Mini project report on

e- Commerce System

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology in

Computer Science &

Engineering

UE22CS351A – DBMS

Project

Submitted by:

Madhumitha V	**PES2UG22CS290**
M K Sumana	**PES2UG22CS282**

Under the guidance of

Prof. Shilpa S and Dr. Suja C M

Assistant Professor

PES University

AUG - DEC 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

CERTIFICATE

This is to certify that the mini project entitled

University Fest Management System

is a bonafide work carried out by

Madhumitha V	**PES2UG22CS290**
M K Sumana	**PES2UG22CS282**

In partial fulfilment for the completion of fifth semester DBMS Project (UE22CS351A) in the Program of Study Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period AUG. 2024 – DEC. 2024. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The project has been approved as it satisfies the 5th semester academic requirements in respect of project work.

Signature
Prof. Shilpa S and Dr. Suja C M
Assistant Professor

DECLARATION

We hereby declare that the DBMS Project entitled **e- commerce System** has been carried out by us under the guidance of **Prof. Shilpa S, Assistant Professor and Dr. Suja C M, Assistant Professor ,** submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester AUG – DEC 2024.

Madhumitha V	**PES2UG22CS290**
M K Sumana	**PES2UG22CS282**

ABSTRACT

The primary goal of this system is to provide a secure, user-friendly platform that facilitates online shopping by allowing users to browse products, compare prices, manage a shopping cart, and complete purchases through multiple payment methods. E-commerce has made it easier for human to reduce physical work and to save time. It is leading a complete change in traditional way of doing business. This significant change in business model is witnessing a tremendous growth around the globe.
The current research has been undertaken to describe the scenario of E-Commerce, analyse the trends of E-Commerce. The following document serves as a guide to the developers, project managers, testers and the stakeholders involved in the project. The study further examines the key features such as user registration and authentication, product browsing, shopping cart management, and order processing are detailed with specific requirements. Additionally, nonfunctional requirements addressing performance, security, and quality are specified to ensure the system's reliability and compliance with industry standards like PCI DSS.

TABLE OF CONTENTS

1. Introduction

1.1 Purpose

This SRS document specifies the software requirements for the development of an e-commerce website. The system enables users to browse products, add them to the shopping cart, and complete the purchase through various payment methods.

1.2 Intended Audience and Reading Suggestions

This document is intended for developers, project managers, testers, marketing staff, users, and documentation writers involved in the e-commerce website project. Each section provides specific information relevant to the respective audience.

1.3 Product Scope

The e-commerce website aims to provide a secure, user-friendly platform for online shopping. It will support product browsing, order placement, payment processing, and order tracking, aligning with the organization's goal of expanding its online retail presence.

2. Product description and requirements

2.1 Product Perspective

The e-commerce website is a new, self-contained product designed to integrate with existing payment gateways, inventory management systems, and logistics providers. The system will interact with external APIs for payment processing and order fulfilment.

2.2 Product Functions

- User registration and authentication.
- Product browsing and filtering.
- Shopping cart management.
- Order placement and payment processing.
- Order tracking and notifications.
- Customer support.

2.3 User Classes and Characteristics

- **Customers:** General users who will browse and purchase products.
- **Admin:** Users responsible for managing products, orders, and user accounts.

2.4 Operating Environment

The website will be accessible on desktop, tablet, and mobile devices, compatible with web browsers like Chrome, Firefox, and Safari, and will operate on major OS platforms (Windows, macOS, iOS, Android).

2.5 Design and Implementation Constraints

- The system must comply with PCI DSS for secure payment processing.
- It must be designed with accessibility standards like WCAG 2.1.
- The system should use SSL/TLS for secure data transmission.

2.6 Assumptions and Dependencies

- The availability of reliable internet connectivity for users.
- Integration with existing payment gateways and inventory systems.

2.7. External Interface Requirements

2.7.1 User Interfaces

- Responsive design compatible with various devices.
- Simple and intuitive navigation with search and filter options.
- Secure login and checkout processes.

2.7.2 Software Interfaces

- APIs for payment gateways (e.g., PayPal, Stripe).
- Integration with inventory management systems for real-time stock updates.

