****

Republic Of Cameroon

\*\*\*\*\*\*

Peace – Work – Fatherland

\*\*\*\*\*\*

AICS Cameroon

E-mail: contact@iaicameroun.com

République Du Cameroun

\*\*\*\*\*\*

Paix – Travail – Patrie

\*\*\*\*\*\*

IAI Cameroun

E-mail : contact@iaicameroun.com

***Academic Year***

***2024-2025***

**THEME : COSMETIC AND SKIN CARE STOCK MANAGEMENT**

**GROUP MEMBERS(GROUP 12) :**

* **TEKENG DJANG YVAN DUPLEX PACOM (50%)**
* **FORCHE GODWIN FORBUWING (50%)**

***Under the supervision of :***

***Mr. FOMEKONG***

Table of Contents

[**I.** **INTRODUCTION** 3](#_Toc191716766)

[**II.** **UML Diagrams** 4](#_Toc191716767)

[**1.** **Use Case Diagram** 4](#_Toc191716768)

[**2.** **Activity Diagram** 8](#_Toc191716769)

[**3.** **Sequence Diagram** 11](#_Toc191716770)

[**4.** **Communication Diagram** 14](#_Toc191716771)

[**5.** **State Machine Diagram** 16](#_Toc191716772)

[**6.** **Class Diagram** 17](#_Toc191716773)

[**7.** **Object Diagram** 18](#_Toc191716774)

[**8.** **Package Diagram** 19](#_Toc191716775)

[**9.** **Component Diagram** 20](#_Toc191716776)

[**10.** **Deployment Diagram** 21](#_Toc191716777)

[**III.** **Database Schema** 22](#_Toc191716778)

[**IV.** **Java Code Implementation** 22](#_Toc191716779)

[**V.** **Some Screenshots of the Application** 23](#_Toc191716780)

[**CONCLUSION** 24](#_Toc191716782)

# **INTRODUCTION**

A stock management application tailored for the cosmetic and skin care industry plays a vital role in streamlining inventory processes. It helps businesses efficiently manage their products, track sales, and respond to market demands. The application is essential for businesses aiming to optimize their inventory processes. By leveraging technology, companies can enhanced operational efficiency, reduce costs, and improve customer satisfaction .The system is built using Java with NetBeans IDE and JDK, and it employs a MySQL database for data storage. The system has three main actors: Super Admin, Admin and Staff. But the staff is subdivided into three main actors that is stock manager, warehouse staff, and Financial Manager. Each actor interacts with the system to perform specific tasks, such as managing stock, recording sales, managing price, and restocking products.

This report provides a detailed explanation of the UML diagrams (use case, activity, sequence, communication, state machine, class, object, package, component, and deployment) and how they relate to the system's functionality. The report also includes an overview of the database schema and Java code implementation.

# **UML Diagrams**

## **Use Case Diagram**

The Use Case Diagram outlines the interactions between the actors and the system.

**Actors**:

**Super Admin**: Manages Users

**Admin**: Manages Users.

**Staff:**

**Stock Manager**: Revise product, View stock details.

**Warehouse staff**: Modify stock, View stock records.

**Financial Manager**: Manage price, View records.

**UseCases**:

**Super Admin**:

* Add Staff
* Delete Staff
* Update Staff
* Add Admin
* Delete Admin
* Update Admin
* Change password

**Admin**:

* Add Staff
* Delete Staff
* Update Staff
* Change password

**Stock Manager**:

* AddItem
* Update Item
* Delete Item
* View details

**Warehouse Staff**:

* Add Stock
* Update Stock
* Delete Stock
* View stock records

**Financial Manager**:

* Set price
* Update Price
* View records

**Relationships**:

* Admin and Staff interact with the system to manage stock, products, price and sales.
* Supplier interacts with the system to restock items. But we have not specify supplier as an actor because we are focus on internal stock management



*Figure 0 : General Use case*



*Figure 0.1:Stock Manager Specific use case*



*Figure 0.2:Financial Manager Specific use case*

## **Activity Diagram**

The Activity Diagram describes the workflow of the system.



*Figure 1.1:Stock Manager Add Product Activity Diagram*



*Figure 1.2:Admin Delete Staff Activity Diagram*



*Figure 1.3:Financial Manager Update Price Activity Diagram*

## **Sequence Diagram**

The Sequence Diagram shows the sequence of interactions between actors and the system.



