



Universidad
del País Vasco

eman ta zabal zazu

Euskal Herriko
Unibertsitatea

BILBOKO
INGENIARITZA
ESKOLA
ESCUELA
DE INGENIERÍA
DE BILBAO

TicTacker

Memoria del Proyecto

Desarrollo Avanzado de Software

4º Curso

<https://github.com/ffernandezco/DAS-Proyecto/>

Grado en Ingeniería Informática de Gestión y Sistemas de
Información

Francisco Fernández Condado

2025-03-16

Índice

1	Introducción	2
1.1	Instalación y configuración	2
2	Elementos utilizados y funcionalidades	2
3	Clases, estructura y bases de datos de la aplicación	4
3.1	Clases	4
3.1.1	Activities	4
3.1.2	Fragments	4
3.1.3	Adaptadores	4
3.1.4	Clases generales	5
3.1.5	Helpers	5
3.1.6	Eventos	5
3.1.7	Workers	5
3.2	Diagrama de clases	6
3.3	Base de datos	6
4	Manual de usuario de TicTacker	7
4.1	Inicio de sesión y registro de usuarios	7
4.2	Ajustes de la aplicación	8
4.3	Fichajes e historial	9
5	Dificultades encontradas al realizar el proyecto	11
6	Fuentes y referencias utilizadas	11

1. Introducción

TicTacker es una aplicación móvil para Android basada en Java + Gradle que permite ayudar a empresas y a autónomos a seguir un registro de su jornada laboral. Para ello, permite realizar un seguimiento completo de las horas y minutos trabajados cada día, permitiendo además configurar una jornada laboral con sus correspondientes horas y días laborales a fin de conocer las horas extraordinarias realizadas cada día y recibir notificaciones en caso de sobrepasar la jornada descrita. Es además posible personalizar su logotipo y el idioma para adaptarse a las necesidades de la empresa, así como generar ficheros CSV con los fichajes registrados que posteriormente puede importarse si así se desea.

1.1. Instalación y configuración

Para hacer uso de *TicTacker*, puede utilizarse el fichero APK para realizar la instalación. Una vez instalada, será necesario autenticarse, para lo que es posible crear una nueva cuenta usando el asistente o bien utilizar el usuario de demostración haciendo uso de las siguientes credenciales:

- **Usuario:** demo
- **Contraseña:** demo

A partir de entonces, podrán comenzar a utilizarse las funcionalidades de la aplicación.

2. Elementos utilizados y funcionalidades

Para construir la aplicación, se han hecho uso de las siguientes características y elementos propios de Android:

- **Uso de RecyclerView + CardView para mostrar listados de elementos con diferentes características:** el historial de fichajes utiliza *RecyclerView* para mostrar listas de fichajes en un formato vertical. Cada elemento de la lista se muestra en una vista personalizada (*CardView*), mostrando información detallada de cada fichaje.
- **Uso de una base de datos local, para listar, añadir y modificar elementos y características de cada elemento:** tal y como se detallará más adelante, el proyecto hace uso de una base de datos SQLite para almacenar los datos de los fichajes, los usuarios y las configuraciones.
- **Uso de diálogos:** la aplicación cuenta con múltiples diálogos, incluyendo los detalles de fichajes, que permiten ver todos los datos o editarlos si es preciso, mensajes de confirmación a la hora de borrar información, o un mensaje que se muestra si aún no ha finalizado la jornada laboral pero se trata de marcar una salida.