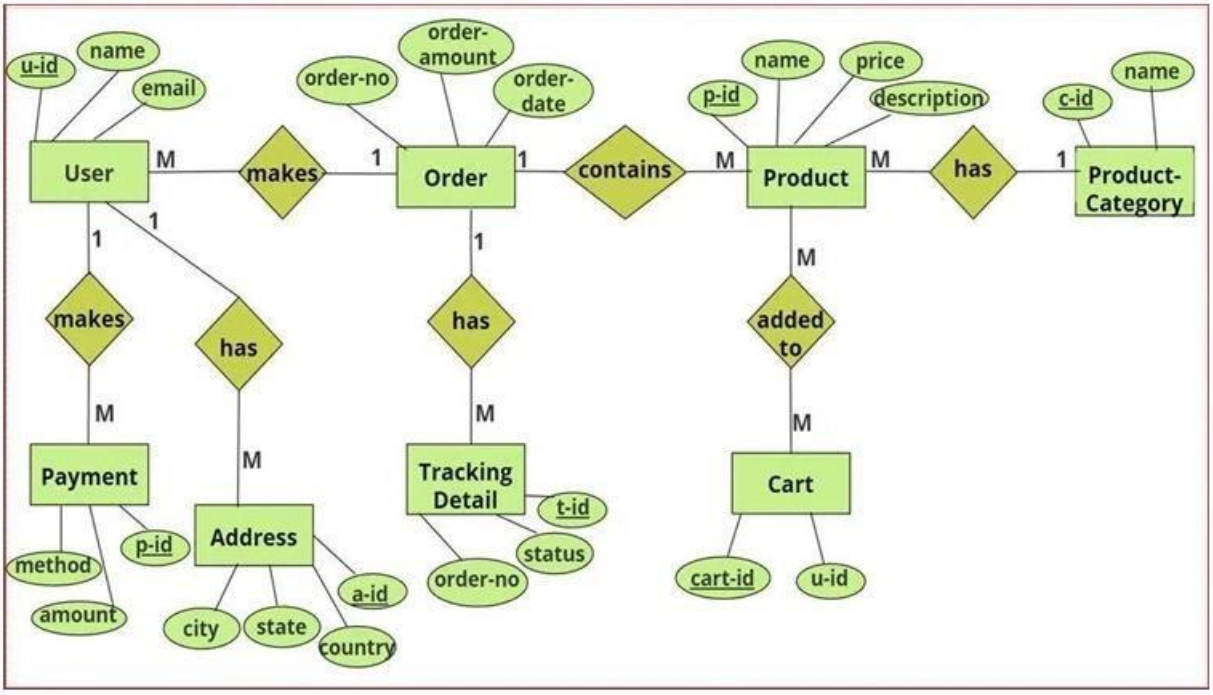
2.7.3 Communication Interfaces

- Secure communication using SSL/TLS protocols.
- Integration with email services for order confirmations and updates.

