# INTRODUCTION TO PROJECT

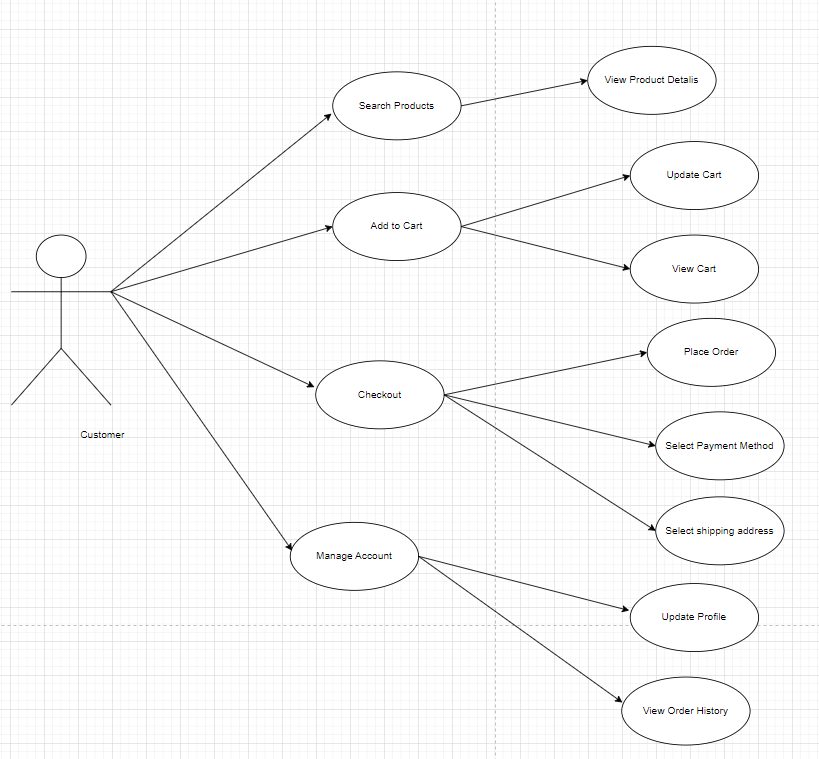
The **Online Medical Store Web Portal** is a cutting-edge platform that revolutionizes the way you purchase medicines and healthcare products online. Customers can effortlessly browse through a diverse range of product categories, search for specific items, and explore detailed product information, including names, prices, and availability, all without the need for logging in. This open access empowers users to fully experience the breadth of offerings available at their convenience.

To complete a purchase, users are required to log in, ensuring a secure and personalized shopping experience. Once logged in, customers can seamlessly add products to their cart, enter delivery details, and finalize their orders using the **Cash on Delivery (COD)** payment method. The system employs **JWT (JSON Web Token) authentication** to safeguard user accounts and transactions, providing a secure and trustworthy environment for online shopping.

Built with a powerful **Spring Boot** backend for robust business logic and API management, a sleek **React** frontend for an engaging user interface, and a reliable **MySQL** database for efficient data handling, the **Online Medical Store Web Portal** delivers a superior, secure, and intuitive shopping experience. This integrated approach ensures that both browsing and purchasing are handled with the utmost ease and security, making it a standout solution in the online medical retail space.

**2.REQUIREMENTS**

**2.1 FUNCTIONAL REQUIREMENTS**

****

**2.1 User Account**

Upon entering the **Online Medical Store Web Portal**, users are presented with three distinct options, depending on their status as either a guest or a registered user, and their intention to either browse or make a purchase. The terms ‘registered user’ and ‘guest’ are defined as follows:

A **registered user** is someone who has previously created an account on the system, providing a username and password. This account, referred to as their ‘profile,’ allows the registered user to log in and access a range of features. Registered users can not only browse the available products but also add items to their cart, proceed with checkout, and place orders.

A **guest**, on the other hand, is a new user who has not yet registered. Guests can browse the product catalog and view detailed information, such as product descriptions, prices, and availability. However, guests are limited to viewing product information and cannot make purchases or place orders.

New users have the option to either:

1. **Register** by providing personal information to create a profile, thereby becoming a registered user.
2. **Browse as a guest** without creating an account, which allows them to explore product information but restricts them from making any purchases.

Even registered users can choose to browse as a guest if they wish to simply view product information without logging in. Throughout the shopping experience, users can exit the system at any time if they choose to discontinue their session.

.

**2.2 Registration and creation of user profile**

To complete any transactions on the **Online Medical Store Web Portal** beyond browsing product information, users are required to register. During the registration process, the system will prompt users to provide the following information: user ID, password, first name, last name .

****

* 1. **Quick Search**

The **Online Medical Store Web Portal** provides a Quick Search facility that allows users to browse and compare product availability and prices without logging in. This feature enables users to easily search for products across various categories and view their prices, facilitating informed decision-making.

Once logged in, either as a registered user or a guest, the system will prompt the user to enter specific details to proceed with their order. Users must provide the following information: origin city and destination city (for address purposes), product category, and any specific product names or criteria.

After submitting these details, the system will access the product database to check for availability. Users will be asked to specify additional parameters such as product quantity, delivery preferences, and any applicable discounts or promotions.

The system will validate the entered details to ensure there are no inconsistencies, such as ordering more products than are available in stock. In case of any errors, the system will notify the user and prevent the order from being processed.

Once the information is verified, the system will query the product database and display the results in a clear and organized manner, typically in a tabular format. This display will include details such as product name, price, available quantity, and any current promotions. Users will see an option to add products to their cart directly from this table.

For each product added to the cart, users will be required to provide personal information for shipping and processing. The system will then calculate the total price for the selected items, taking into account quantities and any applicable discounts.

**2.4 Making Purchases/Order Confirmation**

After a user has completed their search and checked the availability of products as described in Step 2.2, the system will prompt the user to proceed with their purchase.

If the user decides to buy products, the following steps apply: a) **If the user is a guest**, they will need to register to create a profile and log in to the system before proceeding with the purchase. b) **If the user is already a registered user** and is logged in, they can directly proceed with the purchase. If they were browsing as a guest, they must log in first.

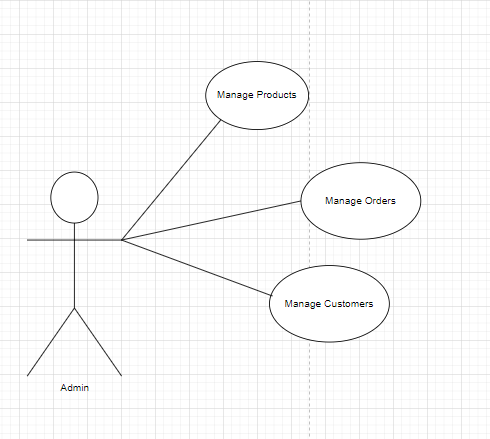
Once logged in, the system will display the total price for the selected products. Users will be required to enter their payment details, such as debit or credit card information. The system will process the payment and charge the amount to the provided payment method.

Upon successful payment, the system will update the inventory database to reflect the purchase and confirm the order. A confirmation email will be sent to the user with details of their order and estimated delivery time.

**2.5 View Order History**

Once logged in, users will be directed to their dashboard, where they can easily access a comprehensive overview of their previous orders. The dashboard will feature a section labeled "Order History," which allows users to view all details related to their past purchases.

By selecting this section, users will see a detailed list of their previous orders displayed in a tabular format. The information will include product names, quantities, prices, order dates, and any applicable status updates. This functionality ensures users can easily track and review their past transactions directly from their user dashboard.

****

**2.6 Admin Functionality**

Admins can securely log in to the control panel and perform the following tasks:

**Manage Orders:** View and update customer orders, including status changes and handling returns.

**Manage Users:** Access and update user profiles and handle user-related issues.

**Manage Products:** Add, update, or remove products, including managing details like names, prices, and stock levels.

* 1. **NON FUNCTIONAL REQUIREMENTS**

**2.2.1 Interface**

Go to Appendix B for user interfaces

**2.2.2 Performance**

* **Number of Concurrent Users:**

ARS shall be able to handle at least 1000 transactions/inquiries per

second

* **Booking of Tickets:**

The system is susceptible to any temporary server failure since it uses the strong feature of Struts 2 and Hibernate. Hence the examination will be continued even if the sever gets disconnected in between the examination.

**2.2.3 Constraint**

ARS shall be able to handle at least 1000 transactions/inquiries persecond

**2.2.4 Other Requirements:**

* **Hardware Interfaces**

The SPMS is expected to function on Intel PIII 900 MHz Processor equivalent or above, 128 MB RAM, 20 GB HDD.

* **Software Interfaces**

The SPMS shall work on MS Windows operating systems family (MS Windows 98, MS Windows NT Workstation, MS Windows 2000, MS Windows XP). It configures to work with Oracle database. This System works on Apache Tomcat server. It uses browser IE 5.0 & above. It uses IIS 5.0 server.

**3. DESIGN**

**3.1 Database Design**

The following table structures depict the database design.