*Figure 2.1:Stock Manager Add Product Sequence Diagram*



*Figure 2.2:Admin Delete Staff Sequence Diagram*



*Figure 2.3:Financial Manager Update Price Sequence Diagram*

## **Communication Diagram**

The Communication Diagram illustrates the interactions between actors and system components.



*Figure 3.1:Stock Manager Add Product Communication Diagram*



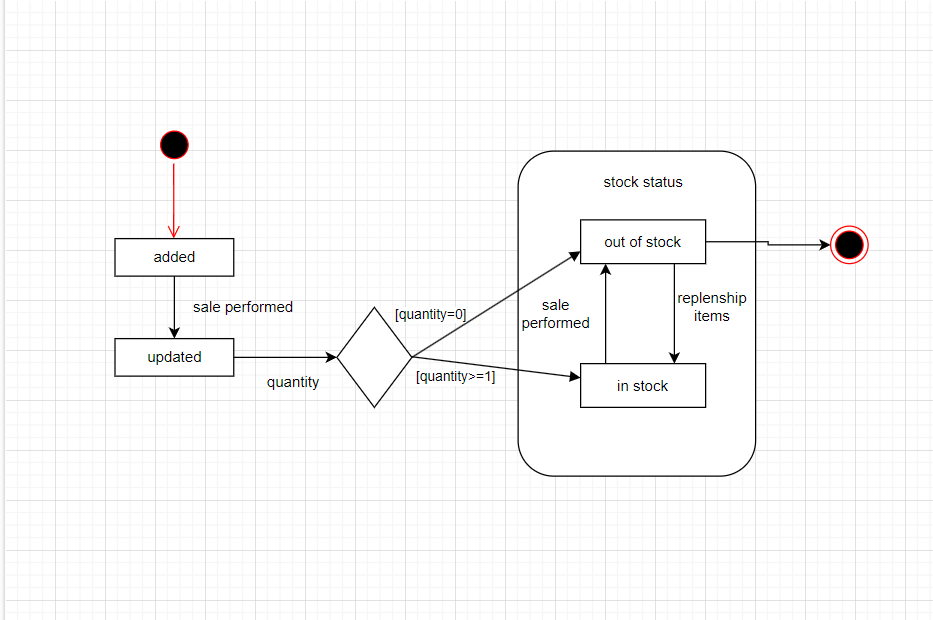
*Figure 3.2:Admin Delete Staff Communication Diagram*



*Figure 3.3:Financial Manager Update Price Communication Diagram*

## **State Machine Diagram**

The State Machine Diagram represents the states and transitions of the system.



*Figure 4:State Machine Diagram*

## **Class Diagram**

The Class Diagramdefines the structure of the system.

**Classes**:

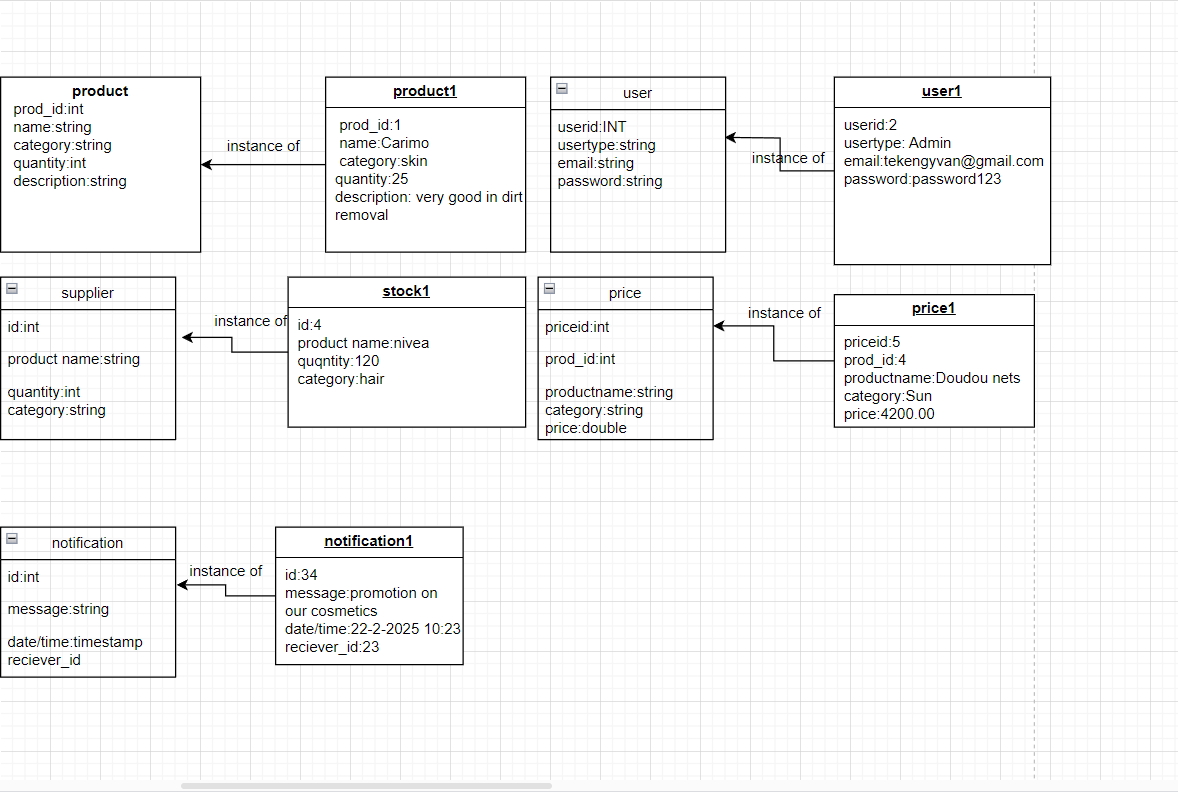
* User (Abstract)
* Super Admin (extends User)
* Admin (extends User)
* Staff [Stock Manager, Warehouse Staff, Financial Manager] (extends User)
* Notification
* Product
* Stock
* Price



