Experiment No:- 06

Name:- Mayank Pahuja **Class:- D15-A RollNo:- 40**

Aim: To Connect Flutter UI with Firebase

Theory:

FlutterFire is a set of Flutter plugins that enable Flutter developers to integrate their applications with various Firebase services. Firebase is a comprehensive mobile and web application development platform provided by Google. FlutterFire is specifically designed to provide Flutter developers with a seamless way to interact with Firebase services.

Key features of FlutterFire include:

1. **Firebase Authentication:** FlutterFire provides plugins to easily integrate Firebase Authentication, allowing developers to implement user sign-up, sign-in, and password recovery features in their Flutter applications. Firebase supports various authentication methods, including email/password, Google Sign-In, Facebook Sign-In, and more.
2. **Cloud Firestore and Realtime Database:** FlutterFire supports both Cloud Firestore and Firebase Realtime Database, enabling developers to store and retrieve data in real-time. Firestore is a NoSQL document database, while Realtime Database is a JSON-based database.
3. **Cloud Functions:** Developers can deploy serverless functions using Cloud Functions for Firebase, and FlutterFire allows Flutter apps to trigger and interact with these functions.
4. **Cloud Storage:** FlutterFire supports Firebase Cloud Storage, allowing developers to upload, download, and manage files in the cloud. This is useful for handling user-generated content, such as images or videos.
5. **Firebase Cloud Messaging (FCM):** FCM enables developers to send push notifications to their Flutter applications. FlutterFire provides plugins for integrating FCM and handling push notifications.
6. **Firebase Performance Monitoring:** Developers can monitor the performance of their Flutter applications using Firebase Performance Monitoring. This includes measuring app startup time, screen rendering, and network performance.
7. **Firebase Analytics:** FlutterFire includes plugins for integrating Firebase Analytics, enabling developers to gain insights into user behavior and app usage.
8. **Firebase Remote Config:** FlutterFire supports Firebase Remote Config, allowing developers to remotely configure app behavior without publishing updates. This is useful for A/B testing and feature toggling.
9. **Firebase Crashlytics:** FlutterFire includes support for Firebase Crashlytics, providing real-time crash reporting to help developers identify and fix issues quickly.
10. **Firebase AdMob:** FlutterFire includes AdMob plugins for integrating advertisements into Flutter applications using Firebase AdMob.

Code & Implementation:-

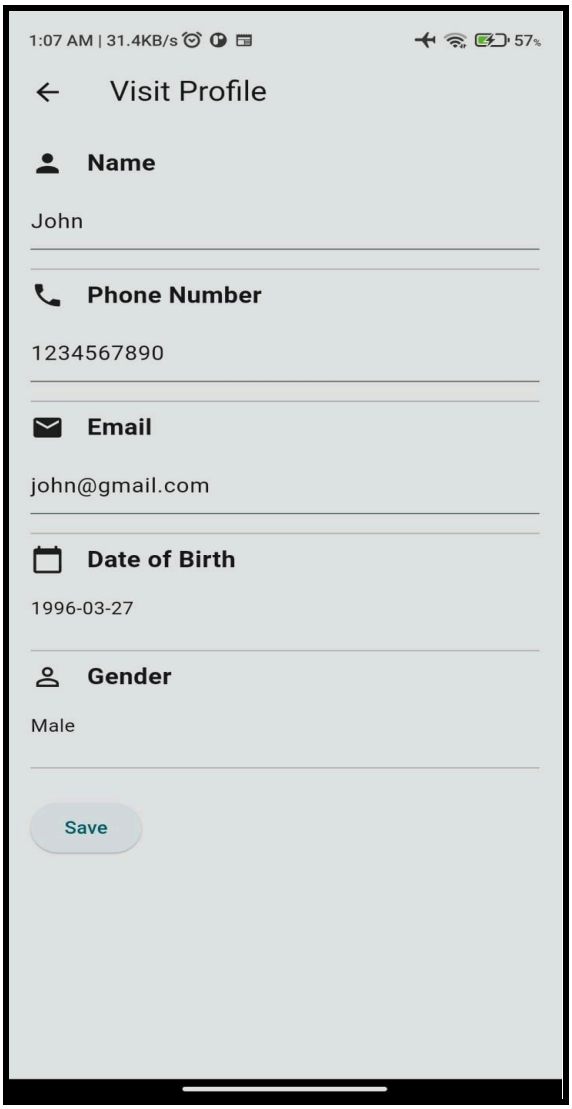
Main.dart file for connecting the firebase

