

## **MAD PWA Lab Exp 6**

### **Aim: To Set Up Firebase with Flutter for iOS and Android Apps**

#### **Theory:**

Firebase is a powerful platform provided by Google that offers a variety of services for mobile and web applications, including real-time databases, authentication, cloud functions, and more. When combined with Flutter, a popular open-source UI software development toolkit, developers can create cross-platform apps for iOS and Android efficiently. This guide will walk you through the process of setting up Firebase with Flutter for both iOS and Android platforms.

#### **Prerequisites**

Before you begin, make sure you have the following prerequisites:

Flutter Installed: Ensure that you have Flutter and Dart installed on your development machine.

Firebase Account: Create a Firebase account if you don't have one already. Visit the Firebase Console to set up a new project.

Flutter IDE: Use an integrated development environment (IDE) like Visual Studio Code or Android Studio with Flutter and Dart plugins installed.

#### **Step 1: Create a Firebase Project**

- 1.1. Go to the Firebase Console, click on "Add Project," and follow the prompts to create a new project.
- 1.2. Once the project is created, click on "Add App" to add both iOS and Android apps to your project.

#### **Step 2: Configure iOS App**

- 2.1. For iOS, click on the iOS icon in the Firebase Console and follow the setup instructions. Download the GoogleService-Info.plist file and add it to your Flutter project's ios/Runner directory.
- 2.2. Update your ios/Podfile to include the necessary Firebase dependencies. Run pod install in the ios directory.

#### **Step 3: Configure Android App**

- 3.1. For Android, click on the Android icon in the Firebase Console and follow the setup instructions. Download the google-services.json file and add it to your Flutter project's android/app directory.

3.2. Update your android/build.gradle and android/app/build.gradle files with the necessary dependencies.

## **Step 4: Add FlutterFire Plugins**

4.1. Open your pubspec.yaml file and add the necessary FlutterFire plugins for the Firebase services you want to use, such as authentication, Firestore, or Cloud Messaging.

4.2. Run flutter pub get in your terminal to fetch the dependencies.

## **Step 5: Initialize Firebase in Your Flutter App**

5.1. Import the Firebase packages in your Dart code and initialize Firebase in the main.dart file.

## **Step 6: Test Firebase Integration**

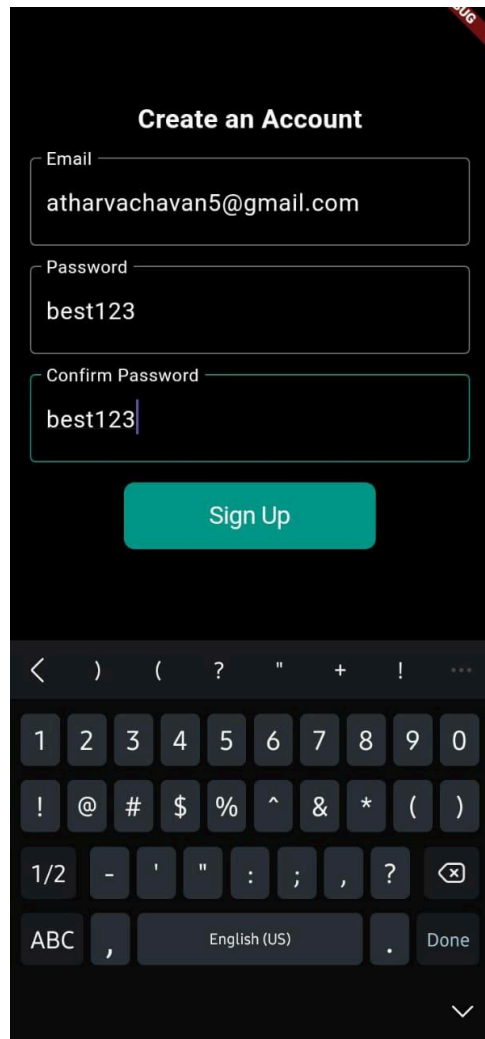
6.1. Write a simple test to ensure Firebase is correctly integrated. For example, try initializing Firestore or authenticate a user.

