Konnekt

Proyecto Integrado 2º DAM
Pablo López Lozano
I.E.S. Hermenegildo Lanz

ÍNDICE

1. ANÁLISIS DEL PROBLEMA

1.1. INTRODUCCIÓN

El proyecto Konnekt consiste en el desarrollo de una aplicación de red social nativa para Android, diseñada para permitir a los usuarios interactuar de manera dinámica y personalizada. La aplicación ofrece funcionalidades como registro e inicio de sesión, creación y personalización de perfiles, publicación de posts, likes, guardar posts, seguimiento de usuarios, chat entre usuarios, y la posibilidad generar comentarios con inteligencia artificial (IA) para enriquecer la experiencia del usuario.

1.2. OBJETIVOS

El objetivo principal del proyecto es crear una aplicación de red social funcional que:

a. Permita a los usuarios registrarse, iniciar sesión y gestionar perfiles personalizados.
b. Facilite la publicación y visualización de contenido multimedia.
c. Ofrezca un sistema de mensajería instantánea entre usuarios.
d. Explore la integración de IA para funcionalidades como comentarios automáticos sobre fotos o un chatbot interactivo.
e. Sea escalable, eficiente y fácil de usar, con una interfaz moderna diseñada en Figma.

1.3. FUNCIONES Y RENDIMIENTOS DESEADOS

El sistema ofrecerá las siguientes funcionalidades:

a. **Registro e Inicio de Sesión**: Autenticación segura mediante correo y contraseña.
b. **Gestión de Perfiles**: Creación y personalización de perfiles con información como nombre de usuario, imagen de perfil, y configuración de privacidad.
c. **Publicación de Posts**: Subida de imágenes con descripciones y soporte para likes y comentarios.
d. **Seguimiento de Usuarios**: Sistema de seguidores y seguidos para conectar usuarios.
e. **Chat**: Mensajería instantánea entre usuarios con indicadores de lectura.
f. **Notificaciones**: Alertas para interacciones como comentarios, likes o nuevos seguidores (opcional, dependiendo de la implementación de Firebase Cloud Messaging).
g. **Integración de IA**: Exploración de un chatbot o generación de comentarios automáticos para fotos, optimizando recursos.

Se espera que la aplicación sea rápida, consuma recursos moderados y ofrezca una experiencia de usuario fluida, con un diseño moderno basado en Jetpack Compose.

1.4. PLANTEAMIENTO Y EVALUACIÓN DE DIVERSAS SOLUCIONES

Se consideraron las siguientes alternativas para el desarrollo:

Lenguaje de Programación:

- Kotlin con Jetpack Compose: Moderno, nativo para Android, y soporta interfaces dinámicas.
- Java con XML: Tradicional, pero menos flexible para UI moderna.
- Flutter: Multiplataforma, pero requiere aprendizaje adicional y no es nativo puro. Backend

Base de Datos:

- Firebase (Firestore y Storage): Fácil integración con Android, escalable, y soporta notificaciones.
- MongoDB con API REST propia: Mayor control, pero requiere más tiempo para configurar el servidor.
- MySQL: Más lento pero mayor seguridad de los datos.

Integración de IA:

- IA local: Alta demanda de recursos en dispositivos móviles.
- IA basada en la nube: Más eficiente, pero depende de conexión a internet.
- Chatbot simple: Menor consumo de recursos y más factible para la primera versión.

1.5. JUSTIFICACIÓN DE LA SOLUCIÓN ELEGIDA

Se eligió ***Kotlin*** con ***Jetpack Compose*** por ser el estándar moderno para desarrollo Android, ofreciendo un diseño de UI declarativo y eficiente.
MongoDB como una base de datos para almacenar datos estructurados, aprovechando su flexibilidad con documentos JSON. También porque algunos servicios de Firebase como Firestore para almacenar imágenes requiere de pago.
Para la ***mensajería instantánea*** y la ***conexión con la base de datos***, un backend en python con websocket para la mensajería y endpoints POST y GET para obtener y modificar la base de datos.
Para la ***IA***, se optó por explorar diversos modelos de IA con Ollama de procesamiento local, finalmente se logró encontrar uno que encaja bien en la aplicación (minicpm-v).
Figma se usó para diseñar la interfaz, permitiendo prototipos rápidos y adaptables.

1.6. MODELADO DE LA SOLUCIÓN

1.6.1. RECURSOS HUMANOS

El proyecto es desarrollado por un único desarrollador, Pablo López Lozano, quien se encarga de:

- Análisis y diseño del sistema.
- Desarrollo de la aplicación (frontend y backend).
- Diseño de la interfaz en Figma.
- Pruebas y documentación

