depresión. Estos estados depresivos los cuantificaban por medio de un cuestionario realizado cada 14 días. Para estudiar la relación de los estados depresivos de los usuarios con su comportamiento de interacción móvil, desarrollaron una aplicación en Android y recogieron datos de 25 usuarios durante un periodo de 30 días. Esta aplicación recogía datos sobre el manejo de notificaciones, uso de aplicaciones y del teléfono, y mediante un conjunto de métricas recogidas de la interacción de los usuarios con los teléfonos móviles, pudieron analizar los datos recogidos. Finalmente, calculaban los coeficientes de correlación para analizar la relación de la puntuación del estado depresivo ofrecidas por los cuestionarios, con las métricas de interacción con el dispositivo móvil. Tras la realización del estudio, llegaron a la conclusión que los datos recibidos en los últimos 14 días, ofrecen mejor precisión de la predicción de la puntuación del nivel de depresión respecto a los del día actual.

Existe otro estudio [20] relacionado con el anterior, en el que desarrollaron un sistema capaz de reconocer estados depresivos y maníacos, y que permitiese detectar cambios de estado de pacientes que sufren trastorno bipolar. Registraron durante 12 semanas distintos datos de cuatro modalidades de detección diferentes de la vida real de diez pacientes provenientes de sensores del dispositivo como el acelerómetro o GPS, pero también mediante datos de interacción social que extraen a

través de llamadas de teléfono, como su duración o el número de estas. Detectaron que en estos 800 días rastrearon 17 cambios de estado y con las características monitorizadas, obtuvieron un reconocimiento de estos estados del 76 % y una precisión de cambios de estado de más del 97 %.

Los fabricantes de dispositivos Android y Apple han desarrollado herramientas que ofrecen a los usuarios estadísticas del uso del móvil y el tiempo empleado usando cada aplicación. No obstante, en este estudio [42] proponen un sistema de reconocimiento de contexto basado en el comportamiento de interacción del usuario con el dispositivo. En concreto, identifica seis contextos y calcula la duración de la experiencia del usuario identificada en cada contexto dentro de una aplicación, diferenciando distintos contextos como ver vídeos, jugar, comprar, navegar, chatear y estudiar.

Por último, en el siguiente estudio [26] muestran la importancia de la monitorización de información, a partir de la interacción de los usuarios con los dispositivos móviles para reconocer la adicción a los mismos y las técnicas o mecanismos que podrían evitar que los usuarios sufran síndromes relacionados con su uso. Para ello, presentan el desarrollo de un modelo de clasificación para el sistema de reconocimiento de adicción a los *smartphones*, que incluye el diseño de la arquitectura del sistema y el modelo de clasificación. Para realizar este estudio, desarrollaron una aplicación Android que se encarga de recoger los datos cada 30 minutos y los preprocesa antes de enviarlos a un servicio en la nube. Las distintas métricas calculadas a partir de los datos recogidos son:

1. Tiempo promedio del uso del móvil en cada desbloqueo
2. Máximo tiempo de uso del móvil en cada desbloqueo
3. Mínimo tiempo de uso del móvil en cada desbloqueo
4. Tiempo total del uso del teléfono móvil
5. Cantidad de pasos dados
6. Periodo de tiempo (6.00-12.00, 12.00-18.00, 18.00-24.00, 24.00-6.00)
7. Número de veces que se ha desbloqueo el teléfono móvil.

El resultado del desarrollo tuvo una precisión de hasta el 78,74 % en reconocer si su teléfono inteligente está inactivo y si presenta una alta o baja probabilidad de adicción a los *smartphones*. Los resultados obtenidos proporcionaban oportunidades de mejora de los teléfonos móviles y prevenir que aumente la adicción de los usuarios a los dispositivos móviles.

### **Aplicaciones existentes en el mercado**

Actualmente, existen dos métodos para saber cuáles son las aplicaciones más usadas en nuestro dispositivo móvil y el tiempo de uso que se ha hecho de esa app. El primer método está integrado en los móviles que presentan un sistema operativo Android y el segundo método son aplicaciones de terceros que se pueden descargar de Google Play Store. La opción más básica que se corresponde con el primer método, consiste en ver el tiempo de uso de tus aplicaciones en los ajustes del teléfono. En la sección de batería aparecerán las apps que han sido recientemente utilizadas junto con el tiempo de uso de las mismas. La opción más avanzada es una aplicación que nos presenta un conjunto de gráficas y estadísticas sobre el tiempo de uso de las aplicaciones y nos proporciona información más estructurada. Algunas de las aplicaciones más usadas y valoradas que permiten ofrecer esta información, se muestran detalladamente en el Anexo G. No obstante, en la Tabla 3.2 se muestra una

comparación de las características que presentan cada una de ellas.

Existen otras aplicaciones específicas que permiten monitorizar las actividades de los usuarios y que son brindadas a cualquier tipo de colectivo, especialmente a padres que presentan inconvenientes para controlar a sus hijos y evitar adicciones a los dispositivos móviles o problemas de salud. Algunas de ellas son:

- **FamiSafe**<sup>1</sup>. Es una aplicación disponible para iOS y Android, que permite monitorizar el tiempo de uso de las aplicaciones usadas y analizar su uso. Además, permite rastrear la ubicación del dispositivo y ver un historial de los sitios visitados.
- **Spyzie**<sup>2</sup>. Es una aplicación disponible tanto para dispositivos Android como iOS y que ayuda a proporcionar información sobre el historial de llamadas, de navegación, mensajes de texto y de WhatsApp, archivos multimedia etc.
- **mSpy**<sup>3</sup>. Esta aplicación de monitorización, está disponible para iOS y Android y proporciona la siguiente información:
  - Registros de llamadas entrantes/salientes con información sobre el contacto, marcas de tiempo y duración de la llamada.
  - Actividad de uso de las redes sociales como Instagram, Facebook o Twitter.
  - Actividad del correo electrónico, calendario, multimedia etc.

### 3.2.3. Conclusiones

Haciendo una comparación entre el sistema BIPapp y los artículos de investigación que estudian la relación de la actividad de los usuarios con el uso de las TICs, junto con las aplicaciones existentes en el mercado que permiten monitorizar la actividad física, se han encontrado diferentes aspectos que no se contemplan en estos artículos y aplicaciones. En el sistema BIPapp, más allá de intentar monitorizar la actividad física de los usuarios y contar el número de pasos, se pretende detectar el tipo de actividad que se está realizando mientras realizas acciones como llamadas telefónicas, escribir letras, símbolos u emoticonos, llamadas y videollamadas de WhatsApp o el uso de aplicaciones concretas como WhatsApp, Instagram, Facebook, Twitter, Linkedin, Gmail y Outlook.

Por otro lado, el sistema BIPapp, en comparación con los artículos de investigación mencionados que estudian la interacción de las personas con los dispositivos móviles y aplicaciones que permiten monitorizar eventos producidos de esas interacciones, recoge una gran diversidad de datos. Mientras que en estas investigaciones y artículos únicamente se basan en recoger datos del tiempo de uso de aplicaciones desde que se desbloquea el dispositivo hasta que se vuelve a bloquear para obtener un conjunto de métricas, en BIPapp se recogen otros datos como frecuencia de pulsación de botones físicos, frecuencia de pulsación de letras, símbolos u emoticonos, tiempo de visualización de vídeos y fotos en WhatsApp, eventos más comunes realizados en las redes sociales, entre otros datos.

Es por ello, que este sistema permite monitorizar a gran escala los eventos realizados por el usuario para estudiar distintos aspectos del uso que hacen de los dispositivos móviles.

---

<sup>1</sup><https://famisafe.wondershare.com/es/>

<sup>2</sup><https://www.spyzie.com/es/>

<sup>3</sup><https://www.msipy.com/>

Tabla 3.2: Tabla comparativa de aplicaciones de monitorización de eventos del dispositivo

Característica	App	Analizador de uso	Apps usage	Your Hour	MyAddictometer	AntiSocial	Smartphonoholic	StayFree
<b>Sistema Operativo</b>	Android	Android	Android	Android e iOS	Android e iOS	Android	Android	Android
<b>Gráficos acotados por tiempo móvil</b>	Sí	Sí	Sí	Sí	Sí	Sí	Sí	Sí
<b>Tiempo de inicio de uso de cada app</b>	Sí	No	Sí	No	No	No	Sí	Sí
<b>Nº de inicialización de una app</b>	No	Sí	Sí	No	Sí	No	Sí	Sí
<b>Nº bloqueo/desbloqueo</b>	No	No	Sí	No	Sí	No	Sí	Sí
<b>Tiempo de uso en categorías</b>	No	Sí	No	No	Sí, solo las RSS	No	Sí	Sí
<b>Historial de apps instaladas</b>	No	Sí	No	No	No	No	No	No
<b>Exportar datos</b>	No	Sí, CSV	Sí, PDF	Sí, CSV	No	No	Sí, CSV	Sí, CSV
<b>Seleccionar apps a monitorizar</b>	No	Sí	No	Sí	No	No	Sí	Sí
<b>Historial de batería o notificaciones</b>	No	Sí	No	No	No	No	No	No
<b>Uso de datos de cada app</b>	Sí	No	No	No	No	No	No	No
<b>Permisos otorgados del sistema a cada app</b>	Sí	No	No	No	No	No	No	No
<b>Almacenamiento usado por cada app</b>	Sí	No	No	No	No	No	No	No

---

## CAPÍTULO 4

# METODOLOGÍA

---

En este capítulo se va a explicar de forma detallada la metodología de desarrollo que se ha seguido para realizar este Trabajo de Fin de Grado. Se especificarán las características y los principios básicos de la metodología que se ha escogido y la forma en que las distintas etapas se han adaptado a este proyecto.

### 4.1. METODOLOGÍA DE TRABAJO

Las metodologías ágiles son un conjunto de métodos y principios diseñados para la gestión de proyectos de desarrollo *software* propuesto en el año 2001 que ofrece ventajas respecto a las metodologías tradicionales. Se diferencian en que las metodologías ágiles se adaptan mejor a los cambios y se centran en integrar al cliente en el proceso de desarrollo del *software*, haciendo una planificación al principio y qué se irá incrementando en las distintas iteraciones o *sprints*. Al final de cada iteración, el cliente puede ver el desarrollo del proyecto y puede sugerir modificaciones desde etapas iniciales.

#### 4.1.1. Manifiesto ágil

El grupo pionero de estas metodologías incorporaron una serie de principios en un manifiesto, denominado manifiesto ágil [8]. En él, establecieron las bases que cualquier metodología ágil debe cumplir:

- **Individuos e interacciones sobre procesos y herramientas.** A lo largo de la historia los procesos y las herramientas han ido evolucionando y han facilitado la creación de proyectos con éxito. Sin embargo, son las personas el componente más importante del manifiesto, porque son las que toman decisiones sobre qué herramientas utilizar o qué procesos se llevan a cabo, proponen iniciativas de cambio, participan, implementan el proyecto y se adaptan a los cambios rápidamente.
- **Software funcionando sobre documentación extensiva.** Las metodologías ágiles se basan en entregar el *software* funcionando, donde en cada incremento se añadan nuevas funcionalidades a lo largo del ciclo de vida del producto y no crear documentación extensa e innecesaria que supone una gran carga de trabajo y aporta poco valor al producto final.

- **Colaboración con el cliente sobre negociación contractual.** El producto final tiene que satisfacer todas las necesidades y requisitos del cliente, es por ello, que los clientes tienen que involucrarse en el proyecto como colaboradores para que al final de cada iteración evalúen el desarrollo y propongan mejoras.
- **Responder ante el cambio sobre seguir un plan.** Este tipo de metodologías se adapta a este proyecto de fin de grado porque los requisitos no están bien definidos desde el inicio del proyecto y pueden surgir muchos cambios a los que habrá que anticiparse para saber cómo actuar y adaptarse a esos cambios. Esto es lo que le hace ser diferente a las metodologías tradicionales basadas en una planificación y en un control que evite desviaciones del plan establecido.

Existen varias metodologías ágiles, entre las más conocidas y usadas están Scrum<sup>1</sup>, eXtreme Programming<sup>2</sup> y Kanban<sup>3</sup>. En este proyecto se ha decidido seguir esta última metodología, debido a que presenta unos requisitos cambiantes que se irán decidiendo según avance el desarrollo del proyecto basado en un conjunto de tareas que pasarán por distintas etapas de desarrollo y en el que se requiere comunicación y validación continua del cliente. Somos conscientes de que Kanban no es estrictamente una metodología de desarrollo *software*. Por ello, en la sección 4.2 se comenta cómo se ha integrado Kanban con el desarrollo iterativo e incremental para conformar la metodología de trabajo de este proyecto.

#### 4.1.2. Kanban

La metodología Kanban surgió a principios del siglo XX de la mano de un Ingeniero Industrial japonés de la empresa de automoción Toyota. A diferencia de otras metodologías ágiles de desarrollo *software* como Scrum o eXtreme Programming, Kanban fue creada para conseguir una mayor eficiencia en la producción de automóviles de esta compañía. Esta empresa se ha convertido en un referente en la industria gracias a esta metodología y es por ello que en el año 2004 se utilizase con éxito en un proyecto de Tecnologías de la Información de Microsoft<sup>4</sup>.

#### Fases de la metodología Kanban

La metodología Kanban se organiza en un tablero dividido en columnas (Figura 4.1). El número de columnas puede variar dependiendo del nivel de complejidad del proceso. Se debe delimitar el número de tareas en cada fase del desarrollo para poder identificar posibles cuellos de botella e intentar terminar y cerrar tareas para poder iniciar otras. Las fases más usadas en el desarrollo *software* [38] son:

- **Lista de tareas – To do.** En esta primera columna se incluye una lista de tareas pendientes y que se pueden realizar sucesivamente. En esa lista se deben de ordenar las tareas según la prioridad que se les ha asignado, colocando arriba las que tengan mayor prioridad y en orden descendente, colocar el resto.

<sup>1</sup><https://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-es.pdf>

<sup>2</sup><http://www.extremeprogramming.org/>

<sup>3</sup><https://kanbantool.com/es/guia-kanban/historia-de-kanban>

<sup>4</sup><https://kanbantool.com/es/guia-kanban/historia-de-kanban>

- **En desarrollo – Development.** En la segunda columna se incluyen todas las tareas que se están desarrollando hasta que se completen correctamente. Si se produce algún problema, esta tarea tendría que retroceder a la columna anterior.
- **Pruebas - Test.** Se realizan comprobaciones sobre el correcto funcionamiento del sistema para comprobar si la tarea se ha realizado de forma exitosa.
- **Despliegue - Deployment.** Cuando ya se ha *testeado* el código, esa tarea pasará a esta columna para llevar a cabo su subida a producción en el sistema en forma de incremento.
- **Terminado – Done.** En esta última columna se incluyen las tareas que se han finalizado.

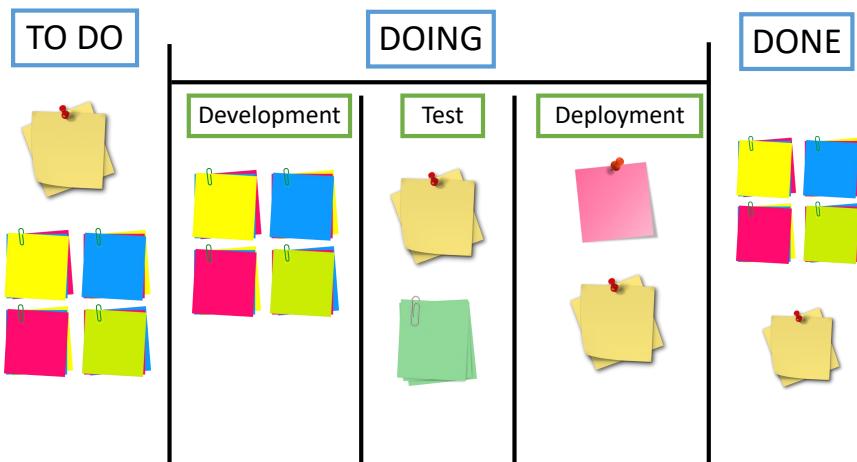


Figura 4.1: Fases de la metodología Kanban

### Principios básicos de la metodología Kanban

David J. Anderson es la persona que formuló la metodología Kanban como una aproximación al proceso iterativo e incremental. Este método de trabajo fue enfocado para llevar a cabo las tareas pendientes y se estructura en los siguientes principios [17]:

- **Calidad garantizada.** Todo el desarrollo debe hacerse bien desde el principio, ya que esta metodología no acepta un margen de error. En Kanban lo más importante es la calidad de las tareas realizadas, para qué, si se produce un cambio en los requisitos, se pueda adaptar fácilmente.
- **Reducción del desperdicio.** Kanban no requiere configuraciones y se basa en hacer bien únicamente lo que se precise.
- **Mejora continua.** Además de ser un método de gestión de proyectos, es un sistema que propone la mejora continua de los proyectos según los objetivos planteados. Propone pequeños y continuos cambios incrementales y evolutivos en el proyecto que eviten los cambios radicales.
- **Flexibilidad.** Cuando se termina una tarea de cada iteración, se evalúa y se comprueba su funcionamiento para decidir qué tareas pendientes se deben realizar posteriormente y se priorizan aquellas tareas que han podido surgir según los errores encontrados. De esta forma se puede hacer frente a improvisaciones y evitar desviaciones en el plan.

## Prácticas en Kanban

Existen seis prácticas identificadas por David J. Anderson [23] que deben estar presentes en un proyecto en el que se aplique esta metodología:

- **Visualizar el flujo de trabajo.** La primera fase en la metodología Kanban consiste en elaborar mediante un tablero la visión general de todas las tareas del proyecto y el flujo o fases de trabajos por las que pasarán. Las distintas etapas serán las columnas y las tarjetas sean las tareas que representan una tarea que hay que desarrollar. De esta forma se podrá hacer un seguimiento de forma visual y detectar posibles cuellos de botella que impiden el correcto desarrollo del sistema.
- **Eliminar las interrupciones.** Las tres columnas intermedias del tablero deben contar con un número máximo de tareas que se deberán ir completando para dar paso a nuevas tareas. Es necesario establecer límites de trabajo en cada etapa para asegurarnos que el trabajo fluye.
- **Gestionar el flujo.** La idea de Kanban es crear un flujo de trabajo continuo que se preocupe de solucionar problemas en situaciones de bloqueos y saturaciones. Se desea que un proyecto esté constantemente creando valor y minimizar el riesgo de posibles incrementos de coste por posibles retrasos en la entrega.
- **Regulaciones.** Todos los procesos y tareas tienen que estar bien definidas para que el equipo entienda el objetivo general del proyecto y puedan trabajar y tomar decisiones al respecto.
- **Circuitos de retroalimentación (Feedback).** Basándose en los valores de las metodologías agiles donde se centran en dar pequeños pasos en forma de iteraciones para entregar constantemente *software* funcionado al cliente, es importante que se realicen reuniones periódicas para recibir *feedback* o críticas por parte del cliente y del *manager* para poder corregir posibles errores rápidamente y compartir constantemente las funciones implementadas.
- **Mejorar colaborando.** Para alcanzar una mejora continua en el desarrollo del proyecto es necesario que los miembros del equipo se esfuerzen por compartir sus opiniones y visiones sobre el flujo de trabajo y los riesgos, para afrontar mejor los problemas y llegar a un consenso sobre posibles soluciones.

## Roles de la metodología Kanban

En Kanban, a diferencia de otras metodologías como eXtreme Programming o Scrum, no existen roles definidos, ya que en muchas ocasiones se consideran innecesarios. Es uno de los motivos por lo que se ha decidido seguir esta metodología, ya que al ser un equipo de desarrollo integrado por una única persona y dos tutores del proyecto, es contradictorio aplicar esas metodologías diseñadas para equipos de un mayor número de componentes. El método Kanban deja que los miembros del equipo se organicen en torno al trabajo y que entre ellos permitan que el trabajo fluya. No obstante, David Anderson junto con su equipo de Lean Kanban University, proponen incorporar dos roles para que la metodología sea más efectiva.

- **Flow Master o Service Delivery Manager.** Es un rol encargado de ayudar al equipo de desarrollo para hacer que las tareas se realicen de forma fluida a lo largo del tablero ajustándose al tiempo establecido. Facilita al equipo en la toma de decisiones para que el sistema funcione.
- **Service Request Manager.** Es el encargado de gestionar los requisitos dentro del proyecto,

manejando las relaciones con el cliente y observando la priorización de las tareas.

### Comparación entre distintas metodologías ágiles

Scrum hace hincapié en la rapidez de los procesos y el control del tiempo por parte del gestor del equipo, pero ese control exige realizar reuniones constantes y discusiones que no hace tan ágil el desarrollo de un proyecto. Sin embargo con Kanban, el Service Request Manager únicamente tiene que determinar las tareas que hay que realizar y asignarles una prioridad que puede cambiar en función de cada situación. Con esta metodología se produce un flujo de trabajo constante y más organizado, que permite aumentar la productividad y la calidad del producto final. No obstante, todas las metodologías ágiles pueden complementarse entre sí y en este proyecto se han incorporado principios de la metodología eXtreme Programming para agregar valor y calidad al proyecto, como la refactorización del código, cambios en los requisitos y comunicación continua con el cliente.

## 4.2. APLICACIÓN DE LA METODOLOGÍA KANBAN AL PROYECTO

El equipo de este proyecto está formado por dos tutores y una alumna, que es a su vez la autora de este Trabajo de Fin de Grado. Los roles y las responsabilidades propuestas por Kanban que se han realizado son las siguientes:

- **Service Delivery Manager.** Este rol ha sido desempeñado por los tutores de este TFG, que se han encargado de guiar en todo momento el desarrollo del sistema y programar reuniones presenciales y no presenciales mediante la herramienta Microsoft Teams.
- **Service request manager y desarrollador.** Este rol ha sido desempeñado por la autora de este Trabajo de Fin de Grado, que se ha encargado de estimar el tiempo de cada tarea, priorizarlas para ajustarse a la planificación, entender las necesidades de los clientes para llevarlas a cabo, diseñar la solución, tomar decisiones en la implementación y desarrollar funcionalidades.

Kanban presenta un conjunto de características fundamentales para llevar a cabo esta metodología, pero debido a las particularidades y a un equipo de desarrollo muy limitado, se han llevado a cabo las siguientes prácticas:

- **Desarrollo incremental e iterativo.** Las metodologías agiles siguen un desarrollo iterativo e incremental. Esto significa que el proyecto se divide inicialmente en fases temporales llamadas iteraciones. En cada iteración se lleva a cabo el mismo procedimiento y siempre termina con la entrega del producto al cliente para obtener una retroalimentación de este y poder evitar fallos en etapas posteriores que aumenten los costes y el esfuerzo. Es incremental porque cada parte del proyecto que se complete en cada iteración se unirá al proyecto final y este irá avanzando hasta alcanzar en la única entrega el sistema definitivo. Es importante que cuando una iteración finaliza, se entregue una parte funcional del proyecto al cliente para que pueda practicar con ella completamente y para ello es primordial dividir el proyecto en partes correctamente para evitar ejecutar un modelo tradicional en cascada. Gracias al desarrollo incremental e iterativo, en este proyecto ha sido posible tener una buena comunicación con el cliente, que en todo momento ha transmitido sus opiniones y *feedback* realizando un proyecto

que se ajuste a sus características y necesidades. Además, al ser un desarrollo incremental se ha podido dividir el proyecto en distintas tareas que no se interponen entre ellas, como por ejemplo la manipulación de la información mediante de la *API REST* y la consumición de esa *API REST* para la representación de los datos. Finalmente, estas partes se han integrado y han permitido contemplar resultados similares al sistema final.

- **Calidad garantizada y mejora continua.** En este proyecto, todas las tareas se debían realizar correctamente desde el principio y si se produce un cambio en los requisitos o una tarea depende de otra, no pasa de la fase de desarrollo hasta que se realicen pruebas conjuntas. Por otro lado, se ha seguido una mejora continua debido a los continuos incrementos que se han producido en el sistema evitando los cambios radicales.
- **Flexibilidad y cambios en los requisitos.** Debido a los requisitos iniciales de este TFG que en su momento no se tenía certeza de si se pudiesen llevar a cabo, han sufrido ligeras modificaciones y se han incorporado otras funcionalidades, adaptándose a los plazos de tiempo y dificultades encontradas.
- **Reducción del desperdicio.** Para aumentar la legibilidad y la mantenibilidad del código, sobre todo de la aplicación móvil, se han hecho refactorizaciones del código durante el desarrollo y se han reescrito ciertas partes, con el objetivo de que el código realice únicamente lo que se precise.
- **Comunicación continua y evaluación del cliente.** A lo largo del proyecto, se han llevado a cabo diferentes reuniones, tanto presenciales como no presenciales, entre cliente y el equipo de desarrollo para aportar *feedback* y opiniones del proyecto. Se realizó una reunión inicial para analizar los requisitos iniciales del proyecto y la elaboración de una planificación. Posteriormente se realizaron varias reuniones para analizar el progreso del proyecto y *test* para comprobar sus funcionalidades.

### 4.3. PLANIFICACIÓN E ITERACIONES

En este proyecto, como se mencionaba anteriormente, se ha seguido una metodología ágil basada en un desarrollo iterativo e incremental. Los distintos objetivos teóricos y prácticos que se definieron en el Capítulo 2, se muestran en el diagrama de Gantt de la Figura 4.2. Como se puede observar la mayor parte de las tareas se solapan y este es el hecho principal por el que se sigue un prototipo iterativo e incremental y no un modelo tradicional o en cascada.



Figura 4.2: Planificación de los objetivos entre 04/02/2020 - 30/06/2020

Estos objetivos se van a realizar en iteraciones que van a suponer un incremento en el sistema final. Cada una de estas iteraciones se desglosa en un conjunto de tareas que se desarrollarán siguiendo la metodología Kanban, pasando por las distintas fases de desarrollo. En la Figura 4.3 se muestra

un Diagrama de Gantt con el tiempo estimado en días y semanas de cada iteración suponiendo un trabajo diario medio de 3 horas al día desde el inicio del proyecto.

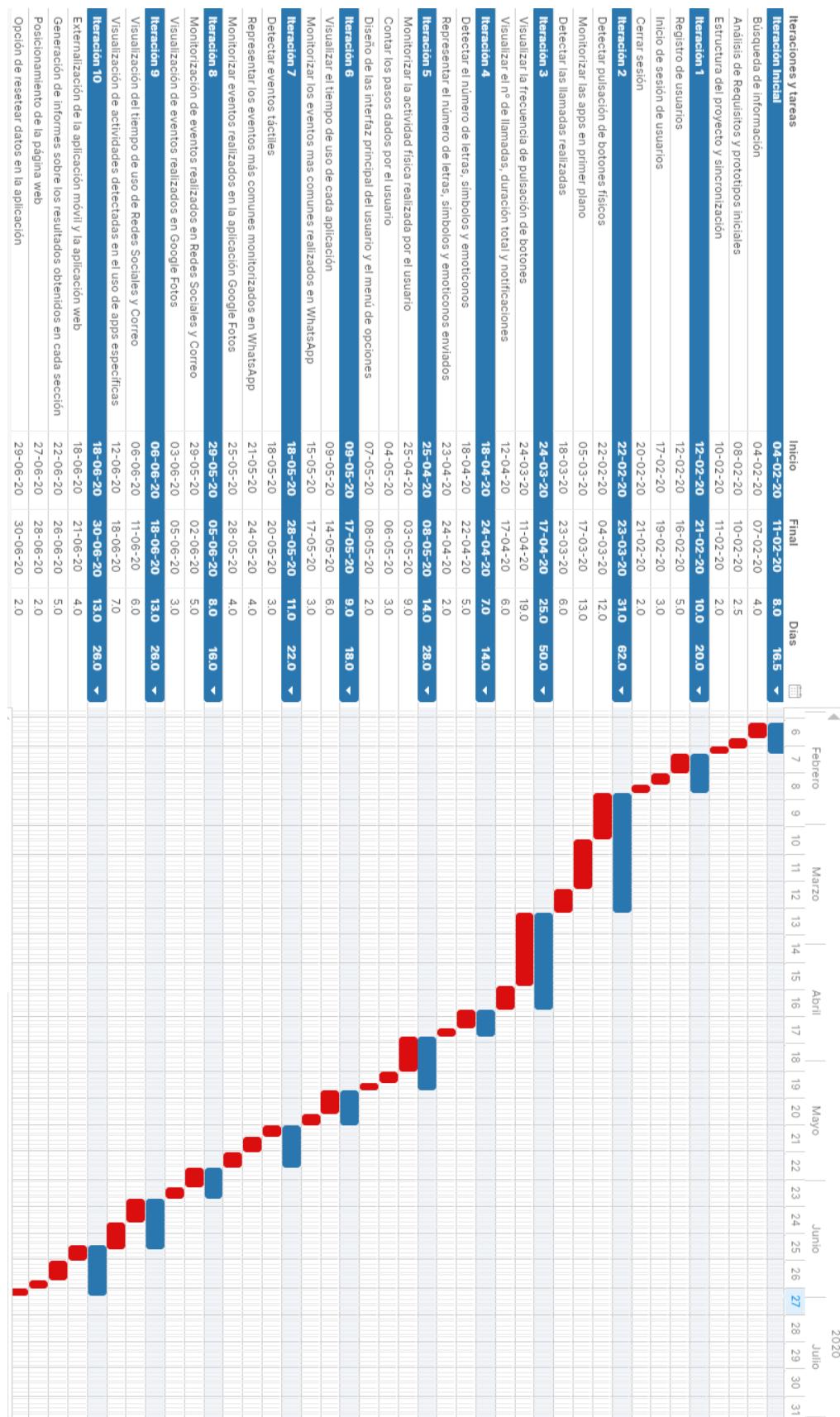


Figura 4.3: Diagrama de Gantt

A continuación, se van a mencionar las distintas actividades que son necesarias realizar para la construcción del sistema y que se desglosan a partir de cada tarea de su iteración correspondiente.

### **Iteración inicial**

- Planificación y estudio de la viabilidad del proyecto
- Análisis de requisitos funcionales y no funcionales
- Prototipos iniciales, estructura del proyecto y sincronización

### **Iteración 1**

#### **Tarea 1. Registro de usuarios**

- Búsqueda de información sobre las distintas formas de hacer un registro de usuario en Android y creación de una tabla en la base de datos para almacenar los datos de los usuarios.

#### **Tarea 2. Inicio de sesión de usuarios**

- Búsqueda de información para la creación de un inicio de sesión de usuarios en Android que compruebe si existe el usuario en la base de datos y opción de mantener la sesión iniciada.

#### **Tarea 3. Cerrar sesión.**

- Investigación de posibles formas para permitir al usuario cerrar sesión.

### **Iteración 2**

#### **Tarea 4. Detectar aplicaciones en primer plano**

- Configuración del servicio de accesibilidad para que permita detectar las aplicaciones que se ejecutan en primer plano.
- Generación de un *script* que permita almacenar todas las aplicaciones que se ejecutan en primer plano en un momento determinado, junto con el id del usuario y la fecha y hora de su inicio.
- Generación de la tabla *primerPlano* en la base de datos con los campos mencionados anteriormente.

#### **Tarea 5. Detectar la pulsación de botones *hardware***

- Investigar sobre distintas formas y medios para detectar eventos realizados por el usuario.
- Configuración e implementación del servicio de accesibilidad y construcción del método *onKeyEvent* que permite detectar la pulsación de los botones subir y bajar volumen.
- Estudiar e implementar distintas formas para detectar el bloqueo y desbloqueo de la pantalla.
- Generación de un método que permita enviar los datos a la base de datos y un *script* que establezca la conexión con ella e inserte los datos.
- Generación de la tabla *botones* que contiene el evento realizado por el usuario, su identificador y la fecha y hora en la que ocurrió el evento.
- Diseño en la interfaz de usuario un *checkbox* para permitir al usuario iniciar el servicio de accesibilidad o detenerlo.

**Tarea 6. Monitorizar las llamadas realizadas**

- Estudio de los eventos que se producen al detectar una aplicación en primer plano en relación con las llamadas e incluir casos alternativos que no se traten.
- Almacenar los datos en la tabla *primerPlano* para calcular el tiempo de duración de las llamadas y el número de llamadas realizadas.

**Iteración 3****Tarea 7. Visualizar la frecuencia de pulsación de los botones físicos**

- Investigación de distintos *Frameworks* para la creación de *API REST* en PHP y generación de la API con las primeras consultas a la base de datos para obtener en primer lugar, la frecuencia de pulsación de los botones físicos y poder representarlos en los gráficos.
- Generación del esquema básico de la aplicación web, incluido el título, menú y *footer* con los lenguajes HTML y CSS y estudio de distintas librerías JavaScript para la generación de gráficos y selección de la que más ventajas y facilidades aporte al proyecto.
- Investigación de las distintas peticiones posibles para consumir una *API REST* en JavaScript.
- Representación de los primeros gráficos con elementos *canvas* y peticiones AJAX a la API.

**Tarea 8. Visualizar el número de llamadas, duración y notificaciones del sistema recibidas.**

- Estudio de distintas formas para integrar la aplicación web en la aplicación móvil e implementar la solución seleccionada.
- Investigación de distintos tipos de gráficos, con barras acumuladas y gráficos de varias líneas para representar los gráficos acotados por semana e histórico.
- Procesamiento de la información de la base de datos mediante la *API REST* para calcular el número de llamadas y la duración que tiene cada una de ellas acotadas por tiempo.
- Representación en gráficos del número de llamadas y duración de las mismas en una semana e histórico.
- Creación de una tabla para añadir el número de notificaciones recibidas en el sistema y el número de llamadas, junto con la duración en el día actual.

**Iteración 4****Tarea 9. Detectar el número de letras, símbolos y emoticonos enviados, junto con la frecuencia de pulsación.**

- Configurar el servicio de accesibilidad para que permita reconocer la pulsación de una letra, símbolos u emoticonos.
- Enviar a la base de datos letra, símbolo u emoticono cada vez que ocurra un evento de este tipo para evitar guardar información confidencial.

**Tarea 10. Visualizar el número de letras, símbolos y emoticonos enviados**

- Calcular mediante consultas a la base de datos, el número de letras, símbolos u emoticonos enviados hoy, en una semana y en el histórico.

- Calcular la frecuencia del número de elementos enviados por segundo y el tiempo que se ha empleado, y visualizar en una tabla todos los datos recogidos.

## Iteración 5

### Tarea 11. Detectar la actividad física realizada por el usuario

- Investigación de todas las API proporcionadas por Android para el reconocimiento de actividad del usuario y los servicios para que la detección de la actividad se realice en segundo plano.
- Implementación de los métodos necesarios para utilizar la API *Activity Recognition* junto con utilidades y constantes que contengan los distintos tipos que son posibles detectar en un *IntentService*.
- Creación de una tabla en la base de datos que registre la actividad realizada por el usuario, junto con el identificador del usuario, fecha y hora en que se produjo ese cambio de actividad.
- Creación de un *script* para enviar los datos desde la aplicación a la base de datos.
- Diseño en la interfaz de usuario un *checkbox* para permitir al usuario iniciar la detección de la actividad física o finalizarlo.

