

Gestor de tareas

Juan Jerez Garcia

**Índice**

1.- Capítulo 1: Introducción

1.1.- …………

Capítulo **1**: **Introducción**

## Introducción del proyecto

En la actualidad, la gestión eficiente del tiempo y las tareas es esencial tanto en el ámbito personal como profesional. Las herramientas digitales han surgido como soluciones efectivas para ayudar a las personas a organizar sus actividades diarias, mejorar su productividad y asegurarse de cumplir con sus compromisos. El propósito de este proyecto es desarrollar una aplicación de gestor de tareas para dispositivos Android utilizando Kotlin y Jetpack Compose, con soporte de backend mediante Firebase. Esta aplicación permitirá a los usuarios registrar y gestionar sus tareas de manera intuitiva y eficiente.

La necesidad de este proyecto surge del creciente número de usuarios que requieren herramientas tecnológicas para organizar sus vidas en un entorno cada vez más digitalizado. Con características como la creación, edición y eliminación de tareas, notificaciones en segundo plano y visualización de tareas en un calendario, esta aplicación está diseñada para mejorar la productividad del usuario. Las ventajas de utilizar esta aplicación incluyen una mayor organización personal, reducción del estrés asociado con la gestión de tareas múltiples y la capacidad de planificar y seguir el progreso de las actividades de manera más efectiva.

## Propósito

El propósito del proyecto es desarrollar una aplicación móvil que ofrezca las siguientes características:

* Registro y autenticación de usuarios.
* Creación, edición y eliminación de tareas.
* Visualización de tareas en un calendario.
* Notificaciones en segundo plano para recordar tareas pendientes.
* Interfaz de usuario intuitiva y fácil de usar.

Una vez desarrollado, el proyecto permitirá a los usuarios gestionar sus tareas diarias de manera eficiente, mejorando así su productividad y organización personal.

## Objetivos del Proyecto

El proyecto tiene como objetivos principales: La implementación de un sistema que gestione las tareas de los usuarios.

En lo que respecta al sistema que gestione las tareas de los usuarios, el objetivo es desarrollar un sistema intuituvo para que los usuarios puedan gestionar sus tareas (crearlas, modificarlas, eliminarlas y visualizarlas). además para facilitar su visualización y control de urgencias se desarrolla un sistema de calendarios y de notificaciones para mantener al usuario siempre alerta de sus tareas.

## Coste del proyecto

En este apartado, se detallan los costes asociados al desarrollo e implantación del proyecto.

### 1.4.1.-Costes de Desarrollo

El costo del desarrollo está asociado a la creación y desarrollo de la aplicación. Este proceso incluye el tiempo y los recursos invertidos en la aplicación, el diseño y las pruebas. Este coste incluye los siguientes elementos claves:

* **salario de los desarrolladores**: Esto sería lo que se paga a los desarrolladores por su trabajo durante la creación de la aplicación. En este caso la retribución que tendría un programador que estuviera trabajando en la aplicación durante 3 meses a media jornada sería de 6.000€ (2.000€ al mes).
* **Herramientas y software utilizandos**: En este caso, el IDE utilizado es de gratuito (Android studio) y todas las librerías y framework utilizadas son gratuitas por lo que la suma del costo de estos es de 0€.
* **pruebas y depuración**: Los costos asociados a las pruebas en el software junto con la identificación y corrección de errores. En este caso solo supondría un aumento en el tiempo de trabajo de un empleado de la empresa o por un otro empleado externo especializado en la depuración. Teniendo en cuenta que en este proyecto solo contamos con un trabajador y que esta actividad estaría incluida en su salario anteriormente mencionado no supondría ningún coste adicional. Las herramientas utilizadas son gratuitas por lo que no representa ningún coste adicional.
* **Diseño UI**: Si es necesario, también tendríamos que incluir los costos del diseño visual de la aplicación. Como en el apartado anterior, al ser solo un trabajador y no realizarse ninguna contratación externa el coste del diseño estaría incluido en el salario del mismo trabajador.

### 1.4.2 Costes de Implantación

Los costos de la implantación son los gastos asociados con poner en marcha nuestro software desarrollado junto con su mantenimiento. Esto implica la instalación, configuración y despliegue de nuestra aplicación para que esté lista para ser utilizada por el usuario final.

En nuestro caso la aplicación no estará alojada en ningún servidor y utilizara como base de datos Firebase que es de Google y su uso es totalmente gratuito.

**Costos totales:**

* Teniendo en cuenta todo lo dicho anteriormente el coste de nuestra aplicación será el coste de nuestro único trabajador los 3 meses que se ha dedicado a está, un total de 6.000€.

Capítulo **2: Análisis del sistema**

## Introducción

El análisis del sistema es una fase crucial en el desarrollo de cualquier proyecto de software, ya que permite definir claramente los requisitos y diseñar una solución que cumpla con las expectativas del usuario final. En este capítulo, se abordarán los aspectos fundamentales del análisis del sistema para la aplicación de gestión de tareas.

Esta aplicación está diseñada para facilitar la organización personal y profesional de los usuarios, permitiéndoles registrar, modificar y eliminar tareas, así como su gestión en un calendario integrado. Para asegurar que la aplicación cumpla con sus objetivos, se llevará a cabo un análisis detallado que incluirá la identificación de los requisitos funcionales y no funcionales, la creación de diagramas de casos de uso y narrativas detalladas de estos casos.

El análisis se centrará en los siguientes componentes clave:

* **Análisis de Requisitos**: Se identificarán y documentarán los requisitos funcionales y no funcionales del sistema. Los requisitos funcionales describen las funciones específicas que el sistema debe realizar, mientras que los requisitos no funcionales especifican las propiedades y restricciones del sistema, como el rendimiento y la seguridad.
* **Diagramas de Casos de Uso**: Los diagramas de casos de uso se utilizarán para representar gráficamente las interacciones entre los actores (usuarios) y el sistema. Estos diagramas ayudarán a visualizar las funcionalidades principales y cómo se relacionan entre sí.
* **Narrativas de Casos de Uso**: Se proporcionara narrativas detalladas de los casos de uso. Estas narrativas describirán paso a paso las interacciones entre los usuarios y el sistema, incluyendo los flujos de eventos principales y los caminos alternativos.

El objetivo de este análisis es proporcionar una comprensión clara y completa de las funcionalidades y requisitos del sistema, lo que permitirá un diseño e implementación efectivos.

## Análisis de requisitos

Este software tiene como propósito principal el proporcionar un sistema de gestión fácil e intuitivo para que un usuario pueda gestionar sus tareas de la forma más rápida posible.

Las propiedades que debe satisfacer son la de organizar las tareas dadas por un usuario y la de realizar notificaciones de las tareas para que el usuario pueda saber en todo momento cuáles tareas son las más urgentes y poner organizar mejor su tiempo.

