MayaWay Dating App with Advanced Features

Jadav Madhav Kumar H

Introduction

This document outlines the plan and technologies required to develop an Android dating application. The app will include core dating features such as finding matches and chatting, alongside additional features like hotel booking, cab booking with live location tracking, and real-time post creation and uploading.

1 App Features

The dating app will provide the following functionalities:

- **User Profiles**: Users can create profiles with details like name, bio, photos, and interests.
- **Matchmaking**: Match users based on preferences such as age, location, and interests.
- **Real-time Chat**: Enable real-time messaging between users.
- **Post Uploading**: Users can upload photos and share posts or stories.
- **Hotel Booking Integration**: Allow users to book hotels through third-party APIs.
- **Cab Booking with Live Location**: Integrate with cab services and allow users to share live location.

2 Technology Stack

To develop the app, the following technologies will be used for both the frontend and backend:

2.1 Frontend (Mobile App)

- **Kotlin**: The main programming language for Android development.
- **Jetpack Compose**: UI toolkit to design the app's user interface.
- **Retrofit**: For network calls and API requests.
- **Room Database**: For local storage of user data.
- **WebSockets**: To enable real-time chat functionality.

2.2 Backend (Server)

- **Node.js (Express.js)** or **Django (Python)**: Server-side frameworks for backend development.
- **MongoDB** or **PostgreSQL**: Databases to store user information, match history, posts, etc.
- **Socket.IO**: For real-time messaging and updates.
- **Firebase Cloud Storage**: For storing images and posts uploaded by users.

2.3 Third-Party Integrations

- **Hotel Booking API**: Integration with services like *Booking.com API*, *Expedia API*, or *Airbnb API*.
- **Google Maps SDK**: For live location sharing during cab rides.
- **Uber API**: For cab booking and real-time tracking.

3 Steps to Develop the App

3.1 Step 1: Design and Prototyping

- Use design tools like *Figma* or *Adobe XD* to create the app's UI and user flow.
- Define user interactions such as signing up, creating profiles, finding matches, and real-time chatting.

3.2 Step 2: Frontend Development (Android)

- Build the project using *Android Studio* and *Kotlin*.
- Implement UI components using *Jetpack Compose* or traditional *XML* layouts.
- Integrate Firebase Authentication for user login.

3.3 Step 3: Backend Development

- Develop backend APIs using *Node.js* (Express.js) or *Django*.
- Use databases such as *MongoDB* or *PostgreSQL* to store user data, chat messages, and posts.
- Implement real-time messaging using *Socket.IO* or *Firebase Realtime Database*.

3.4 Step 4: Matchmaking Algorithm

- Develop an algorithm that matches users based on location, interests, and preferences.
- Store matched profiles in the backend and display them on the frontend.

3.5 Step 5: Real-time Chat and Post Uploading

- Implement real-time chat functionality using *Socket.IO* or *Firebase*.
- Integrate *Firebase Cloud Storage* or *Cloudinary* for handling image uploads.

3.6 Step 6: Hotel and Cab Booking Integration

- Integrate *Booking.com API* or similar for hotel booking options.
- Use *Google Maps SDK* and *Uber API* for cab booking and live location tracking.

3.7 Step 7: Testing and Deployment

- Perform unit testing with *JUnit* and test APIs with *Postman*.
- Deploy the backend to platforms like *Heroku* or *AWS*.
- Publish the Android app on the *Google Play Store*.

4 Key Considerations

- **Security**: Use SSL encryption, secure cloud storage, and token-based authentication.
- **Scalability**: Ensure the backend can handle increasing user load using cloud services such as *AWS Lambda* or *Google Cloud*.
- **Data Privacy**: Ensure the app complies with laws such as the *GDPR* for handling user data securely.

UML Class Diagram for User and Hotel Management System