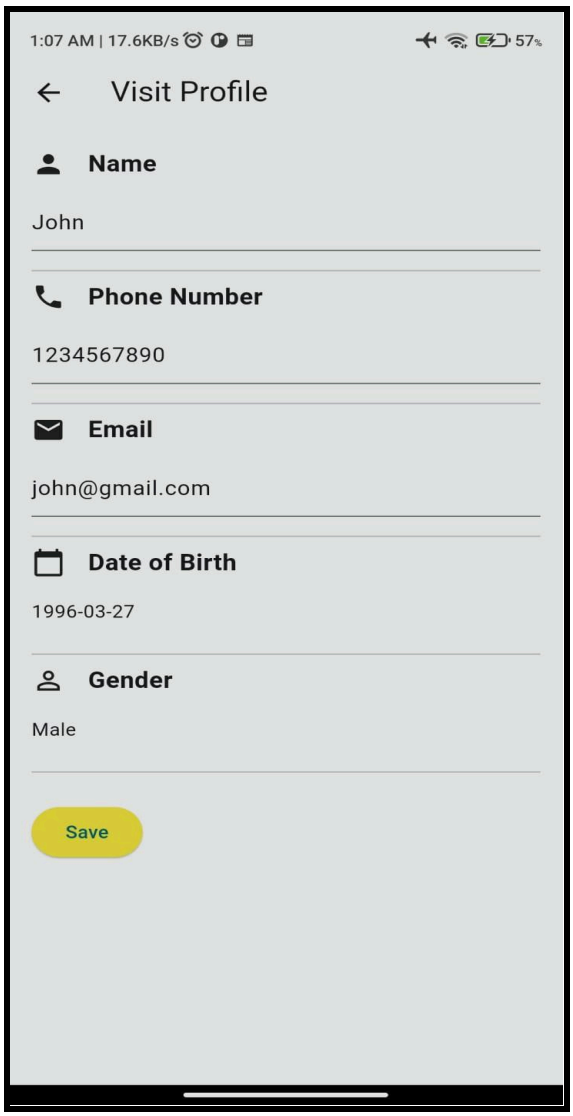
```
import 'dart:io';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import
'package:get/get_navigation/src/root/get_material_ap
p.dart';
import 'package:rapido/screens/about_screen.dart';
import 'package:rapido/screens/history_screen.dart';
import 'package:rapido/screens/splash_screen.dart';
import
'package:rapido/screens/visit_profile_screen.dart';
import 'package:rapido/wrapper.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  Platform.isAndroid
      ? await Firebase.initializeApp(
          options: const FirebaseOptions(
            apiKey:
"AIzaSyDjMB0YuZP3SiJRxpsxy1pBGyBW-OxuNX
k",
            appId:
"1:179713960522:android:391199b92da9ce114fd4b3
",
            messagingSenderId: "179713960522",
            projectId: "rapido-clone-78a08",
            storageBucket:
"rapido-clone-78a08.appspot.com",
          ),
        )
      : await Firebase.initializeApp();
  runApp(const MyApp());
}
```

