# Project Report: Project1-InventoryManagement

# 

**Submitted By** GROUP NO 12

Group Members:

Zain ul abideen Ali 233604 BSCS B

Arfa Imran 233596 BSCS B

Fatima Khalid 233526 BSCS B

**Submitted to Mam Aatka**

**DEPARTMENT OF COMPUTER SCIENCE**

**AIR UNIVERSITY MULTAN CAMPUS**

Table of Contents

[Project Report: Project1-InventoryManagement 1](#_Toc186967552)

[1. Evaluation 4](#_Toc186967554)

[2. Introduction 5](#_Toc186967555)

[2.1 Overview of the Project 5](#_Toc186967556)

[2.2 Problem Statement and Background 5](#_Toc186967557)

[2.3 Significance of the Problem 5](#_Toc186967558)

[3. Objectives 5](#_Toc186967559)

[3.1 Primary Goals 5](#_Toc186967560)

[3.2 Expected Outcomes 6](#_Toc186967561)

[4. Features 6](#_Toc186967562)

[4.1 Key Functionalities 6](#_Toc186967563)

[4.1.1 Admin Features: 6](#_Toc186967564)

[4.1.2 Manager Features: 6](#_Toc186967565)

[4.1.3 Staff Features: 6](#_Toc186967566)

[4.2 Innovative Aspects 7](#_Toc186967567)

[5. Business Logic 7](#_Toc186967568)

[5.1 Description of Core Logic 7](#_Toc186967569)

[5.2 Problem-Solving Approach 8](#_Toc186967570)

[6. Flow Chart 9](#_Toc186967571)

[6.1 Application Flow Representation 9](#_Toc186967572)

[6.2 Annotations and Explanation 10](#_Toc186967573)

[7. Class Diagram 11](#_Toc186967574)

[7.1 Overview of Class Structure 11](#_Toc186967575)

[7.2 Class Relationships 12](#_Toc186967576)

[7.3 Associations and Inheritance 12](#_Toc186967577)

[8. Implementation 13](#_Toc186967578)

[8.1 Development Process 13](#_Toc186967579)

[8.2 Screenshots of all User Interfaces 13](#_Toc186967580)

[Login Page: 13](#_Toc186967581)

[Admin Interfaces 14](#_Toc186967583)

[Manager Interfaces: 16](#_Toc186967586)

[Staff Interfaces 17](#_Toc186967594)

[8.3 Tools and Technologies Used 18](#_Toc186967596)

[8.4 Challenges and Solutions 18](#_Toc186967597)

[1. Challenge: User Authentication and Role-Based Access Control (RBAC) 18](#_Toc186967598)

[2. Challenge: Real-Time Data Synchronization 18](#_Toc186967599)

[3. Challenge: User Interface (UI) Design 19](#_Toc186967600)

[4. Challenge: Data Security 19](#_Toc186967601)

[9. Conclusion and Results 20](#_Toc186967603)

[9.1 Project Outcomes Summary 20](#_Toc186967604)

[9.2 Comparison of Objectives and Results 20](#_Toc186967605)

[9.3 Future Enhancements 21](#_Toc186967606)

[10. References 22](#_Toc186967608)

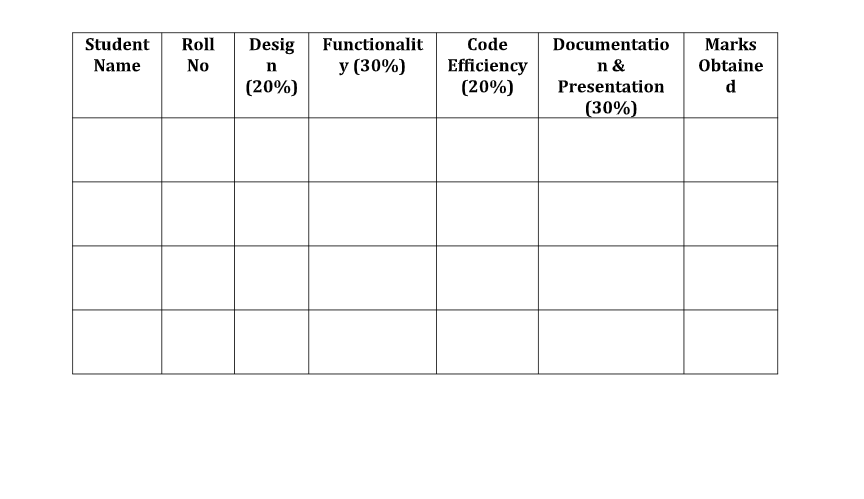
[10.1 Cited Resources 22](#_Toc186967609)

[10.2 Further Reading 22](#_Toc186967610)

[11. Appendices 23](#_Toc186967612)

[11.1 Code Snippets 23](#_Toc186967613)

# 1. Evaluation



# 2. Introduction

2.1 Overview of the Project  
The Inventory Management System (IMS) is designed to optimize inventory management tasks within an organization. The project focuses on developing a reliable database and an intuitive desktop application, empowering users to effectively handle suppliers, products, orders, and stock movements.

