ONLINE RETAIL MANAGEMENT SYSTEM

MEMBERS:

Mohamed Saeed

Kareem Emad

Eslam Mohamed

Mohamed Ahmed

APRIL 25, 2025

INFORMATION TECHNOLOGICAL INSTITUTE

Power BI Development Track

Contents

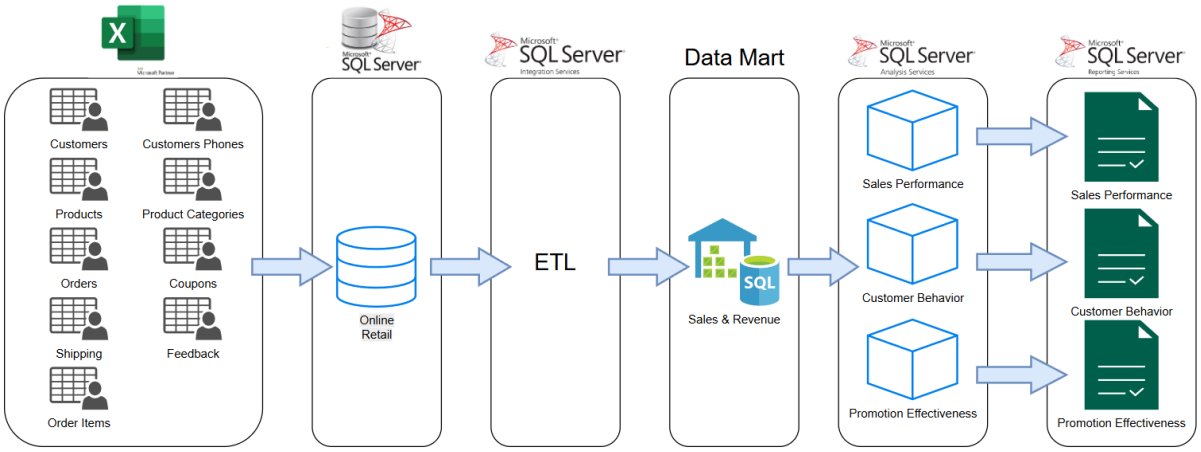
Executive Summary

The *Online Retail Management System* is a comprehensive end-to-end solution that manages customer purchases, product inventory, shipping, and payments. It supports strategic decisions through integrated analytics (SSAS), real-time reporting (SSRS), and a user-friendly interface built with Streamlit for customers to browse and order products.

Problem Statement

Retail businesses need a centralized system to manage inventory, track orders, optimize promotions, and personalize customer experiences. This project meets those needs with a robust database, integrated analytics and reporting, and a user-friendly web interface.

Project Workflow



This is a complete BI solution that consists of:

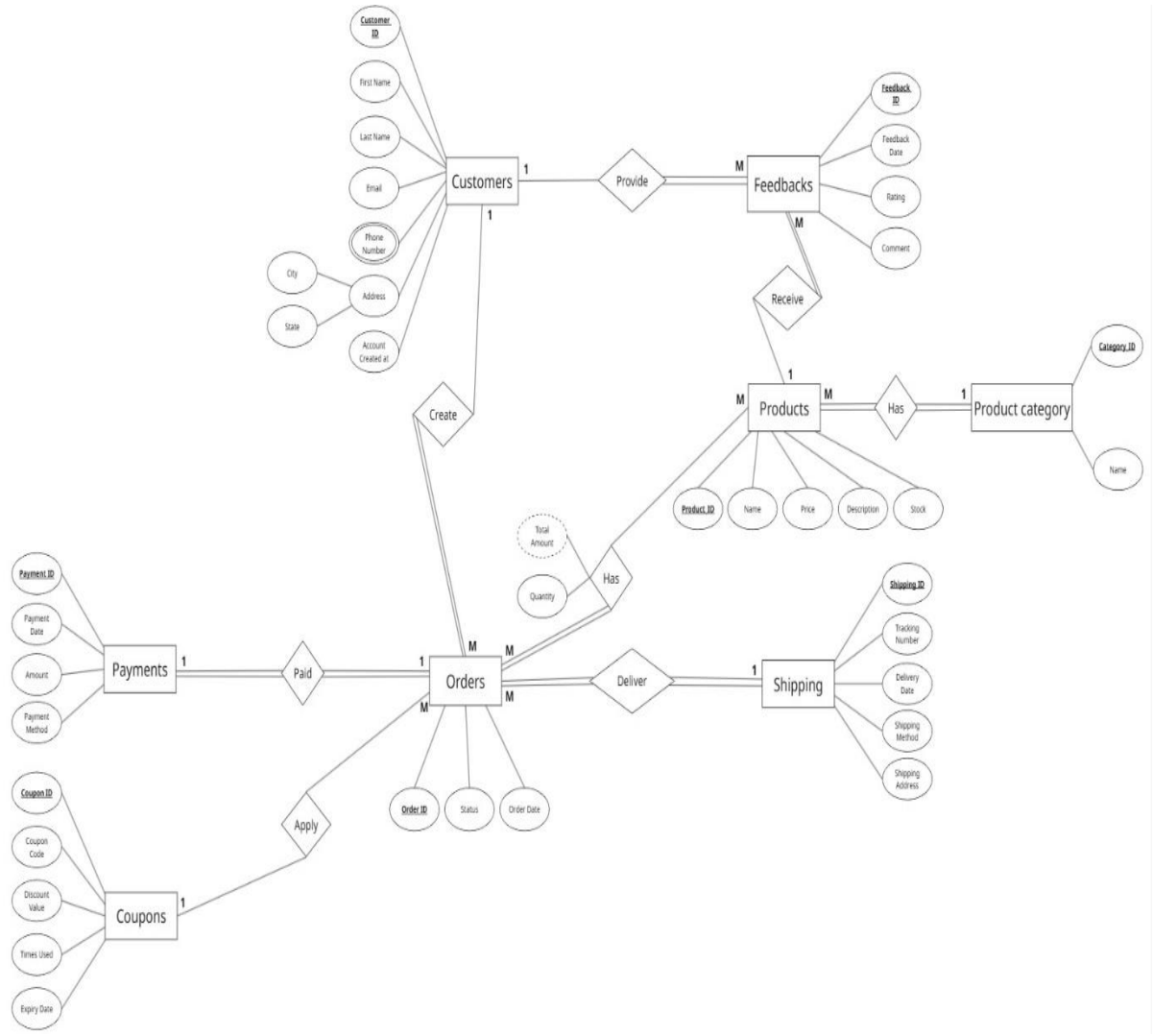
- Database Implementation.
- ETL using SSIS.
- Data Mart Creation.
- Cubes Creation Using SSAS.
- Business Reporting Using SSRS.

Business Case

- Each customer can place multiple orders, allowing for detailed tracking of purchase behaviour and customer segmentation.
- Customers may have multiple contact numbers, enhancing communication flexibility and delivery coordination.
- Customers can leave feedback and ratings for products they've purchased, supporting quality control and customer satisfaction.
- Each order is placed by one customer, includes payment details, may apply a coupon, and links to shipping and multiple products.
- Products are organized into categories, supporting easier navigation, reporting, and stock analysis.
- Products are linked to their categories and can be included in many orders, supporting versatile catalog management.
- Discounts can be applied through coupons, which include usage limits and expiration tracking to manage promotions effectively.
- Each order is associated with a shipping record that stores delivery details, method, and tracking information.

