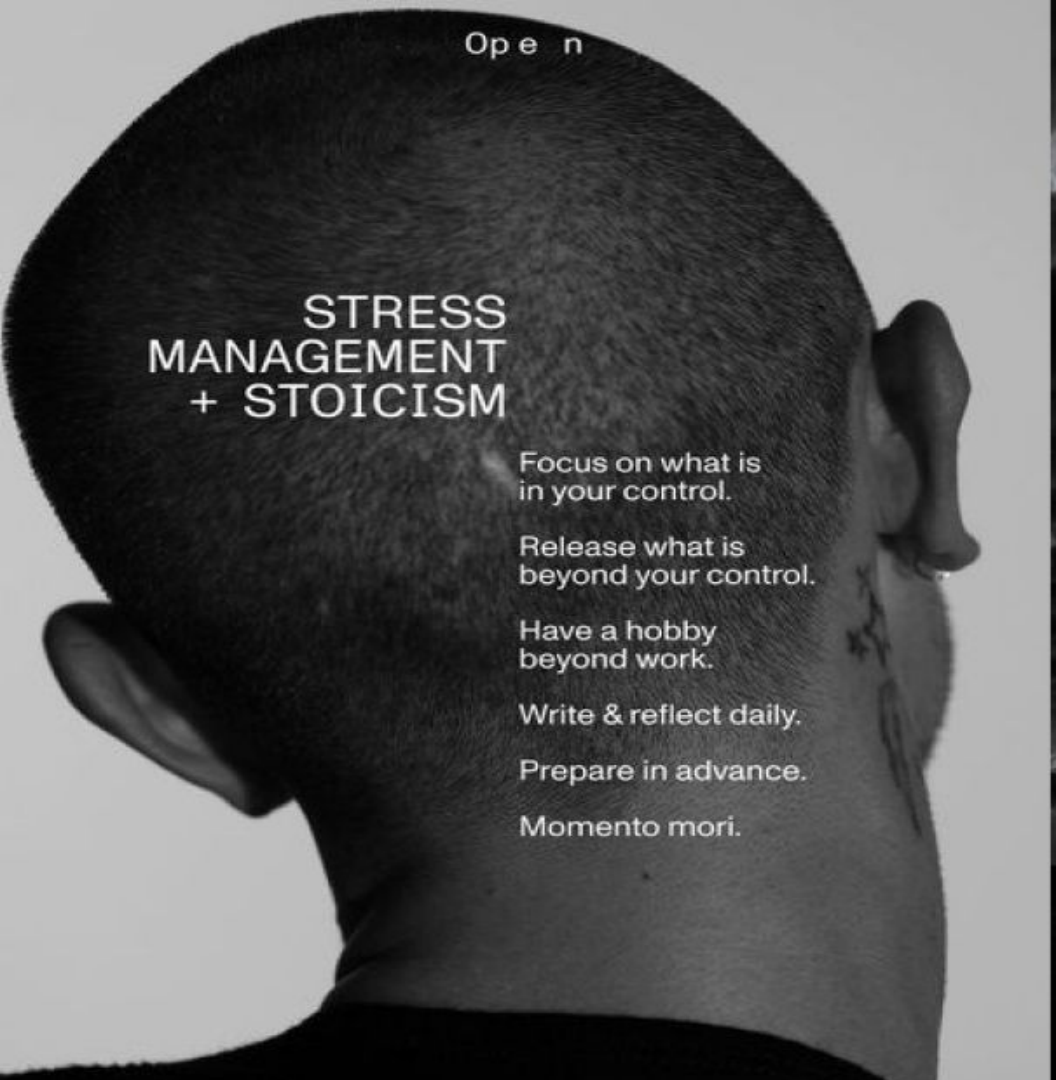

Op e n
STRESS
MANAGEMENT
+ STOICISM
Focus on what is
in your control.
Release what is
beyond your control.
Have a hobby
beyond work.
Write & reflect daily.
Prepare in advance.
Momento mori.


