תמונה שמכילה טקסט, גופן, לוגו, גרפיקה

התיאור נוצר באופן אוטומטי

Software Engineering Department

Braude College

Capstone Project Phase B

**DreamForge: Idea Builder & Project Generator**

**24-1-D-34**

Supervisor: Alex Keselman  
 [Alex@vir-tec.net](mailto:Alex@vir-tec.net)   
Or Yosef   
[oryosef197@gmail.com](mailto:oryosef197@gmail.com)

Yaniv Lippa   
[Yaniv.lippa@gmail.com](mailto:Yaniv.lippa@gmail.com)

**Our GitHub:**

https://github.com/OrYanivProject/DreamForge

Table of Contents

1. Introduction3

1.1. scope of the project3

1.2. Project's code3

2. Project review and project description4

2.1. Project accomplishment description4

2.1.1. Node.js4

2.1.2. Express.js4

2.1.3. React.js4

2.1.4. React Router5

2.1.5. Redux Toolkit5

2.1.6. OpenAI API5

2.1.7. MongoDB5

2.1.8. Firebase5

2.1.8. Agile Development5

2.2. Research Process6

2.2.1. Challenges and solutions6

2.3. Testing Process7

2.3.1. Unit Testing7

2.3.2. Functional Testing8

2.4. Result and Conclusions8

3. User Documentation9

3.1. User's Guide9

3.1.1. Registration and Login9

3.1.2. Create Book11

3.1.3. View, Download or Delete a Book12

3.2. Maintenance Guide 13

3.2.1. Use Case diagram13

3.2.2. Sequence diagram14

3.2.3. Activity diagram15

3.2.4. Hardware and Software requirements16

3.2.5. Api maintenance16

4. Conclusion/Summary18

5. References19

**1. Introduction**

**1.1 scope of the project:**

The goal of our project is to empower anyone, regardless of their expertise, to easily and quickly transform an idea into a formal project book using AI. By leveraging the capabilities of ChatGPT-4 through API calls, we facilitate the creation of detailed project documents and specifications from user-provided input.

Users will be able to register on the website, submit their ideas, and generate personalized project books. These books will be stored in the user’s account, where they can be edited, downloaded, or deleted as desired. The core of the project lies in establishing a seamless interface for users, while performing sophisticated interactions with ChatGPT-4 to manipulate and refine the user’s text.

Upon receiving a user’s idea, the system will request ChatGPT to generate a detailed project specification. This ensures that any missing information is filled in, creating a comprehensive document tailored to the user’s input (e.g., preferred programming language). The system will then make a series of API calls, pulling each chapter of the project book in accordance with a pre-defined structure, and assembling it into a complete and polished document for the user.

**1.2 Projects code:**

<https://github.com/OrYanivProject/DreamForge>

**2. Project review and project description**

**2.1 Project accomplishment description:**

The project consisted of several key phases. Initially, we focused on researching and developing the system architecture, while also familiarizing ourselves with the OpenAI API. Additionally, we explored various programming languages to determine the most suitable options for our project. We developed both the frontend and backend, making well-informed decisions to ensure optimal functionality.

We successfully provided users with a simple and efficient way to generate a complete project book from just an initial idea, offering them the flexibility to explore any concept that made sense. Our approach to handling API calls was designed to produce the book with high accuracy and efficiency.

**2.1.1 Node.js**

Node.js serves as the core server-side technology in this project, responsible for handling backend operations, including API requests and responses. It facilitates communication between the client-side and the MongoDB database, while also managing real-time data processing and user authentication. Node.js was selected due to its asynchronous, event-driven architecture, which supports scalability and high performance for real-time applications.

**2.1.2 Express.js**

Express.js was utilized as the backend framework, built on top of Node.js, to streamline the development of the server-side components. It provides essential features such as routing, middleware, and request handling, enabling a modular and maintainable structure for the API endpoints. The framework's flexibility allows for rapid development and integration of RESTful services required by the application.

