Activity Tracker

Guión de la presentación

Diapositiva 1: Presentación (seguramente lo diga el tribunal)

Buenas tardes, soy Ignacio Martínez Gallardo y voy a presentar mi Trabajo de Fin de Grado. Se titula “Aplicación móvil para la monitorización de actividades”, y ha sido tutorizado por Francisco José Jaime Rodríguez.

Diapositiva 2: Motivación

Lo que ha motivado la realización de este trabajo ha sido la dificultad que tenemos muchas veces las personas para cumplir con los objetivos que nos proponemos.

Algunas de las principales causas por lo que esto ocurre son:

- La falta de motivación
- La poca precisión a la hora de definir la meta del objetivo
- Establecer objetivos muy ambiciosos
- O directamente la ausencia de un seguimiento del progreso.

Estos motivos dan pie a que exista una herramienta que permita facilitar el seguimiento de las diferentes actividades que se pueda proponer una persona.

La solución a la que hemos llegado es crear una aplicación móvil, a la que hemos nombrado ‘Activity Tracker’.

Diapositiva 3: Objetivos

Los objetivos que tiene que cumplir son:

- El usuario pueda crear y gestionar su cuenta.
- Pueda crear actividades de las que quiere hacer un seguimiento y de las que pueda ver detalles y estadísticas.
- Envío de solicitudes de amistad y desafío.
- Sistema de mensajería para que los usuarios se puedan comunicar mediante la aplicación.

Diapositiva 4: Tecnologías

La aplicación ha sido desarrollada con el sistema operativo Android en mente. Para ello, he optado por usar el framework de desarrollo de aplicaciones multiplataforma Flutter.

El lenguaje de programación que utiliza es Dart. Es un lenguaje orientado a objetos cuya sintaxis es similar a Java.

Para la implementación del backend y la creación de la base de datos, he usado la plataforma de desarrollo de aplicaciones Firebase.

Diapositiva 5: Herramientas

En cuanto a las herramientas, he empleado Visual Paradigm para la creación de los diferentes diagramas UML y PencilProject para la creación de los mock-ups de la interfaz de usuario. Quiero destacar también otras herramientas como Latex y Overleaf para la elaboración de la documentación y Git y Github para el control de versiones del código y los documentos.

Diapositiva 6: Proyecto

Para este proyecto, hemos seguido al completo el ciclo de desarrollo de software, abordando las fases de: planificación, análisis y especificación de requisitos, modelado y diseño del sistema, desarrollo e implementación, validación y verificación, despliegue y mantenimiento.

Diapositiva 7: Metodología ágil

Para ello, hemos seguido una metodología ágil basada en Scrum, pero adaptada a las características de nuestro proyecto. Cada sprint ha tenido una duración de dos semanas. Al final de cada una de estas iteraciones se ha organizado una reunión con el tutor para evaluar y validar el trabajo realizado y plantear lo que se iba a realizar de cara al siguiente sprint.

Diapositiva 8: Planificación

Para la planificación del trabajo, he usado la herramienta Trello. En el tablero creado tenemos cuatro listas principales:

- Backlog representa las tareas pendientes del trabajo
- Sprint actual aquellas en las que está planificado trabajar en la iteración actual
- En proceso en las que ya se está trabajando.
- En completado aquellas que se han terminado.
- Adicionalmente, al final de cada iteración hemos creado un nuevo listado para almacenar las tareas completadas en dicha iteración.

Diapositiva 9: Comentar sprint

Aquí se puede ver el listado de tareas para los 6 primeros sprints. Las tareas se representan mediante tarjetas, con etiquetas para las etapas del ciclo software y sprint a las que pertenece.

Diapositiva 10: Análisis y especificación de requisitos

Como hemos mencionado, queremos una aplicación en la que los usuarios pueden gestionar su cuenta, crear actividades con diferentes parámetros de las se quiere monitorizar el progreso, ver estadísticas de las actividades, enviar, recibir y gestionar solicitudes tanto de amistad como de desafío de actividad y establecer conversaciones para enviar mensajes entre otras funcionalidades.

