

Assignment: Task 3

In a similar approach to Workshop on Google Firestore, create a “movies” review HTML/JavaScript web app, based on a Google Firestore NoSQL database. The web app should access the movie review data store in a Google Firestore NoSQL database via JavaScript and present the information via HTML on the web browser. The web app should have the ability to add the movie name, a rating score from 0 to 5 (integers only), the director’s name and the release date. It should have the ability to be sorted in order on any of the fields. It should have the ability to edit and delete individual reviews. There is no requirement for any user authentication for the web app. You should host your application on Aws S3 bucket.

=> Ans:

Firebase:

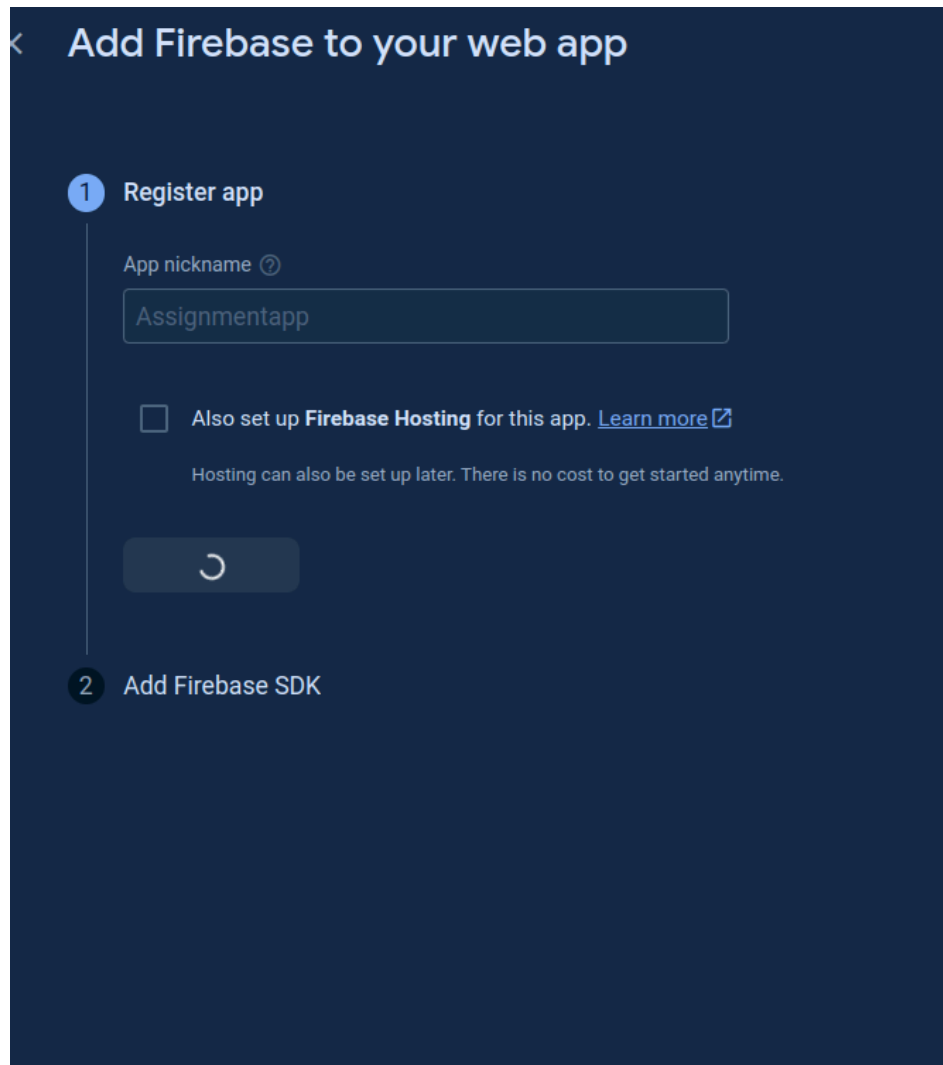
Firebase, created by Google, helps developers build mobile and web apps easily. It provides tools like real-time databases, user login, cloud storage, and machine learning. With Firebase, developers don't need to write server-side code for common tasks, making development faster and simpler. This platform lets developers concentrate on creating great user experiences without worrying about infrastructure. It's particularly useful for startups and small teams who need to build apps quickly and efficiently.

Firestore:

Firestore is a NoSQL database from Firebase, designed for apps that need to scale and work in real-time. It supports flexible data structures, making it great for apps like chat services and online stores. Firestore keeps data synchronized across devices and works offline, so apps stay functional without an internet connection. It integrates smoothly with other Firebase tools like user login and analytics. Firestore automatically scales, performs well, and is secure, making it a strong choice for modern app development.

Let's start with assignment now:

First select a web app option in firebase console and give name to your application. Here, i am giving name Assignmentapp as my application name and i am not allowing the hosting setup provided by firebase.



The screenshot shows the 'Add Firebase to your web app' wizard in the Firebase console. The title is 'Add Firebase to your web app'. The first step, '1 Register app', is active. It contains a label 'App nickname' with a help icon, a text input field containing 'Assignmentapp', and a checkbox labeled 'Also set up **Firebase Hosting** for this app. [Learn more](#)'. Below the checkbox is a note: 'Hosting can also be set up later. There is no cost to get started anytime.' A large button with a circular arrow icon is positioned below the checkbox. The second step, '2 Add Firebase SDK', is visible at the bottom but not yet active.

< Add Firebase to your web app

1 Register app

App nickname ?