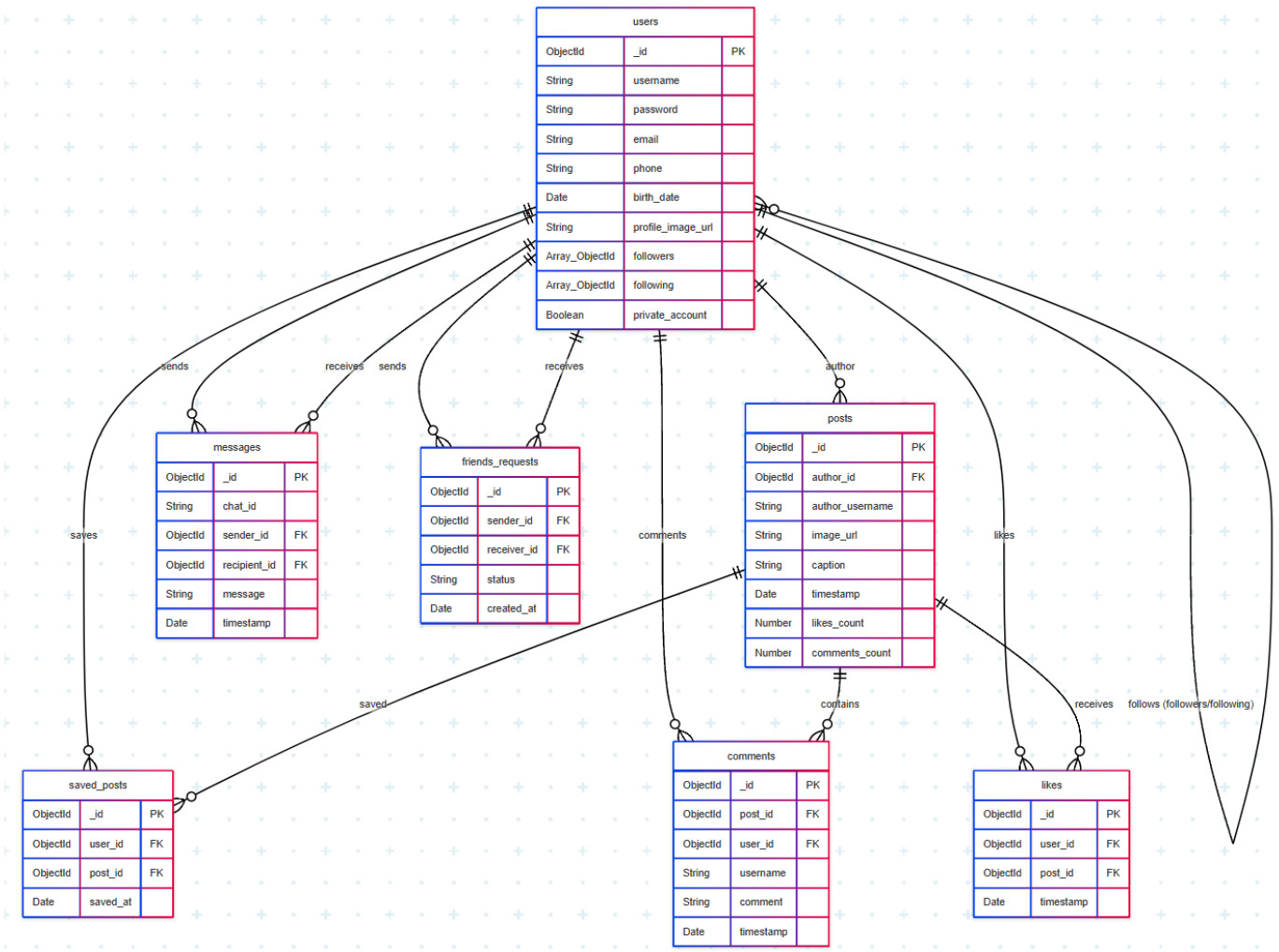
1.6.2. RECURSOS HARDWARE

- Portátil con procesador Intel i5, 16 GB RAM, y SSD con acceso a internet para desarrollo.
- Dispositivos Android para pruebas.

1.6.3. RECURSOS SOFTWARE

- **Android Studio**: IDE para desarrollo Android. Kotlin 1.9+ y Jetpack Compose: Para desarrollo de la aplicación.
- **Backend Python**: para guardar imágenes y API para la conexión entre la base de datos y la aplicación.
- **MongoDB**: Base de datos secundaria para datos estructurados.
- **Figma**: Diseño de prototipos de UI/UX.
- **Git**: Control de versiones.

1.6.4. MODELO DE LA BASE DE DATOS



1.7. PLANIFICACIÓN TEMPORAL

El proyecto se divide en las siguientes fases (duración aproximada):

1. **Análisis y Diseño (2 semanas)**: Definición de requisitos, diseño de UI en Figma, y esquema de base de datos.
2. **Desarrollo del Backend (3 semanas)**: Configuración de Firebase y MongoDB, implementación de autenticación y almacenamiento.
3. **Desarrollo del Frontend (4 semanas)**: Implementación de pantallas (registro, perfil, posts, chat) con Jetpack Compose.
4. **Integración de IA (2 semanas)**: Investigación e implementación de un modelo con Ollama para el procesamiento de imágenes y generación de comentarios.
5. **Pruebas (2 semanas)**: Pruebas de integración, y de usuario.
6. **Documentación (1 semana)**: Redacción de documentación.

2. DISEÑO E IMPLEMENTACIÓN DEL PROYECTO

El diseño del proyecto Konnekt se ha basado en una arquitectura cliente-servidor, donde la app Android funciona como cliente y un backend en Python se encarga de gestionar la lógica del servidor, almacenamiento de datos y comunicación en tiempo real.

El diseño de la interfaz se realizó en Figma, la interfaz fue implementada en Kotlin usando Jetpack Compose, un framework moderno que permite construir interfaces creando componentes.

Para el backend, se empleó Python con WebSockets para la mensajería y un conjunto de endpoints REST desarrollados con FastAPI. La base de datos es MongoDB, organizada en colecciones: users, posts, comments, likes, messages, saved_posts y friends_requests.
Esta estructura permite una alta escalabilidad y flexibilidad en el almacenamiento de información.

Además, se ha integrado una funcionalidad de IA para generar comentarios automáticos a partir de fotos, utilizando el modelo minicpm-v con Ollama, procesado localmente.

