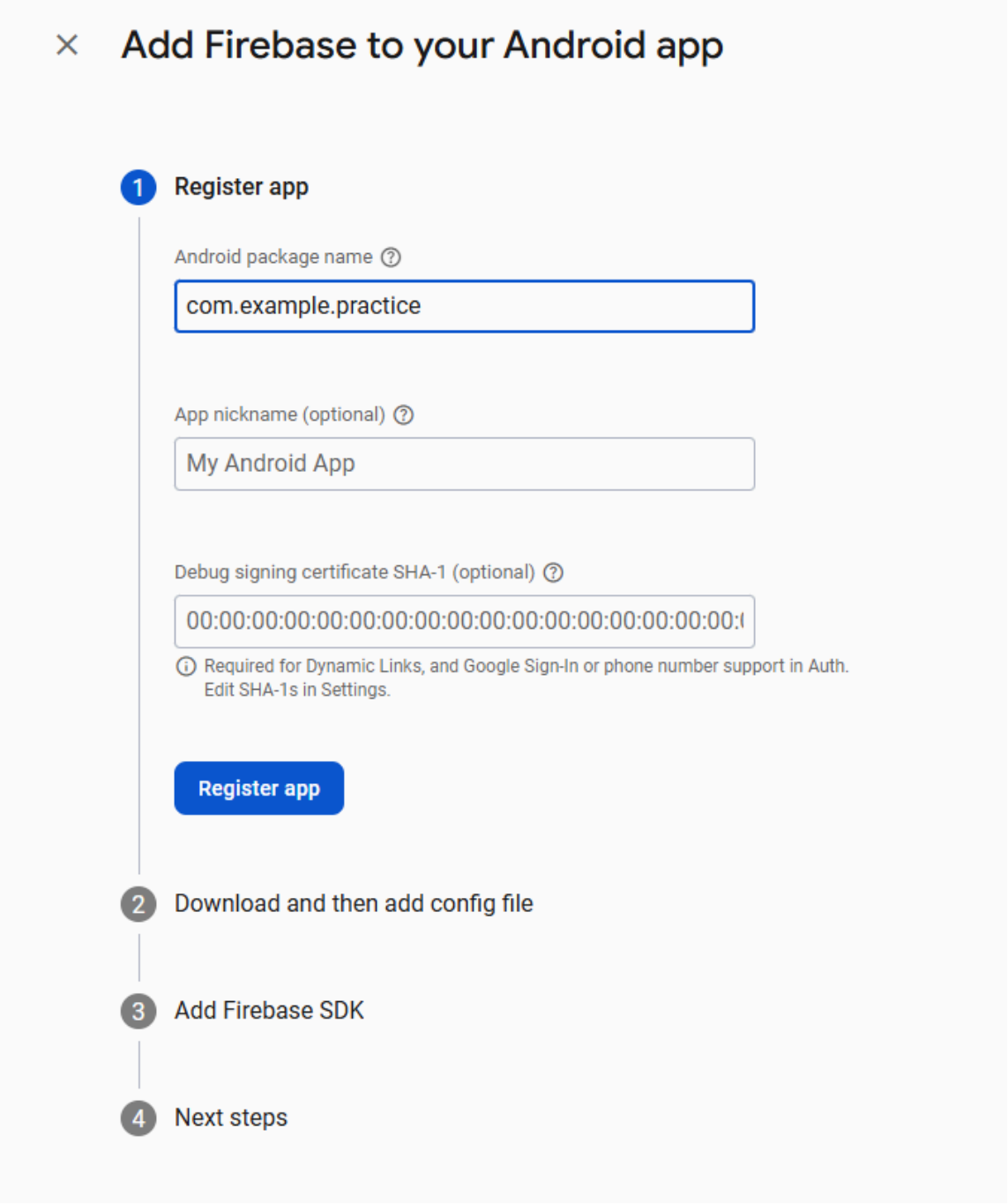
