
OOADJ LAB 9 MVC

K GANESH VAIDYANATHAN

PES1UG21CS253

SEC 'E'

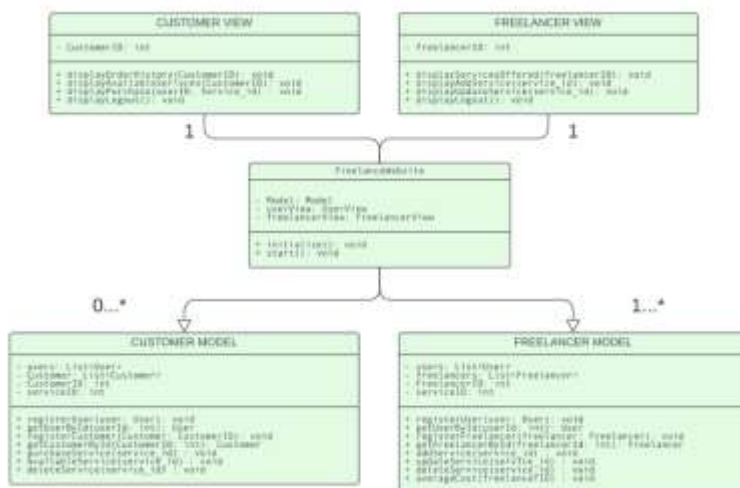
MVC IN OUR PROJECT:

Our project is an online freelance marketplace website. It contains 2 controllers:

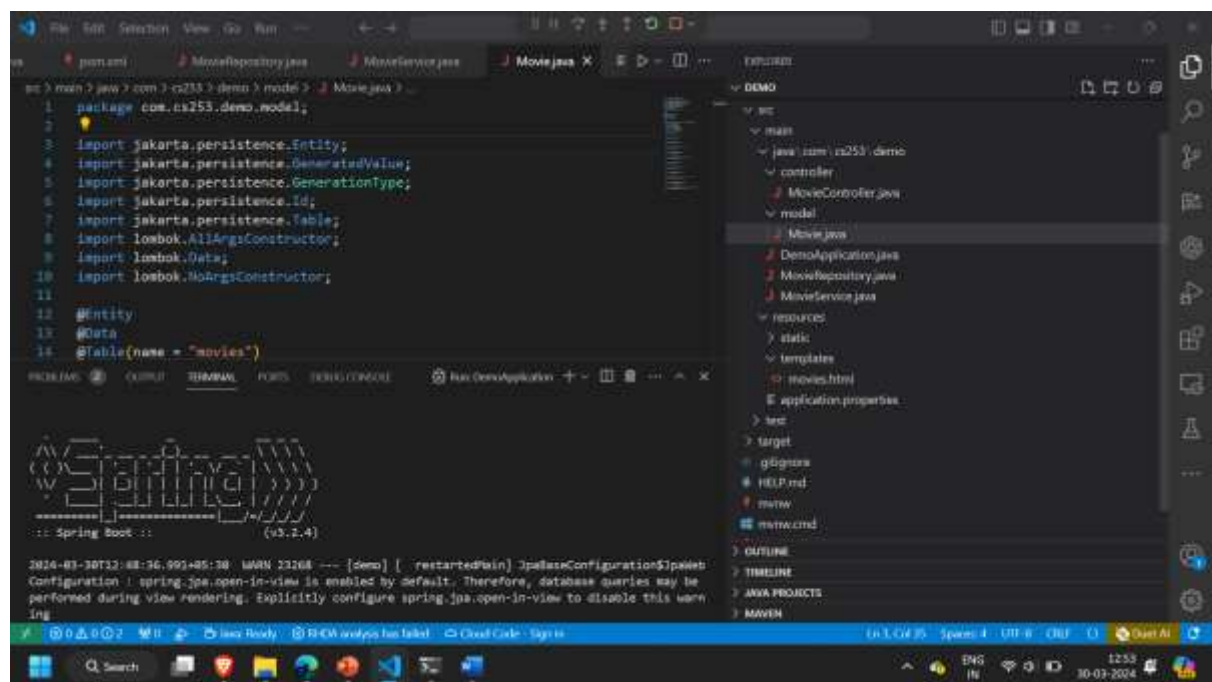
1. Freelancer controller
2. Customer controller

These 2 controllers hold the logic and routing for implementing the respective roles.