### Tarea 12. Contar los pasos realizados por el usuario

- Investigación de todos los sensores incorporados en los dispositivos u otras alternativas que permitan contar los pasos realizados por el usuario.
- Implementación de sensores como el acelerómetro o *Step Counter* para contar los pasos realizados por el usuario de una forma precisa.
- Diseño en la interfaz de usuario un *checkbox* para permitir al usuario iniciar el contador de pasos o finalizarlo.

### Tarea 13. Diseño de la interfaz del usuario de la aplicación móvil

- Generación de estilos de botones, colores y justificación de párrafos de la aplicación.
- Diseño de estilos de alertas y mensajes para cada acción realizada en la aplicación.
- Creación de un menú con distintas opciones para obtener información sobre BIPapp.

## Iteración 6

### Tarea 14. Visualizar el tiempo de uso de cada aplicación

- Procesamiento de la información mediante la *API REST* y visualización de los gráficos en función del paquete detectado.
- Estudio sobre distintas formas para poder realizar los gráficos del tiempo de uso de las aplicaciones mostrando el nombre de la aplicación en vez el del paquete.
- Implementación de un método que permite reconocer las aplicaciones instaladas por el usuario y detección mediante el servicio de accesibilidad el evento instalar o desinstalar aplicación.
- Generación de una tabla denominada *AppsInstaladas* que incluya el paquete y el nombre de la aplicación, junto con el identificador del usuario y creación de un *script* que permita borrar y añadir las aplicaciones instaladas en el dispositivo del usuario cuando se inicia la aplicación o cuando ocurre un evento de instalar o desinstalar aplicación, con el objetivo de actualizar la lista de aplicaciones que tiene instaladas el usuario.

- Cálculo del tiempo de uso de las aplicaciones comprobando que el nombre del paquete de la actividad detectada en primer plano coincide con un paquete de una aplicación instalada en el dispositivo.
- Visualización de los gráficos en función del nombre de la aplicación y únicamente del tiempo de uso de las aplicaciones instaladas en el dispositivo.

#### **Tarea 15. Monitorizar eventos más comunes realizados en la aplicación WhatsApp**

- Configurar el servicio de accesibilidad para que permita reconocer eventos comunes realizados en WhatsApp, como las llamadas y videollamadas realizadas o el número de elementos enviados.
- Enviar todos los eventos detectados a la tabla *apps* de la base de datos.

#### **Iteración 7**

##### **Tarea 16. Visualizar los eventos más comunes realizados en WhatsApp**

- Procesamiento de la información almacenada en la base de datos mediante la *API REST* y consumición de esos recursos para representar los datos monitoreados en forma de gráficos y tablas del número de notificaciones recibidas, llamadas y videollamadas realizadas junto con su duración y número de elementos enviados acotados por tiempo.

##### **Tarea 17. Monitorizar eventos realizados en la aplicación Google Fotos**

- Configurar el servicio de accesibilidad para que reconozca eventos comunes en la app Google Fotos.
- Enviar todos los eventos detectados a la tabla *primerPlano* de la base de datos para poder calcular el tiempo de visualización de vídeos o imágenes.

#### **Iteración 8**

##### **Tarea 18. Monitorización de eventos comunes realizados en las redes sociales y en aplicaciones de correo electrónico**

- Configurar el servicio de accesibilidad para que permita reconocer eventos comunes en las redes sociales y en el correo electrónico.
- Enviar todos los eventos detectados a la tabla *apps* de la base de datos.

##### **Tarea 19. Visualización de los eventos realizados con Google Fotos**

- Procesamiento de la información almacenada en la base de datos y representación de los datos monitorizados en forma de tabla con el tiempo de uso de la aplicación y el tiempo empleado en la visualización de imágenes y vídeos.

#### **Iteración 9**

##### **Tarea 20. Visualización del tiempo de uso y eventos más comunes de las RRSS y de las apps de correo electrónico**

- Procesamiento de la información almacenada en la base de datos mediante la *API REST* y consumición de esos recursos para representar los datos monitoreados.

**Tarea 21. Visualización de las actividades físicas detectadas en el uso de aplicaciones y eventos concretos**

- Generación de una tabla *detectaActividadesLetra* y *primerPlanoLetras* que permita registrar todas las actividades que se están ejecutando en primer plano y el momento en el que se escribe una letra, para que cuando ocurra ese evento se pueda calcular y visualizar el tiempo y tipo de actividad que está realizando.
- Procesamiento de la información de la base de datos y cálculo del tiempo de cada actividad en aplicaciones concretas como las redes sociales o eventos como realizar llamadas.
- Visualización en forma de gráficos de la información que ha sido procesada.

**Iteración 10****Tarea 22. Opción resetear datos en la aplicación**

- Generación de un botón y un *alert dialog* que permita borrar toda la información del usuario almacenada en la base de datos, incluido su identificador de usuario, mediante un *script* que establezca una conexión con la base de datos.

**Tarea 23. Internacionalizar la aplicación móvil y la aplicación web**

- Investigación sobre distintas alternativas para hacer una aplicación móvil Android y una aplicación web multilenguaje.
- Creación de un archivo *strings.xml* en Inglés para poder incluir todo el texto de la aplicación en inglés, referenciados por su identificador.
- Generación de varios archivos PHP, denominados *es.php* y *en.php*, que incluyen los distintos párrafos de la aplicación web en el lenguaje de español e inglés respectivamente.
- Creación de un *ComboBox* que permite seleccionar y cambiar el lenguaje deseado, siendo posible Español e Inglés en la aplicación web.
- Generación de un método en cada ventana de la aplicación que permite reconocer el lenguaje que se ha seleccionado previamente para mostrar la información en el idioma correspondiente.

**Tarea 24. Generación de informes sobre los resultados obtenidos en cada sección**

- Búsqueda de información sobre distintas librerías JavaScript para la generación de informes PDF a partir del contenido de una ventana.
- Implementación de la librería *jsPDF* y generación de informes incluyendo los gráficos y el texto en función del lenguaje que esté seleccionado.

**Tarea 25. Posicionamiento de la aplicación web**

- Mejorar el posicionamiento de la aplicación web incluyendo en cada sección etiquetas y títulos significativos y diferentes, junto con palabras claves por las que pueda ser indexado en el navegador.
- Crear una página de error 404, que redirija al usuario en caso de error a la página de inicio.
- Incluir la aplicación web en la herramienta *Google Search Console*.

---

## CAPÍTULO 5

---

# RESULTADOS

---

En este capítulo se van a explicar detalladamente todos los pasos que se han llevado a cabo hasta la finalización del proyecto dirigidos por la metodología de desarrollo que se presentó en el Capítulo 4. Siguiendo los pasos de la metodología Kanban, el desarrollo se ha estructurado en varias iteraciones que se desglosan en tareas, comenzando por una iteración inicial de partida donde se fijan las primeras ideas y sucesivamente se explicarán 10 iteraciones donde se implementarán las distintas funcionalidades. En cada sección se explicará como la metodología escogida ha permitido satisfacer los objetivos planteados en el Capítulo 2.

### 5.1. VISIÓN GENERAL

El resultado final de este Trabajo de Fin de Grado está esquemáticamente representado en la Figura 5.1. Inicialmente, cuando el usuario se ha instalado la aplicación BIPapp en su dispositivo móvil podrá registrarse o iniciar sesión en el sistema (1). Una vez que se haya registrado en el sistema, tendrá la posibilidad de iniciar sesión y mantener la sesión iniciada, que si se inicia correctamente pasará a la siguiente actividad donde se encuentran las distintas opciones que se podrán habilitar (2) para monitorear los eventos del usuario (4). Tanto la opción de reconocer la actividad del usuario como la de contar pasos, son servicios que se inician desde esta interfaz de usuario. Sin embargo, la opción de monitorear acciones se corresponde con un servicio de accesibilidad que el usuario debe habilitar en los ajustes del teléfono y qué es redirigido automáticamente cuando el usuario selecciona esta opción (3).

Una vez que el usuario ha habilitado las opciones que deseé, todos los datos de los eventos que se produzcan se guardan en la base de datos (5) y estos datos son accedidos y procesados mediante una *API REST*. El usuario podrá visualizar todas las estadísticas y los datos recogidos en forma de tablas y gráficos en una aplicación web que está embebida en la aplicación para que el usuario permanezca en ella y también se podrá visualizar en un navegador de cualquier dispositivo para que los usuarios puedan verlo en un tamaño más ampliado. En la opción 'Información de la aplicación' del menú de la aplicación se explica el funcionamiento de la misma y el identificador del usuario único que se le ha asignado (7). Cuando el usuario desee observar todos los datos, deberá pulsar el botón 'Obtener estadísticas' (6) y se produce el inicio de sesión de la aplicación web (8) donde se deberá introducir el identificador del usuario (9) para comprobar que si ese usuario se ha registrado en el sistema, podrá visualizar sus datos. El usuario podrá navegar entre las distintas secciones de la aplicación web y en

cada sección el sistema realizará varias peticiones a la *API REST* en función del tipo de datos que se necesiten (11 y 12) y la API le devolverá los resultados que el navegador va a representar para que el usuario los pueda visualizar (13 y 14).

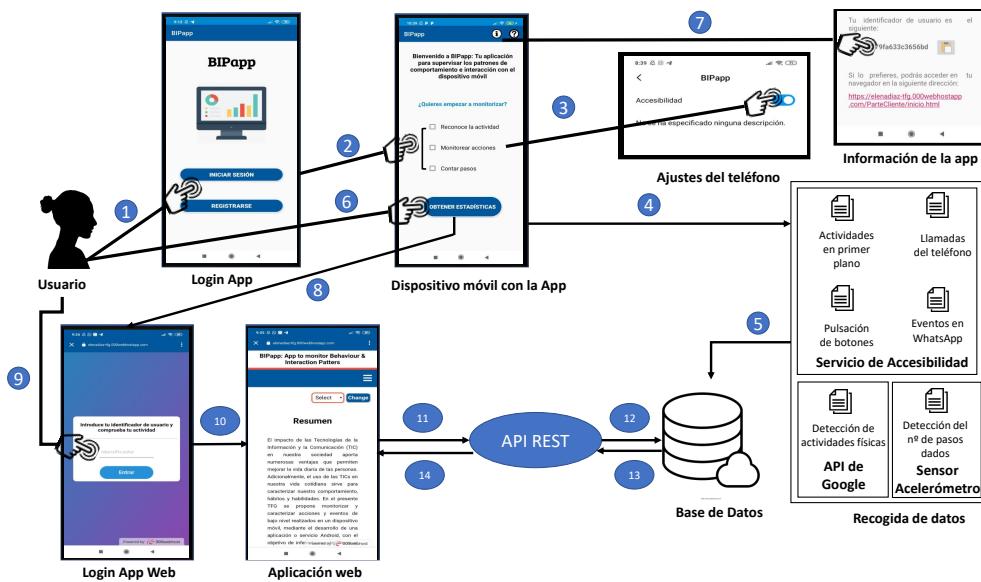


Figura 5.1: Esquema del Trabajo de Fin de Grado

En la Tabla B.1 del Anexo B se muestran los distintos eventos producidos en el sistema y en las aplicaciones más usadas como WhatsApp o en las Redes Sociales, que han permitido obtener una serie de métricas a partir de la monitorización de esos eventos para ser visualizadas.

## 5.2. ITERACIONES

### 5.2.1. Iteración inicial

Un buen proyecto *software* es aquel que presenta una iteración inicial de preparación, donde se mantiene una reunión entre el equipo de desarrollo y el cliente para identificar los requisitos y las necesidades principales que componen el alcance del proyecto. Sucesivamente se aceptará el proyecto y se realizará una planificación para establecer la fecha de inicio y las distintas reuniones donde se entregará el *software* implementado en cada iteración. Esta iteración inicial ha tenido una duración de 8 días y se puede dividir en tres etapas:

- Una primera reunión entre el equipo de desarrollo y el cliente para conocer las necesidades que debía satisfacer el proyecto, donde el usuario presentó los requisitos del sistema.
- Una fase de estudio llevada a cabo por el equipo de desarrollo para estudiar la viabilidad de los requisitos presentados por el cliente, buscando información teórica, alternativas y herramientas necesarias para la realización del proyecto. Se realizaron los primeros prototipos que se presentaron al cliente en la siguiente reunión para acercarse a la solución final y poder obtener el *feedback* del cliente desde esta etapa inicial.
- Una segunda reunión entre el equipo de desarrollo y el cliente, donde el equipo presentó el estudio de viabilidad y de tiempo en función del análisis que han realizado de los requisitos y el cliente priorizó las tareas en función de las estimaciones que se han propuesto.

Al final de cada iteración del proyecto, se mostrará un tablero Kanban con el estado de desarrollo en el que se encuentra cada tarea aplicando la metodología Kanban en el proyecto mediante la herramienta *Monday* mencionada en el Capítulo 2. Se ha decidido no tener más de 5 tareas en las fases *Development*, *Test* y *Deployment* para centrarse en ellas y finalizarlas lo más pronto posible. En la Figura 5.2 se muestra el estado inicial del proyecto, en el que todas las tareas están en la fase To do - Lista de tareas en el tablero Kanban.



**Figura 5.2:** Tablero Kanban estado inicial

### Análisis de requisitos y estudio de la viabilidad del proyecto

La primera fase de la iteración inicial siempre empieza con el análisis de los requisitos presentados por el cliente, que definen las propiedades y las restricciones definidas del proyecto. Esta fase, continúa con una búsqueda de información teórica para hacer un estudio en profundidad sobre las distintas opciones que nos permiten recoger eventos que se producen de las acciones del usuario con el dispositivo móvil, así como las posibles formas para detectar las actividades físicas de los usuarios y el número de pasos dados.

- **Requisitos funcionales.** Los requisitos funcionales son los que describen las funcionalidades que un sistema debe ofrecer cuando un usuario interacciona con él para que se satisfagan sus necesidades y cumplan con los objetivos establecidos. En la Tabla 5.1 se muestran los requisitos funcionales.

**Tabla 5.1:** Requisitos funcionales.

Identificador	Descripción
RF.1	Permitir que los usuarios puedan registrarse
RF.2	Permitir que los usuarios puedan iniciar sesión
RF.3	Permitir que los usuarios puedan cerrar sesión
RF.4	Permitir que los usuarios puedan obtener información sobre el funcionamiento de la aplicación
RF.5	Permitir que los usuarios puedan iniciar y parar el monitoreo de las acciones
RF.6	Visualización del tiempo de uso de las aplicaciones acotados por hoy, semana e histórico
RF.7	Visualización de la frecuencia de pulsación de los botones del dispositivo acotados por hoy, semana e histórico

Identificador	Descripción
RF.8	Visualización del número de llamadas del dispositivo, SMS y notificaciones recibidas acotados por hoy, semana e histórico
RF.9	Visualización del tiempo empleado en cada actividad física y el número de pasos dados acotados por hoy, semana e histórico
RF.10	Visualización del número de aplicaciones instaladas y desinstaladas, letras, símbolos y emoticonos escritos
RF.11	Visualización del número de llamadas y videollamadas y la duración de ambas, número de conversaciones abiertas y de notificaciones recibidas, y el tiempo de visualización de audios, vídeos o imágenes en WhatsApp acotados por hoy, semana e histórico

- **Requisitos no funcionales.** Los requisitos no funcionales son restricciones que se imponen a los requisitos funcionales, es decir, son los que definen las características de la implementación del sistema. En la Tabla 5.2 se muestran los requisitos no funcionales.

Tabla 5.2: Requisitos no funcionales

Identificador	Descripción
RNF.1	La aplicación se tiene que desarrollar para el SO Android
RNF.2	Tiene que ser compatible para distintas versiones Android y distintos dispositivos
RNF.3	La base de datos estará alojada en un servidor remoto y se accederán a los datos mediante una <i>API REST</i>
RNF.4	Las peticiones a la API se harán mediante un atributo encriptado que identifique únicamente a un usuario
RNF.5	El diseño de la aplicación web deberá ser responsive
RNF.6	Tanto la aplicación móvil como la aplicación web deberán ser fáciles de usar e intuitivas
RNF.7	El sistema debe ofrecerse en múltiples idiomas

Antes de comenzar la creación del proyecto y el desarrollo del *software* se ha realizado un estudio inicial para comprobar la viabilidad de este TFG, ya que al tratarse de recoger información sensible del usuario es posible que no existan métodos sencillos que permitan realizar estas acciones. Se realizó una comparación entre distintas alternativas como rootear el teléfono para obtener información, capturar el log de debug de Android o la creación de un servicio de accesibilidad. En este caso, se ha optado desde el primer momento por el servicio de accesibilidad, ya que es la forma más simple de recoger los datos necesarios de los usuarios que permita que usuarios que no posean determinados conocimientos en el ámbito de la informática, puedan usarlos. Sucesivamente, se decidieron algunas herramientas y tecnologías necesarias con las que se va a trabajar. Por ejemplo, como entorno de desarrollo para la aplicación móvil se ha utilizado Android Studio porque proporciona muchas herramientas que son útiles para la elaboración de las interfaces de usuario. Por otro lado, para la visualización de los datos se ha optado por la creación de una aplicación web para aprovechar que los datos pueden ser visualizados tanto en la aplicación móvil como en computadoras o tablets.

Con el análisis que se ha realizado, se han creado los casos de uso que se pueden observar en el Anexo A y se ha realizado una planificación inicial presentada en la Figura 4.3 del Capítulo 4 donde se incluye la estimación de la duración del proyecto y de cada iteración en concreto.

### Estructura del proyecto y sincronización

Antes de empezar el desarrollo del proyecto en Android Studio, hay que establecer la estructura siguiendo el patrón Modelo-Vista-Controlador. No obstante, al ser un proyecto desarrollado con Android, las interfaces de usuario conocidas como *Layouts*, están incluidas desde la creación del proyecto en la carpeta */res*. En esta carpeta se han incluido además de los *Layouts*, las imágenes e iconos utilizados en el menú de navegación, configuraciones adicionales de servicios, estilos de los componentes y los ficheros de *strings* de las cadenas de texto que han permitido la internacionalización de la aplicación. Todas las clases y ficheros escritos en Java que permiten definir la lógica de la aplicación, se encuentran en la primera subcarpeta dentro de la carpeta *java/*. Dentro de esta carpeta se han creado tres subcarpetas que constituyen el patrón MVC:

- **Domain.** Contiene todas las clases necesarias que sirven de soporte a los modelos de la aplicación, donde se definen constantes u otras utilidades.
- **Persistence.** Esta carpeta contiene un único fichero que se encarga de conectar con la base de datos y enviar los datos a la tabla correspondiente en función del tipo de información recogida.
- **Presentation.** Contiene todas las clases Java necesarias para comunicar la lógica de la aplicación con las interfaces de usuario. Algunos de estos ficheros son el servicio de accesibilidad utilizado para la recogida de datos, los servicios e *IntentService* para la detección de la actividad y las *Activities* como *MainActivity* o *Login*.

Con el objetivo de registrar un control de versiones, se ha utilizado Github<sup>1</sup> junto con la herramienta de escritorio Github Desktop que nos permite de una forma fácil y sencilla, sin la necesidad de conocer los comandos git, la sincronización del proyecto en Github.

### Prototipos iniciales

A partir de los requisitos obtenidos en la primera reunión y de los casos de uso realizados a continuación y que se muestran en el Anexo A, se han diseñado con la herramienta Balsamiq Mockups los primeros prototipos del diseño de la app móvil y de la app web. Estos bocetos fueron entregados al cliente en la siguiente reunión con el objetivo de que puedan opinar y modificarlos según sus necesidades. Los prototipos al realizarse en las etapas iniciales pueden sufrir cambios a lo largo del desarrollo dependiendo de las tecnologías y de la implementación desarrollada. En la Figura A.4 del Anexo A, se muestran los bocetos de la app móvil formada por la actividad inicial que permite al usuario registrarse o iniciar sesión y la actividad principal que permite activar las distintas opciones para habilitar la recogida de datos y el ícono del navegador que te permite visualizar la app web. En las Figuras A.5 y A.6 se muestra el diseño de la app web, con las distintas secciones que contendrán los gráficos con los datos recogidos, tanto para el diseño móvil como para el de un ordenador.

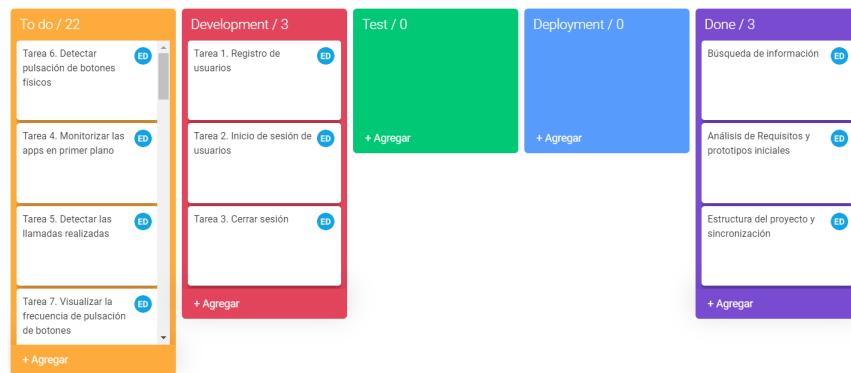
### Obtención de un hosting externo y creación de la base de datos

Con el objetivo de poder publicar la aplicación web en internet para que los usuarios puedan acceder fácilmente, se ha obtenido un alojamiento web de Hostinger que permite alojar todos los archivos en el servidor para que el sitio web funcione correctamente. No es necesario instalar ningún componente

<sup>1</sup>Repositorio del TFG: <https://github.com/elenadiaz-tfg/BIPapp>

adicional y presenta la interfaz *phpMyAdmin* que permite crear la BBDD del sistema y las diferentes tablas que se han utilizado para almacenar distintos tipos de datos de una forma estructurada y relacionados por el identificador del usuario. Las tablas que componen esta BBDD y su significado se irán explicando en cada iteración y el esquema final se puede visualizar en el Anexo E.

Llegando al término de esta iteración inicial, en la Figura 5.3 se muestra el estado del tablero Kanban, donde las tareas correspondientes en esta iteración han sido completadas y las tareas correspondientes a la siguiente iteración empiezan la fase de desarrollo.



**Figura 5.3:** Tablero Kanban iteración 0

### 5.2.2. Iteración 1

En la iteración inicial, el equipo de desarrollo elaboró la planificación con una estimación de duración de cada una de ellas, en función de los requisitos y las tareas definidas por el cliente que fueron creadas y priorizadas según esta planificación. En cada iteración se explicarán los pasos que se han dado hasta alcanzar el objetivo de esa tarea, incluida una etapa de exploración. En esta primera iteración se van a desarrollar las siguientes tareas: registro de usuario, inicio de sesión y cerrar sesión.

#### Tarea 1. Registro de usuario

En la mayoría de los proyectos, la primera tarea consiste en añadir la funcionalidad para que el usuario pueda registrarse en el sistema y que sea identificado para los accesos posteriores únicamente. El primer paso que se realizó es insertar en la interfaz de usuario tres campos de texto que permiten al usuario introducir el nombre, correo electrónico y la contraseña. En esta primera iteración se decidió no modificar el diseño de la interfaz hasta el desarrollo de la actividad principal. La interfaz del usuario desarrollada se muestra en la Figura 5.4(a).

Una vez que se habían decidido los componentes que se iban a guardar en la BBDD, se procedió a crear una tabla denominada *users* formada por tres campos: nombre, correo y contraseña. Los datos almacenados en los campos correo y contraseña se utilizan para comprobar que las credenciales introducidas en el inicio de sesión coincidan con las de esta tabla. Por otro lado, se creó un *script* PHP que permite establecer una conexión con la BBDD e insertar los valores en la tabla *users* si no existe un usuario con los mismos datos. Sucesivamente se creó la lógica de la aplicación que permite realizar el registro del usuario y cuando un usuario complete el formulario se redirigirá a la actividad inicial donde el usuario podrá iniciar sesión.

## Tarea 2. Inicio de sesión

En la segunda tarea, se implementó una interfaz que permite iniciar sesión en la aplicación a los usuarios que se han registrado previamente. Se añadieron dos campos de texto que permiten introducir el correo electrónico y la contraseña del usuario, que se muestran en la interfaz desarrollada en la Figura 5.4(b).

Sucesivamente, se creó otro *script* PHP que permitía establecer una conexión con la base de datos y comprobar en la tabla de usuarios si los datos que se han introducido coinciden con algún registro. Si no existe un usuario registrado con esos datos, el sistema lanza un mensaje de error. En caso contrario, el sistema te redirige a la actividad principal. Por último, se ofrece al usuario la posibilidad de poder mantener la sesión iniciada cada vez que el usuario acceda a la aplicación mediante el servicio *SharedPreferences*<sup>2</sup> que proporciona Android. En primer lugar la aplicación comprueba si existe una sesión abierta y obtiene los datos a través del identificador del usuario que se ha almacenado. En el Listado 5.1 se muestra un ejemplo del código utilizado en el inicio de sesión, donde se comprueba el estado almacenado del *RadioButton*. Si está activado y los datos del usuario son correctos, se guarda la sesión en *SharedPreferences*.

**Listado 5.1:** Mantener la sesión iniciada del usuario

```

1 // Método invocado en el login para almacenar el estado del botón
2 public void estadoRadionBoton() {
3     SharedPreferences preferences = ↵
4         ↵ getSharedPreferences(PREFERENCES, MODE_PRIVATE);
5     preferences.edit().putBoolean("estadoSesion", ↵
6         ↵ sesion.isChecked()).apply();
7 }
8
9 // Método utilizado para comprobar el estado del botón e iniciar la sesión automáticamente
10 public boolean obtenerEstadoBoton() {
11     SharedPreferences preferences = ↵
12         ↵ getSharedPreferences(PREFERENCES, MODE_PRIVATE);
13     return preferences.getBoolean("estadoSesion", false);
14 }
```

## Tarea 3. Cerrar sesión

La última tarea de desarrollo de esta iteración, consiste en ofrecer al usuario la posibilidad de cerrar sesión en la aplicación. Para ello, se ha añadido una opción en el menú de la interfaz de la aplicación que permite cerrar la sesión y que ejecuta el método que se muestra en el Listado 5.2 para eliminar el estado activo del *RadioButton*.

**Listado 5.2:** Cerrar sesión con *SharedPreferences*

```

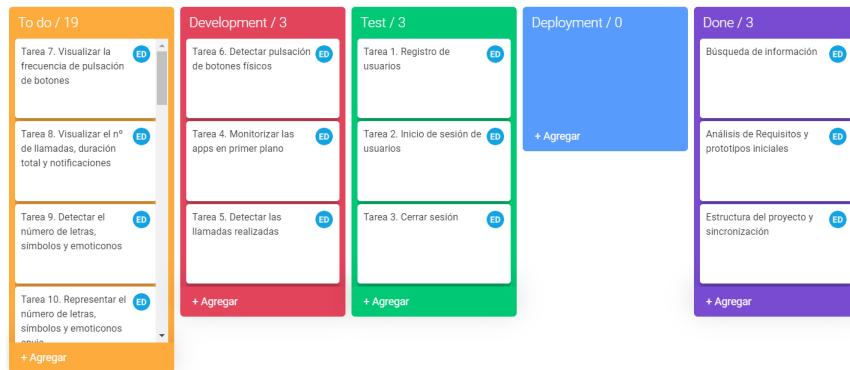
1 public static void cambiarEstadoBoton(Context c, boolean b) {
2     SharedPreferences preferences = ↵
3         ↵ c.getSharedPreferences(PREFERENCES, MODE_PRIVATE);
4     preferences.edit().putBoolean("estadoSesion", b).apply();
```

<sup>2</sup><https://developer.android.com/reference/android/content.SharedPreferences>



**Figura 5.4:** Interfaz del login y registro inicial

Tras la finalización de esta primera iteración, en la Figura 5.5 se muestra el estado del tablero Kanban, donde las tareas correspondientes en esta iteración han sido desarrolladas y van a ser sometidas a *test* en la siguiente iteración, mientras que las tareas correspondientes a la siguiente iteración empiezan la fase de desarrollo.



**Figura 5.5:** Tablero Kanban iteración 1

### 5.2.3. Iteración 2

En esta iteración se desarrollarán las tareas que implican la creación del servicio de accesibilidad. Este servicio es el núcleo fundamental de este proyecto ya que nos permite detectar los eventos más relevantes realizados por el usuario. En esta fase en concreto, se ha detectado la pulsación de los botones físicos, monitorizar las aplicaciones que se ejecutan en primer plano y detectar las llamadas entrantes y realizadas en el dispositivo.

#### Tipos de servicios en Android

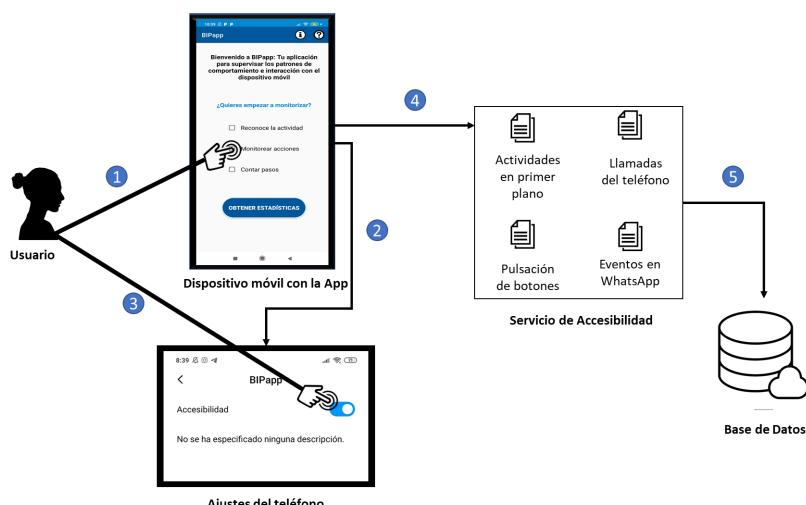
En primer lugar, es necesario diferenciar los dos tipos de servicios proporcionados por Android que se han utilizado en este proyecto. Por un lado, los servicios en segundo plano y por otro lado, los servicios de Accesibilidad. Las funciones de cada servicio son distintas y la única semejanza que

presentan es que se ejecutan en segundo plano en nuestro dispositivo.

- **Servicio en segundo plano** [13]. Casi todas las operaciones que se realizan en una aplicación se ejecutan en primer plano, concretamente en un subprocesso de la interfaz del usuario. El problema de estas acciones se debe a que pueden tener una larga duración y aumentar el tiempo de respuesta de la interfaz de usuario, incluso producir errores en el sistema, por lo que es necesario ejecutar estas operaciones en un hilo secundario. Usando estos servicios se puede controlar su inicio o parada en la misma aplicación a diferencia de los servicios de accesibilidad, donde el usuario tiene que activarlos en los ajustes del sistema.
- **Servicio de accesibilidad** [12]. Muchos usuarios de Android tienen diferentes capacidades o limitaciones que requieren unas facilidades para poder interactuar con sus dispositivos móviles debido a limitaciones visuales, físicas o relacionados con la edad que les impiden ver o usar la pantalla táctil o personas con pérdidas auditivas que no son capaces de reconocer toda la información que se transmite por audio como alertas, notificaciones, multimedia etc. Estos servicios se desarrollan con el objetivo de ayudar a estos usuarios a informarles sobre cambios en la interfaz del usuario, pero en este TFG se propone usar este servicio para recoger eventos realizados por los usuarios. En las siguientes secciones se explicará la configuración de estos servicios y la forma de implementarlos.

### Funcionamiento de los servicios de accesibilidad

Los servicios de accesibilidad se ejecutan en segundo plano y reciben devoluciones de llamadas del sistema cuando se activa un evento de la clase *AccessibilityEvent* [2]. Cada *AccessibilityEvent* es un evento que se produce en nuestro sistema debido a un cambio de estado en la interfaz del usuario, por ejemplo cuando abres una aplicación, recibes una notificación, seleccionas un elemento o haces clic en un botón. En la Figura 5.6 se puede ver el funcionamiento de este servicio en el contexto del TFG. El usuario selecciona la opción *Monitorear acciones* (1) de la aplicación móvil. Esta opción te redirige a la sección de Accesibilidad de los Ajustes del teléfono (2) para que inicies el servicio de accesibilidad de forma manual (3). Sucesivamente, el servicio empieza a recoger los datos de la interacción del usuario con el dispositivo (4) y los guarda en la base de datos (5).



**Figura 5.6:** Funcionamiento del servicio de accesibilidad

## Ciclo de vida de los servicios de accesibilidad

Este tipo de servicios, a diferencia de otros, se activan únicamente cuando el usuario los habilita concretamente en la sección de Accesibilidad de los ajustes del dispositivo. Una vez que el sistema se une a un servicio, es necesario implementar dos métodos:

- ***AccessibilityService#onServiceConnected()***. Este método que se ha implementado es necesario, ya que es ejecutado cuando el sistema se une al servicio.
- ***AccessibilityService#disableSelf()***. Este método sirve para detener el servicio, aunque también se puede detener el servicio cuando el usuario lo apaga en la configuración del dispositivo.

## Declaración del servicio de accesibilidad

Como el resto de servicios, un servicio de accesibilidad se tiene que declarar en el archivo *AndroidManifest.xml* (Figura 5.3), donde además habrá que añadir el permiso *BIND\_ACCESSIBILITY\_SERVICE* que sirve para asegurarse de que solo el sistema pueda vincularse a él y especificar el *INTENT "android.accessibilityservice.AccessibilityService"* que debe ser manejado.

**Listado 5.3:** Declaración del servicio de accesibilidad

```

1 <service
2   android:name=".Presentation.KeyService"
3   android:permission="android.permission.BIND_ACCESSIBILITY_SERVICE">
4     <intent-filter>
5       <action android:name="android.accessibilityservice.□↔
6           ↗ AccessibilityService" />
7     </intent-filter>
</service>
```

## Configuración del servicio de accesibilidad

Los servicios de accesibilidad se pueden configurar para recibir solo unos eventos o paquetes de accesibilidad específicos. La configuración se puede realizar de dos formas distintas:

- Declarando en el archivo *AndroidManifest.xml* un *meta-data* donde se declara el servicio (**C.1**) para que use el recurso *XML* especificado, donde se declaran las propiedades y *flags* de ese servicio para poder monitorear distintos eventos del dispositivo (**C.2**).
- Llamando al método *setServiceInfo(AccessibilityServiceInfo)* de la clase *AccessibilityService*<sup>3</sup> en el método *onServiceConnected()* establecido en el servicio como se muestra en el Listado 5.4.

