universidad de león

Escuela de Ingenierías
Industrial, Informática y Aeroespacial

MÁSTER EN INGENIERÍA INFORMÁTICA

Trabajo de Fin de Máster

Aplicación nativa móvil para dispositivos Android de la aplicación web del proyecto Sano y Feliz

Native mobile application for Android devices of the web application of Sano y Feliz project

Autor: Sergio Rubio Martín
Tutor: José Alberto Benítez Andrades

(Febrero, 2022)

UNIVERSIDAD DE LEÓN
Escuela de Ingenierías Industrial, Informática y Aeroespacial

MÁSTER EN INGENIERÍA INFORMÁTICA
Trabajo de Fin de Máster

ALUMNO: Sergio Rubio Martín

TUTOR: José Alberto Benítez Andrades

TÍTULO: Aplicación nativa móvil para dispositivos Android de la aplicación web del proyecto Sano y Feliz

TITLE: Native mobile application for Android devices of the web application of 'Sano y Feliz' project

CONVOCATORIA: Febrero, 2022

RESUMEN:

Este trabajo tiene como base un proyecto de investigación realizado por el grupo de investigación SALBIS, el cual se llevó a cabo sobre personas que se encontraban en la etapa de la adolescencia de algunos institutos. Dicho proyecto consistió en una intervención sobre la población infato-juvenil de varios institutos haciendo uso de una aplicación web con el objetivo principal de ayudar a prevenir el sobrepeso y la obesidad en dicha población. Este trabajo que se detallará a lo largo de esta memoria consiste en desarrollar e implementar una aplicación móvil de Sano y Feliz, aplicación web que apareció en el proyecto de investigación anteriormente mencionado. La aplicación móvil permite al usuario realizar todas las mismas funciones que en la propia web y que el dispositivo permita. Esta aplicación tiene dos zonas claramente distinguibles, una zona donde todo el mundo tiene acceso a información de dominio público, como por ejemplo, información sobre la investigación, consejos nutricionales y de actividad física, y la otra zona privada, donde solo los alumnos registrados previamente pueden acceder a la red social y otras áreas de interés del usuario. Dentro de esta zona privada el usuario puede utilizar todas las funcionales que una red social puede ofrecer. En esta zona privada puede publicar comentarios, responder a otros usuarios e incluso puede indicar si está de acuerdo o no, o si le ha gustado o no un comentario a través de un sistema de "Me gustas". Los usuarios pueden ver en todo momento cuántas respuestas tiene una

publicación y cuántos “Me Gusta” tiene. También el usuario puede configurar su perfil, no solo información basada en texto como las preferencias o datos personales, sino que puede modificar su foto de perfil, administrar sus amigos e incluso revisar solicitudes de amistad a través de la opción de Avisos. El usuario puede buscar a los diferentes usuarios que componen la red social utilizando un sistema de filtrado por si desea añadir a alguna persona como amigo. A mayores, el usuario puede utilizar de forma íntegra el sistema de gestión de eventos donde podrán decidir si crear un evento, participar en alguno que haya creado otra persona o, en caso de haber ya participado en un evento pasado, poder adjuntar una foto del momento. Por otro lado, el usuario puede seguir su mejora a través de la sección de “Mi Progreso” donde se muestra las medidas anteriormente realizadas por parte del usuario y en qué nivel se encuentra el usuario según su historial de hábitos alimenticios y su historial de hábitos de actividad física. Si el usuario desea comunicarse con uno de sus amigos de forma privada, puede hacerlo de forma directa a través de la opción de chat. Finalmente, el usuario puede cerrar sesión en cualquier momento que quiera. Todas estas funcionalidades serán implementadas en Android Studio con el lenguaje Java.

ABSTRACT:

This work is based on a research project carried out by the SALBIS research group, which was carried out on people who were in the adolescence stage from some institutes. This project consisted of an intervention on the child-adolescent population of several institutes using a web application with the main objective of helping to prevent overweight and obesity in said population. This work that will be detailed throughout this memory consists of developing and implementing a mobile application for Sano y Feliz, a web application that appeared in the aforementioned research project. The mobile application allows the user to perform all the same functions as on the website itself and that the device allows. This application has two clearly distinguishable areas, one area where everyone has access to information of public domain, such as information from the research, nutritional and physical activity, and the other private area, where only previously registered students can access the social network and other areas of interest to the user. Within this private area the user can use all the functions

that a social network can offer. In this private area the user can post comments, respond to other users and can even indicate whether he agree or not, or whether or not he liked a comment through a system of "Likes". Users can see at any time how many responses a post has and how many “Likes” it has. The user can also configure his profile, not only text-based information such as preferences or personal data, but he can modify his profile photo, manage his friends and even review friend requests through the Advertisement option. The user can search for the different users that make up the social network using a filtering system in case he wants to add a person as a friend. Furthermore, the user can fully use the event management system where he can decide whether to create an event, participate in one that someone else has created or, if he have participated in a past event, be able to attach a photo of the moment. On the other hand, the user can follow his improvement through the "My Progress" section where the measurements previously carried out by the user are shown and at what level the user is according to his history of eating habits and his history of physical activity habits. If the user wishes to communicate with one of his friends privately, he can do it directly through the chat option. Finally, the user can close the session at any time he wants. All these functionalities will be implemented in Android Studio with the Java language.

Palabras clave: Android; Fragmento; Salud; Aplicación; Scrum;

Firma del alumno:

RUBIO MARTIN SERGIO - 71470955L

Firmado digitalmente por RUBIO MARTIN SERGIO - 71470955L
DN: C=ES, SERIALNUMBER=IDCES-71470955L, G=SERGIO, SN=RUBIO MARTIN, CN=RUBIO MARTIN SERGIO - 71470955L
Fecha: 2022-02-21 10:34:56

VºBº Tutor/es:

BENITEZ ANDRADES JOSE ALBERTO - 71454586A

Firmado digitalmente por BENITEZ ANDRADES JOSE ALBERTO - 71454586A
Fecha: 2022.02.21 10:48:11 +01'00'

Índice de contenidos

Índice de figuras

Índice de tablas

1. Introducción

Este proyecto lo estoy realizando mientras soy investigador asociado y colaborador externo del grupo de Investigación SALBIS. Este proyecto corresponde al diseño, desarrollo e implementación de una aplicación móvil cuyas funcionalidades sean las mismas que la aplicación web Sano y Feliz, consiguiendo una migración parcial de la aplicación a otra plataforma móvil como es Android. Esta aplicación o plataforma es un portal que busca alentar o fomentar buenos hábitos saludables entre los jóvenes y, por otro lado, de forma más personalizada a través de un sistema de acceso granular, la aplicación ofrece al usuario una red social con diferentes funcionalidades que se detallarán a lo largo de esta memoria.

1.1 Contexto y motivaciones

En primer lugar, esta aplicación es un proyecto de investigación que lo está realizando el Grupo SALBIS (Salud, Bienestar y Sostenibilidad Sociosanitaria) que se encuentra dentro de la Universidad de León.

Figura 1.1 - Grupo de investigación SALBIS (Fuente: [1])

SALBIS tiene varias líneas de investigación, pero el mismo grupo aclara que sus principales líneas de investigación son:

- Análisis de Redes sociales y estrategias sanitarias
- Inteligencia emocional y positiva
- Tecnología e innovación aplicada al bienestar y a la salud.

- Movilidad y equilibrio
- Salud y calidad de vida
- Ingeniería del conocimiento y otras técnicas de inteligencia artificial aplicadas al ámbito de la salud

Con este proyecto se pretende contribuir a que se reduzca el porcentaje de sobrepeso, el cual puede llegar a convertirse en obesidad y producir problemas de salud a nivel físico y también a nivel psicológico. Las personas a las que va destinada esta aplicación es a los jóvenes, sobre todo a aquellos que se encuentran cursando la ESO en Castilla y León. Cabe destacar que el proyecto Sano y Feliz ya ha obtenido resultados positivos en relación al uso de la aplicación como herramienta para prevenir el sobrepeso y la obesidad en la población infanto-juvenil **[2][3]**.

1.2 Objetivos

El objetivo de este trabajo es realizar una migración parcial de la plataforma Sano y Feliz desde un entorno como es una aplicación web, a otro tipo de entorno como es Android, convirtiéndose así en una aplicación móvil y poder existir en dos entornos distintos. La dirección URL de la plataforma que se va a migrar es https://www.sanoyfeliz.es.

Figura 1.2 - Logo Sano y Feliz (Fuente: [4])

Sin entrar en detalles o especificaciones, la aplicación móvil contará con las características que vienen a continuación:

- Debe tener una interfaz que respete los colores de la versión web y ser accesible e intuitiva para la gran mayoría de los usuarios, intentando alcanzar que el mayor número de personas sea capaz de utilizar la aplicación.
- Deberá de tener un control de acceso separando claramente la zona pública de la zona privada.
- Se deberá mostrar al usuario un conjunto de hábitos nutricionales y deportivos de forma que el usuario reciba consejos.
- La red social permitirá a los usuarios interactuar entre sí, para que puedan comunicarse entre ellos y ofrecer, si lo prefieren, apoyo o motivación en la comunidad, fomentando un ambiente positivo y de compañerismo.
- La aplicación debe mostrar la mejora del usuario respecto a sus registros pasados.

1.3 Estructura del trabajo

A lo largo de este documento se mostrará de forma estructurada diferentes puntos principales.

En el capítulo dos, se describe el estado del arte que corresponde a este trabajo. Se desglosa la competencia actual a la que se enfrenta la aplicación tanto competencia directa como indirecta.

En el capítulo tres, se realiza el análisis de todos los requisitos de la aplicación, tanto funcionales como no funcionales.

En el capítulo cuatro, se muestra de forma general qué tecnologías se han utilizados y algunas ventajas que muestran frente a otras tecnologías en el contexto en el que se desarrolla el proyecto.

En el capítulo cinco, se detalla en profundidad qué tipo de metodología se ha utilizado para la planificación de la aplicación. Al utilizar una metodología ágil denominada Scrum, se muestran todas las historias de usuario que se deben de implementar a lo largo de los sprints y también se enseña qué reuniones se han realizado.

En el capítulo seis, se detalla un presupuesto final en euros de la aplicación móvil Sano y Feliz. Este presupuesto contendrá información acerca de los costes de mano de obra y materiales o elementos utilizados.

En el capítulo siete se redacta todo lo relacionado con el desarrollo de la propia aplicación. Este capítulo sin duda es el más extenso de toda la memoria ya que se describirán diferentes puntos. Algunos de estos puntos son:

- Entorno de desarrollo del proyecto.
- Arquitectura que se implementará en la aplicación.
- Diseño e implementación de la base de datos.
- Diferentes elementos o sistemas de desarrollos software especiales implementados en la aplicación.
- Implementar una conexión entre la aplicación móvil y el servidor vía FTP.
- Sacar la APK de la aplicación con un icono personalizado.

En el capítulo ocho se detallan aspectos legales que se deben tener en cuenta a la hora de realizar el trabajo para cumplir con la legalidad vigente, tanto a nivel nacional como europeo.

En el capítulo nueve se trata la importancia de la accesibilidad y la usabilidad de la aplicación final. Para ello, se detallan aspectos como los colores, formas, incluso iconos utilizados para fomentar una navegación sencilla e intuitiva.

En el capítulo diez se abordan las diferentes evaluaciones y pruebas realizadas desde el inicio del proyecto hasta el despliegue o entrega del mismo.

En el capítulo once está disponible un manual de uso para el usuario acerca de la aplicación.

En el capítulo doce se encuentran unas conclusiones del proyecto junto con varias líneas de futuro que se pueden llevar a cabo en un futuro.

En el capítulo trece se expresan unos agradecimientos a una persona en concreto y a un grupo de investigación.

Por último, en el capítulo catorce, se encuentra toda la bibliografía utilizada como respaldo o apoyo para ciertos aspectos de la memoria.

2. Estado del arte

El estado del arte permite mostrar desde un punto de vista de la investigación técnica, científica e industrial, el contexto en el que se encuentra una determinada tecnología utilizada y la competencia a la que se enfrenta de forma indirecta o directa el desarrollo de aplicación móvil.

Para ello se describe en primer lugar el tipo de competencias objetivas que se presentan actualmente en el mercado. Posteriormente, se detalla las principales competencias indirectas que existen.

2.1 Competencias

La competición en el mercado se refiere a todo aquel negocio, aplicación o empresa que ofrece un producto similar o parecido al que ofrece, en este caso, la aplicación móvil de Sano y Feliz. Se ha llevado a cabo un estudio acerca de cuáles son las principales competencias que me puedo encontrar en el mercado. Este estudio saca a relucir que hay dos tipos de competencias, las cuales se explicarán a continuación y qué ejemplos se encuentran en cada una de ellas.

2.1.1 Competencia indirecta

Con competencia indirecta se refiere a toda aplicación, empresa o entidad que ofrece ciertos servicios, en mayor o menor medida, muy similares a los servicios que ofrece la aplicación de Sano y Feliz. Existen varios ejemplos de competencia indirecta:

- La principal competencia indirecta es 8Fit, ya que es una aplicación que ofrece a los usuarios rutinas de ejercicios junto con otros consejos acerca de la alimentación. No obstante, no está pensado para que los usuarios puedan interactuar entre sí, más bien, la aplicación se centra en ver al usuario de forma individual sin ser parte de un colectivo **[5]**.

- Noom es una aplicación que, de forma similar, a través de que el usuario responda una serie de preguntas, se crea un perfil del usuario. Luego establece las tareas que debe realizar el usuario para mejorar y así alcanzar sus objetivos saludables. Noom realiza un seguimiento diario haciendo que los usuarios se midan diariamente y que de esta forma los propios usuarios vean las mejoras paulatinas que van alcanzando. Ofrece comunicación con un entrenador personal. No obstante, de nuevo solo se centra en una comunicación única entre el entrenador y el usuario, sin que el usuario pueda comunicarse con el resto de los usuarios para compartir metas, logros u opiniones **[6]**.

2.1.2 Competencia directa

La competencia directa son aquellas empresas o aplicaciones que ofrecen los mismos servicios que la aplicación resultante de este proyecto. Como competencia directa está Fuertafit, promocionado por Sergio Peinado, que no solo ofrece el servicio de dar consejos acerca de alimentación y rutinas para realizar actividad física, sino que también ofrece una red social o foro interno, la comunidad que contrata Fuertafit, para que los usuarios puedan motivarse entre ellos, se den ánimos y puedan compartir sus logros y alegrarse todos juntos **[7]**.

3. Análisis de requisitos

Todo proyecto necesita de una buena definición y especificación de requisitos para que así el programador pueda saber, qué quiere el grupo de investigación que incluya la aplicación móvil. De esta forma se pueden evitar que aparezcan posibles malentendidos acerca de lo que debe hacer la aplicación y de lo que debe de tener. Para tener todos los requisitos bien estructurados y clasificados haré uso del estándar de IEE 830 **[8]**. Los requisitos se dividen en funcionales y no funcionales. Posteriormente, los no funcionales se clasifican en requisitos de seguridad, portabilidad, disponibilidad y de accesibilidad. A continuación, se muestra un contexto donde surge una necesidad y qué requisito cubre dicha necesidad.

3.1 Requisitos funcionales

Para empezar la especificación de requisitos funcionales, primero hay que definir lo que es un requisito funcional. Este tipo de requisitos son aquellos que declaran los servicios que debe dar la aplicación móvil además de la forma en que dicha aplicación debe reaccionar a las entradas de los usuarios y también, debe definir cómo debe comportarse ante situaciones específicas que no se suelen dar por lo general. Pueden llegar a explicar y enfatizar lo que no debe hacer el sistema.

Todo usuario que entre en la aplicación debe entender qué es Sano y Feliz. También deberán poder conocer quienes están detrás de este proyecto y qué objetivos se quieren conseguir.

Requisito 1 – La portada de la aplicación debe explicar qué es Sano y Feliz.

Requisito 2 – Debe mostrar quienes han participado en el proyecto Sano y Feliz.

Requisito 3 – Debe mostrar cuáles son los objetivos del proyecto Sano y Feliz.

Requisito 4 – Debe contener una forma de recompensa que anime al usuario a seguir utilizando la app. Debe mostrar cuáles son las recompensas de seguir el proyecto Sano y Feliz.

Requisito 5 – Debe especificar rutinas de alimentación para el usuario.

Toda persona al principio puede sentirse perdida cuando está adquiriendo un nuevo estilo de vida más saludable y por lo tanto deberán de poder ser guiados de forma rápida y sencilla a través de la aplicación. Estos consejos alimenticios deben mostrarse en un tablón.

Requisito 6 – Debe especificar consejos de cómo activarse y tener mejor salud.

Requisito 7 – Debe tener una sección para un blog de noticias.

Sano y Feliz cuenta con una red social en la cual los usuarios pueden interactuar entre ellos ofreciendo mensajes positivos y de ánimos. Por ello es imprescindible que exista un sistema de inicio de sesión.

Requisito 8 – Debe tener un sistema de inicio y cerrado de sesión.

Requisito 9 – Debe dejar iniciar sesión a aquellos usuarios que se han registrado a través de la web.

Una vez que el usuario ha entrado en la aplicación móvil, tendrá a su disposición multitud de funcionalidades que ofrece cualquier red social. Además, deberá tener diferentes opciones que permitan personalizar su perfil.

Requisito 10 – Debe permitir al usuario acceder a su propio perfil y pudiendo ver las publicaciones realizadas y los “Me gustas” dados.

Requisito 11 – Debe permitir al usuario configurar su foto de perfil y su privacidad, a la vez que tener la posibilidad de cambiar datos personales y cambiar la contraseña.

Requisito 12 – Debe permitir al usuario ver en una zona de avisos las solicitudes de amistad que otros usuarios le mandan.

Requisito 13 – Debe permitir al usuario ver en forma de lista a sus amigos.

Requisito 14 – Debe permitir al usuario realizar búsqueda de resto de usuarios que están registrados en la aplicación y filtrar a través de ellos para poder encontrar a otro usuario.

Requisito 15 – Debe permitir enviar peticiones de amistad a otros usuarios y también recibirlas.

Requisito 16 – Debe permitir al usuario crear nuevos eventos.

Requisito 17 – Debe permitir asistir a eventos públicos próximos y cancelar la asistencia de sí mismo al evento.

Requisito 18 – Debe permitir al usuario visualizar a qué eventos está asistiendo.

Requisito 19 – Debe permitir al usuario visualizar los eventos asistidos en el pasado y además el usuario podrá adjuntar una imagen para verificar su asistencia.

Requisito 20 – Debe permitir al usuario dar "Me gusta" a los comentarios de sus amigos y cancelar el "Me gusta".

Requisito 21 – Debe permitir al usuario comentar o responder en los comentarios de sus amigos, es decir, el usuario podrá poner comentarios de forma independiente o si lo prefiere podrá responder a otro comentario de otro usuario.

Requisito 22 – Debe permitir al usuario ver el número de "Me gustas" y de comentarios que tiene una publicación.

Requisito 23 – Debe permitir al usuario ver el hilo de comentarios que suceden a un comentario.

Requisito 24 – Debe permitir al usuario ver su progreso. En esta sección el usuario puede ver su registro de medidas que se ha realizado el mismo, su nivel y progreso en el sistema de "BieneSTARS", su historial de hábitos alimenticios y también su historial de hábitos de actividad física.

Requisito 25 – Debe permitir al usuario tener conversaciones por mensaje directo entre usuarios, es decir, habrá un sistema de chat directo donde los usuarios que son amigos entre sí puedan comunicarse.

Requisito 26 – Debe tener una sección donde el usuario que ha iniciado sesión podrá ver a otros usuarios para que éste puede decir si quiere agregar a alguien como amigo. Estas sugerencias de amistad son aleatorias en cada momento.

3.2 Requisitos no funcionales

Al contrario que los requisitos funcionales, los requisitos no funcionales deben definir las limitaciones en los servicios o funciones que ofrece la aplicación. Estas restricciones pueden ser muy diversas, por ejemplo, pueden ser restricciones de temporización o pueden estar relacionadas con el desarrollo en sí de la aplicación. También son requisitos no funcionales aquellos que se encargan de legislar casos especiales, estos últimos se encuentra en el apartado de requisitos de seguridad.

3.2.1 Requisitos de seguridad

La ciberseguridad es un aspecto de la informática que cada día va ganando más relevancia ya que todo va dependiendo más y más de la informatización. Por ello el sistema de iniciar sesión el usuario debe ser seguro, por lo que hay que seguir una política de cambio de contraseñas cada cierto tiempo.

Requisito 27 – Las credenciales de un usuario solo las debe tener el usuario.

Como el sistema de acceso granular basado en usuario y contraseña es el sistema principal de seguridad, se debe tener más preocupación a la hora realizar el tratamiento de estos datos.

Requisito 28 – La contraseña será encriptada y posteriormente almacenada en base de datos de forma que, si se produce un ataque directo a base de datos produciendo que se filtre todos los datos de los usuarios, al menos no tengan un acceso legible a las contraseñas manteniendo el control de la cuenta en las manos del legítimo dueño.

Es importante ser consciente de que como somos personas y estamos continuamente en entornos cambiantes que requieren de la máxima atención, puede que el usuario se olvide de la contraseña. Por eso hay que darle una solución al usuario de cómo recuperar o cambiar dicha contraseña.

Requisito 29 – Se mostrará al usuario un modo de contactar con el administrador de la aplicación para preguntar dudas, notificar errores, solicitar nueva contraseña o pedir modificar la contraseña.

