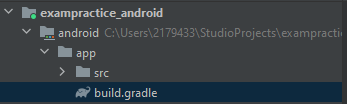
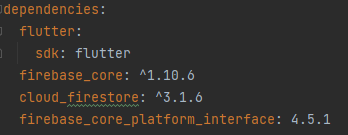
Format: CTRL + ALT + L

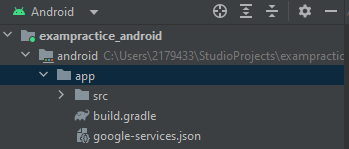
Android {

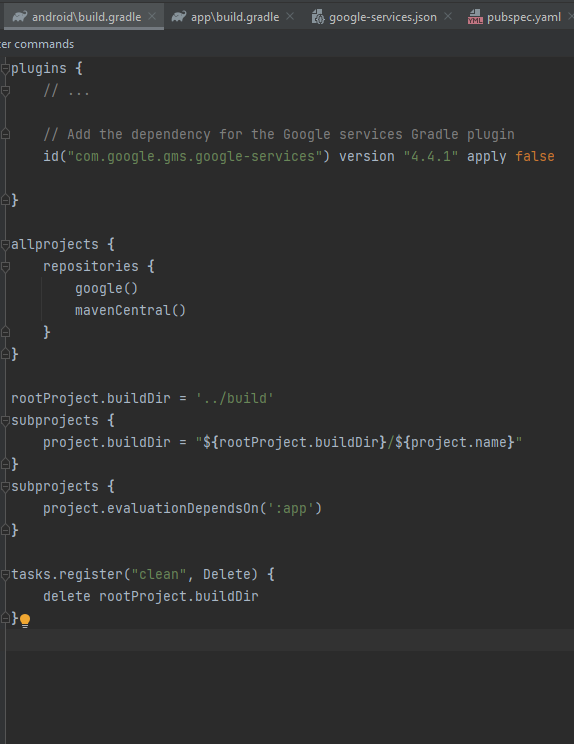
 Namespace “HERE”

Dependencies

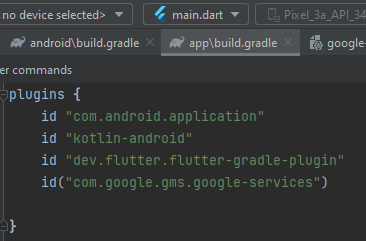


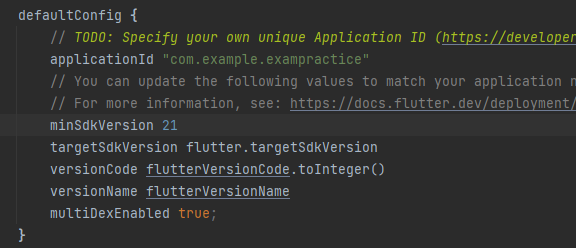
JSON:



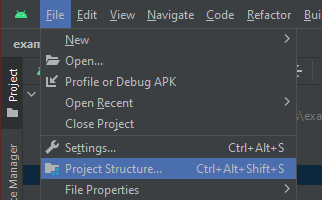


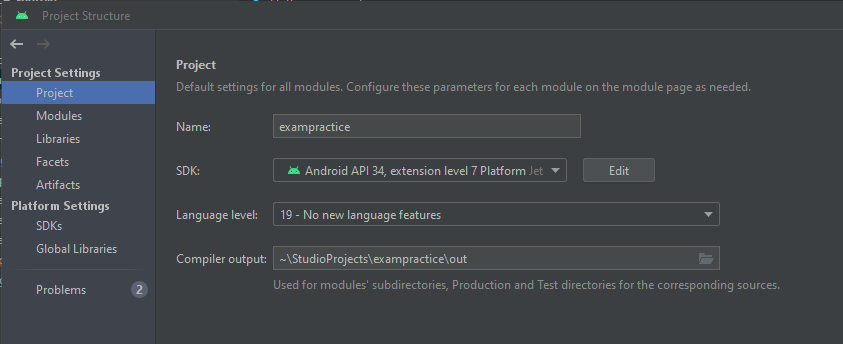
App level

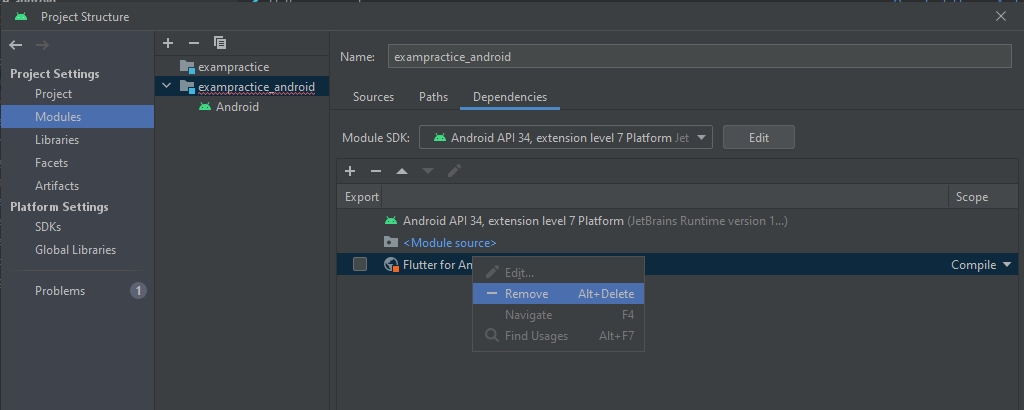




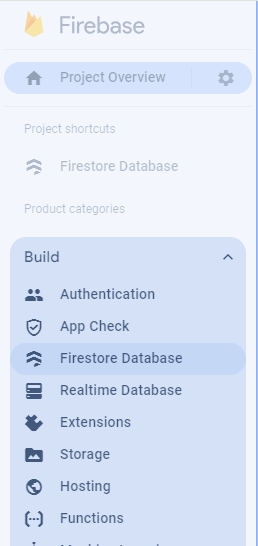


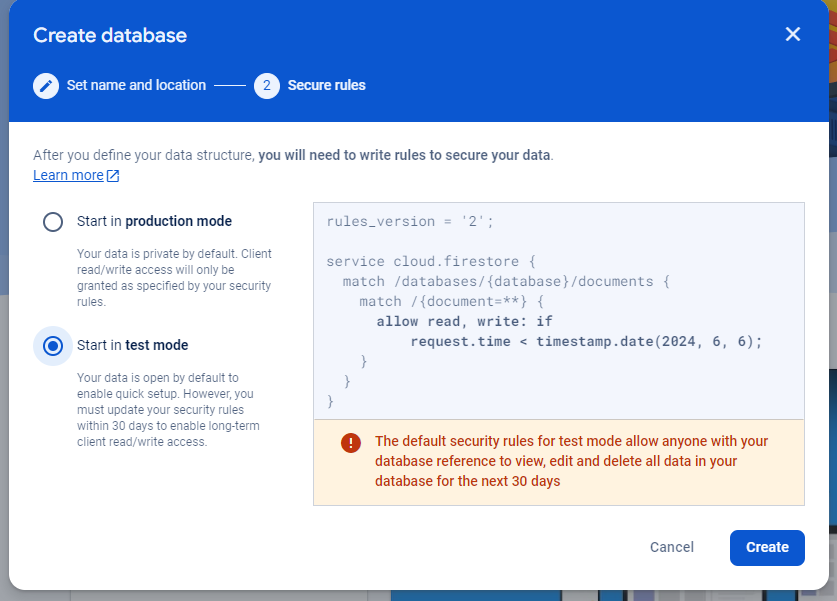






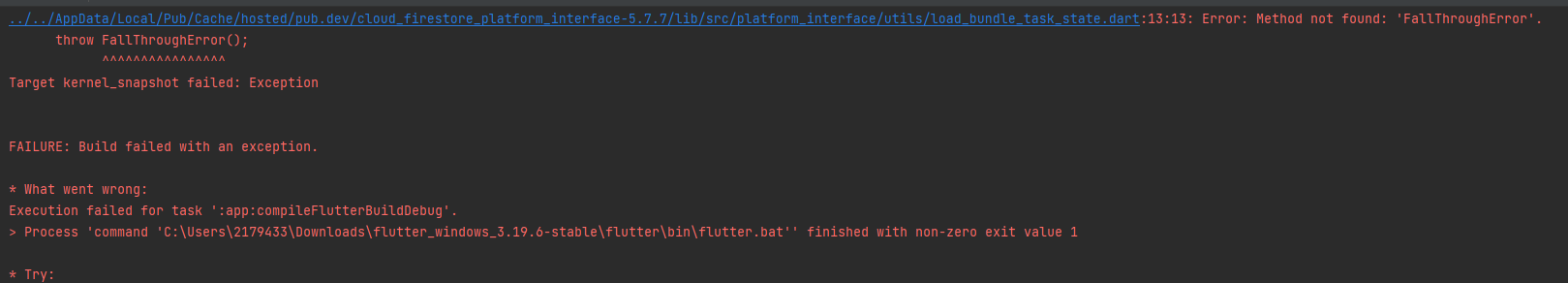
GO TO FIRESTORE IN WEBSITE

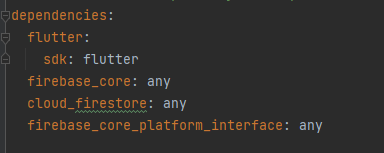


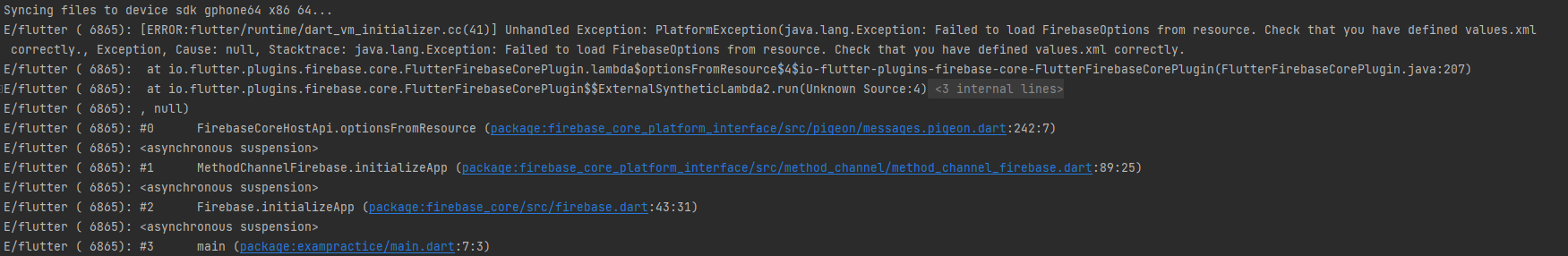


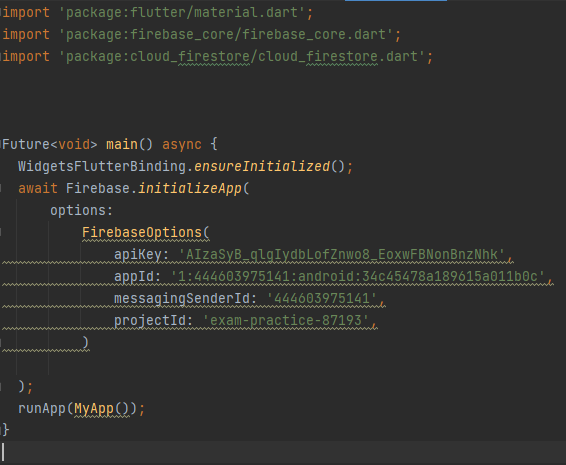
Possible errors:

FallThroughError();

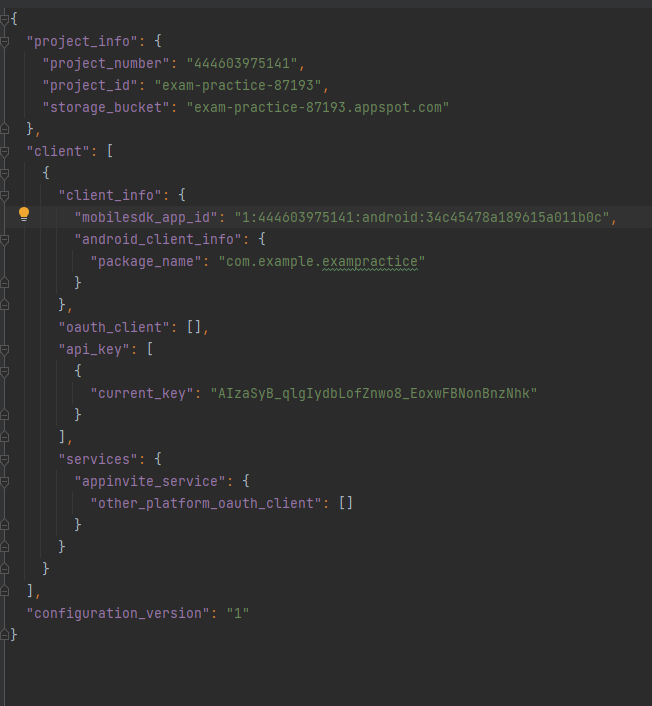








apiKey =



MAIN.DART FOR FIREBASE

import 'package:flutter/material.dart';  
import 'package:firebase\_core/firebase\_core.dart';  
import 'package:cloud\_firestore/cloud\_firestore.dart';  
  
  
  
Future<void> main() async {  
 WidgetsFlutterBinding.*ensureInitialized*();  
 await Firebase.*initializeApp*(  
 options:  
 FirebaseOptions(  
 apiKey: 'AIzaSyB\_qlgIydbLofZnwo8\_EoxwFBNonBnzNhk',  
 appId: '1:444603975141:android:34c45478a189615a011b0c',  
 messagingSenderId: '444603975141',  
 projectId: 'exam-practice-87193',  
 )  
  
