DBMS Project Report

Shoppr

Ahmed Hanoon
2021006

Aajay Ayyappan Devaraj
2021001

Shoppr

An Online Retail Store

Project Scope:

Shopr is an online retail store similar to Amazon in which we maintain a database for customers, vendors, products and so on. Products sold by the vendors will be assigned to warehouses by the store, and Shopr will display a catalog of all the available products to the customers.

Shipping/transportation details, product information and ratings, payment information are all managed in the database management systems and are integrated together. Availability of shipping and products, and analytics are also overseen by the admin.

Interaction with the system takes place through a website.

A physical store's vendor-customer system often comes with many downsides such as limited inventory and retail hours, which are eliminated by an online store.
Shopr makes it possible to enjoy a pleasant experience in buying and selling products online. Inventory and product management are handled by the system and it can operate 24/7.

Technical Requirement:

- Ensures consistency in the data provided by customers and vendors, for example, every customer must provide his delivery address

- Using strict permission access, we make sure customers and vendors only view and edit data they are authorized to. Vendors cannot view customer payment details.

- Handling large data and optimizing them for better website performance. Products can have IDs for direct access.

- Validating information provided by customers and vendors to prevent wrong data from tampering with the database, for example, product price cannot be negative.

- Admin have complete control of the website but cannot change sensitive data like customer payment details

Tech Stack: ReactJS (Frontend JavaScript framework), Django (Python Web Framework), MySQL (Database), TailwindCSS (CSS library)

Functional Requirement:

1. Customer Functionalities

Customers can register and log in to the website and browse the catalog of products.
They can then view Item Details, Availability, and add the desired items to

their cart.

Then the customers may choose to buy the items in their cart by providing their shipment details and payment information. The cart will display the total price including shipping charges along with delivery information. The user can review the items too.

2. Vendor Functionalities

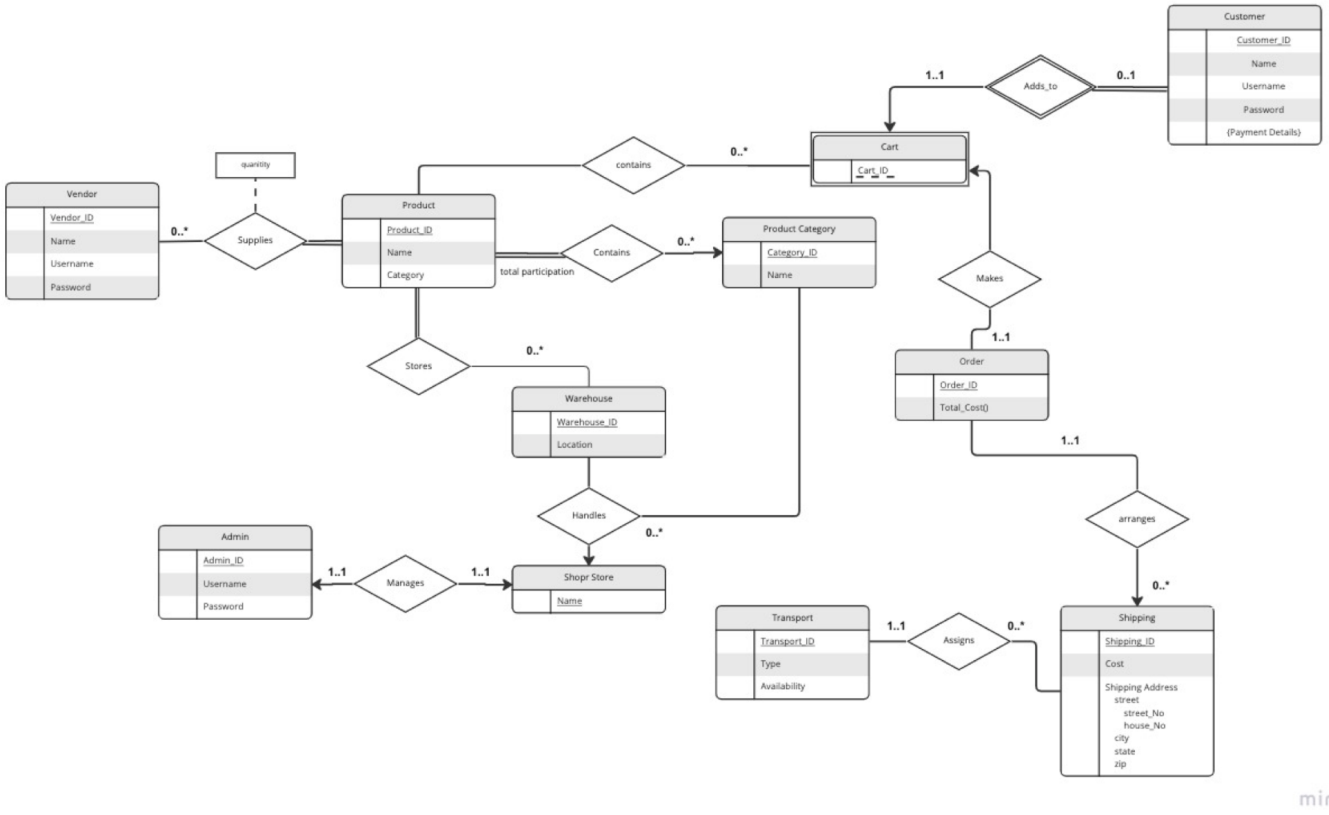
The vendor can register and add products along with their details. Vendors can also then edit details about their products like their stock. Vendors get updates about their sales and stock of their products from the website.

