I'll create a movie recommendation application using HTML, CSS, and JavaScript with a DevOps-inspired UI style. The app will feature a clean, functional interface with container-like movie cards and deployment-inspired visual elements.

1. For a DevOps-inspired design, I'll use color schemes common in monitoring dashboards (blues, greens, grays)
2. Will implement movie cards styled like container boxes with status indicators
3. Need to include:
   - Search functionality
   - Filter/sort options
   - Recommendation engine logic
   - Responsive design
4. Will use https://placehold.co for any placeholder images with detailed alt text
5. Will implement a clean, functional UI with DevOps aesthetics (terminal-like elements, metrics displays)

Here's the complete implementation:

`movie_recommendation.html`
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>DevOps Movie Recommendations</title>
    <style>
        :root {
            --primary: #2962ff;
            --secondary: #00c853;
            --dark: #263238;
            --light: #f5f5f5;
            --warning: #ffab00;
            --error: #d50000;
            --terminal: #0d0d0d;
        }

        body {
            font-family: 'Courier New', monospace;
            margin: 0;
            padding: 0;
            background-color: var(--terminal);
            color: var(--light);
            line-height: 1.6;
        }

        .container {
            max-width: 1200px;
            margin: 0 auto;
            padding: 20px;
        }

        header {
            background-color: var(--dark);
            padding: 20px 0;
```

```
  border-bottom: 3px solid var(--primary);
}

.header-content {
  display: flex;
  justify-content: space-between;
  align-items: center;
}

h1 {
  color: var(--primary);
  margin: 0;
  font-size: 1.8rem;
}

.logo {
  font-weight: bold;
  color: var(--secondary);
  display: flex;
  align-items: center;
}

.logo-icon {
  margin-right: 10px;
  color: var(--secondary);
}

.search-container {
  margin: 30px 0;
}

.search-box {
  display: flex;
  align-items: center;
  background-color: var(--dark);
  padding: 10px;
  border-radius: 4px;
  box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);
}

.search-box input {
  flex-grow: 1;
  background: transparent;
  border: none;
  border-bottom: 1px solid var(--primary);
  padding: 10px;
  color: var(--light);
  font-family: 'Courier New', monospace;
  outline: none;
}

.search-box button {
  background-color: var(--primary);
  color: white;
  border: none;
  padding: 10px 20px;
  border-radius: 4px;
```

```
    cursor: pointer;
    margin-left: 10px;
    transition: background-color 0.3s;
}

.search-box button:hover {
    background-color: #0039cb;
}

.filters {
    display: flex;
    gap: 15px;
    margin-bottom: 20px;
    flex-wrap: wrap;
}

.filter {
    background-color: var(--dark);
    padding: 8px 15px;
    border-radius: 4px;
    cursor: pointer;
    transition: all 0.3s;
    border: 1px solid transparent;
}

.filter:hover {
    border-color: var(--primary);
}

.filter.active {
    background-color: var(--primary);
    color: white;
}

.movie-grid {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(280px, 1fr));
    gap: 20px;
}

.movie-card {
    background-color: var(--dark);
    border-radius: 6px;
    overflow: hidden;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
    transition: transform 0.3s, box-shadow 0.3s;
    border-left: 4px solid var(--primary);
    position: relative;
}

.movie-card:hover {
    transform: translateY(-5px);
    box-shadow: 0 10px 20px rgba(0, 0, 0, 0.3);
}

.movie-image {
    width: 100%;
    height: 180px;
    object-fit: cover;
}

.movie-content {
    padding: 15px;
}

.movie-title {
    margin: 0;
    color: var(--primary);
    font-size: 1.2rem;
    white-space: nowrap;
    overflow: hidden;
    text-overflow: ellipsis;
}

.movie-meta {
    display: flex;
    justify-content: space-between;
    margin: 8px 0;
    color: #b0bec5;
    font-size: 0.9rem;
}

.movie-rating {
    display: flex;
    align-items: center;
}

.rating-star {
    color: var(--warning);
    margin-right: 5px;
}

.movie-description {
    font-size: 0.9rem;
    color: #cfd8dc;
    margin-bottom: 15px;
    display: -webkit-box;
    -webkit-line-clamp: 3;
    -webkit-box-orient: vertical;
    overflow: hidden;
}

