**“VigorX”**

**"Jump into the world of online sportswear shopping where you can snag great deals from the comfort of your couch. For some, it's like a daily victory lap, while others are still warming up to the idea."**

**Table of Contents:**

1. **Introduction………………………………………………………………3**
2. Overview of the E-Commerce Platform**……………………………………3**

**2.2. Objective:……………………………………………………………3**

**2.3. Target Audience:……………………………………………………3**

**3.Product Scope:……………………………………………………………5**

**4.Functional Requirements:……………………………………………6**

**5.User Roles and Permissions:…………………………………………7**

**6.System Architecture:……………………………………………………7**

**7.Technology Stack:………………………………………………………8**

**8.Main Functionalities:………………………………………………8**

**9.UML Diagrams:…………………………………………………………9**

**1. Use Case Diagram -----------------------------------------------------------------------------------------------10**

**2. Activity Diagrams-----------------------------------------------------------------------------------------------14,13,12,11**

**3. Class Diagram -------------------------------------------------------------------------------------------------------16**

**3.1 OCL Constraints Description ------------------------------------------------------------------17**

**4.Sequence Diagram……………………………………………………………..18**

**5.ERD …………………………………………………………………………………..19**

**“Project Title” :**

**{ VigorX e-commerce for sporty clothes }**

**“Project Description “ :**

**online e-commerce platform dedicated to providing a wide range of sporty clothes and products. Aimed at fitness enthusiasts and athletes, the platform offers an extensive collection of high-quality, durable, and stylish sports apparel, .**

1. **INTRODUCTION**
   1. **Purpose:**

**This document aims to provide a comprehensive guide to the development, deployment, and maintenance of an e-commerce platform focused on selling sporty clothes. It serves as a reference for developers, designers, project managers, and stakeholders involved in the creation and management of the platform ,where the document provide as overall description about our e-commerce platform system with UML.**

1.2.1 Overview of the E-Commerce Platform:

**The e-commerce platform for sporty clothes aims to provide a user-friendly and engaging online shopping experience for individuals seeking high-quality sportswear tailored to various athletic activities. With a focus on functionality, performance, and aesthetics, the platform endeavors to meet the diverse needs of sports enthusiasts, athletes, and fitness-conscious individuals**

**1.2.2 Objectives:**

**1.Offer High-Quality Sportswear:**

**The primary objective of the platform is to offer a wide range of sporty clothes made from premium materials, designed to enhance performance and comfort during physical activities.**

**2.Provide a Seamless Shopping Experience:**

**The platform seeks to provide a seamless and intuitive**

**shopping experience, allowing users to browse, search, and**

**purchase sportswear effortlessly.**

**1.2.3 Target Audience:**

**The target audience for the e-commerce platform**

**includes:**

**Sports Enthusiasts: Individuals passionate about various sports and fitness activities, seeking performance-oriented sportswear to enhance their training and competition experiences.**

**Athletes: Professional and amateur athletes looking for specialized sportswear designed to meet the demands of their specific sports disciplines.**

**Fitness Conscious Individuals: Individuals committed to leading active and healthy lifestyles, seeking stylish and functional sportswear for their workouts, gym sessions, and outdoor activities.**

**1.3 Product Scope:**

* **VigorX is an e-commerce platform that covers a wide range of sporty clothes tailored to meet the diverse needs of users engaged in various sports and fitness activities. Each category includes specific examples of products designed to provide comfort, performance, and style during athletic pursuits.**

**Sporty Clothes:**

1. **Tops:**

* **T-Shirts**
* **Tank Tops**
* **Long Sleeve Tops**

1. **Bottoms:**

* **Shorts**
* **Leggings and Tights**

1. **Outerwear:**

* **Jackets**
* **Hoodies and Sweatshirts**
* **Vests**

1. **Functional Requirements**

**1. User Management**

**Registration: users will be able to create accounts by providing basic information such as name, email, and password.**

**Login: Provide a secure login mechanism for registered users to access their accounts.**

**Profile Management: users will be able to view and update their personal information, including shipping addresses and payment methods.**