En ciertos momentos donde el usuario desea llevar a cabo una acción, puede que, durante ese proceso, tenga que introducir alguna palabra o tenga que realizar otro tipo de labor. Sin este paso podría existir vulnerabilidades dentro de la aplicación que podrían ser críticas en caso de un ciberataque. Es por ello, por lo que se tendrá en cuenta este hecho a la hora de realizar la implementación.

Requisito 30 – Cuando el usuario introduzca sus credenciales para iniciar sesión, si éste introduce algún dato erróneamente, se le mostrará un mensaje de advertencia informando acerca de que las credenciales utilizadas no son correctas.

Requisito 31 – Cuando el usuario publica un comentario ya bien sea de tipo respuesta o comentario inicial, la aplicación debe mostrarle un texto informativo si el texto está en blanco.

Requisito 32 – Cuando el usuario quiere chatear con algún amigo, la aplicación mostrará un mensaje en caso de que quiera enviar un mensaje si no ha elegido a un amigo como receptor. Además, mostrará otro mensaje de advertencia en caso de que quiera enviar un mensaje con cero caracteres, es decir, un mensaje vacío.

3.2.2 Requisitos de portabilidad

Un requisito de portabilidad es aquel que solo especifica qué propiedades debe tener o soportar la aplicación para ejecutarse en diferentes plataformas. La aplicación, según el solicitante del proyecto, solo se desplegará en dispositivos Android ya que es el sistema operativo para móviles más extendido, rozando el 87% frente a otros sistemas operativos en España **[9]**. Para evitar que la aplicación no se ejecute adecuadamente en algún dispositivo, sobre todo en dispositivos antiguos, es importante detallar que en esos casos la aplicación directamente no se ejecutará.

Requisito 33 – La aplicación solo está disponible en dispositivos móviles que tengan el sistema operativo Android.

Requisito 34 – La aplicación solo se ejecutará en versiones de Android que tengan como mínimo la versión Android Lollipop que se corresponde con la versión 5.0.

3.2.3 Requisitos de accesibilidad

Un requisito de accesibilidad es aquel que se preocupa de que la aplicación pueda ser utilizada de la forma más intuitiva y fácilmente posible por parte del usuario. La navegación es una parte muy importante cuando se está utilizando una aplicación, por ello se debe buscar que el usuario no pueda perderse.

Requisito 35 – La aplicación debe tener un menú desplegable lateral que permite al usuario ver las diferentes opciones o secciones que hay en la aplicación.

Cualquier persona que haya entrado en la aplicación web de Sano y Feliz debe identificar rápidamente que esta aplicación, es una extensión de Sano y Feliz pero en el mundo de Android a través del conjunto de colores que utiliza en su aplicación web.

Requisito 36 – La aplicación debe ser tener colores similares a la aplicación web de Sano y Feliz.

Requisito 37 – Los iconos utilizados dentro de la aplicación deben facilitar la navegación de forma que reflejen de forma clara e inequívoca qué hay en cada sección correspondiente.

Requisito 38 – El logo de la aplicación deberá estar preparado para los diferentes formatos en los que se debe almacenar para todo tipo de móviles Android.

Requisito 39 – En la sección “El proyecto”, se deben establecer hipervínculos para que el usuario pueda verificar la información. Los elementos que contarán con hipervínculos son el grupo SALBIS, LEO14G18 otorgado por la Junta de Castilla y León y la Universidad de León.

Requisito 40 – En la sección “El proyecto”, se debe utilizar el estilo negrito para resaltar algunas palabras claves de la información.

4. Tecnologías

4.1 Android studio

Android Studio es el principal entorno de desarrollo utilizado por los programadores para llevar a cabo aplicaciones cuyo sistema operativo final es Android. Esta tecnología ofrece varias funciones, por lo se destaca a continuación las principales ventajas frente a otras tecnologías. Muchas de ellas están relacionadas con la mejora del rendimiento del programador **[10]**:

- Permite desplegar emuladores Android de diferentes versiones y tamaños. Esto evita que el programador tenga que desplegar siempre la aplicación en un móvil físico y ver de forma rápida como queda visualmente la aplicación y si existe algún error en la programación.
- Permite al usuario controlar en qué dispositivos puede la aplicación ser instalada y ejecutada. De esta forma, el programador puede especificar a partir de qué versiones de Android la aplicación ya no funcionará.
- El programador puede integrar Github dentro de este entorno de desarrollo para que, si lo ve oportuno, pueda mantener un control de las versiones de las diferentes aplicaciones que esté llevando a cabo. Permite también trabajan con los sistemas de ramas que se encuentran en Github.
- Android Studio ofrece mejor rendimiento que otras tecnologías como React Native.

4.2 Java

Java es uno de los lenguajes de programación más extendidos en el mundo de la programación, por lo que la gran mayoría de dispositivos se han programado en este lenguaje. Java dentro de los diferentes lenguajes de programación, es un lenguaje orientado objetos, o lo que es lo mismo, trabaja con datos y con interfaces como si fueran objetos, permitiendo encapsulación, herencia y polimorfismo. Contiene librerías para interactuar con protocolos como http y ftp. También cabe destacar que Java es un lenguaje

compilado e interpretado al mismo tiempo ya que compila los archivos con código a archivos class, por lo tanto, solo es necesario compilar el programa una vez, pero se interpreta cada vez que se ejecuta en un ordenador gracias a la Máquina Virtual Java (JVM) **[11].** Debido a la gran documentación pública realizada tanto por los creadores de Java y, en mayor medida por los desarrolladores, se encuentran soluciones a posibles problemas que pueden surgir en el desarrollo de la aplicación. Un ejemplo evidente donde los desarrolladores de software pueden encontrar una gran cantidad de documentación para el lenguaje Java es en el foro Stack Overflow **[12]**.

4.3 Myqsl

MySQL es un sistema de gestión de bases de datos relacional, el cual está siendo utilizado por multitud de desarrolladores para sus aplicaciones. Esta tecnología presenta diferentes puntos fuertes, de los cuales se muestran algunos a continuación **[13]**:

- Está basado en la arquitectura Cliente-Servidor, es decir, el rendimiento es bastante eficiente.
- Al estar basado en SQL, que es el lenguaje base más generalizado dentro del mundo de las bases de datos relacionales, permite que diferentes diseñadores y programadores de bases de datos puedan venir de otros entornos basados en SQL.
- MySQL permite usar procedimientos almacenados para agilizar sentencias y dar una capa de abstracción al código.
- MySQL permite automatizar y gestionar algunas tareas dentro de la base de datos cuando se cumplan una serie de características establecidas.
- En cuanto a las transacciones, tiene un sistema de seguridad para preservar la integridad de la base de datos manteniendo segura la información.
- Tiene un sistema de control y gestión gráfico llamado Workbench que permite manejar de forma accesible varias bases de datos.

4.4 Github

Github es un controlador de versiones que hace uso de git y permite que cualquier desarrollador de software puede llevar una gestión de las versiones de su software de forma remota. Así el programador no es necesario que solo trabaje desde un equipo, ya que puede acceder a su código desde otro equipo haciendo uso del repositorio remoto **[14]**.

4.5 Php

PHP es un acrónimo de "Hypertext Preprocessor", es un lenguaje de código abierto enfocado en el desarrollo de aplicaciones web y puede ser utilizado directamente sobre el propio HTML que compone la estructura de la web **[15]**. La principal diferencia que tiene respecto a Javascript es que el código con PHP se ejecuta en el servidor en vez de ser ejecutado en el cliente dando cierta capa de opacidad ya que el cliente verá el resultado final de la ejecución sin saber qué código se ha ejecutado. En este proyecto se utiliza PHP para crear un "web service" que permite al usuario subir desde la propia aplicación una foto de perfil del usuario.

4.6 Filezilla

Filezilla es un programa que permite el traspaso de ficheros y datos entre dos equipos de forma remota. Tiene dos opciones, una para que el equipo funcione como cliente o la otra opción para que funcione como servidor. Este programa, independientemente de la opción escogida, utiliza el protocolo FTP sobre TLS y SFTP para comunicar dos equipos de forma remota y establecer un canal de comunicación por el cual se puede transferir archivos entre ellos **[16]**.

5. Planificación del proyecto

5.1 Scrum

5.1.1 Definición

Scrum es una metodología ágil que acepta el cambio en el software y está diseñada para ser utilizada en entornos o proyectos que se encuentran en constante cambio o evolución **[17]**. Es una de las metodologías más utilizadas en estos días en el ámbito del desarrollo de aplicaciones. El motivo principal es que elimina la idea errónea de que desarrollar software es lo mismo que construir o crear un objeto físico. En la creación de software se necesita abordar e implementar cambios rápidos sin que esto afecte a la planificación o evolución del proyecto. Concretamente Scrum se basa en ciclos iterativos e incrementales, por lo que cada cierto tiempo se va obteniendo un prototipo o entregable de la aplicación de forma periódica a través de las iteraciones. Cada entregable es el resultado del incremento del entregable anteriormente mencionado y que permite ver la evolución del proyecto a través de la implementación de funcionalidades añadidas entre entregables de diferentes iteraciones.

El riesgo es un factor muy a tener en cuenta a la hora de llevar a cabo un proyecto. No obstante, el desarrollo incremental que ofrece Scrum reduce en gran medida los riesgos que surgen, ya que se integran diferentes módulos de mejora en un mismo programa. Si a esto se le añade el sistema iterativo de desarrollo, donde se va desarrollando software por partes, se puede ver que en cada punto se comprueba que el software funciona. De forma que en la siguiente iteración todo lo que se lleve a cabo será sobre la base de un programa que funciona correctamente, facilitando así la búsqueda de errores tanto a nivel gráfico como a nivel de código.

A continuación, se muestran de forma resumida alguna de las principales ventajas que ofrece Scrum frente a otras metodologías:

- ✓ La primera es que ofrece y mantiene una visión real de cómo evoluciona el proyecto y muestra en qué punto se encuentra en cada momento. Así se puede conocer qué requisitos se han implementado ya y cuáles son los que faltan por implementar.
- ✓ La segunda es que al tener un entregable o prototipo al finalizar cada iteración, el dueño del producto puede dar sus impresiones o comentarios que ayudan a mejorar el prototipo en la próxima iteración. De esta forma, se puede saber si las evoluciones del prototipo satisfacen los requisitos establecidos por el cliente.
- ✓ La tercera es que, al dividir el trabajo total en historias de usuario, las cuales se repartirán en las iteraciones, se puede manejar el riesgo del proyecto de forma eficaz y eficiente ya que los requisitos complejos se fragmentan en requisitos pequeños que son más fáciles de cumplir.
- ✓ La cuarta es que por cada iteración que transcurre, la experiencia y el aprendizaje del equipo de desarrollo aumenta significativamente, ya que no solo adquieren más conocimiento, sino que mejoran en el trabajo en equipo elevando así el rendimiento de desarrollo.

Las ventajas mostradas anteriormente son razones de peso suficientes para utilizar Scrum como metodología del proyecto. Sin embargo, no es la única razón por la que se ha elegido utilizar Scrum, sino que ya se tiene experiencia en proyectos cuya metodología era Scrum, por lo que no habrá dificultades durante el proyecto.

5.1.2 Roles

Dentro de la metodología Scrum se pueden identificar tres roles diferentes:

- Product Owner: conocido como dueño del producto, es el rol que toma el cliente final que solicita el desarrollo de la aplicación. Este dueño del producto puede ser un representante que haya asignado la verdadera persona que haya solicitado el proyecto. Por ello, se hace un gran énfasis en que este rol recae en una única e inequívoca persona y es el encargado de establecer el valor de las funcionalidades que tendrá la aplicación. Esta persona se recomienda que esté disponible en todo

momento o al menos el mayor tiempo posible, ya que tiene que ofrecer una respuesta a todas aquellas dudas que puedan surgir al equipo de desarrollo, ofreciendo así, una visión clara y definida de lo que se busca hacer. Además, es el encargado de gestionar u ordenar la pila del producto que forman las funcionalidades finales del sistema. También es recomendable, que revise los resultados obtenidos en cada iteración.

- Equipo de desarrollo: está compuesto por una serie de personas encargadas de desarrollar y llevar a cabo el sistema o la aplicación final. Este equipo es multidisciplinar para que todas las tareas puedan finalizarse satisfactoriamente y a su vez debe ser auto organizado. El equipo también puede poner nuevas funcionalidades o mejoras en común al dueño del producto. Comúnmente, el equipo está compuesto por siete personas, pero no existe una cifra exacta determinada.
- Scrum Master: no hay que confundirlo con jefe del proyecto que suele aparecer en otro tipo de metodologías más enfocadas a las tradicionales y no a las ágiles. Este rol es de vital importancia ya que sirve como coach y forma al resto de personas que se encuentran dentro del proyecto que está utilizando la metodología scrum. Es imprescindible que el dueño del producto no sea al mismo tiempo el Scrum Master.

Una vez explicados cada rol de la metodología scrum, se debe determinar que tanto el rol de Scrum Master como equipo de desarrollo es llevado por mí, ya que este proyecto es un trabajo final de máster. Por otro lado, el dueño del producto es el grupo de investigación SALBIS, que es la entidad que me ha solicitado llevar a cabo la aplicación móvil de Sano y Feliz.

5.1.3 Historias de usuario

Una historia de usuario describe fácilmente una de las múltiples funcionalidades que han sido solicitadas por el dueño del producto y que, por ende, forman parte del proyecto. Estas historias de usuarios se utilizan para definir de forma clara qué debe realizar la aplicación

en vez de enfocarse en cómo se puede implementar. El nivel de detalle de las historias de usuario no debe ser muy elevado porque no es idénticamente igual a una especificación técnica de requisitos, sino que más bien, las historias de usuario deben ser fáciles tanto de recordar como implementar en el producto, es decir, si una historia de usuario es compleja y puede causar varios obstáculos, lo más recomendable es dividir la historia de usuario en otras más pequeñas y alcanzables. Así la dificultad del proyecto se reduce considerablemente.

Cuando se agrupan todas las historias de usuario que componen el producto final se le denomina pila del producto. Dentro de Scrum existen varios estilos de historias de usuario, aunque todos tienen una estructura muy similar salvo algunos detalles. A continuación, en la Figura 5.1 se muestra el estilo de historia que se ha utilizado en este proyecto.

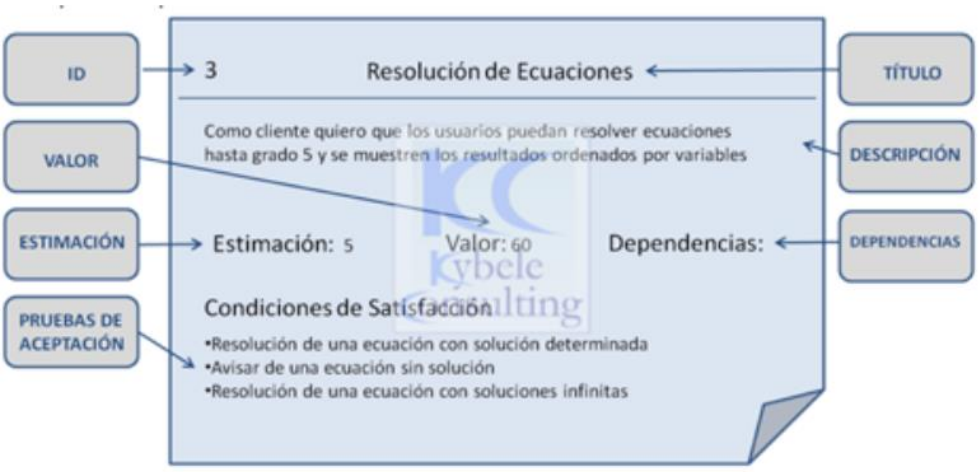

Figura 5.1 - Forma de una historia de usuario (Fuente:[18])

Cada campo que componen la historia de usuario tiene una función importante y son las siguientes:

- ID: es el número que identifica a la historia de usuario y no puede haber números repetidos entre las historias de usuario.
- Título y descripción: el título que recibe la historia sirve para que, a simple vista, solo con leer el título, se sepa de qué va la historia. Justo debajo del título suele ir

una descripción un poco más extensa para detallar la funcionalidad que representa la historia de usuario.

- Estimación: es el esfuerzo estimado que se necesita para llevar a cabo la implementación de la historia de usuario. Esta estimación es realizada por el equipo de desarrollo. No hay una fórmula o método estándar para calcular la estimación. El equipo de desarrolladores suele tomar como referencia, cuánto tiempo necesitarán para llevarlo a cabo, nivel de complejidad de la implementación y cuántas personas y recursos harán falta. La unidad de medida de esta estimación se denomina puntos de historia.
- Valor: es un número que refleja cuánto de valioso es para el dueño del producto implementar la historia de historia, es decir que es el dueño del producto quien asigna este valor a las historias. Si la historia de usuario es muy valiosa e importante para el dueño del producto tendrá un valor más elevado que el resto de historias.
- Dependencias: este campo permite saber qué historias de usuarios deben haberse llevado a cabo obligatoriamente para poder realizar la historia de usuario. En caso de que no tenga dependencias, significa que la historia de usuario es independiente al resto de historias de usuarios. Por otro lado, si la historia de usuario tiene dependencias, se mostrarán con sus respectivos IDs.
- Pruebas de aceptación: muestra las condiciones de aceptación para saber que, cuando se cumplan esas condiciones, la historia de usuario se ha realizado correctamente y queda finalizada.

5.1.4 Sprints

Un sprint es el término utilizado para referirse a una iteración de la metodología Scrum. Normalmente esta iteración tiene una duración entre dos y cuatros semanas, pero una vez empezado el proyecto, los sprints deben tener siempre la misma duración. En cada sprint se tiene que conseguir implementar al completo la pila del sprint. La pila del sprint consiste en todas las historias de usuario ordenadas que han sido planificadas para llevarse a cabo en dicho sprint. Una vez empezado el sprint no puede ser modificado. No obstante, si las

circunstancias externas o internas hacen que haya un cambio de prioridades muy importante, se puede llegar a parar o abortar el sprint, ya que no se estaría avanzando en el proyecto sino más bien perdiendo tiempo y recursos valiosos. En este proyecto se ha considerado oportuno realizar 4 sprints de un mes de duración cada uno de ellos respectivamente. En la Tabla 5.1 se muestra la planificación del proyecto utilizando Scrum.

Nombre	Fecha
Reunión de Planificación Sprint 1	18/11/2020
Sprint 1	18/11/2020 a 20/12/2020
Reunión Revisión Sprint 1	21/12/2020
Reunión Retrospectiva 1	21/12/2020
Reunión Planificación Sprint 2	21/12/2020
Sprint 2	21/12/2020 a 20/01/2021
Reunión Revisión Sprint 2	21/01/2021
Reunión Retrospectiva Sprint 2	21/01/2021
Reunión Planificación Sprint 3	21/01/2021
Sprint 3	21/01/2021 a 21/02/2021
Reunión Revisión Sprint 3	22/02/2021
Reunión Retrospectiva Sprint 3	22/02/2021
Reunión Planificación Sprint 4	22/02/2021
Sprint 4	22/02/2021 a 22/03/2021
Reunión Revisión Sprint 4	23/03/2021
Reunión Retrospectiva Sprint 4	23/03/2021

Tabla 5.1 - Tabla de planificación de los sprints (Fuente: Elaboración propia)

Las reuniones que son de vital importación en la metodología scrum se explicarán en el siguiente apartado. De esta forma se podrá detallar de la mejor manera posible todos los aspectos de cada reunión respectivamente. A continuación, se describe qué historias de

usuario serán implementadas y en qué sprints. Como se puede observar en la Tabla 5.1, el primer sprint empezó el 18 de noviembre de 2021 y finaliza el 21 de diciembre de 2021. Las historias que serán implementadas en el sprint uno se muestran desde la Tabla 5.2 hasta la Tabla 5.7.

ID:1	**Puesta en marcha del entorno de desarrollo**	
Como desarrollador se quiere tener un entorno de desarrollo adaptado para implementar todas las funcionalidades del cliente.		
Estimación: 10	Valor: 50	Dependencias: -
Condiciones de aceptación: ▪ Todos los drivers instalados ▪ Descargado emuladores de Android		

Tabla 5.2 - Historia de usuario 1 (Fuente: Elaboración propia)

ID:2	**Persistencia en base de datos**	
Como cliente se quiere que los todos los datos puedan guardarse en BBDD.		
Estimación: 90	Valor: 100	Dependencias: -
Condiciones de aceptación: ▪ Establecer conexión con la base de datos ▪ Base de datos desplegada		

Tabla 5.3 - Historia de usuario 2 (Fuente: Elaboración propia)

ID:3	**Introducir información relevante en BBDD**	
Como desarrollador se necesita datos de prueba para almacenar en BBDD.		
Estimación: 20	Valor: 90	Dependencias: 2
Condiciones de aceptación: ▪ Se pueden consultar los datos de BBDD		

Tabla 5.4 - Historia de usuario 3 (Fuente: Elaboración propia)

ID:4	**Sistema de menú lateral desplegable**	
Como cliente quiere que los usuarios puedan hacer uso de un menú lateral desplegable que facilite la navegación.		
Estimación: 30	Valor: 50	Dependencias: 1
Condiciones de aceptación: ▪ El usuario puede utilizar el menú lateral para la navegación.		

