UNIVERSITAT POLITÈCNICA DE VALÈNCIA

_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE INGENIERÍA DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería de Telecomunicación

Desarrollo de Aplicación de fichaje para los empleados de una empresa en Android.

Trabajo Fin de Grado

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

AUTOR/A: Pereto Figueres, Maria

Tutor/a: Martínez Zaldívar, Francisco José

CURSO ACADÉMICO: 2021/2022

Resumen

Este trabajo de fin de grado se basa en el desarrollo de una aplicación de fichaje para los empleados de una empresa. En dicha aplicación los usuarios podrán iniciar sesión, consultar y modificar sus datos personales y registrar las horas trabajadas, comprobando la localización en el momento del registro de dichas horas y consultar el resultado de su jornada en forma de informes. En función del tipo de usuario, éste tendrá acceso a distintas funcionalidades. El proyecto surge de la necesidad de registrar la jornada real con respecto al horario teórico conforme a la ley.

A su vez, es una forma de mejorar el rendimiento del trabajador. La aplicación se desarrollará para la plataforma Android empleando el lenguaje de programación Java. Se va a estudiar si conviene mejor implementar una estructura básica a modo de un servidor Apache HTTP alojado en un PC y con un gestor de base de datos MySQL o si realizar una integración en Cloud, por ejemplo, con la herramienta Firebase de Google.

Resum

Aquest treball de fi de grau es basa en el desenvolupament d'una aplicació de fitxatge per als empleats d'una empresa. En aquesta aplicació els usuaris, iniciar sessió, consultar i modificar les seues dades personals i registrar les hores treballades, consultant la localització en el moment del registre d'aquestes hores i consultar el resultat de la seua jornada en forma d'informes. En funció del tipus d'usuari, aquest tindrà accés a diferents funcionalitats. El projecte sorgeix de la necessitat de registrar la jornada real respecte a l'horari teòric conforme a la llei. Al seu torn és una manera de millorar el rendiment del treballador.

L'aplicació es desenvoluparà per a la plataforma Android emprant el llenguatge de programació Java. S'estudiarà si convé millor implementar una estructura bàsica a manera d'un servidor Apatxe HTTP allotjat en un PC i amb un gestor de base de dades MySQL o si realitzar una integració en Cloud, per exemple, amb la ferramenta Firebase de Google.

Summary

This final degree project is based on the development of a work tracking app for the employees of a company. In this app, users will be able to log in, check and modify their personal data and record the daily working hours, checking the location at the time of the recording and check reports of their workday.

Depending on the type of user, they will have access to different functionalities. The project arises from the need to record the actual working day with respect to the theoretical schedule according to the law. In turn, it is a way to improve the worker’s performance.

The application will be developed for the Android platform making use of the Java programming language. It will be studied whether it is better to implement a basic

structure such as an Apache HTTP server hosted on a PC and with a MySQL database manager or whether to perform a Cloud integration, for example, with Google's Firebase tool.

Índice Contenido

Índice Figuras

Índice Tablas

1. Introducción

En 1888 Willard Legrand Bundy, joyero e inventor neoyorquino, patentó la primera máquina registradora de tiempo de trabajo. Consistía en un gran reloj mecánico donde cada trabajador disponía de una llave única para marcar las horas de entrada y salida de sus puestos de trabajo. Dicho invento tenía como fin ahorrar el gasto de contratar vigilantes o cronometradores usados generalmente para registrar dichas horas. [1]

En la actualidad se pueden encontrar muchas soluciones en el mercado que permiten registrar las horas trabajadas, desde simples hojas de papel donde el empleado firma su propio registro a aparatos complejos y de gran coste. El progreso tecnológico que ha habido en los últimos años ha generado un gran impacto en el ámbito laboral, mejorando la productividad de los empleados, de tal forma que desde cualquier lugar y cualquier dispositivo se puede proporcionar acceso a aplicaciones que cubren el propósito de estos sistemas, como es el caso de los teléfonos móviles inteligentes.

Hasta 2019, en España, era decisión de cada empresa si implementar o no un sistema de registro de horas de trabajo. Sin embargo, conforme al Real Decreto-Ley 8/2019, de 8 de marzo, de medidas urgentes de protección social y de lucha contra la precariedad laboral en la jornada de trabajo; Las empresas deben garantizar el registro diario de la jornada, incluyendo el horario concreto de inicio y finalización de la jornada, deben conservar los registros durante cuatro años y dichos registros deben permanecer a disposición de los trabajadores, los representantes legales y la Inspección de Trabajo y Seguridad Social. [2]

Una aplicación móvil como solución favorece a los trabajadores móviles, los cuales tienen más de un lugar de trabajo, de forma que no tengan que desplazarse a propósito a una localización específica con el único propósito de fichar. Este es el caso de muchos empleados de oficina que, con la llegada del virus Covid-19 y el cambio social que ha supuesto, han tenido que cambiar de modalidad de trabajo y optar por el teletrabajo. Por otra parte, favorece la flexibilidad laboral (y por ende, la conciliación familiar) y la productividad del empleado ya que hoy en día, en la sociedad que vivimos, prácticamente todo el mundo dispone de un teléfono móvil con el cual se puede registrar el tiempo de forma más precisa.

1.1 Propósito del presente trabajo de fin de grado (TFH)

El principal propósito de este trabajo es la realización de un proyecto de software desde cero, desde el surgimiento de la idea hasta su empaquetado, ofreciendo así un

producto mínimo viable. Pasando por todas las etapas necesarias; estudio y análisis previo, planificación de tiempo y recursos, y el propio desarrollo de código.

1.2 Objetivos

El principal objetivo es implementar una aplicación móvil mínimamente robusta y segura que permita al trabajador de una empresa registrar la jornada real con respecto al horario teórico. Desde su propio teléfono móvil, el trabajador puede registrar y visualizar las horas trabajadas.

2. Metodología

En este apartado se pretende dar una visión general de cómo se va a llevar a cabo la gestión del proyecto. Es importante tener una estructura clara a seguir en el desarrollo de software, de esta forma, se intenta mejorar la productividad y calidad ya que, sin una gestión adecuada, los proyectos de software corren el riesgo de hacer un uso mayor del necesario de tiempo y recursos.

Para el desarrollo de la aplicación, se va tomar como referencia, el ciclo de vida *Rational Unified Process*, comúnmente conocido como RUP. Este modelo es una implementación del desarrollo en espiral. El proceso se divide en cuatro fases y cada una de ellas se descompone en iteraciones, cuyos resultados son la entrega de un producto ejecutable. El modelo tiene como fin intentar detectar posibles defectos en las fases iniciales y anticiparse a futuras necesidades.

- Fase de inicio F1: Se define el alcance del proyecto, se realiza un estudio de su viabilidad y se hace una definición clara del propósito y los objetivos del proyecto.
- Fase de elaboración F2: Se definen los requerimientos y funcionalidades que debe presentar la aplicación a desarrollar y se desarrolla un flujo de trabajo.
- Fase de construcción F3: Implementación de las fases anteriores.
- Fase de transición F4: Puesta en producción.

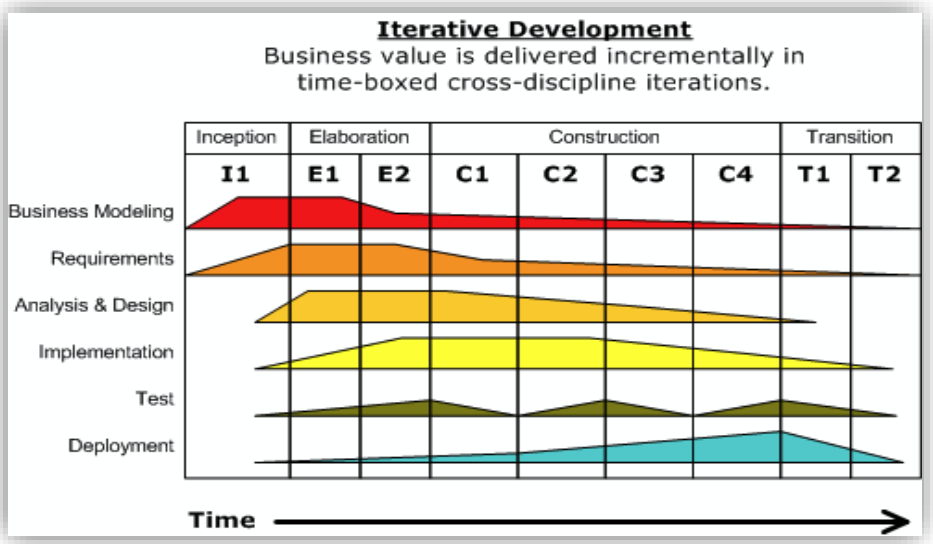

Figura 1. Diagrama RUP [https://fr.wikipedia.org/wiki/Fichier:Development-iterative.png]

3. Requisitos

El desarrollo de software se ve afectado por retrasos y objetivos no alcanzados, muchas veces debido a una mala gestión de prioridades o requisitos ambiguos, cuyo riesgo es la implementación de funcionalidad que no cubre las necesidades, y en consecuencia, supone costes adicionales de implementación y pruebas. Como herramientas contra esos obstáculos existen técnicas de definición de objetivos y priorización que ayudan a avanzar de una forma más eficiente y eficaz.

Haciendo uso de los requisitos SMART, se pueden definir objetivos claros y alcanzables. SMART es un acrónimo de *Specific* 'específico'; *Measurable* 'medible'; *Attainable/Achievable/Appropriate* 'alcanzable'; *Realistic* 'realista' y *Time-bound /traceable* 'con límite de tiempo'. Estos requisitos tienen la ventaja de ser fáciles de entender e implementar, para ello deben ser concretos y estar bien definidos. [4]

Por otra parte, existe el método MoSCoW. Es una técnica que sirve para categorizar cada requisito asignando una prioridad. Consiste en dividir los requisitos en cuatro categorías diferentes: *Must have* 'debe tener' son críticos e imprescindibles; *Should have* 'debería tener' no son imprescindibles; *Could have* 'podría tener' estaría bien implementarlas, pero no son críticas y *Won't have right now* 'no tendrá en estos momentos' pero se puede considerar más adelante. [5]

3.1 Requisitos funcionales

Un requisito funcional es una especificación de una función que el sistema, en este caso la aplicación a desarrollar debe ser capaz de realizar.

Pantalla Inicio de Sesión:

1. La aplicación debe permitir al usuario iniciar sesión mediante usuario/contraseña. El administrador es quien registra al usuario.
2. La aplicación debe mantener la sesión del usuario iniciada hasta que el usuario decida cerrar su sesión
3. Si el usuario dado de alta previamente no dispone de una contraseña, la aplicación debe permitir al usuario reiniciar su contraseña.
4. Si el usuario dado de alta previamente dispone de una contraseña, pero no la recuerda, la aplicación debe permitir al usuario reiniciarla navegando hasta la pantalla de Reinicio de Contraseña.