# Table1: User:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| user\_id | bigint | NO | PRI | NULL | auto\_increment |
| email | varchar(255) | NO | UNI | NULL |  |
| first\_name | varchar(30) | YES |  | NULL |  |
| last\_name | varchar(30) | YES |  | NULL |  |
| mobile\_number | varchar(10) | YES |  | NULL |  |
| password | varchar(255) | YES |  | NULL |  |

**Table2: Address:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| address\_id | bigint | NO | PRI | NULL | auto\_increment |
| building\_name | varchar(255) | YES |  | NULL |  |
| city | varchar(255) | YES |  | NULL |  |
| country | varchar(255) | YES |  | NULL |  |
| pincode | varchar(255) | YES |  | NULL |  |
| state | varchar(255) | YES |  | NULL |  |
| street | varchar(255) | YES |  | NULL |  |

# Table3: user\_address:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| user\_id | bigint | NO | MUL | NULL |  |
| address\_id | bigint | NO | MUL | NULL |  |

### Table4: Roles:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| role\_id | bigint | NO | PRI | NULL |  |
| role\_name | varchar(255) | YES |  | NULL |  |

**Table5:User\_roles:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| role\_id | bigint | NO | PRI | NULL |  |
| role\_name | varchar(255) | YES |  | NULL |  |

**Table6:Products:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| product\_id | bigint | NO | PRI | NULL |  |
| description | varchar(255) | YES |  | NULL |  |
| discount | double | NO |  | NULL |  |
| image | varchar(255) | YES |  | NULL |  |
| price | double | NO |  | NULL |  |
| product\_name | varchar(255) | YES |  | NULL |  |
| quantity | int | YES |  | NULL |  |
| special\_price | double | NO |  | NULL |  |
| category\_id | bigint | YES | MUL | NULL |  |

**Table7: Orders:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| order\_id | bigint | NO | PRI | NULL | auto\_increment |
| email | varchar(255) | NO |  | NULL |  |
| order\_date | date | YES |  | NULL |  |
| order\_status | varchar(255) | YES |  | NULL |  |
| total\_amount | double | YES |  | NULL |  |
| payment\_id | bigint | YES | MUL | NULL |  |

**Table8:Order\_items:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| order\_item\_id | bigint | NO | PRI | NULL | auto\_increment |
| discount | double | NO |  | NULL |  |
| ordered\_product\_price | double | NO |  | NULL |  |
| quantity | int | YES |  | NULL |  |
| order\_id | bigint | YES | MUL | NULL |  |
| product\_id | bigint | YES | MUL | NULL |  |

### Table9:Categories:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| category\_id | bigint | NO | PRI | NULL | auto\_increment |
| category\_name | varchar(255) | YES |  | NULL |  |

**Table10:Payments:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Field** | **Type** | **Null** | **Key** | **Default** | **Extra** |
| payment\_id | bigint | NO | PRI | NULL | auto\_increment |
| payment\_method | varchar(255) | YES |  | NULL |  |

**E-R Diagram,Dataflow diagram and Class Diagram:**

Go to Appendix A

**4. CODING STANDARDS IMPLEMENTED**

### Naming and Capitalization

Below summarizes the naming recommendations for identifiers in Pascal casing is used mainly (i.e. capitalize first letter of each word) with camel casing (capitalize each word except for the first one) being used in certain circumstances.

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | **Case** | **Examples** | **Additional Notes** |
| Class | Pascal | Person, BankVault, SMSMessage, Dept | Class names should be based on "objects" or "real things" and should generally be **nouns**. No ‘\_’ signs allowed. Do not use type prefixes like ‘C’ for class. |
| Method | Camel | getDetails, updateStore | Methods should use **verbs** or verb phrases. |
| Parameter | Camel | personName, bankCode | Use descriptive parameter names. Parameter names should be descriptive enough that the name of the parameter and its type can be used to determine its meaning in most scenarios. |
| Interface | Pascal with "I" prefix | Disposable | Do not use the ‘\_’ sign |
| Property | Pascal | ForeColor, BackColor | Use a noun or noun phrase to name properties. |
| Associated private member variable | \_camelCase | \_foreColor, \_backColor | Use underscore camel casing for the private member variables |
| Exception Class | Pascal with "Exception" suffix | WebException, |  |

### Comments

* Comment each type, each non-public type member, and each region declaration.
* Use end-line comments only on variable declaration lines. End-line comments are comments that follow code on a single line.
* Separate comments from comment delimiters (apostrophe) or // with one space.
* Begin the comment text with an uppercase letter.
* End the comment with a period.
* Explain the code; do not repeat it.

**5. TEST REPORT**

**Another group called Linux did the testing and the report of the testing is given hereunder.**

**GENERAL TESTING:**

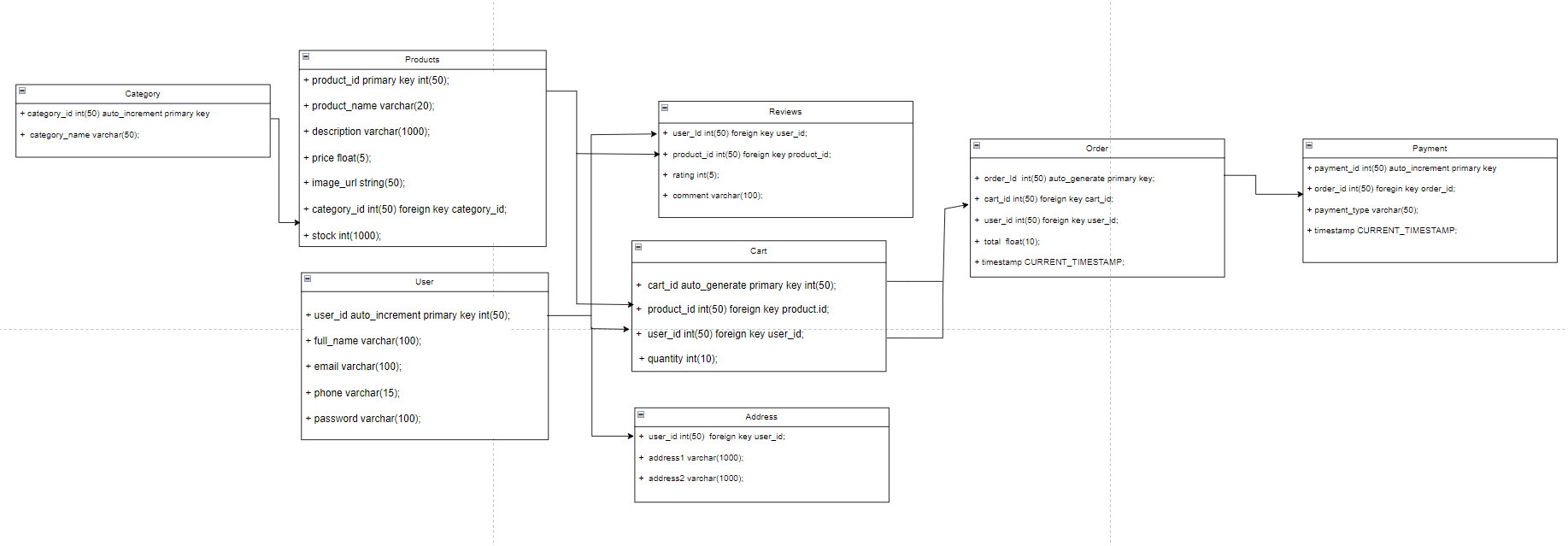
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SR-NO** | **TEST CASE** | **EXPECTED RESULT** | **ACTUAL RESULT** | **ERROR MESSAGE** |
| 1 | Register Page | Redirected to Next page | OK | Nothing |
| 2 | Login Page | Pop-up will come | Ok | Please enter username and password again . |
| 3 | Reset login | Only users password will be reseted | Ok | Nothing |
| 4 | Quick search | Gives all flight details | Ok | Nothing |
| 5 | Ordering product | All the fields should be filled for submission | Ok | Nothing |
| 6 | Checking login or not | User is logged in or not | Ok | Nothing |
| 7 | Add address details for orders | Add informations according to no of seats allocated | Ok | Nothing |
| 8 | Go to user page | Set added information about person | Ok | Nothing |
| 9 | Add information in orders table | Save this all data into booking table | Ok | Nothing |
| 10 | Order detail | On back it should be reverted to previous page | Ok | Nothing |
| 11 | View transaction done | It shows you all transactions done previously | Ok | Nothing |
| 12 | Logout | It will logout from user profile. | Ok | Nothing |
|  | **STATIC TESTING** |  |  |  |
| **SR-NO** | **Deviation** | **Program** |  |  |
| 1 | Commenting not followed | All Web Application |  |  |

