

# Movie and TV Show Explorer

**Main Objective:**

The project aims to develop a comprehensive web application incorporating various functionalities such as Dynamic Single-Page Website, API integration, and Web Socket. The application demonstrates modern web development practices with a focus on both frontend and backend technologies.

**Functionalities and Features:**

**JavaScript Functionality:** Handles and manages the main functions of the dynamic content

**API Integration:** The application will leverage external APIs to enhance its functionality. Specifically, it will utilize:

- TMDb API: To retrieve detailed information about movies.
- News API: To fetch and display news articles and updates.

**Responsive Design:** The application will adapt its layout and functionality to orderly display on various devices.

**Web Socket Implementation:** The application will employ WebSockets to enable real-time search suggestions. This will provide users with instant feedback as they type their search queries.

**Dynamic Content:** Website updates based on user interactions. The application will load content dynamically without full page reloads, resulting in a faster and more seamless user experience.

**Program Structure:**

```
-WEBDEV-FINALS-MAIN
├── assets/
│   ├── images/
│   │   └── placeholder.png
├── scripts/
│   ├── api.js
│   └── websocket.js
├── server/
│   ├── node_modules/
│   ├── package-lock.json
│   ├── package.json
│   └── server.js
├── index.html
├── script.js
└── styles.css
```

Code Documentation:

index.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Movie and TV Show Explorer</title>
7   <link rel="stylesheet" href="styles.css">
8 </head>
9 <body>
10   <header>
11     <h1 id="home-button" style="cursor: pointer;">Movie and TV Show Explorer</h1>
12     <div style="position: relative; display: inline-block;">
13       <input type="text" id="search-input" placeholder="Search for movies or TV shows...">
14       <div id="suggestions" class="suggestions-container"></div>
15     </div>
16     <button id="search-button">Search</button>
17   </header>
18
19   <main>
20     <section id="movie-details-container" class="hidden"></section>
21     <section id="search-results"></section>
22     <section id="news-articles" class="hidden"></section>
23   </main>
24
25   <div id="loading-indicator" style="display: none;">
26     <svg xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" style="margin: auto; background: transparent; display: block; shape-rendering: auto;" width="200px" height="200px" viewBox="0 0 100 100" preserveAspectRatio="xMidYMid">
27       <g transform="rotate(0 50 50)">
28         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
29           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.9166666666666666s" repeatCount="indefinite"></animate>
30         </rect>
31       </g><g transform="rotate(30 50 50)">
32         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
33           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.8333333333333334s" repeatCount="indefinite"></animate>
34         </rect>
35       </g><g transform="rotate(60 50 50)">
36         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
37           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.75s" repeatCount="indefinite"></animate>
38         </rect>
39       </g><g transform="rotate(90 50 50)">
40         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
41           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.6666666666666666s" repeatCount="indefinite"></animate>
42         </rect>
43       </g><g transform="rotate(120 50 50)">
44         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
45           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.5833333333333334s" repeatCount="indefinite"></animate>
46         </rect>
47       </g><g transform="rotate(150 50 50)">
48         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
49           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.5s" repeatCount="indefinite"></animate>
50         </rect>
51       </g><g transform="rotate(180 50 50)">
52         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
53           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.4166666666666667s" repeatCount="indefinite"></animate>
54         </rect>
55       </g><g transform="rotate(210 50 50)">
56         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
57           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.3333333333333333s" repeatCount="indefinite"></animate>
58         </rect>
59       </g><g transform="rotate(240 50 50)">
60         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
61           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.25s" repeatCount="indefinite"></animate>
62         </rect>
63       </g><g transform="rotate(270 50 50)">
64         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
65           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.1666666666666666s" repeatCount="indefinite"></animate>
66         </rect>
67       </g><g transform="rotate(300 50 50)">
68         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
69           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="-0.08333333333333333s" repeatCount="indefinite"></animate>
70         </rect>
71       </g><g transform="rotate(330 50 50)">
72         <rect x="47" y="24" rx="3" ry="6" width="6" height="12" fill="#e15b64">
73           <animate attributeName="opacity" values="1;0" keyTimes="0;1" dur="1s" begin="0s" repeatCount="indefinite"></animate>
74         </rect>
75     </g>
76   </svg>
77 </div>
78
79   <script type="module" src="./scripts/api.js"></script>
80   <script type="module" src="./scripts/websocket.js"></script>
81   <script type="module" src="./script.js"></script>
82 </body>
83 </html>
84
```

# script.js