2.2 Problem Statement and Background  
Organizations often face challenges in maintaining accurate and efficient inventory records, leading to errors, delays, and increased operational costs. Existing manual or outdated systems may lack the capabilities to manage inventory data effectively. This project addresses these challenges by providing a comprehensive and automated solution for inventory management.

2.3 Significance of the Problem  
Efficient inventory management is critical to minimizing operational costs, avoiding stock shortages or surpluses, and improving overall productivity. By implementing a streamlined IMS, organizations can enhance decision-making, ensure data accuracy, and adapt to dynamic market demands. This system will bridge the gap between manual processes and modern technological solutions.

# 

# 3. Objectives

## ****3.1 Primary Goals****

* Develop a robust relational database to manage various inventory-related tasks efficiently.
* Create a WPF application with an intuitive interface for seamless user interaction with the database.
* Ensure data integrity and security through rigorous validation mechanisms and the implementation of user roles.

## ****3.2 Expected Outcomes****

* A fully functional Inventory Management System with a reliable backend database and user-friendly frontend.
* Enhanced user experience through an accessible and responsive WPF interface.
* Improved security and accuracy in inventory data management, reducing errors and unauthorized access.

# 4. Features

## ****Key Functionalities****

### ****4.1.1 Admin Features****:

* + **User Management**: Add, update, and delete user accounts while assigning roles.
  + **Product Management**: Manage products by adding, updating, and deleting items in the inventory.
  + **Supplier Management**: Add new suppliers, update existing supplier details, and manage supplier relationships.
  + **Role-Based Access Control**: Enhance security by defining permissions for different user roles.

### ****Manager Features****:

**Monitor Staff Activities**: The manager ensures check and balance on staff by overseeing their activities, performance, and adherence to organizational standards.

### ****Staff Features****:

* + **Basic Inventory Management**: View details of products, suppliers, and stock movements.
  + **Stock Movement Recording**: Input and update stock movement data accurately.
  + **Sales Order Processing**: Process customer sales orders and update their statuses accordingly.

## ****4.2 Innovative Aspects****

* User-centric design with a WPF application for seamless navigation.
* Advanced role-based access control for heightened security.
* Forecasting tools to support strategic decision-making.

# 5. Business Logic

5.1 Description of Core Logic  
The core logic of the Inventory Management System (IMS) revolves around automating inventory-related processes to ensure accuracy, efficiency, and reliability. The system’s business logic is implemented through the following components:

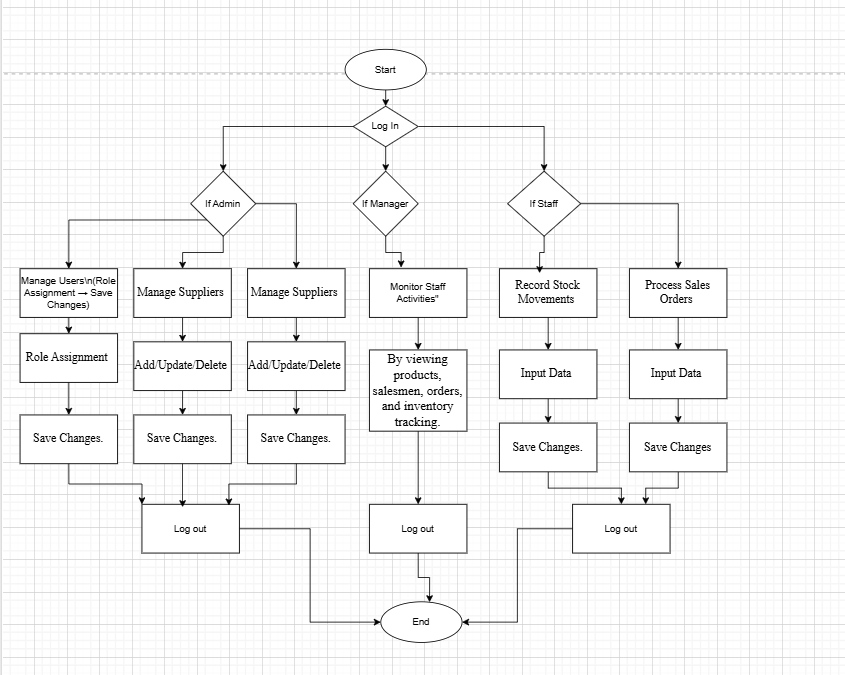
* **Relational Database Operations**:   
  The database structure supports CRUD (Create, Read, Update, and Delete) operations for users, products, suppliers, sales orders, and stock movements, ensuring data consistency.
* **Role-Based Access Control**:   
  User permissions are determined by their roles (Admin, Manager, and Staff). The system restricts access to sensitive functions based on these roles, enhancing security.
* **Order Processing**:   
  Sales orders are processed using a step-by-step workflow that validates data, updates stock levels, and maintains records of transactions.