3. FASE DE PRUEBAS

Durante el desarrollo de Konnekt se han realizado diferentes tipos de pruebas, tanto manuales como automatizadas, en las fases finales del proyecto:

Pruebas realizadas:

- **Pruebas de funcionalidad**:
    - Registro, login
    - Creación, edición y visualización de perfiles
    - Subida de imágenes y visualización de posts
    - Likes, guardado de publicaciones y comentarios
    - Seguimiento y dejar de seguir usuarios
    - Sistema de chat entre usuarios

- **Pruebas de integración**:
    - Comunicación entre frontend y backend a través de WebSockets
    - Envío de datos al backend y persistencia en MongoDB
    - Recuperación de publicaciones y mensajes
- **Pruebas de rendimiento**:
    - Verificación del rendimiento de la aplicación en distintos dispositivos
    - Pruebas de latencia en el sistema de mensajería
    - Medición de carga de la IA al generar comentarios desde imágenes
- **Pruebas de usabilidad**:
    - Interfaz adaptativa probada en diferentes dispositivos
    - Comprobación de accesibilidad básica (colores, botones, navegación clara)

4. DOCUMENTACIÓN DE LA APLICACIÓN

4.1. INTRODUCCIÓN A LA APLICACIÓN

Konnekt es una red social móvil desarrollada para Android donde los usuarios pueden registrarse, crear un perfil, publicar imágenes, seguir a otros usuarios e interactuar mediante mensajes o comentarios. Su diseño busca una experiencia moderna e intuitiva con una funcionalidad única: generación automática de comentarios usando inteligencia artificial.

4.2. MANUAL DE INSTALACIÓN

Frontend:

1. Clonar el repositorio desde GitHub (https://github.com/Pablolopezlo15/Konnekt).
2. Abrir el proyecto en Android Studio.
3. Instalar dependencias requeridas por Gradle.
4. Cambiar la ip de tu servidor en ***config/AppConfig***
5. Conectar el dispositivo Android en modo desarrollador o usar el emulador.
6. Ejecutar la aplicación desde Android Studio.

Backend:

Opción 1 Docker:

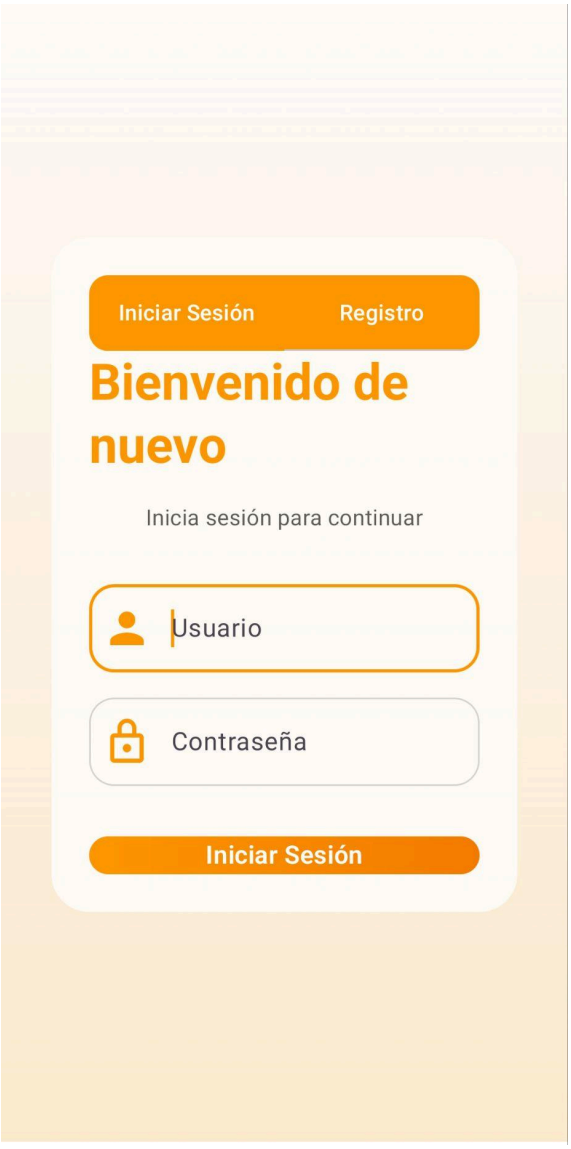
1. Entrar en la carpeta /Backend
2. Lanzar el contenedor Docker → ***docker compose build up***

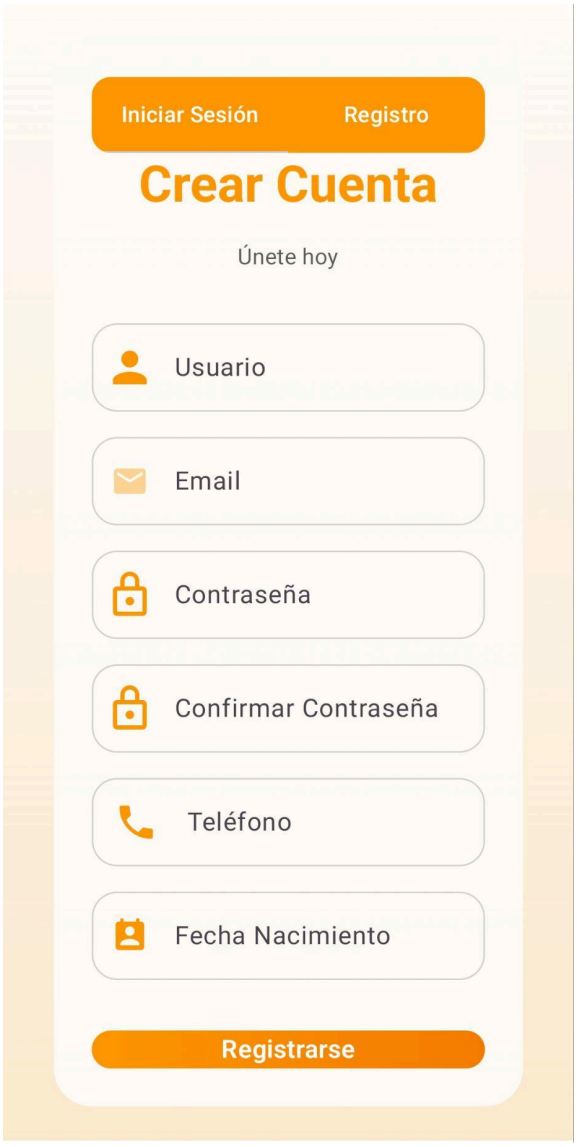
Opción 2 Python:

1. Instalar python (https://www.python.org/downloads/)
2. Entrar en la carpeta /Backend
3. Instalar dependencias necesarias → ***pip install -r requirements.txt***
4. Instalar OpenSSL
5. Lanzar el script start.sh para generar las claves SSL
6. Lanzar el main.py → ***uvicorn main:app --host [TU IP]--port 8000 --ssl-keyfile=certs/key.pem --ssl-certfile=certs/cert.pem***

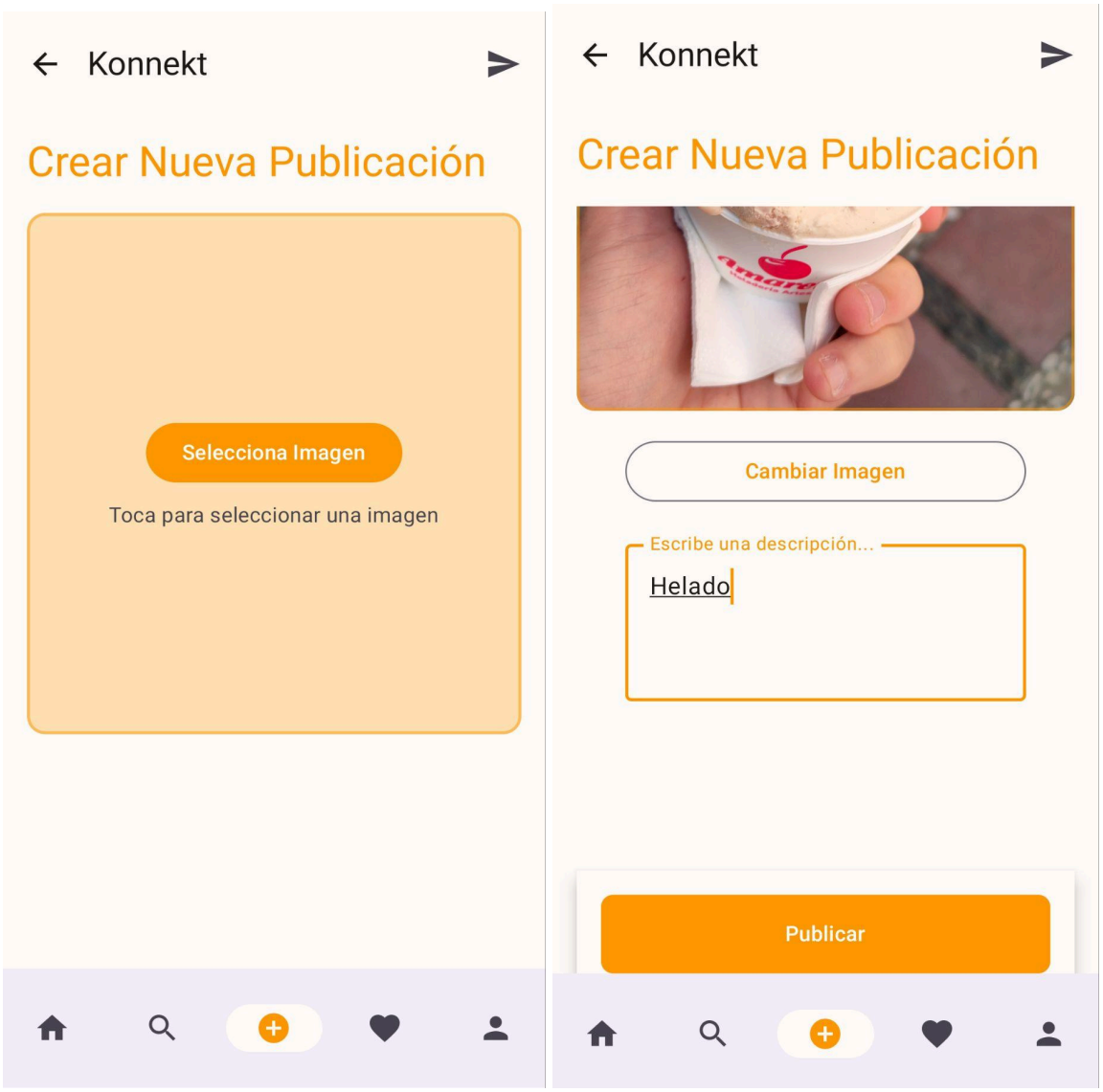
4.3. MANUAL DE USUARIO

Inicio de sesión: Introduce tu email y contraseña.
Registro: Rellena tus datos y crea tu cuenta.