**Guest Checkout: Enable guest users to make purchases without creating an account.**

**Cancel Orders: Allow users to cancel orders within a specified timeframe before shipment.**

**3.Product Management:**

**Product Listing: Display sporty clothes with detailed information, including images , prices, and available sizes/colors.**

**Product Recommendations:Provide personalized product recommendations**

1. **Shopping Cart and Checkout:**

**Add to Cart: Allow users to add products to their shopping cart for later purchase.**

**View Cart: Enable users to view the contents of their shopping cart, including product details, quantities, and total price.**

**Update Cart: Allow users to update the quantities of products in their cart or remove items.**

**Guest Checkout: Support guest users to proceed to checkout without creating an account.**

1. **Admin Panel:**
2. **The system should allow admin to login.**
   1. **The admin shall enter his id and password**.
3. **The system should allow the admin to view and update his profile.**

**3. The system should allow the admin to log out**

**Product Management: Allow administrators to add, edit, and delete products, including uploading product images and updating inventory quantities.**

**Order Management: Provide an interface for administrators to view and manage orders, update order statuses, and process returns and refunds.**

**User Management: Allow administrators to view user accounts, edit user information, and manage user roles and permissions.**

**3.User Roles and Permissions**

* **Administrator**
* **Registered User**

**4.System Architecture**

* **Frontend Architecture (Angular)**
* **Backend Architecture (Spring Boot)**
* **Database Schema**

**5.Technology Stack**

**Frontend Technologies:**

* **Angular**
* **HTML/CSS/JavaScript**
* **Bootstrap/Tailwind CSS**

**Backend Technologies:**

* **Spring Boot**
* **Java**
* **RESTful API**

**Database Technologies:**

* **MySQL**

**Other Tools and Libraries.**

**6.Main Functionalities**

**[User]:**

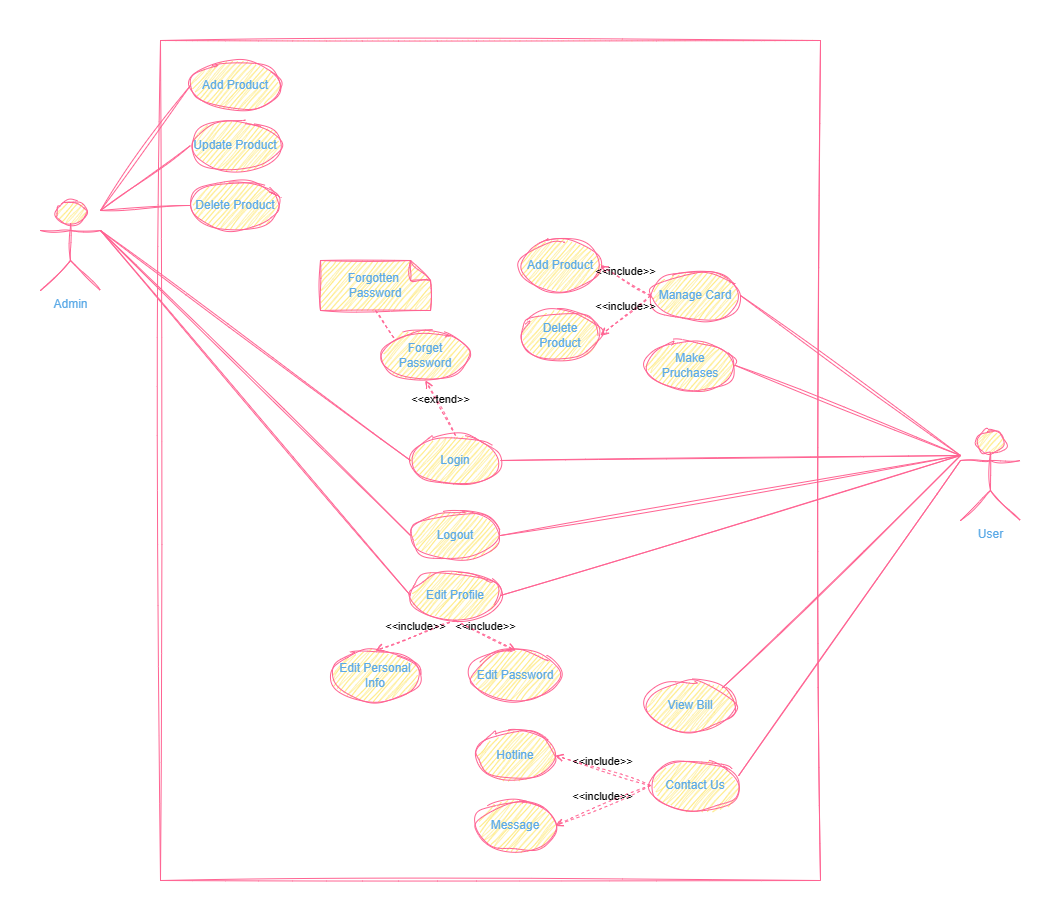
* **Login**
* **Logout**
* **Explore product**
* **Placed order**

