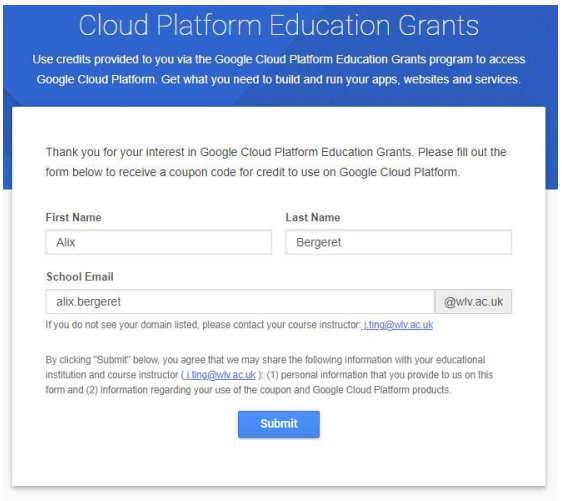
Google Firestore

In this workshop you will learn to create a Google Firestore NoSQL database, then access the data from a simple HTML/JavaScript web app.

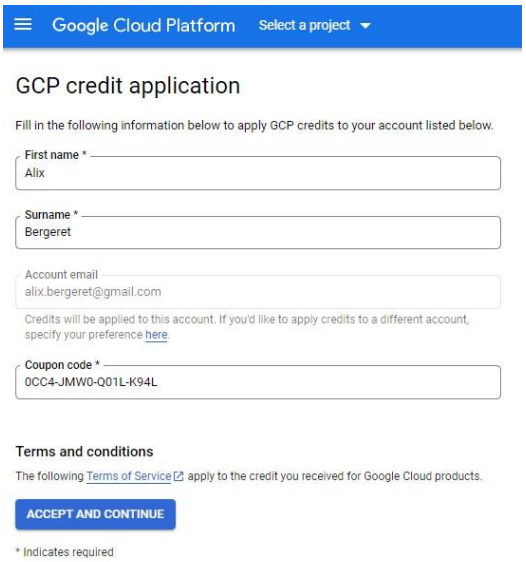
Part 1 – Getting started with Google Firestore

Register for Google Cloud fill

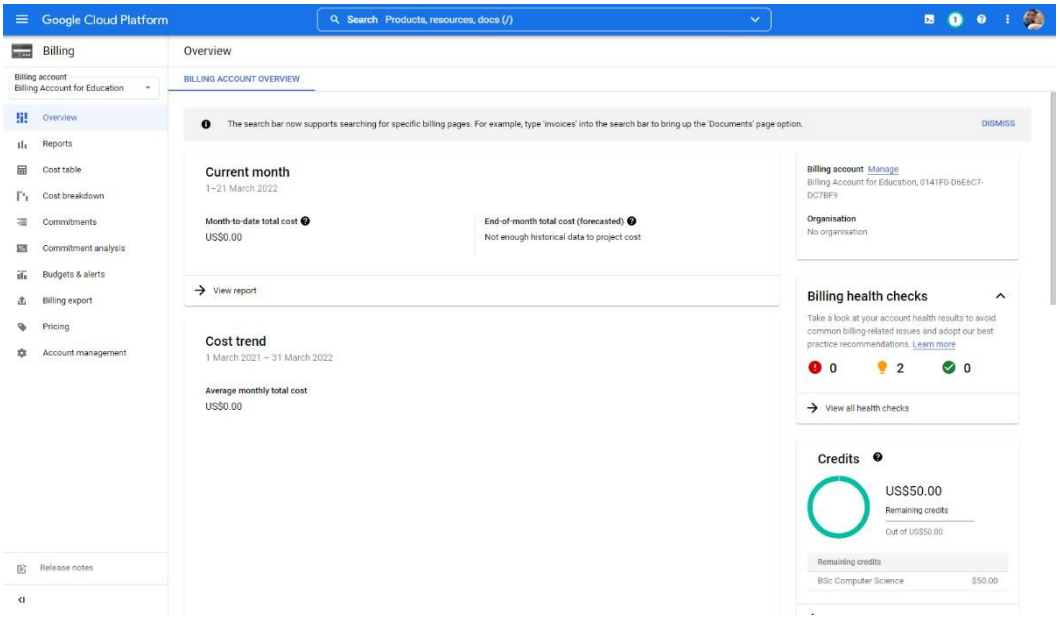
in the form with your details:



A second email will allow you to verify your email address, and once that's done, you will receive a third email containing your coupon. Click on that link to redeem your coupon, and fill the form:



You may have to agree to the Terms of Service (if it's the first time you use Google Cloud), and should then arrive on the Billing Account Overview page:

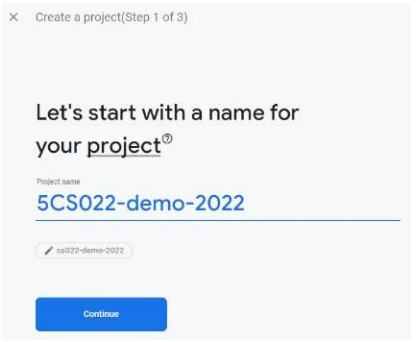


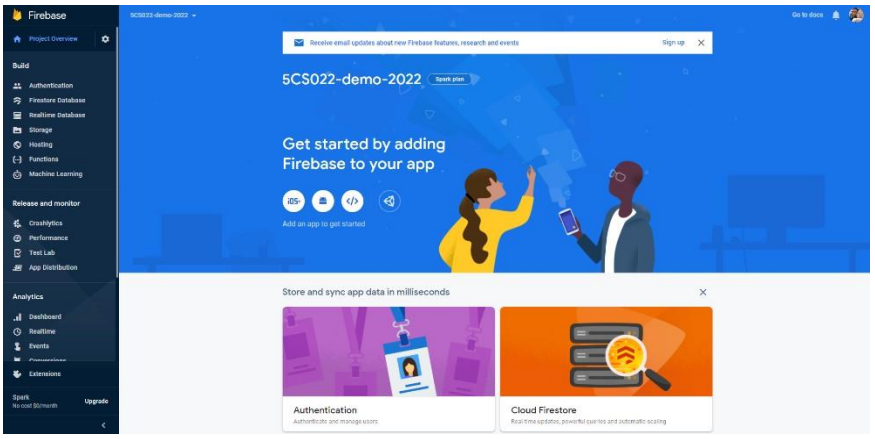
Access Firebase

"Firebase is a platform developed by Google for creating mobile and web applications. It was originally an independent company founded in 2011. In 2014, Google acquired the platform and it is now their flagship offering for app development." (source)

Next, access the **Firebase console**: https://console.firebase.google.com/

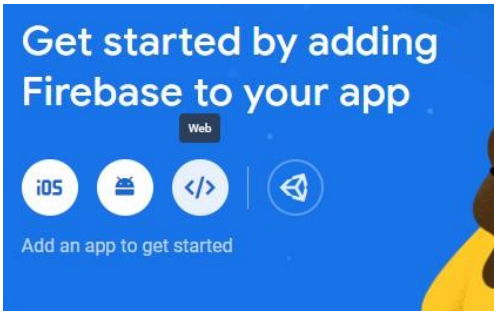
- Press the "Add project" button.
- On the following screen, **enter a new project name (maybe "5CS022" + your student number)** and press "Continue".
- On the "Google Analytics", **deselect** "Enable Google Analytics for this project", as we don't need it for this workshop. Press "Continue".
- Your Firebase project will be created (it takes a few seconds). When ready, press "Continue"… you should arrive on the Firebase Console:



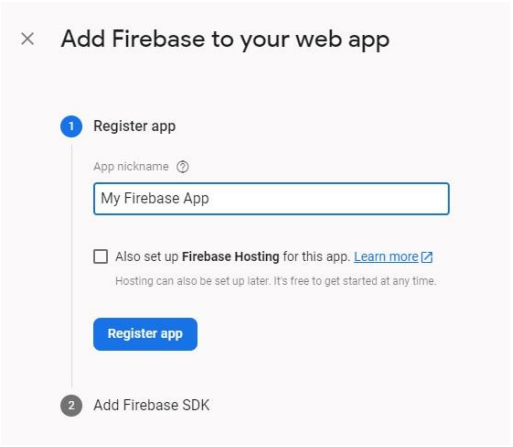


Note: you can come back to your Firebase Console any time, by clicking on the following link then selecting your project: https://console.firebase.google.com/

The last thing we need to do is “Add Firebase to an app” (see the very large blue section on the homepage of your Firebase console.) **Select the “web” option**, as we will be accessing our Firestore from JavaScript:



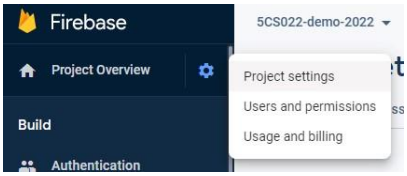
Give your app any name you like and press “Register app”.



On the next screen, copy and paste the JavaScript code extract into a text file somewhere. It is very important as it contains YOUR connection details. We will use it later when writing our JavaScript. Mine looks like this, BUT USE YOUR OWN:

