Stage 1. Idea Exploration & Planning (deadline 2/23/2025)

Define Goals & Objectives

- Currently, most book recommendation systems are built on Collaborative Filtering Systems or Content-Based Filtering Systems. In other words, based on current trends in the BookTok and BookTube communities, and from user's past/completed readings. For new users who are either getting into reading or want a recommendation tailored to specific themes or tropes, a recommendation system based on quick prompts where the users detail the type of book they want to read (including genre, themes, intricacy, mood), similar to how a bookstore clerk would provide guidance, would be ideal.
- The goal is to make the process of finding book recommendations fun and straightforward instead of daunting, especially for new readers.

Market Research & Feasibility Analysis: Identify competitors, market needs, and potential customers. Explore existing solutions.
Current competitors include:

- NextThreeBooks
    - Provides three personalized book recommendations based on user preferences, goals, and habits.
    - The user inputs a bit of information about themselves and their reading goals. This combined with different dropdown selections for the type of book, genre/theme, length, and mood defines the book recommendation
    - Uses GPT-3 for tailored suggestions and explains the relevance of each recommendation.
    - Close to what I am targeting for, although this one focuses more on the person rather than on the book.
- NextRead.AI
    - Offers personalized suggestions based on themes, genres, or previous reads.
    - Uses ChatGPT-4 for tailored recommendations
    - Includes features like genre-based recommendations and similar book suggestions
    - Does not use a prompt-based system. It uses a checkbox system instead.
- Readow
    - Allows users to input books they liked to receive recommendations tailored to their preferences.
    - Focuses on refining suggestions based on multiple inputs for greater accuracy
    - More straightforward, but not ideal for new readers

Risk Assessment: Identify technical, financial, and security risks.
Technical Risks

- Integration and compatibility issues. Since models are being constantly updated, there might be the risk that the pre-trained model integration suffers from compatibility issues.

- Similarly, the stability of a hosting service like this one is unknown to me, so I'll put this warning
- If the model is not the ideal one for this scenario, there is the risk of tailoring inaccurate recommendations.

Financial Risks
- As long as we don't exceed the free thresholds offered by database services like Firebase, there should be no financial risks

Security Risks
- Data privacy violations (although I would expect this to not be a problem with pre-trained models, but don't quote me on that)

Stage 2. Requirements Gathering & Analysis (deadline 2/23/2025)

Functional Requirements: Define core functionalities and features, user interactions, business rules, etc.
Core Functionalities and Features
- The user would enter the page for the first time and, if they want to access more features, sign up.
- The landing page must accept written prompts from the user in an input field
- For now, the landing page will display below the prompt box an array of different book recommendations based on the most popular books
- (Optional) The array for books recommendations in the landing page will have different categories of display (e.g. chips) to choose different book categories/genres.
- Using the pre-trained model, (at the moment, use OpenAI or Perplexity) the page will analyze the best matches based on different books' metadata to get the closest results to what the user's prompt requested/wanted.
- The results will be displayed in a Google-like approached, where the user will get a list of books and click on the one they are more interested in. They will get redirected to a separate page with more information
- The user will have a sorting option within the prompt box (probably called options) to choose options such as sort by rating, sort by popularity, sort by page count.
- The user will be able to modify the same prompt box to search for another type of books.
- Modify the prompt to tailor the recommendations based on the user's preferences.
- To ease navigation, there would
- If the user decides to log in to the page, the webpage must save the history of responses (books recommended)
- If the user is a guest, then they will not be able to save the history of responses and neither will they have better recommendations based on the previous responses

- The user will have the option to share the ‘Detailed Information’ page for a certain book as well as the list of books if they are on the ‘Landing Page’.
- (Optional) - Add an expandable view of the list of books that the user gets recommended based on the prompt.

User Interaction

- **Example of Prompt**: “*I'm looking for a fantasy series with an expansive, richly detailed world. It should have deep, intricate lore and a wide array of characters. I really enjoy themes of destiny, ancient prophecies, and the battle between light and dark. Got any suggestions?*”
- **Answer**: “The Wheel of Time series is the perfect fit for what you are looking for: vast worldbuilding, numerous characters, and overarching themes of prophecy and the struggle between good and evil” accompanied by images of the book covers.