## **Code & Implementation:**

```
//main.dart
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:jio_saavn_auth/firebase_options.dart';
import 'package:jio_saavn_auth/screens/library_screen.dart';
import 'package:jio_saavn_auth/screens/welcome_screen.dart';
import 'package:provider/provider.dart';
```

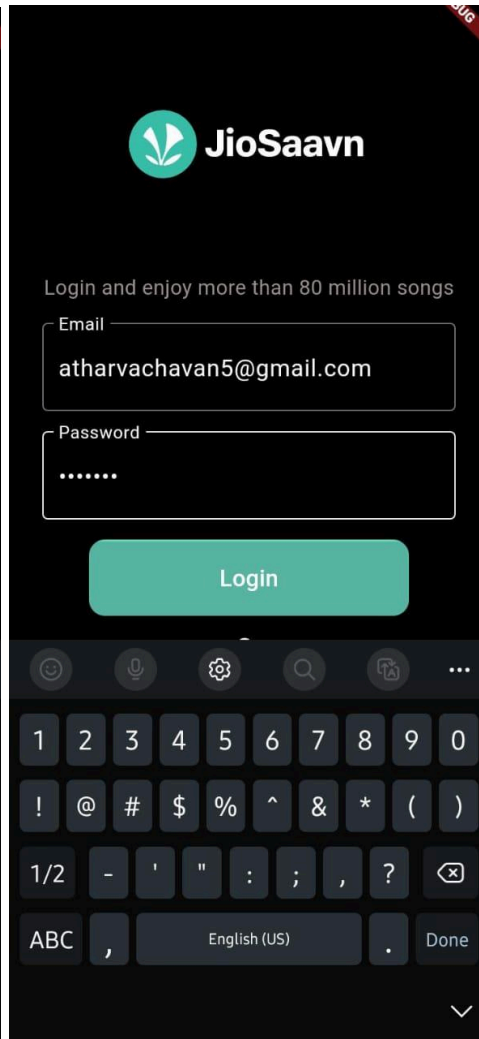
```
Future<void> main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );
  runApp(
    ChangeNotifierProvider(
      create: (context) => LikedSongsModel(),
      child: MyApp(),
    ),
  );
}
```

}



The Sign Up screen features a dark background with a red 'bug' label in the top right corner. The title 'Create an Account' is centered at the top. Below it are three input fields: 'Email' containing 'atharvachavan5@gmail.com', 'Password' containing 'best123', and 'Confirm Password' containing 'best123'. A teal 'Sign Up' button is positioned below the fields. At the bottom, a standard iOS keyboard is visible.

Sign Up Screen



The Login screen has a dark background with a red 'bug' label in the top right corner. The JioSaavn logo is at the top, followed by the text 'Login and enjoy more than 80 million songs'. Below this are two input fields: 'Email' containing 'atharvachavan5@gmail.com' and 'Password' containing six dots. A teal 'Login' button is centered below the fields. At the bottom, a standard iOS keyboard is visible.

Login screen

Phone authentication ▾

Authentication

Users Sign-in method Templates Usage Settings Extensions

Add user

Identifier	Providers	Created ↓	Signed In	User UID
atharvachavan5@gmail...		Feb 20, 2024	Feb 20, 2024	XoD3rijOVAaH1S07xJfd64gGr...
atharvachavan4@gmail...		Feb 17, 2024	Feb 17, 2024	tT6N0f7zLjPgvziaYQffRXrkOpz1
atharvachavan3@gmail...		Feb 17, 2024	Feb 17, 2024	IICZuBuFfZXlxGd8YcfQSEqWC...
atharvachavan2@gmail...		Feb 17, 2024	Feb 17, 2024	Utl4QSk5NAdyq1IZchnlPsF47...
atharvachavan123@gm...		Feb 17, 2024	Feb 20, 2024	aEyv8QylkwhXfXi9zpXu9FfaT...

Rows per page: 50 1 – 5 of 5 < >

## Conclusion:

In conclusion, integrating Firebase with Flutter for iOS and Android apps provides a robust foundation for building feature-rich, cross-platform applications. By following the steps outlined in this guide, you have established a seamless connection between your Flutter project and Firebase, unlocking a plethora of services such as authentication, real-time databases, and cloud functions. This integration not only enhances the app's functionality but also simplifies the management of backend services, making development more efficient.