Project Charter

1. **Project Title**

Movie Search Hub

2. **Project Objectives**

The objectives of the project are:

1. Implement a sorting function(s) to sort movie data by
    a. Release date
    b. Alphabetically
2. Implement a movie search function to find movies by
    a. Title
    b. Ratings
    c. Genres
3. Implement a user interface to display movie details.

We essentially want to provide the end user with a functioning movie database, that is versatile enough to allow them to search for their title with different pieces of information. The sorting function is to implement a way for the information to be presented to the user in a specific way that they feel would work best for them.

3. **Project Scope**

In-Scope:

1. Implementing Searching Functionality:
    - Enabling users to search for movies by title, genre, year, or other attributes.

2. Implementing Sorting Functions:
    - Allowing sorting of search results based on various criteria, such as:
    - Release Date (Newest to Oldest, Oldest to Newest).
    - Alphabetical Order (A-Z, Z-A).
    - Rating (Highest to Lowest, Lowest to Highest).

3. Implementing Display Features:
    - Displaying detailed movie information, such as:
    - Title, Poster, Synopsis, Cast, Director, and Release Date.
    - Presenting multiple views (e.g., grid view, list view) based on user preference.
    - Showing movie trailers, images, and links to streaming services (if available).
    - Providing a movie details page for an in-depth view of a selected movie.

4. User Interaction and Navigation:
    - Creating a responsive user interface for seamless navigation.
    - Enabling interactive elements like "View More Details," "Back to Search Results," etc.

Out-of-Scope:

1. Maintaining the Movie Database:
    - No internal movie database or local storage will be maintained; all data will be fetched through the MoviesDatabase API.

2. Caching and Storing Previous Searches:
    - Previous searches and session data will not be stored or cached for future use.

3. User Accounts and Authentication:
    - No user registration, login, or profile management will be implemented.

4. Creating Custom Ratings or Reviews:
    - The platform will not support adding user-generated content such as custom reviews or ratings.

5. Building a Recommendation Engine:
    - Personalized recommendations based on user preferences or viewing history will not be included.

6. Offline Support:
    - The website will not function offline; it requires a live API connection for searching and displaying data.

4. Team Roles and Responsibilities

Team Member	Role	Responsibilities
Azan Mubashar	Developer	● API Integration ● Search function ● Assist with frontend integration and testing
Muizz Zafar	Developer	● Frontend interface ● Sort function ● Frontend integration and testing
Justin Zoorob	Product Owner	● Defining and maintaining project vision ● Making final decisions on work prioritization ● Acceptance testing
Jerry Soong	Scrum Master	● Planning scrum meetings ● Taking scrum meeting minutes ● Verifying development processes and progress ● Removing blockers

Stakeholder	Relation to Product
Professor Abdel Hamou-Lhadj	Client, overseeing the project
Shabnam Hassaniahari	Teaching assistant, proxy to Professor Hamou-Lhadj
Erfan Elhami	Teaching assistant, proxy to Professor Hamou-Lhadj

5. Initial Risk Assessment

Risk	Impact	Likelihood	Mitigation Strategy
Risk 1: API Downtime	High	Medium/Low	Implement caching, have a fallback dataset for offline usage.
Risk 2: Exceeding API Limits	Medium	Low	Monitor usage, implement rate limiting, and optimize API calls.
Risk 3: Integration Issues	High	Medium	Allocate additional time for integration and testing.
Risk 4: Time Constraints	High	Medium	Prioritize features, have a clear timeline, and track progress weekly.
Risk 5: Scope Creep	Medium	Low	Define clear project scope, avoid adding new features mid-project.
Risk 6: API Updates	High	Low	Monitor API changes and allocate time to refactor code if the API structure is changed

6. Constraints

1. API Rate and Usage Limits:
   - The project is dependent on the MoviesDatabase API, which has a hard limit of 100 requests per day and 10 requests per second. Exceeding these limits will cause service disruptions.

2. API Response Time and Data Accuracy:
   - The performance of the website depends on the API's response time and the accuracy of the data provided by the API. Any delays or inaccuracies from the API will affect the user experience and overall functionality of the website.

