E-Commerce Management System

Group 9:

Jieyu Zheng

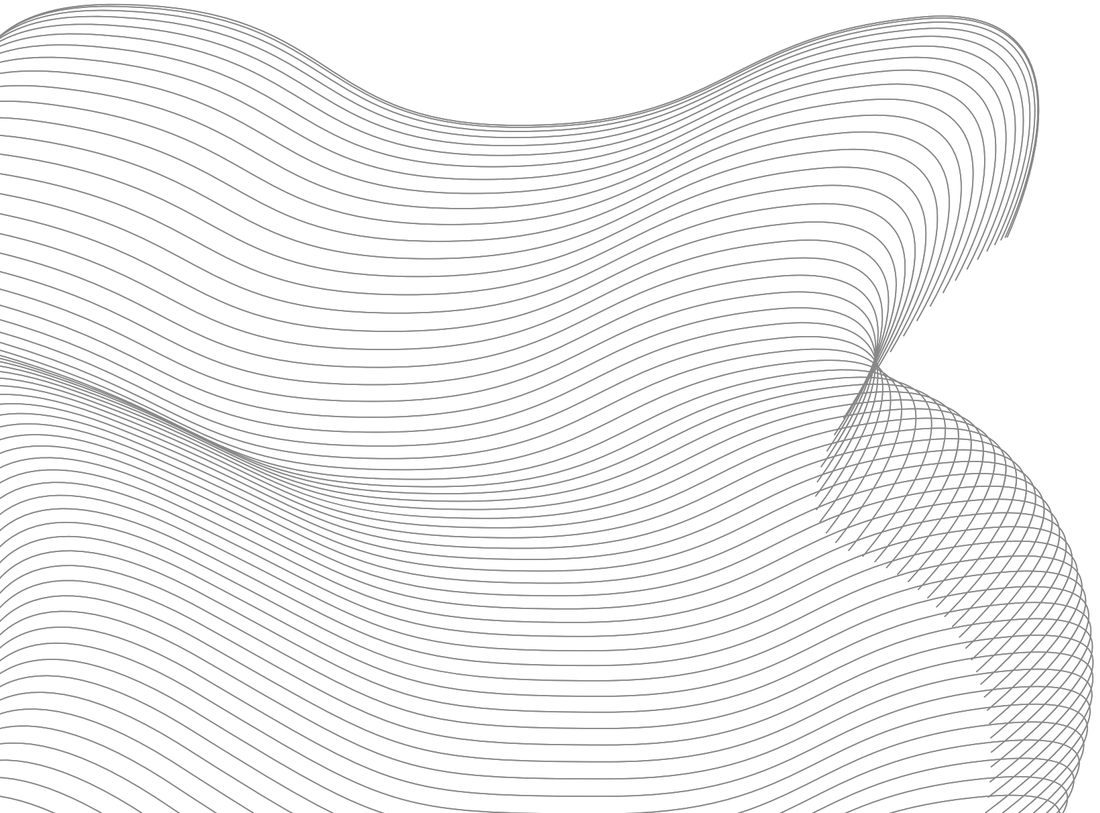
Yuan-Chang Lee

Jiaqi Yu

Rahul Reddy

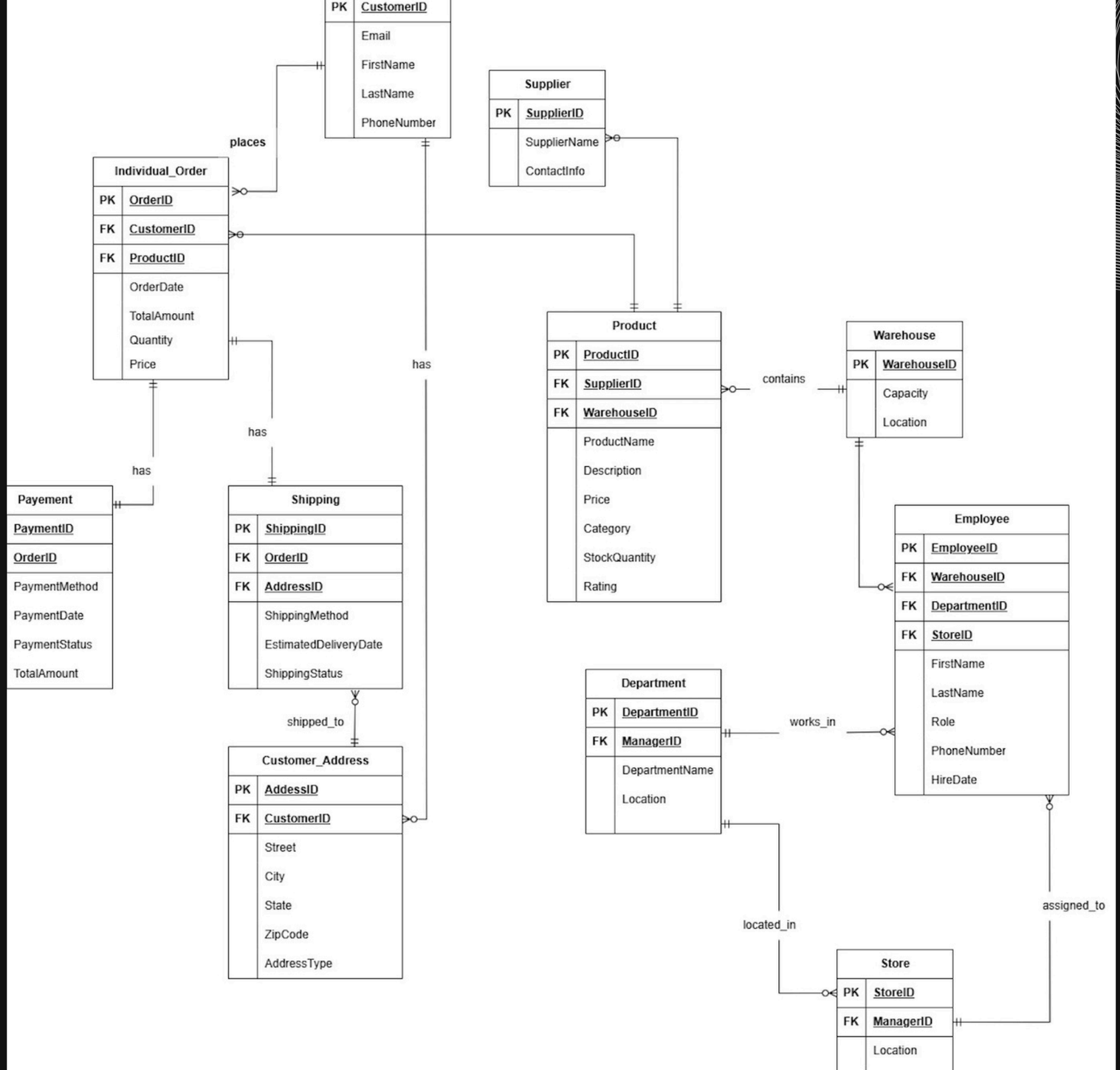
Madhumitha Nandhikatti

Business Probloms Addressed



- **01** Keep customer data organized so you can tailor your marketing just for them.
- **02** Track inventory in real-time to make sure products are always available.
- **03** Support clear roles for employees and good communication between departments.
- **04** Make order processing smooth from start to finish to boost accuracy.
- **05** Ensure secure, multi-option payment handling and accurate transaction tracking
- **06** Improve shipping by working closely with delivery partners.

