Project business details.

The main objective of this project is to create an application E-Commerce for Fashion industry where it provides an online product catalogue about commercial products.

Customers will be able to surf, choose, pay, and ask for delivery on the purchased products.

Merchants will be able to add their product online which can be surfed by the customers.

This new system should provide the following functionalities:

1. Registration for Customer (Customer information, billing details, delivery address)
2. Registration for Merchant (Merchant information, Product details, …)
3. Customer Login
   1. Customers add products to baskets.
   2. Customers remove products from basket.
   3. Customers update product quantities in basket.
   4. Basket Total calculation
   5. Choosing a delivery method
   6. Order management (list of orders, order cancellation)
   7. Payment method management
4. Merchant Login
   1. Merchant inserts product information in the application and choose the price of the item.
   2. Merchant can see his item and he can cancel it.
   3. The sold item the money goes to the merchant account.

Setup env

1. Create back-end application using java spring boot (IntelliJ) downloaded from JetBrains.
   1. Enter IntelliJ go to Fille click at NEW then click on Project…
   2. It opens a page new project choose Language java, built system Maven, JDK adopt-11 V.18
   3. The maven package will be created itself.
   4. Go to pom.xml at this dependencies to run the spring
      1. To start an application we are using (<spring.version>3.0.0</spring.version>) (<https://mvnrepository.com/artifact/org.springframework.boot/spring-boot-starter-web>)

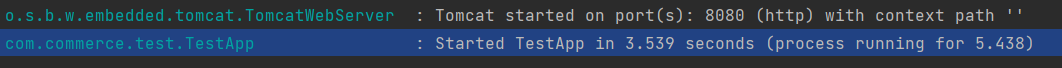
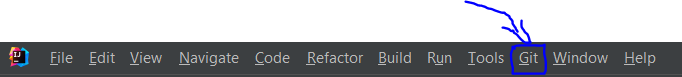
<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

<version>${spring.version}</version>

</dependency>

* + 1. Then run the application.
  1. GIT
     1. Create repository in GitHub for the project and the steps is shown in the GitHub.
     2. Use GIT in IntelliJ to be connected to GitHub.
     3. Choose which link to be connected to in GitHub.

1. Create front-end application using angular (WebStorm) from JetBrains
   1. Change the configuration from localhost -> 127.0.0.1
   2. Run the application by ng serve in the Terminal.
2. Create database PostgreSQL (Data Grip) from JetBrains
3. Create Flyway spring boot
   1. Put the flyway package in maven
   2. Then put another package for JPA in maven so it can read it
   3. After create file application.properties in resources
      1. Inside the file depend in what database(PostgreSQL) you are using:

spring.datasource.url=jdbc:postgresql://localhost:5432/<Database>

spring.datasource.username=

spring.datasource.password=

spring.flyway.url=jdbc:postgresql://localhost:5432

spring.flyway.user=

spring.flyway.password=

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect

spring.jpa.hibernate.ddl-auto=create

* + 1. After create an SQL file in resource and create a package db.migration and put inside it this will make the SQL that you wrote before saved in the history of flyway even you delete the database or lost it when you run the system it will be inserted again.

Use cases diagrams:

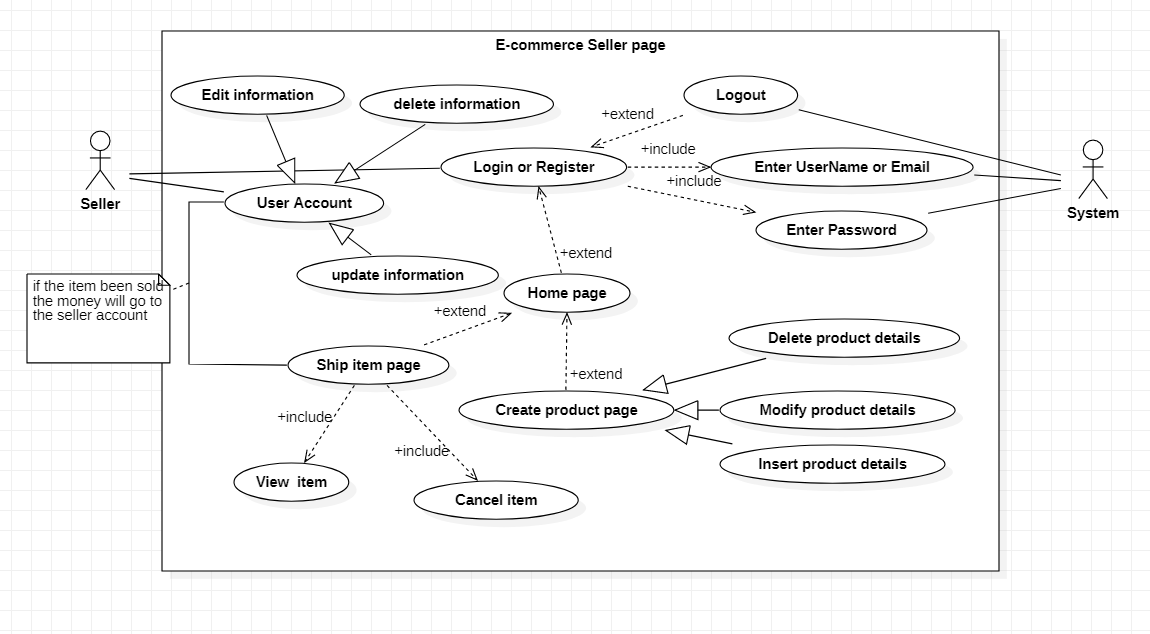
Customer (Buyer):