5.2 Problem-Solving Approach  
The system’s business logic is designed to address the following challenges:

* **Eliminating Manual Errors**:   
  By automating inventory tasks such as stock updates and order processing, the system reduces the risk of human errors in calculations and data entry.
* **Ensuring Real-Time Data Access**:   
  The system integrates data updates in real-time, ensuring users always have the most accurate and current information.
* **Enhancing Data Security**:   
  Role-based access control and validation mechanisms ensure only authorized users can perform critical operations, protecting sensitive data.
* **Simplifying Decision-Making**:   
  Detailed reports and forecasting features provide managers with insights to anticipate trends, prevent stock shortages, and optimize procurement strategies.
* **Improving Workflow Efficiency**:   
  By streamlining tasks like sales order creation, supplier management, and stock tracking, the system reduces time and effort while enhancing productivity.

# 6. Flow Chart

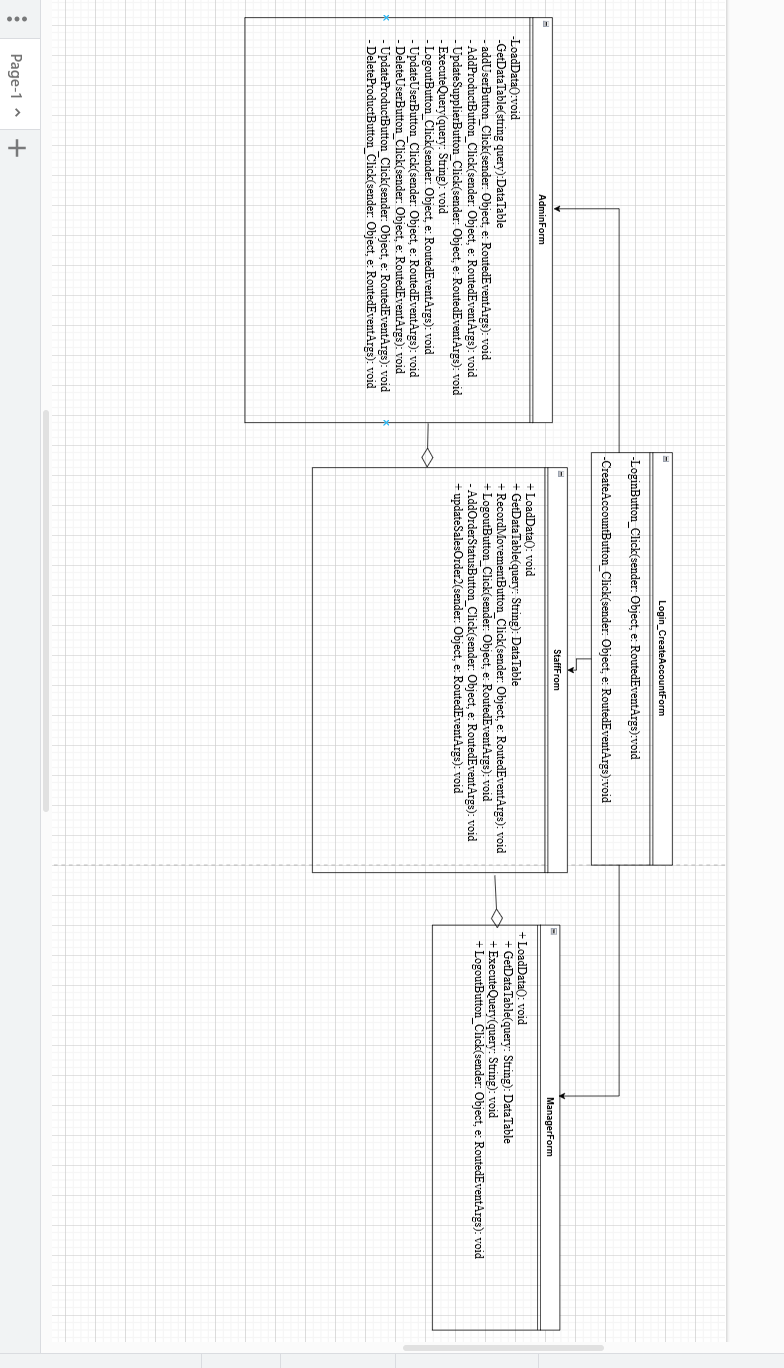
6.1 Application Flow Representation



## ****6.2 Annotations and Explanation****

* **User Authentication**: The flow begins with user authentication, ensuring secure access based on role-based credentials.
* **Role-Specific Dashboards**: Once logged in, users are directed to dashboards tailored to their roles, presenting only the functionalities they are authorized to use.
  + **Admin Dashboard**: Includes comprehensive management options for users, products, and suppliers.
  + **Manager Dashboard**: Focuses on monitoring staff activities.
