**🛍️ E-Commerce Web Application – Explanation Document**

**📄 Project Overview**

This project aims to develop a **mini e-commerce web application** where users can browse products, add them to a shopping cart, and make secure payments online using Razorpay. The system includes user authentication, role-based access control, and an admin panel for managing products and orders.

**🎯 Main Features**

**✅ User Module**

* **Registration & Login**
  + Users can sign up and log in securely.
  + Passwords are stored using hashing for security.
* **Role-based access**
  + **Customers**: can browse products, manage their cart, place orders, and view order history.
  + **Admins**: can manage (add/edit/delete) products and manage all orders.
* **Profile management**
  + Users can update their information.

**✅ Product Module**

* **Product catalog**
  + Display list of all products with details (name, description, price, image).
* **Search and browse**
  + Customers can search for products and view details.
* **Admin product management**
  + Add new products.
  + Update or delete existing products.

**✅ Cart Module**

* Customers can:
  + Add products to their cart.
  + Update product quantities.
  + Remove products from the cart.
* **Cart persistence**
  + The cart remains available during the session until checkout.
* **Total price calculation**
  + Automatically updates as items are added or removed.

**✅ Order Module**

* **Checkout process**
  + Create an order from the cart items at checkout.
* **Order tracking**
  + Status such as: Created, Paid.
* **Order history**
  + Customers can view their own past orders.
  + Admins can view and manage all orders in the system.

**✅ Payment Module**

* **Razorpay integration**
  + Backend creates a payment order with Razorpay and shares payment details with the frontend.
  + Frontend displays Razorpay payment popup for secure payments.
  + After payment, the backend verifies the payment status and updates the order to “Paid”.

**💻 Tech Stack**

| **Component** | **Technology** |
| --- | --- |
| Backend | Spring Boot |
| Frontend | Thymeleaf or React |
| Database | MySQL |
| Security | Spring Security |
| Payments | Razorpay API |
| API Docs | OpenAPI / Swagger |
| Deployment | Render / Railway / Netlify / Vercel |

**🔐 Security**

* **Spring Security**
  + Handles login, registration, and role-based authorization.
  + Restricts admin functionality (e.g., product management) to admins only.
* **Password encryption**
  + User passwords are hashed before being stored in the database.

**⚙️ Testing**

* **Unit Testing**
  + JUnit 5 and Mockito used to write tests for Controllers, Services, and Repositories.
* **End-to-End Testing**
  + Simulate complete user flows: registration, product browsing, cart operations, checkout, and payment.

**🚀 Deployment**

* **Backend**
  + Deployed to Render, Railway, or AWS EC2 (mandatory).
* **Frontend**
  + Deployed to Netlify or Vercel (mandatory).

**📦 Project Structure**

/backend

├── src/main/java/com/example/ecommerce

├── src/main/resources

├── pom.xml

/frontend (if using React)

├── src

├── public

├── package.json

**📝 Submission Instructions**

* **Code Submission**
  + Push complete code to a public GitHub repository.
  + Separate folders: /backend and /frontend.
* **Deployment**
  + Deploy backend and frontend to the specified platforms.
* **Optional**
  + Record a 1–2 minute walkthrough video of your application and upload it to Google Drive with public access.

**🎥 Optional Demo Video**

A short video helps demonstrate your app to reviewers and recruiters. You can include:

* Login and register demo.
* Browsing products.
* Adding to cart and checking out.
* Payment process (using test mode).
* Admin product management.

**✅ Conclusion**

This project provides a **complete, real-world mini e-commerce solution**, integrating backend, frontend, security, and payment processing. It demonstrates key skills including Spring Boot development, database modeling, API design, secure authentication, payment gateway integration, and deployment.