1. **Requerimientos funcionales.**

Los requerimientos funcionales son las especificaciones que describen las funcionalidades que un programa debe cumplir. Estos definen las acciones que el programa debe realizar, las interacciones con los usuarios y los resultados esperados de dichas interacciones. A continuación, se presentarán los requerimientos funcionales de mi aplicación:

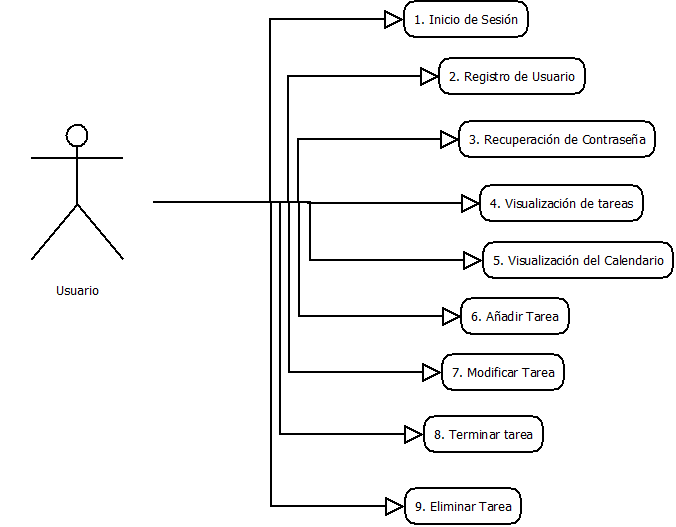
* **Login del usuario**
  + Autenticación de usuario: La aplicación debe verificar la identidad de nuestro usuario mediante su nombre de usuario (email) y contraseña proporcionados.
  + Recuperación de contraseña: La aplicación debe permitir a los usuarios registrados recuperar su contraseña mediante un enlace por correo electrónico.
  + Registro de Usuario: La aplicación debe permitir el registro de nuevos usuarios al sistema proporcionando estos: un email, un nombre, un apellido, una contraseña y una fecha de nacimiento.
* **Pantalla principal**
  + Visualización de tareas: La aplicación debe mostrar las tareas de nuestro usuario divididas por su nivel de urgencia e identificada por colores.
  + Filtrado: La aplicación debe permitir filtrar las tareas por su nivel de urgencia (muy urgente, poco urgente, etc).
  + Creación de tarea: Los usuarios deben poder registrar nuevas tareas, las cuales se crearán a partir de los datos proporcionados por ellos, estos datos son: título, descripción y fecha de finalización.
  + Acceso a los detalles de la tarea: Los usuarios deben poder acceder a estas tareas para modificarlas o eliminarlas.
* **Pantalla de visualización de tareas mediante calendario**
  + Visualización de tareas mediante calendario: La aplicación debe ofrecer una vista de calendario donde los usuarios puedan ver sus tareas organizadas por fechas.
  + Creación de tarea: Los usuarios deben poder registrar nuevas tareas con una fecha de finalización determinada por el calendario, las cuales se crearán a partir de los datos proporcionados por ellos, estos datos son: título, descripción y fecha de finalización.
  + Acceso a los detalles de la tarea: Los usuarios deben poder acceder a estas tareas para modificarlas o eliminarlas.

1. **Requerimientos no funcionales**

* **Seguridad**: El sistema debe garantizar la confidencialidad y protección de los datos de los usuarios, así como medidas de seguridad para evitar accesos no autorizados.
* **Rendimiento**: El tiempo de respuesta de la base de datos debe ser rápido y eficiente, para proporcionar una experiencia fluida al usuario.
* **Usabilidad**: La aplicación móvil debe ser intuitiva y fácil de usar, con una interfaz amigable que permita a los usuarios poder utilizar todas sus funciones sin la necesidad de mirar ningún manual.
* **Escalabilidad**: El sistema debe ser capaz de manejar un crecimiento futuro, sin comprometer el rendimiento ni la funcionalidad.

## Casos de uso: Diagramas y Narrativas de Casos de Uso

Los diagramas de casos de uso reflejarán las funcionalidades que correspondan a cada uno de los ámbitos concretos y más importantes del sistema. En la narrativa de los casos de uso se especifican las funciones, tareas e interacciones que se dan lugar en el sistema, es decir, se describen detalladamente cada uno de los casos de uso.



1. **Caso de uso 1: Inicio de Sesión**

**Nombre del caso de uso**: Inicio de Sesión  
**Actor Principal**: Usuario  
**Condiciones de entrada**: El usuario debe tener una cuenta registrada en el sistema.  
**Flujo de eventos**:

1. El usuario abre la aplicación y selecciona la opción de inicio de sesión.
2. El sistema muestra un formulario de inicio de sesión.
3. El usuario ingresa su correo electrónico y contraseña.
4. El usuario selecciona el botón "Iniciar Sesión".
5. El sistema verifica las credenciales del usuario.
6. Si las credenciales son correctas, el sistema guarda la sesión y redirige al usuario a la pantalla principal.
7. El caso de uso termina.

**Camino alternativo**:

6a. Si las credenciales son incorrectas, el sistema muestra un mensaje de error y permite al usuario intentar nuevamente.

**Condiciones de salida**: El usuario está autenticado y accede a la pantalla principal.

1. **Caso de uso 2: Registro de Usuario**

**Nombre del caso de uso**: Registro de Usuario  
**Actor Principal**: Usuario  
**Condiciones de entrada**: El usuario no debe estar registrado en el sistema.  
**Flujo de eventos**:

1. El usuario abre la aplicación y selecciona la opción de registro.
2. El sistema muestra un formulario de registro.
3. El usuario ingresa sus datos personales (nombre, apellidos, correo electrónico, contraseña y fecha de nacimiento).
4. El usuario selecciona el botón "Registrarse".
5. El sistema valida los datos ingresados.
6. El sistema crea una nueva cuenta para el usuario.
7. El sistema crea una colección de tareas para el usuario.
8. El sistema crea una tarea inicial para el usuario.
9. El sistema redirige al usuario a la pantalla de inicio de sesión.
10. El caso de uso termina.

**Camino alternativo**:

5a. Si los datos ingresados no son válidos, el sistema muestra un mensaje de error y permite al usuario corregir los datos.

**Condiciones de salida**: El usuario tiene una cuenta creada y está listo para iniciar sesión.

1. **Caso de uso 3: Recuperación de Contraseña**

**Nombre del caso de uso**: Recuperación de Contraseña  
**Actor Principal**: Usuario  
**Condiciones de entrada**: El usuario debe haber olvidado su contraseña.  
**Flujo de eventos**:

1. El usuario abre la aplicación y selecciona la opción de "Olvidé mi contraseña".
2. El sistema muestra un formulario de recuperación de contraseña.
3. El usuario ingresa su correo electrónico registrado.
4. El usuario selecciona el botón "Enviar".
5. El sistema envía un correo electrónico con instrucciones para restablecer la contraseña.
6. El usuario sigue las instrucciones del correo electrónico y restablece su contraseña.
7. El sistema restablece su contraseña.
8. El caso de uso termina.

**Camino alternativo**:

5a. Si el correo electrónico ingresado no está registrado, el sistema muestra un mensaje de error.

**Condiciones de salida**: El usuario recibe un correo electrónico con instrucciones para restablecer su contraseña.

1. **Caso de uso 4: Visualización de tareas**

**Nombre del caso de uso**: Visualización de tareas  
**Actor Principal**: Usuario  
**Condiciones de entrada**: El usuario debe estar autenticado en el sistema.  
**Flujo de eventos**:

1. El usuario selecciona la opción de tareas en la barra lateral.
2. El sistema muestra la vista de tareas divididas por sus nivel de urgencia.
3. El usuario puede navegar por la vista para ver sus diferentes tareas.
4. El caso de uso termina.

