

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:

The screenshot shows a web form titled "Cloud Platform Education Grants". It starts with a brief description: "Use credits provided to you via the Google Cloud Platform Education Grants program to access Google Cloud Platform. Get what you need to build and run your apps, websites and services." Below this, there's a message: "Thank you for your interest in Google Cloud Platform Education Grants. Please fill out the form below to receive a coupon code for credit to use on Google Cloud Platform." The form fields include: "First Name" (Alix), "Last Name" (Bergeret), "School Email" (alix.bergeret@wlv.ac.uk), and a note: "If you do not see your domain listed, please contact your course instructor [J.Ting@wlv.ac.uk](#)". At the bottom, there's a note about privacy: "By clicking "Submit" below, you agree that we may share the following information with your educational institution and course instructor ([J.Ting@wlv.ac.uk](#)): (1) personal information that you provide to us on this form and (2) information regarding your use of the coupon and Google Cloud Platform products." A blue "Submit" button is at the bottom right.

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:

The screenshot shows the "GCP credit application" page. At the top, it says "Fill in the following information below to apply GCP credits to your account listed below." The form fields are: "First name *" (Alix), "Surname *" (Bergeret), "Account email" (alix.bergeret@gmail.com), and "Coupon code *" (0CC4-JMW0-Q01L-K94L). Below the form, there's a section titled "Terms and conditions" with the note: "The following Terms of Service [apply to the credit you received for Google Cloud products.](#)" A blue "ACCEPT AND CONTINUE" button is at the bottom left. A small note at the bottom left says: "* Indicates required".

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:

The screenshot shows the Google Cloud Platform Billing Overview page. On the left, a sidebar lists navigation options: Overview, Reports, Cost table, Cost breakdown, Commitments, Commitment analysis, Budgets & alerts, Billing export, Pricing, and Account management. The main content area displays the 'Current month' (1-21 March 2022) with a total cost of US\$0.00. It also shows the 'End-of-month total cost (forecasted)' which is also US\$0.00. A note states 'Not enough historical data to project cost.' Below this is a 'Cost trend' section for the period 1 March 2022 – 31 March 2022, showing an average monthly total cost of US\$0.00. To the right, there's a 'Billing account Manage' section with the account name 'Billing Account for Education, 0141F0-D6Ec7-DCTBF'. It also shows 'Organisation' information: 'No organisation'. Under 'Billing health checks', it says 'Take a look at your account health results to avoid common billing-related issues and adopt our best practice recommendations.' It lists three items: 1 error (red), 2 warnings (yellow), and 0 successes (green). A 'View all health checks' link is provided. Finally, the 'Credits' section shows a balance of US\$50.00, with a note that 'Remaining credits' are 'Out of US\$50.00'. A 'Remaining credits' table shows 'BSc Computer Science' with a value of '\$50.00'.

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.firebaseio.google.com/>

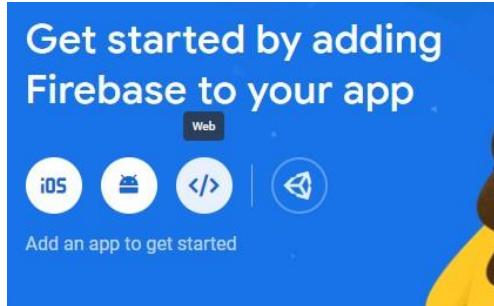
- Press the “Add project” button.
- On the following screen, **enter a new project name (maybe “5CS022” + your student number)** and press “Continue”.
- On the “Google Analytics”, **deselect** “Enable Google Analytics for this project”, as we don’t need it for this workshop. Press “Continue”.
- Your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

The screenshot shows the 'Create a project(Step 1 of 3)' screen. It asks 'Let's start with a name for your project®' and has a text input field containing '5CS022-demo-2022'. Below the input field is a note 'ca022-demo-2022'. A 'Continue' button is at the bottom right.