We also make use of 2 different views, one for each customer and freelancer to display their respective attributes and methods.



FULL FOLDER VIEW:



MOVIECONTROLLER.JAVA (CONTROLLER)

```
package com.cs253.demo.controller;

import org.springframework.ui.Model;
import org.springframework.web.servlet.ModelAndView;
import org.springframework.web.servlet.mvc.support.RedirectAttributes;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

import com.cs253.demo.model.*;
import com.cs253.demo.MovieService;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.servlet.view.RedirectView;

@RestController
public class MovieController {

    @Autowired
    private MovieService movieService;

    @PostMapping("/addMovie")
    public RedirectView postDetails(@RequestParam("title") String
title,
        @RequestParam("releaseYear") int releaseYear,
        @RequestParam("genre") String genre,
        @RequestParam("director") String director,
        @RequestParam("averageRating") double averageRating,
        RedirectAttributes redirectAttributes) {
    try {
        Movie movie = new Movie();
        movie.setTitle(title);
        movie.setReleaseYear(releaseYear);
        movie.setGenre(genre);
        movie.setDirector(director);
        movie.setAverageRating(averageRating);

        movieService.saveDetails(movie);
        redirectAttributes.addFlashAttribute("successMessage",
"Movie added successfully.");
    } catch (Exception e) {
```

```

        redirectAttributes.addFlashAttribute("errorMessage",
"Failed to add movie. Please try again.");
    }
    return new RedirectView("/getMovies", true);
}

@GetMapping("/getMovies")
public ModelAndView getDetails(Model model) {
    Iterable<Movie> movies = movieService.getAllDetails();
    model.addAttribute("movies", movies);
    return new ModelAndView("movies");
}

@PostMapping("/updateRating")
public RedirectView updateRating(@RequestParam("movieId") Integer
movieId,
    @RequestParam("newRating") double newRating,
    RedirectAttributes redirectAttributes) {
    try {
        if (newRating < 1 || newRating > 5) {
            throw new IllegalArgumentException("Rating must be
between 1 and 5.");
        }

        Movie movie = movieService.getById(movieId);
        double oldRating = movie.getAverageRating();
        double updatedRating = (oldRating + newRating) / 2.0;
        movie.setAverageRating(updatedRating);
        movieService.saveDetails(movie);
        redirectAttributes.addFlashAttribute("successMessage",
"Rating updated successfully.");
    } catch (IllegalArgumentException e) {
        redirectAttributes.addFlashAttribute("errorMessage",
"Rating must be between 1 and 5.");
    } catch (Exception e) {
        redirectAttributes.addFlashAttribute("errorMessage",
"Failed to update rating. Please try again.");
    }
    return new RedirectView("/getMovies", true);
}
}

```

MOVIE.JAVA (MODEL)

```
package com.cs253.demo.model;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.Table;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Entity
@Data
@Table(name = "movies")
@NoArgsConstructor
@AllArgsConstructor
public class Movie {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    private Long id;
    private String title;
    private int releaseYear;
    private String genre;
    private String director;
    private double averageRating;

    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public int getReleaseYear() {
```

```
        return releaseYear;
    }

    public void setReleaseYear(int releaseYear) {
        this.releaseYear = releaseYear;
    }

    public String getGenre() {
        return genre;
    }

    public void setGenre(String genre) {
        this.genre = genre;
    }

    public String getDirector() {
        return director;
    }

    public void setDirector(String director) {
        this.director = director;
    }

    public double getAverageRating() {
        return averageRating;
    }

    public void setAverageRating(double averageRating) {
        this.averageRating = averageRating;
    }
}
```