```
1 import { setupWebSocket, sendSearchQuery } from './scripts/websocket.js';
2 import { fetchMovies, fetchMovieDetails, fetchMovieCast, fetchNewsArticles } from './scripts/api.js';
3
4 document.addEventListener('DOMContentLoaded', () => {
5   const searchInput = document.getElementById('search-input');
6   const searchButton = document.getElementById('search-button');
7   const searchResults = document.getElementById('search-results');
8   const movieDetailsContainer = document.getElementById('movie-details-container');
9   const newsArticlesContainer = document.getElementById('news-articles');
10  const loadingIndicator = document.getElementById('loading-indicator');
11  const homeButton = document.getElementById('home-button');
12
13  setupWebSocket();
14
15  movieDetailsContainer.classList.add('hidden');
16  newsArticlesContainer.classList.add('hidden');
17
18  searchInput.addEventListener('input', () => {
19    const query = searchInput.value;
20    if (query) {
21      sendSearchQuery(query);
22    } else {
23      document.getElementById('suggestions').innerHTML = '';
24    }
25  });
26
27  searchButton.addEventListener('click', () => {
28    const query = searchInput.value;
29    if (query) {
30
31      searchResults.innerHTML = '';
32      movieDetailsContainer.innerHTML = '';
33      newsArticlesContainer.innerHTML = '';
34      document.getElementById('suggestions').innerHTML = '';
35
36      movieDetailsContainer.classList.add('hidden');
37      newsArticlesContainer.classList.add('hidden');
38
39      loadingIndicator.style.display = 'block';
40
41      fetchMovies(query).then(displayResults).finally(() => {
42        loadingIndicator.style.display = 'none';
43      });
44    }
45  });
46
47  const displayResults = (movies) => {
48    searchResults.innerHTML = '';
49    if (movies.length === 0) {
50      searchResults.innerHTML = '<p>No results found</p>';
51      return;
52    }
53    movies.forEach(movie => {
54      const movieElement = document.createElement('div');
55      movieElement.className = 'movie';
56      const posterPath = movie.poster_path ? `https://image.tmdb.org/t/p/w200${movie.poster_path}` : 'assets/images/placeholder.png';
57      movieElement.innerHTML = `
58        
59        <h3>${movie.title}</h3>
60        <p>${movie.release_date}</p>
61        <button class="details-button" onclick="showDetails(${movie.id}, '${movie.title}')">Details</button>
62      `;
63      searchResults.appendChild(movieElement);
64    });
65  };
66
67  window.showDetails = async (movieId, title) => {
68
69    loadingIndicator.style.display = 'block';
70
71    const movieDetailsData = await fetchMovieDetails(movieId);
72    const movieCast = await fetchMovieCast(movieId);
73    const newsArticles = await fetchNewsArticles(title);
74
75    loadingIndicator.style.display = 'none';
76
77    if (movieDetailsData) {
78      displayMovieDetails(movieDetailsData, movieCast, newsArticles);
79      scrollToDetails();
80
81      movieDetailsContainer.classList.remove('hidden');
82      newsArticlesContainer.classList.remove('hidden');
83    }
84  };
85 }
```

