Ahmedabad University

Winter Semester 2023
CSE250 Database Management System

Project Title: E-commerce management system

Submitted to Prof. Shefali Naik

Details of Group

Roll Number	Name of Student
AU2140160	Khwahish Patel
AU2140167	Charmi Desai
AU2140214	Riya Patel
AU2140204	Priyal Patel

CONTENTS:

Description

In recent years, the growth of e-commerce has been exponential, with more and more people turning to online shopping for their daily needs. E-commerce has become an indispensable part of our lives, offering a convenient and hassle-free way of purchasing goods and services. However, managing an e-commerce platform is no easy feat. It requires a complex system of databases to keep track of orders, payments, products, and customers.

E-commerce platforms generate vast amounts of data related to customer transactions, product inventories, and order fulfillment. Managing this data effectively requires a robust and scalable database management system (DBMS) that can efficiently store and retrieve data, while also ensuring data integrity and security.

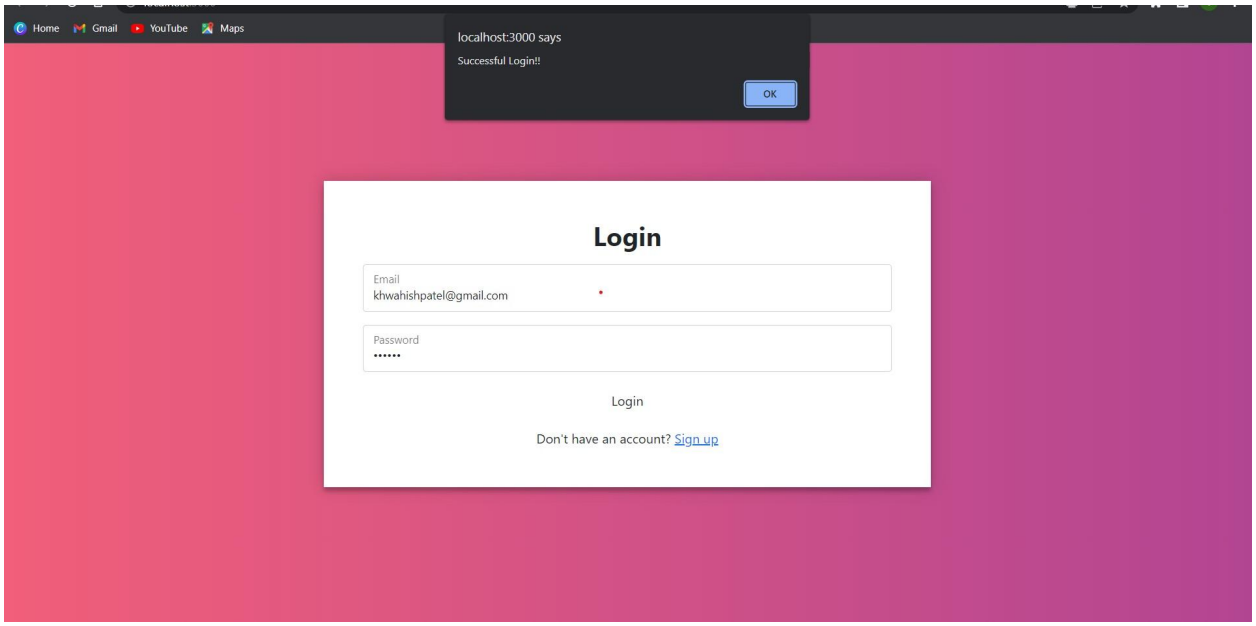
In this project, we aimed to design and implement a DBMS for an e-commerce platform using SQL and PHP. Our goal was to create a database schema that could handle complex relationships between data entities, while also providing efficient query processing and indexing to support high-volume data retrieval.

To achieve this, we focused on creating an optimized database schema that minimized redundancy and maximized data normalization, resulting in a more efficient use of storage space and improved performance. We also implemented various database constraints to ensure data consistency and prevent invalid data entries. The project aimed to provide a comprehensive solution to the challenges of managing an e-commerce platform using a scalable and secure DBMS. Our implementation can serve as a useful reference for others looking to develop similar systems or improve existing ones.

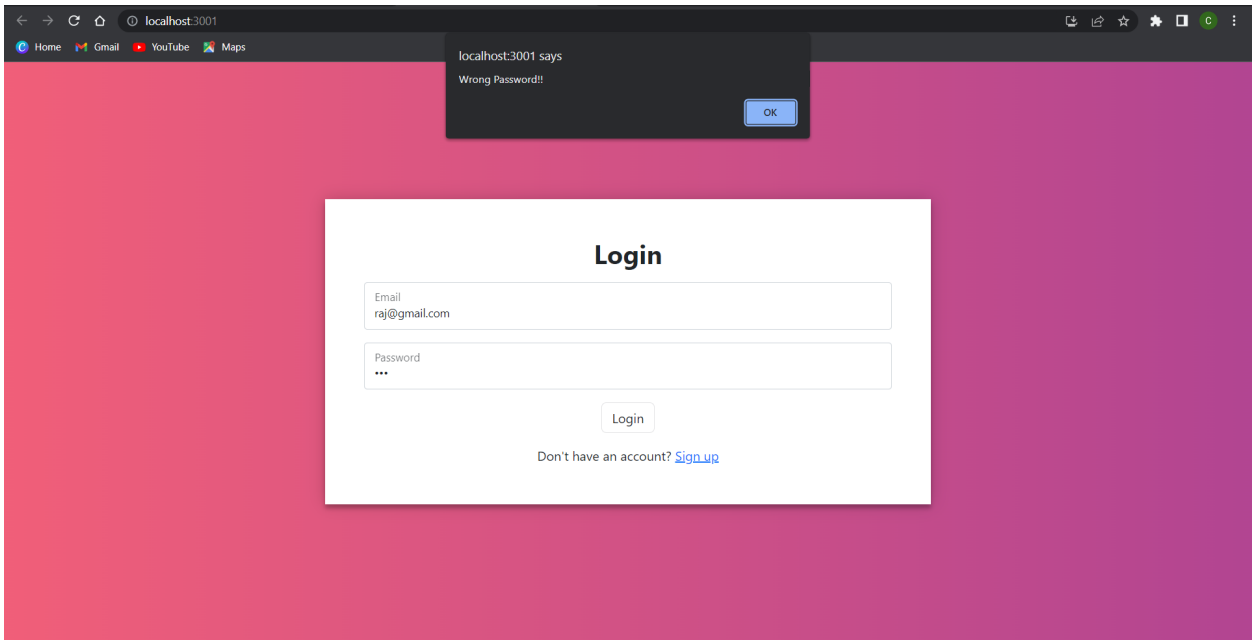
Modules

Login Page

Login page acts as the primary entry point for the system, where the user needs to provide their login credentials for authentication purposes. Upon accessing the login page, the user will be prompted to enter their username and password. After submitting the required login details, the system will verify the user's credentials against the database records. If the user's login information is correct, the system will grant them access to the e-commerce management system, allowing them to view and manage their account and other related activities. This process ensures secure and authorized access to the system. If they do not have the account then they are required to sign up first. Every user of the system would have different credentials since every user may have different rights of accessing certain functions. The trigger is displayed on the page where the message is shown saying that the user has successfully logged in.

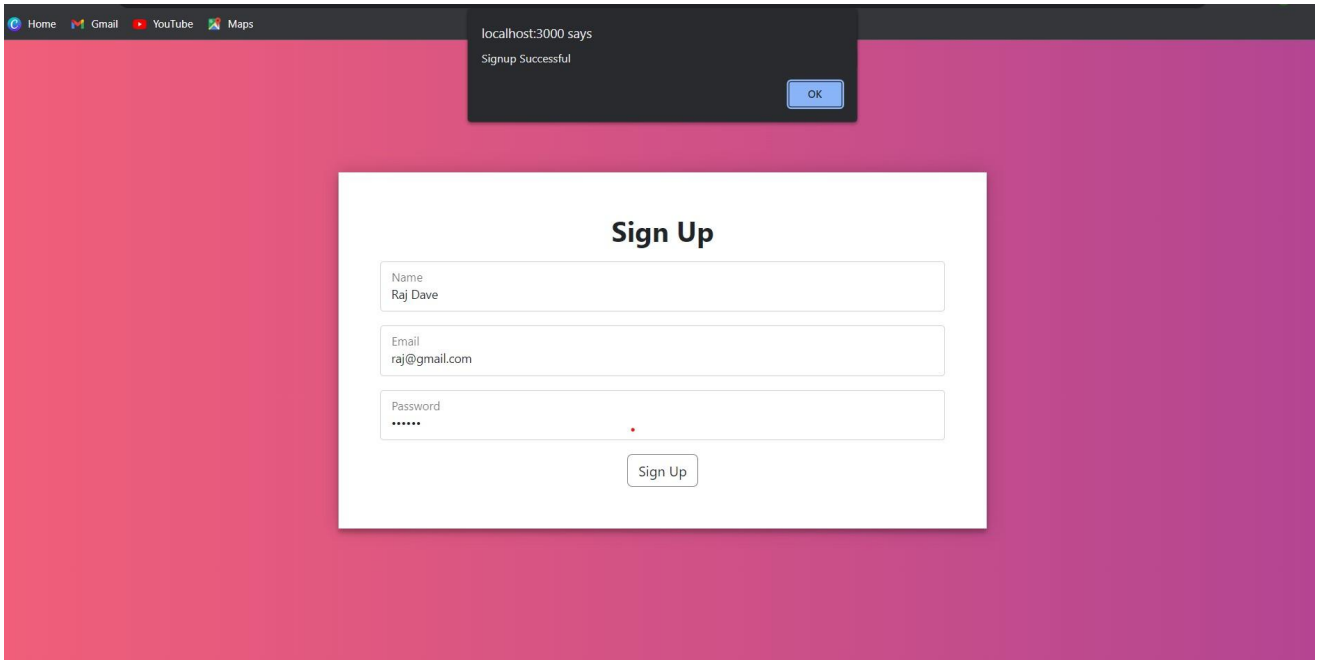


If a user attempts to log in with an incorrect password and does not have an account in the system, an error message will appear on the login page. This error message will notify the user that their login attempt was unsuccessful, and they will be prompted to try again or create a new account. This process prevents unauthorized access to the system and protects user data from potential security breaches.