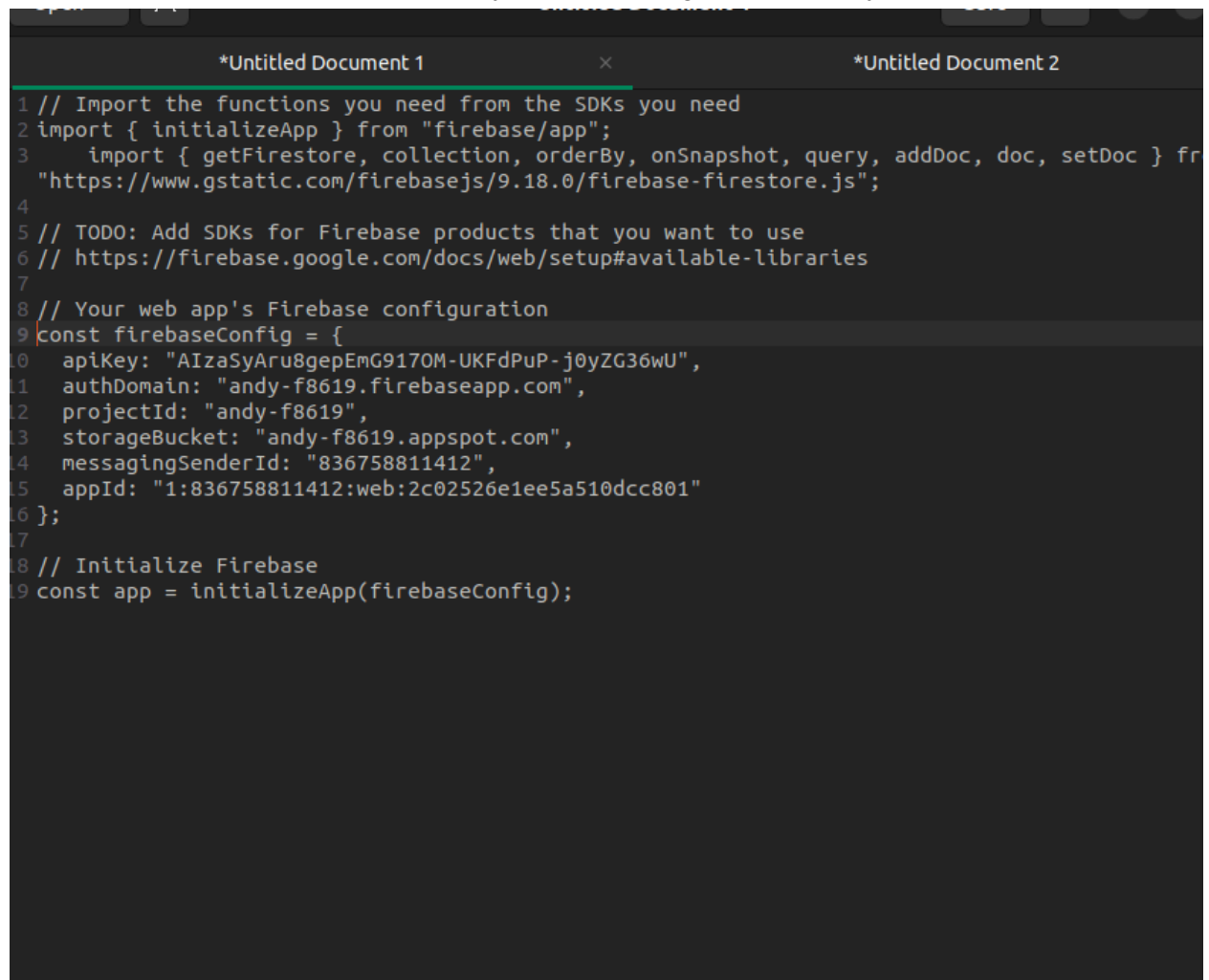
Assignmentapp

☐ 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.

2 Add Firebase SDK

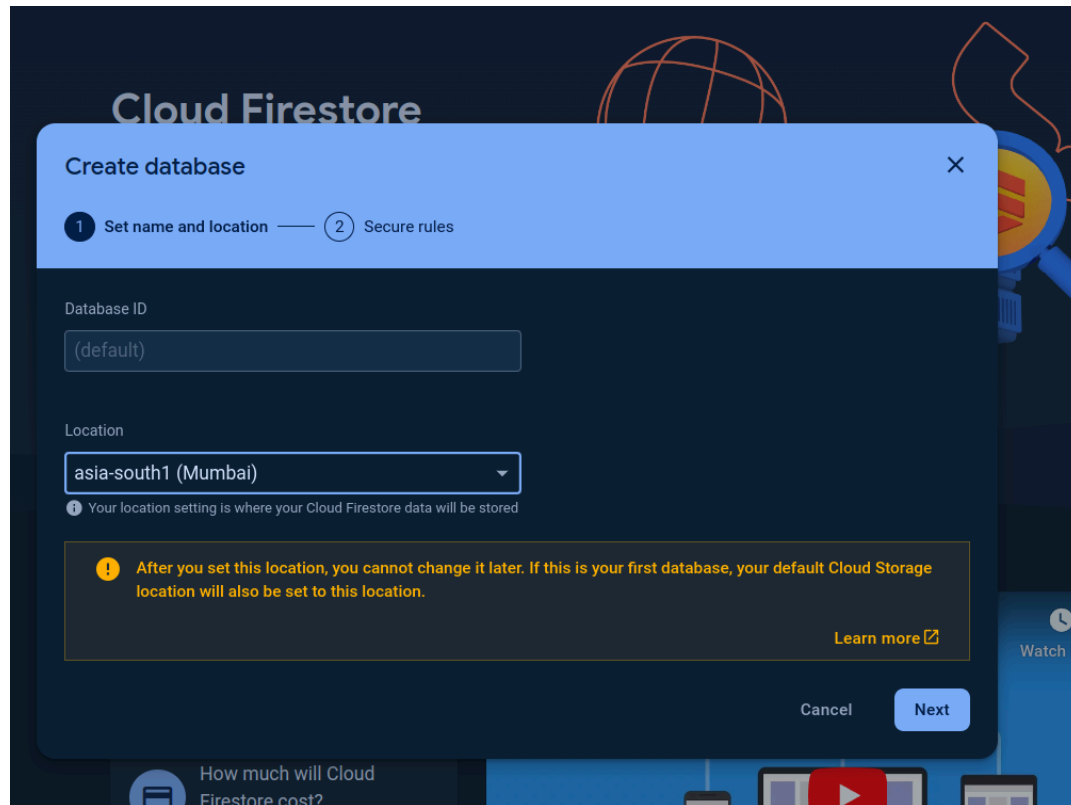
And later on i have copied and paste my firebase configuration into my text editor.

A screenshot of a text editor with a dark theme. The editor has two tabs at the top: '*Untitled Document 1' and '*Untitled Document 2'. The first tab is active and contains JavaScript code for initializing Firebase. The code is as follows:

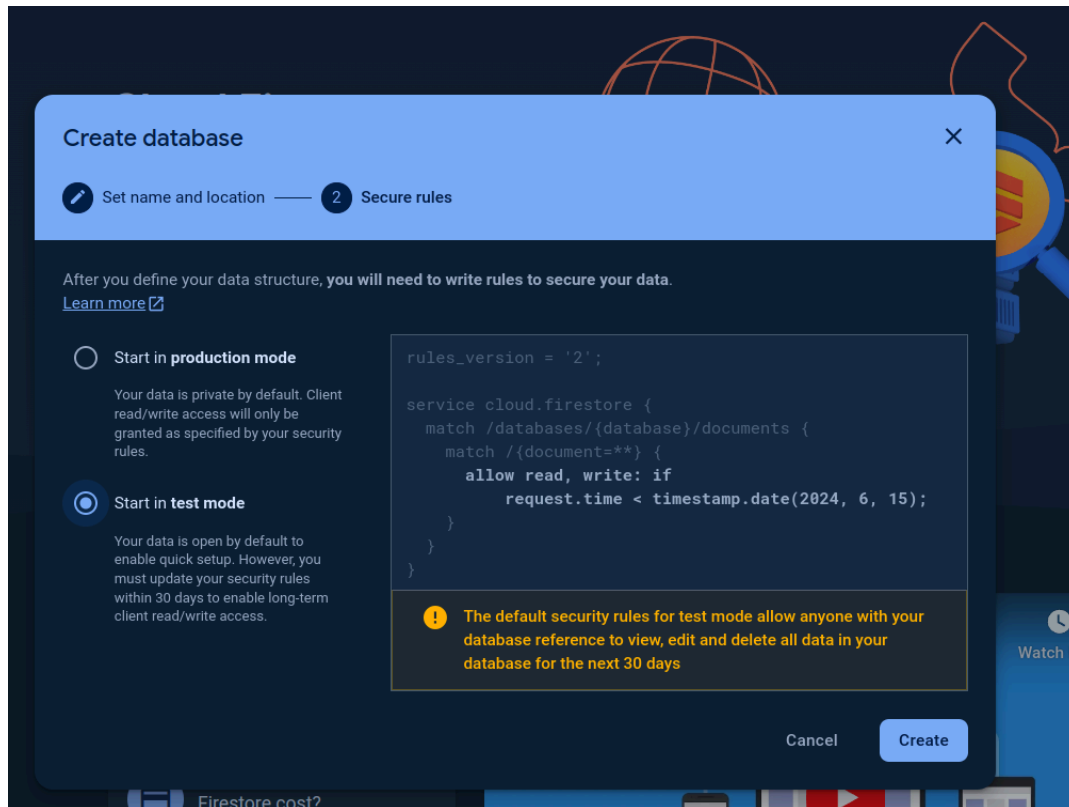
```
1 // Import the functions you need from the SDKs you need
2 import { initializeApp } from "firebase/app";
3   import { getFirestore, collection, orderBy, onSnapshot, query, addDoc, doc, setDoc } fr
   "https://www.gstatic.com/firebasejs/9.18.0/firebase-firestore.js";
4
5 // TODO: Add SDKs for Firebase products that you want to use
6 // https://firebase.google.com/docs/web/setup#available-libraries
7
8 // Your web app's Firebase configuration
9 const firebaseConfig = {
10   apiKey: "AIzaSyAru8gepEmG9170M-UKFdPuP-j0yZG36wU",
11   authDomain: "andy-f8619.firebaseio.com",
12   projectId: "andy-f8619",
13   storageBucket: "andy-f8619.appspot.com",
14   messagingSenderId: "836758811412",
15   appId: "1:836758811412:web:2c02526e1ee5a510dcc801"
16 };
17
18 // Initialize Firebase
19 const app = initializeApp(firebaseConfig);
```

