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.

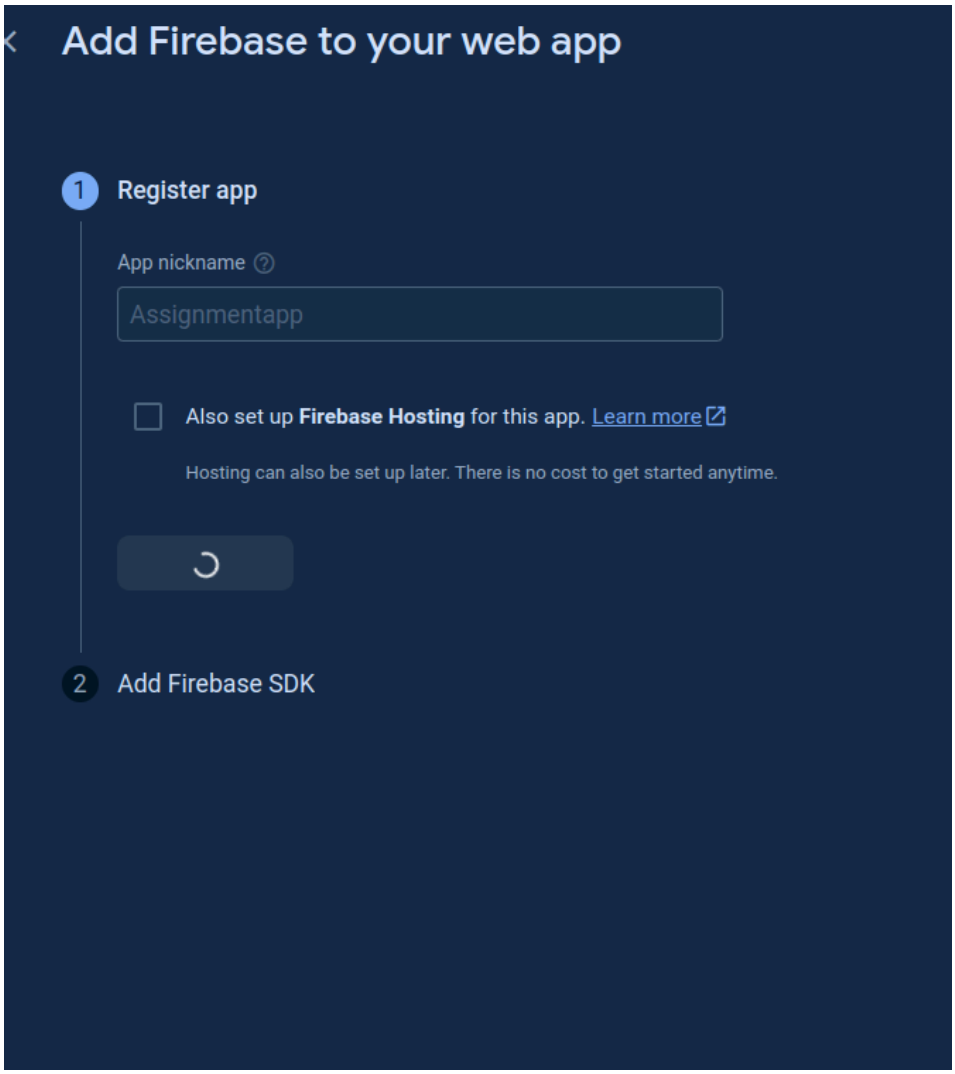
=> Ans:

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.



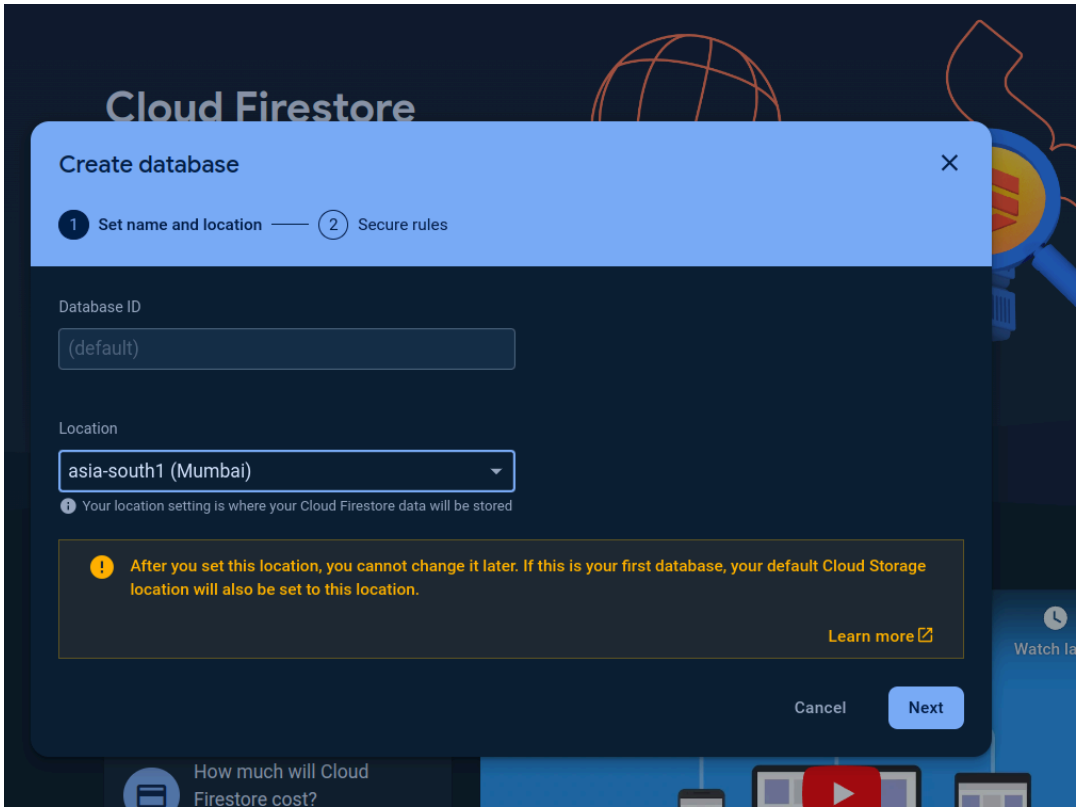
And later on i have copied and paste my firebase configuration into my text editor.