SyntaxSyndicates

INTERACTIVE DEVELOPMENT

DV100 , Term 4 , Final Website & Presentation

By :

John Dippenaar 231268
Zané Olckers 231240
Curtis Alben 241190
Hendrik Odendaal 221140

CSS & STYLING

By John Dippenaar

CSS(CSS Styling Overview)

CSS Styling Overview

- **Themes**: Dark mode background, subtle text shadows, and focus on readability
- **Font and Colours**:
    - `font-family`: Arial, sans-serif for consistency and readability
    - Colours are chosen for high contrast on a dark background

```css
body {
    font-family: 'Arial', sans-serif;
    color: #e0e0e0;
    background-color: #000000 !important;
    margin: 0;
    padding: 0;
    overflow-x: hidden;
}
```

CSS (Body and General Styling)

- **Body**:
  - Dark background
  - Light text colour for contrast
- **Headers and Paragraphs**:
  - Light text with a glowing effect
  - Unified colour for headers, paragraphs, and anchor elements
  - Emphasis on clean, legible typography

```css
h1, h2, p {
    color: #ffffff;
    text-shadow: 0 0 10px rgba(102, 249, 255, 0.5);
}

a {
    color: #ffffff;
    text-decoration: underline;
    transition: color 0.3s;
}

h3{
    color: #ffffff !important;
}

a:hover {
    color: #B2DDF7;
}
```

CSS (Link and Hover Effects)

- **Links**:
  - White colour with underline by default
  - Subtle colour transition on hover
  - Active state for `nav-links` enhances navigation visibility

```css
a:hover {
    color: #B2DDF7;
}

.nav-links a.active {
    text-decoration: underline;
    color: #B2DDF7;
}
```

CSS (Header Design)

- **Header Layout**:
  - Flexbox alignment for centering content and spacing between items
  - Fixed background colour with a border at the bottom for separation
- **Logo**:
  - Large font size (40px) and bright glow effect for visibility
- **Navigation Menu**:
  - Flexbox for horizontal layout, spaced links
  - Simple, clear, and accessible menu design

```css
header {
    display: flex;
    align-items: center;
    justify-content: space-between;
    padding: 20px;
    background-color: #181818;
    border-bottom: 1px solid #222;
    max-width: 100%;
    box-sizing: border-box;
}
```

```css
header .logo h1 {
    font-size: 40px;
    color: #ffffff;
    text-shadow: 0 0 15px rgba(255, 255, 255, 0.5);
    margin: 0;
}
```

```css
nav ul {
    display: flex;
    gap: 20px;
    list-style-type: none;
    margin: 0;
    padding: 0;
}

nav ul li a {
    color: #ffffff;
    font-size: 20px;
    text-decoration: none;
}
```

CSS (Search Bar Styling)

- **Structure**:
  - Positioned within a container for control over layout and styling
  - Rounded input field with no border, ensuring a minimalist aesthetic
- **Icon Positioning**:
  - Search icon aligns within the input box using absolute positioning, aiding usability

```css
.search-bar {
    position: relative;
    max-width: 100%;
}

.search-bar input {
    padding: 10px 20px;
    border-radius: 20px;
    border: none;
    background: #ffffff;
    color: #000000;
    outline: none;
    width: 100%;
    box-sizing: border-box;
}

.search-bar img {
    position: absolute;
    right: 10px;
    top: 50%;
    transform: translateY(-50%);
    width: 20px;
}
```

CSS (Intro and Movies Section)

- **Intro Section**:
  - Centre-aligned text for an inviting introduction
  - Margin adjustments ensure it stands out from surrounding elements
- **Movies Section**:
  - Rounded corners and subtle shadow effects (`box-shadow`) add depth
  - Background colour differentiates it from the main background

```css
.intro {
    text-align: center;
    margin: 40px 20px;
    font-size: 19px;
    color: #ffffff;
}

.movies-section {
    margin: 40px 20px;
    padding: 20px;
    background: #1a1a1a;
    border-radius: 8px;
    box-shadow: 0 0 15px rgba(0, 0, 0, 0.5);
    max-width: 100%;
    box-sizing: border-box;
}

.movies-section h2 {
    margin-bottom: 20px;
    color: #ffffff;
    text-shadow: 0 0 10px rgba(255, 255, 255, 0.3);
}
```

CSS (Carousel Design)