ER-Diagram


PK CustomerID
Email
FirstName
LastName
PhoneNumber
places
Supplier
PK SupplierID
SupplierName
ContactInfo
Individual_Order
PK OrderID
FK CustomerID
FK ProductID
OrderDate
TotalAmount
Quantity
Price
has
has
has
Product
PK ProductID
FK SupplierID
FK WarehouseID
ProductName
Description
Price
Category
StockQuantity
Rating
contains
Warehouse
PK WarehouseID
Capacity
Location
Payement
PaymentID
OrderID
PaymentMethod
PaymentDate
PaymentStatus
TotalAmount
Shipping
PK ShippingID
FK OrderID
FK AddressID
ShippingMethod
EstimatedDeliveryDate
ShippingStatus
shipped_to
Customer_Address
PK AddessID
FK CustomerID
Street
City
State
ZipCode
AddressType
Employee
PK EmployeeID
FK WarehouseID
FK DepartmentID
FK StoreID
FirstName
LastName
Role
PhoneNumber
HireDate
Department
PK DepartmentID
FK ManagerID
DepartmentName
Location
works_in
located_in
assigned_to
Store
PK StoreID
FK ManagerID
Location

Procedure

```sql
-- Stored Procedure 1: Get total amount for a customer
ALTER PROCEDURE usp_GetCustomerTotalAmount
    @CustomerID INT,
    @TotalAmount DECIMAL(10,2) OUTPUT,
    @message NVARCHAR(255) OUTPUT
AS
BEGIN
    BEGIN TRY
        -- Check if the CustomerID exists
        IF NOT EXISTS (SELECT 1 FROM Individual_Order WHERE CustomerID = @CustomerID)
        BEGIN
            -- Handle non-existent customer
            SET @message = 'Error: Customer ID ' + CAST(@CustomerID AS NVARCHAR) + ' do
            SET @TotalAmount = 0;
            RETURN;
        END
        -- Check if the CustomerID is negative
        IF @CustomerID < 0
        BEGIN
            -- Handle negative CustomerID
            SET @message = 'Error: Invalid Customer ID. Customer ID cannot be negative.
            SET @TotalAmount = 0;
            RETURN;
        END
        BEGIN TRANSACTION;
        SELECT @TotalAmount = SUM(TotalAmount)/0
        FROM Individual_Order
        WHERE CustomerID = @CustomerID;
        -- Check if @TotalAmount is NULL (no orders found)
        IF @TotalAmount IS NULL
        BEGIN
            SET @TotalAmount = 0;
            SET @message = 'Customer has no orders.';
        END
        ELSE
        BEGIN
            SET @message = 'Success: Total amount calculated.';
        END
        COMMIT;
    END TRY
    BEGIN CATCH
        IF @@TRANCOUNT > 0
            ROLLBACK;
```

DDL

```sql
CREATE TABLE Customer (
    CustomerID INT IDENTITY(1,1) PRIMARY KEY,
    Email VARCHAR(255) UNIQUE NOT NULL,
    FirstName VARCHAR(100) NOT NULL,
    LastName VARCHAR(100) NOT NULL,
    PhoneNumber VARCHAR(20) NOT NULL CHECK (LEN(PhoneNumber) >= 10)
);
```

View

```sql
--View
-- View 1: Customer basic information with address
CREATE VIEW vw_CustomerFullInfo AS
SELECT
    c.CustomerID,
    c.FirstName + ' ' + c.LastName AS FullName,
    c.Email,
    a.Street,
    a.City,
    a.State,
    a.ZipCode
FROM Customer c
JOIN Customer_Address a ON c.CustomerID = a.CustomerID;
GO

-- Run View 1
SELECT * FROM vw_CustomerFullInfo;
```