- **Uso de notificaciones locales:** la aplicación utiliza notificaciones para alertar a los usuarios cuando han completado su tiempo de trabajo diario o cuando se llega a las horas extra.
- **Control de la pila de actividades:** se controla que la aplicación finalice las actividades que no se están usando para una gestión correcta de la memoria. Además, si se modifica algún parámetro de configuración como el idioma, se recarga la aplicación.
- **Uso de *Fragments*:** la aplicación cuenta con 3 *Fragments* diferentes: uno para el fichaje, otro para el historial y otro para la configuración, pudiéndose acceder a ellos directamente desde el menú y adaptándose a la orientación del dispositivo y el tamaño de la pantalla.
- **Aplicación multiidioma y añadir la opción de cambiar idioma en la propia aplicación:** la aplicación está disponible en inglés y castellano, permitiendo el cambio de idioma desde el apartado de Ajustes. Por defecto, usa el idioma del dispositivo.
- **Uso de ficheros de texto:** se permite exportar e importar los fichajes de un usuario en formato CSV, además de cambiar el logotipo que se muestra en el menú por medio de los ajustes, manipulando tanto ficheros de texto como imágenes.
- **Uso de Preferencias, para guardar las preferencias del usuario en cuanto a mostrar/esconder cierta información:** la aplicación permite controlar el idioma, el logotipo y definir una jornada laboral, marcando tanto los días laborables como las horas que se trabajan cada semana. En función de este dato, se cargará el tiempo restante de la jornada y se enviarán notificaciones.
- **Crear estilos y temas propios, para personalizar fondos, botones, etc.:** se ha hecho uso de un tema propio basado en Material Design de Google, en color verde a juego con el logotipo de la app.
- **Uso de intents implícitos para abrir otras aplicaciones, contactos, etc.:** al registrar un fichaje, se almacenan las coordenadas desde las que se ha realizado, permitiendo posteriormente en el detalle abrir la ubicación en una aplicación que lo permita y que esté instalada (por ejemplo, Google Maps).
- **Pantalla de login (y registro), para guardar credenciales de usuario en la base de datos local:** se ha incluido, además de una pantalla para iniciar sesión, una sección que permite dar de alta un nuevo usuario con su nombre y contraseña para permitir el acceso, pudiendo almacenar fichajes de varios empleados en un mismo dispositivo.
- **Añadir una barra de herramientas (ToolBar) personalizada en la aplicación así como un panel de navegación (Navigation Drawer):** se ha incluido una barra de herramientas adaptada a los dispositivos modernos, así como un panel que permite moverse entre los 3 fragmentos creados para la app fácilmente.

3. Clases, estructura y bases de datos de la aplicación

TicTacker hace uso de diferentes clases que se comunican entre sí y que utilizan los elementos XML de Android para gestionar la vista correspondiente. Adicionalmente, se hace uso de una base de datos SQLite para gestionar los usuarios, los fichajes y las configuraciones personalizadas.

3.1. Clases

La aplicación cuenta con diferentes clases Java diferenciadas que dan respuesta a una o varias de las funcionalidades de la aplicación. A continuación se describen las siguientes clases utilizadas, pudiendo ver las relaciones entre las mismas por medio de diagrama de clases de la Figura 1.

3.1.1. Activities

- **MainActivity**: actividad principal que se encarga de gestionar la navegación y la interfaz de usuario principal de la aplicación.
- **LoginActivity**: gestiona la autenticación de usuarios existentes, validando credenciales y guardando la sesión.
- **SignupActivity**: gestiona el registro de nuevos usuarios, incluyendo validación de los datos, que no se repitan usuarios y creación de cuentas.
- **SettingsActivity**: ofrece opciones de personalización de la aplicación, como cambiar el idioma o la jornada laboral.

3.1.2. Fragments

- **ClockInFragment**: fragment que permite a los usuarios registrar sus fichajes, mostrando el estado actual y el tiempo trabajado.
- **HistoryFragment**: fragment que muestra el historial de fichajes del usuario, permitiendo exportar e importar datos por medio de ficheros CSV.
- **SettingsFragment**: fragment que permite a los usuarios ajustar sus preferencias, como las horas de trabajo semanales y los días laborables.

3.1.3. Adaptadores

- **FichajeAdapter**: utilizado para mostrar listas de fichajes a través de un *RecyclerView*.