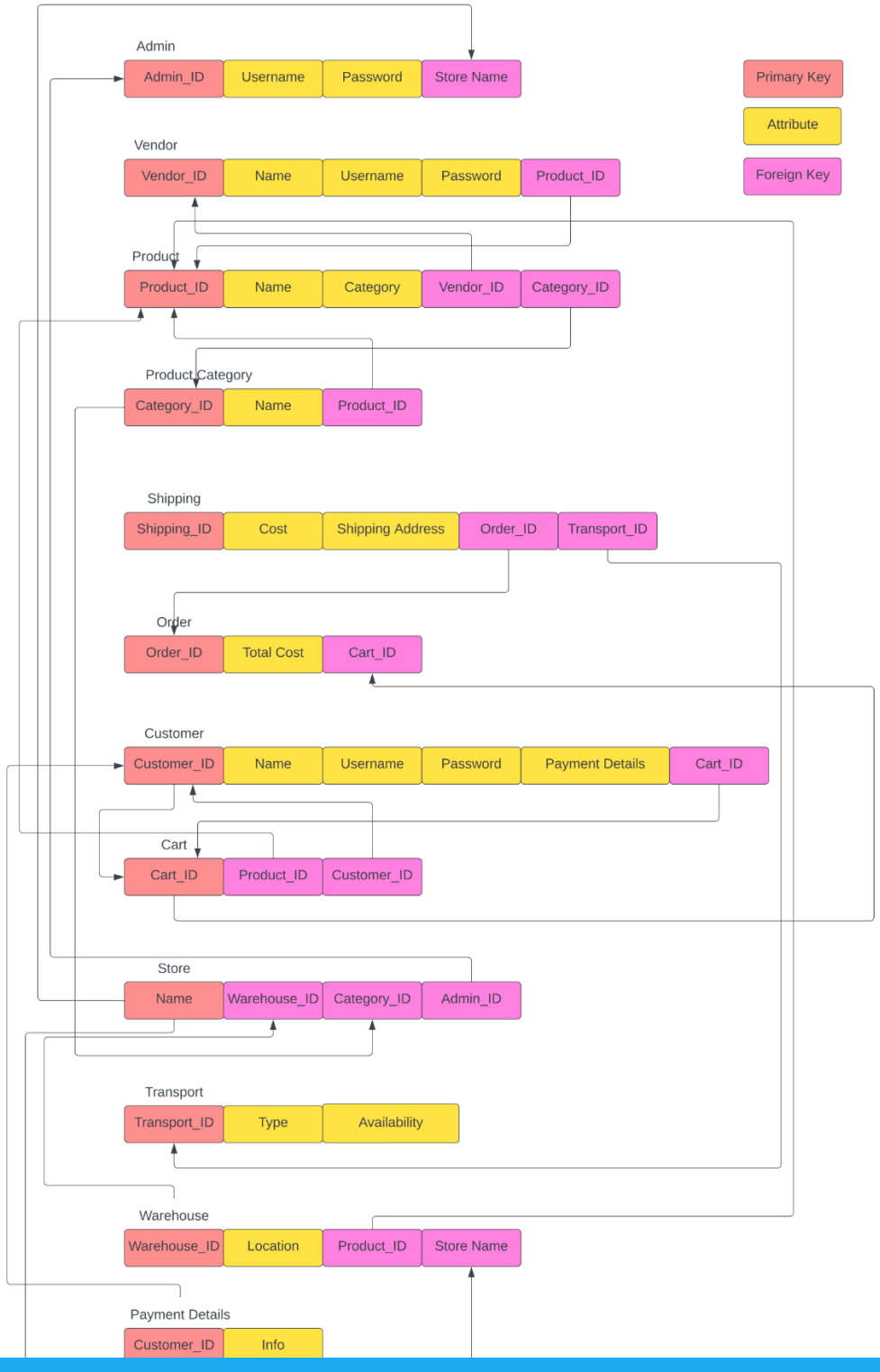
3. Admin Functionalities

The admin can login onto the site and view and manage all listed products. The admin may view the general information and statistics of overall sales/buys on the website, and each particular vendor.

Apart from these user interactions, the database management system manages the warehouse assignment of products, the inventory management and shipping of products and payments.

Conceptual Model Design and Relational model




Admin
Admin_ID
Username
Password
Store Name
Primary Key
Attribute
Foreign Key
Vendor
Vendor_ID
Name
Username
Password
Product_ID
Product
Product_ID
Name
Category
Vendor_ID
Category_ID
Product_Category
Category_ID
Name
Product_ID
Shipping
Shipping_ID
Cost
Shipping Address
Order_ID
Transport_ID
Order
Order_ID
Total Cost
Cart_ID
Customer
Customer_ID
Name
Username
Password
Payment Details
Cart_ID
Cart
Cart_ID
Product_ID
Customer_ID
Store
Name
Warehouse_ID
Category_ID
Admin_ID
Transport
Transport_ID
Type
Availability
Warehouse
Warehouse_ID
Location
Product_ID
Store Name
Payment Details
Customer_ID
Info

The Database schema

```
CREATE TABLE Store (
   Store_Name char(6) NOT NULL PRIMARY KEY
);


CREATE TABLE Admin (
        admin_id INT NOT NULL PRIMARY KEY,
        username VARCHAR(50) NOT NULL,
        password VARCHAR(50) NOT NULL,
        Store_Name char(6) NOT NULL,
        FOREIGN KEY (Store_Name) REFERENCES STORE(Store_Name)
);
#Shoppr Store
INSERT INTO Store VALUES ('Shoppr');
#Admin
INSERT INTO Admin VALUES (1, 'admin', 'admin', 'Shoppr');



CREATE TABLE `Cart` (
                `Cart_ID` int(11) NOT NULL,
                `Product_ID` int(11) NOT NULL,
                `Quantity` int(11) NOT NULL,
                KEY `Cart_ID` (`Cart_ID`),
                KEY `Product_ID` (`Product_ID`),
                CONSTRAINT `Cart_ibfk_1` FOREIGN KEY (`Cart_ID`)
REFERENCES `Customer
(`Customer_ID`),
                CONSTRAINT `Cart_ibfk_2` FOREIGN KEY (`Product_ID`)
REFERENCES `Product`
```

```
(`Product_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;


INSERT INTO `Cart` (`Cart_ID`, `Product_ID`, `Quantity`) VALUES (0, 0, 3);
INSERT INTO `Cart` (`Cart_ID`, `Product_ID`, `Quantity`) VALUES (2, 1, 1);
INSERT INTO `Cart` (`Cart_ID`, `Product_ID`, `Quantity`) VALUES (3, 2, 4);
INSERT INTO `Cart` (`Cart_ID`, `Product_ID`, `Quantity`) VALUES (4, 3, 8);
INSERT INTO `Cart` (`Cart_ID`, `Product_ID`, `Quantity`) VALUES (5, 4, 2);


CREATE TABLE `Customer` (
                        `Customer_ID` int(11) NOT NULL,
                        `Customer_Name` varchar(50) COLLATE
utf8mb4_unicode_ci NOT NULL,
                        `Customer_Username` varchar(50) COLLATE
utf8mb4_unicode_ci NOT NULL,
                        `Customer_Password` varchar(50) COLLATE
utf8mb4_unicode_ci NOT NULL,
                        `Customer_Address` varchar(50) COLLATE
utf8mb4_unicode_ci NOT NULL,
                        `Customer_Payment` varchar(50) COLLATE
utf8mb4_unicode_ci NOT NULL,
                        PRIMARY KEY (`Customer_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;


INSERT INTO `Customer` (`Customer_ID`, `Customer_Name`,
`Customer_Username`, `Customer_Password`, `Customer_Address`,
`Customer_Payment`) VALUES (0, 'et', 'veniam', '67414', '53405 Yundt
Summit\nLeuschkebury, SC 45177-6964', '6011126129078015');

INSERT INTO `Customer` (`Customer_ID`, `Customer_Name`,
`Customer_Username`, `Customer_Password`, `Customer_Address`,
```