ER – Diagram

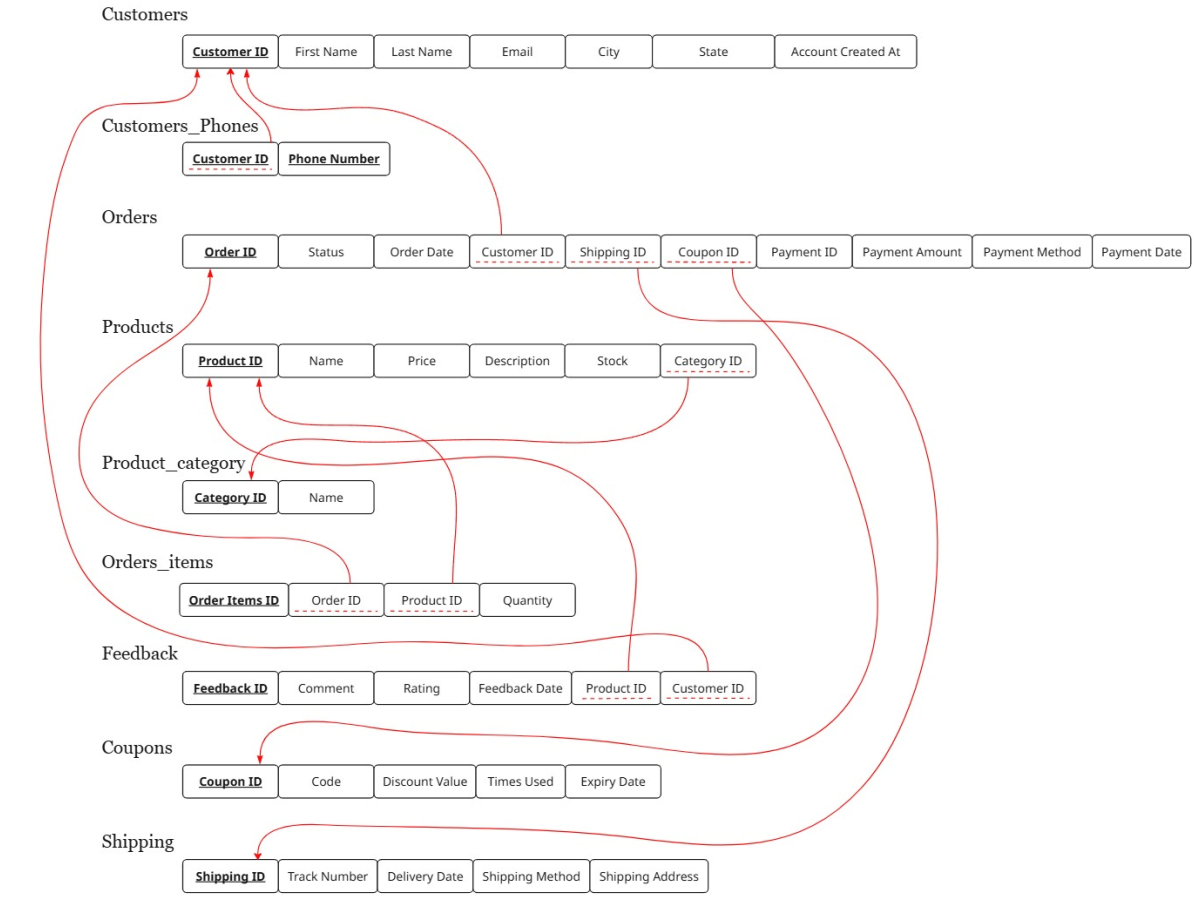
The ERD defines the structural blueprint of the database, capturing the entities, their attributes, and the relationships among them. It ensures data integrity, efficient querying, and logical organization.



This ERD supports the business case and foundational for building a scalable and consistent online retail system that integrates data management, analytics, and user functionality.

Mapping

The logical mapping outlines how the real-world business entities are represented in the database schema, ensuring data normalization, integrity, and consistency.



The “orders_items” table resolves the many-to-many relationship between orders and products.

Database Implementation

```
create table Customers(
    Customer_ID INT IDENTITY(1,1) PRIMARY KEY,
    F_Name varchar(50),
    L_Name varchar(50),
    Email varchar(50),
    City varchar(50),
    State varchar(50),
    Account_Created_At date);

GO

create table Customer_Phones(
    Customer_ID int,
    Phone varchar(50),
    primary key (Customer_ID,phone),
    foreign key (Customer_ID) references Customers(Customer_ID));

GO

create table Product_category(
    Category_ID INT IDENTITY(1,1) PRIMARY KEY,
    Name varchar(50));

GO

create table Products(
    Product_ID INT IDENTITY(1,1) PRIMARY KEY,
    Name varchar(100),
    Price float,
    Description varchar(255),
    Stock smallint,
    Category_ID int,
    foreign key (Category_ID) references Product_category(Category_ID));

GO
```

```sql
create table Coupons(
    Coupon_ID INT IDENTITY(1,1) PRIMARY KEY,
    Coupon_Code varchar(50),
    Discount_Value float,
    Times_Used tinyint,
    Expiry_Date date);

GO

create table Shipping(
    Shipping_ID INT IDENTITY(1,1) PRIMARY KEY,
    Track_Number smallint,
    Delivery_Date date,
    Shipping_Method varchar(50),
    Shipping_Address varchar(50));

GO

create table Orders(
    Order_ID INT IDENTITY(1,1) PRIMARY KEY,
    Status varchar(50),
    Order_date date,
    Customer_ID int,
    Shipping_ID int,
    Coupon_ID int,
    Payment_ID int,
    Payment_Amount smallint,
    Payment_Method varchar(50),
    Payment_Date date,
    Foreign key (Customer_ID) references Customers(Customer_ID),
    Foreign key (Shipping_ID) references Shipping(Shipping_ID),
    Foreign key (Coupon_ID) references Coupons(Coupon_ID));

GO

create table Orders_Items(
    Order_Items_ID INT IDENTITY(1,1) PRIMARY KEY,
    Order_ID int,
    Product_ID int,
    Quantity tinyint,
    foreign key (Order_ID) references Orders(Order_ID),
    foreign key (Product_ID) references Products(Product_ID));

GO

create table Feedback(
    Feedback_ID INT IDENTITY(1,1) PRIMARY KEY,
    Comment varchar(max),
    Rating int,
    Feedback_Date date,
    Product_id int,
    Customer_id int,
    Foreign Key (Product_id) references Products(Product_id),
    Foreign Key (Customer_id) references Customers(Customer_id));
```

Database Ingestion

Used ***Bulk Insert*** to ingest the data into the database.

```
BULK INSERT
    Customers
FROM
    'D:\courses\ITI Power BI\4.DataBase\Database Project\Data Source\Customers.csv'
WITH
    (
    FIRSTROW = 2,
    FIELDTERMINATOR = ',',
    TABLOCK
    );

GO

BULK INSERT
    Orders
FROM
    'D:\courses\ITI Power BI\4.DataBase\Database Project\Data Source\Orders.csv'
WITH
    (
    FIRSTROW = 2,
    FIELDTERMINATOR = ',',
    TABLOCK
    );

GO

BULK INSERT
    Coupons
FROM
    'D:\courses\ITI Power BI\4.DataBase\Database Project\Data Source\Coupons.csv'
WITH
    (
    FIRSTROW = 2,
    FIELDTERMINATOR = ',',
    TABLOCK
    );
```

Database Integration

SQL Server Integration Services (SSIS) was implemented to build and manage the ***Sales & Revenue DataMart*** using the ***Kimball Approach*** within the ***Online Retail Management System***.