3.1.4. Clases generales

- **TicTacker:** clase principal de la aplicación que se encarga de aplicar las preferencias de usuario al iniciar.
- **Fichaje:** entidad que representa un fichaje con los atributos correspondiente, usado para instanciar objetos fácilmente con su fecha, hora de entrada, hora de salida, latitud, longitud y nombre de usuario para almacenar en la base de datos.
- **WorkTimeCalculator:** clase que proporciona métodos para calcular el tiempo trabajado y el tiempo restante.

3.1.5. Helpers

- **DatabaseHelper:** gestiona la base de datos SQLite, incluyendo la creación de tablas y las operaciones.
- **NotificationHelper:** clase que gestiona la creación y envío de notificaciones.

3.1.6. Eventos

- **FichajeEvents:** clase que gestiona los eventos relacionados con los fichajes, permitiendo notificar cambios a los listeners registrados.

3.1.7. Workers

- **WorkTimeCheckWorker:** worker que comprueba periódicamente el tiempo trabajado y envía notificaciones al usuario si alcanza su jornada laboral.

3.2. Diagrama de clases

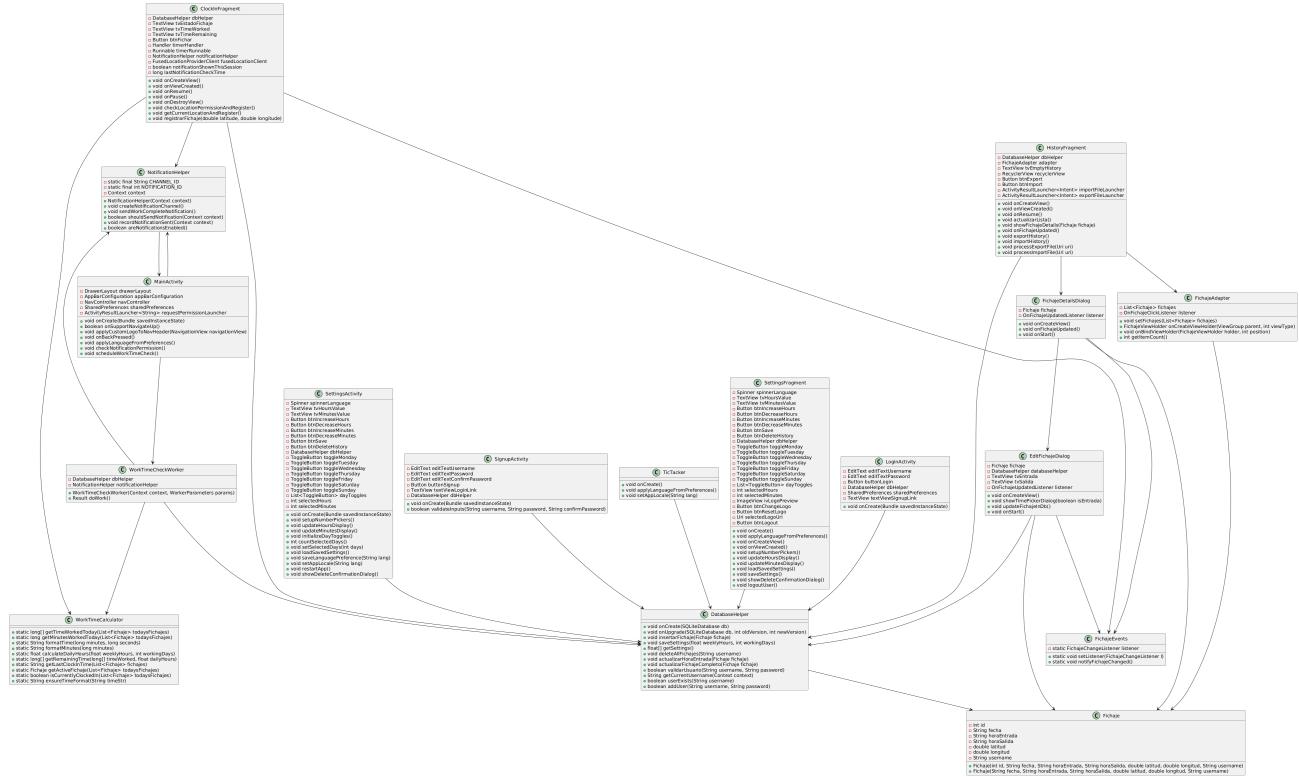


Figura 1: Diagrama de clases del proyecto

3.3. Base de datos

Además de las *SharedPreferences*, para gestionar la persistencia de la aplicación se hace uso de una base de datos de tipo SQLite. La base de datos está definida por la clase *DatabaseHelper*, que extiende de *SQLiteOpenHelper*, y contiene las siguientes tablas:

- **Tabla Fichajes:** almacena la información sobre los fichajes de los usuarios.
 - **Atributos principales:** `id`, `fecha`, `hora_entrada`, `hora_salida`, `latitud`, `longitud`, `username`.
 - **Índices:** `id` es un campo único y clave primaria.
- **Tabla Configuraciones:** almacena las configuraciones de horas de trabajo y días laborables.
 - **Atributos principales:** `id`, `weekly_hours`, `working_days`.

- **Índices:** id es un campo único y clave primaria.
- **Tabla Usuarios:** almacena los usuarios registrados en la aplicación.
 - **Atributos principales:** id, username, password.
 - **Índices:** username es un campo único indexado para prevenir duplicados.

4. Manual de usuario de TicTacker

Una vez se ha realizado la instalación de la aplicación por medio del fichero APK, será posible comenzar a utilizarla. Al lanzarla por primera vez, en función de la versión del sistema operativo Android, se lanzarán alertas para conceder permisos de cara al envío de notificaciones, la ubicación y la gestión de archivos de imagen. Es recomendable habilitar todos ellos mientras la app esté en uso para evitar problemas de funcionamiento y aprovechar todas las características.

4.1. Inicio de sesión y registro de usuarios

Tras aceptar los permisos, se mostrará la pantalla de inicio de sesión representada en la Figura 2a. Es posible hacer uso de la cuenta de demostración, usando “demo” como usuario y contraseña, si bien también puede pulsarse sobre el enlace de registro para dar de alta un nuevo usuario, como representa la Figura 2b. En caso de elegir esta opción, deberá indicarse un nombre de usuario que no puede existir ya en la base de datos de la aplicación, así como una contraseña con 4 o más caracteres que, por motivos de seguridad, deberá introducirse dos veces.

Tras haber creado una nueva cuenta de usuario, se redirigirá a la pantalla de inicio de sesión, donde será posible iniciar sesión usando el usuario y la contraseña registradas. Basta con completar los campos y pulsar sobre el botón “Iniciar sesión” para acceder a la aplicación.

La pantalla de bienvenida de la Figura 2c ilustra una situación de fichaje en curso. La navegación por medio de los distintos apartados de la aplicación se realiza presionando sobre el botón de las tres líneas que se muestra en la esquina superior izquierda. La aplicación presenta tres áreas separadas sobre las que se darán más detalles a continuación:

- **Fichar:** permite consultar el estado de un fichaje en curso o fichar a medida.
- **Historial:** ofrece información sobre los fichajes almacenados, además de permitir exportar e importar datos.
- **Ajustes:** permite definir aspectos de configuración a nivel de la aplicación. Para obtener la mejor experiencia de usuario, se recomienda comenzar accediendo a esta sección.