**[Admin]:**

* **Manage Product"Add, Delete,Update".**
* **Manage Users.**
* **Login**
* **Logout**

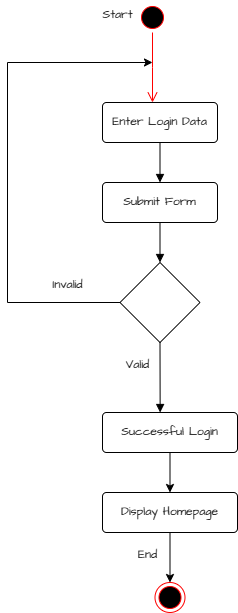
**[ UML Diagrams ]**

**1.UseCase Digram**

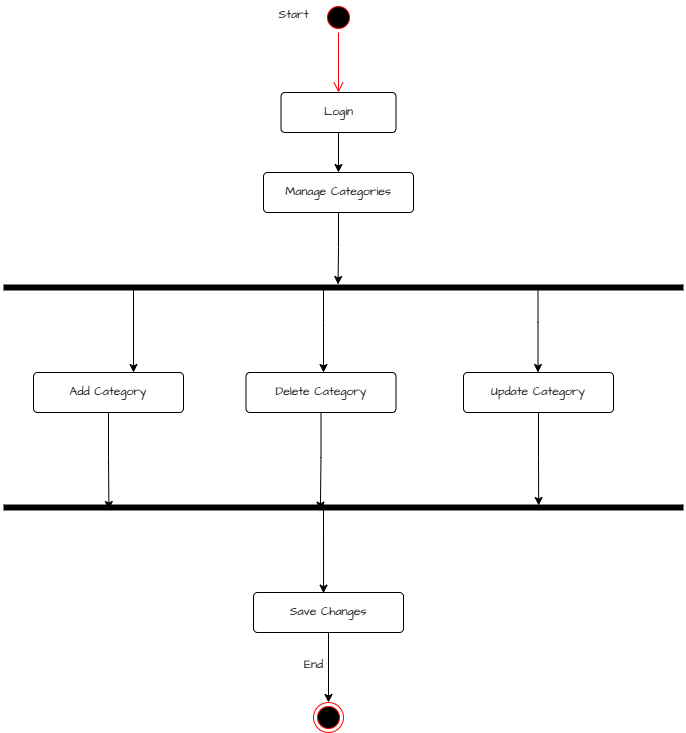
****

**2.Activity Diagrams**

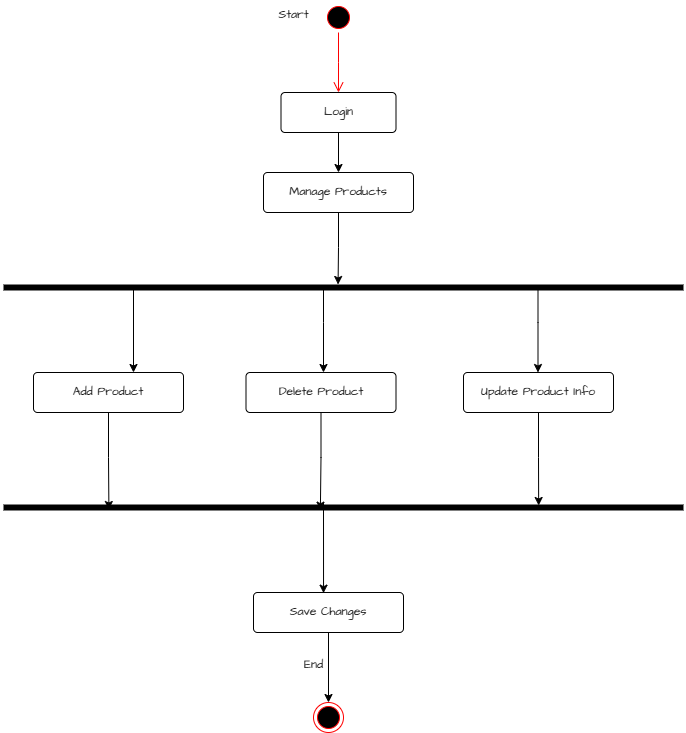
1. **User :Login**

****

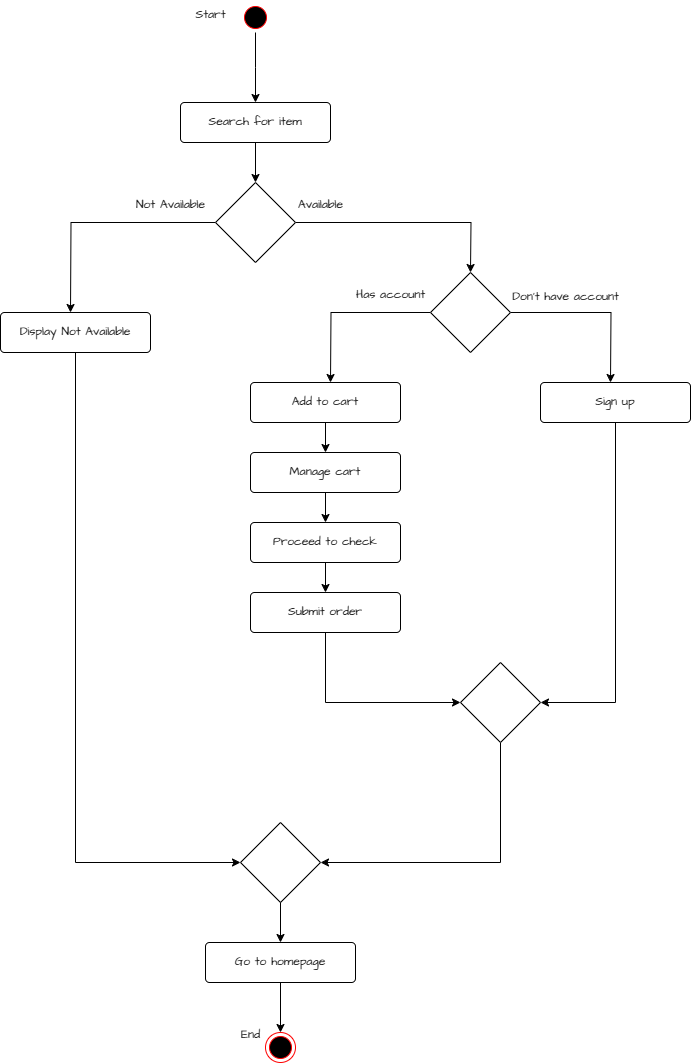
1. **Admin :Manage Categories**