Tabla 5.5 - Historia de usuario 4 (Fuente: Elaboración propia)

ID:5	**Mostrar portada**	
Como cliente quiere que los usuarios puedan ver una portada de lo que es Sano y Feliz. Además, debe ser llamativa.		
Estimación: 30	Valor: 90	Dependencias: 4
Condiciones de aceptación: ▪ El usuario puede ver la portada de la aplicación. ▪ El carrousel dinámico funciona correctamente. ▪ Se muestra una presentación de lo que es sano y Feliz.		

Tabla 5.6 - Historia de usuario 5 (Fuente: Elaboración propia)

ID:6	**Mostrar datos del proyecto de investigación**	
Como cliente quiere que los usuarios puedan conocer quién está a cargo del proyecto junto con otros detalles más específicos del proyecto.		
Estimación: 30	Valor: 70	Dependencias: 4
Condiciones de aceptación: ▪ El usuario puede ver quién está a cargo del proyecto. ▪ El usuario puede ver información complementaria del proyecto. ▪ El usuario puede ver los hipervínculos y los términos importantes en negrita.		

Tabla 5.7 - Historia de usuario 6 (Fuente: Elaboración propia)

Como se ha podido ver, en el primer sprint se ha preparado el entorno de trabajo y se ha implementado una conexión con persistencia en base de datos. Además, se han añadido algunas funcionalidades para la parte pública de la aplicación.

Ahora bien, el segundo sprint empezó el 21 de diciembre de 2020 y finalizó el 20 de enero de 2021. La pila del sprint dos contiene las historias de usuario que aparecen desde la Tabla 5.8 hasta la Tabla 5.12.

ID:7	**Mostrar blog**	
Como cliente quiere que los usuarios puedan ver una serie de noticias o blogs que pueden ser de su interés relacionadas con el ejercicio y la alimentación.		
Estimación: 20	Valor: 50	Dependencias: 3,4
Condiciones de aceptación: ▪ El usuario puede visualizar las noticias disponibles.		

Tabla 5.8 - Historia de usuario 7 (Fuente: Elaboración propia)

ID:8	**Mostrar consejos acerca del ejercicio**	
Como cliente quiere que los usuarios puedan ver diferentes consejos generales que les ayuden a activarse y aumentar la cantidad de ejercicio que realizan.		
Estimación: 30	Valor: 60	Dependencias: 3,4
Condiciones de aceptación: ▪ El usuario puede visualizar diferentes consejos para hacer ejercicio.		

Tabla 5.9 - Historia de usuario 8 (Fuente: Elaboración propia)

ID:9	**Mostrar consejos acerca de la nutrición**	
Como cliente quiere que los usuarios puedan ver diferentes consejos generales que les ayuden a conocer nuevas formas de mejorar su nutrición.		
Estimación: 30	Valor: 60	Dependencias: 3,4
Condiciones de aceptación: ▪ El usuario puede visualizar diferentes consejos para mejorar la nutrición.		

Tabla 5.10 - Historia de usuario 9 (Fuente: Elaboración propia)

ID:10	**Iniciar sesión**	
Como cliente quiere que los usuarios puedan iniciar y cerrar sesión para poder hacer uso de la red social.		
Estimación: 50	Valor: 90	Dependencias: 3,4
Condiciones de aceptación: ▪ El usuario puede iniciar sesión en la aplicación. ▪ El usuario puede cerrar sesión en la aplicación.		

Tabla 5.11 - Historia de usuario 10 (Fuente: Elaboración propia)

ID:11	**Mostrar muro del usuario**	
Como cliente quiere que, los usuarios una vez iniciado sesión, puedan ver que las opciones del menú se han cambiado y que se muestran las publicaciones de los amigos del usuario. También podrán seguir el hilo de un comentario.		
Estimación: 80	Valor: 100	Dependencias: 4, 10
Condiciones de aceptación: ▪ El menú ofrece un abanico de opciones diferente al que ofrece si no ha iniciado sesión. ▪ Se muestran las publicaciones de los amigos del usuario. ▪ El usuario puede ver el hilo de un comentario.		

Tabla 5.12 - Historia de usuario 11 (Fuente: Elaboración propia)

En el segundo sprint se finalizó toda la parte pública de la aplicación, es decir, toda la parte donde no hacía falta iniciar sesión. Además, se ha implementado el muro que se encuentra en la parte privada de la aplicación.

En cuanto al tercer sprint, empezó el 21 de enero de 2021 y finalizó 21 de febrero de 2021. La pila del sprint tres contiene las historias de usuario que aparecen desde la Tabla 5.13 hasta la Tabla 5.18.

ID:12	Implementar un sistema de gestión de 'me gustas'	
Como cliente quiere que los usuarios puedan dar me gusta a las publicaciones que vean en pantalla. También que pueda retirar el me gusta.		
Estimación: 90	Valor: 100	Dependencias: 11
Condiciones de aceptación: ▪ El usuario puede dar me gusta a una publicación. ▪ El usuario puede retirar el me gusta de una publicación.		

Tabla 5.13 - Historia de usuario 12 (Fuente: Elaboración propia)

ID:13	Implementar un sistema de gestión de publicaciones	
Como cliente quiere que los usuarios puedan crear publicaciones nuevas o comentar sobre otras publicaciones en forma de hilo. Incluso el usuario debe poder eliminar sus publicaciones.		
Estimación: 40	Valor: 80	Dependencias: 11
Condiciones de aceptación: ▪ El usuario puede crear una publicación. ▪ El usuario puede comentar sobre otra publicación. ▪ El usuario puede borrar sus publicaciones.		

Tabla 5.14 - Historia de usuario 13 (Fuente: Elaboración propia)

ID:14	Implementar opción Perfil	
Como cliente quiere que los usuarios puedan acceder a su perfil donde podrán ver y gestionar todos los comentarios y me gustas que han realizado.		
Estimación: 40	Valor: 70	Dependencias: 4, 12,13
Condiciones de aceptación: ▪ El usuario puede acceder a su perfil. ▪ El usuario puede gestionar todos sus comentarios. ▪ El usuario puede gestionar todos sus me gustas.		

Tabla 5.15 - Historia de usuario 14 (Fuente: Elaboración propia)

ID:15	Implementar zonas de Avisos	
Como cliente quiere que los usuarios desde la opción de Perfil puedan gestionar las peticiones de amistad que le han enviado otros usuarios.		
Estimación: 30	Valor: 50	Dependencias: 14
Condiciones de aceptación: ▪ El usuario puede ver las peticiones de amistad recibidas. ▪ El usuario puede aceptar una petición de amistad recibida. ▪ El usuario puede rechazar una petición de amistad recibida.		

Tabla 5.16 - Historia de usuario 15 (Fuente: Elaboración propia)

ID:16	Implementar zona de Ajustes	
Como cliente quiere que los usuarios tengan la posibilidad de realizar ajustes en su perfil, su privacidad, cambiar datos personales e incluso cambiar su contraseña.		
Estimación: 40	Valor: 90	Dependencias: 14
Condiciones de aceptación: ▪ El usuario puede cambiar su foto de perfil. ▪ El usuario puede cambiar su configuración de privacidad. ▪ El usuario puede cambiar sus datos personales. ▪ El usuario puede cambiar su contraseña.		

Tabla 5.17 - Historia de usuario 16 (Fuente: Elaboración propia)

ID:17	Implementar zona del Progreso	
Como cliente quiere que, los usuarios puedan ver su historial de medidas, y sus historiales de hábitos alimenticios y de actividad física respectivamente. Además, podrá ver su estado actual en el sistema de BieneSTARS.		
Estimación: 30	Valor: 50	Dependencias: 4
Condiciones de aceptación:		

▪ El usuario puede ver su historial de medidas. ▪ El usuario puede ver su estado actual respecto a su historial de hábitos alimenticios. ▪ El usuario puede ver su estado actual respecto a su historial de actividad física.

Tabla 5.18 - Historia de usuario 17 (Fuente: Elaboración propia)

En el tercer sprint se ha implementado al completo la opción de Perfil con sus opciones de Ajustes y Avisos. Por otro lado, se ha añadido la zona de Progreso del usuario y también se ha implementado los dos sistemas de gestión de Me Gustas y de Publicaciones respectivamente. En cuanto al cuarto sprint, empezó el 22 de febrero de 2021 y finalizó 22 de marzo de 2021. La pila del sprint cuatro contiene las historias de usuario que aparecen desde la Tabla 5.19 hasta la Tabla 5.24.

ID:18	**Implementar zona de Búsqueda**	
Como cliente quiere que los usuarios puedan realizar búsquedas entre todos los usuarios a través de un filtro dinámico.		
Estimación: 40	Valor: 50	Dependencias: 4
Condiciones de aceptación: ▪ El usuario puede buscar entre todos los usuarios de la aplicación. ▪ El usuario puede filtrar dinámicamente.		

Tabla 5.19 - Historia de usuario 18 (Fuente: Elaboración propia)

ID:19	**Implementar zona y sistema de gestión de Amigos**	
Como cliente quiere que los usuarios puedan enviar solicitudes de amistad a otros usuarios y que también puedan visualizar y borrar amigos que tengan agregados.		
Estimación: 50	Valor: 70	Dependencias: 14, 18
Condiciones de aceptación:		

▪ El usuario puede enviar solicitudes de amistad. ▪ El usuario puede visualizar amigos. ▪ El usuario puede eliminar amigos.

Tabla 5.20 - Historia de usuario 19 (Fuente: Elaboración propia)

ID:20	**Implementar sistema de gestión de Eventos**	
Como cliente quiere que los usuarios puedan crear eventos nuevos, participar en otros eventos públicos, gestionar su asistencia a los eventos y adjuntar una foto a los eventos asistidos.		
Estimación: 90	Valor: 80	Dependencias: 4
Condiciones de aceptación: ▪ El usuario puede visualizar eventos futuros y pasados. ▪ El usuario puede crear un evento. ▪ El usuario puede confirmar su asistencia a un evento. ▪ El usuario puede cancelar su asistencia a un evento. ▪ El usuario puede adjuntar una foto como demostración de su asistencia a un evento pasado.		

Tabla 5.21 - Historia de usuario 20 (Fuente: Elaboración propia)

ID:21	**Implementar zona de Chat**	
Como cliente quiere que los usuarios puedan comunicarse de forma directa y privada con sus amigos, es decir, a través de un chat privado.		
Estimación: 50	Valor: 60	Dependencias: 4
Condiciones de aceptación: ▪ El usuario puede chatear directamente con sus amigos.		

Tabla 5.22 - Historia de usuario 21 (Fuente: Elaboración propia)

ID:22	Implementar sistema de Recomendados	
Como cliente quiere que los usuarios puedan visualizar usuarios recomendados por la aplicación.		
Estimación: 20	Valor: 20	Dependencias: 4, 19
Condiciones de aceptación: ▪ El usuario puede ver su historial de medidas. ▪ El usuario puede ver su estado actual respecto a su historial de hábitos alimenticios. ▪ El usuario puede ver su estado actual respecto a su historial de actividad física.		

Tabla 5.23 - Historia de usuario 22 (Fuente: Elaboración propia)

ID:23	Implementar accesorios a las publicaciones	
Como cliente quiero que, los usuarios puedan ver el número de me gustas y de respuestas que tiene una publicación.		
Estimación: 20	Valor: 40	Dependencias: 11
Condiciones de aceptación: ▪ El usuario puede ver el número de me gustas de una publicación. ▪ El usuario puede ver el número de respuestas de una publicación.		

Tabla 5.24 - Historia de usuario 23 (Fuente: Elaboración propia)

En este cuarto y último sprint se ha implementado el sistema de gestión de amigos y el sistema de gestión completo de los eventos. Además, se ha implementado el sistema de búsqueda de usuarios junto con la zona de chat y la zona de usuarios recomendados. Como última implementación se añadió los accesorios a las publicaciones que permiten al usuario ver cuántos Me Gustas tiene una publicación y cuántas respuestas.

En conjunto se han llevado a cabo 23 historias de usuario. No es de extrañar que el número de historias de usuario sea menor que el número total de requisitos especificados en el capítulo 3 porque una historia de usuario puede englobar varios requisitos.

5.1.5 Reuniones

Las reuniones es un factor fundamental de la metodología Scrum, ya que favorecen la comunicación entre todas las partes interesadas que conforman el proyecto. Se han realizado tres tipos de reuniones diferentes:

- Reuniones de planificación del sprint: se realizan antes de empezar el sprint y se utilizan para hablar acerca de qué historias de usuarios se introducirán en el sprint. Su duración ronda entre una hora por cada semana que conlleve el sprint. En esta reunión se pueden distinguir claramente dos partes. En la primera parte se decide qué historias de usuarios se van a introducir en el sprint, donde el dueño del producto es una figura indispensable, ya que es el encargado de revisar con detenimiento las historias más importantes.

  La segunda parte está más enfocada en cómo se va a llevar a cabo todas las historias de usuario. El acta de reunión de planificación de cada sprint se muestra desde la Tabla 5.25 hasta la Tabla 5.28 respectivamente.

Acta de reunión de planificación del sprint 1
N.º sprint: 1
Fecha inicio: 18/11/2020

Objetivos del sprint:

- Obtener programas/drivers
- Menú lateral desplegable
- Base de datos
- Ventana portada
- Ventana proyecto

Tabla 5.25 - Acta de reunión de planificación del sprint 1 (Fuente: Elaboración propia)

Acta de reunión de planificación del sprint 2
N.º sprint: 2
Fecha inicio: 21/12/2020

Objetivos del sprint:

- Ventana actívate
- Ventana nutriconsejos
- Ventana blog
- Ventana inicio de sesión
- Ventana muro

Tabla 5.26 - Acta de reunión de planificación del sprint 2 (Fuente: Elaboración propia)

Acta de reunión de planificación del sprint 3

N.º sprint: 3
Fecha inicio: 21/01/2021

Objetivos del sprint:

- Sistema completo de gestión de Me Gustas.
- Sistema completo de gestión de publicaciones.
- Ventana perfil
- Ventana avisos.
- Ventana ajustes.
- Ventana progreso.

Tabla 5.27 - Acta de reunión de planificación del sprint 3 (Fuente: Elaboración propia)

Acta de reunión de planificación del sprint 4

N.º sprint: 4
Fecha inicio: 22/02/2020

Objetivos del sprint:

- Ventana búsqueda.
- Sistema completo de gestión de amigos.
- Sistema completo de gestión de eventos.
- Ventana chat.
- Ventana recomendaciones.
- Accesorios de publicaciones.

Tabla 5.28 - Acta de reunión de planificación del sprint 4 (Fuente: Elaboración propia)

- Reuniones de revisión del sprint: Estas reuniones se realizan cuando el sprint ha finalizado. El objetivo principal de estas reuniones es analizar el incremento generado en el sprint y, si fuese necesario, adaptar la pila del producto. La duración estimada de estas reuniones es de una hora por cada semana que conlleve el sprint. El acta de reunión de revisión de cada sprint se muestra desde la Tabla 5.29 hasta la Tabla 5.32 respectivamente.

Acta de reunión de revisión del sprint 1

N.º sprint: 1
Fecha inicio: 21/12/2020

Objetivos del sprint:

- Saber qué es lo que se ha finalizado en el sprint 1
- Saber qué es lo que no se ha terminado en el sprint 1

Se ha podido llegar a realizar todas las historias de usuario introducidas en la pila de elementos del sprint. La velocidad del sprint 1 es de 210.

Como se han cumplido todos los objetivos del sprint 1, la velocidad real del sprint 1 es 210.

Tabla 5.29 - Acta de reunión de revisión del sprint 1 (Fuente: Elaboración propia)

Acta de reunión de revisión del sprint 2

N.º sprint: 2
Fecha inicio: 21/01/2021

Objetivos del sprint:

- Saber qué es lo que se ha finalizado en el sprint 2
- Saber qué es lo que no se ha terminado en el sprint 2

La implementación de un blog ha sido rechazada durante el sprint por el dueño del producto. La razón principal es que los datos almacenados en la base de datos para mostrar en el blog no están almacenados de forma flexible para ser expuesto en multiplataformas. Requiere un cambio estructural de la base de datos por parte del dueño del producto.

Como se ha rechazado esa historia de usuario durante el sprint, se considera que se ha eliminado del proyecto por lo que no contará en la suma final de la velocidad del sprint, es decir, es como que no hubiera existido ese requisito. Teniendo esto en

cuenta, se ha podido llegar a realizar todas las historias de usuario introducidas en la pila de elementos del sprint. La velocidad del sprint 2 es de 190. Como se han cumplido todos los objetivos del sprint 2, la velocidad real del sprint 2 es 190.

Tabla 5.30 - Acta de reunión de revisión del sprint 2 (Fuente: Elaboración propia)

Acta de reunión de revisión del sprint 3

N.º sprint: 3
Fecha inicio: 22/02/2021

Objetivos del sprint:

- Saber qué es lo que se ha finalizado en el sprint 3
- Saber qué es lo que no se ha terminado en el sprint 3

Se ha podido llegar a realizar todas las historias de usuario introducidas en la pila de elementos del sprint. La velocidad del sprint 3 es de 270.

Como se han cumplido todos los objetivos del sprint 3, la velocidad real del sprint 3 es 270.

Tabla 5.31 - Acta de reunión de revisión del sprint 3 (Fuente: Elaboración propia)

Acta de reunión de revisión del sprint 4

N.º sprint: 4
Fecha inicio: 23/03/2021

Objetivos del sprint:

- Saber qué es lo que se ha finalizado en el sprint 4
- Saber qué es lo que no se ha terminado en el sprint 4

Se ha podido llegar a realizar todas las historias de usuario introducidas en la pila de elementos del sprint. La velocidad del sprint 4 es de 270.

Como se han cumplido todos los objetivos del sprint 4, la velocidad real del sprint 4 es 270.

Tabla 5.32 - Acta de reunión de revisión del sprint 4 (Fuente: Elaboración propia)

- Reuniones de retrospectiva del sprint: El objetivo principal de estas reuniones es realizar una revisión de lo que ha ocurrido en el sprint, donde el equipo analiza aspectos de carácter operativos, es decir, relacionados con los propios de procesos y el entorno. Se busca obtener un plan de mejora o dar respuesta a qué se puede mejorar respecto al sprint anterior. La duración estimada para estas reuniones es de cuarenta y cinco minutos por cada semana que conlleve el sprint y se lleva a cabo justo después de la reunión de revisión del sprint. En esta reunión cada integrante del equipo de desarrollo pone en común qué piensa que ha ido bien y qué cosas son mejorables sin ser interrumpidos. Casi al final de esta reunión, el Scrum Master trata de recoger todas las sugerencias que se han dado para tenerlas en cuenta. El acta de reunión de retrospectiva de cada sprint se muestra desde la Tabla 5.33 hasta la Tabla 5.36 respectivamente.

Acta de reunión de retrospectiva del sprint 1
N.º sprint: 1
Fecha inicio: 21/12/2021

Objetivos del sprint:

- Mejoras para el futuro
- Posibles obstáculos a tener en cuenta

Debido a la pandemia del Covid-19 se ha hecho vital el uso de Google Meet como herramienta principal para realizar videollamadas y de esta forma mantener la comunicación directa con el dueño del producto. También el uso de correos electrónicos ha servido para mantener una conversión cuyo contenido era de vital importancia que quedara escrito.

Tabla 5.33 - Acta de reunión de retrospectiva del sprint 1 (Fuente: Elaboración propia)

Acta de reunión de retrospectiva del sprint 2
N.º sprint: 2
Fecha inicio: 21/01/2021

Objetivos del sprint:

- Mejoras para el futuro

- Posibles obstáculos a tener en cuenta

Para evitar futuros problemas se debe inspeccionar de antemano todas las estructuras que conforman la base de datos, de esta forma no sería necesario tomar decisiones en mitad del trabajo.

Tabla 5.34 - Acta de reunión de retrospectiva del sprint 2 (Fuente: Elaboración propia)

Acta de reunión de retrospectiva del sprint 3

N.º sprint: 3
Fecha inicio: 22/02/2021

Objetivos del sprint:

- Mejoras para el futuro
- Posibles obstáculos a tener en cuenta

Se ha podido mejorar los tiempos de implementación de historias de usuario que conllevan los procesos involucrados. Esto se debe a que a partir de la finalización del sprint 2 la carga de trabajo externa al proyecto se ha vista reducida bastante, facilitando así, las implementaciones del sprint.

Tabla 5.35 - Acta de reunión de retrospectiva del sprint 3 (Fuente: Elaboración propia)

Acta de reunión de retrospectiva del sprint 4

N.º sprint: 4
Fecha inicio: 23/03/2021

Objetivos del sprint:

- Mejoras para el futuro
- Posibles obstáculos a tener en cuenta

No se han observado obstáculos durante el proceso. La única mejora que se puede aplicar es aplicar en el futuro el hábito de realizar y adelantar trabajo.

Tabla 5.36 - Acta de reunión de retrospectiva del sprint 4 (Fuente: Elaboración propia)

5.2 Controlador de versiones

5.2.1 Definición y motivos

En todo proyecto cuya finalidad es desplegar un producto software, es buena práctica distinguir dos tipos de gestión:

- Gestión de los procesos del proyecto: este tipo de gestión engloba desde que el producto final es una idea hasta la entrega de cada incremento, es decir, se enfoca en la planificación y administración de los recursos, actividades y tiempo.
- Gestión del software: se necesita un sistema de gestión para cada entregable o incremento software. Es por ello que se hace uso de un controlador de versiones.

