**Project: Movie Search App**

**Project Overview**

The Movie Search App allows users to search for movies and view detailed information about each movie, including title, release date, genre, plot summary, and poster. The app will utilize the [OMDb API (Open Movie Database)](http://www.omdbapi.com/) for retrieving movie data.

**Features**

1. **Search Movies:**
   * Users can enter a movie title into a search bar.
   * The app fetches a list of movies matching the search query from the OMDb API.
2. **View Movie Details:**
   * Users can click on a movie from the search results to view detailed information.
   * Detailed information includes the title, release date, genre, plot summary, and poster image.
3. **Error Handling:**
   * Handle errors for network issues or invalid search queries.
   * Provide user-friendly error messages when something goes wrong.
4. **Loading Indicators:**
   * Show a loading indicator while fetching data from the API to improve user experience.

**Technologies Used**

* **HTML:** For the structure of the application.
* **CSS:** For basic styling and layout (optional for students).
* **JavaScript:** For fetching data from the API, updating the DOM, and handling user interactions.
* **OMDb API:** For retrieving movie data.

**Implementation Steps**

1. **Set Up the Project:**
   * Create an HTML file for the app's structure.
   * Create a JavaScript file for handling the logic.
   * Optionally, create a CSS file for basic styling.
2. **Create the User Interface:**
   * Add a search bar where users can enter a movie title.
   * Add a section to display search results.
   * Add a section to display detailed movie information.
3. **Fetch Data from the OMDb API:**
   * Sign up for an API key from the OMDb API.
   * Use fetch to make asynchronous requests to the OMDb API.
   * Handle the response using promises or async/await.
4. **Display Search Results:**
   * Parse the API response to extract movie data.
   * Update the DOM to display the list of movies matching the search query.
5. **Display Movie Details:**
   * Add event listeners to each movie in the search results.
   * Fetch detailed information about the selected movie from the API.
   * Update the DOM to display detailed movie information.
6. **Handle Errors:**
   * Implement error handling for network issues and invalid API responses.
   * Display user-friendly error messages in the UI.
7. **Loading Indicators:**
   * Add a loading indicator to show while data is being fetched.
   * Hide the loading indicator once data is loaded.