**2.1.3 React.js**

React.js was employed for the client-side user interface development. It is a component-based JavaScript library that allows for the creation of a dynamic and interactive user experience. By managing the application’s state effectively, React ensures smooth user interaction, enabling functionalities such as user login, idea submission, and navigation between the bookshelf and project creation modules.

**2.1.4 React Router**

React Router was implemented to manage the routing of the single-page application (SPA). It allows for seamless navigation between different views within the application without requiring full-page reloads. This framework helps maintain the fluidity of the user experience by rendering only the components necessary for each route.

**2.1.5 Redux Toolkit**

Redux Toolkit is used for state management across the React application. It provides a structured approach to managing the global state, which includes user authentication, project ideas, and the dynamic display of project books. The centralized state management offered by Redux ensures consistent data flow and reduces the complexity of handling state changes across various components.

**2.1.6 OpenAI API**

The OpenAI API is at the core of the project book creation process. The system makes a series of calls to the ChatGPT model through the API to generate and refine the content of project books based on user-submitted ideas. Each API call corresponds to the creation of a specific section or chapter in the project book, ensuring that the output adheres to a predefined template.

**2.1.7 MongoDB**

MongoDB, a NoSQL database, was selected for its flexibility in handling unstructured data and its ability to scale horizontally. It stores user data, project ideas, and the generated project books, providing a robust and scalable solution for data persistence. The database’s document-oriented architecture makes it well-suited for managing the dynamic content generated by the application.

**2.1.8 Firebase**

Firebase is integrated into the project to handle the storage of the project books. Once a project book is generated, it is saved in Firebase's cloud storage, which provides a secure and scalable solution for storing large files. Firebase generates a unique URL for each saved project book, allowing users to easily access and download their books. This URL is then stored in MongoDB, associated with the user's account and project, enabling seamless retrieval of the book whenever the user wishes to view, download, or delete it.

**2.1.9 Agile Development**

The project followed Agile development methodologies to ensure flexibility, continuous improvement, and iterative progress. Regular sprints allowed for the continuous assessment of requirements and the timely delivery of functional components. Agile's adaptability was crucial in handling the evolving integration of the OpenAI API and other third-party services.

**2.2 Research Process**

**2.2.1 Challenges and Solutions**

Throughout the development of the project, several significant challenges were encountered, both technical and logistical. A key challenge stemmed from the fact that both team members were balancing full-time jobs alongside their commitment to this project, which made it difficult to find dedicated time for focused collaboration. This limited availability often resulted in fragmented development sessions and required careful scheduling to ensure progress was consistently made. To mitigate this, asynchronous communication tools such as Slack and Trello were employed, enabling both partners to update each other and track progress, even when working at different times.

On the technical front, one of the primary challenges was optimizing the interaction between the backend and the OpenAI API for project book generation. Due to the nature of the API calls, performance bottlenecks occurred, particularly when handling multiple requests for book creation. These bottlenecks were addressed by implementing request throttling and making use of asynchronous processing techniques. This ensured that the system could handle larger volumes of data while maintaining responsiveness for the end user.

Another challenge was ensuring secure and efficient storage of the project books. Firebase was chosen for its cloud storage capabilities, allowing the project books to be saved and easily accessed. However, integrating Firebase with MongoDB required careful coordination. The solution was to store only the URL generated by Firebase in MongoDB, thus minimizing the strain on the database and keeping sensitive data secure. This hybrid approach balanced the need for a scalable cloud storage solution with the flexibility of MongoDB for user and project data management.

In summary, although the project development was hindered by the time constraints of balancing work with project responsibilities, creative solutions like asynchronous collaboration, API optimizations, and thoughtful storage integration helped overcome these obstacles, ensuring the successful delivery of the system.

**2.3. Testing Process**

We tested our program according to the testing plane we provided in the first part of the project. As we decided back then the testing plane consist of two parts: Unit testing and Functional testing.

