Group 10 Project Proposal

Online Bookstore with Shopping Cart

Group Members:

- Nagasaro Ghislaine
- Enzo Batungwanayo
- Samuel Oluwafemi Komaiya
- Kosisochukwu Divinefavour Okeke

1. Project Overview

This project uses the principle of Object-Oriented Programming(OOP) in JavaScript to create a scalable and maintainable backend system. The system is a dynamic book catalogue that allows e-commerce functionalities, including basic CRUD operations. The core components of this system are the User, Book, category and Cart classes, with their respective controllers and routes to allow CRUD operations. The system will use MongoDB for data storage, with Mongoose as the Object Document Mapper (ODM) to enforce schema definitions and streamline database operations.

2. Problem Statement

As online commerce continues to grow, systems that effectively simulate real-world business operations become essential learning tools. This project models a simplified e-commerce bookstore where users can:

- View a catalogue of books sourced from a MongoDB database
- Browse books by category (e.g., Fiction, Non-fiction)
- Add and manage books in a personal shopping cart
- Simulate the checkout process, reducing book stock and clearing the cart

The system aims to demonstrate OOP, proper database integration, and modular design reflective of real-world development practices.

3. Proposed Solution

We will develop a full-stack simulation of a bookstore backend using JavaScript, Node.js, MongoDB and HTML and CSS for a basic frontend. The application will demonstrate clear object modelling and encapsulation using custom classes.

3.1 Class Structure

Book Class

- **Properties:** title, author, price, ISBN, stock, genre (Category), seller
- **Methods:** getInfo(), updateStock(), createStock(), deleteStock(), getAllStock()

Category Class

- **Properties:** name, description
- **Methods:** getInfo(), updateCategory(), createCategory(), deleteCategory(), getAllCategories()
- **Purpose:** Organises books into genres; supports filtering and extensibility.

User Class

- **Properties:** username, email, password, role(buyer, seller)
- **Methods:** register(), login(), logout(), deleteProfile()

Cart Class

- **Properties:** list of Book items and quantities, userID
- **Methods:** addBook(), removeBook(), viewCart(), getTotal(), checkout()

Store Class (Controller Layer)

- **Responsibilities:** Handle database interactions, book listings, stock updates, and purchase processing.

3.2 Functional Workflow

1. Load book inventory from MongoDB.
2. Fetch all categories and allow filtering by category.
3. Allow users to browse and add books to their cart.
4. View and update cart items.
5. Simulate checkout:
    - Decrease book stock
    - Clear the user's cart

4. Technologies Used

Purpose	Technology
Programming Language	JavaScript (ES6+)
Backend Runtime	Node.js
Database	MongoDB
ODM	Mongoose

Version Control	Git & GitHub
Editor	Visual Studio Code
Frontend	HTML/CSS/JS
Testing	Postman, Unit Tests

5. Project Timeline

Day	Milestone
1	● Finalise project plan, define schema, design UML class diagrams. ● Implement core classes: Category, Book, and User
2	● Integrate MongoDB and seed data for books and categories
3	● Develop frontend UI, cart logic and checkout workflow
4	● Conduct testing, fix bugs, and validate functionality
5	● Wrap-up: documentation, prepare demo & submission

6. Team Structure and Responsibilities

Role	Responsibilities
Backend Dev 1 **(Ghislaine)**	Develop User, `Book`, and `Category` classes, define schemas, routes and documentation, and work on the project proposal document.
Backend Dev 2 **(Samuel)**	Develop Cart class, define schema, route and documentation, implement throttling, and work on the system architecture document.
Backend Dev 3 **(Okeke)**	Test the backend system, create the UML diagram, implement documentation and work on the project proposal document.
Frontend Dev **(Enzo)**	Create a basic UI for browsing books and managing the cart, implement business logic, and work on the system architecture document.

7. Deliverables

- Modular JavaScript backend using OOP principles
- MongoDB integration with Mongoose
- Data for books and categories
- Simple frontend
- Full project documentation:
    - UML class diagram
    - README

    - Setup instructions

- GitHub repository with version-controlled source code
- Demo video walkthrough

8. Conclusion

This project offers practical exposure to backend development using OOP principles and MongoDB. The introduction of the `Category` class enhances the realism and scalability of the application.