**BÁO CÁO BÀI TẬP GIỮA KỲ**

|  |  |
| --- | --- |
| **Họ và tên** | Trịnh Nguyễn Nhật Hiếu |
| **Mã SV** | 21IT274 |
| **Lớp HP** | Phát triển ứng dụng đa nền tảng (6) |

1. **Firebase Console tạo project Firebase**

**A screenshot of a computer

Description automatically generated**

1. **Tạo project Flutter**

**A screenshot of a computer program

Description automatically generated**

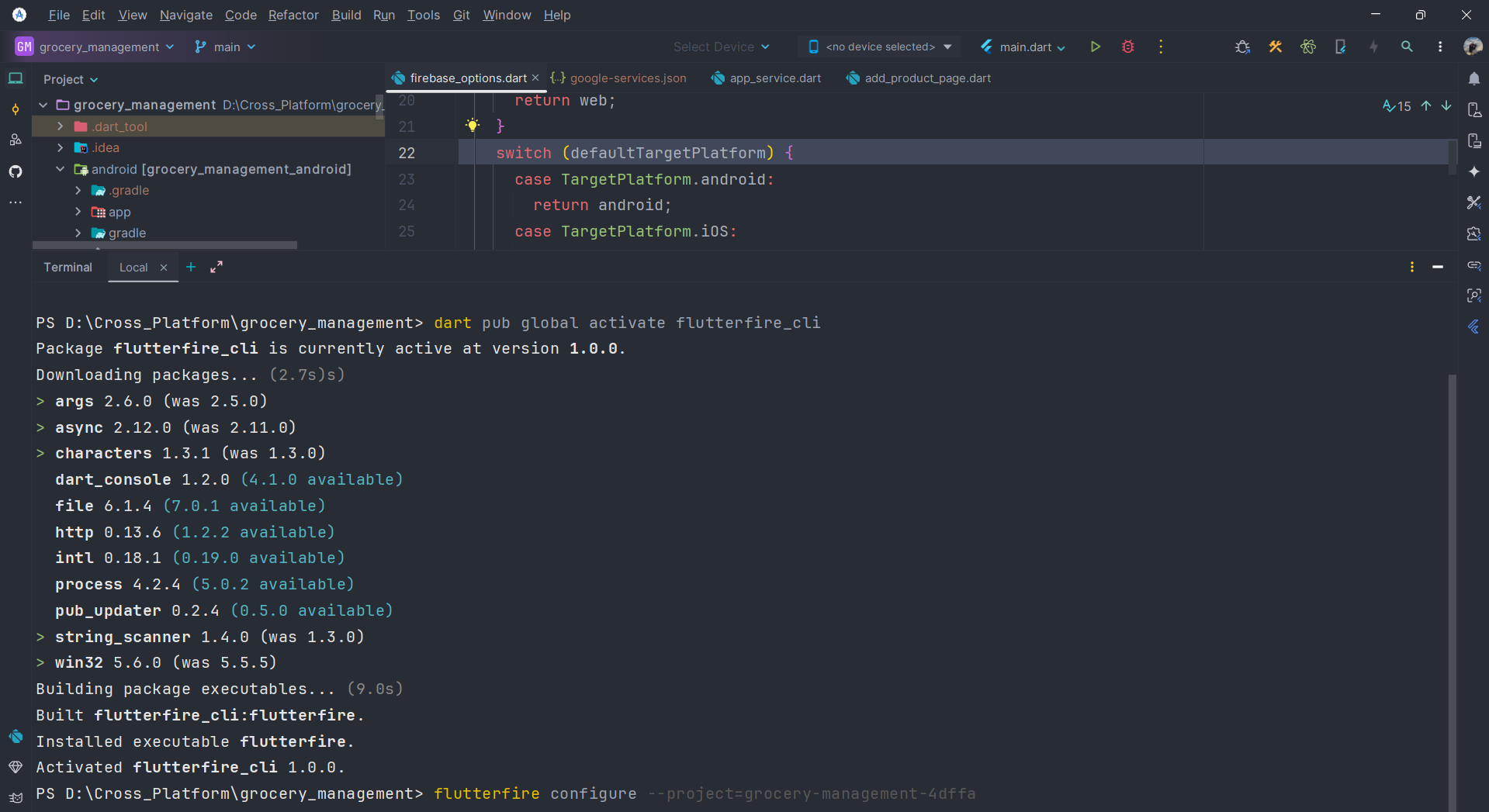
1. **Kết nối project flutter với firebase:**