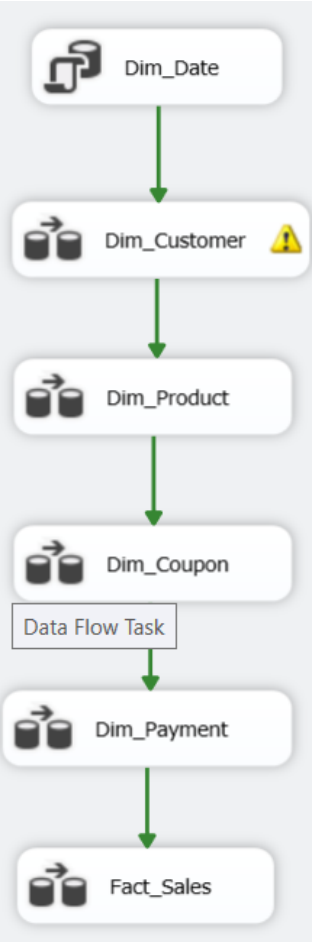
The process follows a logical sequence to extract, transform, and load data from the source system (OLTP) into the data warehouse:

- **Dim_Date:**

A date dimension is generated using a script to support time-based analysis.

- **Dim_Customer:**

Customer data is extracted from the OLTP system and loaded after necessary cleaning.



- **Dim_Product:**

Product details are imported to enable product-level insights.

- **Dim_Coupon:**

Coupon data is processed for promotional analysis.

- **Dim_Payment:**

Payment method data is loaded for transactional context.

- **Fact_Sales:**

Final sales data is loaded, linking all dimensions with key metrics for reporting.

Data Mart Implementation

A Sales & Revenue Data Mart was created to organize and store key business data in a structured way for reporting and analysis. The process included:

- Creating dimension tables like ***Dim_Customer*** to store customer information, ***Dim_Product*** for product details, ***Dim_Coupon*** for discounts, ***Dim_Payment*** for payment methods, and ***Dim_Date*** to track order dates.
- A central ***Fact_Sales*** table was created to record all sales transactions, linking to the dimension tables to provide full context for each order (who bought what, when, how, and with which discount).

```
CREATE TABLE Dim_Customer(
Customer_Key INT PRIMARY KEY IDENTITY(1,1),
Customer_ID INT,
Customer_Full_Name VARCHAR(100),
Customer_Email VARCHAR(50),
Customer_City VARCHAR(50),
Customer_State VARCHAR(50),
Customer_Since DATE,
Customer_Account_Age_Days INT,
Customer_Segment VARCHAR(20)
);

GO
```

```
CREATE TABLE Dim_Product(
Product_Key INT PRIMARY KEY IDENTITY(1,1),
Product_ID INT,
Product_Name VARCHAR(100),
Product_Description VARCHAR(255),
Product_Stock SMALLINT,
Product_Category VARCHAR(50),
Product_Price FLOAT
);
```

```
CREATE TABLE Dim_Coupon(
Coupon_Key INT PRIMARY KEY IDENTITY(1,1),
Coupon_ID INT,
Coupon_Code VARCHAR(50),
Coupon_Discount_Value FLOAT,
Coupon_Usage_Number TINYINT,
Coupon_Expiry_Date DATE,
Is_Expired BIT
);

GO
```

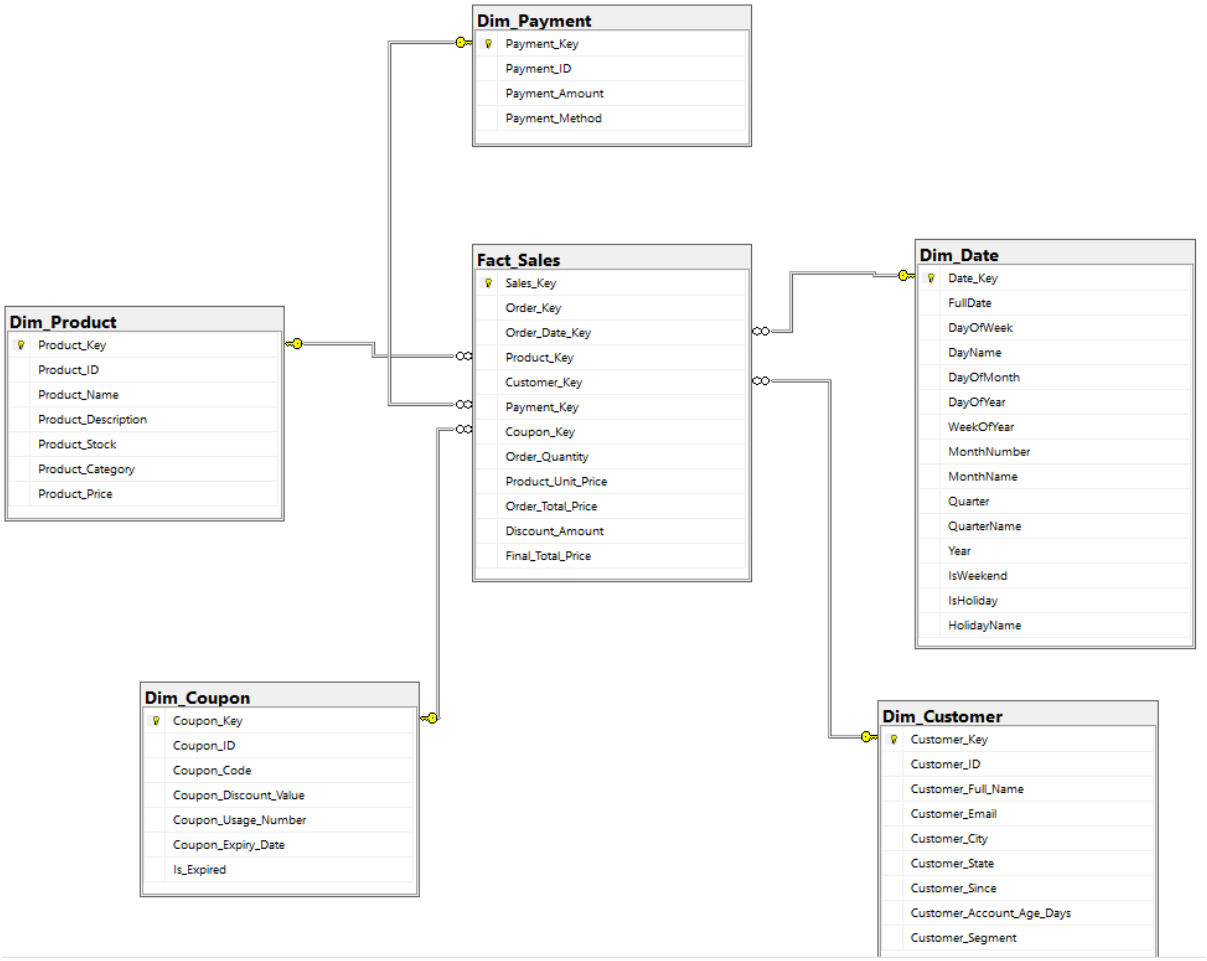
```
CREATE TABLE Dim_Payment(
Payment_Key INT PRIMARY KEY IDENTITY(1,1),
Payment_ID INT,
Payment_Amount SMALLINT,
Payment_Method VARCHAR(50)
);
```

```sql
CREATE TABLE Dim_Date (
    Date_Key INT PRIMARY KEY IDENTITY(1,1),
    FullDate DATE,
    DayOfWeek TINYINT,
    DayName VARCHAR(10),
    DayOfMonth TINYINT,
    DayOfYear SMALLINT,
    WeekOfYear TINYINT,
    MonthNumber TINYINT,
    MonthName VARCHAR(10),
    Quarter TINYINT,
    QuarterName VARCHAR(10),
    Year SMALLINT,
    IsWeekend BIT,
    IsHoliday BIT,
    HolidayName VARCHAR(50)
);

GO

CREATE TABLE Fact_Sales(
Sales_Key INT PRIMARY KEY IDENTITY(1,1),
Order_Key INT,
Order_Date_Key INT,
Product_Key INT,
Customer_Key INT,
Payment_Key INT,
Coupon_Key INT,
Order_Quantity TINYINT,
Product_Unit_Price FLOAT,
Order_Total_Price FLOAT,
Discount_Amount FLOAT,
Final_Total_Price FLOAT,

FOREIGN KEY (Order_Date_Key) REFERENCES Dim_Date(Date_Key),
FOREIGN KEY (Product_Key) REFERENCES Dim_Product(Product_Key),
FOREIGN KEY (Customer_Key) REFERENCES Dim_Customer(Customer_Key),
FOREIGN KEY (Payment_Key) REFERENCES Dim_Payment(Payment_Key),
FOREIGN KEY (Coupon_Key) REFERENCES Dim_Coupon(Coupon_Key)
);
```