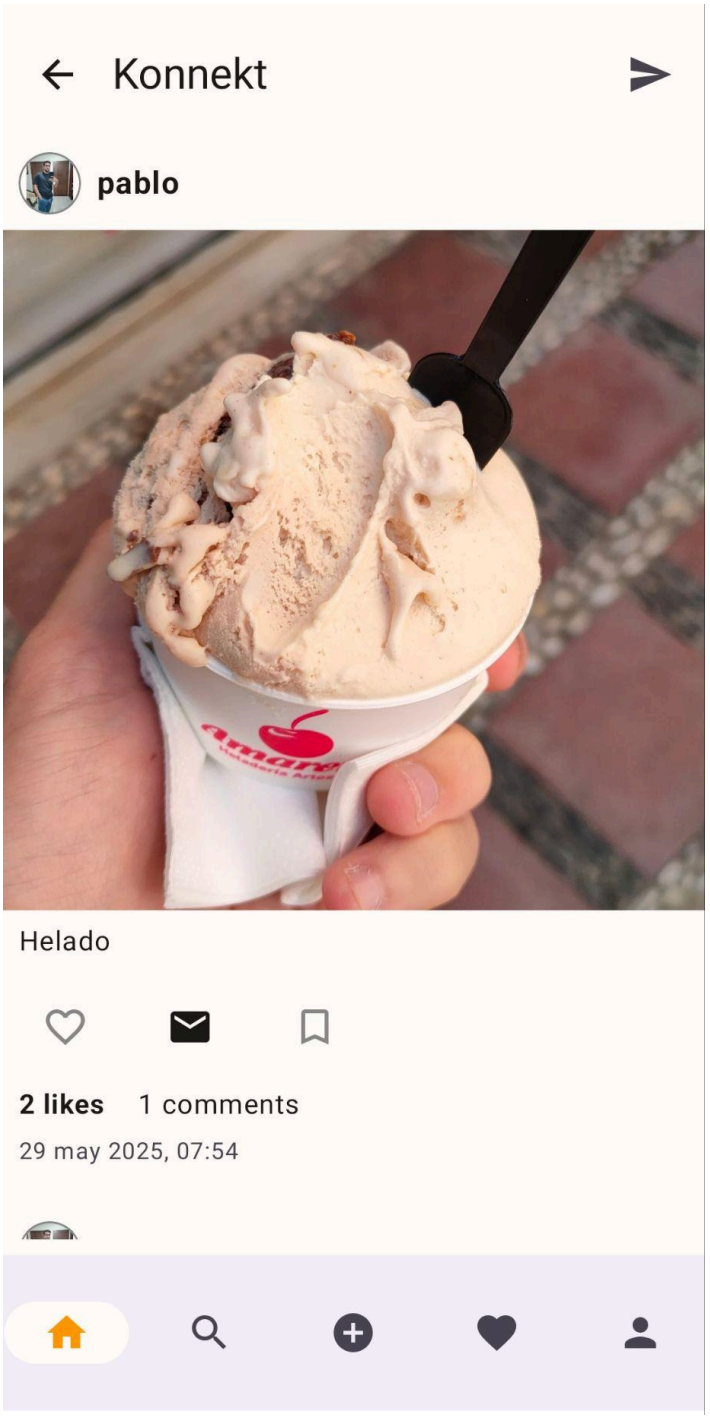




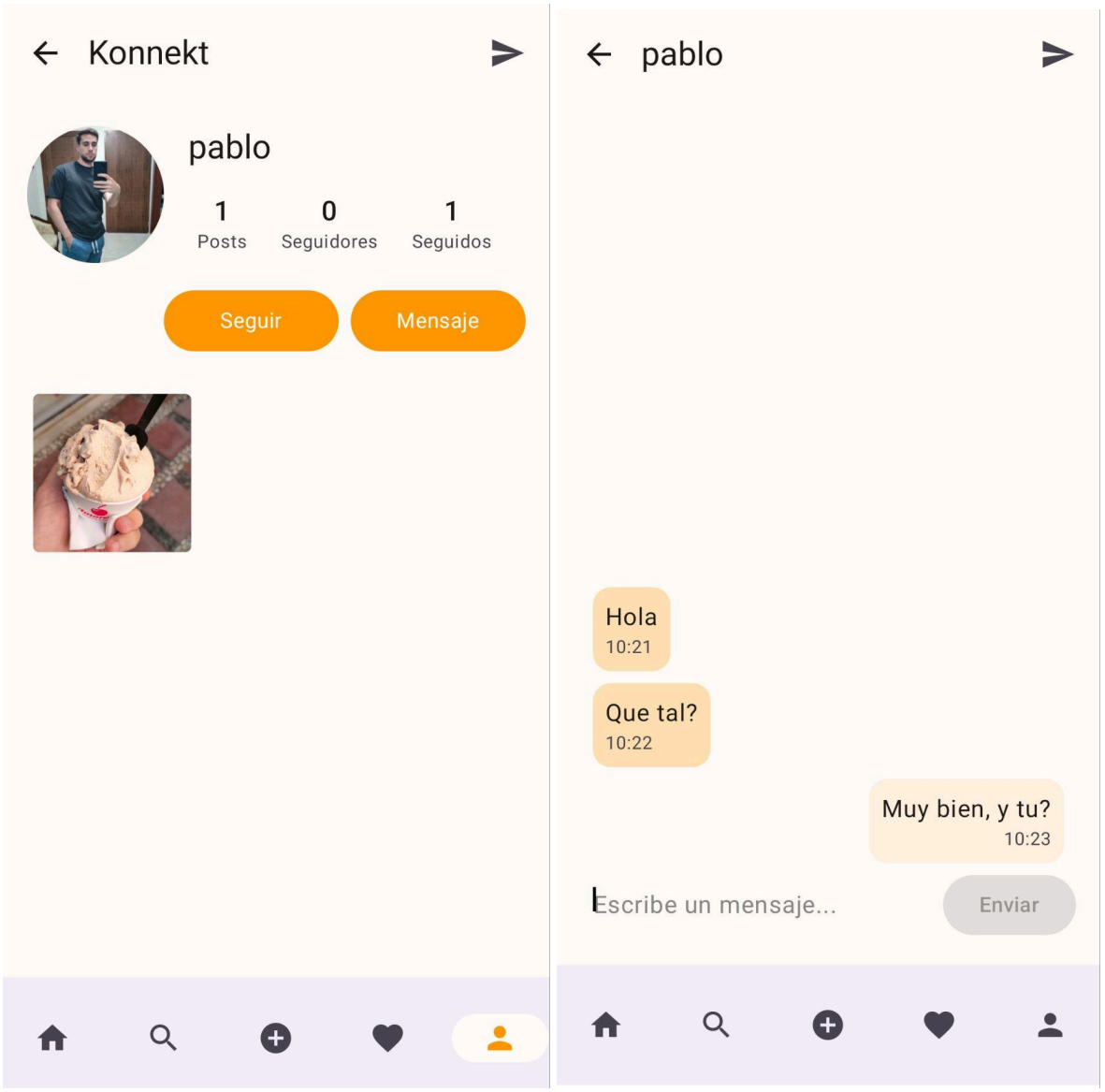
Publicar contenido: Toca el botón de +, selecciona una imagen, añade descripción y publícala.



