

UNIVERSITY PARTNER



UNIVERSITY OF  
WOLVERHAMPTON



HERALD  
COLLEGE  
KATHMANDU

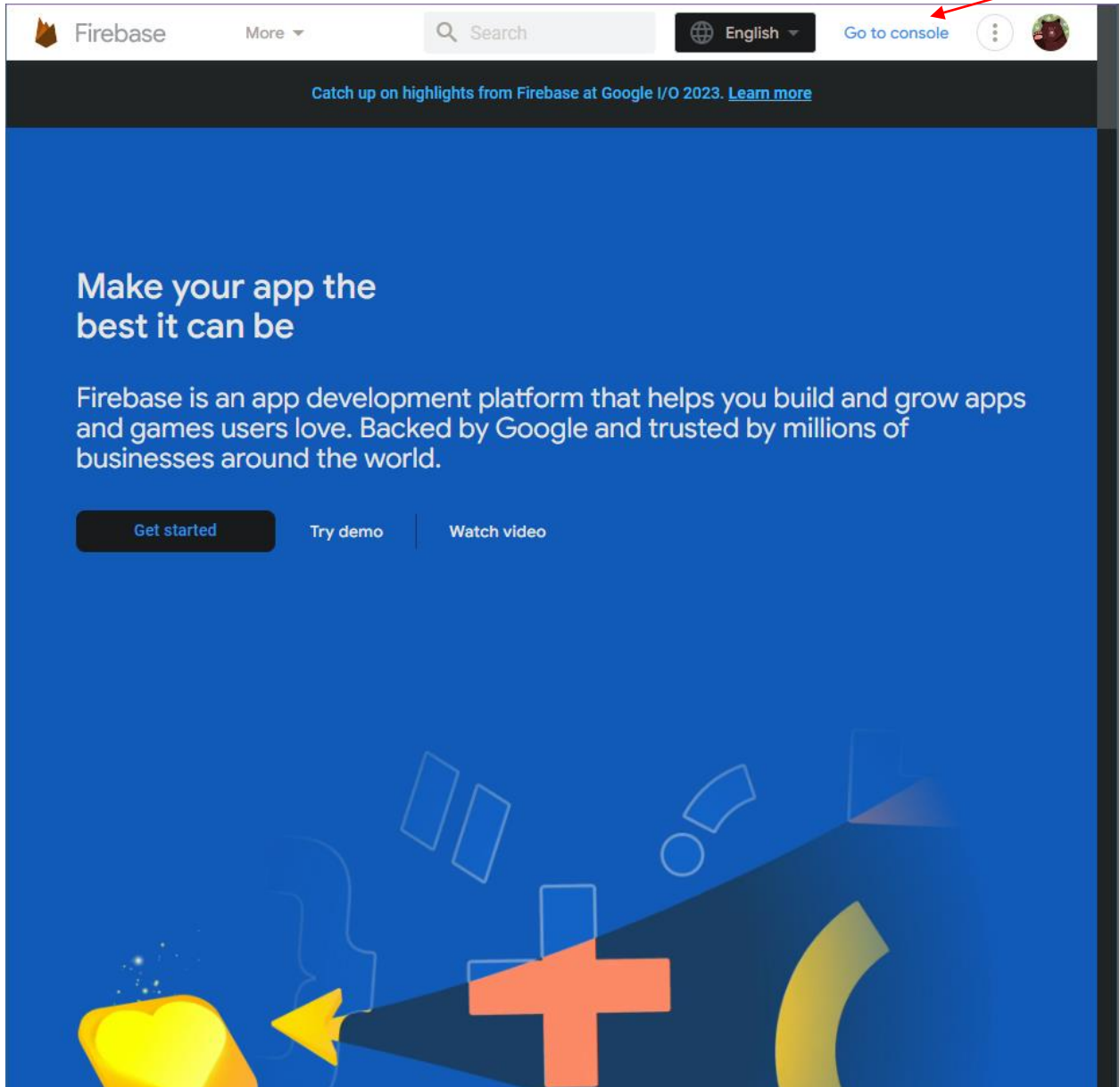
# Distributed and Cloud System Programming (5CS022)

## Task 5 - Movie Review System Documentation

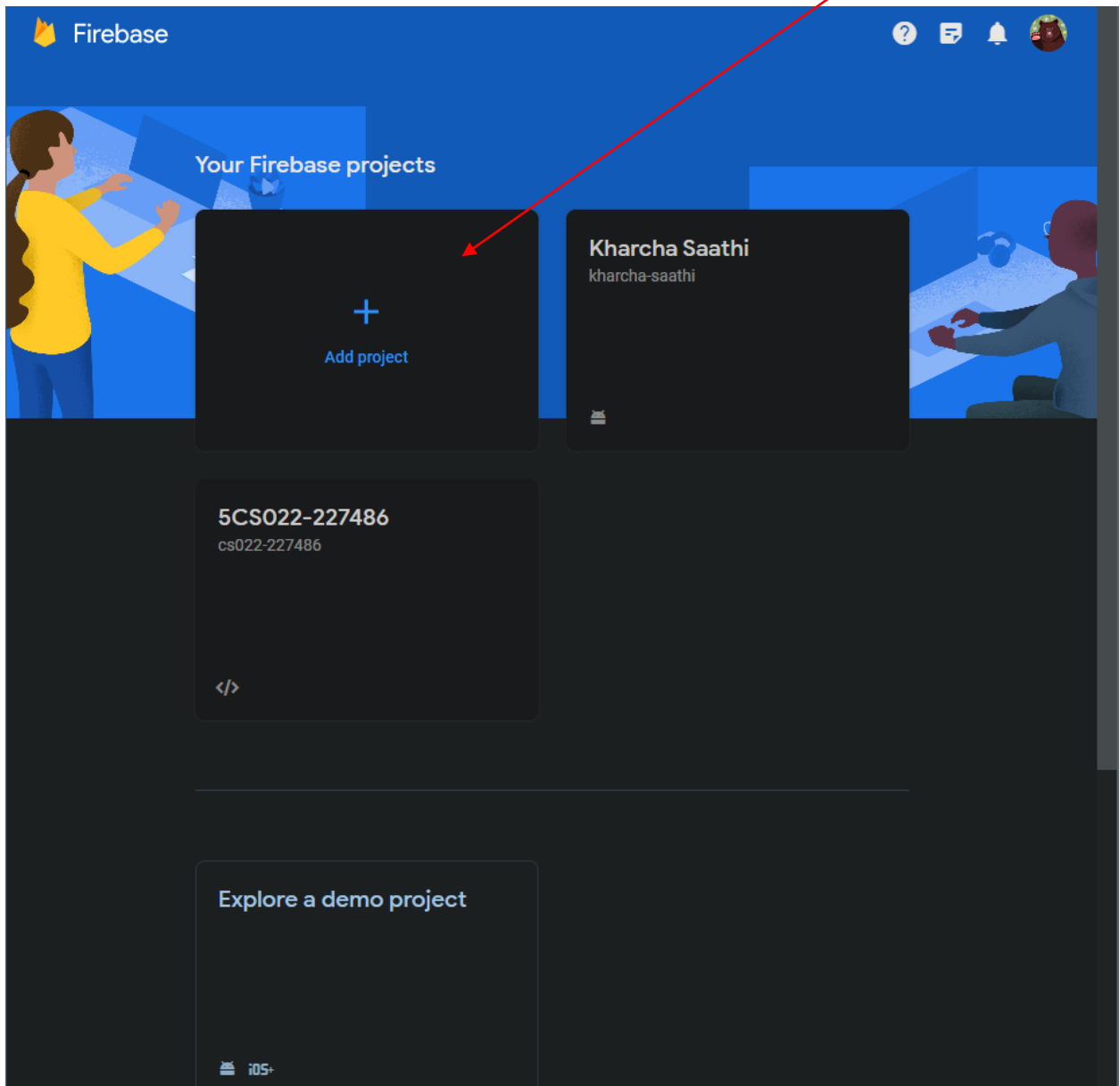
Student Id	: 2227486
Student Name	: Nayan Raj Khanal
Group	: L5CG4
Instructor	: Mr. Prabin Sapkota
Submitted On	: 16.05.2023

## STEP 1: Creating a project

➔ Go to: <https://firebase.google.com/> and click “Go to console”



➔ Click on “Add project” to create a new project




➔ Give a name to your project


× Create a project (Step 1 of 3)

Let's start with a name for your project<sup>?</sup>

Project name

task5-2227486

 task5-2227486

 Select parent resource

Continue

➔ Disable Google Analytics as it is not needed. Click on “create project”.