The screenshot shows the 'Project Overview' screen for the '5CS022-demo-2022' project. The sidebar includes sections for Build (Authentication, Firestore Database, Realtime Database, Storage, Functions, Machine Learning), Release and monitor (Crashlytics, Performance, Test Lab, App Distribution), and Analytics (Dashboard, Heatmap, Events, Extensions). The main content area shows a 'Get started by adding Firebase to your app' section with a 'Add an app to get started' button. Below this are two cards: 'Store and sync app data in milliseconds' (Authenticaiton, Authentication) and 'Cloud Firestore' (Real-time updates, powerful queries and automatic scaling).

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

x Add Firebase to your web app

1 Register app

App nickname

Also set up **Firebase Hosting** for this app. [Learn more](#)

Hosting can also be set up later. It's free to get started at any time.

Register app

2 Add Firebase SDK

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: "AIzaSyA0VPouX305_KJ7V0lwZjh1y_4iFYQ0njc",
  authDomain: "cs022-demo-2022.firebaseio.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”:

Firebase

5CS022-demo-2022

Project Overview 

Build

Authentication

Project settings

Users and permissions

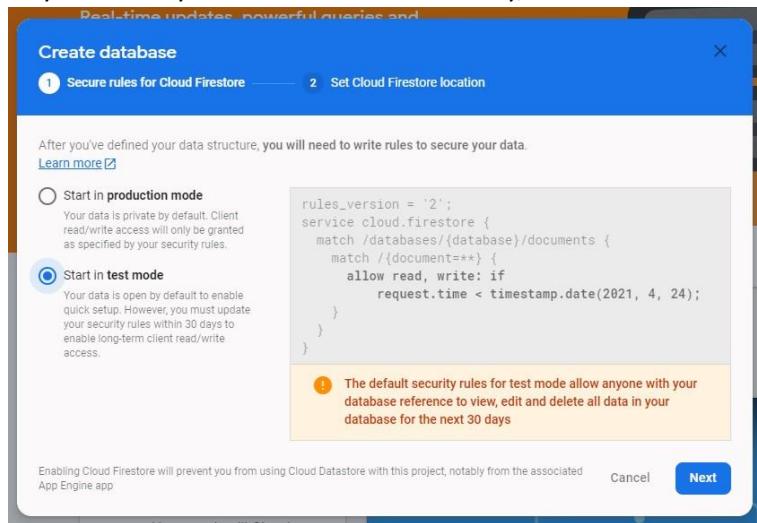
Usage and billing

Create a Firestore

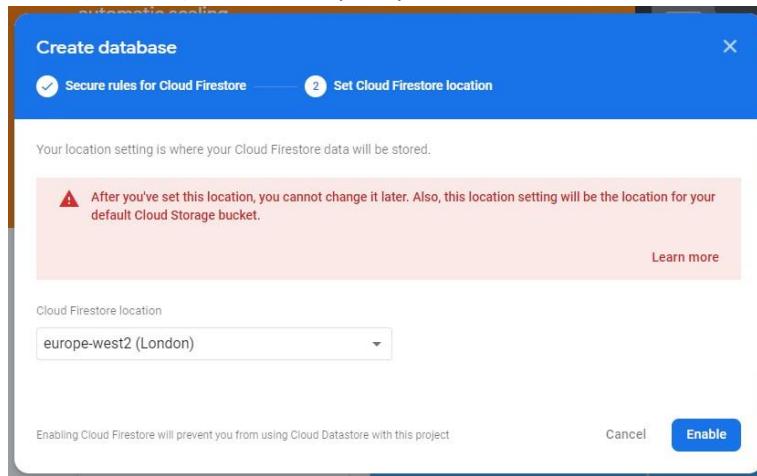
“Easily develop rich applications using a fully managed, scalable, and serverless document database.”
[\(source\)](#)