Configuring Database

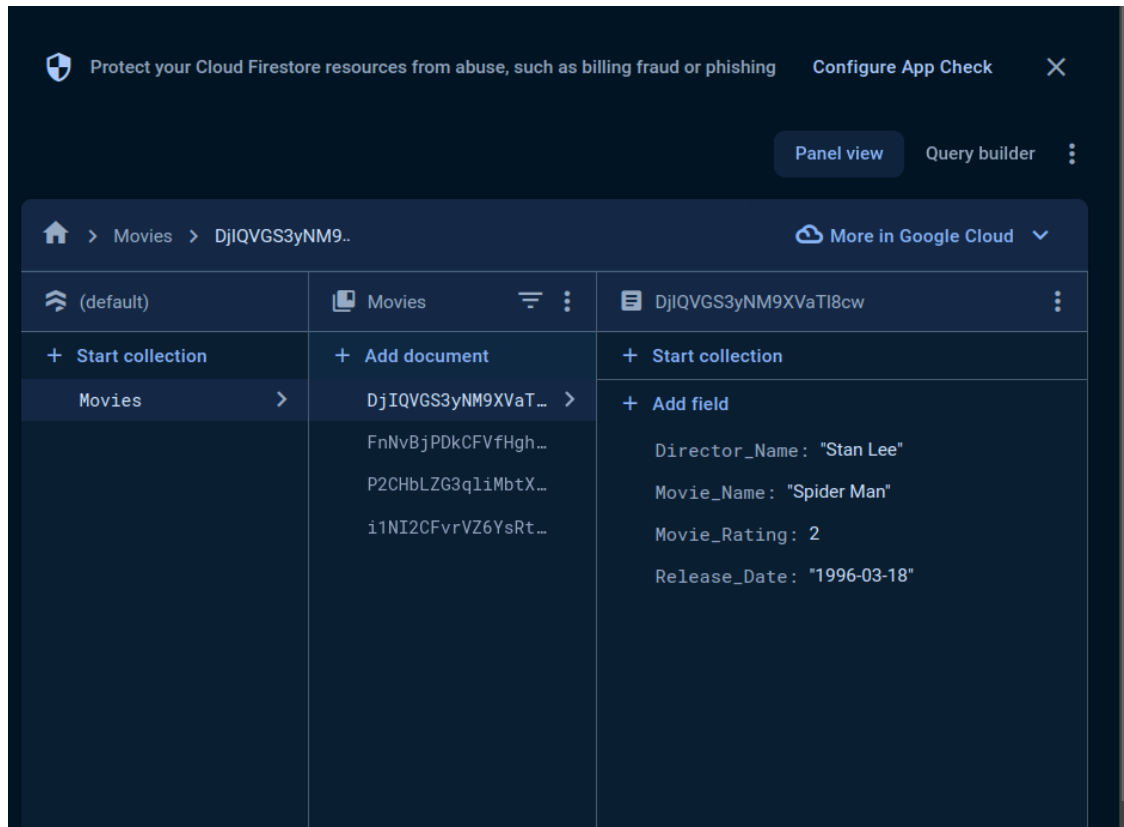
Then i navigate to build option and select firestore database. Then while creating database i choose the location into mumbai server as i am from Nepal and Mumbai is the closest one for me which will provide me with best peformance.



And for rules section, here i choose test section as test mode in Firestore provides relaxed security rules, allowing easy read and write access to the database during development and testing. It simplifies setup, promotes flexibility for experimenting with data, and is ideal for rapid iteration and debugging without strict access restrictions.



And here i give my database name as “Movies” and generate auto id with fields name Movie_Name=”String”, Director_Name=”String”, Release_date=”Timestamp” and Movie_rating=”number”and filled the input fields respectively. Now these are my data inside my database. Here, i have four similar types of info into my database



Accessing data from javascript

Here, i am using vs code editor for editing my java script file. This section of code includes basic html file where we can see four fields for movie name, director name, release date and review and then a add button. And on lower side we can see a tabular view for displaying fetched date from database under topic movie name, rating, director , release date and another one is action on which you can choose and perform edit or delete option on each row.

```
<!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">
<title>My Firebase app</title>
</head>
<body>
<div class="container">
<h1 id="mainTitle">My movies</h1>
<div class="d-flex mb-2">
    <input type="text" class="form-control"
id="movieName" placeholder="Movie name">
```

```
<input type="text" class="form-control mx-2"
id="directorName" placeholder="Director's name">
  <input type="date" class="form-control mx-2"
id="releaseDate">
  <select class="form-control mx-2 w-25"
id="movieRating">
    <option value="0">0/5</option>
    <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">
<thead>
  <tr>
    <th><button class="btn btn-link p-0"
id="sortByName">Movie Name</button></th>
    <th><button class="btn btn-link p-0"
id="sortByRating">Rating</button></th>
    <th><button class="btn btn-link p-0"
id="sortByDirector">Director</button></th>
    <th><button class="btn btn-link p-0"
id="sortByReleaseDate">Release Date</button></th>
    <th>Actions</th>
```



```

        </tr>
</thead>
<tbody id="movieList">
</tbody>
</table>
</div>

<!-- Edit Modal -->
<div class="modal fade" id="editModal"
tabindex="-1" aria-labelledby="editModalLabel"
aria-hidden="true">
<div class="modal-dialog">
    <div class="modal-content">
        <div class="modal-header">
            <h5 class="modal-title"
id="editModalLabel">Edit Movie</h5>
            <button type="button" class="btn-close"
data-bs-dismiss="modal"
aria-label="Close"></button>
        </div>
        <div class="modal-body">
            <input type="hidden" id="editMovieId">
            <input type="text" class="form-control mb-2"
id="editMovieName" placeholder="Movie name">
            <input type="text" class="form-control mb-2"
id="editDirectorName" placeholder="Director's
name">

```

```
        <input type="date" class="form-control mb-2"
id="editReleaseDate">
        <select class="form-control mb-2"
id="editMovieRating">
            <option value="0">0/5</option>
            <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>
    </div>
    <div class="modal-footer">
        <button type="button" class="btn
btn-secondary"
data-bs-dismiss="modal">Close</button>
        <button type="button" class="btn
btn-primary" id="saveChangesButton">Save
changes</button>
    </div>
</div>
</div>
```

Here are different imported queries for jquery, bootstrap and firebase

```
!-- 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>

<!-- Firebase -->
<script
src="https://www.gstatic.com/firebasejs/9.0.0/firebase-app-compat.js"></script>
<script
src="https://www.gstatic.com/firebasejs/9.0.0/firebase-firestore-compat.js"></script>
```