## Cuerpo del servicio de accesibilidad

Un servicio de accesibilidad está formado por tres partes principales que se corresponden con la conexión, detección de eventos y finalización de la ejecución del servicio. A continuación, se hará una breve explicación de cada una de ellas:

<sup>3</sup><https://developer.android.com/reference/android/accessibilityservice/AccessibilityService>

**Listado 5.4:** Configuración del recurso XML

```

1 public void onServiceConnected() {
2
3     AccessibilityServiceInfo info = new AccessibilityServiceInfo();
4     info.flags = AccessibilityServiceInfo.DEFAULT;
5     info.flags = ↵
6         ↵ AccessibilityServiceInfo.FLAG_REQUEST_TOUCH_EXPLORATION_MODE;
7     info.feedbackType = AccessibilityServiceInfo.FEEDBACK_GENERIC;
8     info.flags = AccessibilityServiceInfo.FLAG_REQUEST_FILTER_KEY_EVENTS;
9     setServiceInfo(info);
10 }
```

- **Conexión del servicio.** El primer método que se ejecuta cuando se activa el servicio es el método *onServiceConnected()*<sup>4</sup>. En este método, como se mostraba en la Figura 5.4, se han declarado los distintos eventos o paquetes de accesibilidad que se quieren recibir.
- **Interrupción del servicio** El método *onInterrupt()*<sup>5</sup> es obligatorio y se invoca cuando el sistema quiere interrumpir el funcionamiento del servicio.
- **Detección de eventos.** El método *onAccessibilityEvent()*<sup>6</sup> recibe un evento de la clase *AccessibilityEvent* y es el núcleo principal del servicio de Accesibilidad donde se detectan las distintas acciones llevadas a cabo por los usuarios, se filtran y se guardan en la base de datos. La clase *AccessibilityEvent* representa los distintos eventos de accesibilidad que envía el sistema al servicio cuando ocurre algún cambio importante en la interfaz del usuario, como por ejemplo cuando un usuario pulsa un botón de la interfaz, hace scroll o cambia de aplicación. El servicio de accesibilidad después de que haya recibido los eventos, puede obtener más información de una vista que se representa en forma de árbol mediante la clase *AccessibilityNodeInfo* [3]. Los privilegios para acceder al contenido de la ventana tendrán que estar declarados con anterioridad con una de las dos formas que se indicaban en la configuración del servicio. Cuando ocurre un evento en el sistema, se puede obtener información general y un conjunto de propiedades que nos ha permitido filtrarla para enviar los datos relevantes a la base de datos.
  - **Tipo de evento.** Mediante el método *getEventType()* se puede obtener el tipo de evento que se ha realizado en el dispositivo donde se pueden diferenciar: *Clicked*, *Long clicked*, *selected*, *focused* etc.
  - **Paquete.** Con el método *getPackageName()* se puede obtener el nombre del paquete origen de la aplicación o del sistema que ha producido un evento en particular.
  - **Clase.** Mediante el método *getClassName()* se obtiene el nombre de la clase de la que proviene el paquete del evento producido.
  - **Texto.** Con el método *getEventText()* que se ha implementado, permite obtener información adicional del evento que ha ocurrido en el sistema.

<sup>4</sup>[https://developer.android.com/reference/android/accessibilityservice/AccessibilityService#onServiceConnected\(\)](https://developer.android.com/reference/android/accessibilityservice/AccessibilityService#onServiceConnected())

<sup>5</sup>[https://developer.android.com/reference/android/accessibilityservice/AccessibilityService#onInterrupt\(\)](https://developer.android.com/reference/android/accessibilityservice/AccessibilityService#onInterrupt())

<sup>6</sup>Método *onAccessibilityEvent*

#### Tarea 4. Monitorizar las aplicaciones ejecutadas en primer plano

Una vez que se han realizado los pasos anteriores para configurar el servicio de accesibilidad, se ha llevado a cabo esta tarea mediante el fragmento de código 5.5 que permite detectar las aplicaciones que se están ejecutando en primer plano. A través del *Logcat* de *Android Studio*, se han podido detectar los eventos producidos de la interacción con los tres dispositivos móviles que se han utilizado para la realización de este proyecto. Sucesivamente, se ha seleccionado la información de aquellos eventos relevantes, detectando que cuando se inicia cualquier aplicación, el evento que se realiza es del tipo *TYPE\_WINDOW\_STATE\_CHANGE*. Teniendo en cuenta ese factor, se ha obtenido el nombre del paquete de la aplicación, la fecha y hora en la que se produjo el evento y el identificador del usuario que provoca ese tipo de evento para almacenarlo en la base de datos.

**Listado 5.5:** Detección de las actividades en primer plano

```

1  if (event.getEventType() == ↵
   ↵ AccessibilityEvent.TYPE_WINDOW_STATE_CHANGED) {
2
3      String Paquete = event.getPackageName().toString();
4
5      // Envío a la Base de Datos
6      String sufijoURL = "?Aplicacion=" + Aplicacion + "&Fecha=" + ↵
   ↵ Fecha + "&Hora=" + Hora + "&Usuario=" + Usuario + ↵
   ↵ "&Paquete=" + Paquete;
7      DatosBBDD.PrimerPlano plano = new DatosBBDD.PrimerPlano(this);
8      plano.execute(sufijoURL);
9 }
```

Una vez que se ha realizado un filtrado de los datos que permiten la detección de una aplicación en primer plano, se creó una tabla en la base de datos denominada *primerPlano* con la siguiente estructura:

primerplano	
App	VARCHAR(1024)
Fecha	VARCHAR(1024)
Hora	TIME
Usuario	VARCHAR(1024)
Paquete	VARCHAR(1024)
Id	INT(200)
Dia	TIMESTAMP
Indexes	

**Figura 5.7:** Estructura tabla primerPlano

En segundo lugar, se creó un *script* en PHP alojado en el servidor que permitía insertar los datos en la tabla creada en la base de datos. Por último, se implementó el método que se muestra en el Listado 5.6 que permitía establecer una conexión HTTP con ese *script* y enviar los datos recogidos en un hilo secundario.

**Listado 5.6:** Envío de información a la base de datos

```

1 public static class PrimerPlano extends AsyncTask<String, Void, String> {
2
3     protected String doInBackground (String... strings) {
4         String registrar_url = "http://m4s.mamilab.eu/BIPapp/primerPlano.php";
5         String s = strings[0];
6         BufferedReader bufferedReader = null;
7
8         try {
9             URL url = new URL(registrar_url + s);
10            HttpURLConnection con = (HttpURLConnection) url.openConnection();
11        } catch (Exception e) {
12            return null;
13        }
14    }
15}

```

**Tarea 5. Detección de pulsaciones de botones físicos**

El método `onKeyEvent`<sup>7</sup> en un servicio de accesibilidad permite reconocer eventos físicos como subir o bajar el volumen del dispositivo y en los móviles más antiguos permitían detectar los tres botones físicos inferiores que dejaban retroceder a una ventana anterior, volver a inicio u observar las actividades recientes. Este método que se ha implementado en el sistema se muestra en el Listado 5.7 y hay que utilizarlo de tal forma que se manejen los dos eventos, ya que si manejas el evento *down* pero no el *up* o viceversa, generaría una secuencia de eventos inconsistente. El parámetro que se recibe es un evento, donde *KeyEvent* es el evento que se tiene que procesar.

**Listado 5.7:** Pulsación de teclas subir y bajar volumen

```

1 public boolean onKeyEvent(KeyEvent event) {
2     int action = event.getAction();
3     int keyCode = event.getKeyCode();
4
5     if (action == KeyEvent.ACTION_UP) {
6         if (keyCode == KeyEvent.KEYCODE_VOLUME_UP) {
7             Log.d("App", "Botón subir volumen");
8         } else if (keyCode == KeyEvent.KEYCODE_VOLUME_DOWN) {
9             Log.d("App", "Botón bajar volumen");
10        }
11        return super.onKeyEvent(event);
12    } else {
13        return super.onKeyEvent(event);
14    }
15}

```

Cuando se detecte un evento de subir o bajar el volumen del dispositivo, se almacenará en la base de datos denominada *botones* que contiene aquellos eventos que suponen un cálculo mínimo para ser representados. La estructura de esta tabla se muestra en la figura 5.8(a).

Para detectar la pulsación del botón de bloqueo y desbloqueo de la pantalla del dispositivo no se

<sup>7</sup>[https://developer.android.com/reference/android/accessibilityservice/AccessibilityService#onKeyEvent\(android.view.KeyEvent\)](https://developer.android.com/reference/android/accessibilityservice/AccessibilityService#onKeyEvent(android.view.KeyEvent))

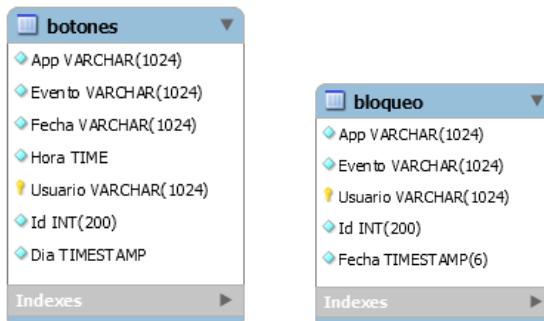
ha podido realizar con el método `onKeyEvent()`, sino con un `BroadcastReceiver`<sup>8</sup> que es una clase que recibe y maneja *Intents* de difusión. Los *Intents* que se reciben son del tipo `ACTION_SCREEN_ON` y `ACTION_SCREEN_OFF`. Cada vez que el dispositivo cambia al estado en suspensión o activación se recibe un evento u otro como se muestra en el Listado 5.8. Estos eventos se envían a las tablas *primerPlano* y *bloqueo*, debido a que es necesario calcular el tiempo de uso de cada aplicación cuando el dispositivo esté desbloqueado y calcular el tiempo que el dispositivo está bloqueado o desbloqueado. Se ha diseñado la tabla *bloqueo* cuya estructura se muestra en la Figura 5.8(b) que ha permitido mediante el tipo de evento y la fecha y hora registradas, calcular los intervalos de tiempo que el dispositivo se encuentra bloqueado o desbloqueado.

**Listado 5.8:** Detección bloqueo y desbloqueo pantalla

```

1 // Registro del broadcast en el método onCreate() de la aplicación
2 registerReceiver(miBroadcast, new IntentFilter(Intent.ACTION_SCREEN_ON));
3 registerReceiver(miBroadcast, new IntentFilter(Intent.ACTION_SCREEN_OFF));
4
5 // Método que permite la detección del bloqueo y desbloqueo de la pantalla
6 BroadcastReceiver miBroadcast = new BroadcastReceiver() {
7     public void onReceive(Context context, Intent intent) {
8         if (intent.getAction().equals(Intent.ACTION_SCREEN_ON)) {
9             Log.i("TAG", "Screen ON");
10        } else if (intent.getAction().equals(Intent.ACTION_SCREEN_OFF)) {
11            Log.i("TAG", "Screen OFF");
12        }
13    }
14};

```



(a) Estructura tabla botones    (b) Estructura tabla bloqueo

**Figura 5.8:** Estructura de las tablas botones y bloqueo

## Tarea 6. Detectar las llamadas realizadas

Para desarrollar esta tarea, se probaron distintas alternativas. En primer lugar se podía detectar cuando se iniciaba una llamada y cuando se producía el evento de finalizar mediante eventos del tipo `TYPE_VIEW_CLICKED` y `TYPE_NOTIFICATION_STATE_CHANGED`, y el paquete '`com.google.android.dialer`', pero haciendo estudios de compatibilidad entre distintos dispositivos este paquete no era común en ambos. Explorando la tabla de detección de las aplicaciones en primer plano, se produce un evento

<sup>8</sup><https://developer.android.com/reference/android/content/BroadcastReceiver?hl=es-419>

cuando se realiza una llamada y a partir de ese evento de la tabla *primerPlano* se han realizado los cálculos necesarios para calcular el número de llamadas y el tiempo total empleado.

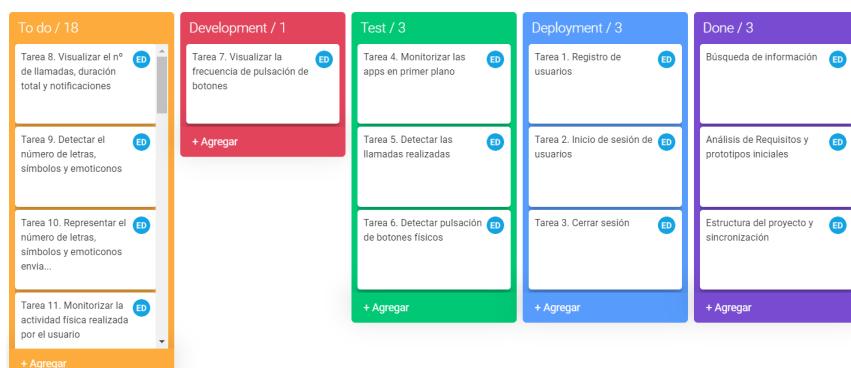
### Resolución de problemas de la iteración anterior

Un servicio de accesibilidad no está vinculado a una interfaz de usuario y podrían funcionar independientemente, es decir, podrías descargar un servicio de accesibilidad desde *Play Store* y poder iniciarla desde la sección de Accesibilidad del dispositivo. Se llegó a la conclusión de que era innecesario construir un *login* y un registro con las credenciales que el usuario deseara para registrarse como se desarrolló en la primera iteración, porque el usuario podría iniciar el servicio de accesibilidad sin haberse registrado en la aplicación. Por este motivo, se necesita un identificador único para cada usuario que provoque cada evento que se guarda en la base de datos y poder mostrar la información en función del mismo. Además, debido a la delicadez de los datos recogidos, se pueden mantener principios de privacidad al asociar los datos simplemente a un identificador. Se estudiaron varias alternativas como la obtención de la dirección *MAC* del dispositivo, el código *IMEI*, ID de publicidad o de instancia, pero estos códigos presentaban varios inconvenientes ya que actualmente no se puede obtener la dirección *MAC* y el código *IMEI* requiere permisos del usuario y es un código privado. Finalmente, se eligió el identificador *ANDROID\_ID*<sup>9</sup> que es un número de 64 bits expresado como una cadena hexadecimal y que es único para cada aplicación, usuario y dispositivo móvil. Este identificador, que se encuentra cifrado en la BBDD, permite cumplir la Ley Orgánica de Protección de Datos Personales y garantía de los derechos digitales (LOPDP) garantizando el honor y la intimidad personal de los ciudadanos según el Art. 18.4 de la Constitución Española.

**Listado 5.9:** Obtención del identificador del usuario

```
1 String uuid = Settings.Secure.getString(getApplicationContext() . ↵
    ↵ getContentResolver() , Settings.Secure.ANDROID_ID);
```

Una vez finalizada esta iteración y resueltos los problemas del inicio de sesión y registro de usuario, en la Figura 5.9 se muestra el estado del tablero Kanban, donde las tareas correspondientes en esta iteración han sido desarrolladas y van a ser sometidas a *test* en la siguiente iteración, mientras que una de las tareas correspondientes a la siguiente iteración empieza la fase de desarrollo. Por otro lado, las tareas correspondientes a la iteración anterior están preparadas para ser desplegadas.



**Figura 5.9:** Tablero Kanban iteración 2

<sup>9</sup>[https://developer.android.com/reference/android/provider/Settings.Secure#ANDROID\\_ID](https://developer.android.com/reference/android/provider/Settings.Secure#ANDROID_ID)

### 5.2.4. Iteración 3

Llegados a esta punto y siguiendo la planificación realizada al inicio del proyecto, esta iteración tendrá una duración superior al resto. Esto es debido a que el proceso más importante junto con la creación del Servicio de Accesibilidad, es la construcción de la aplicación web que conlleva la integración de la aplicación web en la aplicación móvil y la creación de la *API REST* que manipula y sirve la información al sistema para poder representarla en forma de gráficos y tablas. Para realizar estas subtareas, es necesario realizar un estudio sobre un gran abanico de opciones, funcionalidades y diseño.

En esta iteración se llevarán a cabo las tareas 7 y 8 que se corresponden con la visualización de la frecuencia de pulsación de los botones físicos del dispositivo como subir o bajar volumen y el número de llamadas, duración total y notificaciones recibidas en el sistema acotadas por tiempo.

#### Construcción de la *API REST*

*REST* es una arquitectura *software* que sirve para crear una interfaz común entre distintos sistemas que se comunican mediante HTTP con la ventaja de que no guarda los datos que se producen entre dos llamadas. El cliente de una *API REST* puede ser una aplicación *Android* o *iOS* o un navegador web en el caso de este proyecto. En el ámbito web, una API es un servicio de *back-end* utilizado para conectar dos aplicaciones. Esta arquitectura presenta una serie de características [7] que la definen, como pueden ser:

- **Protocolo cliente/servidor sin estado.** Cada petición HTTP que recibe el servidor es independiente y contiene la información necesaria para ejecutarla para qué, tanto el cliente como el servidor no tengan que recordar ningún estado previo para llevarla a cabo.
- Las **operaciones** que se pueden utilizar son *POST*, *GET*, *PUT* y *DELETE* que sirven para la creación, el acceso, actualización o borrado de recursos.
- **Cada objeto se manipula a través de una *URI*** por lo que se convierte en el identificador único de cada recurso de este sistema.
- **Las llamadas se almacenan en cachés** para evitar solicitar varias veces el mismo recurso.
- Ofrece una **arquitectura jerárquica** entre los componentes y cada una lleva a cabo una funcionalidad dentro del sistema *REST*.

Existen distintos lenguajes para la creación de *API REST* que son normalmente usados en el *back-end* como PHP, NodeJS o .NET. Centrándonos en el lenguaje de programación PHP, existen *frameworks* muy potentes como son Symfony<sup>10</sup> o Laravel<sup>11</sup> que permiten acelerar el desarrollo de una API, pero existen otros *microframeworks*, como es el caso de Slim o Lumen<sup>12</sup>, que tienen como objetivo principal el desarrollo de una *API REST* de una forma fácil y sencilla. En este proyecto se ha hecho uso del *microframework* Slim [27] para crear una *API REST* que permita acceder a los datos almacenados en la base de datos, procesarlos y servirlos al *front-end* para que puedan ser manipulados y representados gráficamente. Slim permite crear rápidamente aplicaciones web y APIs simples pero muy potentes. Su misión consiste en recibir solicitudes HTTP, invocar a una rutina de devolución de llamada concreta

---

<sup>10</sup><https://symfony.es/>

<sup>11</sup><https://laravel.com/>

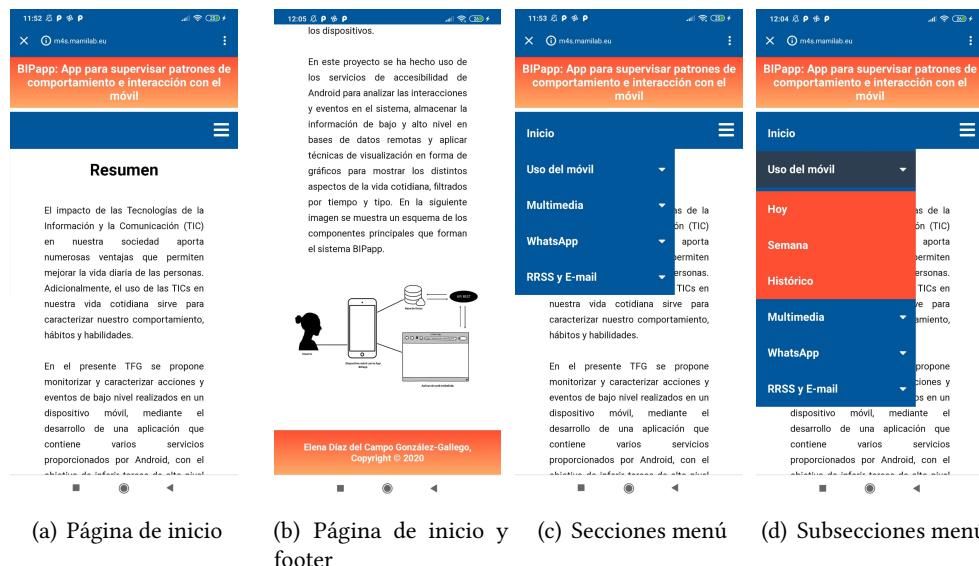
<sup>12</sup><https://lumen.laravel.com/>

como pueden ser sentencias SQL para acceder a los datos de la base de datos, y devolver esa respuesta HTTP en un formato especificado como JSON.

En primer lugar, se necesita de un servidor web en local o en remoto que previamente configurado, envíe todas las solicitudes a un archivo PHP que funciona como un controlador frontal, después lo instancia y ejecuta la aplicación Slim en ese archivo. La configuración de este archivo se puede ver en el Listado C.3 del Anexo C. Después de disponer de un servidor web, es necesario instalar *Composer* que es un manejador de paquetes para PHP que permite descargar e instalar dependencias y librerías de forma automática. La API contiene una serie de *endpoints* que primero invocan una devolución de llamada cuando reciben solicitudes HTTP concretas y después devuelven una respuesta HTTP en un formato determinado. En el Listado C.5 del anexo mencionado anteriormente, se puede ver una ruta de ejemplo que devuelve el tiempo de uso de las 10 aplicaciones más usadas.

### Creación del menú de la aplicación

Para poder representar la información, fue necesario crear un menú de navegación inicial del *front-end* que se dividiera en distintas secciones dependiendo de las categorías de los datos recogidos. En la página de inicio se muestra el contexto en el que se enmarca el desarrollo del sistema (5.10(a), 5.10(b)) y cada una de las opciones que se presentan en el menú se dividen en subsecciones que representan los datos recogidos de esa categoría en el día actual, en la última semana y en el histórico (5.10(c) y 5.10(d)).



**Figura 5.10:** Interfaz del login y registro inicial

### Integración de la aplicación web

Una vez que se había creado el menú de la aplicación y las distintas secciones que permitirán la representación de la información, era necesario integrar la aplicación web en la aplicación móvil *Android* y surgieron varias alternativas. Todas ellas fueron estudiadas y probadas para ver las ventajas y desventajas que proporcionaban, pero finalmente se seleccionó la última opción que permitía realizar las mismas funcionalidades que un navegador. Estas tres alternativas eran:

- **Navegador web.** Esta opción se basaba en enlazar al usuario a un navegador de su dispositivo móvil para que pudiera visualizar la aplicación web. No obstante, fue descartada desde el principio porque el principal objetivo es facilitar al usuario la visualización de esos datos manteniéndolo dentro de la aplicación.
- **WebView [39].** Un *widget* *WebView* es la mejor opción para mostrar contenido web diseñado por el creador de la aplicación y para tener un mayor control sobre la interfaz del usuario, pero carecen de algunas funcionalidades y necesita de configuraciones especiales para permitir algunas acciones. Aspectos como el uso de librerías JavaScript o el almacenamiento en el *LocalStorage* se tienen que configurar previamente y otra particularidad como descargar un informe en PDF que no contenga una URL previamente definida no es posible mediante este componente, por eso se ha utilizado la última opción.
- **Chrome Custom Tabs [11].** *Chrome Custom Tabs* ha sido finalmente la opción elegida para mostrar la aplicación web al usuario debido a que se implementa de una forma fácil como se puede ver en el Listado 5.10 y ofrece numerosas ventajas como:
  - Permite personalizar la interfaz del usuario.
  - Ofrece una navegación segura de Google que protege al usuario en todo momento.
  - Permite compartir *cookies* y el modo autocompletar que permite al usuario no tener que volver a introducir el identificador en accesos posteriores.
  - Con un solo toque se puede volver a la aplicación.

**Listado 5.10:** Configuración Chrome Custom Tabs

```

1 protected void onCreate(Bundle savedInstanceState) {
2
3     botonWeb = findViewById(R.id.btnWeb);
4     botonWeb.setOnClickListener (new View.OnClickListener() {
5         public void onClick(View v) {
6             openChromeCustomTabs("http://m4s.mamilab.eu/BIPapp/←
7                 ↪ ParteCliente/inicio.html");
8         }
9     });
10
11 public void openChromeCustomTabs(String url) {
12     CustomTabsIntent.Builder builder = new CustomTabsIntent.Builder();
13     CustomTabsIntent customTabsIntent = builder.build();
14     customTabsIntent.launchUrl(this, Uri.parse(url));
15 }
```

## Representación de los datos en función del usuario

Después de la integración de la aplicación web, se estudiaron distintas formas para realizar peticiones a la base de datos en función del usuario y representar esos datos en función del mismo. Surgió un problema debido a que las aplicaciones web embebidas son menos flexibles y el identificador del usuario no se le podría pasar directamente a las peticiones generadas por la *API REST*. La solución que se adoptó fue hacer un *login* en el *front-end* donde el usuario tendrá que introducir el identificador

de usuario que se le ha asignado, sucesivamente el sistema comprobará en la tabla *users* de la base de datos si ese usuario ya está registrado y en caso afirmativo le dejará visualizar sus datos. Mediante este *login*, el identificador del usuario se almacenará encriptado en el *LocalStorage* del navegador y el *front-end* realizará consultas a la *API REST* dependiendo de ese identificador.

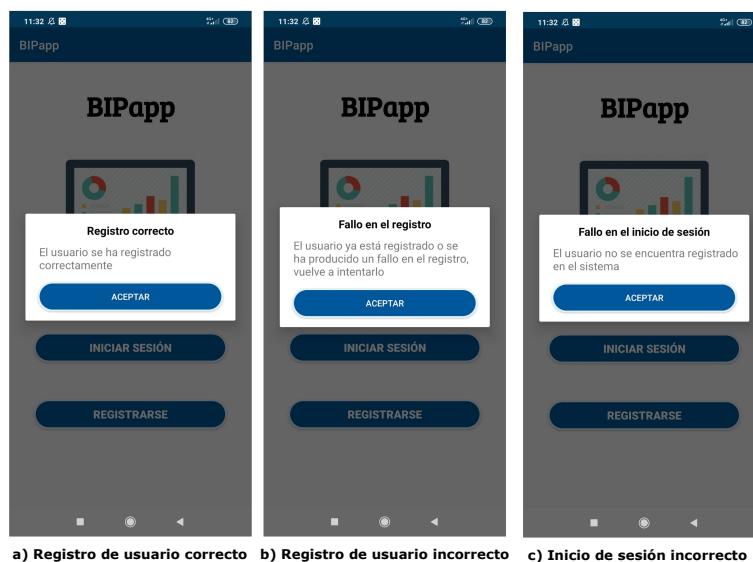
El primer paso fue modificar el inicio de sesión y registro inicial que se construyó inicialmente. Para ello, se eliminaron los campos de texto de introducir datos y se modificó la tabla de registros de usuarios denominada *users* con un único campo que contiene el identificador del usuario cifrado mediante el algoritmo *SHA-512* como se puede ver en la Figura 5.11.



**Figura 5.11:** Ejemplo de usuario registrado en la base de datos

El diseño final de la interfaz del registro e inicio de sesión de usuarios estará formado por dos elementos de tipo *Button* denominados:

- **REGISTRARSE.** Cuando el usuario inicia la aplicación por primera vez debe registrarse en el sistema, de tal forma que cuando haga *click* en este botón, el sistema obtiene el identificador del usuario y comprueba si existe en la tabla de la base de datos. Si el identificador no coincide con un registro de la base de datos, el sistema muestra un mensaje al usuario de que el registro se ha realizado correctamente (Figura 5.12, Escenario a). Sin embargo, si el identificador coincide con un registro de la base de datos, el sistema muestra un mensaje al usuario de error como se muestra en el escenario b de la Figura 5.12.
- **INICIAR SESIÓN.** Cuando el usuario se haya registrado en el sistema, podrá iniciar sesión en la aplicación para empezar a monitorizar todos los eventos. El sistema comprueba que el identificador del usuario se encuentre almacenado en la tabla *users* de la base de datos y si no se encuentra, avisa al usuario para que se registre antes de iniciar sesión (Figura 5.12, Escenario c).



**Figura 5.12:** Inicio de sesión y registro definitivo de la app móvil

Sucesivamente se creó el *login* de la aplicación web que contiene un único campo de texto donde se debe introducir el identificador del usuario. Cuando el sistema comprueba si el usuario existe en la base de datos, ejecuta la función **5.11** JavaScript utilizada para guardar el identificador del usuario cifrado en el *LocalStorage* del navegador mediante la librería *CryptoJS* y con el mismo algoritmo utilizado para cifrar el identificador del usuario que se guarda en la base de datos.

**Listado 5.11:** Inicio de sesión aplicación web

```

1 function setNombre() {
2     var hash = CryptoJS.SHA512(document.getElementById('nombre').value);
3     localStorage.setItem("nombre", hash.toString(CryptoJS.enc.Hex));
4 }
```

Finalmente se mostrarán esos datos en función del usuario obteniendo el identificador del usuario y enviándolo en la URL de la petición de la *API REST* (Listado 5.12).

**Listado 5.12:** Petición *API REST* en función del usuario

```

1 var miDatos = localStorage.getItem("nombre");
2 $(document).ready(function() {
3     $.ajax({
4         url: "http://m4s.mamilab.eu/BIPapp/apiRest/public/api/←
5             ↪ duracionLlamadasD/" + miDatos,
6     })
})
```

## Tarea 7. Visualizar la frecuencia de pulsación de botones físicos

Antes de visualizar la frecuencia de pulsación de los botones físicos, es necesario realizar un estudio de diferentes librerías JavaScript, tanto libres como comerciales, disponibles para la creación de los gráficos de la aplicación. Las bibliotecas o librerías de JavaScript son archivos que nos proporcionan numerosas funciones para realizar diferentes tipos de tareas, agregarles funcionalidades y efectos en las páginas web. En concreto, en este proyecto nos permiten crear gráficas y visualizarlas para mostrar estadísticas sobre los datos recogidos. A continuación se muestra una tabla comparativa (Figura 5.3) de las librerías libres más populares extraídas de las características principales analizadas y presentadas en el Anexo D para representar gráficos, y debido a la simplicidad, facilidad de uso y la documentación extensa de la librería *Chart JS* [16], se decidió usar esta librería para crear los gráficos de la aplicación web.

Para representar un gráfico es necesario consumir la *API REST* para obtener los datos de la base de datos. Existen varias alternativas para comunicarse con una *API REST* desde JavaScript. En este proyecto se ha hecho un estudio sobre *XMLHttpRequest* [31], *Fetch API* [37] y *AJAX* [30].

- **XMLHttpRequest.** Esta opción es un objeto JavaScript que permite obtener información de una URL de una forma fácil y que proviene generalmente de bases de datos remotas. Estas llamadas pueden devolver datos tanto en formato XML<sup>13</sup>, como en formato JSON<sup>14</sup>.

<sup>13</sup><https://developer.mozilla.org/es/docs/Web/XML>

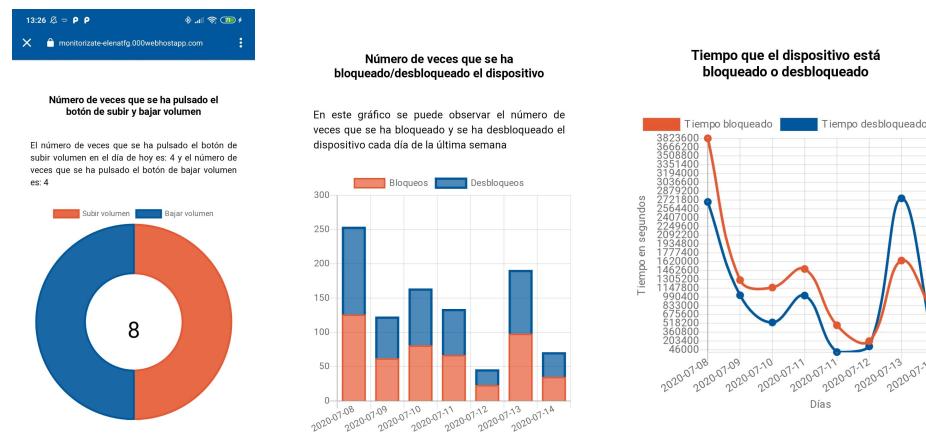
<sup>14</sup><https://developer.mozilla.org/es/docs/Learn/JavaScript/Objects/JSON>

Tabla 5.3: Tabla comparativa librerías gráficos JavaScript

Característica \ Librería	Chart JS	D3 JS	Google Charts	Highcharts	Chartist JS
<b>Open Source</b>	Sí	Sí	No	No	Sí
<b>Cantidad de gráficos</b>	8	Más de 8	Más de 8	Más de 8	3
<b>Documentación y ejemplos</b>	Sí	Sí	Sí	Sí	Sí
<b>Compatibilidad navegadores</b>	Sí	Sí	Sí	Sí	Sí
<b>Uso de plugins</b>	Sí	Sí	Sí	Sí	Sí
<b>Facilidad de uso</b>	Sí	No	Sí	Sí	Sí
<b>Requiere conexión a red</b>	No	No	Sí	No	No

- **Fetch API.** Esta API ofrece una interfaz para obtener recursos de una forma más potente y flexible que *XMLHttpRequest*. Permite interactuar con APIs y obtener datos que se puedan representar en la aplicación web.
- **AJAX.** Es un conjunto de tecnologías que permiten realizar llamadas a los servidores desde las páginas de un navegador web. Es posible utilizarlo tanto en JavaScript como con jQuery.

Inicialmente se implementó la primera opción pero los datos no se actualizaban correctamente y se decidió desarrollar la segunda alternativa, pero produjo problemas para manipular los datos y representarlos en los gráficos con la librería seleccionada. Finalmente, con peticiones *AJAX jQuery*, que son más rápidas y simples que las peticiones *AJAX JavaScript*, se realizaron peticiones al recurso de la *API REST* que permitían consumir los recursos creados para obtener la frecuencia de pulsación de los botones físicos de subir y bajar volumen. Llegados a este punto, se crearon tres *endpoints* que permitían obtener el número de veces que se ha pulsado el botón de subir y bajar volumen en el día y semana actual, y en el histórico, junto con otros tres *endpoints* que permitían obtener el número de veces que se ha bloqueado o desbloqueado el teléfono y el tiempo que el dispositivo permanece bloqueado o desbloqueado acotados por el día y la semana actual, y el histórico. Estos *endpoints* fueron consumidos con peticiones *AJAX* y representados con la librería *Chart.js* obteniendo el resultado que se muestra en la Figura 5.13.



(a) Subir y bajar volumen día actual

(b) Bloqueos y desbloqueos semana actual

(c) Tiempo bloqueado y desbloqueado histórico

Figura 5.13: Pulsación de botones físicos

### Tarea 8. Visualizar el número de llamadas, duración total y notificaciones

Para visualizar el número de llamadas, la duración total y el número de notificaciones totales acotadas por el día y la semana actual, y el histórico, se llevó a cabo el mismo proceso mencionado anteriormente. La única diferencia significativa es la representación de estos datos tanto en gráficas como en tablas como se puede apreciar en la Figura 5.14.

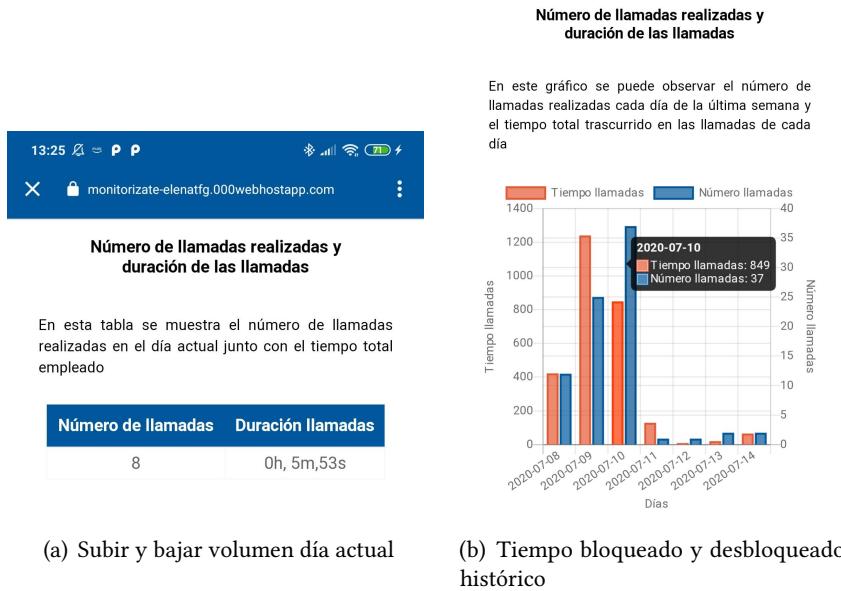


Figura 5.14: Número de llamadas y duración llamadas

Al finalizar esta iteración y haber desarrollado las tareas programadas, en la Figura 5.15 se muestra el estado del tablero Kanban, donde las tareas correspondientes en esta iteración han sido desarrolladas y van a ser sometidas a *test* y despliegue en la siguiente iteración, mientras que una de las tareas correspondientes a la siguiente iteración empieza la fase de desarrollo. Por otro lado, las tareas correspondientes a la iteración anterior se han terminado.