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

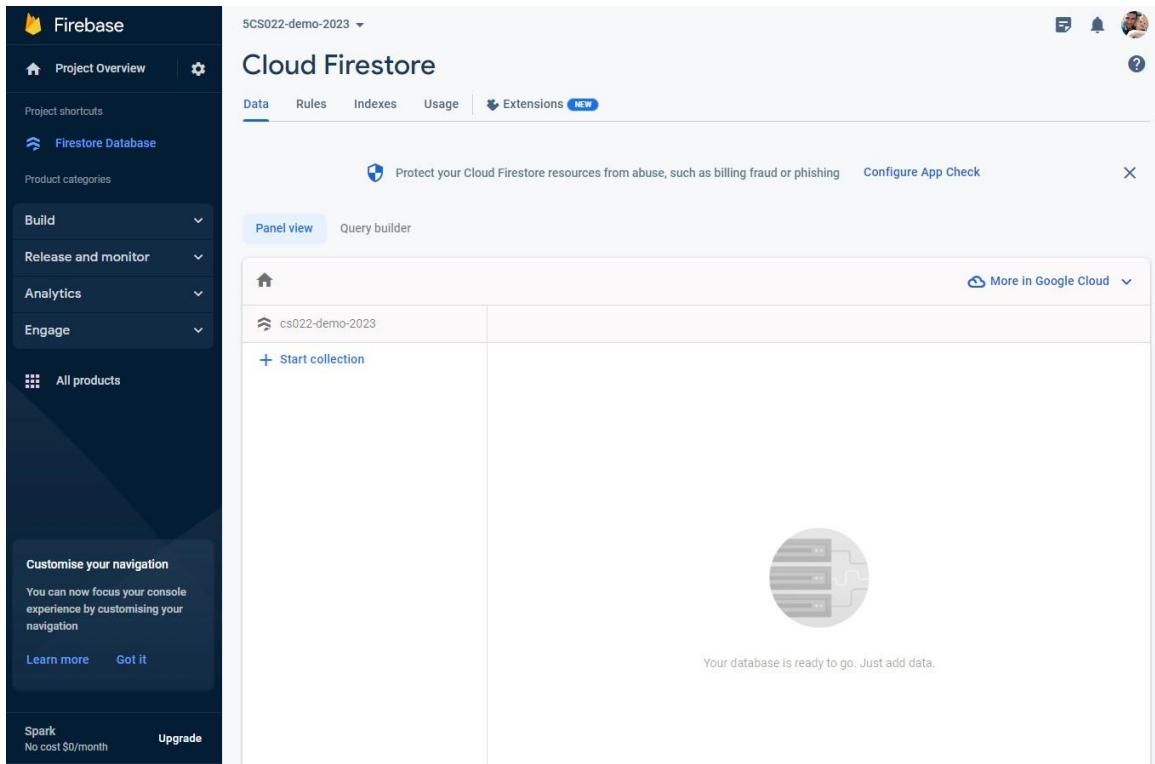
- In your [Firebase console](#), in the menu on the left select **Firestore Database** (under “Build”).
- Click on the “Create database” button in the middle of the screen.
- On the next screen, choose “**Start in test mode**”. This will create an open rule that allows for easy access of your data later 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.firebaseio.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”

The screenshot shows the Google Cloud Platform Firestore interface. On the left, there's a sidebar with options: Firestore (selected), Data, Indexes, Import/Export, Usage, and Security rules. The main area is titled 'Data' and shows a collection named 'Reviews'. Below it, there's a sub-section for 'Root' with a '+ START COLLECTION' button.

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”:

The screenshot shows the 'Start a collection' dialog. It has two main sections: 'Give the collection an ID' (which is checked) and 'Add its first document' (which is step 2). Below this, the 'Document parent path' is set to '/Reviews'. The 'Document ID' field contains 'YhexSBX8pjmtsyANltDS'. Under 'Add its first document', there are two fields defined:

- Field: book_name, Type: string, Value: Dune
- Field: book_rating, Type: number, Value: 5

At the bottom right are 'Cancel' and 'Save' buttons.

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

The screenshot shows the Firestore console. At the top, there's a navigation bar with icons for Home, Reviews, and a document ID. Below it, the 'Reviews' collection is selected. A document with the ID 'iyMKEeLNt0PzVrNKUC3P' is shown. The document details are as follows:

- + Start collection
- + Add document
- iyMKEeLNt0PzVrNKUC3P
- + Start collection
- + Add field
- book_name: "Dune"
- book_rating: 5

