**Full Stack Development with MERN**

**Project Documentation**

**Team ID:** LTVIP2025TMID54367

**1. Introduction**

* **Project Title:** ShopEZ – One-Stop Shop for Online Purchases
* **Team Members:**
  + **Chemalamarri Vyshnavi** – Full Stack Developer: Manages end-to-end development, authentication systems, and deployment.
  + **B Praveen Varma** – Frontend Developer: Responsible for React.js UI components, user experience design, and client-side functionality.
  + **Amballa Vigneswari** – Backend Developer: Handles Node.js/Express.js server development, API creation, and database integration.
  + **B Manogna** – Database Administrator: Designs MongoDB schemas, optimizes queries, and manages data relationships.

**2. Project Overview**

* **Purpose:**  
  ShopEZ is a comprehensive e-commerce platform built to offer a seamless digital shopping experience. It enables users to browse, search, and purchase products with ease while offering sellers and admins robust control panels. The project focuses on scalability, personalization, and real-time features for a next-gen shopping journey.
* **Features:**
  + User authentication and role-based access control
  + Product browsing, search, and filter functionality
  + Add to cart with quantity control
  + Checkout with secure payment integration
  + Admin panel for managing users, products, and orders
  + Real-time order status updates
  + Clean, responsive user interface

**3. Architecture**

* **Frontend:**
  + **Framework:** React.js
  + **Routing:** React Router (with protected and dynamic routes)
  + **State Management:** Context API for global app state (user, cart, UI)
  + **Components:** ProductCard, FilterBar, CartItem, OrderSummary, etc.
  + **Responsive UI:** Mobile-first design approach
* **Backend:**
  + **Runtime/Framework:** Node.js with Express.js
  + **Structure:**
    - app.js: Main entry file
    - Routes, Controllers, Middleware
    - Models (using Mongoose)
    - Utility modules for validation, error handling
  + **APIs:** Follows RESTful conventions with secured endpoints
* **Database:**
  + **Type:** NoSQL (MongoDB)
  + **Collections:**
    - Users
    - Products
    - Cart
    - Orders
    - Categories
    - Banners
    - Admin
  + **Relations:**
    - One-to-Many: Users → Orders, Users → Cart
    - Many-to-Many: Products ↔ Categories (via references)

**4. Setup Instructions**

* **Prerequisites:**
  + Node.js (v14+), npm/yarn
  + MongoDB or MongoDB Atlas
  + Git
  + VS Code or any code editor
* **Installation:**
  + Clone the repository:  
    https://github.com/vyshu0111/ShopSell
  + **Backend Setup:**

bash

Copy code

cd ShopEZ/backend

npm install

npm start

* + **Frontend Setup:**

bash

Copy code

cd ../client

npm install

npm start

**5. Folder Structure**

* **Client:**  
  Contains all React components, pages, styles, and context for state management.
* **Server:**  
  Contains API routes, models, controllers, and middleware for backend logic.

**6. Running the Application**

* **Frontend:**  
  Run npm start inside the client directory.
* **Backend:**  
  Run npm start inside the backend directory.

**7. API Documentation**

* 🔐 **Authentication APIs:**
  + POST /register – Register a new user (admin/user/seller)
  + POST /login – Log in a user
* 📦 **Product APIs:**
  + GET /fetch-products – Get all products
  + GET /fetch-product-details/:id – Get single product details
  + POST /add-new-product – Add new product (admin only)
  + PUT /update-product/:id – Update existing product (admin only)
  + GET /fetch-categories – List all categories
  + POST /update-banner – Update homepage banner (admin only)
  + GET /fetch-banner – Fetch banner image
* 🛒 **Cart APIs:**
  + GET /fetch-cart – Get all items in cart
  + POST /add-to-cart – Add product to cart
  + PUT /increase-cart-quantity – Increase item quantity
  + PUT /decrease-cart-quantity – Decrease/remove item
  + DELETE /remove-item/:id – Remove item by ID
* 📋 **Order APIs:**
  + POST /buy-product – Direct purchase
  + POST /place-cart-order – Place order for cart
  + GET /fetch-orders – Get all orders
  + PUT /cancel-order – Cancel an order
  + PUT /update-order-status – Admin updates order status
* 👥 **User/Admin APIs:**
  + GET /fetch-users – List all users (admin only)

**8. Authentication**

* **JWT-based authentication** is planned (currently handled by session logic).
* Passwords hashed using bcrypt.
* Role-based access control:
  + admin: Full control
  + seller: Seller dashboard (future enhancement)
  + user: Product browsing and order placing

**9. User Interface**

* **Built with React and CSS. Key Pages Include:**
  + Landing Page
  + Login / Signup Page
  + Product Listing & Detail View
  + Cart Page
  + Checkout Page
  + My Orders Page
  + Admin Dashboard for Products, Orders, and Banners

**10. Testing**

* **Manual Testing:**  
  UI tested manually across different devices.
* **API Testing:**  
  Tested using Postman.
* **Automation:**  
  Planned using Jest for future versions.

**11. Known Issues**

* No email notifications implemented
* No real-time messaging or chat
* Minimal front-end validation (to be improved)

**12. Future Enhancements**

* ✉️ Email/SMS Notifications
* 🔌 Real-Time Chat with Socket.IO
* 📊 Admin Analytics Dashboard (charts, metrics)
* 📱 Full mobile responsiveness
* 🔍 Advanced Search by filters, reviews, brands
* 💬 Product Reviews & Ratings
* 🛒 Wishlist and Save for Later features