.movie-actions {
    display: flex;
    justify-content: space-between;
}

.btn {
    padding: 8px 15px;
    border: none;
    border-radius: 4px;
    cursor: pointer;
    font-family: 'Courier New', monospace;
    font-weight: bold;
    transition: background-color 0.3s;
}

.btn-primary {
    background-color: var(--primary);
    color: white;
}

.btn-primary:hover {
    background-color: #0039cb;
}

.btn-secondary {
    background-color: transparent;
    color: var(--primary);
    border: 1px solid var(--primary);
}

.btn-secondary:hover {
    background-color: rgba(41, 98, 255, 0.1);
}

.status-badge {
    position: absolute;
    top: 10px;
    right: 10px;
    padding: 3px 8px;
    border-radius: 4px;
    font-size: 0.8rem;
    font-weight: bold;
}

.status-high {
    background-color: var(--secondary);
    color: var(--terminal);
}

.status-medium {
    background-color: var(--warning);
    color: var(--dark);
}

.status-low {
    background-color: var(--error);
    color: white;
}

.metrics {
    background-color: var(--dark);
    padding: 15px;
    border-radius: 6px;
    margin-bottom: 20px;
    display: flex;
    justify-content: space-between;
    border-left: 4px solid var(--secondary);
}
```

```
.metric {
  text-align: center;
}

.metric-value {
  font-size: 1.8rem;
  font-weight: bold;
  color: var(--secondary);
}

.metric-label {
  font-size: 0.9rem;
  color: #b0bec5;
}

.terminal-output {
  background-color: #0d0d0d;
  border: 1px solid var(--primary);
  border-radius: 6px;
  padding: 15px;
  margin-top: 20px;
  font-family: 'Courier New', monospace;
  color: var(--light);
  height: 150px;
  overflow-y: auto;
  position: relative;
}

.terminal-output::before {
  content: "> Recommendation Engine Logs";
  position: absolute;
  top: -12px;
  left: 15px;
  background-color: var(--terminal);
  padding: 0 10px;
  font-size: 0.8rem;
  color: var(--secondary);
}

.log-entry {
  margin-bottom: 5px;
  padding-left: 15px;
  position: relative;
}

.log-entry::before {
  content: "$";
  position: absolute;
  left: 0;
  color: var(--secondary);
}

@media (max-width: 768px) {
  .movie-grid {
    grid-template-columns: repeat(auto-fill, minmax(220px, 1fr));
  }
```

```
            .filters {
                justify-content: center;
            }

            .metrics {
                flex-direction: column;
                gap: 15px;
            }
        }
    </style>
</head>
<body>
    <header>
        <div class="container">
            <div class="header-content">
                <h1 class="logo">
                    <span class="logo-icon">[MovieOps]</span> DevOps Movie Recommendations
                </h1>
            </div>
        </div>
    </header>

    <main class="container">
        <div class="metrics">
            <div class="metric">
                <div class="metric-value">1287</div>
                <div class="metric-label">Movies Analyzed</div>
            </div>
            <div class="metric">
                <div class="metric-value">94.3%</div>
                <div class="metric-label">Match Accuracy</div>
            </div>
            <div class="metric">
                <div class="metric-value">3.2ms</div>
                <div class="metric-label">Avg. Response Time</div>
            </div>
        </div>

        <div class="search-container">
            <div class="search-box">
                <input type="text" id="searchInput" placeholder="Search for movies...">
                <button id="searchBtn">Search</button>
            </div>
        </div>

        <div class="filters">
            <div class="filter active" data-filter="all">All</div>
            <div class="filter" data-filter="action">Action</div>
            <div class="filter" data-filter="comedy">Comedy</div>
            <div class="filter" data-filter="drama">Drama</div>
            <div class="filter" data-filter="sci-fi">Sci-Fi</div>
            <div class="filter" data-filter="horror">Horror</div>
        </div>