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.css" 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.firebaseio-
app.js"; import { Firestore, getFirestore, onSnapshot,
query, collection,
    orderBy,
    addDoc } from
'https://www.gstatic.com/firebasejs/9.18.0/firebase-firebase.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 Google 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 -->
```

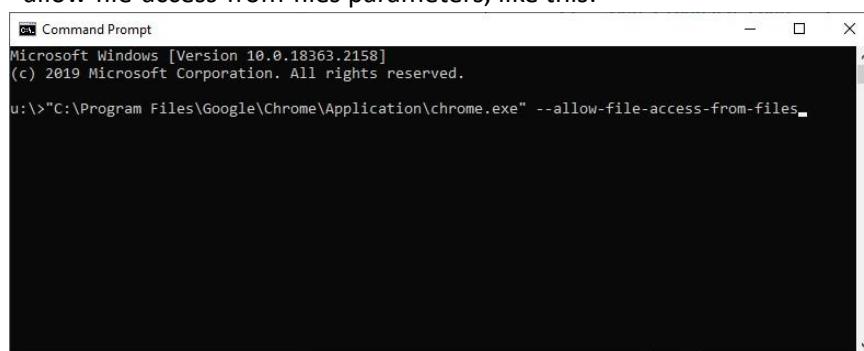
Things to note:

- **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:
 - o 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/firebase/
    import { Firestore,
            getFirestore,
            onSnapshot,
```

- o Uploading your work on a server such as mi-linux, like I have done (more information [here](#))
- o 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!)

6 book reviews in the list

A Journey To The Centre of The Earth	4/5
Dune	5/5
Ender's Game	5/5
Fahrenheit 451	5/5
Neuromancer	4/5
The Forever War	4/5

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.

- 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!

6 book reviews in the list

Book name	1/5	Add
A Journey To The Centre of The Earth	4/5	
Dune	5/5	

- 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>

6 book reviews in the list

Book name	Rating	Action
A Journey To The Centre of The Earth	4/5	Add
Dune	5/5	
Ender's Game	5/5	
Fahrenheit 451	5/5	
Neuromancer	4/5	
The Forever War	4/5	

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 C

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](#)



[X](#) 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 ②
- User segmentation & targeting across Firebase products ②
- Breadcrumb logs in Crashlytics ②
- Event-based Cloud Functions triggers ②
- Free unlimited reporting ②

Enable Google Analytics for this project
Recommended

[Create project](#)



[Previous](#)

Preparing your project, please wait.
task3-2358830



The screenshot shows the Firebase Project Overview page. At the top, there is a circular logo with three yellow/orange dots and the project name "task3-2358830". A green checkmark indicates that the project is ready. A "Continue" button is visible. To the right, there is a cartoon illustration of a person sitting on a large grey sphere, working on a laptop with a yellow flame icon. The background is dark blue with abstract white line art.

Firebase

Project Overview

task3-2358830

Your Firebase project is ready

Continue

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+ Android </> Audio F

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](#)

x Add Firebase to your web app



1 Register app

App nickname

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

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-Dchvx8EdRlwZeJvrFB99P81EGg",
  authDomain: "task3-2358830.firebaseioapp.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

2 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/firebase-app.js"
// 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: "AIzaSyAyjOEvnUchvxBEdIlwZeJvrFB9P81E0g",
  authDomain: "task3-2358830.firebaseioapp.com",
  projectId: "task3-2358830",
  storageBucket: "task3-2358830.appspot.com",
  messagingSenderId: "657959857717",
  appId: "1:657959857717:web:6deccae514f2ec75592e1c"
};

// 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

[✉️](#) 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

Spark No-cost \$0/month [Upgrade](#)

Firebase Project Overview

task3-2358830 ▾

Receive email updates about new Firebase features, research, and events Sign up X

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

The screenshot shows the Firebase Project Overview page for a project named "task3-2358830". On the left, there's a sidebar with "Project categories" and "Release & Monitor" sections. A "Gemini in Firebase" callout box is prominently displayed, encouraging users to try the AI feature. Below it, a card titled "Choose a product to add to your app" offers "Store and sync app data in milliseconds" and features a thumbnail for Cloud Firestore.

