배달 앱(패캠마트)

6 프로젝트 돌아보기

개요 및 학습 목표
돌아보기

Flutter와 Firebase를 활용한 앱을 개발한다.

Firebase 콘솔 설정 방법을 학습한다.
Flutter 프로젝트에 Firebase 프로젝트를 추가하는 방법을 학습한다.
> Firebase와 Flutter 프로젝트 설정방법을 학습한다.

Firebase의 Auth 기능을 활용하여 **로그인, 회원가입 기능**을 학습한다.
Firebase 서비스의 기능을 활용하여 데이터를 저장하고 읽는 과정을 학습한다.
이미지를 Firebase의 데이터베이스에 저장하고 읽는 방법을 학습한다.

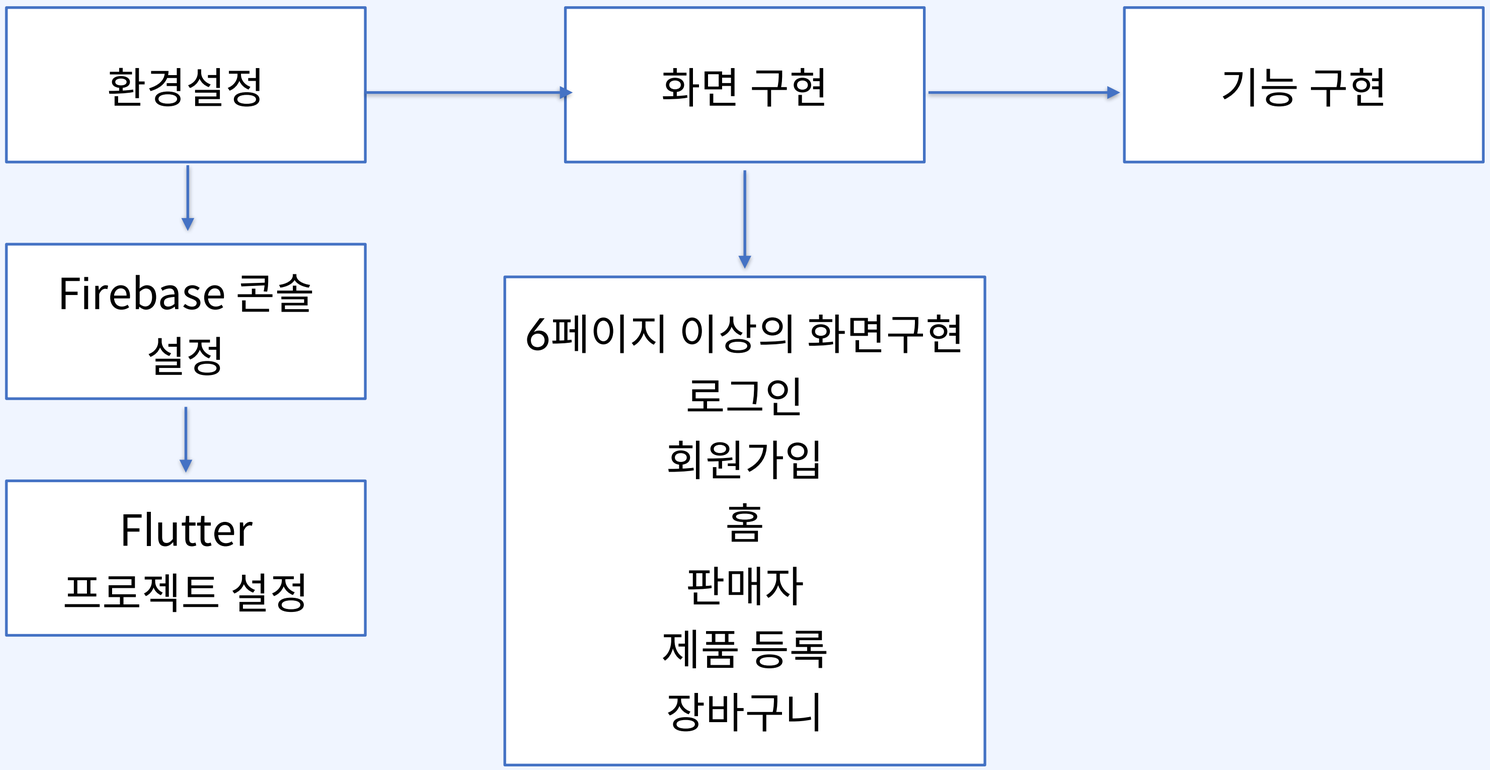
> Firebase 내부 서비스를 활용하여 앱을 구현한다.