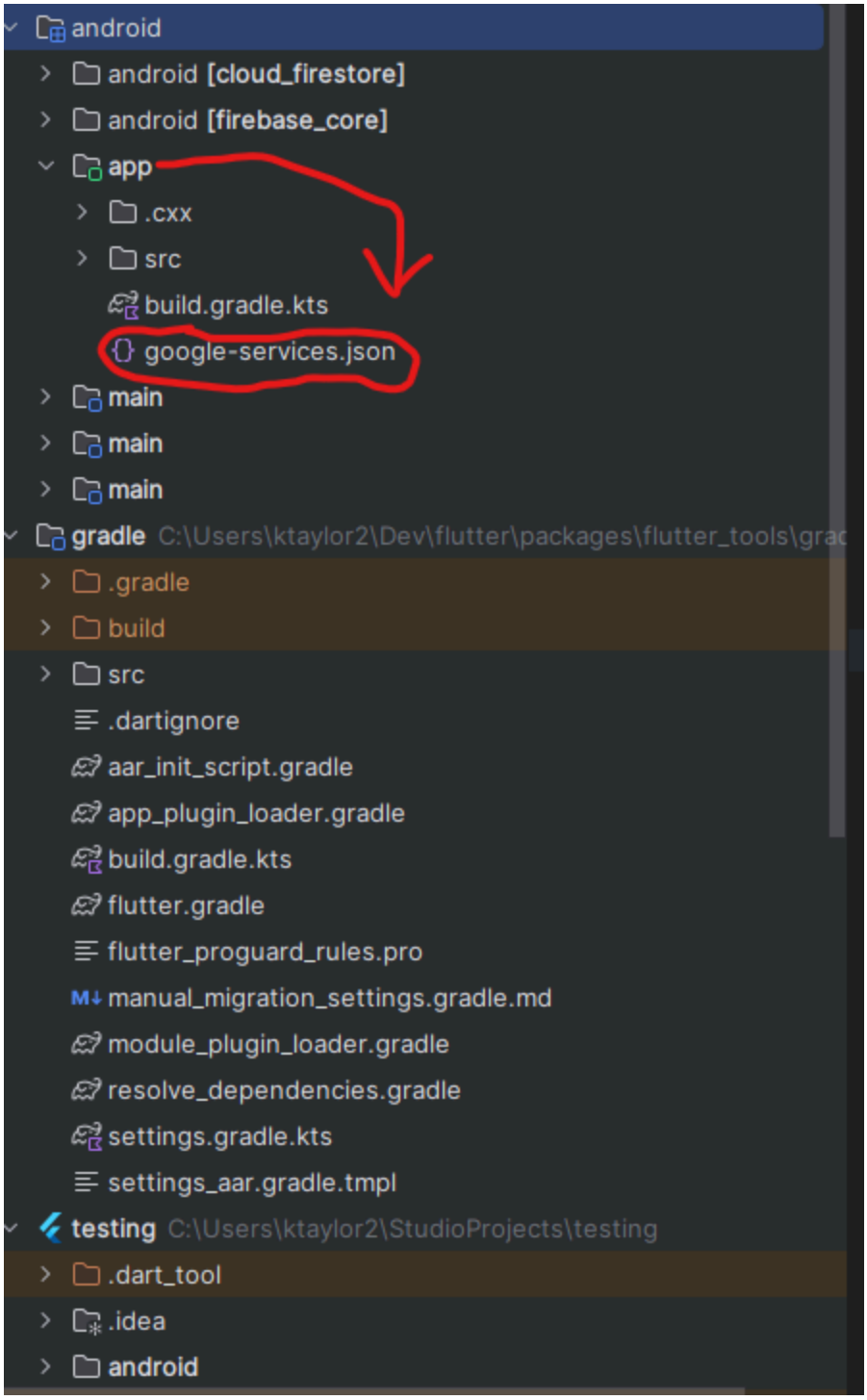
Connecting to Firebase