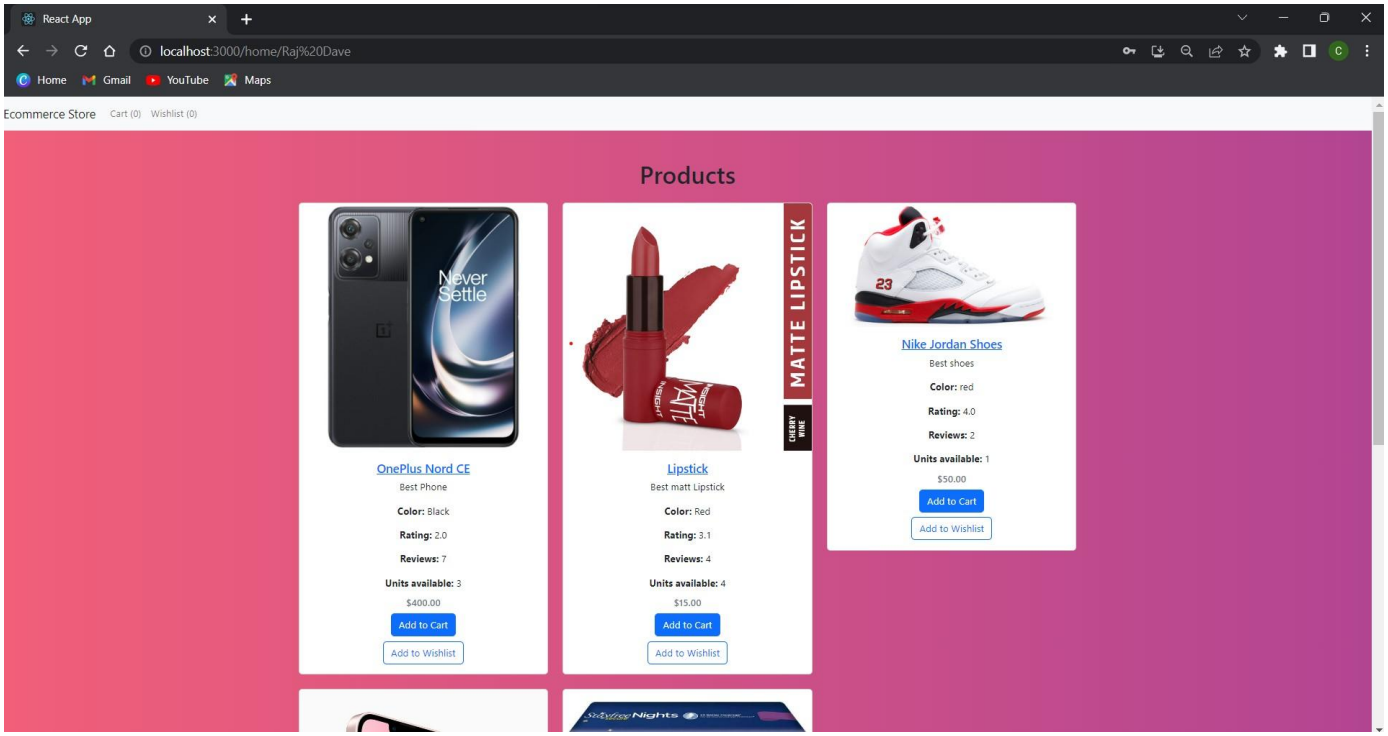
Sign Up Page

The signup page for the e-commerce management system is the first step in creating a new account for users. Once the user completes the signup form and submits their information, the system will validate the data and create a new account in the database. A success message will then appear on the signup page, notifying the user that their account has been successfully created. The user can then proceed to log in and access the e-commerce management system's features



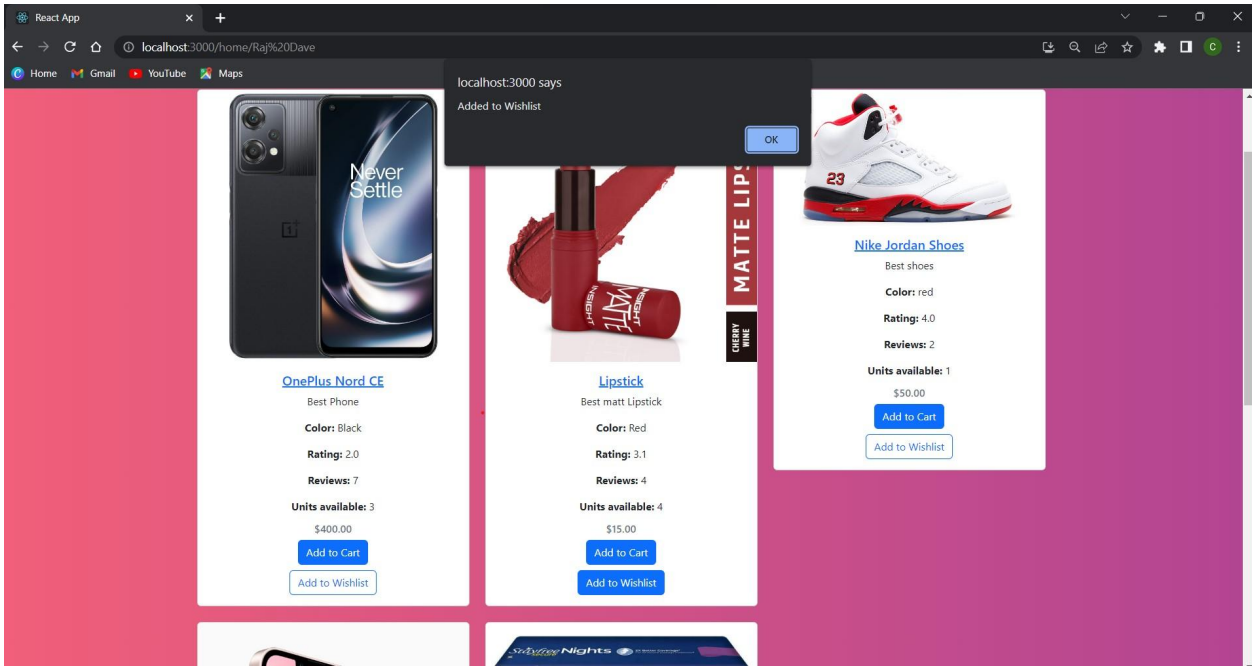
Home Page

The home page of the e-commerce management system serves as the main page for users, providing them with an overview of the available products and their prices.The page will display various products, along with their images, descriptions, and prices. Users can then browse and select the products they want to purchase by clicking on the product image or name. The home page is the starting point for users' shopping experience, making it a crucial aspect of the e-commerce management system. The products also show how many products are available in stock.

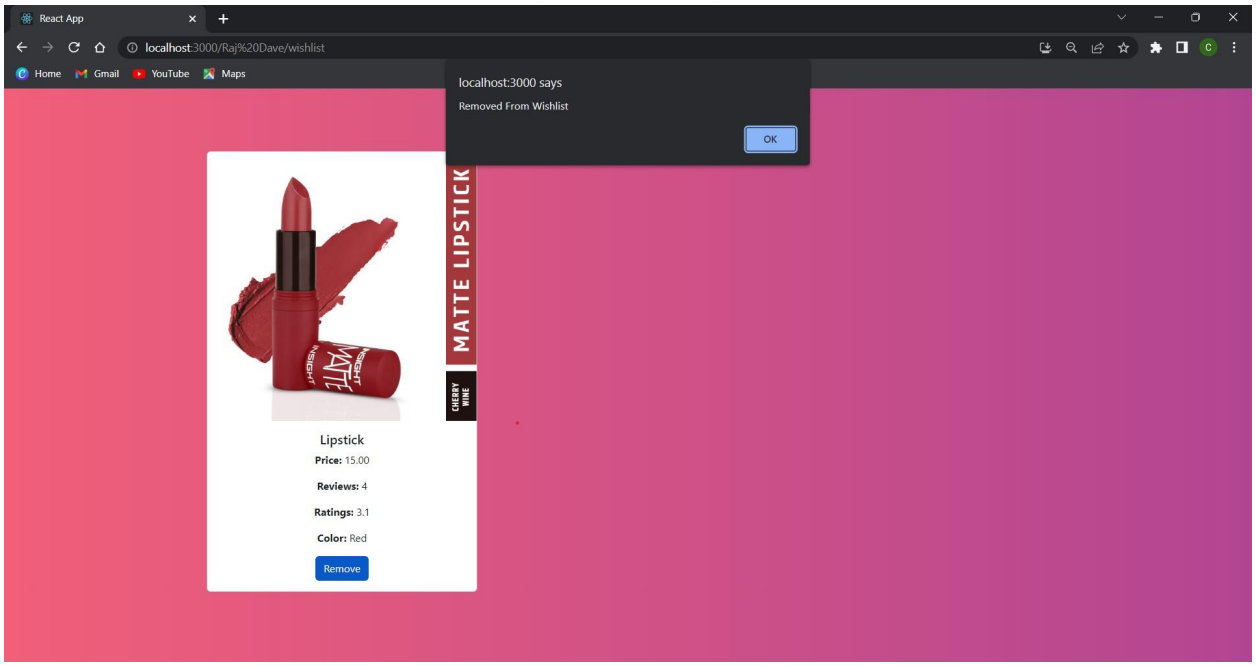


Wishlist Page

The wishlist feature of the e-commerce management system is a convenient tool for users to save products they are interested in purchasing for later. The wishlist page displays various products that the user has added to the list, along with their images, descriptions, and prices. When a user finds a product they want to save for later, they can add it to their wishlist by clicking the "Add to Wishlist" button located on the product page. This trigger will send a request to the database to add the product to the user's wishlist record and display the message of the product being added to the wishlist. Once added, the user can view their wishlist at any time by navigating to the wishlist page. The wishlist feature allows users to keep track of products they may not be ready to purchase yet or want to remember for future reference. It also provides a personalized shopping experience, making it easier for users to find and purchase the products they are interested in.