```

1  const displayMovieDetails = (movie, cast, newsArticles) => {
2      const castList = cast.map(actor => `<li>${actor.name} as ${actor.character}</li>`).join('');
3      const newList = newsArticles.length > 0 ? newsArticles.map(article => `
4          <div class="news-article">
5              <h3>${article.title}</h3>
6              <p>${article.description}</p>
7              <a href="${article.url}" target="_blank">Read more</a>
8          </div>
9      `).join('') : `<p>No articles found</p>`;
10
11      const posterPath = movie.poster_path ? `https://image.tmdb.org/t/p/w500${movie.poster_path}` : 'assets/images/placeholder.png';
12
13      movieDetailsContainer.innerHTML = `
14          <div>
15              
16          </div>
17          <div id="movie-details">
18              <h2>${movie.title}</h2>
19              <p id="movie-overview">${movie.overview}</p>
20              <p><strong>Release Date:</strong> ${movie.release_date}</p>
21              <p><strong>Rating:</strong> ${movie.vote_average}</p>
22              <h3>Cast:</h3>
23              <ul class="cast-list">${castList}</ul>
24              <button id="watch-trailer" onclick="window.open('https://www.youtube.com/results?search_query=${movie.title}+trailer', '_blank')">Watch Trailer</button>
25              <button id="view-articles">View Articles</button>
26          </div>
27      `;
28
29      newsArticlesContainer.innerHTML = `
30          <h2>Related News Articles</h2>
31          ${newList}
32      `;
33
34
35      document.getElementById('view-articles').addEventListener('click', (event) => {
36          event.preventDefault();
37          document.querySelector('#news-articles').scrollIntoView({ behavior: 'smooth' });
38      });
39
40
41      const scrollToDetails = () => {
42          movieDetailsContainer.scrollIntoView({ behavior: 'smooth' });
43      };
44
45
46      homeButton.addEventListener('click', () => {
47
48          searchInput.value = '';
49
50          searchResults.innerHTML = '';
51          movieDetailsContainer.innerHTML = '';
52          newsArticlesContainer.innerHTML = '';
53          document.getElementById('suggestions').innerHTML = '';
54
55          movieDetailsContainer.classList.add('hidden');
56          newsArticlesContainer.classList.add('hidden');
57      });
58  });

```

styles.css

```
1  @import url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;600&display=swap');
2
3  body {
4    font-family: 'Poppins', Arial, sans-serif;
5    text-align: center;
6    background-color: #121212;
7    color: #fff;
8  }
9
10 header {
11   margin-bottom: 20px;
12   position: relative;
13 }
14
15 #search-input {
16   padding: 10px;
17   width: 280px;
18 }
19
20 #search-button {
21   padding: 10px;
22   margin-left: 10px;
23   background-color: #6200ee;
24   color: white;
25   border: none;
26   cursor: pointer;
27   font-family: 'Poppins', Arial, sans-serif;
28   border-radius: 10px;
29 }
30
31 #search-button:hover {
32   background-color: #3700b3;
33 }
34
35 .suggestions-container {
36   position: absolute;
37   top: 100%;
38   left: 0;
39   width: 300px;
40   background-color: white;
41   color: black;
42   border: 1px solid #ccc;
43   max-height: 200px;
44   overflow-y: auto;
45   z-index: 1000;
46 }
47
48 .suggestion {
49   padding: 10px;
50   cursor: pointer;
51   font-family: 'Poppins', Arial, sans-serif;
52 }
53
54 .suggestion:hover {
55   background-color: #eee;
56 }
57
58 #search-results {
59   display: flex;
60   flex-wrap: wrap;
61   justify-content: center;
62   margin-top: 20px;
63 }
64
65 .movie {
66   border: 1px solid #ddd;
67   padding: 10px;
68   margin: 10px;
69   width: 200px;
70   background-color: #1e1e1e;
71   font-family: 'Poppins', Arial, sans-serif;
72 }
73
74 #movie-details-container {
75   display: flex;
76   flex-direction: column;
77   align-items: center;
78   margin-top: 20px;
79   padding: 20px;
80   border: 1px solid #444;
81   background-color: #1e1e1e;
82 }
83
84 #movie-details {
85   padding-left: 20px;
86   font-family: 'Poppins', Arial, sans-serif;
87   text-align: left;
88 }
```