× 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 ⓘ

× Crash-free users ⓘ

× Event-based Cloud Functions triggers ⓘ

× Free unlimited reporting ⓘ

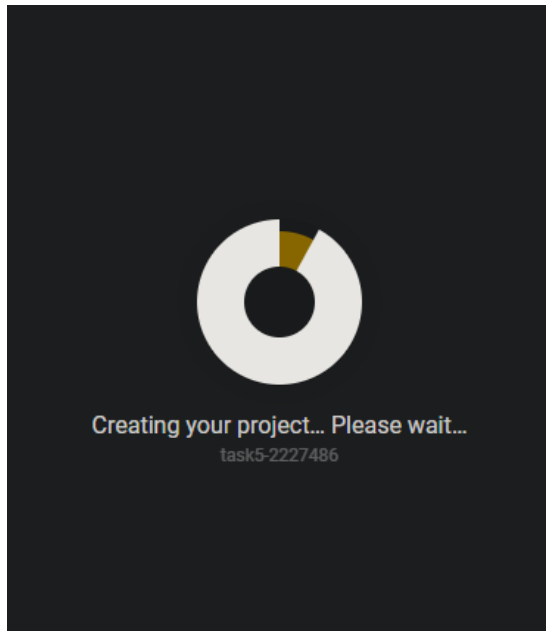
☐ Enable Google Analytics for this project

Recommended

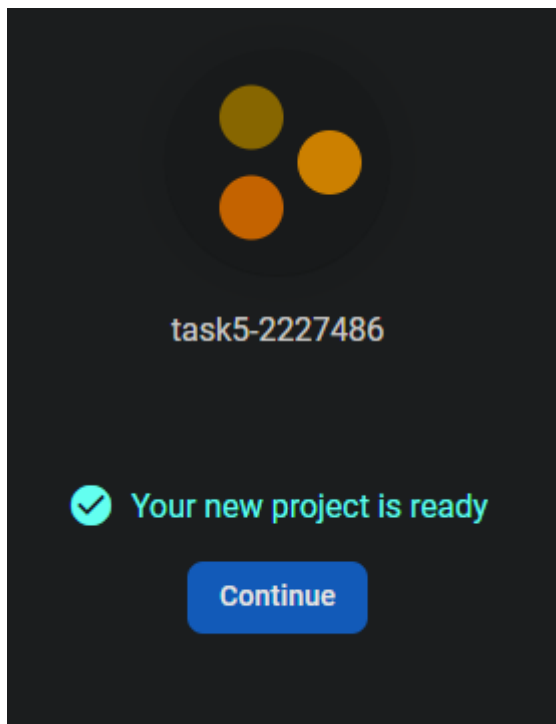
[Previous](#)

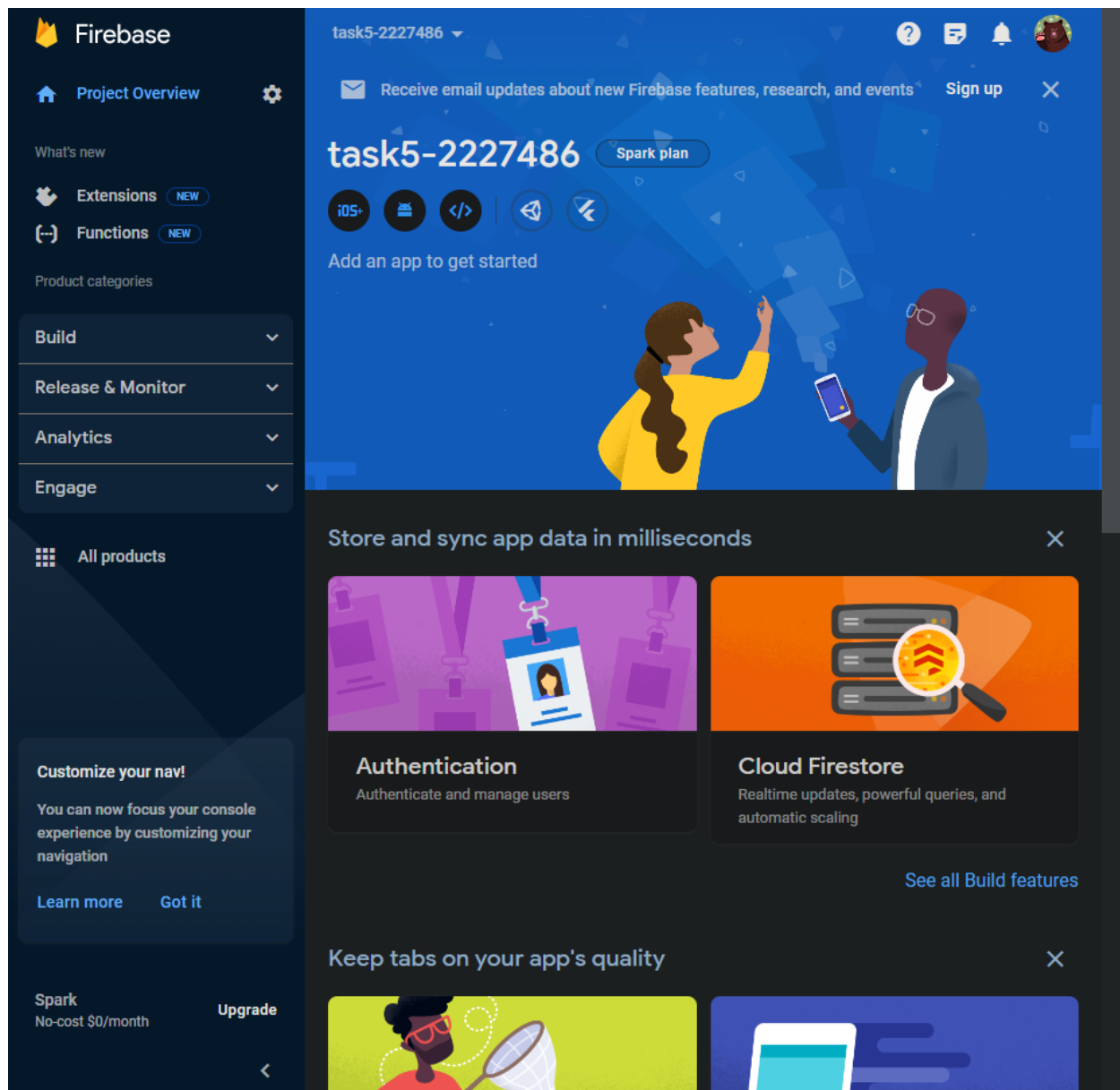
Create project

➔ Please be patient while it loads



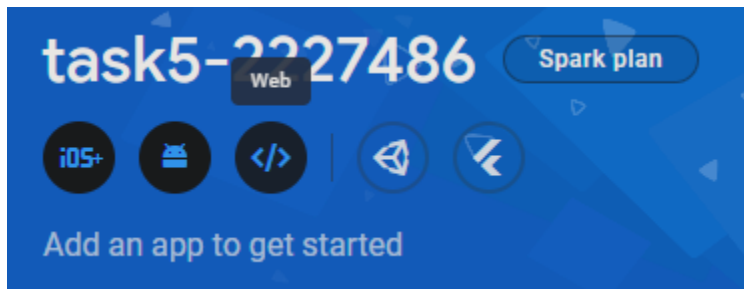
After a while you'll get this interface, click "Continue"





You have created your firebase project.