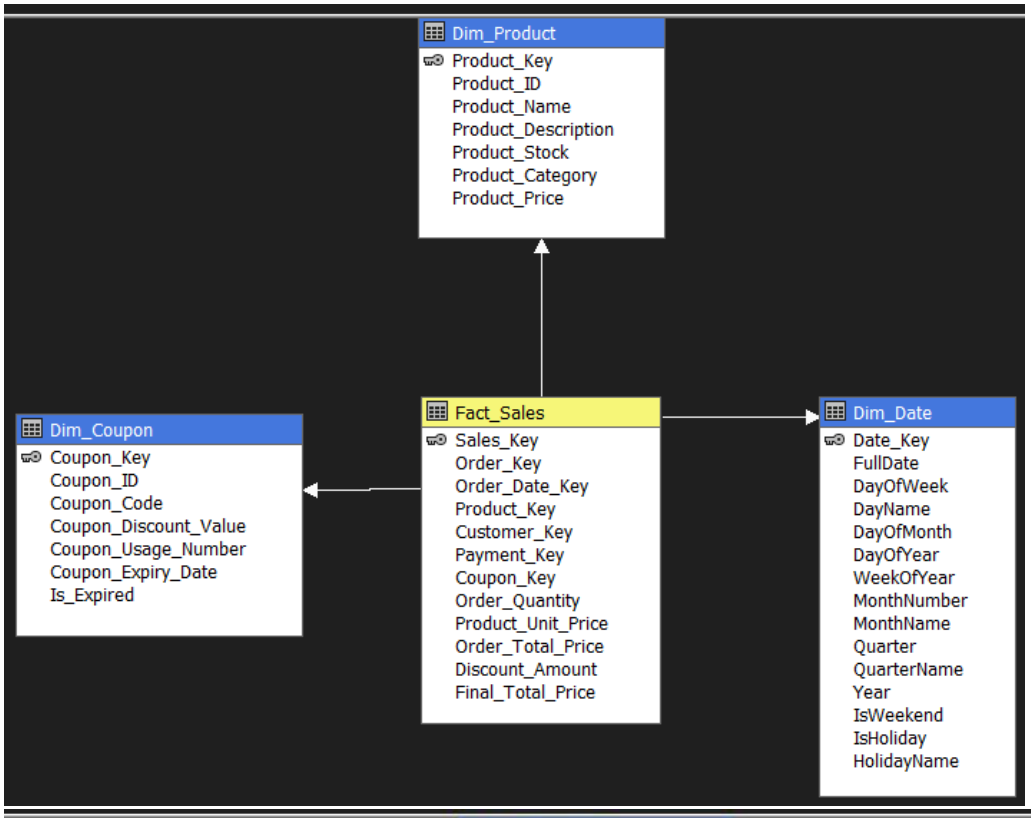
Data Mart Analysis

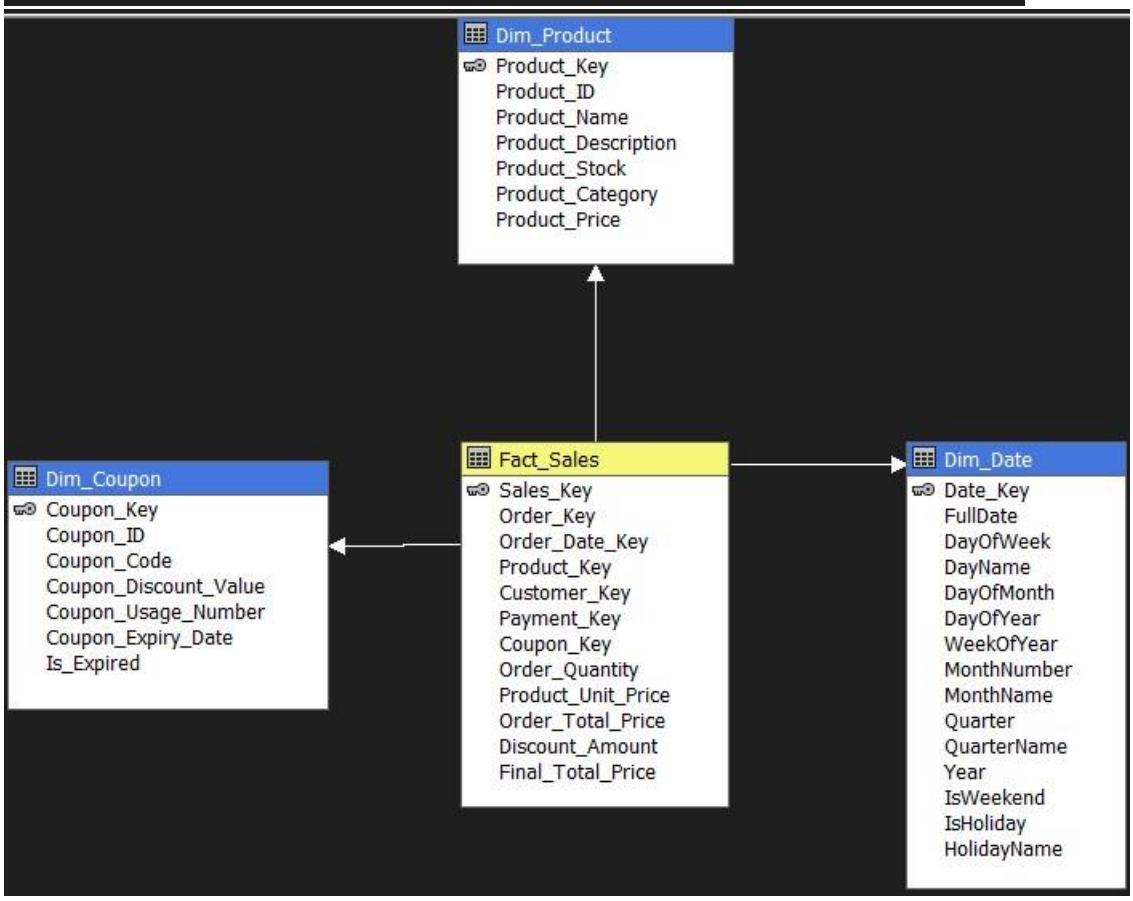
SQL Server Analysis Services (SSAS) was implemented to enable multidimensional analysis and support business intelligence reporting within the Online Retail Management System. SSAS provides a powerful platform to transform raw transactional data into meaningful insights for decision-making.