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyA0VPouX3O5_KJ7VDlwZjh1y_4iFYQ0njc",
  authDomain: "cs022-demo-2022.firebaseapp.com",
  projectId: "cs022-demo-2022",
  storageBucket: "cs022-demo-2022.appspot.com",
  messagingSenderId: "344011870316",
  appId: "1:344011870316:web:0d9bf0a31966ec09350849"
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
```

Note: if you lose the credentials above, you can simply retrieve them from your Firebase Console. Click on your project, then “Project Overview”, then “Project settings”:

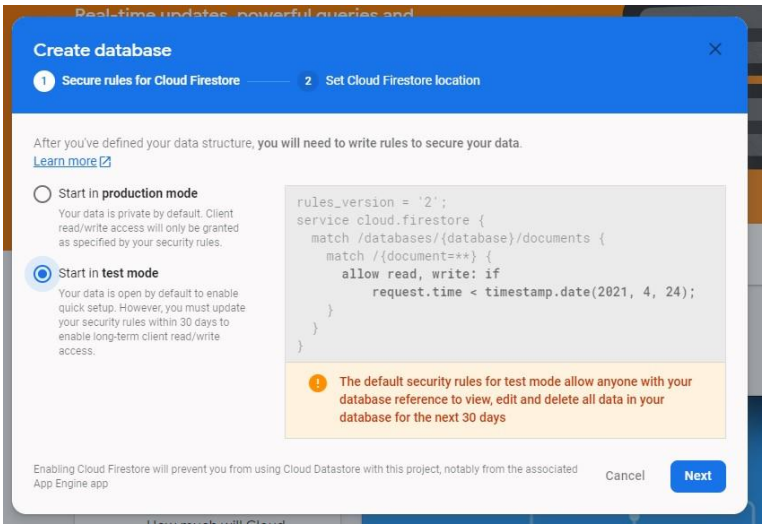


Create a Firestore

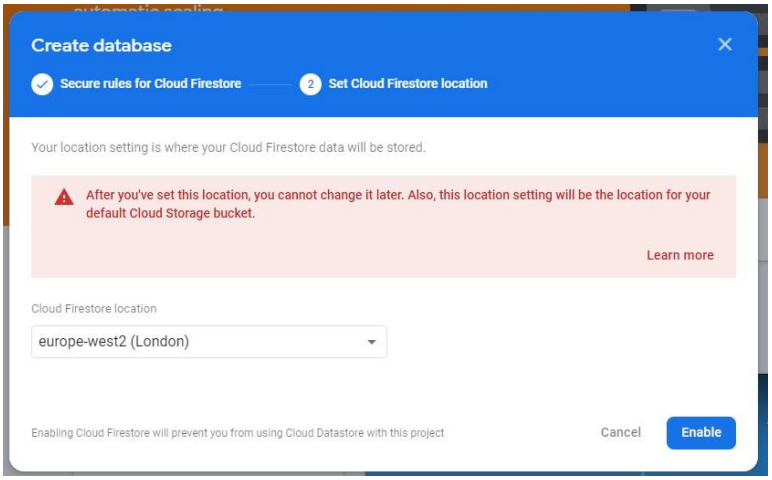
"Easily develop rich applications using a fully managed, scalable, and serverless document database." (*source*)

Next, we need to **create a Firestore** (NoSQL database):

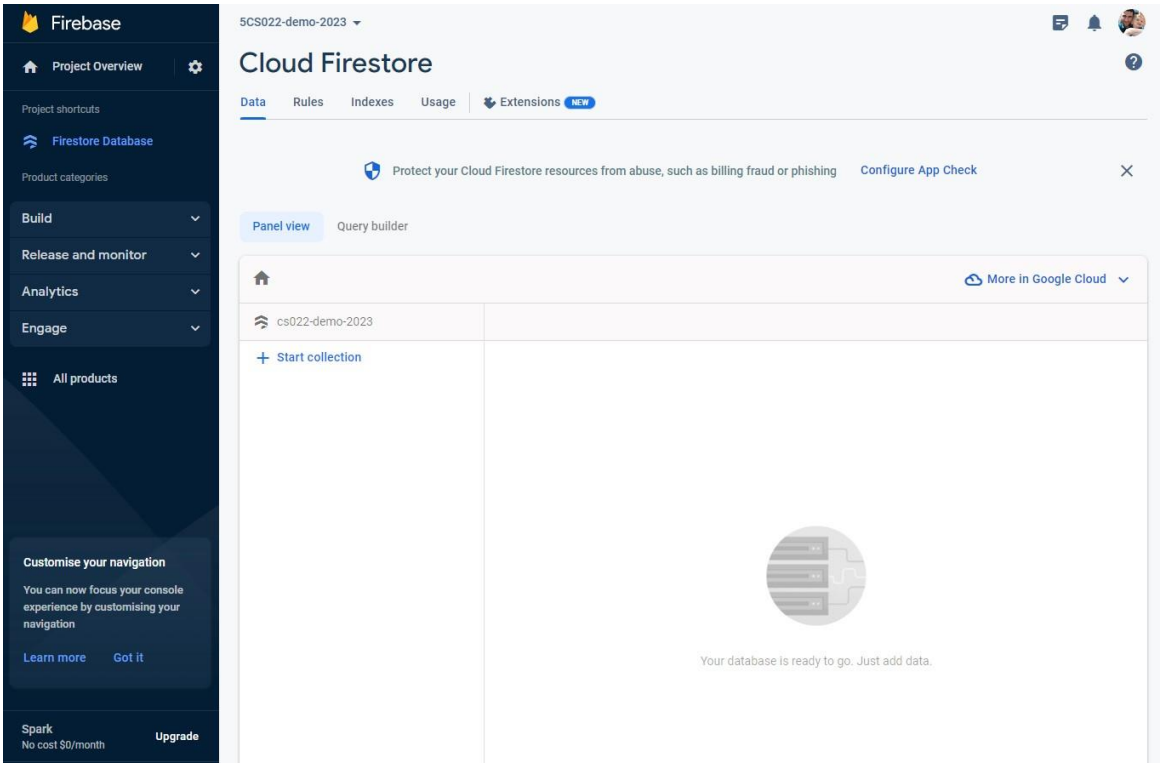
- In your Firebase console, in the menu on the left select **Firestore Database** (under "Build").
- Click on the "Create database" button in the middle of the screen.
- On the next screen, choose "**Start in test mode**". This will create an open rule that allows for easy access of your data later on. Obviously, this is not secure!



- On the next screen, set the location of your Firestore to somewhere nearby, e.g. **Europe-west2 (London)**. This will impact performance and cannot be changed later! Press "Enable".



You should now be in your **Firestore console**:



Explore the tabs at the top:

- "**Data**" is where you can add your collections and documents (see next section)
- "**Rules**" allows you to control access to your data. Because we chose "test mode" earlier, it's open by default.
- "**Indexes**" allows you to create document indexes, just like in a relational database. You will only need this if you use complex queries later on (e.g. if you sort on 2 fields) - "**Usage**" is self-explanatory!

Note that you can access the above via 2 different consoles:

- https://console.cloud.google.com/firestore/
- https://console.firebase.google.com/

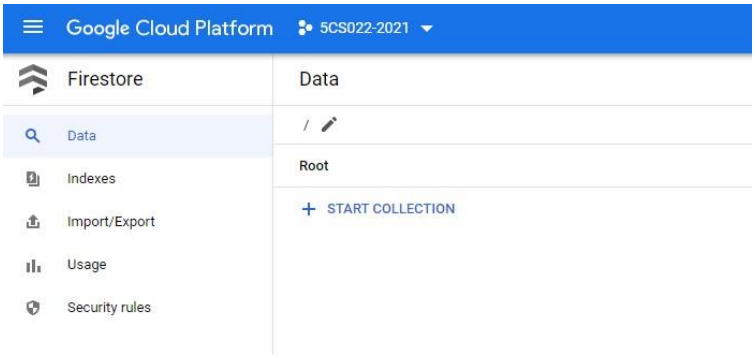
Part 2 – Creating your collections and documents

"Firestore is a NoSQL, document-oriented database. Unlike a SQL database, there are no tables or rows. Instead, you store data in documents, which are organized into collections." (source)

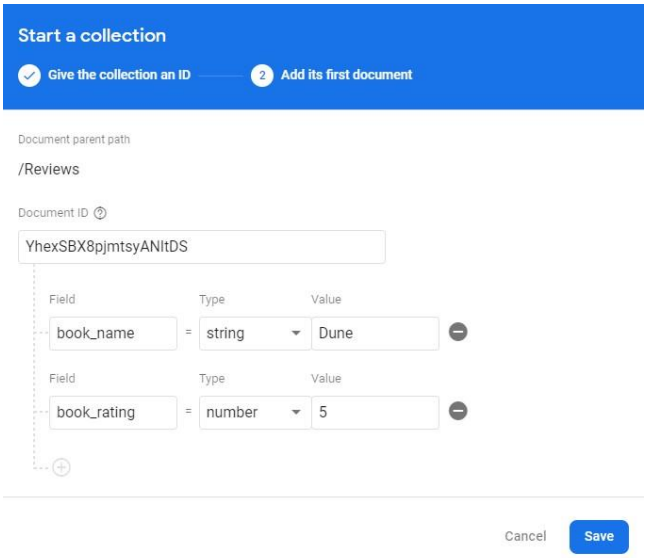
Read about collections and documents here.

Let's create a "Reviews" **collection** with a couple of **documents**.

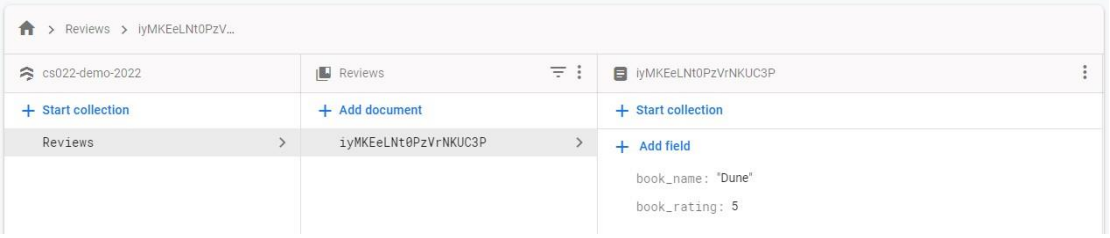
1. In your Firestore console, under "Data", press "+ Start Collection"



2. Type “Reviews” in the Collection ID field, then press “Next”.
3. Add a first Document:
    a. Press “Auto-ID” to populate the Document ID field.
    b. Create 2 “book_name” and “book_rating” fields (with values), as follow, and press “Save”:



4. You should now see your **collection** and single **document**, as follow:



5. Add a few more **documents** in your “Reviews” collection, specifying the same “book_name” and “book_rating” fields every time (but with different values, put your favourite books in there!)

Part 3 – Accessing your data from JavaScript

Let’s create a simple web page that will access the data from part 2.

1. Using your favourite coding text editor or IDE, create a **new HTML file**, e.g. **index.html**, with the following starter template:

```
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.c ss" rel="stylesheet"></head>

    <title>My Firebase app</title>
  </head>
  <body>
      <div class="container">
            <h1 id="mainTitle">My books</h1>
            <table class="table table-striped">
                  <tbody id="reviewList">
                  </tbody>
            </table>
      </div>

      <!-- jQuery -->
      <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

      <!-- Bootstrap JavaScript -->
      <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle. min.js"></script>

  </body>
</html>
```

Things to note:

- We are using the **Bootstrap 5.3.0** starter template located here
- We also include **jQuery**, which drastically simplifies the syntax of our code.
- Our document isn't showing much at the moment, here is mine: https://mi-linux.wlv.ac.uk/~in9352/google-2023/firebase-client-01.html

2. Okay let's connect to our Firestore. Create a **new file** called "**myscripts.js**" with the following code:

```
// Import required Firebase services
import { initializeApp } from
"https://www.gstatic.com/firebasejs/9.18.0/firebase-
app.js"; import { Firestore, getFirestore, onSnapshot,
query, collection,
              orderBy,
              addDoc } from
'https://www.gstatic.com/firebasejs/9.18.0/firebase-firestore.js'

// Your web app's Firebase configuration
const firebaseConfig = { apiKey:
  "use your own", authDomain: " use
  your own ", projectId: "use your
  own", storageBucket: "use your
  own", messagingSenderId: "use
  your own", appId: "use your own"
};

// Initialize Firebase const app =
initializeApp(firebaseConfig); const db =
getFirestore(app);
```

Note: please use your own details, see Part 1 in this document.

3. Then in the **same** "myscripts.js" file, **add** the following code:

```
// Get a live data snapshot (i.e. auto-refresh) of our Reviews
collection const q = query(collection(db, "Reviews"),
orderBy("book_name")); const unsubscribe = onSnapshot(q, (snapshot) => {

  // Empty HTML table
  $('#reviewList').empty();

  // Loop through snapshot data and add to HTML table var
  tableRows = ''; snapshot.forEach((doc) => { tableRows +=
  '<tr>'; tableRows += '<td>' + doc.data().book_name + '</td>';
  tableRows += '<td>' + doc.data().book_rating + '/5</td>';
  tableRows += '</tr>';
  });
  $('#reviewList').append(tableRows);

  // Display review count
  $('#mainTitle').html(snapshot.size + " book reviews in the list");
});
```

Things to note:

- Read the comments carefully, and make sure you understand what the various sections do.
- First, we define our query: we are selecting from our "Reviews" collection, and sorting by book name.

- Then we request live updates by using the onSnapshot event (see here for more details). So later when we add new documents (new books) via other means (e.g. Firestore console or Goole Home Assistant), the list will be **refreshed automatically**.

4. Finally, add an **empty HTML table** (to display your results) and include your JavaScript file in **your index.html file**, like this:

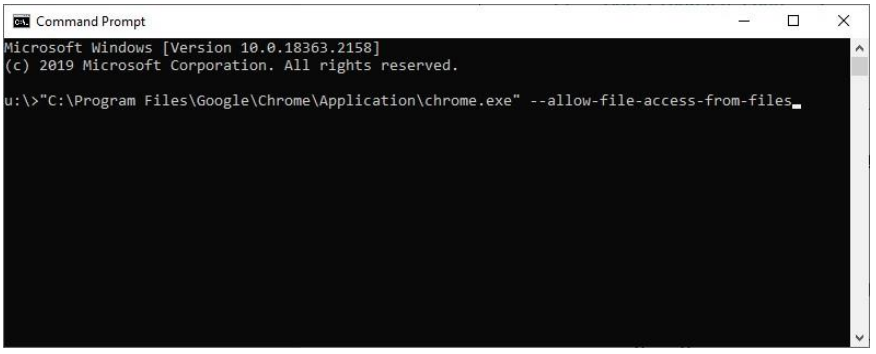
```
    <div class="container">
        <h1 id="mainTitle">My books</h1>
        <table class="table table-striped">
            <tbody id="reviewList">
            </tbody>
        </table>
    </div>

    <!-- jQuery -->

    <!-- Our Firebase code -->
    <script type="module" src="myscripts.js"></script>