Firebase Project Overview

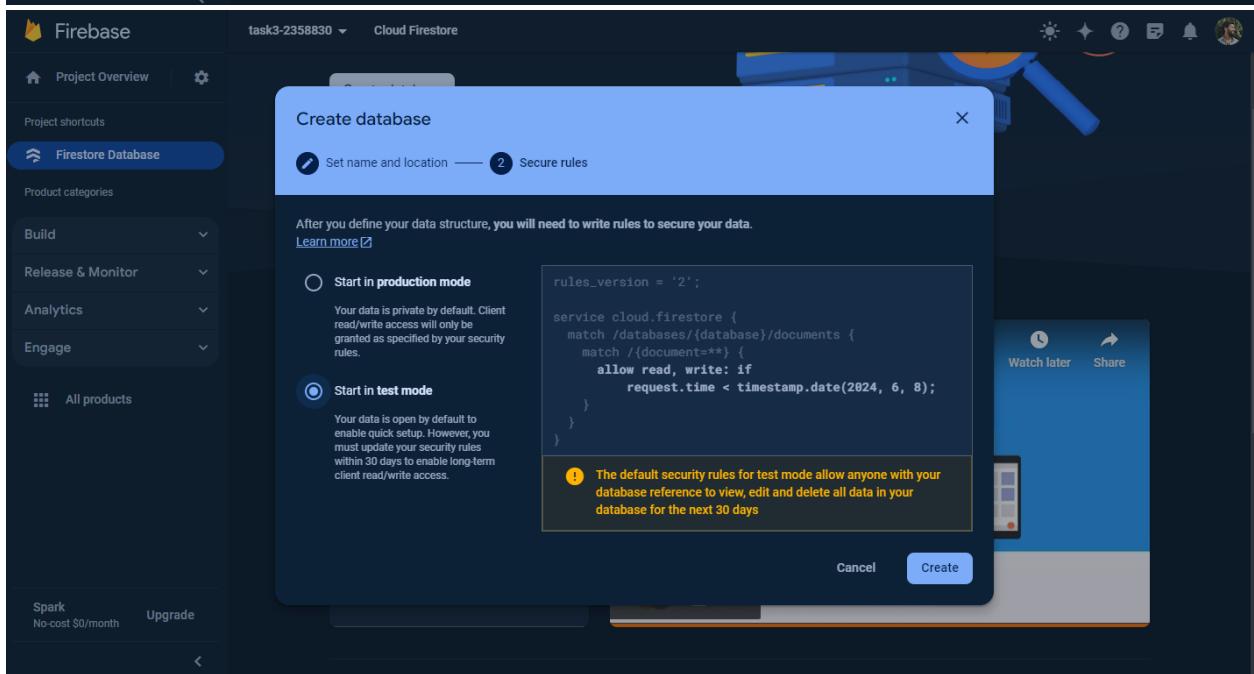
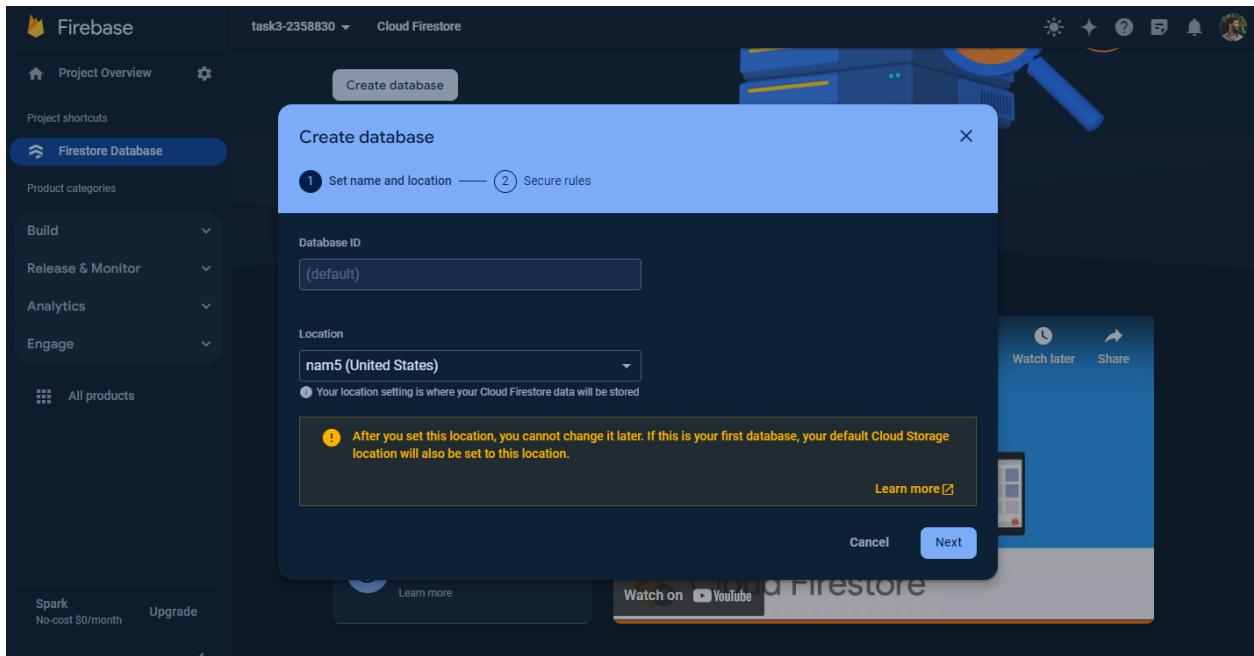
task3-2358830 ▾

