Mobile Apps 2

Assignment Six

Submission Date 24/11/2024

Percent 10%

Include all workings and references

Section A.

1. What are the main issues when considering using any form of BAAS (*3 marks*)

When using Backend-as-a-Service (BaaS), developers should consider the following key issues:

1. Vendor Lock-In

- Relying on a specific BaaS provider can lead to challenges if the provider's services become unavailable, expensive, or insufficient for future needs. Migrating to another platform can be complex and costly due to proprietary APIs and infrastructure dependencies.

2. Data Security and Privacy

- Sensitive user data is often stored on third-party servers, raising concerns about compliance with regulations such as GDPR, CCPA, or HIPAA. It also introduces risks related to data breaches or unauthorized access.

3. Limited Customization and Scalability

- BaaS platforms may restrict customization of backend services, which can limit the application's functionality. Additionally, scaling the application beyond the provider's capabilities or pricing model may pose challenges.

2. Open-Source BaaS has become a viable alternative to commercial software. What are the main considerations when using Open Source generally. (*2 marks*)

Using open-source solutions, including open-source BaaS platforms, involves the following considerations:

1. Licensing and Compliance

- Developers must understand the licensing terms to ensure compliance with redistribution, modification, or commercial use.

2. Support and Maintenance

- Open-source solutions may lack professional support, relying instead on community contributions. Ensuring that the project is actively maintained is critical to avoid using outdated or vulnerable software.

By weighing these factors, organizations can decide whether to adopt commercial or open-source BaaS platforms based on their specific needs and constraints.

Section B.

1. Using whatever commercial or OS BaaS application, implement a means of authentication. You can use an existing app or a very simple example (3.5 *marks*)

2. Implement one of the following in your app
    a. Google Maps
    b. Audio clip
    c. Image Slider
    (*1.5 marks*)

Introduction

This project is developed for the Mobile Apps II course, Assignment 6, which focuses on integrating authentication and API functionalities into a mobile application. The application is inspired by a previous Full Stack / Back-End project, where a web application was built using HTML, CSS, JavaScript, and JSON for querying data. The objective of this project is to build a mobile app that replicates the core features of the web app while meeting the requirements for a mobile platform.

The key technologies used in the mobile app are Google Maps API and Firebase Authentication, ensuring smooth integration of geospatial and user management functionalities. The app retrieves station data from dublinbike.json, originally provided by the Back-End lecturer, and overlays this information onto Google Maps to visualise bike station locations in Dublin.

Project Overview

Objective

The app aims to:

- Authenticate users through Firebase Authentication.
- Display Dublin bike stations on a Google Map using the Google Maps API.
- Allow users to navigate between screens such as login, sign-up, and a list/map of bike stations.

Tools and Technologies

1. Google Maps API: To display bike station locations.
2. Firebase Authentication: For user authentication (sign-up, login, logout).
3. Jetpack Compose: For building the UI components.
4. Android Studio: For development and testing.
5. Kotlin Coroutines: For managing asynchronous tasks, such as Firebase interactions.
6. JSON File: dublinbike.json provides station data, including station names, addresses, and availability status.

Application Features

User Authentication

- Firebase Authentication is used to handle user sign-up and login functionalities.
- Secure password validation is implemented, requiring at least 8 characters, an uppercase letter, a number, and a special symbol.
- A confirmation message is displayed upon successful registration.
- Users are navigated to the login screen after signing up.

Google Maps Integration

- The app integrates Google Maps API to display the bike station locations as markers.
- Upon clicking a marker, users can view detailed station information, such as:
    - Station Name
    - Available Bikes
    - Available Bike Stands
    - Station Status (Open/Closed)

Station List

- A scrollable list of stations is provided for easy navigation.
- Each station card includes details and a button to switch back to the map view.

UI Design

- Jetpack Compose is used for building UI components, ensuring responsiveness and modern design practices.
- Custom dialogs and error messages guide the user during interactions, such as incorrect input during authentication.

Development Process

Architecture

- MVVM Pattern: The app follows the Model-View-ViewModel architecture for separation of concerns and improved maintainability.
- Repository Pattern: Data handling is abstracted for seamless integration of Firebase and JSON file.

Firebase Setup

1. Firebase Console: The project is registered on Firebase Console (https://console.firebase.google.com/).
2. Authentication: Email/Password authentication is enabled.
3. Integration: The Firebase SDK is added to the app, and configuration files (google-services.json) are included.

Google Maps Setup

1. Google Cloud Console: API keys are generated from the Cloud Console (https://console.cloud.google.com/).

JSON Data Integration

- The dublinbike.json file is loaded locally and parsed to display bike station data.
- The file contains:


```
{
  "number": 42,
  "contract_name": "dublin",
  "name": "SMITHFIELD NORTH",
  "address": "Smithfield North",
  "position": {
    "lat": 53.349562,
    "lng": -6.278198
  },
  "banking": false,
  "bonus": false,
  "bike_stands": 30,
  "available_bike_stands": 30,
  "available_bikes": 0,
  "status": "OPEN",
  "last_update": 1729065138000
},
```

Running the Application

Prerequisites

1. Android Studio: Install the latest version.
2. Firebase Configuration: Add your project-specific google-services.json file to the app directory.
3. API Key: Replace the placeholder MAPS_API_KEY in the app with your actual Google Maps API key.
4. Internet Access: Required for Firebase and Google Maps integration.

Steps to Run

1. Clone the repository (Dublin Bike Station) or download the project files.
2. Open the project in Android Studio.
3. Sync Gradle files and resolve dependencies.
4. Run the app on an emulator or physical device.

References

1. Google Maps Platform - Android
2. Firebase Console
3. Google Cloud Console
4. Firebase Authentication with Jetpack Compose
5. Coroutines and Flows with Firebase
6. Building Android Apps with Jetpack Compose and Firebase
7. Dublin Bikes Web App
8. Open Source Initiative. "The Open Source Definition."
9. Karl Fogel. Producing Open Source Software: How to Run a Successful Free Software Project. O'Reilly Media, 2005.
10. What is BaaS?
11. Why Open Source? | Red Hat

Conclusion

This project demonstrates the integration of Firebase Authentication and Google Maps API in a mobile app. It successfully adapts a web application into a mobile app, adhering to modern development standards. The app provides an intuitive user experience, ensuring functionality for both authentication and geospatial data visualization.