Trigger

```sql
-- DML Trigger
-- Customer encryption Trigger: Encrypt PhoneNumber on Customer
CREATE TRIGGER EncryptPhoneNumberTrigger
ON Customer
AFTER INSERT, UPDATE
AS
BEGIN
    SET NOCOUNT ON;
    OPEN SYMMETRIC KEY DAMGSymmetricKey DECRYPTION BY CERTIFICATE DAMGCertificate;

    UPDATE Customer
    SET EncryptedPhoneNumber = ENCRYPTBYKEY(KEY_GUID('DAMGSymmetricKey'), PhoneNumber)
    WHERE CustomerID IN (SELECT CustomerID FROM inserted);

    CLOSE SYMMETRIC KEY DAMGSymmetricKey;
END;
GO

-- Individual_Order Trigger: Recalculate TotalAmount on Individual_Order
CREATE TRIGGER UpdateTotalAmountTrigger
ON Individual_Order
AFTER INSERT, UPDATE
AS
BEGIN
    SET NOCOUNT ON;
    UPDATE IO
    SET TotalAmount = I.Price * I.Quantity
    FROM Individual_Order AS IO
    INNER JOIN inserted AS I ON IO.OrderID = I.OrderID;
END;
GO
```

Non-clustered index

```sql
-- Non-clustered index: Customer.LastName
CREATE NONCLUSTERED INDEX idx_Customer_LastName ON Customer(LastName);
-- Non-clustered index: Product.Category
CREATE NONCLUSTERED INDEX idx_Product_Category ON Product(Category);
-- Non-clustered index: Individual_Order.OrderDate
CREATE NONCLUSTERED INDEX idx_IndividualOrder_OrderDate ON Individual_Order(OrderDate);


-- Testing non-clustered indexes
SELECT * FROM Customer WHERE LastName = 'Smith';
SELECT * FROM Product WHERE Category = 'Electronics';
SELECT * FROM Individual_Order WHERE OrderDate = '2024-03-08';
```

Encryption

```sql
-- Create a master key
CREATE MASTER KEY ENCRYPTION BY PASSWORD = 'DAMG6210Group9';
GO
-- Create a certificate
CREATE CERTIFICATE DAMGCertificate
WITH SUBJECT = 'Certificate for column encryption';
GO
-- Create a symmetric key
CREATE SYMMETRIC KEY DAMGSymmetricKey
WITH ALGORITHM = AES_256
ENCRYPTION BY CERTIFICATE DAMGCertificate;
GO


-- Modify the Customer table to add a new column for storing encrypted phone numbers
ALTER TABLE Customer
ADD EncryptedPhoneNumber VARBINARY(MAX);
GO
-- need to create encryption trigger first
UPDATE Customer
SET EncryptedPhoneNumber =
    EncryptByKey(KEY_GUID('DAMGSymmetricKey'), CAST(PhoneNumber AS NVARCHAR(20)));
GO


-- Query the results
SELECT
    CustomerID,
    Email,
    EncryptedPhoneNumber
FROM Customer;
GO

OPEN SYMMETRIC KEY DAMGSymmetricKey DECRYPTION BY CERTIFICATE DAMGCertificate;
SELECT
    CustomerID,
    Email,
    CONVERT(VARCHAR(20), DecryptByKey(EncryptedPhoneNumber)) AS DecryptedPhoneNumber
FROM Customer;
CLOSE SYMMETRIC KEY DAMGSymmetricKey;
```