**6. PROJECT MANAGEMENT RELATED STATISTICS**

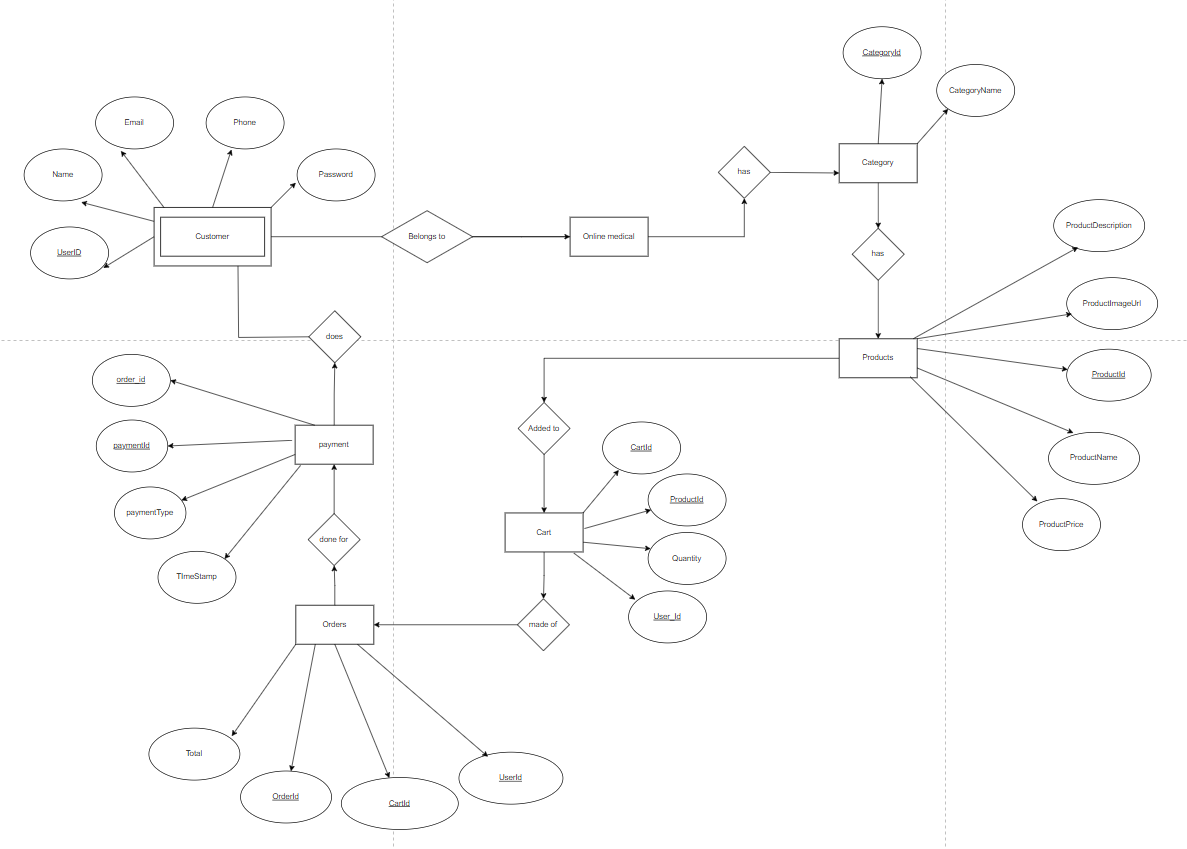
|  |  |  |  |
| --- | --- | --- | --- |
| **DATE** | **WORK PERFORMED** | **SLC Phase** | **Additional Notes** |
| 15-Jul-24 | Project Allotment and User Requirements Gathering | Feasibility Study | Met client Mr. Nitinkudale (CEO, SIIT Pune) to understand requirements |
| 18-Jul-24 | Initial SRS Document Validation and Team Structure Decided | Requirement Analysis (Elicitation) | Presented initial SRS to client for better understanding |
| 22-Jul-24 | Designing use-cases, Class Diagram, Collaboration Diagram, E-R Diagram, User Interfaces | Requirement Analysis & Design Phase | Completed database design |
| 26-Jul-24 | Business Logic Component design Started | Design Phase | - |
| 30-Jul-24 | Coding Phase Started | Coding Phase | 70% of Class Library implemented |
| 01-Aug-24 | Implementation of Web Application and Window Application Started | Coding Phase | Class Library Development ongoing |
| 03-Aug-24 | Off | Off | Off |
| 06-Aug-24 | Implementation of Web Application and Window Application Continued | Coding Phase and Unit Testing | Modified Class Library as per need |
| 08-Aug-24 | Implementation of Web Application and Window Application Continued | Coding Phase and Unit Testing | - |
| 10-Aug-24 | After Ensuring Proper Functioning, Required Validations Implemented | Coding Phase and Unit Testing | Module Integration done by Project Manager |
| 12-Aug-24 | The Project was Tested by Team Leaders and Project Manager | Testing Phase (Module Testing) | - |
| 14-Aug-24 | The Project was Submitted to Other Project Leader for Testing | Testing Phase (Acceptance Testing) | Taken up by team for testing |
| 15-Aug-24 | The Errors Found were Removed | Debugging | Project complete for submission |
| 16-Aug-24 | Final Submission of Project | - | - |