        <div class="movie-grid" id="movieContainer">
            <!-- Movie cards will be dynamically inserted here -->
        </div>
    <div class="terminal-output" id="terminalOutput">
      <div class="log-entry">Initializing recommendation engine...</div>
      <div class="log-entry">Loading movie database...</div>
      <div class="log-entry">Database connection established</div>
      <div class="log-entry">Analyzing user preferences...</div>
      <div class="log-entry">Ready to provide recommendations</div>
    </div>
  </main>

  <script>
    // Movie database
    const movies = [
      {
        id: 1,
        title: "The Matrix Reloaded",
        genre: ["action", "sci-fi"],
        year: 2003,
        rating: 7.2,
        description: "Neo and the rebel leaders estimate they have 72 hours until Zion falls under siege to the Machine army. Neo must decide how he can save Trinity from a dark fate in his dreams.",
        imageAlt: "Neo in black trench coat dodging bullets in a futuristic city"
      },
      {
        id: 2,
        title: "Pulp Fiction",
        genre: ["drama", "crime"],
        year: 1994,
        rating: 8.9,
        description: "The lives of two mob hitmen, a boxer, a gangster's wife, and a pair of diner bandits intertwine in four tales of violence and redemption.",
        imageAlt: "Uma Thurman and John Travolta dancing in a 50s style diner"
      },
      {
        id: 3,
        title: "Inception",
        genre: ["action", "sci-fi", "thriller"],
        year: 2010,
        rating: 8.8,
        description: "A thief who steals corporate secrets through the use of dream-sharing technology is given the inverse task of planting an idea into the mind of a C.E.O.",
        imageAlt: "Dreamlike cityscape folding in on itself with Leonardo DiCaprio standing in foreground"
      },
      {
        id: 4,
        title: "Parasite",
        genre: ["drama", "thriller"],
        year: 2019,
        rating: 8.6,
        description: "Greed and class discrimination threaten the newly formed symbiotic relationship between the wealthy Park family and the destitute Kim clan.",
        imageAlt: "Korean family looking up from a semi-basement apartment window"
      },
      {
        id: 5,
        title: "The Grand Budapest Hotel",
        genre: ["comedy", "drama"],
        year: 2014,
        rating: 8.1,
        description: "The adventures of Gustave H, a legendary concierge at a famous European
hotel between the wars, and Zero Moustafa, the lobby boy who becomes his most trusted friend.",
        imageAlt: "Pink hotel facade with Wes Anderson symmetrical composition style"
      },
      {
        id: 6,
        title: "Get Out",
        genre: ["horror", "thriller"],
        year: 2017,
        rating: 7.7,
        description: "A young African-American visits his white girlfriend's parents for the
weekend, where his simmering uneasiness about their reception of him eventually reaches a boiling
point.",
        imageAlt: "Daniel Kaluuya looking anxious in an upscale suburban home"
      }
    ];

    // DOM elements
    const movieContainer = document.getElementById('movieContainer');
    const searchInput = document.getElementById('searchInput');
    const searchBtn = document.getElementById('searchBtn');
    const filters = document.querySelectorAll('.filter');
    const terminalOutput = document.getElementById('terminalOutput');

    // Initialize the app
    function init() {
      renderMovies(movies);
      addEventListeners();
      logToTerminal("System initialized successfully");
    }

    // Render movies to the DOM
    function renderMovies(moviesToRender) {
      movieContainer.innerHTML = '';

      moviesToRender.forEach(movie => {
        const status = getStatusRating(movie.rating);

        const movieCard = document.createElement('div');
        movieCard.className = 'movie-card';
        movieCard.innerHTML = `
          <img src="https://placehold.co/600x400?text=${movie.title.replace(/\s/g, '+')}"
class="movie-image" alt="${movie.imageAlt}" />
          <span class="status-badge status-${status.class}">${status.text}</span>
          <div class="movie-content">
            <h3 class="movie-title">${movie.title}</h3>
            <div class="movie-meta">
              <span>${movie.year}</span>
              <span class="movie-rating">
                <span class="rating-star">★</span>
                ${movie.rating}
              </span>
            </div>
```