➔ To add Firebase to an app, select “Web” option as we will be accessing our Firestore from JavaScript



➔ Give your app a name and click “Register”

© 2020 Firebase

× 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



➔ Select “Use <script> tag option and save the JavaScript code safely as it contains our connection details.

After saving “Continue to console”.

[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/9.22.0/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: "AIzaSyBUaeBneMi1PyyBCaK9a-6wIIA2N8pPfuw",
    authDomain: "task5-2227486-be9ed.firebaseio.com",
    projectId: "task5-2227486-be9ed",
    storageBucket: "task5-2227486-be9ed.appspot.com",
    messagingSenderId: "701960114309",
    appId: "1:701960114309:web:1233460c34d5cf7a3599ec"
  };

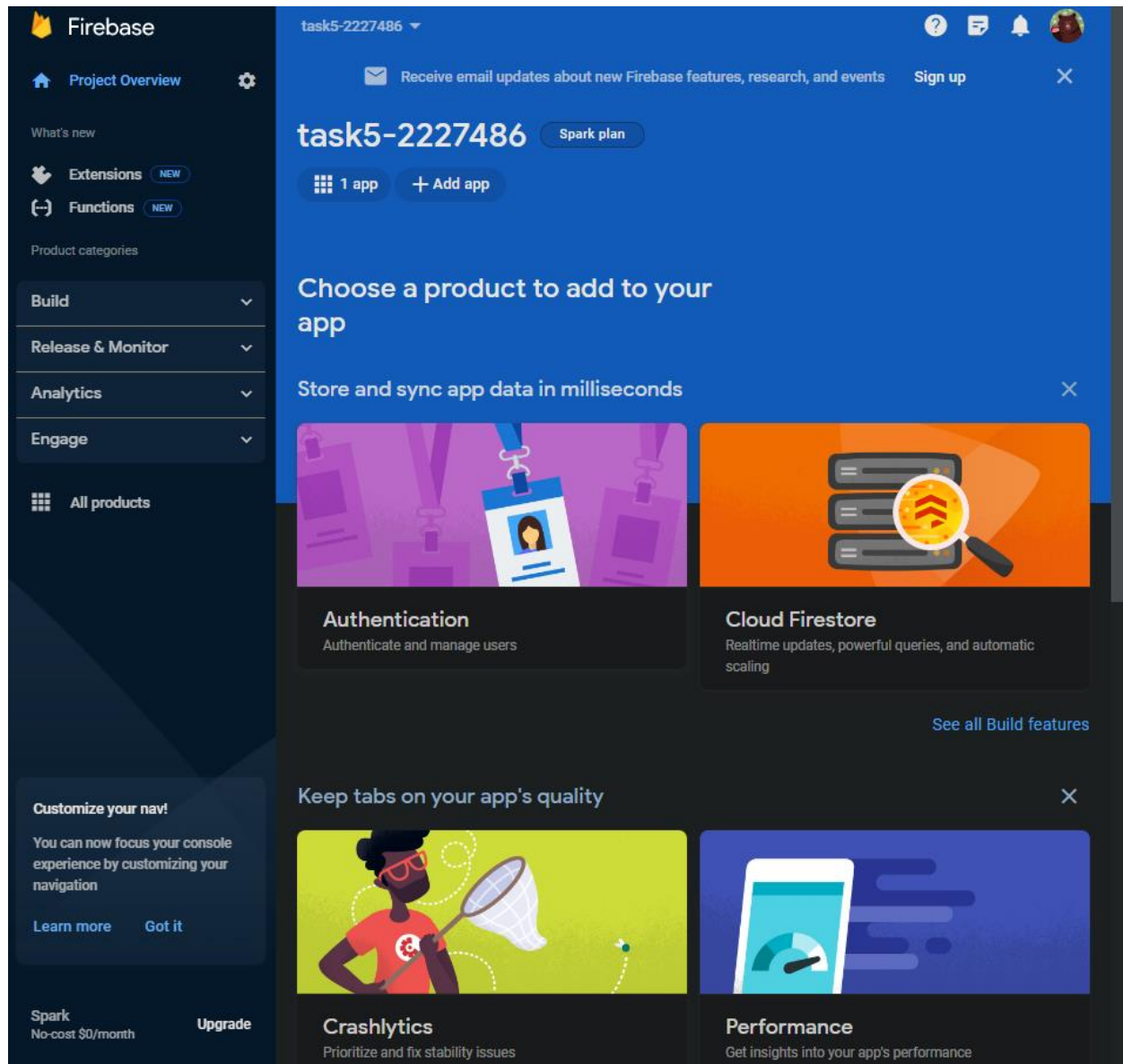
  // Initialize Firebase
  const app = initializeApp(firebaseConfig);
</script>
```

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](#)

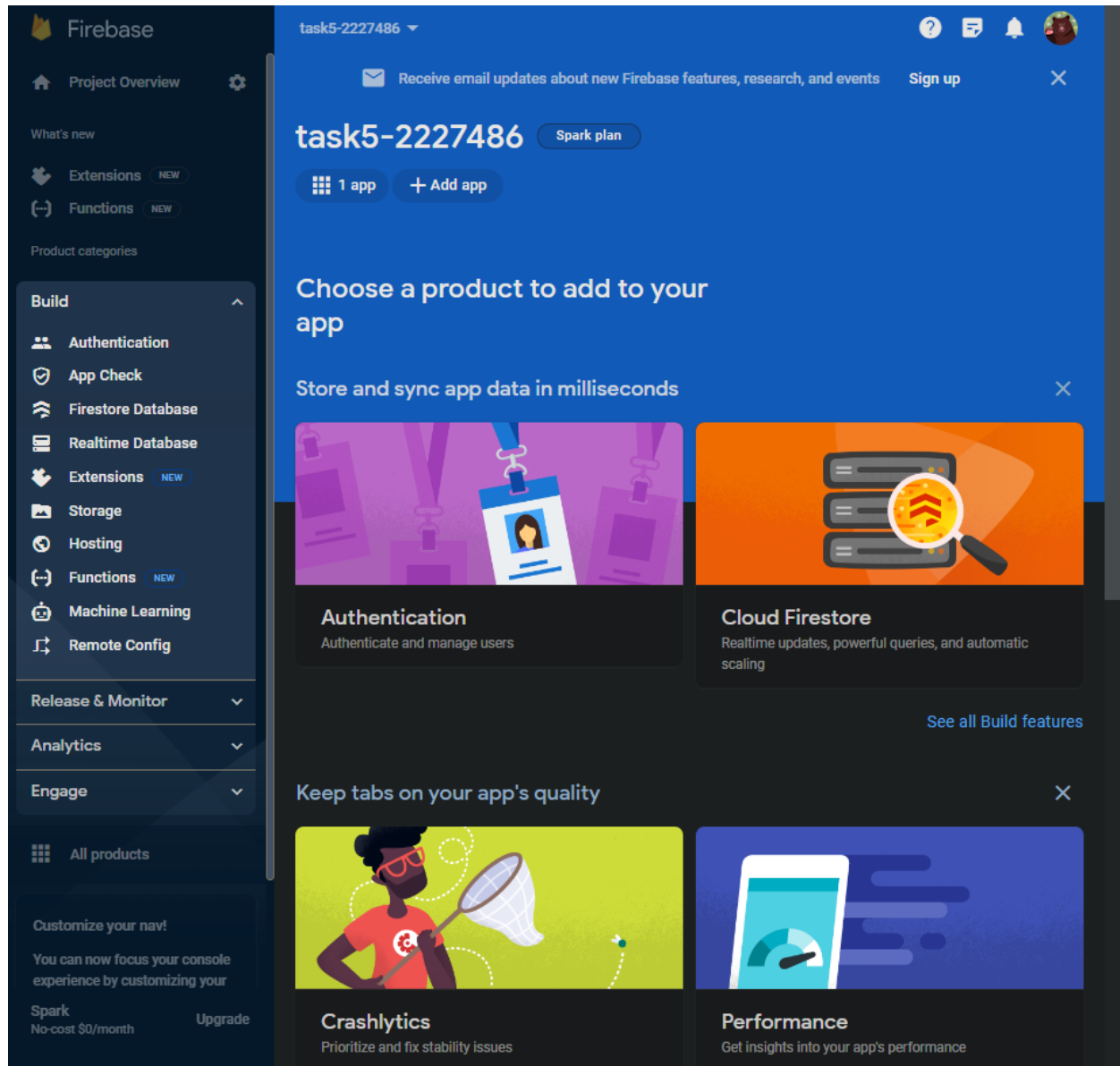
Continue to console

➔ You will be redirected to your console

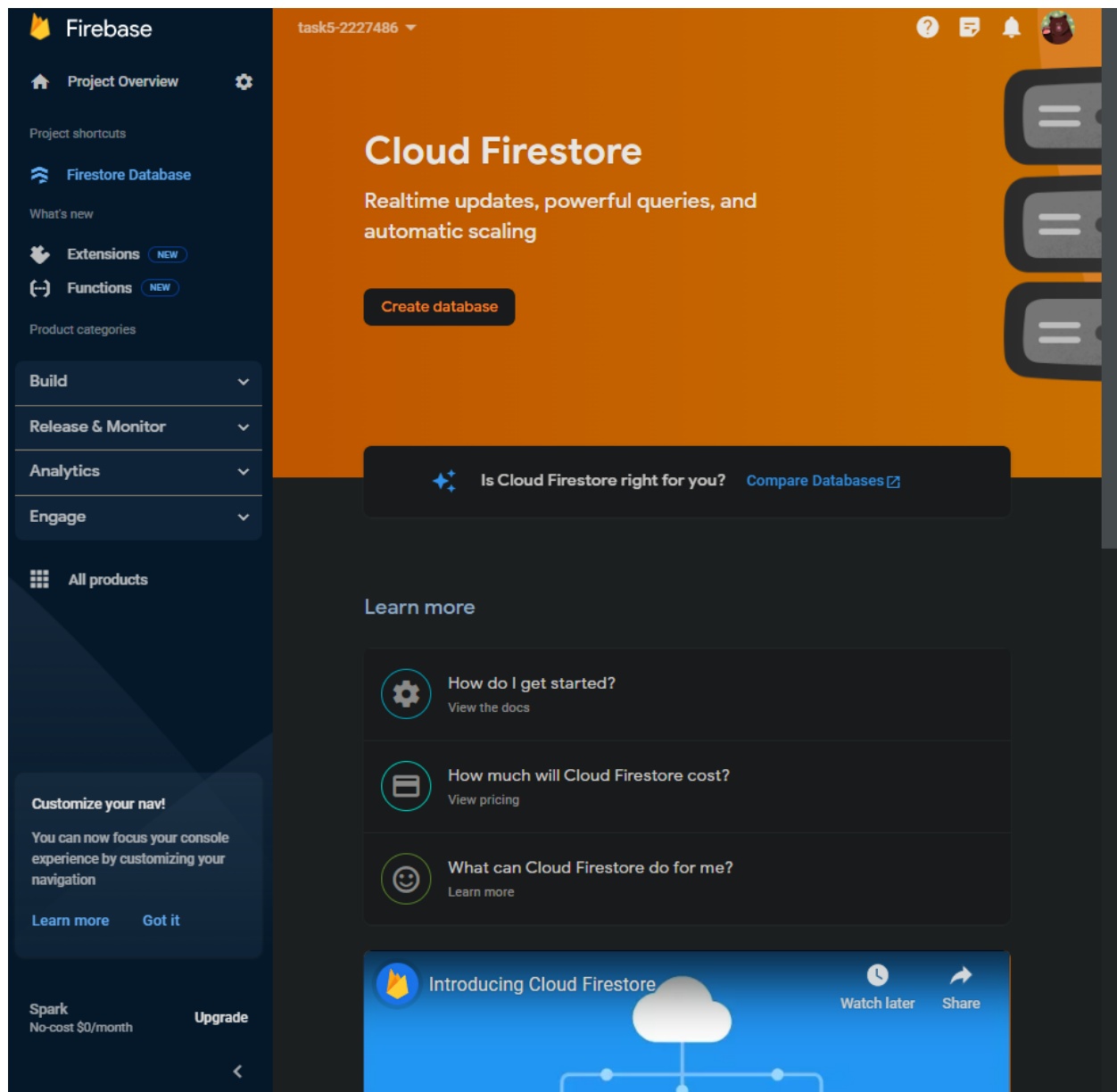


## STEP 2: Creating a Firestore

➔ Click on “Build” and under Build select “Firestore Database”

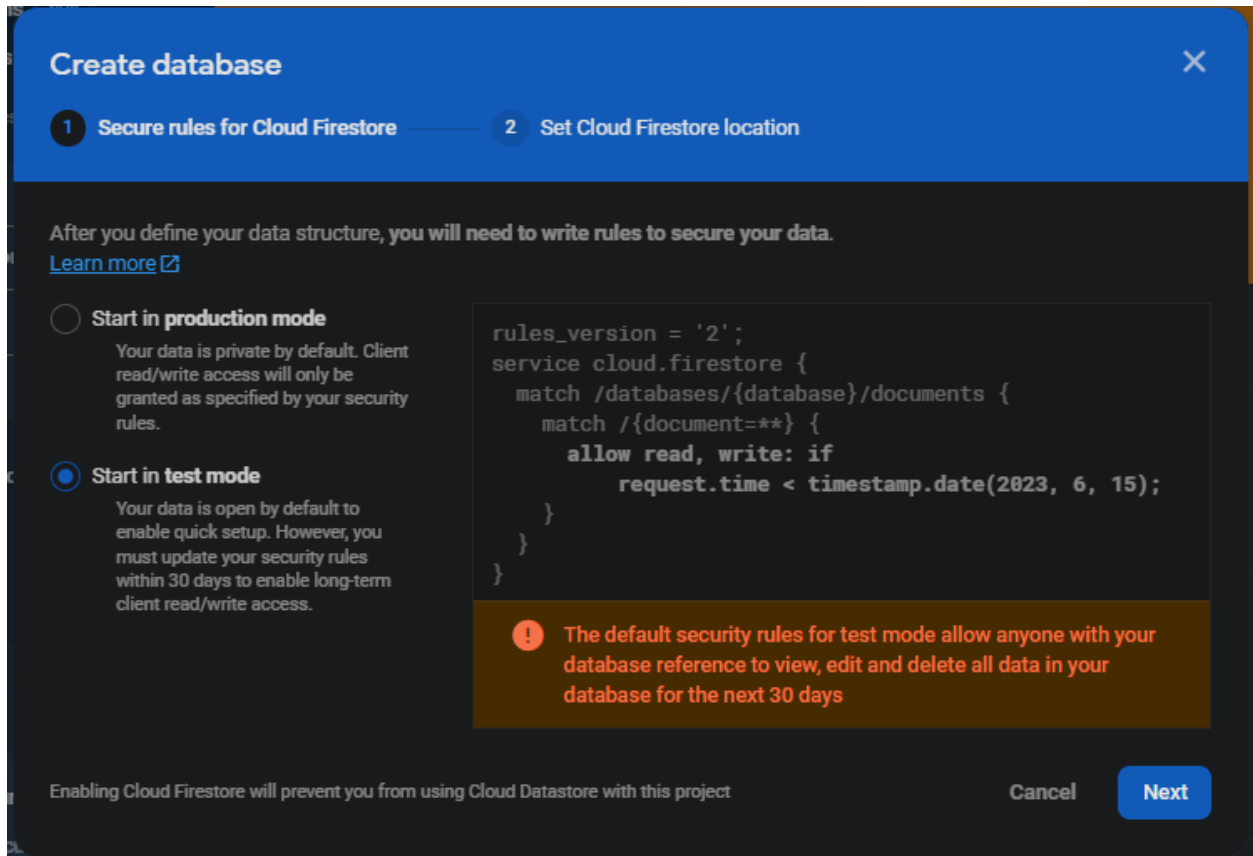


➔ Click on “Create database”



➔ Select “Start in test mode”. This will create an open rule that allows for easy access of the data later.

Click on “Next”.



➔ Set the location of your Firestore to somewhere nearby as this will impact performance and cannot be changed later! Press “Enable”.

The screenshot shows a 'Create database' dialog box with a blue header and a dark grey body. The header contains a close button (X) and two progress steps: '1 Secure rules for Cloud Firestore' (completed) and '2 Set Cloud Firestore location' (active). Below the header, a message states: 'Your location setting is where your Cloud Firestore data will be stored.' A red warning box follows, stating: 'After you set this location, you cannot change it later. Also, this location setting will be the location for your default Cloud Storage bucket.' A 'Learn more' link with an external icon is at the bottom right of the warning box. Below the warning, the 'Cloud Firestore location' is set to 'asia-south1 (Mumbai)' in a dropdown menu. At the bottom, a note says: 'Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project'. There are 'Cancel' and 'Enable' buttons at the bottom right.

