**POS & Northwind Business Simulation – Requirements Document**

**1. Project Overview**

**Purpose:**  
Simulate a wholesale/retail trading company (like Northwind Traders) to manage products, inventory, customers, suppliers, orders, and employees.

**Target Platform:**

* Optional: Database: SQL Server

**Core Modules:**

1. Product & Inventory Management
2. Customer Management
3. Supplier Management
4. Sales / Orders Management
5. Employees & Territories
6. Shippers / Logistics
7. Reporting & Analytics

**2. Functional Requirements**

**2.1 Products & Inventory**

* Create, read, update, delete (CRUD) products
* Product attributes: ID, Name, Category, Supplier, Price, Stock Quantity
* Categories management: CRUD categories
* Suppliers management: CRUD suppliers
* Stock management:
  + Increase stock (restocking)
  + Reduce stock (sales)
* Alerts for low stock

**2.2 Customers**

* CRUD customer information
* Customer attributes: ID, Company Name, Contact Name, Address, City, Country
* Assign orders to customers

**2.3 Suppliers**

* CRUD supplier information
* Supplier attributes: ID, Company Name, Contact Name, Address, City, Country
* Link products to suppliers

**2.4 Employees & Territories**

* CRUD employee information
* Employee attributes: ID, Name, Title, Territory
* CRUD territories
* Assign orders to employees based on territory

**2.5 Orders / Sales**

* Create orders:
  + Select customer
  + Add multiple products with quantities
  + Validate stock availability
* Complete order:
  + Reduce product stock
  + Assign shipper
* Order attributes: ID, Customer, Employee, Shipper, List of products, Total amount, Order date
* Track order history per customer

**2.6 Shippers / Logistics**

* CRUD shippers
* Shipper attributes: ID, Company Name, Contact Info
* Assign shippers to orders

**2.7 Reporting & Analytics**

* Top-selling products
* Sales per customer
* Sales per category
* Revenue per employee / territory
* Inventory reports (stock levels, low stock alerts)

**3. Non-Functional Requirements**

* **Performance:** Handle hundreds of products, customers, and orders efficiently
* **Usability:** User-friendly console/WPF interface with menus
* **Maintainability:** Modular code structure (Modules/Services for Products, Orders, Customers, etc.)
* **Persistence:** Optional database storage (SQL Server / SQLite)
* **Security:** Optional user authentication (Admin, Cashier roles)

**4. Technical Requirements**

* **Language:** C# (.NET 6 or .NET Framework 4.8)
* **Database:** SQL Server / SQLite (optional, can start with in-memory simulation)
* **Architecture:**
  + Layered design (Models → Services → UI)
  + Separate modules per business area (ProductModule, CustomerModule, OrderModule, etc.)
* **Libraries/Tools:**
  + Entity Framework Core (if using database)
  + LINQ for in-memory data manipulation
  + Optional: Newtonsoft.Json for saving/loading in-memory data

**5. UI / Interaction**

* **Console or WPF menus:**
  + Main Menu: Products, Customers, Suppliers, Employees, Orders, Reports, Exit
* Sub-menus for CRUD operations per module
* Order placement workflow:
  + Select customer
  + Select products + quantity
  + Assign employee
  + Assign shipper
  + Confirm order and update stock
* Reports menu:
  + Show top-selling products
  + Show sales per customer / category / employee
  + Show inventory status

**6. Optional Enhancements**

* Discounts / promotions per order
* Multi-currency support
* Authentication and role-based access (Admin, Cashier)
* Export reports to CSV/Excel
* GUI using WPF or Windows Forms