Pantalla Home:

5. En la pantalla de Home aparecerán las acciones disponibles para el usuario
6. En la pantalla de Home la aplicación debe permitir al usuario desplazarse a la pantalla Perfil.
7. En la pantalla de Home la aplicación debe permitir al usuario desplazarse a la pantalla Fichar.
8. En la pantalla de Home la aplicación debe permitir al usuario desplazarse a la pantalla Configuración.
9. En la pantalla de Home la aplicación debe permitir al usuario desplazarse a la pantalla Acerca De.
10. En la pantalla de Home la aplicación debe mostrar un reloj y la fecha actual según la localización del servidor de base de datos.
11. En la pantalla de Home la aplicación debe permitir al usuario cerrar sesión y desplazarse a la pantalla de Inicio de Sesión.

Pantalla Fichar:

12. En la pantalla Fichar, la aplicación debe permitir al usuario registrar el fichaje de entrada mediante un botón dedicado a esa acción.
13. En la pantalla Fichar, la aplicación debe permitir al usuario registrar el fichaje de salida informando el motivo y comentario oportuno mediante un botón dedicado a esa acción.
14. En la pantalla Fichar, la aplicación debe permitir al usuario visualizar los registros del día actual.
15. En la pantalla Fichar, la aplicación debe mostrar un reloj y la fecha actual según la localización del servidor de base de datos.
16. En la pantalla Fichar la aplicación debe permitir al usuario regresar a la pantalla de Home.

Pantalla Configuración:

17. En la pantalla Configuración la aplicación debe permitir al usuario cambiar el tema de la interfaz de usuario (modo día/noche).
18. En la pantalla Configuración la aplicación debe permitir al usuario actualizar su contraseña.
19. En la pantalla Configuración la aplicación debe permitir al usuario regresar a la pantalla de Home.

Pantalla Consultar Fichajes:

20. En la pantalla Fichar, la aplicación debe mostrar la fecha actual según la localización del servidor de base de datos.
21. En la pantalla Consultar Fichajes el usuario debe poder consultar los fichajes en un rango de tiempo específico
22. En la pantalla Consultar Fichajes el usuario debe poder visualizar con detalle información específica de un fichaje entrada-salida.

23. En la pantalla Consultar Fichajes el usuario debe poder descargar en un fichero con extensión '.csv' información de los fichajes consultados por rango de fechas.
24. En la pantalla Consultar Fichajes el usuario debe poder exportar un fichero '.csv' de la información de los fichajes consultados por rango de fechas
25. En la pantalla Consultar Fichajes la aplicación debe permitir al usuario regresar a la pantalla de Home.

Pantalla Acerca De:

26. En la pantalla Acerca De la aplicación muestra al usuario información acerca de la aplicación.
27. En la pantalla Acerca De la aplicación debe permitir al usuario regresar a la pantalla de Home.

Pantalla Perfil:

28. En la pantalla Perfil la aplicación muestra al usuario datos propios del usuario (nombre, apellidos, correo electrónico alternativo de contacto, teléfono de contacto principal….
29. En la pantalla Perfil la aplicación muestra al usuario datos propios de su perfil de empleado (rol en la empresa, tiempo trabajado en la empresa…)
30. En la pantalla Perfil la aplicación debe permitir al usuario regresar a la pantalla de Home.

3.2 Requisitos no funcionales

Disponibilidad:

1. La aplicación debe estar disponible las 24/7

Desempeño

2. La aplicación debe ser usable aun no teniendo acceso a internet en el dispositivo. Sin embargo, algunas funcionalidades dejarán de ser operativas.
3. El tiempo de respuesta a las interacciones del usuario debe ser mínimo.
4. La aplicación debe estar disponible en español e inglés. Si el idioma del dispositivo es español, debe emplear el idioma español, en caso contrario por defecto debe emplear el idioma inglés con la excepción del texto mostrado para representar el día actual qué debe mostrarse según el idioma nativo del dispositivo.
5. La aplicación no debe de requerir un manual de usuario. La aplicación debe ser intuitiva.

Seguridad:

6. Un usuario no registrado en el sistema de autenticación no debe tener acceso a las colecciones y documentos de la base de datos.

3.3 Requisitos adicionales

Adicionalmente se definen una serie de requisitos no prioritarios que podrían implementarse en el desarrollo de la aplicación.

1. Fichar automáticamente en función del tiempo sin que el usuario tenga la aplicación en primer plano.
2. Fichar automáticamente en función de la ubicación GPS sin que el usuario tenga la aplicación en primer plano.
3. Si el usuario tiene asignado un modo de trabajo específico, debe poder fichar entrada solamente cuando esté dentro de un rango de distancia a su lugar de trabajo.

4. Diseño

4.1 Soluciones Firebase

Se ha elegido esta herramienta porque ofrece un modelo Backend-as-a-Service (BaaS) para el desarrollo de aplicaciones. Firebase ofrece un plan sin coste de múltiples servicios y conforme se escala la aplicación ofrece un plan de pago sobre la marcha, a medida que crece el proyecto.

Products	No-cost Spark Plan Generous limits to get started	Pay as you go Blaze Plan Calculate pricing for apps at scale ✓ No-cost usage from Spark plan included*
A/B Testing	No-cost	
Analytics	No-cost	
App Distribution	No-cost	
App Indexing	No-cost	
Authentication		
Phone Auth - US, Canada, and India	10k/month	$0.01/verification
Phone Auth - All other countries	10k/month	$0.06/verification
Other Authentication services	✓	✓
Cloud Firestore		
Stored data	1 GiB total	No-cost up to 1 GiB total Then $0.108 per additional GiB
Network egress	10 GiB/month	No-cost up to 10 GiB/month Then Google Cloud pricing
Document writes	20K writes/day	No-cost up to 20K writes/day Then Google Cloud pricing
Document reads	50K reads/day	No-cost up to 50K reads/day Then Google Cloud pricing
Document deletes	20K deletes/day	No-cost up to 20K deletes/day Then Google Cloud pricing

Figura 2. Plan de costes Firebase

4.1.1 Cloud Firestore

Para este proyecto, se ha hecho uso del servicio de autenticación y base de datos que Firebase ofrece. De esta forma se ha desarrollado una aplicación cliente sin necesidad de desarrollar un servidor. El producto Cloud Firestore permite tener la base de datos en la nube y el servicio de autenticación ofrece múltiples tipos de registro e inicio de sesión.

La base de datos de Cloud Firestore es una base de datos NoSQL orientada a documentos. A diferencia de las bases de datos relacionales, no hay tablas ni filas en ella. La información se almacena en documentos, organizados en colecciones y cada documento consta de juegos de pares de clave-valor.

Una desventaja que tiene el servicio de base de datos es la falta de control cuando más de un usuario realiza operaciones de escritura al mismo tiempo. Sin embargo, para el modelo de datos elegido esto no supone un problema porque cada usuario solo puede acceder a sus documentos y no está contemplado tampoco que un usuario haga operaciones de escritura en más de un dispositivo a la vez.

Otra desventaja es que las consultas realizadas a la base de datos solo pueden obtener documentos en una colección o subcolección. Es decir, si se necesita información que está almacenada en un documento o documentos de otra colección, se necesita un consulta adicional para acceder a esa información. La base de datos de Firestore está optimizada para ser capaz de almacenar grandes colecciones de documentos pequeños. Esto ha condicionado el diseño del modelo de datos que se ha decidido implementar, de forma que se pudiese garantizar un acceso de escritura y lectura rápido y minimizando el número de operaciones.

Una ventaja importante a destacar en la base de datos de Firestore es que en una misma colección, distintos documentos pueden contener distintos juegos de campo-valor. También se pueden añadir dinámicamente nuevos campos en un documento. Esto puede suponer una desventaja cuando se tiene una estructura compleja de documentos, pero en el caso presente de un desarrollo inicial donde ha habido cambios importantes a lo largo del ciclo de vida del desarrollo ha sido ventajoso.

Teniendo en cuenta estas consideraciones, se ha seguido adelante con esta herramienta porque en su conjunto, el servicio que ofrece se adapta a las necesidades de la aplicación a desarrollar además de ser fácil de implementar.

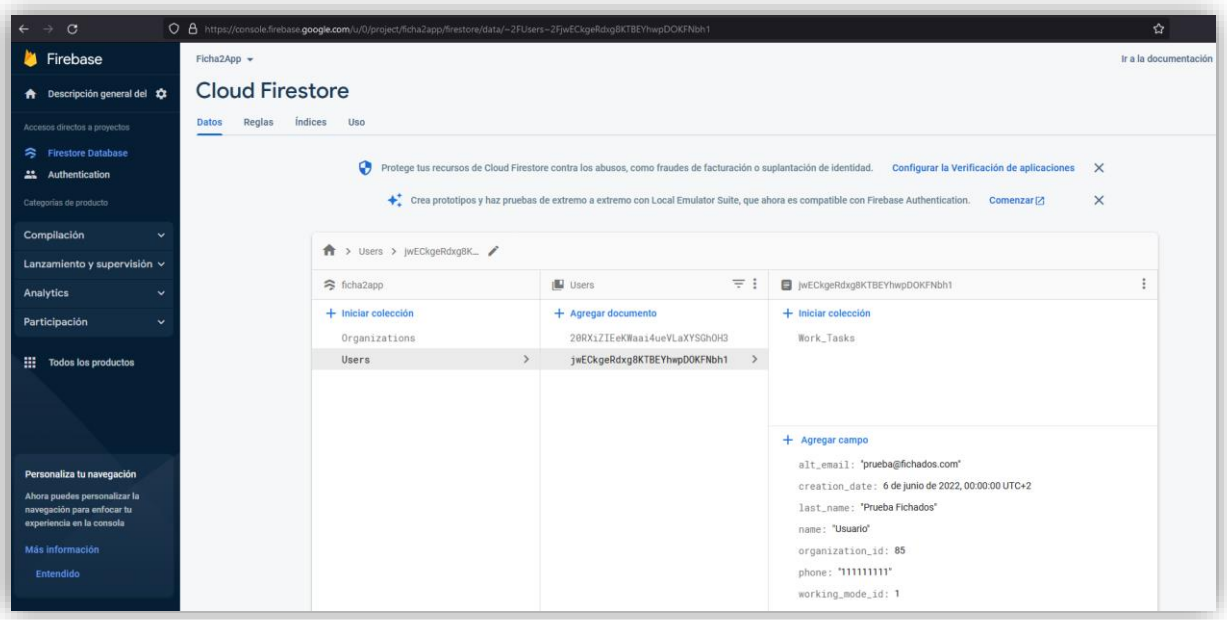

Figura 3. Cloud Firestore

4.1.2 Firebase Authentication

El sistema de autenticación de Firebase ofrece métodos para crear, autenticar y gestionar usuarios dentro del proyecto de Firebase. Ofrece la posibilidad de integrarlo con varios proveedores como Gmail, Facebook, Twitter… Pero para este proyecto se da por hecho que el usuario dispone de una cuenta corporativa y que un usuario administrador ha dado de alta el empleado en la aplicación.