****

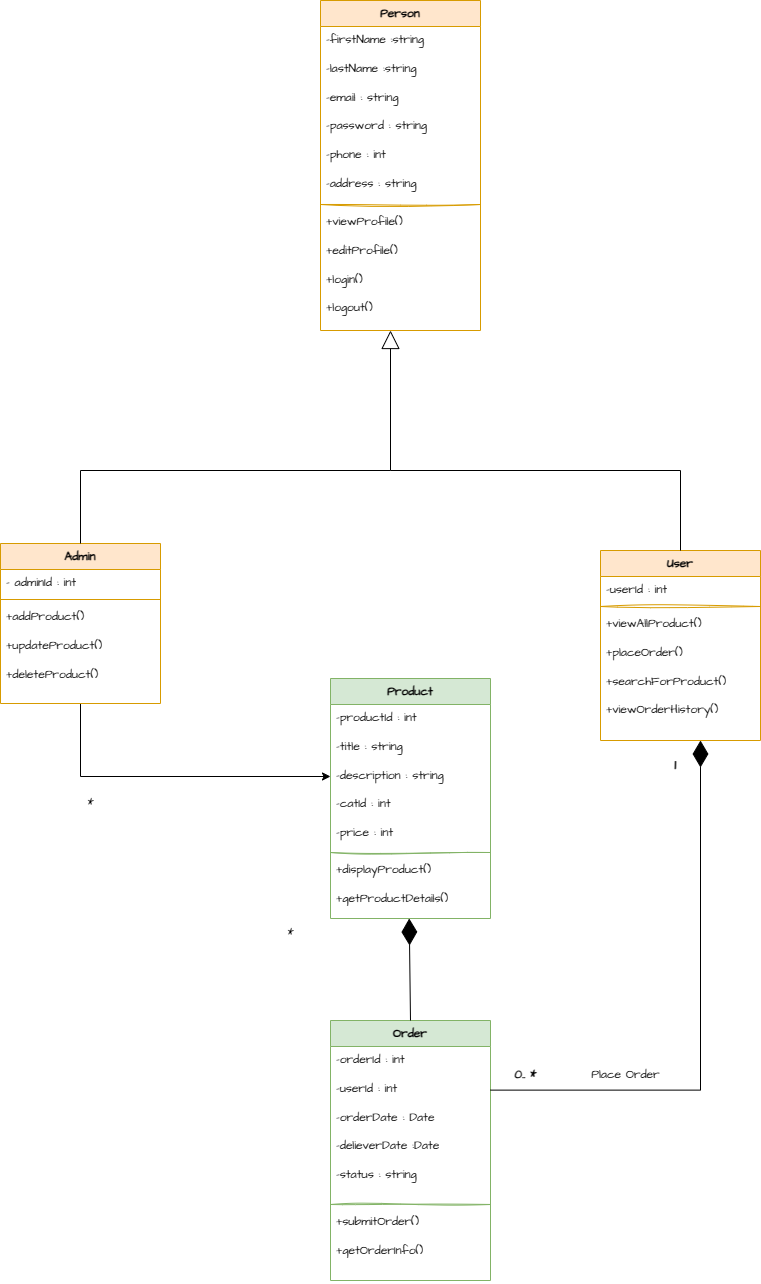
1. **Admin : Manage Product**

****

1. **User :Place an Order**

****

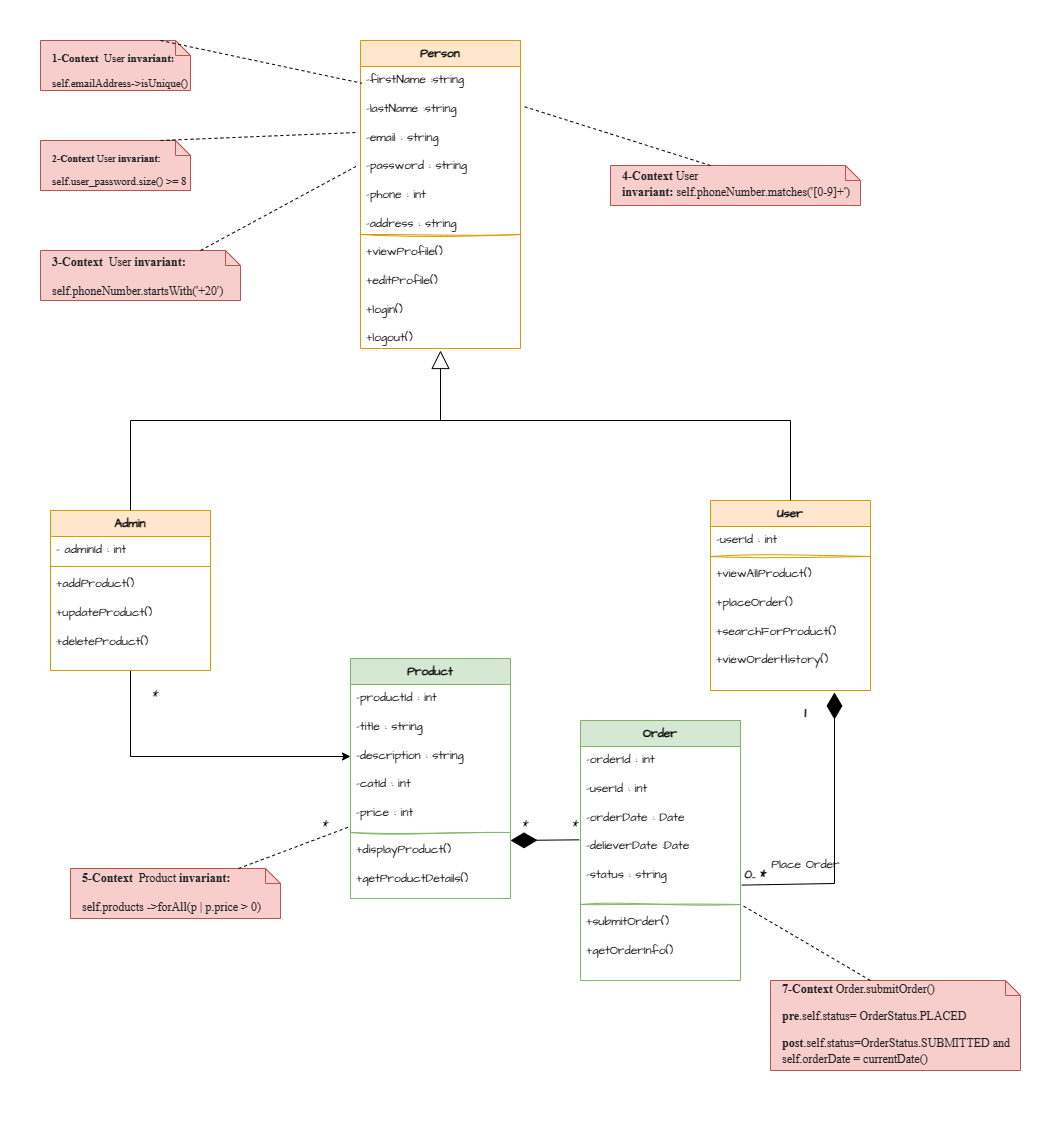
1. **Class Diagram**

****

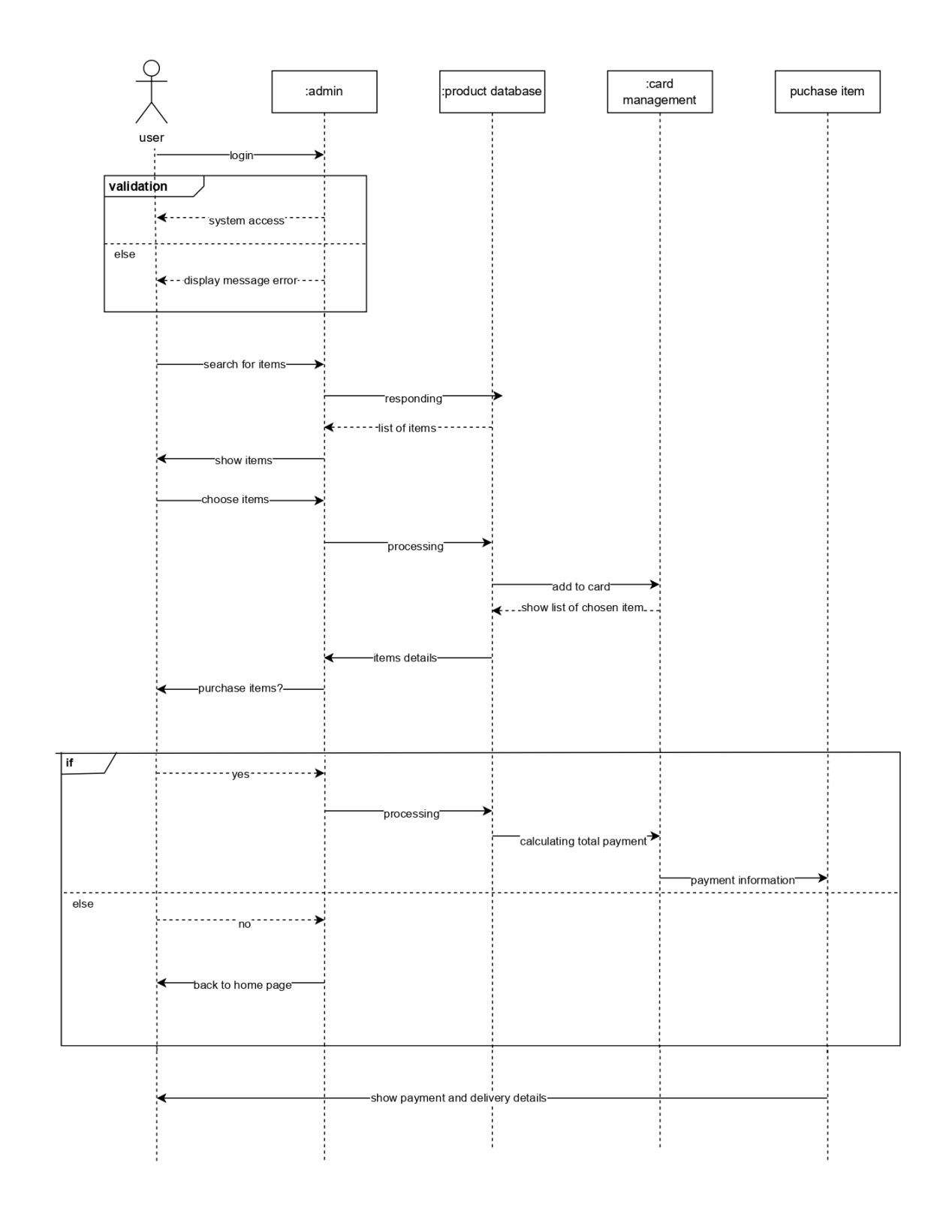
* 1. **OCL Constraints Description:**

|  |  |  |
| --- | --- | --- |
|  | **Constraint** | **Description** |
| **1** | **Context** User **invariant:**  self.emailAddress->isUnique() | This constraint ensures that there are no two or more User instances with the same email address. |
| **2** | **Context** User **invariant:**  self.user\_password.size() >= 8 | This constraint ensures that each user's password must be at least 8 characters long |
| **3** | **Context** User **invariant:**  self.phoneNumber.startsWith('+20') | This constraint ensures that all phone numbers for users in the system are valid Egyptian phone numbers. |
| **4** | **Context** User **invariant:** self.phoneNumber.matches('[0-9]+') | This constraint ensures that all phone numbers for users in the system are valid and contain only digits |
| **5** | **Context** Product **invariant:**  self.products->forAll(p | p.price > 0) | This constraint ensures that all products in the system have a valid price. |
| **6** | **Context** Order.submitOrder()  **pre**.self.status= OrderStatus.PLACED  **post**.self.status=OrderStatus.SUBMITTED and self.orderDate = currentDate() | This constraint describes the precondition and postcondition for the submitOrder operation of an Order class  **Precondition**: Before submitting the order, the order status must be PLACED.  **Postcondition**: After submitting the order, the order status becomes SUBMITTED |

