Group 10 Project Proposal

Project choice: Online Bookstore with Shopping Cart
Group Members:

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

1. Project Overview

This project proposes the development of an online bookstore simulation using Object-Oriented Programming (OOP) in JavaScript, aimed at creating a scalable, and maintainable backend system. It will model real-world e-commerce functionality by allowing users to browse a dynamic catalog of books, manage a shopping cart, and simulate purchases. Core components such as `Book`, `User`, and `ShoppingCart` will be implemented as classes, encapsulating both data and behavior to ensure code reusability and separation of concerns. The system will use MongoDB for data storage, with Mongoose as the Object Document Mapper (ODM) to enforce schema definitions and streamline database operations. Key features will include book management (title, author, price, genre, stock), user account simulation, cart operations (add/remove/update items, calculate totals), and a mock purchase workflow. The architecture will prioritize future extensibility, making it possible to integrate advanced functionality such as recommendation engines, real-time updates, or full authentication layers. Overall, this project serves as a foundational demonstration of applying OOP to backend development in a simulated e-commerce context.

---

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

6. Team Structure and Responsibilities

Role	Responsibilities
Backend Dev 1 **(Ghislaine)**	Develop `Book` and `Category` classes, define schemas, seed MongoDB; assist with testing and documentation
Backend Dev 2 **(Samuel)**	Build `User` class and registration/login logic; cart reference; assist with testing and documentation
Backend Dev 3 **(Okeke)**	Build `Cart` and `Store` classes, implement purchase logic and interactions with database; assist with testing and documentation
Frontend Dev **(Enzo)**	Create a basic UI for browsing books and managing cart; assist with testing and documentation

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. With proper modularisation and database design, this project can serve as a strong foundation for learning real-world backend concepts and e-commerce systems.