**2.3.1 Unit Testing:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Interface Page** | **Test Subject** | **Expected Result** | **Result** |
| **1** | Login | Enter valid user details and press “Login” button | The screen will switch to the “Idea submission” page | Pass |
| **2** | Idea submission | Enter a valid idea and press the “Make It Real” button | The screen will switch to the “Book created” page | Pass |
| **3** | Book created | press the “Download Book” button | The project book will be downloaded to the user’s computer | Pass |
| **4** | Book created | press the “View Book” button | The created book will be shown on the screen | Pass |
| **5** | Bookshelf | press “Remove” button for specific book | A warning message window will appear:  “Are you sure you want to remove the book?” | Pass |
| **6** | Bookshelf (warning) | press “Yes” button after warning appear | The book will be removed from the bookshelf | Pass |
| **7** | Bookshelf (warning) | press “No” button after warning appear | The screen will return to the original bookshelf screen | Pass |
| **8** | Bookshelf | press the “View” button for a specific book | The chosen book will be shown on the screen | Pass |
| **9** | Bookshelf | press the “Download” button for a specific book | The chosen book will be downloaded to the user’s computer | Pass |

**2.3.2 Functional Testing:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Interface Page** | **Test Subject** | **Expected Result** | **Result** |
| 1 | Registration | Enter invalid email | Message: “Invalided email address” | Pass |
| 2 | Registration | Invalid character entered | Message: “Invalid character used” | Pass |
| 3 | Registration | Enter already existing username | Message: “This username already exists” | Pass |
| 4 | Login | Enter wrong username/password | Message: “Wrong username or password” | Pass |
| 5 | Login | Enter an already logged-in user detail | Message: “This user is currently logged-in” | Pass |
| 6 | Login | Login with empty username or password | Message: “Must enter username and password” | Pass |
| 7 | Idea submission | Enter idea with less than four words | Message: “Idea must be more then four words” | Pass |
| 8 | Idea submission | Meaningless idea entered | Message: “Could not understand your idea” | Pass |
| 9 | Idea submission | Offensive text entered | Message: “Warning – your idea may be offensive” | Pass |

**2.1. Result and Conclusions**

Our DreamForge project successfully passed all the tests conducted. One of our primary objectives was to experiment with various ideas to better understand how the chat system responds. With some guidance, we were able to refine the chat's outputs, improving accuracy by continuously adjusting and sharpening the prompts.

Our aim was to enable any user, regardless of technical knowledge, to easily create a comprehensive project book in the most user-friendly way possible. After testing the system with a diverse range of users—some with technical expertise and others without—we can confidently say that we achieved this goal.

While we recognize that there are always opportunities for improvement, we are proud of the current results and believe the project will be valuable for any user.

