**E-Commerce Application Report**

**1. User Documentation**

**Overview**

The application is a console-based E-Commerce system that enables users to register, log in, and perform actions based on their role (Buyer, Seller, or Admin). It provides a role-specific menu, allowing Buyers to browse products, Sellers to manage their product listings, and Admins to manage users and monitor the system.

**Classes and Their Responsibilities**

1. **Main**
   * Entry point for the application.
   * Handles user interactions through a menu-driven interface.
   * Delegates operations to relevant services (UserService, ProductService).
2. **User**
   * Represents a generic user with attributes like username, email, and password.
   * Subclasses:
     + **Buyer**: For browsing and searching products.
     + **Seller**: Manages product-related operations (add, update, delete).
     + **Admin**: Manages users and reviews all system data.
3. **Product**
   * Represents products with attributes like name, price, quantity, and seller ID.
4. **UserService**
   * Handles user-related operations such as registration, login, and user role validation.
5. **ProductService**
   * Handles product-related operations such as CRUD functionality and browsing.

**How to Use**

1. Start the application by running the Main class.
2. Use the menu to:
   * Register as a Buyer, Seller, or Admin.
   * Log in with your credentials.
3. Based on your role, access specific functionalities:
   * **Buyers**: Browse and search products.
   * **Sellers**: Add, update, delete, and view their products.
   * **Admins**: Manage users, view system-wide data.

**Class Diagram**

The following is the association between the main classes (use the provided uploaded diagrams if applicable for the visual representation):

* **Main** interacts with **UserService** and **ProductService**.
* **User** subclasses into **Buyer**, **Seller**, and **Admin**.
* **ProductService** manages **Product** operations.
* **UserService** handles user-related data and operations.

**2. Development Documentation**

**Source Code Directory Structure**

**src/**

**│**

**│── com/**

**│ │── ecommerce/**

**│ │ │──** Main.java

**│ │ │── model/**

**│ │ │ │──** User.java

│ │ │ **│**── Buyer.java

│ │ │ **│**── Seller.java

│ │ │ **│**── Admin.java

│ │ │ └── Product.java

**│ │ │── service/**

**│ │ │ │**── UserService.java

│ │ │ └── ProductService.java

**│ │ └── dao/**

**│ │ │**── UserDAO.java

│ │ └── ProductDAO.java

**│**

**└──── resources/**

└── database.sql

**Build Process**

1. **Dependencies**
   * Maven is used to manage dependencies. Add the following to pom.xml:

<dependencies>

<dependency>

<groupId>org.postgresql</groupId>

<artifactId>postgresql</artifactId>

<version>42.3.6</version>

</dependency>

<dependency>

<groupId>org.mindrot</groupId>

<artifactId>jbcrypt</artifactId>

<version>0.4</version>

</dependency>

</dependencies>

1. **Compilation**

* Use Maven to build the project:

mvn clean package.

1. **Database Setup**

* Create a PostgreSQL database using the resources/database.sql file. Example schema:

CREATE TABLE users (

id SERIAL PRIMARY KEY,

username VARCHAR(255) NOT NULL,

email VARCHAR(255) UNIQUE NOT NULL,

password VARCHAR(255) NOT NULL,

role VARCHAR(50) NOT NULL

);

CREATE TABLE products (

id SERIAL PRIMARY KEY,

name VARCHAR(255) NOT NULL,

price DECIMAL(10, 2) NOT NULL,

quantity INT NOT NULL,

seller\_id INT REFERENCES users(id)

);

1. **Development Standards**

* Follow Java naming conventions for classes and methods
* Use meaningful variable names and consistent indentation.
* Separate concerns by ensuring that DAO, Service, and Model layers are distinct.

**Setting Up Development Environment**

1. Install **Java JDK** (version 13).
2. Install **PostgreSQL** and create the required database.
3. Clone the repository:

git clone <repository\_url>

cd <repository\_directory>

Configure the application.properties file:

db.url=jdbc:postgresql://localhost:5432/<database>

db.username=<db\_username>

db.password=<db\_password>

**3. Deployment Documentation**

**Installation Steps**

1. **Install Prerequisites**
   * Install Java (13).
   * Install PostgreSQL.
2. **Database Configuration**
   * Use the provided SQL script (resources/database.sql) to set up the database.
3. **Run the Application**
   * Compile and run using Maven:

*-bash-*

mvn clean install

java -jar target/ecommerce-application.jar

1. **Accessing the Application**
   * Interact with the console-based interface by running the Main class.

**Environment Requirements**

* **Operating System**: Windows, macOS, or Linux.
* **Java Runtime**: JRE 13.
* **Database**: PostgreSQL.