This is the main script that we are gonna perform in html section. Firstly we will be able to view data into table that are fetched from firestore and on top of it we can see the count of movies we have on database. And here we have attached edit and delete button for each row fetched from firestore

```
<script>
  // Your web app's Firebase configuration
  const firebaseConfig = {
    apiKey: "AIzaSyAru8gepEmG917OM-UKFdPuP-j0yZG36wU",
    authDomain: "andy-f8619.firebaseio.com",
    projectId: "andy-f8619",
    storageBucket: "andy-f8619.appspot.com",
    messagingSenderId: "836758811412",
    appId: "1:836758811412:web:2c02526e1ee5a510dcc801"
  };

  // Initialize Firebase
  firebase.initializeApp(firebaseConfig);
  const db = firebase.firestore();

  let currentSortField = 'Movie_Name';
  let currentSortDirection = 'asc';

  function loadMovies() {
    const q =
db.collection("Movies").orderBy(currentSortField,
currentSortDirection);
    q.onSnapshot((snapshot) => {
```

```

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

        // Loop through snapshot data and add to
HTML table
        var tableRows = '';
        snapshot.forEach((doc) => {
            tableRows += '<tr>';
            tableRows += '<td>' +
doc.data().Movie_Name + '</td>';
            tableRows += '<td>' +
doc.data().Movie_Rating + '/5</td>';
            tableRows += '<td>' +
doc.data().Director_Name + '</td>';
            tableRows += '<td>' +
doc.data().Release_Date + '</td>';
            tableRows += `<td>
                                <button class="btn
btn-warning btn-sm editButton" data-id="${doc.id}"
data-name="${doc.data().Movie_Name}"
data-rating="${doc.data().Movie_Rating}"
data-director="${doc.data().Director_Name}"
data-release="${doc.data().Release_Date}">Edit</but
ton>
                                <button class="btn
btn-danger btn-sm deleteButton"
data-id="${doc.id}">Delete</button>

```

```
                </td>`;
            tableRows += '</tr>';
        });

        $('#movieList').append(tableRows);

        // Display movie count
        $('#mainTitle').html(snapshot.size + "
movies in the list");

        // Attach event listeners for edit and
delete buttons
        $(".editButton").click(function() {
            const id = $(this).data('id');
            const name = $(this).data('name');
            const rating =
$(this).data('rating');
            const director =
$(this).data('director');
            const release =
$(this).data('release');

            $('#editMovieId').val(id);
            $('#editMovieName').val(name);
            $('#editMovieRating').val(rating);

            $('#editDirectorName').val(director);
            $('#editReleaseDate').val(release);
```

```

        $('#editModal').modal('show');
    });

    $(".deleteButton").click(function() {
        const id = $(this).data('id');

        db.collection("Movies").doc(id).delete().then(() => {
            loadMovies();
        }).catch((error) => {
            console.error("Error deleting
document: ", error);
        });
    });
});

loadMovies();

```

From this functionality you will be able to add the database details into firebase database on click of add button

```

// Add button pressed
$("#addButton").click(function() {
    // Add movie to Firestore collection
    db.collection("Movies").add({
        Movie_Name: $("#movieName").val(),

```

```

        Movie_Rating:
parseInt($("#movieRating").val()),
        Director_Name: $("#directorName").val(),
        Release_Date: $("#releaseDate").val()
    })).then(() => {
        // Reset form
        $("#movieName").val('');
        $("#movieRating").val('0');
        $("#directorName").val('');
        $("#releaseDate").val('');
    }).catch((error) => {
        console.error("Error adding document: ",
error);
    });
});

```

And here is the code that will result the edit on row side

```

// Save changes button in edit modal
$("#saveChangesButton").click(function() {
    const id = $('#editMovieId').val();
    const name = $('#editMovieName').val();
    const rating =
parseInt($('#editMovieRating').val());
    const director =
$('#editDirectorName').val();
    const release = $('#editReleaseDate').val();

    db.collection("Movies").doc(id).update({

```



```

        Movie_Name: name,
        Movie_Rating: rating,
        Director_Name: director,
        Release_Date: release
    })).then(() => {
        $('#editModal').modal('hide');
        loadMovies();
    }).catch((error) => {
        console.error("Error updating document:
", error);
    });
});

```

And this last one is for sorting functionalities. On clicking the table header of any movie name or director name or release date or movie rating your table will be sorted according to click.

```

// Sorting functionality
$("#sortByName").click(function() {
    currentSortField = 'Movie_Name';
    currentSortDirection = (currentSortDirection
=== 'asc') ? 'desc' : 'asc';
    loadMovies();
});

$("#sortByRating").click(function() {
    currentSortField = 'Movie_Rating';
    currentSortDirection = (currentSortDirection
=== 'asc') ? 'desc' : 'asc';
    loadMovies();
});

```

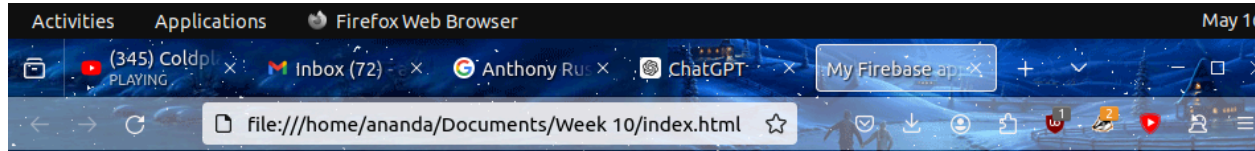
```
});

$("#sortByDirector").click(function() {
    currentSortField = 'Director_Name';
    currentSortDirection = (currentSortDirection
=== 'asc') ? 'desc' : 'asc';
    loadMovies();
});

$("#sortByReleaseDate").click(function() {
    currentSortField = 'Release_Date';
    currentSortDirection = (currentSortDirection
=== 'asc') ? 'desc' : 'asc';
    loadMovies();
});
</script>
</body>
</html>
```

Accessing the index.html file

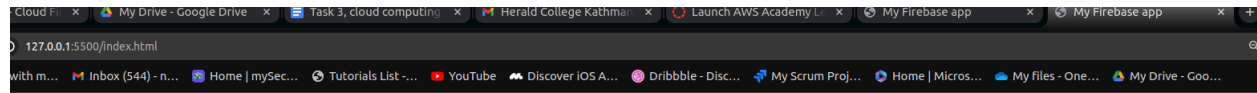
Here is the recent view of html file . There are 4 movies in firebase database which are fetched here.



4 movies in the list

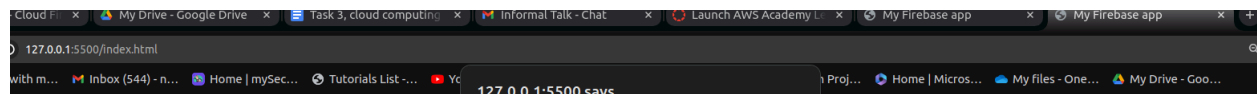
Movie name	Director's name	mm / dd / yyyy	0/5	Add
<u>Movie Name</u>	<u>Rating</u>	<u>Director</u>	<u>Release Date</u>	<u>Actions</u>
BATman	5/5	Christopher Nolan	2008-11-07	<button>Edit</button> <button>Delete</button>
Avengers	4/5	Anthony russo	2014-03-06	<button>Edit</button> <button>Delete</button>
Spider Man	2/5	Stan Lee	1996-03-18	<button>Edit</button> <button>Delete</button>
Dune	0/5	Denis Villeneuve	2019-05-03	<button>Edit</button> <button>Delete</button>

And yes every fields are compulsory to fill before adding other wise it will throw alert to fill all the fields



7 movies in the list

Movie Name	Rating	Director	Release Date	Actions
Avengers	4/5	Anthony russo	2014-03-06	<button>Edit</button> <button>Delete</button>
Dune_edited	4/5	Denis Villene	2019-05-03	<button>Edit</button> <button>Delete</button>
Spider Man	2/5	Stan Lee	1996-03-18	<button>Edit</button> <button>Delete</button>
Superman	3/5	Bret lee	2020-02-16	<button>Edit</button> <button>Delete</button>
Titanic	3/5	Cameron	1998-11-01	<button>Edit</button> <button>Delete</button>
jatrai jatra	1/5	Pradip Bhattarai	2024-05-07	<button>Edit</button> <button>Delete</button>
war	2/5	sidharth ananda	2022-06-09	<button>Edit</button> <button>Delete</button>