Si es la primera vez que el usuario se autentica, este puede hacer click sobre 'He olvidado mi contraseña', introducir su cuenta de correo, y sí esta está dada de alta en el sistema de autenticación de Firebase, el usuario recibirá en el correo electrónico suyo un correo con un enlace para restablecer su contraseña.

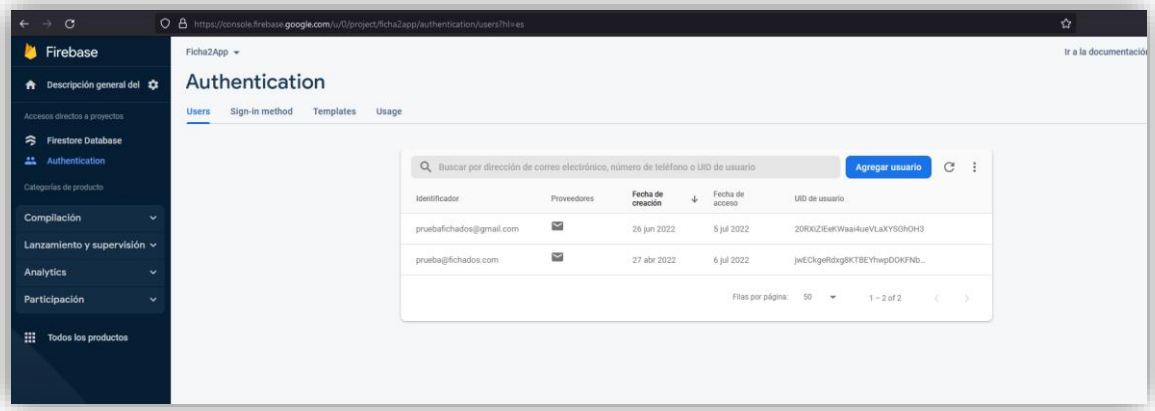

Figura 4. Firebase Authentication

4.2 Modelo de datos

Para empezar a detallar cómo está diseñada la base de datos se va a resumir una serie de conceptos clave:

Dentro de este tipo de base de datos, la unidad de almacenamiento es un documento. Un documento contiene campos con valores asignados. Los objetos anidados en un documento son mapas. Son parecidos a los objetos JSON pero con ligeras diferencias. Por ejemplo, el tamaño del documento está limitado a 1MB, y los mapas admiten tiposde datos adicionales, pero en general se pueden interpretar como objetos JSON.

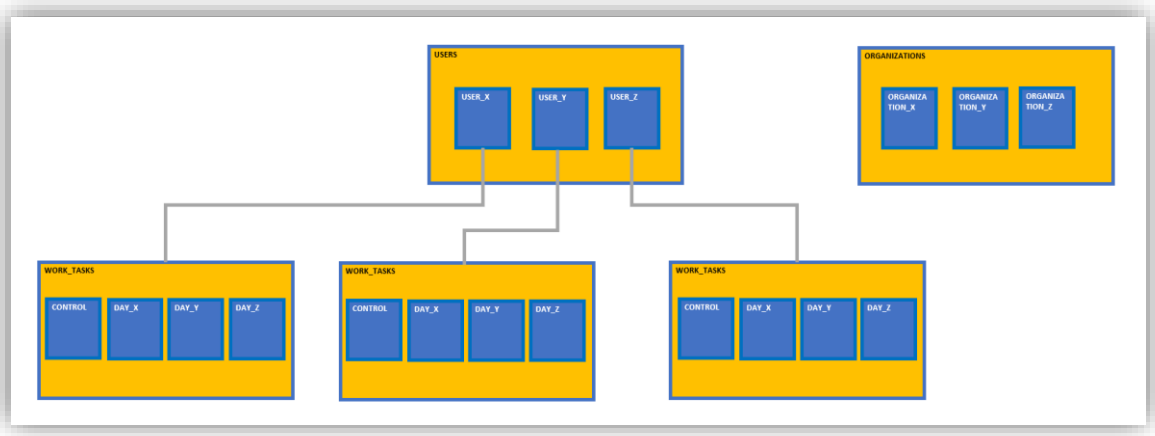

Figura 5. Modelo de datos

Para tratar los datos obtenidos en las consultas a Firebase, en un principio, se iba a implementar objetos JSON para almacenar el documento entero, pero al final se optó por almacenar la información en objetos POJO (Plain Old Java Object), de forma que los datos fueran más accesibles y comprensibles. Para poder implementar esta estructura, en Android, los objetos a almacenar dicha información deben tener un constructor vacío y métodos 'get' para cada campo del documento de base datos.

Documento	Campo	Nivel Profundidad	Tipo	Descripción
User	name	1	String	Nombre del usuario (empleado)
	last_name	1	String	Apellidos del usuario

	alt_email	1	String	Correo electrónico alternativo
	phone	1	String	Teléfono de contacto
	organization_id	1	Número (Long)	ID que representa la organización a la que el usuario pertenece
	working_mode_id	1	Número (Long)	ID que representa el modelo de trabajo del empleado (Oficina, teletrabajo…)
	creation_date	1	Timestamp	Fecha creación del usuario
Organization	name	1	String	Nombre de la organización
	office_location	1	Geopoint	Geolocalización de la organización
	clock_in_distance_meters	1	Número (Long)	Rango de distancia en metros en el cual un usuario en modo oficina tiene habilitada la opción de fichar entrada
	start_active_date	1	Timestamp	Marca de tiempo en el que la organización está activa
Control	last_ctl_timestamp	1	Timestamp	Fecha actualizada del sistema en el momento de su lectura
Work_Task	document_name	1	String	Nombre del documento
	last_in_flag	1	String	Flag Y/N que determina si el último fichaje que hubo en el día fue de entrada
	work_timestamp	1	Timestamp	Marca de tiempo del documento de fichaje
	last_work_ctl	1	Número (Long)	Número de control para identificar dentro del documento cada par fichaje entrada-salida (id secuencial)
	fichajes	1	Map	Lista de fichajes (Mapa)
	id secuencial	2	Map	Número de control para identificar dentro

				del documento cada par fichaje entrada-salida (id secuencial) (Mapa)
	comment	3	String	Comentario que el usuario introduce en el fichaje de salida
	out_reason_id	3	Número (Long)	Identificador del motivo de salida
	timestamp_in	3	Timestamp	Marca de tiempo del fichaje de entrada
	timestamp_out	3	Timestamp	Marca de tiempo del fichaje de salida
	work_place_id	3	Número (Long)	Identificador del modo de trabajo (Oficina, teletrabajo...)

Tabla 1. Modelo de datos

Se ha creado la colección Usuarios, la cual contiene documentos que identifican a cada usuario. Cada documento contiene información clave del usuario. Cada documento Usuario se identifica de forma unívoca en toda la base datos mediante un id autogenerado por Firebase cuyo origen es el sistema de autenticación.

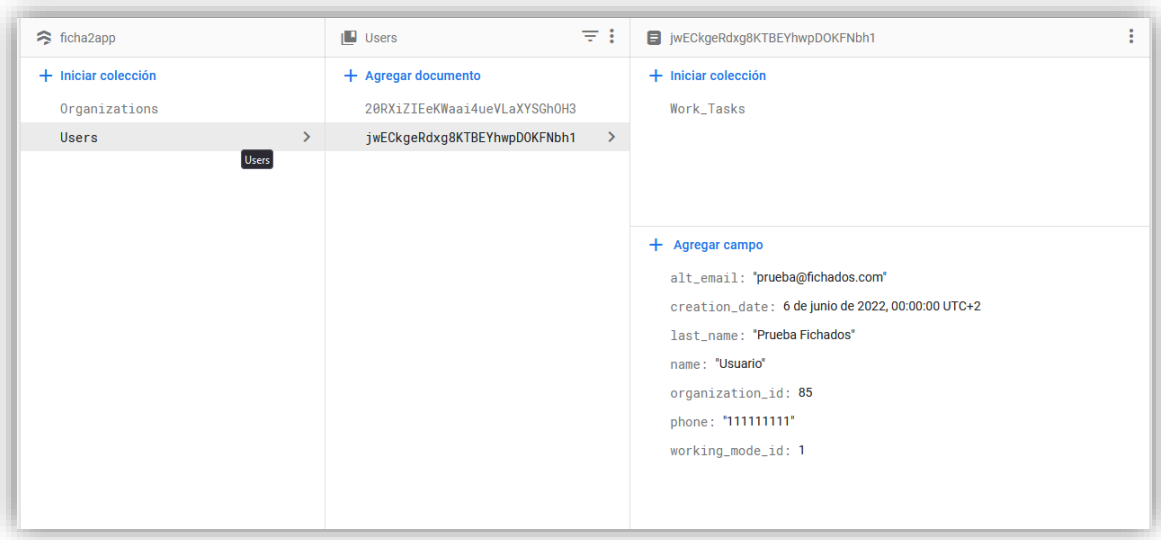

Figura 6. Extracto Documento Usuario

El documento Usuario tiene referenciado a su vez una colección llamada Tareas de Trabajo, la cual contiene documentos que representan cada día de fichaje. Esta colección contiene adicionalmente un documento llamado Control, el cual se usa para calcular la fecha actual del sistema según las necesidades de la aplicación.

Los documentos que contienen información de cada día de fichaje se identifican por su nombre único solamente dentro de la colección de Tareas de Trabajo. Esto es una decisión de diseño en la que se ha considerado que dado que el documento de fichaje está anidado en una colección referenciada por un usuario único en el sistema no va a haber ambigüedad a la hora de acceder a dicho documento, ya sea en operación de escritura o lectura. Por lo tanto, se ha optado por identificar cada fichaje de usuario por el día en que se crea el documento en un formato 'yyyymmdd' (año, mes y día concatenado).