Todo lo que se requería que hiciera el sistema se ha recogido en el listado de requisitos funcionales. Todas las restricciones que se aplican sobre el sistema y sus funcionalidades se han añadido a los requisitos no funcionales. Por último, hemos establecido un listado de requisitos opcionales, que son aquellos a realizar en un futuro para aumentar las funcionalidades de la aplicación.

Diapositiva 11: Modelado y diseño del sistema

Continuamos con el modelado y diseño del sistema, etapa en la que se han elaborado los casos de uso junto a su diagrama, el modelo estructural mediante un diagrama de clases y el modelador comportamental mediante diagramas de secuencia y el diseño de la interfaz de usuario.

Diapositiva 12: Casos de uso

Los casos de uso describen el comportamiento del sistema desde el punto de vista del usuario para lograr un objetivo. Aquí podemos ver un caso de uso particular. Por cada caso de uso tenemos una tabla con su identificador, el título, una descripción, sus precondiciones y postcondiciones, su escenario principal y los posibles escenarios alternativos.

Diapositiva 13: Diagrama casos de uso

Todos los casos de uso están representados en un diagrama que está disponible en la memoria. Aquí mostramos un subdiagrama centrado en la gestión de usuarios. Tenemos un único actor que es el usuario, el cual está asociado a los casos de uso. Hemos empleado la relación de extensión para aquellos casos de uso que permiten ejecutar otros casos. En este caso, para que un usuario editar su perfil o eliminar su cuenta, debe ejecutar primero el caso de uso de ‘Ver los detalles del perfil’.

Diapositiva 14: Diagrama de clases

Para representar de manera gráfica el modelado estructural, hemos diseñado un diagrama de clases completo está en la memoria.

Diapositiva 15: Subdiagrama de clases 1

En este primer subdiagrama se muestran las clases principales del sistema. Por ejemplo un actividad pertenece a un solo y usuario y este puede tener un número indeterminado de actividades. Estas actividades están compuestas por progresos, de manera que cada progreso es exclusivo de una actividad. Podemos ver también diferentes tipos enumerados que se utilizan en las atributos de las clases así como la clase ActivityUtils que contiene todos los métodos que se utilizan para las actividades.

Diapositiva 16: Subdiagrama de clases 2

En este segundo subdiagrama están algunas de las clases servicio, las cuales contienen para cada entidad los diferentes métodos que nos permiten crear, actualizar, borrar y tomar documentos de la base de datos.

Diapositiva 17: Diagrama de secuencia

El modelado comportamental del sistema se representa mediante diagramas de secuencia, los cuales describen los eventos e intercambios de mensajes entre los actores y objetos. Cada uno de estos diagramas representa uno de los escenarios definidos en los casos de uso.

Diapositiva 18: Diagrama de secuencia petición amistad

Vamos a comentar el siguiente diagrama de secuencia, que representa el escenario de éxito para el envío de una solicitud de amistad a un usuario. Mientras que el usuario introduzca un nombre de usuario que no es válido, UserService lo comprobará y enviará un mensaje a AddFriendPage para que le muestre un mensaje informativo al usuario. Una vez que haya introducido uno correcto, confirmará el envío de la petición. UserService se comunicará con FriendshipRequestService para comprobar si la petición entre ambos usuarios existe. Al informarle de vuelta a UserService de que no existe este pedirá que se cree la solicitud para ambos usuarios. FriendshipRequestService crea una nueva instancia de la clase FriendshipRequest, y posteriormente llama al método de la base de datos FirebaseFirestore para almacenarlo. Cuando haya quedado guardada la nueva petición, se enviarán mensajes de confirmación de vuelta hasta mostrarle al usuario un mensaje informativo indicando que la solicitud se ha creado con éxito.