- **Carousel**:
  - Carousel images and items are designed with a glowing shadow effect
  - Rounded corners for a polished look and a bright shadow effect around images
- **Controls**:
  - Background-colour transitions and rounded shapes for prev/next controls
  - Icon colours inverted for contrast against the dark background

```css
.carousel-inner {
    height: 900px;
}

.carousel-image {
    height: 300px;
    object-fit: cover;
}

.carousel-item img {
    border-radius: 8px;
    box-shadow: 0 0 20px rgba(255, 255, 255, 0.3);
}

.carousel-control-prev-icon,
.carousel-control-next-icon {
    filter: brightness(0) invert(1);
    background-size: 50%;
}

.carousel-control-prev,
.carousel-control-next {
    background-color: rgba(0, 0, 0, 0.5);
    border-radius: 50%;
}

button.carousel-control-prev,
button.carousel-control-next {
    transition: background-color 0.3s;
}
```

CSS (Footer Layout)

- **Structure**:
  - Dark grey background and padding for separation from the main content
  - Flexbox layout divides content into sections for easier navigation
- **Link Styling**:
  - Colour adjustments ensure links stand out while fitting the design
  - Social media icons in a horizontal layout add visual interest

```css
.footer {
    background-color: #999;
    padding: 50px 20px 0 20px;
    margin-bottom: 0;
    max-width: 100%;
    box-sizing: border-box;
}

.footerContent {
    display: flex;
    justify-content: space-between;
    align-items: center;
    max-width: 100%;
    box-sizing: border-box;
}
```

CSS (Responsive Design)

- **Mobile Adjustments**:
  - Flex direction changes and font size adjustments for smaller screens
  - Padding reductions and repositioning for navigation and footer
  - Responsive carousel height to adapt to screen sizes, optimising mobile experience

```css
@media (max-width: 468px) {
    header {
        flex-direction: column;
        padding: 20px;
    }

    header .logo h1 {
        font-size: 28px;
        text-align: center;
    }

    nav ul {
        flex-direction: column;
        align-items: center;
        gap: 10px;
    }

    nav ul li a {
        font-size: 18px;
    }

    .search-bar {
        margin-top: 20px;
        width: 100%;
    }

    .intro {
        margin: 20px 10px;
    }

    .movies-section {
        margin: 20px 10px;
        padding: 15px;
    }

    .carousel-inner {
        height: auto;
    }


    .footerContent {
        flex-direction: column;
        text-align: center;
    }

    .linkSection {
        padding-left: 0;
    }

    .footerBottom {
        flex-direction: column;
        text-align: center;
    }
}
```

HTML

By Hendrik Odendaal

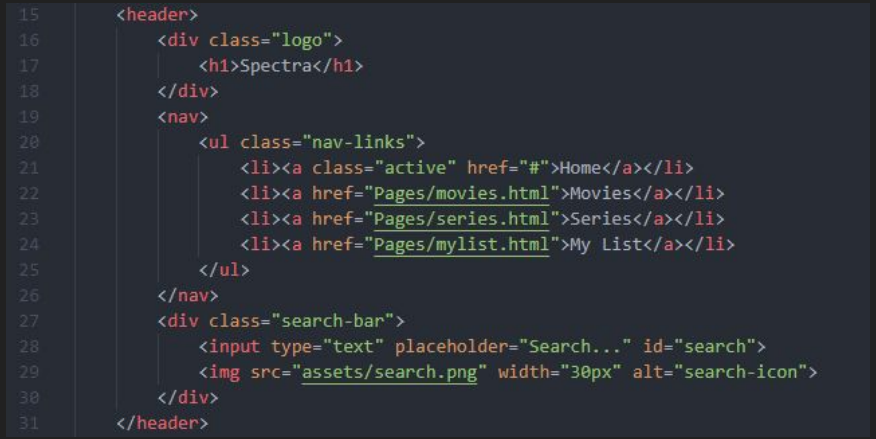
HTML

Each page of the Spectra site follows a standardized HTML structure. We start with the <!DOCTYPE html> declaration to ensure proper rendering across all browsers. The document opens with the <html> tag and includes essential metadata in the <head>, such as linking to external CSS and JavaScript resources from Bootstrap to keep the site responsive and interactive.

```
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Spectra - Series</title>
<link rel="stylesheet" href="../CSS/main.css">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>
```

Header Section

In the header, we have our navigation links, and a search bar. We have links for easy navigation across pages—like Home, Movies, Series, and My List—to keep the user experience consistent. This setup makes the header a core navigational element for our users.



```
<header>
    <div class="logo">
        <h1>Spectra</h1>
    </div>
    <nav>
        <ul class="nav-links">
            <li><a class="active" href="#">Home</a></li>
            <li><a href="Pages/movies.html">Movies</a></li>
            <li><a href="Pages/series.html">Series</a></li>
            <li><a href="Pages/mylist.html">My List</a></li>
        </ul>
    </nav>
    <div class="search-bar">
        <input type="text" placeholder="Search..." id="search">
        <img src="assets/search.png" width="30px" alt="search-icon">
    </div>
</header>
```