기획
돌아보기

요구사항

1. 회원가입과 로그인 기능을 만들어주세요
2. 판매자가 상품을 등록할 수 있도록 해주세요
3. 카테고리별 상품 등록도 될 수 있어야해요
4. 할인하는 제품도 사용자가 확인할 수 있어야해요
5. 사용자는 장바구니에 상품을 담을 수 있어야해요

돌아보기



돌아보기-Firebase Firestore

컬렉션 설계하기

1.사용자 정보 >> users
2.카테고리 정보 >> category
3.제품 정보 >> products
4.장바구니 >> cart

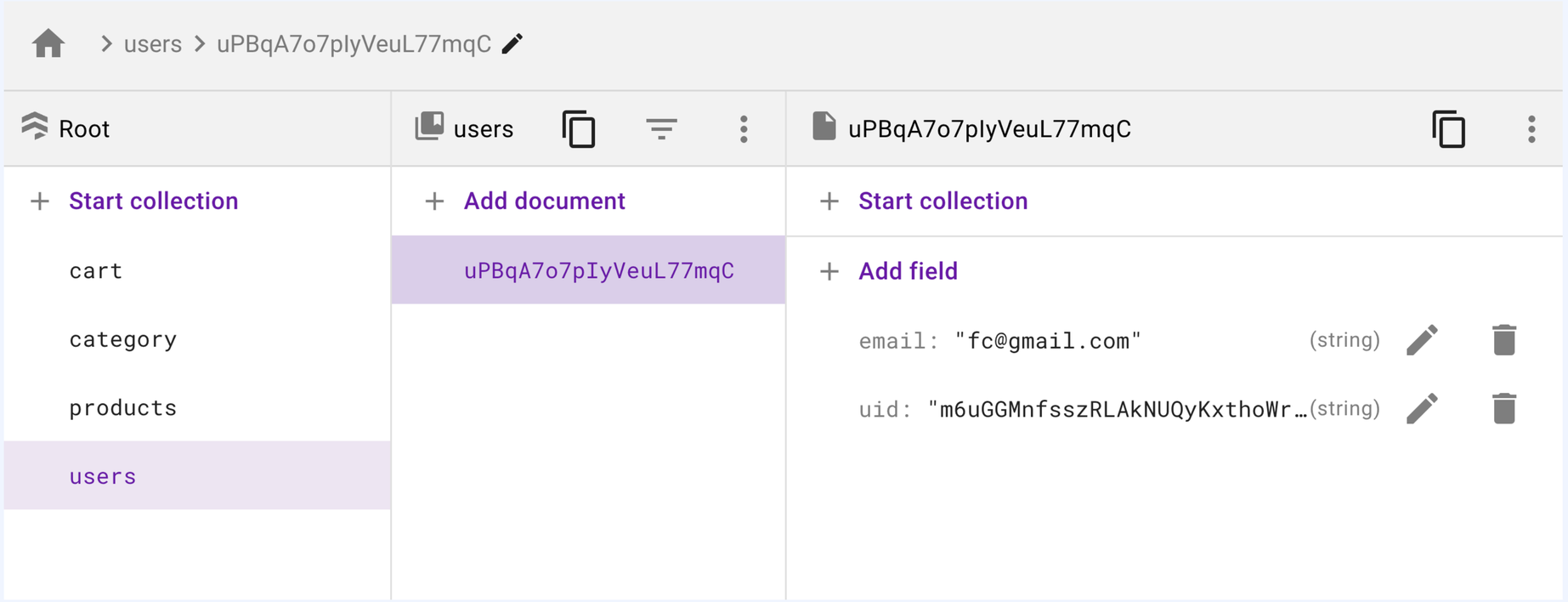
컬렉션과 도큐먼트의 구성으로 설계를 진행

돌아보기-Firebase Firestore

컬렉션 설계하기

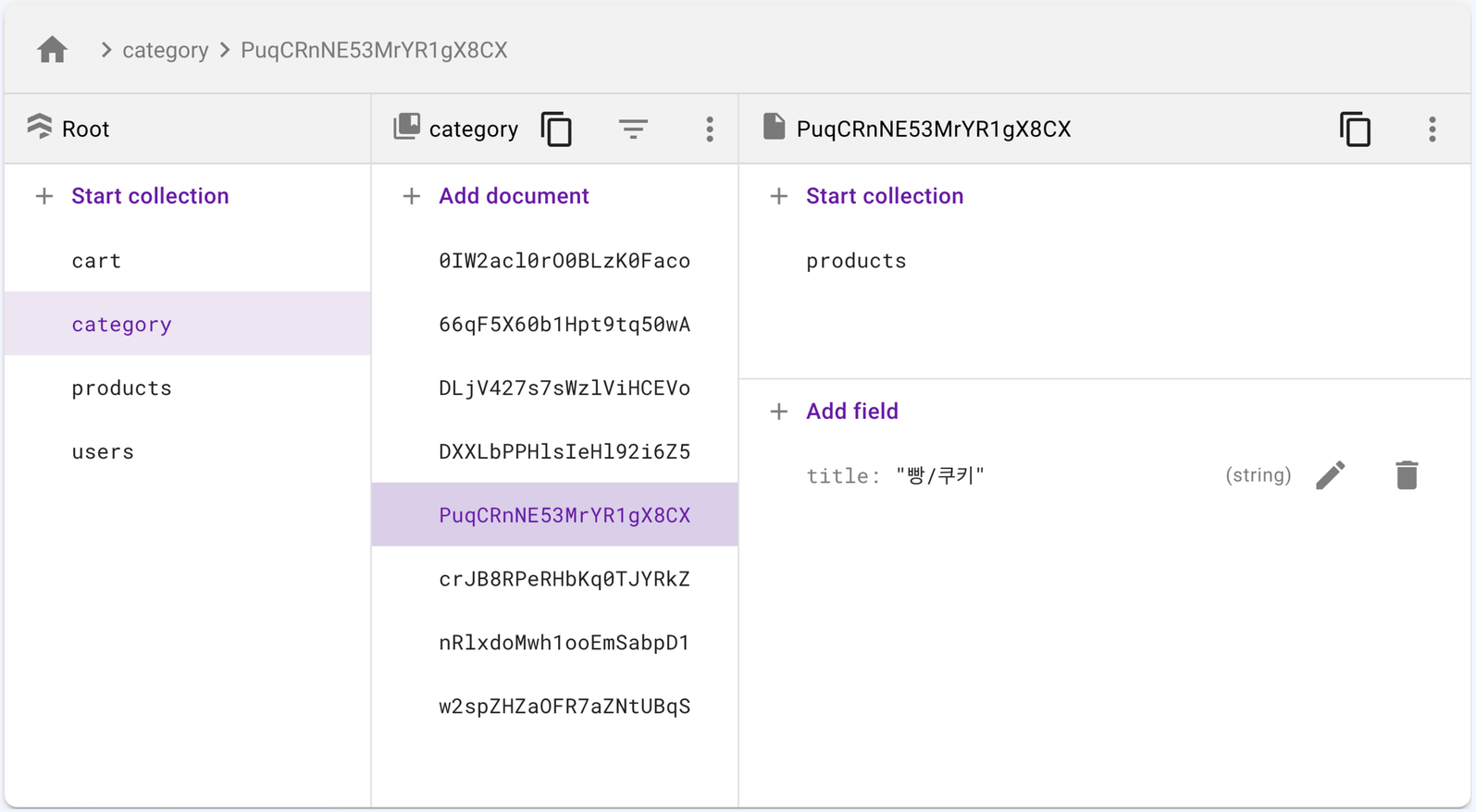
1.사용자 정보

- 이메일
- Uid



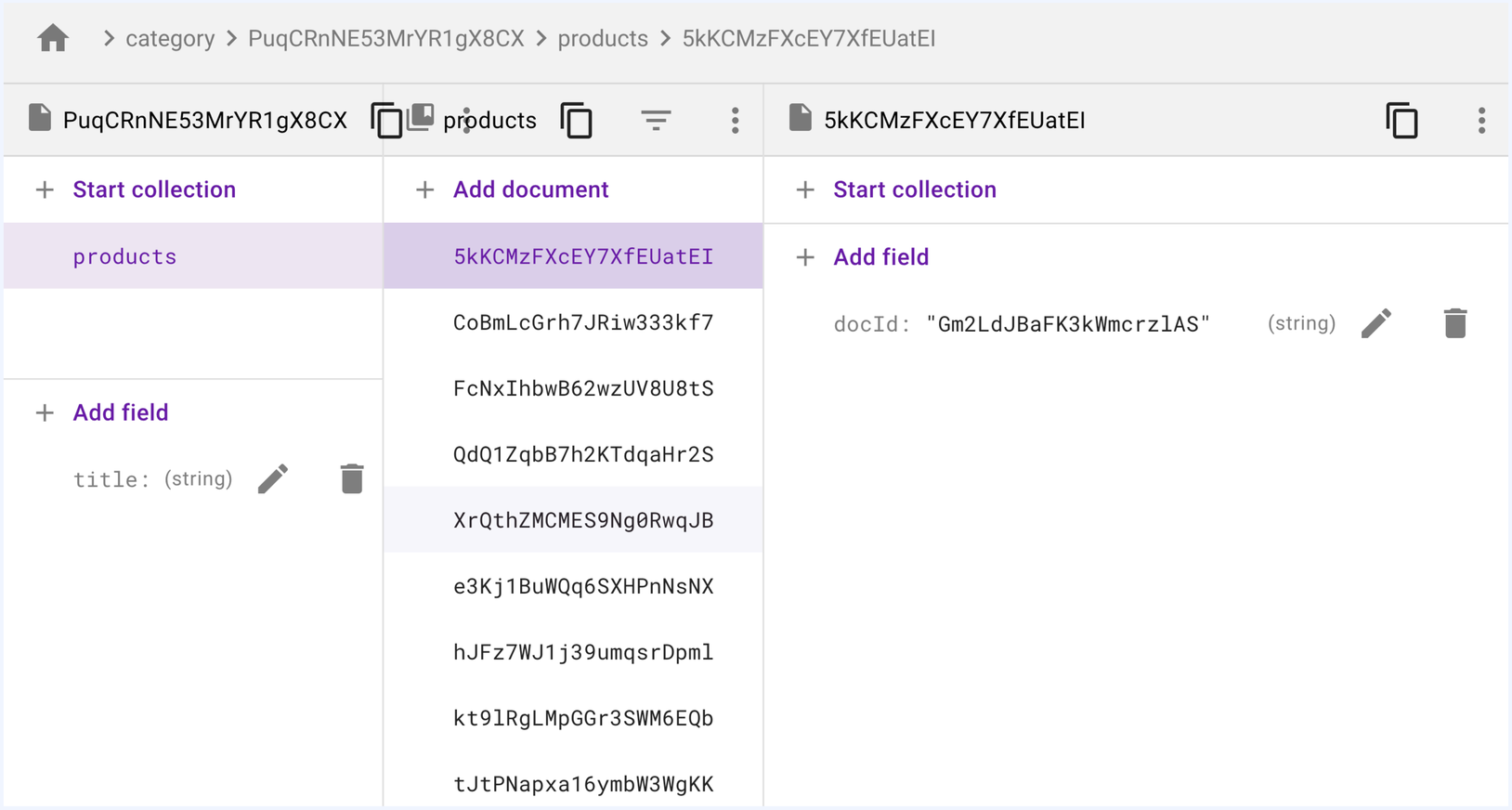
돌아보기-Firebase Firestore

컬렉션 설계하기
2. 카테고리 정보