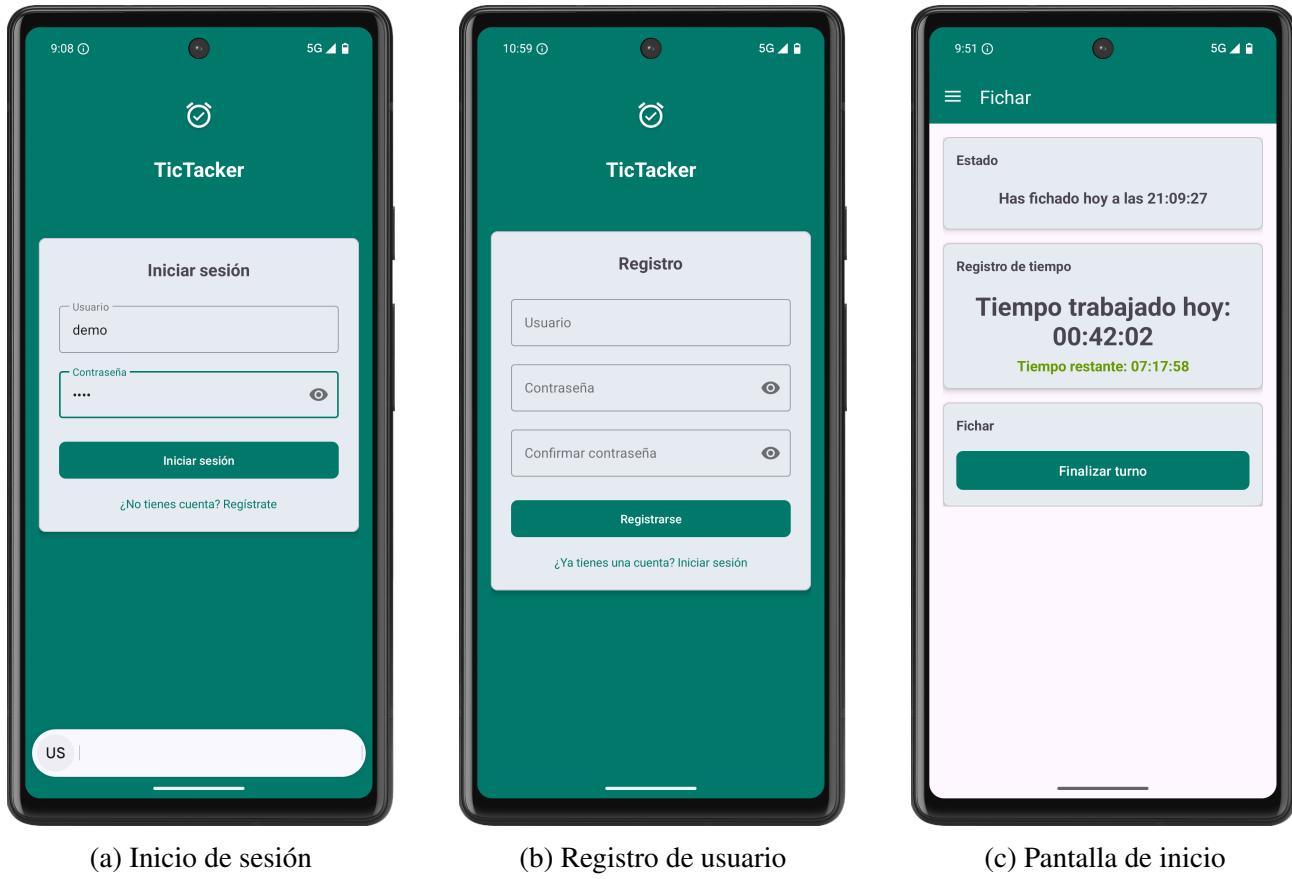


Figura 2: Pantalla de bienvenida e inicio de sesión

4.2. Ajustes de la aplicación

Pulsando sobre la sección de “Ajustes” en el menú de la aplicación, se accederá a la configuración. Es importante presionar sobre el botón de “Guardar” para que se refresque la vista de la aplicación y se puedan ver los cambios realizados de forma correcta si se modifica alguno de los capos.

Para obtener la mejor experiencia posible, se permite elegir, tal y como se muestra en la Figura 3a, entre Español e Inglés como idiomas, siendo suficiente con elegir en el desplegable el preferido para configurarlo como el idioma por defecto para toda la aplicación. A continuación, el usuario deberá completar los datos de su jornada laboral, especificando los días laborables y las horas que se trabaja a la semana usando los controles. Esta información permite a la aplicación calcular las horas que se trabajará cada día, así como enviar notificaciones en caso de superar la previsión diaria y mostrar un cuadro de diálogo si se trata de salir antes de lo previsto.