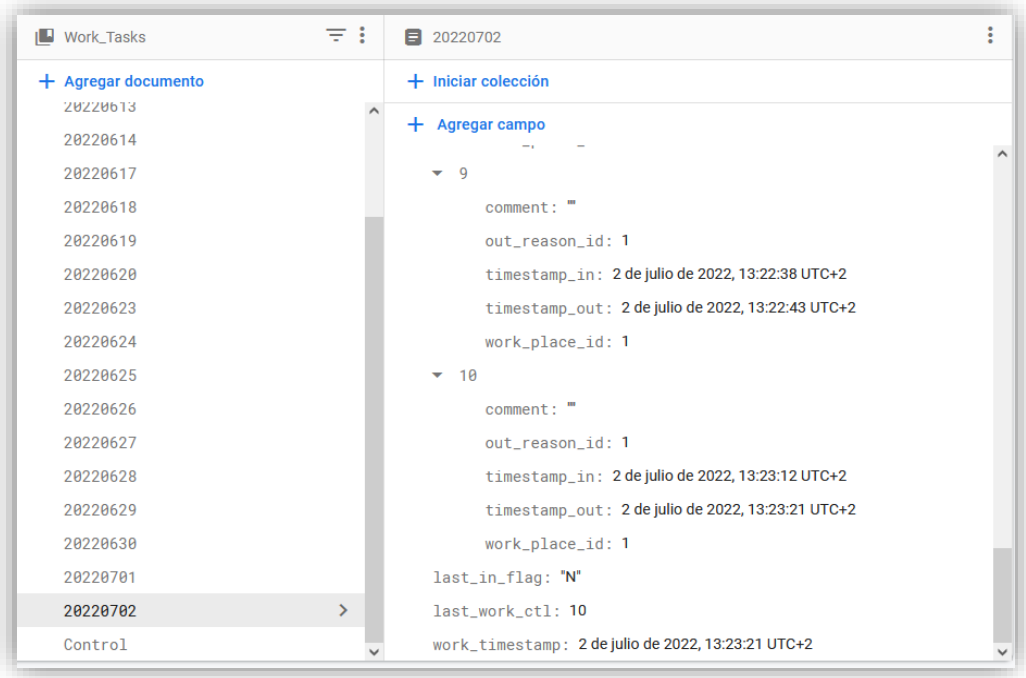

Figura 7. Extracto Documento Fichaje

En la colección de organizaciones, la organización se identifica por el nombre del documento. Se podría haber optado por un identificador automático, pero para poder identificar fácilmente la organización, se ha definido un nombre numérico único.

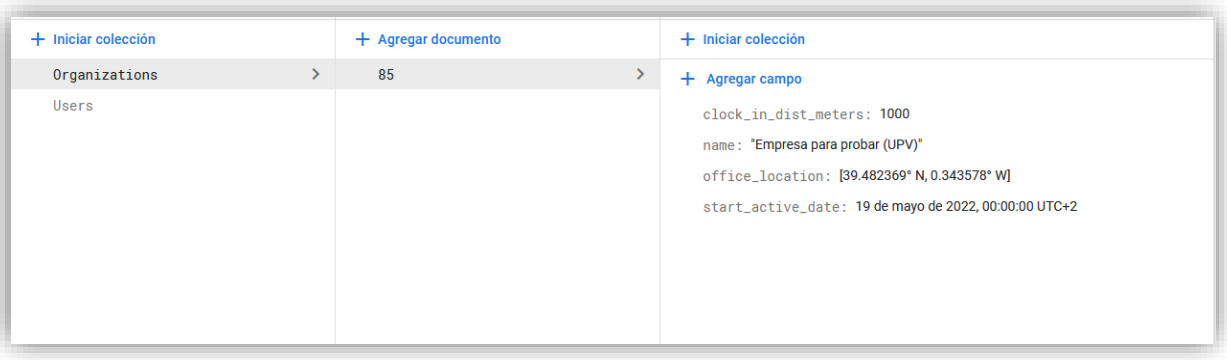

Figura 8. Extracto Documento Organización

Se debe tener en cuenta que una organización no es sinónimo de empresa. Se puede interpretar como empresa, oficina… Es decisión de la empresa que adquiere la aplicación.

5. Implementación

5.1 Entorno de desarrollo

El desarrollo se ha llevado a cabo en el entorno de desarrollo Android Studio. Esta herramienta ofrece, entre otras cosas, un editor de código, emulador, gestor de recursos para la aplicación, integración con Github… Es una herramienta bastante completa y potente que facilita el proceso de desarrollo.

El idioma de la aplicación por defecto es el inglés. Sin embargo, mediante el uso de ficheros con extensión '.xml' en directorios de la aplicación, es posible almacenar textos en diferentes idiomas. Para el desarrollo de esta aplicación se ha especificado una serie de *strings* (cadenas de texto) en inglés con su correspondiente traducción al español de forma el idioma de los textos presentados en la aplicación dependa de la configuración del idioma del dispositivo del usuario.

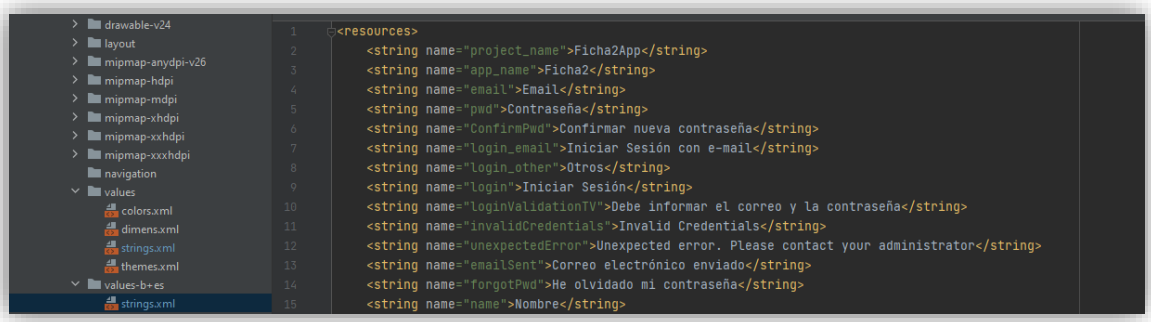

Figura 9. Fichero strings.xml

Por otra parte, en la aplicación se da la posibilidad de configurar el tema de la aplicación (Modo Día/Noche). Esto se ha conseguido utilizando una combinación de los ficheros colors.xml y themes.xml para gestionar los colores que se muestran en la aplicación dependiendo del tema escogido.

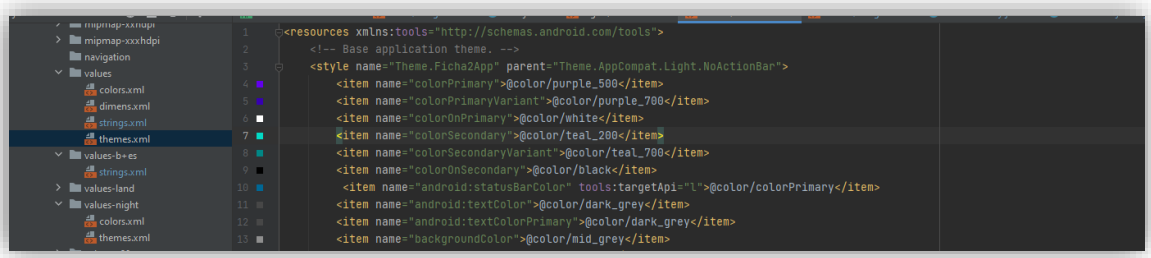

Figura 10. Fichero themes.xml

5.2 Fecha y Hora

En la aplicación se depende de la fecha y hora en distintas operaciones. En la pantalla de Home, la pantalla que permite al usuario fichar muestra de forma actualizada la fecha y hora con notación UTC (*Universal Time Coordinated).* En la pantalla de consulta de fichajes también se muestra la fecha actualizada.

Por otra parte, la fecha se emplea para decidir en qué documento se almacena la información de los fichajes, por lo tanto, es importante tener una fuente fiable a la hora de usar dicha fecha. Por ese motivo se decide descartar la fecha del dispositivo actual ya que es fácil cambiarlo a ajuste manual y que el usuario manipule la hora a su antojo.

A raíz de eso, se ofrece como alternativa la obtención de la fecha que consta en la base de datos. En Firestore no existe una variable del sistema que albergue tal información. Sin embargo, Firestore permite escribir en base de datos, dígase en el campo valor de un documento la fecha actual del servidor en el momento de la escritura. Por lo tanto, una forma de obtener la fecha es realizar una operación de escritura y otra de lectura sobre el mismo campo. Esto es una forma fiable de obtener la fecha, pero para no abusar de las operaciones de escritura y lectura sobre la base de datos se ha decidido buscar una nueva alternativa.

La nueva alternativa se basa en consultar mediante protocolo UDP la fecha y hora en servidores NTP los cuales están dedicados a ofrecer una fecha actual precisa. Debido a que se necesita obtener la fecha y hora de forma actualizada para poder mostrarla en la aplicación habría que hacer cada minuto una petición a los servidores NTP. Se corre el riesgo de que el servidor de internet interprete que estamos realizando un ataque de denegación de servicio al hacer demasiadas peticiones seguidas por lo que se ha optado finalmente por la siguiente vía:

Se guarda en un fichero dentro de la aplicación una serie de URLs de servidores NTP. Se itera dicho fichero para obtener la fecha y hora especificando un tiempo de timeout para evitar la latencia en la aplicación. Si no se consigue obtener una fecha mediante esa iteración, la aplicación realiza una operación de escritura y lectura en base de datos para obtener la fecha.

Con esa fecha y hora, se consulta la fecha actual del dispositivo y se calcula la diferencia entre ambas fechas guardando esa diferencia en una variable llamada offset y se comprueba si el reloj del dispositivo está atrasado o adelantado con respecto a la fecha obtenida del servidor. En función de esto trabajaremos con la hora local del dispositivo sumándole o restándole ese offset y así evitar estar continuamente haciendo consultas a los servidores. De forma asíncrona, cada cierto tiempo se vuelve a consultar la hora del servidor y a realizar el mismo cálculo con la hora local del dispositivo.

Si por algún motivo la fecha obtenida de los servidores NTP fuera errónea, esto no afectaría ni en la creación del documento de la base de datos ni en los tiempos registrados en los fichajes ya que dependen únicamente de la fecha del servidor de Firestore.

Para obtener la fecha y hora es necesario crear un hilo en paralelo porque se necesita acceder a los servicios de red que ofrece el dispositivo y el sistema no permite las operaciones de red en el hilo principal. Para controlar la gestión de los hilos se emplean las interfaces *Runnable* y *Executor*.