  <!-- Bootstrap JavaScript -->
```

Things to note:

- **Important**: the bits in grey are already there!
- The id of the HTML table body (reviewList) is referenced in your JavaScript code, when looping through the data and "appending".
- For security reasons, including an external JS file is not allowed when browsing local files (e.g. URLs starting with C:/). Solutions include: ○ Don't use an external JS file 😉 Simply type your JS code in your HTML file, inside <script type="module"> block:

```
<!-- Our Firebase code -->
<script type="module">

    // Import required Firebase services
    import { initializeApp } from 'https://www.gstatic.com/firebas
    import { Firestore,
             getFirestore,
             onSnapshot,
```

  - Uploading your work on a server such as mi-linux, like I have done (more information here)
  - Starting Google Chrome from the command line with --allow-file-access-from-files parameters, like this:



5. All done. Browse to your file, and you should see a nice HTML table displaying your data from Part 2. Here is mine for reference: https://mi-linux.wlv.ac.uk/~in9352/google-2023/firebase-client-02.html (remember you can view the full source code in the console!)

Try **adding new reviews via the Firestore console** (as per Part 2) and see how your web page **is refreshed automatically**!

Part 4 – Adding more data from JavaScript

Finally, we would like to be able to **add new reviews directly from our web page**.

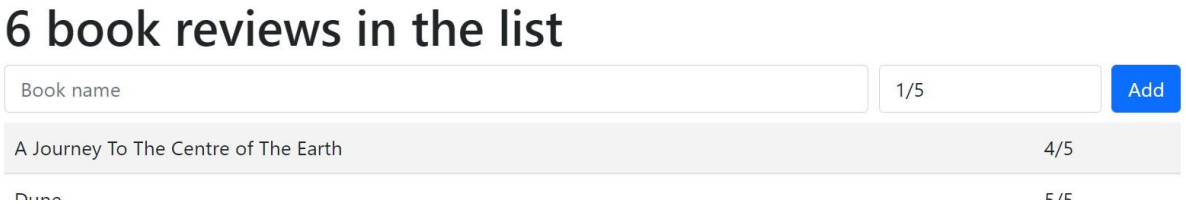
1. **Add** the following HTML in **index.html**, just **above your HTML table**:

```
<div class="d-flex mb-2">
  <input type="text" class="form-control" id="bookName" placeholder="Book name">
  <select class="form-control mx-2 w-25" id="bookRating">
    <option value="1">1/5</option>
    <option value="2">2/5</option>
    <option value="3">3/5</option>
    <option value="4">4/5</option>
    <option value="5">5/5</option>
  </select>
  <button type="button" class="btn btn-primary" id="addButton">Add</button>
</div>
```

Things to note:

- The first element is a text box, for the user to enter the name of the book. - The second element is a dropdown, with rating values from 1 to 5 - The third element is a button that will trigger the JS code below.

Here's what the top of our page now looks like. The button doesn't do anything just yet!



2. **Add** the following code in your existing **myscripts.js** file:

```
// Add button pressed
$("#addButton").click(function() {
```

```
        // Add review to Firestore collection
        const docRef = addDoc(collection(db, "Reviews"), {
          book_name: $("#bookName").val(),
          book_rating: parseInt($("#bookRating").val())
        });