De forma opcional, es posible incluir una imagen como logotipo, que se mostrará en el menú de la aplicación tal y como ilustra la Figura 3c. Para ello, basta con pulsar sobre el botón de “Cambiar” y elegir el nuevo logotipo en el explorador, como muestra la Figura 3b. Debe ser un fichero de imagen,

y se recomienda que sea un PNG con fondo transparente y un color claro para mejor visibilidad. En caso de querer restaurar el logotipo por defecto, puede usarse el botón de “Quitar”.

De forma adicional, esta pantalla también permite cerrar sesión para cambiar de usuario, lo que nos llevaría directamente a la pantalla de inicio de sesión, así como eliminar los datos de fichaje almacenados, lo que no se recomienda salvo que se hayan exportado previamente.

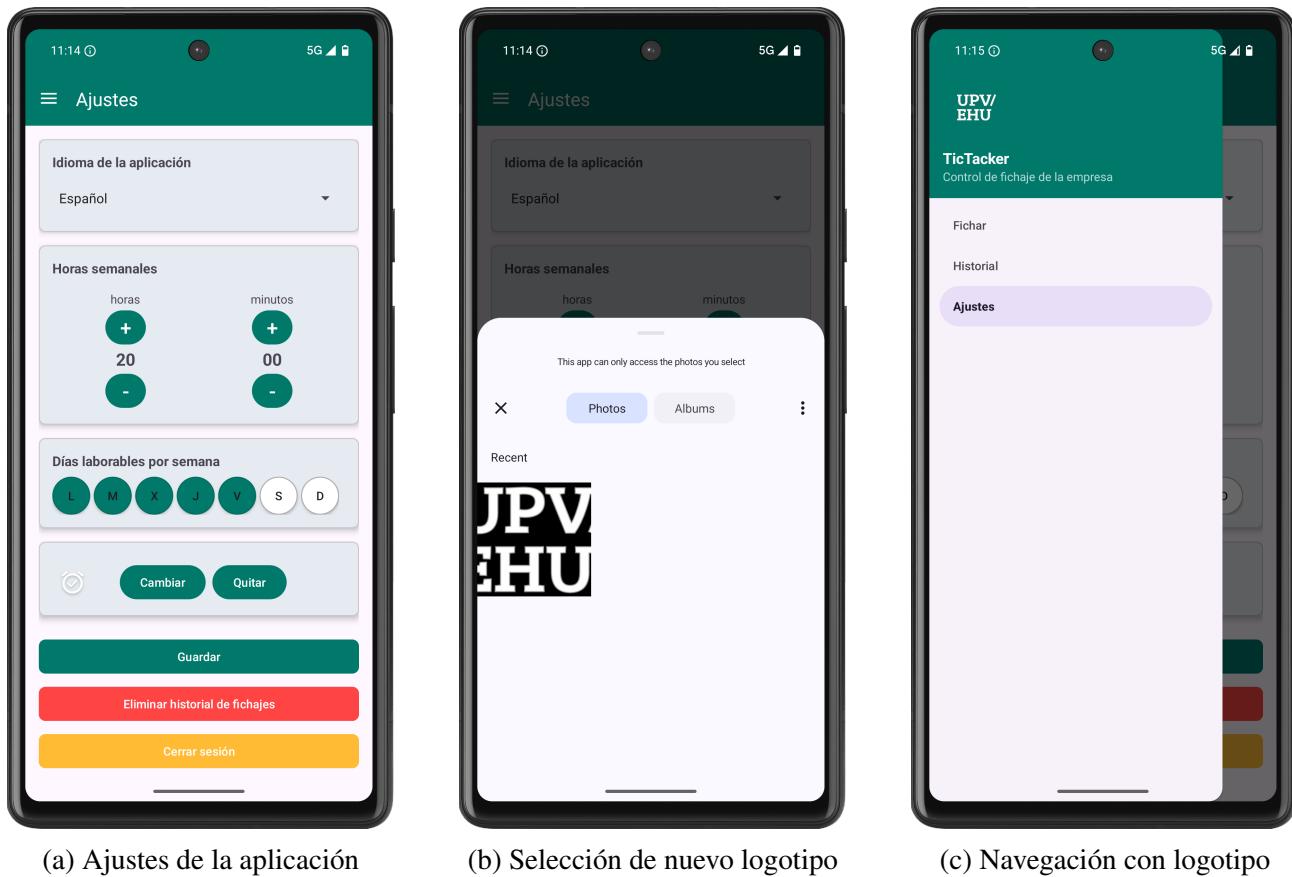


Figura 3: Ajustes de la aplicación

4.3. Fichajes e historial

Accediendo a la pantalla para fichar, es posible consultar de un vistazo los datos de un fichaje activo, tal y como muestra la Figura 4a. Es posible marcar una entrada y una salida por medio del botón asociado de forma sencilla, y consultar la hora, minuto y segundo exacto en el que se fichó, así como la duración actual, el tiempo restante para alcanzar la jornada o, en su defecto, las horas extraordinarias realizadas. Al llegar a las horas estimadas, se recibe una notificación si la aplicación está activa.

Una vez guardado el fichaje, desde el menú es posible acceder al historial que muestra la figura

4b. En él, puede verse un registro de todos los fichajes almacenados en la aplicación, mostrándose primero el más reciente. Si se pulsa sobre uno de ellos, como se muestra en la Figura 4c, es posible ver el detalle completo, consultando la fecha y hora exactas de salidas y entradas, así como las coordenadas geográficas desde las que se produjo el fichaje en caso de haber otorgado permisos para acceder a la ubicación (en su defecto, se mostrará que no está disponible).

En la parte inferior de la ventana de detalles, existen opciones para editar un fichaje guardado, que permite modificar la hora de entrada o salida en caso de haberse identificado algún error, así como la opción de ver en un mapa la ubicación registrada. Al pulsar sobre esta opción, se abre la aplicación de mapas asociada del dispositivo (Google Maps en gran parte de las ocasiones).