```
1  #movie-poster {
2      max-width: 300px;
3      max-height: 450px;
4  }
5
6  ul {
7      list-style-type: none;
8      padding: 0;
9  }
10
11  li {
12      text-align: left;
13      margin: 5px 0;
14      font-family: 'Poppins', Arial, sans-serif;
15  }
16
17  .cast-list {
18      max-height: 150px;
19      overflow-y: auto;
20      border: 1px solid #444;
21      padding: 10px;
22      background-color: #2a2a2a;
23  }
24
25  .cast-list li {
26      color: #ccc;
27      font-family: 'Poppins', Arial, sans-serif;
28  }
29
30  #loading-indicator {
31      position: fixed;
32      top: 50%;
33      left: 50%;
34      transform: translate(-50%, -50%);
35      z-index: 9999;
36      fill: #6200ee;
37  }
38
39  #outline {
40      stroke-dasharray: 2.42777px, 242.77666px;
41      stroke-dashoffset: 0;
42      -webkit-animation: anim 1.6s linear infinite;
43      animation: anim 1.6s linear infinite;
44  }
45
46  #news-articles {
47      margin-top: 20px;
48  }
49
50  .news-article {
51      background-color: #1c1c1c;
52      border-radius: 8px;
53      padding: 15px;
54      margin-bottom: 10px;
55      box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
56  }
57
58  .news-article h3 {
59      margin-top: 0;
60      color: #ff6b6b;
61      font-size: 1.2em;
62  }
63
64  .news-article p {
65      color: #ddd;
66      margin-bottom: 10px;
67  }
68
69  .news-article a {
70      color: #1e90ff;
71      text-decoration: none;
72      font-weight: bold;
73  }
74
75  .news-article a:hover {
76      text-decoration: underline;
77  }
78
79  #watch-trailer, #view-articles, .details-button {
80      padding: 10px 20px;
81      background-color: #6200ee;
82      color: white;
83      border: none;
84      cursor: pointer;
85      font-family: 'Poppins', Arial, sans-serif;
86      margin: 5px;
87      border-radius: 50px;
88  }
```

```
1  #watch-trailer:hover, #view-articles:hover, .details-button:hover {
2    background-color: #3700b3;
3  }
4
5
6  @media (min-width: 600px) {
7    #movie-details-container {
8      flex-direction: row;
9    }
10
11    #movie-details {
12      flex: 1;
13    }
14
15    #movie-poster {
16      margin-right: 20px;
17    }
18  }
19
20  @media (max-width: 599px) {
21    #search-input, #search-button {
22      width: 100%;
23      margin-bottom: 10px;
24    }
25
26    #movie-details-container {
27      flex-direction: column;
28      align-items: center;
29    }
30
31    #movie-details {
32      padding: 0;
33    }
34
35    .cast-list {
36      width: 100%;
37    }
38  }
39  @media (min-width: 600px) {
40    #movie-details-container {
41      flex-direction: row;
42    }
43
44    #movie-details {
45      flex: 1;
46    }
47
48    #movie-poster {
49      margin-right: 20px;
50    }
51  }
52
53  @media (max-width: 599px) {
54    #search-input {
55      width: 90%;
56      margin-bottom: 0px;
57    }
58
59    #search-button {
60      width: 70px;
61    }
62
63    #movie-details-container {
64      flex-direction: column;
65      align-items: center;
66    }
67
68    #movie-details {
69      padding: 0;
70    }
71
72    .cast-list {
73      width: 100%;
74    }
75
76    .suggestions-container {
77      width: 195px;
78    }
79
80    .suggestion {
81      padding: 10px;
82    }
83  }
84
```

## websocket.js