A su vez, se necesita declarar en el fichero AndroidManifest.xml el permiso de acceso a internet:

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

```
public void startClock(){
    mBroadcastReceiver = (BroadcastReceiver) (ctx, intent) -> {
            if (intent.getAction().compareTo(Intent.ACTION_TIME_TICK) == 0){
                if (mCustomDateObj!=null){
                    horaTV.setText(Utils.getDateStr(mCustomDateObj.getNewCustomDate(), Utils.TIME_FORMAT_HOUR_MIN,getApplicationContext()));
                }
            }
    };
    registerReceiver(mBroadcastReceiver, new IntentFilter(Intent.ACTION_TIME_TICK));
}
public void setTime() {
    Runnable mRunnable = () -> {
        Date serverDate = null;
        while (serverDate == null) {
            serverDate = CurrentTimeManager.getTimeNow(mNTPListObject);
        }
        diaTV.setText(Utils.getDateStr(serverDate, Utils.DATE_FORMAT_LONG_DAY_ZONE, getApplicationContext()));
        mCustomDateObj = new CustomDateObj(serverDate);
        mBundle.putParcelable(CustomDateObj.CUSTOM_DATE_OBJECT, mCustomDateObj);
    };
    executor = Executors.newScheduledThreadPool( corePoolSize: 1);
    executor.scheduleAtFixedRate(mRunnable,  1, mNTPListObject.getAwait(), TimeUnit.SECONDS);
}
```

Figura 11. Actualización Fecha y Hora

Se ha creado la clase CurrentTimeManager especificamente para obtener la fecha y hora del servidor NTP o del servidor de base de datos de Firestore (método workAroundDate()). Dicha información se almacena en un objeto CustomDateObj cuya clase se ha creado implementando la interface Parcelable para poder llevar el objeto de una actividad a otra.

Para poder realizar la consulta por UDP se han importado las librerias de org.apache.commons.net.time.TimeUDPClient y en el fichero build.gradle se ha añadido su dependencia implementation 'commons-net:commons-net:3.8.0'

```
private static  Date getTimeNowUDP(String host) {
    Date date=null;
    TimeUDPClient client = new TimeUDPClient();
    try {
        client.setDefaultTimeout(3000);
        client.open();
        date = client.getDate(InetAddress.getByName(host));
        client.close();
    }
    catch (IOException e) {
        e.printStackTrace();
    } finally {
        client.close();
    }
    return date;
}
```

Figura 12. Conexión UDP con servidor NTP

5.3 Geolocalización

En el caso de que un usuario quiera fichar entrada. Se comprueba el modo de trabajo que tiene configurado por defecto (Oficina, teletrabajo…). En el caso de que el modo configurado sea Oficina, el sistema comprueba que el usuario esté dentro de un rango de distancia de la oficina. Para que la aplicación pueda acceder al servicio de geolocalización primero se debe pedir permisos al usuario para poder proseguir.

Para ello en el fichero AndroidManifest.xml se debe declarar el permiso <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION "/>. Este permiso incluye permisos para usar el proveedor de red y el proveedor GPS. Se usa como proveedor el GPS para no depender de la conectividad de red.

```
if (ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED
        && ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_COARSE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
    ActivityCompat.requestPermissions( activity: FicharActivity.this, new String[]{android.Manifest.permission.ACCESS_FINE_LOCATION}, Utils.FICHAR_LOCATION_PERMISSION);
    Toast.makeText( context: FicharActivity.this, "Location permission needed, please grant permission in ...", Toast.LENGTH_SHORT).show();
    return;
}
```

Figura 13. Requerir permisos de localización al usuario

El proceso de obtención de localización consiste en lo siguiente: Se solicitan actualizaciones de localización. Una vez obtenida la localización se comprueba si la distancia hacia el puesto de trabajo, cuyas coordenadas se obtienen previamente por consulta a la base de datos (2) y se guardan como atributo del objeto que representa al usuario. Si la localización es mayor del rango informado en la organización (1), también informado previamente por consulta a la base de datos, se hace saber al usuario que no está dentro del rango permitido para poder fichar. En caso contrario ficha entrada.

Cuando el proceso termina se informa al listener onLocationChanged que debe dejar de escuchar ya que hemos obtenido correctamente la localización.

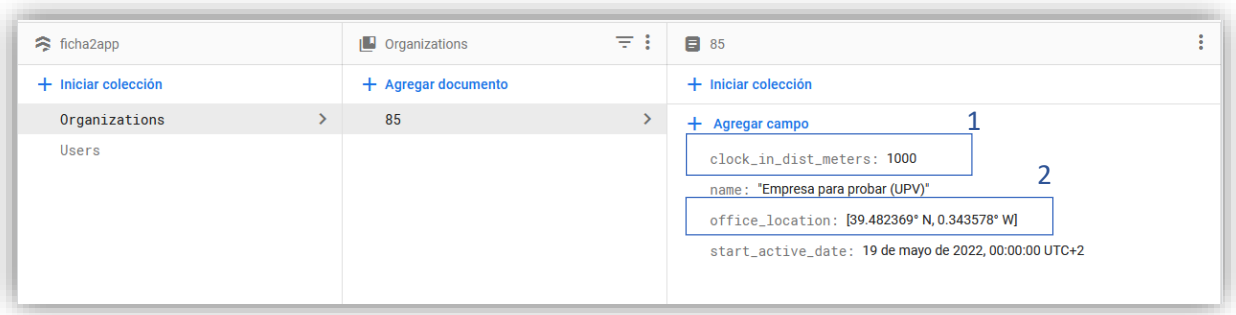

Figura 14. Parámetros localización de la organización

```
mLocationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER,  minTimeMs: 5000,  minDistanceM: 5, new LocationListener() {
    @Override
    public void onLocationChanged(@NonNull Location location) {

        float[] results = new float[1];
        Location.distanceBetween(mUser.getOrganization().getLatitude(), mUser.getOrganization().getLongitude(),
                location.getLatitude(), location.getLongitude(), results);

        if (results[0] > mUser.getOrganization().getClock_in_dist_meters()) {
                final TextView alertTV = new TextView( context: FicharActivity.this);
                AlertDialog.Builder mBuilder = new AlertDialog.Builder( context: FicharActivity.this);
                mBuilder.setTitle("Warning");
                mBuilder.setView(alertTV);
                alertTV.setGravity(Gravity.CENTER);
                alertTV.setText("You are not within range. You need to get closer in ord...");
                mBuilder.setPositiveButton("OK", new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        ((ViewGroup) alertTV.getParent()).removeView(alertTV);
                    }
                });
                mBuilder.show();
            } else {
                clockInFlag = "Y";
                isClicked = true;
                readWorkDoc(Utils.getDateStr(mCustomDateObj.getNewCustomDate(), Utils.DATE_FORMAT_CONCAT_STR_DAY, getApplicationContext()));
            }
            mLocationManager.removeUpdates( listener: this);
        }
});
```

Figura 15. Obtención de la localización

5.4 Actividades

Una actividad es un elemento de la aplicación que interactúa con el usuario. Comúnmente se identifica como pantalla y a lo largo de este documento nos referiremos indistintamente como pantalla o actividad. En su conjunto, las actividades ofrecen una experiencia uniforme cara al usuario, pero la funcionalidad de cada una es independiente de la demás.

El mapa de navegación entre las actividades de esta aplicación consta de 3 niveles de profundidad. Los bloques son las actividades y las líneas son la relación entre ellas. La primera actividad que se ejecuta es MainActivity que puede navegar hacia HomeActivity o LoginActivity en función de si el usuario está autenticado o no. Desde la actividad LoginActivity solo se puede navegar hacia HomeActivity y desde HomeActivity se puede navegar a LoginActivity (si el usuario cierra sesión) o a las distintas actividades del nivel inferior. Las cuales solo pueden navegar de vuelta a la actividad HomeActivity.

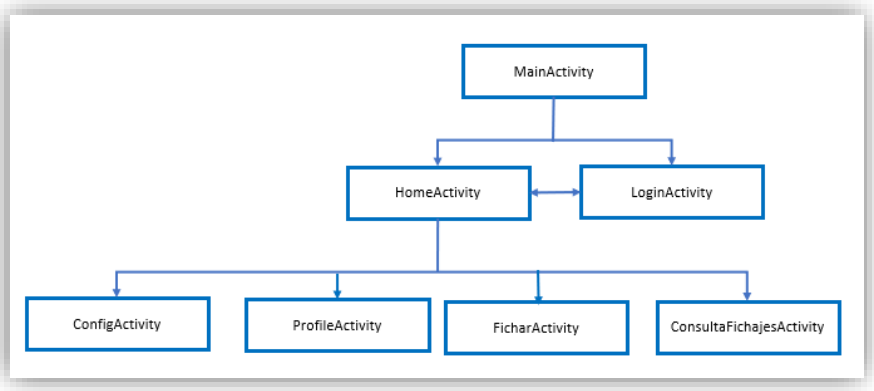

Figura 16. Mapa de navegación en la aplicación

5.4.1 Pantalla Inicial (MainActivity)

Esta actividad muestra la primera pantalla que se carga cuando el usuario abre la aplicación por primera vez o después de haber cerrado la aplicación. Se crea un *bundle*, que es un objeto que se utiliza generalmente para almacenar datos y pasarlos de una actividad a otra. Los objetos cuyas clases se han creado implementan la interfaz *Parceable* para poder pasarlos de una actividad a otra.

Como se indica en el punto 3.3.1 para obtener la fecha y hora es necesario crear un hilo en paralelo porque se necesita acceder a los servicios de red que ofrece el dispositivo y el sistema no permite las operaciones de red en el hilo principal.

Para que el hilo principal se entere de que ya se ha obtenido la fecha se hace uso del objeto *handler*, el cual permite enviar y procesar un objeto de Tipo *Message*. Para procesar dicho mensaje, en el hilo principal se debe redefinir el método *handleMessage* donde se define el código a ejecutar cuando el hilo principal recibe el mensaje.

```
public void setTime() {
    Runnable mRunnable = new Runnable() {
        public void run() {
            Date serverDate = null;
            while (serverDate == null) {
                serverDate = CurrentTimeManager.getTimeNow(mNTPListObject);
            }
            CustomDateObj mCustomDateObj = new CustomDateObj(serverDate);
            mBundle.putParcelable(CustomDateObj.CUSTOM_DATE_OBJECT, mCustomDateObj);
            mHandler.sendEmptyMessage( what: 1);
        }
    };
    executor = Executors.newSingleThreadExecutor();
    executor.execute(mRunnable);
}
```

Figura 17. Nuevo hilo

```
mHandler = new Handler(Looper.getMainLooper()){
    @Override
    public void handleMessage(@NonNull Message msg) {
        if (msg.what == 1){
            findViewById(R.id.loadingPanel).setVisibility(View.GONE);
            if (User.isLoggedUser()) Utils.moveToNextActivity(HomeActivity.class, mBundle,  MainActivity.this);
            else Utils.moveToNextActivity(LoginActivity.class, mBundle,  MainActivity.this);
        }
    }
};
```

Figura 18. Handler

En el hilo principal, donde se gestiona la interfaz de usuario, el usuario percibe una barra de progreso que terminará cuando el hilo secundario avise al principal de que ya ha terminado su función y dará paso a la siguiente pantalla. Si el usuario estaba autenticado en el sistema la siguiente pantalla será la pantalla Home y en caso contrario la pantalla de Inicio de sesión.

