***YallaMarket***

Youssef Elraggal 7806

Youssef Magdy 7825

Bishoy Adel 7824

Zeyad Shawki 7383

**1) Functional and Non-Functional Requirements**

**Functional Requirements**

* **User Management**: User registration, authentication, and role-based access control for Admin, Vendor, Retail Store, and Customer roles
* **Authentication System**: JWT-based authentication with email verification
* **Product Management**: Vendors can create, update, and manage products with categories, pricing, and inventory
* **Order Processing**: Complete order lifecycle management with status tracking
* **Payment System**: Multiple payment methods with status tracking and refund capabilities
* **Delivery Tracking**: Multi-stage delivery status monitoring
* **Rating System**: User and vendor rating functionality

**Non-Functional Requirements**

* **Security**: Spring Security with JWT authentication and role-based authorization
* **Scalability**: PostgreSQL database with connection pooling and batch processing
* **Performance**: Hibernate optimizations and SQL logging for debugging
* **Reliability**: Email notifications and verification system

**2) Requirements Analysis**

**User Requirements**

* **Vendors:** Need to manage products, track orders, and view sales statistics
* **Retail Stores:** Require order placement, inventory management, and paymentprocessing**,** Need product browsing, order placement, and delivery tracking
* **Administrators:** System management, user oversight, and payment monitoring

**System Requirements**

* **Runtime Environment:** Java 17 with Spring Boot 3.4.5
* **Database:** PostgreSQL with JPA/Hibernate ORM
* **Security:** JWT tokens with configurable expiration
* **Email Service:** SMTP integration for notifications

**3) Software Process**

**Suggested Process: Agile/Iterative Development**

**Phases Division**:

1. **Requirements Analysis & Planning** - User story definition and system design
2. **Core Infrastructure** - Database schema, security, and authentication
3. **User Management Module** - Registration, login, and role management
4. **Product Management Module** - CRUD operations for vendors
5. **Order Processing Module** - Order lifecycle and status management
6. **Payment Integration** - Payment processing and refund system
7. **Testing & Quality Assurance** - Unit, integration, and system testing

**4) Architectural Design**

**System Architecture: Layered N-Tier Architecture**

* **Presentation Layer**: REST Controllers and Web Controllers
* **Business Logic Layer**: Service layer with business rules
* **Data Access Layer**: JPA repositories and entities
* **Database Layer**: PostgreSQL with entity relationships

**Application Architecture: Spring Boot MVC**

* **Model**: JPA entities representing business objects
* **View**: Thymeleaf templates and REST API responses
* **Controller**: Request handling and response formatting
* **Configuration**: Security, database, and application configurations

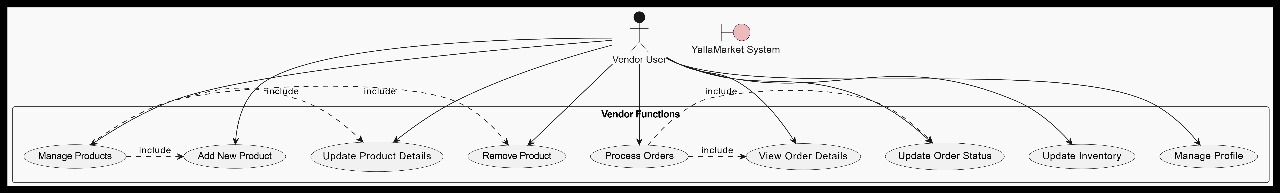
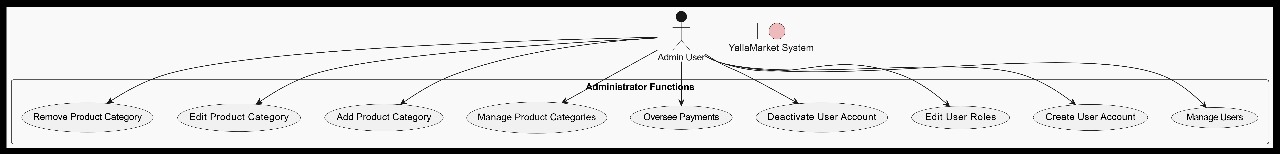
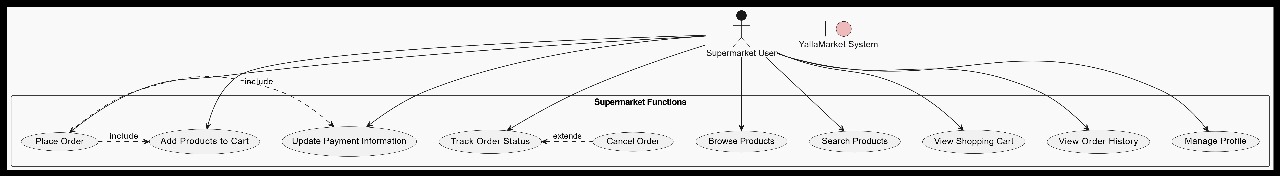
**5) Design & Implementation**