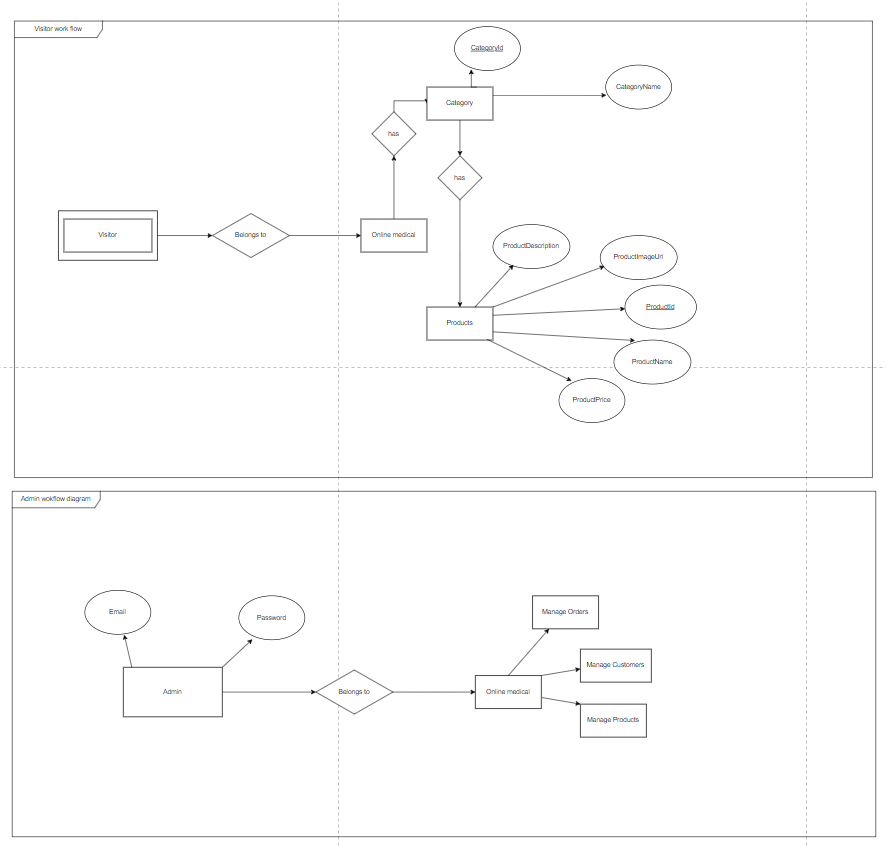
Appendix A

Entity Relationship Diagram

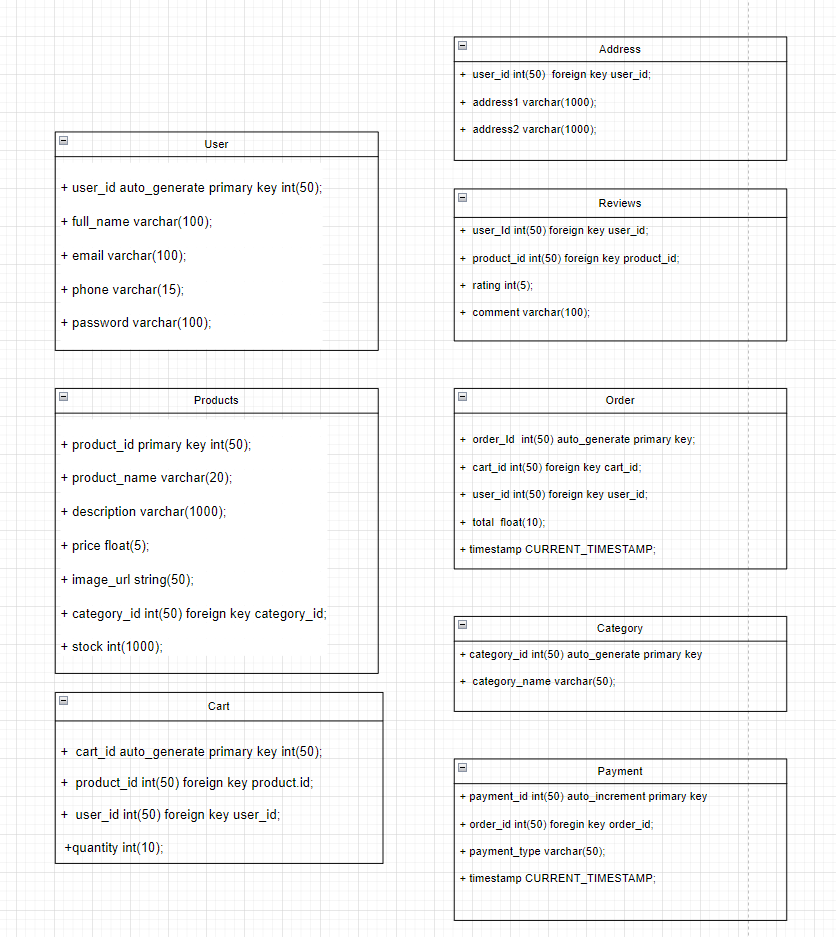


**Data Flow Diagram:**



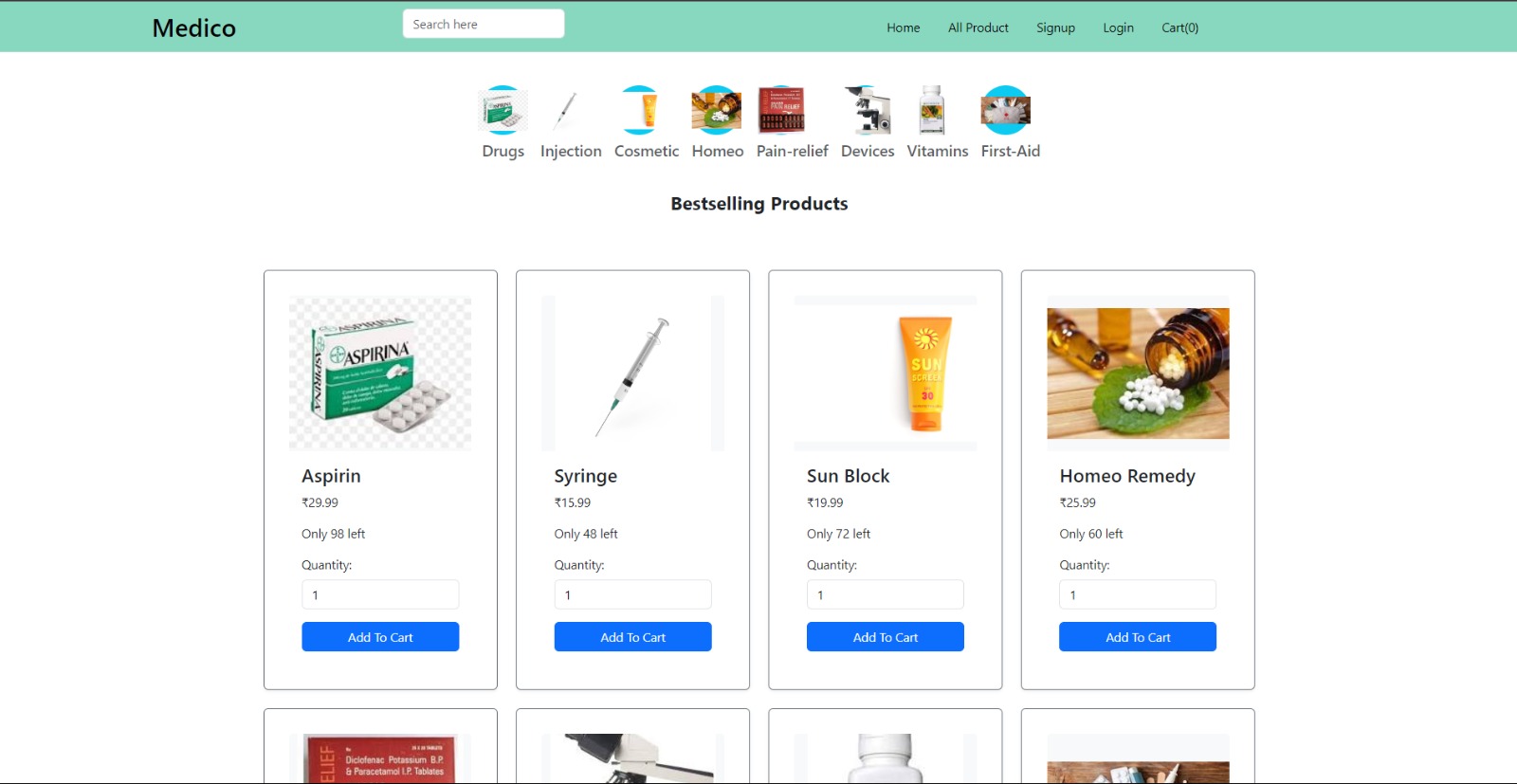