**7. Suggested Development Steps**

1. **Module 1 – Products & Inventory** (CRUD, stock management)
2. **Module 2 – Customers** (CRUD)
3. **Module 3 – Suppliers** (CRUD, product link)
4. **Module 4 – Employees & Territories** (CRUD)
5. **Module 5 – Shippers** (CRUD)
6. **Module 6 – Orders / Sales** (workflow: select customer → products → employee → shipper → complete order)
7. **Module 7 – Reporting & Analytics** (LINQ queries, revenue calculation, top-selling products)
8. **Module 8 – Optional Enhancements** (authentication, GUI, export reports)

## ****1️⃣ Categories Table****

CREATE TABLE Categories (

CategoryId INT IDENTITY PRIMARY KEY,

CategoryName NVARCHAR(100) NOT NULL

);

## ****2️⃣ Suppliers Table****

CREATE TABLE Suppliers (

SupplierId INT IDENTITY PRIMARY KEY,

CompanyName NVARCHAR(150) NOT NULL,

ContactName NVARCHAR(100),

Address NVARCHAR(200),

City NVARCHAR(100),

Country NVARCHAR(50),

Phone NVARCHAR(50)

);

## ****3️⃣ Products Table****

CREATE TABLE Products (

ProductId INT IDENTITY PRIMARY KEY,

ProductName NVARCHAR(150) NOT NULL,

CategoryId INT NOT NULL,

SupplierId INT NOT NULL,

Price DECIMAL(18,2) NOT NULL,

Stock INT NOT NULL DEFAULT 0,

CONSTRAINT FK\_Product\_Category FOREIGN KEY (CategoryId) REFERENCES Categories(CategoryId),

CONSTRAINT FK\_Product\_Supplier FOREIGN KEY (SupplierId) REFERENCES Suppliers(SupplierId)

);

## ****4️⃣ Customers Table****

CREATE TABLE Customers (

CustomerId INT IDENTITY PRIMARY KEY,

CompanyName NVARCHAR(150) NOT NULL,

ContactName NVARCHAR(100),

Address NVARCHAR(200),

City NVARCHAR(100),

Country NVARCHAR(50),

Phone NVARCHAR(50)

);

## ****5️⃣ Employees Table****

CREATE TABLE Employees (

EmployeeId INT IDENTITY PRIMARY KEY,

FirstName NVARCHAR(50) NOT NULL,

LastName NVARCHAR(50) NOT NULL,

Title NVARCHAR(100),

HireDate DATETIME DEFAULT GETDATE()

);

## ****6️⃣ Territories Table****

CREATE TABLE Territories (

TerritoryId INT IDENTITY PRIMARY KEY,

TerritoryName NVARCHAR(100) NOT NULL

);

### ****EmployeeTerritories (Many-to-Many)****

CREATE TABLE EmployeeTerritories (

EmployeeId INT NOT NULL,

TerritoryId INT NOT NULL,

PRIMARY KEY(EmployeeId, TerritoryId),

CONSTRAINT FK\_EmployeeTerritory\_Employee FOREIGN KEY(EmployeeId) REFERENCES Employees(EmployeeId),

CONSTRAINT FK\_EmployeeTerritory\_Territory FOREIGN KEY(TerritoryId) REFERENCES Territories(TerritoryId)

);

## ****7️⃣ Shippers Table****

CREATE TABLE Shippers (

ShipperId INT IDENTITY PRIMARY KEY,

CompanyName NVARCHAR(150) NOT NULL,

Phone NVARCHAR(50)

);

## ****8️⃣ Orders Table****

CREATE TABLE Orders (

OrderId INT IDENTITY PRIMARY KEY,

CustomerId INT NOT NULL,

EmployeeId INT NOT NULL,

ShipperId INT NOT NULL,

OrderDate DATETIME DEFAULT GETDATE(),

CONSTRAINT FK\_Order\_Customer FOREIGN KEY(CustomerId) REFERENCES Customers(CustomerId),

CONSTRAINT FK\_Order\_Employee FOREIGN KEY(EmployeeId) REFERENCES Employees(EmployeeId),

CONSTRAINT FK\_Order\_Shipper FOREIGN KEY(ShipperId) REFERENCES Shippers(ShipperId)