Create database

1 Secure rules for Cloud Firestore 2 Set Cloud Firestore location

Your location setting is where your Cloud Firestore data will be stored.

⚠ After you set this location, you cannot change it later. Also, this location setting will be the location for your default Cloud Storage bucket. [Learn more](#)

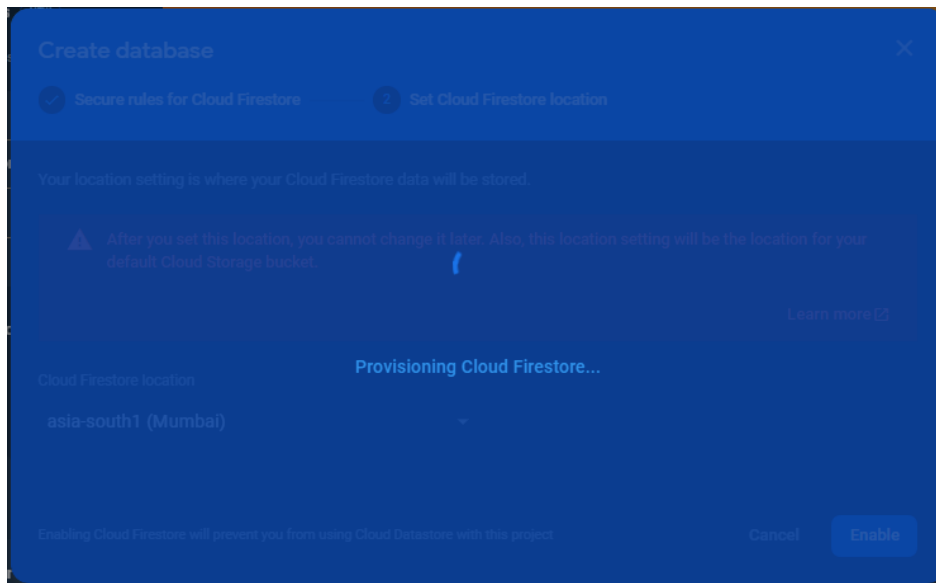
Cloud Firestore location

asia-south1 (Mumbai)

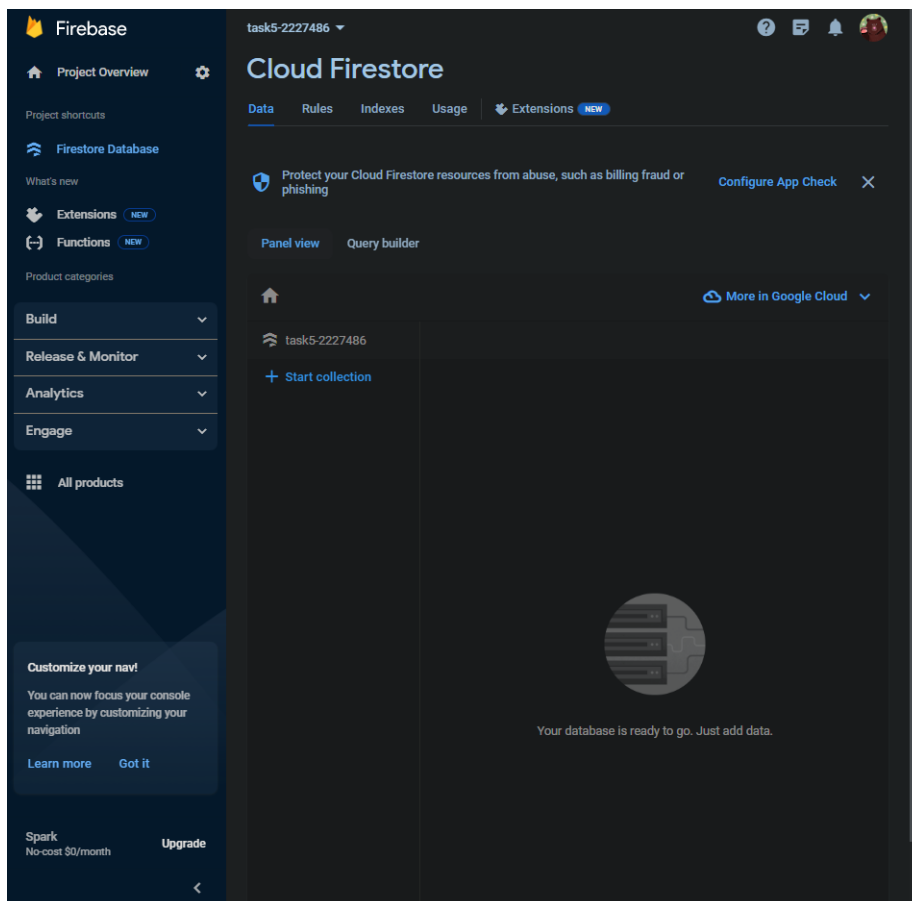
Enabling Cloud Firestore will prevent you from using Cloud Datastore with this project

Cancel Enable

➔ Please be patient while it loads

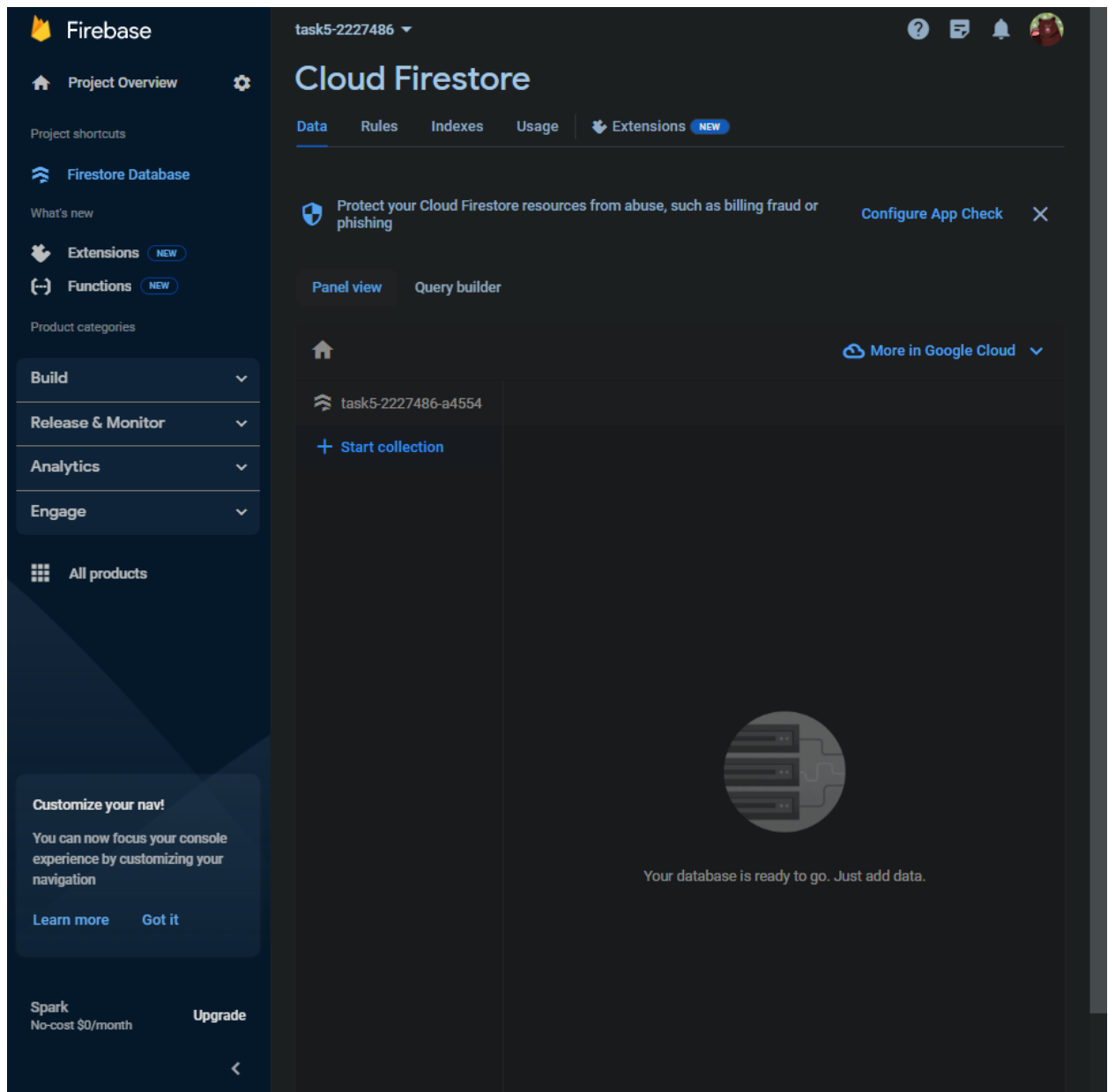


You will be greeted with this interface once it loads



## STEP 3: Creating collections and documents

➔ Click on “Start Collection”





➔ Enter “Reviews” in the Collection ID field, then press “Next”

**Start a collection**

1 Give the collection an ID — 2 Add its first document

Parent path

/

Collection ID ⓘ

Reviews

Cancel Next

➔ Add documents in “Reviews” accordingly and click on “Save”

### Add a document

Parent path  
/Reviews

Document ID ?  
m0yqVbRnEqXtQvLT1dtT

Field	Type	Value	
director	= string	Christopher Nola	−
movieName	= string	Oppenheimer	−
rating	= number	5	−
releaseDate	= string	2023-07-23	−

+

Cancel **Save**

NOTE:

Press “Auto-ID” to populate the Document ID field

➔ You should see your single document in the firestore like this:

The screenshot displays the Firebase Cloud Firestore console interface. On the left is a dark sidebar with the 'Firebase' logo and navigation options: 'Project Overview', 'Firestore Database', 'Extensions' (marked NEW), 'Functions' (marked NEW), and 'All products'. The main area is titled 'Cloud Firestore' for project 'task5-2227486'. It includes tabs for 'Data', 'Rules', 'Indexes', 'Usage', and 'Extensions' (marked NEW). A security warning banner is present. Below, there are 'Panel view' and 'Query builder' options. The 'Data' panel shows a breadcrumb path: 'Home > Reviews > m0yqVbRnEqXt...'. A table lists collections and documents. The 'Reviews' collection is expanded, showing a document 'm0yqVbRnEqXtQvLT1dT'. The document's JSON content is displayed on the right: 