DATA POPULATION

```sql
-- Insert into Product (40 records with diverse categories)
INSERT INTO Product (SupplierID, WarehouseID, ProductName, Description, Price, Category, StockQuantity, Rating) VALUES
-- Electronics
(1, 1, 'Premium Laptop', 'High-end gaming laptop', 1499.99, 'Electronics', 50, 4.7),
(2, 2, 'Smartphone Pro', 'Latest model smartphone', 999.99, 'Electronics', 100, 4.5),
(11, 3, 'Wireless Earbuds', 'Noise cancelling earbuds', 149.99, 'Electronics', 200, 4.3),
(1, 4, 'Smart TV 55"', '4K Ultra HD Smart TV', 699.99, 'Electronics', 30, 4.6),
(11, 5, 'Digital Camera', 'Professional DSLR camera', 899.99, 'Electronics', 25, 4.8),
(2, 6, 'Smart Watch', 'Fitness and health tracker', 249.99, 'Electronics', 75, 4.4),
(1, 7, 'Bluetooth Speaker', 'Waterproof portable speaker', 89.99, 'Electronics', 120, 4.2),
(11, 8, 'Tablet Pro', '10-inch tablet with stylus', 449.99, 'Electronics', 50, 4.5),

-- Home Appliances
(3, 9, 'Blender Pro', 'High-speed kitchen blender', 129.99, 'Home Appliances', 40, 4.3),
(12, 10, 'Coffee Maker', 'Programmable coffee machine', 79.99, 'Home Appliances', 60, 4.1),
(3, 11, 'Robot Vacuum', 'Smart robot vacuum cleaner', 299.99, 'Home Appliances', 35, 4.7),
(12, 12, 'Toaster Oven', 'Convection toaster oven', 119.99, 'Home Appliances', 45, 4.0),

-- Furniture
(4, 13, 'Leather Sofa', 'Luxury leather sofa', 1299.99, 'Furniture', 15, 4.8),
(4, 14, 'Dining Table Set', 'Wooden dining table with 6 chairs', 799.99, 'Furniture', 20, 4.6),
(4, 15, 'Bookshelf', 'Modern 5-tier bookshelf', 189.99, 'Furniture', 30, 4.3),
(14, 16, 'Office Desk', 'Ergonomic office desk', 249.99, 'Furniture', 25, 4.5),

-- Automotive
(5, 17, 'Car Tires Set', 'All-season tire set of 4', 399.99, 'Automotive', 30, 4.4),
```