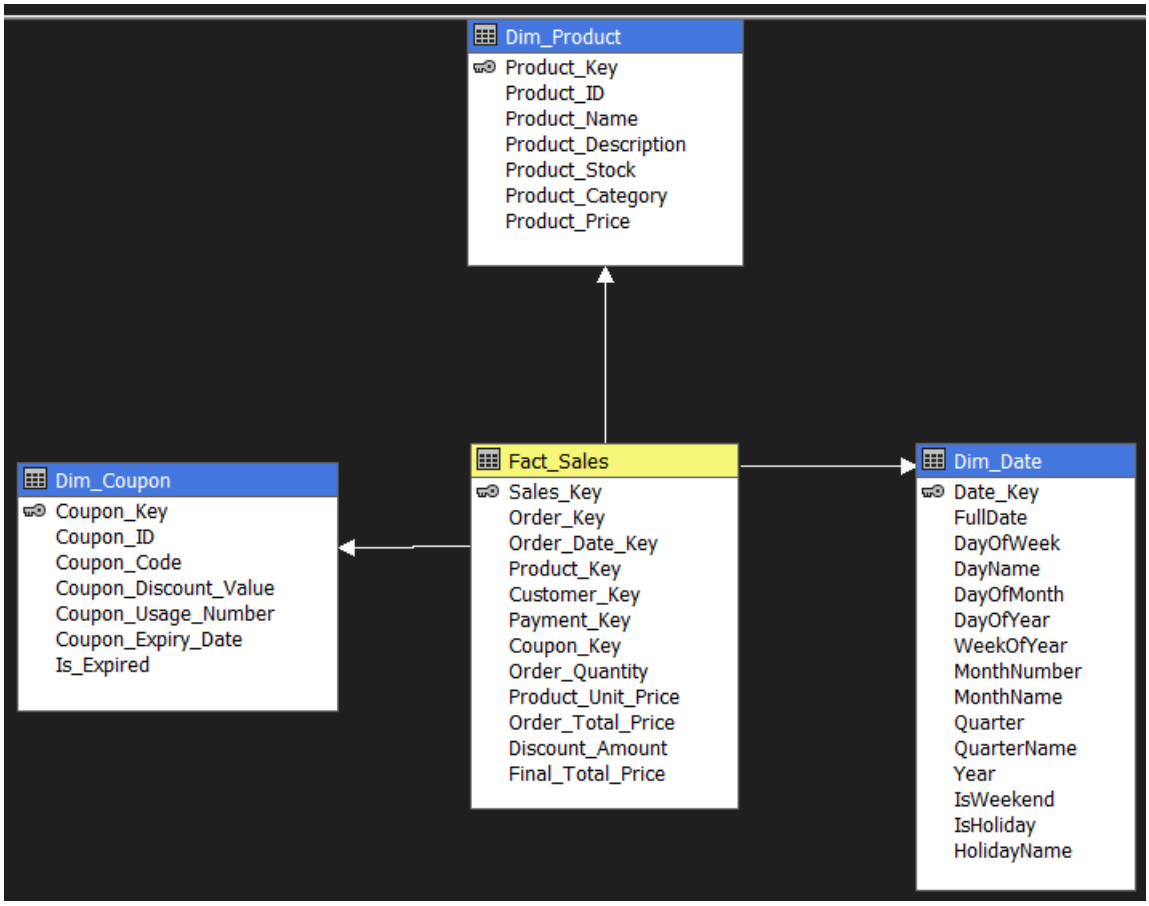
Three analytical cubes were created to support business intelligence reporting within the Online Retail Management System:

- **Sales Performance Cube:** Focuses on analyzing overall revenue, sales trends, and product performance across time and geography.
- **Customer Behavior Cube:** Helps understand customer purchasing patterns, segments, and lifecycle metrics like frequency and recency.

- **Promotion Effectiveness Cube:** Measures the impact of coupons and discounts on sales, tracking redemption rates and sales lift.







Data Mart Reporting

Three interactive reports were developed using SSRS to visualize insights drawn from the analytical cubes:

- **Sales Performance Report:** Displays key metrics such as total sales, revenue trends, and top-performing products across various time periods and regions.
- **Customer Behavior Report:** Highlights customer segments, repeat purchase behavior, and retention insights to better understand buying patterns.
- **Promotion Effectiveness Report:** Evaluates the success of promotional campaigns by tracking coupon usage, discount impact, and resulting sales lift.

Executive Sales Dashboard

Year	Quarter	Month	Category	Total Sales	Quantity Sold
2020	4	December	Furniture	823.61	99
2020	4	December	Furniture	13390.5	125
2020	4	December	Furniture	3149.01	29
2020	4	December	Furniture	869382.04	73
2020	4	December	Furniture	3670.29	30
2020	4	December	Furniture	13514.95	109
2020	4	December	Furniture	5633.08	46
2020	4	December	Furniture	5424.67	30
2020	4	December	Furniture	2097.87	115
2020	4	December	Furniture	6303.41	29
2020	4	December	Furniture	6039.74	24
2020	4	December	Furniture	389.16	17
2020	4	December	Furniture	1198.92	47
2020	4	December	Furniture	28984.35	110
2020	4	December	Furniture	38557.66	127

Customer Health Monitor

Customer Segment	Customer Count	Churn Rate	Top Category	Customer Order Count	Last Purchase Date
Loyal	24	0.05	Technology	34	9/15/2021 12:00:00 AM
Loyal	12	0.02	Office Supplies	23	9/19/2021 12:00:00 AM
Loyal	20	0.04	Office Supplies	26	6/29/2021 12:00:00 AM
Loyal	10	0.02	Office Supplies	12	7/22/2021 12:00:00 AM
Loyal	16	0.03	Office Supplies	27	9/15/2021 12:00:00 AM

Promotion Effectiveness

Coupon Code	Discount Efficiency	Final Total Price	Discount Efficiency Indicator
A11EYCKTG465YJ	21.43	1933.45	High
A11YIDLENN53FX	110.06	16612.76	High
A121IMNDKBYC2	58.33	6818.57	High
A13PWRAEW1WMG6	4.77	356.88	Medium
A165WEMMCLK24J	129.77	9503.11	High
A16ARWY4RXPLWY	18.97	1292.22	High

Web Application

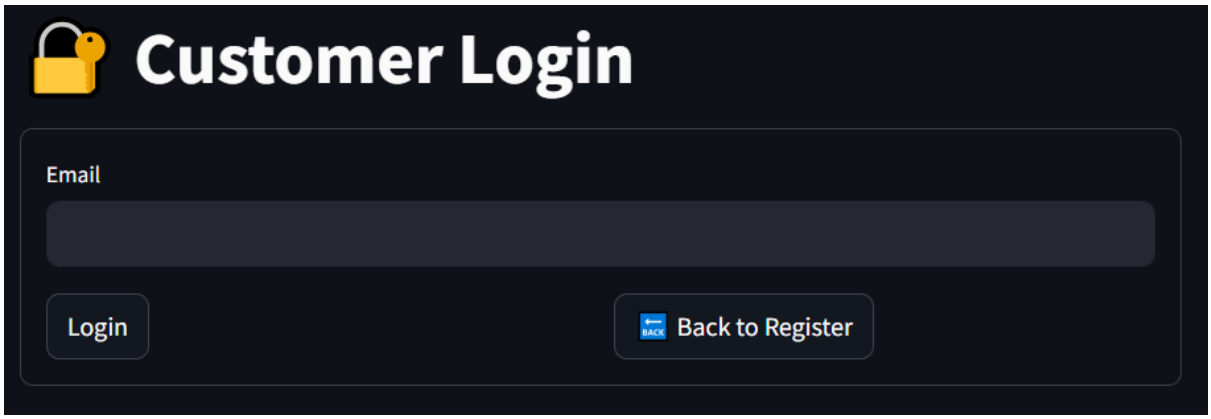
An interactive web application was developed using Streamlit to serve as the customer-facing panel of the Online Retail Management System. The application provides a user-friendly interface for browsing products, placing orders, and managing customer interactions.