*Untitled Document 1 *Untitled Document 2

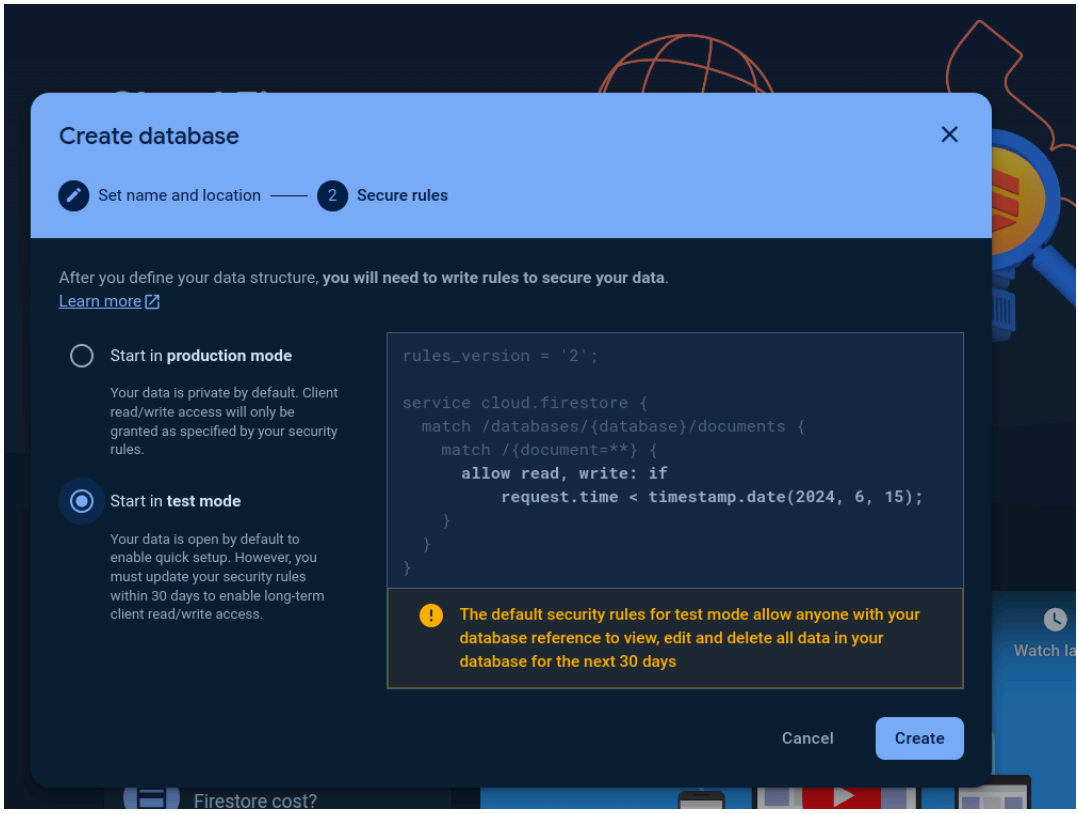
```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
    import { getFirestore, collection, orderBy, onSnapshot, query, addDoc, doc, setDoc } fr
"https://www.gstatic.com/firebasejs/9.18.0/firebase-firestore.js";

// 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: "AIzaSyAru8gepEmG917OM-UKFdPuP-j0yZG36wU",
  authDomain: "andy-f8619.firebaseapp.com",
  projectId: "andy-f8619",
  storageBucket: "andy-f8619.appspot.com",
  messagingSenderId: "836758811412",
  appId: "1:836758811412:web:2c02526e1ee5a510dcc801"
};

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

Configuring Databse

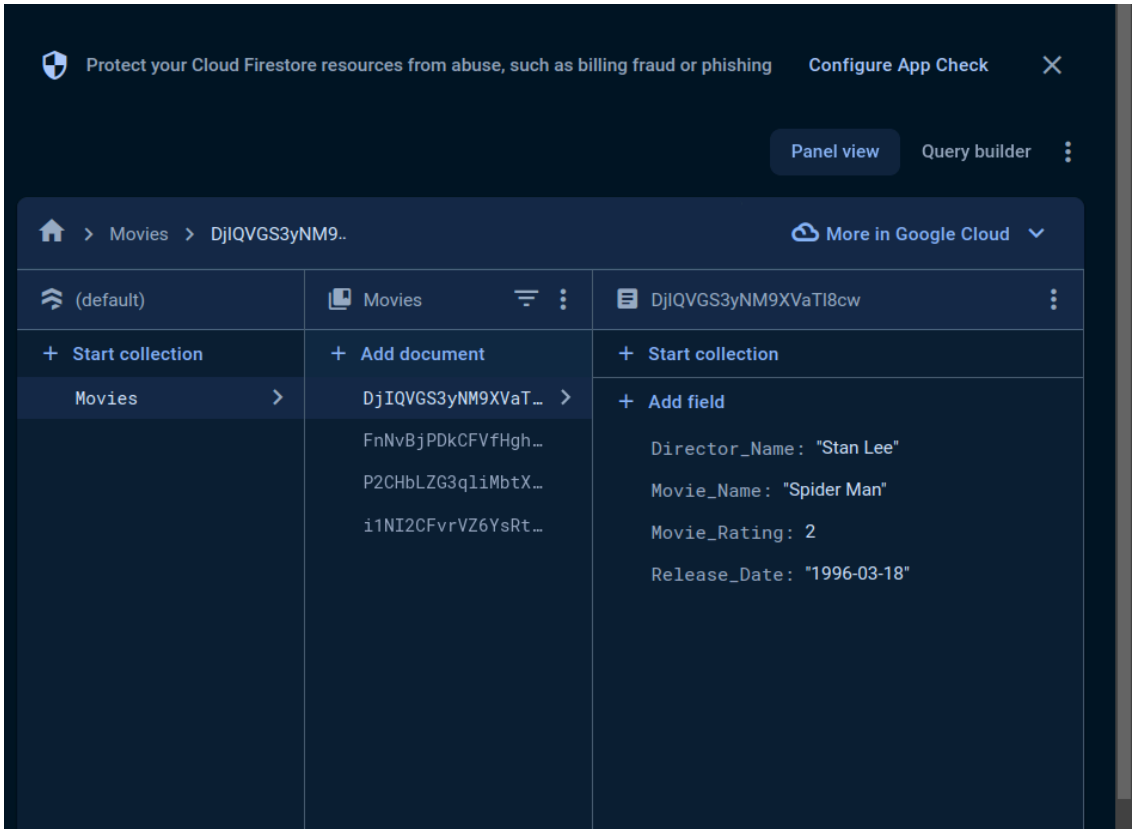
Then i navigate to build option and select firestore database. Then while creating database i choose the location into mumbai server as i am from Nepal and Mumbai is the closest one for me which will provide me with best peformance.



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>
</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-a
lpha1/dist/js/bootstrap.bundle.min.js"></script>

<!-- Firebase -->
<script
src="https://www.gstatic.com/firebasejs/9.0.0/fireb
ase-app-compat.js"></script>
<script
src="https://www.gstatic.com/firebasejs/9.0.0/fireb
ase-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.firebaseapp.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);
        });
    });
```

