1. Introduction

For this assignment, your task is to develop a straightforward library management system that allows users to borrow books. The core components to be implemented include APIs for user management, book management, and borrowing management. Furthermore, it is essential to incorporate role-based authentication functionality. Detailed instructions and additional information will be provided in subsequent sections.

Please be mindful that your code will be evaluated based on coding best practices, adherence. to REST standards, and the security of your implementation. To facilitate this, you are required. to create a GitHub repository for your project. We will closely examine your commits to assess. your progress and the development process. Additionally, maintaining comprehensive API documentation using Postman is a mandatory part of this assignment. This will ensure clarity. and accessibility for reviewers and future users of the system.

2. Setup

- Create a new Node.js project.
- Set up a MongoDB database.
- Initialize your project with npm, and install the necessary dependencies (Express, Mongoose, etc.).

3. Data Models

3.1 AUTH

The primary objective of this model is to securely store and maintain users' login credentials.

- _id: String (User’s email address)
- password: String

3.2 Users

The primary objective of this model is to maintain user data.

- _id: ObjectId
- authId: String (Reference to the Auth document)
- name: String
- type: String (Admin/User)

3.3 Books

The primary objective of this model is to maintain books.

- _id: ObjectId
- name: String
- totalCopies: Number (use this attribute to maintain total number of copies)
- availableCopies: Number (use this attribute to maintain available number of copies)

3.4 Borrowings

The primary objective of this model is to manage and track user book borrowing records.

- _id: String
- User: {

  _id: ObjectId (Reference to the User document)

  name: String

  }
- book: {

  _id: ObjectId (Reference to the Book document)

  name: String

  }
- isReturned: Boolean

4. API Endpoints

4.1 Users

It's important to note that when creating a user, an authentication document should be generated. Equally, when deleting a user, the associated authentication document must be removed as well.

4.1.1. Create User

This endpoint should only allow regular users to register themselves. You should implement seeder function to create admin accounts.

4.1.2. Delete User

Only the owner is allowed to delete the user.

4.2 AUTH

4.2.1. Login

Both admins and regular users should be able to use this endpoint to login to the system.

4.3 Books

4.3.1. Create Book

This endpoint should allow only admins to create books.

4.3.2. Delete Book

This endpoint should allow only admins to delete books.

4.3.3. View All Book

This endpoint should allow all user roles to view books in the system

4.4 Borrowings

4.4.1. Borrow Book

This should allow only regular users to borrow books.

4.4.2. Return Book

This endpoint should allow only admins to mark when the users returns them.

4.4.3. View Self Borrowings

This endpoint should allow only regular users to view their borrowings. Implement a filter to return all borrowings and currently active borrowings.

4.4.4. View Borrowings by User

This endpoint should allow only admins to view borrowings of each user by user ID. Implement a filter to return all borrowings and currently active borrowings.

5. Requirements

5.1 Mandatory Requirements

Candidates must fulfill mandatory requirements to pass this assignment.

1. Implement given API endpoints by following proper best practices in building REST APIs.
2. Maintain a GitHub repository and push your changes to the repository continuously.
3. Maintain a Postman API documentation.
4. Deploy your backend. You may use https://render.com/ to deploy your backend or any cloud service of your preference.
5. Use MongoDB Atlas to host your database.
6. Create a readme.md file in your GitHub repository that explains your project and instructions to run it.

5.2 Optional Requirements

These requirements are optional and may be rewarded if fulfilled.

1. Use Typescript instead of JavaScript.
2. Write unit tests (Following TDD principles may be rewarded).

3. Index the database to optimize query performance.

6. Deliverables

1. Publicly accessible GitHub link to the repository.
2. Publicly accessible Postman documentation link.