```
                <p class="movie-description">${movie.description}</p>
                <div class="movie-actions">
                    <button class="btn btn-primary">Add to Queue</button>
                    <button class="btn btn-secondary">Details</button>
                </div>
            </div>
        `;

        movieContainer.appendChild(movieCard);
    });
}

// Get status rating based on score
function getStatusRating(rating) {
    if (rating >= 8.0) return { class: 'high', text: 'Highly Recommended' };
    if (rating >= 6.5) return { class: 'medium', text: 'Recommended' };
    return { class: 'low', text: 'Fair' };
}

// Filter movies by genre
function filterMovies(genre) {
    if (genre === 'all') {
        renderMovies(movies);
        logToTerminal("Displaying all movies");
        return;
    }

    const filteredMovies = movies.filter(movie =>
        movie.genre.includes(genre)
    );

    renderMovies(filteredMovies);
    logToTerminal(`Filtered movies by genre: ${genre}`);
}

// Search movies
function searchMovies(query) {
    const normalizedQuery = query.toLowerCase();

    const filteredMovies = movies.filter(movie =>
        movie.title.toLowerCase().includes(normalizedQuery) ||
        movie.description.toLowerCase().includes(normalizedQuery)
    );

    renderMovies(filteredMovies);
    logToTerminal(`Search results for: ${query}`);
}

// Add event listeners
function addEventListeners() {
    // Search button
    searchBtn.addEventListener('click', () => {
        const query = searchInput.value.trim();
        if (query) {
            searchMovies(query);
        }
    });
```

~~~
            // Enter key in search input
            searchInput.addEventListener('keypress', (e) => {
                if (e.key === 'Enter') {
                    const query = searchInput.value.trim();
                    if (query) {
                        searchMovies(query);
                    }
                }
            });

            // Filter buttons
            filters.forEach(filter => {
                filter.addEventListener('click', () => {
                    filters.forEach(f => f.classList.remove('active'));
                    filter.classList.add('active');
                    filterMovies(filter.dataset.filter);
                });
            });
        }

        // Log to terminal
        function logToTerminal(message) {
            const logEntry = document.createElement('div');
            logEntry.className = 'log-entry';
            logEntry.textContent = message;
            terminalOutput.appendChild(logEntry);
            terminalOutput.scrollTop = terminalOutput.scrollHeight;
        }

        // Initialize the app
        document.addEventListener('DOMContentLoaded', init);
    </script>
</body>
</html>

```
~~~

Generating images based on descriptions...

✓ **Images generated successfully! Here's your updated content:**

`movie_recommendation.html`

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>DevOps Movie Recommendations</title>
    <style>
        :root {
            --primary: #2962ff;
            --secondary: #00c853;
            --dark: #263238;
            --light: #f5f5f5;
            --warning: #ffab00;
```

```
    --error: #d50000;
    --terminal: #0d0d0d;
}

body {
    font-family: 'Courier New', monospace;
    margin: 0;
    padding: 0;
    background-color: var(--terminal);
    color: var(--light);
    line-height: 1.6;
}

.container {
    max-width: 1200px;
    margin: 0 auto;
    padding: 20px;
}

header {
    background-color: var(--dark);
    padding: 20px 0;
    border-bottom: 3px solid var(--primary);
}

.header-content {
    display: flex;
    justify-content: space-between;
    align-items: center;
}

h1 {
    color: var(--primary);
    margin: 0;
    font-size: 1.8rem;
}

.logo {
    font-weight: bold;
    color: var(--secondary);
    display: flex;
    align-items: center;
}

.logo-icon {
    margin-right: 10px;
    color: var(--secondary);
}

.search-container {
    margin: 30px 0;
}

.search-box {
    display: flex;
    align-items: center;
    background-color: var(--dark);
    padding: 10px;
    border-radius: 4px;
    box-shadow: 0 2px 4px rgba(0, 0, 0, 0.2);
}

.search-box input {
    flex-grow: 1;
    background: transparent;
    border: none;
    border-bottom: 1px solid var(--primary);
    padding: 10px;
    color: var(--light);
    font-family: 'Courier New', monospace;
    outline: none;
}

.search-box button {
    background-color: var(--primary);
    color: white;
    border: none;
    padding: 10px 20px;
    border-radius: 4px;
    cursor: pointer;
    margin-left: 10px;
    transition: background-color 0.3s;
}