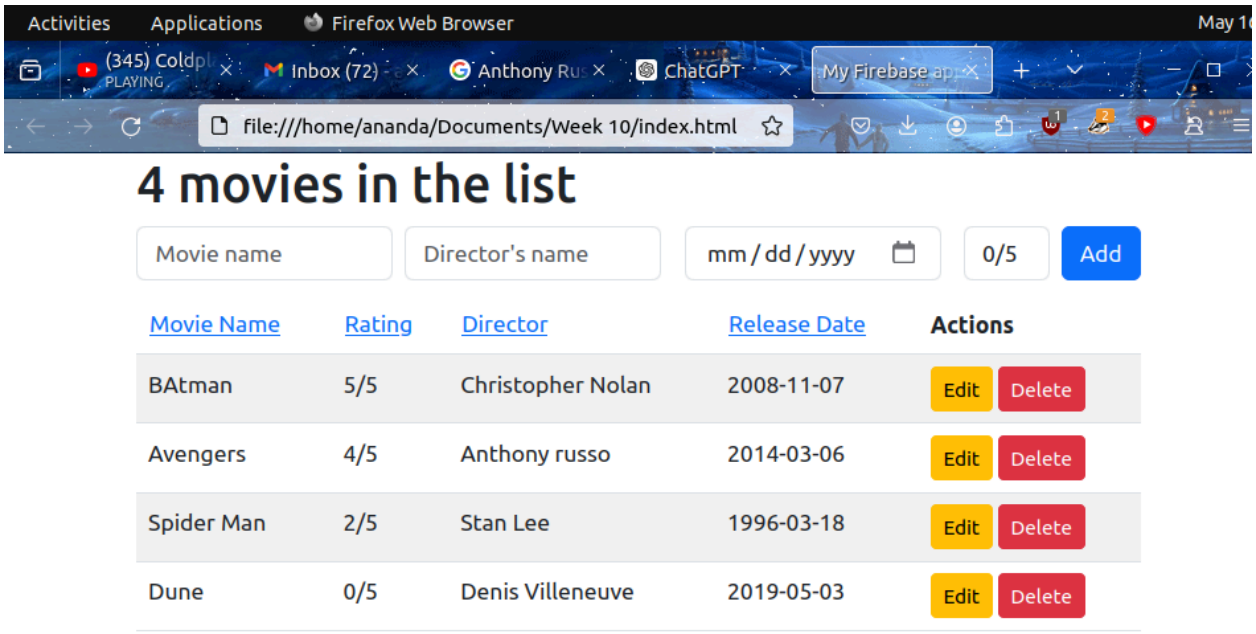
And this last one is for sorting functionalities. On clicking the table header of any movie name or director name or release date or movie rationing your table will be sorted according to click.

```
    // 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();
```

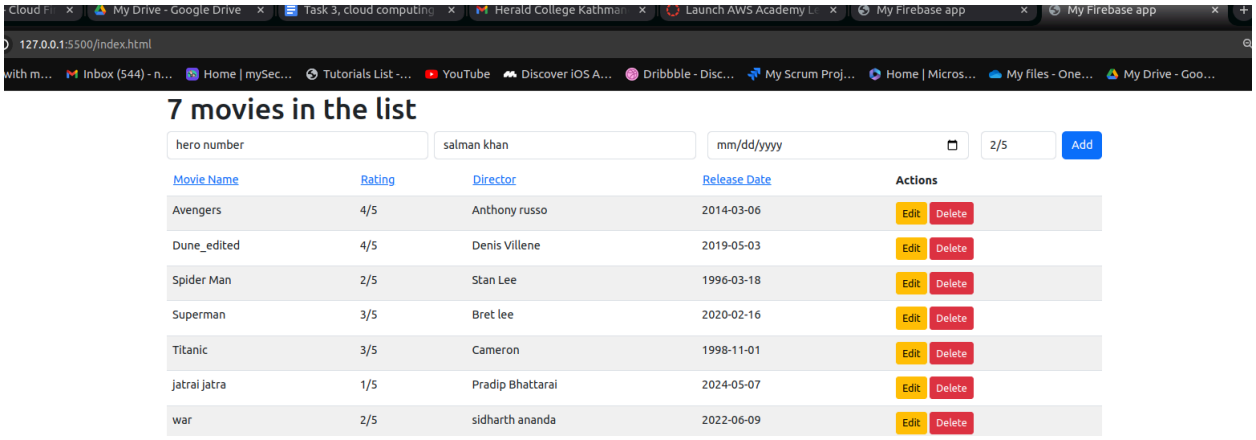
```
    });

    $("#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>
```