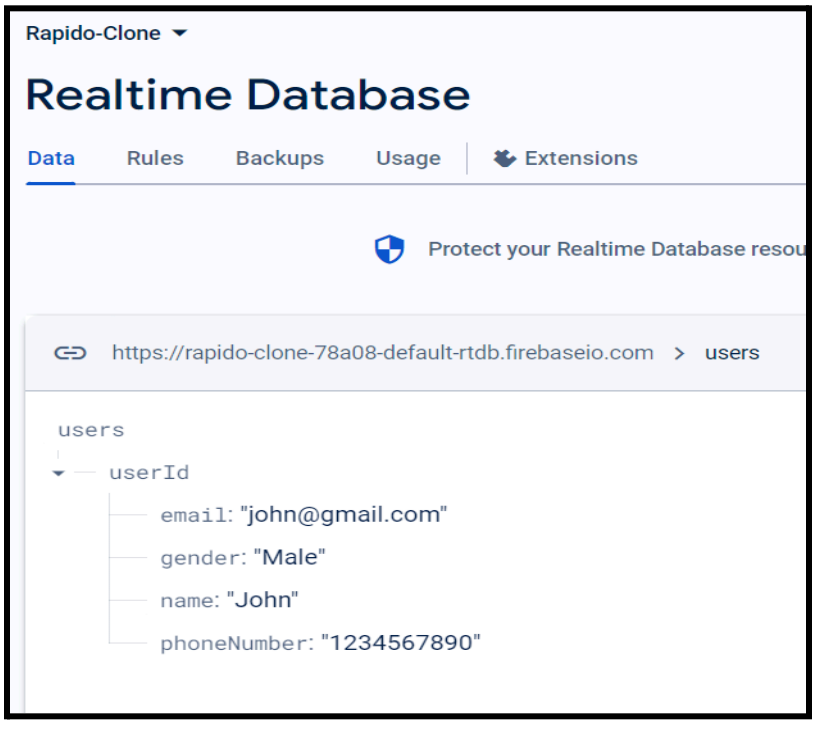
Store user data from visited_profile.dart

```
import
'package:firebase_database/firebase_database.dart';
import 'package:flutter/material.dart';

class VisitProfileScreen extends StatefulWidget {
  const VisitProfileScreen({Key? key}) : super(key:
key);
  @override
  _VisitProfileScreenState createState() =>
_VisitProfileScreenState();
}
class _VisitProfileScreenState extends
State<VisitProfileScreen> {
  TextEditingController nameController =
TextEditingController();
  TextEditingController phoneNumberController =
TextEditingController();
  TextEditingController emailController =
TextEditingController();

  String name = '';
  String phoneNumber = '';
  String email = '';
  String gender = ''; // Default gender
  DateTime? dob; // Date of Birth
  bool isDataSaved = false; // To track if data is saved

  final DatabaseReference _userRef =

FirebaseDatabase.instance.reference().child('users');

  @override
  void initState() {
    super.initState();
    // Set initial values to controllers
    nameController.text = name;
    phoneNumberController.text = phoneNumber;
    emailController.text = email;
  }
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Visit Profile'),
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          crossAxisAlignment:
CrossAxisAlignment.start,
          children: [
            _buildProfileItem(Icons.person, 'Name',
nameController),
```

```
          _buildProfileItem(Icons.phone, 'Phone
Number', phoneNumberController),
          _buildProfileItem(Icons.email, 'Email',
emailController),
          _buildProfileItem(
              Icons.calendar_today,
              'Date of Birth',
              null,
              text: dob != null ?
dob.toString().substring(0, 10) : 'Select Date',
              onTap: () => _selectDate(context),
            ),

          _buildProfileItem(
            Icons.person_outline,
            'Gender',
            null,
            text: gender.isNotEmpty ? gender : 'Select
Gender',
            onTap: () => _showGenderDialog(context),
          ),

          SizedBox(height: 16),
          ElevatedButton(
            onPressed: () {
              _saveUserData();
            },
            style: ElevatedButton.styleFrom(
              backgroundColor: isDataSaved ?
Colors.yellow : null,
            ),
            child: Text('Save'),
          ),
        ],
      ),
    ),
  );
}
void _updateUserInfo(String fieldName, String
newValue) {
  setState(() {
    // Update local state based on field name
    switch (fieldName) {
      case 'Name':
        name = newValue;
        break;
      case 'Phone Number':
        phoneNumber = newValue;
        break;
      case 'Email':
        email = newValue;
        break;
      case 'Gender':
        gender = newValue;
        break;
      default:
        break;
    }
  });
}

void _saveUserData() {
  _userRef.child('userId').set({
    'name': name,
    'phoneNumber': phoneNumber,
    'email': email,
    'gender': gender,
    // Add other fields as needed
  }).then((_) {
    setState(() {
      isDataSaved = true;
    });
  });
}
```


1:07 AM | 31.4KB/s
57%
Visit Profile
Name
John
Phone Number
1234567890
Email
john@gmail.com
Date of Birth
1996-03-27
Gender
Male
Save


1:07 AM | 17.6KB/s
57%
Visit Profile
Name
John
Phone Number
1234567890
Email
john@gmail.com
Date of Birth
1996-03-27
Gender
Male
Save


Rapido-Clone
Realtime Database
Data
Rules
Backups
Usage
Extensions
Protect your Realtime Database resou
https://rapido-clone-78a08-default-rtdb.firebaseio.com > users
users
userId
email: "john@gmail.com"
gender: "Male"
name: "John"
phoneNumber: "1234567890"