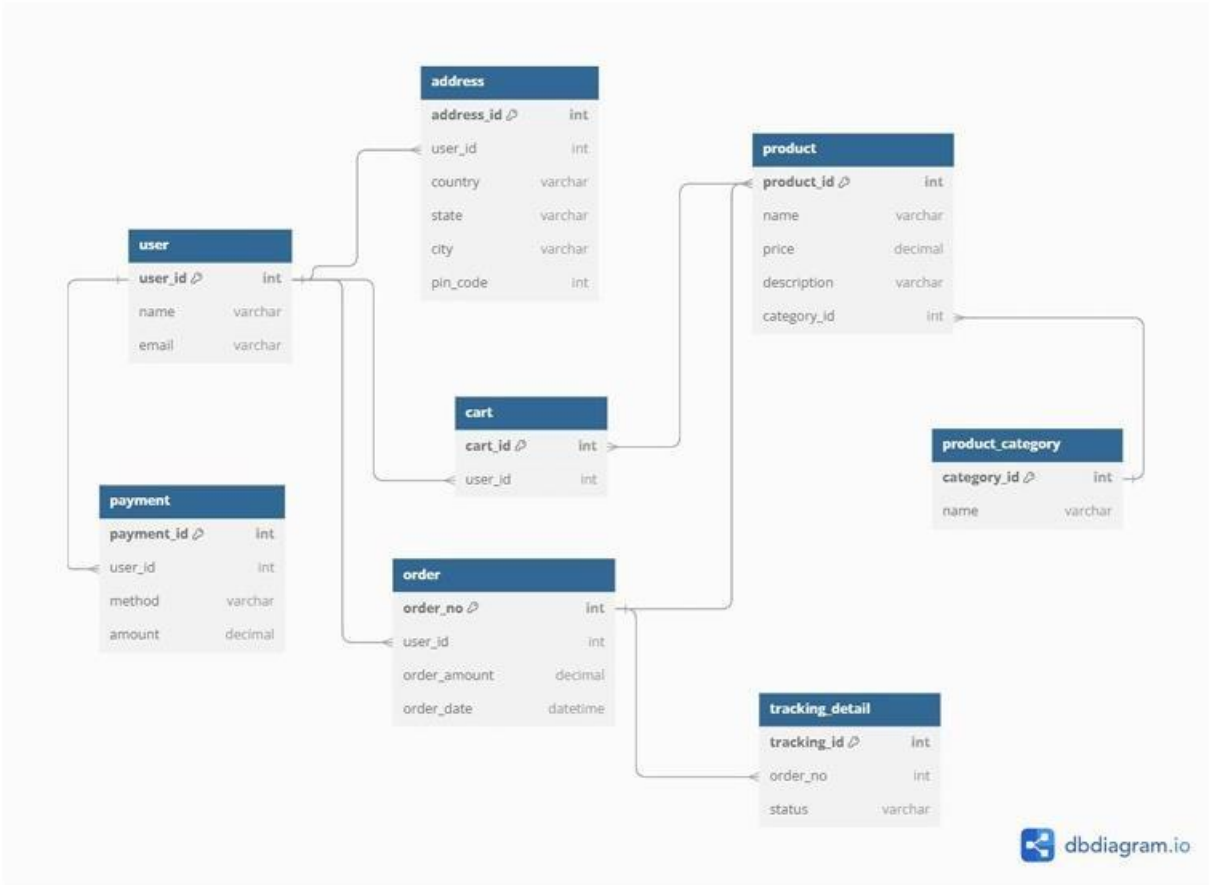
3. Software tools:

- MySQL
- Python Tkinter

4. ER Diagram



5. Relational schema



6. DDL statements

```
CREATE DATABASE ecom2;
USE ecom2;
-- Create user table
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    user_id VARCHAR(255) UNIQUE,
    name VARCHAR(50),
    pno int UNIQUE,
    email VARCHAR(50) UNIQUE,
    address varchar(255),
    password VARCHAR(25),
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

```
CREATE TABLE product_category (
    id INT AUTO_INCREMENT PRIMARY KEY,
    cat_id VARCHAR(100) UNIQUE, -- Matches type with product table
    cat_name VARCHAR(200) UNIQUE
);
```

```
CREATE TABLE order_products (
    order_no INT,
    product_id varchar(90),
    quantity INT,
    PRIMARY KEY (order_no, product_id),
    FOREIGN KEY (order_no) REFERENCES order_table(order_no),
    FOREIGN KEY (product_id) REFERENCES product(product_id)
);
```

```
TRUNCATE TABLE product;
```

```sql
CREATE TABLE product (
    id INT AUTO_INCREMENT PRIMARY KEY,
    product_id VARCHAR(255) UNIQUE,
    name VARCHAR(255),
    price DECIMAL(10, 2),
    img_link varchar(200),
    category_id VARCHAR(100),
    category_name VARCHAR(200),
    FOREIGN KEY (category_id) REFERENCES product_category(cat_id),
    FOREIGN KEY (category_name) REFERENCES product_category(cat_name)
);
```

```sql
CREATE TABLE cart (
    cart_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id varchar(100),
    product_id varchar(100),
    quantity int,
    FOREIGN KEY (product_id) REFERENCES product(product_id),
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

```sql
CREATE TABLE order_table(
    order_no INT AUTO_INCREMENT PRIMARY KEY,
    user_id varchar(90),
    order_amount DECIMAL,
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);
```