`Customer_Payment`) VALUES (2, 'porro', 'aut', '5429401', '1428 Candelario Radial\nWest Salvador, PA 77005-847', '4929580778464');

INSERT INTO `Customer` (`Customer_ID`, `Customer_Name`, `Customer_Username`, `Customer_Password`, `Customer_Address`, `Customer_Payment`) VALUES (3, 'ipsum', 'est', '', '43795 Paucek Spring Apt. 903\nWest Julien, UT 93850', '342521549280215');

INSERT INTO `Customer` (`Customer_ID`, `Customer_Name`, `Customer_Username`, `Customer_Password`, `Customer_Address`, `Customer_Payment`) VALUES (4, 'ratione', 'quas', '17895', '123 Harris Greens\nSerenityhaven, AZ 39204', '379999663505682');

INSERT INTO `Customer` (`Customer_ID`, `Customer_Name`, `Customer_Username`, `Customer_Password`, `Customer_Address`, `Customer_Payment`) VALUES (5, 'qui', 'nihil', '8', '0039 Jordan Plaza Apt. 679\nMarcelinoburgh, UT 9688', '5374609951617978');

```
CREATE TABLE `Orders` (
                `Order_ID` int(11) NOT NULL,
                `Order_Cost` int(11) NOT NULL,
                `CART_ID` int(11) NOT NULL,
                PRIMARY KEY (`Order_ID`),
                KEY `CART_ID` (`CART_ID`),
                CONSTRAINT `Orders_ibfk_1` FOREIGN KEY (`CART_ID`)
REFERENCES `Cart` (`Cart_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

INSERT INTO `Orders` (`Order_ID`, `Order_Cost`, `CART_ID`) VALUES (0, 0, 127);

```
INSERT INTO `Orders` (`Order_ID`, `Order_Cost`, `CART_ID`) VALUES (1, 1,
3804374);
INSERT INTO `Orders` (`Order_ID`, `Order_Cost`, `CART_ID`) VALUES (2, 1,
4920);
INSERT INTO `Orders` (`Order_ID`, `Order_Cost`, `CART_ID`) VALUES (3,
1794, 7429);
INSERT INTO `Orders` (`Order_ID`, `Order_Cost`, `CART_ID`) VALUES (4, 270
, 94);


CREATE TABLE `Product` (
                `Product_ID` int(11) NOT NULL,
                `Product_Name` varchar(50) COLLATE utf8mb4_unicode_ci
NOT NULL,
                `Price` int(11) NOT NULL,
                `Warehouse_ID` int(11) NOT NULL,
                `Vendor_ID` int(11) NOT NULL,
                `Category_ID` int(11) NOT NULL,
                PRIMARY KEY (`Product_ID`),
                KEY `Warehouse_ID` (`Warehouse_ID`),
                KEY `Vendor_ID` (`Vendor_ID`),
                KEY `Category_ID` (`Category_ID`),
                CONSTRAINT `Product_ibfk_1` FOREIGN KEY
(`Warehouse_ID`) REFERENCES
`Warehouse` (`Warehouse_ID`),
                CONSTRAINT `Product_ibfk_2` FOREIGN KEY (`Vendor_ID`)
REFERENCES
`Vendor` (`Vendor_ID`),
                CONSTRAINT `Product_ibfk_3` FOREIGN KEY
(`Category_ID`) REFERENCES
`Product_Category` (`Category_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

#

INSERT INTO `Product` (`Product_ID`, `Product_Name`, `Price`, `Warehouse_ID`, `Vendor_ID`, `Category_ID`) VALUES (0, 'iPhone 13 Pro Max', 1499, 1001, 11, 101);

INSERT INTO `Product` (`Product_ID`, `Product_Name`, `Price`, `Warehouse_ID`, `Vendor_ID`, `Category_ID`) VALUES (1, 'Samsung Galaxy S21 Ultra', 1199, 1002, 12, 202);

INSERT INTO `Product` (`Product_ID`, `Product_Name`, `Price`, `Warehouse_ID`, `Vendor_ID`, `Category_ID`) VALUES (2, 'iPad Pro', 999, 1003, 13, 303);

INSERT INTO `Product` (`Product_ID`, `Product_Name`, `Price`, `Warehouse_ID`, `Vendor_ID`, `Category_ID`) VALUES (3, 'Apple Watch Series 7', 399, 1004, 14, 404);

INSERT INTO `Product` (`Product_ID`, `Product_Name`, `Price`, `Warehouse_ID`, `Vendor_ID`, `Category_ID`) VALUES (4, 'Sony WH-1000XM4', 349, 1005, 15, 505);

CREATE TABLE `Product_Category` (
`Category_ID` int(11) NOT NULL,
`Category_Name` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,
PRIMARY KEY (`Category_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

INSERT INTO `Product_Category` (`Category_ID`, `Category_Name`) VALUES

```
(101, 'Technology');
INSERT INTO `Product_Category` (`Category_ID`, `Category_Name`) VALUES
(202, 'Appliances');
INSERT INTO `Product_Category` (`Category_ID`, `Category_Name`) VALUES
(303, 'Apparels');

INSERT INTO `Product_Category` (`Category_ID`, `Category_Name`) VALUES
(404, 'Shoes');
INSERT INTO `Product_Category` (`Category_ID`, `Category_Name`) VALUES
(505, 'Sale');


CREATE TABLE `Shipping` (
                        `Shipping_ID` int(11) NOT NULL,
                        `Shipping_Cost` int(11) NOT NULL,
                        `Shipping_Address` varchar(50) COLLATE
utf8mb4_unicode_ci NOT NULL,
                        `Order_ID` int(11) NOT NULL,
                        PRIMARY KEY (`Shipping_ID`),
                        KEY `Order_ID` (`Order_ID`),
                        CONSTRAINT `Shipping_ibfk_1` FOREIGN KEY (`Order_ID`)
REFERENCES
`Orders` (`Order_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;


INSERT INTO `Shipping` (`Shipping_ID`, `Shipping_Cost`, `Shipping_Address`,
`Order_ID`) VALUES (0, 6, '1908 Carol Highway\nNorth Alessandro, DE
63971', 26);

INSERT INTO `Shipping` (`Shipping_ID`, `Shipping_Cost`, `Shipping_Address`,
`Order_ID`) VALUES (3, 5, '5263 Liam Springs Apt. 897\nHalvorsontown, OR
85716', 9);
```

INSERT INTO `Shipping` (`Shipping_ID`, `Shipping_Cost`, `Shipping_Address`, `Order_ID`) VALUES (4, 3, '41764 Beulah Inlet\nPort Dariusmouth, OK 85240-0016', 7309);