**Condiciones de salida**: El usuario visualiza las tareas.

1. **Caso de uso 5: Visualización del Calendario**

**Nombre del caso de uso**: Visualización del Calendario  
**Actor Principal**: Usuario  
**Condiciones de entrada**: El usuario debe estar autenticado en el sistema.  
**Flujo de eventos**:

1. El usuario selecciona la opción de calendario en la pantalla principal.
2. El sistema muestra la vista del calendario con las tareas programadas.
3. El usuario puede navegar por el calendario para ver sus tareas en diferentes fechas.
4. El caso de uso termina.

**Condiciones de salida**: El usuario visualiza el calendario con tareas.

1. **Caso de uso 6: Añadir Tarea**

**Nombre del caso de uso**: Añadir Tarea  
**Actor Principal**: Usuario  
**Condiciones de entrada**: El usuario debe estar autenticado en el sistema.  
**Flujo de eventos**:

1. El usuario selecciona la opción de "Añadir Tarea".
2. El sistema muestra un formulario para agregar una nueva tarea.
3. El usuario ingresa los detalles de la tarea (título, descripción, fecha de vencimiento).
4. El usuario selecciona el botón "Guardar".
5. El sistema guarda la nueva tarea y la muestra en la vista de tareas y calendario.
6. El caso de uso termina

**Camino alternativo**:

5a. Si el titulo de la tarea ya existe, el sistema mostrara un mensaje de error.

**Condiciones de salida**: La nueva tarea está registrada y visible en la aplicación.

1. **Caso de uso 7: Modificar Tarea**

**Nombre del caso de uso**: Modificar Tarea  
**Actor Principal**: Usuario  
**Condiciones de entrada**: El usuario debe estar autenticado en el sistema y tener al menos una tarea registrada.  
**Flujo de eventos**:

1. El usuario selecciona una tarea existente.
2. El sistema muestra los detalles de la tarea seleccionada.
3. El sistema permite al usuario editar los detalles de la tarea.
4. El usuario modifica los detalles y selecciona el botón "Guardar".
5. El sistema actualiza la tarea con los nuevos detalles y vuelve a la vista de tareas.
6. El caso de uso termina

**Camino alternativo**:

5a. Si el titulo de la tarea ya existe, el sistema mostrara un mensaje de error.

**Condiciones de salida**: La tarea está actualizada con la nueva información.

1. **Caso de uso 8: Terminar tarea**

**Nombre del caso de uso**: Terminar Tarea  
**Actor Principal**: Usuario  
**Condiciones de entrada**: El usuario debe estar autenticado en el sistema y tener al menos una tarea registrada.  
**Flujo de eventos**:

1. El usuario selecciona una tarea existente.
2. El sistema muestra los detalles de la tarea seleccionada.
3. El usuario selecciona la opción de "Terminar tarea".
4. El sistema da por finalizado la tarea y vuelve a la vista de tareas.
5. El caso de uso termina

**Condiciones de salida**: La tarea está actualizada con la nueva información.

1. **Caso de uso 9: Eliminar Tarea**

**Nombre del caso de uso**: Eliminar Tarea  
**Actor Principal**: Usuario  
**Condiciones de entrada**: El usuario debe estar autenticado en el sistema y tener al menos una tarea registrada.  
**Flujo de eventos**:

1. El usuario selecciona una tarea existente.
2. El sistema muestra los detalles de la tarea seleccionada.
3. El usuario selecciona la opción de "Eliminar".
4. El sistema solicita confirmación para eliminar la tarea.
5. El usuario confirma la eliminación.
6. El sistema elimina la tarea y la elimina de la vista de tareas y calendario.
7. El caso de uso termina

**Condiciones de salida**: La tarea seleccionada es eliminada del sistema.

Capítulo 3: **Diseño del sistema**

## Introducción.

Este proyecto surge de una necesidad personal: siempre he preferido tener mis tareas y actividades organizadas de la mejor manera posible. Habitualmente, realizaba esta organización mentalmente sin registrarla en ningún lugar. Con el tiempo, me di cuenta de que algunas actividades no las podía completar a tiempo, y mi tiempo libre se reducía debido a mi falta de organización y gestión del tiempo.

Por esta razón, comencé a buscar una aplicación que me ayudara a gestionar todas mis tareas de la manera más rápida y sencilla posible. Sin embargo, después de probar varias aplicaciones, descubrí que ninguna me resultaba útil. La mayoría tenían demasiadas funciones, lo que hacía que la tarea principal de crear una tarea se volviera demasiado complicada.

Con esto en mente, decidí investigar las funcionalidades clave de un gestor de tareas, así como otras características interesantes, como las notificaciones en segundo plano. Basándome en toda la información recopilada, me propuse crear mi propia aplicación desde cero.

El objetivo era gestionar mis tareas de una forma mucho más sencilla e intuitiva que las demás aplicaciones disponibles, además de aportar un toque **innovador** implementando el uso de notificaciones en segundo plano utilizando una API proporcionada por la biblioteca de Android Jetpack.

1. **Diagrama de Clases.**

En este apartado, se detallará la construcción de la aplicación. Se abordarán los siguientes aspectos:

* **Diagrama de Clases**: Representación gráfica que muestra la estructura del sistema mediante clases, sus atributos, métodos y las relaciones entre ellas. Este diagrama ayuda a visualizar la organización y las interacciones dentro del sistema.
* **Estructura del Proyecto**: Descripción de la organización del código fuente, los diferentes módulos y componentes que conforman la aplicación.
* **Paquetes Externos (dependencias de la aplicación)**: Lista y descripción de las bibliotecas y paquetes externos utilizados en el proyecto. Esto incluye herramientas de terceros que proporcionan funcionalidades adicionales y facilitan el desarrollo, como las librerías de Android Jetpack, Firebase, entre otros.

Cada uno de estos elementos es crucial para comprender cómo está construida la aplicación, asegurando que sea mantenible, escalable y eficiente.

**Estructura del proyecto**

* Componentes

Aquí se incluyen los componentes reutilizables que se utilizaran en las diferentes clases

* Modelos

Esta carpeta contiene las clases de datos que representan el modelo de la aplicación.

* ViewModels

Los ViewModels contienen la lógica de presentación y gestionan la comunicación entre los modelos y las vistas. Aqui se encontrara todo lo referente a la conexión con nuestra base de datos.

* Navegación

Maneja la navegación entre las diferentes vistas de la aplicación utilizando NavGraph y Vistas.

* notificaciones

Esta carpeta contiene clases relacionadas con el uso de las notificaciones en segundo plano que utiliza la aplicación.

* Log

Esta carpeta contiene clases relacionadas con el sistema de log de nuestra aplicación.

* vista

Aquí se definen las diferentes vistas (UI) de la aplicación.

* MainActivity

El punto de entrada de la aplicación.

**Paquetes Externos (dependencias de la aplicación)**

* Firebase Authentication

implementation(libs.firebase.auth)

Proporciona servicios de autenticación seguros y fáciles de implementar para gestionar usuarios.

* Firebase Firestore

implementation(libs.firebase.firestore)

Base de datos NoSQL en tiempo real para almacenar y sincronizar datos entre usuarios.

* Navegación

implementation("androidx.navigation:navigation-compose:2.7.5")

Biblioteca para gestionar la navegación dentro de aplicaciones Jetpack Compose.