Un controlador de versión es un sistema de gestión de código que permite llevar un registro de todos los cambios y/o incrementos que se van añadiendo al producto al finalizar cada iteración **[19]**. El equipo de desarrollo es el encargado de llevar un registro de todo el software del proyecto. Los cambios que se pueden registrar pueden llegar a ser desde un cambio en el nombre de una variable hasta la implementación completa de una funcionalidad, pero se aconseja realizar un registro actual del software cuando se implementa una funcionalidad importante y que se ha comprobado que funciona correctamente. Otra gran ventaja que ofrecen los controladores de versiones es que permite comparar con otras versiones anteriores del software, mostrando así qué cambios se han realizado entre versiones y quién las ha realizado. Para este proyecto, como se mostró en el capítulo de tecnologías, se ha utilizado Github como controlador de versiones y repositorio remoto. El motivo principal es la facilidad de uso que ofrece esta herramienta a los desarrolladores. Además, es una de las herramientas más conocidas en el ámbito del desarrollo software y a su vez es utilizada también como método de reclutamiento por parte de empresas del sector privado.

Github, dentro de los diferentes tipos de controlador de versiones, se clasifica como un controlador de versiones distribuido basado en Git. El funcionamiento de este tipo se explica a continuación. En la parte del servidor funciona como repositorio y el desarrollador no descarga la instantánea más actual de los archivos, más bien copia íntegramente el

repositorio de forma completa en el equipo. Este tipo de controlador de versiones en concreto es muy favorable para el desarrollador debido a que en caso de que el servidor deje de funcionar o se vuelva fuera de servicio y se tenga que cambiar, el desarrollador podrá clonar, del repositorio local que tiene copiado en su equipo al nuevo servidor en remoto, todo el proyecto software. En la Figura 5.2 se muestra de forma gráfica como está estructurado un controlador de versiones distribuido.

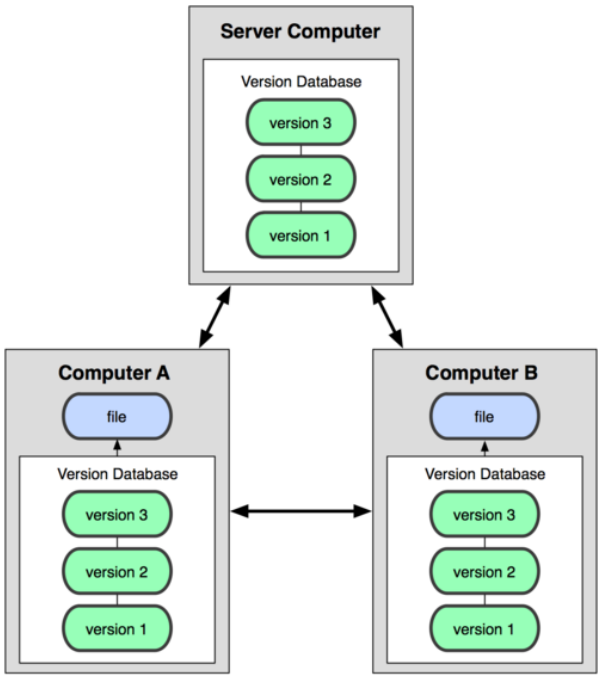

Figura 5.2 - Estructura de un controlador de versiones distribuido (Fuente: [20])

5.2.2 Ramas

El controlador de versiones utilizado en este proyecto permite trabajar con ramas. Una rama es un puntero que puede moverse y que apunta a una de las confirmaciones de los cambios en los archivos **[21]**. Las ramas son muy útiles cuando en un proyecto están trabajando varias personas y todas ellas están desarrollando código para partes del programa que son independientes entre sí. Al principio de un proyecto existe solo una única rama denominada “master”. El funcionamiento del sistema de ramas es el siguiente:

Cada confirmación que se realiza, Git, que es la base de GitHub, guarda una instantánea del trabajo preparado almacenando información acerca del autor junto con un mensaje explicativo y a mayores los punteros a las confirmaciones anteriores a la que se acaba de realizar. La Figura 5.3 es una representación donde se puede observar bien cómo se trabaja con dos ramas.

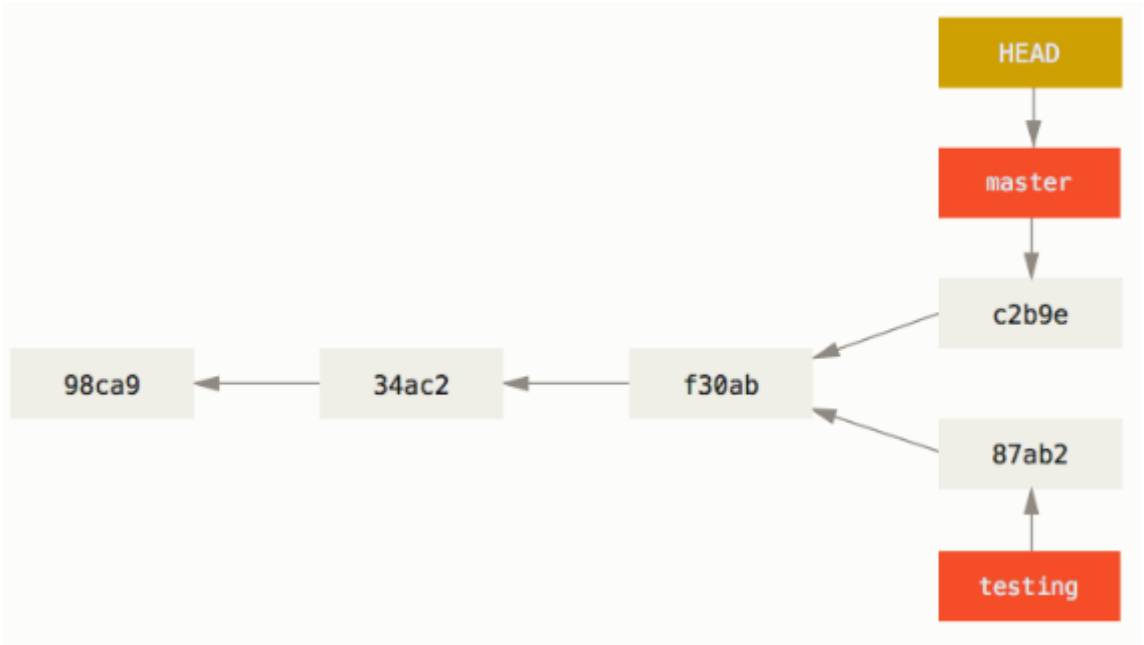

Figura 5.3 - Uso de ramas en un proyecto Git (Fuente:[21])

El puntero que aparece con el nombre de HEAD muestra en qué rama está el desarrollador trabajando actualmente. Las ramas permiten trabajar a varios desarrolladores en diferentes cosas al mismo tiempo y también permite al programador desarrollar y testear una implementación del producto final sin tener riesgos de provocar errores en otras implementaciones del producto final. Una vez que se ha testeado que la implementación es correcta y funciona bien, como se ha desarrollado en una rama, se puede fusionar a la rama master para que todos los desarrolladores puedan trabajar sobre el incremento del producto.

Para este proyecto he hecho uso de 5 ramas aparte de la master, donde han servido de gran ayuda a la hora de implementar funcionalidades complejas. Para la parte pública de

la aplicación se utilizó una rama para cada zona. Posteriormente, para la parte privada de la aplicación se ha utilizado la gestión de ramas recomendadas del artículo “A successful Git Branching Model” **[22]**. Las ramas utilizadas se muestran en la Figura 5.4.

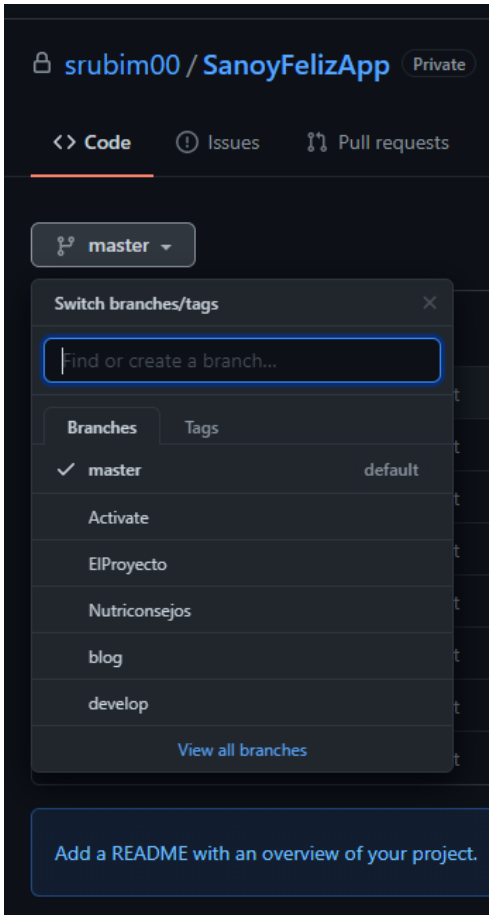

Figura 3.4 - Número de ramas en el repositorio (Fuente: Elaboración propia)

6. Presupuesto

El presupuesto establecido para el desarrollo completo del trabajo se desglosa en dos tablas donde en una se detalla los costes asociados a la mano de obra, es decir, todo lo referente a los diferentes perfiles de informáticos que han trabajado dentro del proyecto. En la otra tabla se detalla los costes relacionados con materias primas, software y otros elementos que no están relacionados con las personas.

6.1 Mano de obra

Este presupuesto se ha realizado para el TFM, por lo que soy el único integrante que compone al equipo de desarrollo. Sin embargo, tal y como se hizo mención en la introducción de este apartado 6, se realizan diferentes labores, es decir, tomaré diferentes roles dentro de la metodología scrum, pero también realizaré tareas de otros perfiles de informáticos. En este caso se han encontrados 3 tipos de perfiles que hacen faltan para llevar a cabo el proyecto y los salarios medios que perciben personas con la preparación y el título que les acredita para realizar esa labor. En primer lugar, habrá un programador con experiencia en desarrollo de aplicaciones Android (≈ 27.000€/año) **[23]**, un scrum master (≈ 31.000€/año) **[24]** y finalmente un diseñador gráfico (≈ 21.000€/año) **[25]**. Se entiende que el programador trabajará un total de cuatro horas diarias durante veinte días laborales al mes, es decir, descansando sábados y domingos. El Scrum Master trabajará aproximadamente un tercio de tiempo que el programador y el diseñador trabajará la mitad del tiempo que el programador, es decir, dos horas al día. El presupuesto referente a este subapartado se muestra en la Tabla 6.1.

Tipo de mano de obra	Cantidad [horas]	Coste unitario [€/h]	Coste total [€]
Programador Android	320	14,00	4.480,00
Scrum master	100	16,00	1.600,00

Diseñador	160	11,00	1.760,00
Subtotal			**7.840,00**

Tabla 6.1 - Presupuesto de mano de obra (Fuente: Elaboración propia)

6.2 Materiales y elementos

En este otro subapartado se tiene en cuenta todos aquellos elementos, ajenos a la mano de obra, que se necesitan para poder realizar el proyecto. Es por ello por lo que el presupuesto referente a esta parte contendrá el equipo/ordenador comprado específicamente para el proyecto donde se desarrolla toda la aplicación, las licencias no gratuitas de software, etc. Por último, también se tendrá en cuenta gastos de electricidad y contratación de fibra de internet.

En cuanto al precio del Kwh de la línea de corriente eléctrica, aunque su precio fluctúa cada hora y cada día, se establece un precio medio de 0,19€/Kwh **[26]**. Para el internet se contrata fibra óptica de 600Mbps por 21€/mes.

Material/licencia utilizados en el proyecto	Concepto	Cantidad	Importe total [€]
Ordenador	Ordenador portátil	1	1.750,00
Licencia Office	Licencia Microsoft Office	2	99,00
Electricidad	Línea de corriente eléctrica	1	60,80
Internet	Tarifa de internet proveída por Vodafone	1	84,00
Subtotal			1.993,80

Tabla 6.2 - Presupuesto de materiales y licencias (Fuente: Elaboración propia)

6.3 Presupuesto final

Concepto	Importe [€]	Importe total [letra]
Mano de obra	7.840,00	Siete mil ochocientos cuarenta euros
Material y licencias utilizados en el proyecto	1.993,80	Mil novecientos noventa y tres euros con ochenta céntimos
PRESUPUESTO TOTAL	9.833,80	Nueve mil ochocientos treinta y tres euros con ochenta céntimos

Tabla 6.3 - Presupuesto total del proyecto (Fuente: Elaboración propia)

7. Desarrollo de la aplicación

7.1 Entorno de desarrollo

Es evidente las diferentes ventajas que tiene Android Studio frente a otros programas de desarrollo de aplicaciones móviles. En este entorno de desarrollo existen dos lenguajes de programación diferentes que se pueden utilizar, Java y Kotlin. La ventaja que ofrece Kotlin respecto a Java es que permite realizar las mismas funciones de Java, pero en menos líneas de código generalmente. No obstante, existe un cambio bastante claro en la forma de programar en Kotlin respecto de Java. Además, actualmente existe mucha más documentación para programar en Java, por lo que existe una barrera bastante grande para poder cambiar de la programación Java a la programación Kotlin. Otra de las razones por las que la aplicación final se programa en Java es porque se tienen unas bases bastante bien asentadas de Java, ya que ya he realizado otros proyectos Android en Java.

7.2 Arquitectura de la aplicación

La arquitectura utilizada en este proyecto es una de las más utilizadas a nivel mundial, se trata de la arquitectura cliente-servidor. En esta arquitectura, los clientes, que son los dispositivos Android de las personas, realizan las peticiones al servidor para que éste les devuelva datos o cualquier otro tipo de información que la aplicación requiera **[27]**.

En la Figura 7.1 se muestra de forma más visual cómo funciona esta arquitectura.

Para la realización de este proyecto, el grupo de investigación ha facilitado el acceso a su servidor, donde está almacenada la base de datos.

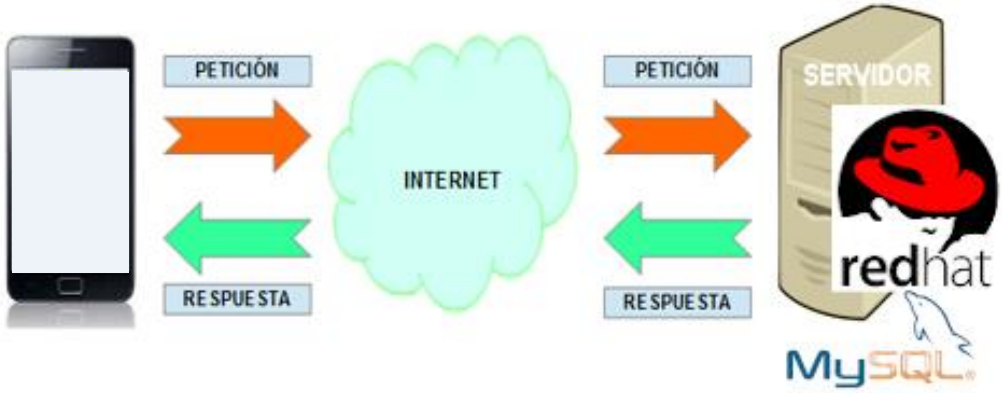

Figura 7.1 - Arquitectura cliente-servidor (Fuente: Elaboración propia)

7.3 Base de datos

Debido a que este proyecto consiste en migrar una aplicación web a una aplicación Android, ya existe una base de datos creada y además rellena de datos importante para permitir el buen funcionamiento del sistema. A continuación, se explica de forma detallada todo lo relacionado con las bases de datos, concretamente desde dos perspectivas de vital importancia, el diseño interno que compone la base de datos y por otro lado cómo se puede implementar la programación necesaria para que la aplicación móvil pueda realizar peticiones a la base de datos del servidor.

7.3.1 Diseño de la base de datos

Como gestor de bases de datos se utiliza MySQL, debido a que la base de datos originalmente está almacenada e implementada en el servidor Ubuntu del grupo de investigación. A continuación, en la Figura 7.2 se muestra un diagrama relacional exportado desde el gestor de MySQL. Obviamente, no se corresponde con el diagrama relacional completo de la aplicación web, ya que en la Figura 7.2 solo se tiene en cuenta todas las tablas que han sido requeridas y utilizadas para el desarrollo de la aplicación móvil, ya bien sea para realizar funciones necesarias como acceder a información, editar o directamente eliminar un dato de la base de datos.

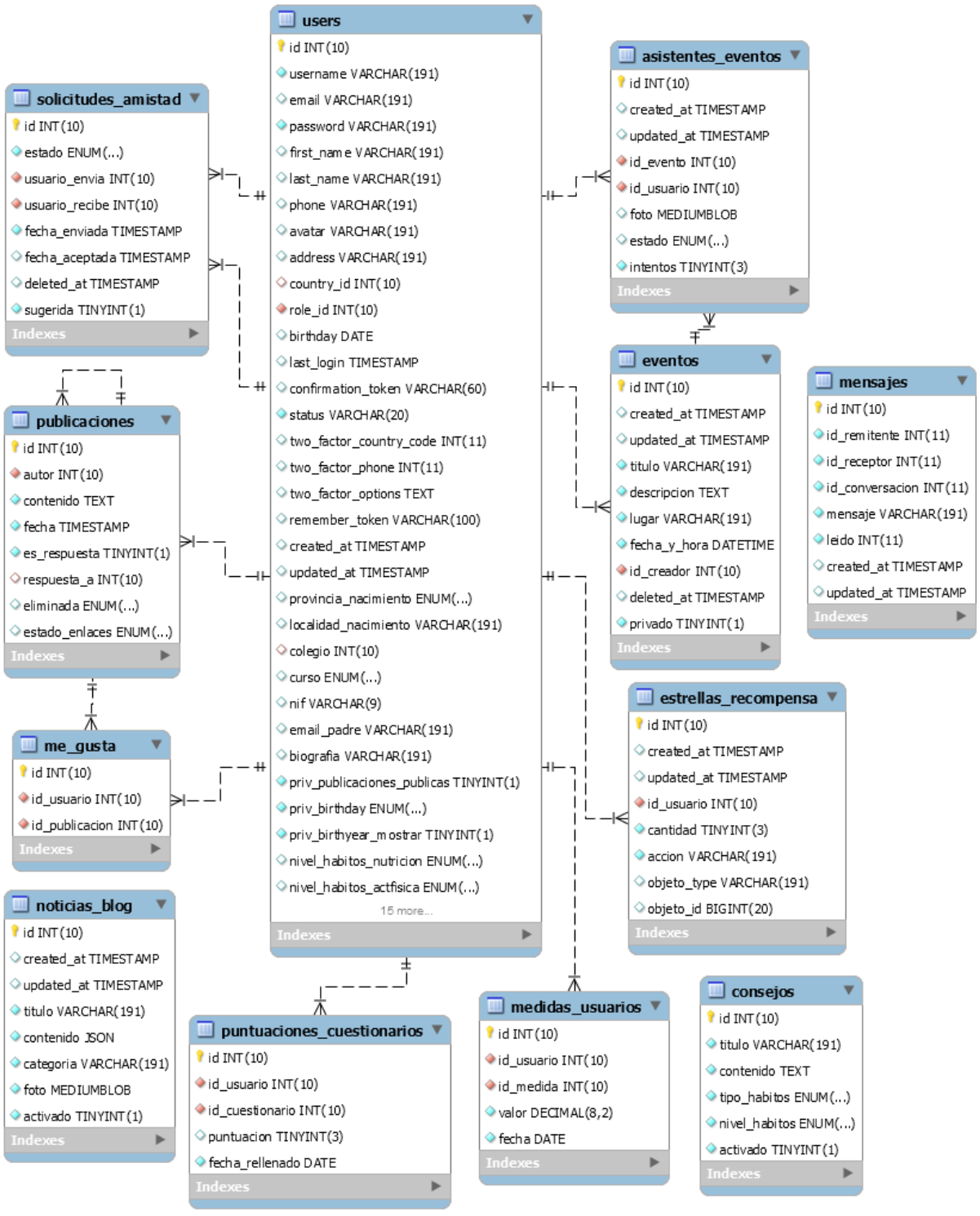

Figura 4.2 - Diagrama relacional de la base de datos (Fuente: Elaboración propia)

Una vez observado las tablas utilizadas en el proyecto en la Figura 7.2, se detalla a continuación para qué sirve cada tabla:

- La tabla **Consejos** almacena información acerca de los consejos relacionados con la nutrición y la actividad física. Esta tabla permite mostrar dichos datos en el apartado de Nutriconsejos y Actívate de la aplicación.
- La tabla **Noticias_blog** almacena información acerca de noticias que puede ser útiles al usuario. Esta tabla se iba a utilizar para la zona de Blog.
- La tabla **Users** es la tabla más grande de la base de datos y es la que más relaciones tiene con otras tablas. En esta tabla se almacena todo lo relacionado con los usuarios que representan a las personas en la red social. Entre los datos almacenados se encuentran datos de identificación del usuario incluyendo el propio nombre de usuario y contraseña para el inicio de sesión. Además, esta tabla guarda las opciones de privacidad establecidas por el usuario y los datos privados modificables.
- La tabla **Eventos** almacena toda la información relevante acerca de los eventos creados. Alberga información detallada acerca de la configuración del evento y quién lo creó.
- La tabla **Asistentes_eventos** almacena la información acerca de qué usuarios han asistido o asistirán a los eventos creados. También almacena la foto que puede subir el usuario que ha participado en un evento pasado concreto para verificar que estuvo en el evento.
- La tabla **Publicaciones** almacena información acerca de todas las publicaciones que han sido realizadas por los usuarios, ya bien sea publicaciones nuevas o respuestas a otras publicaciones. A su vez, guarda quién es el que ha realizado la publicación.
- La tabla **Me_gusta** almacena todo los me gustas que se han dado a todas las publicaciones existentes. Sirve para saber si un usuario ha dado me gusta a una publicación o cuántos me gustas tiene una publicación concreta.
- La tabla **Solicitudes_amistad** almacena todas las peticiones de amistad que han sido enviadas entre los usuarios y el estado actual de estas peticiones utilizando la marca de tiempo.

