SECRETGIFT

Miguel Abelleira

Daniel Hernández

Daniel Pérez

Contenido

Descripción

SecretGift es una aplicación diseñada para replicar el popular juego del Amigo Invisible. Los usuarios podrán crear salas privadas y enviar invitaciones a amigos mediante correo electrónico. Una vez dentro de una sala, los participantes se asignan aleatoriamente entre ellos para intercambiar regalos.

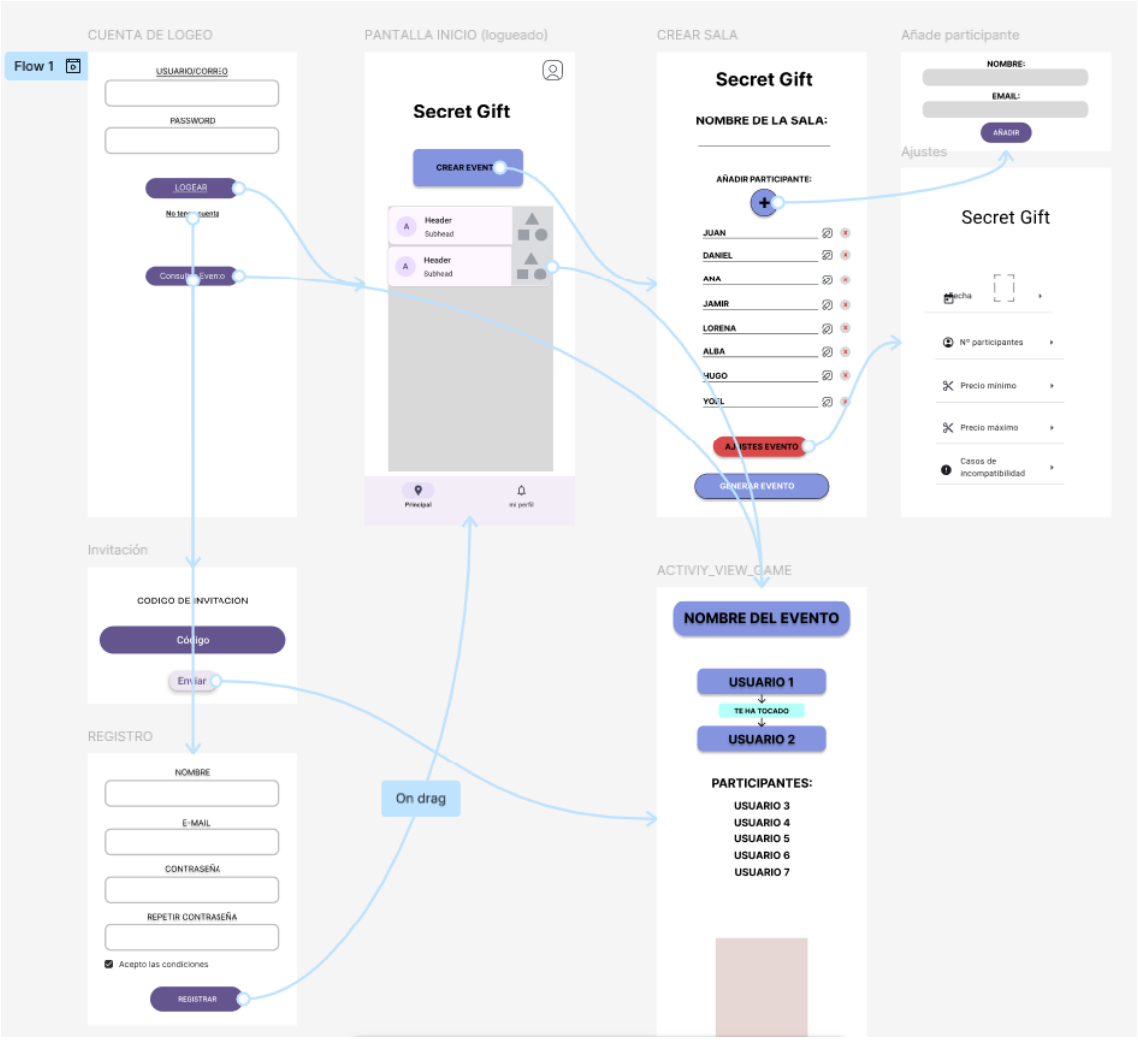
La aplicación ofrece una interfaz intuitiva que permite gestionar fácilmente los eventos, configurar reglas específicas como presupuestos máximos y mínimos, y establecer restricciones entre participantes.