.search-box button:hover {
    background-color: #0039cb;
}

.filters {
    display: flex;
    gap: 15px;
    margin-bottom: 20px;
    flex-wrap: wrap;
}

.filter {
    background-color: var(--dark);
    padding: 8px 15px;
    border-radius: 4px;
    cursor: pointer;
    transition: all 0.3s;
    border: 1px solid transparent;
}

.filter:hover {
    border-color: var(--primary);
}

.filter.active {
    background-color: var(--primary);
    color: white;
}

.movie-grid {
    display: grid;
    grid-template-columns: repeat(auto-fill, minmax(280px, 1fr));
    gap: 20px;
}

.movie-card {
    background-color: var(--dark);
    border-radius: 6px;
    overflow: hidden;
    box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
    transition: transform 0.3s, box-shadow 0.3s;
    border-left: 4px solid var(--primary);
    position: relative;
}

.movie-card:hover {
    transform: translateY(-5px);
    box-shadow: 0 10px 20px rgba(0, 0, 0, 0.3);
}

.movie-image {
    width: 100%;
    height: 180px;
    object-fit: cover;
}

.movie-content {
    padding: 15px;
}

.movie-title {
    margin: 0;
    color: var(--primary);
    font-size: 1.2rem;
    white-space: nowrap;
    overflow: hidden;
    text-overflow: ellipsis;
}

.movie-meta {
    display: flex;
    justify-content: space-between;
    margin: 8px 0;
    color: #b0bec5;
    font-size: 0.9rem;
}

.movie-rating {
    display: flex;
    align-items: center;
}

.rating-star {
    color: var(--warning);
    margin-right: 5px;
}
```

```
.movie-description {
  font-size: 0.9rem;
  color: #cfd8dc;
  margin-bottom: 15px;
  display: -webkit-box;
  -webkit-line-clamp: 3;
  -webkit-box-orient: vertical;
  overflow: hidden;
}

.movie-actions {
  display: flex;
  justify-content: space-between;
}

.btn {
  padding: 8px 15px;
  border: none;
  border-radius: 4px;
  cursor: pointer;
  font-family: 'Courier New', monospace;
  font-weight: bold;
  transition: background-color 0.3s;
}

.btn-primary {
  background-color: var(--primary);
  color: white;
}

.btn-primary:hover {
  background-color: #0039cb;
}

.btn-secondary {
  background-color: transparent;
  color: var(--primary);
  border: 1px solid var(--primary);
}

.btn-secondary:hover {
  background-color: rgba(41, 98, 255, 0.1);
}

.status-badge {
  position: absolute;
  top: 10px;
  right: 10px;
  padding: 3px 8px;
  border-radius: 4px;
  font-size: 0.8rem;
  font-weight: bold;
}

.status-high {
  background-color: var(--secondary);
  color: var(--terminal);
```

```
}

.status-medium {
   background-color: var(--warning);
   color: var(--dark);
}

.status-low {
   background-color: var(--error);
   color: white;
}

.metrics {
   background-color: var(--dark);
   padding: 15px;
   border-radius: 6px;
   margin-bottom: 20px;
   display: flex;
   justify-content: space-between;
   border-left: 4px solid var(--secondary);
}

.metric {
   text-align: center;
}

.metric-value {
   font-size: 1.8rem;
   font-weight: bold;
   color: var(--secondary);
}

.metric-label {
   font-size: 0.9rem;
   color: #b0bec5;
}

.terminal-output {
   background-color: #0d0d0d;
   border: 1px solid var(--primary);
   border-radius: 6px;
   padding: 15px;
   margin-top: 20px;
   font-family: 'Courier New', monospace;
   color: var(--light);
   height: 150px;
   overflow-y: auto;
   position: relative;
}

.terminal-output::before {
   content: "> Recommendation Engine Logs";
   position: absolute;
   top: -12px;
   left: 15px;
   background-color: var(--terminal);
   padding: 0 10px;
```

```
            font-size: 0.8rem;
            color: var(--secondary);
        }

        .log-entry {
            margin-bottom: 5px;
            padding-left: 15px;
            position: relative;
        }