7. DML statements

```
insert into product_category(cat_id,cat_name) values
('CAT001','electronics'),
('CAT002','fashion'),
('CAT003','footwear'),
('CAT004','grocery');
```

```
INSERT INTO product (name, price, img_link, category_id, category_name) VALUES
('Product 1', 10.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\E1.png','CAT001','electronics'),
('Product 2', 20.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\E2.png','CAT001','electronics'),
('Product 3', 30.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\E3.png','CAT001','electronics'),
('Product 4', 40.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\E4.png','CAT001','electronics'),
('Product 5', 50.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\F1.png','CAT002','fashion'),
('Product 6', 60.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\F2.png','CAT002','fashion'),
('Product 7', 70.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\F3.png','CAT002','fashion'),
('Product 8', 90.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\F4.png','CAT002','fashion'),
('Product 9', 100.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\FW1.png','CAT003','footwear'),
('Product 10', 20.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\FW2.png','CAT003','footwear'),
('Product 11', 10.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\FW3.png','CAT003','footwear'),
('Product 12', 30.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\FW4.png','CAT003','footwear'),
('Product 13', 80.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\G1.png','CAT004','grocery'),
('Product 14', 20.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\G2.png','CAT004','grocery'),
('Product 15', 70.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\G3.png','CAT004','grocery'),
('Product 16', 200.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\G4.png','CAT004','grocery'),
('Product 17', 90.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\G5.png','CAT004','grocery'),
('Product 18', 30.00, 'C:\\Users\\MADHUMITHA V\\CSE\\SEM-5\\DBMS\\project_rough\\G6.png','CAT004','grocery');
```

```
if category:
    query = "SELECT product_id ,name, price, img_link, category_id, category_name FROM product WHERE category_name LIKE %s"
    cursor.execute(query, ('%' + category + '%',))
else:
    query = "SELECT product_id ,name, price, img_link, category_id, category_name FROM product"
    cursor.execute(query)

products = cursor.fetchall()
```

```
if existing_item:
    cursor.execute("UPDATE cart SET quantity = quantity + 1 WHERE user_id = %s AND product_id = %s", (current_user_id, product_id))
else:
    cursor.execute("INSERT INTO cart (user_id, product_id, quantity) VALUES (%s, %s, 1)", (current_user_id, product_id))
```

```
def remove_from_cart(product_id, user_id, connection):
    try:
        # Step 1: Remove the product from the cart in the database
        cursor = connection.cursor()
        cursor.execute("DELETE FROM cart WHERE user_id = %s AND product_id = %s", (user_id, product_id))
        connection.commit()

        # Step 2: Refresh the cart page to remove the product from the UI
        load_cart_page(user_id, connection)

        messagebox.showinfo("Success", "Product removed from cart!")
```

8. Queries

```
42      SELECT
43          u.name AS user_name,
44          SUM(p.price * c.quantity) AS total_cart_value
45      FROM cart c
46      JOIN users u ON c.user_id = u.user_id
47      JOIN product p ON c.product_id = p.product_id
48      GROUP BY u.user_id, u.name
49      ORDER BY total_cart_value DESC;
50
```

user_name	total_cart_value
RAHUL	300.00
ARJUN	260.00
CHARLIE	60.00
SUMANA	30.00

```sql
SELECT u.name AS user_name,p.name AS product_name,pice AS product_price,c.quantity
FROM cart c
JOIN users u ON c.user_id = u.user_id
JOIN product p ON c.product_id = p.product_id
ORDER BY u.name, p.name;
```

```sql
SELECT p.category_name,p.name AS product_name,p.price
FROM product p
WHERE p.price = (
        SELECT MAX(price)
        FROM product p2
        WHERE p2.category_name = p.category_name
    )
ORDER BY p.category_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

category_name	product_name	price
electronics	Product 4	40.00
fashion	Product 8	90.00
footwear	Product 9	100.00
grocery	Product 16	200.00

9. Procedures, functions and triggers

```
DELIMITER $$
CREATE PROCEDURE GetTotalUserSpending(
    IN user_id INT)
BEGIN
    SELECT u.name AS UserName,SUM(op.quantity * p.price) AS TotalSpent
    FROM order_table o
    INNER JOIN order_products op ON o.order_no = op.order_no
    INNER JOIN product p ON op.product_id = p.product_id
    INNER JOIN users u ON o.user_id = u.user_id
    WHERE u.user_id = user_id;
END $$
DELIMITER ;
CALL GetTotalUserSpending(1);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap

cart_id	user_id	product_id	quantity
15	USR001	PROD000023	1
16	USR001	PROD000026	1
17	USR004	PROD000001	1
18	USR004	PROD000021	1
19	USR004	PROD000025	1
20	USR004	PROD000026	1
21	USR004	PROD000024	1