* Material Icons Extended

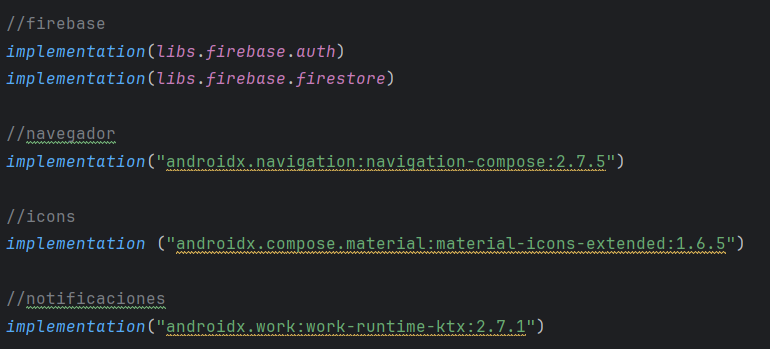
implementation ("androidx.compose.material:material-icons-extended:1.6.5")

Conjunto extendido de íconos de Material Design para su uso en Jetpack Compose.

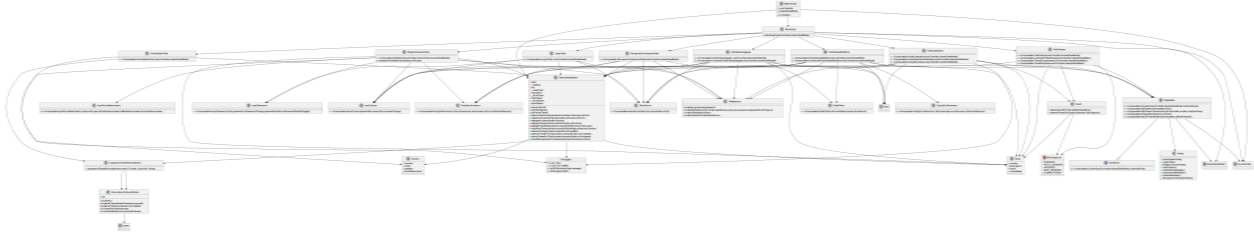
* WorkManager

implementation("androidx.work:work-runtime-ktx:2.7.1")

Biblioteca para gestionar tareas en segundo plano de manera eficiente y programar trabajos periódicos.



**Diagrama de Clases**



1. **Diseño de la Base de Datos.**

En este apartado se detallará el diseño de la base de datos utilizada en la aplicación, la cual está basada en Firebase Firestore, un sistema de base de datos NoSQL en tiempo real. Firebase Firestore permite almacenar y sincronizar datos de manera flexible y eficiente entre usuarios, proporcionando una solución escalable y fácil de integrar con aplicaciones móviles.

**3.3.1.- Diseño Lógico**

El diseño lógico de la base de datos está estructurado en torno a dos colecciones principales: usuarios y tareas.

* **Colección usuarios**
* Documentos: Cada documento en la colección usuarios representa a un usuario individual. El identificador del documento es único para cada usuario, en este caso su ID sera el email que el usuario introduzca al registrarse.
* Campos:
  + - nombre: String, el nombre del usuario.
    - apellido: String, el apellido del usuario.
    - email: String, el correo electrónico del usuario.
    - Fecha de nacimiento: Timestamp, la fecha de nacimiento del usuario.
* **Subcolección tareas**

Cada documento de usuario contiene una subcolección llamada tareas, que almacena las tareas asociadas a ese usuario.

* Documentos: Cada documento en la subcolección tareas representa una tarea individual.
* Campos:
  + - titulo: String, el título de la tarea.
    - descripción: String, la descripción detallada de la tarea.
    - Fecha de Vencimiento: Timestamp, la fecha y hora en que la tarea debe completarse.
    - completada: Boolean, valor para saber si esta completada la tarea.

**3.3.2.- Diseño Relacional**

Aunque Firebase Firestore es una base de datos NoSQL y no utiliza relaciones tradicionales como en las bases de datos SQL, la estructura de documentos y colecciones permite modelar relaciones jerárquicas de manera eficiente. La relación principal en el diseño de la base de datos es la que existe entre los usuarios y sus respectivas tareas:

* **Relación Usuario-Tarea:**

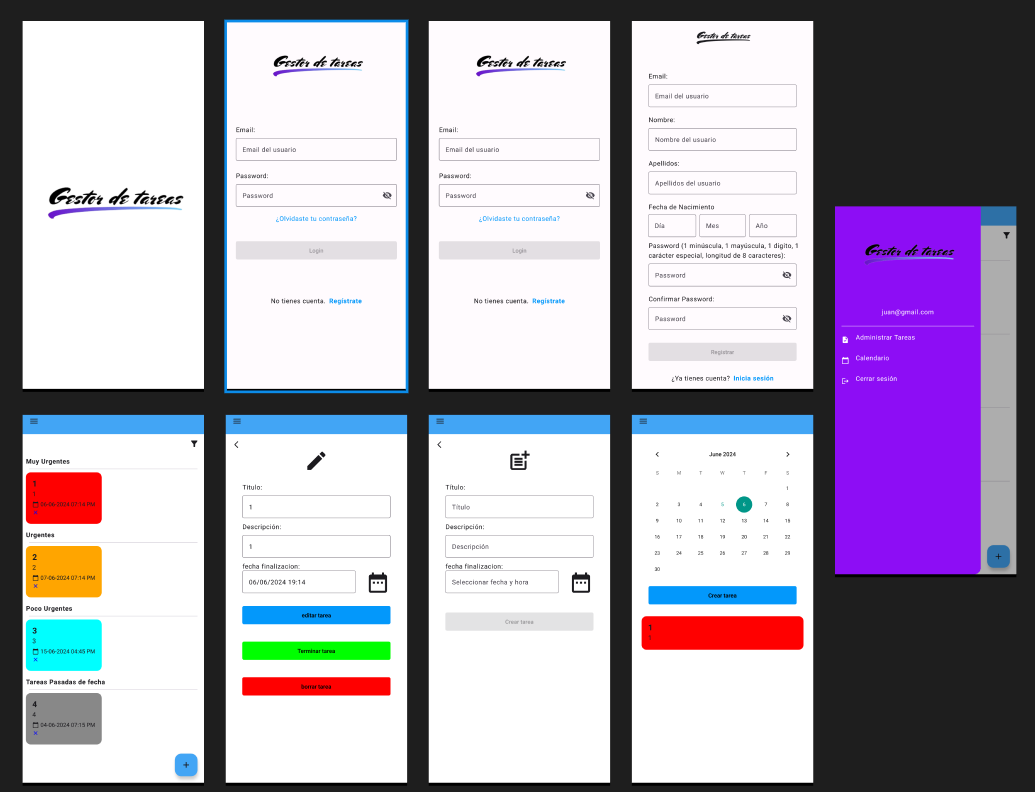
Un usuario tiene una relación de uno a muchos con sus tareas. Esto se modela mediante la subcolección tareas dentro de cada documento de usuario.

1. **Diseño de la Interfaz.**

En este apartado se define cuál va a ser la apariencia visual de la aplicación. Para la creación de la aplicación se ha utilizado Jetpack Compose, una herramienta moderna y potente para el diseño de interfaces de usuario en Android. Jetpack Compose permite crear componentes de interfaz de usuario de manera más eficiente y con menos código, facilitando así el desarrollo de aplicaciones con una apariencia y comportamiento coherentes.

**Diseño Preliminar con Figma**

Antes de proceder con la implementación en Jetpack Compose, se realizó un diseño preliminar utilizando Figma. Figma es una herramienta de diseño colaborativo que permite a los desarrolladores y diseñadores crear prototipos. Este paso es crucial para visualizar la estructura y el flujo de la aplicación antes de comenzar a codificar.