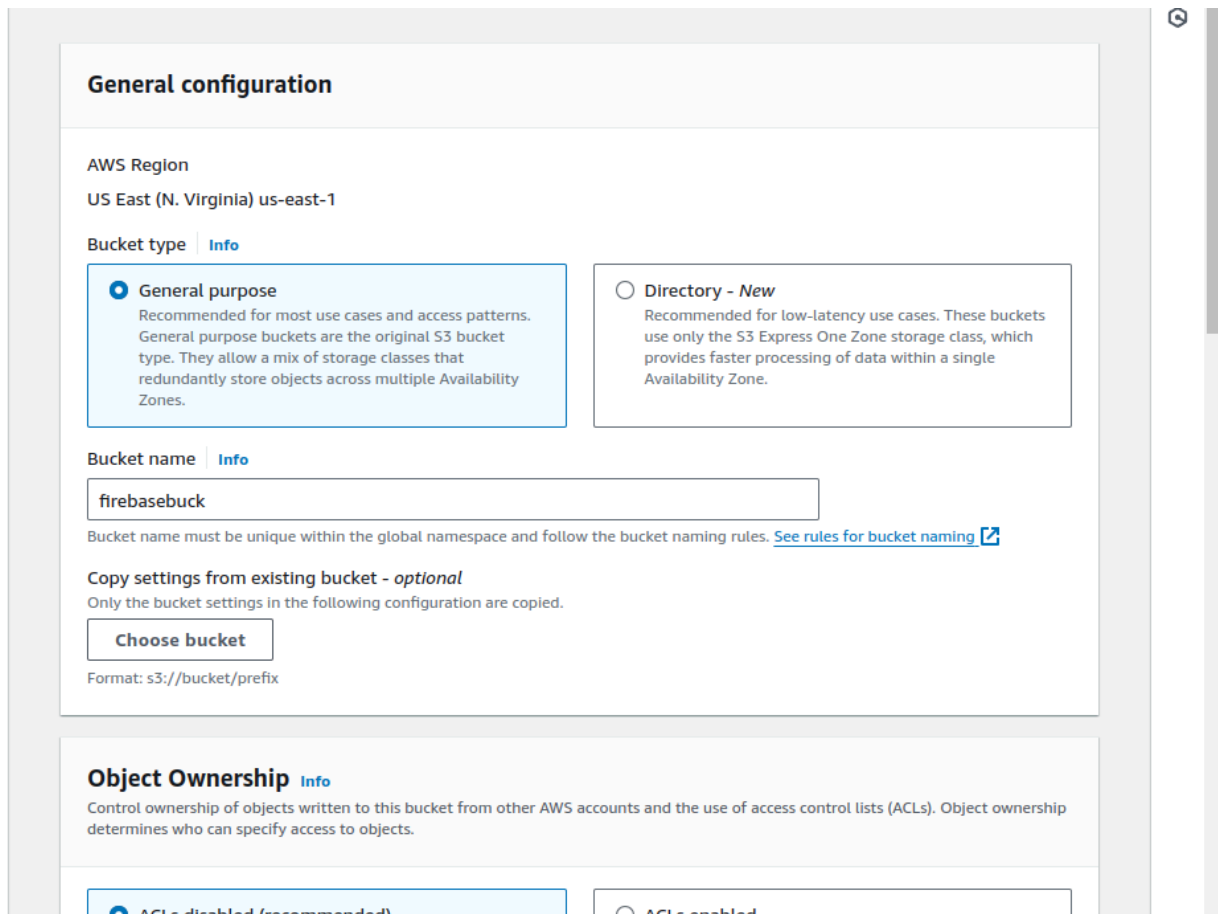


7 movies in the list

Movie Name	Rating	Director	Release Date	Actions
Avengers	4/5	Anthony russo	2014-03-06	<button>Edit</button> <button>Delete</button>
Dune_edited	4/5	Denis Villene	2019-05-03	<button>Edit</button> <button>Delete</button>
Spider Man	2/5	Stan Lee	1996-03-18	<button>Edit</button> <button>Delete</button>
Superman	3/5	Bret lee	2020-02-16	<button>Edit</button> <button>Delete</button>
Titanic	3/5	Cameron	1998-11-01	<button>Edit</button> <button>Delete</button>
jatrai jatra	1/5	Pradip Bhattarai	2024-05-07	<button>Edit</button> <button>Delete</button>
war	2/5	sidharth ananda	2022-06-09	<button>Edit</button> <button>Delete</button>

Now lets host this site into aws s3 bucket

First redirect to aws console and search for s3 in service menu. After clicking on s3 choose create bucket option. Give a name to your bucket, here i am giving **firebasebucket**. I am keeping all the other fields as default and press the button create.



The screenshot shows the 'General configuration' section of the AWS S3 'Create bucket' wizard. The 'AWS Region' is set to 'US East (N. Virginia) us-east-1'. Under 'Bucket type', the 'General purpose' option is selected with a radio button. The 'Bucket name' field contains 'firebasebucket'. Below this, there is a note about bucket naming rules and a link 'See rules for bucket naming'. The 'Copy settings from existing bucket - optional' section has a 'Choose bucket' button. At the bottom, the 'Object Ownership' section shows the 'ACLs disabled (recommended)' option selected.

General configuration

AWS Region
US East (N. Virginia) us-east-1

Bucket type [Info](#)

☒ **General purpose**
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

☐ **Directory - New**
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name [Info](#)
firebasebucket

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

[Choose bucket](#)