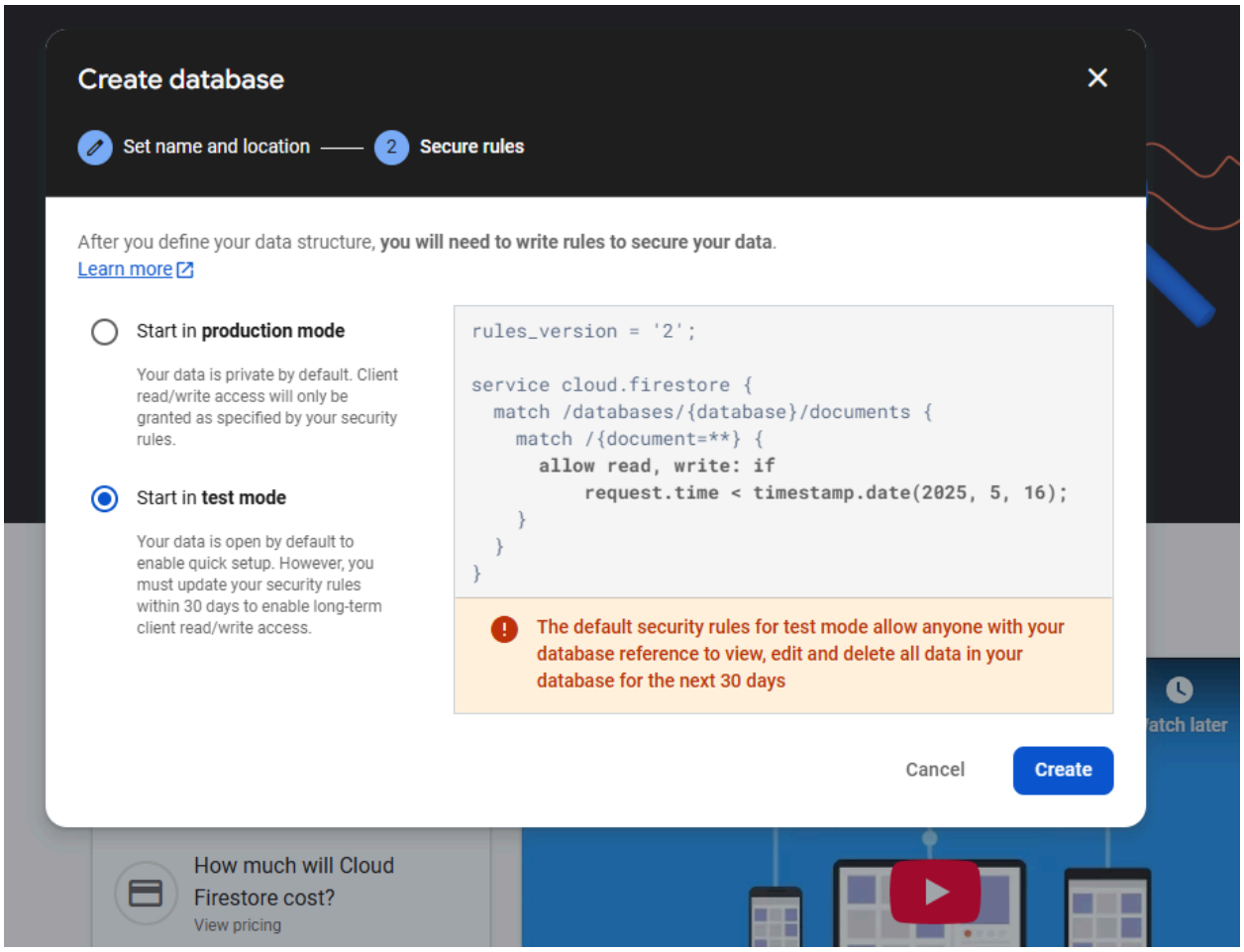
1. Go to Firebase Website and Scroll Down until you see “Get started in console”
    a. Open that
2. Click “Create a Firebase project” and go through it the steps and create it
3. In your new project at the top you will have different options in terms of what kind of app you want to make, for this documentation, we will be following the Android app route
4. Clicking it should take you to this page