**Diseño Final con Jetpack Compose**

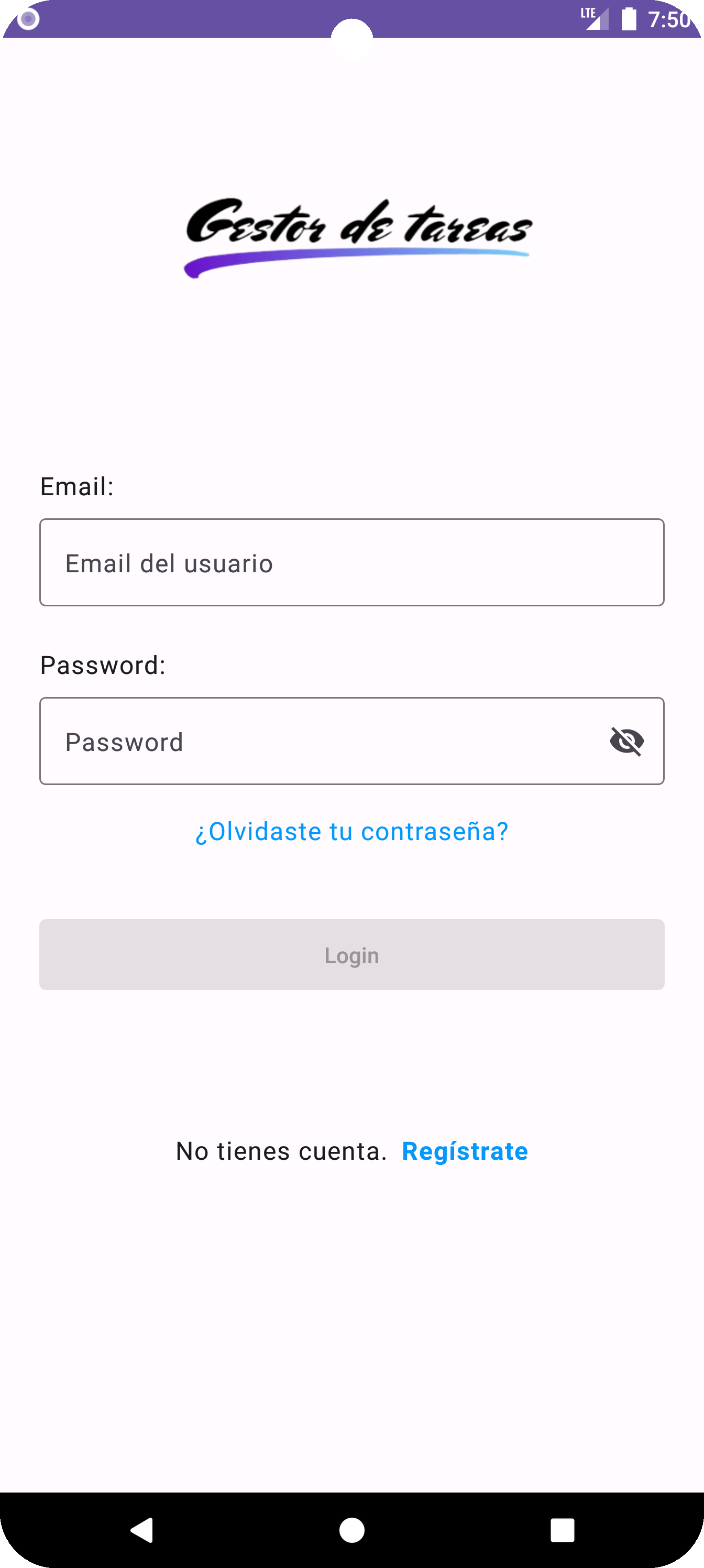
Tras la aprobación del diseño preliminar, se procedió a implementar las pantallas utilizando Jetpack Compose.

A continuación se presentan las pantallas de la aplicación con sus respectivas funcionalidades y diseños finales:

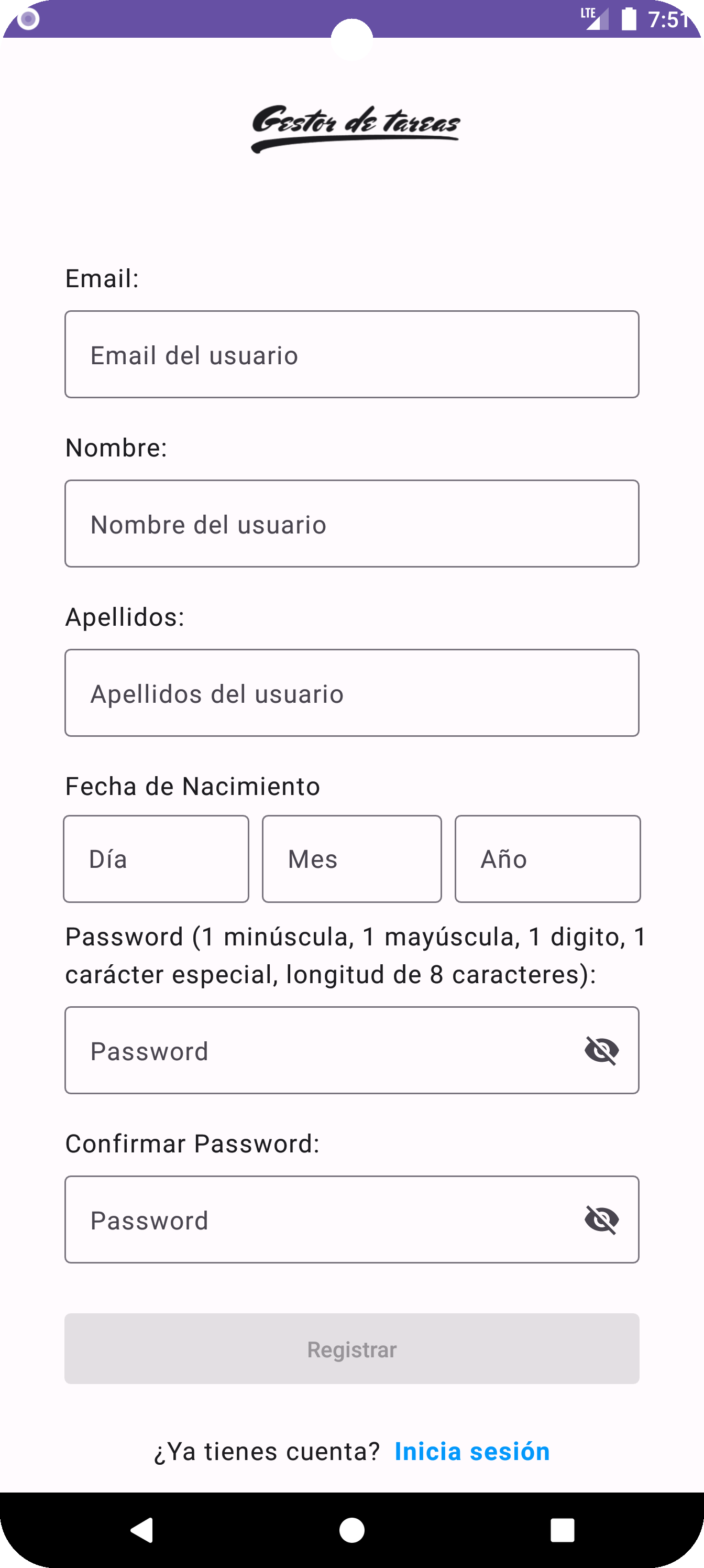
* Pantalla de Inicio: Una pantalla simple que muestra el logo de la aplicación mientras se inicializan los recursos necesarios.