```
{  "director": "Christopher Nolan",  "movieName": "Oppenheimer",  "rating": 5,  "releaseDate": "2023-07-23"}
```

task5-2227486-a4554	Reviews	m0yqVbRnEqXtQvLT1dT
<a href="#">+ Start collection</a>	<a href="#">+ Add document</a>	<a href="#">+ Start collection</a>
Reviews >	m0yqVbRnEqXtQvLT1dT	<a href="#">+ Add field</a>
		<pre>director: "Christopher Nolan" movieName: "Oppenheimer" rating: 5 releaseDate: "2023-07-23"</pre>

→ Let's add one more document

### Add a document

Parent path  
/Reviews


Document ID ?

oeObYGdt7ip9LDZB8mja

Field	Type	Value	
director	= string	Nayan Raj Khana	—
movieName	= string	Life of CS Studer	—
rating	= number	5	—
releaseDate	= string	2023-03-16	—

+ Add field

Cancel Save

 **Firebase**

Project Overview

Project shortcuts

Firestore Database

What's new

Extensions NEW

Functions NEW

Product categories

Build

Release & Monitor

Analytics

Engage

All products

Customize your nav!

You can now focus your console experience by customizing your navigation

[Learn more](#) [Got it](#)

Spark

No-cost \$0/month

Upgrade

task5-2227486



## Cloud Firestore

Data Rules Indexes Usage Extensions NEW

Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing [Configure App Check](#)

Panel view Query builder

Home > Reviews > oeObYGdt7ip9L...

More in Google Cloud

task5-2227486-a4554	Reviews	oeObYGdt7ip9LDZB8mja
<a href="#">+ Start collection</a>	<a href="#">+ Add document</a>	<a href="#">+ Start collection</a>
Reviews >	m0yqVbRnEqXtQvLT1d...	<a href="#">+ Add field</a>
	oeObYGdt7ip9LDZB8m...	<div>director : "Nayan Raj Khanal"</div> <div>movieName: "Life of CS Student"</div> <div>rating: 5</div> <div>releaseDate: "2023-03-16"</div>

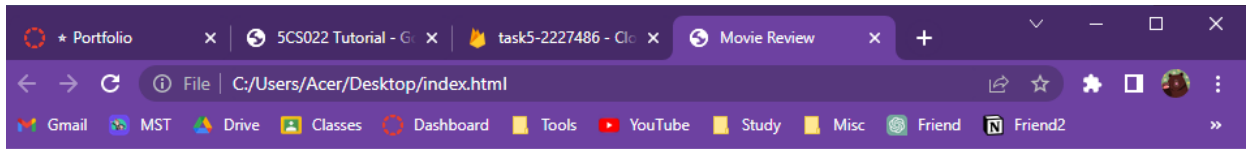
## PART 4: Accessing data from JavaScript

➔ Let's create a simple web page