        .log-entry::before {
            content: "$";
            position: absolute;
            left: 0;
            color: var(--secondary);
        }

        @media (max-width: 768px) {
            .movie-grid {
                grid-template-columns: repeat(auto-fill, minmax(220px, 1fr));
            }

            .filters {
                justify-content: center;
            }

            .metrics {
                flex-direction: column;
                gap: 15px;
            }
        }
    </style>
</head>
<body>
    <header>
        <div class="container">
            <div class="header-content">
                <h1 class="logo">
                    <span class="logo-icon">[MovieOps]</span> DevOps Movie Recommendations
                </h1>
            </div>
        </div>
    </header>

    <main class="container">
        <div class="metrics">
            <div class="metric">
                <div class="metric-value">1287</div>
                <div class="metric-label">Movies Analyzed</div>
            </div>
            <div class="metric">
                <div class="metric-value">94.3%</div>
                <div class="metric-label">Match Accuracy</div>
            </div>
            <div class="metric">
                <div class="metric-value">3.2ms</div>
                <div class="metric-label">Avg. Response Time</div>
            </div>
        </div>

        <div class="search-container">
            <div class="search-box">
                <input type="text" id="searchInput" placeholder="Search for movies...">
                <button id="searchBtn">Search</button>
            </div>
        </div>

        <div class="filters">
            <div class="filter active" data-filter="all">All</div>
            <div class="filter" data-filter="action">Action</div>
            <div class="filter" data-filter="comedy">Comedy</div>
            <div class="filter" data-filter="drama">Drama</div>
            <div class="filter" data-filter="sci-fi">Sci-Fi</div>
            <div class="filter" data-filter="horror">Horror</div>
        </div>

        <div class="movie-grid" id="movieContainer">
            <!-- Movie cards will be dynamically inserted here -->
        </div>

        <div class="terminal-output" id="terminalOutput">
            <div class="log-entry">Initializing recommendation engine...</div>
            <div class="log-entry">Loading movie database...</div>
            <div class="log-entry">Database connection established</div>
            <div class="log-entry">Analyzing user preferences...</div>
            <div class="log-entry">Ready to provide recommendations</div>
        </div>
    </main>

    <script>
        // Movie database
        const movies = [
            {
                id: 1,
                title: "The Matrix Reloaded",
                genre: ["action", "sci-fi"],
                year: 2003,
                rating: 7.2,
                description: "Neo and the rebel leaders estimate they have 72 hours until Zion falls under siege to the Machine army. Neo must decide how he can save Trinity from a dark fate in his dreams.",
                imageAlt: "Neo in black trench coat dodging bullets in a futuristic city"
            },
            {
                id: 2,
                title: "Pulp Fiction",
                genre: ["drama", "crime"],
                year: 1994,
                rating: 8.9,
                description: "The lives of two mob hitmen, a boxer, a gangster's wife, and a pair of diner bandits intertwine in four tales of violence and redemption.",
                imageAlt: "Uma Thurman and John Travolta dancing in a 50s style diner"
            },
            {
```

```
        id: 3,
        title: "Inception",
        genre: ["action", "sci-fi", "thriller"],
        year: 2010,
        rating: 8.8,
        description: "A thief who steals corporate secrets through the use of dream-sharing technology is given the inverse task of planting an idea into the mind of a C.E.O.",
        imageAlt: "Dreamlike cityscape folding in on itself with Leonardo DiCaprio standing in foreground"
      },
      {
        id: 4,
        title: "Parasite",
        genre: ["drama", "thriller"],
        year: 2019,
        rating: 8.6,
        description: "Greed and class discrimination threaten the newly formed symbiotic relationship between the wealthy Park family and the destitute Kim clan.",
        imageAlt: "Korean family looking up from a semi-basement apartment window"
      },
      {
        id: 5,
        title: "The Grand Budapest Hotel",
        genre: ["comedy", "drama"],
        year: 2014,
        rating: 8.1,
        description: "The adventures of Gustave H, a legendary concierge at a famous European hotel between the wars, and Zero Moustafa, the lobby boy who becomes his most trusted friend.",
        imageAlt: "Pink hotel facade with Wes Anderson symmetrical composition style"
      },
      {
        id: 6,
        title: "Get Out",
        genre: ["horror", "thriller"],
        year: 2017,
        rating: 7.7,
        description: "A young African-American visits his white girlfriend's parents for the weekend, where his simmering uneasiness about their reception of him eventually reaches a boiling point.",
        imageAlt: "Daniel Kaluuya looking anxious in an upscale suburban home"
      }
    ];