3. Network Requirement:
   - The website requires an active internet connection to function, since all of the movie data will be fetched in real-time.

4. No Local Data Storage:
   - Due to scope and time limitations, there will be no local storage or caching functionality to save previous searches or user preferences.

5. Scalability Constraints:
   - This implementation will be focused on basic search and display functionalities. Any additional features like user authentication, personalized recommendations, or reviewing functionality are beyond the scope of the current project.

6. Project Timeline:
   - The project timeline is set to 4 weeks. This means our main focus will be implementing the project's core features only.

7. Success Criteria

The project will be deemed successful if the following criteria is met successfully:

- **Functionality:**
  - Allow users to navigate through the database of movies provided by the API and search for movies based on their name, genre, and release year
  - Allow users to sort the results of their search alphabetically and/or by rating in ascending or descending order
  - Users should be able to view all the movie details provided by the API
- **Performance:**
  - The website should have a quick search load time
- **Time Delivery:**
  - The Project should be completed within the allotted sprints without sacrificing the functionality or user experience
- **Team Collaboration:**
  - The team should communicate regularly following the practices of agile methodology and achieve the milestones set for each sprint

8. **Timeline/Release Plan**

Sprint 1: Frontend Skeleton, Search Function (1 Week)

- Tasks:
    - Implement search bar w/ search function to search by movie title.
    - Implement filter options (e.g., genre and rating)
    - Display initial search results with essential information (title, year, and poster).
    - Create a basic user interface layout for search results.
- Sprint Goal: Barebones initial working prototype of site, w/ working search function.

Sprint 2: Sorting Function (1 Week)

- Tasks:
    - Implement sorting functionality for the search results (e.g., alphabetical, rating, release date).
    - Add advanced filtering options (genre, language, and rating).
    - Create a UI for selecting sorting and filtering options.
- Sprint Goal: Added sorting functionality to the site.

Sprint 3: Increased Movie Details (1 Week)

- Tasks:
    - Create a separate movie details page for displaying more in-depth information.
    - Display detailed movie information like the synopsis, cast, crew, and which streaming services can the movie be viewed through.
- Sprint Goal: Improved/Full display of movie information.

Sprint 4: UI/UX Refinement (1 Week)

- Tasks:
    - Develop responsive UI elements for search results and details pages.
    - Implement consistent and functional navigation, styling, and layout design.
    - Ensure the website is mobile-friendly and visually appealing.
- Sprint Goal: Fleshed out UI/UX.

High-Level Release Plan:

- Sprint 1: Frontend Skeleton, Search Function
- Sprint 2: Sorting Function
- Sprint 3: Increased Movie Details
- Sprint 4: UI/UX Refinement

9. Communication Plan

The team will communicate through Discord. The team will have scrum meetings twice per week during which we will talk about what we worked on the previous day, what we will work on today, and if there are any blockers. These stand-ups will synchronize the team on where each task stands, whether a task needs more hands or a prerequisite task completed, and notify our Scrum Master for any blockers they need to remove.

At the end of each sprint, the team plans to have both an end-of-sprint review and an end-of-sprint retrospective. In the review, we will review how the product is progressing and discuss if the product design needs revisions. In the retrospective, we will go over what went well throughout the sprint, and what didn't go so well and discuss how to improve for the next sprint. Both discussions will help us plan our next sprint, whether it is adjusting tasks and priorities to fit a change in product direction or changing our scrum structure to better support development efforts. With these discussions in mind, we will conclude the sprint by organizing the sprint backlog and tasks for the next sprint.

We will also be meeting with at least one stakeholder every Friday, filling them in on the progress of the project. We expect that the stakeholders are all in communication with each other, and that stakeholder contact will be forwarding the information to the other stakeholders.

10. Tools & Resources

The following tools and resources are planned to be used in the project:

- Discord for communication
- Jira for task management
- React for frontend
- Python FastAPI for backend
- Axios for creating and managing HTTP calls and responses
- GitHub for code collaboration