Business Rules

- The system shall comply with all facets of the copyright law and the rules/requirements of the pre-trained model host.

Non-Functional Requirements: Performance, Security, Scalability, Accessibility, Compliance.

- The webpage shall return the result of the user’s prompt in no more than 5 seconds (considering the optimal inference time to be around 1.4 seconds according to Google How to optimize the inference time of your machine learning model - UbiOps - AI model serving, orchestration & training)
- All sensitive user data must be encrypted using AES-256 encryption both in transit and at rest (maybe use OpenSSL or PyCrypto?)
- User accounts must lock after five failed login attempts, requiring identity verification to unlock.
- The system must support scaling up to 1,000, although this number will depend on the resources used to build the webpage.
- The system must adhere to GDPR and CCPA regulations, ensuring proper handling of user data and providing mechanisms for data deletion upon request

Use Cases

- Users who are just starting to get into reading will be able to describe the type of story, genre, or themes they find most intriguing (based on other forms of entertainment). The system will return book recommendations tailored towards the user based on their prompt (not relying on previous reads to get those recommendations)
- Users can explore books outside their usual genres or discover diverse authors and translated works (harder to get with conventional collaborative filtering or content-based filtering recommendation systems, which are based on existing trends or user’s reads).
- Users can get suggestions from highly specific categories (e.g., "steampunk novels with strong female protagonists" or "non-fiction about AI ethics").

Stage 3. System Design, Architecture & UI/UX Design (deadline 3/2/2025)

High-Level Architectural Planning
Architecture Style: Based on the relatively small-scale of the project and the limitants in budget, I am learning towards a monolithic architectural approach.

Tech Stack Selection:
Frameworks and Languages for the Frontend:

- Possible frontend frameworks/languages
    - **React** (JavaScript framework)
- Possible backend framework/languages
    - **Node.js**

Database Systems

- **MySQL** (manage structured book data easily)
- **PostgreSQL** (for more advanced actions. Not sure if I will need it)

Integration Strategy:

- Data Sources
    - Google Books API (retrieve book information, viewability and eBook availability)
    - Enhanced Penguin Random House API (alternative to Google Books API??)
    - Hugging Face API (most likely source to get the pre-trained model for the prompt-based recommendation system)
- Integration Method
    - Use HTTP requests towards the Google Books API to gather metadata about books and the Hugging Face API for the pre-trained model.
    - Since the project will most likely be done in JavaScript, I'll use the "axios" library for HTTP requests.
- Data Transformation
    - Parsing of book data (e.g. title, author, description, cover)
    - Data Mapping?? (apparently is the matching fields from one database to another. I'm not sure how or if I will use it. Have to research more about the topic)
- Data Aggregation
    - Combine all the collected book data into a single formatted and visually appealing layout (e.g. GoodReads, StoryGraph)

API & Data Management
API Design: Outline API structure (REST vs. GraphQL), versioning strategy, and documentation

practices (Swagger, Postman).

- API Structure
    - REST (more suited due to the third-party Hugging Face model integration)
- Versioning Strategy
    - URL Versioning? (i'm confused, isn't this a task for those managing the API?)
- Documentation Practices
    - Postman

Database Design: Develop initial data models, consider indexing strategies, partitioning/sharding, and backup/disaster recovery plans.

UI/UX Design
Wireframes & Prototyping (Figma)
Style Guides: Define color schemes, typography, and UI components to ensure a consistent look and feel.

Text Font: Afacad

Pages:

1. Log In
2. Log Out
3. Landing Page
4. Detailed Info Page
5. Account/Profile Page (delete profile, basic user details)
6. Bookmark Page (favorites, save for later)

Dialog/Popup Section

- Settings

Component Libraries: Consider using or building a reusable component library to speed up development and maintain uniformity.
Accessibility Standards: Ensure designs comply with WCAG guidelines for accessibility.
Responsive Layouts: Design for various screen sizes and devices, emphasizing mobile-first design principles.