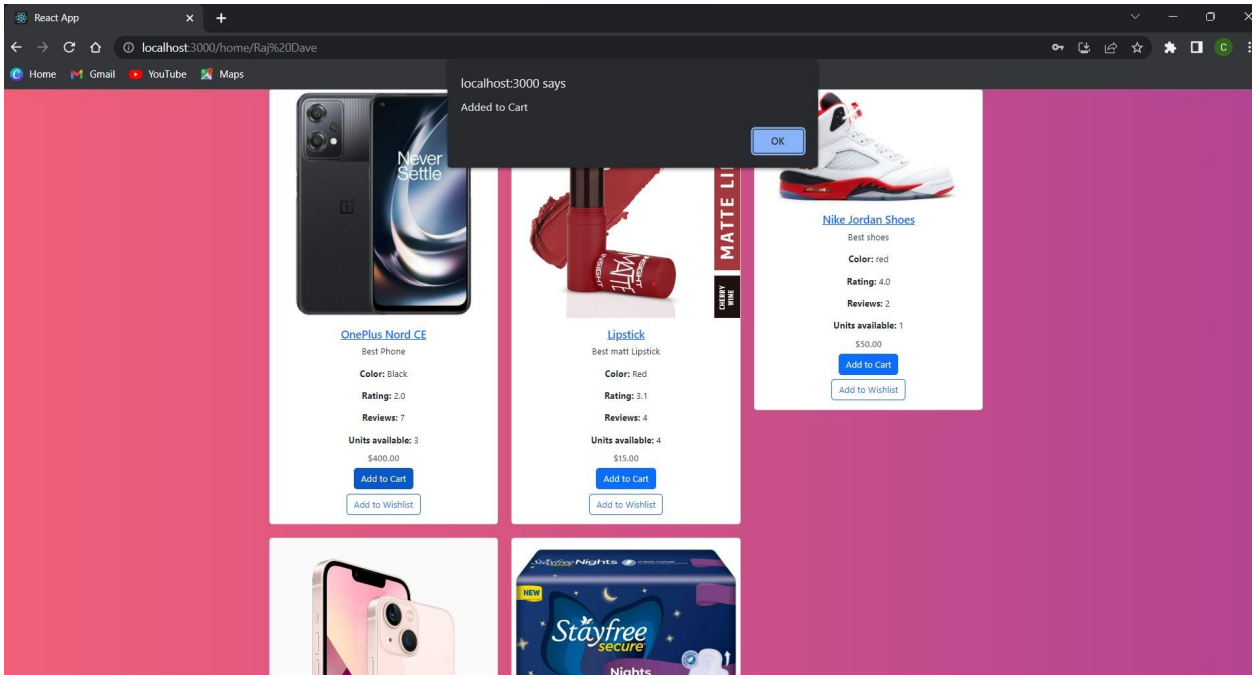


When a user no longer wants to save a product on their wishlist, they can remove it by clicking the "Remove" button located on the product's entry in their wishlist record. This will display a message on the screen showing that the product has been removed from the wishlist. This trigger will send a request to the database to remove the product from the user's wishlist record. Once removed, the user will no longer see the product on their wishlist. The ability to remove products from the wishlist is essential to keep the list up-to-date and relevant. Users may change their minds about a product or find it elsewhere, and the ability to remove products ensures the wishlist remains useful and organized.



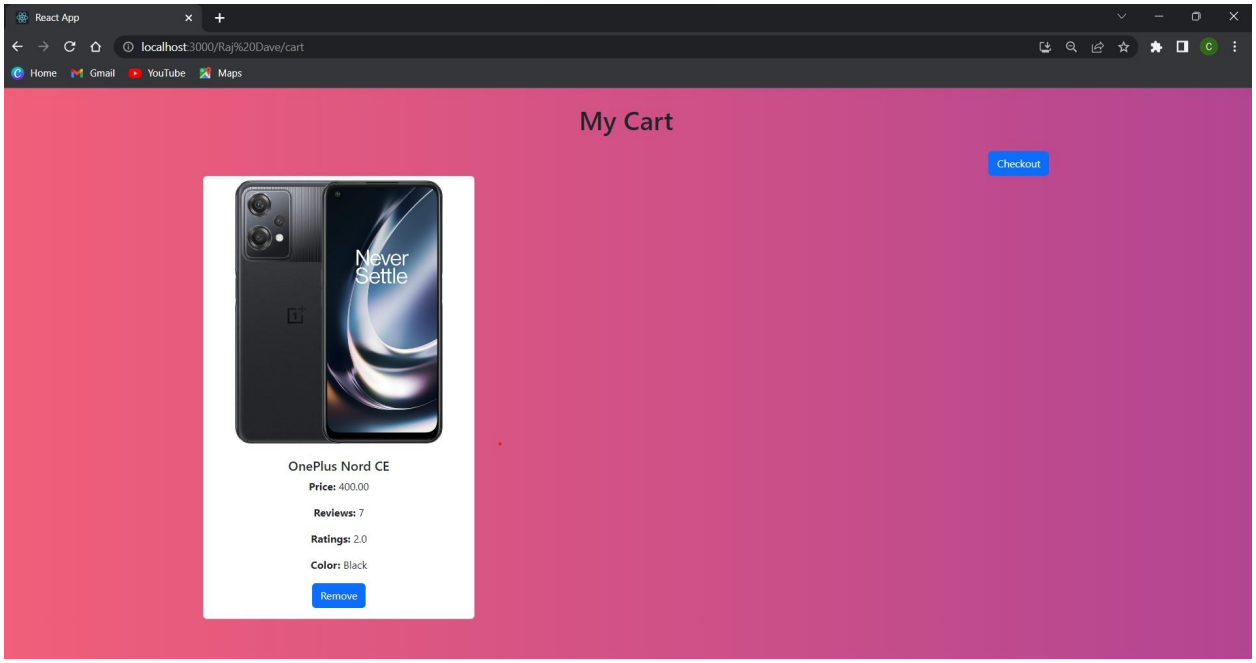
Add to Cart Page

The shopping cart feature of the e-commerce management system allows users to add products they want to purchase and keep track of the total cost before checkout.When a user finds a product they want to purchase, they can add it to their shopping cart by clicking the "Add to Cart" button located on the product page which will display the message of the product being added to the cart. This trigger will send a request to the database to add the product to the user's cart record. The cart page will display all products added to the cart, along with their respective prices and quantities.The ability to add products to the shopping cart provides a convenient way for users to keep track of their intended purchases and calculate the total cost before checkout.The shopping cart trigger also provides valuable data for system administrators, allowing them to analyze which products users add to their carts and adjust the product offerings to better align with user preferences.



The shopping cart feature of the e-commerce management system provides users with a way to keep track of their intended purchases before checkout. In addition to the ability to add products to the cart, users also have the ability to remove products from the cart or proceed to checkout. When a user decides they no longer want a product in their cart, they can remove it by clicking the "Remove" button located on the product's entry in their cart record. This trigger will send a request to the database to remove the product from the user's cart record. Once removed, the user will no longer see the product on their cart page. Once a user has added all desired products to their cart, they can proceed to checkout by clicking the "Checkout" button on the cart page. This trigger will send a request to the database to create a new order record for the user and redirect

them to the payment page.The order placement feature is critical to the success of the e-commerce management system. It ensures that all necessary information about the order is recorded accurately and allows for efficient processing and shipping.



The order placement trigger begins once the user has completed payment and submitted their order. The system will validate the payment and check for any issues.

Frameworks Used

GUI:

The frontend part of the system has a GUI and is made using VB .NET and is connected to the Database using MySQL Connector Nuget Package.

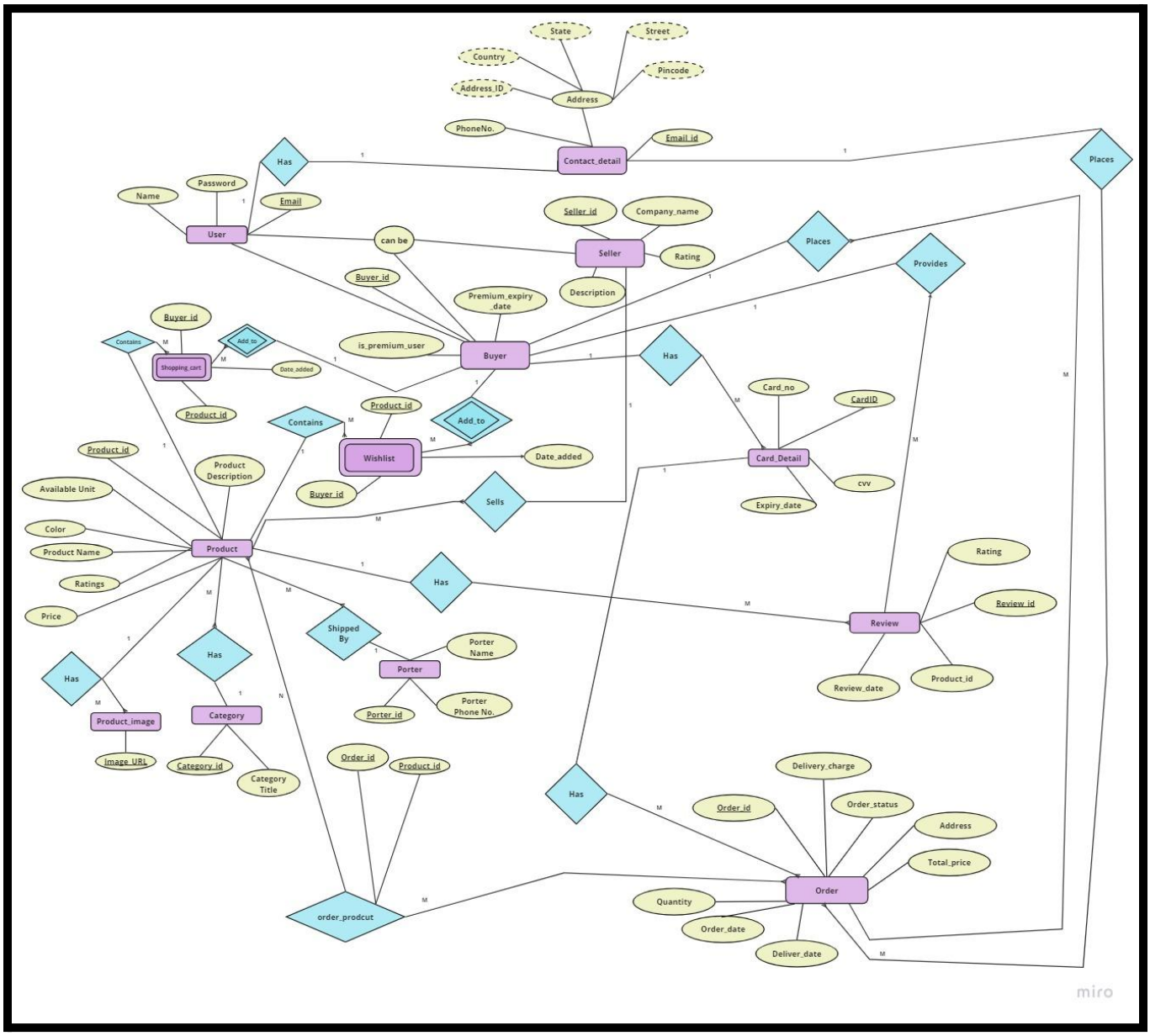
Relational Database Management System:

MySQL Database Server for Windows 11

System Requirements

Windows 10 Creators Update or later (for native support of .NET 5.0) .NET Framework 5.0 MySQL 8.0 or newer

Entity - Relationship Diagram



The relation table as 1:M denotes a one-to-many relationship, M:1 denotes a many-to-one relationship, and M:N denotes a many-to-many relationship for all the given for the tables for ecommerce.