- La tabla **Estrellas_recompensa** almacena todas las estrellas que ha conseguido el usuario a través del sistema de BieneSTARS.
- La tabla **Medidas_usuario** almacena información acerca de las medidas corporales del usuario. Son las medidas que se mostrarán al usuario en el apartado de Progreso, en la zona privada.
- La tabla **Puntuaciones_cuestionarios** almacena la puntuación que han recibido los usuarios cuando han realizado los cuestionarios relacionados con la nutrición y el ejercicio físico.
- La tabla **Mensajes** almacena información acerca de todas las conversaciones que mantenido los usuarios que son amigos entre sí dentro de la aplicación a través de la zona de Chat.

7.3.2 Conexión a base de datos

7.3.2.1 Archivo manifest

Para poder establecer una conexión desde la aplicación móvil a la base de datos del servidor se debe, en primer lugar, descargar e importar el conector que ofrece la propia impresa que está detrás de MySQL. Para este proyecto se utiliza concretamente el conector MySQL-Java con la versión 5.1.48. Además, para que la aplicación pueda acceder a internet, se debe modificar el archivo "Manifest" de la aplicación, el cual contiene toda la configuración de la aplicación en cuanto a la relación hardware-software se refiere. En este archivo mencionado recientemente debemos poner las dos siguientes líneas que aparecen en la Figura 7.3:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
```

Figura 7.3 - Configuración adicional en el archivo Manifest (Fuente: Elaboración propia)

La primera línea que aparece en la Figura 7.3 permite a la aplicación de internet conectarse a internet y la segunda línea permite leer archivos externos a la aplicación.

7.3.2.2 Clase mysql.java

Una vez que la aplicación ya está disponible para establecer una conexión, se debe crear una clase Java. En el caso de este proyecto se denomina Mysql.java. En esta clase se debe crear un método que será el encargado de ser el receptor de las solicitudes de la aplicación cuando se desea acceder a la base de datos. Pueden existir varios métodos, tantos como se desee, ya que cada método estará enfocado para una solicitud en concreto. Dentro de la clase donde están todos los métodos, se crean clases internas. Entonces la clase Mysql.java está compuesta de métodos y debajo de la zona de los métodos, simplemente para dar claridad al archivo, se establecen las clases que son las encargadas de acceder directamente a la base de datos. Se puede ver la disposición del archivo en la Figura 7.4:

```
public void guardarInicioUser(String user_id){

    InicioUser inicio_u=new InicioUser();
    try {

        inicio_u.execute(user_id).get();

    } catch (ExecutionException e) {
        e.printStackTrace();
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

class InicioUser extends AsyncTask<String, Void, String> {
    Connection connection;
    String error="";

    @Override
    protected String doInBackground(String... array) {

        String id_user=array[0];
        try
        {
            Class.forName("com.mysql.jdbc.Driver");
            connection = DriverManager.getConnection( url:
                                                      user:
                                                  password:
                                                    );
            Statement statement = connection.createStatement();
            int resultSet = statement.executeUpdate( s
        }
        catch(Exception e)
        {
            System.out.println("MAL:::"+e);
            error = e.toString();
        }
        return "";
    }
}
```

Figura 7.4 - Parte de la clase Mysql.java (Fuente: Elaboración Propia)

Para explicar el funcionamiento correcto de este sistema se toma como ejemplo el querer guardar el último momento que un usuario accedió a su cuenta dentro de la aplicación. Primer lugar, cuando el cliente se encuentra en la ventana de iniciar sesión, rellena tanto el cambio usuario como contraseña y hace clic en el botón “Entrar”, la aplicación lo que está haciendo es, una comprobado de que las credenciales son correctas. Entonces llama al método “guardarInicioUser” que recibe el id del usuario. Dentro de este método se crea un objeto o instancia de la clase interna que está dentro del mismo fichero que el método, en este caso la clase interna se denomina “InicioUser”.

7.3.2.3 Clases internas con herencia de asynctask

Ahora bien, como se puede ver, estas clases internas heredan de “AsyncTask”, es decir, todo lo que se ejecute en la clase se realiza a través de un hilo destinado para eso exclusivamente. Es por ello que en el método “guardarInicioUser” se utiliza la función “execute(user_id).get()”. Esto es debido a que, si no se utilizara ese tipo de clase multitarea o multihilo, el usuario no podría interactuar con la pantalla debido a que el hilo principal está ocupado comunicándose con el servidor remoto y solicitando realizar alguna operación en la base de datos, es decir, el hilo principal estaría siendo monopolizado por los accesos a base de datos **[28]**. En cambio, al establecer estas clases internas con su herencia de “AsyncTask”, se evita este problema, ya que los hilos se encargan de acceder a base de datos y el hilo principal se centra en la aplicación principal, por ejemplo, estando alerta para cuando el usuario hace clic en un botón o cuando genera algún tipo de evento que requiere ser atendido.

Las clases que heredan de “AsyncTask” tienen varios métodos que se pueden sobrescribir. No obstante, el método más importante y que es necesario en esta aplicación es el método “doInBackground”. En este método se puede hacer todo lo que se necesite en el hilo en segundo plano. Este método es de tipo protegido y puede devolver varios tipos de elementos. Como se puede ver en la Figura 7.4, el argumento del método es un array de un mismo tipo de variable. En esto caso como solo mandamos una cadena de texto que contiene el número del usuario, se especifica que recibe un array de cadenas de texto.

Una vez dentro del método protegido se almacena la variable pasada como argumento y se define el conector a bases de datos de MySQL, como aparece dentro del bloque de try-catch. Se establece la IP del servidor junto con el nombre de la base de datos, el usuario de la base de datos y la contraseña de acceso al servidor de MySQL. Posteriormente se crea un "statement" que permite ejecutar directamente sentencias de MySQL. Por razones de privacidad y seguridad se ha decidido pixelar tanto las credenciales como la sentencia de MySQL.

7.4 Clase de tipo interface

Las clases de tipo "interface" son aquellas que son absolutamente abstractas y que están formadas por propiedades constantes y una serie de métodos abstractos, es decir, que los métodos están vacíos y solo aparecen con su signatura, con el nombre de variable del argumento del método y su tipo. **[29]**. Por lo tanto, esta peculiaridad que ofrecen este tipo de clases es muy útil en la aplicación. En la Figura 7.5 se muestra la clase desarrollada de tipo interface:

```
public interface CallBackInterface {
    public void callBackMethod(Usuarios user);
    public void callBackImage(Bitmap foto,String hel);
}
```

Figura 7.5 - Clase Interface de la aplicación (Fuente: Elaboración propia)

En la Figura 7.5 se puede ver que la clase está compuesta por dos métodos:

- *CallBackMethod*: este método se utiliza en el momento en el que el usuario inicia sesión porque es necesario que se devuelva a la clase Inicio toda la información del usuario que ha iniciado sesión. Esto se realiza a través del argumento "user". En esa variable se puede almacenar toda la información del usuario necesario para el buen funcionamiento de la aplicación. Para que las llamadas al método de la clase interface sean correctas se procede a explicar a continuación cómo se debe hacer.

En la clase inicio, en la signatura de la clase se añade la implementación de la interface CallBackInterface como se puede ver en la figura 7.6.

```
public class Inicio extends AppCompatActivity
        implements NavigationView.OnNavigationItemSelectedListener,CallBackInterface{

    private DrawerLayout drawerLayout;
    private ActionBarDrawerToggle actionBarDrawerToggle;
    private Toolbar toolbar;
```

Figura 7.6 - Clase Inicio implementa CallBackInterface (Fuente: Elaboración propia)

Se establece que se quiere utilizar dicha interface dentro del código que comunica las ventanas de Inicio e Inicio de Sesión como se puede ver en la Figura 7.7.

```
case R.id.iniciarsesion:

    IniciarSesion iniciar_sesion=new IniciarSesion();
    fragmentManager=getSupportFragmentManager();
    fragmentTransaction=fragmentManager.beginTransaction();
    iniciar_sesion.setCallBackInterface(this);
    fragmentTransaction.replace(R.id.frgprincipal,iniciar_sesion);
    fragmentTransaction.commit();
    break;
```

Figura 7.7 - Definición de la interface en clase Inicio (Fuente: Elaboración propia)

Más adelante, dentro de la clase Inicio se añade el método de la interface y se rellena para que se guarde el valor del usuario como se puede ver en la Figura 7.8, concretamente se ve en la zona verde remarcada.

```
@Override
public void callBackMethod(Usuarios user) {
    user_logged=user;
    View headerView =navigationView.getHeaderView( index: 0);
    Mysql mysql_call=new Mysql();
```

Figura 7.8 - Método callBackMethod de Inicio (Fuente: Elaboración propia)

Solo queda establecer la interface en la clase donde se inicia sesión por parte del usuario que se establece de la forma que se muestra en la Figura 7.9.

```
public void setCallBackInterface(CallBackInterface callBackInterface){
    this.callBackInterface=callBackInterface;
}
```

Figura 7.9 - Llamada a la interface desde donde se inicia sesión (Fuente: Elaboración propia)

Posteriormente cuando dentro de la clase se obtiene el dato que se quiere devolver a través de la interface se debe programar como aparece en la Figura 7.10.

```
if(user_logged!=null){
    if(callBackInterface!=null){
        callBackInterface.callBackMethod(user_logged);

    }
```

Figura 7.10 - Devolver dato desde Inicio de Sesión a Inicio (Fuente: Elaboración propia)

- *CallBackImage*: es el segundo método que contiene la clase interface. La función principal de este método es devolver la imagen que el usuario ha seleccionado de su galería para colocarla en un "ImageView". En este caso las clases involucradas son el adaptador de eventos y la clase galería.

  De nuevo, se vuelve a indicar en la clase del adaptador que implementa la interface tal y como se vio en la Figura 7.6. Luego se rellena el método callBackImage, donde se obtiene la foto final, como se puede ver en la Figura 7.11.

```
@Override
public void callBackImage(Bitmap foto,String hola) {

    bitmap=foto;

    ByteArrayOutputStream bos=new ByteArrayOutputStream();
    bitmap.compress(Bitmap.CompressFormat.PNG,  quality: 100, bos);
    byte[] bArray = bos.toByteArray();
```

Figura 7.11 - CallBackImage en el adaptador de eventos (Fuente: Elaboración propia)

Además, se debe realizar la llamada al método que ofrece dicha clase para establecer la conexión a través de la interface como se muestra en la Figura 7.12.

```
Galeria galeria=new Galeria(fragmentManager,fragment);
FragmentTransaction fragmentTransaction=fragmentManager.beginTransaction();
galeria.setCallBackInterface(cbi);
```

Figura 7.12 - Se establece la conexión a través de la interface (Fuente: Elaboración propia)

Una vez ya en la clase de Galería se rellena el método que se mostró en la Figura 7.8. Durante la ejecución de la clase Galería se llama al método de la interface para devolver la imagen resultante.

```
bitmap= MediaStore.Images.Media.getBitmap(getContext().getContentResolver(),uri);
if(callBackInterface!=null) {
    String hola="pop";
    callBackInterface.callBackImage(bitmap,hola);
```

Figura 7.13 - Se devuelve la imagen al adaptador por la interface (Fuente: Elaboración propia)

7.5 Sistema de bienstars

El sistema de BienSTARS es un sistema de recompensas que ofrece la aplicación web de Sano y Feliz. Este sistema de premios o bonificaciones entrega al usuario una serie de BienSTARS o estrellas cada vez que el usuario realice ciertas acciones como pueden ser leer una noticia en el blog, publicar un mensaje, crear un evento, etc. Este sistema solo ofrece estas recompensas en la versión web. En la versión móvil el usuario solo puede ver su estado actual dentro del sistema de BienSTARS, es decir, cuantas estrellas ha acumulado hasta el momento, en qué nivel se encuentra y cuántas estrellas le quedan para subir al siguiente nivel. En la Figura 7.14 se muestra el estado actual de un usuario dentro del sistema de BieneSTARS.

Figura 7.14 – Estado del usuario en el sistema de BieneSTARS desde app móvil (Fuente: Elaboración propia)

Para la versión móvil se ha tenido en cuenta el mismo algoritmo utilizado que en la aplicación web. En primer lugar, todos los usuarios parten desde el nivel cero y si consiguen diez estrellas alcanzan el nivel uno. Ahora bien, a medida que el usuario va escalando niveles dentro del sistema, cada vez tendrá que hacer más esfuerzo para poder alcanzar el siguiente nivel, ya que por cada nivel que el usuario suba, costará diez estrellas a mayores de lo que costó subir el nivel anterior. En definitiva, las cuentas que se realizarían para alcanzar el nivel cuatro serían las que aparecen en la Figura 7.15.

NIVEL	Estrellas necesarias
0	0
1	10
2	30
3	60
4	100

0 + 10

10 + 20

30 + 30

60 + 40

Figura 7.15 - Algoritmo del sistema de BieneSTARS (Fuente: Elaboración propia)

Para visualizar mejor la tendencia que adquiere este sistema de recompensas de forma gráfica, se puede observar dicha tendencia en la Figura 7.16.

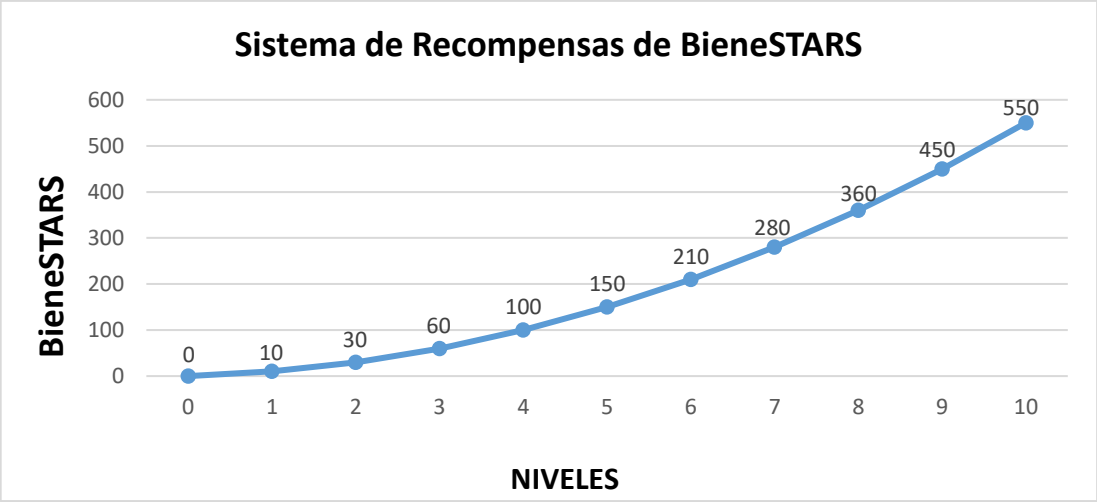

Figura 7.16 - Gráfica del sistema de recompensas (Fuente: Elaboración propia)

7.6 Sistema de menú lateral desplegable

La navegación a través de las diferentes zonas de la aplicación se realiza a través de un menú lateral desplegable personalizado. Este menú es el tipo de navegación más utilizado en todas las aplicaciones destinadas a móviles, tabletas e incluso en aplicaciones web que están programadas como "responsive" de forma que dispositivos que tengan una pantalla pequeña aparece dicho menú. En este caso, la aplicación móvil de Sano y Feliz tiene dos menús, que se muestran conjuntamente en la Figura 7.17. El menú de la zona pública aparece a la izquierda y el menú de la zona privada aparece a la derecha.

Figura 7.17 - Menús de la aplicación móvil de Sano y Feliz (Fuente: Elaboración propia)

Ambos menús son importantes para el buen funcionamiento de la aplicación y detrás de ello hay una programación importante que permite que todo se lleve a cabo bien. En primer lugar, se debe crear un fichero de XML para indicar los diferentes ítems que compondrán el menú como se ve en la Figura 7.18. Para cada menú se necesitará un fichero XML distinto por lo que la aplicación contará con dos ficheros XML. Estos ficheros se deben almacenar en la carpeta menú que tiene específicamente el proyecto de Android Studio.

```xml
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/portada"
        android:icon="@drawable/icono_home"
        android:title="Portada" />
    <item
        android:id="@+id/elproyecto"
        android:icon="@drawable/icono_proyecto"
        android:title="El proyecto" />
    <item
        android:id="@+id/nutriconsejos"
        android:icon="@drawable/icono_nutriconsejos"
        android:title="Nutriconsejos" />
    <item
        android:id="@+id/activate"
        android:icon="@drawable/icono_activate"
        android:title="Actívate" />
    <item
        android:id="@+id/blog"
        android:icon="@drawable/icono_blog"
        android:title="Blog" />
    <item
        android:id="@+id/iniciarsesion"
        android:icon="@drawable/icono_login"
        android:title="Iniciar sesión" />
</menu>
```

Figura 7.18 - Definición del menú de la zona pública en XML (Fuente: Elaboración propia)

Se aprovecha el sistema de gestión de “widgets” de Android Studio para implementar un “widget” denominado “NavegationView”, que será el desplegable donde se muestra el menú. En este caso como la clase Inicio es donde se ejecuta inicialmente la aplicación, se implementará el menú en esa clase. Para ello, como se puede ver en la Figura 7.19, se relaciona la variable de tipo “NavegationView” con el elemento visual creado anteriormente, se establece una imagen como fondo del menú, se especifica que mantenga los colores originales de los iconos del menú y se le establece un “Listener” al menú.

```
navigationView=findViewById(R.id.nvview);
navigationView.setBackground(getResources().getDrawable(R.drawable.fondo_menu));
navigationView.setItemIconTintList(null);

navigationView.setNavigationItemSelectedListener(this);
```

Figura 7.19 - Definición del menú en Java (Fuente: Elaboración propia)

Posteriormente se sobrescribe el método que permite ejecutar las acciones de cada opción del menú. Se utiliza "FragmentManager" para realizar las transacciones o cambios entre ventanas, es decir, entre opciones del menú, como se puede ver en la Figura 7.20.

```
@Override
public boolean onNavigationItemSelected(@NonNull MenuItem item) {
    switch (item.getItemId()){
        case R.id.portada:

            fragmentManager=getSupportFragmentManager();
            fragmentTransaction=fragmentManager.beginTransaction();
            fragmentTransaction.replace(R.id.frgprincipal,new Portada());
            fragmentTransaction.commit();
            break;

        case R.id.elproyecto:

            fragmentManager=getSupportFragmentManager();
            fragmentTransaction=fragmentManager.beginTransaction();
            fragmentTransaction.replace(R.id.frgprincipal,new Miproyecto());
            fragmentTransaction.commit();
            break;

        case R.id.nutriconsejos:

            fragmentManager=getSupportFragmentManager();
            fragmentTransaction=fragmentManager.beginTransaction();
            fragmentTransaction.replace(R.id.frgprincipal,new Nutriconsejos());
            fragmentTransaction.commit();
            break;
```

Figura 7.20 - Definición de la navegación de los menús (Fuente: Elaboración propia)

En la zona de la clase Inicio, donde se devuelve el usuario que ha iniciado sesión, se limpia todos los elementos del menú y se establece el menú de la zona privada como se puede ver en la Figura 7.20.

