FULL STACK PROJECT IDEA

## # Multi-Vendor Shop System Documentation

**1. Introduction**

The Multi-Vendor Shop System is a full-stack web application where customers can browse and purchase products from various vendors. Vendors manage their own shops, and a superadmin oversees all operations. The system includes location-based filtering to help users find nearby shops.

**2. Technology Stack**

* **Frontend:** React (Vite), Tailwind CSS, React Router, Axios
* **Backend:** Node.js, Express.js
* **Database:** MongoDB (MongoDB Atlas)
* **Authentication:** JWT (JSON Web Tokens), bcrypt.js (password hashing)
* **State Management:** Context API / Redux
* **Additional Features:** Geolocation API, MongoDB Geospatial Queries

**3. System Features & Roles**

**User Roles:**

1. **Customer**:
   * Browse shops & products
   * Add products to cart & checkout
   * View past orders
2. **Vendor**:
   * Manage their shop (add/edit products, update stock)
   * View customer orders for their shop
3. **Superadmin**:
   * Manage all shops & vendors
   * Remove products or shops
   * Manage users (approve vendors, remove accounts)

**4. Project Roadmap**

**Phase 1: Project Setup**

1. Initialize Backend (Node.js + Express)
2. Initialize Frontend (React + Vite, Tailwind CSS)
3. Set up MongoDB Atlas database

**Phase 2: Authentication & Authorization**

1. Implement role-based authentication (JWT, bcrypt.js)
2. Create protected API routes for each role

**Phase 3: Shop & Product Management**

1. Create shops API (CRUD operations, location filtering)
2. Implement product management (vendors can only edit their own products)

**Phase 4: Customer Features**

1. Build cart & checkout system
2. Implement orders API for customers and vendors

**Phase 5: Location-Based Filtering**

1. Integrate Geolocation API for shop filtering
2. Implement MongoDB Geospatial Queries to find nearby shops

**Phase 6: Admin & Vendor Dashboard**

1. Superadmin dashboard for shop and vendor management
2. Vendor dashboard for managing products and orders

**Phase 7: Deployment & Testing**

1. Test APIs using Postman
2. Deploy backend (Render/Railway/Vercel) and frontend (Vercel/Netlify)
3. Deploy MongoDB on MongoDB Atlas

**5. Future Enhancements**

* **Real-time Order Updates** (WebSockets / Socket.io for instant notifications)
* **Payment Gateway Integration** (Stripe/Razorpay for online payments)
* **Google Maps API** (Visual representation of shop locations)
* **Admin Analytics Dashboard** (Track sales, user activities, and shop performance)

**6. Conclusion**

The Multi-Vendor Shop System provides a seamless experience for customers, vendors, and administrators. With role-based access control and location-based filtering, the platform enhances user convenience while ensuring data security and operational efficiency. Future enhancements such as real-time order updates and payment integration will further improve usability and performance.

**7. References & Tools**

* **Postman** (API Testing)
* **MongoDB Atlas** (Cloud Database Management)
* **Vercel / Netlify** (Frontend Deployment)
* **Render / Railway** (Backend Deployment)
* **GitHub** (Version Control)