Table 1	Relationship	Table 2
user_detail	1:1	contact_detail
card_detail	M:1 (buyer_id)	buyer
buyer	1:M (orders)	order_detail
category	1:M (products)	product
product	M:1 (seller_id)	seller
product	M:1 (porter_id)	porter
product_image	M:1 (product_id)	product
shopping_cart	1:M (product_shoppingcart)	product_shoppingcart
product_shoppingcart	M:1 (product_id)	product
product_shoppingcart	M:1 (buyer_id)	buyer

wish_list	**M:1 (product_id)**	**product**
wish_list	**M:1 (buyer_id)**	**buyer**
order_detail	**M:1 (card_id)**	**card_detail**
order_detail	**M:1 (user_email)**	**contact_detail**
order_detail	**1: (product_id, order_id)**	**order_product**
order_product	**M:1 (product_id)**	**product**
review	**M:1 (product_id)**	**product**
review	**M:1 (buyer_id)**	**buyer**

Schema Design

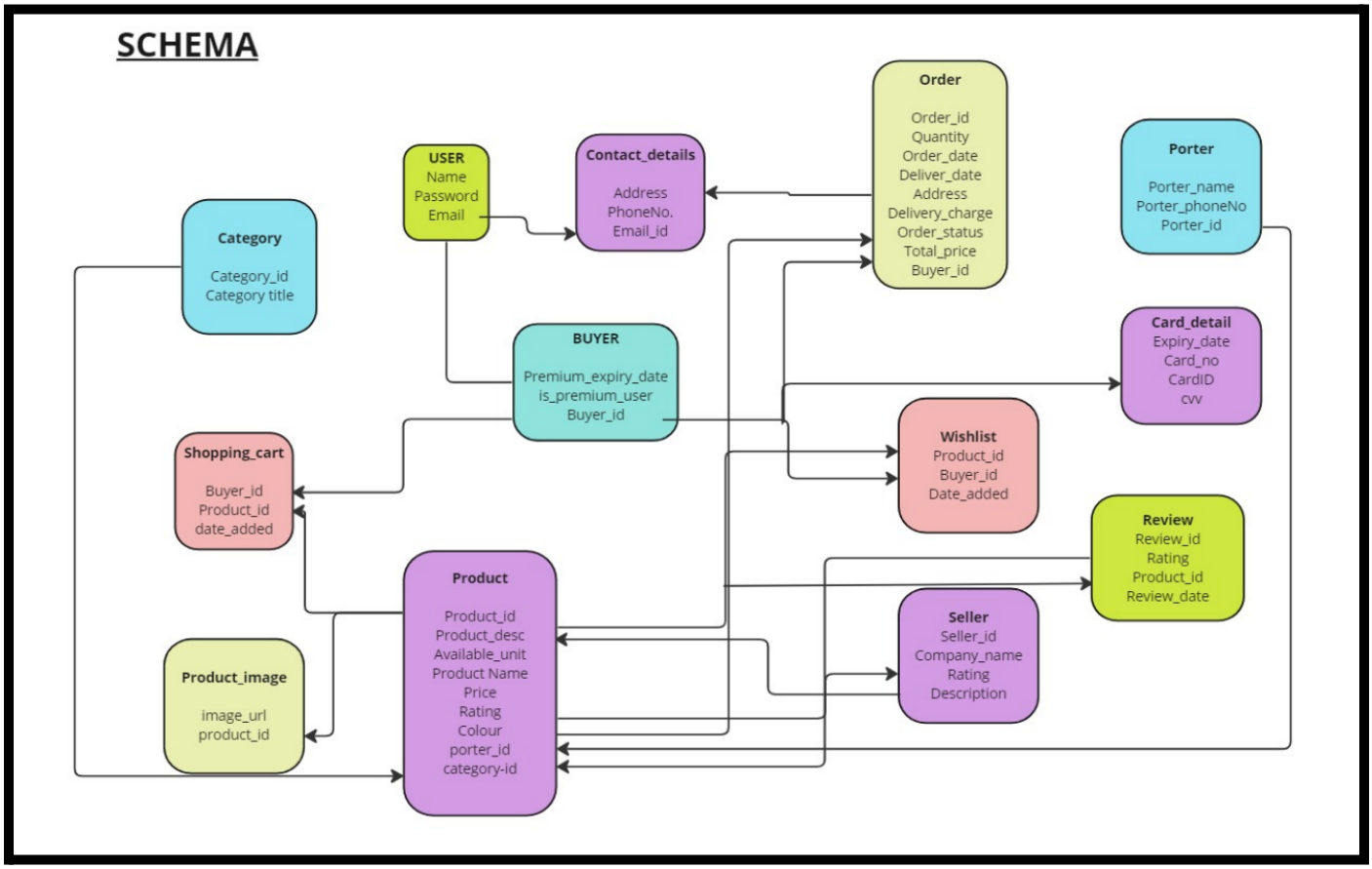


Table Design

Table Name	Attributes	Table Description
user_detail	email (Primary Key), name (Not Null), password (Not Null)	Table for storing user details, including email, name, and password
contact_detail	user_email (Foreign Key references user_detail), address_id, street (Not Null), state (Not Null), country (Not Null), pincode (Not Null), phone (Not Null)	Table for storing user contact details, including address, phone, and email
card_detail	card_id (Primary Key), card_no (Not Null), expiry_date (Not Null), cvv (Not Null), buyer_id (Foreign Key references buyer)	Table for storing card details, including card ID, card number, and CVV
buyer	buyer_id (Primary Key), is_premium_user (Default 0), premium_expiry_date	Table for storing buyer details, including buyer ID, premium status, and expiry date
seller	seller_id (Primary Key), company_name (Not Null), description, rating (Default 2.5), rating_count (Default 0)	Table for storing seller details, including seller ID, company name, and rating

category	category_id (Primary Key), category_title (Not Null)	Table for storing category details, including category ID and title
product	product_id (Primary Key), product_name (Not Null), seller_id (Foreign Key references seller), price (Not Null), rating, category_id (Foreign Key references category), product_description, available_units, color, porter_id (Foreign Key references porter), review_count (Default 0)	Table for storing product details, including product ID, name, price, and rating
product_image	product_id (Foreign Key references product), image_url (Primary Key)	Table for storing product image details, including product ID and image URL
shopping_cart	buyer_id (Foreign Key references buyer), date_added	Table for storing shopping cart details, including buyer ID and date added
product_shoppingcart	product_id (Foreign Key references product), buyer_id (Foreign Key references buyer)	Table for storing product and shopping cart details, including product ID and buyer ID
wish_list	buyer_id (Foreign Key references buyer), date_added, product_id (Foreign Key references product)	Table for storing wish list details, including buyer ID, date added, and product ID

order_detail	order_id (Primary Key), buyer_id (Foreign Key references buyer), card_id (Foreign Key references card_detail), total_price, order_date, delivery_charge (Default 10), delivery_address (Foreign Key references contact_detail), delivery_date, order_status (Not Null), quantity (Not Null), user_email (Foreign Key references user_detail)	Table for storing order details, including order ID, buyer ID, card ID, and order status
order_product	order_id (Foreign Key references order_detail), product_id (Foreign Key references product)	Table for storing order and product details, including order ID and product ID
review	review_id (Primary Key), product_id (Foreign Key references product), buyer_id (Foreign Key references buyer), rating, review_date	Table for storing review details, including review ID, product ID, buyer ID, and rating
porter	porter_id (Primary Key), porter_name (Not Null), porter_phone (Not Null)	Table for storing porter details, including porter ID, name, and phone number

Database design with all constraints

database.js | database.sql | app.js | mysql

D: > Ahmedabad_University > Semester_4 > CSE250 > Project > database.sql

```sql
CREATE TABLE user_detail (email VARCHAR(255) PRIMARY KEY,name VARCHAR(255) NOT NULL, password VARCHAR(30) NOT NULL);

CREATE TABLE contact_detail (user_email VARCHAR(255) PRIMARY KEY, address_id VARCHAR(255), street VARCHAR(255) NOT NULL,state VARCHAR(50) NOT NULL,
        country VARCHAR(50) NOT NULL,pincode INT NOT NULL,phone VARCHAR(20) NOT NULL);

CREATE TABLE card_detail (
    card_id       INTEGER PRIMARY KEY,
    card_nO   INT NOT NULL,
    expiry_date   DATE NOT NULL,
    cvv           INT NOT NULL,
    buyer_id      VARCHAR(255) NOT NULL
);

CREATE TABLE buyer (
    buyer_id            VARCHAR(255) PRIMARY KEY,
    is_premium_user          INT DEFAULT 0,
    premium_expiry_date   DATE
);

CREATE TABLE seller (
    seller_id        VARCHAR(255) PRIMARY KEY,
    company_name     VARCHAR(255) NOT NULL,
    description      VARCHAR(255),
    rating           DECIMAL(3, 1) DEFAULT 2.5,
    rating_count     INTEGER DEFAULT 0
);

CREATE TABLE category (
    category_id     INTEGER PRIMARY KEY,
    category_title   VARCHAR(255) NOT NULL
);
```