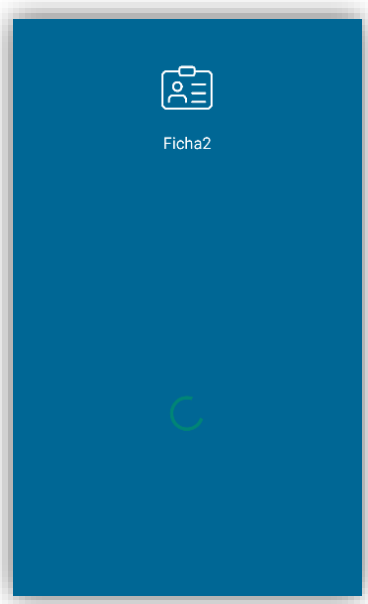

Figura 19. Pantalla Inicial

5.4.2 Pantalla Login (LoginActivity)

Si el usuario no está autenticado, ya sea tras abrir la aplicación o al cerrar sesión desde otra pantalla el usuario llegará a esta pantalla. Al usuario se le presenta una pantalla de inicio de sesión convencional en la que introducir y autenticar sus credenciales con la posibilidad de reiniciar su contraseña en caso de no acordarse de ella. La complejidad de esta actividad reside en la construcción de su interfaz de usuario. El resultado es una pantalla con fragmentos, en este caso dos, donde al hacer click sobre su pestaña o deslizar en una dirección concreta el usuario navega entre fragmentos.

Para conseguir está disposición, en la clase contenedora de la actividad LoginActivity se instacia un objeto de tipo *TabLayout* que permite mostrar pestañas dispuestas horizontalmente, y un objeto*ViewPager2* que permite navegar entre pestañas con un gesto deslizante horizontal del dedo, también conocido como paginación horizontal. Ambos objetos hacen referencia a elementos definidos en el layout de la actividad. También se ha creado una clase que se utiliza como adaptadora. Esta clase extiende a la clase *FragmentStateAdapter* y controla el ciclo de vida de los fragmentos conforme se cambia de pestaña (LOGIN WITH EMAIL o OTHER)

Figura 20. Pantalla Login

Se ha definido el primer fragmento como la clase que se encarga de autenticar al usuario o reiniciar la contraseña en caso de que el usuario no se acuerde de ella. Para autenticar al usuario primero se instancia el servicio de autenticación y cuando el usuario haga click sobre el botón de inicio de sesión se indica al servicio de autenticación las credenciales con las que el usuario quiere acceder. Si las credenciales son correctas se llama a la actividad HomeActivity. Por otra parte, si el usuario no recuerda su contraseña y quiere reiniciarla debe hacer click sobre el botón con texto descriptivo y se abrirá un diálogo informando al usuario de que se enviará un correo a la bandeja de entrada.

El segundo fragmento se ha definido como una clase que muestra simplemente un texto informativo. El planteamiento de este fragmento es dejar abierta la posibilidad de añadir complejidad al servicio de autenticación.

```
loginButton.setOnClickListener(v -> {
    email=inputEmail.getText().toString().trim();
    pwd=inputPwd.getText().toString().trim();
    if (email.isEmpty()|| pwd.isEmpty()){
        fieldValidationTV.setText(getString(R.string.loginValidationTV));
        return;
    }
    fieldValidationTV.setText(getString(R.string.emptyValue));
    mAuth.signInWithEmailAndPassword(email, pwd)
            .addOnCompleteListener(requireActivity(), task -> {
                if (task.isSuccessful()) {
                    // Se hace login con exito
                    Utils.moveToNextActivity(HomeActivity.class, mBundle, getActivity());
                } else {
                    // Se hace saber al usuario que el login ha fallado
                    Toast.makeText( getActivity(), getString(R.string.invalidCredentials), Toast.LENGTH_SHORT).show();
                }
            });
});

    forgotPwdButton.setOnClickListener(v->{
        ResetPwd resetPwdDialog = new ResetPwd();
        resetPwdDialog.show(requireActivity().getSupportFragmentManager(), "Detail");});
```

Figura 21. Autenticación usuario

Figura 22. Diálogo 'He olvidado mi contraseña'

```
        auth.sendPasswordResetEmail(emailAddress)
                .addOnCompleteListener(task -> {
                    if (task.isSuccessful()) {
                        emailValTV.setText(getString(R.string.emptyValue));
                        Toast.makeText( getActivity(), getString(R.string.emailSent), Toast.LENGTH_SHORT).show();
                        dialog.dismiss();
                    } else {
                        emailValTV.setText(getString(R.string.invalidEmail));
                    }
                });
```

Figura 23. Reinicio de contraseña

Como se ha mencionado previamente, las operaciones de red se ejecutan en hilos distintos al principal, de forma asíncrona. Los métodos de Firebase que 'escuchan' la respuesta de los servidores ejecutan el código en el hilo principal.

5.4.3 Pantalla Home (HomeActivity)

A esta actividad se puede acceder desde el resto de las actividades. La pantalla Home hace de menú principal de la aplicación. Existe la posibilidad de configurar ajustes de la aplicación, ver información acerca de la aplicación, ver y editar ciertos datos de perfil, generar nuevos registros de fichaje o consultar fichajes en un rango de tiempo, pudiéndolos exportar o descargar en el directorio de descargas del dispositivo.

Figura 24. Pantalla Home

En la figura 15 se muestra las interacciones del usuario con la pantalla

1. Se muestra fecha y hora (explicado con detalle en el punto 3.3.1).
2. El usuario puede acceder a la pantalla de configuración.
3. El usuario puede abrir un diálogo informativo
4. El usuario puede acceder a la pantalla Perfil.
5. Se muestra nombre y apellidos del usuario.
6. El usuario puede acceder a la pantalla Fichar.
7. El usuario puede acceder a la pantalla Consultar fichajes.
8. El usuario puede cerrar sesión.

5.4.4 Pantalla Configuración (ConfigActivity)

En esta pantalla el usuario puede determinar el tema de la aplicación haciendo click en un switch (on/off). Si el switch está en estado 'on' se establece el tema oscuro en la aplicación. En caso contrario el tema claro.

Por otra parte, el usuario también puede cambiar la contraseña de su cuenta en el servicio de autenticación de Firebase. Cuando el usuario hace click sobre esta opción se abre un díalogo donde debe introducir la contraseña antigua y la nueva dos veces para confirmar su decisión.

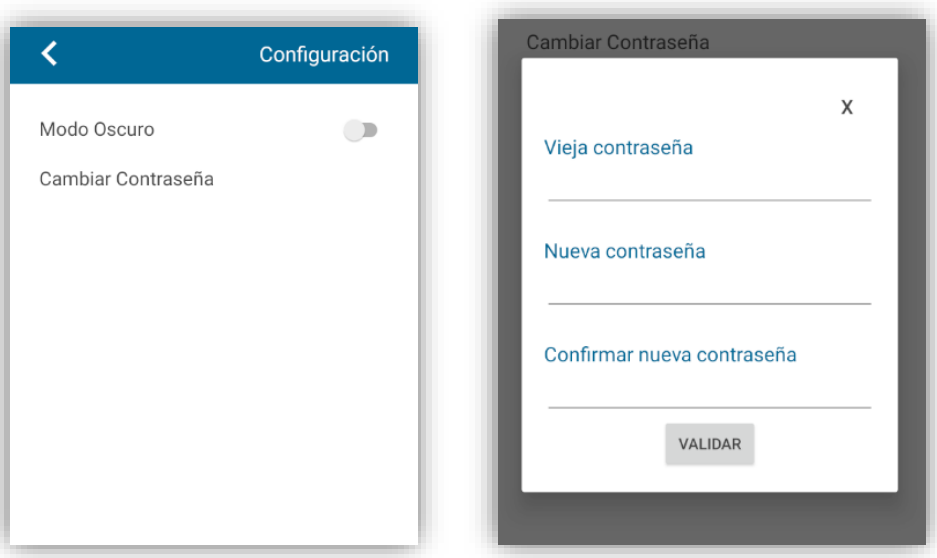

Figura 25. Pantalla Configuración

5.4.5 Pantalla Perfil (ProfileActivity)

En esta pantalla el usuario puede ver ciertos datos de su perfil. Al hacer click sobre el botón editar, se activa el modo edición. En el modo edición el usuario puede editar ciertos datos de su perfil. En particular su dirección de correo alternativa y su número de teléfono. En el modo edición se habilitan dos botones, uno para cancelar y otro para guardar los cambios.

Al guardar los cambios se realiza una operación de escritura en base de datos con la nueva información del usuario.

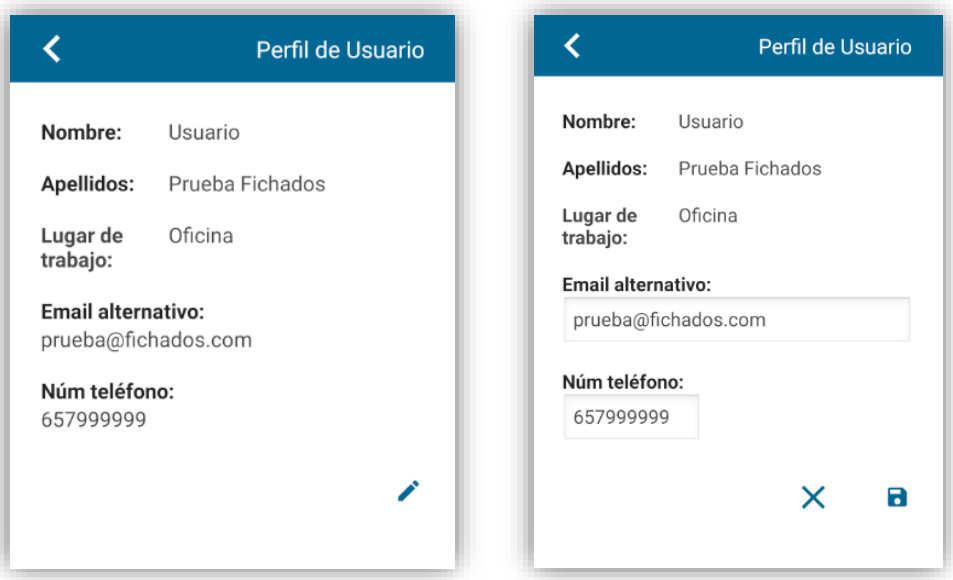

Figura 26. Pantalla Perfil

5.4.6 Pantalla Consultar Fichajes (ConsultaFichajesActivity)

Al entrar en la pantalla Consultar Fichajes, se muestra una lista de los últimos 30 días y las horas registradas cada día. Si el usuario ha registrado fichajes en algún día en particular, se habilita un botón que le permite desplegar ese día concreto y visualizar las horas de entrada y de salida. Mediante su icono asignado se define si el fichaje es de entrada o de salida. A su vez, se habilita un botón para ver más información del par fichaje entrada-salida en cuestión mostrando un diálogo con detalle del fichaje.