Diapositiva 19: Mockups interfaz

Para terminar esta sección de modelado y diseño, se construyeron bocetos de la interfaz de usuario para visualizar el diseño y flujo entre pantallas antes del desarrollo y validarlo con el cliente. Aquí podemos ver unos cuantos ejemplos.

Durante cada iteración, ha sido necesario mejorar partes del modelado y diseño del sistema.

Diapositiva 20: Desarrollo e implementación

Como hemos mencionado anteriormente, la aplicación se ha desarrollado utilizando Flutter junto a Firebase. Estas tecnologías suelen aplicarse conjuntamente muy a menudo, por lo que la creación y configuración del proyecto ha sido sencilla.

Diapositiva 21: Firebase Authentication

Para la autenticación de los usuarios, hemos utilizado Firebase Authentication. Esto nos ha permitido implementar el método de acceso a la aplicación mediante correo electrónico y contraseña, configurar plantillas de correo electrónico para la verificación de la cuenta y el restablecimiento de la contraseña, y por último

definir una política de contraseñas robusta en la que se exigen al menos 8 caracteres de los cuales tiene que haber uno en mayúscula, otro en minúscula, uno numérico y otro que sea un carácter especial.

Firebase Authentication almacena las contraseñas internamente de forma segura, aplicandoles un hash. El script mediante el que se hace esto tiene unos parámetros que dependen de cada proyecto, lo cual aporta una capa más de seguridad.

Otro aspecto de la seguridad que queremos comentar es que por ejemplo a la hora de restablecer la contraseña debemos introducir el correo electrónico de la cuenta. Aunque la cuenta introducida en el sistema no exista, Firebase Authentication no lanzará ningún error. Esto es de forma intencional, ya que lo hace para evitar ataques de enumeración de cuentas. De forma que los atacantes no pueden comprobar si una dirección de correo electrónico está registrada en la aplicación.

Diapositiva 22: Firebase Cloud Firestore

Otro servicio que hemos empleado es Firebase Cloud Firestore. Se trata de una base de datos no relacional en la que tenemos para cada entidad una colección compuesta por documentos con datos.

Diapositiva 23: Notificaciones

Para cada actividad, el usuario puede decidir si quiere recibir un recordatorio a una hora concreta. Para implementarlo, hemos hecho uso del paquete Flutter Local Notifications, el cual nos permite establecer un canal para programar el envío de las notificaciones al dispositivo del usuario. Una vez programadas, las notificaciones se mostrarán incluso si la aplicación está cerrada.

Diapositiva 24: Widget (eliminada)

Uno de los requisitos opcionales que se ha definido es la implementación de un widget para la pantalla de inicio desde el que el usuario pudiese ver sus actividades del día de hoy junto a su progreso. Esto lo hemos implementado gracias al paquete Home Widget, que facilita la creación de widgets en la pantalla mediante código nativo.

Diapositiva 25: Interfaz

Flutter utiliza bloques de construcción para crear la interfaz, llamados widgets. Siguen una estructura jerárquica basada en composición, de forma que cada widget está dentro de otro widget padre. Existen widgets con estado y sin estado, siendo los primeros aquellos cuyas propiedades pueden verse modificadas por la interacción del usuario. Algunos de los widgets que más hemos usado han sido el Scaffold que crea la estructura visual básica, StreamBuilder para mostrar los datos mediante un stream y que aparezcan constantemente actualizados y FlutterToast para mostrar avisos en la pantalla a modo de feedback al usuario cada vez que realiza una acción importante. Se ha seguido un diseño minimalista, responsivo y fácil de navegar.

Diapositiva 26: Validación y verificación

Para asegurar la calidad del software desarrollado y comprobar el correcto funcionamiento del sistema, se han realizado diferentes pruebas.

Diapositiva 27: Pruebas unitarias

Las primeras pruebas que hemos realizado han sido las unitarias. Dado que el proyecto presenta unas restricciones de tiempo y recursos, no se ha creado una batería de pruebas exhaustiva de todas las funcionalidades. Nos hemos centrado en testear las del servicio de actividades.