D: > Ahmedabad_University > Semester_4 > CSE250 > Project > database.sql

```sql
CREATE TABLE product (
    product_id         INTEGER PRIMARY KEY,
    product_name       VARCHAR(255) NOT NULL,
    seller_id          VARCHAR(255) NOT NULL,
    price              DECIMAL(10, 2) NOT NULL,
    rating             DECIMAL(2, 1),
    category_id        INTEGER,
    product_description VARCHAR(255),
    available_units    INTEGER,
    color              VARCHAR(30),
    porter_id          INTEGER,
    review_count       INTEGER DEFAULT 0
);

CREATE TABLE product_image (
    product_id   INTEGER,
    image_url    VARCHAR(255),
    PRIMARY KEY ( product_id,
                  image_url )
);

CREATE TABLE shopping_cart (
    buyer_id     VARCHAR(255),
    date_added   DATE
);

CREATE TABLE product_shoppingcart (
    product_id   INTEGER,
    buyer_id     VARCHAR(255),
    PRIMARY KEY ( product_id,
                  buyer_id )
);
```

database.js | database.sql | app.js | mysql

D: > Ahmedabad_University > Semester_4 > CSE250 > Project > database.sql

```sql
CREATE TABLE wish_list (
    buyer_id     VARCHAR(255),
    date_added   DATE,
    product_id   INTEGER,
    PRIMARY KEY ( product_id,
                  buyer_id )
);

CREATE TABLE order_detail (
    order_id          INTEGER PRIMARY KEY,
    buyer_id          VARCHAR(255) NOT NULL,
    card_id           INTEGER NOT NULL,
    total_price       DECIMAL(10, 2),
    order_date        DATE,
    delivery_charge   DECIMAL(4, 2) DEFAULT 10,
    delivery_address  VARCHAR(255),
    delivery_date     DATE,
    order_status      CHAR(1) NOT NULL,
    quantity          INTEGER NOT NULL,
    user_email        VARCHAR(255)
);

CREATE TABLE order_product (
    order_id     INTEGER,
    product_id   INTEGER,
    PRIMARY KEY ( order_id,
                  product_id )
);

CREATE TABLE review (
    review_id     INTEGER PRIMARY KEY,
    product_id    INTEGER NOT NULL,
    buyer_id      VARCHAR(255) NOT NULL,
    rating        DECIMAL(2, 1),
    review_date   DATE
);
```

database.js | database.sql | app.js | mysql

D: > Ahmedabad_University > Semester_4 > CSE250 > Project > database.sql

```sql
CREATE TABLE porter (
    porter_id      INTEGER PRIMARY KEY,
    porter_name    VARCHAR(255) NOT NULL,
    porter_phone   DECIMAL(10) NOT NULL
);

ALTER TABLE contact_detail
    ADD CONSTRAINT contact_detail_user_id_fk FOREIGN KEY ( user_email )
        REFERENCES user_detail ( email )
            ON DELETE CASCADE;

ALTER TABLE card_detail
    ADD CONSTRAINT card_info_buyer_id_fk FOREIGN KEY ( buyer_id )
        REFERENCES buyer ( buyer_id )
            ON DELETE CASCADE;

ALTER TABLE product
    ADD CONSTRAINT product_seller_id_fk FOREIGN KEY ( seller_id )
        REFERENCES seller ( seller_id )
            ON DELETE CASCADE;

ALTER TABLE product
    ADD CONSTRAINT product_category_id_fk FOREIGN KEY ( category_id )
        REFERENCES category ( category_id )
            ON DELETE CASCADE;

ALTER TABLE product
    ADD CONSTRAINT product_carrier_id_fk FOREIGN KEY ( porter_id )
        REFERENCES porter ( porter_id )
            ON DELETE CASCADE;

ALTER TABLE product_image
    ADD CONSTRAINT product_image_product_id_fk FOREIGN KEY ( product_id )
        REFERENCES product ( product_id )
            ON DELETE CASCADE;
```

database.js | database.sql | app.js | mysql

D: > Ahmedabad_University > Semester_4 > CSE250 > Project > database.sql

```
139  ALTER TABLE shopping_cart
140      ADD CONSTRAINT shopping_cart_buyer_id_fk FOREIGN KEY ( buyer_id )
141          REFERENCES buyer ( buyer_id )
142              ON DELETE CASCADE;
143
144  ALTER TABLE product_shoppingcart
145      ADD CONSTRAINT product_sc_buyer_id_fk FOREIGN KEY ( buyer_id )
146          REFERENCES buyer ( buyer_id )
147              ON DELETE CASCADE;
148
149  ALTER TABLE product_shoppingcart
150      ADD CONSTRAINT product_sc_product_id_fk FOREIGN KEY ( product_id )
151          REFERENCES product ( product_id )
152              ON DELETE CASCADE;
153
154  ALTER TABLE wish_list
155      ADD CONSTRAINT wishlist_buyer_id_fk FOREIGN KEY ( buyer_id )
156          REFERENCES buyer ( buyer_id )
157              ON DELETE CASCADE;
158
159  ALTER TABLE wish_list
160      ADD CONSTRAINT wishlist_product_id_fk FOREIGN KEY ( product_id )
161          REFERENCES product ( product_id )
162              ON DELETE CASCADE;
163
164  ALTER TABLE order_detail
165      ADD CONSTRAINT order_buyer_id_fk FOREIGN KEY ( buyer_id )
166          REFERENCES buyer ( buyer_id )
167              ON DELETE CASCADE;
168
169  ALTER TABLE order_detail
170      ADD CONSTRAINT order_card_id_fk FOREIGN KEY ( card_id )
171          REFERENCES card_detail ( card_id )
172              ON DELETE CASCADE;
```

database.js | database.sql | app.js | mysql

D: > Ahmedabad_University > Semester_4 > CSE250 > Project > database.sql

```
174  ALTER TABLE order_detail
175      ADD CONSTRAINT order_delivery_address_id_fk FOREIGN KEY ( user_email)
176          REFERENCES contact_detail (user_email)
177              ON DELETE CASCADE;
178
179  ALTER TABLE order_product
180      ADD CONSTRAINT order_product_order_id_fk FOREIGN KEY ( order_id )
181          REFERENCES order_detail ( order_id )
182              ON DELETE CASCADE;
183
184  ALTER TABLE order_product
185      ADD CONSTRAINT order_product_product_id_fk FOREIGN KEY ( product_id )
186          REFERENCES product ( product_id )
187              ON DELETE CASCADE;
188
189  ALTER TABLE review
190      ADD CONSTRAINT review_product_id_fk FOREIGN KEY ( product_id )
191          REFERENCES product ( product_id )
192              ON DELETE CASCADE;
193
194  ALTER TABLE review
195      ADD CONSTRAINT review_buyer_id_fk FOREIGN KEY ( buyer_id )
196          REFERENCES buyer ( buyer_id )
197              ON DELETE CASCADE;
198
199  -----------------------------------------------------------------------------------------
200
```

Stored Procedures/ Functions

1. **Login page:**

```
mysql> DELIMITER //
mysql>
mysql> CREATE OR REPLACE PROCEDURE login_procedure()
    -> BEGIN
    ->     DECLARE message VARCHAR(255);
    ->     IF NEW.email = 'user_email' AND NEW.password = 'user_password' THEN
    ->         SET message = 'User logged in successfully';
    ->         INSERT INTO user_login (user_id, message) VALUES (NEW.id, message);
    ->     END IF;
    -> END//
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
```

2. **Adding Buyer Details for the user:**

```
mysql> DELIMITER //
mysql> CREATE OR REPLACE PROCEDURE register_buyer (
    ->     IN email VARCHAR(255),
    ->     IN name VARCHAR(255),
    ->     IN password VARCHAR(255)
    -> )
    -> BEGIN
    ->     INSERT INTO user_detail (email, name, password) VALUES (email, name, password);
    ->
    ->     INSERT INTO buyer (buyer_id, is_premium_user, premium_expiry_date) VALUES (email, 0, NULL);
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> CALL register_buyer('charmidesai@gmail.com', 'Charmi', '123abc');
Query OK, 2 rows affected (0.01 sec)

mysql> CALL register_buyer('riyapatel@gmail.com', 'Riya', '123abc');
Query OK, 2 rows affected (0.00 sec)

mysql> CALL register_buyer('khwahishpatel@gmail.com', 'Khwahish', '123abc');
Query OK, 2 rows affected (0.00 sec)

mysql> select * from user_detail;
+------------------------+----------+----------+
| email                  | name     | password |
+------------------------+----------+----------+
| charmidesai@gmail.com  | Charmi   | 123abc   |
| khwahishpatel@gmail.com | Khwahish | 123abc   |
| riyapatel@gmail.com    | Riya     | 123abc   |
+------------------------+----------+----------+
3 rows in set (0.00 sec)
```

3. Adding seller details:

```
mysql> DELIMITER //
mysql>
mysql> CREATE OR REPLACE PROCEDURE register_seller (
    ->     IN email        VARCHAR(255),
    ->     IN name         VARCHAR(255),
    ->     IN password     VARCHAR(255),
    ->     IN company_name VARCHAR(255),
    ->     IN description  VARCHAR(255)
    -> )
    -> BEGIN
    ->     INSERT INTO user_detail (email, name, password)
    ->     VALUES (email, name, password);
    ->
    ->     INSERT INTO seller (seller_id, company_name, description, rating, rating_count)
    ->     VALUES (email, company_name, description, 2.5, 1);
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL register_seller('kushagradar@gmail.com', 'kushagra', '123abc', 'kushagra Co and Co', 'company of shoes');
Query OK, 2 rows affected (0.01 sec)

mysql> CALL register_seller('ruchisingh@gmail.com', 'ruchi', '123abc', 'ruchi Co and Co', 'company of metals');
Query OK, 2 rows affected (0.00 sec)

mysql> CALL register_seller('anantprakash@gmail.com', 'anant', '123abc', 'anant Co and Co', 'company of iphones');
Query OK, 2 rows affected (0.00 sec)

mysql> select * from seller;
+----------------------+--------------------+--------------------+--------+--------------+
| seller_id            | company_name       | description        | rating | rating_count |
+----------------------+--------------------+--------------------+--------+--------------+
| anantprakash@gmail.com | anant Co and Co    | company of iphones |    2.5 |            1 |
| kushagradar@gmail.com  | kushagra Co and Co | company of shoes   |    2.5 |            1 |
| ruchisingh@gmail.com   | ruchi Co and Co    | company of metals  |    2.5 |            1 |
+----------------------+--------------------+--------------------+--------+--------------+
3 rows in set (0.00 sec)
```