Movies and Series Pages

Moving to the Movies and Series pages, each of these are designed to display a collection of content using a section called movies-section. The layout includes rows that will hold individual movie or series cards. JavaScript will populate this section dynamically, allowing us to update the content without altering the HTML. This approach keeps the code modular and easily expandable.

```
<section class="intro">
    <h2>Explore Our Movie Collection</h2>

</section>

<!-- Movies Section -->
<section class="movies-section">
    <h2>All Movies</h2>
    <div class="movies-row" id="movies-list">
        <!-- Movie Cards will be added here dynamically -->

    </div>
</section>
```

My List Page (Zané Olckers)

The My List page consolidates all favorite movies and series in one place. This layout is user-friendly, with separate sections for series and movies, so users can quickly find what they've saved. Again, JavaScript will dynamically load this content, making it flexible and personalised for each user.

```
<!-- Intro Section -->
<section class="intro">
    <h2>My List</h2>
    <p>SEE ALL YOUR FAVORITE MOVIES AND SERIES IN ONE PLACE</p>
</section>

<!-- Series Section -->
<section class="movies-section">
    <h2>All Series</h2>
    <div class="movies-row" id="series-list">
        <!-- Series Cards will be added here dynamically -->
    </div>
</section>

<section class="movies-section">
    <h2>All Movies</h2>
    <div class="movies-row" id="movies-list">
        <!-- Series Cards will be added here dynamically -->
    </div>
</section>
```

Sign-In and Sign-Up Pages

Our Sign-In and Sign-Up pages are built using form elements. Both pages feature required fields to ensure essential information is collected. For example, Sign-In requires a username and password, while Sign-Up requires additional fields like email. There are also links for users to easily switch between Sign-In and Sign-Up. These forms are crucial for user authentication and data collection.

```
<div class="container">
    <h1>Sign Up</h1>
    <form id="signup-form">
        <label for="username">Username</label>
        <input type="text" id="username" name="username" required>

        <label for="email">Email</label>
        <input type="email" id="email" name="email" required>

        <label for="password">Password</label>
        <input type="password" id="password" name="password" required>

        <button type="submit">Sign Up</button>
        <div class="form-footer">
            <p>Already have an account? <a href="signin.html">Sign In</a></p>
        </div>
    </form>
</div>
```

```
<div class="container">
    <h1>Sign In</h1>
    <form id="signin-form">
        <label for="signin-username">Username</label>
        <input type="text" id="signin-username" name="username" required>

        <label for="signin-password">Password</label>
        <input type="password" id="signin-password" name="password" required>

        <button type="submit">Sign In</button>
        <div class="form-footer">
            <p>Don't have an account? <a href="signup.html">Sign Up</a></p>
        </div>
    </form>
</div>
```

Footer Section (Zané Olckers)

Each page concludes with a footer section, featuring company links and social media icons to connect users with related content. The footer also includes a copyright notice and legal links to the Terms and Conditions and Privacy Policy. This adds a professional touch to the site and serves as a resource hub for users.

```
<footer class="footer">
    <div class="footerContent">
        <div class="logo">
            <h1>Spectra</h1>
        </div>
        <div class="linkSection">
          <h4 class="linkHeading">Company</h4>
          <ul class="links">
            <li><a href="../index.html">Home</a></li>
            <li><a href="../Pages/movies.html">Movies</a></li>
            <li><a href="../Pages/mylist.html">Series</a></li>
            <li><a href="../Pages/mylist.html">My list</a></li>
          </ul>
        </div>
        <div class="socialMediaSection">
          <h4 class="socialMediaHeading">Social Media Links</h4>
          <div class="socialMediaIcons">
            <a href="https://www.instagram.com/openwindowinstitute/?hl=en"><img src="../assets/instagram-white-icon.png"></a>
            <a href="https://twitter.com/open_window_?ref_src=twsrc%5Egoogle%7Ctwcamp%5Eserp%7Ctwgr%5Eauthor"><img src="../assets/x-social-media-white-icon.png"></a>
            <a href="https://www.tiktok.com/discover/Open-window"><img src="../assets/tiktok-round-white-icon.png"></a>
            <a href="https://www.facebook.com/theopenwindow/"><img src="../assets/facebook-app-round-white-icon.png"></a>
          </div>
        </div>
    </div>
    <hr>
    <div class="footerBottom">
      <p class="rightsReserved">All rights reserved @ Open Window 2024</p>
      <div class="legalLinks">
        <a href="#">Terms and Conditions</a>
        <a href="#">Privacy Policy</a>
      </div>
    </div>
</footer>
```