*Figure 5:Class Diagram*

## **Object Diagram**

The Object Diagram shows instances of classes.



## **Package Diagram**

The Package Diagram organizes the system into modules.

**Packages**:

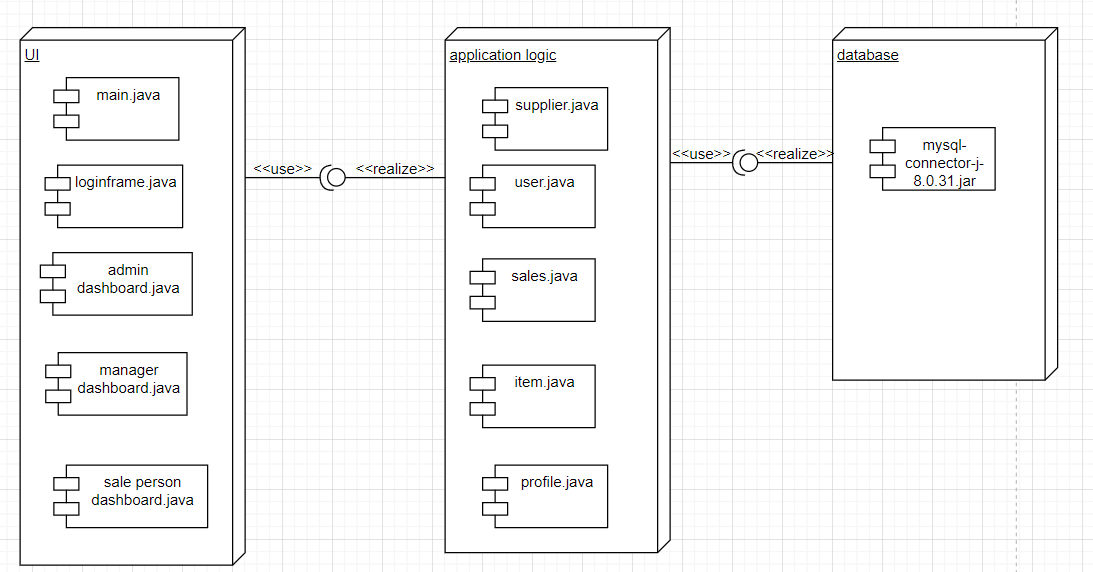
* User Management (Admin, Super Admin)
* Stock Management (Warehouse Staff)
* Price Management (Financial Manager)
* Product Management (Stock Manager)
* Notification Management
* Authentication Management



*Figure 7:Package Diagram*

## **Component Diagram**

The Component Diagram shows the system's components.



**managep**.**java**

**Stock**.**java**

**pass**.**java**

**ware**.**java**

## **Deployment Diagram**

The Deployment Diagram describes the system's hardware and software environment.