Purpose

The web app bridges the gap between customers and the backend system, offering an intuitive shopping experience while capturing real-time transactional data directly into the SQL Server database.

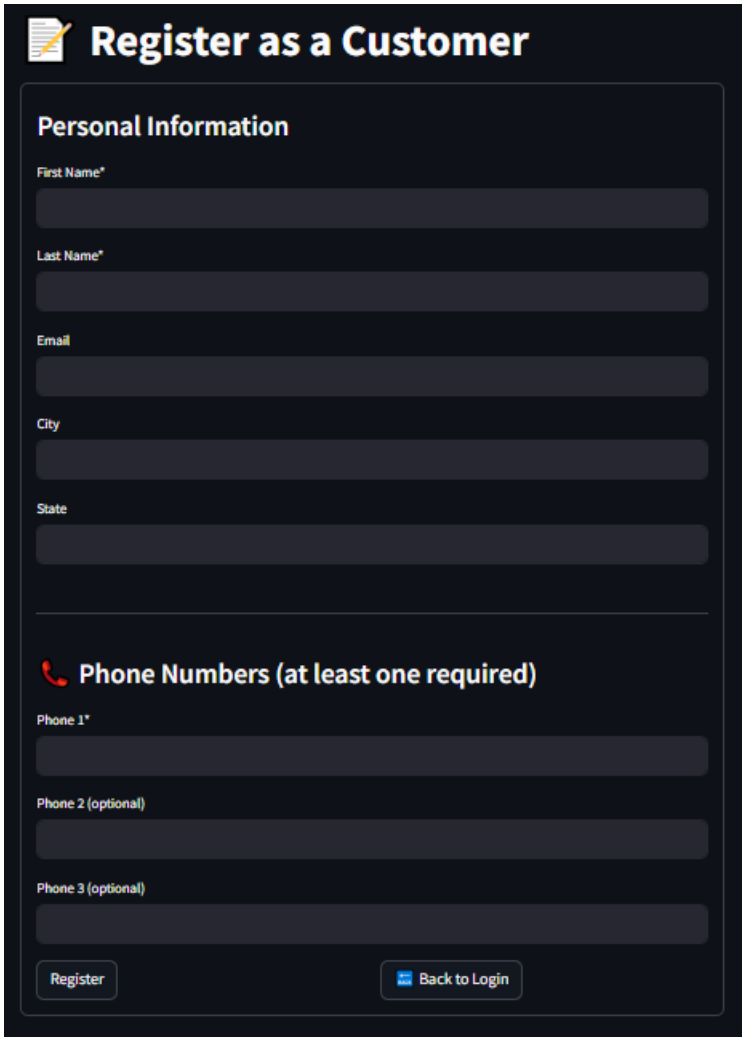
Key Features

- Customer Registration & Login:



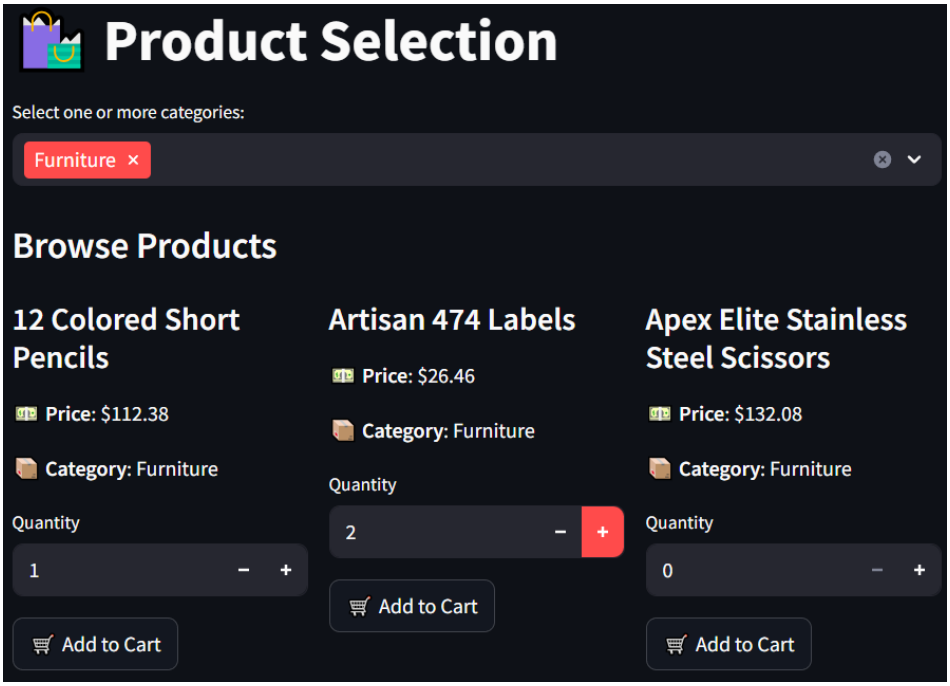
When a user enters their email address, the backend system verifies whether it already exists in the database to prevent duplicate registrations and ensure account integrity.

During registration, the customer is required to enter their first name, last name, and at least one phone number. The backend system then inserts a new record into the Customers table and stores the phone number in the Customer_Phones table.