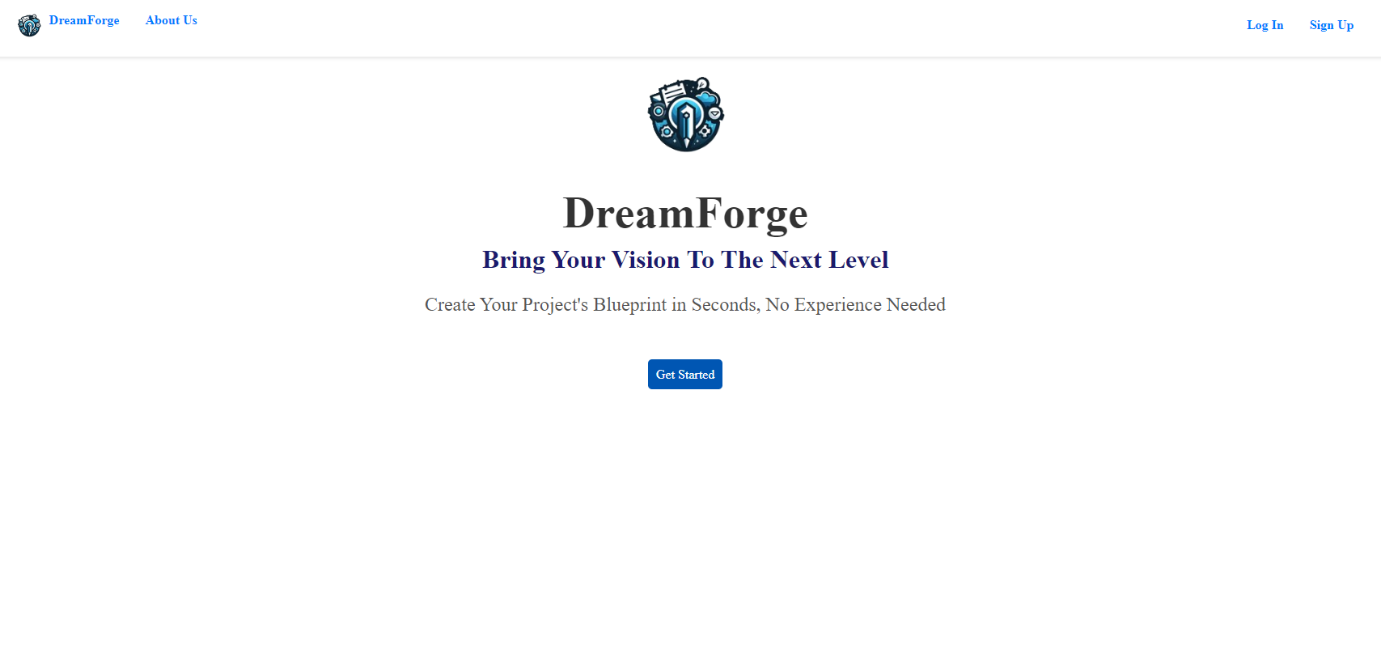
**3. User Documentation**

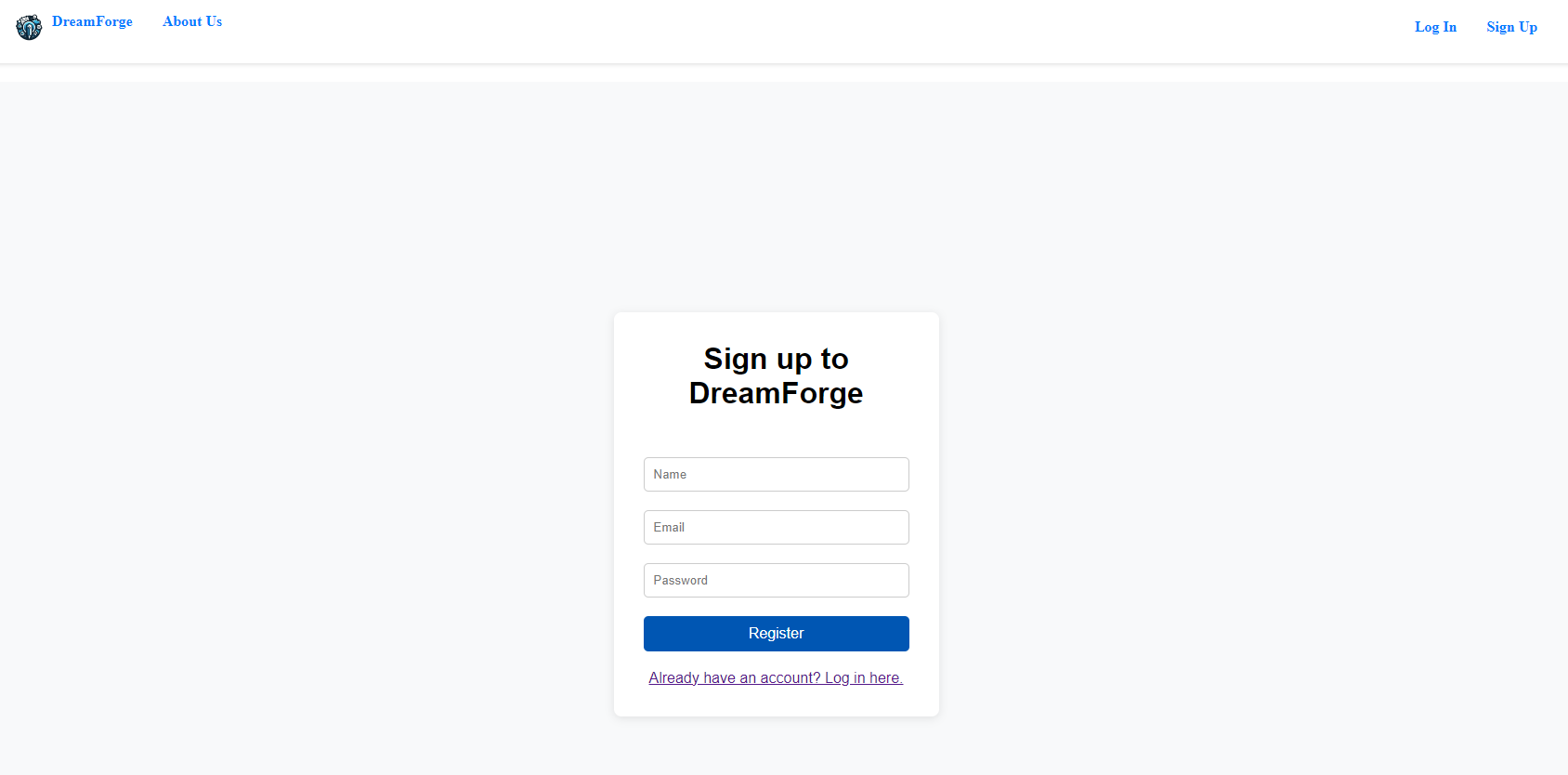
**3.1. User's Guide**

This section provides a detailed overview of how end users interact with the system, highlighting its key functionalities. The website is intentionally designed to be as simple and minimalistic as possible, ensuring that users enjoy an optimal and intuitive experience.

**3.1.1. Registration and Login**

To begin using DreamForge, users must first register. The registration process is straightforward, requiring only a username (at least 4 characters long and free of special symbols like "." or "/"), a valid email address, and a password with a minimum of 6 characters. Users can access the registration page by clicking the "Get Started" button on the homepage or by selecting the "Sign Up" button located on the right side of the toolbar.