**Node**:

Client (NetBeans IDE, JDK)

Server (Database Server, Application Server)



*Figure 9:Package Diagram*

# **Database Schema**

The database consists of the following tables:

* 1. **Users Table**: Stores information about Admins and Staffs.
  2. **Stock Managers Table**: Stores information about stock managers.
  3. **Warehouse Staff Table**: Stores information about Warehouse staffs.
  4. **Financial Stock Managers Table**: Stores information about financial managers.
  5. **Product Table**: Stores information about Products.
  6. **Stock Table**: Tracks the quantity of items in stock.
  7. **Price Table**: Stores information of price about particular products.

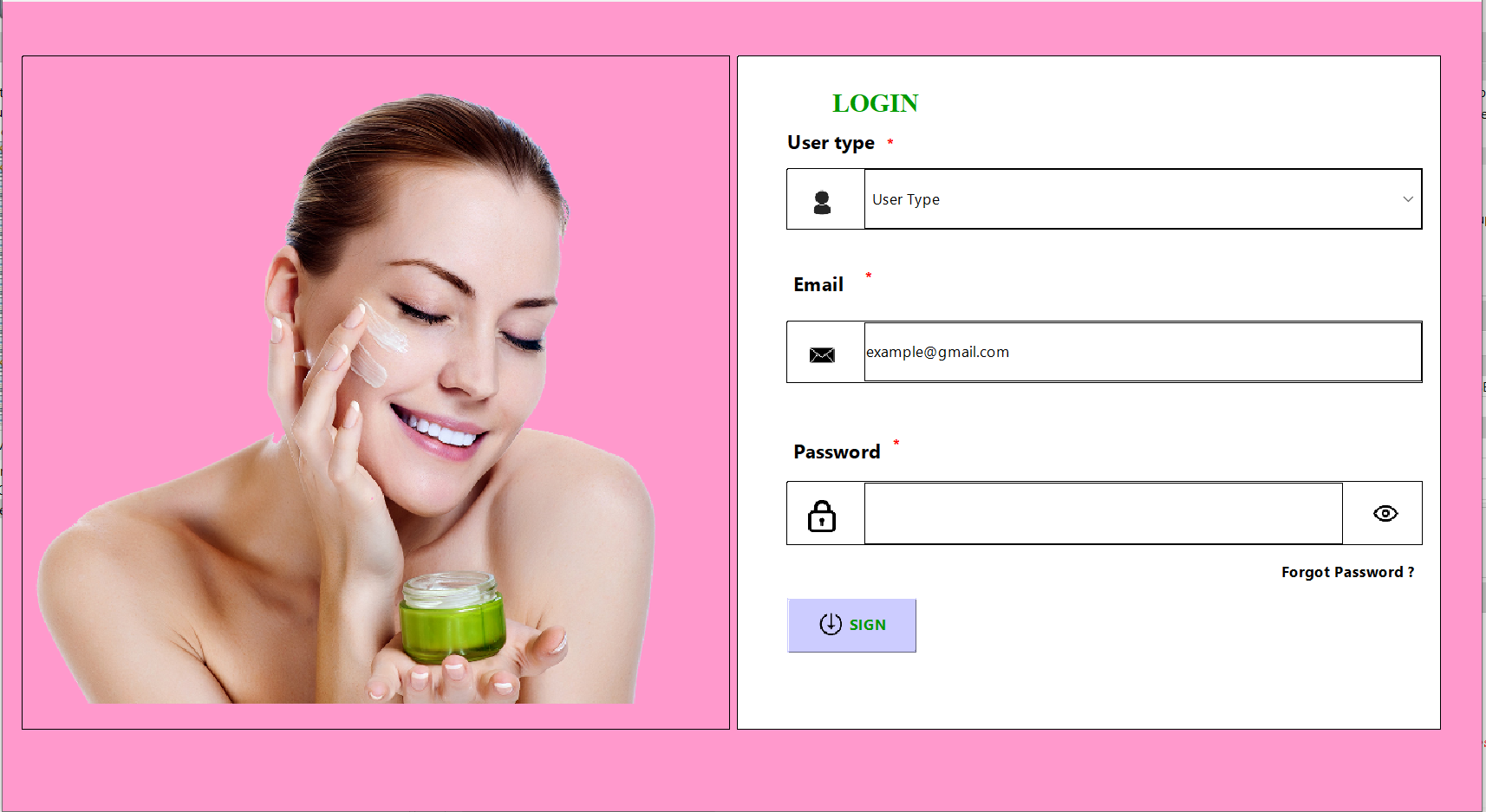
# **Java Code Implementation**

The system is implemented in Java, with classes for each entity (e.g., User, Super Admin, Admin, Staffs such as Stock Manager, Warehouse Staff, Financial Manager ,Product, Stock, Price, Notification). The Database class handles database connections, and DAO (Data Access Object) classes manage CRUD operations.