Total Revenue
84.96K

Orders
100

Avg Order Value
849.60

Customers
20

Order Date
05-01-2020 25-03-2024

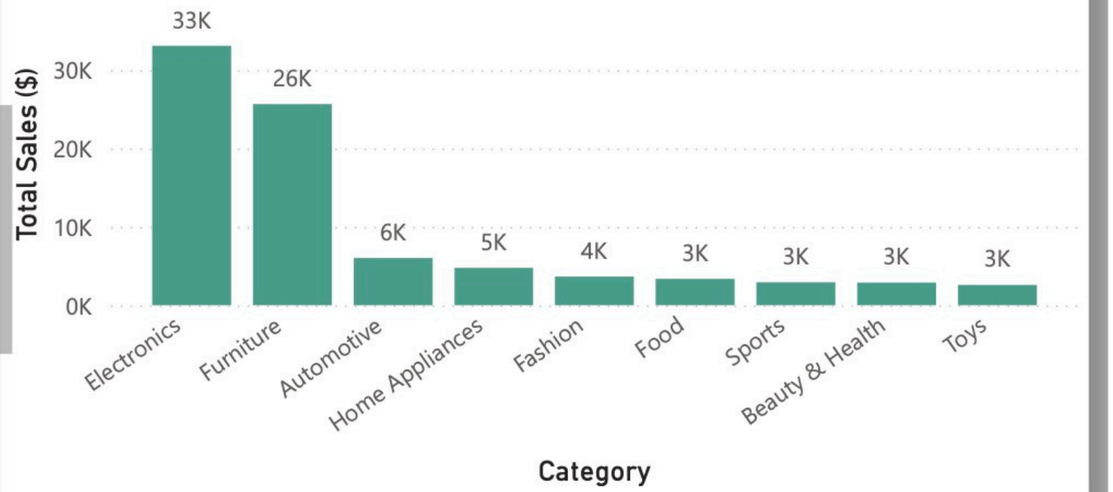
Location
All

Category
All

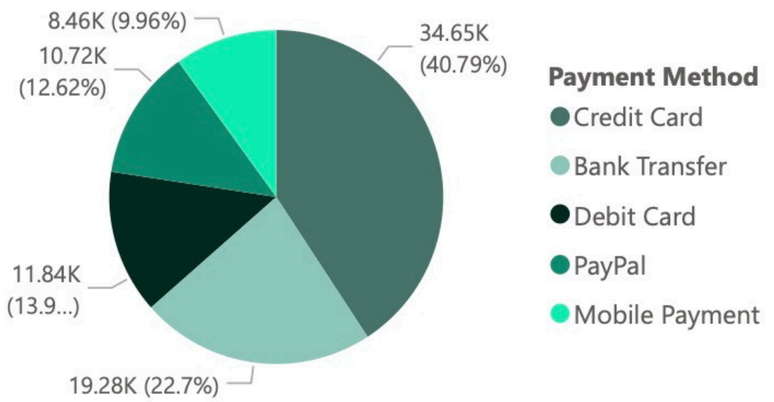
Payment Method
- [ ] Bank Transfer
- [ ] Credit Card
- [ ] Debit Card
- [ ] Mobile Payment
- [ ] PayPal