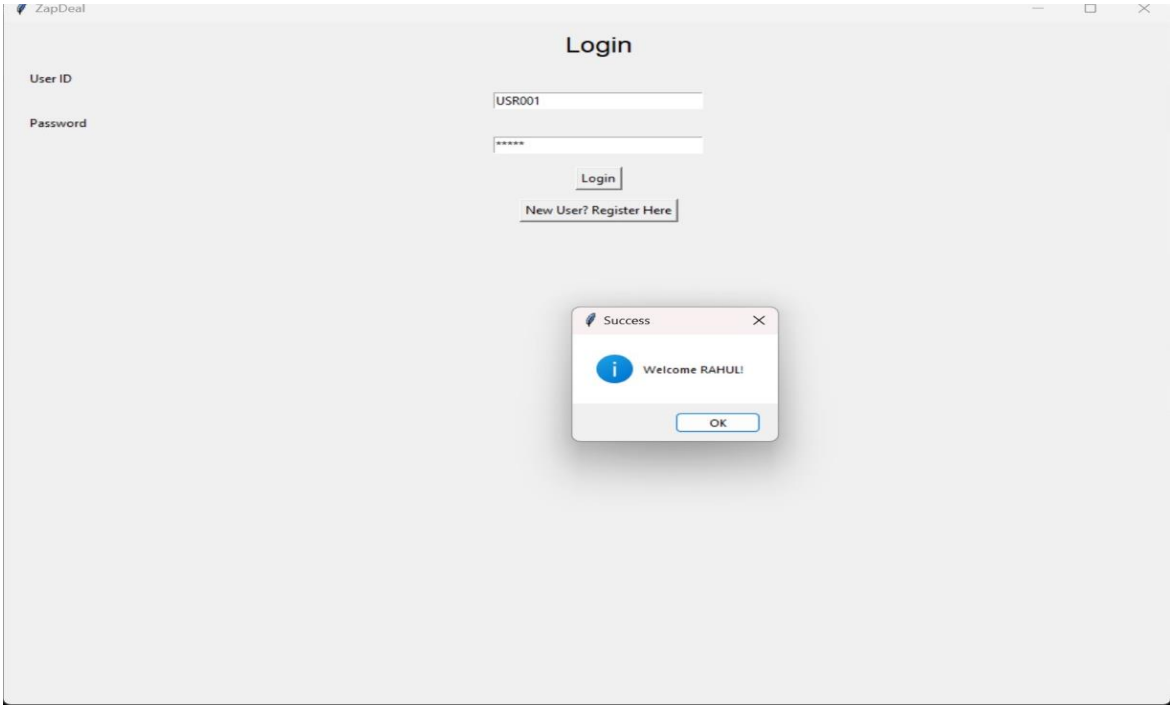
```
DELIMITER //
CREATE FUNCTION calculate_total(order_no INT)
RETURNS DECIMAL(10,2)
BEGIN
  RETURN (
    SELECT SUM(p.price * op.quantity)
    FROM products p
    JOIN order_products op ON p.product_id = op.product_id
    WHERE op.order_no = order_no
  )
END;
DELIMITER //
```

```
DELIMITER //
CREATE TRIGGER before_product_insert
BEFORE INSERT ON product
FOR EACH ROW
BEGIN
    DECLARE max_id INT;
    SELECT IFNULL(MAX(id), 0) + 1 INTO max_id FROM product;
    SET NEW.product_id = CONCAT('PROD', LPAD(max_id, 6, '0'));
END//

DELIMITER ;
```