```
navigationView.getMenu().clear();
navigationView.inflateMenu(R.menu.menu_drawer);
```

Figura 7.21 - Cambio de menú (Fuente: Elaboración propia)

7.7 Reciclado de vistas y adaptadores de los elementos

7.7.1 Objetos de tipo recycledview

En ciertos momentos de la aplicación es necesario mostrar información en estructuras muy similares o quizás idénticas que faciliten la comprensión del usuario de toda la información que está viendo. En estos casos se hace uso de objetos denominados “RecyclerViews”, los cuales permiten de manera eficiente, mostrar al usuario grandes bloques de información. Los “RecyclerViews” son como listas que reciclan objetos individuales que se desplazan fuera de la pantalla, no los destruye. De esta forma, la ventaja que ofrecen estos objetos principalmente sería que mejora bastante el rendimiento y la capacidad de respuesta de la aplicación móvil, reduciendo así el consumo de energía **[30]**. Suponiendo que no se utilizara estos objetos, habría que programar las estructuras de los elementos de forma manual lo cual es bastante tedioso e ineficiente.

Concretamente se han definido en cuatro casos un “RecyclerView”. Estos casos serían:

- Todo lugar donde se muestren diferentes usuarios, por ejemplo, en las zonas de búsqueda y de amigos sugeridos.
- Todo lugar donde se muestren diferentes publicaciones, como es en el caso de la zona del muro y en la zona de perfil ya que se muestran las publicaciones realizas por el usuario y aquella en las cuales el usuario ha dado me gusta.
- En la zona de eventos para mostrar todos los eventos, ya bien sean eventos pasados, futuros o a los cuales el usuario ha confirmado su asistencia.
- En la zona de notificaciones, ya que las solicitudes de amistad son todas igual salvo que cambia el nombre del emisor de la solicitud.

A continuación, en la Figura 7.22 se muestra un ejemplo del uso de “RecyclerView” en el caso de las publicaciones. Se puede ver en dicha figura, de forma clara, la repetición de la estructura utilizada para mostrar las publicaciones, en este caso en la zona del muro.

Figura 7.22 - Resultados de utilizar RecyclerView (Fuente: Elaboración propia)

7.7.2 Adaptador del recyclerview

7.7.2.1 Frontend

Para crear un adaptador se debe crear una vista para dicho adaptador. En el ejemplo de las publicaciones, se tiene que la parte visual del adaptador es la estructura de una única publicación, que servirá como plantilla para mostrar varias publicaciones a la vez. Se puede ver la parte visual del adaptador en la Figura 7.23.

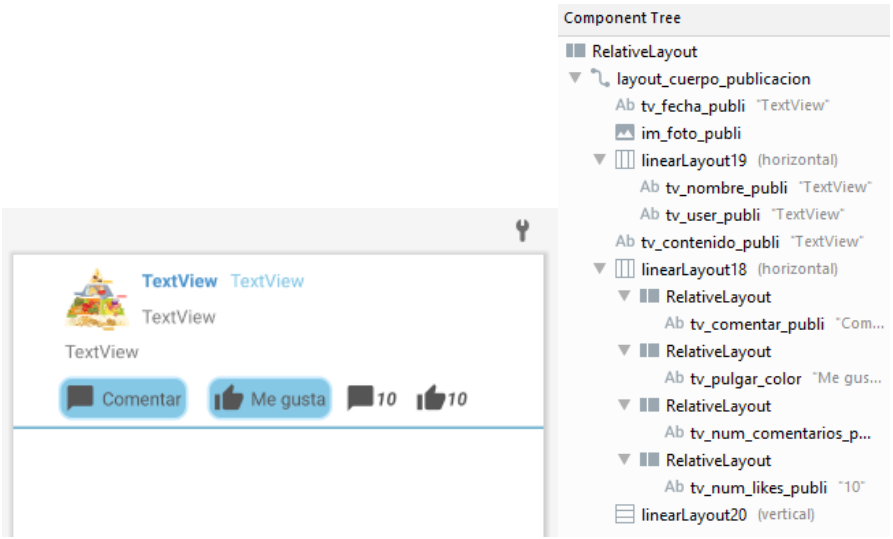

Figura 7.23 - Plantilla creada para el adaptador de publicaciones (Fuente: Elaboración propia)

7.7.2.2 Backend

Una vez establecido en la ventana donde se quiere utilizar un objeto de tipo "RecyclerView", se debe utilizar la clase "Adapter", que permitirá definir la estructura o forma en la que se mostrarán los datos. Antes de entrar más en profundidad en la clase que funciona como "Adapter", se debe establecer en la clase correspondiente a la ventana, que se va a mostrar el "RecyclerView", tal y como se ve en la Figura 7.24.

```
adapter=new ListaPublicacionesAdapter( tipo_fragment: 0,user_logged,getContext(),
        getFragmentManager(),publicaciones,megustas,user_logged);
recycler_muro.setAdapter(adapter);
```

Figura 7.24 - Definición del adaptador del RecyclerView (Fuente: Elaboración propia)

Como se puede ver en la figura anterior, se crea una instancia del adaptador programado para el "RecyclerView", en este caso del muro. Posteriormente se establece que el adaptador que utiliza el "recyclerView" es el que se instanció anteriormente.

Antes de nada, se debe crear la parte visual del adaptador, es decir, el elemento que se utiliza como plantilla para mostrar datos, como se observó en la Figura 7.23. Ahora bien, la clase que se crea para funcionar de adaptador debe heredar la clase "Adapter", específico para los "RecyclerView", como se puede ver en la Figura 7.25.

```
public class ListaPublicacionesAdapter extends RecyclerView.Adapter<ListaPublicacionesAdapter.PublicacionesViewHolder> {
    private ArrayList<Publicacion> listaPublicaciones;
    private ArrayList<String> likes;
    private Usuarios user_buscado;
    private FragmentManager fragmentManager;
    private Context context;
    private Mysql mysql;
    private Usuarios user_logged;
    private int tipo_fragment;
```

Figura 7.25 - Herencia de la clase Adapter (Fuente: Elaboración propia)

Una vez realizada la herencia, se debe crear una clase interna denominada PublicacionesViewHolder que hereda de "ViewHolder". La finalidad de esta clase interna es establecer la relación de los objetos creados en la parte visual con sus instancias en Java.

De esta forma se crea la conexión entre Java y la vista de la aplicación. Se puede ver un ejemplo en la Figura 7.26.

```
public class PublicacionesViewHolder extends RecyclerView.ViewHolder {
    TextView tv_nombre_publi,tv_user_publi,tv_fecha_publi,tv_contenido_publi,tv_num_comentarios_publi,tv_num_likes_publi;
    TextView tv_comentar_publi,tv_gustar_publi;
    ImageView im_foto_publi;
    ConstraintLayout ly_publicacion;
    TextView btn_asistir;
    public PublicacionesViewHolder(@NonNull View itemView) {
        super(itemView);
        tv_nombre_publi=itemView.findViewById(R.id.tv_nombre_publi);
        tv_user_publi=itemView.findViewById(R.id.tv_user_publi);
        tv_fecha_publi=itemView.findViewById(R.id.tv_fecha_publi);
        tv_contenido_publi=itemView.findViewById(R.id.tv_contenido_publi);
        tv_num_comentarios_publi=itemView.findViewById(R.id.tv_num_comentarios_publi);
        tv_num_likes_publi=itemView.findViewById(R.id.tv_num_likes_publi);
        tv_comentar_publi=itemView.findViewById(R.id.tv_comentar_publi);
        tv_gustar_publi=itemView.findViewById(R.id.tv_pulgar_color);
        im_foto_publi=itemView.findViewById(R.id.im_foto_publi);
        ly_publicacion=itemView.findViewById(R.id.layout_cuerpo_publicacion);
```

Figura 7.26 - Creación de la clase interna del adaptador (Fuente: Elaboración propia)

Además, se debe sobrescribir los tres métodos clave de la clase principal:

- Método onCreateViewHolder(): se debe llamar a este método cuando se necesita crear un "ViewHolder" y su vista asociada del elemento.
- Método onBindViewHolder(): en este método se realiza todo el desarrollo del adaptador, desde configuraciones visuales como colores de un botón, hasta crear "Listeners" para los eventos que generan todos los objetos. Como tiene las variables que hacen referencia a los elementos de la vista de la aplicación, puede realizar todas las acciones que se quiera con esos elementos.
- Método getItemCount(): este método sirve para saber el número total de elementos, en este caso publicaciones, que se mostrarán en el "RecyclerView".

```
@NonNull
@Override
public PublicacionesViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
    View view= LayoutInflater.from(parent.getContext()).inflate(R.layout.lista_item_publicacion, root: null, attachToRoot: false);
    PublicacionesViewHolder viewHolder=new PublicacionesViewHolder(view);
    return viewHolder;
}

@Override
public void onBindViewHolder(@NonNull ListaPublicacionesAdapter.PublicacionesViewHolder holder, int position) {...}

@Override
public int getItemCount() { return listaPublicaciones.size(); }
```

Figura 7.27 - Métodos del adaptador del RecyclerView (Fuente: Elaboración propia)

7.8 Texto interactico y dinámico en android (java)

En Java se permite dar cierta personalización a los textos que se quieren mostrar en las ventanas de la aplicación. Esta personalización está presente en varios editores de texto como Word, Google e incluso en Excel junto con otros muchos programas. Esta personalización consiste en mostrar una palabra o conjunto de palabras que se encuentran dentro de un texto de forma distinta, como son el que aparezcan resaltadas en negrita o cursiva. En Java también cabe la posibilidad de mostrar hiperenlaces dentro de un texto.

7.8.1 Cursiva y negrita

El establecer alguna palaba en negrita o cursiva dentro de un texto puede llegar a ser un poco tedioso, pero no imposible en Java. En la aplicación estos recursos son utilizado en la zona de El Proyecto, que se encuentra en la zona pública. En primer lugar, se debe almacenar el texto completo en una variable para poder realizar acciones sobre el texto. Luego, se debe crear dos objetos, uno objeto de tipo “SpannableStringBuilder” que albergará la variable de texto creada anteriormente haciéndola editable o personalizable, y otro objeto de tipo “StyleSpan” que permite establece el estilo de texto, en este caso se establece que dicho texto sea mostrado en negrita y al mismo tiempo en cursiva **[31]**. Para realizar las personalizaciones en el texto se utilizada el método “setSpan”, que ofrece la clase “SpannableString”. Se explica un caso como ejemplo a continuación. Como se puede ver en la Figura 7.28, se indica en el método que el resultado debe estar en cursiva y negrita pero solo desde donde se encuentre la palabra alumnos y finalice cuando acabe la palabra alumnos. El último parámetro se trata de SPAN_EXCLUSIVE_EXCLUSIVE, que significa que en caso de que se añade posteriormente alguna letra o palabra justo después, estas letras no se tomarán en cuenta luego para ser resaltadas.

```
String recompensas = "Para los alumnos : aumentar su bienestar, sentirse actores claves y protag
        "Para los centros educativos : la colaboración en la transferencia de conocimiento con
        "Para los padres : acompañar a sus hijos en una experiencia de auto-gestión que contribu

SpannableStringBuilder ssBuilder = new SpannableStringBuilder(recompensas);
StyleSpan boldItalicSpan = new StyleSpan(Typeface.BOLD_ITALIC);

//Aplica italic al texto
ssBuilder.setSpan(
        boldItalicSpan,
        recompensas.indexOf("alumnos"),
        end: recompensas.indexOf("alumnos") + String.valueOf("alumnos").length(),
        Spanned.SPAN_EXCLUSIVE_EXCLUSIVE
);

tv_recompensas.setText(ssBuilder);
```

Figura 7.28 - Personalización de texto a cursiva y negrita (Fuente: Elaboración propia)

7.8.2 Texto como hiperenlaces

En la zona El Proyecto de la aplicación se ha necesitado introducir hiperenlaces dentro de bloques de texto plano. En primer lugar, se crea un variable de tipo String que almacena el texto al completo que se quiere mostrar y otra variable de tipo SpannableString para hacer el texto modificable. Después en otra variable de tipo String, se define la dirección URL de la página que se quiere introducir como destino del hiperenlace. A continuación, se crea un objeto de tipo ClickableSpan que sirve para que, si se hace clic en la cadena de texto definida (url_somos2), se abra dicho enlace en una nueva página. Como se acaba de establecer el evento de abrir el enlace, solo falta enlazarlo con el texto que servirá como texto de hiperenlace. Para ello, se delimita la cadena de texto que funcionará como hiperenlace y se añade al SpannableString. Para darle un resaltado mejor al hiperenlace, se establece un color azul típico en de los hiperenlaces y que indica que el texto completo contiene eventos o que tiene ciertos movimientos, en este caso de hiperenlaces. A continuación, en la Figura 7.29 se muestra cómo se realizó la implementación de los hiperenlaces en un bloque de texto que necesitaba el uso de dichos hiperenlaces.

```
String text_somos="Somos el Grupo SALBIS (Salud, Bienestar y Sostenibilidad Sociosanitaria) de la Univers
SpannableString spanString = new SpannableString(text_somos);
String url_somos2="https://www.salbis.es";
ClickableSpan clickableSpan2=new ClickableSpan() {
    @Override
    public void onClick(@NonNull View view) {
        Intent intent= new Intent(Intent.ACTION_VIEW);
        intent.setData(Uri.parse(url_somos2));
        startActivity(intent);
    }

    @Override
    public void updateDrawState(TextPaint textPaint) { super.updateDrawState(textPaint); }
};
startIndex=text_somos.indexOf("Grupo SALBIS (Salud, Bienestar y Sostenibilidad Sociosanitaria)");
endIndex=startIndex+"Grupo SALBIS (Salud, Bienestar y Sostenibilidad Sociosanitaria)".length();

spanString.setSpan(clickableSpan2,startIndex,endIndex,Spanned.SPAN_EXCLUSIVE_EXCLUSIVE);
spanString.setSpan(new ForegroundColorSpan(ContextCompat.getColor(getContext(), R.color.color_enlaces)),
        startIndex,endIndex,Spanned.SPAN_EXCLUSIVE_EXCLUSIVE);

tv_somos.setText(spanString,TextView.BufferType.SPANNABLE);
tv_somos.setMovementMethod(LinkMovementMethod.getInstance());
```

Figura 7.29 - Hiperenlaces en bloques de texto (Fuente: Elaboración propia)

7.9 Implementación de un sistema para solicitar permisos

Por motivos de mantener la privacidad del usuario y cumplir la protección de datos, la cual se explica en el capítulo ocho de esta memoria, se debe implementar un sistema en el cual cuando se intente acceder a archivos, imágenes o herramientas del propio móvil del usuario, se debe pedir consentimiento al usuario para poder realizar dicha acción. En este caso en concreto, solo se pide permiso al usuario cuando se quiere acceder a la galería de imágenes cuando, por ejemplo, el usuario quiere cambiar su foto de perfil.

En este caso todo se activa cuando el usuario, en la zona de Ajustes, hace clic en el botón de cambiar foto de perfil. En ese entonces, si es la primera vez que accede a la aplicación le saldrá un recuadro solicitando permiso al usuario para acceder a la galería, tal y como aparece en la Figura 7.30.

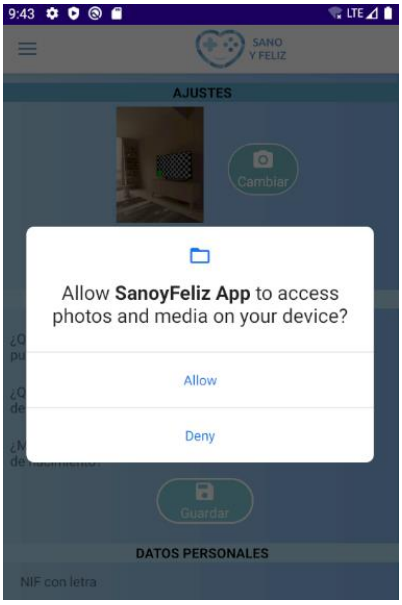

Figura 7.30 - Solicitud de permisos al usuario (Fuente: Elaboración propia)

En caso de que el usuario acepte, la aplicación abrirá la galería y en caso de que el usuario rechace la solicitud de acceso no se abrirá nada y no podrá cambiar su foto de perfil. No obstante, si ha rechazado la solicitud, le volverá a salir el mensaje cuando intente realizar la misma acción de cambiar la foto de perfil.

En primer lugar, se comprueba si el usuario ya ha aceptado que la aplicación acceda a su galería. En caso negativo, se solicita acceso a la persona como se puede ver en la Figura 7.31. Todo esto se puede realizar utilizando los métodos “checkSelfPermission” y “requestPermissions” **[32]**.

```
tv_foto.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        //Pedir permiso al usuario para acceso a galería
        if (ContextCompat.checkSelfPermission(getContext(),Manifest.permission.WRITE_EXTERNAL_STORAGE)
                != PackageManager.PERMISSION_GRANTED ||
                ContextCompat.checkSelfPermission(getContext(), android.Manifest.permission.READ_EXTERNAL_STORAGE)
                        != PackageManager.PERMISSION_GRANTED) {
            requestPermissions(new String[]{android.Manifest.permission.WRITE_EXTERNAL_STORAGE,
                            android.Manifest.permission.READ_EXTERNAL_STORAGE},
                    requestCode: 1);
        }
    }
});
```

Figura 7.31 - Implementación de la solicitud de acceso a galería (Fuente: Elaboración propia)

En caso de que el usuario acepte el permiso de acceso, se ejecuta la función "onRequestPermissionResult" que llamará a la función "cargarImagen", como se puede ver en la Figura 7.32.

```
//Función para pedir permiso de acceso a galería
@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
                                       @NonNull int[] grantResults) {
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);
    switch(requestCode)
    {
        case 1:
            if (grantResults.length > 0  && grantResults[0] == PackageManager.PERMISSION_GRANTED) {
                cargarImagen();
            }
            break;

    }
}
```

Figura 7.32 - Gestión de la respuesta a la solicitud de permiso (Fuente: Elaboración propia)

La función "cargarImagen", que se muestra en la Figura 7.33, es la encargada de abrir la galería. Por ello pregunta al usuario con qué aplicación desea entrar a su galería de fotos y de esta forma seleccionar la foto que más le guste para ponérsela de perfil.

```
//Cargar la imagen en el imageview
private void cargarImagen(){
    Intent intent=new Intent(Intent.ACTION_PICK, MediaStore.Images.Media.EXTERNAL_CONTENT_URI);
    intent.setType("image/");
    startActivityForResult(intent.createChooser(intent, title: "Selecciona la aplicación"), requestCode: 10);
}
```

Figura 7.33 - Función que muestra al usuario las aplicaciones para abrir la galería (Fuente: Elaboración propia)

Posteriormente la función "onActivityResult", que aparece en la Figura 7.34, es la encargada de coger la foto seleccionada por el usuario y almacenarla en una variable de

tipo Uri, para que de esta forma se pueda colocar de forma inmediata en el cuadro habilitado en la interfaz de la aplicación como foto de perfil.

```
@Override
public void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    if(resultCode==RESULT_OK){
        Uri path=data.getData();
        im_foto.setImageURI(path);  //Se coloca la imagen obtenida de la galería

        try {
            if(nuevo){
                mysql.guardarAvatar(user_logged.getId(),nombre_avatar); //Se guarda el nombre de
            }

            bit_http =MediaStore.Images.Media.getBitmap(getContext().getContentResolver(),path);
            cargarWebService();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Figura 7.34 - Función que coloca la foto selecciona en la interfaz (Fuente: Elaboración propia)

7.10 Conexión con el servidor para el tratamiento de archivos

En este apartado se detalla los dos tipos de conexiones que se han realizado entre la aplicación móvil de Sano y Feliz con el servidor de SALBIS, que es donde están almacenados diferentes archivos. En este caso en concreto, se trata de los archivos PNG, es decir, las imágenes de perfil de los usuarios, las cuales no están almacenadas en la base de datos, sino que están alojadas en el propio servidor en forma de archivo. No se ha almacenado dichas imágenes en la base de datos para evitar que la tabla donde está contenido todos los datos de los usuarios, se vuelva muy pesada de forma que ralentice las conexiones y las consultas a dicha tabla. Solo se ha almacenado en la tabla de los usuarios el nombre de la imagen que se corresponde con la foto de perfil.