MOVIESERVICE.JAVA

```
package com.cs253.demo;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.cs253.demo.model.*;
import com.cs253.demo.MovieService;

@Service
public class MovieService {

    @Autowired
    private MovieRepository movieRepository;

    @SuppressWarnings("null")
    public Movie saveDetails(Movie movie) {
        return movieRepository.save(movie);
    }

    public Iterable<Movie> getAllDetails() {
        return movieRepository.findAll();
    }

    @SuppressWarnings("null")
    public Movie getById(Integer movieId) {
        return movieRepository.findById(movieId).orElse(null);
    }
}
```

MOVIES.HTML (VIEW)

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Movies Review Website</title>
  <link
href="https://fonts.googleapis.com/css2?family=Roboto:wght@400;700&disp
lay=swap" rel="stylesheet">
  <style>
    /* Reset CSS */
    * {
      margin: 0;
      padding: 0;
      box-sizing: border-box;
    }
    body {
      font-family: 'Roboto', sans-serif;
      background-color: #f4f4f4;
      color: #333;
      line-height: 1.6;
      margin: 0;
      padding: 0;
    }
    /* Container */
    .container {
      max-width: 1200px;
      margin: 0 auto;
      padding: 20px;
    }
    /* Header */
    header {
      background-color: #007bff;
      color: #fff;
      padding: 20px 0;
      text-align: center;
    }
    header h1 {
      font-size: 36px;
    }
    /* Main Content */
    main {
```



```
        padding: 20px;
        background-color: #fff;
        border-radius: 8px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        margin-top: 20px;
    }
    h2 {
        margin-bottom: 10px;
        font-size: 24px;
        color: #007bff;
    }
    /* Forms */
    form {
        margin-bottom: 20px;
    }
    label {
        display: block;
        margin-bottom: 5px;
    }
    input[type="text"],
    input[type="number"] {
        width: 100%;
        padding: 10px;
        margin-bottom: 10px;
        border-radius: 5px;
        border: 1px solid #ccc;
    }
    button {
        padding: 10px 20px;
        background-color: #007bff;
        color: #fff;
        border: none;
        border-radius: 5px;
        cursor: pointer;
    }
    button:hover {
        background-color: #0056b3;
    }
    /* Table */
    table {
        width: 100%;
        border-collapse: collapse;
        margin-bottom: 20px;
    }
    th, td {
        padding: 10px;
```

```

border-bottom: 1px solid #ddd;
text-align: center; /* Center align table values */
}
th {
background-color: #007bff;
color: #fff;
}
/* Messages */
.success-message {
color: green;
margin-bottom: 10px;
}
.error-message {
color: red;
margin-bottom: 10px;
}
</style>
</head>
<body>
<header>
<h1>Movies Review Website</h1>
</header>
<div class="container">
<main>
<section>
<h2>Add Ratings</h2>
<form action="/updateRating" method="post">
<label for="movieId">Movie ID:</label>
<input type="number" id="movieId" name="movieId"
required>

<label for="newRating">New Rating:</label>
<input type="number" id="newRating"
name="newRating" step="0.1" min="1" max="5" required>
<button type="submit">Update Rating</button>
</form>
</section>

<hr>

<section>
<h2>Movies List</h2>
<table>
<thead>
<tr>
<th>ID</th>
<th>Title</th>

```

```

        <th>Release Year</th>
        <th>Genre</th>
        <th>Director</th>
        <th>Current Rating</th>
    </tr>
</thead>
<tbody>
    <tr th:each="movie : ${movies}">
        <td th:text="${movie.id}"></td>
        <td th:text="${movie.title}"></td>
        <td th:text="${movie.releaseYear}"></td>
        <td th:text="${movie.genre}"></td>
        <td th:text="${movie.director}"></td>
        <td th:text="${movie.averageRating}"></td>
    </tr>
</tbody>
</table>
</section>

<hr>

<section>
    <h2>Add Movies</h2>
    <form action="/addMovie" method="post">
        <label for="title">Title:</label>
        <input type="text" id="title" name="title"
required>