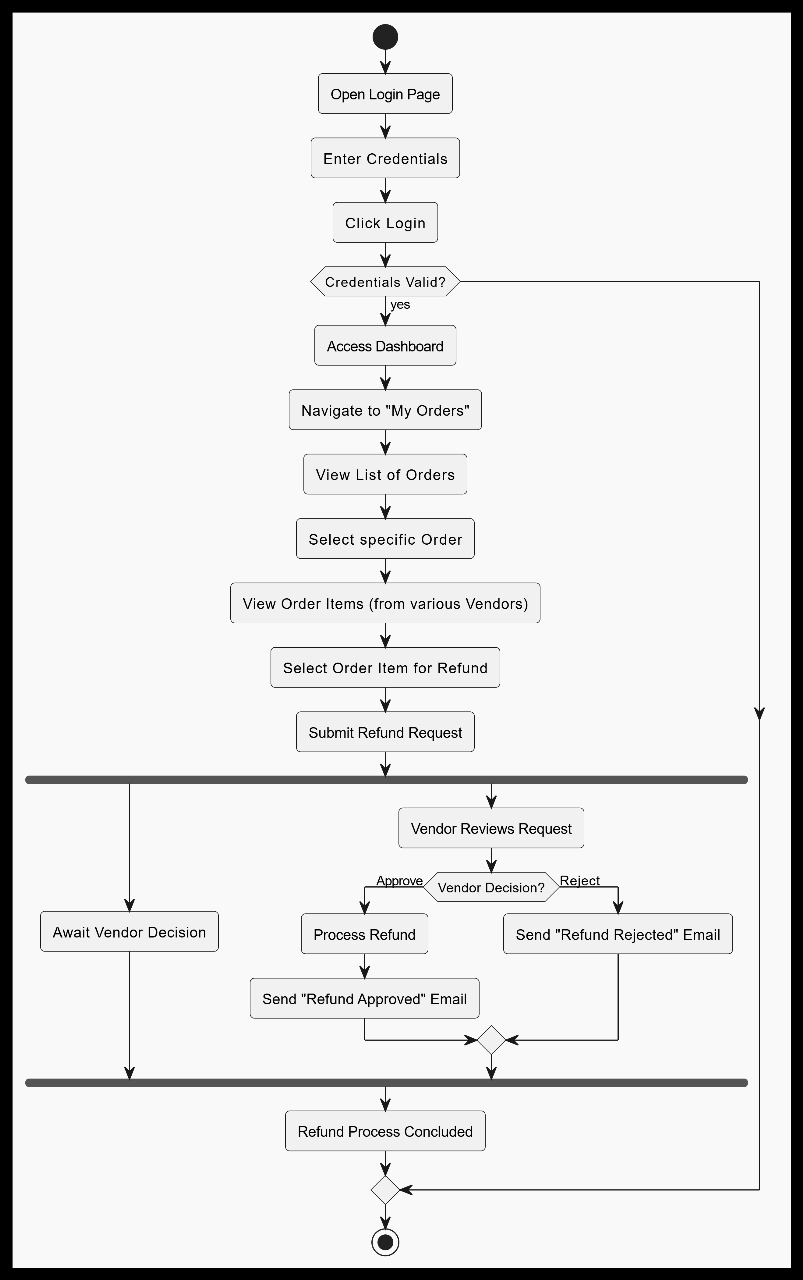
**Design Description**

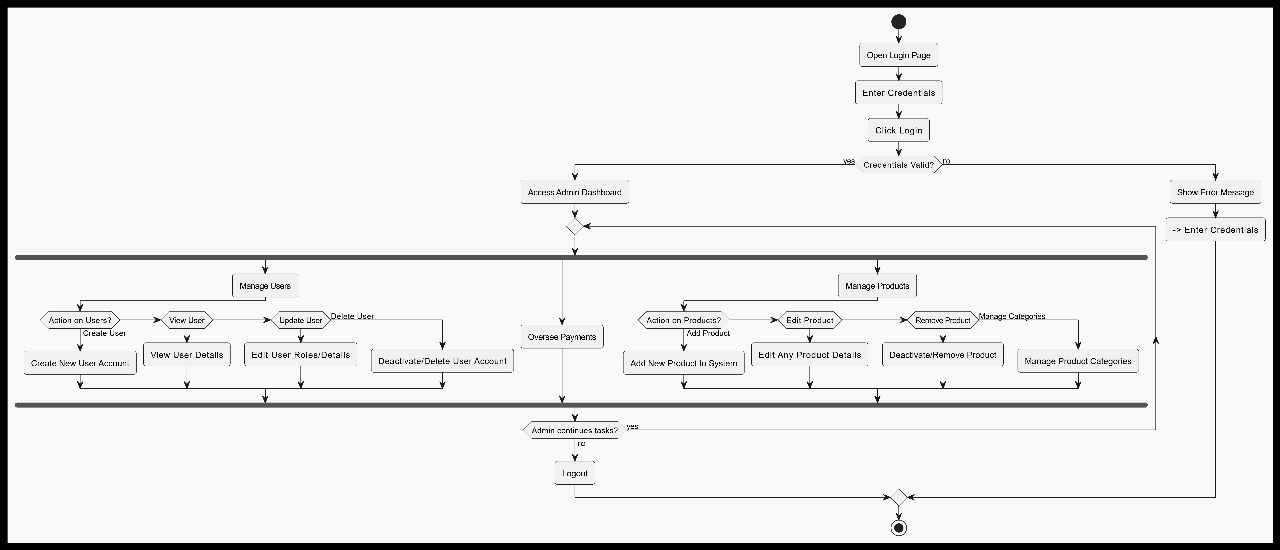
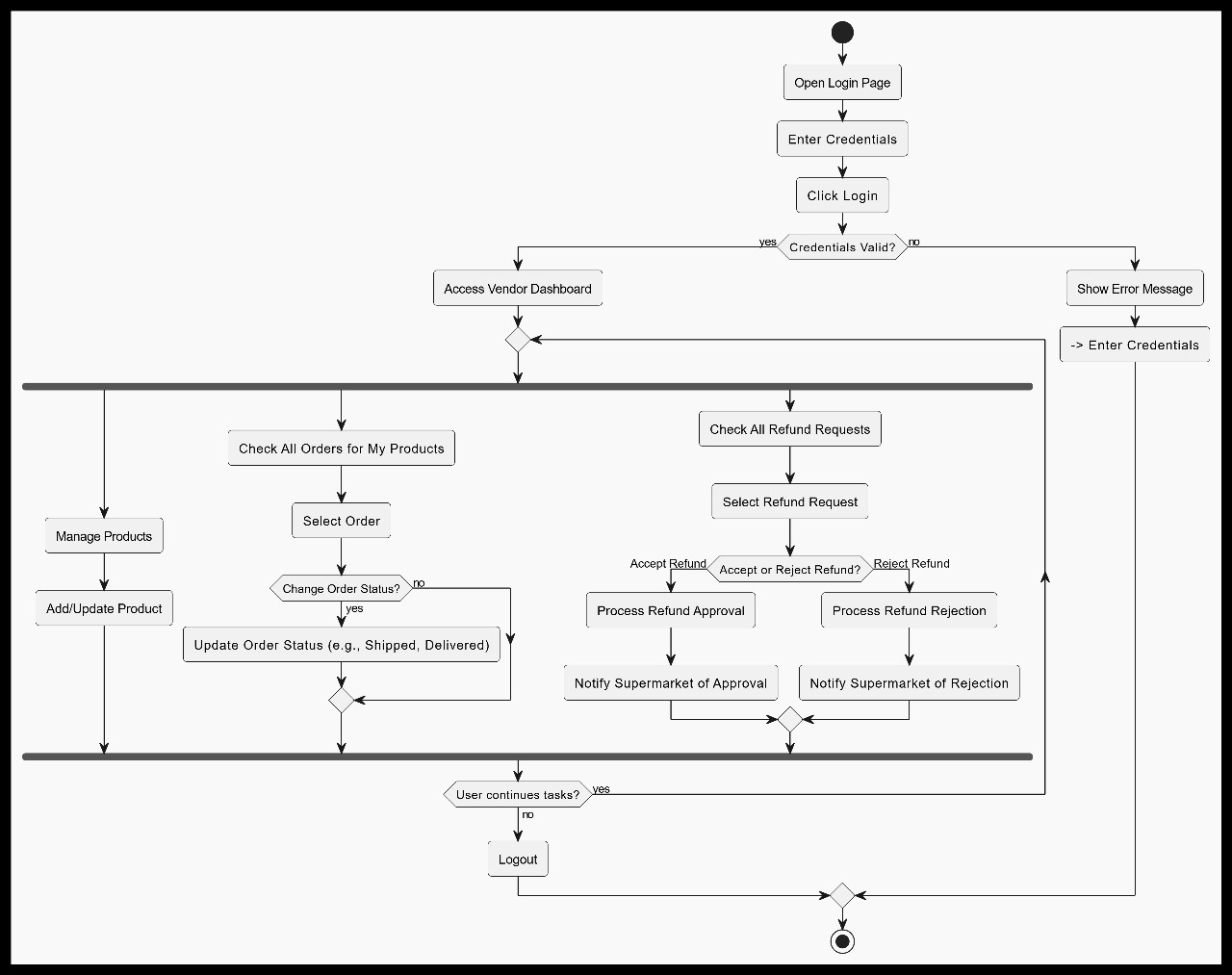
The system follows Domain-Driven Design principles with clear separation of concerns. Entity relationships are managed through JPA annotations with proper handling of circular references