    // DOM elements
    const movieContainer = document.getElementById('movieContainer');
    const searchInput = document.getElementById('searchInput');
    const searchBtn = document.getElementById('searchBtn');
    const filters = document.querySelectorAll('.filter');
    const terminalOutput = document.getElementById('terminalOutput');

    // Initialize the app
    function init() {
      renderMovies(movies);
      addEventListeners();
      logToTerminal("System initialized successfully");
    }
```

```
// Render movies to the DOM
function renderMovies(moviesToRender) {
  movieContainer.innerHTML = '';

  moviesToRender.forEach(movie => {
    const status = getStatusRating(movie.rating);

    const movieCard = document.createElement('div');
    movieCard.className = 'movie-card';
    movieCard.innerHTML = `
      <img src="https://storage.googleapis.com/workspace-0f70711f-8b4e-4d94-86f1-2a93ccde5887/image/d7387753-12ac-4ea0-9906-1ec9749c6a3e.png'+')}" class="movie-image" alt="${movie.imageAlt}" />
      <span class="status-badge status-${status.class}">${status.text}</span>
      <div class="movie-content">
        <h3 class="movie-title">${movie.title}</h3>
        <div class="movie-meta">
          <span>${movie.year}</span>
          <span class="movie-rating">
            <span class="rating-star">★</span>
            ${movie.rating}
          </span>
        </div>
        <p class="movie-description">${movie.description}</p>
        <div class="movie-actions">
          <button class="btn btn-primary">Add to Queue</button>
          <button class="btn btn-secondary">Details</button>
        </div>
      </div>
    `;

    movieContainer.appendChild(movieCard);
  });
}

// Get status rating based on score
function getStatusRating(rating) {
  if (rating >= 8.0) return { class: 'high', text: 'Highly Recommended' };
  if (rating >= 6.5) return { class: 'medium', text: 'Recommended' };
  return { class: 'low', text: 'Fair' };
}

// Filter movies by genre
function filterMovies(genre) {
  if (genre === 'all') {
    renderMovies(movies);
    logToTerminal("Displaying all movies");
    return;
  }

  const filteredMovies = movies.filter(movie =>
    movie.genre.includes(genre)
  );

  renderMovies(filteredMovies);
  logToTerminal(`Filtered movies by genre: ${genre}`);
}
```

```
    // Search movies
    function searchMovies(query) {
      const normalizedQuery = query.toLowerCase();

      const filteredMovies = movies.filter(movie =>
        movie.title.toLowerCase().includes(normalizedQuery) ||
        movie.description.toLowerCase().includes(normalizedQuery)
      );

      renderMovies(filteredMovies);
      logToTerminal(`Search results for: ${query}`);
    }

    // Add event listeners
    function addEventListeners() {
      // Search button
      searchBtn.addEventListener('click', () => {
        const query = searchInput.value.trim();
        if (query) {
          searchMovies(query);
        }
      });

      // Enter key in search input
      searchInput.addEventListener('keypress', (e) => {
        if (e.key === 'Enter') {
          const query = searchInput.value.trim();
          if (query) {
            searchMovies(query);
          }
        }
      });

      // Filter buttons
      filters.forEach(filter => {
        filter.addEventListener('click', () => {
          filters.forEach(f => f.classList.remove('active'));
          filter.classList.add('active');
          filterMovies(filter.dataset.filter);
        });
      });
    }

    // Log to terminal
    function logToTerminal(message) {
      const logEntry = document.createElement('div');
      logEntry.className = 'log-entry';
      logEntry.textContent = message;
      terminalOutput.appendChild(logEntry);
      terminalOutput.scrollTop = terminalOutput.scrollHeight;
    }

    // Initialize the app
    document.addEventListener('DOMContentLoaded', init);
  </script>
</body>
```

</html>

```
```