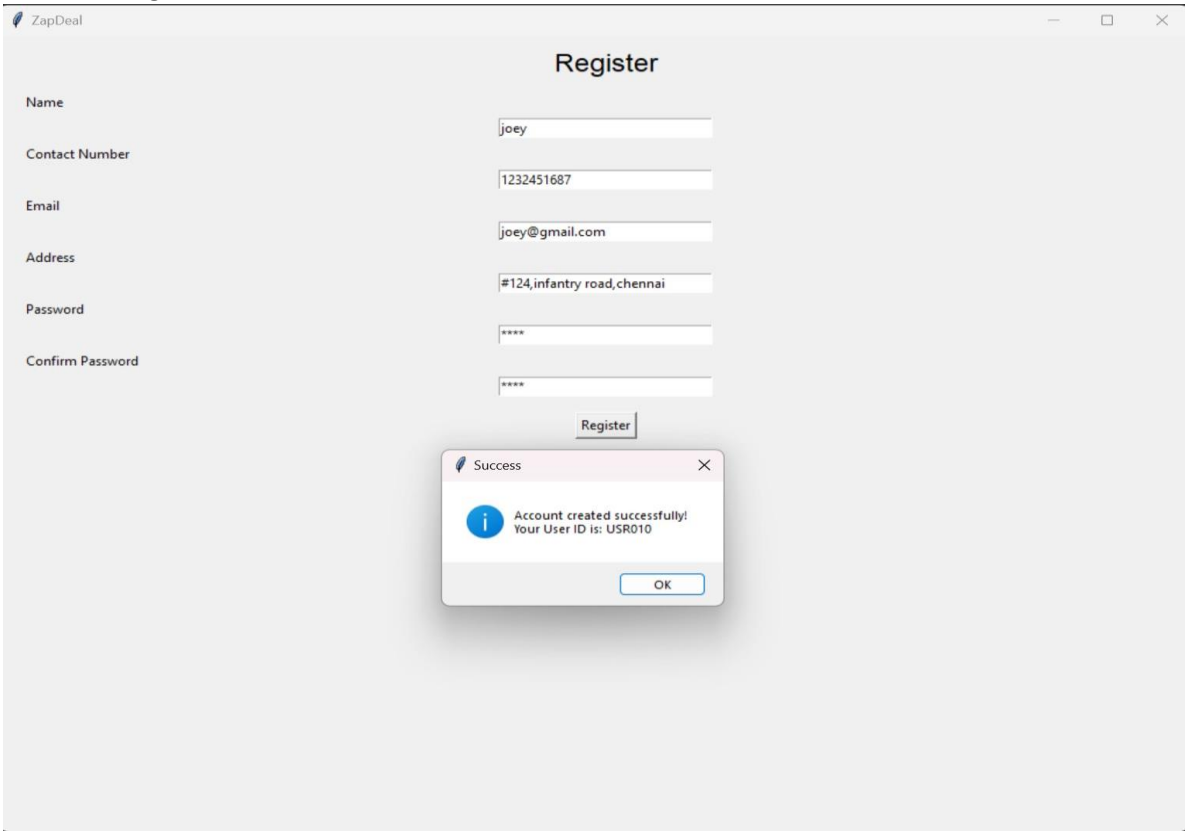
```
DELIMITER //

CREATE TRIGGER before_user_insert
BEFORE INSERT ON users
FOR EACH ROW
BEGIN
    IF NEW.user_id IS NULL THEN
        SET NEW.user_id = CONCAT('USR', LPAD((SELECT COUNT(*) FROM users) + 1, 3, '0'));
    END IF;
END//

DELIMITER ;
```