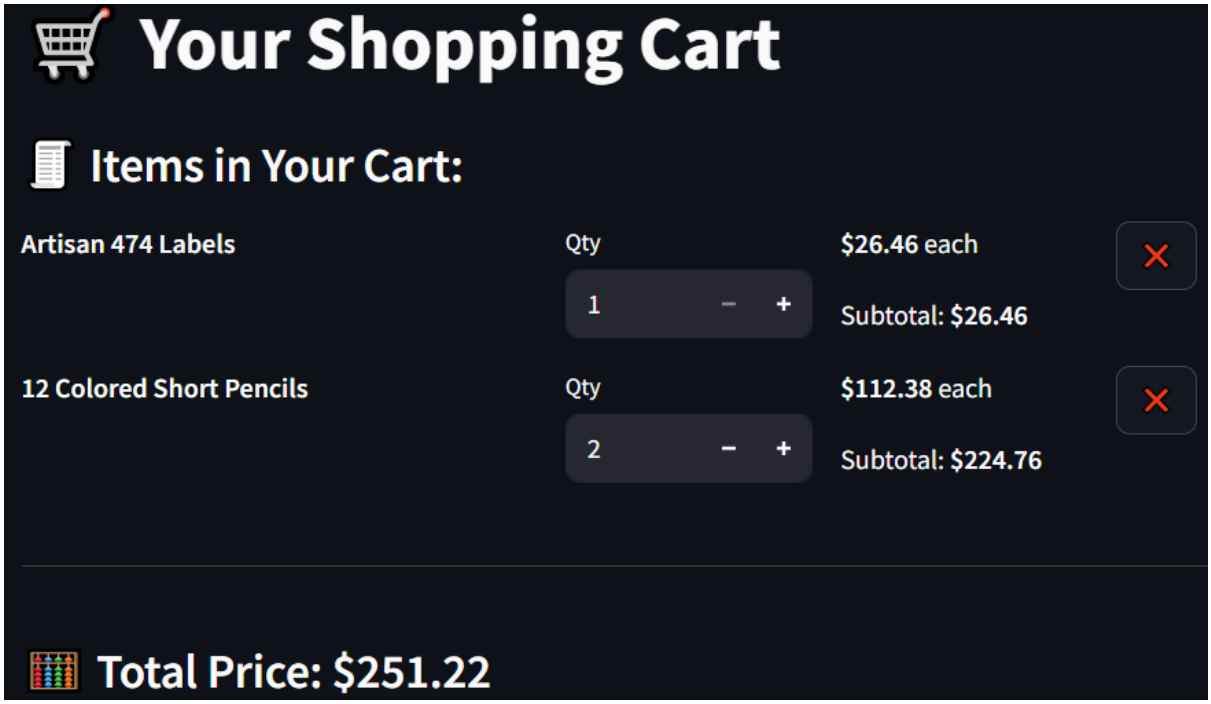


- Product Catalog Browsing:

Customers can select product categories and specify the desired quantities for each item.

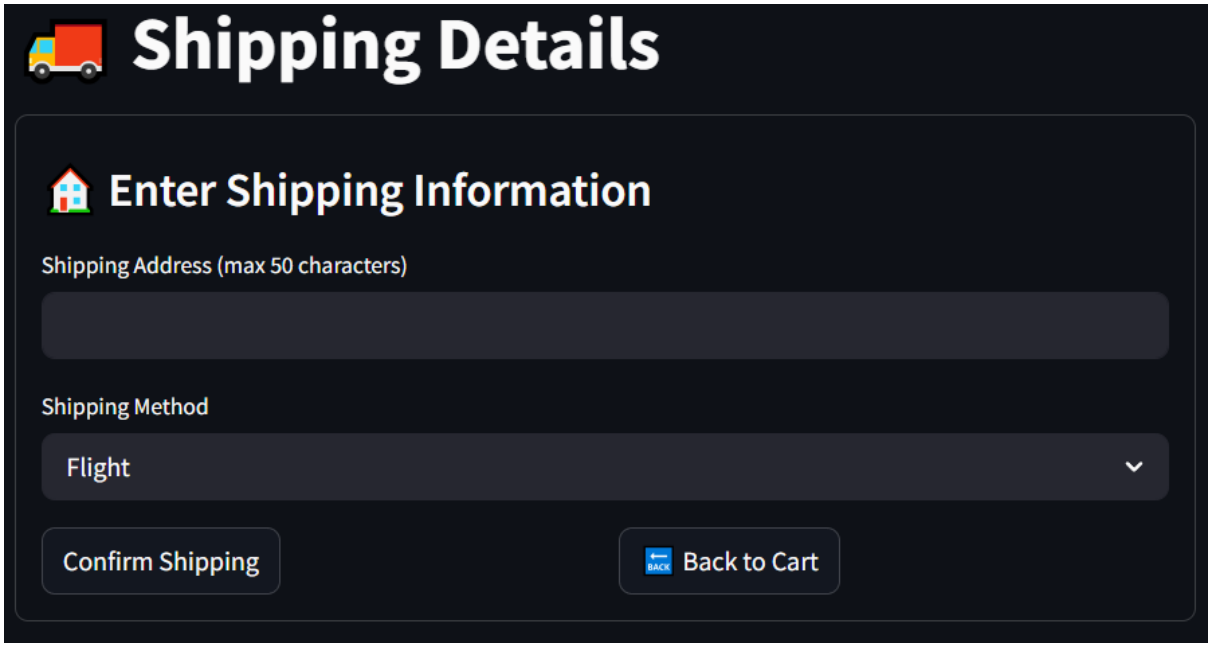


- Cart System:

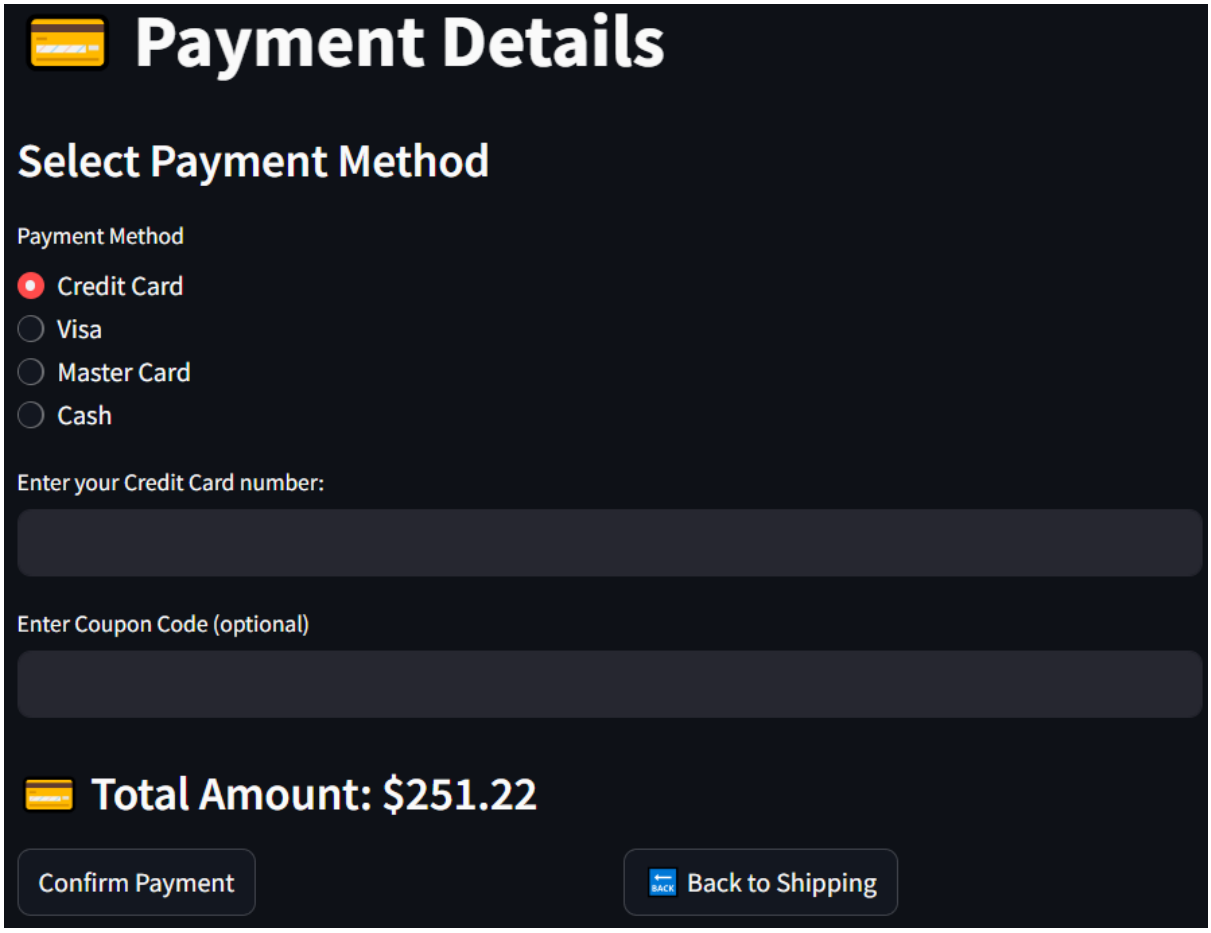


Customers can remove products from their cart or adjust quantities as needed, with real-time updates to the subtotal and total amounts displayed.