Integration with CSS and JavaScript

While I focused on structuring the HTML, my teammates are handling the styling with CSS and adding functionality with JavaScript. For instance, our CSS will enhance the visual appeal of the navigation, cards, and forms, while JavaScript will handle dynamic features like the search bar, loading content into sections, and interactive navigation elements. Together, our contributions create a polished and user-friendly experience.

```
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Spectra - Series</title>
<link rel="stylesheet" href="../CSS/main.css">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet">
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"></script>
```

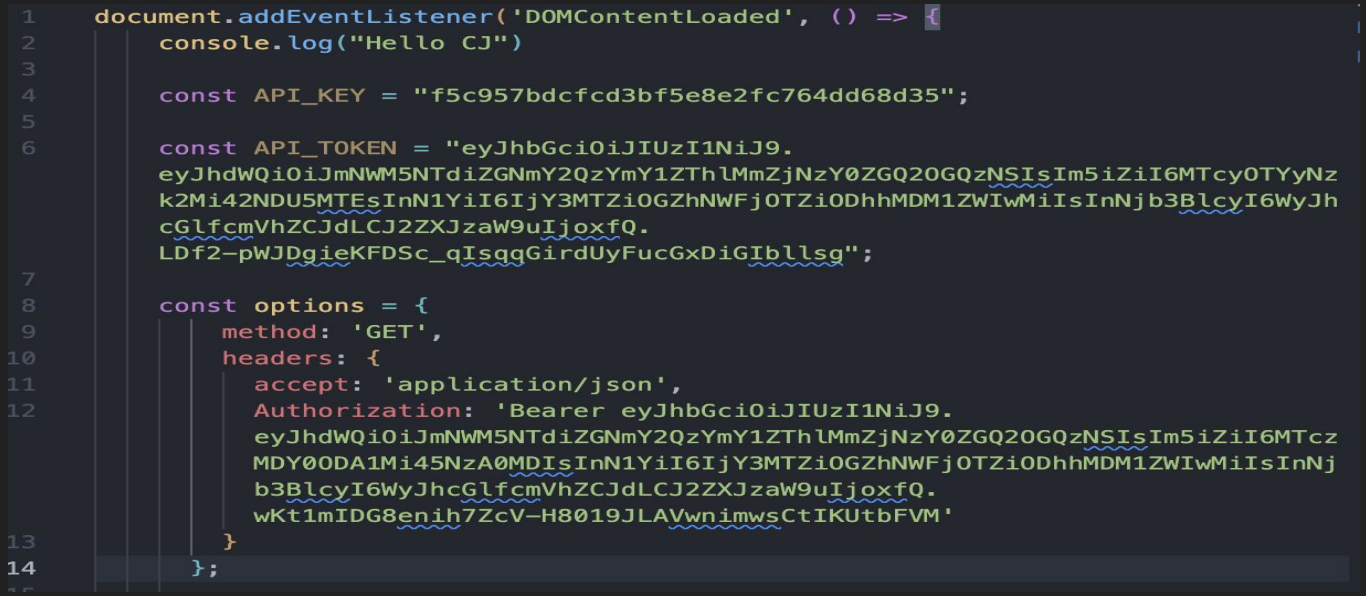
JAVASCRIPT

By Curtis J Alben

```
<!--New Movie Sections -->
<section class="movies-section">
    <h2>New</h2>
    <div class="carousel-container">
    <div class="carousel slide" id="new-movies" data-bs-ride="carousel">
        <div class="carousel-inner">

            <!-- <div class="carousel-item active">
                <img src="assets/big-data-meme.jpg" class="d-block w-100" alt="Image 1" width="50%">
            </div>
            <div class="carousel-item">
                <img src="assets/big-data-meme.jpg" class="d-block w-100" alt="Image 2">
            </div>
            <div class="carousel-item">
                <img src="assets/big-data-meme.jpg" class="d-block w-100" alt="Image 3">
            </div> -->
        </div>


        <button class="carousel-control-prev" type="button" data-bs-target="#new-movies" data-bs-slide="prev">
            <span class="carousel-control-prev-icon" aria-hidden="true"></span>
            <span class="visually-hidden">Previous</span>
        </button>
        <button class="carousel-control-next" type="button" data-bs-target="#new-movies" data-bs-slide="next">
            <span class="carousel-control-next-icon" aria-hidden="true"></span>
            <span class="visually-hidden">Next</span>
        </button>
    </div>
</div>