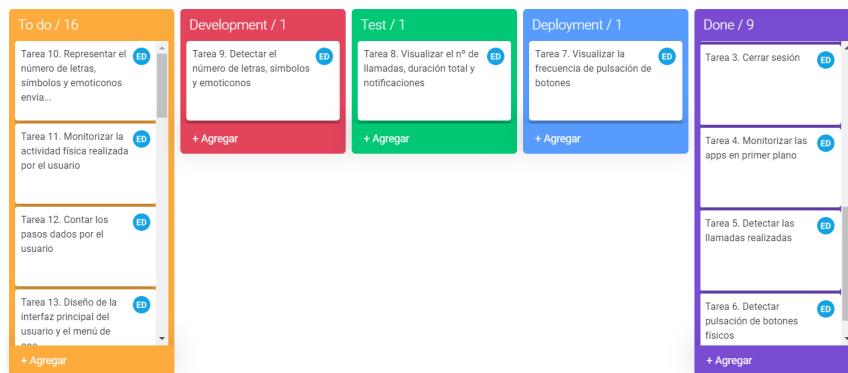


Figura 5.15: Tablero Kanban iteración 3

#### 5.2.5. Iteración 4

Esta iteración tiene una duración de una semana y en ella se desarrollarán las tareas 9 y 10 correspondientes a la monitorización y visualización del número de elementos escritos con el dispositivo, diferenciando letras, símbolos u emoticonos.

### Tarea 9. Detectar el número de letras, símbolos y emoticonos

Un servicio de accesibilidad puede detectar las letras, símbolos u emoticonos que un usuario escribe. No obstante, para garantizar la máxima confidencialidad de los datos recogidos, se ha configurado el sistema de tal forma que cuando ocurre un evento de este tipo, se registre en la base de datos en forma de letra, símbolo u emoticono y nunca guardando el contenido original escrito por el usuario. Este contenido es posible detectarlo a través del nombre de la clase '*android.widget.EditText*' y para diferenciar los tipos de elementos escritos ha sido necesario filtrarlos con condiciones *if-else*. Este contenido se almacenará en la tabla *apps* como se muestra en la Figura 5.16, que contiene eventos que implican realizar un conteo básico sobre el número de veces que ocurren, y en la tabla *primerPlanoLetras*, cuya estructura y objetivo se explicará en la Tarea 5.2.10.

App	Evento	Fecha	Hora	Usuario
Whatsapp	Letra	2020-06-22	10:34:23.000000	30d7a6083a561e8dbc2988dc322e
Whatsapp	Letra	2020-06-22	10:34:23.000000	30d7a6083a561e8dbc2988dc322e
Whatsapp	Letra	2020-06-22	10:34:33.000000	30d7a6083a561e8dbc2988dc322e

Figura 5.16: Ejemplo de registro de letras escritas

### Tarea 10. Representar el número de letras, símbolos y emoticonos enviados

Una vez que se han monitorizado los eventos de enviar letras, símbolos y emoticonos, fue necesario realizar un análisis de la información relevante que se debe representar. En la Tabla 5.4 se muestran las métricas que han sido procesadas en la API a partir de los eventos monitorizados en la tarea 9.

Tabla 5.4: Métricas de letras, símbolos y emoticonos representadas

Evento	Métrica
Letras enviadas	Número de letras enviadas, frecuencia de pulsación de letras por segundo y tiempo total empleado escribiendo, en el día y semana actual y en el histórico.
Símbolos enviados	Número de símbolos enviados, frecuencia de pulsación de símbolos por segundo y tiempo total empleado escribiendo, en el día y semana actual y en el histórico.
Emoticonos enviados	Número de emoticonos enviados, frecuencia de pulsación de emoticonos por segundo y tiempo total empleado escribiendo, en el día y semana actual y en el histórico.

En la Figura 5.17 se muestra una gráfica que compara el número de cada tipo de elemento enviado cada día de la última semana y una tabla que contiene el tiempo total que se ha empleado escribiendo y la frecuencia del número de elementos escritos en cada segundo.

Una vez que se han desarrollado las tareas programadas en esta iteración las tareas serán sometidas a *test* para desplegarlas sucesivamente en la siguiente iteración, mientras que las tareas de la siguiente iteración empiezan la fase de desarrollo como se muestra en la Figura 5.18.

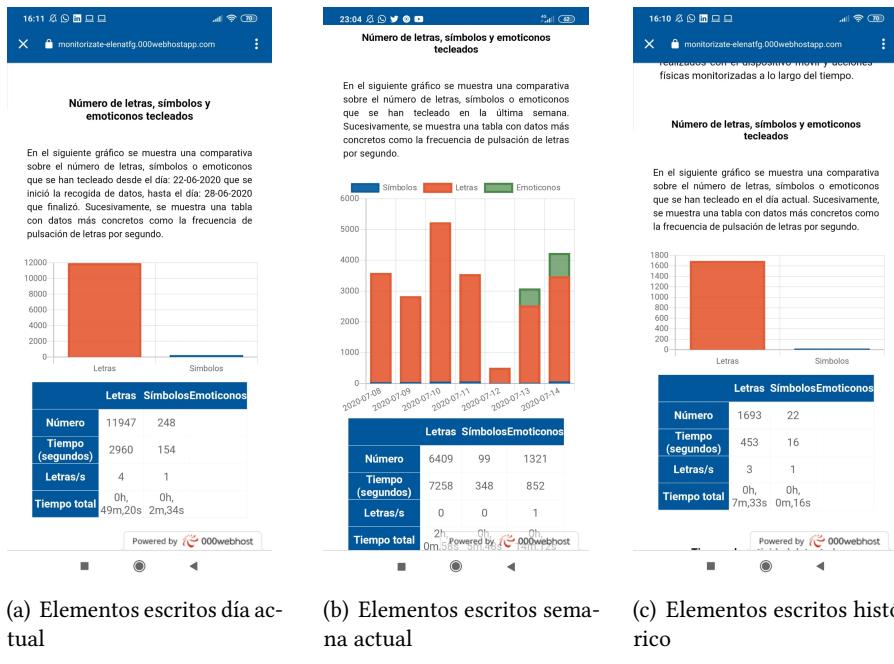


Figura 5.17: Gráficos letras, símbolos y emoticonos enviados

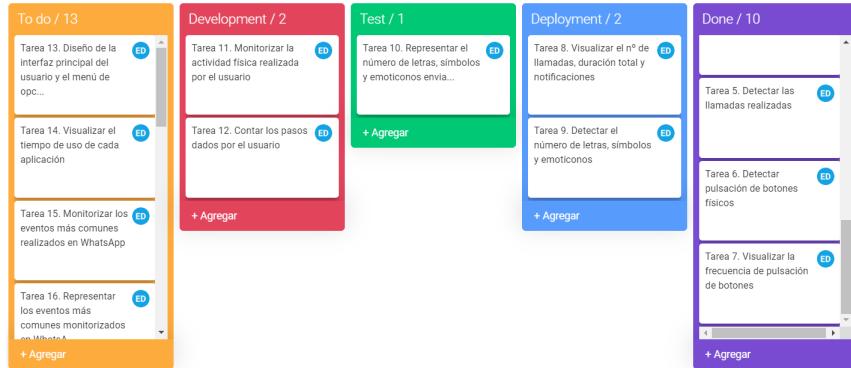


Figura 5.18: Tablero Kanban iteración 4

## 5.2.6. Iteración 5

En esta iteración se llevará a cabo la implementación del detector de actividades físicas y el contador de pasos, que permitirán relacionar los eventos realizados en el dispositivo con el tipo de actividad que realiza el usuario en un momento determinado. Por último, se realizará el diseño de la interfaz de la actividad principal de la aplicación y el menú de navegación.

### Tarea 11. Monitorizar la actividad física realizada por el usuario

Un servicio es un componente *software* que se ejecuta en segundo plano y que realiza un conjunto de operaciones cada cierto intervalo de tiempo. Los servicios no se les proporcionan a los usuarios en forma de interfaces gráficas, sino que se utilizan para ejecutar otras funcionalidades como puede ser la reproducción de música desde un dispositivo. No obstante, estos servicios pueden ser iniciados en las actividades y pueden funcionar de dos formas distintas [19]:

- **Autónomo.** Se corresponde cuando una actividad de una aplicación o un componente de esta actividad inicia el servicio mediante el método *StartService()*<sup>15</sup>. Una vez que se ha iniciado este servicio se puede ejecutar indefinidamente a lo largo del tiempo, incluso si el componente que lo inició se destruye.
- **Dependiente o ligado (*Bind Service*).** Se corresponde cuando un elemento de la aplicación se une al servicio mediante el método *onBind()*<sup>16</sup>. Los servicios ligados permiten a los componentes de una aplicación interactuar con el servicio enviando peticiones y recibiendo resultados. Estos servicios solo se ejecutan mientras que los componentes de la aplicación están unidos a él y cuando todos estos elementos se desligan, se destruye el servicio.

Para el reconocimiento de la actividad del usuario se ha seleccionado un *IntentService*, que es una clase que permite crear servicios autónomos y realizar tareas en segundo plano ejecutándose en un hilo secundario. La principal diferencia respecto a la clase *Service* es que la instancia del *IntentService* termina automáticamente una vez finaliza la ejecución de su tarea. La clase *IntentService* dispone del método *onHandleIntent()*<sup>17</sup> que es invocado de forma asíncrona por el sistema y es donde se tiene que implementar la funcionalidad de este tipo de servicio. Actualmente, existen varias APIs desarrolladas por Google que podemos usar libremente para desempeñar distintas acciones y que permiten la comunicación e integración de los Servicios de Google con otros servicios. En este proyecto se han usado las siguientes APIs:

- **Activity Recognition Client [4].** La API *Activity Recognition Client*, proveniente de Google y sucesora de las APIs *LocationClient* y *ActivityRecognitionAPI*, es el primer punto de entrada para detectar el reconocimiento de la actividad de los usuarios en tiempo real. Esta API ofrece a su vez dos APIs, la API de transición de reconocimiento de actividad y la API de muestreo de reconocimiento de actividad. Para el reconocimiento de la actividad se ha usado la segunda API ya que mejora la precisión, consume menos energía y aumenta la productividad.
- **Activity Recognition Result [5].** Por otro lado, se ha hecho uso de la API *Activity Recognition Result*, que también proviene de Google y se utiliza para obtener las actividades más probables que se han detectado con la API mencionada anteriormente. Ese conjunto de actividades más probables, se almacenan en un *ArrayList* que contiene un conjunto de actividades detectadas concretas, filtradas mediante la clase *DetectedActivity*[14] y a cada actividad detectada se le asocia un grado de confianza que indica la probabilidad de esa actividad.

La lógica del reconocimiento de la actividad consiste en extraer el resultado de un reconocimiento de actividad mediante un *Intent* producido por la API *ActivityRecognitionClient* y que recibe la clase *ActivityRecognitionResult*, para después almacenar en un *array* las distintas actividades más probables realizadas por el usuario que se hayan extraído anteriormente. Para cada tipo de actividad detectada se obtiene el nivel de confianza y el tipo de actividad con los métodos *getConfidence()* y *getType()* que ofrece la clase *DetectedActivity*. Estos datos se almacenan en una tabla denominada *detectaActividades* cuya estructura se muestra en la Figura 5.19 y que permitirán calcular el tiempo realizando cada tipo de actividad física.

<sup>15</sup>[https://developer.android.com/reference/android/content/Context?hl=es-419#startService\(android.content.Intent\)](https://developer.android.com/reference/android/content/Context?hl=es-419#startService(android.content.Intent))

<sup>16</sup>[https://developer.android.com/reference/android/app/Service?hl=es-419#onBind\(android.content.Intent\)](https://developer.android.com/reference/android/app/Service?hl=es-419#onBind(android.content.Intent))

<sup>17</sup>[https://developer.android.com/reference/android/app/IntentService#onHandleIntent\(android.content.Intent\)](https://developer.android.com/reference/android/app/IntentService#onHandleIntent(android.content.Intent))



**Figura 5.19:** Estructura tabla detectaActividades

### Tarea 12. Contar los pasos dados por el usuario

Cuando las actividades realizadas por el usuario fueron detectadas, se empezó a desarrollar esta tarea, cuyo objetivo es potencialmente relacionar las acciones y eventos monitorizados con la cadencia de paso. En primer lugar, se realizó un estudio sobre distintas formas para obtener el número de pasos dados por el usuario en un momento determinado a través del *framework FUNF* [6] y de los sensores del dispositivo. *FUNF* es un *framework* de código abierto creado por *MIT Media Lab* utilizado para recoger y analizar información de los *smartphones*. Está formado por sondas, que son módulos de *software* utilizados para recopilar los datos que ofrecen los sensores de teléfono de bajo nivel y contiene otras sondas específicas para recopilar datos de mayor nivel. Presenta un total de treinta sondas implementadas [25] y algunos ejemplos de ellas son:

- Registro de llamadas y de SMS
- Contactos
- Historial de Navegación
- Aplicaciones en ejecución
- Aplicaciones instaladas
- Estado encendido/apagado de la pantalla
- Estado de la batería
- GPS, ubicación, bluetooth y acelerómetro

A través de su API se podría usar en las aplicaciones que se desarrolle en *Android*, pero se decidió recoger información directamente de los sensores sin utilizar esta API, ya que el objetivo es contar los pasos y eso se podría realizar con los sensores integrados en los dispositivos móviles. Los sensores son un conjunto de dispositivos que nos proporcionan información del exterior y *Android* nos permite acceder a ellos. Podemos encontrar tres tipos de sensores: de movimiento, posición o entorno, aunque dependiendo del dispositivo estarán disponibles o no. Para contar los pasos dados por el usuario se realizaron pruebas implementando dos tipos de sensores de movimiento:

- **Sensor contador de pasos** [33]. Muestra la cantidad de pasos que da el usuario desde que se activó el sensor. Tiene más latencia que el sensor de detector de pasos pero es más preciso.
- **Sensor detector de pasos** [34]. Este sensor activa un evento cada vez que el usuario da un paso. La latencia es inferior que en el sensor de contador de pasos.

Ambos proporcionaban el número de pasos dados de forma correcta y precisa, pero tenían el inconveniente de que no todos los dispositivos lo tienen integrado y se decidió usar el sensor del acelerómetro [1] que si está incluido en todos los dispositivos móviles. El sensor del acelerómetro permite detectar si el dispositivo se mueve en una determinada dirección, si se agita o si vibra y mediante un conjunto de operaciones es posible contar los pasos.

Para contabilizar el número de pasos se han creado tres clases Java en el paquete *Presentation*:

- **SensorFilter.java.** Contiene un algoritmo<sup>18</sup> que sirve para filtrar los valores recogidos por el sensor del acelerómetro y que tengan una aproximación de un paso.
- **StepDetector.java.** Es una clase que contiene un método que permite actualizar el número de pasos dados cuando recibe cambios de estado del sensor.
- **StepListener.java.** Es una interfaz que recibe alertas sobre los pasos que se han detectado y se implementa en la actividad que inicia el sensor, que en este caso se corresponde con la clase *MainActivity.java*.

En la clase **MainActivity** se inicia el sensor del acelerómetro haciendo uso del método *registerListener()*<sup>19</sup> cuando se selecciona la opción del *checkbox* 'Contar pasos' y se detiene cuando se deselecciona esa opción con el método *unregisterListener()*<sup>20</sup>. Cada vez que cambia el sensor del acelerómetro, se ejecuta el método *onSensorChanged()* y este notifica a la clase *StepDetector* para comprobar si se ha producido un nuevo paso para después, mediante el método *step()*, contabilizar un paso más y guardarlo en la base de datos.

**Listado 5.13:** Detección de cambios en el sensor acelerómetro

```

1 public void onSensorChanged(SensorEvent event) {
2     if (event.sensor.getType() == Sensor.TYPE_ACCELEROMETER) {
3         simpleStepDetector.updateAccel(event.timestamp, ←
4             ↗ event.values[0], event.values[1], event.values[2]);
5     }

```

### Tarea 13. Diseño de la interfaz principal de la aplicación móvil y el menú de opciones

En esta tarea se realiza el diseño de la actividad principal de la aplicación móvil con distintas opciones que pueden ser seleccionadas por los usuarios para iniciar o finalizar la recogida de datos. De la misma forma que se mantenía la sesión iniciada en el *Login* con *SharedPreferences*, se ha guardado el estado de selección de los *checkbox* y se ha hecho uso de los mismos colores en tonos azules en las distintas actividades de la aplicación y la misma fuente de letra resultando la interfaz que se muestra en la Figura 5.20(a).

En la carpeta */res/menu* se ha creado un archivo denominado *menu\_aplicacion.xml* utilizado para diseñar el menú de la aplicación y en el que se han añadido tres iconos creados y editados con la herramienta Gimp. Cada una de estas opciones tienen los siguientes objetivos:

- **Información de la aplicación.** Se corresponde con la primera opción del menú empezando por la izquierda (5.20(b) y 5.20(c)), cuyo objetivo principal es informar sobre el propósito de la aplicación y los distintos pasos que tiene que realizar el usuario para que el sistema pueda monitorizar los eventos. Sucesivamente, se informa sobre el identificador del usuario que se le ha asignado y la opción de poder copiarlo para ingresar en la aplicación web de una forma rápida y eficaz. Por último, se muestra el enlace de la aplicación web como alternativa

<sup>18</sup><http://www.gadgetsaint.com/android/create-pedometer-step-counter-android/#.XwyEWygzbcc>

<sup>19</sup>Método *registerListener()*

<sup>20</sup>Método *unregisterListener()*

para visualizar los datos en cualquier dispositivo y la opción de borrar todos los datos de la aplicación. Para la justificación del texto se ha hecho uso de una librería externa<sup>21</sup> y que previamente se ha tenido que incluir como dependencia en el archivo *build.gradle*.

- **Acerca de la aplicación.** La segunda opción es un mensaje de información sobre el creador del sistema y el propósito del mismo que se muestra en la Figura 5.20(d).
- **Cerrar sesión.** La tercera opción permite al usuario cerrar la sesión iniciada en la aplicación.

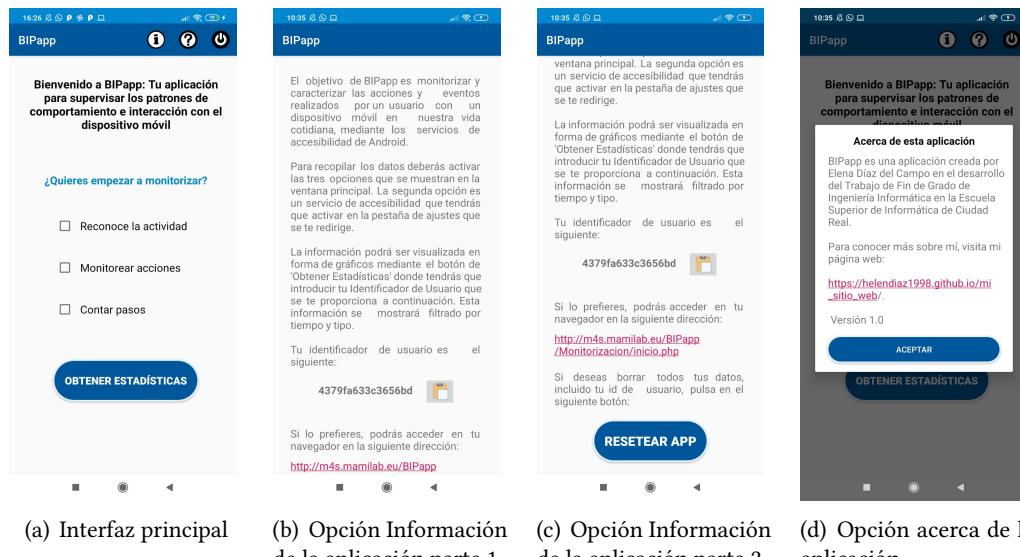


Figura 5.20: Menú de opciones de la aplicación móvil

Al finalizar esta iteración y haber desarrollado las tareas programadas, en la Figura 5.21 se muestra el estado del tablero Kanban, donde las tareas correspondientes en esta iteración han sido desarrolladas y van a ser sometidas a *test* y despliegue en la siguiente iteración mientras que una de las tareas correspondientes a la siguiente iteración empieza la fase de desarrollo.

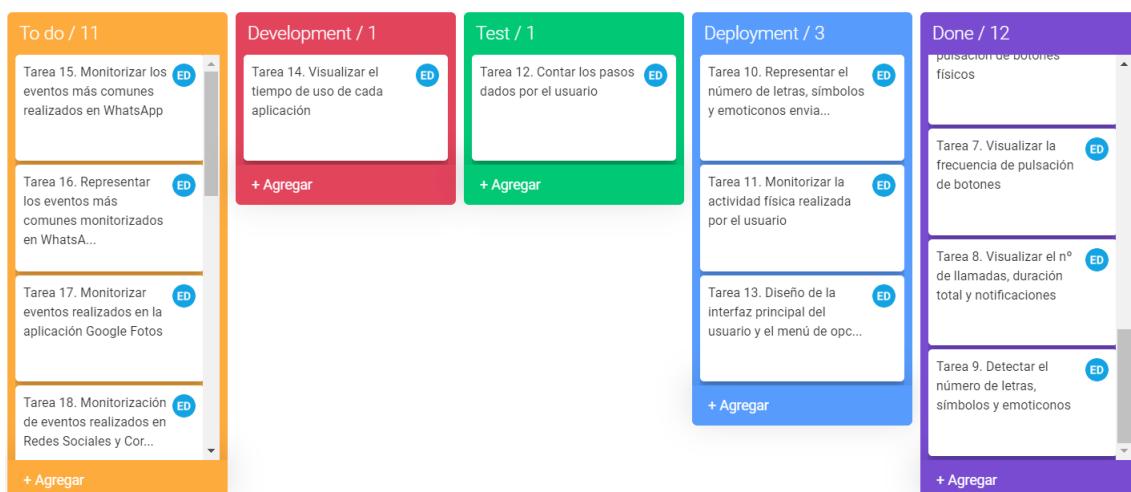


Figura 5.21: Tablero Kanban iteración 5

<sup>21</sup><https://github.com/amilcar-sr/JustifiedTextView>

### 5.2.7. Iteración 6

Esta iteración tiene una duración de 9 días y en ella se han desarrollado las tareas 14 y 15 que se corresponden con la visualización en gráficos del tiempo de uso de las aplicaciones y la monitorización de los eventos más comunes realizados en la aplicación WhatsApp.

#### Tarea 14. Visualización del tiempo de uso de las aplicaciones

Esta tarea consiste en realizar gráficos o tablas con el tiempo de uso de cada aplicación. Para el día y la semana actual se ha optado por representarlo en un gráfico circular, mientras que en el histórico se ha optado por realizar una tabla con el tiempo de uso de las diez aplicaciones más utilizadas. Se crearon varios *endpoints* en la *API REST* que consistían en sumar la diferencia del tiempo entre la hora de registro de eventos consecutivos acotados por el día y semana actual, y el histórico. Se representaron de la misma forma que los gráficos anteriores pero surgió un problema debido a que el tiempo de uso de las aplicaciones se calculaba en función de los paquetes reconocidos y al no ser posible conocer el nombre de la aplicación, las leyendas de los gráficos se representaban en función del paquete. Además, en la detección de actividades en primer plano se registran otros eventos como la actividad de inicio del dispositivo que impedían representar únicamente el tiempo de uso de aplicaciones utilizadas por el usuario.

Se consiguió implementar un método que permitía reconocer los paquetes y el nombre de las aplicaciones instaladas en el dispositivo. Se decidió almacenar esta información en la tabla de la base de datos que se muestra en la Figura 5.22 junto con el identificador del usuario. Estas aplicaciones serán registradas cuando el usuario acceda a la ventana principal de la aplicación y se actualizará cada vez que el servicio de accesibilidad detecte un evento de instalar o desinstalar una aplicación.

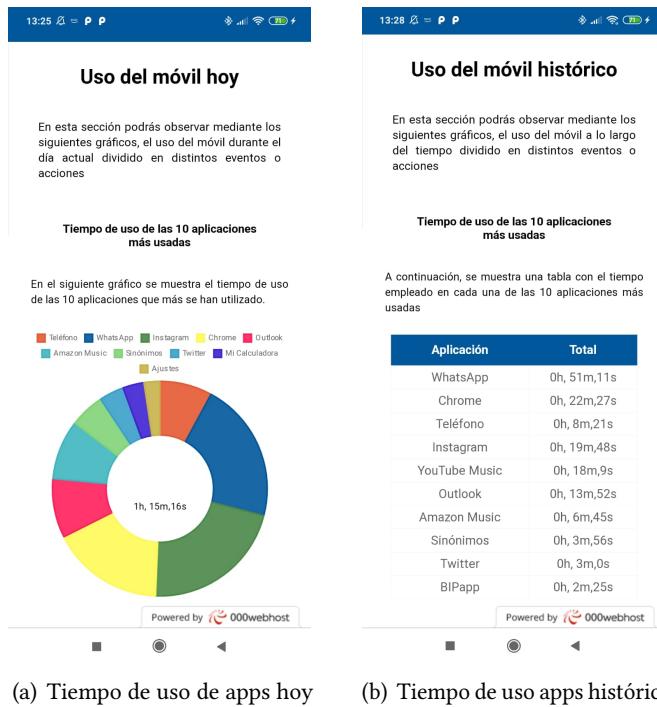
appsInstaladas	
Paquete	VARCHAR(1024)
Usuario	VARCHAR(1024)
App	VARCHAR(1024)
Id	INT(100)
Indexes	

Figura 5.22: Estructura tabla appsInstaladas

De esta forma, se calculará el tiempo de uso de cada aplicación registrada en la tabla anterior con el objetivo de filtrar paquetes que no se corresponden con aplicaciones del dispositivo y los paquetes al ser iguales en ambas tablas, se podía obtener el nombre original de la aplicación registrado en la tabla *appsInstaladas* y representar los gráficos en función de esos nombres como se muestra en la Figura 5.23.

#### Tarea 15. Monitorizar los eventos más comunes realizados en WhatsApp

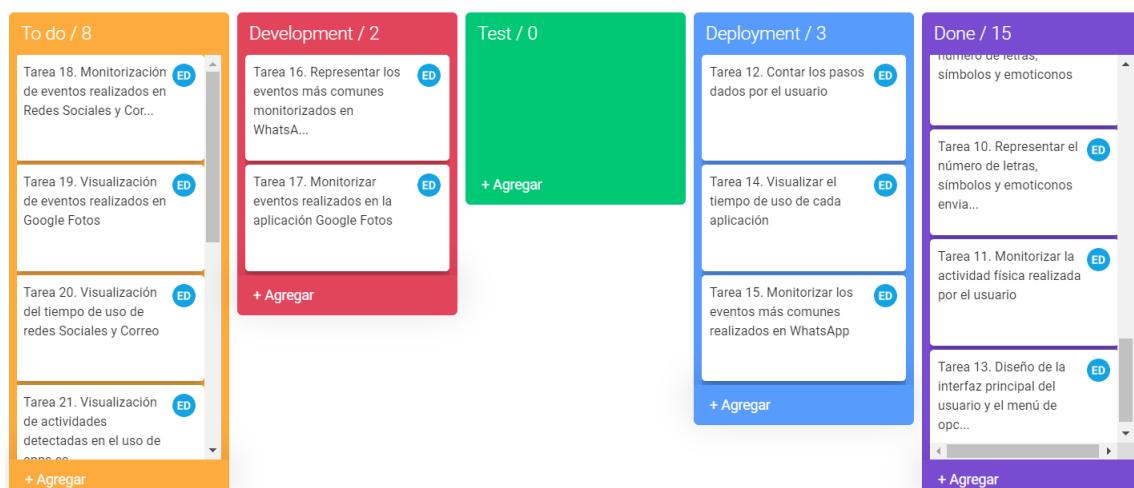
En esta tarea se ha añadido en el servicio de accesibilidad la posibilidad de detectar eventos comunes realizados en WhatsApp. Estos eventos se corresponden con acciones de enviar documentos, imágenes, ubicaciones etc., eventos de notificaciones recibidas y llamadas o videollamadas realizadas. Los eventos



**Figura 5.23:** Tiempo de uso de las aplicaciones más usadas

de envío de archivos o cualquier otro tipo de información se registrarán en la tabla *apps*, ya que en esa tabla se registran eventos que implican realizar un conteo básico del número de veces que se producen al cabo de un día, semana o histórico. Sin embargo, eventos relacionados con acciones de visualizar un vídeo o una imagen y escuchar un audio, se registrarán en la tabla *primerPlano* con el objetivo de poder calcular el tiempo empleado en estas actividades.

Tras haber desarrollado las tareas programadas en esta iteración, en la Figura 5.24 se muestra el estado del tablero Kanban, donde las tareas correspondientes en esta iteración han sido desarrolladas y van a ser integradas en el sistema final y las tareas de la siguiente iteración pasan al estado de desarrollo.



**Figura 5.24:** Tablero Kanban iteración 6

### 5.2.8. Iteración 7

Esta iteración tiene una duración de 11 días y en ella se han desarrollado las tareas 16 y 17 que se corresponden con la representación de eventos realizados en WhatsApp y la monitorización de eventos realizados en la aplicación Google Fotos.

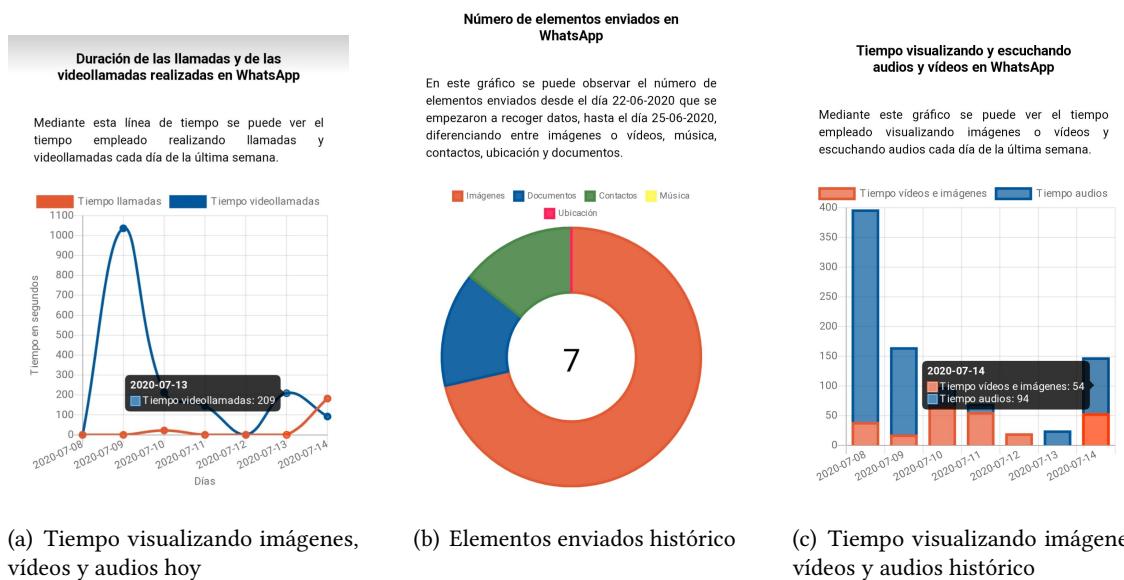
#### Tarea 16. Representar los eventos más comunes monitorizados en WhatsApp

Una vez que se han monitorizado los eventos más comunes realizados en WhatsApp, es necesario realizar un análisis de la información relevante que se debe representar. En la Tabla 5.5 se muestran las métricas que han sido procesadas en la *API REST* mediante la creación de varios *endpoints* y representadas sucesivamente a partir de los eventos monitorizados en la tarea 15.

**Tabla 5.5:** Métricas de eventos realizados en WhatsApp para ser representados

Evento	Métrica
Elementos enviados	Número de imágenes, documentos, contactos, archivos de música o ubicaciones enviadas en el día y en la semana actual y en el histórico.
Llamadas y videollamadas	Calcular el número de llamadas y videollamadas, junto con la duración total de las mismas, realizadas en el día y la semana actual y en el histórico.
Visualización de imágenes o vídeos y reproducción de audios	Tiempo empleado visualizando imágenes o vídeos y escuchando audios en el día y semana actual y en el histórico.
Otros datos	Nº de conversaciones abiertas y nº de notificaciones recibidas en el día y en la semana actual y en el histórico.

Algunos de los gráficos resultantes de las métricas obtenidas se muestran en la Figura 5.25.



**Figura 5.25:** Gráficos métricas WhatsApp

### Tarea 17. Monitorizar eventos realizados en la aplicación Google Fotos

En esta tarea se ha realizado en primer lugar un estudio de distintos eventos que ocurren en la aplicación de Google Fotos y que se pueden detectar con el servicio de accesibilidad para obtener una serie de métricas relevantes que se puedan representar. Los eventos reconocidos por el servicio de accesibilidad permiten detectar la visualización de imágenes o vídeos en esta aplicación. Estos datos serán enviados a la tabla *primerPlano* para poder calcular el tiempo de visualización de imágenes y vídeos mientras la aplicación se ejecuta en primer plano.

Una vez que se han desarrollado las tareas programadas en esta iteración las tareas serán sometidas a *test* para desplegarlas sucesivamente en la siguiente iteración mientras que la primera tarea de la siguiente iteración empieza la fase de desarrollo como se muestra en la Figura 5.26.