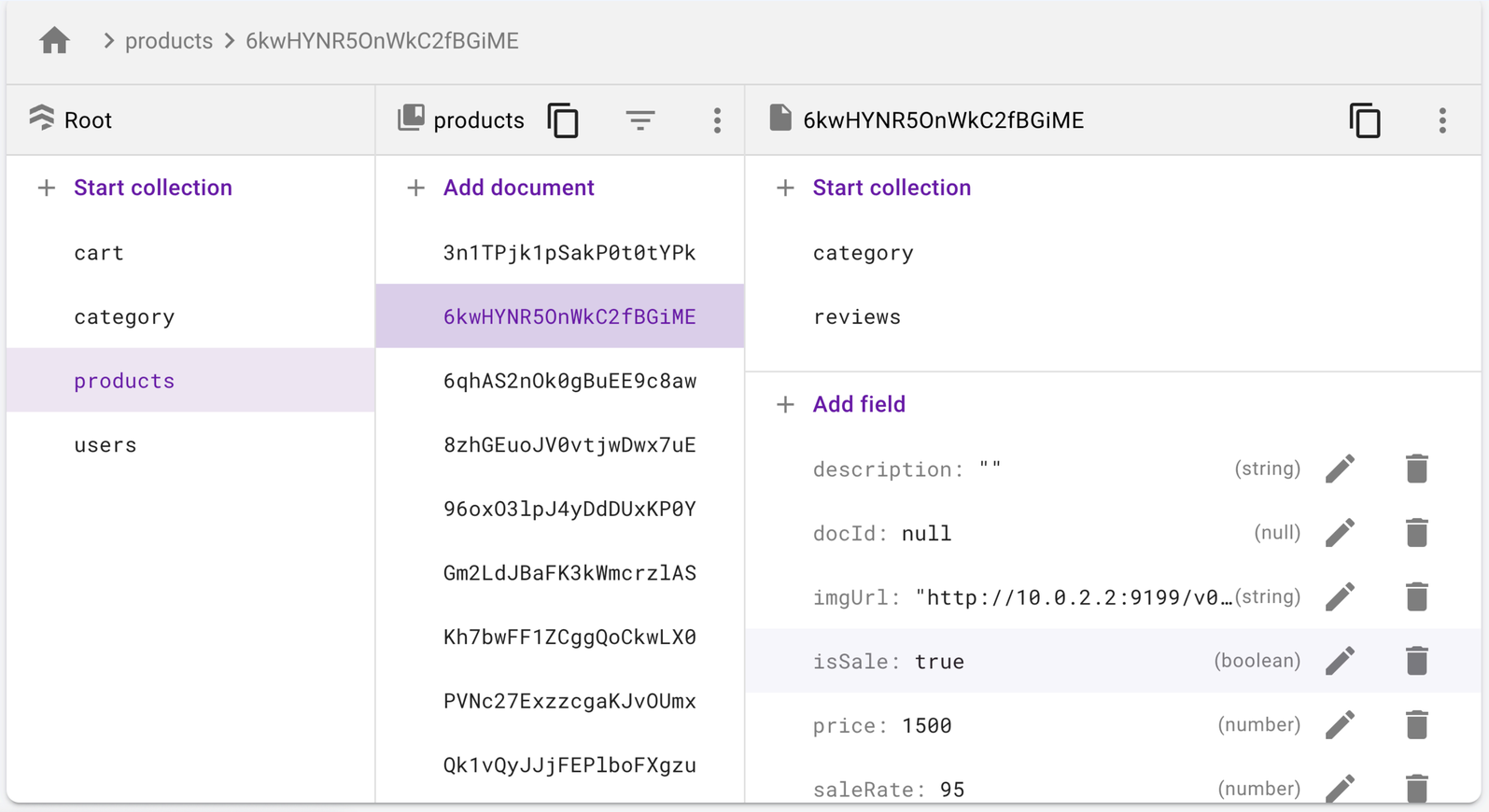
돌아보기-Firebase Firestore

컬렉션 설계하기
2. 카테고리 정보



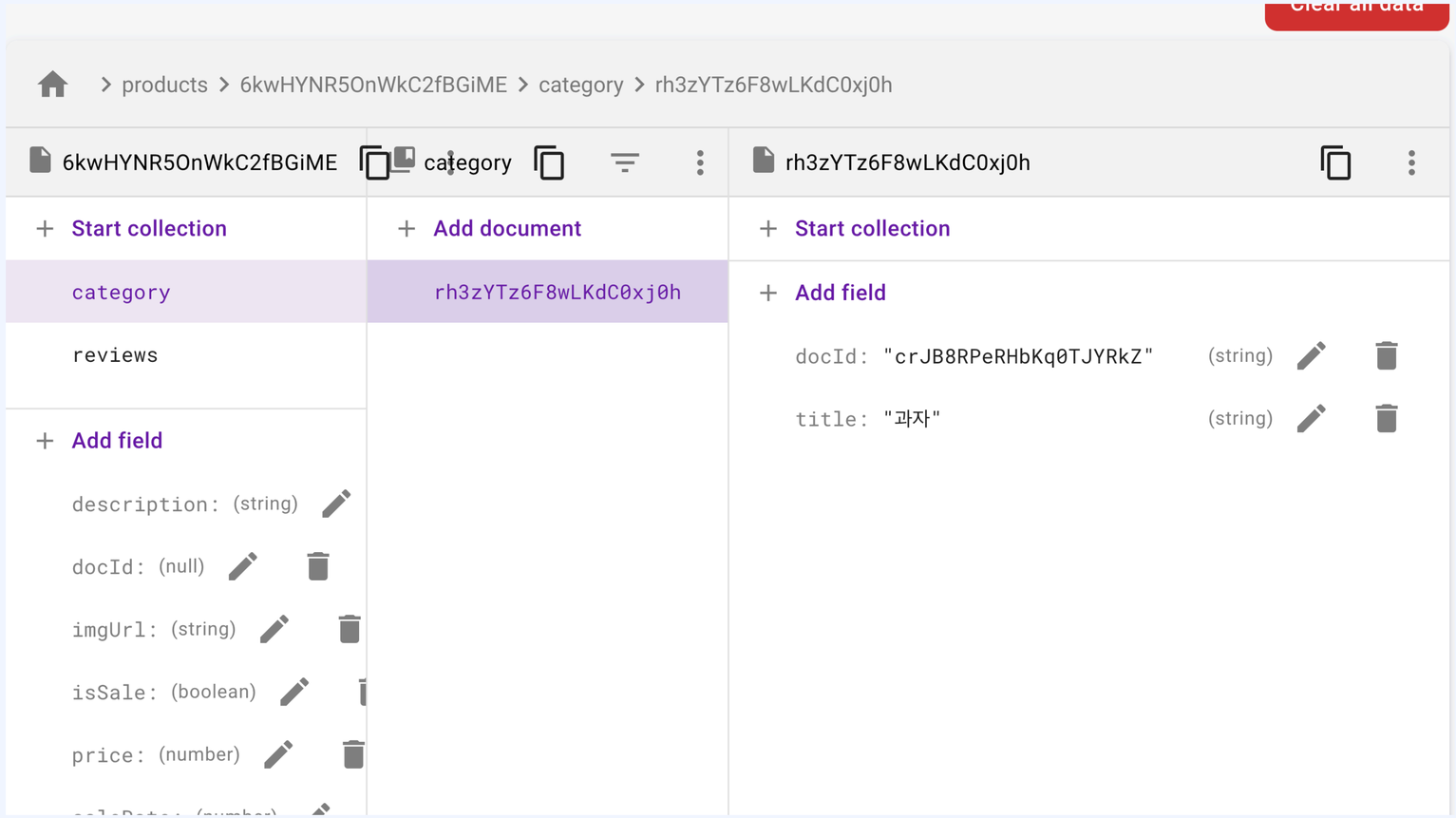
돌아보기-Firebase Firestore

컬렉션 설계하기
3. 제품정보



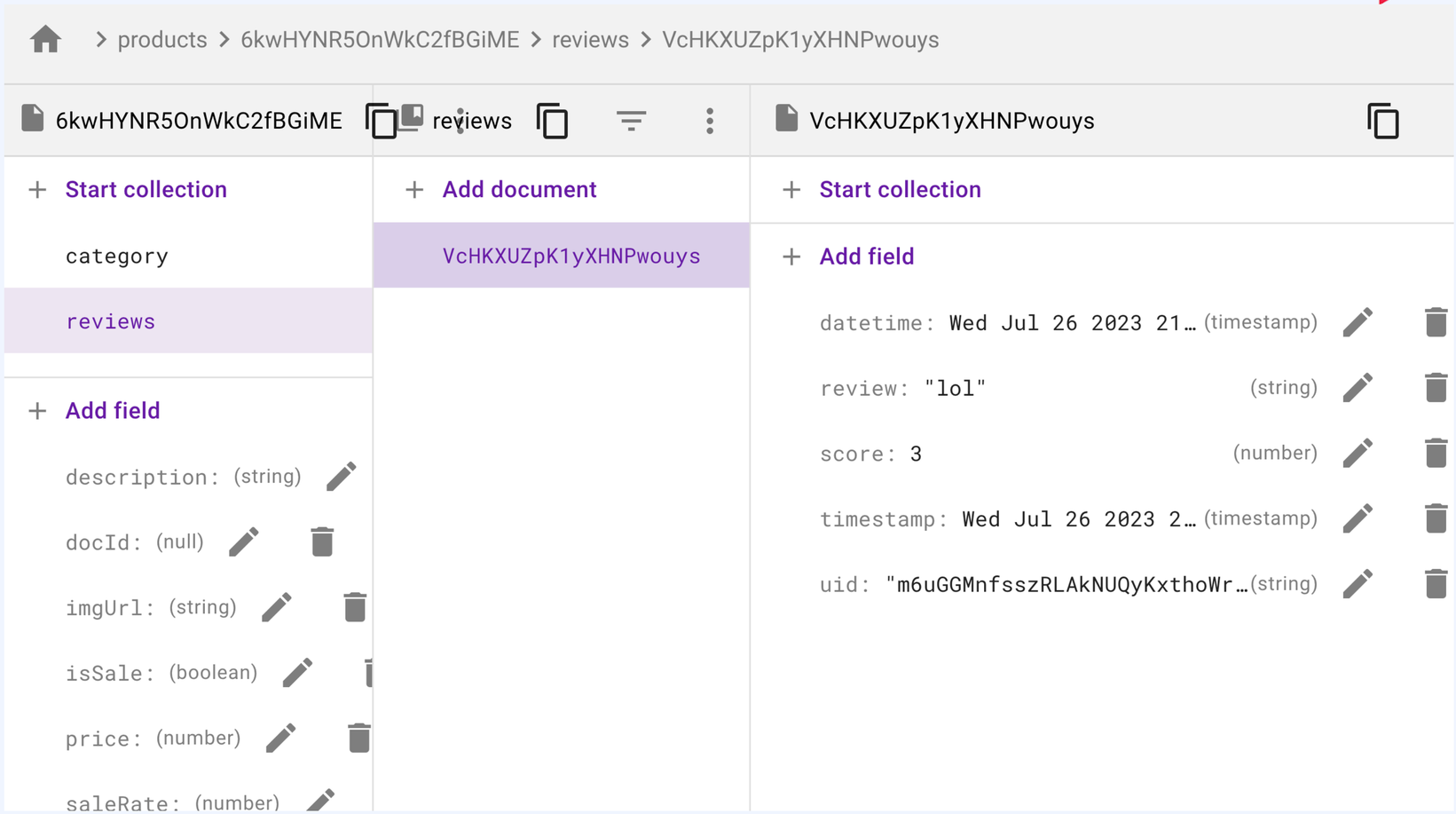
돌아보기-Firebase Firestore

컬렉션 설계하기
3. 제품정보



돌아보기-Firebase Firestore

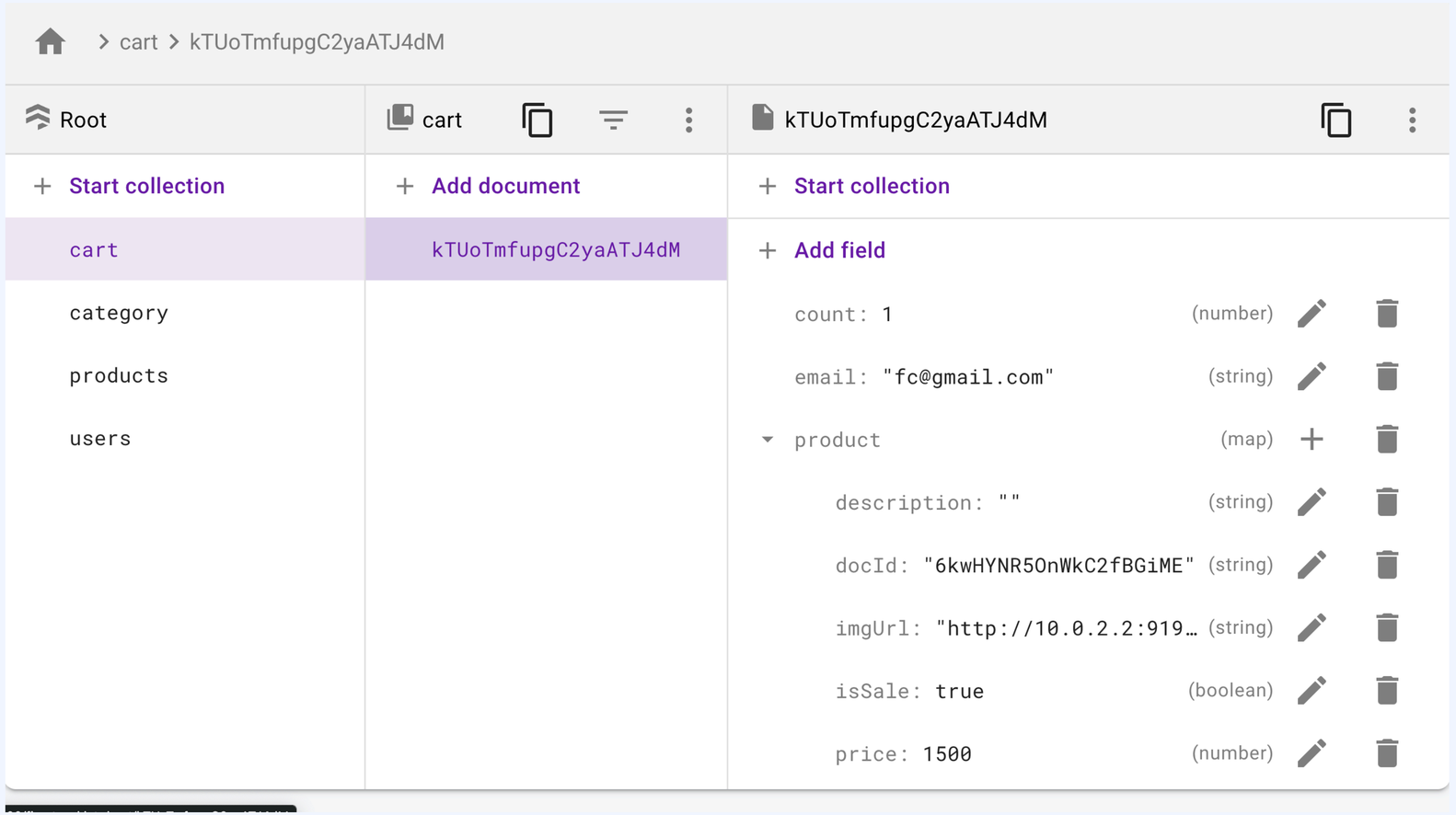
컬렉션 설계하기
3. 제품정보



돌아보기-Firebase Firestore

컬렉션 설계하기

4. 장바구니



돌아보기 - 구현

Firebase 로그인

```
Future<UserCredential?> signIn(String emailAddress, String password) async {
    try {
      final credential = await FirebaseAuth.instance