        <label for="releaseYear">Release Year:</label>
        <input type="number" id="releaseYear"
name="releaseYear" required>

        <label for="genre">Genre:</label>
        <input type="text" id="genre" name="genre"
required>

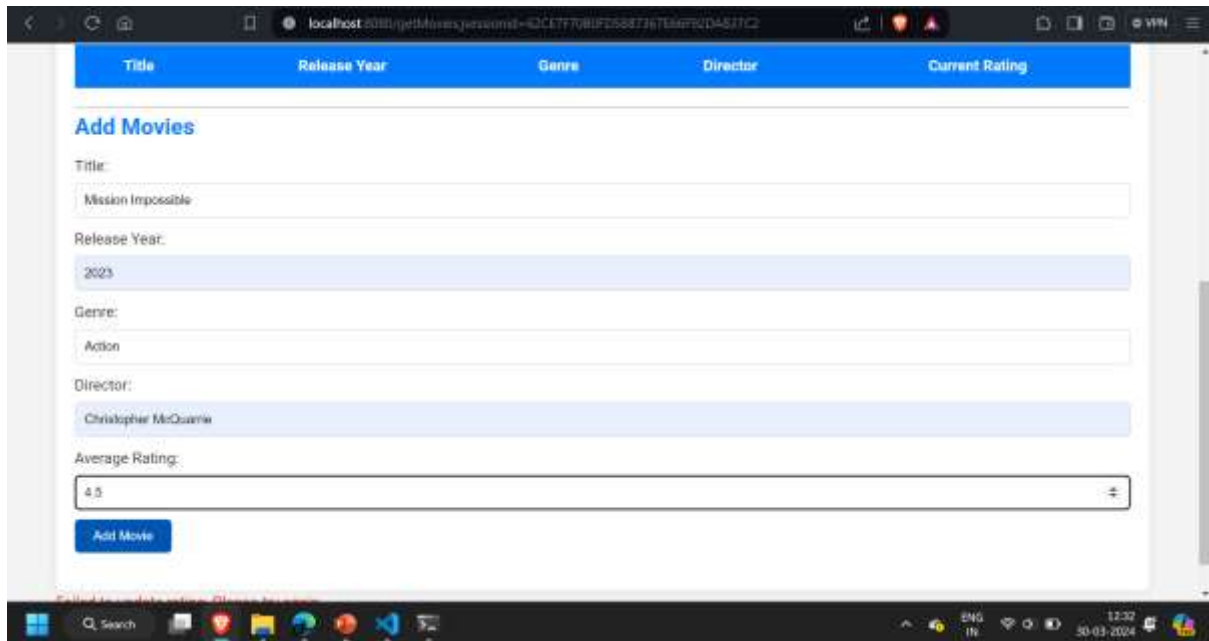
        <label for="director">Director:</label>
        <input type="text" id="director" name="director"
required>

        <label for="averageRating">Average Rating:</label>
        <input type="number" id="averageRating"
name="averageRating" step="0.1" min="1" max="5" required>
        <button type="submit">Add Movie</button>
    </form>
</section>
</main>
<div class="success-message" th:if="${successMessage}"
th:text="${successMessage}"></div>

```