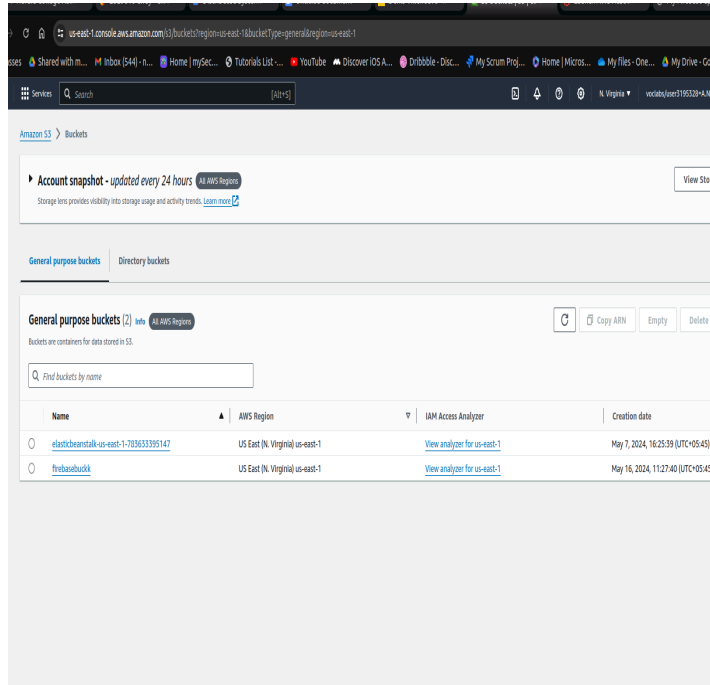
Format: s3://bucket/prefix

Object Ownership [Info](#)
Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

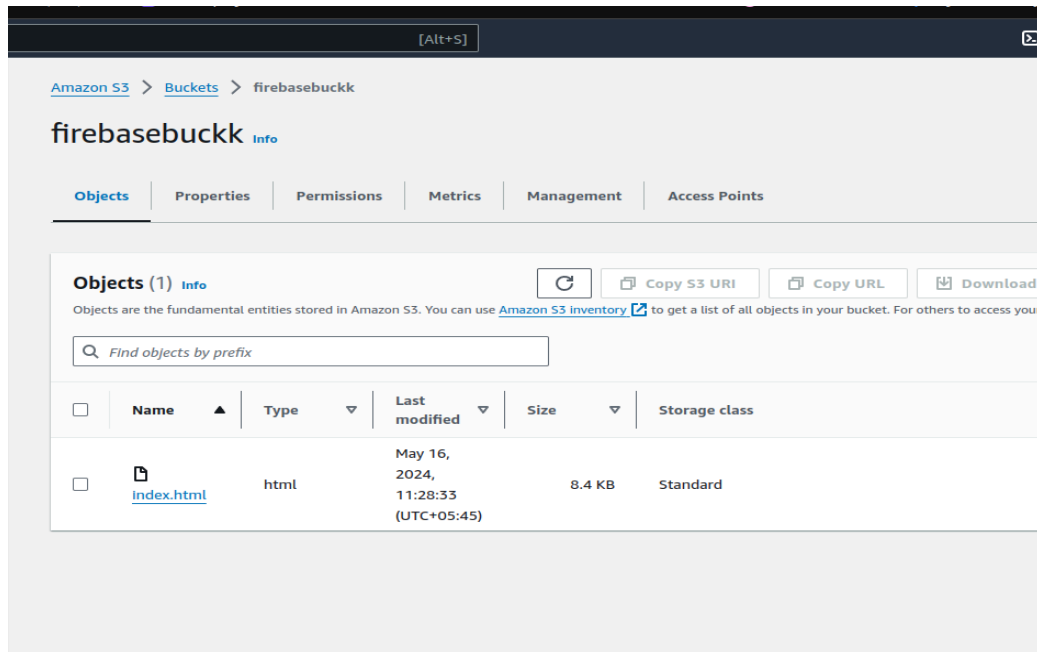
☒ **ACLs disabled (recommended)**

☐ **ACLs enabled**

After your bucket is successfully created, then select the bucket name and choose the option upload and deploy.



After clicking on bucket name, i have uploaded my index.html file, which you can see on next page.



Then go back to your bucket section and click on bucket name. In details view you can see a label name object url.

The screenshot shows the Amazon S3 console interface. The breadcrumb navigation at the top reads: Amazon S3 > Buckets > firebasebucket > index.html. The main heading is "index.html" with an "Info" link. To the right of the heading are three buttons: "Copy S3 URI", "Download", and "Open". Below the heading is a tabbed interface with "Properties", "Permissions", and "Versions". The "Properties" tab is active, displaying an "Object overview" section. This section is divided into two columns. The left column contains: Owner (awslabsc0w497213111670409279), AWS Region (US East (N. Virginia) us-east-1), Last modified (May 16, 2024, 11:28:33 (UTC+05:45)), Size (8.4 KB), Type (html), and Key (index.html). The right column contains: S3 URI (s3://firebasebucket/index.html), Amazon Resource Name (ARN) (arn:aws:s3:::firebasebucket/index.html), Entity tag (ETag) (a7158a813d9675419c4bb7abb9ad3878), and Object URL (https://firebasebucket.s3.amazonaws.com/index.html). At the bottom of the console, there is a section titled "Object management overview".

Amazon S3 > Buckets > firebasebucket > index.html

index.html [Info](#) [Copy S3 URI](#) [Download](#) [Open](#)

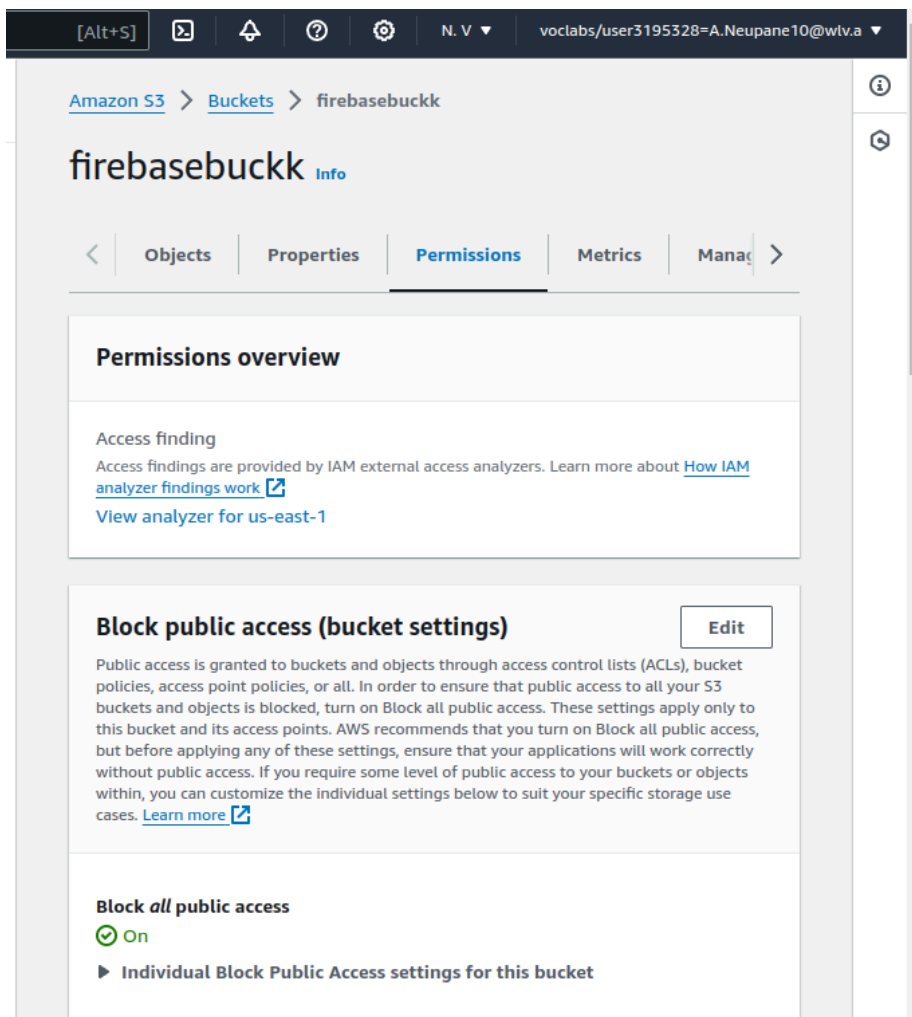
Properties | Permissions | Versions

Object overview

Owner	awslabsc0w497213111670409279	S3 URI	s3://firebasebucket/index.html
AWS Region	US East (N. Virginia) us-east-1	Amazon Resource Name (ARN)	arn:aws:s3:::firebasebucket/index.html
Last modified	May 16, 2024, 11:28:33 (UTC+05:45)	Entity tag (ETag)	a7158a813d9675419c4bb7abb9ad3878
Size	8.4 KB	Object URL	https://firebasebucket.s3.amazonaws.com/index.html
Type	html		
Key	index.html		

Object management overview

Copy the url and redirect to the page this will throw you an error as we have block the public access. Now redirect to the permission section



The screenshot shows the AWS Management Console interface for the 'firebasebuckk' bucket. The breadcrumb navigation at the top indicates the path: Amazon S3 > Buckets > firebasebuckk. The bucket name 'firebasebuckk' is displayed with an 'Info' link. Below this, a horizontal tab bar contains 'Objects', 'Properties', 'Permissions' (which is the active tab), 'Metrics', and 'Managed Objects'. The 'Permissions overview' section is visible, containing an 'Access finding' card with a link to 'View analyzer for us-east-1'. The 'Block public access (bucket settings)' section is also visible, featuring an 'Edit' button and a paragraph explaining that public access is blocked. Below this, the 'Block all public access' setting is shown as 'On' with a green checkmark icon. A section titled 'Individual Block Public Access settings for this bucket' is partially visible at the bottom.