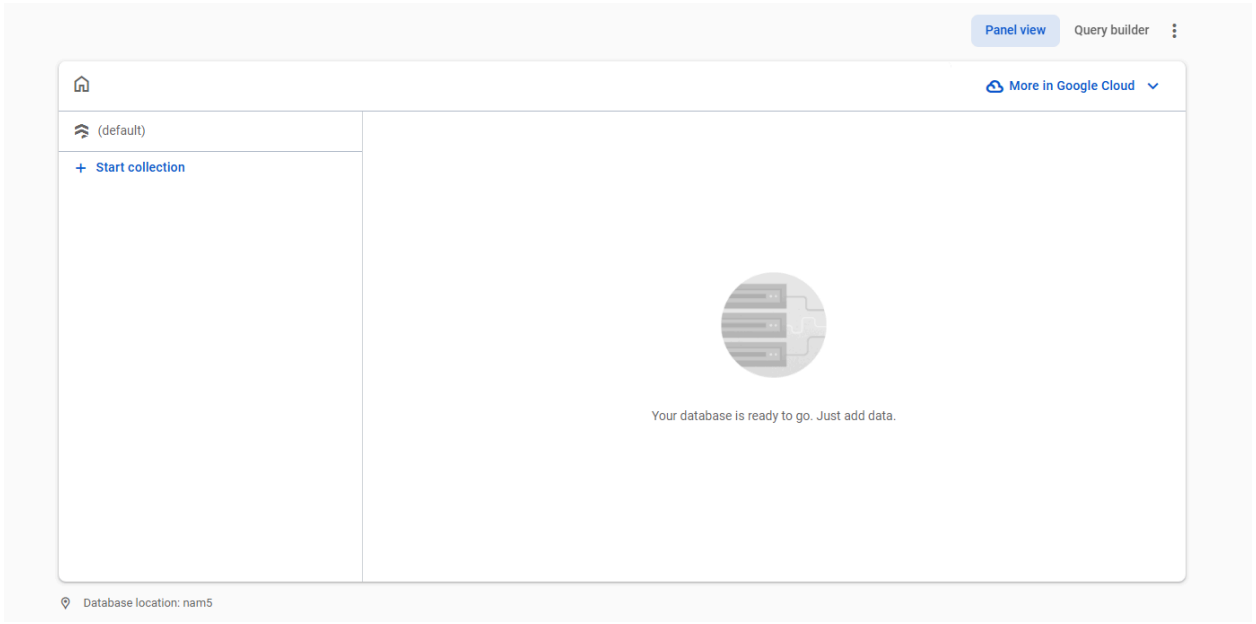


5.
6. Give the company name, this case the first half should be com.example followed by what name you want, REMEMBER this
7. Click register app and now it is going to have you download a “google-services.json” file
    a. Put that file in your Android/App Directory in your project of where your coding

8. 