4. Adding contact details:

```
mysql> CREATE PROCEDURE add_contact_details (
    ->     IN user_email VARCHAR(255),
    ->     IN address_id INT,
    ->     IN street VARCHAR(255),
    ->     IN state VARCHAR(50),
    ->     IN country VARCHAR(50),
    ->     IN pincode INT,
    ->     IN phone VARCHAR(20)
    -> )
    -> BEGIN
    ->     INSERT INTO contact_detail (user_email, address_id, street, state, country, pincode, phone)
    ->     VALUES (user_email, address_id, street, state, country, pincode, phone);
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL add_contact_details('charmidesai@gmail.com', 1, 'Bhattha', 'Gujarat', 'India', 380007, 7777777777);
Query OK, 1 row affected (0.01 sec)

mysql> CALL add_contact_details('khwahishpatel@gmail.com', 2, 'Satellite', 'Gujarat', 'India', 380015, 8888888888);
Query OK, 1 row affected (0.00 sec)

mysql> CALL add_contact_details('riyapatel@gmail.com', 3, 'Lalbaug', 'Gujarat', 'India', 380026, 6969696969);
Query OK, 1 row affected (0.00 sec)

mysql> select * from contact_detail;
+-------------------------+------------+-----------+---------+---------+---------+------------+
| user_email              | address_id | street    | state   | country | pincode | phone      |
+-------------------------+------------+-----------+---------+---------+---------+------------+
| charmidesai@gmail.com   | 1          | Bhattha   | Gujarat | India   |  380007 | 7777777777 |
| khwahishpatel@gmail.com | 2          | Satellite | Gujarat | India   |  380015 | 8888888888 |
| riyapatel@gmail.com     | 3          | Lalbaug   | Gujarat | India   |  380026 | 6969696969 |
+-------------------------+------------+-----------+---------+---------+---------+------------+
3 rows in set (0.00 sec)
```

5. Adding Card Details of the user:

```
mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE add_card_info (
    ->     IN buyer_id VARCHAR(255),
    ->     IN card_id INT,
    ->     IN card_nO DECIMAL(30),
    ->     IN expiry_date DATE,
    ->     IN cvv INT
    -> )
    -> BEGIN
    ->     INSERT INTO card_detail (card_id, card_nO, expiry_date, cvv, buyer_id)
    ->     VALUES (card_id, card_nO, expiry_date, cvv, buyer_id);
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL add_card_info('khwahishpatel@gmail.com', 1, 1234123412341234, STR_TO_DATE('2025-06-23', '%Y-%m-%d'), 696);
Query OK, 1 row affected, 1 warning (0.01 sec)

mysql> CALL add_card_info('riyapatel@gmail.com', 2, 4567456745674567, STR_TO_DATE('2024-04-10', '%Y-%m-%d'), 123);
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> CALL add_card_info('charmidesai@gmail.com', 3, 6987698769876987, STR_TO_DATE('2023-12-09', '%Y-%m-%d'), 444);
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> select * from card_detail;
+---------+------------+-------------+-----+-------------------------+
| card_id | card_nO    | expiry_date | cvv | buyer_id                |
+---------+------------+-------------+-----+-------------------------+
|       1 | 2147483647 | 2025-06-23  | 696 | khwahishpatel@gmail.com |
|       2 | 2147483647 | 2024-04-10  | 123 | riyapatel@gmail.com     |
|       3 | 2147483647 | 2023-12-09  | 444 | charmidesai@gmail.com   |
+---------+------------+-------------+-----+-------------------------+
3 rows in set (0.00 sec)

mysql>
```

6. Adding products by the seller to buy:

```
mysql> CREATE OR REPLACE PROCEDURE add_product (
    ->     IN product_id INT,
    ->     IN product_name VARCHAR(255),
    ->     IN seller_id VARCHAR(255),
    ->     IN price DECIMAL(10,2),
    ->     IN rating INT,
    ->     IN category_id INT,
    ->     IN product_description TEXT,
    ->     IN available_units INT,
    ->     IN color VARCHAR(255),
    ->     IN porter_id INT,
    ->     IN review_count INT,
    ->     IN image_url VARCHAR(255)
    -> )
    -> BEGIN
    ->     INSERT INTO product (product_id, product_name, seller_id, price, rating, category_id, product_description, available_units, color, porter_id, review_count)
    ->     VALUES (product_id, product_name, seller_id, price, rating, category_id, product_description, available_units, color, porter_id, review_count);
    ->
    ->     INSERT INTO product_image (product_id, image_url)
    ->     VALUES (product_id, image_url);
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL add_product(1, 'OnePlus Nord CE', 'kushagradar@gmail.com', 400, 2, 3,
    ->                 'Best Phone', 3, 'Black', 2, 5,
    ->                 'https://m.media-amazon.com/images/W/IMAGERENDERING_521856-T1/images/I/71V--WZVUIL._SL1500_.jpg');
Query OK, 2 rows affected (0.01 sec)

mysql>
mysql> CALL add_product(2, 'Lipstick', 'ruchisingh@gmail.com', 15, 3, 1,
    ->                 'Best matt Lipstick', 4, 'Red', 3, 3,
    ->                 'https://images-static.nykaa.com/media/catalog/product/f/6/f6a87bc8904207500689_1.jpg');
Query OK, 2 rows affected (0.01 sec)
```

```
mysql> select * from product;
+------------+-----------------+----------------------+--------+--------+-------------+---------------------+-----------------+-----------+-----------+--------------+
| product_id | product_name    | seller_id            | price  | rating | category_id | product_description | available_units | color     | porter_id | review_count |
+------------+-----------------+----------------------+--------+--------+-------------+---------------------+-----------------+-----------+-----------+--------------+
|          1 | OnePlus Nord CE | kushagradar@gmail.com | 400.00 |    2.0 |           3 | Best Phone          |               3 | Black     |         2 |            5 |
|          2 | Lipstick        | ruchisingh@gmail.com  |  15.00 |    3.0 |           1 | Best matt Lipstick  |               4 | Red       |         3 |            3 |
|          3 | Nike Jordan Shoes | anantprakash@gmail.com |  50.00 |    5.0 |           1 | Best shoes          |               1 | red       |         1 |            1 |
|          4 | I phone 13      | anantprakash@gmail.com | 500.00 |    4.0 |           3 | Better than android |               3 | Rose gold |         2 |            2 |
|          5 | Sanitary Napkins | ruchisingh@gmail.com  |  20.00 |    4.0 |           2 | Best sanitary napkins |              12 | Blue      |         1 |            1 |
+------------+-----------------+----------------------+--------+--------+-------------+---------------------+-----------------+-----------+-----------+--------------+
5 rows in set (0.00 sec)
```

7. **Adding to Shopping Cart:**

```
mysql> CREATE PROCEDURE add_to_shopping_cart (
    ->     IN buyer_id VARCHAR(255),
    ->     IN product_id INT
    -> )
    -> BEGIN
    ->     INSERT INTO shopping_cart (buyer_id, date_added)
    ->     VALUES (buyer_id, DATE_FORMAT(CURDATE(), '%Y-%m-%d'));
    ->
    ->     INSERT INTO product_shoppingcart (product_id, buyer_id)
    ->     VALUES (product_id, buyer_id);
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL add_to_shopping_cart('riyapatel@gmail.com', 1);
Query OK, 2 rows affected (0.01 sec)

mysql> CALL add_to_shopping_cart('charmidesai@gmail.com', 3);
Query OK, 2 rows affected (0.00 sec)

mysql> CALL add_to_shopping_cart('khwahishpatel@gmail.com', 2);
Query OK, 2 rows affected (0.00 sec)

mysql> CALL add_to_shopping_cart('khwahishpatel@gmail.com', 4);
Query OK, 2 rows affected (0.00 sec)

mysql> select * from shopping_cart;
+------------------------+------------+
| buyer_id               | date_added |
+------------------------+------------+
| riyapatel@gmail.com    | 2023-04-20 |
| charmidesai@gmail.com  | 2023-04-20 |
| khwahishpatel@gmail.com | 2023-04-20 |
| khwahishpatel@gmail.com | 2023-04-20 |
+------------------------+------------+
4 rows in set (0.00 sec)
```

8. Adding to Wishlist:

```
mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE add_to_wish_list (
    ->     IN buyer_id VARCHAR(255),
    ->     IN product_id INT
    -> )
    -> BEGIN
    ->     INSERT INTO wish_list (buyer_id, date_added, product_id)
    ->     VALUES (buyer_id, DATE_FORMAT(CURDATE(), '%Y-%m-%d'), product_id);
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL add_to_wish_list('charmidesai@gmail.com', 1);
Query OK, 1 row affected (0.00 sec)

mysql> CALL add_to_wish_list('charmidesai@gmail.com', 4);
Query OK, 1 row affected (0.00 sec)

mysql> CALL add_to_wish_list('riyapatel@gmail.com', 3);
Query OK, 1 row affected (0.00 sec)

mysql> CALL add_to_wish_list('khwahishpatel@gmail.com', 5);
Query OK, 1 row affected (0.00 sec)

mysql> select * from wish_list;
+-------------------------+------------+------------+
| buyer_id                | date_added | product_id |
+-------------------------+------------+------------+
| charmidesai@gmail.com   | 2023-04-20 |          1 |
| riyapatel@gmail.com     | 2023-04-20 |          3 |
| charmidesai@gmail.com   | 2023-04-20 |          4 |
| khwahishpatel@gmail.com | 2023-04-20 |          5 |
+-------------------------+------------+------------+
4 rows in set (0.00 sec)
```