```

HTML :

Added a section including a carousel and two buttons for scrolling to the index page



```
document.addEventListener('DOMContentLoaded', () => {
    console.log("Hello CJ")

    const API_KEY = "f5c957bdcfcd3bf5e8e2fc764dd68d35";

    const API_TOKEN = "eyJhbGciOiJIUzI1NiJ9.
    eyJhdWQiOiJmNWM5NTdiZGNmY2QzYmY1ZThlMmZjNzY0ZGQ2OGQzNSIsIm5iZiI6MTcyOTYyNz
    k2Mi42NDU5MTEsInN1YiI6IjY3MTZiOGZhNWFjOTZiODhhMDM1ZWIwMiIsInNjb3BlcyI6WyJh
    cGlfcmVhZCJdLCJ2ZXJzaW9uIjoxfQ.
    LDf2-pWJDgieKFDSc_qIsqgGirdUyFucGxDiGIbllsg";

    const options = {
      method: 'GET',
      headers: {
        accept: 'application/json',
        Authorization: 'Bearer eyJhbGciOiJIUzI1NiJ9.
        eyJhdWQiOiJmNWM5NTdiZGNmY2QzYmY1ZThlMmZjNzY0ZGQ2OGQzNSIsIm5iZiI6MTcz
        MDY0ODA1Mi45NzA0MDIsInN1YiI6IjY3MTZiOGZhNWFjOTZiODhhMDM1ZWIwMiIsInNj
        b3BlcyI6WyJhcGlfcmVhZCJdLCJ2ZXJzaW9uIjoxfQ.
        wKt1mIDG8enih7ZcV-H8019JLAVwnimwsCtIKUtbFVM'
      }
    };
```

API Key & API Token :

Above you will see the API Key and API token that was used to retrieve the data from The Movie Database or “TMDB” , which was our chosen API

```
    // fetch movie data
    fetch('https://api.themoviedb.org/3/trending/movie/day?language=en-US', options)
    .then(res => res.json())
    .then(res => {
      const movies = res.results.slice(0, 5); // Accessing the first 5 movies
      const moviesList = document.getElementById('movies-list'); // Get the container element

      movies.forEach(movie => {
        // Create the movie card element
        const movieCard = document.createElement('div');
        movieCard.classList.add('movie-card');

        // Movie poster image
        const moviePoster = document.createElement('img');
        moviePoster.src = `https://image.tmdb.org/t/p/w500${movie.poster_path}`;
        moviePoster.alt = `${movie.title} poster`;
        movieCard.appendChild(moviePoster);

        // Movie title
        const movieTitle = document.createElement('h3');
        movieTitle.textContent = movie.title;
        movieCard.appendChild(movieTitle);

        // Movie release date
        const movieReleaseDate = document.createElement('p');
        movieReleaseDate.textContent = `Release Date: ${movie.release_date}`;
        movieCard.appendChild(movieReleaseDate);

        // Append the movie card to the movies list
        moviesList.appendChild(movieCard);
      });
    })
    .catch(err => console.error(err));
});

```

My List Page :

Added fetch request to filter specific data in the MyList Page , used forEach loop ,

created movie card element , included different data to include namely image , movie title , release date and movie card.