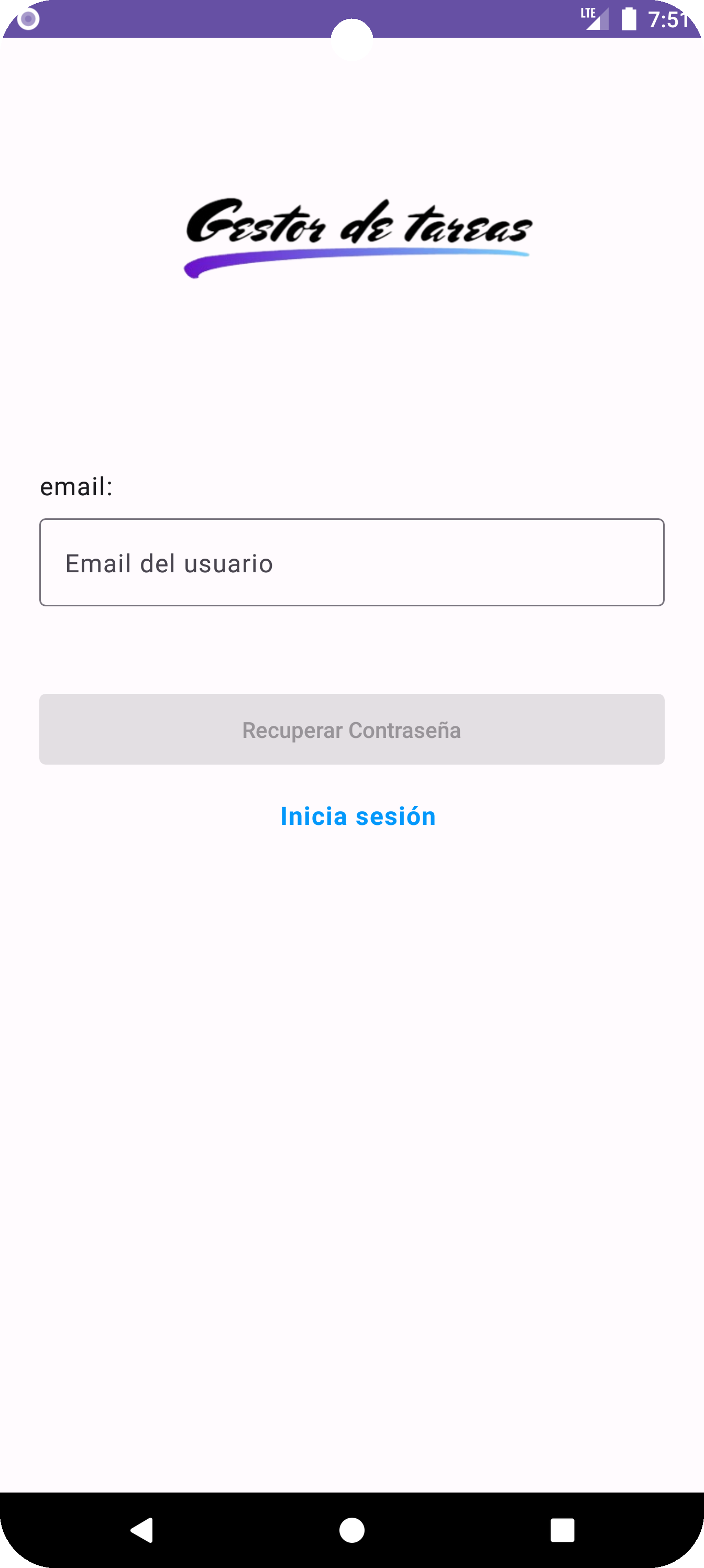
* Pantalla de Inicio de Sesión: Incluye campos de entrada para el correo electrónico y la contraseña, y botones para iniciar sesión, registrarse y recuperar la contraseña.



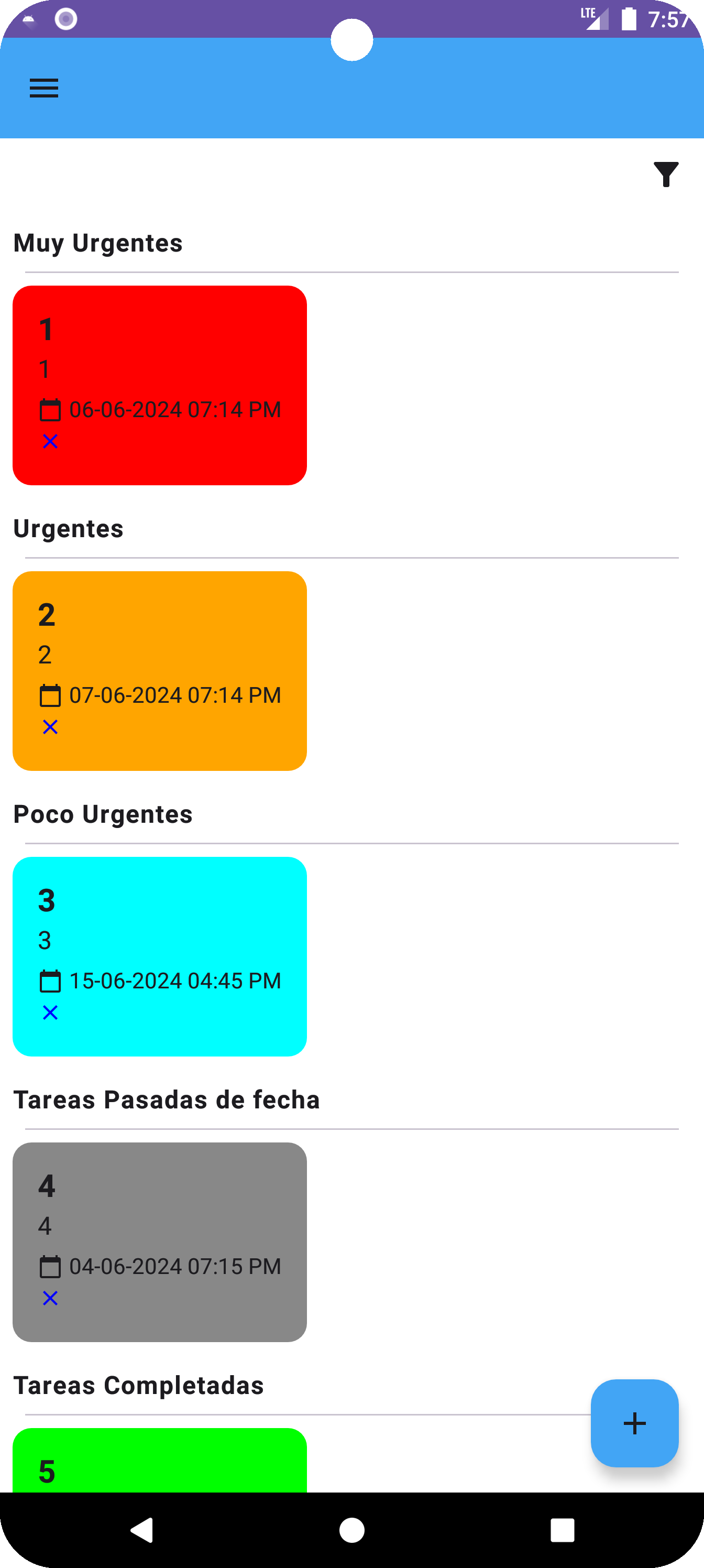
* Pantalla de Registro: Un formulario extenso que captura información del usuario como nombre, apellidos, fecha de nacimiento, correo electrónico y contraseña.