        // Reset form
        $("#bookName").val('');
        $("#bookRating").val('1');
});
```

Things to note:

- The first line detects that the user has clicked the "add" button.
- The next block of code **adds a new document** to our existing "Reviews" collection, grabbing the values from our text box and dropdown list. Note the parseInt, required here to store the rating as a number (rather than a string). -More on adding documents here.

3. All done! You should now be able to add reviews to your list (which will be refreshed automatically). Here is mine for reference: https://mi-linux.wlv.ac.uk/~in9352/google-2023/firebase-client-03.html

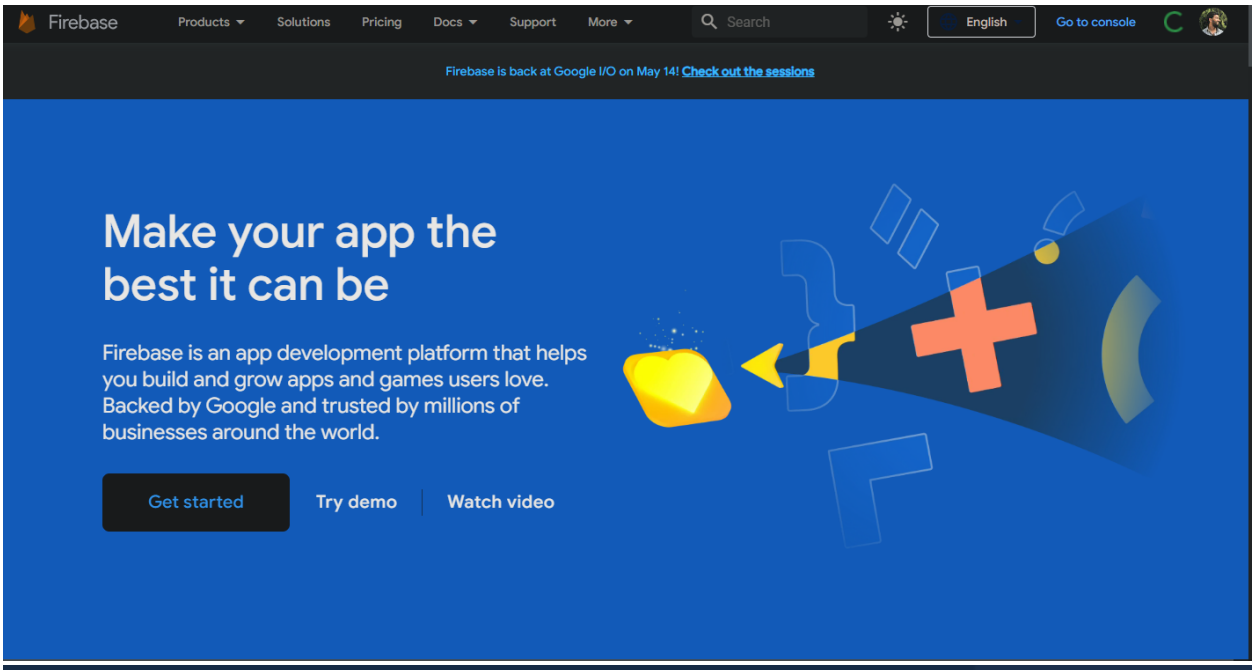
Note: Something not working? Make sure you have your browser's Developer Tools and console open, as the error messages are often very helpful.
Important: The security of our application is a bit flaky at the moment… here is a good place to get started if you want to know more: https://cloud.google.com/firestore/docs/security/get-started

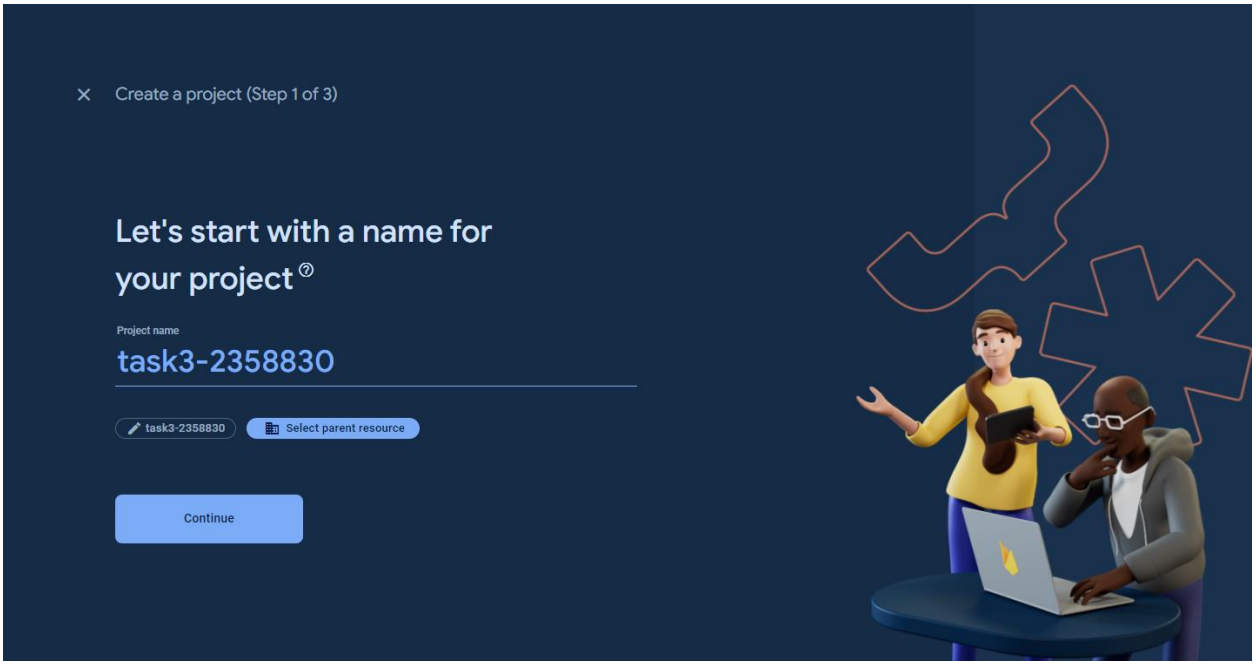
Part 5 – Optional work (if you want to explore this topic further)

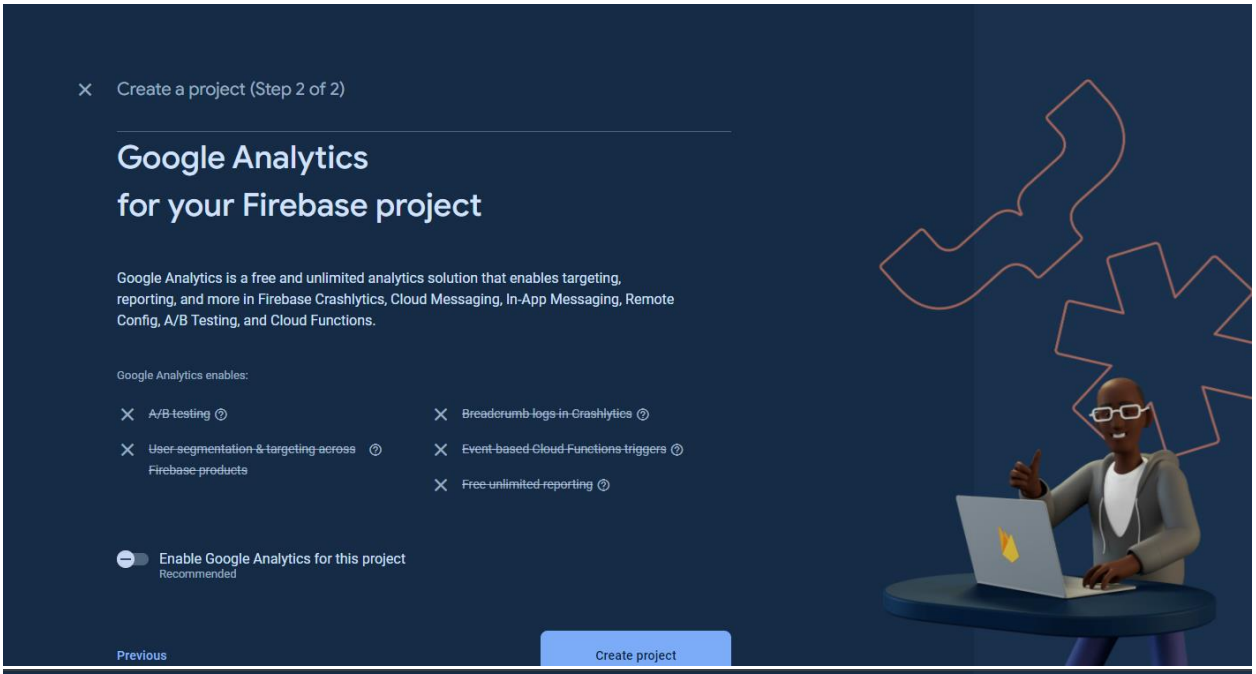
Add the following features to your application:

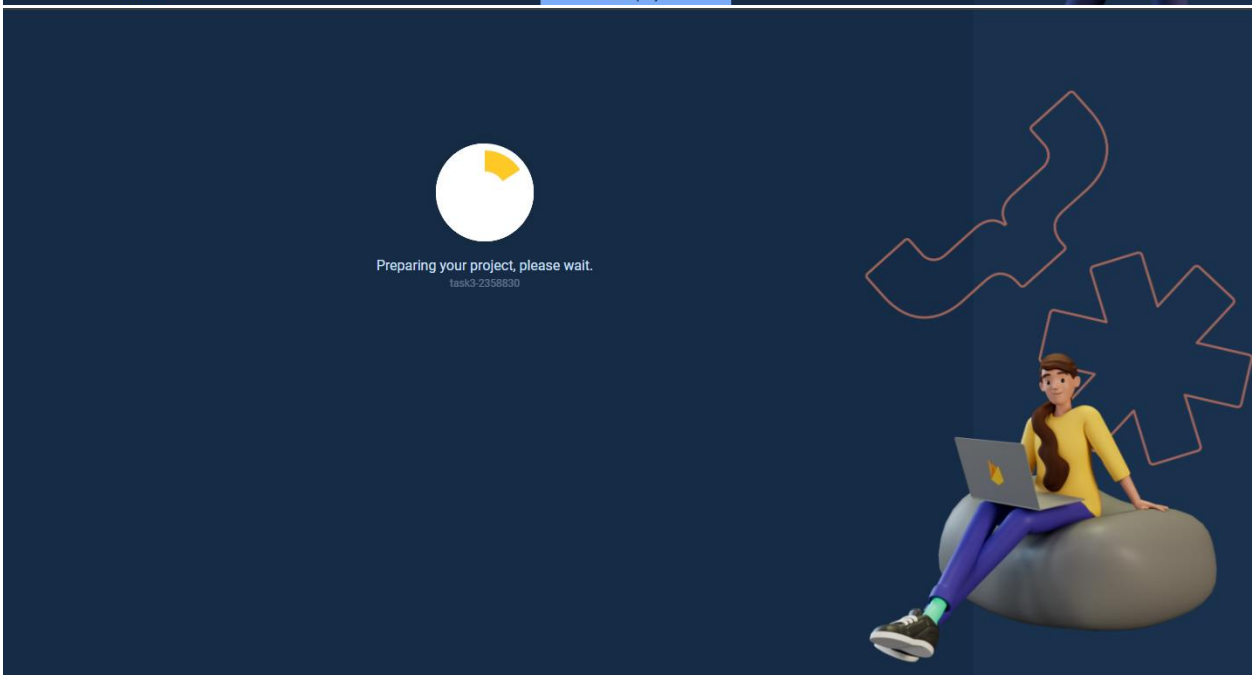
1. Add a "review_date" field to your existing documents and display it in your HTML table as a third column.
    - When **adding** new reviews, the field should be set automatically to the **current date and time**.

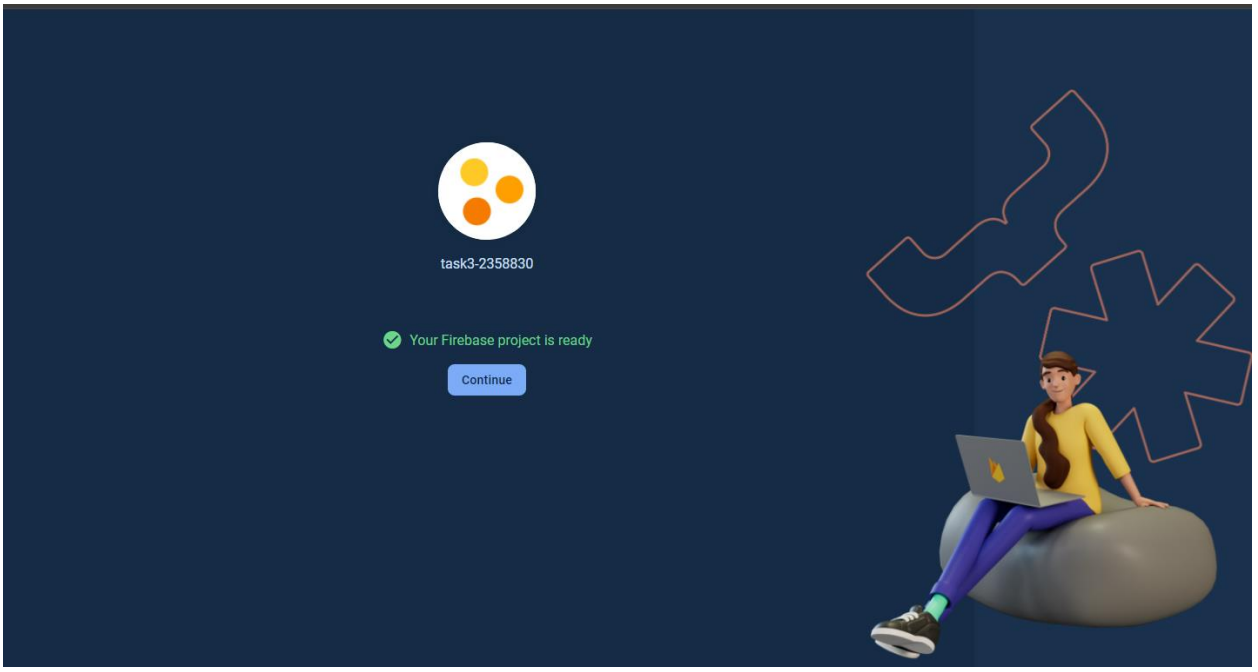
2. Add the ability to delete reviews from the web interface.
    - Here is a "how-to" guide on how to delete a Firestore document: https://cloud.google.com/firestore/docs/manage-data/delete-data
    - The trick will be to pass the correct document ID. You could store the document ID for each document in your HTML table when displaying it, in order to retrieve it when trying to delete an item.

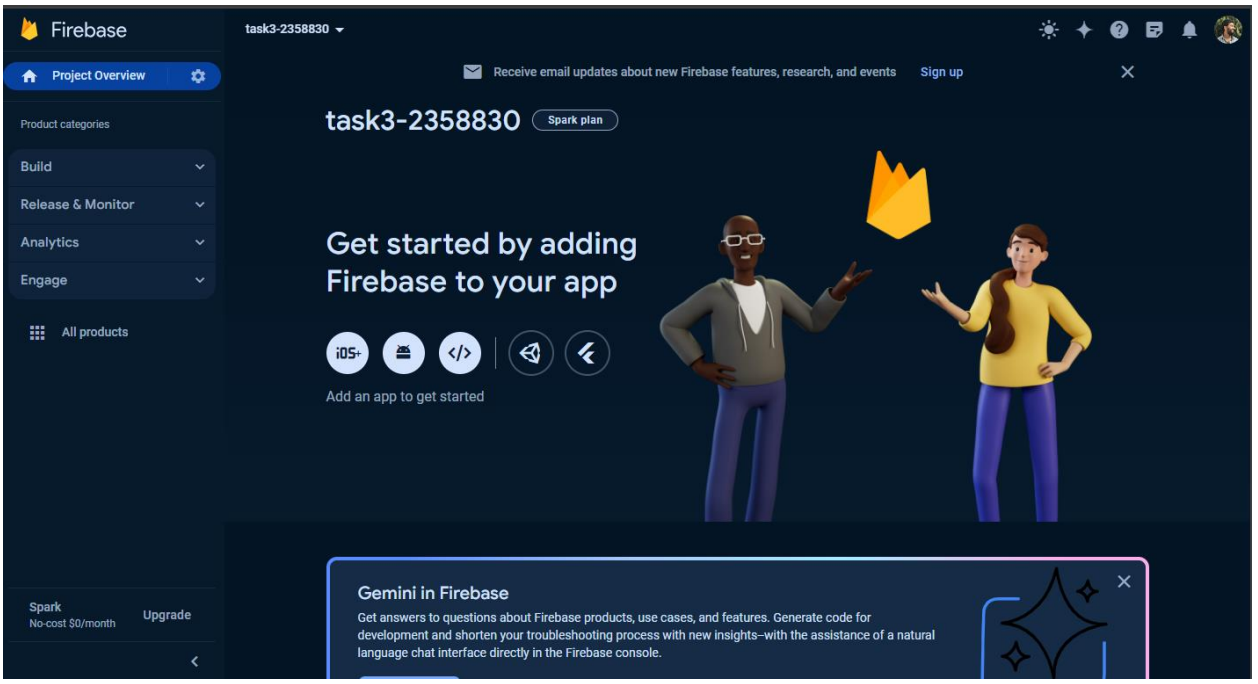

Firebase
Products
Solutions
Pricing
Docs
Support
More
Search
English
Go to console
Firebase is back at Google I/O on May 14! Check out the sessions
Make your app the best it can be
Firebase is an app development platform that helps you build and grow apps and games users love. Backed by Google and trusted by millions of businesses around the world.
Get started
Try demo
Watch video


Create a project (Step 1 of 3)
Let's start with a name for your project
Project name
task3-2358830
task3-2358830
Select parent resource
Continue


Create a project (Step 2 of 2)
Google Analytics
for your Firebase project
Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, and Cloud Functions.
Google Analytics enables:
A/B testing
User segmentation & targeting across Firebase products
Breadcrumb logs in Crashlytics
Event-based Cloud Functions triggers
Free unlimited reporting
Enable Google Analytics for this project
Recommended
Previous
Create project


Preparing your project, please wait.
task3-2358830


task3-2358830
Your Firebase project is ready
Continue


Firebase
task3-2358830
Project Overview
Product categories
Build
Release & Monitor
Analytics
Engage
All products
Spark
No-cost $0/month
Upgrade
Receive email updates about new Firebase features, research, and events
Sign up
task3-2358830
Spark plan
Get started by adding
Firebase to your app
iOS+
Add an app to get started
Gemini in Firebase
Get answers to questions about Firebase products, use cases, and features. Generate code for development and shorten your troubleshooting process with new insights–with the assistance of a natural language chat interface directly in the Firebase console.

Go to docs

Add Firebase to your web app

1 Register app

App nickname

2358830_Movie

Also set up **Firebase Hosting** for this app. Learn more

Hosting can also be set up later. There is no cost to get started anytime.

Register app

2 Add Firebase SDK

Go to docs

2 Add Firebase SDK

Use npm Use a <script> tag

If you're already using npm and a module bundler such as webpack or Rollup, you can run the following command to install the latest SDK (Learn more):