```
C:\Users\Ace\Desktop> index.html > html > body > div.container.w-50.bg-light.shadow.rounded.p-5 > table.reviews-table.table.bg-success
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8" />
5   <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6   <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7   <title>Movie Reviews</title>
8   <!-- Bootstrap CSS -->
9   <link
10     rel="stylesheet"
11     href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
12   />
13 </head>
14 <body>
15   <div class="container w-50 bg-light shadow rounded p-5">
16     <h1 class="my-4">Movie Reviews</h1>
17     <!-- Table to display reviews -->
18     <table class="table bg-success" id="reviews-table">
19       <thead class="thead-dark">
20         <tr>
21           <th id="movie-name-header"
22             onmouseenter="this.style.cursor = 'pointer';"
23           >
24             Movie Name</th>
25           <th
26             id="rating-header"
27             onmouseenter="this.style.cursor = 'pointer';"
28           >
29             Rating</th>
30           <th id="director-header"
31             onmouseenter="this.style.cursor = 'pointer';"
32           >
33             Director</th>
34           <th id="release-date-header"
35             onmouseenter="this.style.cursor = 'pointer';"
36           >
37             Release Date</th>
38           <th>Actions</th>
39         </tr>
40         <tr id="loading-row">
41           <td></td>
42           <td></td>
43           <td>Loading...</td>
44           <td></td>
45           <td></td>
46         </tr>
47       </thead>
48     </table>
49   </div>
```

This index.html contains both HTML and CSS

## Output:



## Movie Reviews

Movie Name	Rating	Director	Release Date	Actions
Loading...				

Movie Name:

Rating (0-5):

Director Name:

Release Date:

Add Review

➔ Now let's connect it to our Firestore, below is the code necessary to make the application functional according to the requirements.