Amazon S3 > Buckets > firebasebuckk

firebasebuckk [Info](#)

< Objects Properties **Permissions** Metrics Managed Objects >

Permissions overview

Access finding

Access findings are provided by IAM external access analyzers. Learn more about [How IAM analyzer findings work](#)

[View analyzer for us-east-1](#)

Block public access (bucket settings) [Edit](#)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access

✔ On

► Individual Block Public Access settings for this bucket

Click on the check box and remove the tick mark and then click on save changes which we allow us to redirect to the url.

Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

☐ **Block all public access**

Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

☐ **Block public access to buckets and objects granted through new access control lists (ACLs)**

S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.

☐ **Block public access to buckets and objects granted through any access control lists (ACLs)**

S3 will ignore all ACLs that grant public access to buckets and objects.

☐ **Block public access to buckets and objects granted through new public bucket or access point policies**

S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

☐ **Block public and cross-account access to buckets and objects through any public bucket or access point policies**

S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Cancel Save changes

Amazon S3 > Buckets > firebasebuckk

firebasebuckk [Info](#)

< Objects Properties Permissions Metrics Manage >

Permissions overview

Access finding
Access findings are provided by IAM external access analyzers. Learn more about [How IAM analyzer findings work](#)
[View analyzer for us-east-1](#)

Block public access (bucket settings) Edit

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Block all public access
Off

► Individual Block Public Access settings for this bucket

Bucket policy Edit Delete

Then just below there is bucket policy section edit your bucket policy as shown in the picture and press save changes this will update your url to be accessible from everywhere.


[Amazon S3](#) > [Buckets](#) > [firebasebuckk](#) > Edit bucket policy

Edit bucket policy [Info](#) [Info](#)