```
$ npm install firebase
```

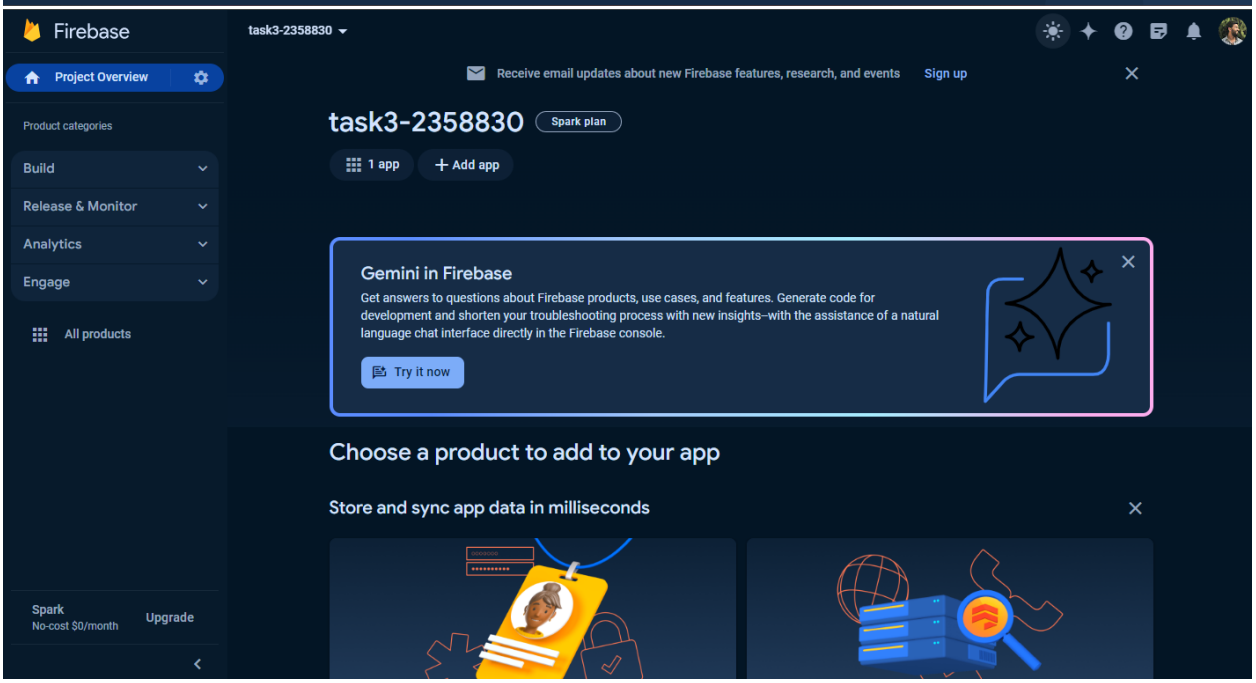
Then, initialize Firebase and begin using the SDKs for the products you'd like to use.

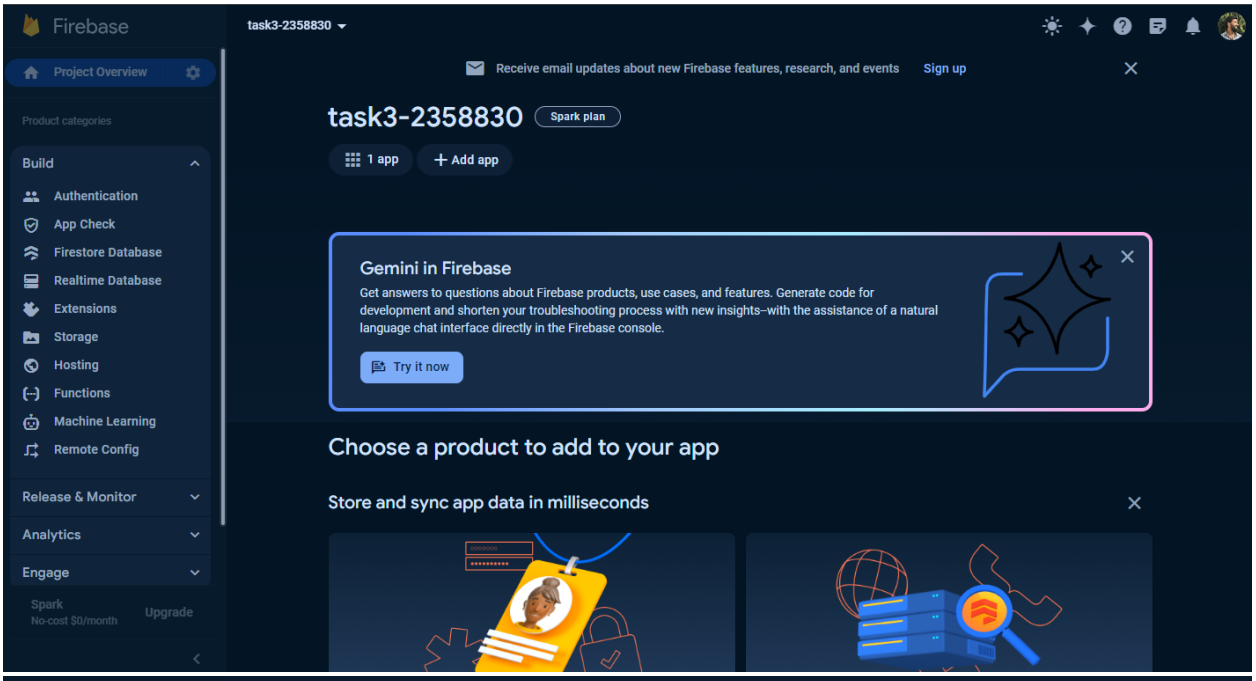
```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyBXyDEvq-DchvxBEdRlwZeJvrFB9P81EGg",
  authDomain: "task3-2358830.firebaseapp.com",
  projectId: "task3-2358830",
  storageBucket: "task3-2358830.appspot.com",
  messagingSenderId: "657959837717",
  appId: "1:657959837717:web:66eccae514f2ec75592e1c"
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
```

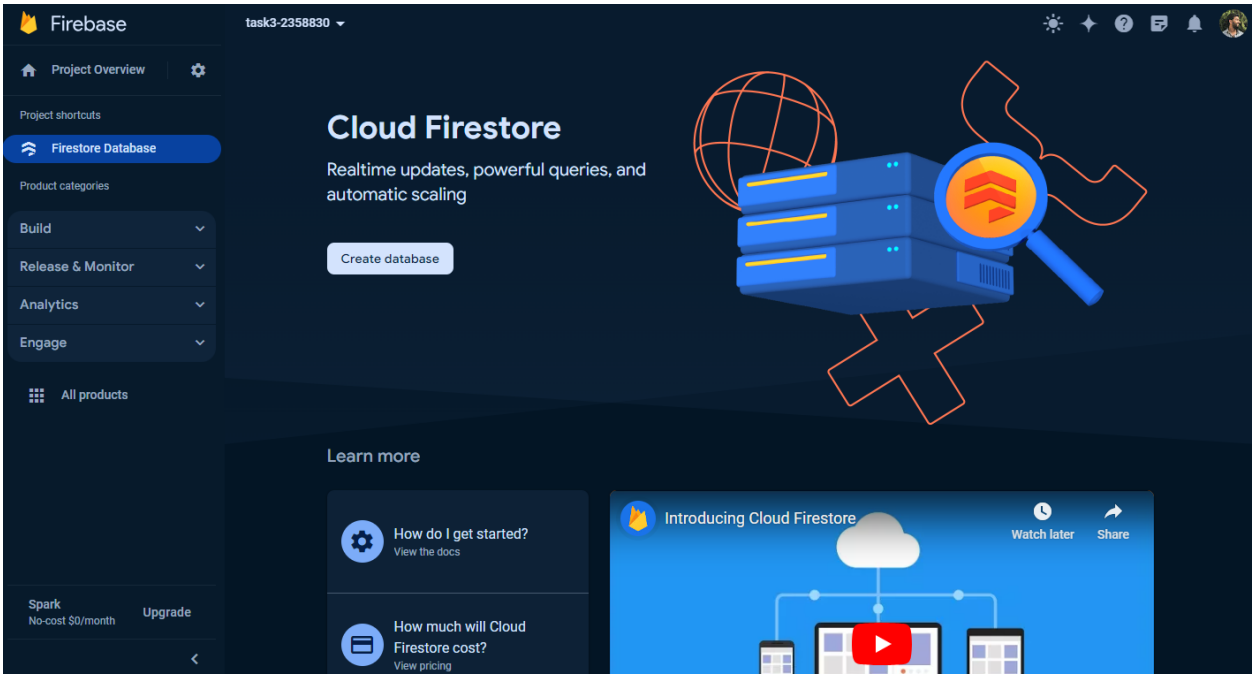
Note: This option uses the modular JavaScript SDK, which provides reduced SDK size.

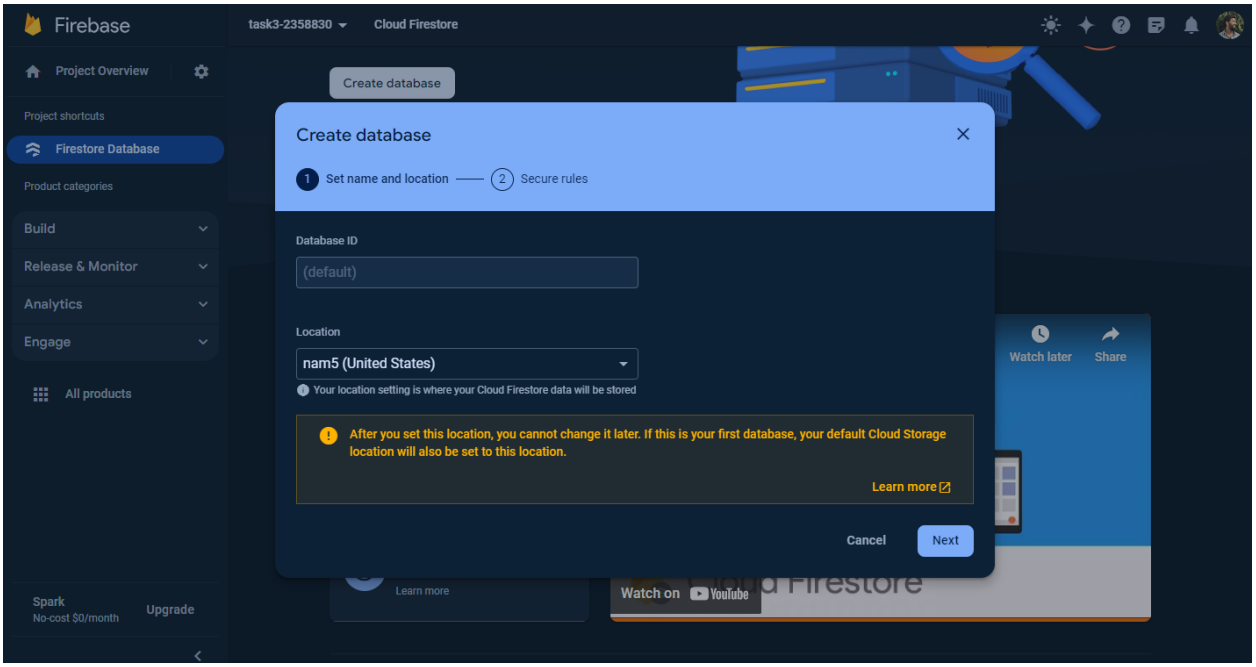
Learn more about Firebase for web: Get Started, Web SDK API Reference, Samples

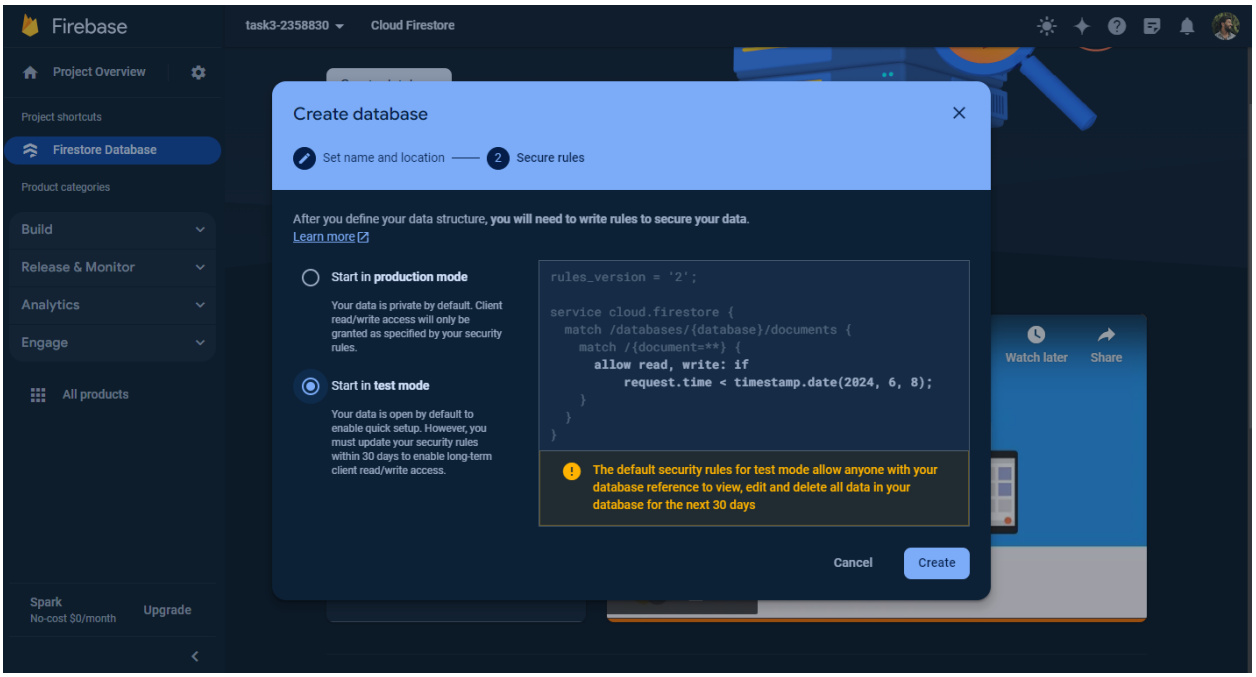

Go to docs
Add Firebase to your web app
Register app
Add Firebase SDK
Use npm
Use a <script> tag
If you don't use build tools, use this option to add and use the Firebase JS SDK. Use this option to get started, but it's not recommended for production apps. Learn more.
Copy and paste these scripts into the bottom of your <body> tag, but before you use any Firebase services:
<script type="module">
// Import the functions you need from the SDKs you need
import { initializeApp } from "https://www.gstatic.com/firebasejs/10.11.1/fi
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries
// Your web app's Firebase configuration
const firebaseConfig = {
apiKey: "AIzaSyBXyDEvq-DchvxBEdRlwZeJvrFB9P81EGg",
authDomain: "task3-2358830.firebaseapp.com",
projectId: "task3-2358830",
storageBucket: "task3-2358830.appspot.com",
messagingSenderId: "657959837717",
appId: "1:657959837717:web:66eccae514f2ec75592e1c"
};
// Initialize Firebase
const app = initializeApp(firebaseConfig);
</script>
Are you using npm and a bundler like webpack or Rollup? Check out the modular SDK.
Learn more about Firebase for web: Get Started, Web SDK API Reference, Samples
Continue to console

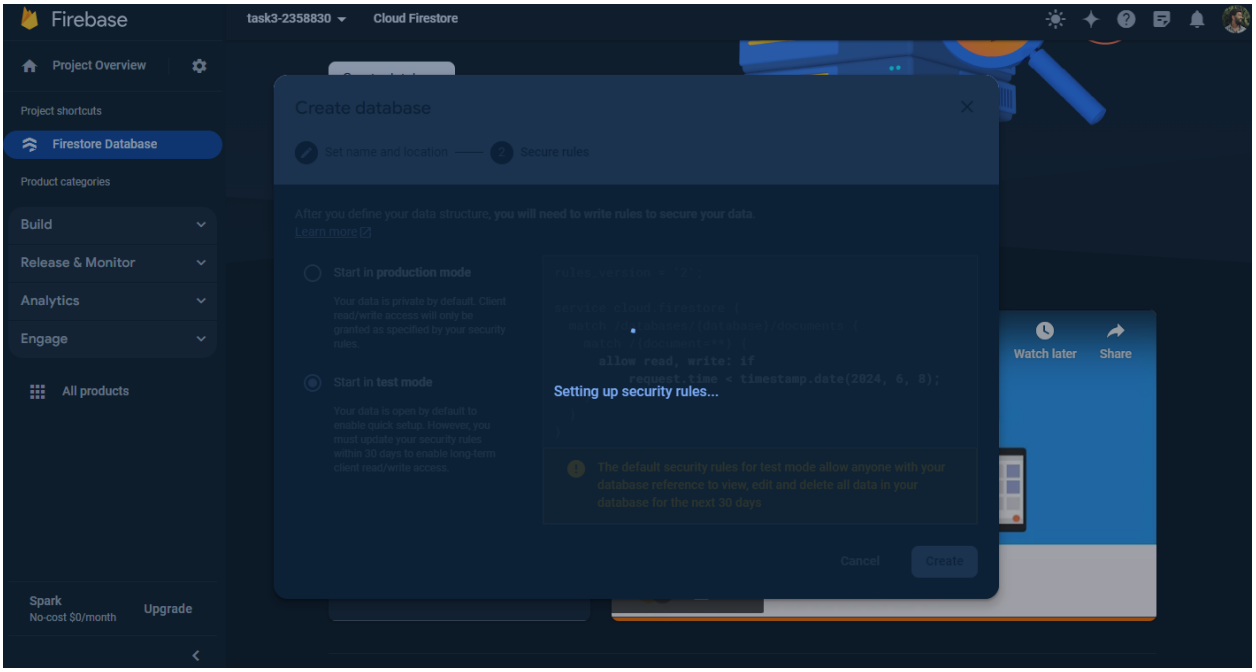

Firebase
task3-2358830
Project Overview
Product categories
Build
Release & Monitor
Analytics
Engage
All products
Spark
No-cost $0/month
Upgrade
Receive email updates about new Firebase features, research, and events
Sign up
task3-2358830
Spark plan
1 app
Add app
Gemini in Firebase
Get answers to questions about Firebase products, use cases, and features. Generate code for development and shorten your troubleshooting process with new insights–with the assistance of a natural language chat interface directly in the Firebase console.
Try it now
Choose a product to add to your app