INSERT INTO `Shipping` (`Shipping_ID`, `Shipping_Cost`, `Shipping_Address`, `Order_ID`) VALUES (5, 3, '011 Kunze Ridge\nKulasborough, FL 73660-7383', 39);

INSERT INTO `Shipping` (`Shipping_ID`, `Shipping_Cost`, `Shipping_Address`, `Order_ID`) VALUES (6, 4, '905 Katlynn Mills Apt. 711\nNew Mabelle, WI 44110-7', 483);

```
CREATE TABLE `Transport` (
                        `Transport_ID` int(11) NOT NULL,
                        `Transport_Type` varchar(50) COLLATE
utf8mb4_unicode_ci NOT NULL,
                        `Transport_Availability` varchar(50) COLLATE
utf8mb4_unicode_ci NOT NULL,
                        `Shipping_ID` int(11) NOT NULL,
                        PRIMARY KEY (`Transport_ID`),
                        KEY `Shipping_ID` (`Shipping_ID`),
                        CONSTRAINT `Transport_ibfk_1` FOREIGN KEY
(`Shipping_ID`) REFERENCES
`Shipping` (`Shipping_ID`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_unicode_ci;
```

INSERT INTO `Transport` (`Transport_ID`, `Transport_Type`, `Transport_Availability`, `Shipping_ID`) VALUES (0, 'consequatur', '', 8);

INSERT INTO `Transport` (`Transport_ID`, `Transport_Type`, `Transport_Availability`, `Shipping_ID`) VALUES (1, 'rerum', '', 181);

INSERT INTO `Transport` (`Transport_ID`, `Transport_Type`, `Transport_Availability`, `Shipping_ID`) VALUES (2, 'praesentium', '1', 31);

INSERT INTO `Transport` (`Transport_ID`, `Transport_Type`, `Transport_Availability`, `Shipping_ID`) VALUES (4, 'quis', '', 2825586);

INSERT INTO `Transport` (`Transport_ID`, `Transport_Type`, `Transport_Availability`, `Shipping_ID`) VALUES (6, 'et', '1', 75506);

CREATE TABLE `Vendor` (

`Vendor_ID` int(11) NOT NULL,

`Vendor_Name` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,

`Vendor_Username` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,

`Vendor_Password` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,

PRIMARY KEY (`Vendor_ID`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

INSERT INTO `Vendor` (`Vendor_ID`, `Vendor_Name`, `Vendor_Username`, `Vendor_Password`) VALUES (0, 'doloremque', 'qui', '6');

INSERT INTO `Vendor` (`Vendor_ID`, `Vendor_Name`, `Vendor_Username`, `Vendor_Password`) VALUES (1, 'sapiente', 'harum', '8');

INSERT INTO `Vendor` (`Vendor_ID`, `Vendor_Name`, `Vendor_Username`, `Vendor_Password`) VALUES (2, 'numquam', 'qui', '1');

INSERT INTO `Vendor` (`Vendor_ID`, `Vendor_Name`, `Vendor_Username`, `Vendor_Password`) VALUES (3, 'adipisci', 'est', '9');

INSERT INTO `Vendor` (`Vendor_ID`, `Vendor_Name`, `Vendor_Username`, `Vendor_Password`) VALUES (4, 'autem', 'excepturi', '6');

CREATE TABLE `Warehouse` (

`Warehouse_ID` int(11) NOT NULL,

`Location` varchar(50) COLLATE utf8mb4_unicode_ci NOT NULL,

PRIMARY KEY (`Warehouse_ID`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

INSERT INTO `Warehouse` (`Warehouse_ID`, `Location`) VALUES (0, '293 Cooper Well\nNorth Opheliatown, MI 37207-2183');

INSERT INTO `Warehouse` (`Warehouse_ID`, `Location`) VALUES (1, '9738 Buckridge Springs\nGloverville, MI 25957-6766');

INSERT INTO `Warehouse` (`Warehouse_ID`, `Location`) VALUES (2, '148 Bobbie Hill\nDareview, MO 93576-5632');

INSERT INTO `Warehouse` (`Warehouse_ID`, `Location`) VALUES (3, '2822 Tremblay Plaza\nLake Wavaburgh, DC 01992-9396');

INSERT INTO `Warehouse` (`Warehouse_ID`, `Location`) VALUES (4, '432 Darien Center\nSouth Sydni, NM 18554-0674');

SQL queries supporting the application features

Queries Used in the main program:

1)

UPDATE shoppr.product SET Product_Name = ?, Price = ?, Category_ID = ?, Vendor_ID = ?, Warehouse_ID = ? WHERE Product_ID = ?

[productName, productPrice, categoryID, vendorID, warehouseID, productID]

2)

SELECT * FROM shoppr.customer WHERE Customer_Username = ? AND Customer_Password = ?

[username, password]

3)

SELECT Product.Product_ID, Product.Product_Name, Product.Price, Cart.Quantity FROM shoppr.product INNER JOIN shoppr.cart ON Product.Product_ID = Cart.Product_ID WHERE Cart.Cart_ID = ?'

[customer_id]

4)

DELETE FROM shoppr.cart WHERE Cart_ID = ? AND Product_ID = ?

[cartID, productID]

5)
SELECT * FROM shoppr.product WHERE Price > 10000 AND Category_ID IN (' + category + ')

Some of the queries other queries to interact with the database:

6)

SELECT t.Transport_ID, t.Transport_Type, s.Shipping_Cost FROM Transport t

JOIN Shipping s ON t.Shipping_ID = s.Shipping_ID

WHERE t.Transport_Availability = 1 AND s.Shipping_Cost < 5;

7)

SELECT Transport_Type, COUNT(Transport_Availability) FROM transport

GROUP BY Transport_Type

ORDER BY COUNT(Transport_Availability) DESC;

8)

SELECT v.Vendor_Name, SUM(p.Price) AS Total_Value FROM Product p

INNER JOIN Vendor v ON p.Vendor_ID = v.Vendor_ID GROUP BY v.Vendor_ID;

9)

SELECT c.Customer_ID, c.Customer_Name, c.Customer_Username, c.Customer_Address FROM Customer c

WHERE EXISTS (

SELECT *

FROM Orders o

INNER JOIN Shipping s ON o.Order_ID = s.Order_ID WHERE o.CART_ID = c.Customer_ID

);

10)
SELECT c.Customer_Name,

(SELECT SUM(o.Order_Cost) FROM Orders o WHERE o.Cart_ID = c.Customer_ID) AS Total_Order_Cost,