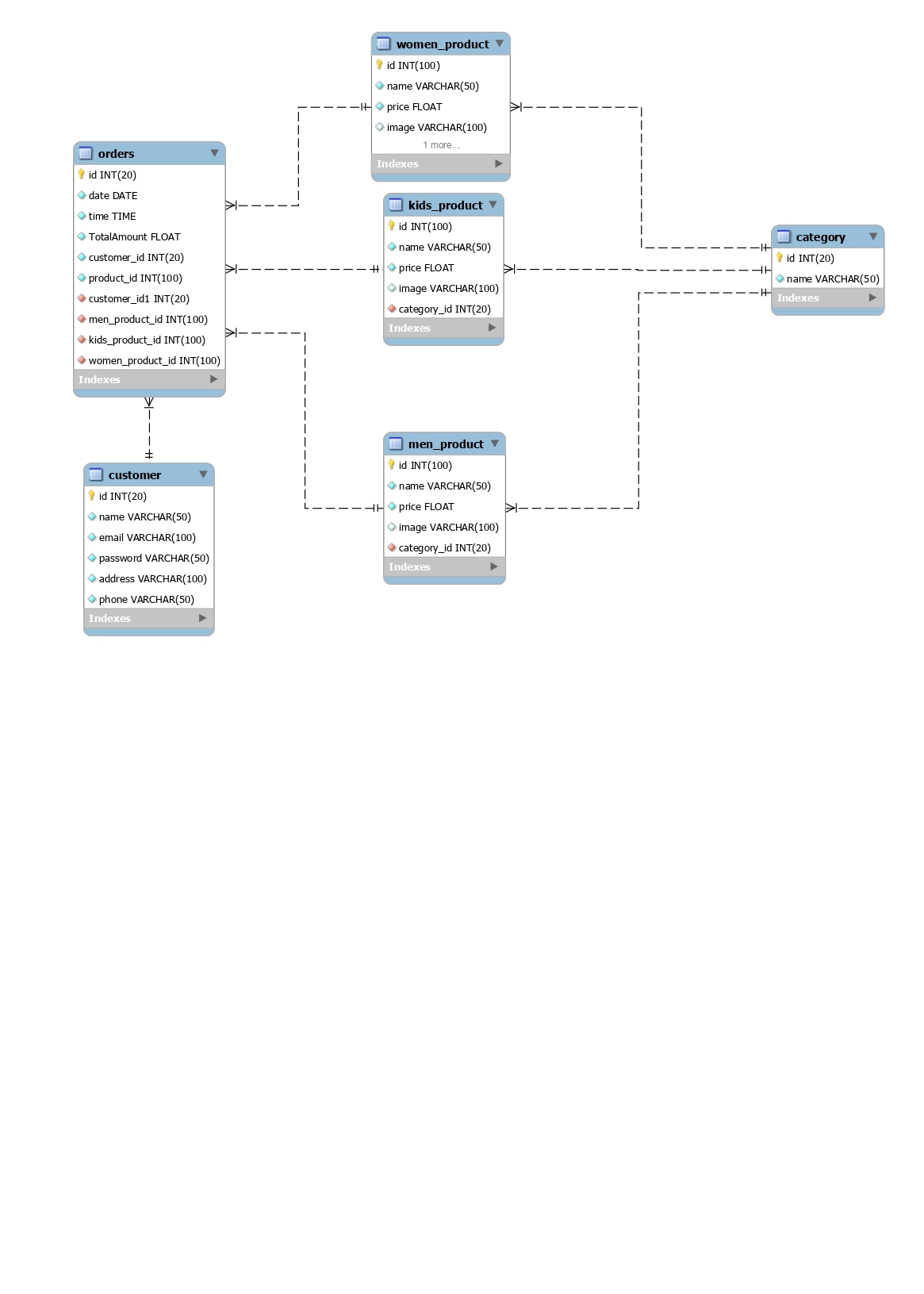
**[Class Diagram After Applying OCL Constraints ]**

****

**4.Sequence Diagram**

****

**5.Erd**

****