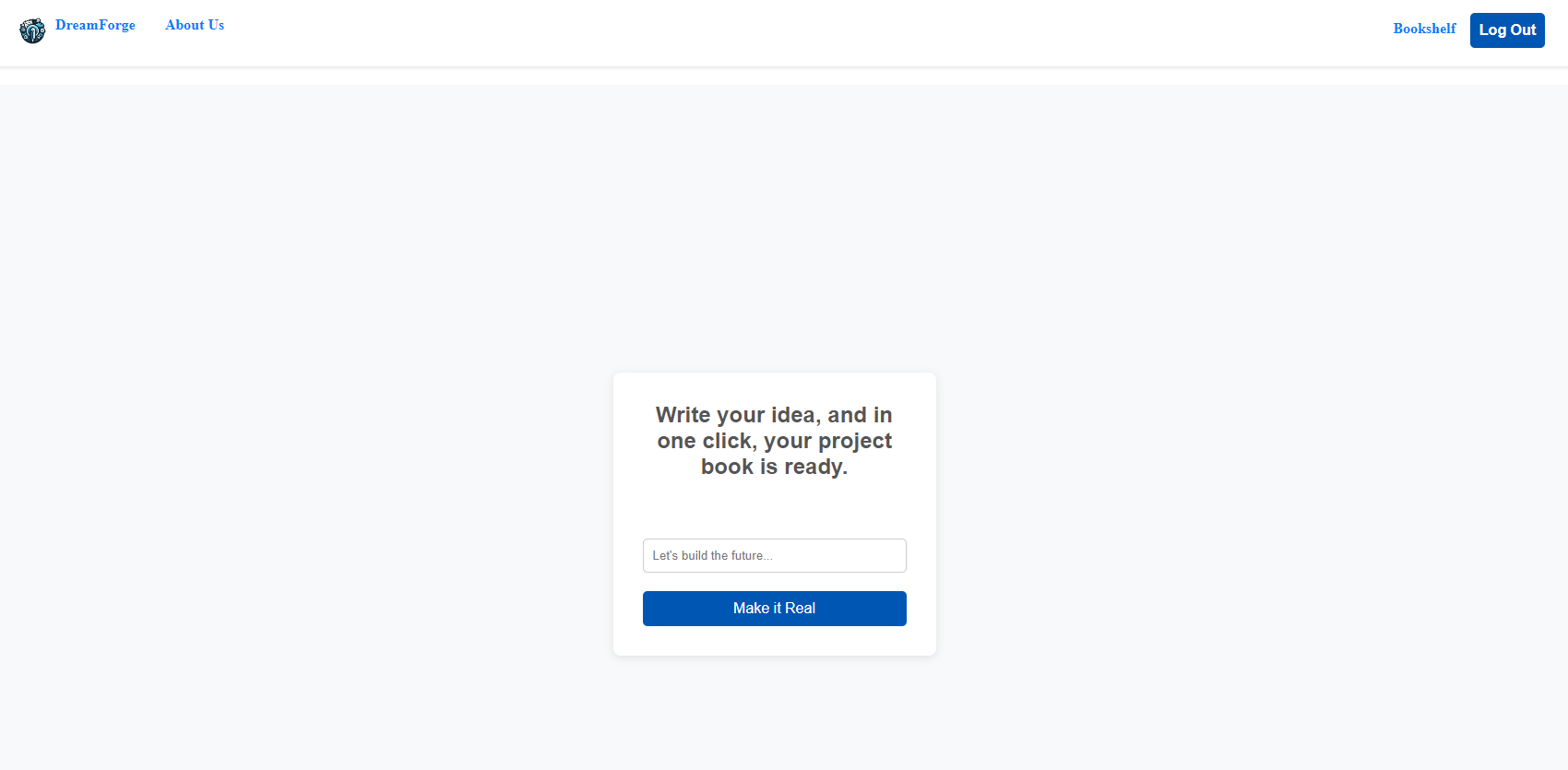


**תמונה שמכילה טקסט, צילום מסך, תוכנה, מערכת הפעלה

התיאור נוצר באופן אוטומטי**Users who have already signed up can click the "Already have an account? Log in here" link located below the registration button on the registration page, or they can navigate to the login page by selecting the "Log In" button on the right side of the toolbar, available on any page.

**3.1.1. Create Book**

After been logged in you will be able to create a new project book by entering the idea of the project in the text box. After pressing the "Make It Real" button the page will change to loading page. When the book ready the page will change to bookshelf and it will give the user option to view, download or delete the book.



**2.1.3. View, Download or Delete a Book**

תמונה שמכילה טקסט, גופן, צילום מסך

התיאור נוצר באופן אוטומטיAny logged in user can easily view his project books by clicking the "bookshelf" button at the right side of the toolbar. At the bookshelf page you will be able to view any book online by pressing the "View" button next to it, download any book to your computer by pressing "Download" button next to it or pressing "Delete" to delete the requested book.