(SELECT SUM(s.Shipping_Cost) FROM Shipping s JOIN Orders o ON s.Order_ID = o.Order_ID WHERE o.Cart_ID = c.Customer_ID) AS Total_Shipping_Cost

FROM Customer c;

Embedded SQL, OLAP queries, and Triggers

Embedded Queries -- JavaScript

a. User login

```
app.get('/customer/login' , (req, res) => {
 const username = req.query.username;
 const password = req.query.password;
 const q = 'SELECT * FROM shoppr.customer WHERE

Customer_Username = ? AND Customer_Password = ?'; db.query(q,
[username, password], (err, result) => {

if(err) throw err;

res.send(result); }

); });
```

b. Display all products

```
const category = req.query.category;
 var q;
```

```
 if(category!== undefined) q = `SELECT * FROM shoppr.product

WHERE Category_ID IN (${category})`;
 else q = `SELECT * FROM shoppr.product`; db.query(q, (err, result) => {

if(err) throw err;

res.send(result); }

);
```

OLAP Queries

```
#total revenue from each order
 SELECT Orders.Order_ID, SUM(Product.Price*Cart.Quantity) AS Revenue

FROM Orders
 JOIN Cart ON Orders.CART_ID = Cart.Cart_ID
 JOIN Product ON Cart.Product_ID = Product.Product_ID GROUP BY
Orders.Order_ID;

#total revenue from each category
 SELECT Product_Category.Category_Name,
SUM(Product.Price*Cart.Quantity)

AS Revenue FROM Product
 JOIN Cart ON Product.Product_ID = Cart.Product_ID
 JOIN Product_Category ON Product.Category_ID =
Product_Category.Category_ID
 GROUP BY Product_Category.Category_Name;
```

3)

```
SELECT pc.Category_Name, SUM(p.Price * o.Order_Cost) as Total_Revenue
FROM Product p
 JOIN Product_Category pc ON p.Category_ID = pc.Category_ID
 JOIN Cart c ON p.Product_ID = c.Product_ID

JOIN Orders o ON c.Cart_ID = o.CART_ID GROUP BY pc.Category_Name
WITH ROLLUP;
```

4)

```
SELECT c.Customer_ID, COUNT(o.Order_ID) as total_orders FROM
Customer c
 JOIN Cart ct ON ct.Cart_ID = c.Customer_ID
 JOIN Orders o ON o.Cart_ID = ct.Cart_ID

GROUP BY c.Customer_ID WITH ROLLUP;
```

Triggers

1)

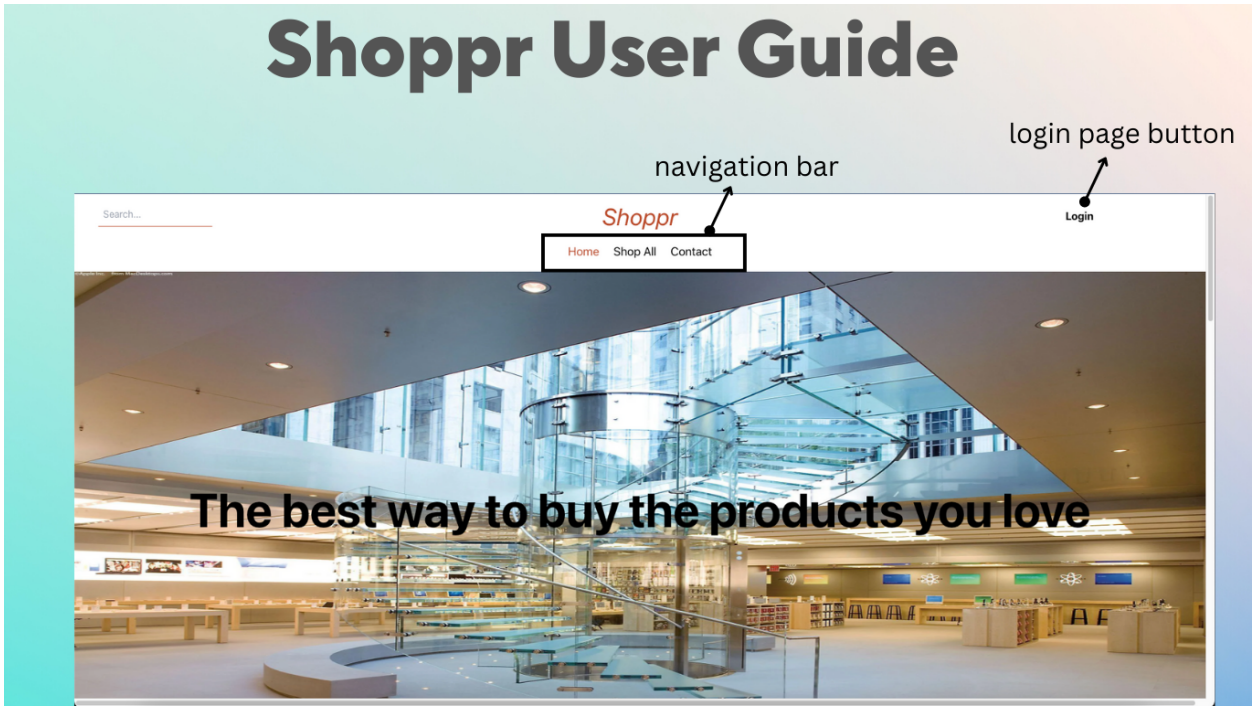
```
DELIMITER //
 CREATE TRIGGER enforce_price_minimum BEFORE INSERT ON Product

FOR EACH ROW
BEGIN

        IF NEW.Price <= 0 THEN
                SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Product
price must be

                greater than zero.';

                END IF;
```