Mở terminal, copy và dán các câu lệnh sau:

* **dart pub global activate flutterfire\_cli**
* **flutterfire configure --project= “grocery-management-4dffa......” (Thay thế “...” là tên dự án flutter tạo trên firebase).**

**A screenshot of a computer

Description automatically generated**

****

**Firebase Authentication**

**A screenshot of a computer

Description automatically generated**

**Firestore Database**

**A screenshot of a computer

Description automatically generated**

**Storage**

**A screenshot of a computer

Description automatically generated**

1. **Trong project flutter, vào pubspec.yaml, thêm các thư viện sau**A screenshot of a computer program

   Description automatically generated
2. **Trong lib, tạo cấu trúc dự án như sau:**A screenshot of a computer

   Description automatically generated
3. **Tiến hành triển khai, xây dựng app:**
4. **File main.dart**

Future<void> main() async {  
 WidgetsFlutterBinding.ensureInitialized();  
 await Firebase.initializeApp(options: DefaultFirebaseOptions.currentPlatform);  
 runApp(const MyApp());  
}  
  
class MyApp extends StatelessWidget {  
 const MyApp({super.key});  
  
 @override  
 Widget build(BuildContext context) {  
 return const MaterialApp(  
 debugShowCheckedModeBanner: false,  
 title: "My Grocery",  
 home: AuthGate(),  
 );  
 }  
}

1. **File services/app\_service.dart**

class AppService {  
 final FirebaseAuth \_auth = FirebaseAuth.instance;  
 final CollectionReference \_firestore =  
 FirebaseFirestore.instance.collection("products");  
 final FirebaseStorage \_storage = FirebaseStorage.instance;  
 final ImagePicker \_picker = ImagePicker();  
  
 // Get product stream  
 Stream<List<Product>> getProductStream({String searchQuery = ""}) {  
 return \_firestore  
 .where('userId', isEqualTo: \_auth.currentUser?.uid)  
 .snapshots()  
 .map(  
 (snapshot) {  
 return snapshot.docs  
 .map(  
 (doc) {  
 final productData = doc.data() as Map<String, dynamic>;  
 return Product(  
 id: doc.id,  
 name: productData['name'],  
 category: productData['category'],  
 price: productData['price'],  
 imageUrl: productData['imageUrl'],  
 userId: \_auth.currentUser!.uid,  
 );  
 },  
 )  
 .where((product) =>  
 searchQuery.isEmpty ||  
 product.name.toLowerCase().contains(  
 searchQuery.toLowerCase(),  
 ))  
 .toList();  
 },  
 );  
 }  
  
 // Upload Image  
 Future<String?> uploadImage(File image) async {  
 try {  
 String userId = \_auth.currentUser!.uid;  
 String filePath =  
 "products/$userId/${DateTime.now().millisecondsSinceEpoch}.png";  
 // tham chieu duong dan file toi producs  
 Reference reference = \_storage.ref().child(filePath);  
 // upload image  
 UploadTask uploadTask = reference.putFile(image);  
 await uploadTask.whenComplete(() => null);  
 String imageUrl = await reference.getDownloadURL();  
 return imageUrl;  
 } on FirebaseException catch (e) {  
 throw Exception(e.code);  
 }  
 }  
  
 // Delete Image  
 Future<void> deleteImage(String imageUrl) async {  
 try {  
 Reference reference = \_storage.refFromURL(imageUrl);  
 await reference.delete();  
 } on FirebaseException catch (e) {  
 throw Exception(e.code);  
 }  
 }  
  
 // Add product  
 Future<void> addProduct(  
 String name, String category, String price, File? image) async {  
 String? imageUrl;  
  
 if (image != null) {  
 imageUrl = await uploadImage(image);  
 }  
  
 await \_firestore.add({  
 'name': name,  
 'category': category,  
 'price': price,  
 'userId': \_auth.currentUser?.uid.toString(),  
 'imageUrl': imageUrl ?? "",  
 });  
 }  
  