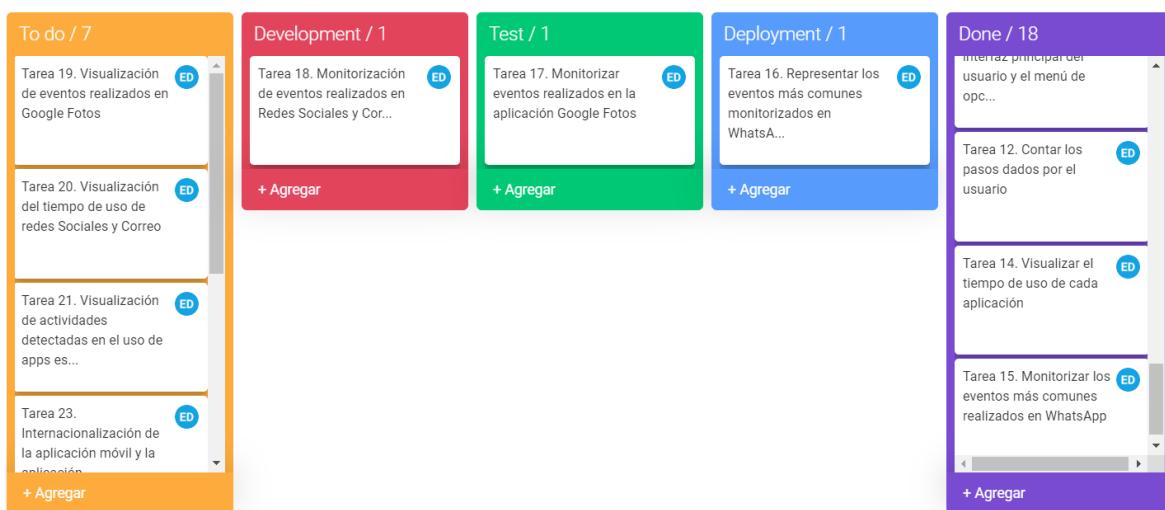


Figura 5.26: Tablero Kanban iteración 7

### 5.2.9. Iteración 8

Esta iteración tiene una duración de 8 días y en ella se han desarrollado las tareas 18 y 19 que se corresponden con la monitorización de distintos eventos realizados en las redes sociales y aplicaciones de correo electrónico más utilizadas, y la visualización de métricas obtenidas de la aplicación Google Fotos.

### Tarea 18. Monitorización de eventos realizados en Redes Sociales y Correo

En esta tarea se han realizado las últimas modificaciones en el servicio de accesibilidad que han permitido monitorizar eventos comunes realizados por el usuario en las aplicaciones de redes sociales más utilizadas como son Instagram, Facebook, Twitter y Linkedin, junto con las aplicaciones de correo electrónico más usadas como son Gmail y Outlook. En las siguientes tablas se muestran una serie de eventos que se han detectado con el servicio de accesibilidad una vez que se ha configurado de la misma forma que en iteraciones anteriores y las métricas que se desean representar a partir de esa información.

**Tabla 5.6:** Métricas de eventos realizados en Instagram

<b>Instagram</b>	
<b>Evento</b>	<b>Métrica</b>
Notificaciones	Número de notificaciones recibidas de esta aplicación en el día y semana actual y en el histórico.
Me gusta	Número de me gustas en publicaciones en el día y semana actual y en el histórico.
Comentarios escritos	Número de comentarios escritos en publicaciones en el día y semana actual y en el histórico.
Comentarios visualizados	Número de comentarios visualizados en publicaciones en el día y semana actual y en el histórico.
Publicaciones compartidas	Número de publicaciones compartidas en publicaciones en el día y semana actual y en el histórico.
Tiempo de uso	Tiempo de uso de la app en el día y semana actual y en el histórico.

**Tabla 5.7:** Métricas de eventos realizados en Facebook

<b>Facebook</b>	
<b>Evento</b>	<b>Métrica</b>
Notificaciones	Número de notificaciones recibidas de esta aplicación en el día y semana actual y en el histórico.
Me gusta	Número de me gustas en publicaciones en el día y semana actual y en el histórico.
Comentarios escritos	Número de comentarios escritos en publicaciones en el día y semana actual y en el histórico.
Publicaciones compartidas	Número de publicaciones compartidas en publicaciones en el día y semana actual y en el histórico.
Tiempo de uso	Tiempo de uso de esta aplicación en el día y en la semana actual, y en el histórico.

**Tabla 5.8:** Métricas de eventos realizados en Twitter

<b>Twitter</b>	
<b>Evento</b>	<b>Métrica</b>
Notificaciones	Número de notificaciones recibidas de esta aplicación en el día y semana actual y en el histórico.
Tweets compartidos	Número de Tweets compartidos en el día y semana actual y en el histórico.
Tweets respondidos	Número de Tweets respondidos en el día y semana actual y en el histórico.
Tweets escritos	Número de Tweets escritos en el día y semana actual y en el histórico.
Tweets eliminados	Número de Tweets eliminados en el día y semana actual y en el histórico.
Tiempo de uso	Tiempo de uso de esta aplicación en el día y en la semana actual, y en el histórico.

**Tabla 5.9:** Métricas de eventos realizados en Linkedin

<b>Linkedin</b>	
<b>Evento</b>	<b>Métrica</b>
Notificaciones	Nº de notificaciones recibidas en el día y semana actual y en el histórico.
Publicaciones escritas	Nº de publicaciones escritas en el día y semana actual y en el histórico.

Publicaciones eliminadas	Número de publicaciones eliminadas en el día y semana actual y en el histórico.
Publicaciones recomendadas	Número de publicaciones recomendadas en el día y semana actual y en el histórico.
Comentarios	Número de comentarios escritos en publicaciones en el día y semana actual y en el histórico.
Tiempo de uso	Tiempo de uso de esta aplicación en el día y en la semana actual, y en el histórico.

**Tabla 5.10:** Métricas de eventos realizados en Gmail

Gmail	
Evento	Métrica
Notificaciones	Número de notificaciones recibidas de esta aplicación en el día y semana actual y en el histórico.
Correos enviados	Número de correos enviados en el día y en la semana actual, y en el histórico.
Correos eliminados	Número de correos eliminados en el día y en la semana actual, y en el histórico.
Tiempo de uso	Tiempo de uso en el día y en la semana actual, y en el histórico.

**Tabla 5.11:** Métricas de eventos realizados en Outlook

Outlook	
Evento	Métrica
Notificaciones	Nº de notificaciones recibidas de esta aplicación en el día y semana actual y en el histórico.
Correos enviados	Calcular el nº de correos enviados en el día y en la semana actual, y en el histórico.
Tiempo de uso	Calcular el tiempo de uso de esta aplicación en el día y en la semana actual, y en el histórico.

### Tarea 19. Visualización de eventos realizados en Google Fotos

Una vez que se han monitorizado los eventos de visualizar imágenes o vídeos en Google Fotos, es necesario realizar un análisis de la información relevante que se debe representar. En la Tabla 5.12 se muestran las métricas que han sido procesadas en la API a partir de los eventos monitorizados en la tarea 17.

**Tabla 5.12:** Métricas de eventos realizados en Google Fotos para ser representados

Evento	Métrica
Nº de imágenes visualizadas	Calcular el número de imágenes visualizadas en Google Fotos en el día y en la semana actual, y en el histórico.
Nº de vídeos visualizados	Calcular el número de vídeos visualizados en Google Fotos en el día y en la semana actual, y en el histórico.
Tiempo empleado viendo vídeos	Calcular el tiempo empleado visualizando vídeos en Google Fotos en el día y en la semana actual, y en el histórico.
Tiempo empleado viendo imágenes	Calcular el tiempo empleado visualizando imágenes en Google Fotos en el día y en la semana actual, y en el histórico.
Tiempo de uso de la app	Tiempo de uso en el día y en la semana actual, y en el histórico.

Algunos de los gráficos resultantes de las métricas obtenidas se muestran en la Figura 5.27.

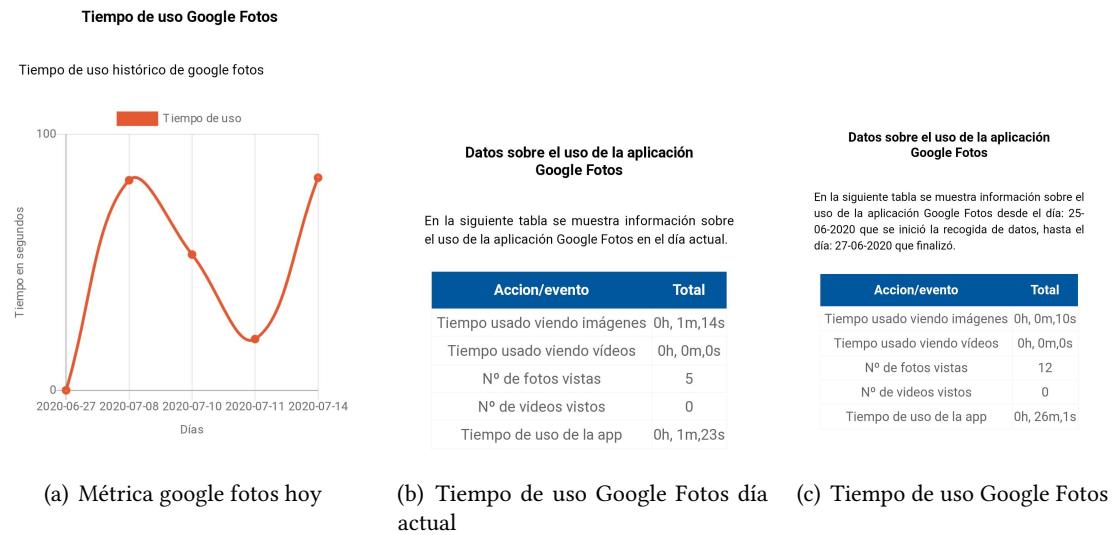


Figura 5.27: Gráficos métricas Google Fotos

Al finalizar esta iteración y haber desarrollado las tareas programadas, en la Figura 5.28 se muestra el estado del tablero Kanban, donde las tareas correspondientes en esta iteración han sido desarrolladas y van a ser sometidas a *test* en la siguiente iteración mientras que las tareas correspondientes a la siguiente iteración empiezan la fase de desarrollo.

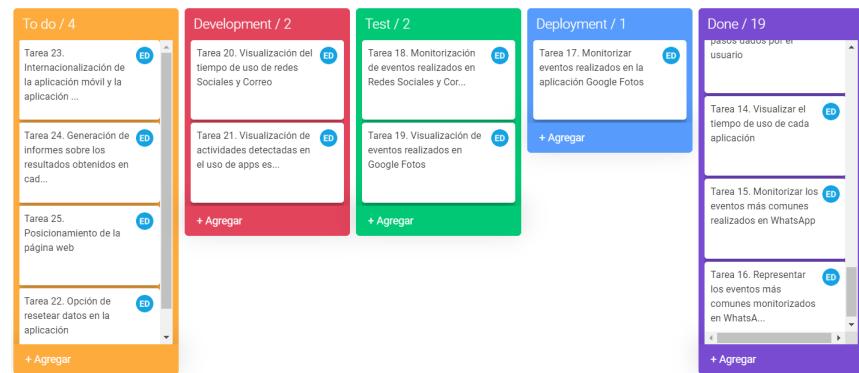


Figura 5.28: Tablero Kanban iteración 8

### 5.2.10. Iteración 9

Esta penúltima iteración tiene una duración de 13 días y en ella se han desarrollado las tareas 20 y 21 que se corresponden con la visualización del tiempo de uso y métricas recogidas del uso de las redes sociales y correo electrónico junto con la visualización de actividades detectadas en el uso de aplicaciones concretas.

#### Tarea 20. Visualización del tiempo de uso y métricas de Redes Sociales y Correo

Una vez que se han monitorizado los eventos más comunes realizados en las redes sociales y en el correo electrónico se han construido distintos gráficos con las métricas obtenidas que han sido

procesadas en la *API REST* mediante la creación de varios *endpoints* como se muestra en la Figura 5.29.

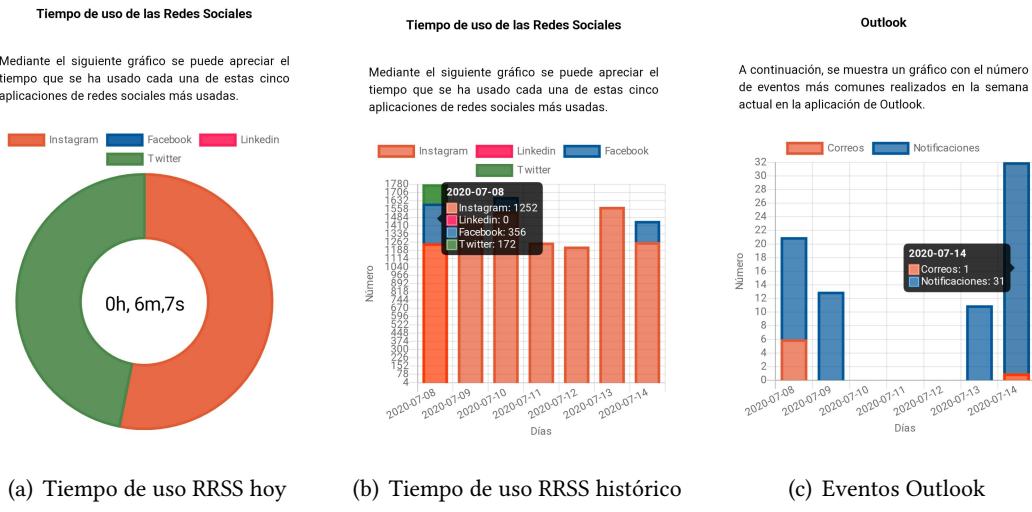


Figura 5.29: Gráficos métricas RRSS y correo

### Tarea 21. Visualización de actividades detectadas en el uso de apps específicas

En la realización de esta tarea fue necesario llevar a cabo el pensamiento a la máxima abstracción para conseguir relacionar las actividades físicas que realiza un usuario con el tipo de evento o aplicación que está utilizando en ese momento. Para ello, se seleccionaron un conjunto de eventos y aplicaciones más usadas por los usuarios para realizar gráficos del tiempo de uso de esas aplicaciones y eventos mientras el usuario está parado o andando para permitir analizar la capacidad que tiene el usuario de realizar actividades en movimiento. Las aplicaciones y eventos que fueron seleccionadas para realizar esta tarea son algunas como: llamadas realizadas con el dispositivo o con WhatsApp, acción de escribir letras y uso de las aplicaciones: WhatsApp, Gmail, Outlook, Instagram, Facebook o Twitter.

Para detectar el evento de escribir en movimiento, fue necesario crear una nueva tabla denominada *primerPlanoLetras* que registra además de las aplicaciones en primer plano, el evento de escribir letras, símbolos u emoticonos. Esta acción no se podía realizar conjuntamente en la tabla *primerPlano*, debido a que puedes estar escribiendo en diferentes aplicaciones y si se produce ese evento no se calcularía adecuadamente el tiempo de uso de las aplicaciones. Igualmente, se creó una tabla denominada *detectaActividadesLetra* que registra el tipo de actividad realizada en un momento determinado junto con el último registro de la tabla mencionada anteriormente. Ambas tablas tienen la misma estructura que las tablas *primerPlano* y *detectaActividades*. En la Figura 5.30 se muestra un ejemplo de las gráficas realizadas, donde se muestra el tiempo que se ha estado escribiendo en el dispositivo realizando una determinada actividad física en el día actual.

Llegando al término de esta penúltima iteración, en la Figura 5.31 se muestra el estado del tablero Kanban, donde las tareas de esta iteración han sido desarrolladas y serán sometidas a *test* mientras que dos de las cuatro tareas correspondientes a la siguiente iteración empiezan la fase de desarrollo.

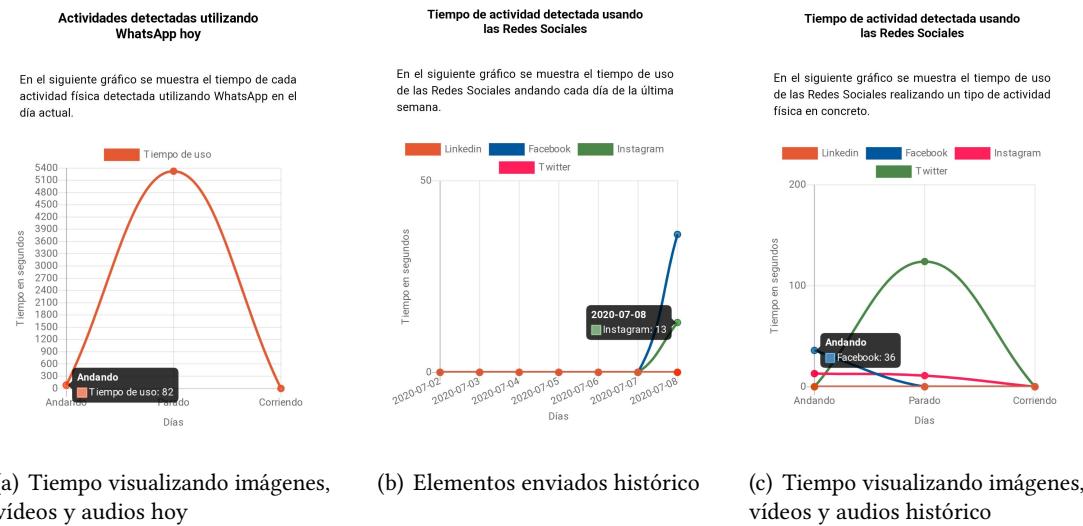


Figura 5.30: Gráficos métricas WhatsApp

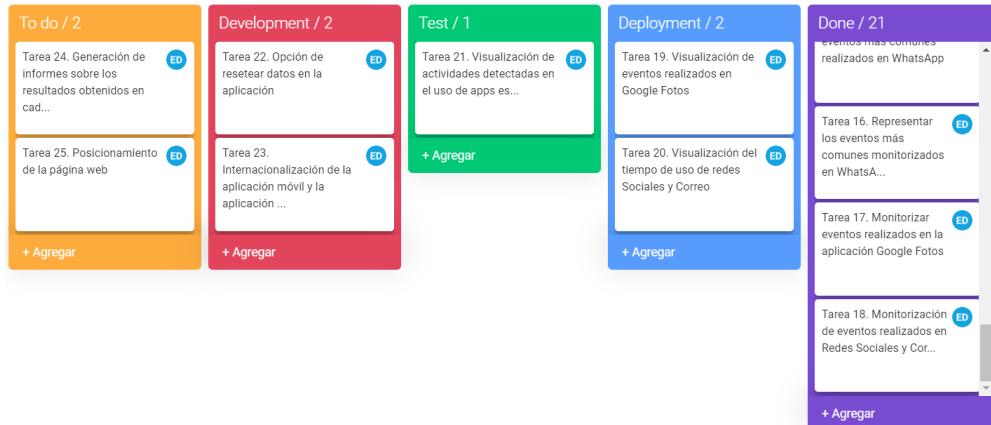


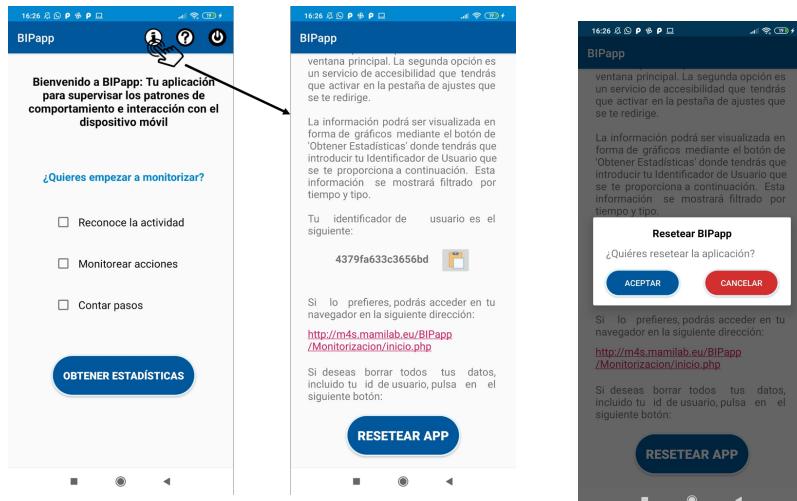
Figura 5.31: Tablero Kanban iteración 9

### 5.2.11. Iteración 10

Esta iteración ha tenido una duración de 13 días y en ella se realizan las últimas tareas restantes para finalizar el proceso de desarrollo del proyecto.

#### Tarea 22. Opción resetear datos en la aplicación

Por motivos de seguridad, es necesario ofrecer una opción en la aplicación móvil que permita al usuario borrar toda la información recogida de su dispositivo, incluido su identificador de usuario. Esta opción se ha implementado en la actividad 'Información sobre la aplicación' que se encuentra en una opción del menú que se muestra en la Figura 5.32(a), debido a que es conveniente que no esté disponible a simple vista para evitar que pueda ser seleccionada accidentalmente. No obstante, se ha añadido un mensaje de alerta (Figura 5.32(b)) para que el usuario pueda confirmar la acción de resetear los datos de la aplicación. Si el usuario acepta, se ejecutará un *script* alojado en el servidor que borrará todos los datos de las tablas de la base de datos que contengan su identificador de usuario y la aplicación le redirigirá a la actividad de inicio de sesión y registro.



(a) Opción resetear información

(b) Mensaje de alerta opción resetear

**Figura 5.32:** Opción resetear información

### Tarea 23. Internacionalización de la aplicación móvil y la aplicación web

Ofrecer una aplicación en varios idiomas impulsa la internacionalización y comercialización de la aplicación y un aumento en el número de descargas que implica un mayor número de usuarios que hacen uso de la aplicación. Es el motivo principal por el que se ha decidido ofrecer el sistema en dos idiomas: inglés y español.

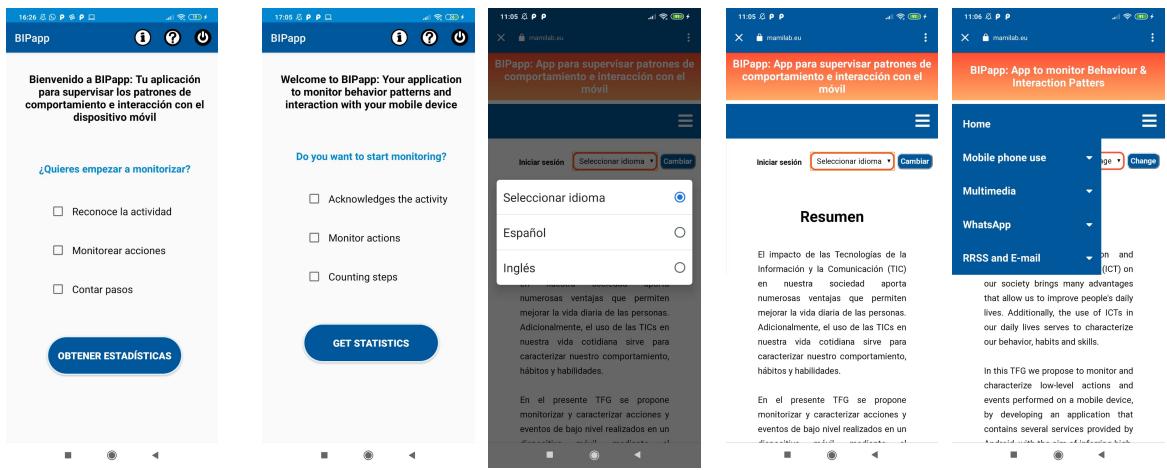
#### Internacionalización de la aplicación móvil

Para ofrecer la aplicación tanto en inglés como en español, se han incluido recursos específicos de un idioma en concreto. En el directorio *res/values* del proyecto es posible crear tantos archivos *strings.xml* como número de idiomas se quiera ofrecer la aplicación. Cada componente que se incluye en una interfaz de usuario tiene que tener declarado un identificador en el archivo *strings.xml*. Esta aplicación contiene dos archivos de este tipo que se corresponden con el idioma español e inglés. Ambos archivos presentan los mismos identificadores de cada componente pero con distinta descripción en función del idioma de ese archivo. Cuando la aplicación se inicia, comprueba el idioma que tiene configurado el dispositivo y por defecto mostrará la aplicación en español. Si el idioma del dispositivo se corresponde con inglés, toda la información de la aplicación se presentará en ese lenguaje. En la Figura 5.33(a) se muestra la aplicación en español y en la Figura 5.33(b) se muestra en inglés.

#### Internacionalización de la aplicación web

Para internacionalizar la aplicación web se ha optado por incluir un *ComboBox* en la página de inicio que permita al usuario seleccionar el idioma deseado, aunque por defecto la aplicación se mostrará en español. La internacionalización de la aplicación se ha implementado en PHP y consiste en mostrar el contenido de la página según la opción que se ha seleccionado. Si el idioma es español se cargará el archivo *es.php* que contiene todos los títulos, párrafos o descripciones en español. En cambio, si el idioma es inglés se cargará el archivo *en.php* que contiene el mismo contenido

definido con el mismo identificador en inglés. En la Figura 5.33(d) se muestra la aplicación web en un ordenador en español y en la Figura 5.33(e) se muestra en Inglés.



(a) Aplicación móvil en español (b) Aplicación móvil en inglés (c) Selección idiomas aplicación web (d) Aplicación web en español (e) Aplicación web en inglés

**Figura 5.33:** Internacionalización de la aplicación móvil y web

#### Tarea 24. Generación de informes sobre los resultados obtenidos en cada sección

Un documento PDF es una forma muy útil de observar la información en un archivo sin conexión a internet para analizar los datos. En este proyecto se han generado informes de cada sección con el objetivo de poder guardar un registro de todos los gráficos en un momento determinado y ofrecer a los investigadores o especialistas la posibilidad de analizar esos informes. Existen varias librerías de JavaScript que permiten generar archivos PDF a partir de HTML, pero en este caso, se ha hecho uso de la librería *jsPDF* [29]. Es una de las mejores librerías para realizar esta función y es simple de utilizar. En primer lugar hay que importarla en el proyecto como un *script* y sucesivamente hay que generar en cada sección el método que se muestra en el Listado F.2 del Anexo F donde se configuran todos los títulos, párrafos, gráficos o tablas que se deben incluir en el documento a partir de los identificadores de la página HTML y que se invoca cuando el usuario selecciona la opción '*Descargar informe*'.

#### Tarea 25. Posicionamiento de la aplicación web

Si la internacionalización de la aplicación ofrece ventajas competitivas, el hecho de ofrecer el sistema en una aplicación web permite mejorar el *posicionamiento SEO* del sistema, ya que puede ganar visibilidad y llegar a los clientes potenciales por distintos caminos. Por este motivo, en este proyecto se han analizado las palabras clave que definen el proyecto para incluirlas en los títulos, metadescipciones y en el contenido. Mejorar el posicionamiento implica que cuando un usuario introduzca algunas de estas palabras en el navegador, la aplicación aparezca en los resultados más relevantes. Estas descripciones se incluyen entre las etiquetas *head* de los archivos HTML. Un ejemplo del posicionamiento de la sección WhatsApp del día actual se muestra en el Listado F.1. Además, para evitar que se produzcan errores en las búsquedas y el usuario abandone la aplicación, se ha diseñado una página de *error 404* (Figuras F.1 y F.2) que ofrece al usuario la posibilidad de ser redirigido a la

página de inicio. Por otro lado, la herramienta gratuita *Google Search Console* [18] ayuda a supervisar, mantener y solucionar los problemas de la página web en relación con los resultados de la búsqueda. Esta herramienta te confirma si se ha indexado el sitio en el navegador, soluciona los problemas, consulta los datos de tráfico del sitio e informa de las páginas que contienen enlaces a esta aplicación web. En este Trabajo de Fin de Grado se ha hecho uso de esta herramienta y se ha indexado la aplicación web (Figura F.3), que ha implicado la creación de un archivo *sitemap.xml*, para conseguir posicionarla en los primeros puestos de los resultados de búsqueda.

Tras la finalización de la última iteración del desarrollo de este proyecto, las distintas tareas planificadas al inicio del proyecto se han completado como se muestra en el tablero Kanban en la Figura 5.34.



Figura 5.34: Tablero Kanban iteración 10

## 5.3. EVALUACIÓN

Una vez que se ha desarrollado el sistema al completo, y tras haber realizado pruebas en cada iteración incremental, es necesario comprobar que funciona correctamente y que cumple con el objetivo principal propuesto en el Capítulo 2. Para ello, se ha estado probando durante una semana en tres dispositivos al mismo tiempo, que se corresponden con los teléfonos móviles de los dos tutores de este proyecto y la autora del mismo. En primer lugar, se han detectado distintas anomalías y errores que presenta el sistema y se han corregido sucesivamente. Por otro lado, se ha realizado un análisis y comparación de los datos más relevantes recogidos de estos usuarios para poder detectar distintos comportamientos y hábitos en el uso de los dispositivos móviles. Este análisis no pretende tener ninguna significancia estadística debido a la baja población ( $n=3$ ) pero demuestra que el sistema cumple el objetivo principal y muestra el potencial para el análisis de hábitos y comportamientos del sistema presentado en este TFG.

### 5.3.1. Errores detectados y solucionados en la fase de evaluación

A continuación, se detallan diferentes errores detectados después de realizar el análisis de los distintos datos recogidos por estos tres usuarios y la forma en la que se han abordado para solucionarlos.

- **Conteo exacto del número de letras, símbolos y emoticonos enviados.** Uno de los errores que se detectaron fue el conteo incorrecto del número de letras, símbolos u emoticonos escritos. Esto era debido a que la detección de cada elemento se hacía correctamente, pero

también se contaban los caracteres eliminados. Sin embargo, con la detección del evento `TYPE_VIEW_TEXT_CHANGED` y con la clase `AccessibilityRecord`<sup>22</sup> que contiene información sobre el cambio en un componente `View`, se consiguió calcular el número exacto de elementos escritos. Esta clase ofrece los métodos `getAddedCount()` que permite obtener el número de caracteres agregados y `getRemovedCount()` que indica el número de caracteres eliminados. De esta forma, se comprueba qué, si el número de elementos añadidos menos el número de elementos eliminados es igual a uno, se ha añadido un nuevo elemento. Si se produce el proceso inverso, significa que se ha borrado un elemento y se añade en el número de elementos eliminados.

- **Detección del bloqueo y desbloqueo de pantalla en segundo plano.** En primer lugar la detección del estado de bloqueo y desbloqueo de la pantalla se implementó en el hilo de la actividad principal y únicamente se detectaban estos eventos cuando la aplicación móvil BIPapp estuviese iniciada. Este fragmento de código indicado en la tarea 5 y que se trata de un `broadcastReceiver`, fue incorporado en un servicio que se ejecuta indefinidamente en segundo plano y que se inicia a la vez que el servicio de accesibilidad, cuando se selecciona ese `checkbox` en la aplicación móvil. De esta forma, es posible calcular el tiempo que el dispositivo permanece bloqueado y desbloqueado.
- **Contar los pasos en segundo plano.** El conteo de pasos realizados por el usuario se detectaba en primer plano y cuando la aplicación finalizaba se dejaba de contabilizar. Para solucionar este error, se implementó un servicio en segundo plano que realizaba las mismas funciones, permitiendo de esta forma contar los pasos dados por los usuarios indefinidamente.
- **Mejoras en los gráficos.** Se han realizado mejoras en algunos gráficos para aumentar la visibilidad en el diseño móvil y se han añadido gráficos en las secciones semana e histórico de la pestaña multimedia que muestran el número de pasos dados cada día.

### 5.3.2. Análisis de hábitos y comportamientos

En esta sección, se ha realizado un análisis y comparación de los datos más relevantes recogidos de los tres dispositivos móviles de los dos tutores y de la autora de este proyecto, para poder detectar distintos comportamientos y hábitos en el uso de los mismos. En concreto, se ha realizado una comparación del tiempo de uso de las cinco aplicaciones más usadas por los usuarios, el número de conversaciones abiertas y notificaciones recibidas en WhatsApp, el tiempo de visualización de imágenes o vídeos y reproducción de audios en WhatsApp, el porcentaje de letras, símbolos o emoticonos escritos, y el tiempo de uso de WhatsApp en comparación con el tiempo que se ha usado esta app mientras los usuarios están andando.

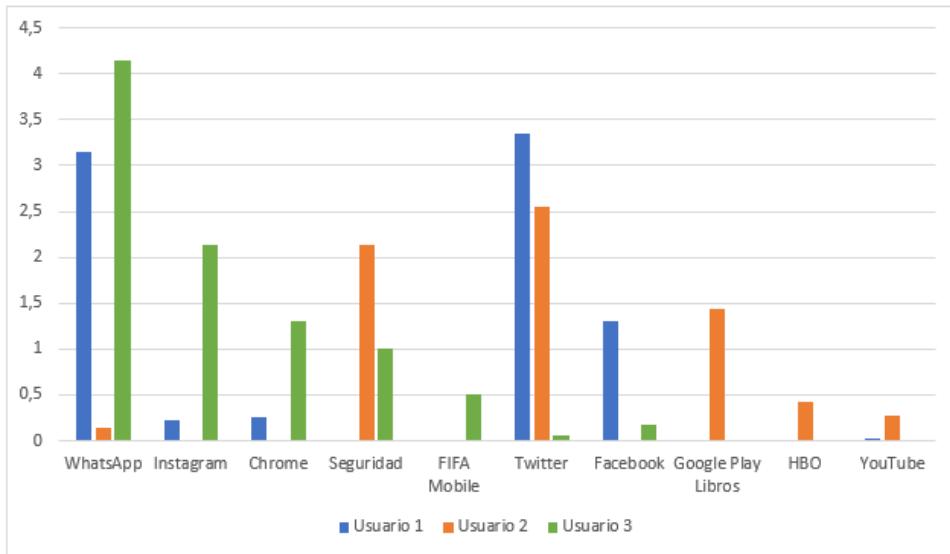
#### Comparación del tiempo de uso de las cinco aplicaciones más usadas por los usuarios

Uno de los datos más relevantes que nos proporciona el sistema, es el tiempo de uso de cada aplicación. La Figura 5.35 muestra una comparación del tiempo de uso de las cinco aplicaciones más utilizadas por los tres usuarios (medido en horas), teniendo en cuenta que algunas aplicaciones son usadas por los tres y otras no. Se puede observar que las aplicaciones más utilizadas se corresponden con redes sociales, destacando un mayor uso de Instagram por parte del primer y tercer usuario y un mayor

---

<sup>22</sup><https://developer.android.com/reference/android/view/accessibility/AccessibilityRecord>

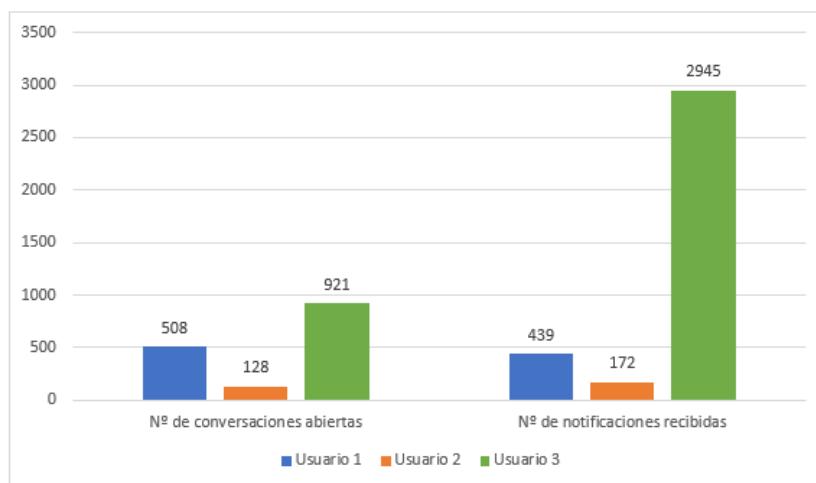
uso de Twitter en el caso del primer y segundo usuario. El resto de aplicaciones se corresponde con ocio, como juegos, visualización de vídeos y películas en YouTube y HBO, o lectura.