Store and sync app data in milliseconds

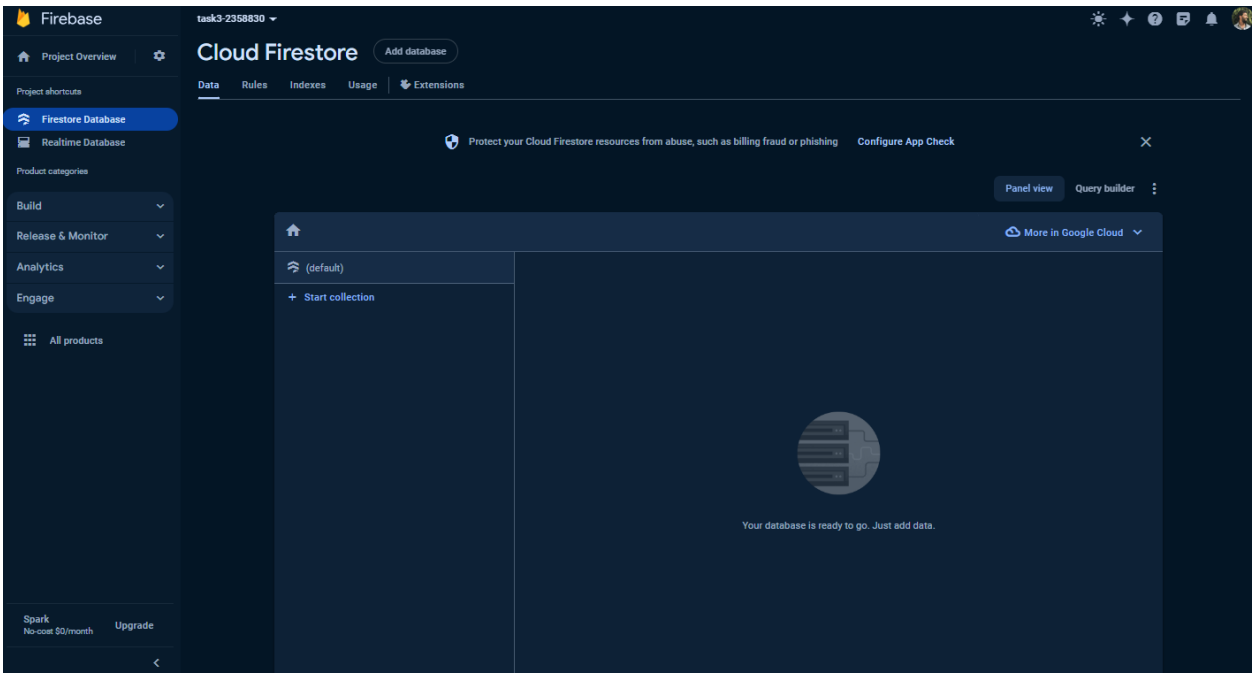

Firebase
task3-2358830
Project Overview
Product categories
Build
Authentication
App Check
Firestore Database
Realtime Database
Extensions
Storage
Hosting
Functions
Machine Learning
Remote Config
Release & Monitor
Analytics
Engage
Spark
No-cost $0/month
Upgrade
Receive email updates about new Firebase features, research, and events
Sign up
task3-2358830
Spark plan
1 app
Add app
Gemini in Firebase
Get answers to questions about Firebase products, use cases, and features. Generate code for development and shorten your troubleshooting process with new insights–with the assistance of a natural language chat interface directly in the Firebase console.
Try it now
Choose a product to add to your app
Store and sync app data in milliseconds

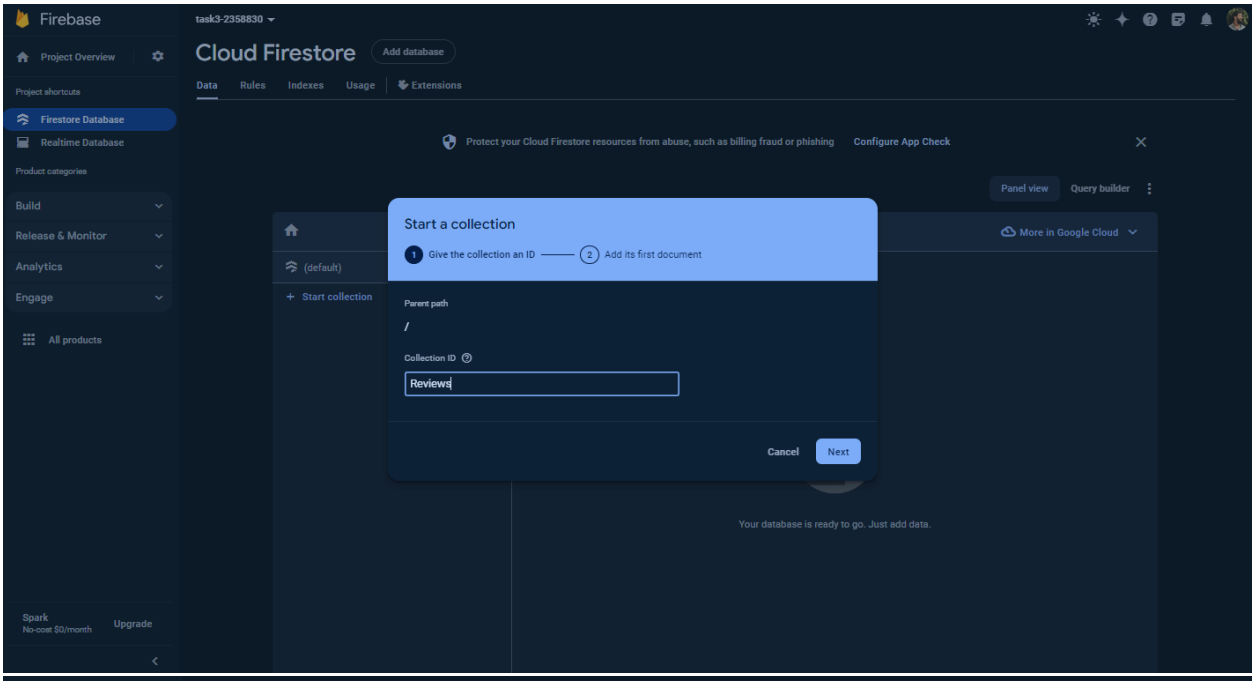

Firebase
task3-2358830
Project Overview
Project shortcuts
Firestore Database
Product categories
Build
Release & Monitor
Analytics
Engage
All products
Spark
No-cost $0/month
Upgrade
Cloud Firestore
Realtime updates, powerful queries, and automatic scaling
Create database
Learn more
How do I get started?
View the docs
How much will Cloud Firestore cost?
View pricing
Introducing Cloud Firestore
Watch later
Share

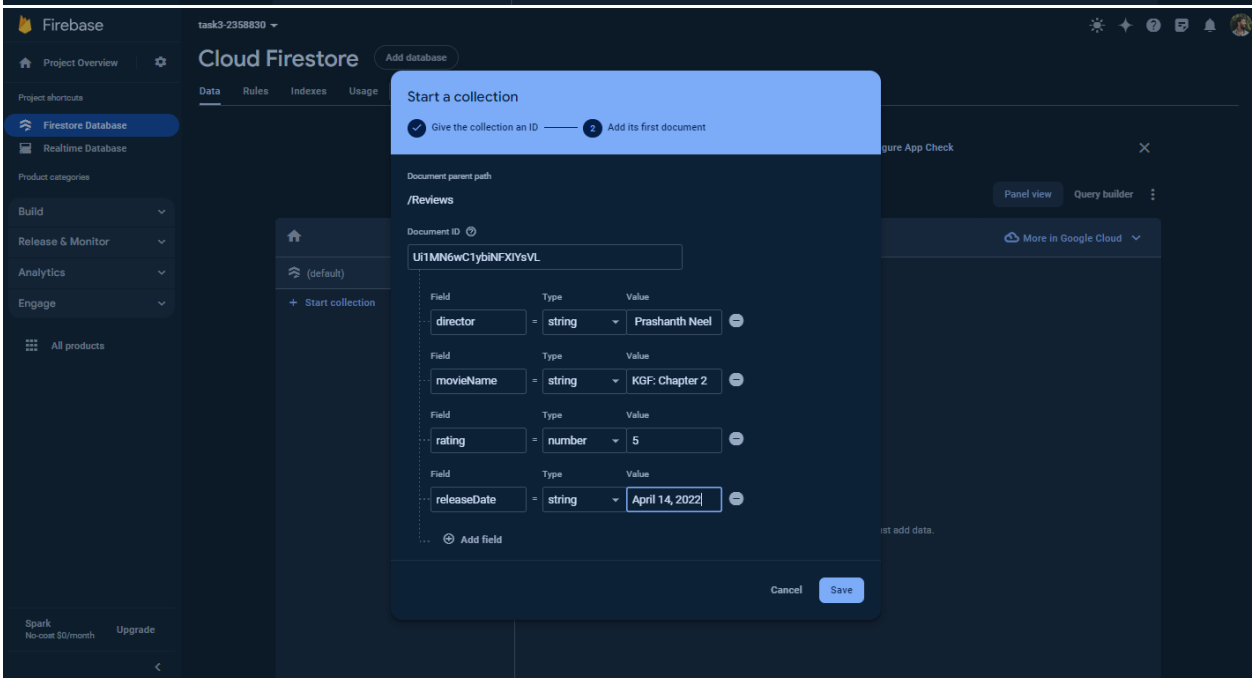

Firebase
task3-2358830
Cloud Firestore
Project Overview
Project shortcuts
Firestore Database
Product categories
Build
Release & Monitor
Analytics
Engage
All products
Spark
No-cost $0/month
Upgrade
Create database
Create database
1 Set name and location
2 Secure rules
Database ID
(default)
Location
nam5 (United States)
Your location setting is where your Cloud Firestore data will be stored
After you set this location, you cannot change it later. If this is your first database, your default Cloud Storage location will also be set to this location.
Learn more
Cancel
Next
Watch later
Share
Watch on YouTube

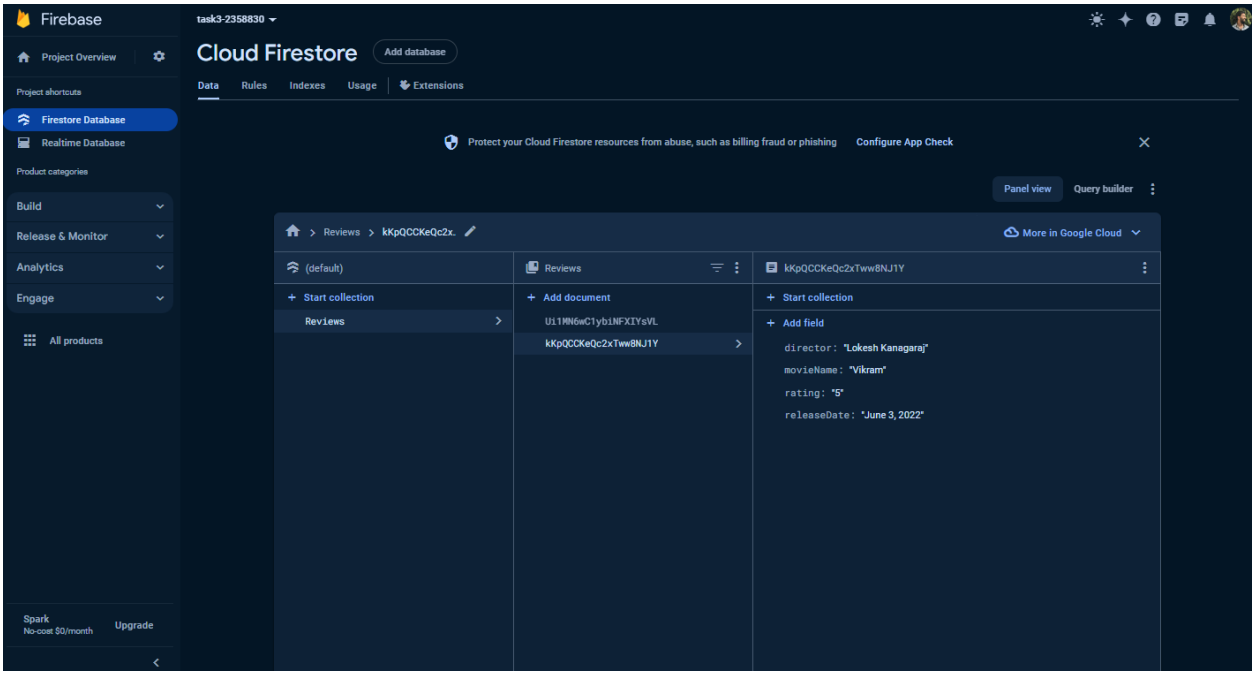

Firebase
task3-2358830
Cloud Firestore
Project Overview
Project shortcuts
Firestore Database
Product categories
Build
Release & Monitor
Analytics
Engage
All products
Spark
No-cost $0/month
Upgrade
Create database
Set name and location
2 Secure rules
After you define your data structure, you will need to write rules to secure your data.
Learn more
Start in production mode
Your data is private by default. Client read/write access will only be granted as specified by your security rules.
Start in test mode
Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.
rules_version = '2';
service cloud.firestore {
match /databases/{database}/documents {
match /{document=**} {
allow read, write: if