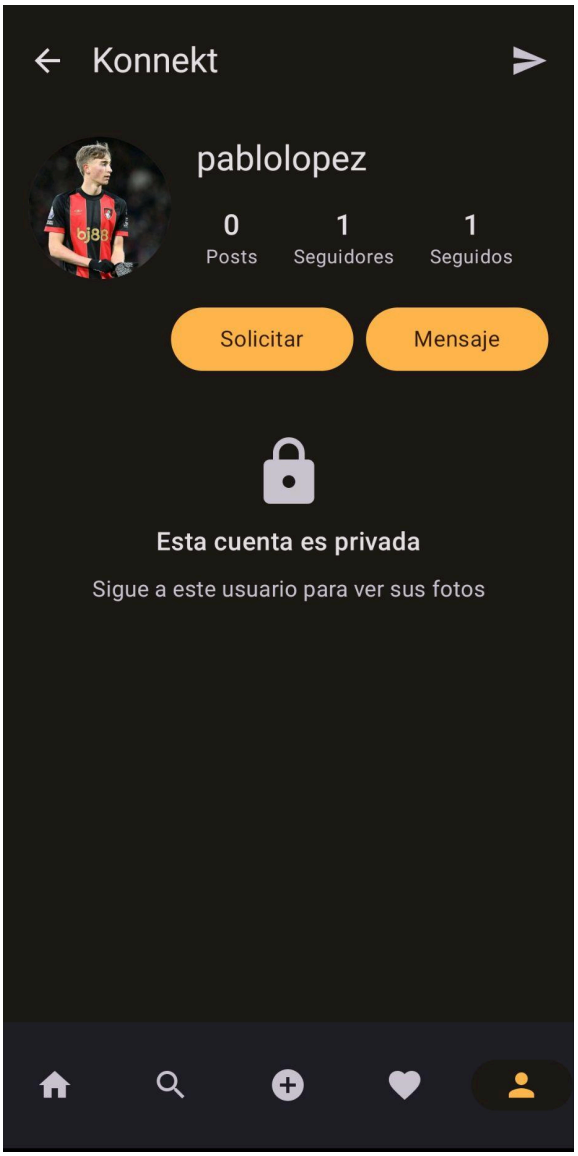
Interacción: Puedes dar “me gusta” dándole al corazón o dándole 2 veces a la pantalla en la foto, comentar y guardar publicaciones.

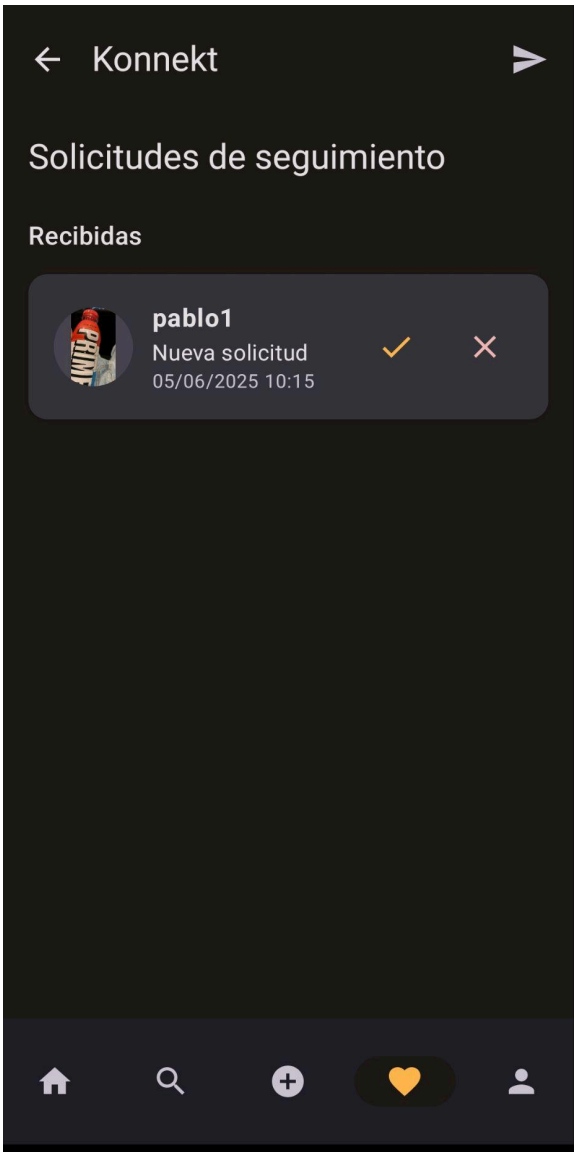


Chat: Entra al perfil de un usuario y pulsa "Mensaje".