```
<!-- Firebase CDN -->
<script src="https://www.gstatic.com/firebasejs/8.7.0/firebase-app.js"></script>
<script src="https://www.gstatic.com/firebasejs/8.7.0/firebase-database.js"></script>
<script>
  // Initialize Firebase
  const firebaseConfig = {
    apiKey: "AIzaSyBUae8neM1Pyy8Cak9a-6wIIA2N8pPFuw",
    authDomain: "task5-2227486-be9ed.firebaseio.com",
    projectId: "task5-2227486-be9ed",
    storageBucket: "task5-2227486-be9ed.appspot.com",
    messagingSenderId: "701960114309",
    appId: "1:701960114309:web:1233460c34d5cf7a3599ec"
  };

  firebase.initializeApp(firebaseConfig);

  // Get a reference to the movie reviews collection in Firestore
  var reviewsRef = firebase.database().ref("reviews");
  showFirestoreData();

  const movieNameHeader = document.getElementById("movie-name-header");
  const ratingHeader = document.getElementById("rating-header");
  const directorHeader = document.getElementById("director-header");
  const releaseDateHeader = document.getElementById("release-date-header");

  movieNameHeader.addEventListener("click", sortByMovieName);
  ratingHeader.addEventListener("click", sortByRating);
  directorHeader.addEventListener("click", sortByDirectorName);
  releaseDateHeader.addEventListener("click", sortByReleaseDate);

  // Function to add a new review to Firestore
  Codeium: Refactor | Explain
  function addReview(movieName, rating, director, releaseDate) {
    // Generate a new key for the review
    var newReviewRef = reviewsRef.push();

    // Set the data for the new review using the generated key
    newReviewRef
      .set({
        movieName: movieName,
        rating: rating,
        director: director,
        releaseDate: releaseDate,
      })
      .then(function () {
        console.log("Review added successfully!");
        document.getElementById("review-form").reset(); // Reset the form
      })
      .catch(function (error) {
        console.error("Error adding review: ", error);
      });
  }

  Codeium: Refactor | Explain | Generate Docstring
  function deleteReview(reviewKey) {
    // Remove the review from the database
    reviewsRef
      .child(reviewKey)
      .remove()
      .then(function () {
        console.log("Review deleted successfully!");
      })
      .catch(function (error) {
        console.error("Error deleting review: ", error);
      });
  }

  Codeium: Refactor | Explain | Generate Docstring
  function clearTable() {
    var table = document.getElementById("reviews-table");
    var rowCount = table.rows.length;
    for (var i = rowCount - 1; i > 0; i--) {
      table.deleteRow(i);
    }
  }
}
```



```

    var rowCount = table.rows.length;
    for (var i = rowCount - 1; i > 0; i--) {
        table.deleteRow(i);
    }
}

Codeium: Refactor | Explain | Generate Docs
function showFirestoreData() {
    var table = document.getElementById("reviews-table");
    // Get the Firestore data
    reviewsRef.on(
        "value",
        function (snapshot) {
            clearTable();
            if (snapshot.val() == null) {
                var row = table.insertRow(-1);
                var cell1 = row.insertCell(0);
                var cell2 = row.insertCell(1);
                var cell3 = row.insertCell(2);
                var cell4 = row.insertCell(3);
                var cell5 = row.insertCell(4);
                cell3.innerHTML = "No reviews found";
            }

            snapshot.forEach(function (childSnapshot) {
                var childData = childSnapshot.val();
                var row = table.insertRow(-1);
                var cell1 = row.insertCell(0);
                var cell2 = row.insertCell(1);
                var cell3 = row.insertCell(2);
                var cell4 = row.insertCell(3);
                var cell5 = row.insertCell(4);
                cell1.innerHTML = childData.movieName;
                cell2.innerHTML = childData.rating;
                cell3.innerHTML = childData.director;
                cell4.innerHTML = childData.releaseDate;
                cell5.innerHTML =
                    '<button class="btn btn-sm btn-warning" id="edit-btn">Edit</button> <button class="btn btn-sm btn-danger" id="delete-btn">Delete</button>';
                cell5
                    .querySelector("#delete-btn")
                    .addEventListener("click", function () {
                        // Get the key of the review to be deleted
                        var reviewKey = childSnapshot.key;
                        // Remove the review from the database
                        deleteReview(reviewKey);
                    });
                cell5
                    .querySelector("#edit-btn")
                    .addEventListener("click", function () {
                        // Populate the form with current values
                        document.getElementById("movieName").value =
                            childData.movieName;
                        document.getElementById("movieRating").value =
                            childData.rating;
                        document.getElementById("movieDirector").value =
                            childData.director;
                        document.getElementById("releaseDate").value =
                            childData.releaseDate;

                        // Show the modal dialog
                        var editModal = new bootstrap.Modal(
                            document.getElementById("editModal")
                        );
                        editModal.show();

                        // Add event listener to save changes button
                        document
                            .getElementById("save-changes-btn")
                            .addEventListener("click", function () {
                                // Get the values from the form fields
                                var movieName =
                                    document.getElementById("movieName").value;
                                var rating = document.getElementById("movieRating").value;
                                var director =

```

```

292         document.getElementById("movieDirector").value;
293     var releaseDate =
294         document.getElementById("releaseDate").value;
295
296     // Update the review
297     childSnapshot.ref.update({
298         movieName: movieName,
299         rating: rating,
300         director: director,
301         releaseDate: releaseDate,
302     });
303
304     // Hide the modal dialog
305     editModal.hide();
306     });
307 });
308 });
309 },
310 function (error) {
311     console.log("Error: " + error.code);
312 }
313 );
314 }
315
316
317

```

Let ascendingOrder = true;

Codeium: Refactor | Explain | Generate Docstring

```

318 function resetArrow(){
319     ratingHeader.innerHTML = "Rating";
320     movieNameHeader.innerHTML = "Movie Name";
321     directorHeader.innerHTML = "Director";
322     releaseDateHeader.innerHTML = "Release Date";
323 }
324

```

Codeium: Refactor | Explain | Generate Docstring

```

325 function sortByRating() {
326     resetArrow();
327     if (ascendingOrder) {
328         ratingHeader.innerHTML = "Rating &#8593";
329     } else {
330         ratingHeader.innerHTML = "Rating &#8595";
331     }
332
333     let table, rows, switching, i, x, y, shouldSwitch;
334     table = document.getElementById("reviews-table");
335     switching = true;
336     while (switching) {
337         switching = false;
338         rows = table.rows;
339         for (i = 1; i < rows.length - 1; i++) {
340             shouldSwitch = false;
341             x = rows[i].getElementsByTagName("TD")[1];
342             y = rows[i + 1].getElementsByTagName("TD")[1];
343             if (ascendingOrder) {
344                 if (Number(x.innerHTML) > Number(y.innerHTML)) {
345                     shouldSwitch = true;
346                     break;
347                 }
348             } else {
349                 if (Number(x.innerHTML) < Number(y.innerHTML)) {
350                     shouldSwitch = true;
351                     break;
352                 }
353             }
354         }
355         if (shouldSwitch) {
356             rows[i].parentNode.insertBefore(rows[i + 1], rows[i]);
357             switching = true;
358         }
359     }
360     ascendingOrder = !ascendingOrder;

```

```
ascendingOrder = !ascendingOrder;  
}
```

Codeium: Refactor | Explain | Generate Docstring

```
function sortByMovieName(){  
  resetArrow();  
  if (ascendingOrder) {  
    movieNameHeader.innerHTML = "Movie Name &#8593";  
  } else {  
    movieNameHeader.innerHTML = "Movie Name &#8595";  
  }  
  
  let table, rows, switching, i, x, y, shouldSwitch;  
  table = document.getElementById("reviews-table");  
  switching = true;  
  while (switching) {  
    switching = false;  
    rows = table.rows;  
    for (i = 1; i < rows.length - 1; i++) {  
      shouldSwitch = false;  
      x = rows[i].getElementsByTagName("TD")[0];  
      y = rows[i + 1].getElementsByTagName("TD")[0];  
      if (ascendingOrder) {  
        if (x.innerHTML.toLowerCase() > y.innerHTML.toLowerCase()) {  
          shouldSwitch = true;  
          break;  
        }  
      } else {  
        if (x.innerHTML.toLowerCase() < y.innerHTML.toLowerCase()) {  
          shouldSwitch = true;  
          break;  
        }  
      }  
    }  
    if (shouldSwitch) {  
      rows[i].parentNode.insertBefore(rows[i + 1], rows[i]);  
      switching = true;  
    }  
  }  
  ascendingOrder = !ascendingOrder;  
}
```

Codeium: Refactor | Explain | Generate Docstring