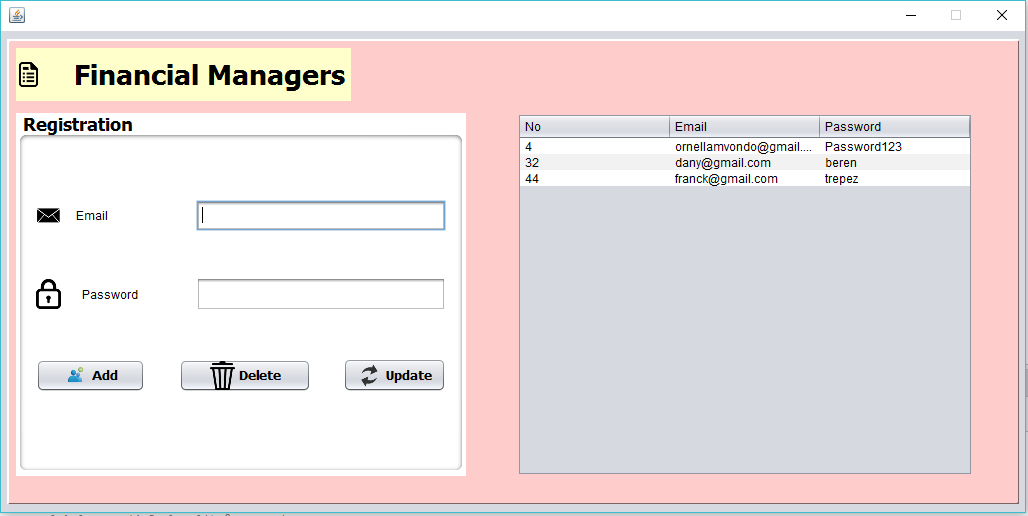
# C:\Users\USER\Desktop\mer\a.PNG

# **Some Screenshots of the Application**

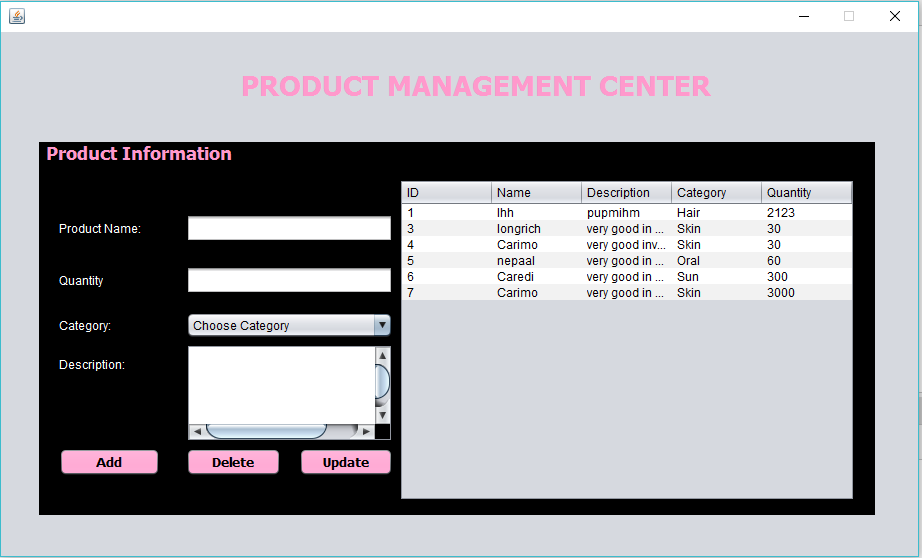
Here we have the Login page Add Admin page, change password, product management and price management