// Update product  
 Future<void> updateProduct(String id, String name, String category,  
 String price, String imageUrl, File? image) async {  
 String? newImageUrl;  
  
 if (image != null) {  
 await deleteImage(imageUrl);  
 newImageUrl = await uploadImage(image);  
 } else {  
 newImageUrl = imageUrl;  
 }  
 await \_firestore.doc(id).update(  
 {  
 'name': name,  
 'category': category,  
 'price': price,  
 'imageUrl': newImageUrl,  
 },  
 );  
 }  
  
 //Delete product  
 Future<void> deleteProduct(String id, String imageUrl) async {  
 await deleteImage(imageUrl);  
 await \_firestore.doc(id).delete();  
 }  
  
 // Get image gallery  
 Future<File?> pickImageFromGallery() async {  
 final pickedFile = await \_picker.pickImage(  
 source: ImageSource.gallery,  
 imageQuality: 80,  
 );  
 return pickedFile != null ? File(pickedFile.path) : null;  
 }  
}

1. **File services/auth\_gate.dart**

class AuthGate extends StatelessWidget {  
 const AuthGate({super.key});  
  
 @override  
 Widget build(BuildContext context) {  
 return Scaffold(  
 body: StreamBuilder(  
 stream: FirebaseAuth.instance.authStateChanges(),  
 builder: (context, snapshot) {  
 if (snapshot.hasData) {  
 return const HomePage();  
 } else {  
 return const LoginOrRegister();  
 }  
 },  
 ),  
 );  
 }  
}

1. **File services/auth\_service.dart**

class AuthService {  
 final FirebaseAuth \_auth = FirebaseAuth.instance;  
  
 // Get current user  
 User? getCurrentUser() {  
 return \_auth.currentUser;  
 }  
  
//sign in  
 Future<UserCredential> signInWithEmailPassword(  
 String email, String password) async {  
 try {  
 UserCredential userCredential = await \_auth.signInWithEmailAndPassword(  
 email: email,  
 password: password,  
 );  
 return userCredential;  
 } on FirebaseAuthException catch (e) {  
 throw Exception(e.code);  
 }  
 }  
  
 // sign up  
 Future<UserCredential> signUpWithEmailPassword(  
 String email, String password) async {  
 try {  
 UserCredential userCredential =  
 await \_auth.createUserWithEmailAndPassword(  
 email: email,  
 password: password,  
 );  
 return userCredential;  
 } on FirebaseAuthException catch (e) {  
 throw Exception(e.code);  
 }  
 }  
  
 // sign out  
 Future<void> signOut() async {  
 return await \_auth.signOut();  
 }  
}

1. **File services/login\_or\_register.dart**

class LoginOrRegister extends StatefulWidget {  
 const LoginOrRegister({super.key});  
  
 @override  
 State<LoginOrRegister> createState() => \_LoginOrRegisterState();  
}  
  
class \_LoginOrRegisterState extends State<LoginOrRegister> {  
 bool showLoginPage = true;  
  
 void togglePages() {  
 setState(() {  
 showLoginPage = !showLoginPage;  
 });  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 if (showLoginPage) {  
 return LoginPage(onTap: togglePages);  
 } else {  
 return RegisterPage(onTap: togglePages);  
 }  
 }  
}

**Source code:** [**tnnHiu/flutter\_crud\_firebase (github.com)**](https://github.com/tnnHiu/flutter_crud_firebase)

1. **Kết quả chạy chương trình:**

**Giao diện đăng nhập:**

**A screenshot of a login screen

Description automatically generated**

**Giao diện đăng ký  
A screenshot of a login screen

Description automatically generated**

**Sau khi đăng ký, thông tin sẽ được lưu trên firebase Auth**

**A screenshot of a computer

Description automatically generated**

**Giao diện Thêm sản phẩm  
A screenshot of a phone

Description automatically generated**

**Giao diện trang chủ  
A screenshot of a phone

Description automatically generated**

Dữ liệu trên firebase firestore sau khi thêm sản phẩm

A screenshot of a computer

Description automatically generated

Dữ liệu trên firebase storage sau khi thêm sản phẩm

A screenshot of a computer

Description automatically generated