 );  
 runApp(MyApp());  
}  
  
  
  
class MyApp extends StatelessWidget {  
 const MyApp({super.key});  
  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 home: MyHomePage(),  
 );  
 }  
}  
  
class MyHomePage extends StatefulWidget {  
 const MyHomePage({super.key});  
  
 @override  
 State<MyHomePage> createState() => \_MyHomePageState();  
}  
  
class \_MyHomePageState extends State<MyHomePage> {  
 final Stream<QuerySnapshot> \_userStream =  
 FirebaseFirestore.*instance*.collection('Users').snapshots();  
  
 CollectionReference users = FirebaseFirestore.*instance*.collection('Users');  
 String name = ' ';  
  
 Future<void> addUser() {  
 return users  
 .add({  
 'name': name,  
 })  
 .then((value) => print("User added"))  
 .catchError((error) => print("Failed to add the user"));  
 }  
  
 Future<void> updateUser(String id, String newName) async {  
 await users  
 .doc(id)  
 .update({'name': newName})  
 .then((value) => print("User updated"))  
 .catchError((error) => print("Failed to update"));  
 }  
  
 Future<void> deleteUser(String id) async {  
 await users  
 .doc(id)  
 .delete()  
 .then((value) => print("User deleted"))  
 .catchError((error) => print("Failed to delete"));  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: AppBar(  
 title: Text('Firestore Demo'),  
 ),  
 body: SingleChildScrollView(  
 child: Container(  
 margin: EdgeInsets.*zero*,  
 padding: EdgeInsets.all(16.0),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.start,  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Text(  
 'Enter user Name',  
 style: TextStyle(fontSize: 16),  
 ),  
 SizedBox(  
 height: 10,  
 ),  
 TextField(  
 onChanged: (value) {  
 name = value;  
 },  
 decoration: InputDecoration(  
 hintText: 'Enter user name',  
 contentPadding: EdgeInsets.symmetric(  
 vertical: 10.0, horizontal: 20.0),  
 border: OutlineInputBorder(  
 borderRadius: BorderRadius.all(Radius.circular(6.0)),  
 ),  
 ),  
 ),  
 SizedBox(  
 height: 10,  
 ),  
 ElevatedButton(  
 onPressed: () {  
 addUser();  
 },  
 child: Text('Add User'),  
 ),  
 SizedBox(  
 height: 20.0,  
 ),  
 StreamBuilder<QuerySnapshot>(  
 stream: \_userStream,  
 builder: (BuildContext context,  
 AsyncSnapshot<QuerySnapshot> snapshot) {  
 if (snapshot.hasError) {  
 return Text('Something went wrong');  
 }  
 if (snapshot.connectionState == ConnectionState.waiting) {  
 return CircularProgressIndicator(); // Show loading indicator  
 }  
 return ListView.builder(  
 shrinkWrap: true,  
 itemCount: snapshot.data!.docs.length,  
 itemBuilder: (BuildContext context, int index) {  
 Map<String, dynamic> data = snapshot.data!.docs[index]  
 .data() as Map<String, dynamic>;  
 String docId = snapshot.data!.docs[index].id;  
 return ListTile(  
 title: Text(data['name']),  
 trailing: Row(  
 mainAxisSize: MainAxisSize.min,  
 children: [  
 IconButton(  
 icon: Icon(Icons.*edit*),  
 onPressed: () {  
 showDialog(  
 context: context,  
 builder: (BuildContext context) {  
 String updatedName = data['name'];  
 return AlertDialog(  
 title: Text('Edit User'),  
 content: TextField(  
 onChanged: (value) {  
 updatedName = value;  
 },  
 decoration: InputDecoration(  
 hintText: 'Enter new name',  
 ),  
 ),  
 actions: [  
 TextButton(  
 onPressed: () {  
 Navigator.*pop*(context);  
 },  
 child: Text('Cancel'),  
 ),  
 TextButton(  
 onPressed: () {  
 updateUser(docId, updatedName);  
 Navigator.*pop*(context);  
 },  
 child: Text('Update'),  
 ),  
 ],  
 );  
 },  
 );  
 },  
 ),  
 IconButton(  
 icon: Icon(Icons.*delete*),  
 onPressed: () {  
 deleteUser(docId);  
 },  
 ),  
 ],  
 ),  
 );  
 },  
 );  
 },  
 ),  
 ],  
 ),  
 ),  
 ),  
 );  
 }  
}