CSE 341 Final project Proposal

General Info

Group Members

Daniel C. Opute,

Nigel Ndlovu,

Sakhile Mamba

Proposed Application Name

BookSphare API

Contents:

Contents

Application Info

What will the API do?

BookSphere API is a tool that makes managing and accessing library resources easy and efficient.

- **For Developers Building for Readers:** The API provides endpoints to search for books, manage borrowing, and leave reviews. It enables features like tracking due dates, and accessing digital content like eBooks or audiobooks.
- **For Developers Building for Librarians:** The API offers endpoints to manage the library catalog, track inventory, handle user accounts, and enforce borrowing policies.

In short, **BookSphere API** is designed to power applications that connect people with books while simplifying library operations through a developer-friendly interface.

How will your API utilize a login system?

The API will utilise the traditional email and password login, or OAuth 2.0, for authentication, allowing users to log in via third-party services (Github).

What database will you use?

We will be using the MangoDb database.

How will the data be stored in your database?

There will be four collections with fields, which are:

1. Users {
   “userId”: “unique_id”,
   “firstname”: “name”,
   “lastname”: “name”
   “email”: example@example.come
   “passwordHased”: “hased_password”,
   “oauthProvider”: “GitHub”,
   “oauthToken”: “token_string”,

```
                “createdAt”: “timestamp”,
                “role”: “admin/user”
        }
    2.  Books {
                "bookId": "unique_id",
                "title": “book title",
                "author": "writer/s name",
                "genre": "Fiction",
                "publishedDate": "date of publication (DATE)",
                "summary": "Brief summary of the book",
                "totalCopies": 15,
                "availableCopies": 9,
                "createdAt": "timestamp"
        }
    3.  BorrowRecords {
                "recordId": "unique_id",
                "userId": "unique_id",
                "bookId": "unique_id",
                "borrowDate": "timestamp",
                "returnDate": "timestamp",
                "isReturned": false
                “penalise” : false
        }
    4.  Reviews {
                "reviewId": "unique_id",
                "bookId": "unique_id",
                "userId": "unique_id",
                "rating": 4,
                "comment": "Awesome Book!",
                "createdAt": "timestamp"
        }
```

How would a frontend be able to manage authentication state based on the data you provide?

On successful login, the frontend will receive a JWT from the backend (BookSphere API). This token will be stored in a sessionStorage and be used for subsequent requests to authenticate the user.

What pieces of data in your app will need to be secured? How will you demonstrate web security principles in the development of this app?

Sensitive information such as the API client ID, client secret, database connection string, and others will be stored in a .env file and accessed using the dotenv module in the application.

What file structure and program architecture will you use for this project (how will you organize your node project)? Why?

For this Node project, we will use a **modular file structure** and follow a basic **MVC (Model-View-Controller)** architecture. This approach ensures that the project remains organized, scalable, and easy to maintain, even as it grows in complexity.

What are potential stretch challenges that you could implement to go above and beyond?

Prevent Duplicate Records

Ensure unique constraints on fields like email, book title, etc., and handle duplicates gracefully.

API Endpoint Planning

For this section, you'll plan out what API endpoints you'll need for your project.

- users :
  - POST/users --- Create a new user
  - GET/users --- Get all users
  - GET/users/login --- Login a user
  - GET/users/logout --- Logout a user
  - GET/users/userId --- get a user by their ID
  - PUT/users/userId --- update a user
  - DELETE/users/userId – delete a user
- books:
  - POST/books --- Add a new book
  - GET/books – Get all books
  - GET/books/bookId --- get a book
  - GET/books/title --- get books by title (a search query)
  - PUT/books/bookId --- update a book
  - DELETE/books/bookId --- delete a book
- borrowRecords:
  - POST/borrow --- Add a new borrow record
  - GET/borrow – Get all borrow records
  - GET/borrow/borrowId --- get a borrow record
  - PUT/borrow/borrowId --- update a borrow record
  - DELETE/borrow/borrowId --- delete a borrow record
- reviews:
  - POST/review --- Add a new review
  - GET/review – Get all reviews
  - GET/review/reviewId --- get a review by ID
  - PUT/review/reviewId --- update a review
  - DELETE/review/reviewId --- delete a review

Project Scheduling and Delegation

Plan out what tasks will get completed with each lesson remaining in the semester (Only edit highlighted text).

Week 04 Tasks	*Project Proposal*
Week 05 Tasks	● *Create Git Repo* ● *Push to Heroku* ● *API DOCUMENTATION is complete and available at route '/api-docs'*
Week 06 Tasks	● Implement Authentication and Authorization Mechanisms ● Add our stretch challenge ● Publish to render
Week 07 Task	● Do a thorough test on the API *…Video Presentation…*

How will you divide up work in your team to ensure the following tasks all get completed?

- HTTP GET, GET (all, single) --- Nigel Ndlovu
- HTTP POST --- Daniel C. Opute
- HTTP PUT --- Daniel C. Opute
- HTTP DELETE --- Sakhile Mamba
- Node.js project creation --- Daniel C. Opute
- Create git repo and share it with group --- Nigel Ndlovu
- MongoDB setup --- Sakhile Mamba
- API Swagger documentation for all API routes --- Sosa Maximiliano Bustios
- Video presentation of node project, all routes functioning, MongoDB data being modified, and API documentation. --- Daniel C. Opute

Potential Risks and Risk Mitigation Techniques

What are the risks involved with you being able to finish this project in a timely manner?

Lack of commitment to completing individual tasks as assigned could hinder or delay the project's progress.

How will you mitigate or overcome these risks?

We will exchange contact information to remind and follow up with each other, ensuring tasks are completed promptly to avoid delaying the project's completion.