Para cada método de esta clase, se ha creado un test en el que se utiliza un mock de la base de datos. En estos tests se comprueba que para unos datos dados, tras aplicar el método correspondiente, obtenemos los resultados esperados. Esto se hace utilizando la función expect, en la que se compara el valor obtenido con el esperado.

Diapositiva 28: Pruebas de usuario

Las segundas pruebas han consistido en que los usuarios utilicen la versión terminada de la aplicación y posteriormente respondan un formulario que contiene preguntas tratan tanto de las funcionalidades, problemas e interfaz y experiencia de usuario. Además, los usuarios han podido aportar feedback con el fin de poder mejorar la aplicación.

Diapositiva 29: Demo de la aplicación

Diapositiva 30: Conclusiones

Para terminar, quiero comentar que el proyecto ha concluido superando las expectativas y objetivos propuestos, convirtiéndose en una aplicación muy completa como se ha podido ver.

Uno de los principales objetivos que he mencionado fue el de aprender nuevas tecnologías populares para el desarrollo móvil como Flutter y Firebase. Gracias al estudio inicial que realicé, una vez se comprendieron sus bases el desarrollo fue en general fluido. Sin embargo como en todo proyecto, me he enfrentado a varios retos, como a la hora de implementar las notificaciones de las actividades o el requisito opcional del widget de la aplicación. Considero que este proyecto me ha permitido poner en práctica gran parte de los conocimientos que he adquirido en la carrera.

Diapositiva 31: Líneas futuras

Como trabajos futuros existe un gran margen de mejora para la aplicación. Gracias a que Flutter es un framework multiplataforma, se podría:

- Desarrollar una versión para iOS a partir de lo que ya tenemos.
- Añadir y ampliar las funcionalidades existentes como mejorar los desafíos de actividad incluyendo datos como el progreso de ambos usuarios, añadir un archivo de actividades de las que no se quiera seguir realizando un

seguimiento, notificaciones para las solicitudes y los mensajes, nuevas estadísticas o un sistema que permita a los usuarios proponer plantillas de actividad.

Diapositiva 32: Fin

Eso ha sido todo, gracias por vuestra atención.

Cosas que debo llevar abiertas

- Presentación
- Memoria
- VSCode del proyecto abierto
- Firebase console
- Repositorio de Github
- Trello
- Formulario
- Cable teléfono, consola y opciones de desarrollador activadas con la depuración por USB
- Ocultar iconos del escritorio y poner fondo claro

Posibles preguntas y respuestas

De lo que debo hablar

- Presentar el trabajo
- Motivación y objetivos
- Tecnologías y herramientas
- Metodología de trabajo
- Análisis y especificación de requisitos
- Casos de uso
- Diagrama de clases
- Diagramas de secuencia
- Diseño de la UI (mockups)
- Estructura proyecto
- Funcionalidades
- UI/UX
- Validación y verificación
- Conclusiones
- Trabajos futuros
- Hablar de los entregables y/o apéndices?

Búscate una plantilla bonita y más o menos los contenidos que debes incluir son los siguientes:

1. Motivación (de dónde surge la idea del TFG, qué problema intenta resolver, contexto del problema, ...)

2. Objetivos (contar en una diapositiva en qué ha consistido el TFG; como si se lo quisieras explicar a alguien; no son necesarios detalles técnicos)

A partir de aquí comienza la parte técnica:
3. Tecnologías y herramienta utilizadas
4. Especificación y análisis (actores del sistema, requisitos, casos de uso principales, ...)
5. Diseño (modelo de datos, diseño arquitectónico, ...)
6. Implementación, pruebas y despliegue

Aquí se hace la demo
7. Conclusiones y líneas futuras (se repiten cuáles han sido los objetivos y si han cumplido o no; ses comentan las dificultades encontradas y cómo las has resuelto; posibles líneas futuras del trabajo)