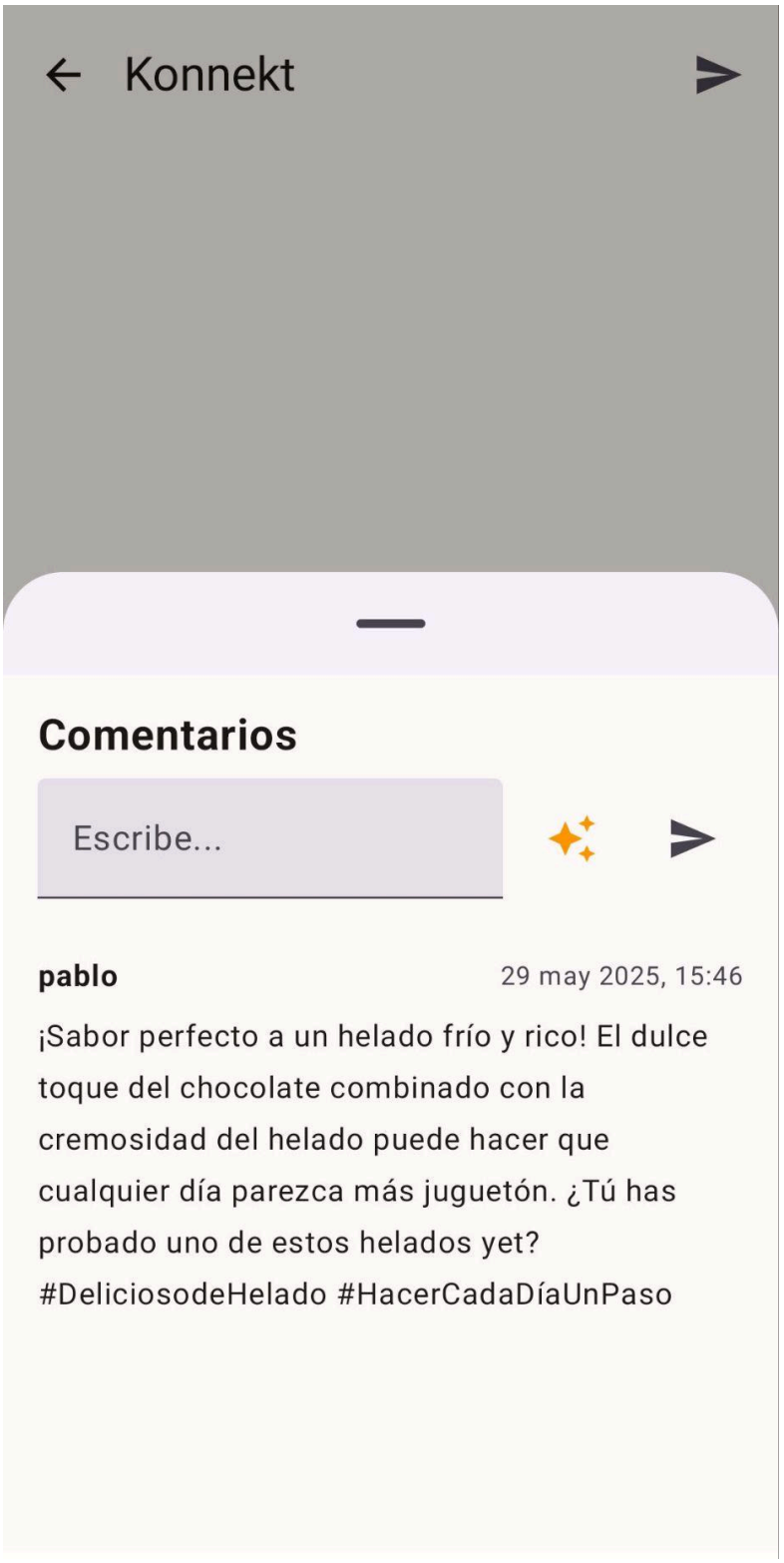


Solicitudes de seguimiento: Para poder ver fotos de cuentas privadas, debes ser seguidor, dale a solicitar. El otro usuario recibirá una solicitud que puede aceptar o rechazar.