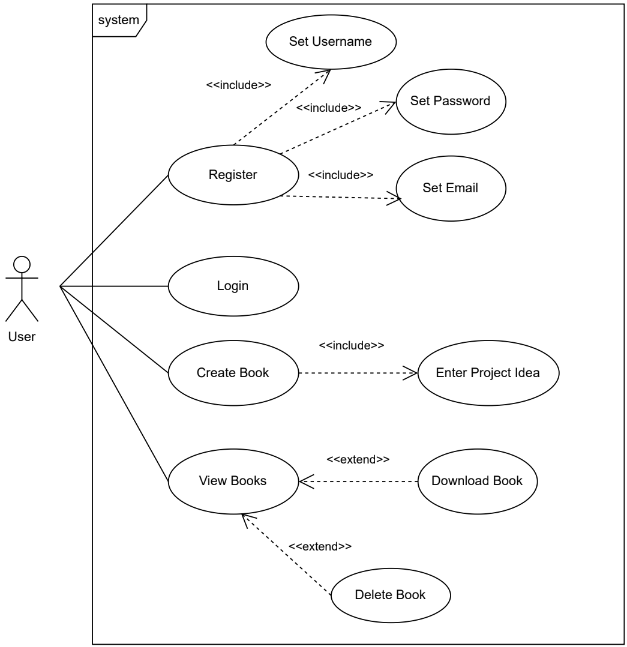
תמונה שמכילה טקסט, תוכנה, דף אינטרנט, אתר

התיאור נוצר באופן אוטומטי

**3.2. Maintenance Guide**

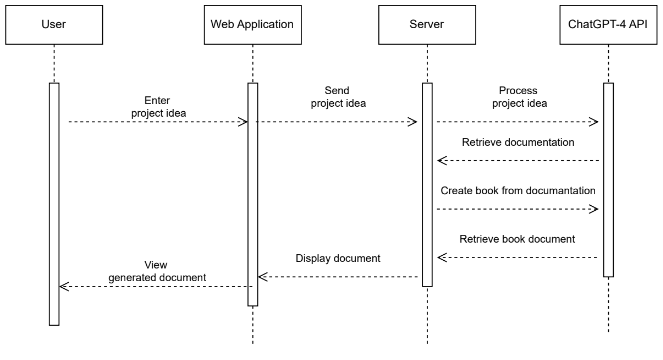
**3.2.1. use case Diagram**

The Use Case diagram provides a high-level overview of the system's functionality, helping to define the scope of the project and understand user requirements. It highlights the key features and scenarios in which the system will be used, guiding the design and development process.

****

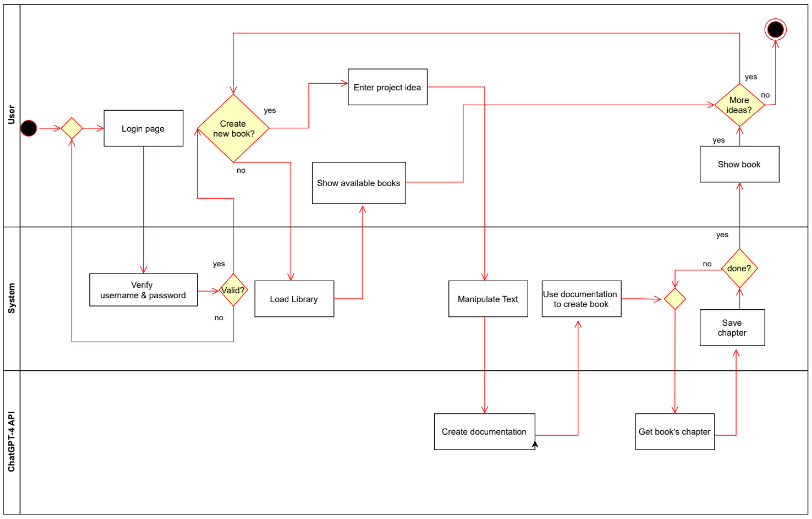
**3.2.2. Sequence Diagram**

This sequence diagram provides a visual representation of the flow of actions, starting from the user's interface with the web application, passing through the server, and reaching the ChatGPT4 API system

****

**3.2.3. use case Diagram**

This is the activity diagram, illustrating the sequential actions and interactions within the project's workflow.