My List Page :

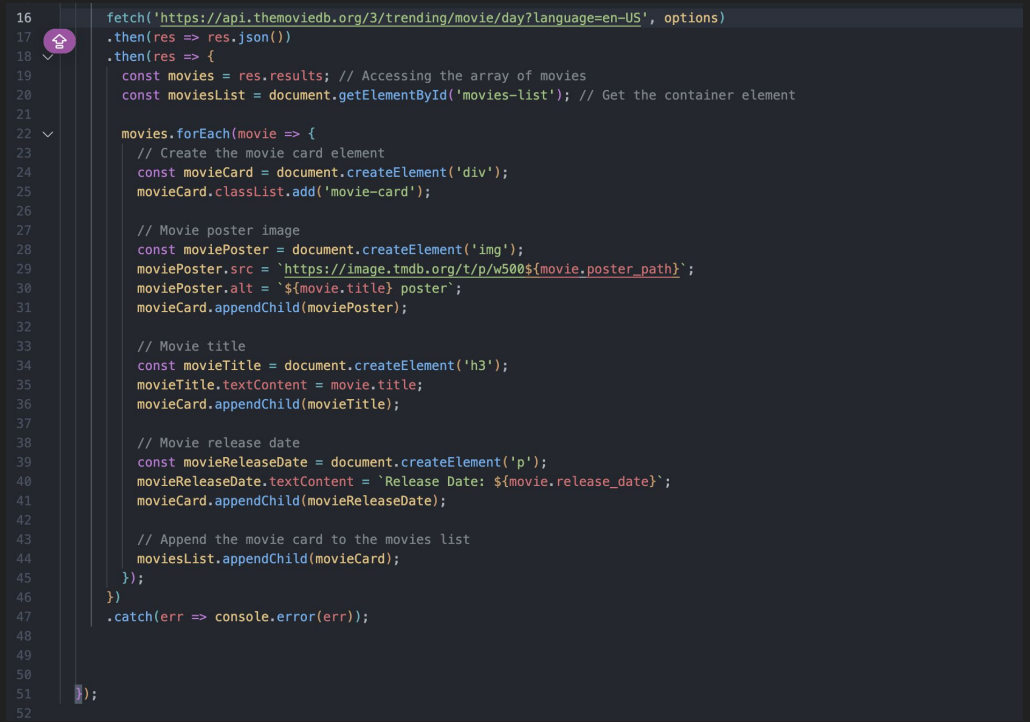
Fetch request retrieving data for movies from API .

```
51      // fetch movie data
52      fetch('https://api.themoviedb.org/3/trending/movie/day?language=en-US', options)
53      .then(res => res.json())
54      .then(res => {
55        const movies = res.results.slice(0, 5); // Accessing the first 5 movies
56        const moviesList = document.getElementById('movies-list'); // Get the container element
57
58        movies.forEach(movie => {
59          // Create the movie card element
60          const movieCard = document.createElement('div');
61          movieCard.classList.add('movie-card');
62
63          // Movie poster image
64          const moviePoster = document.createElement('img');
65          moviePoster.src = `https://image.tmdb.org/t/p/w500${movie.poster_path}`;
66          moviePoster.alt = `${movie.title} poster`;
67          movieCard.appendChild(moviePoster);
68
69          // Movie title
70          const movieTitle = document.createElement('h3');
71          movieTitle.textContent = movie.title;
72          movieCard.appendChild(movieTitle);
73
74          // Movie release date
75          const movieReleaseDate = document.createElement('p');
76          movieReleaseDate.textContent = `Release Date: ${movie.release_date}`;
77          movieCard.appendChild(movieReleaseDate);
78
79          // Append the movie card to the movies list
80          moviesList.appendChild(movieCard);
81        });
82      })
83      .catch(err => console.error(err));
84  });
85
```

Movies Page :

Added fetch request to filter specific data in the Movies Page , used forEach loop ,

created movie card element , included different data to include namely image , movie title , release date and movie card.



```
fetch('https://api.themoviedb.org/3/trending/movie/day?language=en-US', options)
.then(res => res.json())
.then(res => {
  const movies = res.results; // Accessing the array of movies
  const moviesList = document.getElementById('movies-list'); // Get the container element

  movies.forEach(movie => {
    // Create the movie card element
    const movieCard = document.createElement('div');
    movieCard.classList.add('movie-card');

    // Movie poster image
    const moviePoster = document.createElement('img');
    moviePoster.src = `https://image.tmdb.org/t/p/w500${movie.poster_path}`;
    moviePoster.alt = `${movie.title} poster`;
    movieCard.appendChild(moviePoster);

    // Movie title
    const movieTitle = document.createElement('h3');
    movieTitle.textContent = movie.title;
    movieCard.appendChild(movieTitle);

    // Movie release date
    const movieReleaseDate = document.createElement('p');
    movieReleaseDate.textContent = `Release Date: ${movie.release_date}`;
    movieCard.appendChild(movieReleaseDate);

    // Append the movie card to the movies list
    moviesList.appendChild(movieCard);
  });
})
.catch(err => console.error(err));



});
```