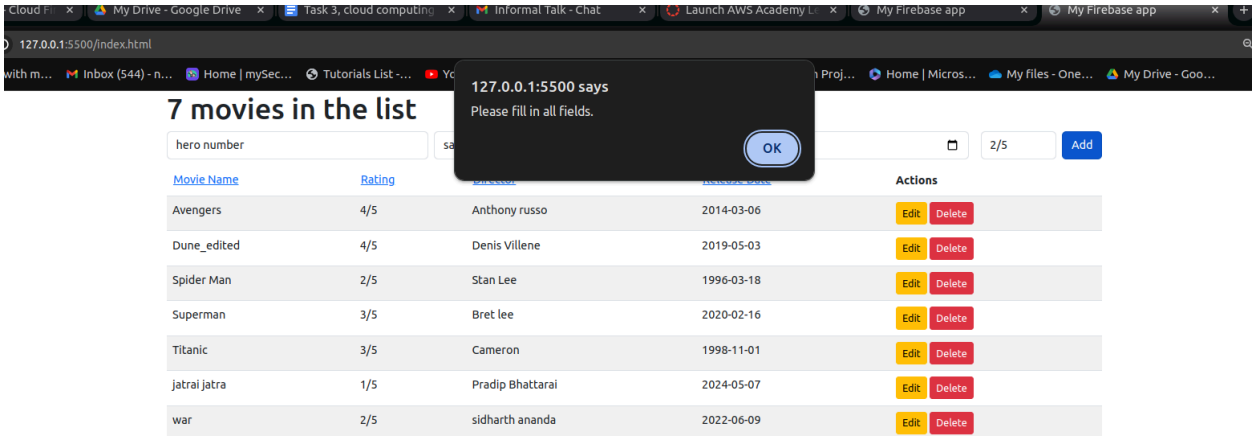
Accessing the index.html file

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



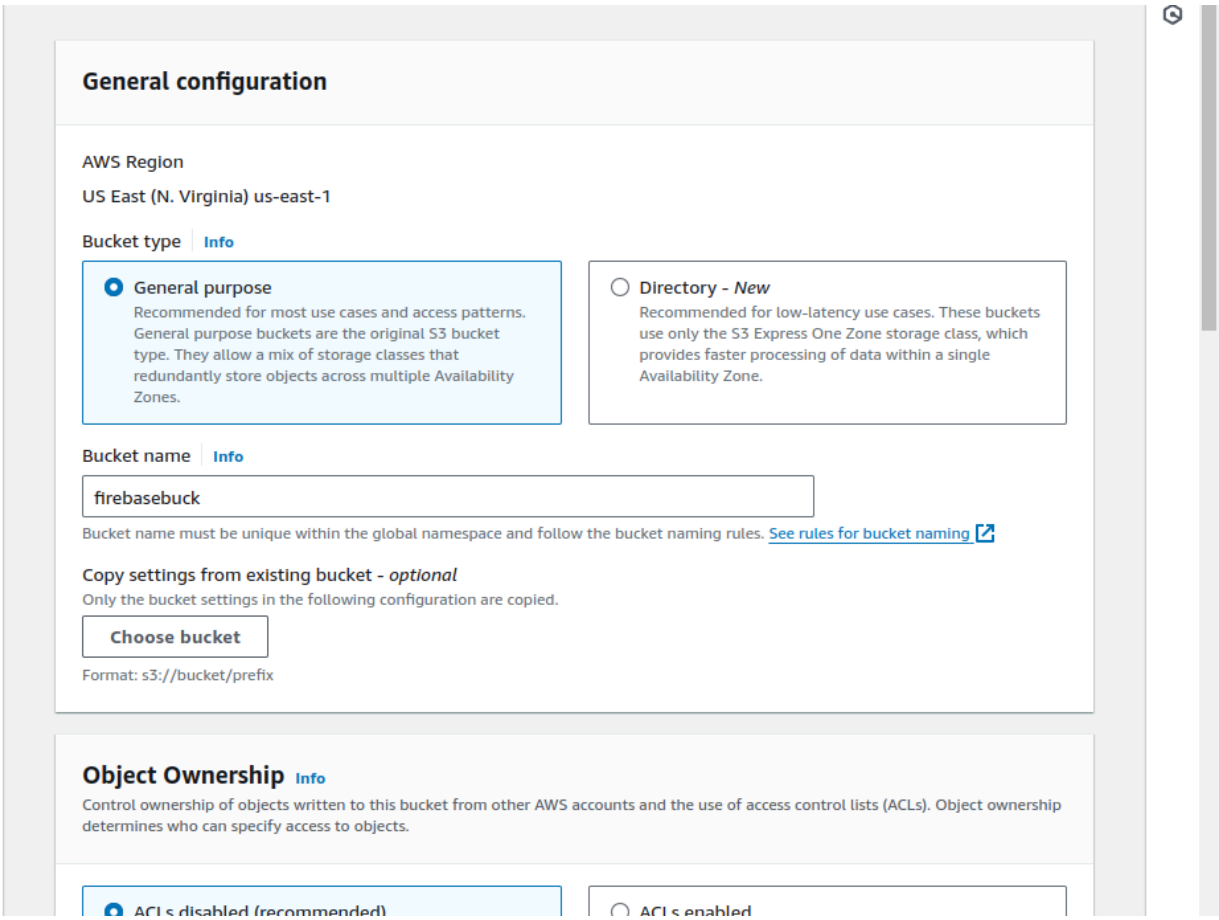
And yes every fields are compulsory to fill before adding other wise it will throw alert to fill all the fields



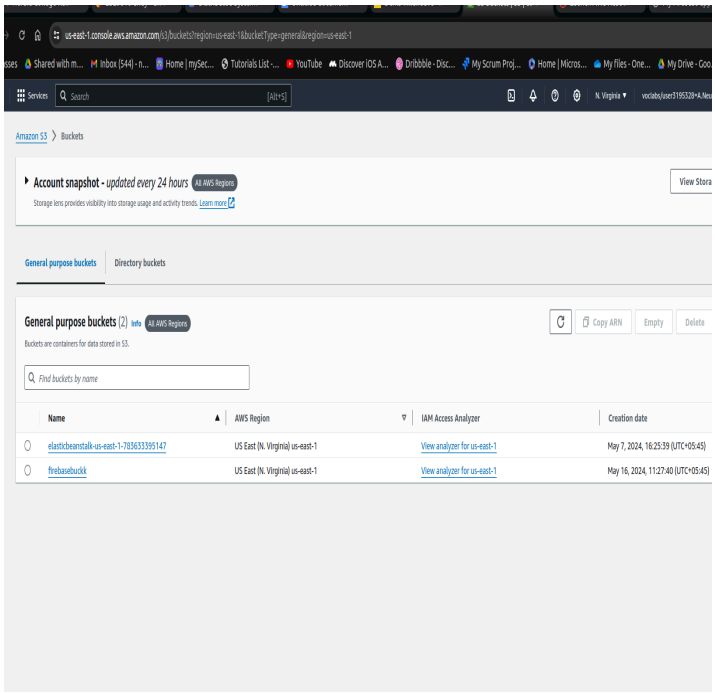


Now lets host this site into aws s3 bucket

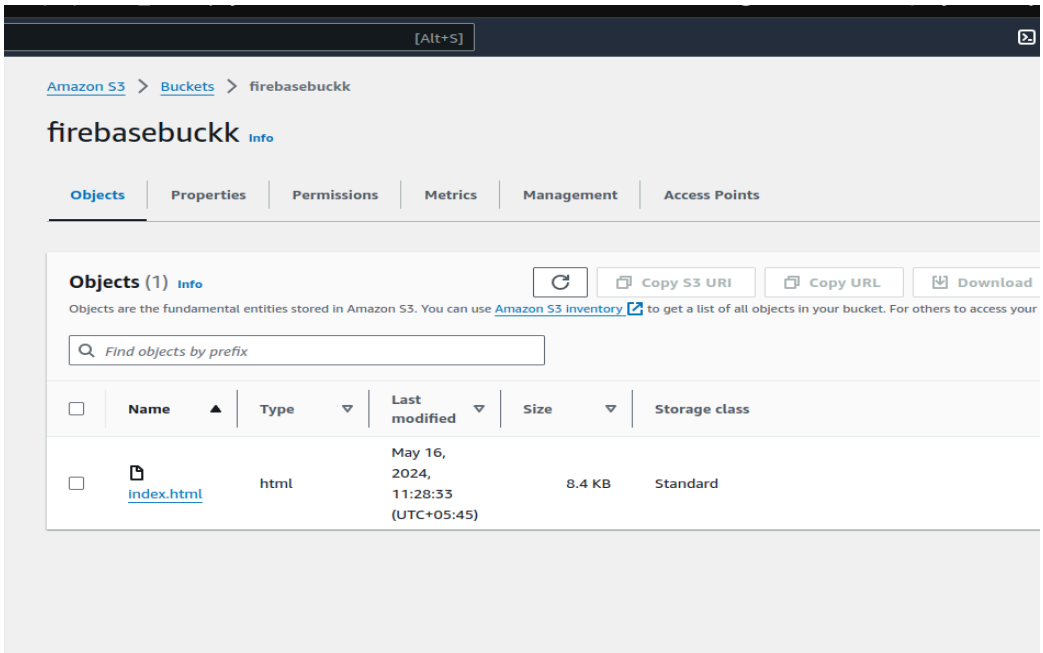
First redirect to aws console and search for s3 in service menu. After clicking on s3 choose create bucket option. Give a name to your bucket, here i am giving **firebasebuck**. I am keeping all the other fields as default and press the button create.



