```
<!DOCTYPE html>
<html lang="en">
    <meta charset="UTF-8">
    <meta name="viewport" content=</pre>
"width=device-width, initial-scale=1.0">
    <title>Movie Info and Reviews</title>
    <link rel="stylesheet" href="styles.css">
<body>
    <div class="container">
        <h1>Movie Information and Reviews</h1>
        <input type="text" id="search"</pre>
placeholder="Enter movie name" />
        <button onclick="fetchMovieData()">
Search</button>
        <div id="results"></div>
    </div>
    <script src="script.js"></script>
```

```
function displayMovieData(movie, reviews) {
   const results = document.getElementById('results');
   results.innerHTML =
            <h2>${movie.title} (${movie.release_date.split()})
            ${movie.overview}
            <img src="https://image.tmdb.org/t/p/w200${movie.</pre>
poster_path}" alt="${movie.title}">
            IMDB: ${reviews.imdbRating} ${convertToStars(
reviews.imdbRating)}
           Rotten Tomatoes: ${reviews.Ratings.find(rating)}
 => rating.Source === "Rotten Tomatoes")?.Value || 'N/A'}
function convertToStars(rating) {
   const ratingOutOf5 = (rating / 10) * starsTotal;
    for (let i = 0; i < starsTotal; i++) {
        if (i < ratingOutOf5) {
           starDisplay += '*;
        } else {
```

```
const ws = new WebSocket('ws://localhost:8080');

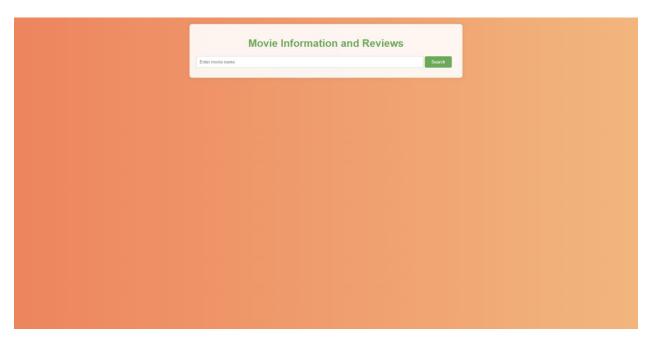
ws.onmessage = (event) => {
    const data = JSON.parse(event.data);
    if (data.error) {
        document.getElementById('results').innerHTML = `${
        data.error}
} else {
        displayMovieData(data.movie, data.ratings);
}

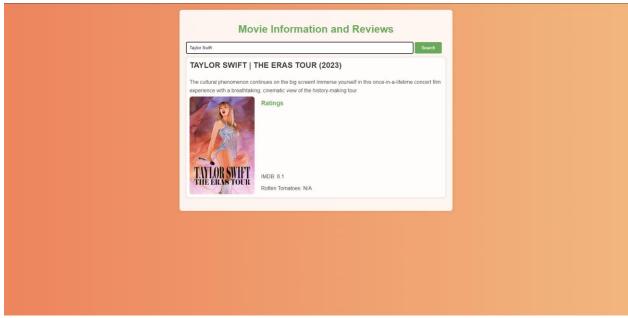
function fetchMovieData() {
    const query = document.getElementById('search').value;
    ws.send(query);
}
```

```
font-family: Arial, sans-serif;
   background: linear-gradient(to right, #ff7e5f, #feb47b);
   margin: 0;
   padding: 0;
.container {
   max-width: 800px;
   margin: 20px auto;
   padding: 20px;
   background-color: rgba(255, 255, 255, 0.9);
   box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
   border-radius: 8px;
   text-align: center;
   margin-bottom: 20px;
input[type="text"] {
   width: calc(100% - 100px);
   padding: 10px;
   margin-bottom: 10px;
   border-radius: 4px;
   box-sizing: border-box;
   padding: 10px 20px;
   background-color: #4CAF50;
   border: none;
   border-radius: 4px;
   background-color: #45a049;
   margin-bottom: 20px;
   padding: 10px;
   background-color: rgba(255, 255, 255, 0.8);
   border: 1px solid #ddd;
   border-radius: 8px;
   display: flex;
   flex-wrap: wrap;
   box-shadow: 0 0 5px rgba(0, 0, 0, 0.1);
```

```
.movie img {
   max-width: 100%;
   border-radius: 8px;
   margin-right: 20px;
.movie h2 {
   margin-top: 0;
   flex-basis: 100%;
.movie h3 {
   color: #4CAF50;
   flex-basis: 100%;
   margin-top: 0;
   display: flex;
   flex-direction: column;
   margin-top: 10px;
.ratings p {
   margin: 5px 0;
.movie p {
   margin: 5px 0;
   line-height: 1.6;
   margin-left: 5px;
@media (max-width: 600px) {
       padding: 10px;
   .movie {
        align-items: center;
   .movie img {
       margin: 0 0 20px 0;
   input[type="text"] {
       width: calc(100% - 90px);
```

```
const express = require('express');
const path = require('path');
const WebSocket = require('ws');
const http = require('http');
const app = express();
const server = http.createServer(app);
const wss = new WebSocket.Server({ server });
const TMDB_API_KEY = 'af5d59325a310881a3af0db5878d94f3';
const OMDB_API_KEY = '23d3a334';
app.use(express.static(path.join(__dirname, 'public')));
wss.on('connection', (ws) => {
    ws.on('message', async (message) => {
        const query = message.toString();
        try {
            const fetch = await import('node-fetch').then(
module => module.default);
            const tmdbResponse = await fetch(
`https://api.themoviedb.org/3/search/movie?api_key=${
TMDB_API_KEY}&query=${query}`);
            const tmdbData = await tmdbResponse.json();
            const movie = tmdbData.results[0];
            if (movie) {
                const omdbResponse = await fetch(
`http://www.omdbapi.com/?t=${movie.title}&apikey=${
OMDB_API_KEY}`);
               const omdbData = await omdbResponse.json();
               ws.send(JSON.stringify({ movie, ratings:
omdbData }));
            } else {
                ws.send(JSON.stringify({ error:
       } catch (error) {
            ws.send(JSON.stringify({ error:
'Error fetching movie data' }));
   });
});
server.listen(8080, () => {
   console.log('Server is listening on port 8080');
});
```





| 7              | ation and Reviews |  |
|----------------|-------------------|--|
| ir0n man       | Search            |  |
| No movie found |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |
|                |                   |  |