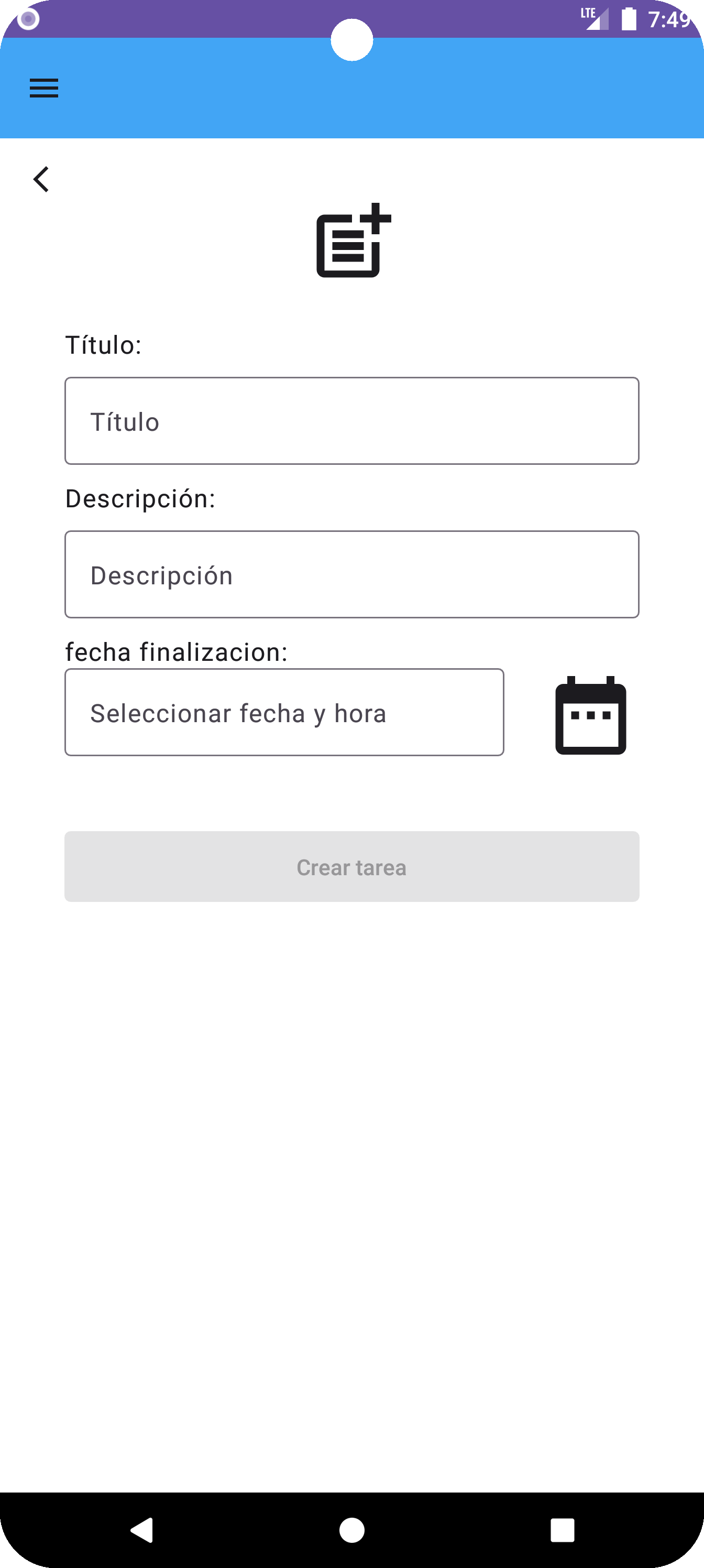
* Pantalla de Recuperación de Contraseña: Permite a los usuarios recuperar su contraseña ingresando su correo electrónico.



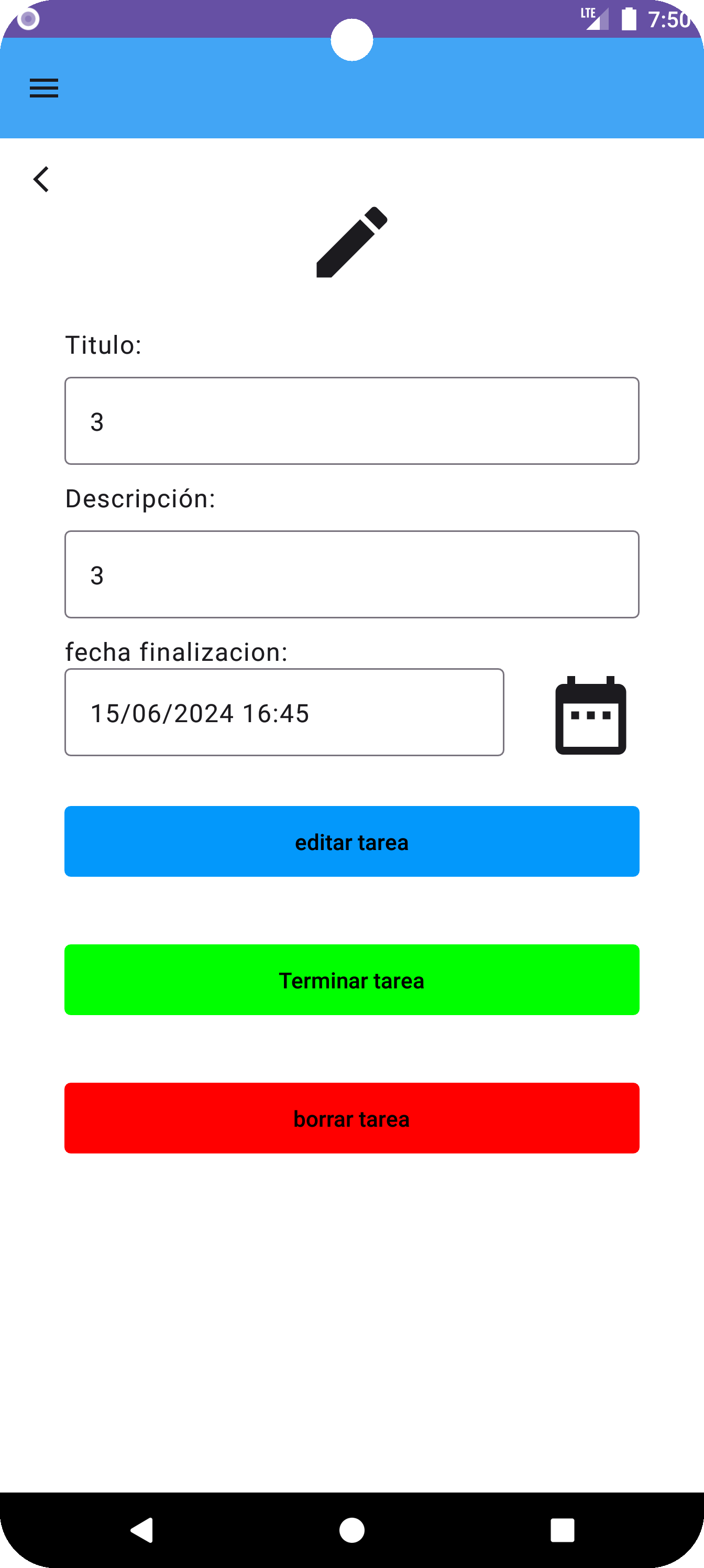
* Pantalla Principal: Muestra una lista categorizada de tareas, destacando aquellas que son muy urgentes, urgentes, menos urgentes, pasadas de fecha y completadas.