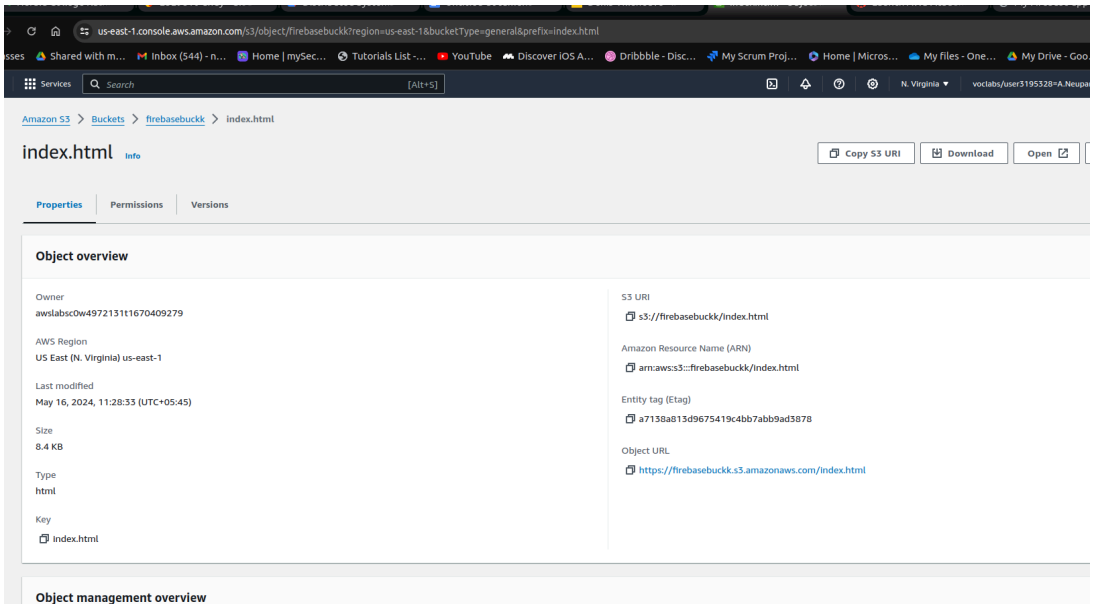
After your bucket is successfully created, then select the bucket name and choose the option upload and deploy.



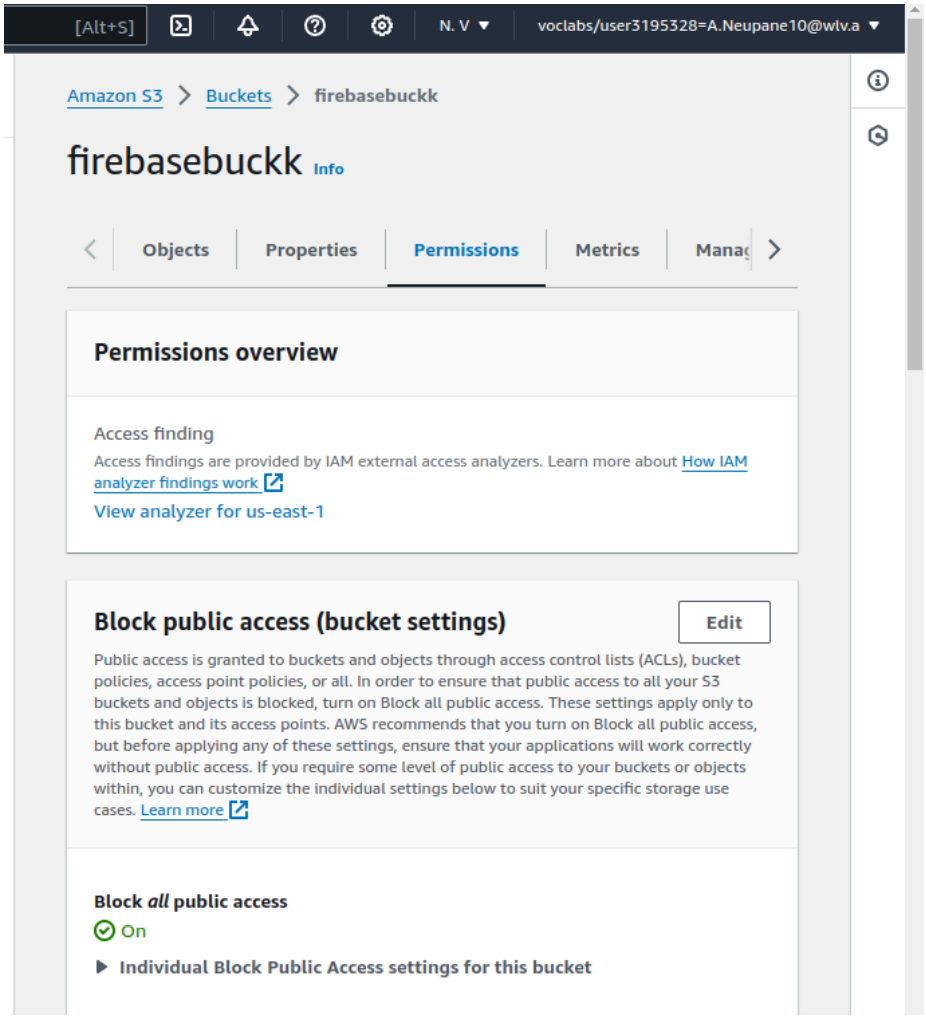
After clicking on bucket name, i have uploaded my index.html file, which you can see on next page.



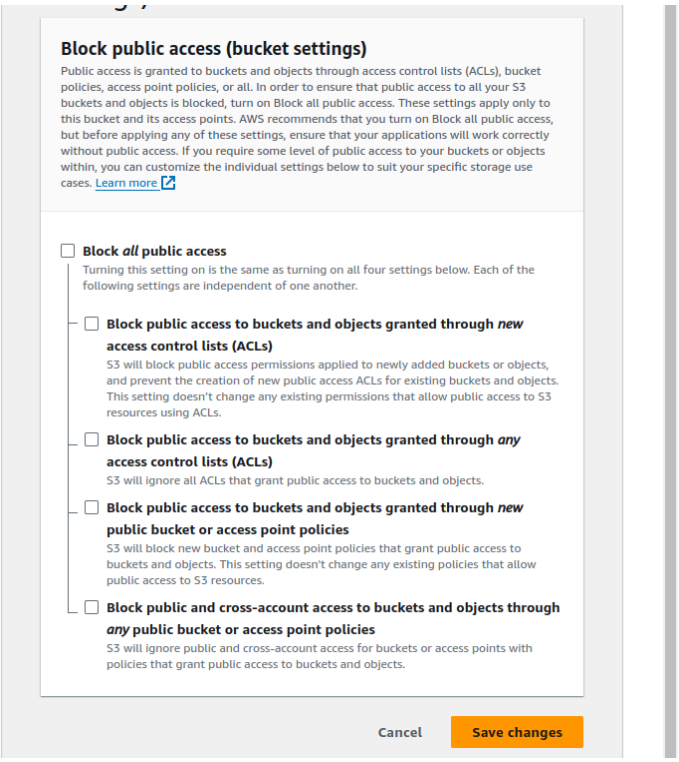
Then go back to your bucket section and click on bucket name. In details view you can see a label name object url.