Diagram

Description automatically generated

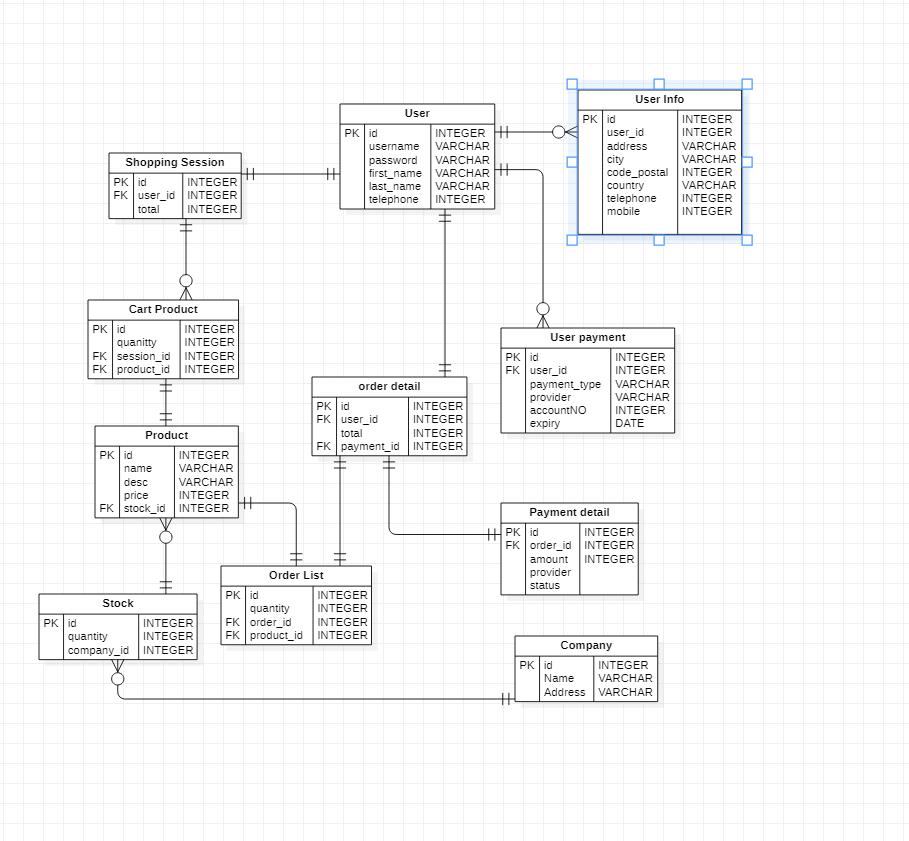
1. Customer registers or login to the System
2. After login using the username and password it will take him to the Home Page
3. From the homepage the user can search and view products.
4. Clicking on the product it will take the user to another page which will give more detailed information (size, style, price …) about the product and then he can add it to the basket.
5. From the Homepage the user can go to the “**Go to basket** “where he can add another products or update quantity or delete it
6. In the Go to basket the user clicks on check-in to finalize his product. it will take him to another page View order List to see all the product that he wants to buy with total price, and he can choose the delivery method (Home Delivery, Pickup Delivery or Drive) by his address that he putted in his account, or he can cancel the order.
7. After check in the order list, he click on the proceed the payment to take him to another page which he will choose payment method.
8. The user can manage his account by modifying, editing or deleting profile.

Merchant (Seller) Later:



1. Seller registers or login to the System
2. After login using the username and password it will take him to the Home Page
3. From the homepage can go to **Create product page** he can insert or update or delete product information
4. The product will be on the ship item page where the user can view his product, or he can cancel it.
5. If the product been sold on the ship item page, they will go to the user account.

Model diagram:



DAO and CRUD operation Junit Test

1. Create a class stock and product.
2. Create conf.properties create(URL, user, password) in resources
3. Create a class Configuration where I create singleton instance synchronized method and methods for the database for the url, user and password to be not shown.
4. Create a class ApplicationConfiguration where I use spring (Bean) to be connect to class Configuration.
5. Create class DAO for StockDao and ProductDAO with the connection of datasource of ApplicationConfiguration and create CRUD methods to initialize the testing method.
6. Comment: Create DatabaseManager for the DB connection.
7. Then do the integration test to see if it is working by using HikaryDatabse to be connected to conf.properties and ApplicationConfiguration where I inject DataSource from it.