          .signInWithEmailAndPassword(email: emailAddress, password: password);
      print(credential);
      return credential;
    } on FirebaseAuthException catch (e) {
      if (e.code == 'user-not-found') {
        print('No user found for that email.');
      } else if (e.code == 'wrong-password') {
        print('Wrong password provided for that user.');
      }
    }
  }
```

돌아보기 - 구현

Firebase
회원가입

firestore에
저장하기

```dart
Future<bool> signUp(String emailAddress, String password) async {
  try {
    final credential =
        await FirebaseAuth.instance.createUserWithEmailAndPassword(
      email: emailAddress,
      password: password,
    );
    await FirebaseFirestore.instance.collection("users").add({
      "uid": credential.user?.uid ?? "",
      "email": credential.user?.email ?? ""
    });
    return true;
  } on FirebaseAuthException catch (e) {
    if (e.code == 'weak-password') {
      print('The password provided is too weak.');
    } else if (e.code == 'email-already-in-use') {
      print('The account already exists for that email.');
    }
    return false;
  } catch (e) {
    print(e);
    return false;
  }
}
```

돌아보기 - 구현

이미지 압축

```
Future<Uint8List> imageCompressList(Uint8List list) async {
   var result = await FlutterImageCompress.compressWithList(
     list,
     quality: 50,
   );
   return result;
 }
```

돌아보기 - 구현

Storage 올리기
다운로드 링크 얻기

```
final storageRef = storage.ref().child(
"${DateTime.now().millisecondsSinceEpoch}_${image?.name ?? "${DateTime.now()}.jpg"}");
final compressedData = await imageCompressList(imageData!);