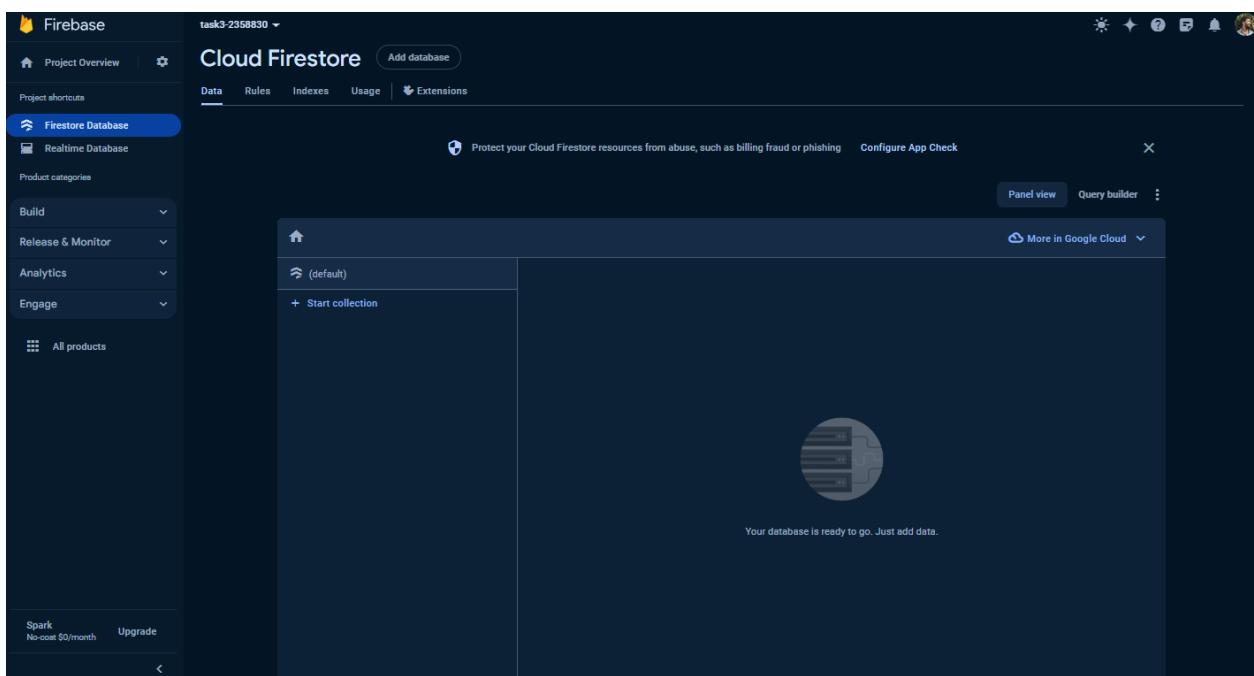
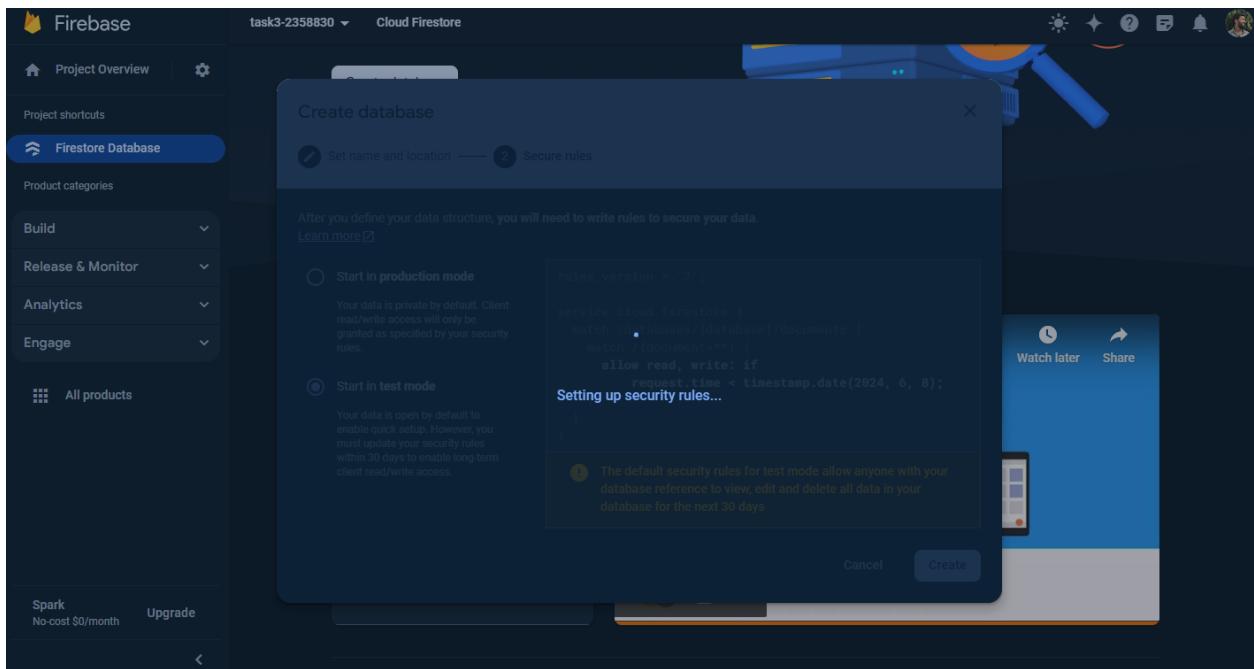
Cloud Firestore