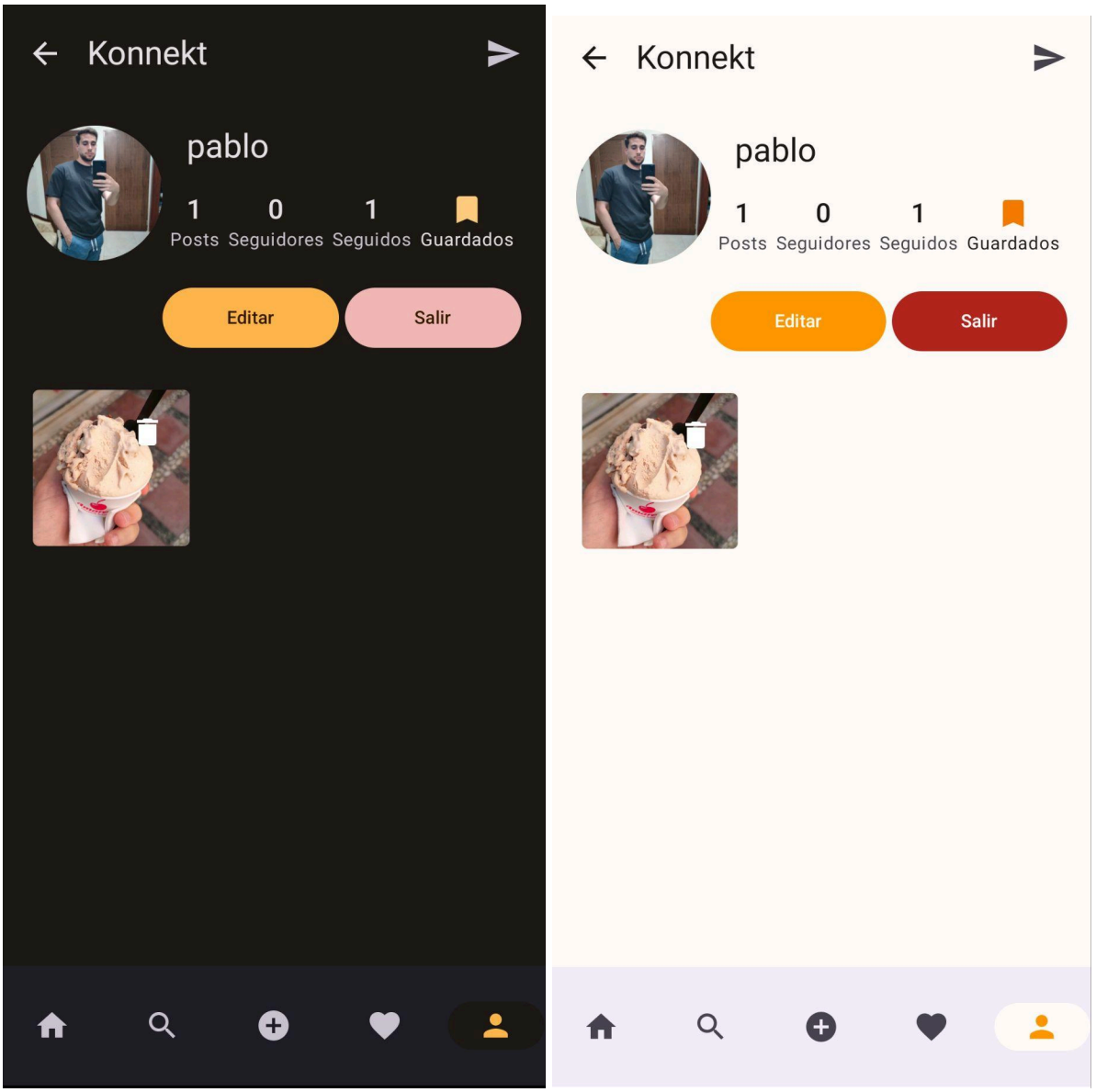




IA: Puedes generar un comentario automático en cualquier post con IA.



Tema de la app: Tema claro u oscuro según la configuración del dispositivo



4.4. MANUAL DE ADMINISTRACIÓN

Actualmente no hay panel administrativo tradicional. La administración del sistema se realiza a través de herramientas de base de datos como MongoDB Compass o Postman para gestionar endpoints y probar API.

5. CONCLUSIONES FINALES

5.1. GRADO DE CUMPLIMIENTO DE LOS OBJETIVOS

El proyecto ha cumplido la mayoría de los objetivos establecidos:

- ☑ ~~Registro, login y gestión de usuarios~~
- ☑ ~~Publicación de imágenes con likes y comentarios~~
- ☑ ~~Sistema de seguidores y mensajería~~
- ☑ ~~Pruebas funcionales y de integración completadas~~
- ☑ ~~Exploración e integración de IA local~~

5.2. PROPUESTA DE MODIFICACIONES O AMPLIACIONES

- ☐ Mejorar la velocidad de la IA para generación de comentarios
- ☐ Implementar historias al estilo Instagram
- ☐ Incluir notificaciones push
- ☐ Publicar la app en Google Play Store

6. BIBLIOGRAFÍA

- **Jetpack Compose Documentation**. Documentación oficial de Jetpack Compose, utilizada como referencia para el diseño e implementación de interfaces gráficas en Android. https://developer.android.com/jetpack/compose

- **Kotlin Language Reference**. Manual oficial del lenguaje Kotlin, consultado para la programación de lógica de la aplicación y manejo de estructuras de datos. https://kotlinlang.org/docs/home.html

- **Flask Documentation**. Guía oficial de Flask (Python), utilizada como base para la creación de la API REST del backend. https://flask.palletsprojects.com/

- **MongoDB Manual**. Documentación oficial de MongoDB, empleada para el diseño de colecciones, consultas y conexión desde el backend en Python. https://www.mongodb.com/docs/

- **Ollama - minicpm-v**. Modelo de inteligencia artificial utilizado para pruebas de generación de comentarios automáticos a partir de imágenes. https://ollama.com/library/minicpm-v

- **Figma**. Plataforma de diseño colaborativo utilizada para el prototipado de interfaces de usuario y diseño de pantallas de la aplicación. https://www.figma.com/

- **Git Documentation**. Manual oficial del sistema de control de versiones Git, utilizado para gestionar el código fuente y las versiones del proyecto.
  https://git-scm.com/doc