);

### ****OrderItems Table****

CREATE TABLE OrderItems (

OrderItemId INT IDENTITY PRIMARY KEY,

OrderId INT NOT NULL,

ProductId INT NOT NULL,

Quantity INT NOT NULL,

UnitPrice DECIMAL(18,2) NOT NULL,

CONSTRAINT FK\_OrderItem\_Order FOREIGN KEY(OrderId) REFERENCES Orders(OrderId),

CONSTRAINT FK\_OrderItem\_Product FOREIGN KEY(ProductId) REFERENCES Products(ProductId)

);

## ****9️⃣ Purchases Table****

CREATE TABLE Purchases (

PurchaseId INT IDENTITY PRIMARY KEY,

SupplierId INT NOT NULL,

EmployeeId INT NOT NULL,

PurchaseDate DATETIME DEFAULT GETDATE(),

CONSTRAINT FK\_Purchase\_Supplier FOREIGN KEY(SupplierId) REFERENCES Suppliers(SupplierId),

CONSTRAINT FK\_Purchase\_Employee FOREIGN KEY(EmployeeId) REFERENCES Employees(EmployeeId)

);

### ****PurchaseItems Table****

CREATE TABLE PurchaseItems (

PurchaseItemId INT IDENTITY PRIMARY KEY,

PurchaseId INT NOT NULL,

ProductId INT NOT NULL,

Quantity INT NOT NULL,

UnitPrice DECIMAL(18,2) NOT NULL,

CONSTRAINT FK\_PurchaseItem\_Purchase FOREIGN KEY(PurchaseId) REFERENCES Purchases(PurchaseId),

CONSTRAINT FK\_PurchaseItem\_Product FOREIGN KEY(ProductId) REFERENCES Products(ProductId)

);

## ****10️⃣ CustomerReturns Table****

CREATE TABLE CustomerReturns (

CustomerReturnId INT IDENTITY PRIMARY KEY,

OrderId INT NOT NULL,

ReturnDate DATETIME DEFAULT GETDATE(),

CONSTRAINT FK\_CustomerReturn\_Order FOREIGN KEY(OrderId) REFERENCES Orders(OrderId)

);

### ****CustomerReturnItems Table****

CREATE TABLE CustomerReturnItems (

CustomerReturnItemId INT IDENTITY PRIMARY KEY,

CustomerReturnId INT NOT NULL,

ProductId INT NOT NULL,

Quantity INT NOT NULL,

RefundAmount DECIMAL(18,2) NOT NULL,

CONSTRAINT FK\_CustomerReturnItem\_Return FOREIGN KEY(CustomerReturnId) REFERENCES CustomerReturns(CustomerReturnId),

CONSTRAINT FK\_CustomerReturnItem\_Product FOREIGN KEY(ProductId) REFERENCES Products(ProductId)

);

## ****11️⃣ SupplierReturns Table****

CREATE TABLE SupplierReturns (

SupplierReturnId INT IDENTITY PRIMARY KEY,

SupplierId INT NOT NULL,

ReturnDate DATETIME DEFAULT GETDATE(),

CONSTRAINT FK\_SupplierReturn\_Supplier FOREIGN KEY(SupplierId) REFERENCES Suppliers(SupplierId)

);

### ****SupplierReturnItems Table****

CREATE TABLE SupplierReturnItems (

SupplierReturnItemId INT IDENTITY PRIMARY KEY,

SupplierReturnId INT NOT NULL,

ProductId INT NOT NULL,

Quantity INT NOT NULL,

CONSTRAINT FK\_SupplierReturnItem\_Return FOREIGN KEY(SupplierReturnId) REFERENCES SupplierReturns(SupplierReturnId),

CONSTRAINT FK\_SupplierReturnItem\_Product FOREIGN KEY(ProductId) REFERENCES Products(ProductId)

);