9. Placing an order:

```
mysql> CREATE OR REPLACE PROCEDURE insert_order_detail (IN order_id INT, IN buyer_id VARCHAR(255), IN card_id INT, IN total_price DECIMAL(10,2), IN order_date DATE, IN delivery_charge DECIMAL(4,2),
    -> IN delivery_address VARCHAR(255), IN delivery_date DATE, IN order_status CHAR(1), IN quantity INT, IN user_email VARCHAR(255))
    -> BEGIN
    ->     INSERT INTO order_detail (order_id, buyer_id, card_id, total_price, order_date, delivery_charge, delivery_address, delivery_date, order_status, quantity, user_email)
    ->     VALUES (order_id, buyer_id, card_id, total_price, order_date, COALESCE(delivery_charge, 10), COALESCE(delivery_address, NULL), COALESCE(delivery_date, NULL),
    ->     COALESCE(order_status, 'C'), quantity, user_email);
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql> DELIMITER ;
mysql> CALL insert_order_detail(1, 'riyapatel@gmail.com', 2, 50.00, DATE(SYSDATE()), 10.00, 'Lalbaug', DATE(DATE_ADD(SYSDATE(), INTERVAL 1 MONTH)), 'C',
    ->                     3, 'riyapatel@gmail.com');
Query OK, 1 row affected (0.01 sec)

mysql> CALL insert_order_detail(2, 'charmidesai@gmail.com', 3, 100.00, DATE(SYSDATE()), 10.00, 'Bhattha', DATE(DATE_ADD(SYSDATE(), INTERVAL 1 MONTH)), 'C',
    ->                     2, 'charmidesai@gmail.com');
Query OK, 1 row affected (0.00 sec)

mysql> select * from order_detail;
+----------+-------------------------+---------+-------------+------------+-----------------+------------------+---------------+--------------+----------+-------------------------+
| order_id | buyer_id                | card_id | total_price | order_date | delivery_charge | delivery_address | delivery_date | order_status | quantity | user_email              |
+----------+-------------------------+---------+-------------+------------+-----------------+------------------+---------------+--------------+----------+-------------------------+
|        1 | riyapatel@gmail.com     |       2 |       50.00 | 2023-04-20 |           10.00 | Lalbaug          | 2023-05-20    | C            |        3 | riyapatel@gmail.com     |
|        2 | charmidesai@gmail.com   |       3 |      100.00 | 2023-04-20 |           10.00 | Bhattha          | 2023-05-20    | C            |        2 | charmidesai@gmail.com   |
+----------+-------------------------+---------+-------------+------------+-----------------+------------------+---------------+--------------+----------+-------------------------+
2 rows in set (0.00 sec)
```

```
mysql> CREATE OR REPLACE PROCEDURE place_order(IN order_id INT, IN buyer_id_var VARCHAR(255))
    -> BEGIN
    ->     DECLARE card_id_var INT;
    ->     DECLARE address_id_var INT;
    ->     DECLARE total_price_var DECIMAL(10, 2) DEFAULT 0;
    ->     DECLARE total_qty_var DECIMAL(10, 2) DEFAULT 0;
    ->     DECLARE available_units_var DECIMAL(10, 2);
    ->     DECLARE delivery_charge_var DECIMAL(10, 2) DEFAULT 10;
    ->     DECLARE is_premium_user_var INT DEFAULT 0;
    ->     DECLARE product_id_var INT;
    ->
    ->     DECLARE products_cur CURSOR FOR
    ->         SELECT product_id
    ->         FROM product_shoppingcart
    ->         WHERE buyer_id = buyer_id_var;
    ->
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET @done = TRUE;
    ->
    ->     OPEN products_cur;
    ->     SET @done = FALSE;
    ->
    ->     read_loop: LOOP
    ->         FETCH products_cur INTO product_id_var;
    ->         IF @done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         SELECT
    ->             price,
    ->             available_units
    ->         INTO
    ->             total_price_var,
    ->             available_units_var
    ->         FROM
    ->             product
    ->         WHERE
    ->             product_id = product_id_var;
    ->
    ->         IF available_units_var > 0 THEN
    ->             SET total_price_var := total_price_var + 0;
    ->             SET total_qty_var := total_qty_var + 1;
    ->             INSERT INTO order_product VALUES (
    ->                 order_id,
    ->                 product_id_var
```

```
    ->     IF is_premium_user_var = 1 THEN
    ->         SET delivery_charge_var := 0;
    ->     END IF;
    ->
    ->     SELECT
    ->         card_id
    ->     INTO card_id_var
    ->     FROM
    ->         card_detail
    ->     WHERE
    ->         buyer_id = buyer_id_var;
    ->
    ->     SELECT
    ->         address_id
    ->     INTO address_id_var
    ->     FROM
    ->         contact_detail
    ->     WHERE
    ->         user_email = buyer_id_var;
    ->
    ->     SET total_price_var := total_price_var + delivery_charge_var + 10;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> CALL place_order(1, 'riyapatel@gmail.com');
Query OK, 5 rows affected (0.00 sec)

mysql> CALL place_order(2, 'charmidesai@gmail.com');
Query OK, 5 rows affected (0.01 sec)

mysql> select * from product_shoppingcart;
+------------+-------------------------+
| product_id | buyer_id                |
+------------+-------------------------+
|          1 | riyapatel@gmail.com     |
|          2 | khwahishpatel@gmail.com |
|          3 | charmidesai@gmail.com   |
|          4 | khwahishpatel@gmail.com |
+------------+-------------------------+
4 rows in set (0.00 sec)
```

10. Removing from Wishlist:

```
mysql> DELIMITER //
mysql>
mysql> CREATE PROCEDURE remove_wishlist (
    ->     IN buyer_id VARCHAR(255),
    ->     IN product_id INT
    -> )
    -> BEGIN
    ->     DELETE FROM wish_list
    ->     WHERE buyer_id = buyer_id AND product_id = product_id
    ->     ORDER BY date_added ASC
    ->     LIMIT 1;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> call remove_wishlist('charmidesai@gmail.com',1);
Query OK, 1 row affected (0.00 sec)

mysql> select * from wishlist;
ERROR 1146 (42S02): Table 'commerce.wishlist' doesn't exist
mysql> select * from wish_list;
+-------------------------+------------+------------+
| buyer_id                | date_added | product_id |
+-------------------------+------------+------------+
| riyapatel@gmail.com     | 2023-04-20 |          3 |
| charmidesai@gmail.com   | 2023-04-20 |          4 |
| khwahishpatel@gmail.com | 2023-04-20 |          5 |
+-------------------------+------------+------------+
3 rows in set (0.00 sec)
```

11. Removing from Cart:

```
mysql> CREATE PROCEDURE remove_shopping_cart (
    ->     IN buyer_id VARCHAR(255),
    ->     IN product_id INT
    -> )
    -> BEGIN
    ->     DELETE FROM product_shoppingcart
    ->     WHERE buyer_id = buyer_id AND product_id = product_id
    ->     LIMIT 1;
    ->
    ->     DELETE FROM shopping_cart
    ->     WHERE buyer_id = buyer_id
    ->     AND NOT EXISTS (
    ->         SELECT * FROM product_shoppingcart
    ->         WHERE product_shoppingcart.buyer_id = shopping_cart.buyer_id
    ->     );
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> call remove_shopping_cart('riyapatel@gmail.com',1);
Query OK, 2 rows affected (0.01 sec)

mysql> select * from shopping_cart;
+------------------------+------------+
| buyer_id               | date_added |
+------------------------+------------+
| charmidesai@gmail.com  | 2023-04-20 |
| khwahishpatel@gmail.com | 2023-04-20 |
| khwahishpatel@gmail.com | 2023-04-20 |
+------------------------+------------+
3 rows in set (0.00 sec)
```

Triggers

1. **Trigger to auto increment category_id:**

```
mysql> DELIMITER //
mysql> CREATE OR REPLACE TRIGGER tr_category_auto_increment
    -> BEFORE INSERT ON category FOR EACH ROW
    -> BEGIN
    ->     SET NEW.category_id = (SELECT COALESCE(MAX(category_id), 0) + 1 FROM category);
    -> END;
    -> //
Query OK, 0 rows affected (0.01 sec)
```

2. **Trigger to auto increment porter_id:**

```
mysql> DELIMITER ;
mysql> DELIMITER //
mysql> CREATE OR REPLACE TRIGGER porter_auto_increment
    -> BEFORE INSERT ON porter FOR EACH ROW
    -> BEGIN
    ->     DECLARE last_id INT;
    ->     SET last_id = (SELECT MAX(porter_id) FROM porter);
    ->     SET NEW.porter_id = COALESCE(last_id, 0) + 1;
    -> END;
    -> //
Query OK, 0 rows affected (0.02 sec)
```

3. To update the product rating:

```
mysql> CREATE TRIGGER update_product_rating
    -> AFTER INSERT ON review FOR EACH ROW
    -> BEGIN
    ->     DECLARE new_rating DECIMAL(2, 1);
    ->     DECLARE review_count_old INT;
    ->
    ->     SELECT review_count INTO review_count_old
    ->     FROM product
    ->     WHERE product_id = NEW.product_id;
    ->
    ->     SET new_rating = NEW.rating;
    ->     UPDATE product
    ->     SET rating = ((rating * review_count_old) + new_rating) / (review_count_old + 1),
    ->         review_count = review_count_old + 1
    ->     WHERE product_id = NEW.product_id;
    -> END //
Query OK, 0 rows affected (0.02 sec)

mysql>
mysql> DELIMITER ;
mysql> select * from review;
Empty set (0.00 sec)

mysql> insert into review values(1,1,'charmidesai@gmail.com', 4, STR_TO_DATE('2023-04-20', '%Y-%m-%d'));
Query OK, 1 row affected (0.01 sec)

mysql> select * from review;
+-----------+------------+-----------------------+--------+-------------+
| review_id | product_id | buyer_id              | rating | review_date |
+-----------+------------+-----------------------+--------+-------------+
|         1 |          1 | charmidesai@gmail.com |    4.0 | 2023-04-20  |
+-----------+------------+-----------------------+--------+-------------+
1 row in set (0.00 sec)
```