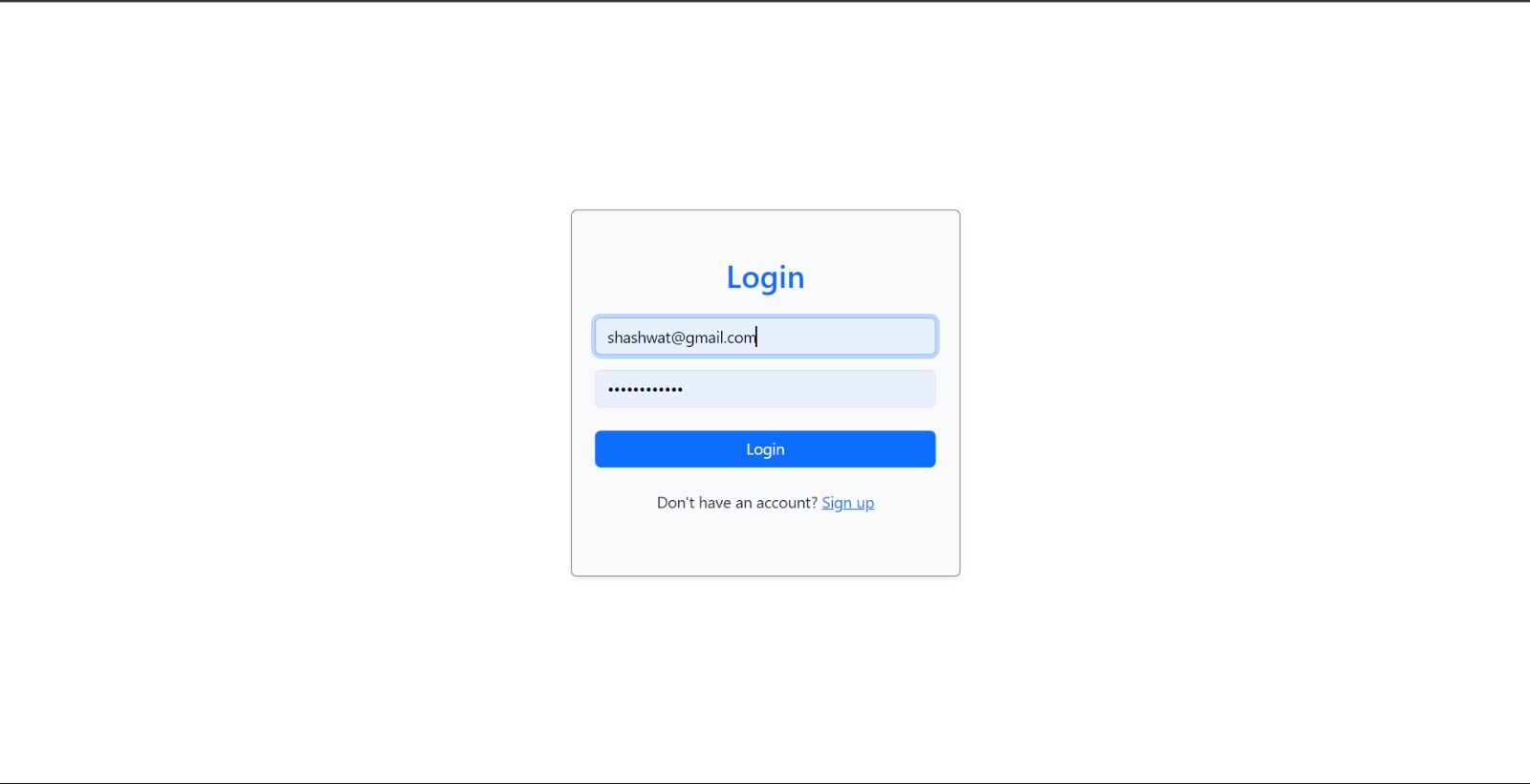
**Class Diagram**



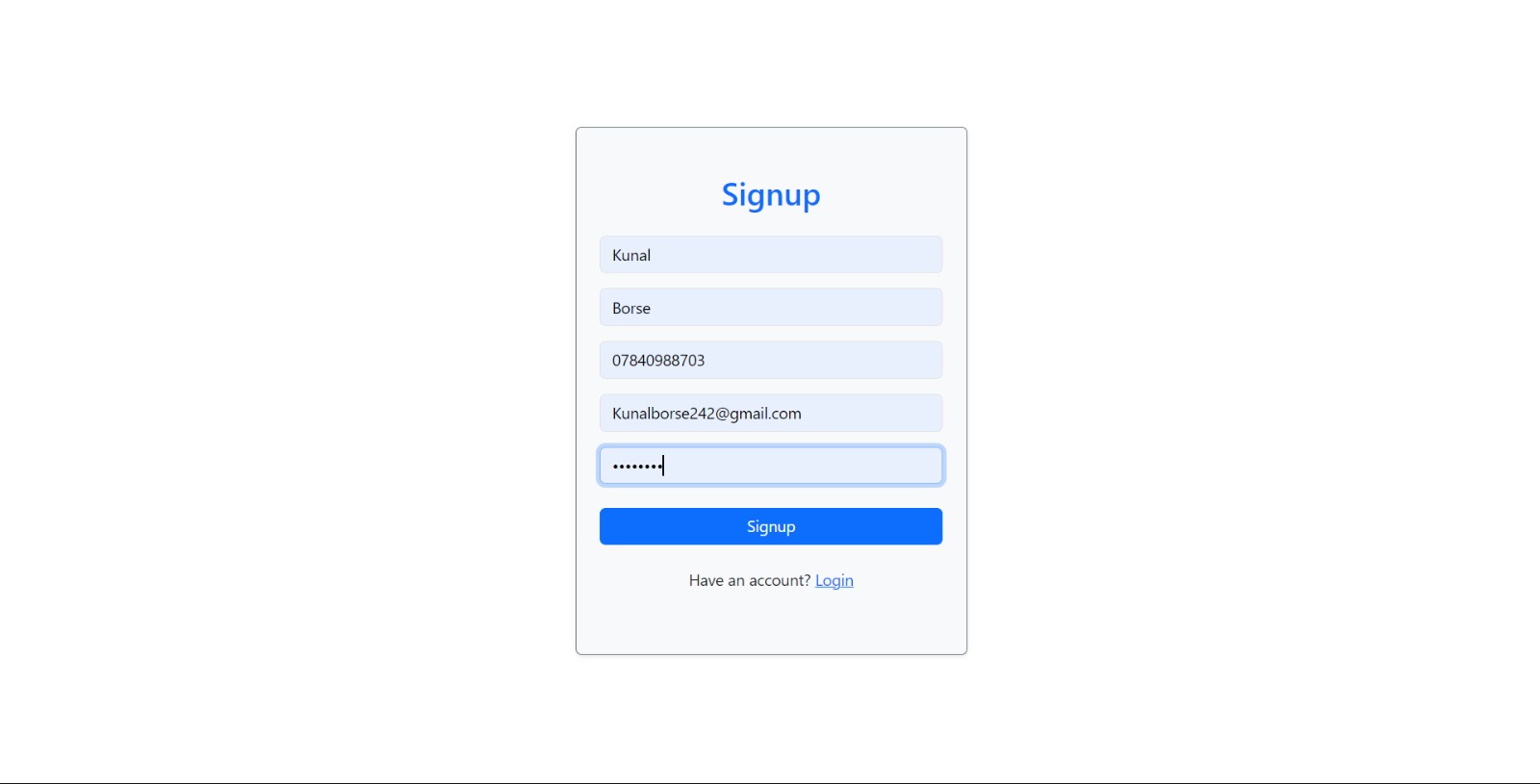
Appendix B

Homepage:

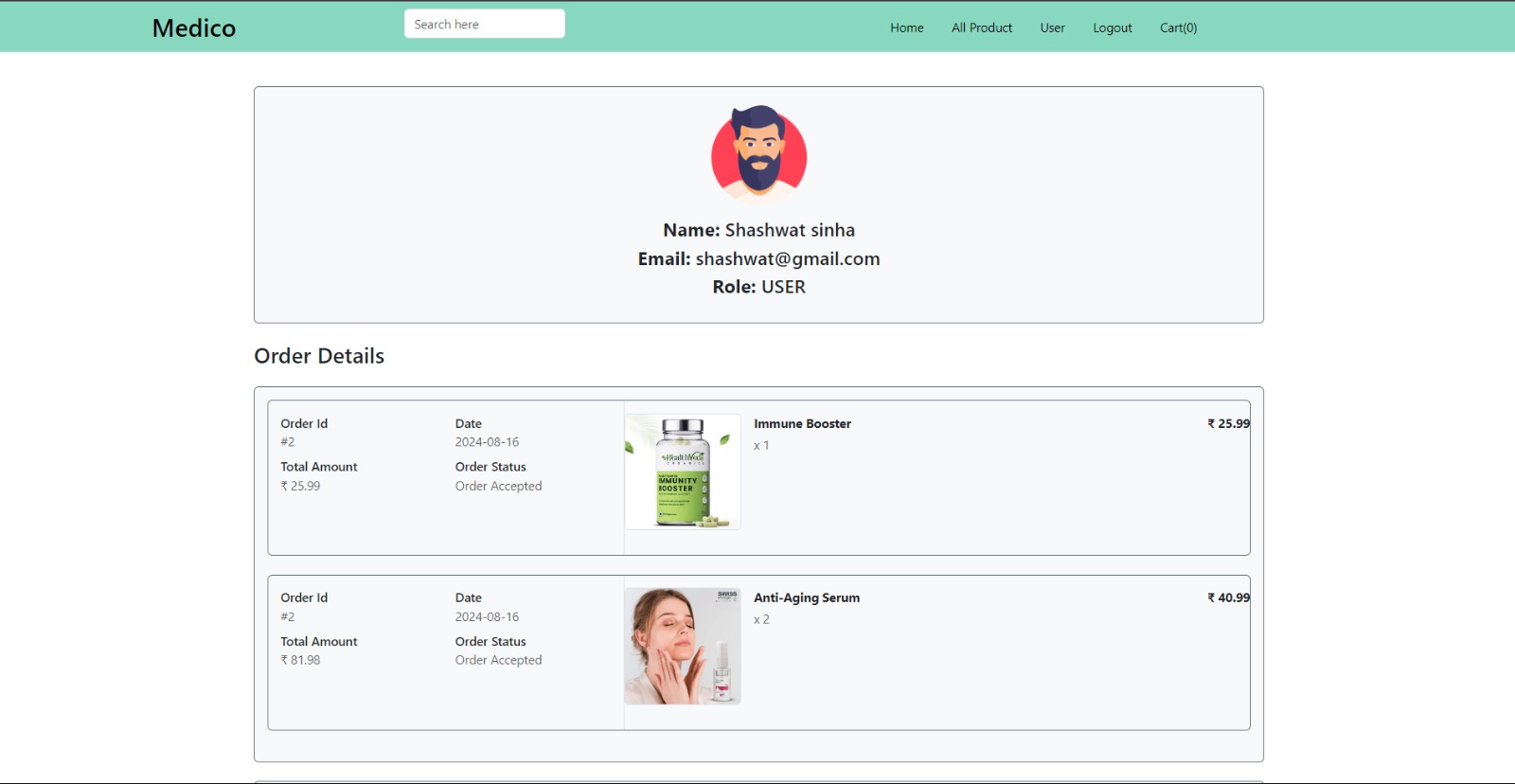


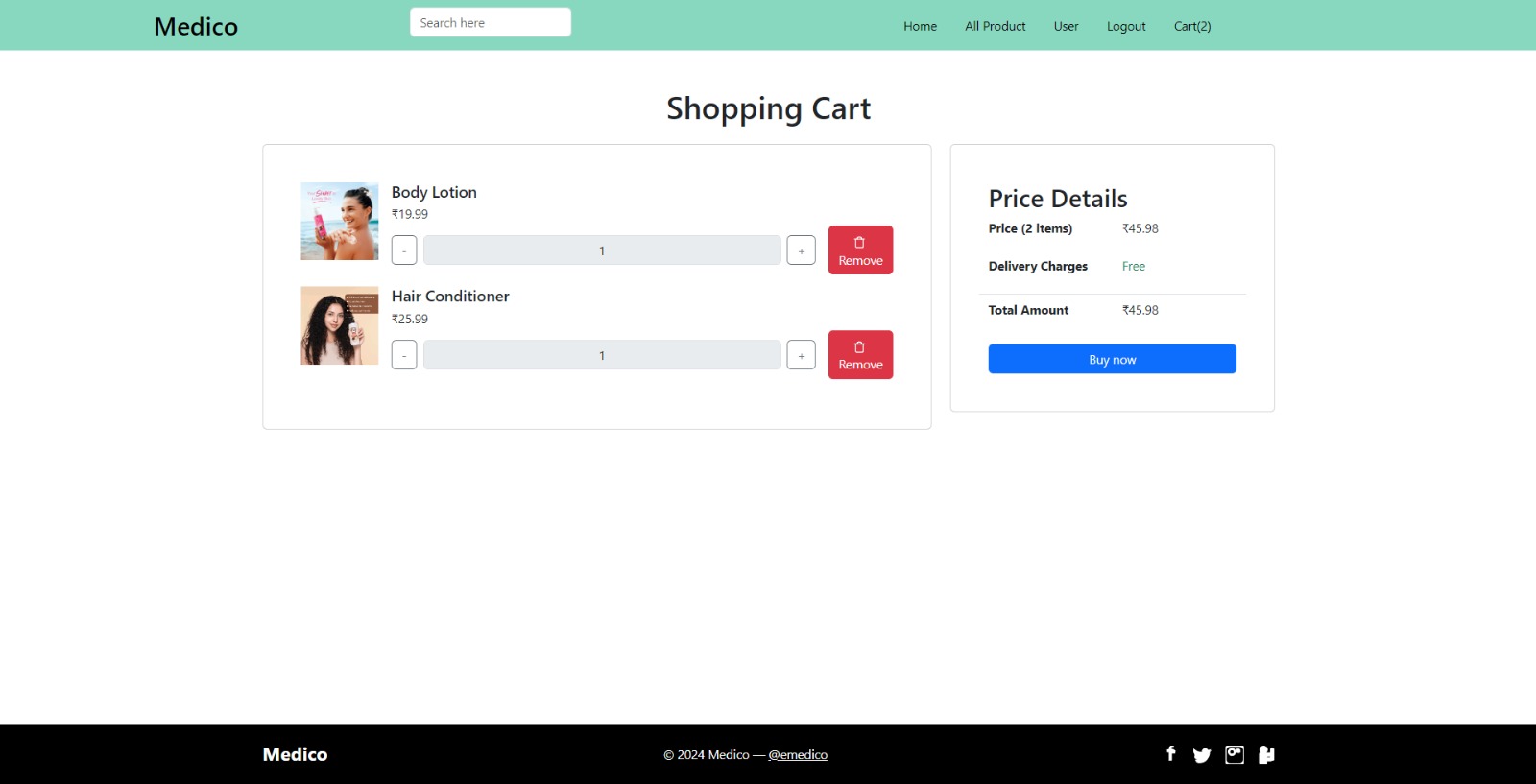
LoginPopup:

User Registration:

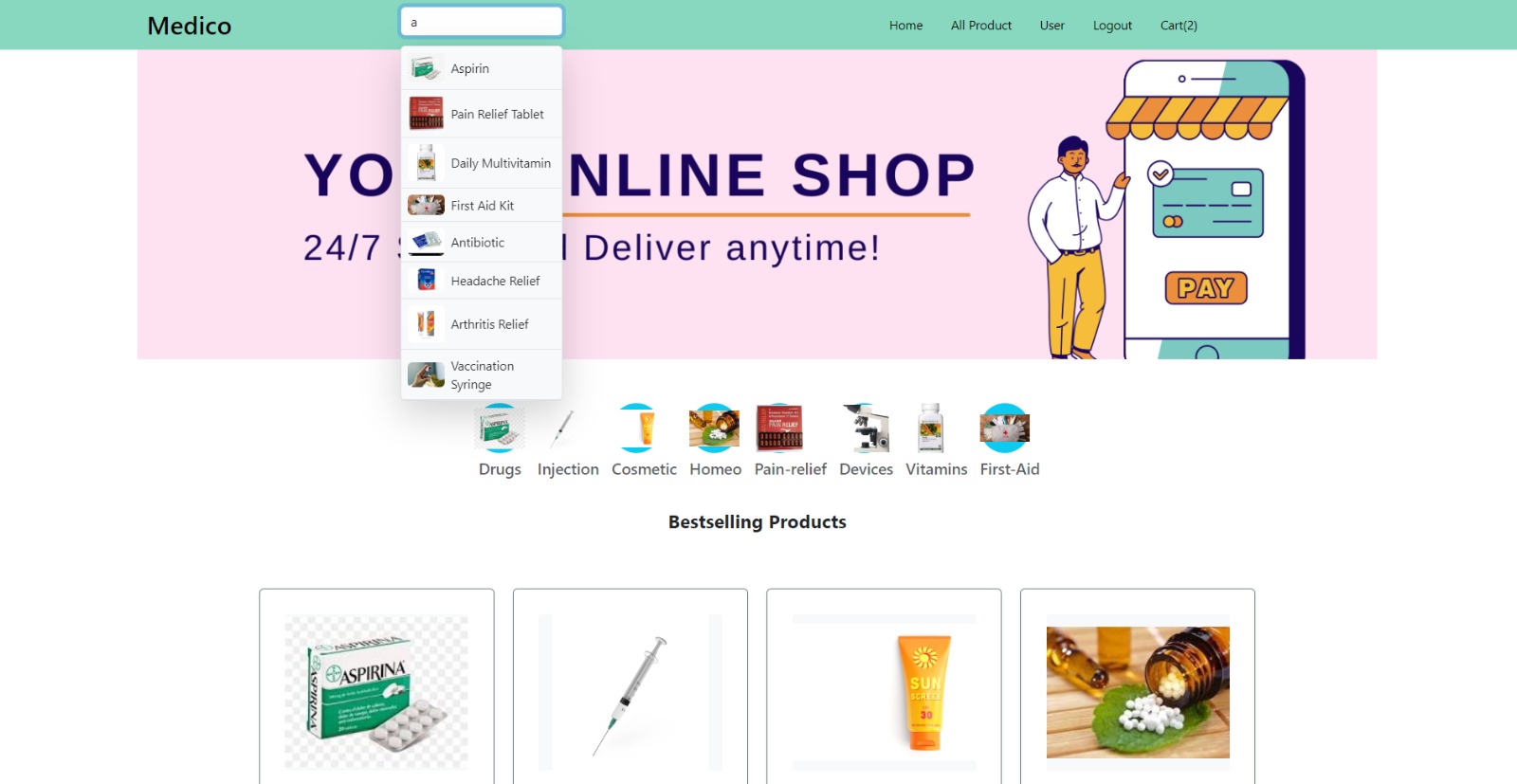


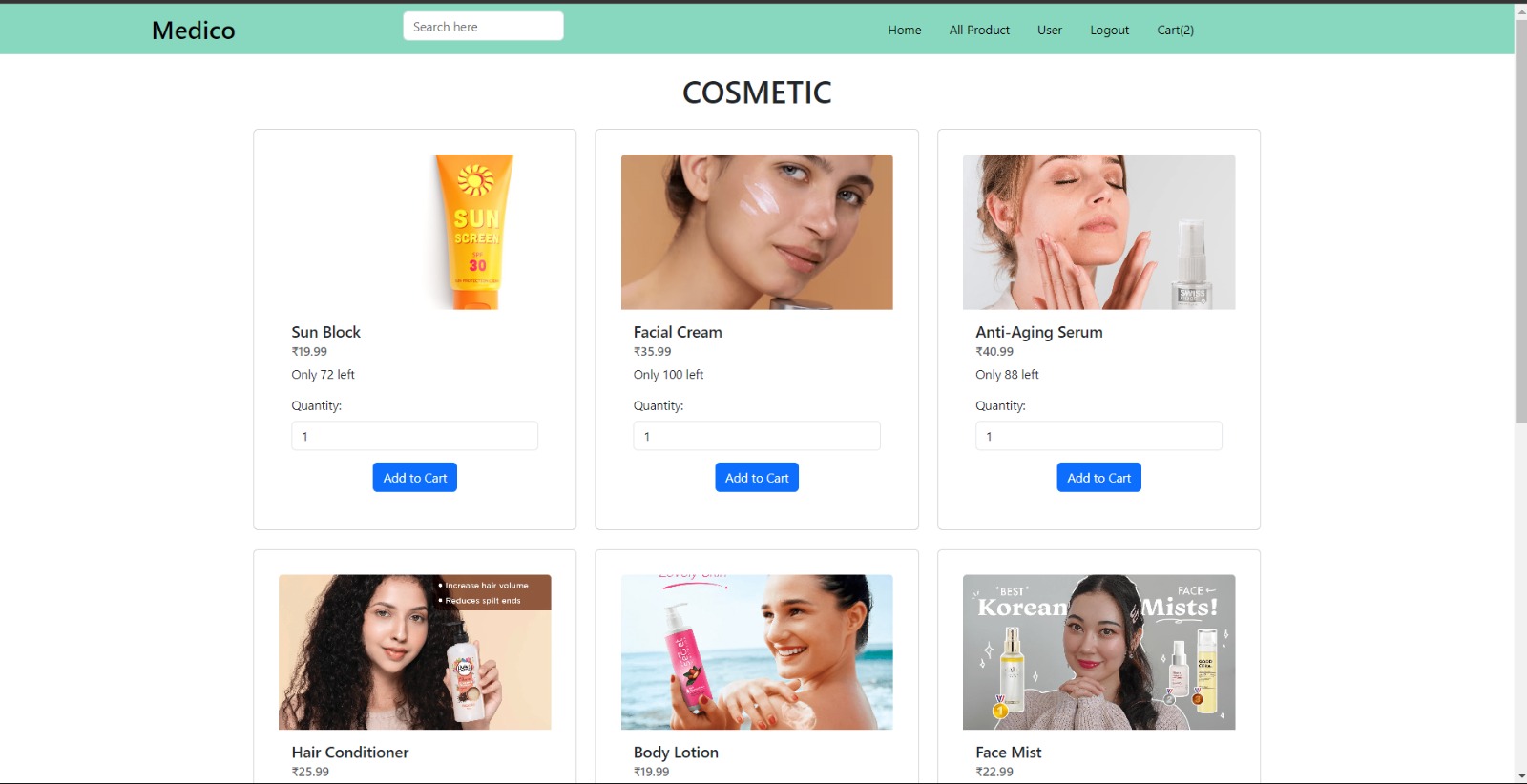
**User-DashBoard**

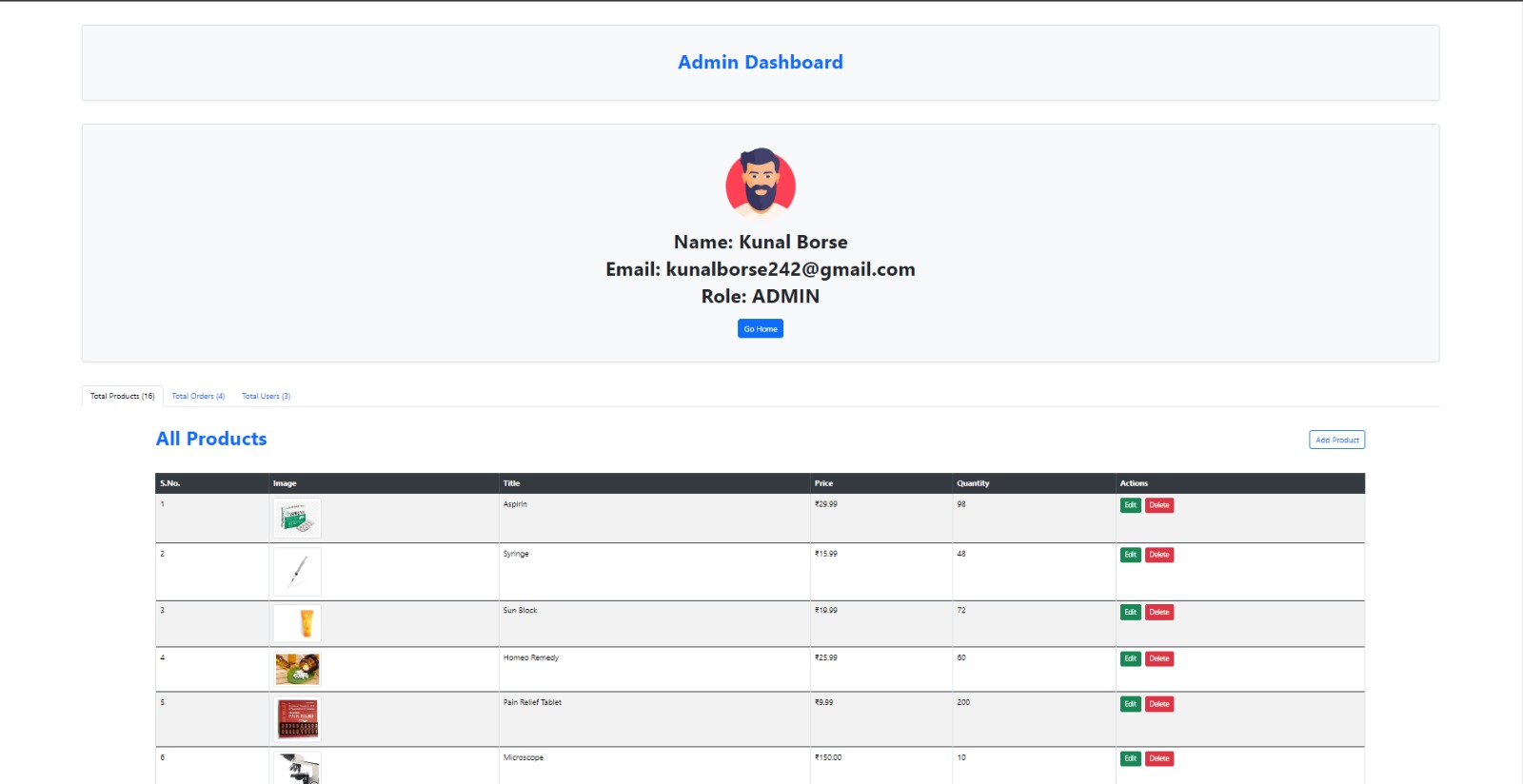


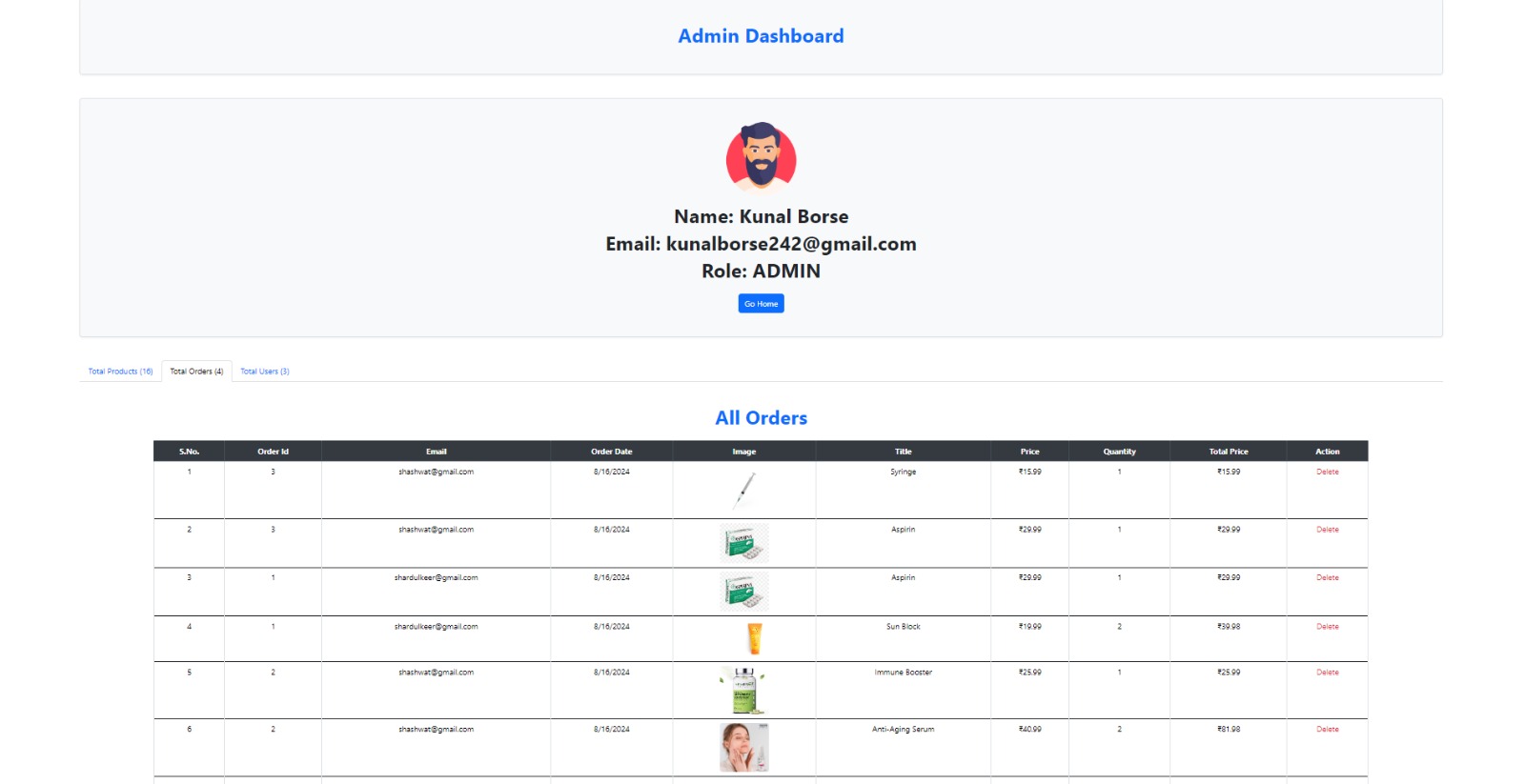
