DinoPass

Juan Marcial Padrón Cedrés

Desarrollo de aplicaciones multiplataforma(I.E.S Haría)

2º GS

05 de abril de 2024

Resumen
Este documento proporciona una descripción detallada de la aplicación DinoPass, un servicio de game pass que ofrece tres planes mensuales diferentes. La aplicación utiliza Kotlin con Jetpack Compose para la interfaz de usuario, se integra con la API RAWG para obtener información sobre juegos, y utiliza las bibliotecas Room y Retrofit para la gestión de bases de datos y solicitudes HTTP, respectivamente.

Introducción
DinoPass es una aplicación móvil desarrollada para brindar a los usuarios acceso a un catálogo de juegos mediante suscripción mensual. Ofrece tres planes de suscripción diferentes para adaptarse a las necesidades y preferencias de los usuarios. La aplicación se integra con la API RAWG para obtener información detallada sobre los juegos, lo que permite a los usuarios explorar y seleccionar los juegos que desean jugar dentro de su suscripción. Utilizando Kotlin con Jetpack Compose, DinoPass ofrece una interfaz de usuario moderna y atractiva para mejorar la experiencia del usuario.

Propósito y Audiencia
El propósito de DinoPass es proporcionar a los entusiastas de los juegos una forma conveniente y accesible de acceder a una amplia variedad de juegos mediante una suscripción mensual. La aplicación está dirigida a jugadores de todas las edades que buscan descubrir y jugar nuevos títulos sin tener que realizar compras individuales.

Funcionalidades Principales
Exploración de Juegos: Los usuarios pueden explorar el catálogo de juegos disponibles y obtener información detallada sobre cada juego.
Suscripciones: DinoPass ofrece tres planes de suscripción diferentes: DinoBasic, DinoPro y DinoPremium, cada uno con diferentes beneficios y precios.
Búsqueda de Juegos: Los usuarios pueden buscar juegos específicos dentro del catálogo utilizando la función de búsqueda.

Arquitectura
DinoPass sigue una arquitectura de cliente-servidor, donde la aplicación cliente (front-end) está desarrollada utilizando Kotlin con Jetpack Compose y se comunica con un servidor a través de solicitudes HTTP. Los componentes principales de la arquitectura son:

Interfaz de Usuario (UI): Desarrollada con Kotlin y Jetpack Compose.
API RAWG: Utilizada para obtener información sobre juegos.
Base de Datos Local: Implementada con Room para almacenar datos de usuario y configuración de suscripción.
Comunicación con el Servidor: Implementada con Retrofit para realizar solicitudes HTTP a un servidor remoto.

Tecnologías Utilizadas

Lenguaje de Programación: Kotlin.
Biblioteca de Interfaz de Usuario: Jetpack Compose.
Base de Datos: Room.
Cliente HTTP: Retrofit.
API Externa: RAWG.

Consideraciones de Diseño

La aplicación DinoPass se diseñó teniendo en cuenta la usabilidad y la experiencia del usuario. Se hizo especial énfasis en los siguientes principios de diseño:
Simplicidad: La interfaz de usuario se mantiene simple y fácil de usar para permitir a los usuarios navegar sin esfuerzo por la aplicación.
Claridad: Se proporciona información clara y concisa sobre los planes de suscripción y los juegos disponibles para garantizar que los usuarios puedan tomar decisiones informadas.
Retroalimentación Instantánea: Se incorporan elementos interactivos y animaciones para proporcionar retroalimentación instantánea a las acciones del usuario y mejorar la experiencia de navegación.

Conclusiones

En conclusión, DinoPass es una aplicación diseñada para satisfacer las necesidades de los entusiastas de los juegos al proporcionar un acceso fácil y conveniente a una amplia variedad de juegos mediante una suscripción mensual. Con una interfaz de usuario moderna y atractiva, y una integración fluida con la API RAWG, DinoPass ofrece una experiencia de juego sin complicaciones para usuarios de todas las edades.

Referencias

RAWG API Documentation. (s.f.). Recuperado de https://rawg.io/apidocs

Jetpack Compose Codelabs. (s.f.). Recuperado de https://developer.android.com/codelabs/jetpack-compose

Retrofit Documentation. (s.f.). Recuperado de https://square.github.io/retrofit/

Android Developers. (s.f.). Room Persistence Library. Recuperado de https://developer.android.com/topic/libraries/architecture/room