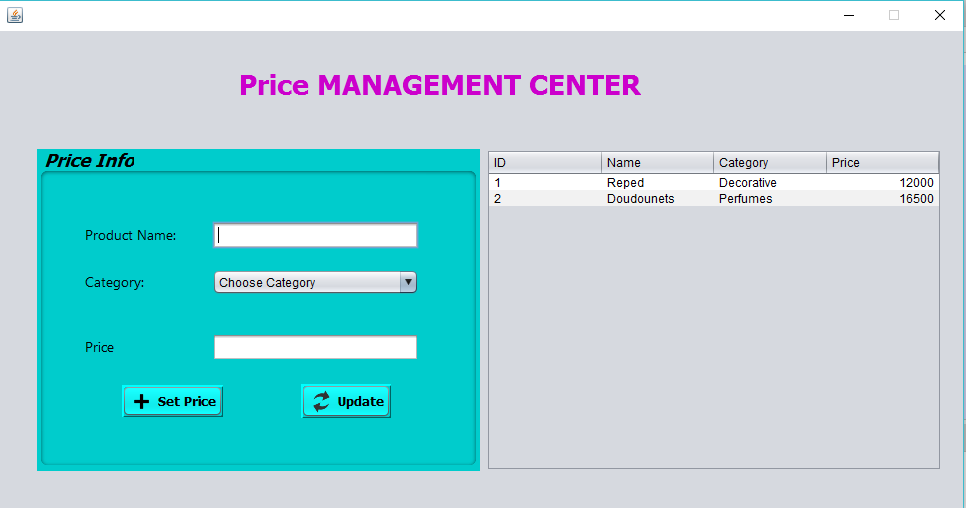
# *Login page form*



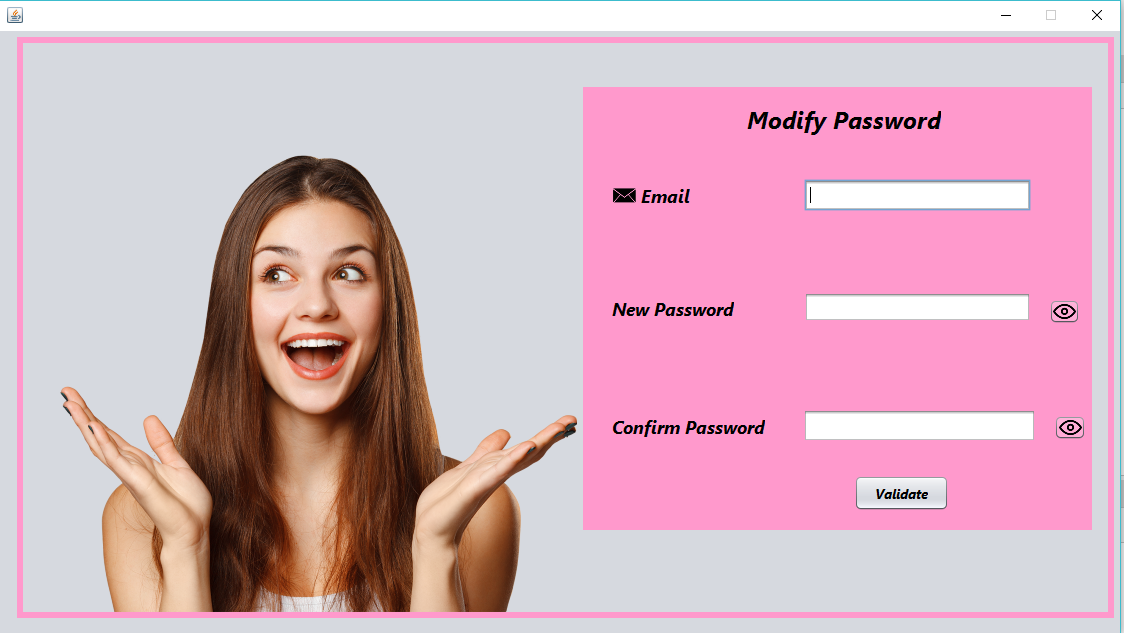
*Add financial Manager form*



***Product Management form***



***Price Management form***



# **CONCLUSION**

**CONCLUSION**

In the fast evolving cosmetic and skin care industry, effective stock management is not just a logical necessity but a strategic advantage. A dedicated stock management application offers a comprehensive solution to the challenges of inventory control enabling businesses to respond swiftly to consumer trends and preferences. The UML diagrams provide a detailed overview of the system's structure and behaviour, while the database schema and Java code ensure efficient data management and system functionality.

This system can be further enhanced by adding features such as automated restocking, advanced reporting, and integration with external suppliers. Overall, it provides a robust foundation for managing cosmetics and skin care in an organized and efficient manner.