10. Front end

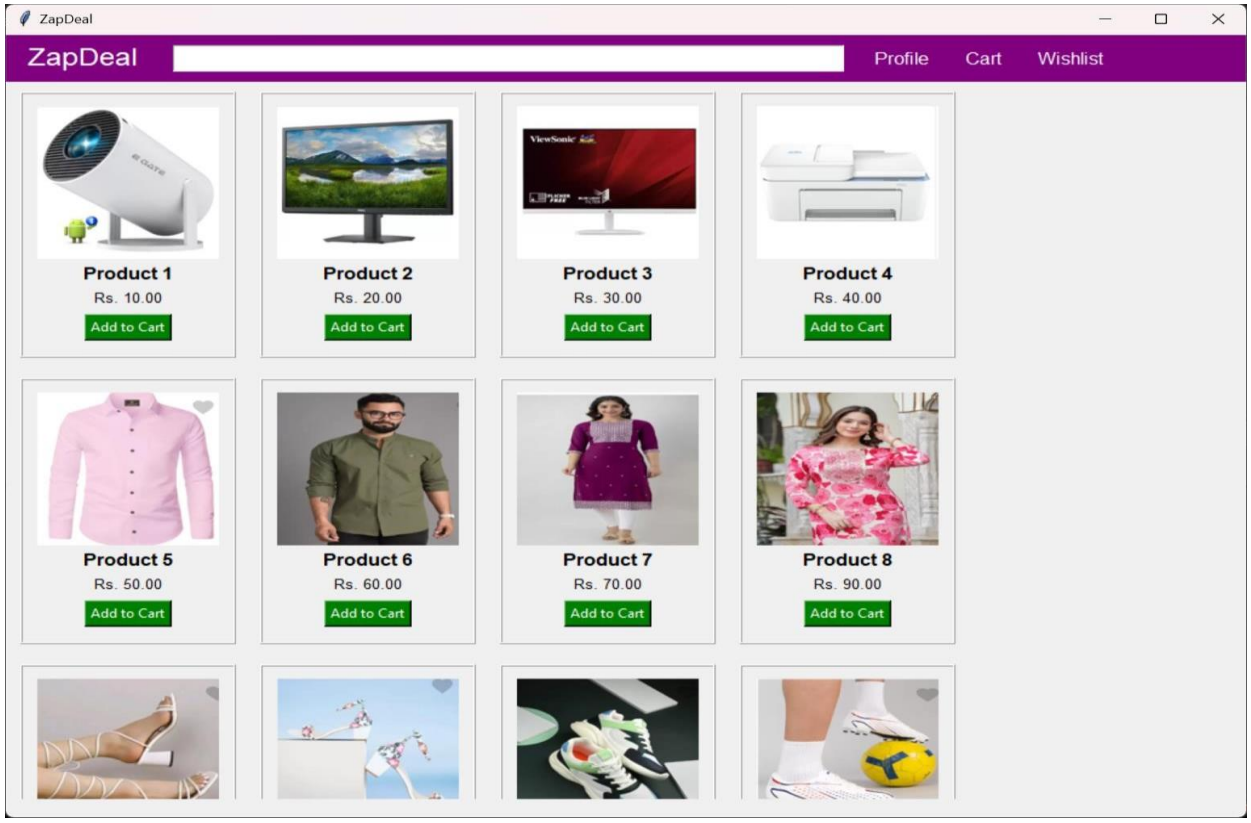
Login page



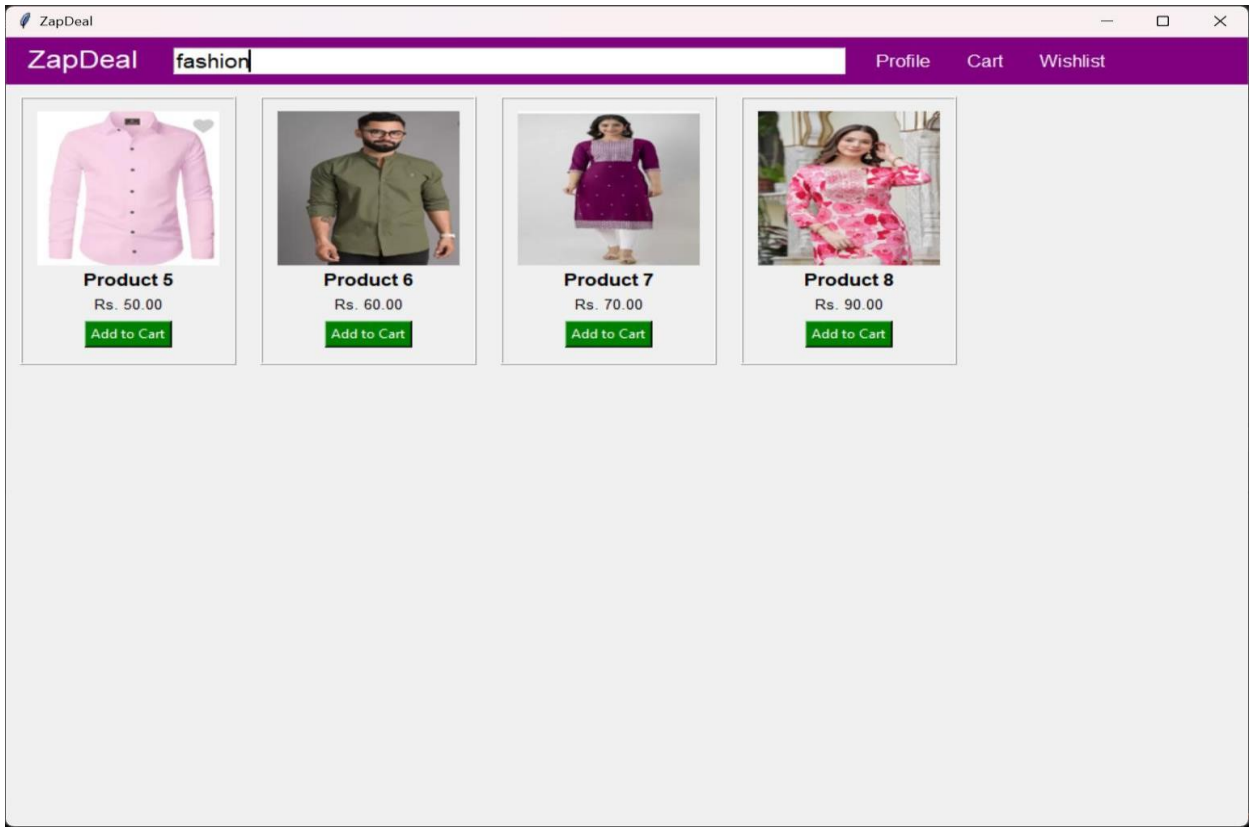
