1. Introduction

App Name: Taxi Booking App

Description: The Taxi Booking App allows users to sign up, log in, view and edit their profiles, view a list of available rides, book rides, and manage payments. The app aims to provide a seamless and user-friendly experience for booking taxis.

Purpose: The app was developed to demonstrate the implementation of various Android development concepts, including feature implementation, Material Design, RecyclerView, SharedPreferences for login functionality, and Room Database for CRUD operations.

2. Architecture Overview

Architecture Pattern: MVVM (Model-View-ViewModel)

Components:

Activities:

BookRideActivity: Manages the booking of rides and payment confirmation.

LoginActivity: Handles user login via email/password or phone number with OTP.

ProfileActivity: Displays the user's profile information.

ProfileSetupActivity: Allows users to set up or edit their profile.

RideListActivity: Shows a list of available rides.

SignUpActivity: Manages user registration.

Adapters:

RideAdapter: Binds ride data to the RecyclerView in RideListActivity.

Database:

AppDatabase: The Room database instance.

RideDao: Data Access Object for accessing ride data.

Model:

Ride: Data model representing a ride.

RideDao: Interface for database operations related to Ride.

Main Activity:

MainActivity: The launcher activity providing navigation options to the user.

3. Key Design Decisions

3.1 Use of MVVM Architecture

Reason: MVVM helps in separating the business logic from the UI, making the code more modular, testable, and maintainable.

Implementation: ViewModels were used to manage UI-related data, while the Model layer handles the data logic.

3.2 Room Database for Persistent Storage

Reason: Room provides an abstraction layer over SQLite, enabling more robust database access while harnessing the full power of SQLite.

Implementation: Entities and DAOs were defined to handle ride data, with the AppDatabase class managing the database instance.

3.3 Firebase for Authentication

Reason: Firebase provides a secure and scalable authentication system, supporting various login methods including email/password and phone number OTP.

Implementation: FirebaseAuth was used for managing authentication, with listeners set up for handling login states and errors.

3.4 Material Design for UI

Reason: Material Design provides guidelines for visual, motion, and interaction design, resulting in a cohesive and pleasant user experience.

Implementation: Material components were used throughout the app, including buttons, text fields, and dialogs.

3.5 SharedPreferences for Simple Data Storage

Reason: SharedPreferences is a simple way to store small amounts of primitive data as key-value pairs.

Implementation: Used for storing user login state and basic profile information.

4. Challenges Faced

4.1 Integrating Firebase Authentication

Challenge: Handling different authentication methods and managing the authentication flow.

Solution: Followed Firebase documentation closely and used FirebaseAuth's built-in methods and listeners for seamless integration.

4.2 Implementing RecyclerView

Challenge: Properly displaying a list of rides with different states (available, booked, etc.).

Solution: Created a custom adapter (RideAdapter) to bind ride data to the RecyclerView and handled item clicks to book rides.

4.3 Managing Room Database Migrations

Challenge: Handling database schema changes without losing user data.

Solution: Used Room's migration support to define migration paths and tested thoroughly to ensure data integrity.

4.4 Designing a User-Friendly UI

Challenge: Ensuring the app is intuitive and easy to navigate.

Solution: Followed Material Design guidelines and performed user testing to gather feedback and make improvements.

5. Conclusion

The Taxi Booking App was developed to provide a practical implementation of various Android development concepts. By using MVVM architecture, Room Database, Firebase Authentication, and Material Design, the app ensures a robust, scalable, and user-friendly experience. The challenges faced during development provided valuable learning opportunities, resulting in a well-rounded and functional application.