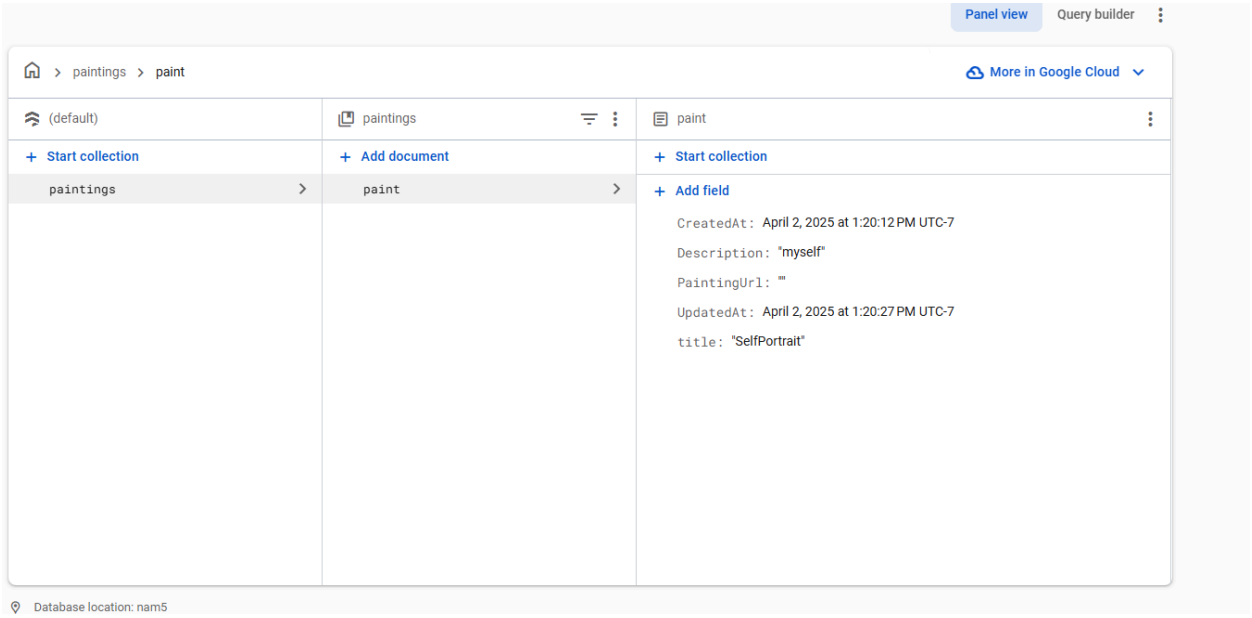
9. Follow the Kotlin DSL process and add in the required plugins and dependencies
10. Now continue to console
11. Now we are going to create a database
    a. On the left of the screen, expand Build and you will see a “Firestore Database” option
        i. Select it
12. Click create database and make sure “Start in test mode is selected”

13. 

14. Your page should now look like this
15. 

16. We are going to start a collection – think of it like a folder to store our “files” of which will hold our data or key-value pairs

17. Inside that collection we are going to make a document of which will hold our data, name these appropriately and give it a temporary field to see how it looks



18.
19. We finished establishing the connection, now we need to be able to handle data between our app and the database

Code and Data Handling

1. Handling Data Storage

```
Future<void> savePainting(String title, String description) async {
 try {
   await FirebaseFirestore.instance.collection('paintings').add({
     'title': title,
     'Description': description,
     'PaintingUrl': '',
     'CreatedAt': Timestamp.now(),
     'UpdatedAt': Timestamp.now(),
   });
   debugPrint("Painting saved!");
 } catch (e) {
   debugPrint("Error saving painting: $e");
 }
}
```

   o. We use future<void> here so that the function can run in the background and as well we use async so that it runs asynchronously with our app
      i. We accept two parameters, title and description (for now) so we can test if we can handle data with our database
   p. We then us an await so that the code waits for this to finish before we continue

i. In this we save in our collections named “paintings” in our database these fields of which we will pass, for now we are not working with PaintingUrl and the Description and Title will be passed from when we enter these in a text field later on and click save

2. Handling User Text Input

```
final TextEditingController _titleController =
TextEditingController();
final TextEditingController _descController =
TextEditingController();
```