****

**3.2.4. Hardware and Software requirements**

**Dependencies:**

    React: 18.3.1

    React-dom: 18.3.1

    React-router-dom: 6.26.1

Node.js v21.1.0.

  Mongodb: 6.8.0

mongoose: 8.6.0

multer: 1.4.5-lts.1

    Axios: 1.7.7

    Dotenv: 16.4.5

    Express-session: 1.18.0

    firebase: 10.13.1

    jsonwebtoken: 9.0.2

**3.2.5. API maintenance**

The platform relies heavily on the ChatGPT API to generate the content of the project books. Maintaining this API integration is crucial for ensuring the system’s functionality and performance. The createBook API call is the most critical part of the system, as it initiates the process of generating a structured project book from a user’s idea. Proper maintenance of this API involves several key considerations:

1. **Handling Conditional Logic**: The API calls within the createBook function allow for customization in how the project book is generated. Developers can introduce conditions to modify the flow of API requests, such as altering chapter names or changing the structure of the entire book. This flexibility is crucial for future adjustments or expanding the system’s capabilities. Proper testing of these conditional changes is necessary to ensure that the resulting book remains cohesive and correctly structured.
2. **API Key Management**: Since the ChatGPT API requires an active API key for usage, it’s important to manage key expiration and rotation carefully. The system must monitor the validity of the API key and ensure it is updated when necessary. Failure to do so will result in failed API requests and an inability to generate project books. Developers should implement an alert or error-handling mechanism to notify administrators when the key is nearing expiration. This prevents service interruptions and ensures continuous access to the API.
3. **Performance Optimization**: ChatGPT API calls, especially when generating large volumes of content, can introduce latency into the system. To mitigate this, asynchronous API handling has been implemented to prevent blocking other requests. Additionally, throttling mechanisms have been applied to ensure the system adheres to rate limits imposed by the API, preventing any overuse that could lead to temporary suspension of access.
4. **Version Control and Updates**: As the ChatGPT API evolves, newer versions may introduce changes in behavior or new features. Regularly monitoring for updates or changes in the API is essential to ensure compatibility and to take advantage of any new functionality that could improve the book generation process. Any structural changes to the book template must be tested across multiple API versions to maintain consistency and quality of the generated content.

In summary, API maintenance not only involves managing the core technical aspects, such as key expiration and rate limits, but also adapting to evolving project requirements through conditional logic and template adjustments. Regular monitoring and proactive key management will help ensure the smooth functioning of the book creation process.

**4. Conclusions / Summary**

In conclusion, this project successfully demonstrates the integration of modern web technologies and artificial intelligence to create a user-friendly platform for generating structured project documentation. By leveraging Node.js, Express.js, React.js, and MongoDB, the system provides a scalable and efficient solution for managing user data and dynamically generating project books.

The core feature of the platform is its seamless interaction with the OpenAI API, which allows users to input their ideas and receive fully formatted project books based on a predefined structure. This automation streamlines the process of project documentation, making it accessible to users with varying levels of technical expertise.

Throughout the development process, various challenges were encountered, particularly in optimizing the system's performance and ensuring secure data handling. However, by following Agile development methodologies and employing thorough testing, these challenges were effectively addressed. The result is a robust and scalable platform that meets the project's original objectives.

This project demonstrates the potential of combining AI-driven content generation with modern web development frameworks to provide practical solutions in the field of project management and documentation. Future enhancements may include further optimization of API usage, expanding template customization options, and incorporating additional features such as collaborative editing or project tracking.

**5. References**

1. Node.js - <https://nodejs.org/en>
2. Express.js - <https://expressjs.com/>
3. React.js - <https://react.dev/>
4. React Router - <https://reactrouter.com/en/main>
5. Redux Toolkit - https://redux-toolkit.js.org/
6. OpenAI API - https://platform.openai.com/docs/api-reference/introduction
7. MongoDB - https://www.mongodb.com/docs/
8. Firebase - <https://firebase.google.com/>
9. Apryse - <https://docs.apryse.com/>