**Cart** 

**Quick Search:**

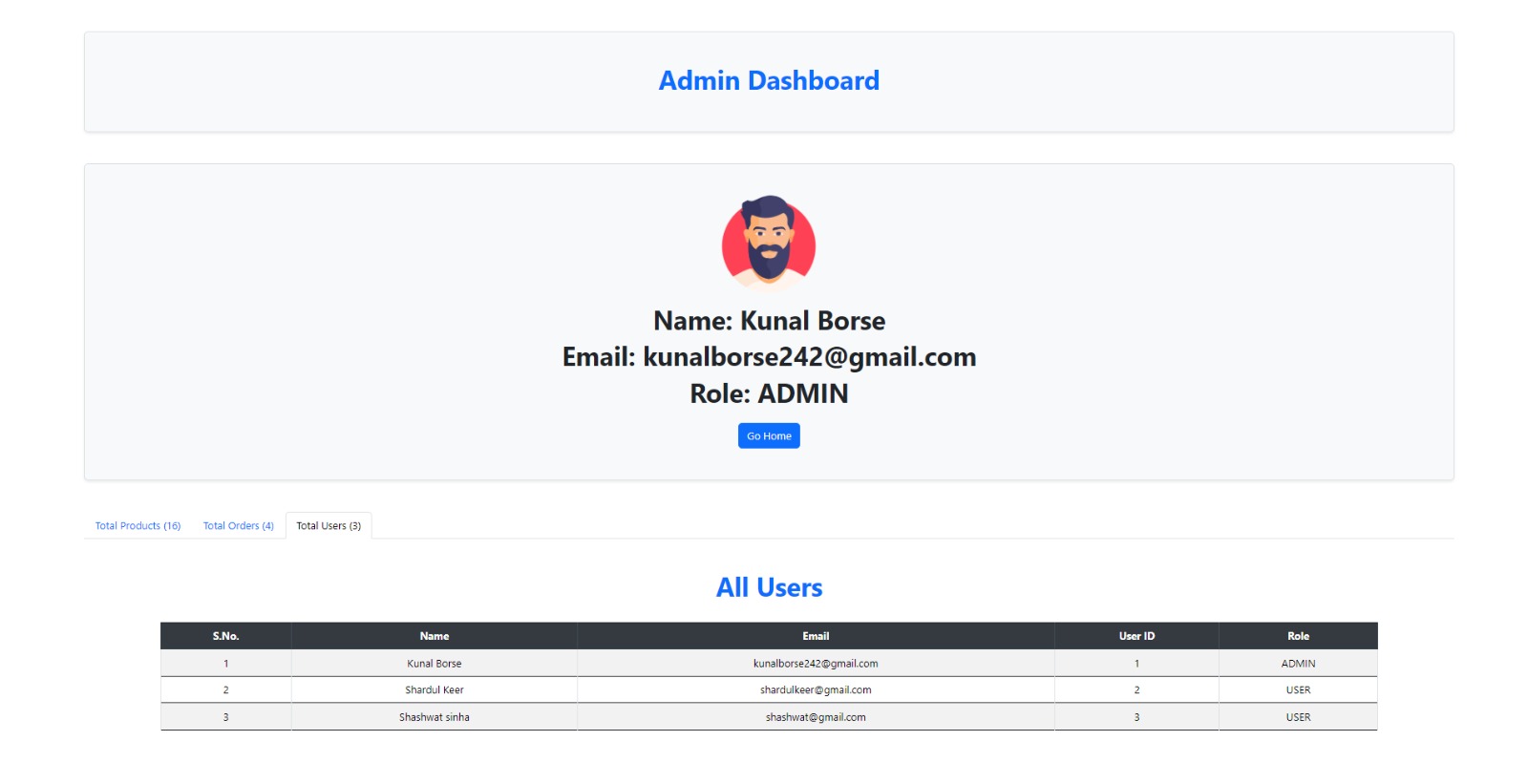
****

**Category\_products:**

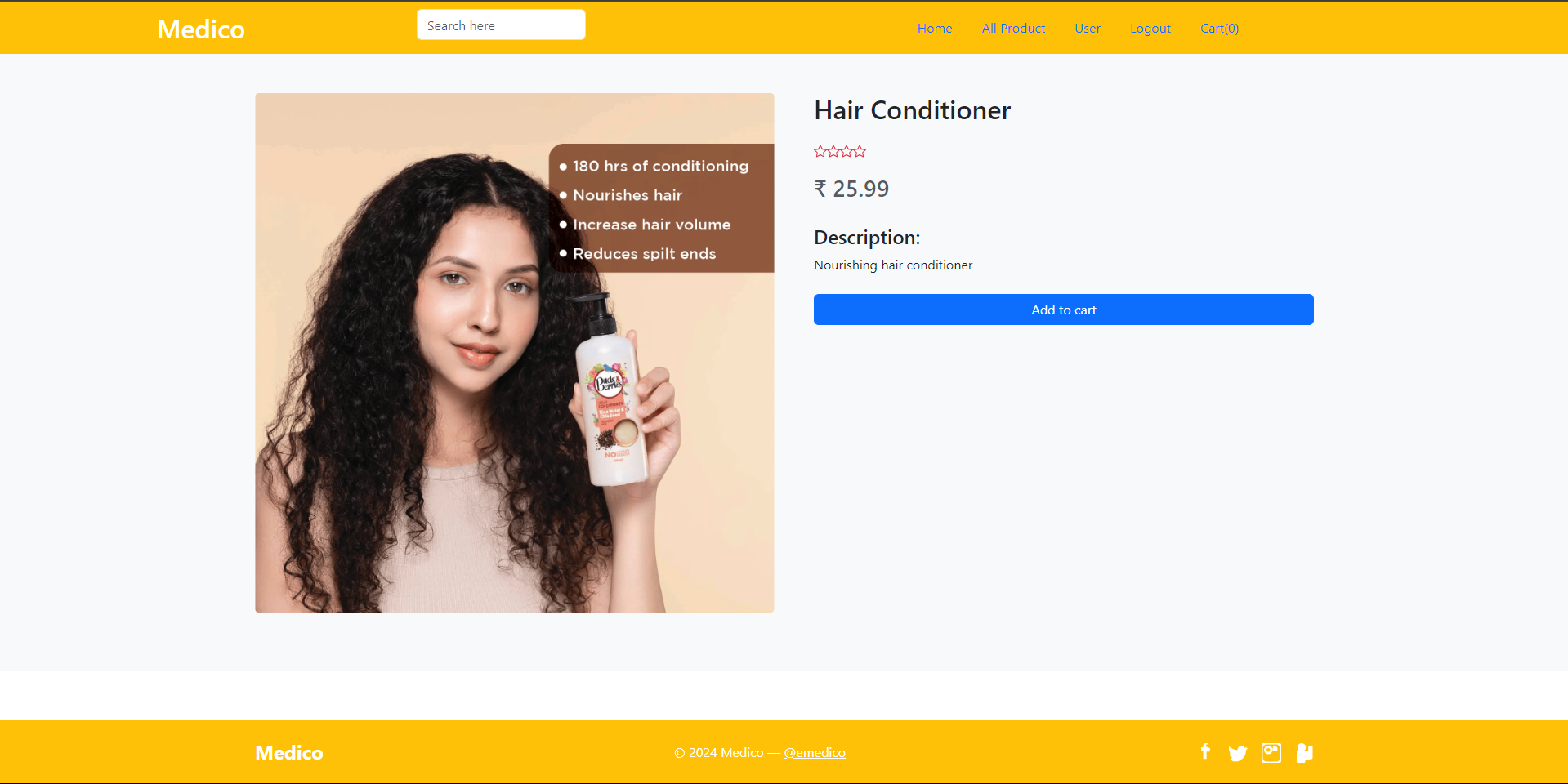
Admin\_dashboard\_products

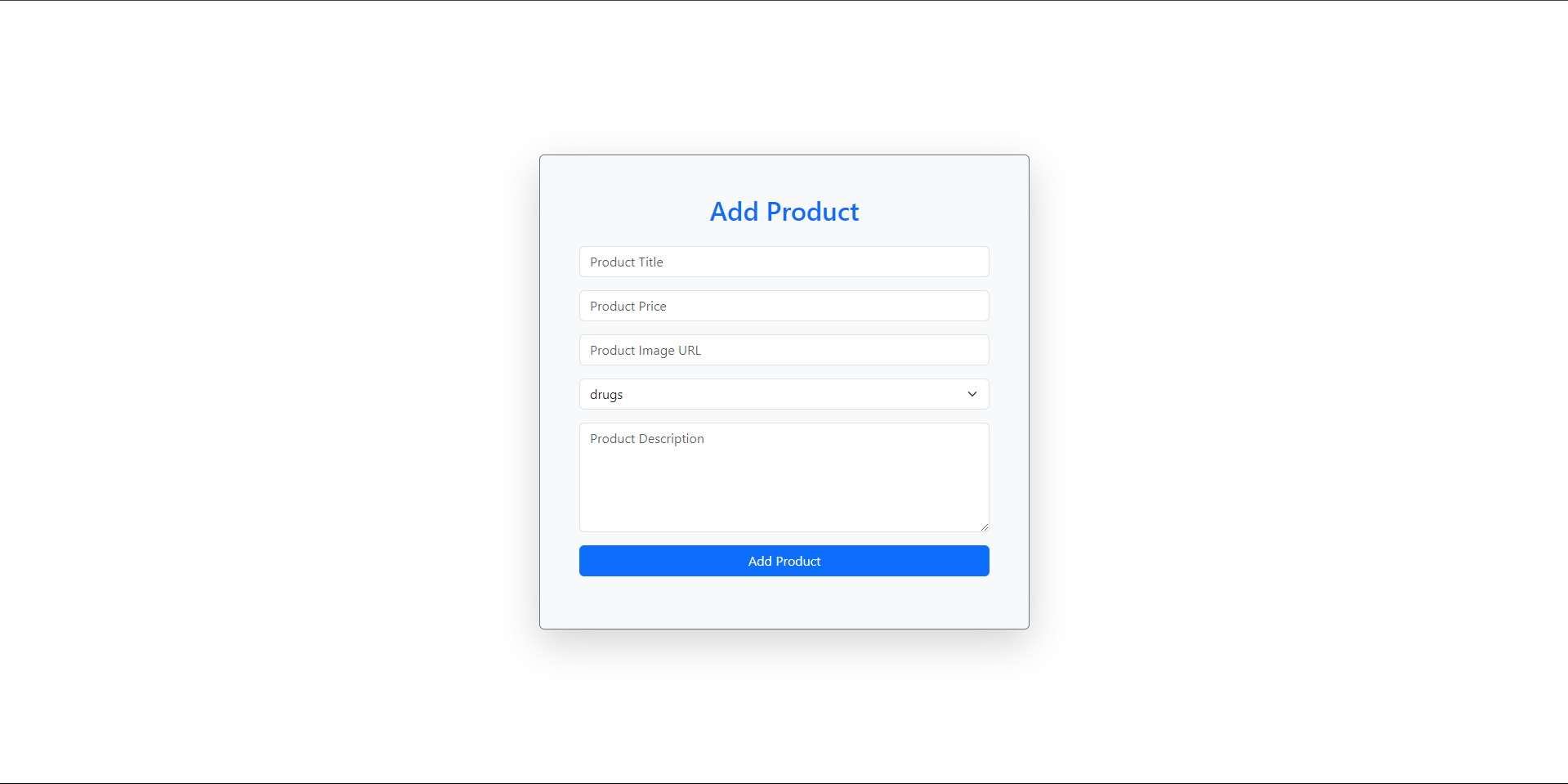
Admin\_Dashboard\_orders

Admin\_Dashboard

Users

ProductInfoPage:



**Admin\_addProductPage:**

**7.REFERENCES:**

<http://www.google.com>

https://www.1mg.com/

<http://www.webdevelopersjournal.com/>

<https://www.webmd.com/>

http://www.wikipedia.org

<https://pharmeasy.in/>

https://medlineplus.gov/