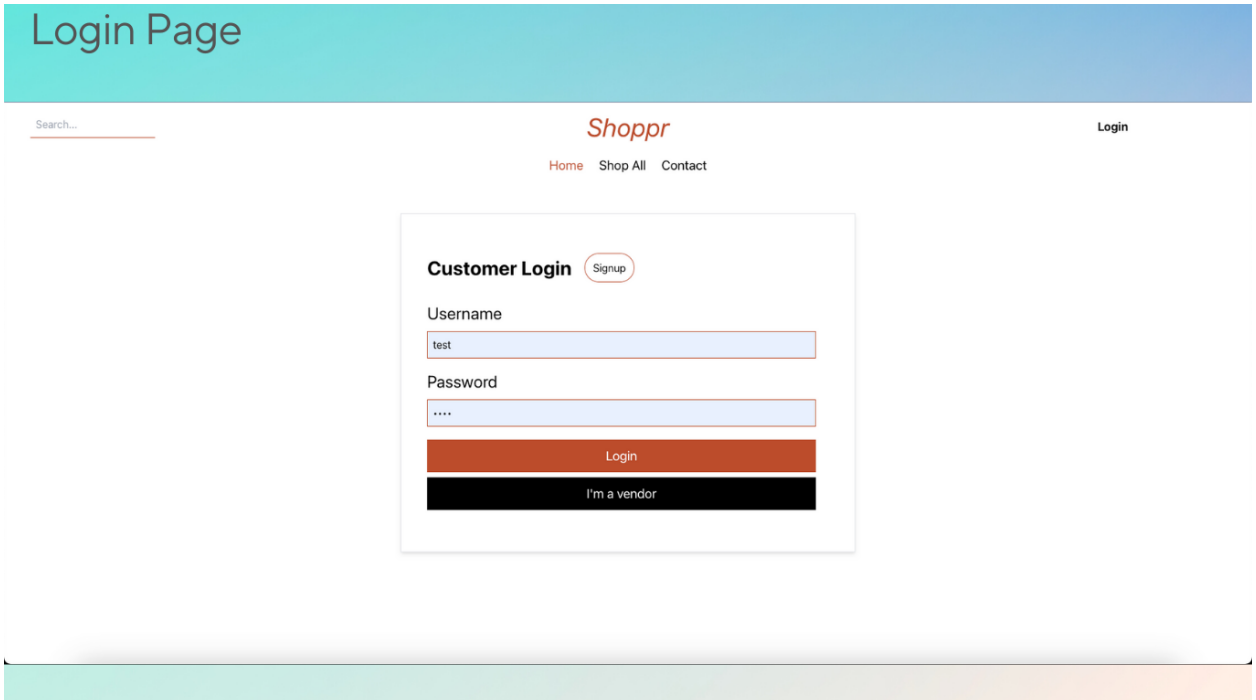
```
END;//

INSERT INTO `Product` (`Product_ID`, `Product_Name`, `Price`,
`Warehouse_ID`, `Vendor_ID`, `Category_ID`) VALUES (140, 'Nintendo Switch
OLED', -1, 1005, 11, 303);
```

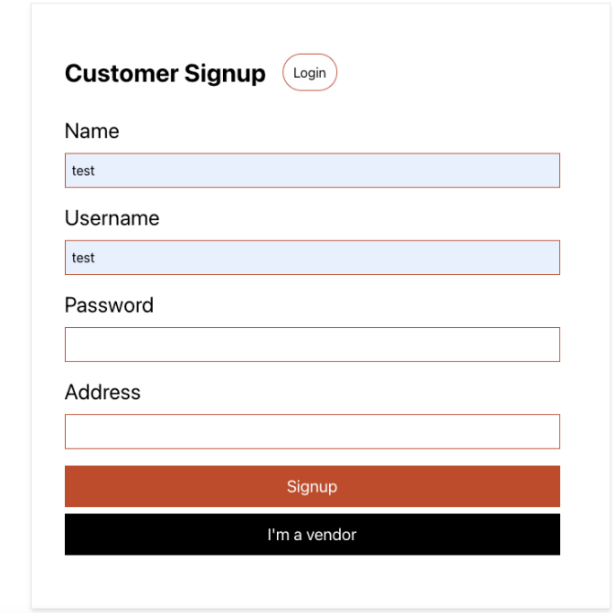
2)

```
DELIMITER //
 CREATE TRIGGER remove_product_from_carts AFTER DELETE ON Product
 FOR EACH ROW
 BEGIN

        DELETE FROM Cart WHERE product_id = OLD.product_id;

        END;//

DELETE FROM Product WHERE product_id = 1;
```