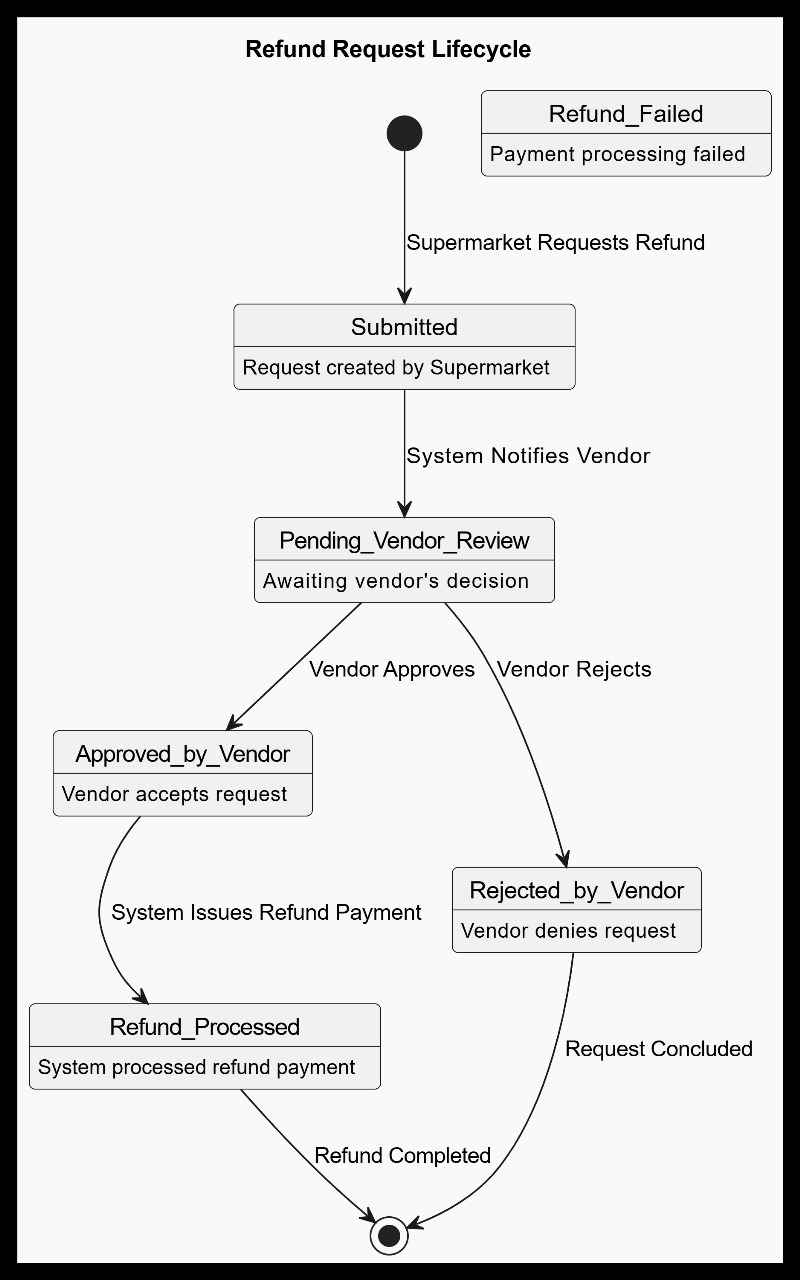
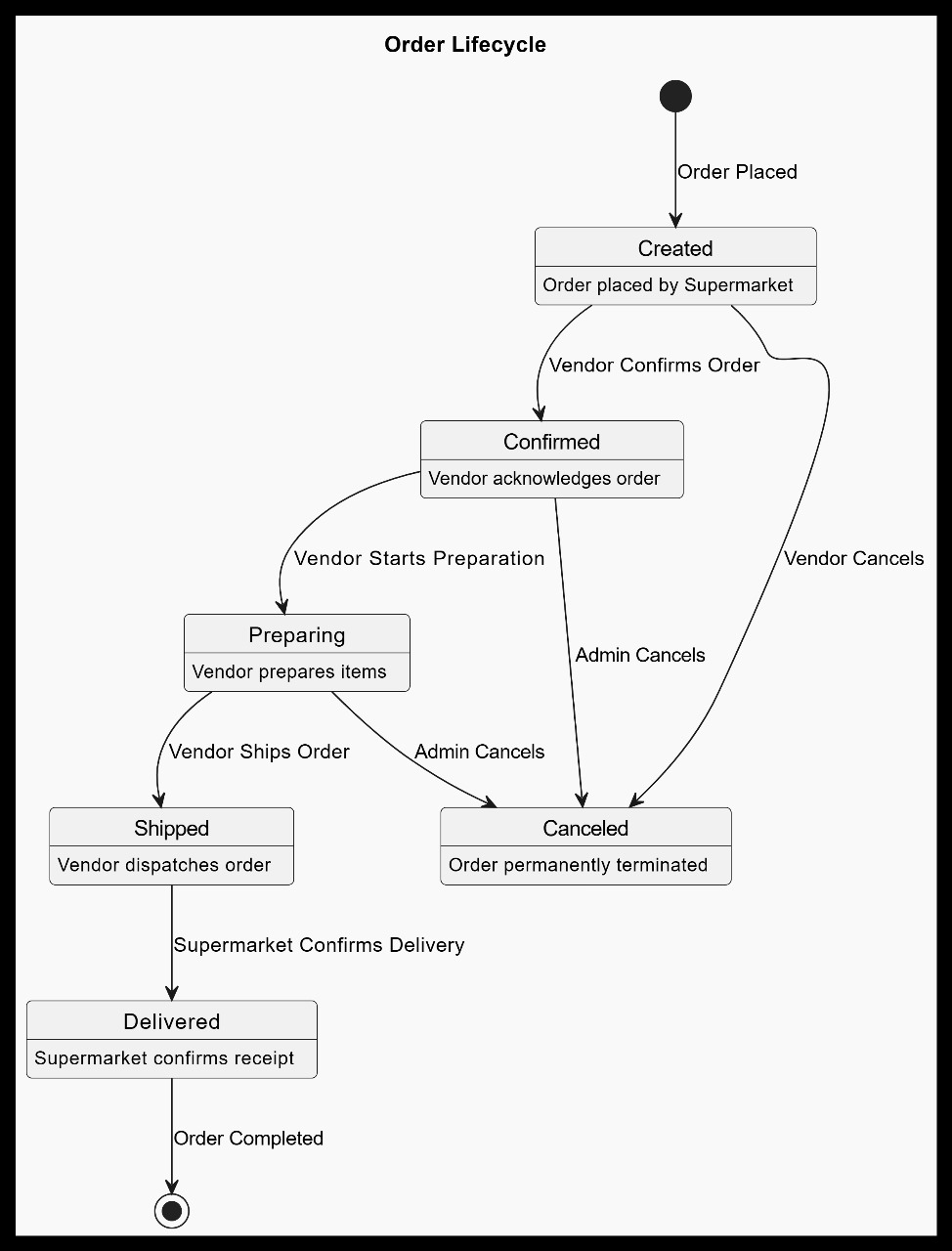
**Development Environment & Coding (Implementation)**

* **Build Tool**: Maven with Spring Boot parent
* **Dependencies**: Spring Boot Starters for Web, Security, JPA, and Mail
* **Code Generation**: Lombok for reducing boilerplate code
* **Application Entry Point**: Standard Spring Boot application class

**Use Case Diagrams**



A screenshot of a computer

AI-generated content may be incorrect.

A computer screen shot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A computer screen shot of a diagram

AI-generated content may be incorrect.