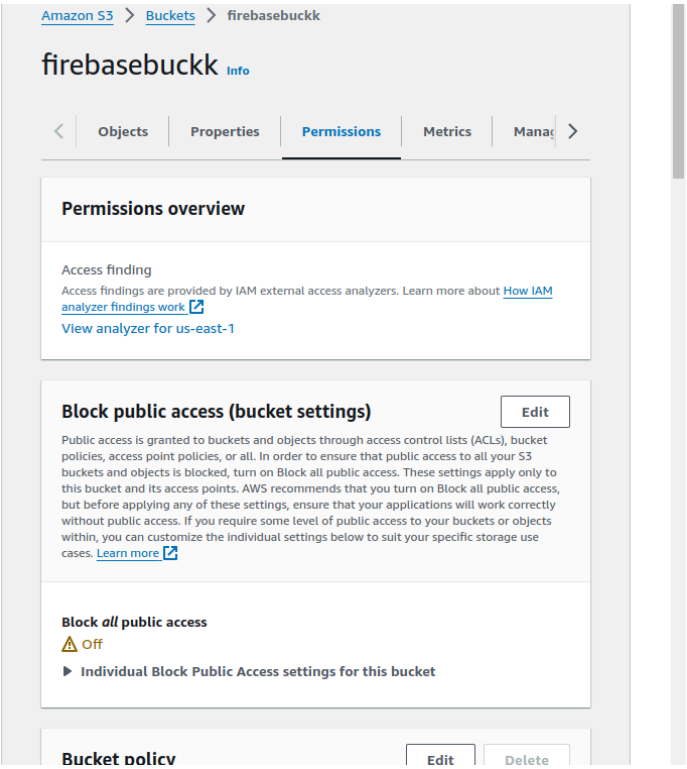


Copy the url and redirect to the page this will throw you an error as we have block the public access. Now redirect to the permission section

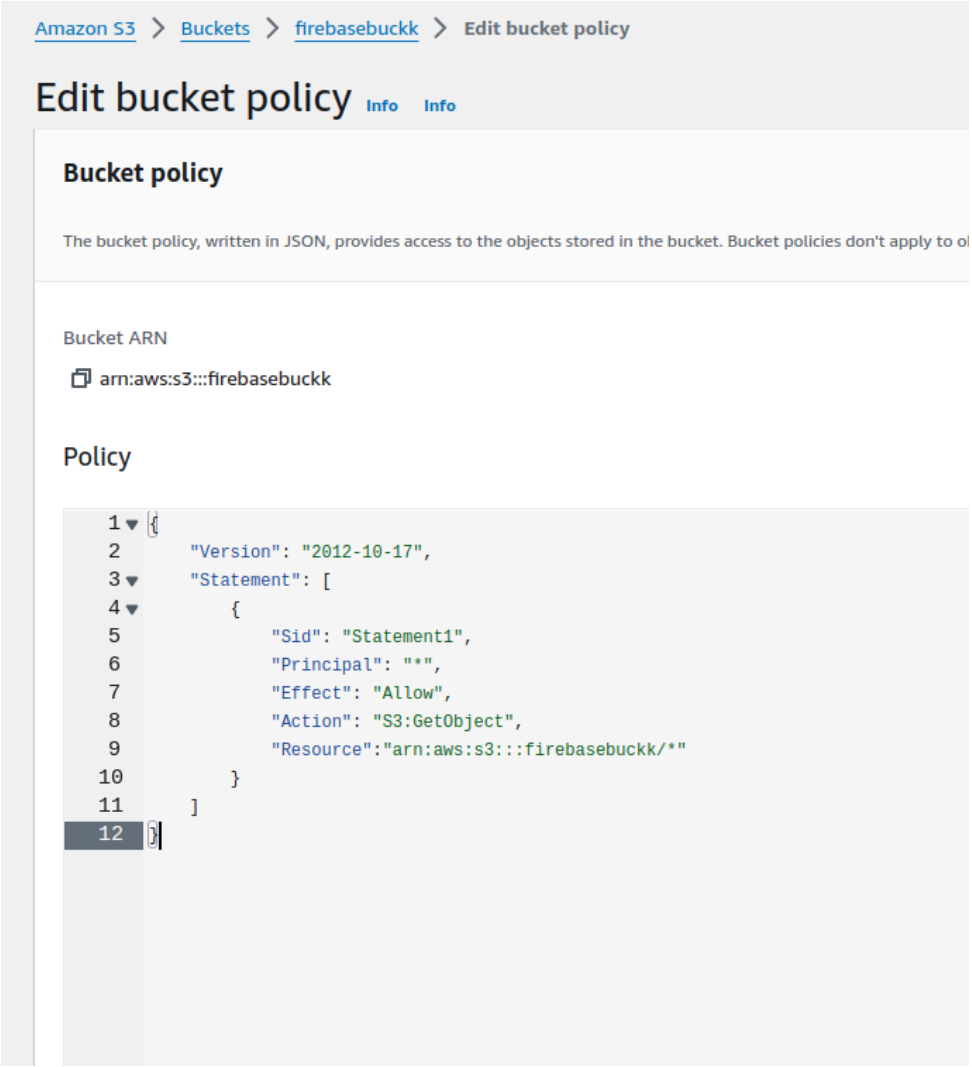


Click on the check box and remove the tick mark and then click on save changes which we allow us to redirect to the url.



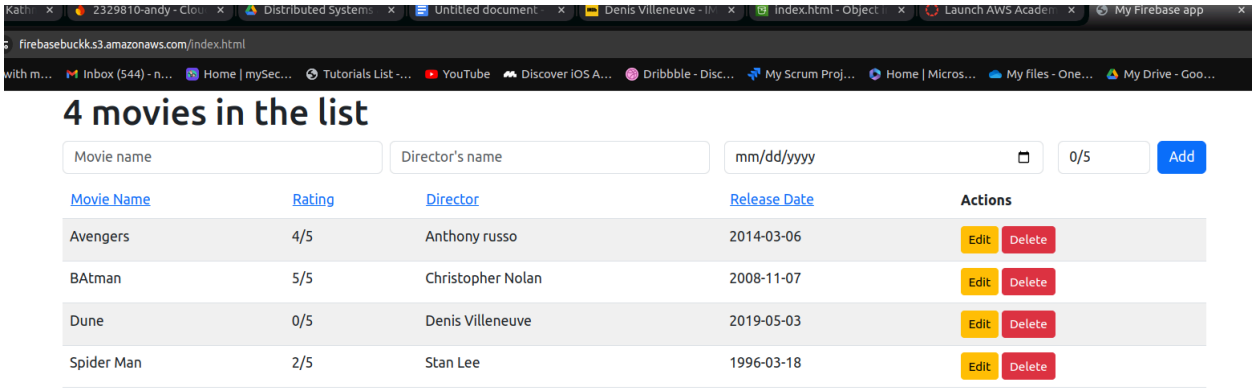


Then just below there is bucket policy section edit your bucket policy as shown in the picture and press save changes this will update your url to be accessible from everywhere.

