**Ecommerce Multi Vendor**

**Tools**

* intellij idea ( ”CODEWITHZOSH” for three month free licence)
* Vs code
* node js
* my sql

**Technolodgy**

* spring boot
* mysql
* spring security
* java mail sender
* jwt
* react
* typescript
* redux toolkit
* mui
* tailwind css
* react chart
* formik
* yup
* react router dom
* axios
* Payment Gateway: Razorpay (for indian student) Stripe (for international student)

**Features**

**Customer Features**

1. **Chatbot for Queries**
   * A chatbot allows users to ask questions about:
     + **Order History**: View past orders.
     + **Cart**: Inquire about items in their cart.
     + **Product Details**: Ask for detailed information about products.
2. **Product Management**
   * **Fetch Product List**: Users can browse through available products.
   * **Filter & Sort**: Filter products by categories, price, etc., and sort them by price.
   * **Pagination**: Display products in multiple pages to improve performance and user experience.
   * **Product Details**: View detailed information about a specific product.
3. **Cart Management**
   * **Add Item to Cart**: Add products to the shopping cart.
   * **Update Cart Item**: Modify item quantities or remove items from the cart.
4. **Checkout Process**
   * **Apply Coupon**: Users can apply discount coupons to their cart.
   * **Add New Address**: Add and manage shipping addresses during checkout.
   * **Payment Gateways**: Checkout using payment options like Razorpay or Stripe.
5. **Order History**
   * **View Past Orders**: Users can see a list of their previous purchases and order details.
   * cancel order
6. **User Account Management**
   * Manage personal details, view order history, and track account settings.
7. Review & Rating
   * Write Review
8. wishlist
   * add and remove product from wishlist

**Seller Features**

1. **Seller Dashboard**
   * Earning Graph (Today, last 7 days, last 12 month), and seller Report
2. **Seller Reports**
   * **Total Sales**: View the total number of products sold.
   * **Total Earnings**: Track overall earnings from sales.
   * **Refunds & Cancellations**: Monitor refunds and canceled orders.
3. **Product Management**
   * **Create Products**: Add new products to the store.
   * **Orders Management**: View and manage customer orders.
4. **Payment & Transactions**
   * **Track Payments**: Monitor incoming payments for orders.
   * **Transaction History**: Detailed history of all transactions.
5. **Seller Account Management**
   * **Profile Management**: Update and manage seller profiles.

**Admin Features**

1. **Admin Dashboard**
2. **Seller Management**
   * Handle all sellers, including approval, and suspension.
3. **Coupon Management**
   * **Create, Edit, Delete Coupons**: Manage discount codes available for customers.
4. **Home Page Management**
   * Update and customize the home page through the admin panel.
5. **Deal Management**
   * Create and manage promotional deals and offers on products.

**Entity Relationships**

1. **User**
   * One-to-Many with **Address**: A user can have multiple addresses.
   * Many-to-Many with **Coupon**: Users can use multiple coupons, and a coupon can be used by multiple users.
   * One-to-Many with **Cart**: Each user has one cart.
   * One-to-Many with **Order**: A user can have multiple orders.
   * One-to-Many with **Review**: A user can leave multiple reviews.
   * One-to-Many with **Transaction**: A user can have multiple transactions.
   * One-to-One with **Wishlist**: Each user has one wishlist.
2. **Address**
   * Many-to-One with **User**: An address belongs to one user.
   * Many-to-One with **Order**: An order has one shipping address.
3. **Cart**
   * One-to-One with **User**: Each user has one cart.
   * One-to-Many with **CartItem**: A cart can contain multiple cart items.
4. **CartItem**
   * Many-to-One with **Cart**: A cart item belongs to one cart.
   * Many-to-One with **Product**: A cart item refers to one product.
5. **Product**
   * Many-to-One with **Category**: A product belongs to one category.
   * Many-to-One with **Seller**: A product is sold by one seller.
   * One-to-Many with **Review**: A product can have multiple reviews.
6. **Category**
   * Many-to-One with **Category**: A category can have a parent category (for subcategories).
7. **Coupon**
   * Many-to-Many with **User**: A coupon can be used by multiple users.
8. **Order**
   * Many-to-One with **User**: An order belongs to one user.
   * One-to-Many with **OrderItem**: An order can have multiple order items.
   * Many-to-One with **Address**: An order has one shipping address.
9. **OrderItem**
   * Many-to-One with **Order**: An order item belongs to one order.
   * Many-to-One with **Product**: An order item refers to one product.
10. **PaymentOrder**
    * Many-to-One with **User**: A payment order belongs to one user.
    * One-to-Many with **Order**: A payment order can include multiple orders.
11. **Seller**
    * One-to-One with **Address**: A seller has one pickup address.
    * One-to-Many with **Product**: A seller can sell multiple products.
    * One-to-Many with **Transaction**: A seller can be involved in multiple transactions.
12. **Transaction**
    * Many-to-One with **User**: A transaction is associated with one user.
    * Many-to-One with **Seller**: A transaction is associated with one seller.
    * One-to-One with **Order**: A transaction corresponds to one order.
13. **Review**
    * Many-to-One with **Product**: A review is for one product.
    * Many-to-One with **User**: A review is written by one user.
14. **Wishlist**
    * One-to-One with **User**: Each user has one wishlist.
    * Many-to-Many with **Product**: A wishlist can contain multiple products.
15. **VerificationCode**
    * One-to-One with **User**: A verification code can be associated with one user.
    * One-to-One with **Seller**: A verification code can be associated with one seller.
16. **SellerReport**
    * One-to-One with **Seller**: A report corresponds to one seller.

**Connect Frontend With Backend**

**User Interaction**

* User makes a request (e.g., button click) in the React component.

**API Call**

* React component sends a request to the backend API.

**Backend Processing**

* Backend processes the request and retrieves data from the database or other source.

**Response from API**

* Backend sends the requested data back to the React component as a response.

**Store Data in Local State**

* React component receives the response and stores the data in local state (using useState or similar).

**Render Data**

* React component renders the data in the UI.

**Project Step**

* set up mysql database
* create first api
* create entity model
* spring security (login, signup)
* service and impl
* controller
* test all api end points