c. Used to store user text of which we will retrieve later on

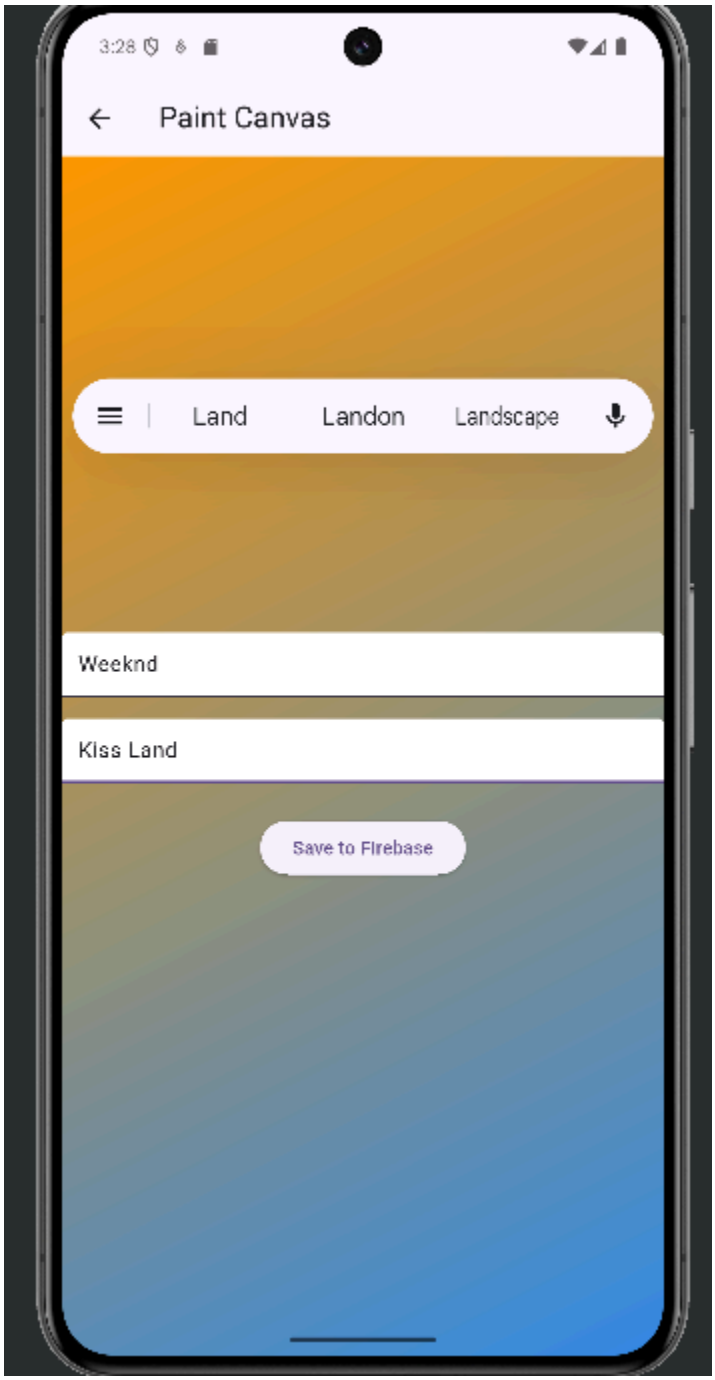
```
children: [
 TextField(
   controller: _titleController,
   decoration: const InputDecoration(
     hintText: 'Enter title',
     filled: true,
     fillColor: Colors.white,
   ),
 ),
 const SizedBox(height: 16), // spacing between the fields
 TextField(
   controller: _descController,
   decoration: const InputDecoration(
     hintText: 'Enter description',
     filled: true,
     fillColor: Colors.white,
   ),
 ),
 const SizedBox(height: 24),
 ElevatedButton(
   onPressed: () {
     final title = _titleController.text.trim();
     final desc = _descController.text.trim();
     savePainting(title, desc);
   },
   child: const Text("Save to Firebase"),
 ),
],
```

ff. Here we have two text fields implementing the controllers we have to store user text into two input boxes - Title and Description