Realtime updates, powerful queries, and automatic scaling

Create database

The screenshot shows the Cloud Firestore product page. It features a large illustration of a stack of blue server racks with a magnifying glass focusing on the top one, which has the Firestore logo. The page includes a "Create database" button and a "Learn more" section with links to "How do I get started?" and "How much will Cloud Firestore cost?".





Firebase task3-2358830  Cloud Firestore 

Project Overview 

Firestore Database  Realtime Database

Product categories

Build  Release & Monitor

Analytics  Engage

All products

Spark No-cost \$0/month Upgrade

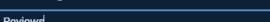
Cloud Firestore

Data Rules Indexes Usage 

Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing 

Panel view Query builder 

Start a collection

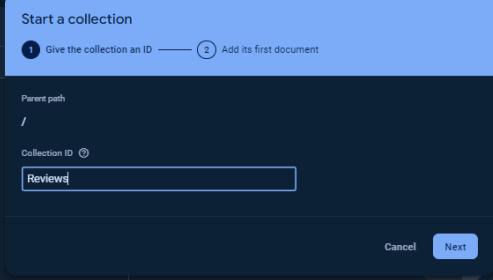
1 Give the collection an ID  2 Add its first document

Parent path /

Collection ID  Reviews

Cancel Next

Your database is ready to go. Just add data.