New user registration



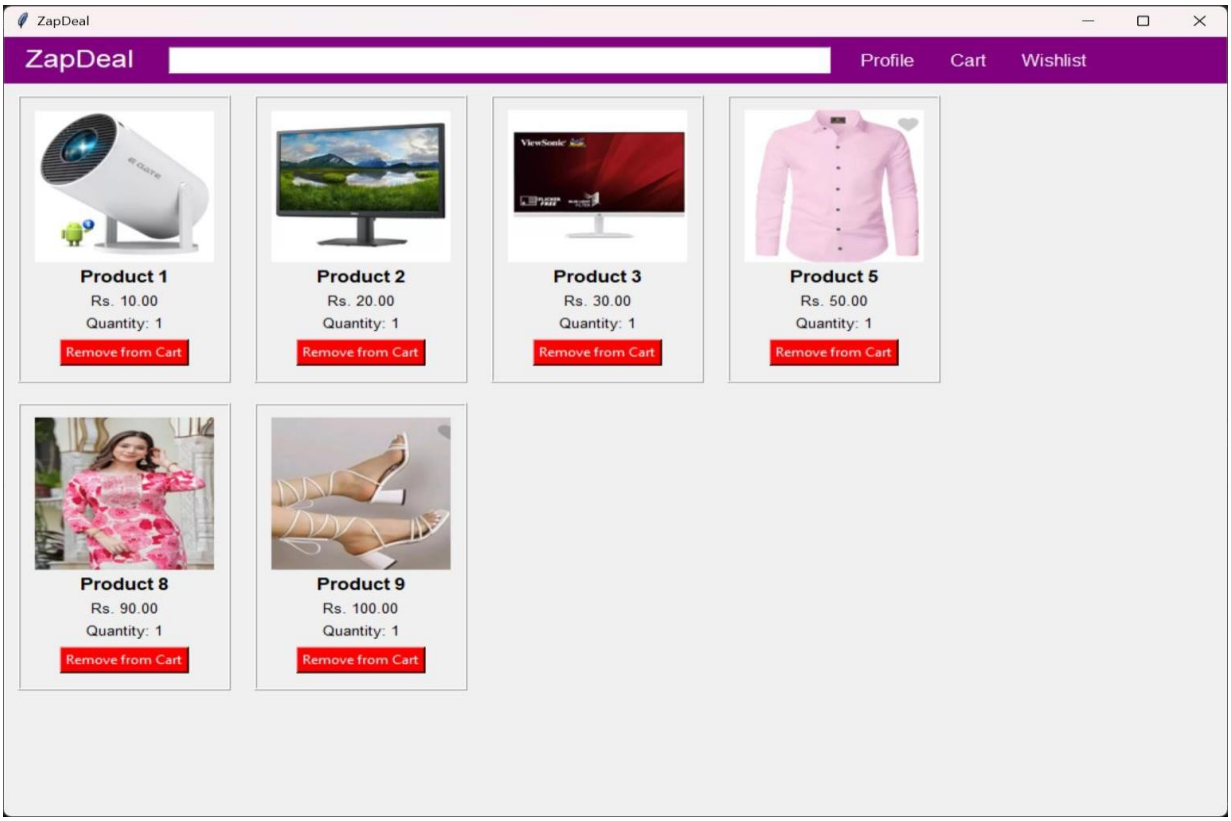
Home page



Search bar



cart



profile