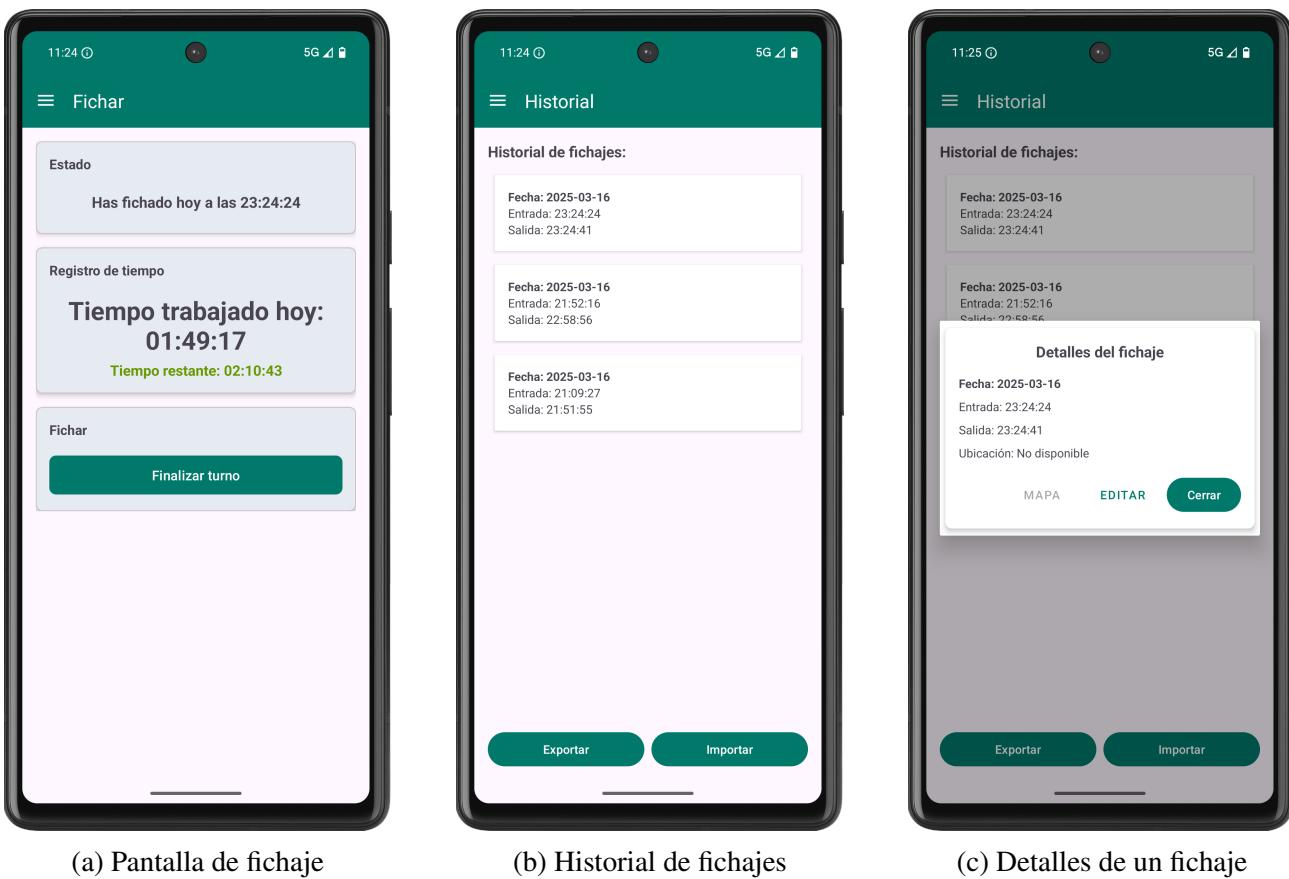


Figura 4: Uso general de la aplicación

De forma adicional, la parte inferior de la aplicación permite importar o exportar una hoja de fichajes. Al presionar sobre exportar, se genera un fichero CSV con los fichados registrados, para el que deberá elegirse una ubicación dentro del dispositivo en la que guardarla, mientras que si se selecciona importar, se puede elegir un fichero generado previamente por la aplicación o de forma manual con el mismo formato. Al hacerlo, se almacenarán los detalles en la base de datos y se podrá

ver en el historial junto al resto de datos.

5. Dificultades encontradas al realizar el proyecto

A lo largo del proyecto, han surgido múltiples dificultades. En primer lugar, el rendimiento de Android Studio y del emulador no es el esperado, lo que ha dificultado algo la tarea. Además, tuve problemas a la hora de gestionar los permisos y cómo se realizaba el conteo del tiempo, pues había que pulsar varias veces para que se registrase una entrada o una salida, y las notificaciones no siempre se enviaban o, peor aún, se enviaban cada segundo obligando a detener la aplicación, algo que fue corregido.

También ha resultado algo complicado implementar la lógica del inicio de sesión, principalmente porque la aplicación se creó siendo pensada para un solo usuario y, por tanto, ha sido necesario adaptar cada método para que funcione correctamente con múltiples usuarios, siendo necesario acceder y añadir un parámetro más en cada consulta.

6. Fuentes y referencias utilizadas

A la hora de realizar el proyecto, ha sido preciso realizar consultas a diferentes sitios de Internet, bien para resolver dudas o para obtener información y guías sobre cómo realizar diferentes acciones. A continuación se exponen las fuentes y referencias más relevantes que se han utilizado:

- Apuntes de la asignatura “Desarrollo Avanzado de Software”. Iker Sobrón Polancos. eGela UP-V/EHU.
- Android Developers Guide.
- *Android Studio mensaje: Missing Constraints in ConstraintLayout.* Stack Overflow.
- Launcher icon generator.
- Material design color palette.