USER GUIDE FOR THE INTERFACE



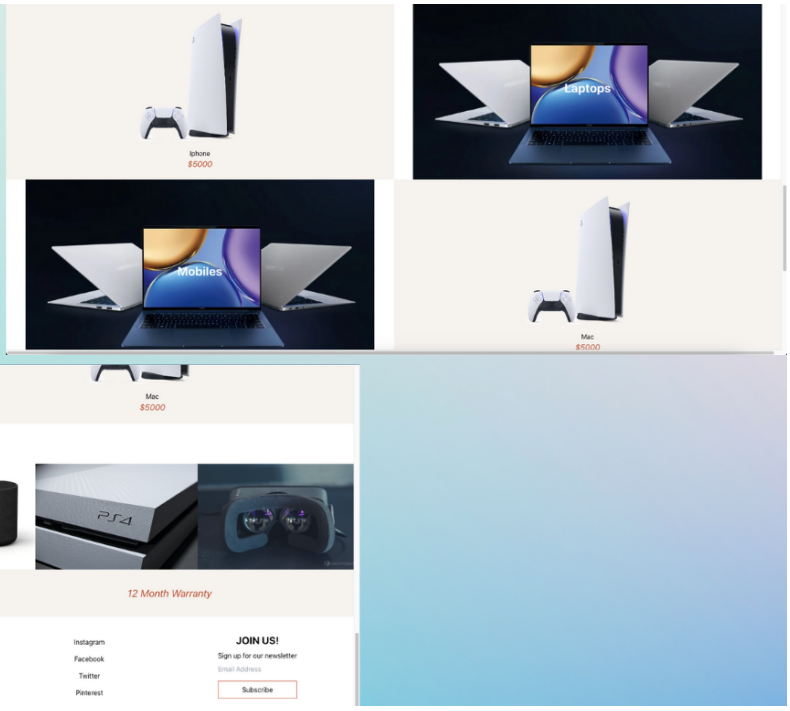


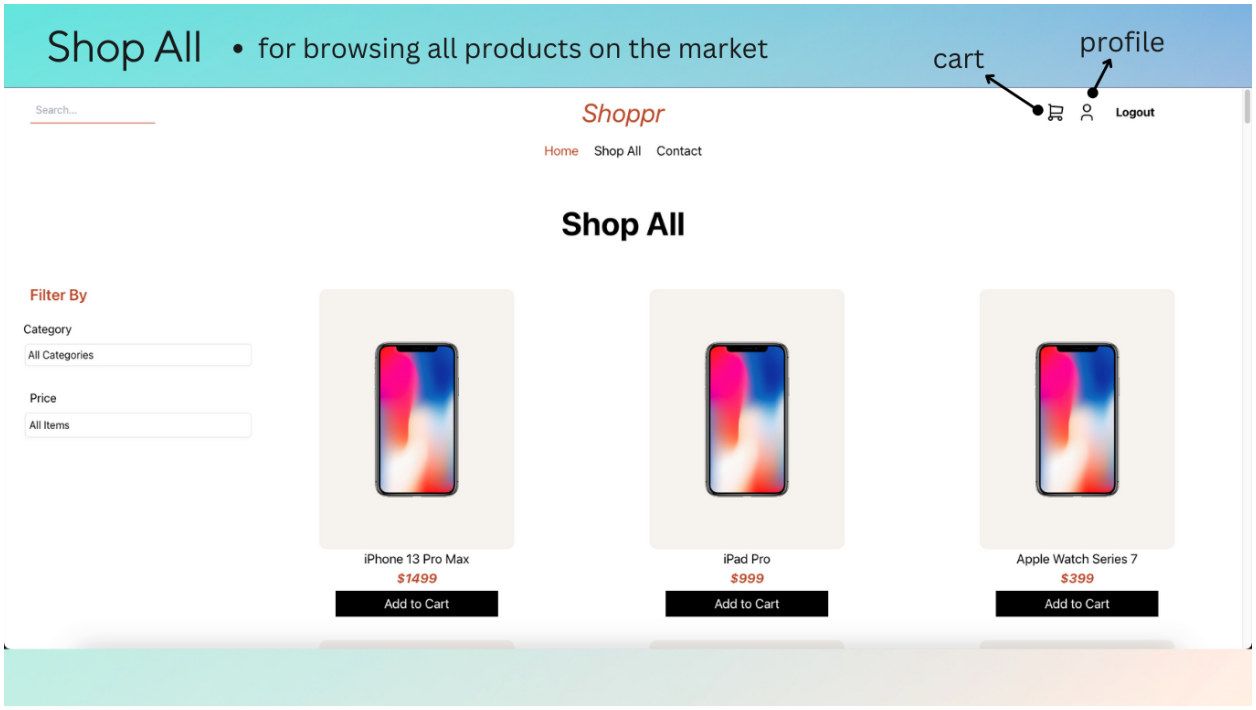


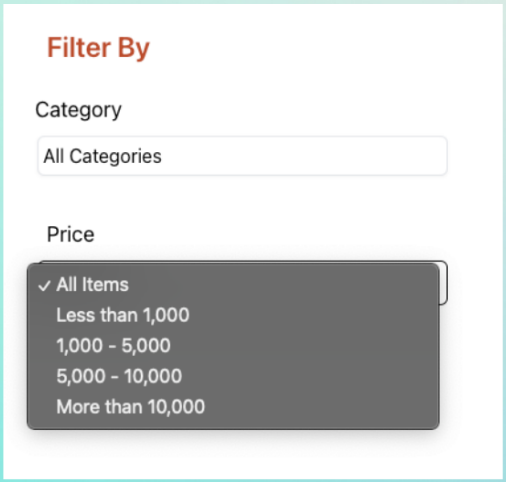
- users can also sign up to create a new account

Homepage

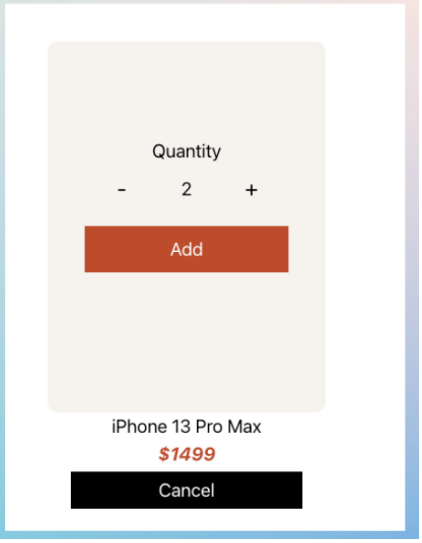
- refer title page







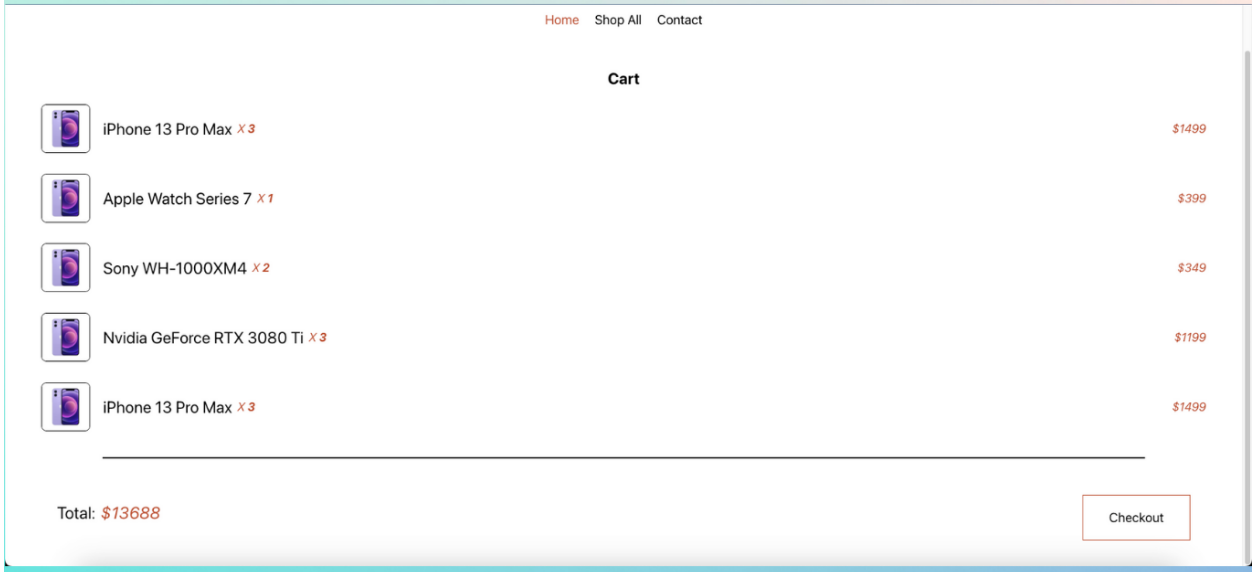
- Users can Filter products by category and price