**Figura 5.35:** Comparativa del tiempo de uso de las aplicaciones más usadas por los usuarios medida en horas

#### Comparación número de conversaciones abiertas y notificaciones recibidas en WhatsApp

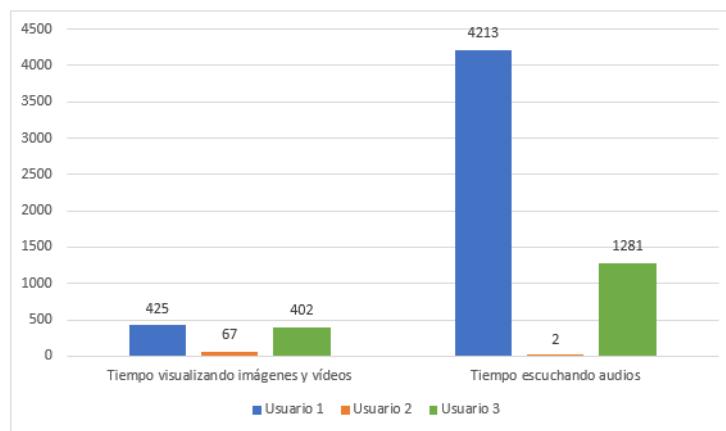
Con el objetivo de estudiar el uso que hace el usuario de la aplicación WhatsApp, se ha realizado una comparación entre el número de conversaciones que ha abierto el usuario, junto con el número de notificaciones recibidas que puede indicar la actividad diaria del uso de esta app. Como se ilustra en la Figura 5.36, el tercer usuario ha recibido más notificaciones y eso puede implicar que haya abierto un mayor número de conversaciones. Por otra parte, como se mostraba en la Figura 5.35, este usuario ha utilizado más tiempo esta app seguido por el primer usuario qué, como se puede apreciar, ha recibido un mayor número de notificaciones y ha abierto un mayor número de conversaciones que el segundo usuario.



**Figura 5.36:** Comparativa del número de conversaciones abiertas y notificaciones recibidas en WhatsApp de cada usuario

### Comparación del tiempo de visualización de imágenes o vídeos y reproduciendo de audios en WhatsApp

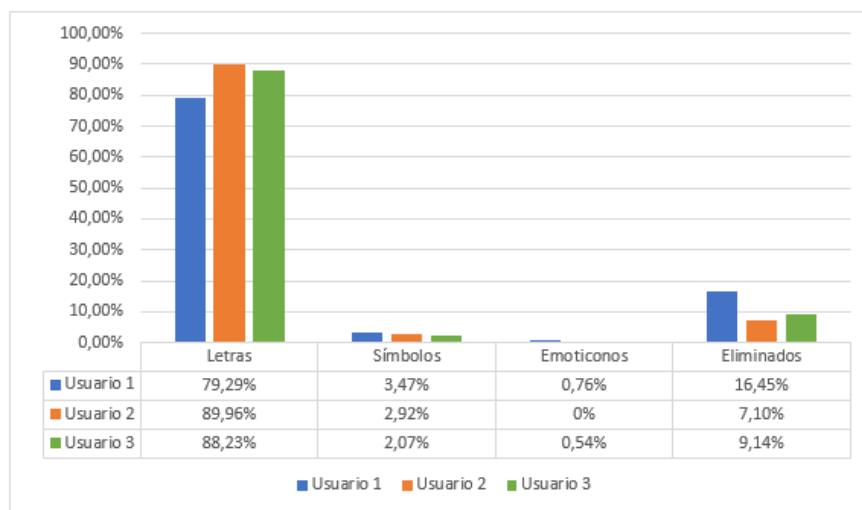
Por otro lado, centrándonos en aspectos más concretos como el tiempo de visualización de imágenes o vídeos y el tiempo de reproducción de audios (medido en segundos), en la Figura 5.37 se muestra una comparativa entre los tres usuarios, que corrobora que el segundo usuario al usar menos esta aplicación y haber abierto un menor número de conversaciones, el tiempo de visualización de imágenes y vídeos es escaso. Sin embargo, los otros dos usuarios emplean un tiempo similar en el tiempo de visualización de vídeos e imágenes, pero un tiempo significativamente diferente en la reproducción de audios.



**Figura 5.37:** Comparativa del tiempo en segundos, de visualización de vídeos e imágenes y reproducción de audios en WhatsApp

### Comparación del porcentaje de letras, símbolos o emoticonos escritos

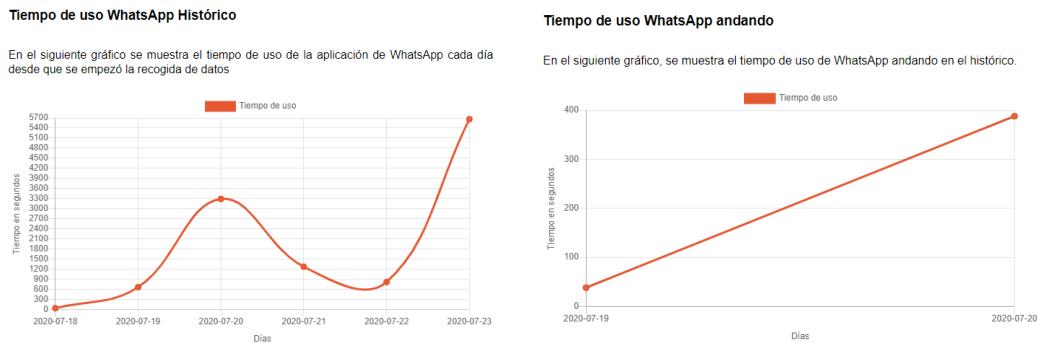
En la Figura 5.38 se muestra el porcentaje de letras, símbolos y emoticonos enviados en la última semana. Los tres usuarios coinciden en que el mayor porcentaje del número de elementos escritos, se corresponden con letras. El porcentaje de emoticonos es muy inferior, donde se puede apreciar que el segundo usuario no ha escrito ningún emoticono.



**Figura 5.38:** Comparativa del porcentaje de letras, símbolos y emoticonos enviados

### Comparación del tiempo de uso de WhatsApp y el tiempo que se ha usado mientras los usuarios están andando

Analizando el tiempo de uso de la aplicación WhatsApp de estos tres usuarios, se puede comparar la relación entre el tiempo de uso de esta aplicación y el tiempo que se ha hecho uso de ella mientras se realizaba la actividad de andar. Se puede comprobar que el tercer usuario ha utilizado esta aplicación más tiempo andando que otros usuarios, destacando en concreto el segundo usuario, que al utilizar esta aplicación un tiempo muy inferior al anterior, no ha utilizado esta aplicación realizando este tipo de actividad física.



**Figura 5.39:** Comparativa tiempo de uso WhatsApp y el tiempo que se ha usado mientras el primer usuario está andando



**Figura 5.40:** Comparativa tiempo de uso WhatsApp y el tiempo que se ha usado mientras el segundo usuario está andando



**Figura 5.41:** Comparativa del tiempo de uso WhatsApp y el tiempo que se ha usado mientras el tercer usuario está andando

---

## CAPÍTULO 6

# CONCLUSIONES

---

En este último capítulo que pone fin al desarrollo de la memoria de este Trabajo de Fin de Grado, se analizarán si se han cumplido los objetivos que fueron definidos en el Capítulo 2. También, se estudiarán y se justificarán las competencias obtenidas de la intensificación de Tecnologías de la Información cursada por la autora de este proyecto. En último lugar, se propondrán algunas mejoras y trabajos futuros en el sistema y un balance a nivel personal sobre los conocimientos aprendidos y la satisfacción de la realización de este proyecto.

### 6.1. OBJETIVOS ALCANZADOS

En el Capítulo 2 se definió un objetivo principal desglosado en varios objetivos específicos, junto con un conjunto de objetivos teóricos-prácticos que se deseaban alcanzar con la realización de este proyecto. Ahora, tras finalizarlo, es el momento de analizar el sistema que se ha desarrollado para comprobar si se han completado esos objetivos. El objetivo principal que se propuso consistía en *caracterizar el comportamiento y hábitos de uso del dispositivo móvil mediante el desarrollo de una aplicación que haga uso de servicios Android y monitorice acciones y eventos, tanto interactivos como del sistema, relacionándolos con la actividad física realizada en cada momento*. Para verificar si este objetivo general se ha cumplido, es necesario repasar uno por uno los objetivos secundarios especificados:

#### **Captura de acciones y eventos, tanto interactivos, como del sistema, en dispositivos Android**

Para llevar a cabo este objetivo se han implementado un conjunto de servicios que permiten monitorizar distintos tipos de acciones y eventos:

1. Monitorización de eventos producidos por la interacción del usuario con el dispositivo móvil mediante un servicio de accesibilidad, que ha sido configurado para capturar el conjunto de eventos producidos en dispositivos Android que se enumeran a continuación:
  - Detección de la pulsación de botones físicos como subir o bajar el volumen, bloquear o desbloquear el dispositivo (Tarea 5, iteración 5.2.2)
  - Detección de las aplicaciones ejecutadas en primer plano (Tarea 4, iteración 5.2.2)
  - Monitorizar el número de letras, símbolos u emoticonos enviados (Tarea 6, iteración 5.2.7).
  - Detección de las llamadas realizadas con el dispositivo (Tarea 6, iteración 5.2.2)

- Monitorizar los eventos más comunes realizados en WhatsApp (Tarea 15, iteración 5.2.7), Google Fotos (Tarea 17, iteración 5.2.8), redes sociales y correo electrónico (Tarea 18, iteración 5.2.9).
2. Monitorización de la actividad física realizada por el usuario mediante servicios ejecutados en segundo plano y dos APIs proporcionadas por Android: *Activity Recognition Client* y *Activity Recognition Result* (Tarea 11, iteración 5.2.6).
  3. Contabilizar el número de pasos dados por el usuario con el sensor del acelerómetro incorporado en el dispositivo móvil (Tarea 12, iteración 5.2.6).

### Almacenamiento de la información en bases de datos

La información producida por las acciones y eventos capturados de los dispositivos móviles se han almacenado en bruto en distintas tablas de una base de datos proporcionada por Hostinger. Estas tablas se han ido explicando en cada iteración correspondiente junto con su estructura.

### Diseño e implementación de técnicas de visualización de información que muestre los hábitos y comportamientos

1. Implementación de una *API REST* (5.2.4) que permite establecer una conexión con la base de datos para acceder a la información y procesarla, a fin de obtener un conjunto de métricas que se deben representar en la aplicación web con el propósito de ser analizados por el usuario.
2. Diseño de una aplicación móvil (Tarea 12, iteración 5.2.6) que le permite al usuario registrarse, iniciar sesión y cerrar sesión en la aplicación (Tareas 1, 2 y 3, iteración 5.2.2), obtener información sobre la aplicación y el identificador de usuario único que se le asignado, borrar todos los datos registrados (Tarea 21, iteración 5.2.11), iniciar o finalizar los servicios que permiten monitorizar las acciones y visualizar la información en una aplicación web.
3. Diseño de una aplicación web (5.2.4) integrada en la aplicación móvil para que el usuario pueda visualizar en forma de gráficos y tablas un conjunto de métricas obtenidas de los eventos detectados, acotados por tiempo y tipo. Además, se permite la posibilidad de poder visualizar esos datos en otros dispositivos como un ordenador.
4. Construcción de gráficos con la librería JavaScript *Chart.js* que ha permitido visualizar toda la información monitorizada en la aplicación web. (Tareas 7 y 8 iteración 5.2.4, Tarea 10 iteración 5.2.5, Tarea 14 iteración 5.2.7, Tarea 16 iteración 5.2.8, Tarea 19 iteración 5.2.9, Tarea 20 y 21 5.2.10)

Además, mediante la evaluación realizada en el Capítulo 5, se ha demostrado también la consecución del objetivo general y ha ayudado a hacer mejoras en el sistema.

## 6.2. COMPETENCIAS OBTENIDAS

En esta sección se muestran las distintas competencias que se dan por demostradas en la realización de este Trabajo de Fin de Grado correspondientes a la intensificación de Tecnologías de la Información cursada por la autora de este proyecto.

- *Capacidad para seleccionar, diseñar, desplegar, integrar, evaluar, construir, gestionar, explotar y mantener las tecnologías de hardware, software y redes, dentro de los paráme-*

***etros de coste y calidad adecuados.*** En las reuniones iniciales se ha realizado una planificación en un diagrama de Gantt del tiempo que se emplearía en cada iteración de la implementación del proyecto. Sucesivamente, se ha realizado un estudio para escoger las herramientas, tecnologías y lenguajes que mejor se adapten al desarrollo del proyecto garantizando la calidad y obteniendo un coste del proyecto de 0 euros, exceptuando los costes de personal y equipos utilizados.

- ***Capacidad para emplear metodologías centradas en el usuario y la organización para el desarrollo, evaluación y gestión de aplicaciones y sistemas basados en tecnologías de la información que aseguren la accesibilidad, ergonomía y usabilidad de los sistemas.*** Esto se ha conseguido con la aplicación de metodologías ágiles que permiten un desarrollo iterativo e incremental y que han permitido involucrar al cliente en todas las fases de desarrollo mediante la obtención de *feedback* y la utilización de la herramienta Monday que le ha permitido observar el estado de desarrollo de las tareas del sistema.
- ***Capacidad de concebir sistemas, aplicaciones y servicios basados en tecnologías de red, incluyendo Internet, web, comercio electrónico, multimedia, servicios interactivos y computación móvil.*** En este proyecto se ha proporcionado un alojamiento de Hostinger para almacenar en ese servidor la base de datos, los *scripts* utilizados por la aplicación móvil que permiten introducir datos en la base de datos, la API REST y el *front-end* de la aplicación web. Desde la aplicación móvil se han realizado peticiones HTTP para insertar los datos en la base de datos y desde la aplicación web se han implementado peticiones HTTP para acceder a esta API. Por otro lado, se ha mejorado el posicionamiento del sistema, implementando una página de error 404, indexando la aplicación web en el buscador Google mediante la herramienta *Google Search Console* y añadiendo palabras claves en las etiquetas de cada sección para conseguir posicionarla en los primeros resultados de la búsqueda.
- ***Capacidad para comprender, aplicar y gestionar la garantía y seguridad de los sistemas informáticos.*** Esto se ha conseguido mediante el cifrado del identificador del usuario en la base de datos y en las peticiones de la API REST evitando en todo momento el envío del identificador en la URL de la aplicación web. Además, garantizar la seguridad y fiabilidad del usuario añadiendo la opción de resetear los datos de la aplicación.

### 6.3. TRABAJO FUTURO

Como continuación a este proyecto y una vez que ha sido probado por varios usuarios, se proponen diferentes mejoras y trabajos futuros que se podrán llevar a cabo, estas son:

- **Mejoras de código y optimización.** Es importante que antes de desarrollar nuevas funcionalidades en el sistema, se realicen pruebas intensivas para mejorar la calidad del código. En concreto, mejorar algunas consultas a base de datos y refactorizar algunos métodos de la aplicación móvil.
- **Monitorizar eventos táctiles.** Un aspecto importante que nos permite recoger distintas métricas sobre el uso que hacen los usuarios del dispositivo móvil es la detección del pixel pulsado, gestos realizados sobre la pantalla y el movimiento scroll. Detectando estos eventos el sistema ayudaría a los diseñadores y desarrolladores a estudiar y analizar la interacción de los

usuarios con las pantallas táctiles para mejorar la creación de interfaces.

- **Diferenciar entre tiempo activo o pasivo del uso de un dispositivo móvil.** La posible mejora mencionada anteriormente abre la puerta a la oportunidad de poder diferenciar si un usuario interacciona con el dispositivo de forma activa o pasiva, es decir, poder diferenciar si el usuario realiza algún movimiento táctil mientras está utilizando el dispositivo o si por el contrario está visualizando vídeos o realizando llamadas telefónicas, con el objetivo de recoger más información sobre la interacción de los usuarios con los dispositivos móviles.
- **Diferenciación de roles.** Añadir un rol de administrador que tenga la oportunidad de visualizar los datos de cada usuario, con el objetivo de realizar análisis en profundidad de la interacción del usuario con el dispositivo.
- **Mejoras de usabilidad en el sistema.** Durante todo el desarrollo del proyecto, se ha realizado un sistema fácil y sencillo de usar por cualquier tipo de usuario, indicando los pasos que se deben seguir en la sección de información de la aplicación y explicando el contexto en el que se enmarca. No obstante, siempre es posible añadir ciertas mejoras que puedan resultar útiles para el usuario como evitar que una vez registrado, tenga que copiar y pegar su identificador en la aplicación web para acceder a las estadísticas.
- **Generación de gráficos del tiempo de uso de aplicaciones agrupados por categorías y día de la semana.** Ofrecer gráficos que permitan comparar el tiempo de uso de las aplicaciones agrupadas en distintas categorías, tales como juegos, entretenimiento, redes sociales, aplicaciones del sistema etc. Por otro lado, ofrecer gráficos que permitan comparar el uso de las aplicaciones cada día de la semana, para analizar qué días se utiliza más el dispositivo móvil.

## 6.4. OPINIÓN PERSONAL

El desarrollo de este Trabajo de Fin de Grado ha supuesto en mi vida un gran reto profesional y personal. Este proyecto me ha permitido aplicar todos los conocimientos que he aprendido durante estos 4 años cursando el Grado de Ingeniería Informática de Ciudad Real y profundizar en aquellas tecnologías y fundamentos de la intensificación de Tecnologías de la Información en la que me he especializado estos dos últimos años, logrando así construir una solución íntegra guiada por una metodología ágil, que combina el desarrollo móvil y web.

## **ANEXOS**



---

## ANEXO A

---

# DIAGRAMAS DE CASOS DE USO Y PROTOTIPOS INICIALES

---

En este primer anexo se muestran todos los diagramas de casos de uso del sistema desarrollado y los prototipos iniciales del desarrollo de la aplicación móvil y de la aplicación web. Debido a que existe un gran número de casos de uso, se ha decidido agruparse según los datos que se manejan y las diferentes acciones realizadas por el usuario en la aplicación móvil y en la aplicación web.

### A.1. DIAGRAMA DE CASOS DE USO

En primer lugar se explicarán los diagramas de casos de uso realizados a partir de los requisitos obtenidos en la primera reunión.

#### A.1.1. Diagrama de casos de uso de la gestión de usuarios

En la Figura A.1 se muestra el diagrama de casos de uso que permite la gestión de los usuarios, donde se incluye el registro, inicio y cierre de sesión, junto con las opciones de obtener información de la aplicación que incluye la posibilidad de borrar toda la información registrada del usuario.

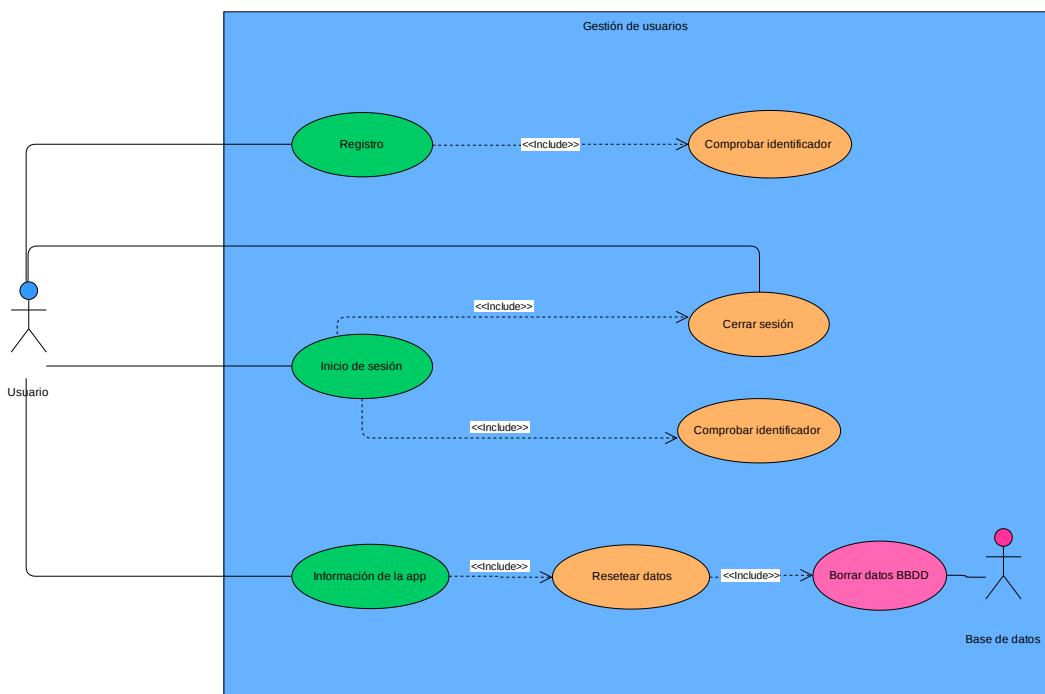
En la siguiente tabla se explica de forma detallada las funcionalidades del sistema representadas por un flujo de eventos en el primer diagrama de casos de uso, mostrando las condiciones de entrada y de salida necesarias para llevar a cabo estas funcionalidades y el escenario principal o alternativos posibles.

**Tabla A.1:** Especificación del primer caso de uso

<b>Nombre</b>	Gestión de usuarios
<b>Descripción</b>	Registro, inicio de sesión, cerrar sesión y obtener información de la aplicación
<b>Actor</b>	Usuario y base de datos
<b>Condiciones de entrada</b>	Obtención de un identificador de usuario único
<b>Condiciones de salida</b>	Monitorizar y visualizar los datos recogidos

<b>Escenario</b>	<ol style="list-style-type: none"> <li>1. El usuario se registra en la aplicación</li> <li>2. El sistema envía el identificador único del usuario</li> <li>3. El sistema comprueba que ese identificador no se encuentre en la base de datos y registra al usuario</li> <li>4. El usuario inicia sesión en la aplicación</li> <li>5. El sistema comprueba si el identificador se encuentra registrado en la base de datos</li> <li>6. El sistema proporciona información sobre el uso de la aplicación y muestra el identificador de usuario</li> <li>7. El usuario puede resetear los datos de la aplicación o cancelar la acción y volver a la actividad principal</li> </ol>
<b>Escenario alternativo 1</b>	<ol style="list-style-type: none"> <li>1. El usuario se registra en la aplicación</li> <li>2. El sistema envía el identificador único del usuario</li> <li>3. El sistema comprueba que ese identificador se encuentra en la base de datos y muestra un mensaje de error al usuario</li> </ol>
<b>Escenario alternativo 2</b>	<ol style="list-style-type: none"> <li>1. El usuario inicia sesión en la aplicación</li> <li>2. El sistema comprueba si el identificador se encuentra registrado en la base de datos</li> <li>3. El sistema comprueba que ese identificador no se encuentra registrado y muestra un mensaje de error al usuario</li> </ol>

Visual Paradigm Online Diagrams Express Edition

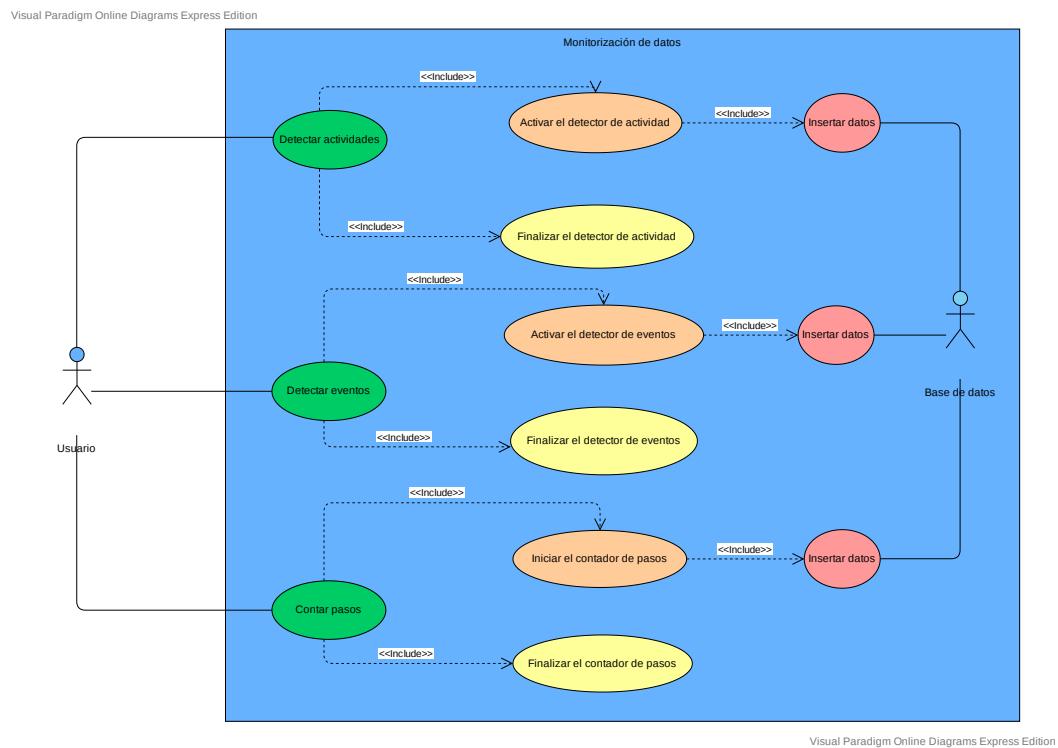


Visual Paradigm Online Diagrams Express Edition

**Figura A.1:** Diagrama de casos de uso gestión usuarios

### A.1.2. Diagrama de casos de uso de la monitorización de eventos

En la Figura A.2 se muestra el diagrama de casos de uso de las distintas opciones que permite al usuario iniciar o finalizar la monitorización de eventos.



**Figura A.2:** Diagrama de casos de uso monitorización de eventos

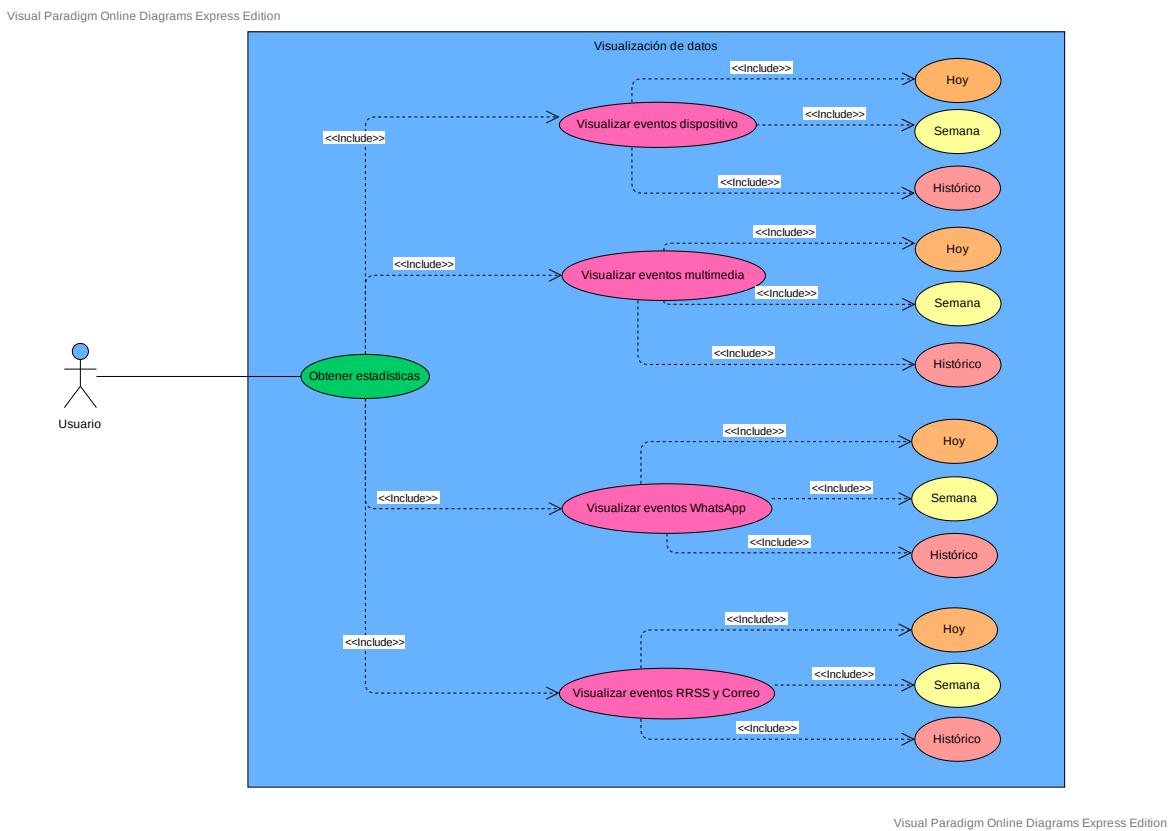
En la siguiente tabla se explica de forma detallada las funcionalidades del sistema representadas por el segundo diagrama de casos de uso, mostrando las condiciones de entrada y de salida necesarias para llevar a cabo estas funcionalidades y el escenario principal o alternativos posibles.

**Tabla A.2:** Especificación del segundo caso de uso

<b>Nombre</b>	Monitorización de eventos
<b>Descripción</b>	Detectar actividades físicas, eventos o contar pasos
<b>Actor</b>	Usuario y base de datos
<b>Condiciones de entrada</b>	Registro e inicio de sesión del usuario
<b>Condiciones de salida</b>	Visualizar los datos recogidos
<b>Escenario</b>	<ol style="list-style-type: none"> <li>El usuario inicia el detector de actividades físicas</li> <li>El usuario inicia el detector de eventos</li> <li>El usuario inicia el contador de pasos</li> </ol>
<b>Escenario alternativo 1</b>	<ol style="list-style-type: none"> <li>El usuario finaliza el detector de actividades físicas</li> <li>El usuario finaliza el detector de eventos</li> <li>El usuario finaliza el contador de pasos</li> </ol>

### A.1.3. Diagrama de casos de uso de la visualización de datos

En la Figura A.3 se muestra el diagrama de casos de uso de la visualización de la información monitorizada. Incluye los casos de uso de visualización de información relacionada con el dispositivo, aspectos multimedia como Google Fotos o actividades físicas, acciones realizadas en WhatsApp y en las redes sociales y el correo electrónico.



**Figura A.3:** Diagrama de casos de uso visualización de datos

En la siguiente tabla se explica de forma detallada cada una de las funcionalidades del sistema representadas por el tercer diagrama de casos de uso que permite la visualización de los datos, mostrando las condiciones de entrada y de salida necesarias para llevar a cabo estas funcionalidades y el escenario principal o alternativos posibles.

**Tabla A.3:** Especificación del tercer caso de uso

<b>Nombre</b>	Visualización de datos
<b>Descripción</b>	Visualizar todos los datos recogidos acotados por hoy, semana e histórico
<b>Actor</b>	Usuario y base de datos
<b>Condiciones de entrada</b>	Activación de las opciones que permiten la monitorización de datos
<b>Condiciones de salida</b>	Analizar los datos recogidos

<b>Escenario</b>	<ol style="list-style-type: none"> <li>1. El usuario selecciona la opción obtener estadísticas</li> <li>2. El usuario introduce su identificador de usuario en la aplicación web</li> <li>3. El sistema comprueba que el usuario esté registrado y muestra la página de inicio de la aplicación web.</li> <li>4. El usuario visualiza las pestañas hoy, semana o histórico de las secciones Dispositivo, Multimedia, WhatsApp, RRSS y correo</li> </ol>
<b>Escenario alternativo 1</b>	<ol style="list-style-type: none"> <li>1. El usuario accede en el navegador a la aplicación web</li> <li>2. El usuario introduce su identificador de usuario</li> <li>3. El sistema comprueba que el usuario no está registrado en la aplicación móvil y muestra un mensaje de error al usuario</li> </ol>

## A.2. PROTOTIPOS INICIALES

A continuación, se muestran los prototipos diseñados en la iteración inicial y que fueron entregados en la segunda reunión mantenida con el cliente.

### A.2.1. Prototipos aplicación móvil

En la Figura A.4 se muestra el diseño de la aplicación móvil formada por la actividad inicial que permite al usuario registrarse o iniciar sesión y la actividad principal que permite activar las distintas opciones para habilitar la recogida de datos.

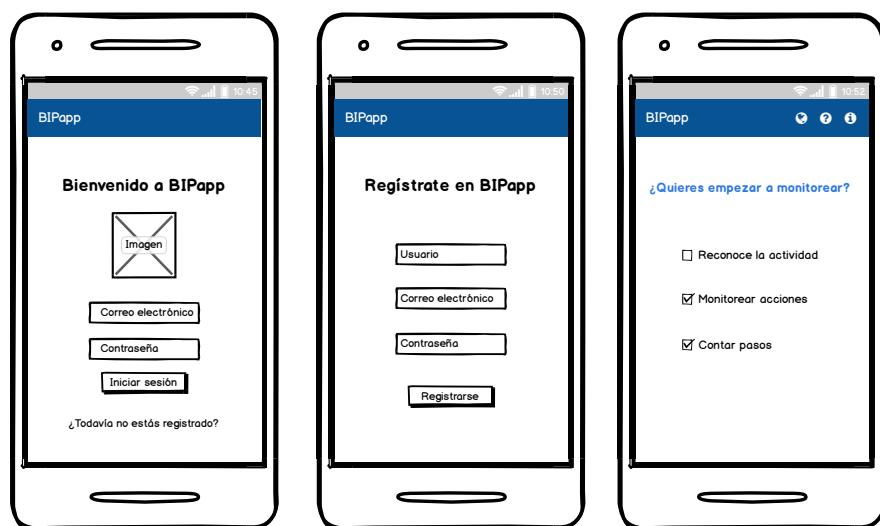


Figura A.4: Bocetos aplicación móvil

### A.2.2. Prototipos aplicación web

En las Figuras A.5 y A.6 se muestra el diseño de la aplicación web, con las distintas secciones que contendrán los gráficos con los datos recogidos, tanto para el diseño móvil como para el de un ordenador.