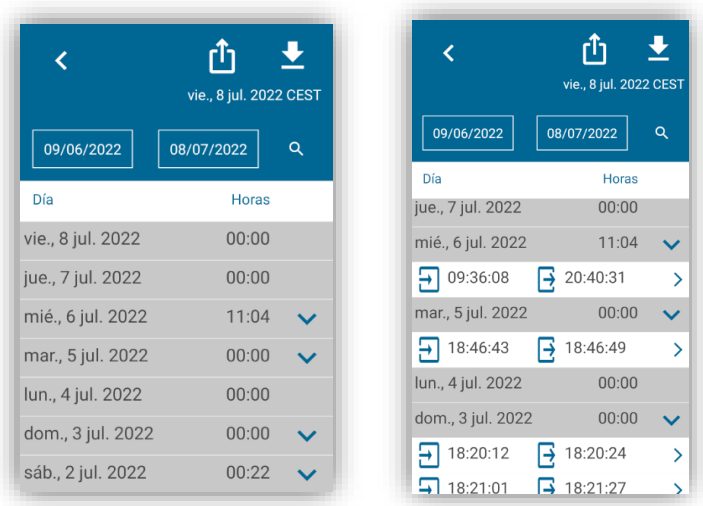

Figura 27. Pantalla Consultar Fichajes

Para representar los elementos en la vista se ha hecho uso de dos objetos *recyclerView*. La clase *RecyclerView* permite mostrar listas dinámicas. En el presente caso los objetos coexisten uno anidado en el otro.

En el detalle del fichaje se muestra su lugar de trabajo (modo oficina, modo teletrabajo…), el motivo de salida y el comentario que indicó el usuario con respecto a su salida.

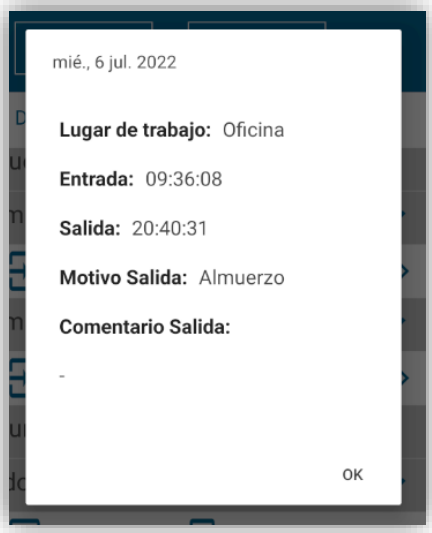

Figura 28. Diálogo Detalle de fichaje

El usuario puede consultar fichajes en un rango de fechas concreto. Según la figura 29, al hacer click en los botones de calendario, representados por rectángulos que muestran fechas en formato 'dd/mm/yyyy', se abre un diálogo para elegir la fecha deseada (desde(1) /hasta(2)). Para confirmar la consulta el usuario debe hacer click sobre el botón de búsqueda (3). De esta forma se actualizará la lista mostrada.

Si el usuario desea realizar una extracción de la consulta realizada puede hacerlo haciendo click sobre los botones de la esquina superior derecha. La extracción se realiza en forma de un fichero excel con extensión '.csv'. El botón de exportar (4) permite al usuario compartir con otras aplicaciones el fichero generado y el botón descargar (5) permite al usuario guardar el fichero generado en la carpeta de descargas del dispositivo.

Usando el botón atrás del dispositivo o haciendo click sobre el botón habilitado (6) para volver atrás, el usuario regresará a la pantalla Home.

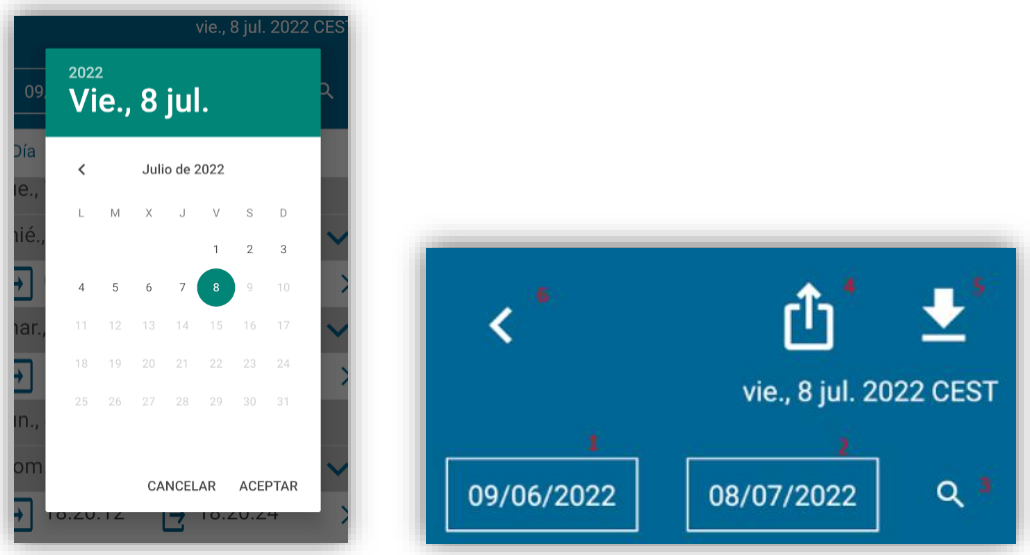

Figura 29. Botones Consultar Fichajes

5.4.7 Pantalla Fichar (FicharActivity)

Esta pantalla permite al usuario fichar. Se muestra el día y hora actual, los botones para fichar entrada o salida, su lugar de trabajo (modo oficina, modo teletrabajo…) y un espacio reservado para visualizar los fichajes del día

Existen varios casos de uso:

1. Si el lugar de trabajo del usuario en el momento de fichar es modo oficina:
    a. Si el usuario está dentro de un rango de distancia determinado a nivel de organización: El usuario puede fichar entrada.
    b. Si el usuario esta fuera de dicho rango de distancia determinado a nivel de organización: El usuario no puede fichar entrada.
2. Si el lugar de trabajo del usuario en el momento de fichar es uno distinto a oficina (i.e. teletrabajo): El usuario puede fichar entrada.
3. Si es el primer registro del día: El usuario no puede fichar salida hasta que se registre un fichaje de entrada. Para ello se inhabilita el botón fichar salida (su fondo coloreado en negro)
4. Si el usuario intenta registrar dos veces el mismo tipo de fichaje (entrada o salida) aparece un diálogo informando al usuario de que se sobrescribirá el registro.

Figura 30. Pantalla Fichar

Al registrar un fichaje de salida se abrirá un diálogo que permitirá al usuario informar el motivo de la salida y un comentario opcional. Se requerirá un comentario en caso de que el motivo de la salida sea una incidencia.

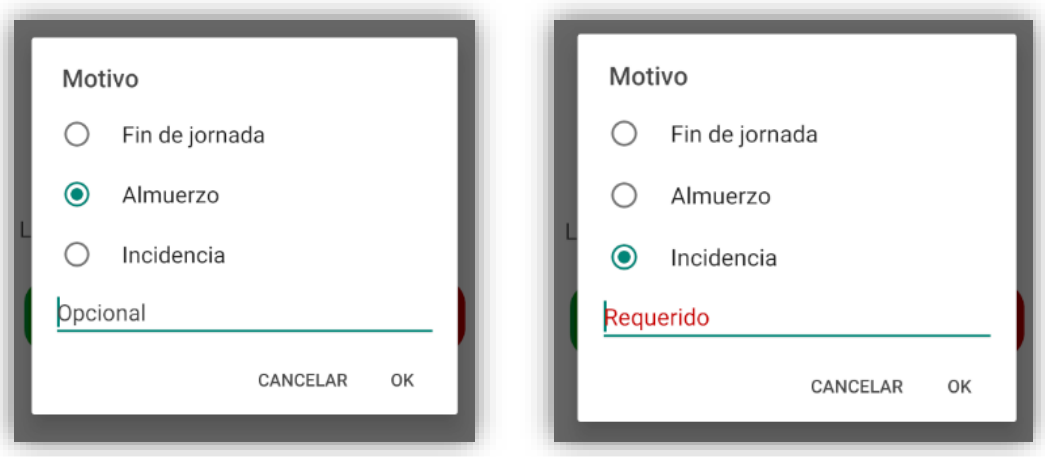

Figura 31. Diálogo Fichaje Salida

6. Pruebas unitarias

Las pruebas unitarias consisten en comprobar que determinadas partes del código funcionan correctamente. Validan el comportamiento de la lógica y ayudan a detectar errores que podrían pasar desapercibidos en otros casos.

PLAN DE PRUEBAS UNITARIO					
id	**Actividad**	**prerrequisitos**	**pasos**	**resultado esperado**	**Estado prueba**
1	MainActivity	- Acceso a internet habilitado - El usuario no está autenticado en la aplicación	Abrir la aplicación	Se abre una pantalla con una animación de barra de progreso circular y automáticamente se navega a la pantalla de Login	OK
2		- Acceso a internet habilitado - El usuario está autenticado en la aplicación	Abrir la aplicación	Se abre una pantalla con una animación de barra de progreso circular y automáticamente se navega a la pantalla de Home	OK
3	LoginActivity	- Acceso a internet habilitado - El usuario no está autenticado en la aplicación	Abrir la aplicación En la pantalla de Inicio de sesión hacer click sobre 'He olvidado mi contraseña'	Se abre un diálogo de aviso al usuario con respecto al reinicio de contraseña	OK
4		- Acceso a internet habilitado - El usuario no está autenticado en la aplicación	Abrir la aplicación -En la pantalla de Inicio de sesión hacer click sobre 'He olvidado mi contraseña' -Introducir dirección de correo errónea	Se informa al usuario que esa cuenta no existe.	OK