7.10.1 Subir una foto al servidor

Suponiendo que el usuario ha elegido la nueva foto que quiere utilizar como foto de perfil, no basta solo con mostrarla en la ventana de la aplicación, sino que es necesario colocar el nombre la foto en la base de datos y también es importante subir la foto al servidor. Primero, se debe definir, mientras se crea la parte lógica de la aplicación, una variable de tipo RequestQueue que será la encargada de ser la propia solicitud que se enviará al servidor **[33].** Este tipo de variable está dentro de la librería Volley y la definición de la variable se puede ver en la Figura 7.35.

```
request= Volley.newRequestQueue(getContext()); //Preparar request para foto
```

Figura 7.35 - Instancia de la petición al servidor (Fuente: Elaboración propia)

Justo cuando la nueva imagen de perfil seleccionada se muestra en la ventana de la aplicación, se añade en la base de datos el nombre de la imagen de perfil donde el registro del usuario siempre y cuando no tuviese antes otra imagen asociada a su perfil en cuyo caso se mantiene el mismo nombre. Posteriormente se llama a la función "cargarWebService", que es la que aparece en la Figura 7.36, donde se crea una solicitud para el servidor de tipo POST, es decir, de envío de archivos, y se establece la dirección url del Web Service del servidor que responderá a la solicitud. Para confirmar que se ha enviado al servidor y que se ha almacenado correctamente se implementa que devuelva una variable de confirmación. Los parámetros que se envían en la solicitud de tipo POST son el nombre del archivo y la propia imagen, pero en tipo String, en vez de en Bitmap. El método utilizado para convertir la imagen de tipo Bitmap a String se muestra en la Figura 7.37.

```
private void cargarWebService(){
    String url="https://www.sanoyfeliz.es/subirAvatar.php?";
    stringRequest=new StringRequest(Request.Method.POST, url, new Response.Listener<String>() {
        @Override
        public void onResponse(String response) {
            if(response.trim().equalsIgnoreCase( anotherString: "registra")){
               Toast.makeText(getContext(), text: "Se ha guardado la imagen.",Toast.LENGTH_SHORT).show();

            }else{
                Toast.makeText(getContext(), text: "Error al guardar la imagen.",Toast.LENGTH_SHORT).show();

            }
        }
    }, new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {
            Toast.makeText(getContext(), text: "Ha habido un error"+error.toString(),Toast.LENGTH_SHORT).show();
        }
    }){
        @Override
        protected Map<String, String> getParams() throws AuthFailureError {

            String imagen=convertirImgString(bit_http);
            Map<String, String> parametros= new HashMap<>();
            parametros.put( k: "nombre",nombre_avatar);
            parametros.put( k: "imagen",imagen);

            return parametros;
        }
    };
    request.add(stringRequest);
}
```

Figura 7.36 - Preparación de la solicitud al servidor (Fuente: Elaboración propia)

```
//Convertir imagen bitmap a String
private String convertirImgString(Bitmap bitmap){

    ByteArrayOutputStream array=new ByteArrayOutputStream();
    bitmap.compress(Bitmap.CompressFormat.PNG, quality: 100,array);
    byte[] imageByte=array.toByteArray();
    String imagenString= Base64.encodeToString(imageByte, android.util.Base64.DEFAULT);


    return imagenString;
}
```

Figura 7.37 - Conversor de Bitmap a String (Fuente: Elaboración propia)

Para que la solicitud sea respondida por parte del servidor, es necesario crear un Web Service que dé una respuesta a la aplicación móvil. Es por ello por lo que se crea un fichero

PHP específico con las acciones necesarias para gestionar la solicitud de la aplicación. El archivo PHP es subido al servidor utilizando Filezilla. En la Figura 7.38 se muestra cómo está construido el Web Service. Se puede observar que se define la url del servidor, y que se guardan las variables u objetos enviados por la aplicación, en este caso la imagen y su nombre. Posteriormente, se establece la url o path dentro del servidor donde se almacenará la imagen estableciéndole ya el nombre. Por último, se almacena la imagen convirtiéndola desde tipo String en Base64.

```php
<?PHP
$hostname="www.sanoyfeliz.es";
$database="";
$username="";
$password="";




$imagen=$_POST["imagen"];
$nombre=$_POST["nombre"];
$path="public/upload/users/$nombre";

$url="https://$hostname/$path";

file_put_contents($path, base64_decode($imagen));
$bytesArchivo=file_get_contents($path);

echo "registra";
```

Figura 57.38 - Web Service del servidor (Fuente: Elaboración propia)

El nombre de la imagen se define siguiendo el siguiente protocolo. Antes de que el usuario si quiera haya decidido cambiar la imagen, se devuelve de la base de datos el nombre de la foto de perfil del usuario. En caso de que tengan una foto almacenada en el servidor, se reutilizará el mismo nombre almacenado en la base de datos. En caso contrario, se crea un nombre para la imagen el cual constará de 16 caracteres y tendrá formato PNG. Por lo que suponiendo que el usuario decida cambiar de foto de perfil, ya estará el nombre de la imagen definido y preparado para su uso como se puede ver en la Figura 7.39.

```
//Avatar no nuevo
if(nombresAvatar.size()==1){
    nombre_avatar=nombresAvatar.get(0);
    nuevo=false;
}
else{   //Si es nuevo
    String caracteres="aAbBcCdDeEfFgGhHiIjJlLmMnNoOpPqQrRsStTuUvVwWxXyYzZ0123456789";
    Random random= new Random();
    nombre_avatar="";

    do {
        for (int i = 0; i < 16; i++) {
            int randomInt = random.nextInt(caracteres.length());
            char randomChar = caracteres.charAt(randomInt);
            nombre_avatar += randomChar;

        }

        nombre_avatar += ".png";

    }while(nombresAvatar.contains(nombre_avatar));

}
```

Figura 7.39 - Sistema de creación de nombres para imágenes (Fuente: Elaboración propia)

7.10.2 Traer una foto del servidor

Se necesita traer del servidor la imagen de foto de perfil del usuario cuando entra la zona de Perfil. Se debe definir la ruta donde se encuentra la imagen dentro del servidor junto con el nombre de dicha imagen. También se debe crear una clase interna llamada “LoadImage” que hereda de “AsyncTask”. En esta clase interna se recibe el objeto de la interfaz donde se muestra la imagen. Además, a través del método “doInBackground” se utiliza la ruta en la que se encuentra la imagen en el servidor para poder traerla a la aplicación. La conexión es posible gracias a la instancia de tipo URL **[34]**, que es la que se encarga de crear un canal con el servidor para obtener la imagen. La imagen llega a la aplicación en forma de “InputStream” por lo que se debe convertir a Bitmap. Luego se coloca en el espacio reservado dentro de la interfaz. La llamada a la clase interna se encuentra en la Figura 7.40 y la propia clase interna se encuentra en la Figura 7.41.

```
Bitmap bit=null;
Ajustes ajustes=null;
if(user_logged.getNombre_imagen().length()!=0){

    String ruta_imagen="https://www.sanoyfeliz.es/public/upload/users/"+user_logged.getNombre_imagen();
    LoadImage loadImage =new LoadImage(im_foto_perfil);

    try {
      bit=loadImage.execute(ruta_imagen).get();
    } catch (ExecutionException e) {
        e.printStackTrace();
    } catch (InterruptedException e) {
        e.printStackTrace();
    }

}
```

Figura 7.40 - Llamada a la clase interna (Fuente: Elaboración propia)

```
//Clase para poner la imagen obtenida por HTTP
private class LoadImage extends AsyncTask<String, Void, Bitmap> {
    ImageView imageview;
    public LoadImage(ImageView ivResult){

        this.imageview=ivResult;
    }

    @Override
    protected Bitmap doInBackground(String... strings) {

        String url=strings[0];
        Bitmap bitmap=null;

        try {
            InputStream inputStream=new URL(url).openStream();
            bitmap =BitmapFactory.decodeStream(inputStream);

        } catch (IOException e) {
            e.printStackTrace();
        }
        return bitmap;
    }

    @Override
    protected void onPostExecute(Bitmap bitmap) {

        im_foto_perfil.setImageBitmap(bitmap);

    }
}
```

Figura 7.41 - Clase interna que trae imagen del servidor a la aplicación (Fuente: Elaboración propia)

7.11 Exportar apk con icono de la app móvil

Para desplegar la aplicación y que así el dueño del producto pudiera instalar la aplicación en su teléfono móvil, se decide exportar la aplicación como archivo APK. Un APK es un archivo ejecutable de aplicaciones que sirve para ser desplegado, en este caso, en dispositivos móviles con el sistema operativo Android **[35]**. De esta forma la aplicación se puede llevar de forma portable siendo transferible y ejecutable en otros móviles Android.

La ventaja que ofrece este archivo APK, es que se puede probar la aplicación en móviles físicos de Android, evitando así el uso de emuladores de Android en ordenadores, ya que es más probable que surjan errores ajenos a la aplicación debido a configuraciones o procesos internos del emulador.

Antes de sacar el archivo de la aplicación es necesario decidir qué icono debe representar, como acceso directo, la aplicación dentro del teléfono. Para ello se debe escoger la imagen que se desea utilizar, en este caso la imagen escogida es el logo de SALBIS, y posteriormente se introduce en la página de MakeAppIcon **[36]**, la cual hace llegar por email, la imagen elegida con varios tamaños para los diferentes tipos de resoluciones de dispositivos que existen. Una vez descargas las carpetas en las cuales se encuentran dichas imágenes respectivamente, se debe ir al directorio mipmap del proyecto y sustituir aquellas carpetas del directorio por las que nos descargamos desde el email. Existe una carpeta para cada tipo de resolución como se puede ver en la Figura 7.42.

Figura 7.42 - Icono de aplicación para diferentes resoluciones (Fuente: Elaboración propia)

Una vez realizado la sustitución de carpetas, se debe ir al archivo de configuración de la aplicación, es decir, el Manifest e indicar que el icono de la aplicación se encuentra en la carpeta mipmap y el nombre de la imagen para las diferentes resoluciones como se puede ver en la Figura 7.43.

```
<application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="SanoyFeliz App"
```

Figura 7.43 - Configurar el icono en el archivo Manifest (Fuente: Elaboración propia)

Por último, se comprueba que, al sacar el APK de la aplicación e instalarla en un dispositivo físico de Android, aparece el icono de acceso directo de la aplicación como se puede ver en la Figura 7.44. Además, si se hace clic en el icono se ejecuta adecuadamente la aplicación.

Figura 7.44 - Icono de la aplicación en dispositivo físico (Fuente: Elaboración propia)

8. Aviso legal

8.1. Introducción

Un dato, en este contexto, es aquello que da cierta información sobre alguna persona física en concreto, un ejemplo puede ser el teléfono o el correo electrónico privado de una persona. Desde hace unos años atrás, las personas también están adquiriendo cada vez más conciencia de lo importante que son sus datos y de lo vulnerables que serían si no se utilizaran dichos datos de forma correcta y respetuosa.

8.2. Reglamento general de protección de datos – 2016

Este pensamiento llevó al parlamento europeo ha sacar en 2016 una serie de normas de protección de datos, denominadas como Reglamento General de Protección de Datos (RGPD) **[37]**. Estas normas explican los diferentes contextos en el que un autónomo, empresa u organización está autorizada para recopilar o reutilizar información personal de cada persona. Con anterioridad a este reglamento, se aplicaba un protocolo de protección de datos más laxa, con pocas obligaciones comparadas con la actualidad.

8.3. Ley orgánica de protección de datos personales – 2018

Después de la obligada aplicación del RGPD en toda Europa, España instauró en 2018 la aplicación de la ley a nivel nacional como una nueva ley **[38]**. Antes de esta ley, toda entidad debía llevar a cabo un fichero con los datos de clientes, trabajadores u otras personas y este fichero se entregaba a la Agencia Española de Protección de Datos, pero actualmente en vez de realizar dicho fichero, se lleva a cabo un registro de actividad de parte de la entidad para conocer cómo se están tratando los datos.

Hoy en día, la empresa debe asignar a un trabajador cualificado, a partir de ahora denominado responsable, que valide la información y compruebe que dicha información está actualizada. En caso de que un usuario pidiera todos los datos que tiene una entidad sobre él, el responsable debe tener en cuenta los derechos del usuario, los cuales se muestran algunos de ellos a continuación. El usuario que ofrece datos personales a una entidad tiene diferentes derechos que se explican a continuación:

- Derecho a la trazabilidad: el responsable debe proporcionar al usuario un registro con la trazabilidad de por dónde han pasado sus datos.
- Derecho a la rectificación: el responsable debe proporcionar al usuario la posibilidad de modificar sus datos.
- Derecho al olvido: el responsable debe proporcionar al usuario la posibilidad de borrar todo dato que la empresa tenga sobre el usuario.
- Derecho a la finalidad: el responsable debe proporcionar la finalidad para la que se almacenan los datos del usuario.

8.4. Implementación de la normativa en la programación

Una vez explicadas las leyes involucradas en el proceso de desarrollo del trabajo, a continuación, se explica qué hacer para que código de la aplicación respete dichas normas. En este caso los únicos datos personales que el usuario proporciona son utilizados íntegramente por el grupo de investigación, o lo que es lo mismo, no se pasan a otras terceras personas o entidades, por lo que es fácil seguir la trazabilidad de los datos y se proporciona información a los usuarios de que sus datos son utilizados con la finalidad de ser usados solo en la aplicación, no con otras intenciones.

También, el usuario puede cambiar en cualquier momento de forma directa los datos que se tienen sobre él mismo a través de la opción ajustes de la aplicación. Incluso puede cambiar su foto de perfil, lo cual requiere acceder a la galería que es un lugar privado del usuario. Es por ello, por lo que antes de acceder a la galería, la aplicación solicita al usuario que la autorice para acceder a la galería y así obtener una foto y modificar su foto de perfil. En todo momento el usuario puede aceptar o rechazar la solicitud, en caso de que la decline

no podrá cambiar de foto de perfil, pero podrá dar su consentimiento en cualquier momento que quiera. En el apartado 7.9 de esta memoria se muestra cómo se puede implementar en la aplicación para solicitar permiso de acceso al usuario.

9. Accesibilidad

Este es un capítulo muy importante que a veces algunos programadores no tienen en cuenta. De nada sirve tener una aplicación capaz de realizar varias funciones complejas, sobre todo una aplicación en un dispositivo móvil, si la navegación es tediosa, compleja y además el usuario al que está destinado es una persona común sin conocimientos amplios de informática. Es por todo esto, que en este proyecto se ha tomado la accesibilidad como un aspecto crucial a tener en cuenta durante el desarrollo.

9.1 Iconos

Los iconos son elementos que representan acciones u objetos de forma que una persona al verlo sabe qué funcionalidad tiene dicho icono. Los iconos no se utilizan solo en el desarrollo software, sino que están fuertemente arraigados en la sociedad. Por ejemplo, si una persona necesita acceder al baño, la persona solo con echar un vistazo al icono que hay al lado de la puerta sabe si es el baño de chicos o de chicas. La mayoría de los iconos están tan arraigados en la sociedad que son utilizados a nivel mundial por lo que todo tipo de personas conocen el significado, por ejemplo, del icono que representa el wifi.

En el desarrollo software se utilizan iconos principalmente para favorecer la experiencia y la navegación del usuario dentro de la aplicación. Esto hace que todo tipo de personas, aunque no sepan en su totalidad cómo funciona la aplicación, puedan usar los iconos, ya que les ayudan a comprender de forma fácil e intuitiva qué se puede hacer en cada momento.

A continuación, concretamente en la Figura 9.1, se muestran los iconos que se utilizan en la navegación de los menús. Como se puede comprobar, los iconos mostrados son fáciles de comprender y se puede observar que no difieren muchos frente a iconos en otras aplicaciones móviles.

Figura 9.1 - Iconos de los menús (Fuente: [39])

En la aplicación se pueden apreciar el uso de más iconos aparte de los mostrados anterioremente. Sin embargo, son iconos utilizados dentro de las zonas por lo cuál no tiene una finalidad de navegación como tal, sino más bien tienen una función de mostrar al usuario la funcionalidad que ejecuta el icono. Por ejemplo, el icono de un engranaje muestra la opción de Ajustes o el icono de una campana muestra las notificaciones. Ambas funcionalidades se encuentran dentro de la zona de Perfil.

Si se desea ver en profundidad qué iconos se utilizan para representar las funcionalidades dentro de la aplicación, se pueden observar íntegramenten en el manual de usuario que se encuentra dentro de esta memoria. En este capítulo, aparte de explciar al usuario qué se puede hacer en cada ventana, se muestra a través de una serie de capturas cómo es visualmente la aplicación para que en caso de que el usuario quiera seguir el manual, puede guiarse, no solo a través de explicaciones redactadas sino también con estímulos visuales.

9.2 Disposición de la información

La información bien estructura y mostrada de forma vistosa siempre es agradecida por todos los usuarios. Para que la información sea útil debe ir acompañada de un contexto o de una aclaración acerca de qué es la información mostrada. Se utilizado tres métodos principalmente para mostrar la información o hacerla más accesible y comprensible al usuario.

- Texto como hiperenlace: El modo de implementar de forma programática en Java este tipo de texto se detalla en el apartado 7.8. No obstante, este tipo de texto, que

se encuentra en la zona de “El proyecto”, en la zona pública, ofrece una mayor accesibilidad al usuario ya que no solo sirve como texto informativo, sino como texto interactivo. Este texto se utiliza para que, si el usuario lo desea y lo viera oportuno, pueda ser redirigido fuera de la aplicación en tres ocasiones, concretamente a la orden donde se resuelve la subvención otorgada al proyecto, a la página web oficial del grupo de investigación SALBIS y por último a la página web oficial de la Universidad de León. Se pueden ver los hiperenlaces en la Figura 9.2.

Figura 9.2 - Texto como hiperenlace (Fuente: Elaboración propia)

- Texto en tabla: Este tipo de texto aparece en la zona de Progreso y se utiliza para mostrar los datos del historial de medidas del usuario con sus respectivas unidades de medida. La mejor manera de mostrar esa información es un formato con tabla, ya que son datos que forman parte del mismo origen, que son las medidas del usuario, y por lo tanto no pueden separarse los datos para su representación. Además, se le ha añadido a la tabla una barra de “scroll” lateral que permite ver la tabla al completo sin ocupar grandes extensiones de la pantalla. En el supuesto caso de no utilizar la tabla en esta ocasión, dichos datos ocuparían una extensión muy amplia de la pantalla en vertical, quitando la visibilidad de otros aspectos

importante que se muestran en esta zona como son el progreso en el programa de BienSTARS, el historial de hábitos alimenticios y el historial de hábitos de actividad física. Se puede ver la tabla utilizada en la Figura 9.3.

Figura 9.3 - Texto en tabla (Fuente: Elaboración propia)

- Texto en modo chat: este formato de texto se encuentra en la zona de Chat, dentro de la zona privada donde el usuario necesita iniciar sesión. Como se puede ver en la Figura 9.4, es necesario dividir la pantalla en dos, es decir, a la izquierda se muestra los mensajes de la persona con la que está hablando el usuario y en la parte derecha aparecen los mensajes del usuario que ha iniciado sesión. Cabe destacar también que el texto que escribe cada uno aparece en una zona de un color blanco si es el destinario y de color verde si es el usuario. De esta forma la información aparece de forma entendible por el usuario. En el supuesto caso de que no se repartiera en dos zonas la pantalla y que no se coloque un color para cada persona, el usuario no habría sabido a quién pertenece cada mensaje y la comunicación se haría muy difícil de seguir e ineficaz.

Figura 9.4 - Texto en chat (Fuente: Elaboración propia)

9.3 Colores

Los colores tienen una función muy importante en el apartado de la accesibilidad, debido a que puede generar dentro de los usuarios que utilizan la aplicación sentimientos de sosiego y lugar seguro mientras que por el contrario puede generar desconfianza e intranquilidad. Este efecto lo podemos observar cuando alguna web por la que se navega normalmente, de la noche a la mañana, cambia toda su estructura y los colores son diferentes, realizando un cambio disruptivo de la web antigua a la nueva. La primera vez que el usuario entra en dicha web piensa que está entrando en "territorio desconocido". Es por ello que se recomienda que los cambios visuales o estructurales de una aplicación web o móvil se deben realizar de forma paulatina y en diferentes versiones consecutivas, dando tiempo al usuario a adaptarse a pequeños cambios poco a poco.

Por lo tanto, se ha decidido mantener los colores predominantes de la versión web de Sano y Feliz, es decir, los colores azul y blanco como colores principales, también colores como el verde y el rojo en sus tonos más claros y suaves para opciones y por último el amarillo claro para el sistema de BienSTARS.

10. Evaluación y pruebas

Para este proyecto se han llevado a cabo diferentes tipos de evaluaciones y de pruebas. Entre ellas se encuentran la evaluación de los requisitos de la aplicación, pruebas del propio software, y las pruebas de validación de la aplicación.

10.1 Evaluación de requisitos

Antes de empezar a programar nada, es decir, estando en la etapa de planificación del proyecto, se mantienen varias conversaciones con el dueño del producto para especificar todos los requisitos que debe cumplir el entregable final. Una vez escuchado de forma paciente todos los requisitos del cliente y después de haberlos documentado, se debe realizar una evaluación de requisitos, o lo que es lo mismo, realizar un estudio de viabilidad y de realismo de cada requisito. Esta evaluación está destinada a responder preguntas como:

- ¿Se puede llevar a cabo a nivel informático?
- ¿Se tienen los recursos necesarios para cumplir con el requisito?
- ¿Los posibles obstáculos para implementar el requisito son superables?

En caso de que la respuesta a estas preguntas sea sí, se acepta el requisito cabiendo la posibilidad de que dentro de un sprint se pueda refutar la posibilidad de implementar un requisito por algún factor que no se contempló a priori y por lo tanto ese requisito, o esa implementación en particular, queda desechada del entregable final.

10.2 Pruebas del software

El código que compone todo el sistema se divide en módulos. Por lo tanto, se realizar varias pruebas de integración, las cuales permiten detectar posibles errores o fallas que conlleva la interacción entre diferentes módulos. Lo que se busca es que a medida que se van

testeando o comprobando cada módulo, estos se juntan, realizando así una integración incremental ascendente.

La prueba de integración utilizada se denomina prueba de regresión, la cual es una actividad cuyo fin es asegurarse de que los cambios que se han introducido no han propagado efectos colaterales negativos. Hay que tener en cuenta que los módulos más pequeños e independientes se dan por testeados con resultados favorables, es por ello por lo que solo se pueden obtener errores a la hora de añadir o juntar módulos entre sí, ya que existe un cambio de entorno muy importante como son:

- Se crean nuevas rutas de flujo de datos.
- Pueden aparecer nuevas entradas y salidas.
- Posibles nuevas opciones o caminos de ejecución disponibles.

10.3 Pruebas de validación

Las pruebas de validación son aquellas que se realizan para conocer si el software desarrollado cumple con los requisitos del cliente. Gracias a estas pruebas se puede afirmar que se han satisfecho todos los requisitos funcionales. Por su puesto, también se han cumplido todos los requisitos no funcionales. Además, para entregar una documentación adecuada, correcta e inteligible se presenta en el capítulo 11 de esta memoria un manual de usuario para ayudar a la persona que utiliza la aplicación. Por lo tanto, al cumplir con los requisitos de toda índole, en la lista de deficiencias no hay ninguna reclamación. Se han utilizado dos tipos de pruebas de validación:

10.3.1 Pruebas de validación alfa

El cliente utiliza la aplicación final en el lugar donde se desarrolló, es decir, se utiliza el software estando en presencia de los desarrolladores, que esta vez tendrán el papel de observadores cuya función principal es registrar errores y problemas de uso. En definitiva, se realiza la prueba en un entorno predeterminado y controlado.

10.3.2 Pruebas de validación beta

Por otro lado, en las pruebas de validación beta son los usuarios finales los que utilizan la aplicación en diferentes lugares elegidos por los propios usuarios, donde los desarrolladores no estarán presentes observando, sino que es el propio cliente el que registra cualquier incidencia o error que ocurra y debe notificarlo a los desarrolladores. Las pruebas beta se deben llevar a cabo en entornos no predefinidos o no controlados.

11. Manual de usuario

Este capítulo sirve como manual o guía para el usuario para que pueda navegar de forma rápida a través de la aplicación. En primer lugar, nada más iniciar la aplicación aparece un mensaje informativo tal y como se ve en la Figura 11.1.

