E-Commerce Web App — AI-Assisted Development (Report-2)

Tools Used

- **ChatGPT** – Debugging signup/login routes, fixing MongoDB Atlas connection errors, generating sample frontend with Bootstrap cards, and helping restructure backend into routes/controllers/schemas.
- **Claude AI** – Suggested best practices for Express middleware, modular backend design, and error handling strategies.
- **GitHub Copilot** – Auto-completed repetitive JSX in product cards and boilerplate in Mongoose schemas and controllers.

Project Overview

This phase focused on:

1. Finalizing **signup and login backend functionality** with MongoDB Atlas.
2. Building a **sample frontend main page** with a product grid using React + Bootstrap.
3. **Restructuring the backend** into separate folders for routes, controllers, and schemas for scalability.
4. Beginning **product catalogue API development** to serve product data to the frontend.

AI Interaction & Prompts

- **Signup/Login Backend**
    - ChatGPT initially provided a modular backend setup (routes, controllers, models). I requested a **simplified single-file index.js** version for testing, which worked better.
    - When MongoDB Compass failed to connect, I asked ChatGPT for help. It explained how to properly use **MongoDB Atlas connection strings**, replacing <db_password> and appending the database name.

  - AI flagged that useNewUrlParser and useUnifiedTopology are no longer needed in Mongoose v6+.
  - When hitting 500 Internal Server Error, ChatGPT pointed out a **frontend-backend mismatch** (/register vs /api/v1/signup/register) and guided me to check backend logs for clarity.
- **Frontend Main Page**
  - Started with a minimal header: 🛒 MyShop and subtitle.
  - Prompted AI to add a **Bootstrap container + row** → AI inserted a products array and console logged it.
  - Asked for **product cards** → AI added Bootstrap card grid with product image, name, and price.
  - Enhanced with **“Add to Cart” button**, Bootstrap shadows, and spacing for a polished look.
- **Backend Restructuring**
  - Asked AI for recommended folder structure. It suggested:

```
backend/
        routes/
        controllers/
        models/
        config/
        server.js
```

  - ChatGPT explained how to separate signup logic into controllers and export routes cleanly.
  - Claude suggested using **async/await with try/catch** in controllers for better error handling.
- **Product Catalog API**
  - Prompted AI to create a sample product schema and REST API.
  - Copilot assisted by completing the schema fields and Express routes (GET /products).
  - AI provided sample test data and showed how to integrate it into MongoDB Atlas.

AI’s Role

- **ChatGPT**: Debugging signup/login flow, simplifying backend into single file, restructuring backend folder, and generating React frontend product cards.
- **Claude AI**: Provided architectural advice on backend modularization and error handling.
- **GitHub Copilot**: Auto-completed repetitive Mongoose and JSX code to speed up development.

My Implementation

- Implemented **SignupForm.jsx** with Axios POST requests to backend.
- Created **LoginForm.jsx** (basic structure; functionality under development).
- Built **index.js backend** for signup/login, later restructured into modular folders.
- Set up **MongoDB Atlas cluster** successfully with working connection string.
- Developed **Main.jsx frontend page** displaying sample product cards in Bootstrap.
- Added **Product model, controller, and API route** for fetching catalog data.

Code Evolution

- **Signup Backend**
    - *AI’s initial version*: Modular with routes/controllers; too complex to test quickly.
    - *Final version*: Simplified single-file backend for signup, then restructured into modular folders for maintainability.
- **Frontend Main Page**
    - *AI’s initial version*: Minimal header with shop name and subtitle.
    - *Final version*: Bootstrap-based product grid with images, names, prices, shadows, and Add-to-Cart buttons.
- **Product Catalog API**

    - *AI's initial version*: Hardcoded array in frontend.
    - *Final version*: Express + MongoDB API with GET /products serving real product data.

Component Working

- **SignupForm.jsx** – Uses useState to capture input; Axios POST to /register; backend validates and saves to Atlas.
- **LoginForm.jsx** – Captures credentials; backend route prepared for JWT-based validation (to be completed).
- **Main.jsx** – Maps products array into Bootstrap cards with images, name, price, and "Add to Cart" button.
- **Backend** – Modular structure: routes/ (signup, login, products), controllers/ (logic), models/ (Mongoose schemas).
- **Product API** – Returns product data in JSON, ready to integrate with frontend.

Advantages

- Signup/login now working with MongoDB Atlas.
- Product catalog backend ready to integrate with frontend.
- Modular backend folder structure makes scaling easier.
- Bootstrap frontend is responsive and visually appealing.
- AI-assisted debugging significantly reduced trial-and-error time.