request.time < timestamp.date(2024, 6, 8);
}
}
}
The default security rules for test mode allow anyone with your database reference to view, edit and delete all data in your database for the next 30 days
Cancel
Create
Watch later
Share


Firebase
task3-2358830
Cloud Firestore
Project Overview
Project shortcuts
Firestore Database
Product categories
Build
Release & Monitor
Analytics
Engage
All products
Spark
No-cost $0/month
Upgrade
Create database
Set name and location
Secure rules
After you define your data structure, you will need to write rules to secure your data.
Learn more
Start in production mode
Your data is private by default. Client read/write access will only be granted as specified by your security rules.
Start in test mode
Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.
Setting up security rules...
The default security rules for test mode allow anyone with your database reference to view, edit and delete all data in your database for the next 30 days
Cancel
Create
Watch later
Share


Firebase
task3-2358830
Project Overview
Project shortcuts
Firestore Database
Realtime Database
Product categories
Build
Release & Monitor
Analytics
Engage
All products
Spark
No-cost $0/month
Upgrade
Cloud Firestore
Add database
Data
Rules
Indexes
Usage
Extensions
Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing
Configure App Check
Panel view
Query builder
More in Google Cloud
(default)
Start collection
Your database is ready to go. Just add data.


Firebase
task3-2358830
Project Overview
Cloud Firestore
Add database
Project shortcuts
Data
Rules
Indexes
Usage
Extensions
Firestore Database
Realtime Database
Product categories
Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing
Configure App Check
Build
Release & Monitor
Analytics
Engage
Panel view
Query builder
All products
More in Google Cloud
(default)
Start collection
Start a collection
1 Give the collection an ID
2 Add its first document
Parent path
/
Collection ID
Reviews
Cancel
Next
Your database is ready to go. Just add data.
Spark
No-cost $0/month
Upgrade


Firebase
task3-2358830
Project Overview
Cloud Firestore
Add database
Project shortcuts
Data
Rules
Indexes
Usage
Firestore Database
Realtime Database
Product categories
Start a collection
Give the collection an ID
2 Add its first document
Build
Release & Monitor
Analytics
Engage
Panel view
Query builder
All products
More in Google Cloud
(default)
Start collection
Document parent path
/Reviews
Document ID
Ui1MN6wC1ybiNFXIYsVL
Field
Type
Value
director
string
Prashanth Neel
movieName
string
KGF: Chapter 2
rating
number
5
releaseDate
string
April 14, 2022
Add field
Cancel
Save
Spark
No-cost $0/month
Upgrade