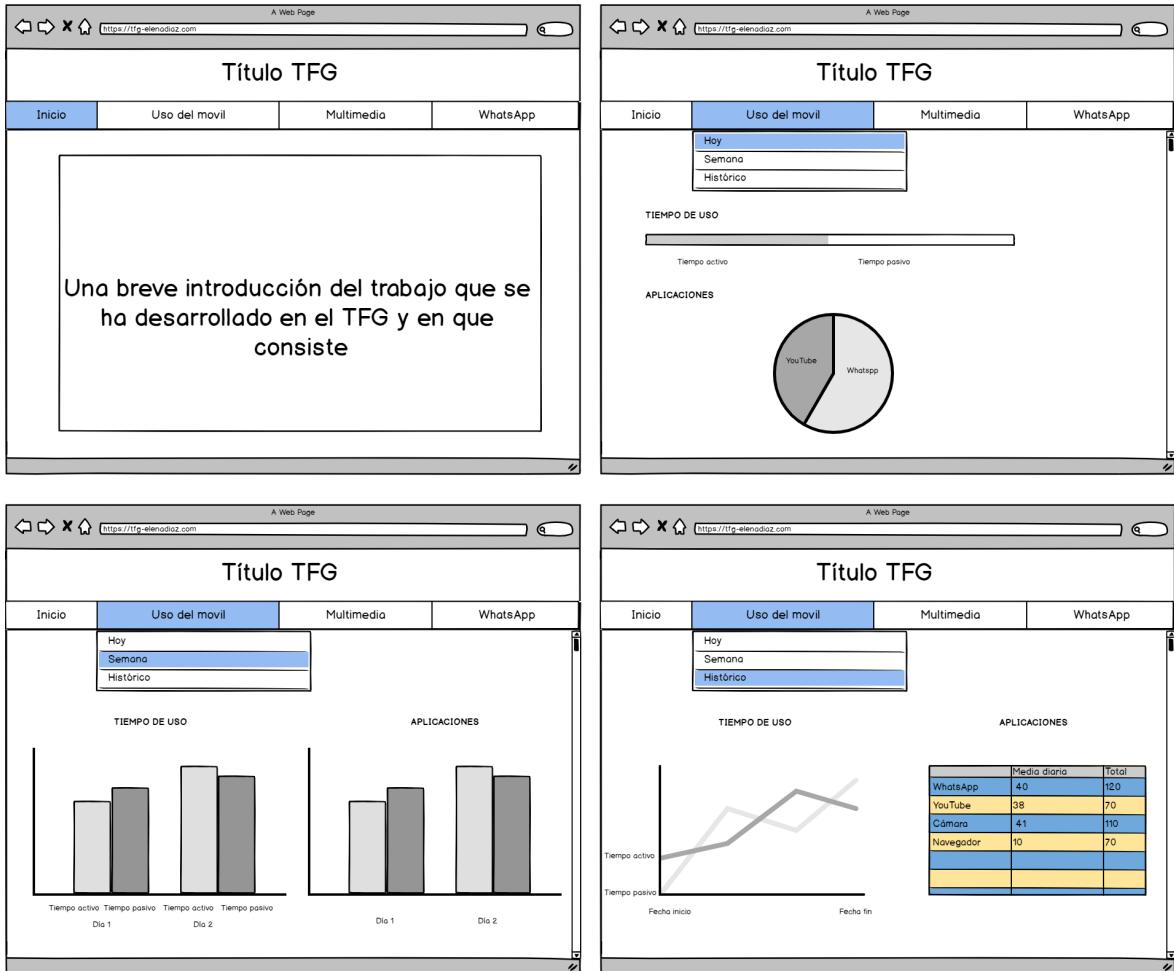


Figura A.5: Bocetos aplicación web ordenador

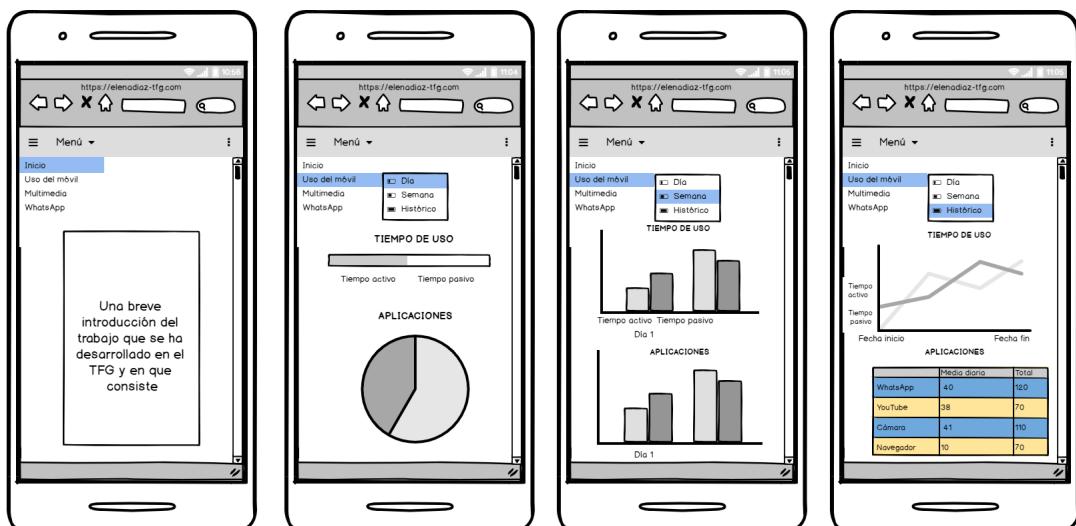


Figura A.6: Bocetos aplicación web móvil

---

## ANEXO B

---

# EVENTOS MONITORIZADOS Y MÉTRICAS OBTENIDAS

---

En la siguiente tabla se muestran los distintos eventos producidos en el sistema o en aplicaciones concretas que han sido monitorizados y han permitido obtener una serie de métricas para ser visualizadas. Cada una de las columnas que componen esta tabla, presenta el siguiente significado:

- **App.** Aplicación o parte del sistema en el que surge un determinado evento producido de la interacción del usuario con el dispositivo.
- **Evento(s).** Tipo de evento que se produce en esa app o parte del sistema.
- **Paquete.** Paquete que ha permitido detectar las distintas clases de eventos producidos en el sistema.
- **Métricas.** Información de interés que ha sido procesada a partir de los datos monitorizados y almacenados en la BBDD, para ser representado.
- **Vi.** Eventos y métricas que han sido visualizadas
- **Mv.** Eventos que han sido monitorizados

**Tabla B.1:** Tabla de eventos y métricas recogidas de cada aplicación

Métricas					
App	Evento(s)	Paquete	Métricas	Vi	Mv
Sistema	Aplicación o servicio en primer plano	TYPE_WINDOW  _STATE_CHANGED	Tiempo de uso de cada aplicación en el día y semana actual, y en el histórico	Si	Si
			Comparativa del uso de las aplicaciones detectadas en primer plano	Si	No
			Cambios de una aplicación a otra	No	-
Sistema	Notificación recibida	TYPE_NOTIFICATION  _STATE_CHANGED y android.app.Notification	Número total de notificaciones recibidas en el día y semana actual, y en el histórico	Si	No

Métricas						
App	Evento(s)	Paquete	Métricas	Vi	Mv	
Sistema	Aplicaciones instaladas	com.android.packageinstaller.UninstallerActivity y com.android.vending	Número de aplicaciones instaladas	No	-	
			Frecuencia de cambio de aplicaciones (instalar o borrar) en el día y semana actual, y en el histórico	Si	Si	
			Nombre y paquete de las aplicaciones instaladas	No	Si	
Sistema	Bloqueo o desbloquear pantalla	Intent.ACTION_SCREEN_OFF e Intent.ACTION_SCREEN_ON	Fallos en el desbloqueo de la pantalla	No	-	
			Frecuencia de bloqueo y desbloqueo de pantalla en el día y semana actual, y en el histórico	Si	Si	
			Tiempo acumulado con el móvil bloqueado y desbloqueado en el día y semana actual, y en el histórico	Si	Si	
Sistema	Subir o bajar volumen	KeyEvent.KEYCODE_VOLUME_UP y KeyEvent.KEYCODE_VOLUME_DOWN	Frecuencia de cambio de volumen en el día y semana actual, y en el histórico	Si	Si	
Sistema	Llamadas	TYPE_WINDOW_STATE	Número de llamadas realizadas en el dispositivo junto con el tiempo total empleado, en el día y semana actual, y en el histórico	Si	Si	
Sistema	Cámara	com.android.camera	Número de usos de la cámara del dispositivo en el día y semana actual, y en el histórico	No	Si	
WhatsApp	Visualización de imágenes o videos	com.whatsapp.mediaview.MediaViewActivity	Tiempo empleado visualizando imágenes o videos en el día y semana actual y en el histórico	Si	Si	
WhatsApp	Reproducción de audios	com.whatsapp.mediaview.AudioViewActivity	Tiempo empleado escuchando audios en el día y semana actual y en el histórico	Si	Si	
WhatsApp	Otros datos	com.whatsapp.Conversation	Número de conversaciones abiertas y nº de notificaciones recibidas en el día y en la semana actual y en el histórico	Si	Si	

Métricas					
App	Evento(s)	Paquete	Métricas	Vi	Mv
WhatsApp	Envío de elementos	com.whatsapp	Número de imágenes, documentos, contactos, archivos de música o ubicaciones enviadas en el día y en la semana actual, y en el histórico.	Si	Si
WhatsApp	Videollamadas	com.whatsapp	Calcular el número de videollamadas, junto con la duración total de las mismas, realizadas en el día y la semana actual, y en el histórico	Si	Si
WhatsApp	Llamadas	com.whatsapp	Calcular el número de llamadas, junto con la duración total de las mismas, realizadas en el día y la semana actual, y en el histórico	Si	Si
Google Fotos	Visualización de vídeos o imágenes	com.google.android.apps.photos	Calcular el número de imágenes y vídeos visualizados en Google Fotos, junto con el tiempo empleado visualizando dichas imágenes y vídeos en el día y en la semana actual, y en el histórico	Si	Si
Gmail	Correos enviados o eliminados, notificaciones y tiempo de uso	com.google.android.gm	Número de correos enviados o eliminados y notificaciones, junto con el tiempo de uso de esta aplicación en el día y en la semana actual, y en el histórico	Si	Si
Outlook	Correos enviados, notificaciones y tiempo de uso	com.microsoft.office.outlook	Número de correos enviados y notificaciones, junto con el tiempo de uso de esta aplicación en el día y en la semana actual, y en el histórico	Si	Si
Instagram	Notificaciones, me gusta, comentarios escritos o visualizados, publicaciones compartidas y tiempo de uso	com.instagram.android	Número de notificaciones recibidas, me gustas en publicaciones, comentarios escritos o visualizados en publicaciones y publicaciones compartidas junto con el tiempo de uso de esta app en el día y en la semana actual, y en el histórico	Si	Si

<b>Métricas</b>					
<b>App</b>	<b>Evento(s)</b>	<b>Paquete</b>	<b>Métricas</b>	<b>Vi</b>	<b>Mv</b>
Facebook	Notificaciones, me gusta, comentarios escritos, publicaciones compartidas y tiempo de uso	com.facebook.katana	Nº de notificaciones recibidas, me gustas y comentarios escritos en publicaciones, publicaciones compartidas y tiempo de uso de la app en el día y en la semana actual, y en el histórico	Si	Si
Twitter	Tweets compartidos, respondidos, escritos u eliminados, notificaciones y tiempo de uso	com.twitter.android	Nº de notificaciones recibidas, tweets compartidos, respondidos, escritos u eliminados, junto con el tiempo de uso de esta app en el día y en la semana actual, y en el histórico	Si	Si
Linkedin	Publicaciones escritas, eliminadas o recomendadas, comentarios, notificaciones y tiempo de uso	com.linkedin.android	Nº de notificaciones recibidas, publicaciones escritas, eliminadas o recomendadas, comentarios realizados en publicaciones y tiempo de uso de esta app en el día y en la semana actual, y en el histórico	Si	Si
Teclado	Letras, símbolos y emoticonos escritos	com.widget.edittext	Número de letras, símbolos y emoticonos escritos y eliminados, frecuencia de pulsación de letras por segundo y tiempo total empleado escribiendo, en el día y semana actual y en el histórico	Si	Si
Sensor	Paso dado por el usuario	Sensor del acelerómetro del dispositivo	Nº de pasos dados por el usuario en el día y en la semana actual, y en el histórico	Si	Si
Google APIs	Actividad física del usuario	APIs Activity Recognition Client y Activity Recognition Result	Calcular el tiempo de actividad realizada por el usuario, diferenciando entre andar, correr o parado, mientras realiza distintos tipos de eventos con el dispositivo como llamar por teléfono o escribir letras.	Si	Si

---

## ANEXO C

# CONFIGURACIÓN SERVICIO DE ACCESIBILIDAD Y API REST SLIM

---

## C.1. CONFIGURACIÓN SERVICIO DE ACCESIBILIDAD

Según se explicaba en la iteración 5.2.3 existen dos formas posibles de configurar el servicio de accesibilidad. La primera forma consistía en configurar el servicio para que haga uso de un archivo meta-data (Listado C.1) donde se configuren los distintos permisos necesarios para recoger el contenido de una ventana, filtrar eventos, permitir la detección de botones físicos etc., como se muestra en el Listado C.2.

Listado C.1: Configuración del servicio de accesibilidad

```
1 <service
2   <meta-data
3     android:name="android.accessibilityservice"
4     android:resource="@xml/my_key_service"
5   />
6 </service>
```

Listado C.2: Configuración del recurso XML

```
1 <accessibility-service
2   android:accessibilityEventTypes="typeContextClicked | ↴
3     ↪ typeViewClicked | typeAllMask | typeViewFocused"
4   android:accessibilityFlags="flagRequestFilterKeyEvents |
5     ↪ flagRequestTouchExplorationMode"
6   android:accessibilityFeedbackType="feedbackAllMask | ↴
7     ↪ feedbackGeneric | feedbackSpoken"
8   android:canRetrieveWindowContent="true"
9   android:canRequestFilterKeyEvents="true"
10  android:canRequestTouchExplorationMode="true"
11  android:packageName="com.example.andres.eventcapture"
12 />
```

## C.2. CONFIGURACIÓN API REST CON SLIM

La primera tarea fue construir la carpeta del proyecto para instalar en el directorio raíz del proyecto el *framework* Slim y sus dependencias en el directorio denominado *vendor/* del proyecto mediante el comando que se indica a continuación.

```
composer require slim/slim:"4.*"
```

Sucesivamente se creó otro directorio denominado *public* que contenía un archivo llamado *index.php*, que permitía configurar la API e incluir todas las rutas a los archivos necesarios como se muestra en el Listado C.3.

**Listado C.3:** Configuración API REST Slim

```
1 <?php
2 use Psr\Http\Message\ResponseInterface as Response;
3 use Psr\Http\Message\ServerRequestInterface as Request;
4
5 require '../vendor/autoload.php';
6 require '../src/config/db.php';
7
8 $config = ["settings" => [ "displayErrorDetails" => true]];
9
10 $app = new Slim\App();
11
12 // Incluir rutas a archivos de manipulación de la información
13 require '../src/rutas/infor.php';
14
15 $app->run();
```

En el directorio *apiRest/src/config* se creó un archivo denominado *db.php* que permitía establecer una conexión con la base de datos (Listado C.4) y que será utilizada en cada *endpoint*.

**Listado C.4:** Conexión BBDD

```
1 <?php
2 $config = ["settings" => [ "displayErrorDetails" => true]];
3
4 class db{
5
6     public function conectDB() {
7
8         $conexion=mysqli_connect("localhost", ↵
9             "id13191673_id2_elenadiaz_tfg", "Monitorizacion/2020", ↵
10            "id13191673_id1_elenadiaz_tfg");
11
12     return $conexion;
13 }
```

Por último, en el directorio *apiRest/src/config/rutas* se crearon distintos archivos similares al que se muestra en el Listado C.5 que permiten crear distintos *endpoints* para acceder y manipular la información para obtener las métricas requeridas para ser representadas.

**Listado C.5:** Ejemplo de petición HTTP que calcula el tiempo de uso de las 10 aplicaciones más usadas

```
1 <?php
2 use Psr\Http\Message\ResponseInterface as Response;
3 use Psr\Http\Message\ServerRequestInterface as Request;
4
5 $app -> get('/api/tiempoApps/{name}', function(Request $request, ←
6     Response $response, array $args) {
7
8     $name = $args['name'];
9     $conexion = new db();
10    $conexion = $conexion->conectDB();
11
12    $sql = "SELECT t1.Paquete_App, SUM((t2.Hora-t1.Hora)) Tiempo FROM ←
13        primerPlano t1, primerPlano t2 WHERE (t2.id=t1.id+1) && ←
14        t2.Hora-t1.Hora>3 GROUP BY t1.Paquete ORDER BY ←
15        SUM((t2.Hora-t1.Hora)) DESC LIMIT 10";
16
17    mysqli_set_charset($conexion, "utf8");
18    $result=mysqli_query($conexion, $sql);
19
20    $clientes = array();
21
22    while($row = mysqli_fetch_array($result)) {
23        $TotalI=$row['App'];
24        $Tiempo=$row['Tiempo'];
25        $clientes[] = array('App'=> $TotalI, 'Tiempo' => $Tiempo);
26    }
27
28    $close = mysqli_close($conexion) or die("Ha sucedido un error ←
29        inesperado en la desconexion de la base de datos");
30
31    $json_string = json_encode($clientes);
32    echo $json_string;
33
34});
```



---

## ANEXO D

# CARACTERÍSTICAS PRINCIPALES LIBRERÍAS DE GRÁFICOS JS

---

Como se mencionaba en la iteración 5.2.4 se realizó la tabla comparativa 5.3 a partir del estudio de las distintas características que se muestran a continuación, ofrecidas por cinco librerías populares para la elaboración de gráficos con JavaScript.

- **Chart JS.** Chart JS es una librería JavaScript popular debido a que sus componentes de visualización de datos tienen un buen rendimiento en distintos navegadores web y en distintas resoluciones haciendo uso de *canvas* de HTML5. Es una biblioteca sencilla, potente y flexible para desarrollar gráficos. Entre sus principales características se encuentran:
  - Librería gratuita para todo tipo de uso y dispone únicamente de 8 tipos de gráficos: lineal, área, barra, circular, polar, burbuja y dispersión.
  - Los componentes de visualización de datos son personalizables y animados.
  - Existencia de una documentación técnica y ejemplos a seguir por una amplia comunidad de programadores.
  - Soporte en múltiples navegadores web.
  - Es modular, ya que las funcionalidades se presentan separadas y es posible hacer uso de *plugins* para aumentar las funcionalidades.
- **D3 JS**<sup>1</sup>. D3 JS es una librería, aunque más bien se trata de un *framework*, para representar datos de forma potente.
  - Librería gratuita para todo tipo de uso, pero más difícil de usar que el resto de librerías, aunque se obtienen mejores resultados.
  - Dispone de muchos tipos de gráficos en comparación con otras librerías y ofrece la posibilidad de generar curvas mediante el uso de diversas funciones.
  - Existencia de un gran número de ejemplos de su uso en la red.
  - Combina los componentes de visualización potentes SVG, CSS y HTML5, con un enfoque basado en los datos para la manipulación de DOM.
  - Tiene funcionalidad *Drag & Drop*.
  - Es fácil de depurar con el inspector de elementos del navegador.

---

<sup>1</sup><https://d3js.org/>

- **Google Charts**<sup>2</sup>. Google Charts es una librería que ofrece herramientas potentes para la construcción de gráficos interactivos y fáciles de personalizar. Presenta numerosas ventajas, como pueden ser:
  - La librería no es de código libre y la versión gratuita ofrece funciones limitadas salvo para las organizaciones sin fines de lucro.
  - Tiene una gran cantidad de gráficos y los elementos gráficos se basan en el uso de HTML5, SVG y VML y permite generar gráficos a partir de hojas de cálculo, bases de datos SQL o archivos CSV.
  - Presenta la posibilidad de conectar los datos en tiempo real utilizando protocolos de conexión y ofrece una amplia documentación sobre su uso.
  - Es compatible con distintos navegadores y portabilidad multiplataforma para iOS y Android.
  - Los archivos de Google deben ser incrustados en la aplicación web.
- **Highcharts**<sup>3</sup>. Highcharts es una biblioteca que ofrece una gran diversidad de tipos de gráficos con un alto rendimiento. Es una librería completa y ligera. Presenta las siguientes características:
  - La librería no es gratuita, salvo para organizaciones sin ánimo de lucro.
  - Los elementos gráficos se procesan en SVG o VML basado en HTML5.
  - Es multiplataforma, usada por grandes firmas tecnológicas y presenta gráficos interactivos.
  - Los datos se pueden cargar al gráfico de forma externa al documento HTML.
  - Los gráficos pueden ser descargados en formato PDF, PNG, JPG o SVG
- **Chartist JS**<sup>4</sup>. Chartist JS es una biblioteca utilizada para cualquier tipo de uso. Es fácil de usar de implementar y la visualización de los datos se obtiene a partir de gráficos simples y *responsive*. Las principales características que presenta son:
  - Librería gratuita para todo tipo de uso y presenta una amplia documentación para su uso.
  - Dispone únicamente de tres tipos de gráficos: lineales, de barras y circulares y se construyen con el uso de SVG.
  - Es compatible con navegadores antiguos, pero no todos soportan sus animaciones.
  - No depende de terceras librerías, es fácil de configurar y permite usar *plugins* para aumentar las funcionalidades.

---

<sup>2</sup><https://developers.google.com/chart/>

<sup>3</sup><https://www.highcharts.com/>

<sup>4</sup><https://gionkunz.github.io/chartist-js/>

---

## ANEXO E

---

# ESQUEMA BASE DE DATOS

---

En este anexo, se muestra el esquema del conjunto de tablas que componen la base de datos del sistema y que se han ido explicando a lo largo de la memoria.

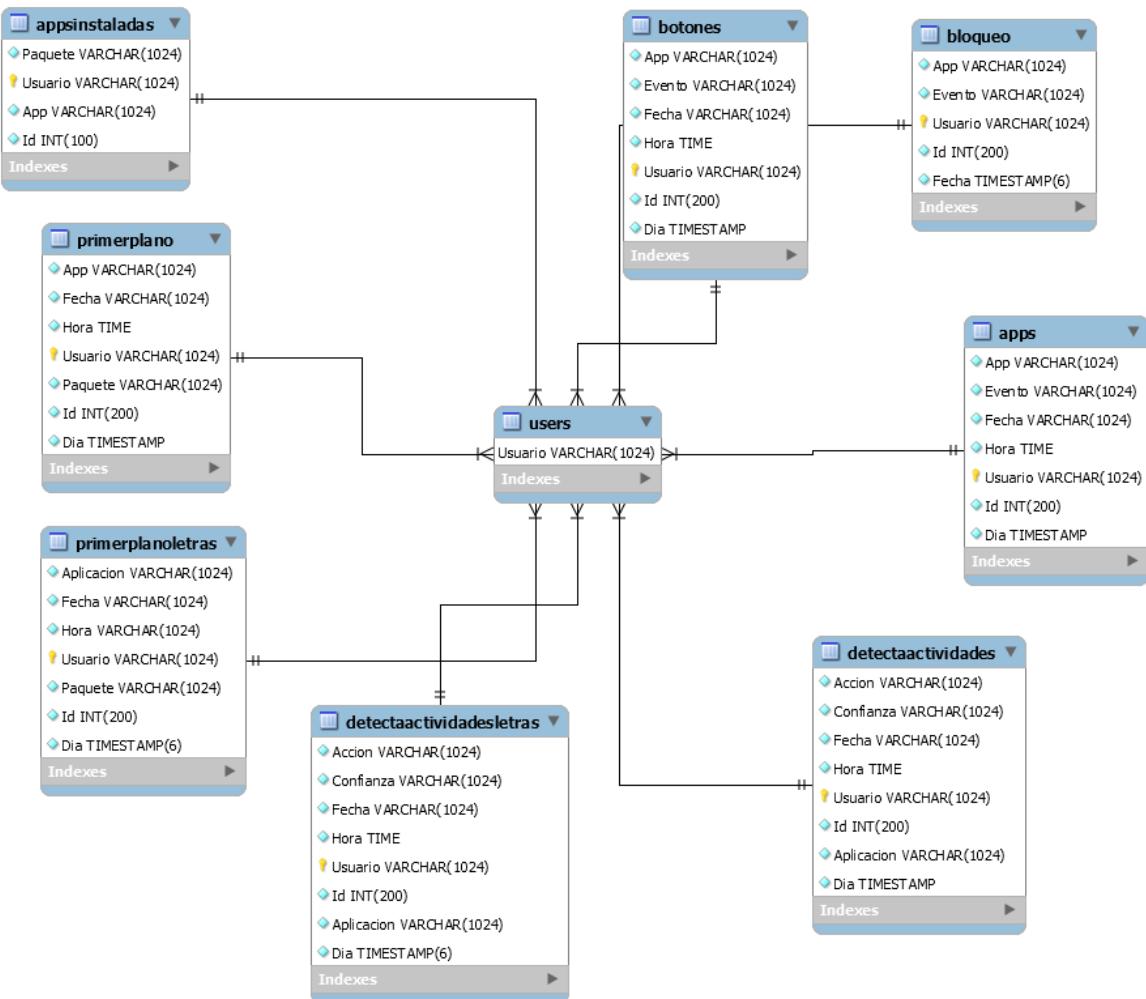


Figura E.1: Esquema relacional de la base de datos



---

## ANEXO F

# GENERACIÓN DE INFORMES Y POSICIONAMIENTO

---

- Página de error 404 de la aplicación web

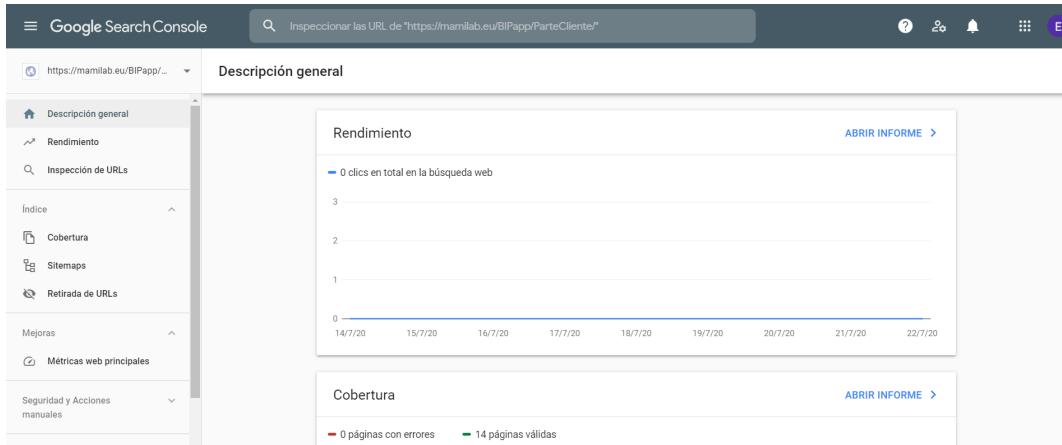


Figura F.1: Página de error 404 ordenador



Figura F.2: Página de error 404 móvil

■ Indexación de la aplicación web en Google Search Console



**Figura F.3:** Indexación de la app web en Google Search Console

- Ejemplo de etiquetas y palabras clave en cada sección de la aplicación web para mejorar el posicionamiento y colocarla en los primeros resultados de la búsqueda.

**Listado F.1:** Posicionamiento de la aplicación web

```

1 <head>
2   <title>BIPapp. Monitorización de la interacción con el dispositivo ←
3     ↪ móvil</title>
4   <meta name="author" content="Elena Diaz del Campo Gonzalez-Gallego">
5   <meta name="description" content="BIPapp. En esta sección podrás ←
6     ↪ observar el uso de la aplicación WhatsApp en el día actual">
7   <meta name="title" content="Observa el uso del WhatsApp en el día ←
8     ↪ actual con BIPapp">
9   <meta name="keywords" content="BIPapp, monitorización, ←
10    ↪ monitorizar, monitorear, uso, uso del móvil, monitoreo, ←
11    ↪ comportamiento, patrones de comportamiento, dispositivo, ←
12    ↪ dispositivo móvil, multimedia, whatsapp, cámara, videos, ←
13    ↪ WhatsApp, WhatsApp, google, google fotos, vicio, botones, ←
14    ↪ adicción, adicción, multimedia, videos, letras, simbolos, ←
15    ↪ emoticonos, llamadas, videollamadas, aplicación, hoy">
16 </head>

```

- Código implementado para la generación de informes con la librería *jsPDF*.

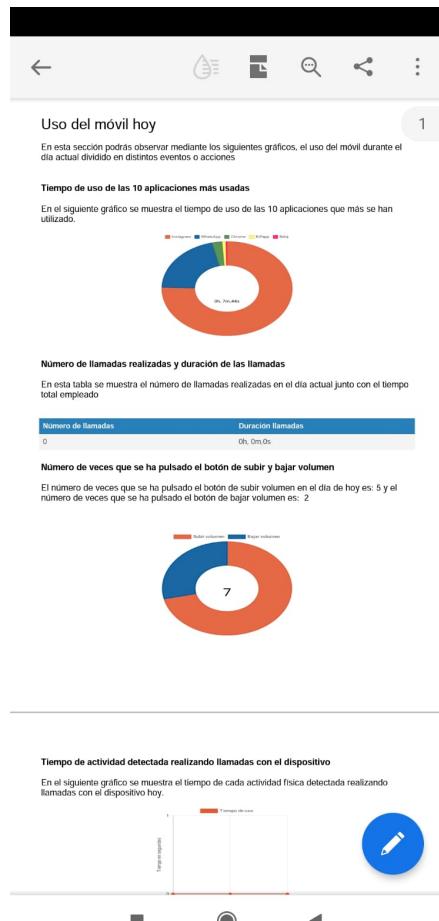
**Listado F.2:** Generación de informes PDF

```

1 <script>
2
3     function downloadPDF() {
4
5         var canvas1 = document.querySelector('#graficoVolumen');
6             var canvasImg1 = canvas1.toDataURL("image/png", 1.0);
7         var doc = new jsPDF('p');
8
9         doc.text(15, 15, "Uso del móvil hoy");
10        doc.text(15, 55, "Número de veces que se ha pulsado el botón de ←
11            ↪ subir y bajar volumen");
12        doc.addImage(canvasImg1, 'JPEG', 15, 70, 180, 50 );
13        doc.setProperties({ title: "Informe diario" });
14        var blob = doc.output("blob");
15            window.open(URL.createObjectURL(blob));
16
17    </script>

```

- Ejemplo de informe generado con la librería *jsPDF*.



**Figura F.4:** Ejemplo de informe PDF



---

## ANEXO G

# APLICACIONES EXISTENTES EN EL MERCADO

---

Como se mencionaba en el Capítulo 3 se realizaron dos tablas comparativas de aplicaciones existentes en el mercado que permiten monitorizar la actividad física del usuario y los eventos producidos de la interacción con los dispositivos móviles, a partir del estudio de las distintas características, funcionalidades y objetivos que presentan y que se enumeran a continuación.

### G.1. ACTIVIDAD FÍSICA

En primer lugar, se realizó un estudio de distintas aplicaciones que permiten la monitorización de actividades físicas.

- **Adidas Running by Runtastic<sup>1</sup>.** Adidas Running es una aplicación disponible para Android e iOS que permite registrar todas las actividades realizadas por el usuario y analizar las estadísticas. Proporciona la distancia recorrida, la duración o el tiempo transcurrido en cada actividad y la velocidad o la altitud entre otros datos relevantes.

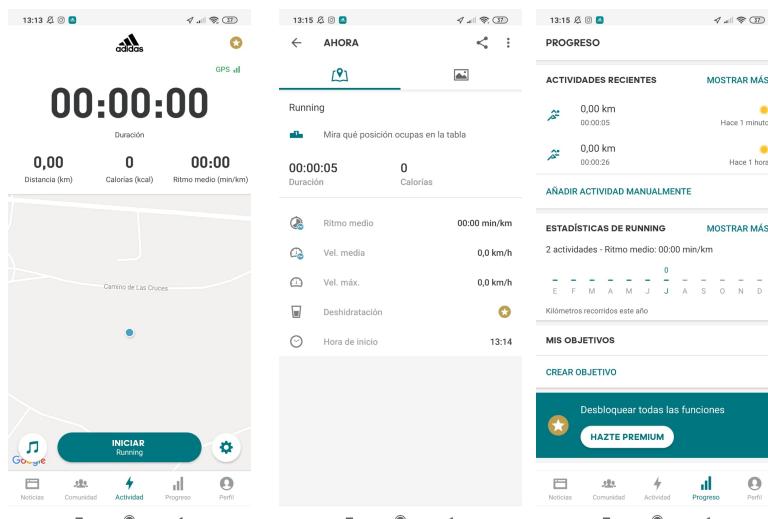
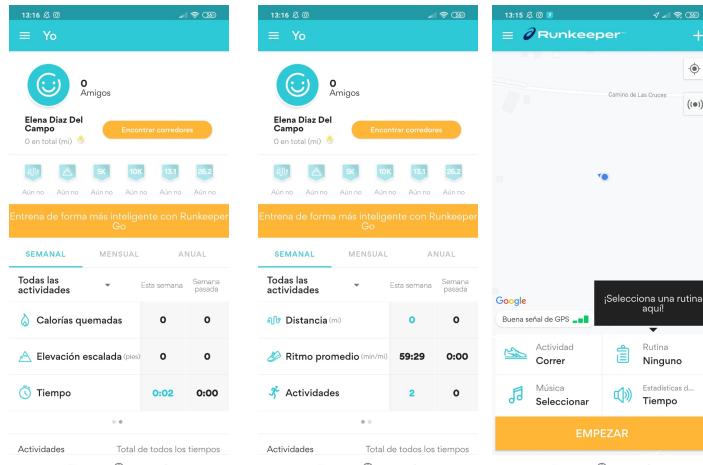


Figura G.1: Aplicación Adidas Running by Runtastic

---

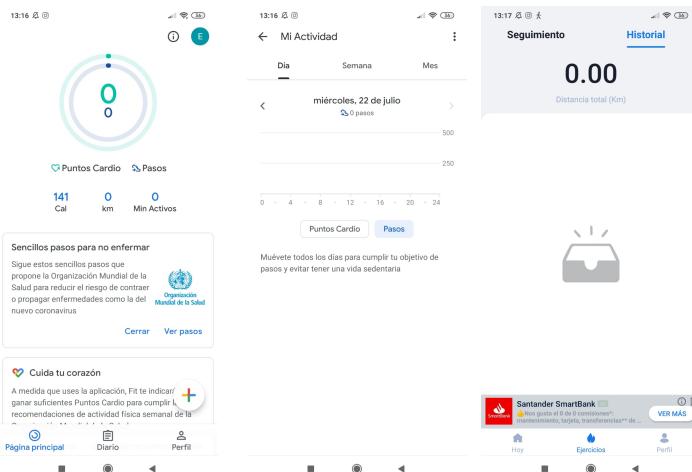
<sup>1</sup><https://www.runtastic.com/es/>

- **RunKeeper<sup>2</sup>**. RunKeeper es una aplicación disponible tanto para sistemas operativos Android como iOS. Al igual que Adidas Running u otras aplicaciones similares como Nike Training Club, permite monitorizar las distintas actividades deportivas o ver estadísticas sobre la distancia recorrida y la duración, además de proponerte metas y objetivos.



**Figura G.2:** Aplicación RunKeeper

- **Google Fit<sup>3</sup>**. Google Fit no solo ayuda a mejorar el estado de salud sino que permite consultar estadísticas del conteo de pasos y de la detección de acciones como correr, ir en bicicleta o andar, usando los sensores del dispositivo. Está sincronizado con otras aplicaciones como pulseras Xiaomi o con la aplicación mencionada anteriormente, para mostrar información sobre el estado de salud y poder analizar el progreso.



**Figura G.3:** Aplicación Google Fit

- **Seguimiento de pasos – Podómetro<sup>4</sup>**. Esta aplicación disponible para dispositivos Android, permite hacer un seguimiento diario del número de pasos dados y otros datos como la distancia recorrida, duración, ritmo, datos de salud, calorías etc., con un diseño sencillo que proporciona informes en forma de gráficos.