await storageRef.putData(compressedData);
final downloadLink = await storageRef.getDownloadURL();
```

돌아보기 - 구현

Firestore 데이터 쓰기

```
final doc = await db.collection("products").add(sampleData.toJson());
await doc.collection("category").add(selectedCategory?.toJson() ?? {});
final catRef = db.collection("category").doc(selectedCategory?.docId);
await catRef.collection("products").add({"docId": doc.id});
```

돌아보기 - 구현

Firestore 실시간으로 데이터 받아오기
Stream

```
Stream<QuerySnapshot<Map<String, dynamic>>> fetchCartItems() {
  return FirebaseFirestore.instance
      .collection("cart")
      .where("email", isEqualTo: widget.uid)
      .orderBy("timestamp")
      .snapshots();
}
```

돌아보기 - 구현

Firestore 실시간으로 데이터 받아오기
Stream

```
StreamBuilder(
  stream: fetchCartItems(),
  builder: (BuildContext context,
      AsyncSnapshot<QuerySnapshot<Map<String, dynamic>>> snapshot) {
```

더 나아가기
Next Step

1. 다양한 로그인 방법 붙여보기
2. 주문하기 화면 구현해보기
3. 리뷰 화면 추가 구현해보기

구현이란 100% 정답이 없습니다.
많이 만들어보고 반복해보면서 시스템에 맞는 여러분만의 적합한 방법을 찾아보세요!
끝까지 고생많으셨습니다. 화이팅!