```
<!doctype html>
<html lang="en">
<head>
<!-- Required meta tags -->
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<!-- Bootstrap CSS -->
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.c
ss" rel="stylesheet"></head>
<title>My Firebase app</title>
</head>
<body>
<div class="container">
<h1 id="mainTitle">My Movie Review App</h1>
<div class="d-flex mb-2">
<input type="text" class="form-control" id="bookName" placeholder="Book name">
<!-- <input type="text" class="form-control" id="directorName"
placeholder="Director name">
<input type="text" class="form-control" id="releaseDate" placeholder="Release
date"> -->
<select class="form-control mx-2 w-25" id="bookRating">
<option value="1">1/5</option>
<option value="2">2/5</option>
<option value="3">3/5</option>
<option value="4">4/5</option>
```

```
<option value="5">5/5</option>
</select>
<button type="button" class="btn btn-primary" id="addButton">Add</button>
</div>

<table class="table table-striped">
<tbody id="reviewList">
</tbody>
</table>
</div>

<!-- jQuery -->
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<!-- Bootstrap JavaScript -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.
min.js"></script>
</body>
</html>

<script type="module">
import { initializeApp } from
"https://www.gstatic.com/firebasejs/9.18.0/firebase-app.js";
import { Firestore,
getFirestore,
onSnapshot,
query,
collection,
orderBy,
addDoc } from
'https://www.gstatic.com/firebasejs/9.18.0/firebase-firestore.js'
// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyB9P1akYdqGbnUdYzV-FOSJtDlmcbqQl2A",
  authDomain: "cs022-2358830-sumitshah.firebaseapp.com",
  projectId: "cs022-2358830-sumitshah",
  storageBucket: "cs022-2358830-sumitshah.appspot.com",
  messagingSenderId: "225477100760",
  appId: "1:225477100760:web:bac46662a971359af343ad"
};
// Initialize Firebase
const app = initializeApp(firebaseConfig);
const db = getFirestore(app);
// Get a live data snapshot (i.e. auto-refresh) of our Reviews collection
const q = query(collection(db, "Reviews"), orderBy("book_name"));
const unsubscribe = onSnapshot(q, (snapshot) => {
// Empty HTML table
$('#reviewList').empty();
```

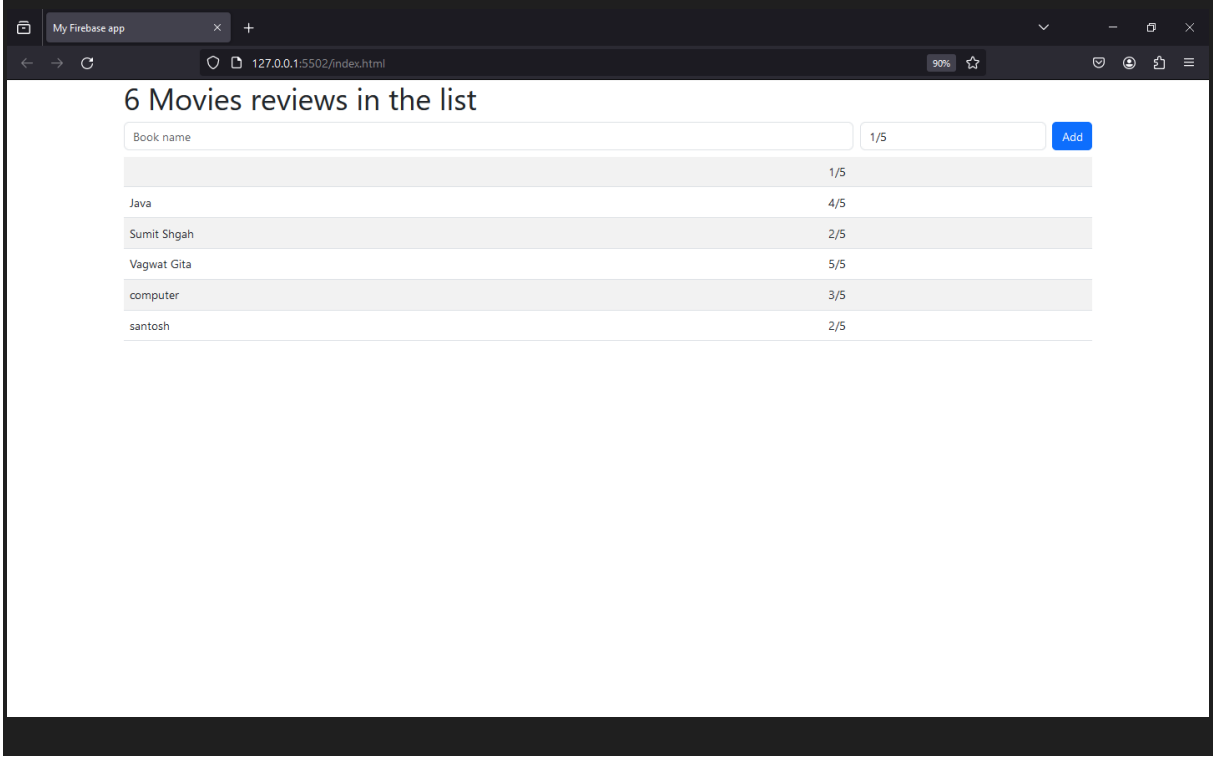
```
// Loop through snapshot data and add to HTML table
var tableRows = '';
snapshot.forEach((doc) => {
tableRows += '<tr>';
tableRows += '<td>' + doc.data().book_name + '</td>';
tableRows += '<td>' + doc.data().book_rating + '/5</td>';
tableRows += '</tr>';
});
$('#reviewList').append(tableRows);
// Display review count
$('#mainTitle').html(snapshot.size + " Movies reviews in the list");
});

$("#addButton").click(function() {
// Add review to Firestore collection
const docRef = addDoc(collection(db, "Reviews"), {
book_name: $("#bookName").val(),
book_rating: parseInt($("#bookRating").val())
});
// Reset form
$("#bookName").val('');
$("#bookRating").val('1');
});
</script>
<!doctype html>
<html lang="en">
<head>
<!-- Required meta tags -->
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<!-- Bootstrap CSS -->
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.c
ss" rel="stylesheet"></head>
<title>My Firebase app</title>
</head>
<body>
<div class="container">
<h1 id="mainTitle">My Movie Review App</h1>
<div class="d-flex mb-2">
<input type="text" class="form-control" id="bookName" placeholder="Book name">
<!-- <input type="text" class="form-control" id="directorName"
placeholder="Director name">
<input type="text" class="form-control" id="releaseDate" placeholder="Release
date"> -->
<select class="form-control mx-2 w-25" id="bookRating">
<option value="1">1/5</option>
<option value="2">2/5</option>
```

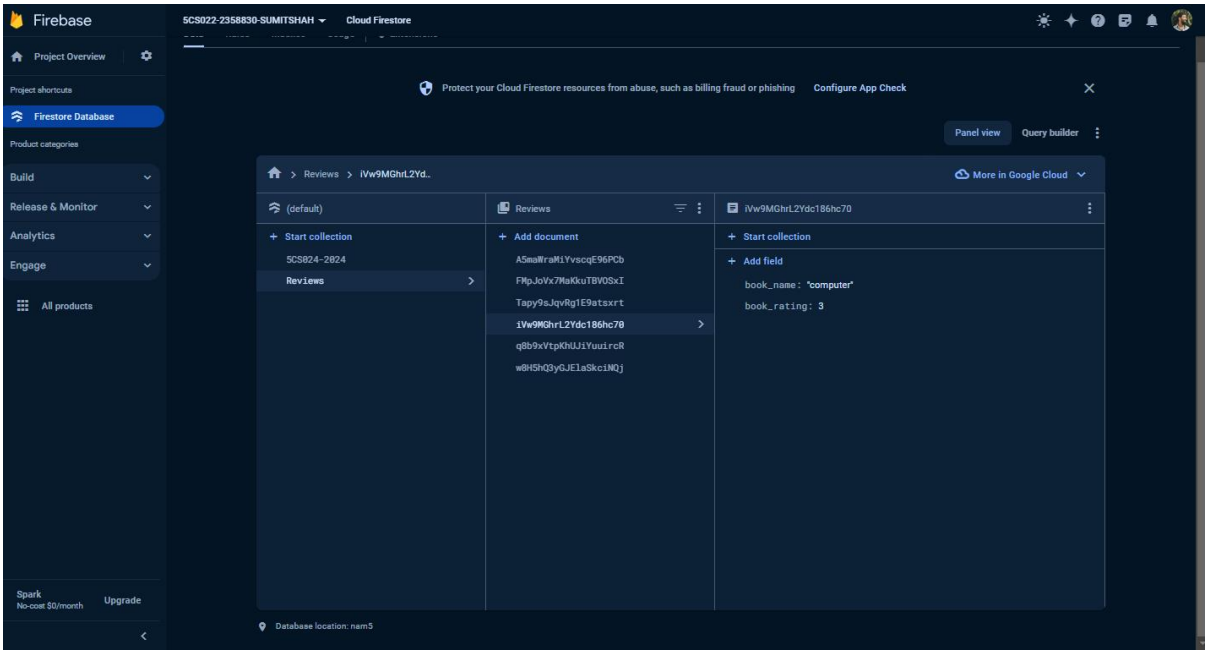
```
<option value="3">3/5</option>
<option value="4">4/5</option>
<option value="5">5/5</option>
</select>
<button type="button" class="btn btn-primary" id="addButton">Add</button>
</div>

<table class="table table-striped">
<tbody id="reviewList">
</tbody>
</table>
</div>

<!-- jQuery -->
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<!-- Bootstrap JavaScript -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.
min.js"></script>
</body>
</html>

<script type="module">
import { initializeApp } from
"https://www.gstatic.com/firebasejs/9.18.0/firebase-app.js";
import { Firestore,
getFirestore,
onSnapshot,
query,
collection,
orderBy,
addDoc } from
'https://www.gstatic.com/firebasejs/9.18.0/firebase-firestore.js'
// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyB9P1akYdqGbnUdYzV-FOSJtDlmcbqQl2A",
  authDomain: "cs022-2358830-sumitshah.firebaseapp.com",
  projectId: "cs022-2358830-sumitshah",
  storageBucket: "cs022-2358830-sumitshah.appspot.com",
  messagingSenderId: "225477100760",
  appId: "1:225477100760:web:bac46662a971359af343ad"
};
// Initialize Firebase
const app = initializeApp(firebaseConfig);
const db = getFirestore(app);
// Get a live data snapshot (i.e. auto-refresh) of our Reviews collection
const q = query(collection(db, "Reviews"), orderBy("book_name"));
const unsubscribe = onSnapshot(q, (snapshot) => {
```

```
// Empty HTML table
$('#reviewList').empty();
// Loop through snapshot data and add to HTML table
var tableRows = '';
snapshot.forEach((doc) => {
tableRows += '<tr>';
tableRows += '<td>' + doc.data().book_name + '</td>';
tableRows += '<td>' + doc.data().book_rating + '/5</td>';
tableRows += '</tr>';
});
$('#reviewList').append(tableRows);
// Display review count
$('#mainTitle').html(snapshot.size + " Movies reviews in the list");
});

$("#addButton").click(function() {
// Add review to Firestore collection
const docRef = addDoc(collection(db, "Reviews"), {
book_name: $("#bookName").val(),
book_rating: parseInt($("#bookRating").val())
});
// Reset form
$("#bookName").val('');
$("#bookRating").val('1');
});
</script>
<!doctype html>
<html lang="en">
<head>
<!-- Required meta tags -->
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<!-- Bootstrap CSS -->
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/css/bootstrap.min.c
ss" rel="stylesheet"></head>
<title>My Firebase app</title>
</head>
<body>
<div class="container">
<h1 id="mainTitle">My Movie Review App</h1>
<div class="d-flex mb-2">
<input type="text" class="form-control" id="bookName" placeholder="Book name">
<!-- <input type="text" class="form-control" id="directorName"
placeholder="Director name">
<input type="text" class="form-control" id="releaseDate" placeholder="Release
date"> -->
<select class="form-control mx-2 w-25" id="bookRating">
```

```
<option value="1">1/5</option>
<option value="2">2/5</option>
<option value="3">3/5</option>
<option value="4">4/5</option>
<option value="5">5/5</option>
</select>
<button type="button" class="btn btn-primary" id="addButton">Add</button>
</div>

<table class="table table-striped">
<tbody id="reviewList">
</tbody>
</table>
</div>

<!-- jQuery -->
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
<!-- Bootstrap JavaScript -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-
alpha1/dist/js/bootstrap.bundle.
min.js"></script>
</body>
</html>

<script type="module">
import { initializeApp } from
"https://www.gstatic.com/firebasejs/9.18.0/firebase-app.js";
import { Firestore,
getFirestore,
onSnapshot,
query,
collection,
orderBy,
addDoc } from
'https://www.gstatic.com/firebasejs/9.18.0/firebase-firestore.js'
// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyB9P1akYdqGbnUdYzV-FOSJtDlmcbqQl2A",
  authDomain: "cs022-2358830-sumitshah.firebaseapp.com",
  projectId: "cs022-2358830-sumitshah",
  storageBucket: "cs022-2358830-sumitshah.appspot.com",
  messagingSenderId: "225477100760",
  appId: "1:225477100760:web:bac46662a971359af343ad"
};
// Initialize Firebase
const app = initializeApp(firebaseConfig);
const db = getFirestore(app);
// Get a live data snapshot (i.e. auto-refresh) of our Reviews collection
```

```
const q = query(collection(db, "Reviews"), orderBy("book_name"));
const unsubscribe = onSnapshot(q, (snapshot) => {
// Empty HTML table
$('#reviewList').empty();
// Loop through snapshot data and add to HTML table
var tableRows = '';
snapshot.forEach((doc) => {
tableRows += '<tr>';
tableRows += '<td>' + doc.data().book_name + '</td>';
tableRows += '<td>' + doc.data().book_rating + '/5</td>';
tableRows += '</tr>';
});
$('#reviewList').append(tableRows);
// Display review count
$('#mainTitle').html(snapshot.size + " Movies reviews in the list");
});

$("#addButton").click(function() {
// Add review to Firestore collection
const docRef = addDoc(collection(db, "Reviews"), {
book_name: $("#bookName").val(),
book_rating: parseInt($("#bookRating").val())
});
// Reset form
$("#bookName").val('');
$("#bookRating").val('1');
});
</script>
```




Firebase
5CS022-2358830-SUMITSHAH
Cloud Firestore
Project Overview
Project shortcuts
Firestore Database
Product categories
Build
Release & Monitor
Analytics
Engage
All products
Spark
No-cost $0/month
Upgrade
Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing
Configure App Check
Panel view
Query builder
Reviews
iVw9MGhrL2Yd..
More in Google Cloud
(default)
Start collection
5CS024-2024
Reviews
Reviews
Add document
A5maWraMiYvscqE96PCb
FMpJoVx7MaKkuTBVOSxI
Tapy9sJqvRg1E9atsxrt
iVw9MGhrL2Ydc186hc70
q8b9xVtpKhUJiYuuircR
w8H5hQ3yGJElaSkciNQj
iVw9MGhrL2Ydc186hc70
Start collection
Add field
book_name: "computer"
book_rating: 3
Database location: nam5