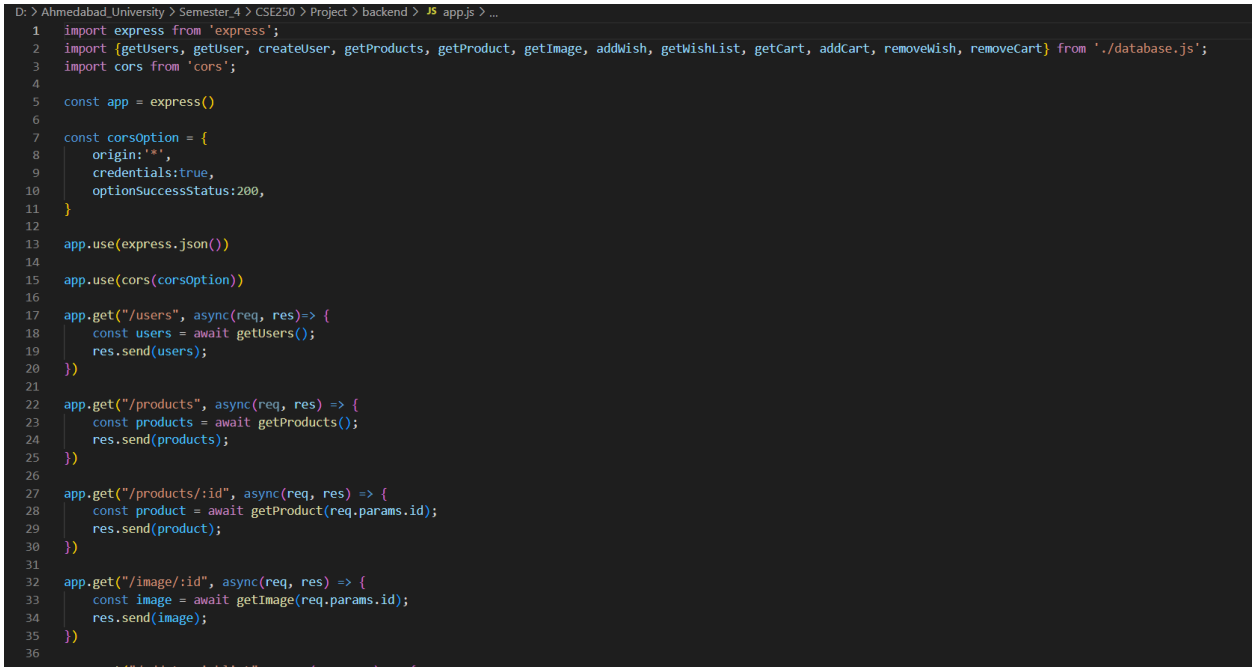
4. To update seller rating:

```
mysql> CREATE TRIGGER update_seller_rating
    -> AFTER INSERT ON review FOR EACH ROW
    -> BEGIN
    ->     DECLARE new_rating DECIMAL(2, 1);
    ->     DECLARE seller_id_to_update VARCHAR(255);
    ->
    ->     SET new_rating = NEW.rating;
    ->     SELECT seller_id INTO seller_id_to_update
    ->     FROM product
    ->     WHERE product_id = NEW.product_id;
    ->
    ->     UPDATE seller
    ->     SET rating = ((rating * rating_count) + new_rating) / (rating_count + 1),
    ->         rating_count = rating_count + 1
    ->     WHERE seller_id = seller_id_to_update;
    -> END //
Query OK, 0 rows affected (0.01 sec)

mysql>
mysql> DELIMITER ;
mysql> select * from seller;
+-----------------------+-------------------+--------------------+--------+--------------+
| seller_id             | company_name      | description        | rating | rating_count |
+-----------------------+-------------------+--------------------+--------+--------------+
| anantprakash@gmail.com | anant Co and Co   | company of iphones |    2.5 |            1 |
| kushagradar@gmail.com  | kushagra Co and Co | company of shoes   |    2.5 |            1 |
| ruchisingh@gmail.com   | ruchi Co and Co   | company of metals  |    2.5 |            1 |
+-----------------------+-------------------+--------------------+--------+--------------+
3 rows in set (0.00 sec)

mysql> insert into review values(2,2,'charmidesai@gmail.com', 3, STR_TO_DATE('2023-04-20', '%Y-%m-%d'));
Query OK, 1 row affected (0.00 sec)

mysql> select * from seller;
+-----------------------+-------------------+--------------------+--------+--------------+
| seller_id             | company_name      | description        | rating | rating_count |
+-----------------------+-------------------+--------------------+--------+--------------+
| anantprakash@gmail.com | anant Co and Co   | company of iphones |    2.5 |            1 |
| kushagradar@gmail.com  | kushagra Co and Co | company of shoes   |    2.5 |            1 |
| ruchisingh@gmail.com   | ruchi Co and Co   | company of metals  |    2.8 |            2 |
+-----------------------+-------------------+--------------------+--------+--------------+
3 rows in set (0.00 sec)
```

5. **To update the available units for the products:**

```
mysql> DELIMITER //
mysql> CREATE TRIGGER update_available_units AFTER INSERT ON order_detail FOR EACH ROW
    -> BEGIN
    ->     DECLARE product_id_var INT;
    ->     DECLARE available_units_var INT;
    ->     DECLARE done INT DEFAULT 0;
    ->     DECLARE products_cur CURSOR FOR
    ->         SELECT product_id FROM order_product WHERE order_id = NEW.order_id;
    ->     DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    ->
    ->     OPEN products_cur;
    ->
    ->     read_loop: LOOP
    ->         FETCH products_cur INTO product_id_var;
    ->         IF done THEN
    ->             LEAVE read_loop;
    ->         END IF;
    ->
    ->         SELECT available_units INTO available_units_var FROM product WHERE product_id = product_id_var;
    ->
    ->         IF available_units_var >= 2 THEN
    ->             UPDATE product SET available_units = available_units - 1 WHERE product_id = product_id_var;
    ->         ELSEIF available_units_var = 1 THEN
    ->             UPDATE product SET available_units = available_units - 1 WHERE product_id = product_id_var;
    ->         END IF;
    ->     END LOOP;
    ->
    ->     CLOSE products_cur;
    -> END;
    -> //
Query OK, 0 rows affected (0.01 sec)
```

Connecting procedures with the frontend



D: > Ahmedabad_University > Semester_4 > CSE250 > Project > backend > JS app.js > ...

```
import express from 'express';
import {getUsers, getUser, createUser, getProducts, getProduct, getImage, addWish, getWishList, getCart, addCart, removeWish, removeCart} from './database.js';
import cors from 'cors';

const app = express()

const corsOption = {
    origin:'*',
    credentials:true,
    optionSuccessStatus:200,
}

app.use(express.json())

app.use(cors(corsOption))

app.get("/users", async(req, res)=> {
    const users = await getUsers();
    res.send(users);
})

app.get("/products", async(req, res) => {
    const products = await getProducts();
    res.send(products);
})

app.get("/products/:id", async(req, res) => {
    const product = await getProduct(req.params.id);
    res.send(product);
})

app.get("/image/:id", async(req, res) => {
    const image = await getImage(req.params.id);
    res.send(image);
})

```

D: > Ahmedabad_University > Semester_4 > CSE250 > Project > backend > JS app.js > ...

```
app.post("/add_to_wishlist", async(req, res) => {
    const { buyer_id, prod_id } = req.body;
    const out = await addWish(buyer_id, prod_id);
    res.send(out);
})

app.post("/remove_wishlist", async(req, res) => {
    const { buyer_id, prod_id } = req.body;
    const out = await removeWish(buyer_id, prod_id);
    res.send(out);
})

app.post("/remove_cart", async(req, res) => {
    const { buyer_id, prod_id } = req.body;
    const out = await removeCart(buyer_id, prod_id);
    res.send(out);
})

app.post("/cart", async(req, res) => {
    const { buyer_id, prod_id } = req.body;
    const out = await addCart(buyer_id, prod_id);
    res.send(out);
})

app.get("/users/:id", async(req, res) => {
    const user = await getUser(req.params.id);
    res.send(user);
})

app.get("/wishlist", async(req, res) => {
    const wish = await getWishList();
    res.send(wish);
})

app.get("/cart", async(req, res) => {
    const cart = await getCart();
    res.send(cart);
})

app.post("/users", async(req, res) => {
    const { email, name, password } = req.body;
    const user = await createUser(email, name, password);
    res.status(201).send(user);
})

app.listen(8081, () => {
    console.log('Server is running on port 8081');
})
```

D: > Ahmedabad_University > Semester_4 > CSE250 > Project > backend > JS database.js > getImage

```
import mysql from 'mysql2';

const ecommerce = mysql.createPool({
    host: '127.0.0.1',
    user: 'joe root',
    password: '123',
    database: 'ecommerce',
}).promise();

export async function getUsers() {
    const [result] = await ecommerce.query("SELECT * FROM user_detail");
    return result;
}

export async function getProducts() {
    const [result] = await ecommerce.query("SELECT * FROM product");
    return result;
}

export async function getProduct(id) {
    const [result] = await ecommerce.query(`
        SELECT *
        FROM product
        WHERE product_id = ?
        `, [id])
    return result[0];
}

export async function getImage(id) {
    const [result] = await ecommerce.query(`
        SELECT *
        FROM product_image
        WHERE product_id = ?
        `, [id])
    return result[0];
}
```

D: > Ahmedabad_University > Semester_4 > CSE250 > Project > backend > JS database.js > getImage

```
38  export async function getUser(name) {
39      const [result] = await ecommerce.query(`
40          SELECT *
41          FROM user_detail
42          WHERE name = ?
43      `, [name])
44      return result[0];
45  }
46
47  export async function createUser(email, name, password) {
48      const result = await ecommerce.query(`
49          CALL register_buyer(?, ?, ?)
50      `, [email, name, password]);
51      return result;
52  }
53
54  export async function addWish(buyer_id, prod_id) {
55      const result = await ecommerce.query(`CALL add_to_wish_list(? ,?)`, [buyer_id, prod_id]);
56      return result;
57  }
58
59  export async function removeWish(buyer_id, prod_id) {
60      const result = await ecommerce.query(`CALL remove_wishlist(? ,?)`, [buyer_id, prod_id]);
61      return result;
62  }
63
64  export async function removeCart(buyer_id, prod_id) {
65      const result = await ecommerce.query(`CALL remove_shopping_cart(? ,?)`, [buyer_id, prod_id]);
66      return result;
67  }
68
69  export async function addCart(buyer_id, prod_id) {
70      const result = await ecommerce.query(`CALL add_to_shopping_cart(? ,?)`, [buyer_id, prod_id]);
71      return result;
72  }
```

```
73
74  export async function getWishList() {
75      const [result] = await ecommerce.query(`select * from wish_list`);
76      return result;
77  }
78
79  export async function getCart() {
80      const [result] = await ecommerce.query(`select * from product_shoppingcart`);
81      return result;
82  }
83
84
```