<sup>2</sup><https://runkeeper.com/>

<sup>3</sup><https://www.google.com/fit/>

<sup>4</sup><https://play.google.com/store/apps/details?id=steptracker.healthandfitness.walkingtracker.pedometer&hl=es>

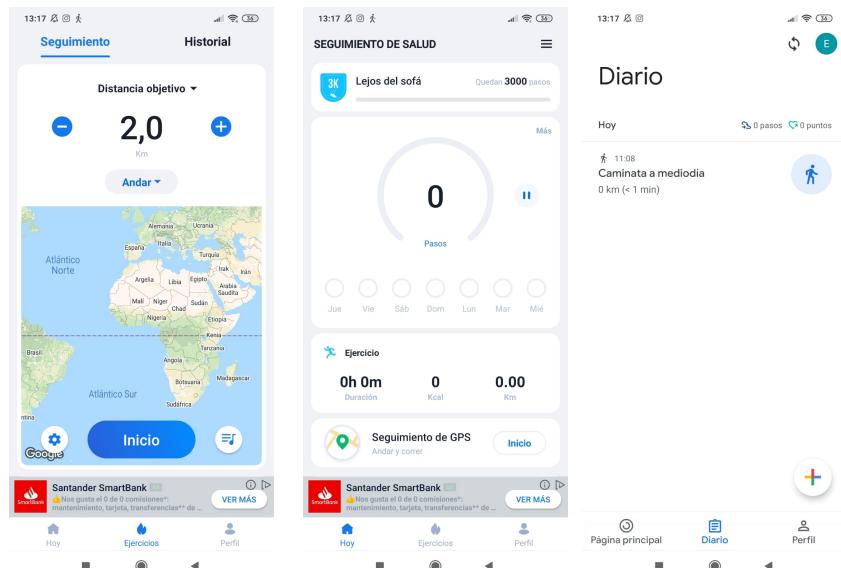


Figura G.4: Aplicación seguimiento de pasos - podómetro

## G.2. INTERACCIÓN CON EL DISPOSITIVO

Sucesivamente, se realizó un estudio de distintas aplicaciones que permiten la monitorización de eventos y acciones producidos por la interacción de los usuarios con los dispositivos móviles.

- **Analizador de uso<sup>5</sup>**. Esta aplicación nos permite analizar el tiempo que pasamos usando una determinada aplicación y gestionar ese tiempo. La información se estructura en días, semanas, meses e histórico mediante gráficas y estadísticas.

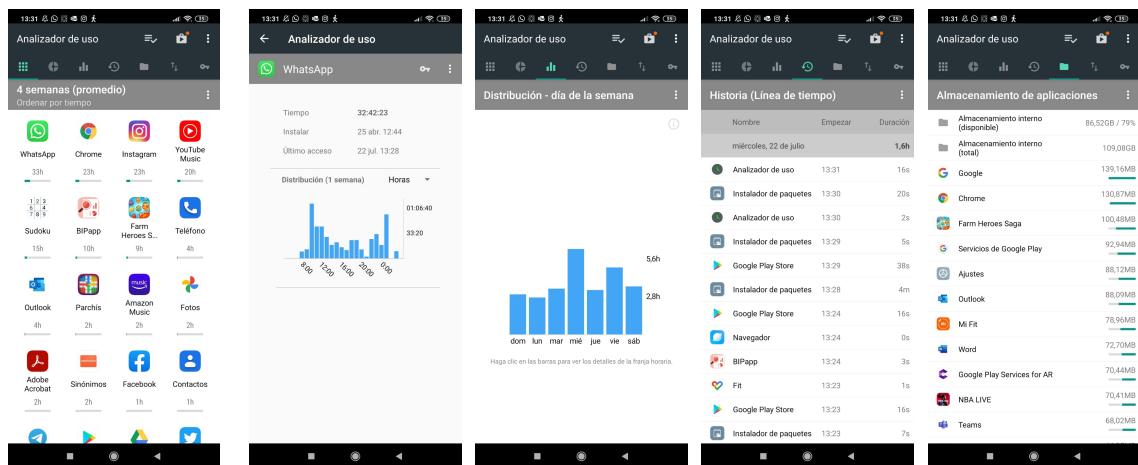


Figura G.5: Aplicación analizador de uso

- **Your Hour<sup>6</sup>**. Esta app ofrece varias funciones para rastrear y administrar el uso que hacemos de los dispositivos móviles e incluso la categoría de adicción al teléfono a la que perteneces. En el panel de control se puede obtener información sobre el uso del teléfono durante todo

<sup>5</sup><https://play.google.com/store/apps/details?id=info.kfsoft.usageanalyzer&hl=es>

<sup>6</sup><https://www.yourhour.app/>

el día e incluye el tiempo que ha estado activada la pantalla y el número de desbloqueos que se han realizado. Después realiza una comparativa de los últimos 7 días. Si seleccionas una aplicación se puede ver de forma individual el tiempo que se ha estado usando a lo largo del día y el tiempo dedicado fuera del uso que se ha establecido. También muestra una línea de tiempo sobre lo que ha estado sucediendo en el dispositivo móvil a lo largo de un día, es decir, el tiempo que usaste esa aplicación, la app a la que cambiaste sucesivamente etc.

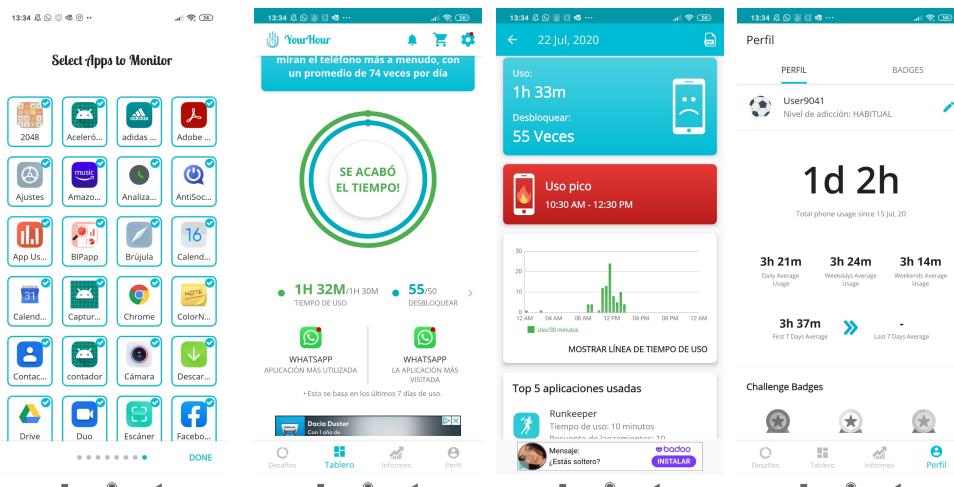


Figura G.6: Aplicación YourHour

- **Apps Usage<sup>7</sup>**. Esta segunda aplicación es más completa que la anterior, ya que nos permite obtener estadísticas del uso de aplicaciones diarias, semanales, historial de notificaciones o de consultas e incluso las aplicaciones más usadas por el usuario. Presenta una cronología del uso en tiempo real con el que se puede observar la inversión de nuestro tiempo minuto a minuto.

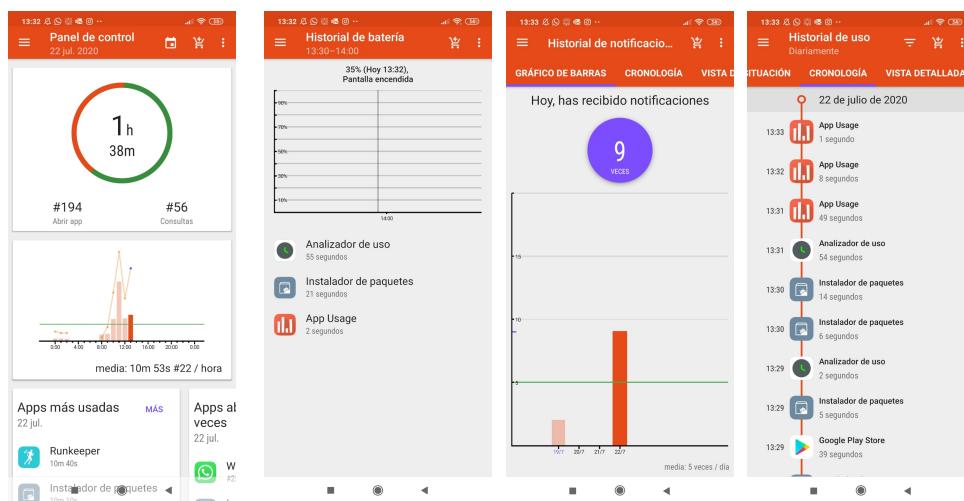
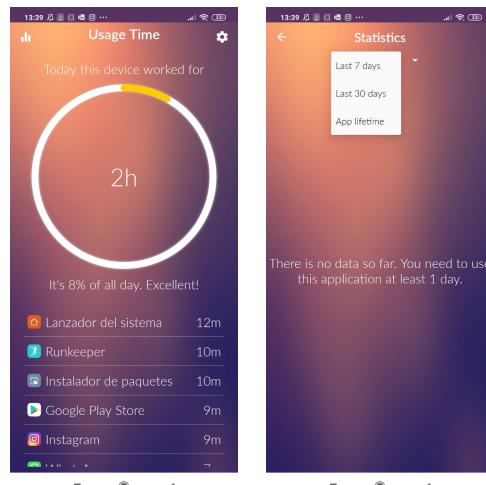


Figura G.7: Aplicación Apps Usage

<sup>7</sup><https://play.google.com/store/apps/details?id=com.a0soft.gphone.uninstaller&hl=es>

- **Smartphonoholic<sup>8</sup>**. Esta es otra aplicación simple que permite controlar el tiempo que se ha usado el dispositivo. En la pantalla principal se puede observar el tiempo de uso del dispositivo en el día actual y el tiempo que se ha usado cada aplicación. Además, se puede visualizar esta información de días anteriores.



**Figura G.8:** Aplicación Smartphonoholic

- **AntiSocial<sup>9</sup>**. Aplicación que contiene una interfaz simple y clara, que proporciona informes detallados con una variedad de datos precisos sobre el uso de las aplicaciones recogidas después de dos semanas y el registro de desbloqueos.



**Figura G.9:** Aplicación AntiSocial

- **MyAddictometer<sup>10</sup>**. Es una herramienta que permite ver la productividad de las personas y controlar su adicción a los dispositivos móviles mediante el análisis del tiempo que usas el teléfono. Es una app divertida y atractiva que incluye:

<sup>8</sup>[https://play.google.com/store/apps/details?id=com.jackypot.smartphonoholic&hl=es\\_419](https://play.google.com/store/apps/details?id=com.jackypot.smartphonoholic&hl=es_419)

<sup>9</sup><http://antisocial.io/>

<sup>10</sup><http://myaddictometer.com/>

- Seguimiento de uso diario del teléfono móvil y muestra una tendencia de uso durante el día, la semana y el mes.
- Número de desbloqueos del móvil para revisar la tendencia de realizar esta acción. Nos muestra una línea de tiempo con todos los bloqueos y desbloqueos para ver el tiempo que los hemos usado desde el inicio hasta el fin del día.
- Tiempo total dedicado al dispositivo móvil, comparativa del uso del móvil en días y detalles de uso de cada aplicación.

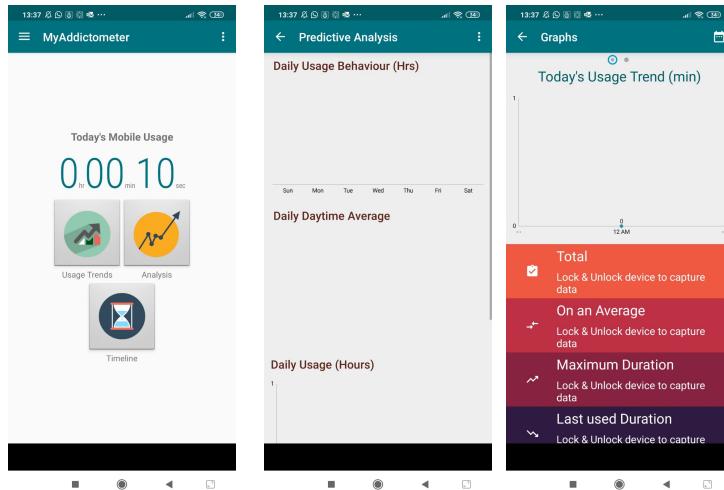


Figura G.10: Aplicación MyAddictometer

- **StayFree<sup>11</sup>**. Es una aplicación para dispositivos Android que muestra el tiempo que invertimos haciendo uso de nuestro dispositivo móvil, de las distintas aplicaciones y tus aplicaciones favoritas. El historial de uso se puede visualizar con una gráfica, permite exportar el historial en archivos CSV o Excel y mediante un gráfico circular muestra los porcentajes de uso diario y mensual de la aplicación.



Figura G.11: Aplicación StayFree

<sup>11</sup><https://play.google.com/store/apps/details?id=com.burockgames.timeclocker&hl=es>

---

## ANEXO H

---

# DOCUMENTACIÓN API REST

---

En este anexo, se enumeran los distintos *endpoints* utilizados para procesar la información almacenada en la base de datos y obtener un conjunto de métricas que van a ser representadas. Estos *endpoints* se presentan divididos en las categorías que fueron definidas en el Capítulo 1.

### BOTONES FÍSICOS

- **Botón bloquear y desbloquear**

- **Total número de bloqueos y desbloqueos en el histórico**

**URL:** /api/totalBloqueos/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Total número de bloqueos y desbloqueos en el día actual**

**URL:** /api/totalBloqueosD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Total número de bloqueos y desbloqueos en la semana actual**

**URL:** /api/totalBloqueosS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo total que el dispositivo ha permanecido bloqueado y desbloqueado en la semana actual**

**URL:** /api/tiempoDesbloqueadoS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo total que el dispositivo ha permanecido bloqueado y desbloqueado en el histórico**

**URL:** /api/tiempoDesbloqueadoH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo total que el dispositivo ha permanecido bloqueado y desbloqueado en el día actual**

**URL:** /api/tiempoDesbloqueadoD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Botón subir y bajar volumen**

- **Total número de veces pulsado el botón subir y bajar volumen en el día actual**

**URL:** /api/volumenD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Total número de veces pulsado el botón subir y bajar volumen en la semana actual**

**URL:** /api/volumenS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Total número de veces pulsado el botón subir y bajar volumen en el histórico**

**URL:** /api/volumen/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

## SISTEMA

- **Tiempo de uso de aplicaciones y servicios en primer plano en el día actual**

**URL:** /api/tiempoApps2/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo de uso de aplicaciones y servicios en primer plano en el histórico**

**URL:** /api/tiempoAppsH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo de uso de aplicaciones y servicios en primer plano en la semana actual**

**URL:** /api/tiempoAppsSemanal/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de aplicaciones instaladas y desinstaladas en el día actual**

**URL:** /api/appsInstaladasDesinstaladasD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de aplicaciones instaladas y desinstaladas en la semana actual**

**URL:** /api/appsInstaladasDesinstaladasS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de aplicaciones instaladas y desinstaladas en el histórico**

**URL:** /api/appsInstaladasDesinstaladasH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Duración llamadas teléfono en la semana actual**

**URL:** /api/duracionLlamadasS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Duración llamadas teléfono en el histórico**

**URL:** /api/duracionLlamadasH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Duración llamadas teléfono en el día actual**

**URL:** /api/duracionLlamadasD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de llamadas teléfono en el día actual**

**URL:** /api/numeroLlamadasD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de llamadas teléfono en el histórico**

**URL:** /api/numeroLlamadasH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de llamadas teléfono en la semana actual**

**URL:** /api/numeroLlamadasS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de notificaciones en el histórico**

**URL:** /api/notificacionesH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de notificaciones en la semana actual**

**URL:** /api/notificacionesS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de notificaciones en el día actual**

**URL:** /api/notificacionesD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

## REDES SOCIALES

- **Instagram**

Número de veces que se ha producido un determinado evento en Instagram en el día y semana actual, y en el histórico respectivamente:

- **URL:** /api/instagramD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **URL:** /api/instagramS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **URL:** /api/tiempoInstagramHistorico/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Facebook**

Número de veces que se ha producido un determinado evento en Facebook en el día y semana actual, y en el histórico respectivamente:

- **URL:** api/facebookD/name

**Método HTTP:** GET**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **URL:** /api/facebookS/name

**Método HTTP:** GET**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **URL:** /api/tiempoFacebookHistorico/name

**Método HTTP:** GET**Parámetros requeridos:** Name = [String] – Identificador del usuario**▪ Linkedin**

Número de veces que se ha producido un determinado evento en Linkedin en el día y semana actual, y en el histórico respectivamente:

- **URL:** /api/linkedinD/name

**Método HTTP:** GET**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **URL:** /api/linkedinS/name

**Método HTTP:** GET**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **URL:** /api/tiempoLinkedinHistorico/name

**Método HTTP:** GET**Parámetros requeridos:** Name = [String] – Identificador del usuario**▪ Twitter**

Número de veces que se ha producido un determinado evento en Twitter en el día y semana actual, y en el histórico respectivamente:

- **URL:** /api/twitterD/name

**Método HTTP:** GET**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **URL:** /api/twitterS/name

**Método HTTP:** GET**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **URL:** /api/tiempoTwitterHistorico/name

**Método HTTP:** GET**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo de uso de las aplicaciones de redes sociales en el día actual**
  - **URL:** /api/tiempoRRSSD/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo de uso de las aplicaciones de redes sociales en la semana actual**  
**URL:**/api/tiempoRRSSS/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario

## GMAIL Y OUTLOOK

### ▪ Gmail

Número de veces que se ha producido un determinado evento en Gmail en junto con el tiempo de uso de esta aplicación, en el día y semana actual, y en el histórico respectivamente:

- **URL:** /api/gmailD/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **URL:** /api/gmailS/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **URL:** /api/tiempoGmailHistorico/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario

### ▪ Outlook

Número de veces que se ha producido un determinado evento en Outlook en junto con el tiempo de uso de esta aplicación, en el día y semana actual, y en el histórico respectivamente:

- **URL:** /api/outlookD/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **URL:** /api/outlookS/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **URL:** /api/tiempoOutlookHistorico/name  
**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo de uso de las aplicaciones de correo electrónico GMail y Outlook en la semana actual**

**URL:** /api/tiempoCorreoS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

## WHATSAPP

- **Número de conversaciones abiertas y notificaciones recibidas en WhatsApp en el día actual**

**URL:** /api/totalConversacionesD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de conversaciones abiertas y notificaciones recibidas en WhatsApp en el histórico**

**URL:** /api/totalConversacionesH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de conversaciones abiertas y notificaciones recibidas en WhatsApp en la semana actual**

**URL:** /api/totalConversacionesS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de elementos enviados en WhatsApp en el día actual**

**URL:** /api/envioWAD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de elementos enviados en WhatsApp en la semana actual**

**URL:** /api/envioWAS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de elementos enviados en WhatsApp en el histórico**

**URL:** /api/envioWAH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de llamadas en WhatsApp en el histórico**

**URL:** /api/NllamadaWaH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de llamadas en WhatsApp en el día actual**

**URL:** /api/NllamadaWaD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de llamadas en WhatsApp en la semana actual**

**URL:** /api/NllamadaWaS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo visualizando imágenes o videos y escuchando audios en WhatsApp en el histórico**

**URL:** /api/tiempoIAV/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo visualizando imágenes o videos y escuchando audios en WhatsApp en la semana actual**

**URL:** /api/tiempoIAVS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo visualizando imágenes o videos y escuchando audios en WhatsApp en el día actual**

**URL:** /api/tiempoIAVD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo llamadas realizadas en WhatsApp en el histórico**

**URL:** /api/llamadaWaH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo llamadas realizadas en WhatsApp en el día actual**

**URL:** /api/llamadaWaD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo llamadas realizadas en WhatsApp en la semana actual**  
**URL:** /api/llamadaWaS/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo de uso aplicación WhatsApp en el histórico**  
**URL:** /api/tiempoWhatsappHistorico/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo de uso aplicación WhatsApp en la semana actual**  
**URL:** /api/tiempoWhatsappS/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario

## GOOGLE FOTOS

- **Tiempo empleado visualizando imágenes y videos en Google Fotos en la semana actual**  
**URL:** /api/tiempoGoogleS/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo empleado visualizando imágenes y videos en Google Fotos en el histórico**  
**URL:** /api/tiempoGoogleH/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo empleado visualizando imágenes y videos en Google Fotos en el día actual**  
**URL:** /api/tiempoGoogle/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo de uso de la aplicación Google Fotos en el histórico**  
**URL:** /api/graficoGoogleHistorico/name  
**Método HTTP:** GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario

## TECLADO

- **Número de letras, símbolos y emoticonos escritos, junto con el número de elementos eliminados en el histórico**  
**Letras, símbolos y emoticonos**

**URL:** /api/totalLetrasSimbolosH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de letras, símbolos y emoticonos escritos, junto con el número de elementos eliminados en la semana actual**

**URL:** /api/totalLetrasSimbolosS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **URL:** /api/totalLetrasSimbolosSS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número de letras, símbolos y emoticonos escritos, junto con el número de elementos eliminados en el día actual**

**URL:** /api/totalLetrasSimbolosD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

## SENSORES Y GOOGLE APIS

- **Sensores**

- **Número total de pasos dados en el histórico**

**URL:** /api/totalPasos/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número total de pasos dados en el día actual**

**URL:** /api/totalPasosD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número total de pasos dados en la semana actual**

**URL:** /api/totalPasosS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número total de pasos dados cada día de la semana actual**

**URL:** /api/pasosS/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Número total de pasos dados cada día en el histórico**

**URL:** /api/pasosH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Google APIs**

- **Tiempo realizando un tipo de actividad física mientras se hace uso de las redes sociales en el día actual**

**URL:** /api/tiempoAppsTD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras se hace uso de las aplicaciones de correo electrónico Outlook y Gmail en el día actual**

**URL:** /api/tiempoCorreoTD/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras se hace uso de las redes sociales en el histórico**

**URL:** /api/tiempoAAH/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- Tiempo realizando un tipo de actividad física mientras se hace uso de las aplicaciones de correo electrónico Outlook y Gmail en el día actual

**URL:** /api/tiempoCorreoTHisto/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras se hace uso de las redes sociales en la semana actual**

**URL:** /api/tiempoARRSSs/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- Tiempo realizando un tipo de actividad física mientras se hace uso de las aplicaciones de correo electrónico Outlook y Gmail en la semana actual

**URL:** /api/tiempoACorreos/name

**Método HTTP:** GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras el usuario escribe cualquier tipo de elemento en el día actual**

URL: /api/tiempoALetrasD/name

Método HTTP: GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras el usuario escribe cualquier tipo de elemento en la semana actual**

URL: /api/tiempoALetrasS/name

Método HTTP: GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras el usuario escribe cualquier tipo de elemento en el histórico**

URL: /api/tiempoALetrasH/name

Método HTTP: GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras se hace uso de la aplicación WhatsApp en el día actual**

URL: /api/tiempoAWD/name

Método HTTP: GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras se hace uso de la aplicación WhatsApp en la semana actual**

URL: /api/tiempoAWS/name

Método HTTP: GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras se hace uso de la aplicación WhatsApp en el histórico**

URL: /api/tiempoAWH/name

Método HTTP: GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras se realizan llamadas y videollamadas en WhatsApp en el día actual**

URL: /api/tiempoATWD/name

Método HTTP: GET

**Parámetros requeridos:** Name = [String] – Identificador del usuario

- **Tiempo realizando un tipo de actividad física mientras se realizan llamadas y videollamadas en WhatsApp en el histórico**  
URL: /api/tiempoATWH/name  
Método HTTP: GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo realizando un tipo de actividad física mientras se realizan llamadas y videollamadas en WhatsApp en la semana actual**  
URL: /api/tiempoATWS/name  
Método HTTP: GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo realizando un tipo de actividad física mientras se visualizan videos o imágenes y se reproducen audios en WhatsApp en la semana actual**  
URL: /api/tiempoATWVS/name  
Método HTTP: GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo realizando un tipo de actividad física mientras se realizan llamadas en el dispositivo en el día actual**  
URL: /api/tiempoATD/name  
Método HTTP: GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo realizando un tipo de actividad física mientras se realizan llamadas en el dispositivo en el histórico**  
URL: /api/tiempoATH/name  
Método HTTP: GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario
- **Tiempo realizando un tipo de actividad física mientras se realizan llamadas en el dispositivo en la semana actual**  
URL: /api/tiempoATS/name  
Método HTTP: GET  
**Parámetros requeridos:** Name = [String] – Identificador del usuario



# BIBLIOGRAFÍA

---

- [1] *Accelerometer Sensor*, Sensores de movimiento Android. dirección: [https://developer.android.com/reference/android/hardware/Sensor?hl=es#TYPE\\_ACCELEROMETER](https://developer.android.com/reference/android/hardware/Sensor?hl=es#TYPE_ACCELEROMETER) (visitado 06-04-2020).
- [2] *Accessibility Event*, Android Developers. dirección: <https://developer.android.com/reference/android/view/accessibility/AccessibilityEvent> (visitado 07-02-2020).
- [3] *Accessibility Node Info*, Android Developers. dirección: <https://developer.android.com/reference/android/view/accessibility/AccessibilityNodeInfo> (visitado 24-02-2020).
- [4] *Activity Recognition Client*, API de Google para Android. dirección: <https://developers.google.com/android/reference/com/google/android/gms/location/ActivityRecognitionClient> (visitado 28-04-2020).
- [5] *Activity Recognition Result*, API de Google para Android. dirección: <https://developers.google.com/android/reference/com/google/android/gms/location/ActivityRecognitionResult> (visitado 28-04-2020).
- [6] N. Aharony y col. (2015). «Funf, Open Sensing Framework», dirección: <http://www.funf.org/about.html> (visitado 07-02-2020).
- [7] BBVAOPEN4U. (2016). «API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos», dirección: <https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos> (visitado 28-03-2020).
- [8] K. Beck y col., *Manifiesto por el Desarrollo Ágil de Software*. 2001. dirección: <https://agilemanifesto.org/iso/es/manifesto.html> (visitado 17-02-2020).
- [9] L. Cabañero y col., *Characterisation of Mobile-Device Tasks by Their Associated Cognitive Load through EEG Data Processing*, 1.<sup>a</sup> ed. Department of Information Systems, Technologies y Department of Psychology, University of Castilla-La Mancha, Ciudad Real, 2019. dirección: <https://doi.org/10.3390/proceedings2019031070> (visitado 13-03-2020).
- [10] N. Chaín-Pinzón y B. Briñez, *Actividad física en adolescentes y su relación con agresividad, impulsividad, internet y videojuegos*. Psychologia: Avances de la disciplina. Vol 5. Nº 1, 2011. dirección: <https://revistas.usb.edu.co/index.php/Psychologia/article/view/1118/910> (visitado 20-05-2020).
- [11] *Chrome Custom Tabs*, Chrome. dirección: <https://developer.chrome.com/multidevice/android/customtabs> (visitado 05-03-2020).
- [12] *Crea tu propio servicio de accesibilidad*, Android Developers. dirección: <https://developer.android.com/guide/topics/ui/accessibility/service?hl=es-419> (visitado 06-02-2020).

- [13] *Descripción general de los servicios*, Android Developers. dirección: <https://developer.android.com/guide/components/services?hl=es-419> (visitado 25-04-2020).
- [14] *Detected Activity*, Android Developers. dirección: <https://developers.google.com/android/reference/com/google/android/gms/location/DetectedActivity> (visitado 25-04-2020).
- [15] D. Domínguez, *La forma de interactuar con el móvil puede revelar trastornos mentales*. 2019. dirección: [https://www.economiadigital.es/tecnologia-y-tendencias/la-forma-de-interactuar-con-el-movil-puede-revelar-trastornos-mentales\\_20025209\\_102.html](https://www.economiadigital.es/tecnologia-y-tendencias/la-forma-de-interactuar-con-el-movil-puede-revelar-trastornos-mentales_20025209_102.html) (visitado 03-07-2020).
- [16] N. Downie, *Chart JS, Simple yet flexible JavaScript charting for designers and developers*. dirección: <https://www.chartjs.org/> (visitado 24-03-2020).
- [17] L. Gilibets, *Qué es la metodología Kanban y cómo utilizarla*. 2013. dirección: <https://www.iebschool.com/blog/metodologia-kanban-agile-scrum/> (visitado 15-02-2020).
- [18] *Google Search Console, Mejora tus resultados en la Búsqueda de Google*, Google. dirección: <https://search.google.com/search-console/about?hl=es> (visitado 28-06-2020).
- [19] D. Griffiths y D. Griffiths, *Head First Android Development*, 1.<sup>a</sup> ed. O'REILLY, 2015.
- [20] A. Grünerbl y col., *Smartphone-Based Recognition of States and State Changes in Bipolar Disorder Patients*. IEEE, 2015. dirección: <https://ieeexplore.ieee.org/document/6866115/figures#figures> (visitado 05-02-2020).
- [21] S. Hoober, *How Do Users Really Hold Mobile Devices?* 2013. dirección: <https://www.uxmatters.com/mt/archives/2013/02/how-do-users-really-hold-mobile-devices.php> (visitado 12-07-2020).
- [22] *Interacción persona-computadora*, Wikipedia. dirección: [https://es.wikipedia.org/wiki/Interacci%C3%B3n\\_persona-computadora\\_%C3%A1reas\\_hist%C3%B3ricas](https://es.wikipedia.org/wiki/Interacci%C3%B3n_persona-computadora_%C3%A1reas_hist%C3%B3ricas) (visitado 12-07-2020).
- [23] Kanbanize, *Kanban: explicación para principiantes*. dirección: <https://kanbanize.com/es/recursos-de-kanban/primeros-pasos/que-es-kanban> (visitado 28-06-2020).
- [24] M. Karam y m. c Schraefel, *A Taxonomy of Gestures in Human Computer Interactions*. Electronics y Computer Science, 2005.
- [25] M. M. Lab, *Sondas implementadas Framework FUNF*. dirección: <https://github.com/funf-org/funf-core-android/tree/master/src/edu/mit/media/funf/probe/builtin> (visitado 04-05-2020).
- [26] W. Lawanont y M. Inoue, *A Development of Classification Model for Smartphone Addiction Recognition System Based on Smartphone Usage Data*. Shibaura Intitute of Technology, 2018. dirección: [https://link.springer.com/chapter/10.1007%2F978-3-319-59424-8\\_1](https://link.springer.com/chapter/10.1007%2F978-3-319-59424-8_1) (visitado 14-03-2020).
- [27] J. Lockhart y col., *Slim PHP Framework*. dirección: <http://www.slimframework.com/docs/v4/> (visitado 28-03-2020).
- [28] A. Mehrotra, R. J. Hendley y M. Musolesi, *Towards Multi-modal Anticipatory Monitoring of Depressive States through the Analysis of Human-Smartphone Interaction*. UbiComp '16 Proceedings of the 2016 ACM International Joint Conference on Pervasive y Ubiquitous Computing: Adjunct. Association for Computing Machinery , pp. 1132-1138, 2016. dirección: [http://pure-oai.bham.ac.uk/ws/files/29362980/MHSI\\_2016\\_paper\\_6\\_2.pdf](http://pure-oai.bham.ac.uk/ws/files/29362980/MHSI_2016_paper_6_2.pdf) (visitado 14-03-2020).

- [29] MrRios, *jsPDF*. dirección: <https://rawgit.com/MrRio/jsPDF/master/docs/index.html> (visitado 22-06-2020).
- [30] E. I. G. Pérez. (2019). «¿Cómo Crear Peticiones Con Javascript?», Código facilito, dirección: <https://codigofacilito.com/articulos/como-crear-peticiones-js> (visitado 12-04-2020).
- [31] *Peticiones XMLHttpRequest*, MDN web docs. dirección: <https://developer.mozilla.org/es/docs/Web/API/XMLHttpRequest> (visitado 04-04-2020).
- [32] A. M. M. Roffarello y L. D. Russis, *Towards detecting and mitigating smartphone habits*. Adjunct Proceedings of the 2019 ACM International Joint Conference on Pervasive, Ubiquitous Computing y Proceedings of the 2019 ACM International Symposium on Wearable Computers, 2019. dirección: <https://dl.acm.org/doi/10.1145/3341162.3343770> (visitado 14-03-2020).
- [33] *Step Counter*, Sensores de movimiento Android. dirección: [https://developer.android.com/guide/topics/sensors/sensors\\_motion?hl=es-419#sensors-motion-stepcounter](https://developer.android.com/guide/topics/sensors/sensors_motion?hl=es-419#sensors-motion-stepcounter) (visitado 05-05-2020).
- [34] *Step Detector*, Sensores de movimiento Android. dirección: [https://developer.android.com/guide/topics/sensors/sensors\\_motion?hl=es-419#sensors-motion-stepdetector](https://developer.android.com/guide/topics/sensors/sensors_motion?hl=es-419#sensors-motion-stepdetector) (visitado 04-05-2020).
- [35] C. A. L. Tacca, *El uso de las TICs y su relación con la actividad física en escolares de la institución educativa secundaria "Mariano Melgar"de Ayaviri*. 2015. dirección: [http://repositorio.unap.edu.pe/bitstream/handle/UNAP/4739/Lopez\\_Tacca\\_Cesar\\_Augusto.pdf?sequence=1&isAllowed=y](http://repositorio.unap.edu.pe/bitstream/handle/UNAP/4739/Lopez_Tacca_Cesar_Augusto.pdf?sequence=1&isAllowed=y) (visitado 17-05-2020).
- [36] M. R. Turró, *Evaluación y tendencias de la interacción persona-ordenador*. 2005. dirección: <http://deposit.ub.edu/dspace/bitstream/2445/24346/1/526223.pdf> (visitado 12-07-2020).
- [37] *Uso de Fetch*, MDN web docs. dirección: [https://developer.mozilla.org/es/docs/Web/API/Fetch\\_API/Utilizando\\_Fetch](https://developer.mozilla.org/es/docs/Web/API/Fetch_API/Utilizando_Fetch) (visitado 07-04-2020).
- [38] Viewnext, *Kanban aplicado al desarrollo de software*. 2019. dirección: <https://www.viewnext.com/kanban-desarrollo-software/> (visitado 22-02-2020).
- [39] *Web View*, Android Developers. dirección: <https://developer.android.com/reference/android/webkit/WebView> (visitado 30-03-2020).
- [40] T. D. Wilcockson, D. A. Ellis y H. Shaw, *Determining Typical Smartphone Usage: What Data Do We Need?* Mary Ann Liebert, Inc, 2018. dirección: <https://www.liebertpub.com/doi/pdf/10.1089/cyber.2017.0652> (visitado 14-03-2020).
- [41] L. Wroblewski, *Touch Gesture Reference Guide*. (n.d.) dirección: <https://www.lukew.com/ff/entry.asp?1071> (visitado 13-07-2020).
- [42] X. Yan, F. Hong y Z. Huo, *Research on application recognition technology based on user's touch screen behavior*. ACM TURC '19, 2019. dirección: <https://dl.acm.org/doi/10.1145/3321408.3321415> (visitado 17-06-2020).
- [43] S. Zhao y col., *Discovering different kinds of smartphone users through their application usage behaviors*. Proceedings of the 2016 ACM International Joint Conference on Pervasive y Ubiquitous Computing, 2016. dirección: <https://dl.acm.org/doi/10.1145/2971648.2971696> (visitado 13-02-2020).