Payment Status
- [ ] Cancelled
- [ ] Delivered
- [ ] Pending
- [ ] Shipped

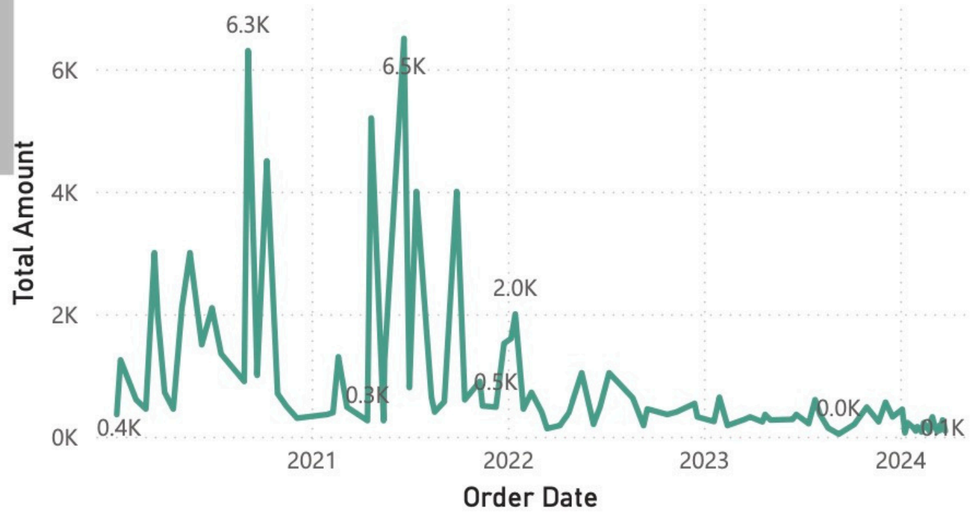
Sales by Category



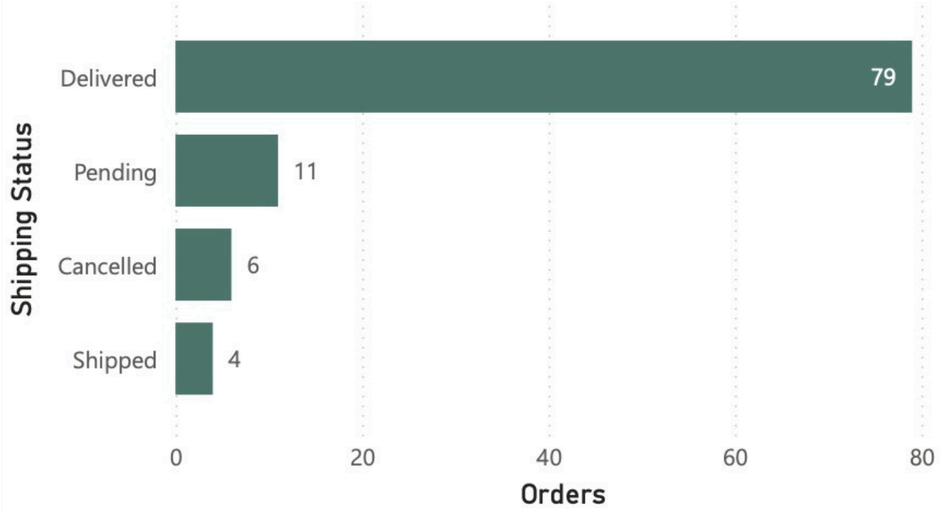
Payment Method Distribution



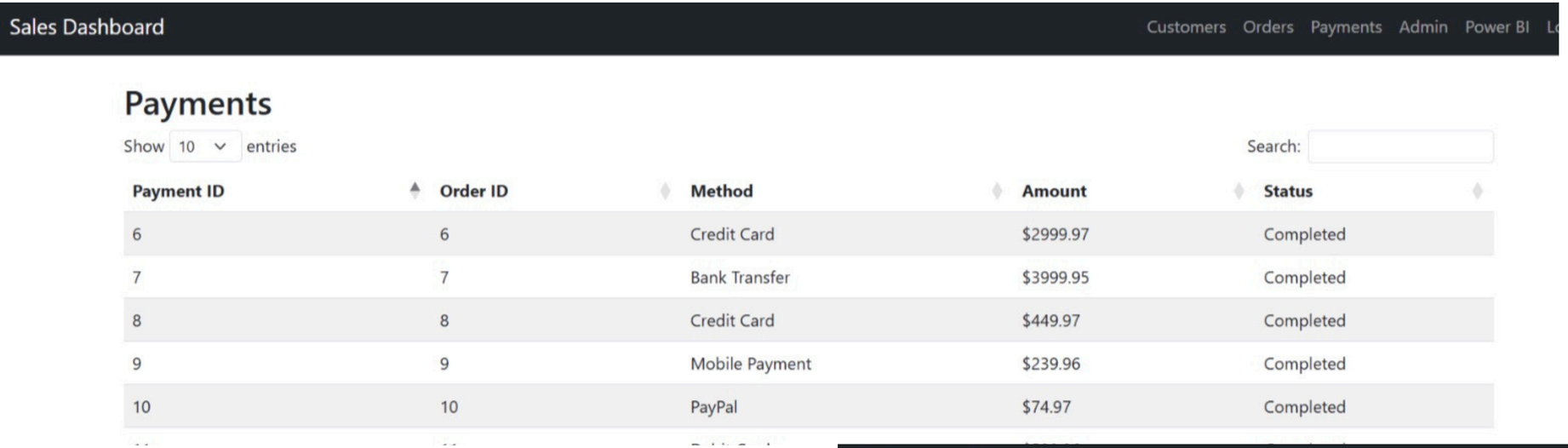
Daily Sales Trend

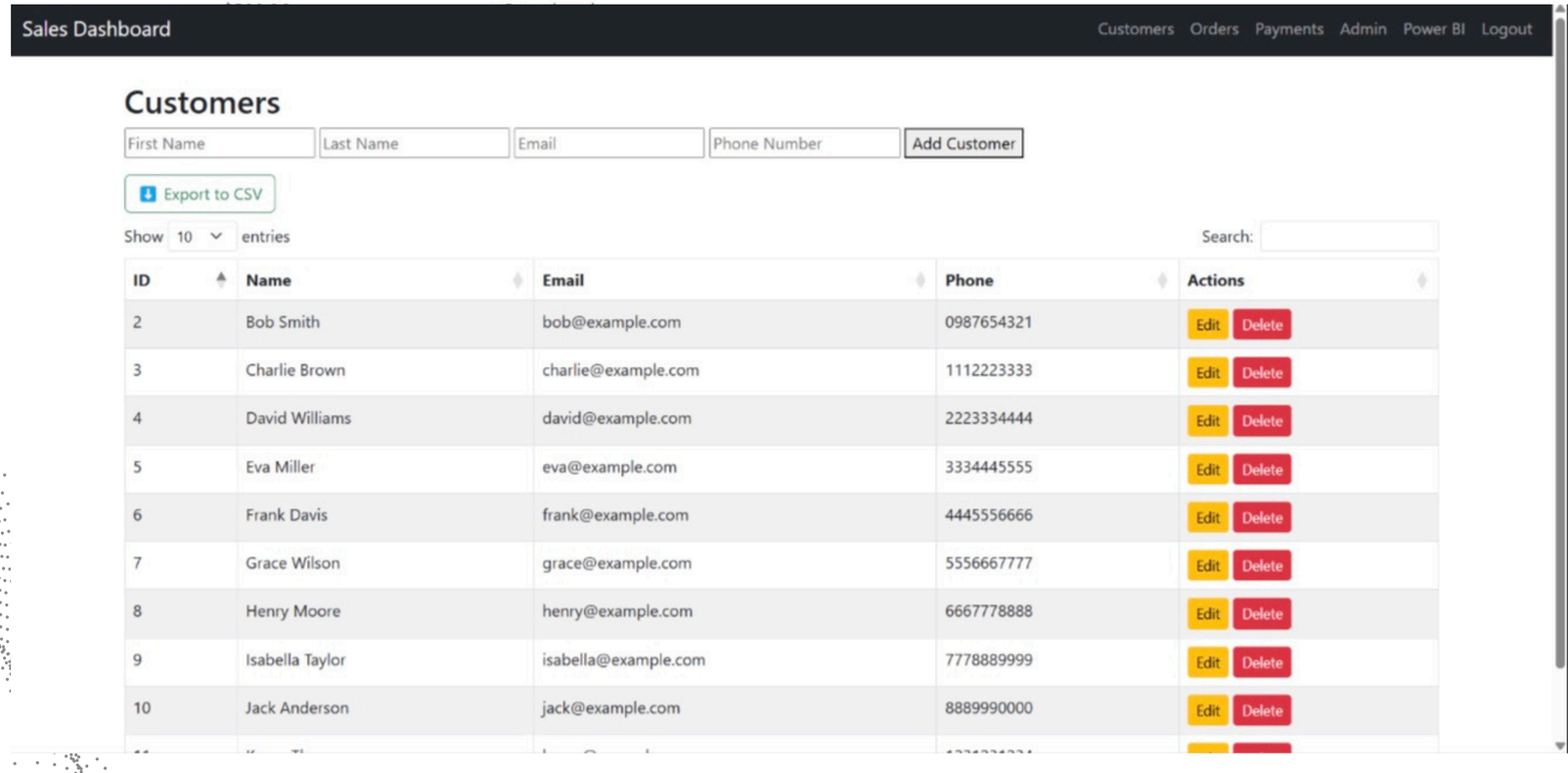


Shipping Status



GUI


Sales Dashboard
Customers Orders Payments Admin Power BI
Payments
Show 10 entries
Search:
Payment ID
Order ID
Method
Amount
Status
6 6 Credit Card $2999.97 Completed
7 7 Bank Transfer $3999.95 Completed
8 8 Credit Card $449.97 Completed
9 9 Mobile Payment $239.96 Completed
10 10 PayPal $74.97 Completed


Sales Dashboard
Customers Orders Payments Admin Power BI Logout
Customers
First Name
Last Name
Email
Phone Number
Add Customer
Export to CSV
Show 10 entries
Search:
ID
Name
Email
Phone
Actions
2 Bob Smith bob@example.com 0987654321 Edit Delete
3 Charlie Brown charlie@example.com 1112223333 Edit Delete
4 David Williams david@example.com 2223334444 Edit Delete
5 Eva Miller eva@example.com 3334445555 Edit Delete
6 Frank Davis frank@example.com 4445556666 Edit Delete
7 Grace Wilson grace@example.com 5556667777 Edit Delete
8 Henry Moore henry@example.com 6667778888 Edit Delete
9 Isabella Taylor isabella@example.com 7778889999 Edit Delete
10 Jack Anderson jack@example.com 8889990000 Edit Delete

Thank you!