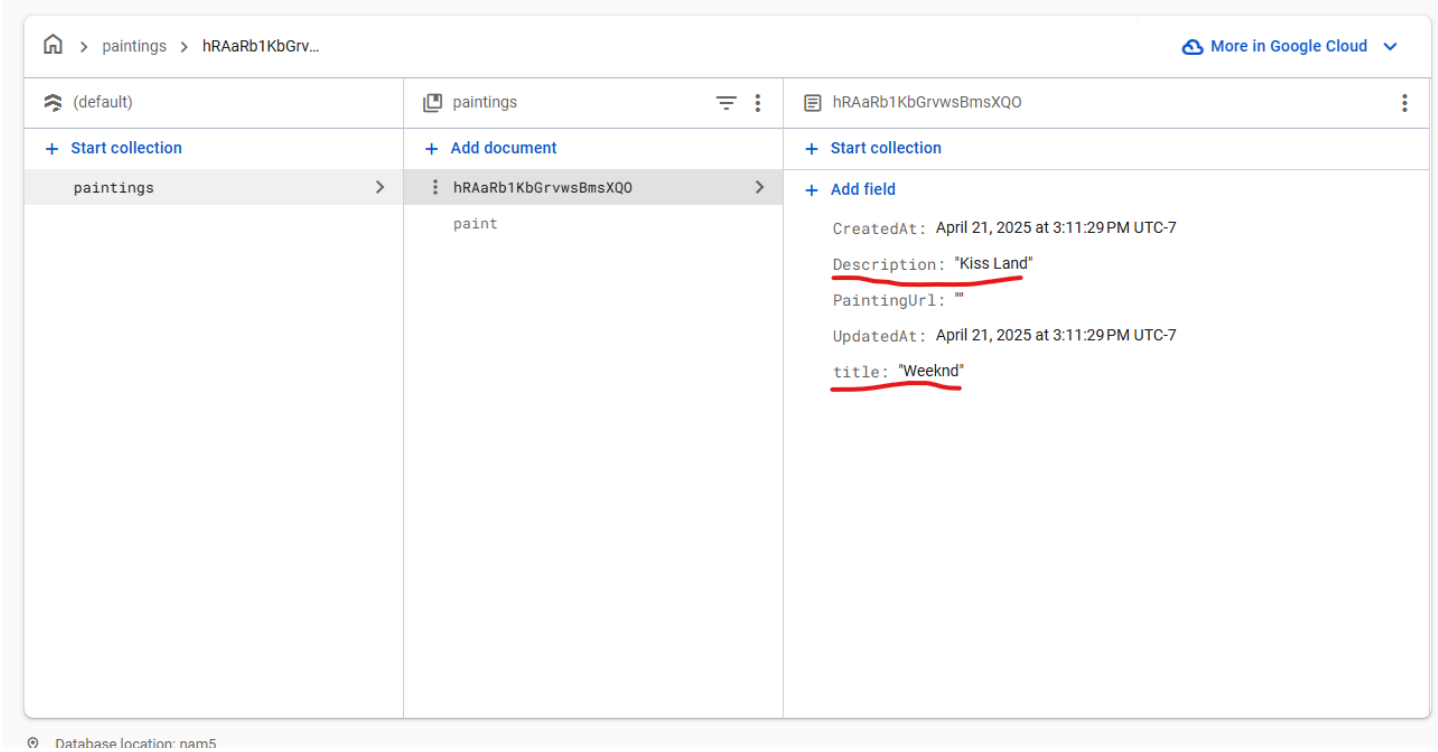
gg. We then make a button that is titled “Save to Firebase” which assigns the variables to the text in the fields for our title and description and then calls the

savePainting function we made earlier (the one with Future<void> of which those fields will be saved and updated in our database

Example:



→


paintings
hRAaRb1KbGrv...
More in Google Cloud
(default)
paintings
hRAaRb1KbGrvwsBmsXQO
Start collection
Add document
Start collection
paintings
hRAaRb1KbGrvwsBmsXQO
Add field
paint
CreatedAt: April 21, 2025 at 3:11:29 PM UTC-7
Description: "Kiss Land"
PaintingUrl: ""
UpdatedAt: April 21, 2025 at 3:11:29 PM UTC-7
title: "Weeknd"
Database location: nam5