Bucket policy

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to o

Bucket ARN

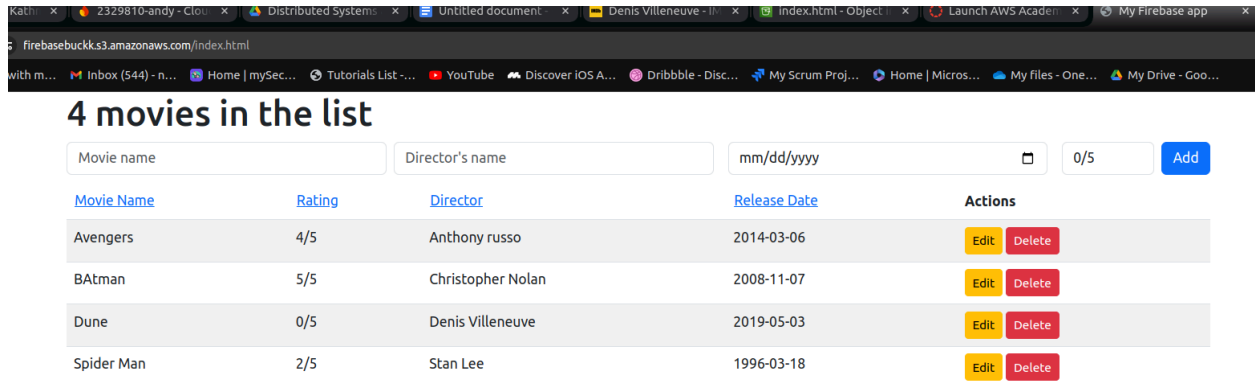
 `arn:aws:s3:::firebasebuckk`

Policy

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Sid": "Statement1",
6       "Principal": "*",
7       "Effect": "Allow",
8       "Action": "S3:GetObject",
9       "Resource": "arn:aws:s3:::firebasebuckk/*"
10    }
11  ]
12 }
```

Lastly, press on the url link and you will see your site has been hosted on s3 bucket. Here i am gonna show you all the actions you can perform within website

1. First face of url

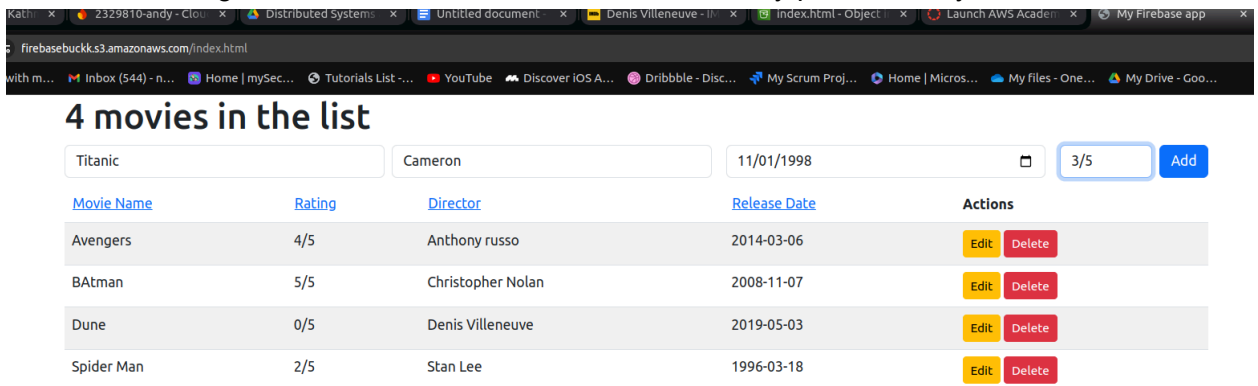


The screenshot shows a web browser with the URL `firebasebuckets3.amazonaws.com/index.html`. The page title is "4 movies in the list". Below the title, there are three input fields: "Movie name", "Director's name", and "mm/dd/yyyy". To the right of these fields is a counter "0/5" and an "Add" button. Below the input fields is a table with 5 columns: "Movie Name", "Rating", "Director", "Release Date", and "Actions". The table contains 4 rows of movie data.

Movie Name	Rating	Director	Release Date	Actions
Avengers	4/5	Anthony russo	2014-03-06	Edit Delete
BATman	5/5	Christopher Nolan	2008-11-07	Edit Delete
Dune	0/5	Denis Villeneuve	2019-05-03	Edit Delete
Spider Man	2/5	Stan Lee	1996-03-18	Edit Delete

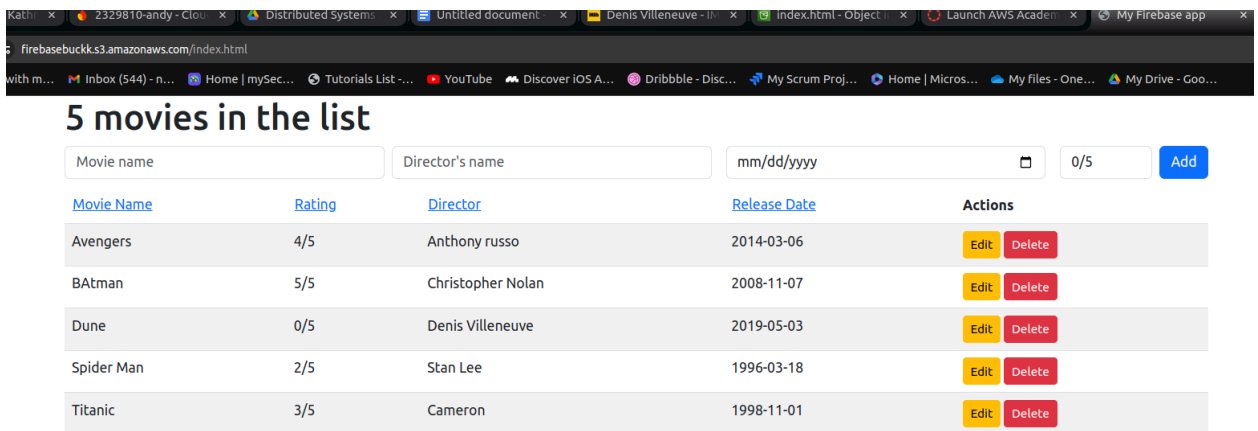
2. Adding a information

Here, i am adding info for movie titanic, which is successfully performed by the site



The screenshot shows the same website as before, but with the "Add" button highlighted. The "Movie name" field now contains "Titanic", the "Director's name" field contains "Cameron", and the "mm/dd/yyyy" field contains "11/01/1998". The counter now shows "3/5".

Movie Name	Rating	Director	Release Date	Actions
Avengers	4/5	Anthony russo	2014-03-06	Edit Delete
BATman	5/5	Christopher Nolan	2008-11-07	Edit Delete
Dune	0/5	Denis Villeneuve	2019-05-03	Edit Delete
Spider Man	2/5	Stan Lee	1996-03-18	Edit Delete



The screenshot shows the website after adding the movie "Titanic". The title is now "5 movies in the list". The "Add" button is no longer highlighted. The "Movie name" field is empty, and the "Director's name" field contains "Cameron". The "mm/dd/yyyy" field contains "mm/dd/yyyy". The counter now shows "0/5". The table now contains 5 rows of movie data.

Movie Name	Rating	Director	Release Date	Actions
Avengers	4/5	Anthony russo	2014-03-06	Edit Delete
BATman	5/5	Christopher Nolan	2008-11-07	Edit Delete
Dune	0/5	Denis Villeneuve	2019-05-03	Edit Delete
Spider Man	2/5	Stan Lee	1996-03-18	Edit Delete
Titanic	3/5	Cameron	1998-11-01	Edit Delete

3. Edit a row

Here i am gonna show you by editing info for movie Dune. Here i have edited dune name as dune_edited and has been saved in database on click save changes

The screenshot shows a web application with a table titled "5 movies in the list". The table has columns for "Movie Name", "Rating", "Director's name", "Release Date", and "Actions". The data rows are: Avengers (4/5, Anthony russo, 2014-03-06), Batman (5/5, Christopher Nolan, 2008-11-07), Dune_edited (4/5, Denis Villene, 2019-05-03), Spider Man (2/5, Stan Lee, 1996-03-18), and Titanic (3/5, Cameron, 1998-11-01). An "Edit Movie" modal dialog is open, showing the current values for the selected row: "Dune_edited", "Denis Villene", "05/03/2019", and "4/5". The dialog has "Close" and "Save changes" buttons.

Movie Name	Rating	Director's name	Release Date	Actions
Avengers	4/5	Anthony russo	2014-03-06	Edit Delete
Batman	5/5	Christopher Nolan	2008-11-07	Edit Delete
Dune_edited	4/5	Denis Villene	2019-05-03	Edit Delete
Spider Man	2/5	Stan Lee	1996-03-18	Edit Delete
Titanic	3/5	Cameron	1998-11-01	Edit Delete

4. Delete a row

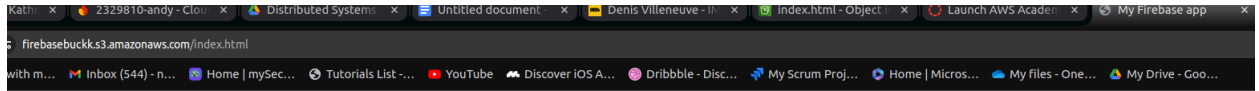
Here, i am gonna show you by deleting Batman movie row. Movie has successfully been deleted and database has updated accordingly

The screenshot shows the same web application after deleting the Batman row. The table now displays "4 movies in the list". The data rows are: Avengers (4/5, Anthony russo, 2014-03-06), Dune_edited (4/5, Denis Villene, 2019-05-03), Spider Man (2/5, Stan Lee, 1996-03-18), and Titanic (3/5, Cameron, 1998-11-01). The "Batman" row has been removed.

Movie Name	Rating	Director's name	Release Date	Actions
Avengers	4/5	Anthony russo	2014-03-06	Edit Delete
Dune_edited	4/5	Denis Villene	2019-05-03	Edit Delete
Spider Man	2/5	Stan Lee	1996-03-18	Edit Delete
Titanic	3/5	Cameron	1998-11-01	Edit Delete

5. Sorting by movie name:

Here, i am gonna sort movie according to their name, You can perform this action simply by clicking on MovieName header of table

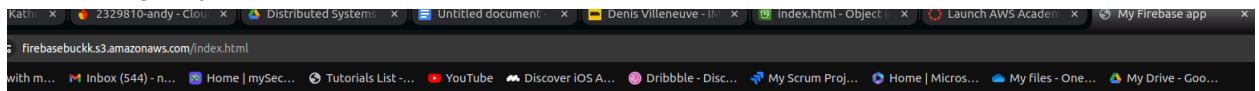


4 movies in the list

Movie Name	Rating	Director	Release Date	Actions
Avengers	4/5	Anthony russo	2014-03-06	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Dune_edited	4/5	Denis Villene	2019-05-03	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Spider Man	2/5	Stan Lee	1996-03-18	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Titanic	3/5	Cameron	1998-11-01	<input type="button" value="Edit"/> <input type="button" value="Delete"/>

6. Sorting moving by director name

Here, i am gonna show you table after sorting upon director name. This goes on descending way

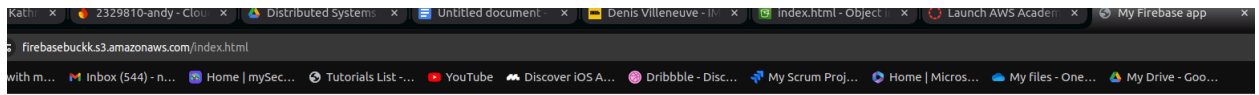


4 movies in the list

Movie Name	Rating	Director	Release Date	Actions
Spider Man	2/5	Stan Lee	1996-03-18	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Dune_edited	4/5	Denis Villene	2019-05-03	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Titanic	3/5	Cameron	1998-11-01	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Avengers	4/5	Anthony russo	2014-03-06	<input type="button" value="Edit"/> <input type="button" value="Delete"/>

7. Sorting on basis of rating

Here, i am gonna show you table after sorting on review header



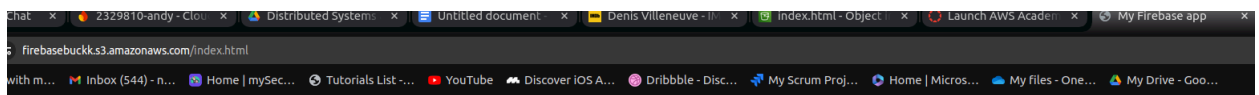
4 movies in the list

Movie name Director's name mm/dd/yyyy 0/5

Movie Name	Rating	Director	Release Date	Actions
Spider Man	2/5	Stan Lee	1996-03-18	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Titanic	3/5	Cameron	1998-11-01	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Dune_edited	4/5	Denis Villene	2019-05-03	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Avengers	4/5	Anthony russo	2014-03-06	<input type="button" value="Edit"/> <input type="button" value="Delete"/>

8. Sorting on basis of release date

here , i am gonna show you table after sorting on release date.



4 movies in the list

Movie name Director's name mm/dd/yyyy 0/5

Movie Name	Rating	Director	Release Date	Actions
Dune_edited	4/5	Denis Villene	2019-05-03	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Avengers	4/5	Anthony russo	2014-03-06	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Titanic	3/5	Cameron	1998-11-01	<input type="button" value="Edit"/> <input type="button" value="Delete"/>
Spider Man	2/5	Stan Lee	1996-03-18	<input type="button" value="Edit"/> <input type="button" value="Delete"/>