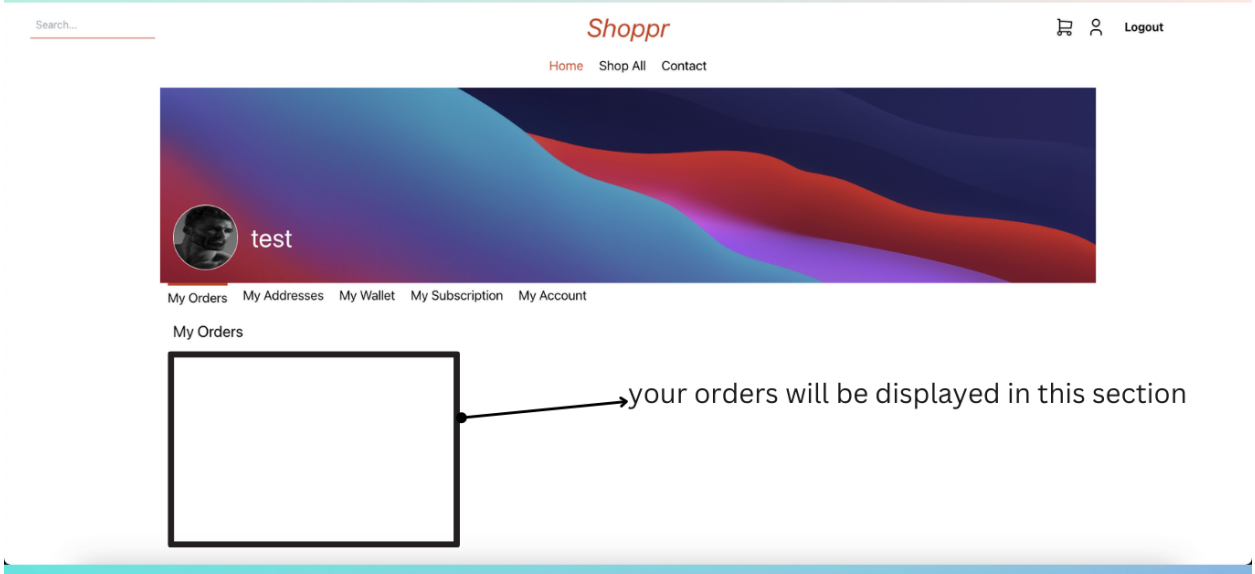
- Add products to cart after specifying quantity

Cart

- Review your cart and checkout

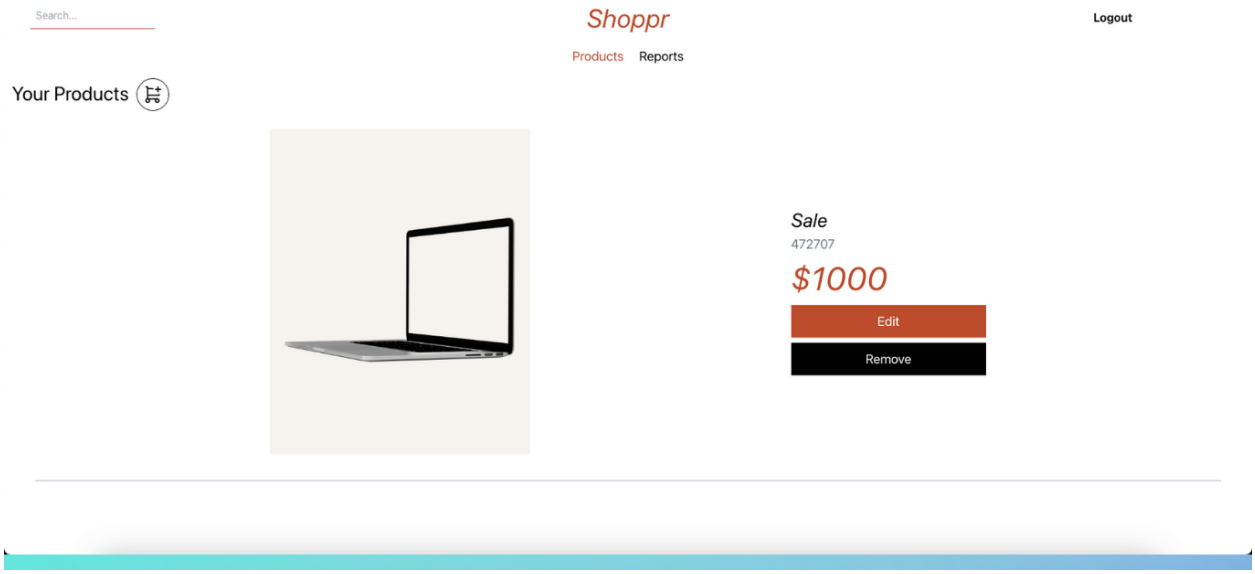


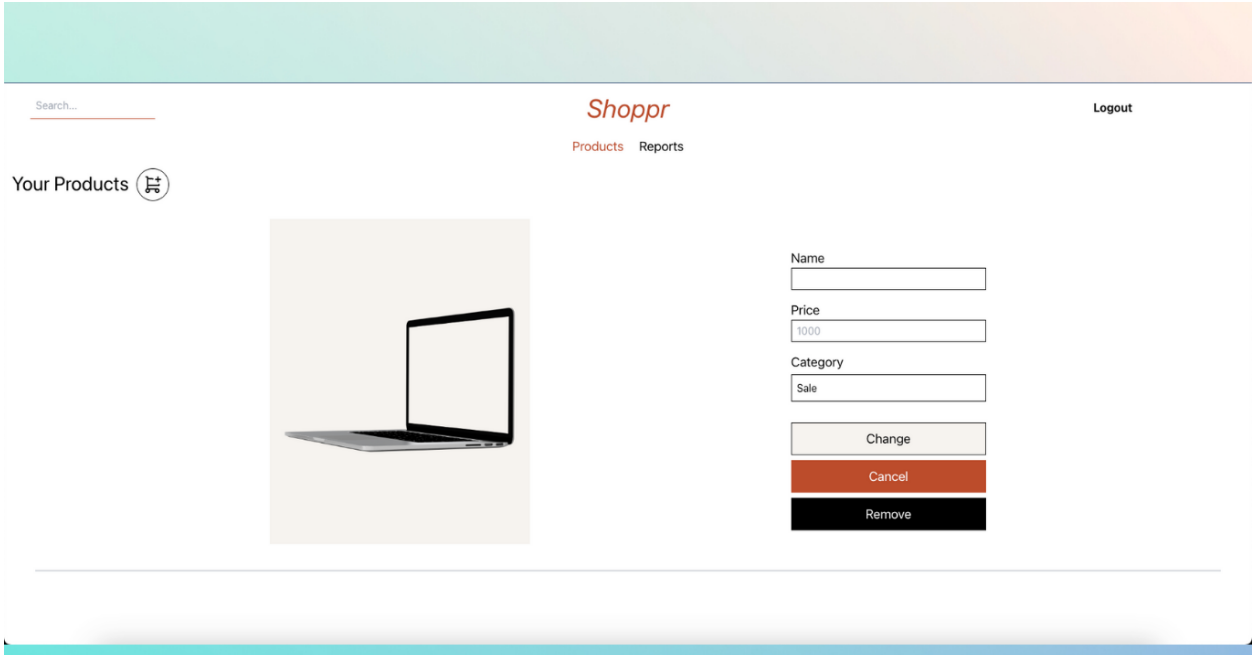
Profile



Vendor Home

vendor login/signup will take you into this page, where vendors can review their products and edit details





- Admin page can be used to run transactions that create a user, vendor and product