Figura 11.1 - Mensaje inicial (Fuente: Elaboración propia)

Cuando se cierra el mensaje informatico se puede ver la primera ventana de la aplicación que se corresponde con la portada, Figura 11.2, donde se comunica al usuario el nombre del proyecto y cuál es el objetivo de la misma.

Figura 11.2 - Portada de la aplicación (Fuente: Elaboración propia)

Siguiendo en sentido secuencuial las opciones del menú desplegable lateral de la sección pública, la siguiente opción es "El proyecto", Figura 11.3. Allí se puede ver más en profundidad quién está a cargo del proyecto además de otros factores relacionados con el proyecto.

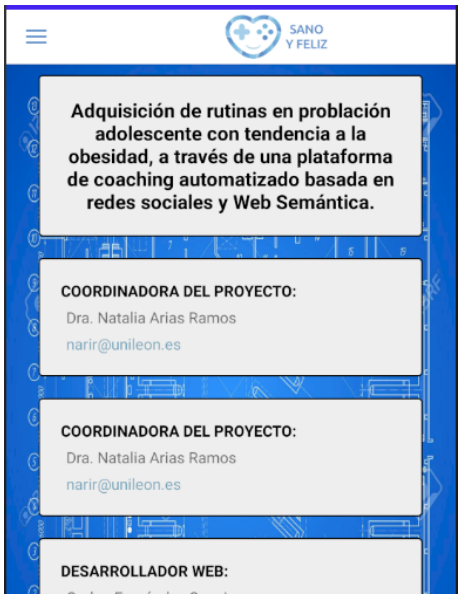

Figura 11.3 - Ventana acerca del proyecto de la aplicación (Fuente: Elaboración propia)

Luego se encuentra la opción de Nutriconsejos, Figura 11.4, donde se pueden observar diferentes consejos saludables relaciones con la alimentación. Por otro lado, en la Figura 11.5 se muestra la opción o sección de Actívate donde se muestran consejos relaciones con el ejercicio físico.

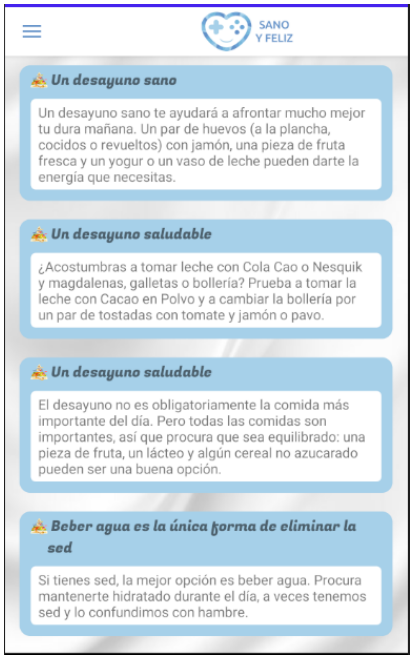

Figura 11.4 - Ventana Nutriconsejos (Fuente: Elaboración propia)

Figura 11.5 - Ventana Actívate (Fuente: Elaboración propia)

En la Figura 11.6, se muestra la ventana para iniciar sesión y de esta forma poder acceder a la sección privada o la red social de la aplicación.

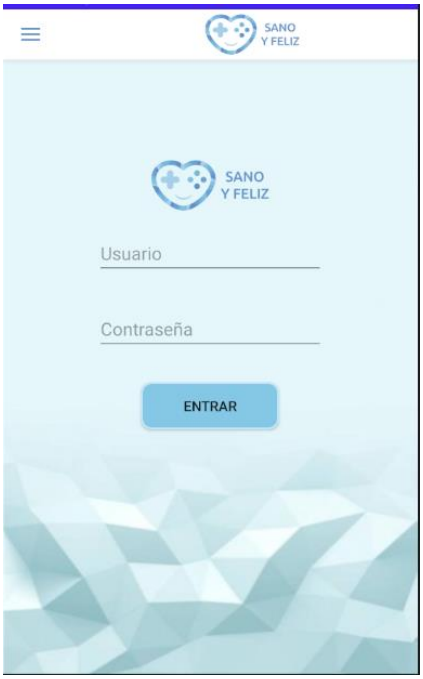

Figura 11.6 - Ventana Iniciar Sesión (Fuente: Elaboración propia)

Una vez iniciado sesión correctamente, se muestra de forma directa la ventana del Muro, donde se muestran todas las publicaciones que han realizado los amigos del usuario. Además permite al usuario responder a comentarios, dar me gusta a los comentarios, comprobar el número de respuestas de la publicación y a su vez el número de me gustas que tiene dicha publicación. El Muro se muestra en la Figura 11.7. En la Figura 11.8, se puede ver el hilo de respuestas que tiene una publicación haciendo clic sobre dicha publicación. Si se desea escribir una nueva publicación, se debe hacer clic en el icono del lápiz, donde el muro, y le aparece el cuadro para escribir una publicación, el cual se muestrta en la Figura 11.9.

Figura 11.7 - Ventana Muro (Fuente: Elaboración propia)

Figura 11.8 - Hilo de respuesta de una publicación (Fuente: Elaboración propia)

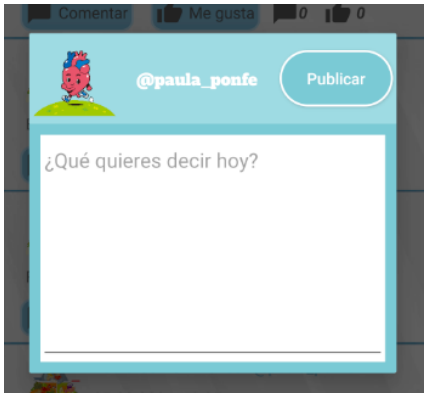

Figura 11.9 - Cuadro para publicar (Fuente: Elaboración propia)

La primera sección a la cual se permite navegar a través del menú lateral desplegable de la zona privada, es la sección de Pefil, Figura 11.10. En esta sección se puede ver las publicaciones y los me gustas realizados por el usuario. También desde la sección de Perfil se puede navegar a las ventana de Ajustes, de Notificaciones y de Lista de Amigos.

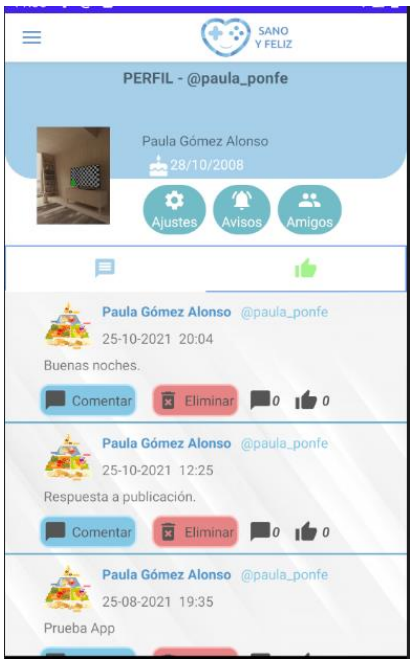

Figura 11.10 - Ventana Perfil (Fuente: Elaboración propia)

En la ventana de Ajustes, Figura 11.11, se puede modificar las preferencias de privacidad del usuario. También puede modificar datos personales desde ese mismo panel. Por otro lado puede modificar la contraseña anterior por una nueva.

Figura 11.11 - Ventana Ajustes (Fuente: Elaboración propia)

La ventana Notificaciones, Figura 11.12, muestra todas las peticiones de amistad recibidas por el usuario, pudiendo rechazarlas o aceptarlas.

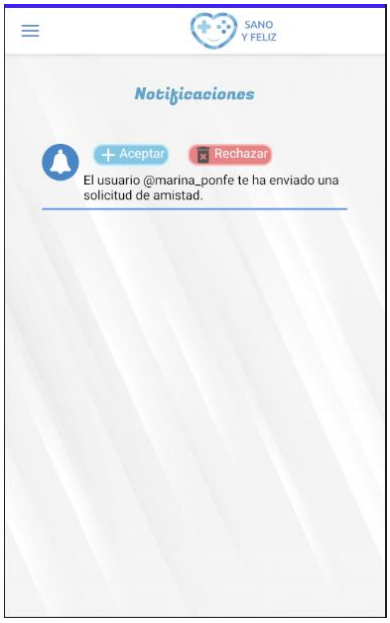

Figura 11.12 - Ventana Notificaciones (Fuente: Elaboración propia)

La ventana Lista de Amigos, Figura 11.13, muestra en forma de lista todos los amigos del usuario. Incluso tiene la opción de borrar amigos.

Figura 11.13 - Ventana Lista de Amigos (Fuente: Elaboración propia)

Volviendo de nuevo al menú lateral, la siguiente opción es la de Búsqueda, Figura 11.14, que permite buscar a cualquier usuario dentro de la aplicación pudiendo añadirle como amigo o solo ver su perfil.

Figura 11.14 - Ventana Búsqueda (Fuente: Elaboración propia)

La siguiente sección es la de Eventos, Figura 11.15, donde se puede gestionar todos los eventos existentes.

Figura 11.15 - Ventana Eventos (Fuente: Elaboración propia)

En esta ventana existe la opción de crear un evento nuevo, el cual puede ser configurado por el usuario. La creación de un nuevo evento se puede ver en la Figura 11.16. Además, dentro de la ventana de gestión de Eventos, existen tres tipos de eventos. Están los Eventos Próximos, Figura 11.17, que son aquellos a los que el usuario puede apuntarse para confirmar su asistencia. También están los Eventos Asistirás, Figura 11.18, que son aquellos a los que el usuario va a asistir pudiendo cancelar su asistencia si no es el creador del evento. Por último, están los Eventos Asististe, Figura 11.19, que son aquellos eventos a los cuales el usuario asistió en el pasado y que puede adjuntar una imagen para verificar su asistencia.

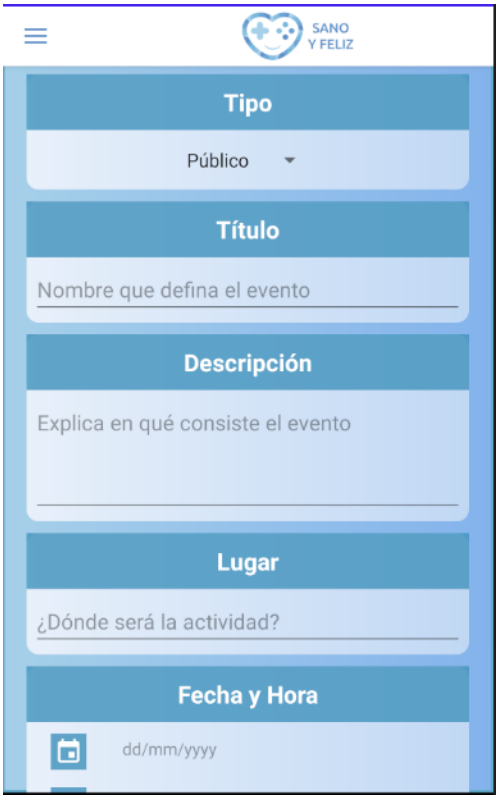

Figura 11.16 - Ventana de creación de eventos (Fuente: Elaboración propia)

Figura 11.17 - Ventana Eventos Próximos (Fuente: Elaboración propia)

Figura 11.18 - Ventana Eventos Asistirás (Fuente: Elaboración propia)

Figura 11.19 - Ventana Eventos Asistidos (Fuente: Elaboración propia)

En el menú desplegable de la aplciación se encuentra también la sección de Progreso, Figura 11.20, que es donde se muestra todo el progreso del usuario en el sistema de BienSTARS. A mayores, se muestra la evolución de las medidas del usuario, su puntuación según su historial de hábitos alimenticios y su puntuación según su historial de hábitos en la actividad física.

Figura 11.20 - Ventana Progreso (Fuente: Elaboración propia)

Por último, dentro de la aplicación quedan dos secciones, la de Chat y la de Amigos Sugeridos, que se corresponden con la Figura 11.21 y la Figura 11.22 respectivamente. En el chat, el usuario solo debe escoger al amigo con el que quiere conversar, escribir un mensaje y darle al botón de enviar. Por otro lado, en la última sección de Amigos Sugeridos, se muestran cinco usuarios de forma aleatoria para que el usuario, si quiere, pueda enviar una solicitud de amistad a alguna de esas personas.

Figura 11.21 - Ventana Chat (Fuente: Elaboración propia)

Figura 11.22 - Ventana Amigos Sugeridos (Fuente: Elaboración propia)

12. Conclusión

Como conclusión de este proyecto, se ha percibido la gran mejora en el desarrollo de las aplicaciones móviles respecto a aplicaciones anteriormente realizadas. En este proyecto se puede decir que se han implementado todas las funcionales requeridas. Además, se ha comprobado que la aplicación se ejecuta correctamente.

Es cierto que la implementación del blog no se pudo realizar, no por problemas internos de este proyecto, sino por la estructura predefinida de la base de datos la cual puede ser utilizada en Javascript, lenguaje utilizado para aplicaciones web, pero no para Android (Java). Como el cliente del producto vio que el problema radicaba en algo externo al proyecto y que la solución que tenía que adoptar era muy costosa para implementar una única funcionalidad, decidió quitarla. Es por ello que se considera que todas las funcionalidades se han implementado.

La aplicación resultante de este proyecto tiene la ventaja de que es escalable, es decir, se puede implementar otra sección privada si eres profesor u otra distinta si eres administrador.

Por último, esta aplicación ofrece diferentes oportunidades para desarrollar varias líneas de futuro que pueden ser bastante interesantes:

- Desarrollar procedimientos almacenados para dar una capa de abstracción y protección a la aplicación.
- Desarrollar una inteligencia artificial para mostrar usuarios recomendados basándose en diferentes criterios que ha obtenido del usuario que ha iniciado sesión.
- Como se indicó anteriormente, desarrollar la parte privada de los profesores y la parte del administrador.
- Implementar el sistema de gestión de BieneSTARS dentro de la aplicación, ya que actualmente solo muestra el estado actual en el sistema realizando la función de mostrar información.

13. Agradecimientos

Aprovecho este apartado dentro de la memoria para pronunciar unas palabras de agradecimiento a algunas personas que han estado presentes en mayor o menor medida a lo largo del desarrollo de este proyecto de fin de máster.

Mi agradecimiento principal me gustaría dirigírselo a mi director del trabajo y profesor, José Alberto, ya que ha sido como el viento que mueve la vela de un barco en plano mar. Me ha mostrado diferentes formas que podría utilizar para estructurar mi proyecto. Cada vez que me he enfrentado a un obstáculo, para el cuál tras mucho tiempo no encontraba una solución, José Alberto me tendió una mano, lo que conllevó que todo obstáculo fuera solucionado.

Por otro lado, ha mantenido una vía de comunicación conmigo, no solo a través de teléfono o correos electrónicos, sino que estableció reuniones online a través de Google Meet conmigo para llevar un seguimiento de la evolución del trabajo. A nivel docente, veo obligado destacar la rapidez con la que ayuda a los alumnos con problemas que pueden surgir y la facilidad que tiene para enseñar conocimientos que a simple vista pueden verse complejos.

Por último, me gustaría agradecer también al Grupo de Investigación SALBIS por darme la oportunidad de poder realizar la versión móvil para Android de Sano y Feliz, además de permitirme ser parte del grupo como investigador asociado.

14. Bibliografía

[1] Andrades, J. Salud, Bienestar, Ingeniería y Sostenibilidad Sociosanitaria - SALBIS Grupo de Investigación. Retrieved from https://www.salbis.es/

[2] C. Benavides, J. A. Benítez-Andrades, P. Marqués-Sánchez, y N. Arias, «eHealth Intervention to Improve Health Habits in the Adolescent Population: Mixed Methods Study», JMIR Mhealth Uhealth, vol. 9, n.º 2, p. e20217, feb. 2021, doi: 10.2196/20217.

[3] J. A. Benítez-Andrades, N. Arias, M. T. García-Ordás, M. Martínez-Martínez, y I. García-Rodríguez, «Feasibility of Social-Network-Based eHealth Intervention on the Improvement of Healthy Habits among Children», Sensors, vol. 20, n.º 5, p. 1404, mar. 2020, doi: 10.3390/s20051404.

[4] Portada | Sano y Feliz. Retrieved from https://www.sanoyfeliz.es/public/portada

[5] 8fit | Entrenamientos personalizados en casa y alimentación saludable. Retrieved from https://8fit.com/es/

[6] Noom Es. Noom para Negocios - Noom Es. Retrieved from https://web2.noom.com/es/business/

[7] Fuertafit - Inicio. Retrieved from https://fuertafit.com/

[8] IEEE. (1984). 830-1984 - IEEE Guide for Software Requirements Specifications.

[9] Olivero, E. (2018). Hábitos de Consumo Mobile en España y en el Mundo en 2018. Retrieved from https://pickaso.com/2018/informe-consumo-mobile-2018#:~:text=Los%20espa%C3%B1oles%20prefieren%20utilizar%20el,puntos%20porcentuales%20respecto%20a%202016

[10] Introducción a Android Studio | Desarrolladores de Android. (2020). Retrieved from https://developer.android.com/studio/intro?hl=es-419

[11] Sc.ehu.es. 2021. *La Máquina Virtual Java*. Retrieved from http://www.sc.ehu.es/sbweb/fisica/cursoJava/fundamentos/introduccion/virtual.htm

[12] Stack Overflow en español. Retrieved from https://es.stackoverflow.com/

[13] Robledano, Á., 2019. *Qué Es Mysql: Características Y Ventajas*. OpenWebinars.net. Retrieved from https://openwebinars.net/blog/que-es-mysql/

[14] Chacon, S., & Straub, B. (2014). *Pro Git* (2nd ed.). Apress.

[15] Php.net. *PHP:* ¿Qué es PHP? - Manual. Retrieved from https://www.php.net/manual/es/intro-whatis.php

[16] Filezilla-project.org. FileZilla - The free FTP solution. Retrieved from https://filezilla-project.org/

[17] Layton, M. (2018). *Scrum*. Wiley & Sons Canada, Limited, John.

[18] Garzás Parra, J., Enríquez de S, J., & Irrazábal, E. (2012). Gestión ágil de proyectos software. Madrid: Kybele Consulting

[19] Git - Acerca del Control de Versiones. Retrieved from https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Acerca-del-Control-de-Versiones

[20] 1.1. Acerca del control de versiones (Pro Git, el libro oficial de Git). Retrieved from https://uniwebsidad.com/libros/pro-git/capitulo-1/acerca-del-control-de-versiones

[21] Git - ¿Qué es una rama? Retrieved from https://git-scm.com/book/es/v2/Ramificaciones-en-Git-%C2%BFQu%C3%A9-es-una-rama%3F

[22] Driessen V (2010). "A Successful Git Branching Model." Retrieved from http://nvie.com/posts/a-successful-git-branching-model/

[23] Es.indeed.com. 2021. *Salario de un Analista programador android en España*. Retrieved from https://es.indeed.com/career/analista-programador-android/salaries?from=top_sb

[24] 2021. Salario de un Scrum master en España. Es.indeed.com. Retrieved from https://es.indeed.com/career/scrum-master/salaries

[25] Es.indeed.com. 2021. Salario de un Diseñador gráfico en España. Retrieved from https://es.indeed.com/career/dise%C3%B1ador-gr%C3%A1fico/salaries

[26] tarifasgasluz.com. 2021. Consulta el precio de la luz (€/kWh): tarifas y comparativa. Retrieved from https://tarifasgasluz.com/comparador/precio-kwh

[27] Modelo cliente servidor: ¿Qué es? Características, Ventajas y Desventajas. Retrieved from https://blog.infranetworking.com/modelo-cliente-servidor/

[28] Invarato, R. (2013). Multitarea en Android - Jarroba. Retrieved from https://jarroba.com/multitarea-en-android/

[29] Geek, E. (2019). Clases abstractas e interfaces: ¿Qué son?. Retrieved from https://ifgeekthen.nttdata.com/es/clases-abstractas-e-interfaces

[30] Cómo crear listas dinámicas con RecyclerView | Desarrolladores de Android | Android Developers. (2021). Retrieved from https://developer.android.com/guide/topics/ui/layout/recyclerview?hl=es-419

[31] Android Developers. 2021. *SpannableStringBuilder | Android Developers*. Retrieved from https://developer.android.com/reference/android/text/SpannableStringBuilder

[32] Request app permissions | Android Developers. (2021). Retrieved from https://developer.android.com/training/permissions/requesting

[33] Send a simple request | Android Developers. (2020). Retrieved from https://developer.android.com/training/volley/simple

[34] Creating a URL (The Java™ Tutorials > Custom Networking > Working with URLs). Retrieved from https://docs.oracle.com/javase/tutorial/networking/urls/creatingUrls.html

[35] Aguilar, R., 2020. Qué es un APK de Android, cómo se instala y diferencias con las apps normales. *Xataka Android*, Retrieved from https://www.xatakandroid.com/aplicaciones-android/que-apk-android-como-se-instala-diferencias-apps-normales

[36] Makeappicon.com. n.d. *Developer Tool - App Icon Resizer*. Retrieved from https://makeappicon.com/

[37] *Relativo a la protección de las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos y por el que se deroga la Directiva 95/46/CE (Reglamento general de protección de datos) (2016)*.

[38] Jefatura de Estado, 2018. *Ley Orgánica 3/2018, de 5 de diciembre, de Protección de Datos Personales y garantía de los derechos digitales*. BOE.

[39] Pngaaa.com. Millions of PNG Images, Backgrounds and Vectors for Free Download - pngaaa.com. Retrieved from https://www.pngaaa.com/