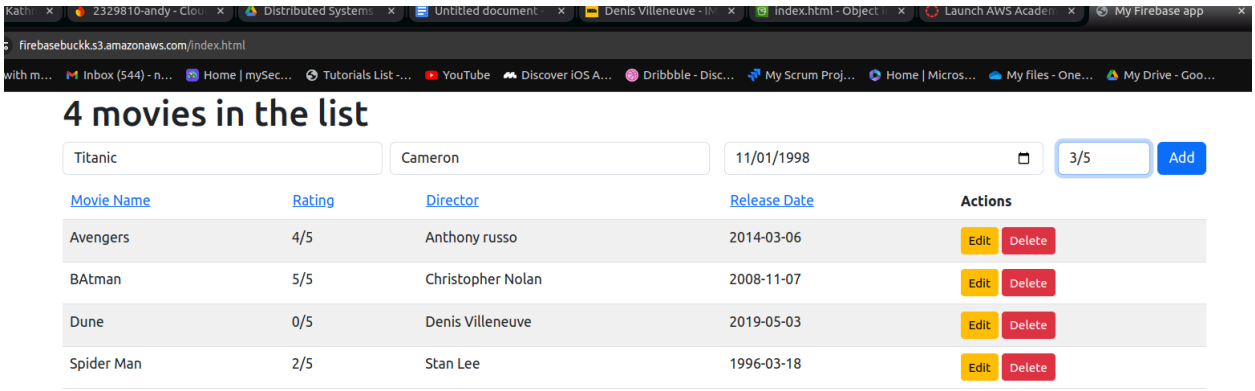


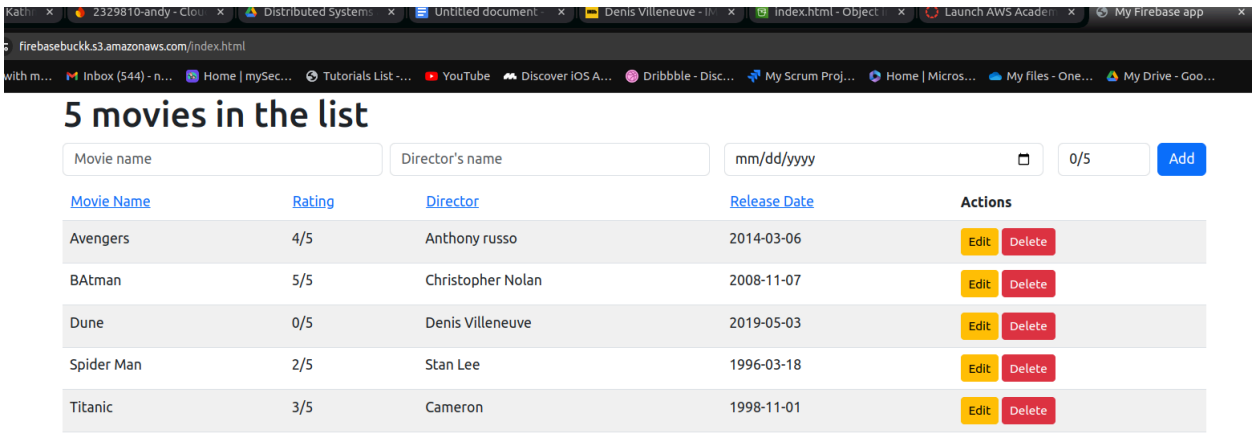
Lastly, press on the url link and you will see your site has been hosted on s3 bucket. Here i am gonna show you all the actions you can perform within website

1. First face of url



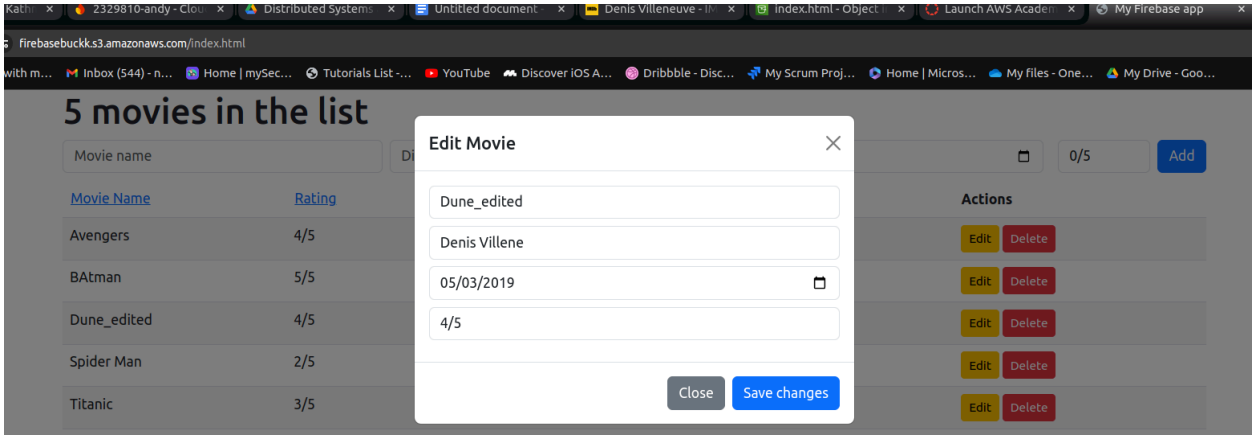
2. Adding a information
   Here, i am adding info for movie titanic, which is successfully performed by the site