Firebase task3-2358830  Cloud Firestore 

Project Overview 

Firestore Database  Realtime Database

Product categories

Build  Release & Monitor

Analytics  Engage

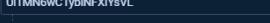
All products

Spark No-cost \$0/month Upgrade

Cloud Firestore

Data Rules Indexes Usage 

Document parent path /Reviews

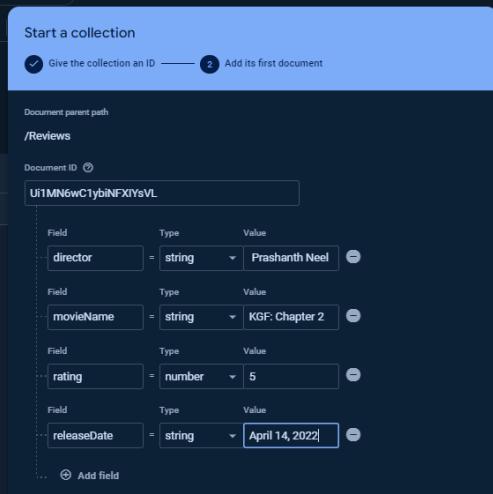
Document ID  Uf1MN6wC1ybiNFXIysVL

Field Type Value
director = string Prashanth Neel 
Field Type Value
movieName = string KGF: Chapter 2 
Field Type Value
rating = number 5 
Field Type Value
releaseDate = string April 14, 2022 

Add field

Cancel Save

Your database is ready to go. Just add data.



The screenshot shows the Firebase Cloud Firestore interface. On the left, the navigation bar includes 'Project Overview', 'Realtime Database', 'Firestore Database' (selected), and 'Analytics'. The main area displays a document structure under 'Reviews'. The root 'Reviews' collection contains two documents: 'U11MNGwC1yb1NFXIYsVL' and 'kKpQCCKeQc2xTww8NJ1Y'. The second document is expanded, showing fields: director: "Lokesh Kanagaraj", movieName: "Vikram", rating: "5", and releaseDate: "June 3, 2022".

```
<!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.css"
  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>


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

<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.firebaseio-app.js";
import { Firestore,
getFirestore,
onSnapshot,
query,
collection,
orderBy,
addDoc } from
'https://www.gstatic.com/firebasejs/9.18.0/firebase-firebase.js'
// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyB9P1akYdqGbnUdYzV-FOSJtDlmcBqQl2A",
  authDomain: "cs022-2358830-sumitshah.firebaseioapp.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.css"
    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.firebaseio-app.js";
import { Firestore,
getFirestore,
onSnapshot,
query,
collection,
orderBy,
addDoc } from
'https://www.gstatic.com/firebasejs/9.18.0/firebase-firebase.js'
// Your web app's Firebase configuration
const firebaseConfig = {
  apiKey: "AIzaSyB9P1akYdqGbnUdYzV-FOSJtDlmcqQl2A",
  authDomain: "cs022-2358830-sumitshah.firebaseioapp.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>

<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha1/dist/css/bootstrap.min.css" 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">
<!-- &lt;input type="text" class="form-control" id="directorName" placeholder="Director name"&gt;
&lt;input type="text" class="form-control" id="releaseDate" placeholder="Release date"&gt; --&gt;
&lt;select class="form-control mx-2 w-25" id="bookRating"&gt;</pre>
```

```
<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.firebaseio-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.firebaseioapp.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>

```

Book name	Rating
Java	4/5
Sumit Shgah	2/5
Vagwat Gita	5/5
computer	3/5
santosh	2/5

The screenshot shows the Firebase Cloud Firestore interface. On the left, a sidebar navigation includes 'Project Overview', 'Project shortcuts', 'Build', 'Release & Monitor', 'Analytics', 'Engage', and 'All products'. The 'Firestore Database' tab is selected. The main area displays a 'Cloud Firestore' dashboard with a 'Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing' banner and a 'Configure App Check' button. Below this is a 'Panel view' and 'Query builder' section. The main content area shows a hierarchical navigation path: Home > Reviews > iVw9MGhrL2Ydc186hc70. The 'Reviews' collection has a single document, 'iVw9MGhrL2Ydc186hc70', which is expanded. This document contains fields: 'book_name' (set to 'computer') and 'book_rating' (set to 3). There are also buttons for '+ Add document' and '+ Add field'.