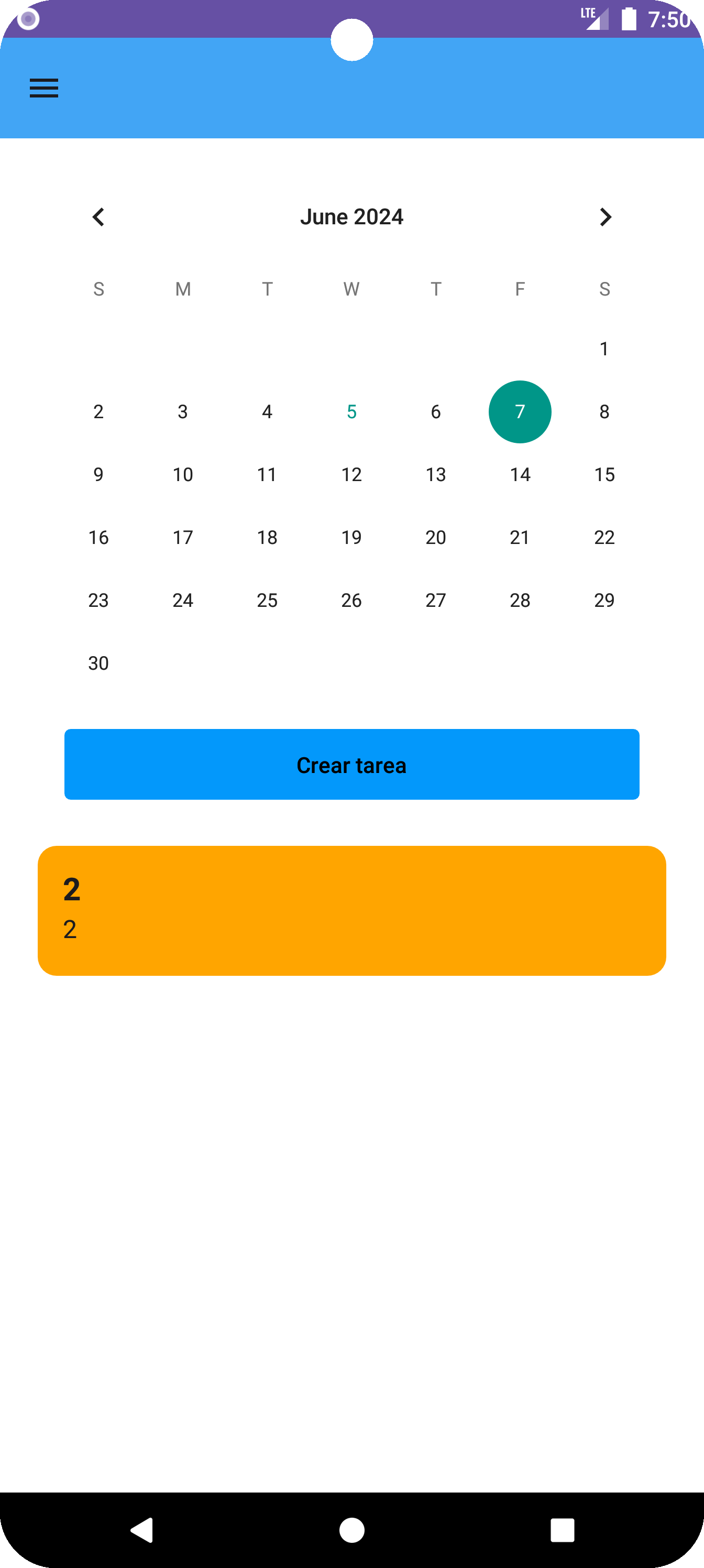
* Pantalla de Creación de Tarea: Un formulario para que los usuarios añadan nuevas tareas, incluyendo el título, descripción y fecha de vencimiento.



* Pantalla de Modificación de Tarea: Permite a los usuarios editar los detalles de una tarea existente.



* Pantalla de Calendario: Una vista de calendario que muestra las fechas de las tareas, permitiendo a los usuarios navegar por los días y ver sus actividades planificadas.



Estas pantallas conforman la interfaz de usuario de la aplicación, diseñada para ser intuitiva, eficiente y visualmente atractiva, proporcionando una experiencia de usuario superior.

Capítulo **4**: **Implementación**

1. **Introducción.**

En este momento ya se encuentra definido el problema y la solución, por lo que lo siguiente será transformar el modelo obtenido en las actividades anteriores en código fuente.

1. **Arquitectura cliente/servidor**

Explicar en qué consiste

1. **Lenguajes de Programación.**

Explicar un poco cada uno de los lenguajes de programación que has utilizado para la realización de tu proyecto.

1. **Herramientas de Desarrollo.**

Explicar que herramientas has utilizado para desarrollar tu aplicación

1. **Codificación.**

Todo el código desarrollado se encuentra disponible en el pendrive / moodle que se adjunta con esta memoria (a determinar por el departamento de informática).

Capítulo **5**: **Pruebas de software**

**5.1.-Introducción.**

En el desarrollo de cualquier software, una de las actividades asociadas a este proceso es la prueba. En este capítulo se detallarán las pruebas realizadas al software desarrollado. Se intentará valorar la calidad de la aplicación generada, así como detectar y corregir posibles errores.

Breve descripción de que son las pruebas del software.

**5.2.- Técnicas de Prueba.**

Explicación de las pruebas de caja blanca y caja negra

**5.2.1.- Pruebas de caja blanca o enfoque estructural.**

Explicar qué son y en qué consisten este tipo de pruebas.

Puedes hacer diagrama de flujo del sistema.

**Nota: en caso de proyectos amplios es posible que no se realice documentación sobre ninguna de estas pruebas puesto que la documentación de las mismas sería muy tediosa. Menciona los aspectos que se han tenido en cuenta en caso de estar en esta situación.**

**5.2.2.- Pruebas de caja negra o enfoque funcional.**

Explicar qué son y en qué consisten este tipo de pruebas.

Estas pruebas se realizan tras la fase de codificación y nos van a determinar si el requisito de una aplicación es parcial o completamente satisfactoria.

Puedes aplicar pruebas mediante JUnit. Si tu programa es muy grande pon las que te resulten más importantes.

Capítulo **6**: **Conclusiones.**

**6.1.- Conclusiones**

**6.2.- propuestas Futuras (opcional).**

Ampliaciones y/o mejoras que podrías añadir a tu proyecto en un futuro.

Capítulo **7**: **Bibliografía y referencias**

**7.1.-Referencias bibliográficas.**

En este apartado se citan las referencias bibliográficas, los artículos electrónicos y las páginas Web utilizadas en el proyecto.

Anexo **1**: **Manual de Instalación**

(Detallad paso a paso la instalación de las aplicaciones necesarias para desarrollar dicha aplicación con los apartados que veáis necesarios)

Anexo 2: **Manual de Usuario**

(Detallad paso a paso, con los apartados que consideres necesarios, para que el usuario entienda todas las opciones que hay en tu aplicación para su uso)