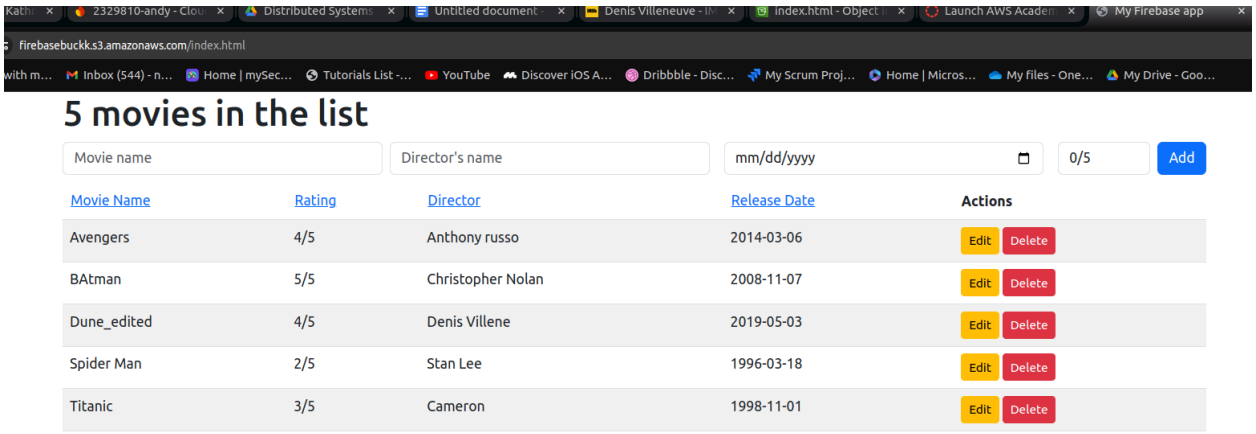




3. Edit a row

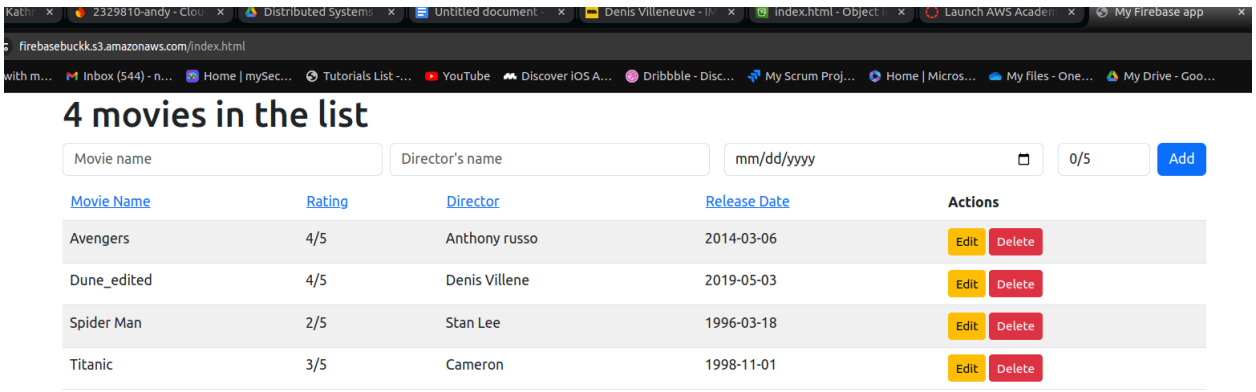
   Here i am gonna show you by editing info for movie Dune. Here i have edited dune name as dune_edited and has been saved in database on click save changes



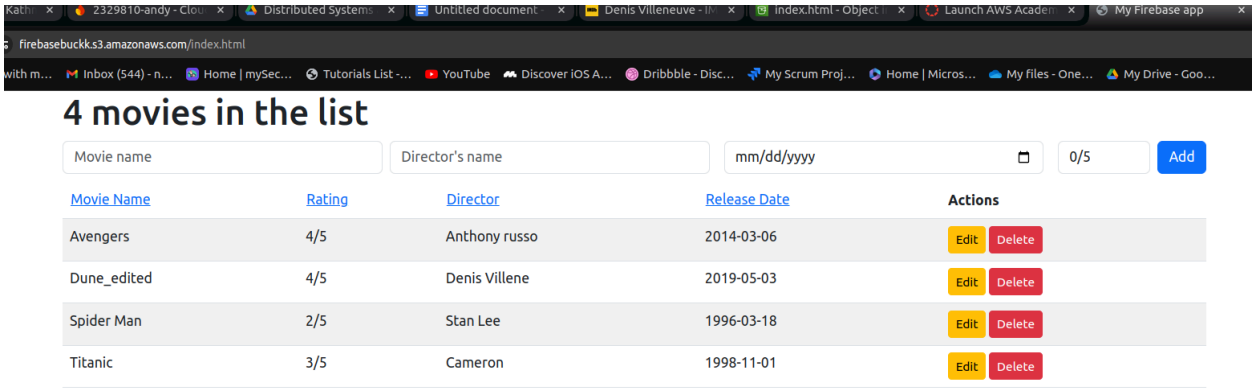


4. Delete a row

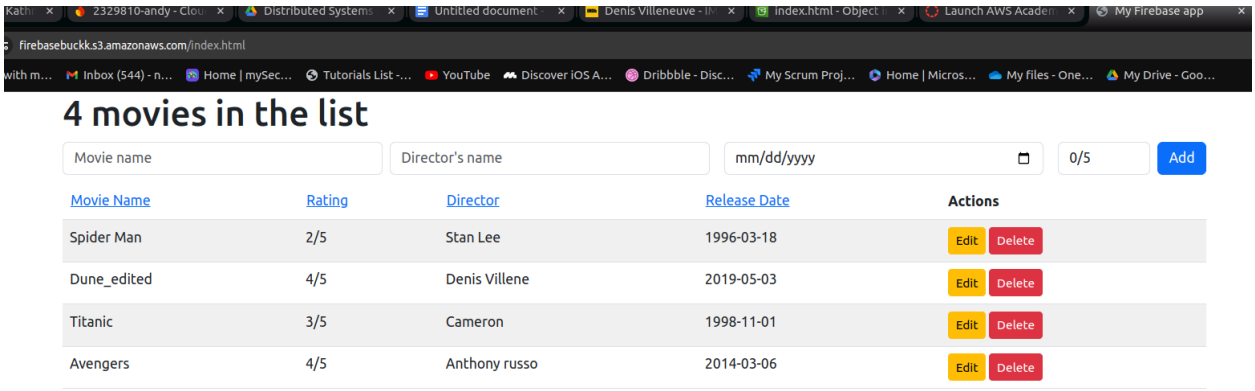
   Here, i am gonna show you by deleting Batman movie row. Movie has successfully been deleted and database has updated accordingly



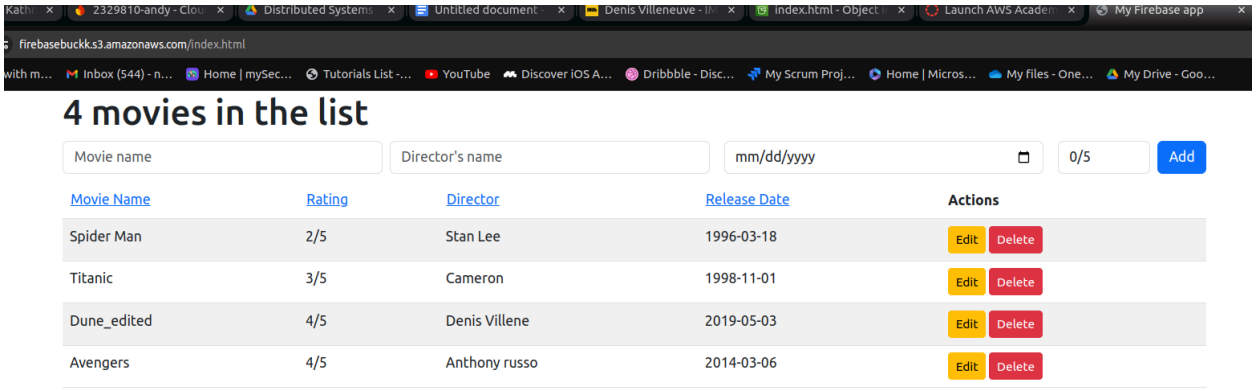
5. Sorting by movie name:
   Here, i am gonna sort movie according to their name, You can perform this action simply by clicking on MovieName header of table



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



7. Sorting on basis of rating
   Here, i am gonna show you table after sorting on review header



8. Sorting on basis of release date
   here , i am gonna show you table after sorting on release date.