```
function sortByDirectorName(){  
  resetArrow();  
  if (ascendingOrder) {  
    directorHeader.innerHTML = "Director &#8593";  
  } else {  
    directorHeader.innerHTML = "Director &#8595";  
  }  
  
  let table, rows, switching, i, x, y, shouldSwitch;  
  table = document.getElementById("reviews-table");  
  switching = true;  
  while (switching) {  
    switching = false;  
    rows = table.rows;  
    for (i = 1; i < rows.length - 1; i++) {  
      shouldSwitch = false;  
      x = rows[i].getElementsByTagName("TD")[2];  
      y = rows[i + 1].getElementsByTagName("TD")[2];  
      if (ascendingOrder) {  
        if (x.innerHTML.toLowerCase() > y.innerHTML.toLowerCase()) {  
          shouldSwitch = true;  
          break;  
        }  
      } else {  
        if (x.innerHTML.toLowerCase() < y.innerHTML.toLowerCase()) {  
          shouldSwitch = true;  
          break;  
        }  
      }  
    }  
    if (shouldSwitch) {  
      rows[i].parentNode.insertBefore(rows[i + 1], rows[i]);  
      switching = true;  
    }  
  }  
}
```

```

        if (shouldSwitch) {
            rows[i].parentNode.insertBefore(rows[i + 1], rows[i]);
            switching = true;
        }
    }
    ascendingOrder = !ascendingOrder;
}

```

Codeium: Refactor | Explain | Generate Docstring

```

function sortByReleaseDate(){
    resetArrow();
    if (ascendingOrder) {
        releaseDateHeader.innerHTML = "Release Date &#8593";
    } else {
        releaseDateHeader.innerHTML = "Release Date &#8595";
    }

    let table, rows, switching, i, x, y, shouldSwitch;
    table = document.getElementById("reviews-table");
    switching = true;
    while (switching) {
        switching = false;
        rows = table.rows;
        for (i = 1; i < rows.length - 1; i++) {
            shouldSwitch = false;
            x = rows[i].getElementsByTagName("TD")[3];
            y = rows[i + 1].getElementsByTagName("TD")[3];
            if (ascendingOrder) {
                if (x.innerHTML.toLowerCase() > y.innerHTML.toLowerCase()) {
                    shouldSwitch = true;
                    break;
                }
            } else {
                if (x.innerHTML.toLowerCase() < y.innerHTML.toLowerCase()) {
                    shouldSwitch = true;
                    break;
                }
            }
        }
        if (shouldSwitch) {
            rows[i].parentNode.insertBefore(rows[i + 1], rows[i]);
            switching = true;
        }
    }
    ascendingOrder = !ascendingOrder;
}

```

```

// When the user submits the review form
document
    .getElementById("review-form")
    .addEventListener("submit", function (event) {
        event.preventDefault(); // Prevent the default form submit behavior

```

```

        // Get the values from the input fields
        var movieName = document.getElementById("movie-name").value;
        var rating = document.getElementById("rating").value;
        var director = document.getElementById("director").value;
        var releaseDate = document.getElementById("release-date").value;

```

```

        // Call the addReview function with the values
        addReview(movieName, rating, director, releaseDate);
        showFirestoreData();
    });

```

```

</script>

```

```

<!-- Bootstrap JS -->

```

```

<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"></script>

```

```

<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></script>

```

```

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js"></script>

```

```

</body>

```

```

</html>

```

➔ Now when you press refresh:

## Movie Reviews

Movie Name	Rating	Director	Release Date	Actions	
Oppenheimer	5	Christopher Nolan	2023-07-23	Edit	Delete
Life of CS Student	5	Nayan Raj Khanal	2001-03-16	Edit	Delete

Movie Name:

Rating (0-5):

Director Name:

Release Date:

Add Review

➔ Now let's try adding a new review:

## Movie Reviews

Movie Name	Rating	Director	Release Date	Actions
Oppenheimer	5	Christopher Nolan	2023-07-23	<button>Edit</button> <button>Delete</button>
Life of CS Student	5	Nayan Raj Khanal	2001-03-16	<button>Edit</button> <button>Delete</button>
Testing	1	Tester	2023-05-16	<button>Edit</button> <button>Delete</button>

Movie Name:

Rating (0-5):

Director Name:

Release Date:

We can see it's being added,

and if we check the database:

task5-2227486

?

## Cloud Firestore


Data

Rules


Indexes

Usage

Extensions NEW


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

[Configure App Check](#)







Panel view

Query builder

 > Reviews > 1iX7XfB1lZRz1...

[More in Google Cloud](#)

 task5-2227486-be9ed	 Reviews 	 1iX7XfB1lZRz1G3XY034
<a href="#">+ Start collection</a>	<a href="#">+ Add document</a>	<a href="#">+ Start collection</a>
Reviews >	1iX7XfB1lZRz1G3XY034 efXnkSxXoWJhNJQhv3f taNkFp7uBAzZpkPsZpk	<a href="#">+ Add field</a> director: "Tester" movieName: "Testing" rating: 1 releaseDate: "2023-05-16"

➔ Now let's try editing:

Swapped "Test" and "Testing" and we can see it's being edited.

## Movie Reviews

Movie Name	Rating	Director	Release Date	Actions	
Oppenheimer	5	Christopher Nolan	2023-07-23	Edit	Delete
Life of CS Student	5	Nayan Raj Khanal	2001-03-16	Edit	Delete
Test	5	Testing	2001-03-16	Edit	Delete

Movie Name:

Rating (0-5):

Director Name:

Release Date:

Add Review



It's also being updated in database:

task5-2227486

?

## Cloud Firestore


Data

Rules


Indexes

Usage

Extensions NEW


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


[Configure App Check](#)








Panel view

Query builder

 > Reviews > 1iX7XfB1lZRz1...

 More in Google Cloud

 task5-2227486-be9ed	 Reviews 	 1iX7XfB1lZRz1G3XY034 
<a href="#">+ Start collection</a>	<a href="#">+ Add document</a>	<a href="#">+ Start collection</a>
Reviews >	1iX7XfB1lZRz1G3XY034 efXnkSxXoWJhNJQhv3t taNkFp7uBAzZpkPsZpk	<a href="#">+ Add field</a> director: "Testing" movieName: "Tester" rating: 1 releaseDate: "2023-05-16"

➔ Now let's try to delete:

We can see the entry is being deleted.

## Movie Reviews


Movie Name	Rating	Director	Release Date	Actions	
Oppenheimer	5	Christopher Nolan	2023-07-23	Edit	Delete
Life of CS Student	5	Nayan Raj Khanal	2001-03-16	Edit	Delete

Movie Name:

Rating (0-5):

Director Name:

Release Date:

Add Review

Now to see in the database:

The screenshot shows the Google Cloud Firestore console for project 'task5-2227486'. The 'Data' tab is selected, showing a collection named 'Reviews'. A document with ID 'efXnkSxXoWJhNJQhv3t8' is selected, displaying its fields: 'director' (Christopher Nolan), 'movieName' (Oppenheimer), 'rating' (5), and 'releaseDate' (07-23-2023). The document is highlighted in grey, indicating it is the selected item.

task5-2227486 ▾

Cloud Firestore

Data Rules Indexes Usage Extensions NEW

Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing [Configure App Check](#) ✕

Panel view Query builder

Home > Reviews > efXnkSxXoWJh...

task5-2227486-be9ed Reviews efXnkSxXoWJhNJQhv3t8

+ Start collection + Add document + Start collection

Reviews > efXnkSxXoWJhNJQhv3t8 + Add field

director: "Christopher Nolan"

movieName: "Oppenheimer"

rating: 5

releaseDate: "07-23-2023"

It is being deleted from the database too.

With that task 5 has been concluded.