SUPPORT

By Zané Olckers

```css
.footer {
    background-color: #999;
    padding: 50px 20px 0 20px;
    margin-bottom: 0;
    max-width: 100%;
    box-sizing: border-box;
}

.footerContent {
    display: flex;
    justify-content: space-between;
    align-items: center;
    max-width: 100%;
    box-sizing: border-box;
}

.linkSection {
    padding-left: 100px;
}

.links {
    list-style-type: none;
    padding: 0;
}

.links li {
    margin-bottom: 5px;
}

.links a,
.rightsReserved,
.legalLinks a {
    text-decoration: none;
    color: #333;
}

.socialMediaIcons img {
    width: 40px;
    margin-right: 10px;
}
```

CSS for footer:

After I did Html was made for the footer, I did some basic CSS to ensure that the footer displays at a good size and that everything is aligned properly.

```css
.footerBottom {
    margin-top: 20px;
    display: flex;
    align-items: center;
    justify-content: space-between;
    max-width: 100%;
    box-sizing: border-box;
}
```

```html
<button id="back-to-top" title="Go to top">Top</button>
```

JQuery Function :

Started by adding a back to top button in the html, I then added CSS to make sure it is displaying at the right place and the right size.

I then implemented the necessary JS to make it functional.

```css
#back-to-top {
    display: none; /* Initially hidden */
    position: fixed;
    bottom: 40px;
    right: 40px;
    z-index: 100;
    background-color: #2c3e50;
    color: white;
    border: none;
    padding: 10px 20px;
    border-radius: 5px;
    cursor: pointer;
    font-size: 16px;
    transition: background-color 0.3s ease;
}

#back-to-top:hover {
    background-color: #B2DDF7;
    color: #000000;
}
```

```javascript
$(document).ready(function() {
    // Show or hide the button when scrolling
    $(window).scroll(function() {
        if ($(this).scrollTop() > 200) {
            $('#back-to-top').fadeIn();
        } else {
            $('#back-to-top').fadeOut();
        }
    });

    // Smooth scroll to top when button is clicked
    $('#back-to-top').click(function() {
        $('html, body').animate({scrollTop: 0}, 50);
        return false;
    });
});
```

To Explain some of the Javascript:

$(window).scroll(function() {...}): This tracks the user when they scroll, when the scroll position passes 200 px, the button fades in otherwise it hides it.

$('html, body').animate({scrollTop: 0}, 50); : It ensures that it smoothly scrolls back up to the top of the page once the button is clicked.

$('#back-to-top').fadeIn(): It make sure the button fades in

```html
<div class="search-bar">
    <input type="text" placeholder="Search..." id="search">
    <button id="search-button">
        <img src="assets/search.png" width="30px" alt="search-icon">
    </button>
</div>
```

```css
.search-bar {
    position: relative;
    max-width: 100%;
}

.search-bar input {
    padding: 10px 20px;
    border: none;
    outline: none;
    border-radius: 20px;
    background: #ffffff;
    color: #000000;
    width: 100%;
    box-sizing: border-box;
}

.search-bar button {
    position: absolute;
    right: 10px;
    top: 50%;
    transform: translateY(-50%);
    background: none;
    border: none;
    padding: 0;
    cursor: pointer;
}

.search-bar img {
    position: absolute;
    right: 10px;
    top: 50%;
    transform: translateY(-50%);
    width: 20px;
}
```

```javascript
$(document).ready(function() {
    $('#search-button').click(function(event) {
        event.preventDefault(); // Prevents form submission if inside a form

        // Get the search input value
        const query = $('#search').val().trim();

        if (query) {
            // Perform the search action (redirect to search results page with query)
            window.location.href = `search-results.html?query=${encodeURIComponent(query)}`;
        } else {
            alert('Please enter a search term.');
        }
    });
});
```

Functional Search Bar using JQuery:

Started by changing the HTML a bit, I then added and changed CSS to make sure it is displaying at the right place and the right size.

I then implemented the necessary JS to make it functional.

To Explain some of the Javascript:

event.preventDefault() : This stops the default form submission behaviour and ensures that the custom search function works properly

const query = $('#search').val().trim(); : Retrieves and trims the input value to remove any unnecessary whitespace