```
        <div class="error-message" th:if="${errorMessage}"  
th:text="${errorMessage}"></div>  
    </div>  
</body>  
</html>
```

ADDING A MOVIE:

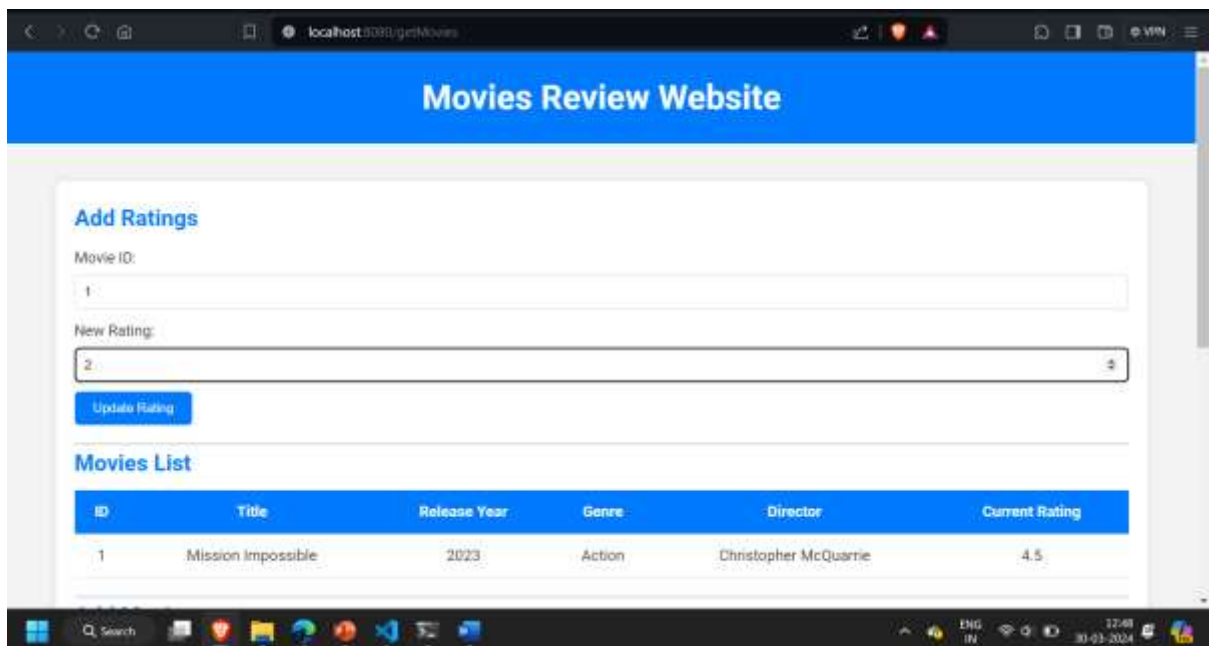


The screenshot shows a web browser window with the URL `localhost:8080/getMovies.js?movieId=42CCT770BIF2583776T6MFP2D4837C2`. The page has a blue header with the following columns: Title, Release Year, Genre, Director, and Current Rating. Below the header is a form titled "Add Movies". The form contains the following fields:

- Title:
- Release Year:
- Genre:
- Director:
- Average Rating:

At the bottom of the form is a blue button labeled "Add Movie".

ADDING A RATING:



The screenshot shows a web browser window with the URL `localhost:8080/getMovies`. The page has a blue header with the text "Movies Review Website". Below the header is a form titled "Add Ratings". The form contains the following fields:

- Movie ID:
- New Rating:

At the bottom of the form is a blue button labeled "Update Rating".

Below the form is a table titled "Movies List". The table has the following columns: ID, Title, Release Year, Genre, Director, and Current Rating. The table contains one row of data:

ID	Title	Release Year	Genre	Director	Current Rating
1	Mission Impossible	2023	Action	Christopher McQuarrie	4.5

UPDATED AVERAGE RATING

Movies List

ID	Title	Release Year	Genre	Director	Current Rating
1	Mission Impossible	2023	Action	Christopher McQuarrie	3.25

ERROR CONDITIONS:

Add Ratings

Movie ID:

New Rating:

Update Rating

 Value must be less than or equal to 5.

ERROR CONDITIONS:

Add Ratings

Movie ID:

New Rating:

Update Rating

Movies List

ID	Title	Release Year
1	Mission Impossible	2023

Add Movies

Title:

Failed to update rating. Please try again.

DATABASE VIEW:

```
mysql> use movies
Database changed
mysql> SELECT * FROM MOVIES;
```

id	average_rating	director	genre	release_year	title
1	3.25	Christopher McQuarrie	Action	2023	Mission Impossible
2	5	Joseph Kosinski	ACTION THRILLER	2022	TOP GUN MAVERICK
3	4.9	RISHABH SHETTY	THRILLER	2022	KANTARA

```
3 rows in set (0.00 sec)

mysql> |
```