SecretGift está diseñada para proporcionar una experiencia segura y fluida, integrando autenticación mediante Firebase Authentication y almacenamiento de datos con NodeJs y MongoDB. También cuenta con publicidad a través de Google AdMob y una arquitectura optimizada basada en Clean MVVM

Tecnologías utilizadas

- **Lenguaje:** Kotlin
- **IDE:** Android Studio
- **Arquitectura:** Clean MVVM
- **Base de datos local:** Room
- **Backend:** NodeJs + MongoDB
- **Autenticación:** Firebase Authentication
- **Inyección de dependencias:** Hilt
- **Networking:** Retrofit
- **Gestor de estados y UI:** LiveData
- **Publicidad:** Google AdMob

Flujo de la aplicación



1. **Inicio** → (Login, Registro o Código de Evento).
2. **Pantalla Principal** → (Ver salas o crear una nueva).
3. **Creación de Sala** → (Nombre y participantes).
4. **Ajustes de Sala** → (Reglas, fecha y presupuesto).
5. **Generación del Evento** → (Asignación automática de regalos).
6. **Visualización del Evento** → (Acceder desde menú o código).

Estructura de paquetes

La aplicación sigue una arquitectura Clean MVVM con la siguiente organización de paquetes:

- **Data:** Contiene las implementaciones de repositorios, DAOs, fuentes de datos remotas y locales. Se encarga de manejar la obtención y almacenamiento de la información utilizada en la aplicación.
- **Di:** Contiene los módulos de inyección de dependencias utilizando Hilt. Se encarga de proveer instancias de los repositorios, fuentes de datos y otros componentes necesarios en la aplicación.
- **Domain:** Define la lógica de negocio mediante casos de uso y entidades. Aquí se encapsulan las reglas y validaciones que rigen el comportamiento de la aplicación.
- **Presentation:** Contiene los ViewModels y la UI de la aplicación. Se encarga de manejar la comunicación entre la capa de dominio y la interfaz de usuario, asegurando que los datos mostrados sean correctos y estén actualizado

Dependencias utilizadas en SecretGift

SecretGift utiliza diversas librerías para garantizar un desarrollo eficiente y optimizado en Android. A continuación, se detallan algunas de las principales dependencias incluidas en el proyecto:

Arquitectura y Gestión del Ciclo de Vida

- **Fragment KTX:** Facilita el uso de Fragmentos con extensiones de Kotlin.
- **Lifecycle ViewModel:** Permite gestionar el ciclo de vida de la UI y almacenar datos de manera persistente.
- **LiveData:** Proporciona datos reactivos que se actualizan automáticamente en la UI.

Inyección de Dependencias

- **Hilt:** Framework de inyección de dependencias oficial de Android que simplifica la gestión de dependencias en el proyecto.

Base de Datos Local

- **Room:** Abstracción sobre SQLite que facilita la persistencia de datos y reduce el código repetitivo.

Red y Serialización

- **Retrofit:** Cliente HTTP para consumir APIs REST de manera sencilla.
- **Gson:** Biblioteca para convertir objetos Java/Kotlin en JSON y viceversa.
- **Logging Interceptor:** Permite depurar y monitorear las solicitudes HTTP realizadas con Retrofit.

Autenticación y Almacenamiento en la Nube

- **Firebase Authentication:** Sistema de autenticación de usuarios mediante correo,entre otros.
- **Firestore:** Base de datos en la nube de Firebase, utilizada para almacenar información en tiempo real.

Interfaz de Usuario y Diseño

- **Flexbox:** Permite diseñar interfaces flexibles y adaptables en la aplicación.

Publicidad

- **AdMob:** Plataforma de monetización que permite incluir anuncios en la aplicación.

Análisis de Requisitos

Requisitos Funcionales

1. **Registro de usuario**: Los usuarios pueden registrarse proporcionando su correo electrónico y una contraseña.
2. **Inicio de sesión**: Los usuarios pueden iniciar sesión con sus credenciales de correo electrónico y contraseña, con sesión persistente.
3. **Creación de eventos**: Los usuarios pueden crear eventos, asignar participantes y definir reglas como el presupuesto máximo y las incompatibilidades.
4. **Gestión de participantes**: Los organizadores pueden agregar, eliminar y ver los detalles de los participantes, asignando los regalos de forma aleatoria.
5. **Asignación de regalos**: La aplicación asigna los regalos de manera aleatoria, garantizando que no se asignen regalos a sí mismos.
6. **Visualización de eventos**: Los usuarios pueden ver eventos creados, acceder a los detalles y consultar la lista de participantes.

Requisitos No Funcionales

1. **Seguridad**: Protección de los datos personales mediante Firebase Authentication
2. **Usabilidad**: Interfaz intuitiva para facilitar la creación y gestión de eventos.
3. **Rendimiento**: Uso eficiente de recursos, garantizando una experiencia fluida.
4. **Escalabilidad**: Arquitectura basada en MVVM que facilita la expansión futura de la aplicación.