5		- Acceso a internet habilitado - El usuario no está autenticado en la aplicación	Abrir la aplicación -En la pantalla de Inicio de sesión hacer click sobre 'He olvidado mi contraseña' -Introducir dirección de correo correcta	Se cierra el diálogo y se informa al usuario de que se ha enviado el correo electrónico de reinicio de contraseña	OK
6		- Acceso a internet habilitado - El usuario no está autenticado en la aplicación	Abrir la aplicación -En la pantalla de Inicio de sesión hacer click sobre 'He olvidado mi contraseña' -Introducir dirección de correo correcta -Comprobar que llega el correo electrónico a la cuenta de correo del usuario -Iniciar sesión en la aplicación con las nuevas credenciales	El usuario inicia sesión correctamente y es navegado hacia la pantalla Home	OK
7	HomeActivity	- Acceso a internet habilitado	Desplazarse a la pantalla Home	Se muestra fecha y hora actualizada y nombre de usuario. Con el paso del tiempo la hora se actualiza correctamente	OK
8		- Acceso a internet habilitado	Desplazarse a la pantalla Home -Cerrar sesión	El usuario es navegado hacia la pantalla de Inicio de Sesión	OK
9		- Acceso a internet habilitado	Desplazarse a la pantalla Home -Hacer click sobre el logo de la aplicación	Se abre un diálogo informativo	OK
10		- Acceso a internet habilitado	Desplazarse a la pantalla Home -Hacer click sobre el	El usuario es navegado hacia	OK

			icono de Configuración	la pantalla de Configuración	
11		- Acceso a internet habilitado	Desplazarse a la pantalla Home -Hacer click sobre el icono de Perfil	El usuario es navegado hacia la pantalla de Perfil	OK
12		- Acceso a internet habilitado	Desplazarse a la pantalla Home -Hacer click sobre el botón Fichar	El usuario es navegado hacia la pantalla de Fichar	OK
13		- Acceso a internet habilitado	Desplazarse a la pantalla Home -Hacer click sobre el botón Consultar Fichajes	El usuario es navegado hacia la pantalla de Consultar Fichajes	OK
14	ConfigActivity	- Acceso a internet habilitado	Desplazarse a la pantalla Configuración -Hacer click sobre el botón Volver	El usuario es navegado hacia la pantlla Home	OK
15		- Acceso a internet habilitado	Desplazarse a la pantalla Configuración -Hacer click sobre el switch Cambiar Tema -Navegar hasta la pantalla Home, cerrar sesión y volver a entrar en la aplicación	Se actualiza el tema de la aplicación	OK
16		- Acceso a internet habilitado	Desplazarse a la pantalla Configuración -Hacer click sobre la opción Cambiar Contraseña	Se abre un dialogo que permite introducir datos	OK
17		- Acceso a internet habilitado	Desplazarse a la pantalla Configuración -Hacer click sobre la opción Cambiar Contraseña	Se abre un dialogo que permite introducir datos	OK

18		- Acceso a internet habilitado	Desplazarse a la pantalla Configuración -Hacer click sobre la opción Cambiar Contraseña -Dejarse campos vacios y darle al botón validar	Aparece texto en color rojo sobre los campos vacios	OK
19		- Acceso a internet habilitado	Desplazarse a la pantalla Configuración -Hacer click sobre la opción Cambiar Contraseña -Introducir contraseña nueva y de confirmación disitintas y hacer click sobre el botón validar	Aparece texto en color rojo indicando que las contraseñas no coinciden	OK
20		- Acceso a internet habilitado	Desplazarse a la pantalla Configuración -Hacer click sobre la opción Cambiar Contraseña -Introducir contraseña antigua -Introducir contraseña nueva y de confirmación iguales y hacer click sobre el botón validar	Aparece texto en color rojo indicando que las contraseñas no coinciden	OK
21		- Acceso a internet habilitado	Desplazarse a la pantalla Configuración -Hacer click sobre la opción Cambiar Contraseña -Introducir contraseña antigua correctamente -Introducir contraseña nueva y	Se actualiza la contraseña. Se cierra el diálogo indicando que la contraseña se ha actualizado con éxito	OK

			de confirmación iguales y hacer click sobre el botón validar		
22		- Acceso a internet habilitado	Desplazarse a la pantalla Configuración -Hacer click sobre la opción Cambiar Contraseña -Introducir contraseña antigua correctamente -Introducir contraseña nueva y de confirmación iguales y hacer click sobre el botón validar -Cerrar sesión y volver a iniciar sesión con la nueva contraseña	Se inicia sesión correctamente	OK
23	ProfileActivity	- Acceso a internet habilitado	Desplazarse a la pantalla Perfil -Hacer click sobre el botón Editar -Modificar los datos de correo eléctronico alternativo o número de teléfono y hacer click sobre el botón guardar - Salir a la pantalla Home con el botón volver o cerrar sesión y volver a entrar a la pantalla Perfil	Los datos se muestran actualizados	OK

24	FicharActivity	- Acceso a internet habilitado	Desplazarse a la pantalla Fichar	Se muestra fecha y hora actualizada y nombre de usuario. Con el paso del tiempo la hora se actualiza correctamente	OK
25		- Acceso a internet habilitado - No se ha fichado en el día actual	Desplazarse a la pantalla Fichar	El botón Salida está inhabilitado. No permite fichar salida	OK
26		- Acceso a internet habilitado - Usuario modo teletrabajo	Desplazarse a la pantalla Fichar -Fichar entrada y salida -Registrar comentario en la salida	Se actualiza la lista de fichajes	OK
27		- Acceso a internet habilitado - Localización habilitada - usuario modo oficina - usuario a más de x metros de distancia de su oficina	Desplazarse a la pantalla Fichar - Hacer click sobre el botón Entrada	Se avisa al usuario de que no puede fichar entrada porque no está dentro del rango de distancia adecuado para fichar	OK
28		- Acceso a internet habilitado - Localización habilitada - usuario modo oficina - usuario a menos de x metros de distancia de su oficina	Desplazarse a la pantalla Fichar - Hacer click sobre el botón Entrada	Se registra el fichaje de entrada y se actualiza el listado de fichajes	OK

29		- Acceso a internet no habilitado - Localización no habilitada - usuario modo oficina - usuario a más de x metros de distancia de su oficina	Desplazarse a la pantalla Fichar - Hacer click sobre el botón Entrada	Se avisa al usuario de que no puede fichar entrada porque la localización está inactiva	OK
30		- Acceso a internet habilitado - Localización no habilitada - usuario modo oficina - usuario a menos de x metros de distancia de su oficina	Desplazarse a la pantalla Fichar - Hacer click sobre el botón Entrada	Se avisa al usuario de que no puede fichar entrada porque la localización está inactiva	OK
31	ConsultaFichajesActivity	- Acceso a internet habilitado	Desplazarse a la pantalla Consulta Fichajes	Se muestra fecha y hora actualizada. Con el paso del tiempo la hora se actualiza correctamente Se muestra un listado de un resumen de los fichajes de los últimos 30 días.	OK
32		- Acceso a internet habilitado	Desplazarse a la pantalla Consulta Fichajes - Cambiar fechas de calendarios - Hacer click sobre el botón buscar	Se muestra un listado actualizado con los fichajes	OK

33		- Acceso a internet habilitado	Desplazarse a la pantalla Consulta Fichajes - Seleccionar un rango de fechas y no pulsar botón buscar - Hacer click sobre el botón exportar	Se informa al usuario que debe pulsar el botón buscar antes de proceder	OK
34		- Acceso a internet habilitado	Desplazarse a la pantalla Consulta Fichajes - Seleccionar un rango de fechas y no pulsar botón buscar - Hacer click sobre el botón descargar	Se informa al usuario que debe pulsar el botón buscar antes de proceder	OK
35		- Acceso a internet habilitado	Desplazarse a la pantalla Consulta Fichajes - Seleccionar un rango de fechas y pulsar botón buscar - Hacer click sobre el botón exportar	Se abre una lista de aplicaciones con las que poder compartir el fichero.	OK
36		- Acceso a internet habilitado	Desplazarse a la pantalla Consulta Fichajes - Seleccionar un rango de fechas y pulsar botón buscar - Hacer click sobre el botón descargar	Se abre un administrador de archivos que permite guardar el fichero a exportar	OK

Tabla 2. Plan de pruebas unitario

7. Conclusión y futuras implementaciones

7.1 Conclusión

El propósito de este proyecto no es un despliegue comercial, sino la evaluación de una prueba de concepto. Se ha planteado este proyecto con el objetivo de obtener un producto mínimo viable. Un proceso iterativo de generación de ideas, análisis y aprendizaje cuyo resultado ha sido el cumplimiento de los objetivos propuestos al principio de este proyecto: Una aplicación donde los usuarios puedan registrar las horas trabajadas, consultarlas y exportarlas con una interfaz sencilla y fácil de usar.

Para la realización del proyecto, me he bastado de mis conocimientos de programación en Java y bases de datos adquiridos a lo largo del grado junto con los recursos e información ofrecidos por Google en su sitio web Google Developers para el desarrollo de aplicaciones con las herramientas Android y Firebase.

El mayor reto de este trabajo ha sido el diseño e implementación de una aproximación de un modelo de datos que facilite minimizar el número de operaciones y escrituras en la base de datos y el desarrollo de código eficiente que minimice el consumo de batería y datos del dispositivo sobre el cual se ejecuta la aplicación.

7.2 Futuras implementaciones

Como se ha mencionado anteriormente, el propósito de este proyecto no es un despliegue comercial que requeriría de características adicionales a nivel de funcionalidad complementaria, capacidad de integración con sistemas ERP (*Enterprise Resource Planning*), HR (*Human Resources)*, BI (*Business Inteligence)* … Los cuales son programas que integran y gestionan procesos empresariales, así como de un planteamiento de desarrollo industrial con entornos de desarrollos, pruebas y producción. A continuación, enumero algunas de las características funcionales complementarias que se proponen como futuras implementaciones.

- Adjuntar justificantes médicos
- Fichar automáticamente en función del tiempo y localización GPS sin que el usuario tenga la aplicación en primer plano.
- Sistema de solicitud de disfrute de días de vacaciones
- Tener en cuenta diferentes zonas horarias
- Fichar mediante tecnología NFC, bluetooth, Wi-Fi…
- Desarrollar una aplicación web que permita a los usuarios administradores de una empresa realizar un proceso de carga de usuarios para importar los empleados de la empresa a la aplicación móvil.

- Desarrollar en la plataforma iOS.

Todo esto con el fin de poder ofrecer una aplicación con una estructura multiempresa, que se pueda desarrollar y mejorar de forma conjunta sin que afecte negativamente a las empresas que utilizan la aplicación.

8. Bibliografía

[1] W. L. BUNDY., «TIME RECORDER.,» [En línea]. Available: https://patents.google.com/patent/US393205A/en. [Último acceso: 07 07 2022].

[2] «Real Decreto-ley 8/2019,» 12 Marzo 2019. [En línea]. Available: https://www.boe.es/eli/es/rdl/2019/03/08/8. [Último acceso: 07 Julio 2022].

[3] «Android Developers,» [En línea]. Available: https://developer.android.com/. [Último acceso: 07 07 2022].

[4] «Requerimientos SMART,» [En línea]. Available: https://theecommmanager.com/smart-requirements/. [Último acceso: 07 07 2022].

[5] «Método MoSCoW,» [En línea]. Available: https://www.itdo.com/blog/moscow-que-es-y-como-priorizar-en-el-desarrollo-de-tu-aplicacion/. [Último acceso: 07 07 2022].