```
1  let socket;
2
3  export const setupWebSocket = () => {
4    socket = new WebSocket('ws://localhost:8080');
5
6    socket.onopen = () => {
7      console.log('WebSocket connection established');
8    };
9
10   socket.onmessage = (event) => {
11     const suggestions = JSON.parse(event.data);
12     console.log('WebSocket message received:', suggestions);
13     displaySuggestions(suggestions);
14   };
15
16   socket.onclose = () => {
17     console.log('WebSocket connection closed');
18   };
19
20   socket.onerror = (error) => {
21     console.error('WebSocket error:', error);
22   };
23
24   return socket;
25 };
26
27 const displaySuggestions = (suggestions) => {
28   const suggestionsContainer = document.getElementById('suggestions');
29   suggestionsContainer.innerHTML = '';
30
31   const uniqueTitles = new Set();
32   suggestions.forEach(suggestion => {
33     if (!uniqueTitles.has(suggestion.title)) {
34       uniqueTitles.add(suggestion.title);
35       const suggestionElement = document.createElement('div');
36       suggestionElement.className = 'suggestion';
37       suggestionElement.innerText = suggestion.title;
38       suggestionElement.onclick = () => {
39         document.getElementById('search-input').value = suggestion.title;
40         suggestionsContainer.innerHTML = '';
41
42         document.getElementById('search-button').click();
43       };
44       suggestionsContainer.appendChild(suggestionElement);
45     }
46   });
47 };
48
49 export const sendSearchQuery = (query) => {
50   if (socket && socket.readyState === WebSocket.OPEN) {
51     socket.send(query);
52   }
53 };
54
```



## server.js

```
1  const WebSocket = require('ws');
2
3  let fetch;
4  (async () => {
5    fetch = (await import('node-fetch')).default;
6  })();
7
8  const wss = new WebSocket.Server({ port: 8080 });
9
10 wss.on('connection', (ws) => {
11   console.log('New client connected');
12
13   ws.on('message', async (message) => {
14     const query = message.toString();
15     console.log(`Received search query: ${query}`);
16
17     const suggestions = await fetchSearchSuggestions(query);
18
19     ws.send(JSON.stringify(suggestions));
20   });
21
22   ws.on('close', () => {
23     console.log('Client disconnected');
24   });
25 });
26
27 console.log('WebSocket server is running on ws://localhost:8080');
28
29 async function fetchSearchSuggestions(query) {
30   const apiKey = '4fc23459f91f0511543f8721c6be3214';
31   const url = `https://api.themoviedb.org/3/search/movie?api_key=${apiKey}&query=${query}`;
32   try {
33     const response = await fetch(url);
34     const data = await response.json();
35     return data.results.map(movie => ({
36       id: movie.id,
37       title: movie.title,
38     }));
39   } catch (error) {
40     console.error('Error fetching search suggestions:', error);
41     return [];
42   }
43 }
44
```

## api.js

```
1  const tmdbApiKey = '4fc23459f91f0511543f8721c6be3214';
2  const newsApiKey = '13f157f4c52c41ac840247c96f187c7c';
3  const newsApiUrl = 'https://newsapi.org/v2/everything';
4
5  export const fetchMovies = async (query) => {
6    const url = `https://api.themoviedb.org/3/search/movie?api_key=${tmdbApiKey}&query=${query}`;
7    try {
8      const response = await fetch(url);
9      const data = await response.json();
10     return data.results;
11   } catch (error) {
12     console.error('Error fetching movies:', error);
13     return [];
14   }
15 };
16
17 export const fetchMovieDetails = async (movieId) => {
18   const url = `https://api.themoviedb.org/3/movie/${movieId}?api_key=${tmdbApiKey}`;
19   try {
20     const response = await fetch(url);
21     const movie = await response.json();
22     return movie;
23   } catch (error) {
24     console.error('Error fetching movie details:', error);
25     return null;
26   }
27 };
28
29 export const fetchMovieCast = async (movieId) => {
30   const url = `https://api.themoviedb.org/3/movie/${movieId}/credits?api_key=${tmdbApiKey}`;
31   try {
32     const response = await fetch(url);
33     const data = await response.json();
34     return data.cast;
35   } catch (error) {
36     console.error('Error fetching movie cast:', error);
37     return [];
38   }
39 };
40
41 export const fetchNewsArticles = async (query) => {
42   const url = `${newsApiUrl}?q=${encodeURIComponent(query)}&apiKey=${newsApiKey}`;
43   try {
44     const response = await fetch(url);
45     const data = await response.json();
46     return data.articles.slice(0, 5);
47   } catch (error) {
48     console.error('Error fetching news articles:', error);
49     return [];
50   }
51 };
52
```