- Shipping & Payment:



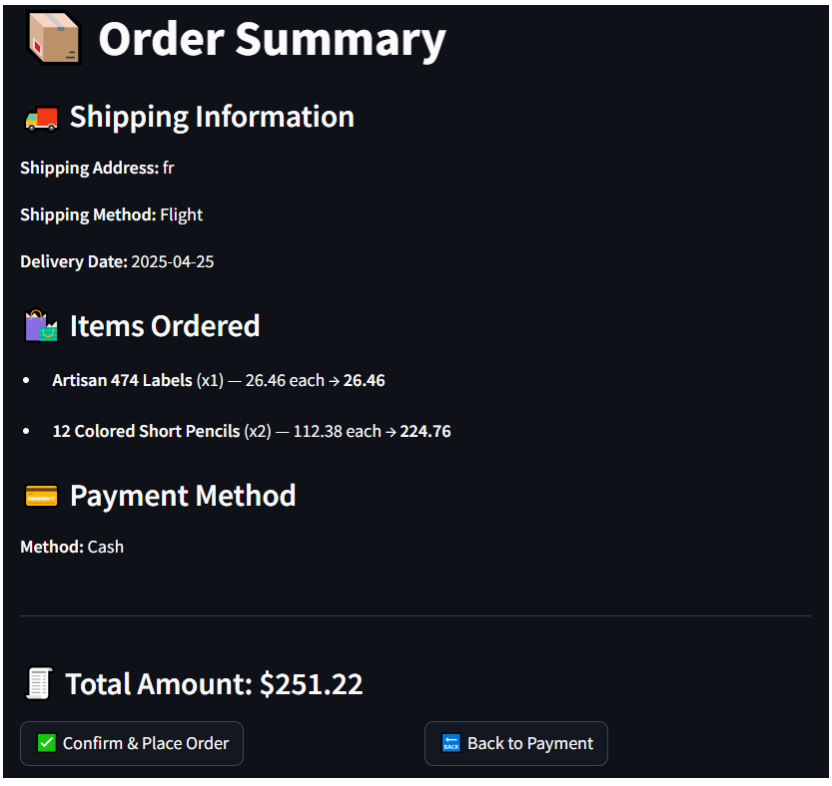
Customers provide their shipping address and select a preferred shipping method during checkout.



Customers can choose a payment method and apply a coupon code if available.

- Order Summary:

The shipping details, items ordered, selected payment method, and total amount paid are displayed for customer confirmation.



Database Integration

The app connects to the SQL Server database and uses stored procedures to:

- Insert customer data

```
CREATE PROCEDURE InsertCustomerWithPhone
@FName NVARCHAR(20),
@LName NVARCHAR(20),
@Email NVARCHAR(50),
@City NVARCHAR(20),
@State NVARCHAR(20)
AS
BEGIN
    INSERT INTO Customers (F_Name, L_Name, email, city, state, Account_Created_At)
    VALUES (@FName, @LName, @Email, @City, @State, GETDATE());

END;
```

```
CREATE PROCEDURE AddCustomerPhones
    @Phone1 VARCHAR(50),
    @Phone2 VARCHAR(50),
    @Phone3 VARCHAR(50)
AS
BEGIN

    DECLARE @Customer_ID INT;

    SELECT
        TOP 1 @Customer_ID = Customer_ID
    FROM
        Customers
    ORDER BY
        Customer_ID DESC;

    IF @Phone1 IS NOT NULL
        INSERT INTO Customer_Phones (Customer_ID, Phone) VALUES (@Customer_ID, @Phone1);

    IF @Phone2 IS NOT NULL
        INSERT INTO Customer_Phones (Customer_ID, Phone) VALUES (@Customer_ID, @Phone2);

    IF @Phone3 IS NOT NULL
        INSERT INTO Customer_Phones (Customer_ID, Phone) VALUES (@Customer_ID, @Phone3);
END;
```

- Check if customer email exists in the database or not

```
CREATE PROCEDURE ValidateCustomerLogin
@Email VARCHAR(50)
AS
BEGIN
    SELECT F_Name, L_Name
    FROM Customers
    WHERE Email = @Email
END;
```

- Filter products based on customer selection of categories

```
CREATE PROCEDURE GetProductsByCategory
@CategoryName VARCHAR(MAX)
AS
BEGIN
    SELECT
        P.Product_ID,
        P.Name,
        P.Price,
        P.Stock,
        C.Name AS Category
    FROM
        Products P
    JOIN
        Product_category C
    ON
        P.Category_ID = C.Category_ID
    WHERE
        C.Name IN (SELECT VALUE FROM string_split(@CategoryName, ','))
END;
```

- Insert shipping details

```
CREATE PROCEDURE InsertShippingDetails
@Track_Number SMALLINT,
@Delivery_Date DATE,
@Shipping_Method VARCHAR(50),
@Shipping_Address VARCHAR(50)
AS
BEGIN
    INSERT INTO Shipping (Track_Number, Delivery_Date, Shipping_Method, Shipping_Address)
    VALUES (@Track_Number, @Delivery_Date, @Shipping_Method, @Shipping_Address)
END;
```

- Check if coupon is valid or not

```
CREATE PROCEDURE CheckCouponValidity
    @Coupon_Code VARCHAR(50)
AS
BEGIN
    -- Return discount if valid
    SELECT
        Discount_Value
    FROM
        Coupons
    WHERE
        Coupon_Code = @Coupon_Code
        AND Expiry_Date >= CAST(GETDATE() AS DATE);
END;
```

- Insert order and payment details

```
CREATE PROCEDURE InsertOrder
    @Coupon_ID INT = NULL,
    @Payment_Amount SMALLINT,
    @Payment_Method VARCHAR(50)
AS
BEGIN
    DECLARE @Customer_ID INT;

    SELECT
        TOP 1 @Customer_ID = Customer_ID
    FROM
        Customers
    ORDER BY
        Customer_ID DESC;

    DECLARE @Shipping_ID INT;

    SELECT
        TOP 1 @Shipping_ID = Shipping_ID
    FROM
        Shipping
    ORDER BY
        Shipping_ID DESC;

    DECLARE @Payment_ID INT

    -- Get the max current Payment_ID and add 1, or start from 1 if table is empty
    SELECT @Payment_ID = ISNULL(MAX(Payment_ID), 0) + 1 FROM Orders

    INSERT INTO Orders (Status, Order_date, Customer_ID, Shipping_ID, Coupon_ID, Payment_ID, Payment_Amount, Payment_Method, Payment_Date)
    VALUES ('Confirmed', GETDATE(), @Customer_ID, @Shipping_ID, @Coupon_ID, @Payment_ID, @Payment_Amount, @Payment_Method, GETDATE())
END;
```

Conclusion & Future Enhancements

This project provides a scalable and data-driven system suitable for small to medium online retail businesses. Future improvements may include:

- Integration with real-time payment gateways.
- Admin panel for order, product, and coupon management.
- Advanced machine learning models for recommendation systems.