  + **Staff Dashboard**: Allows viewing inventory details, recording stock movements, and managing sales orders.
* **Core Processes**: Each functionality (e.g., product management, sales orders, reporting) is executed step-by-step to ensure data integrity and successful task completion.
* **Logout**: Users can securely end their session to prevent unauthorized access.

# 7. Class Diagram

7.1 Overview of Class Structure

7.2 Class Relationships

This section focuses on how the classes relate to one another:

1. **Login Class**
   * Has associations with **Staff Class**, **Manager Class**, and **Admin Class**. Each user logs in using the **Login Class**, which determines the user Type (e.g., Staff, Manager, and Admin).
2. **Admin Class**
   * Inherits from **Manager Class** because an Admin may have extended permissions that include managerial capabilities
3. **Manager Class**

* May supervise or manage multiple **Staff Class** instances.

## ****7.3 Associations and Inheritance****

**Associations:**

* **Login Class** ↔ **Staff Class**, **Manager Class**, and **Admin Class**:
  + A user Type in the Login Class determines which class is instantiated.
* **Manager Class** ↔ **Staff Class**:
  + A Manager assigns tasks to Staff.

**Inheritance:**

* **Admin Class** inherits from **Manager Class** because an Admin has broader capabilities than a Manager.

# 8. Implementation

## 8.1 Development Process

The implementation process for a WPF application where Admins and Managers can view activities of Staff follows a structured and iterative development process

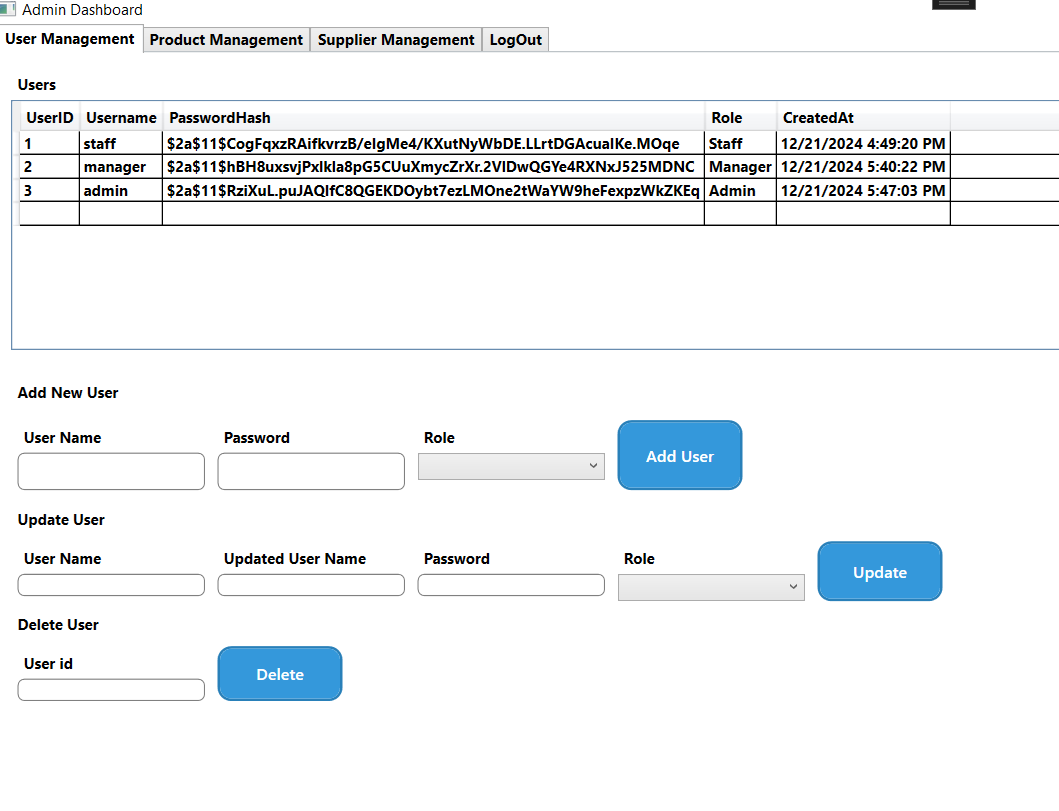
## 8.2 Screenshots of all User Interfaces

### Login Page:

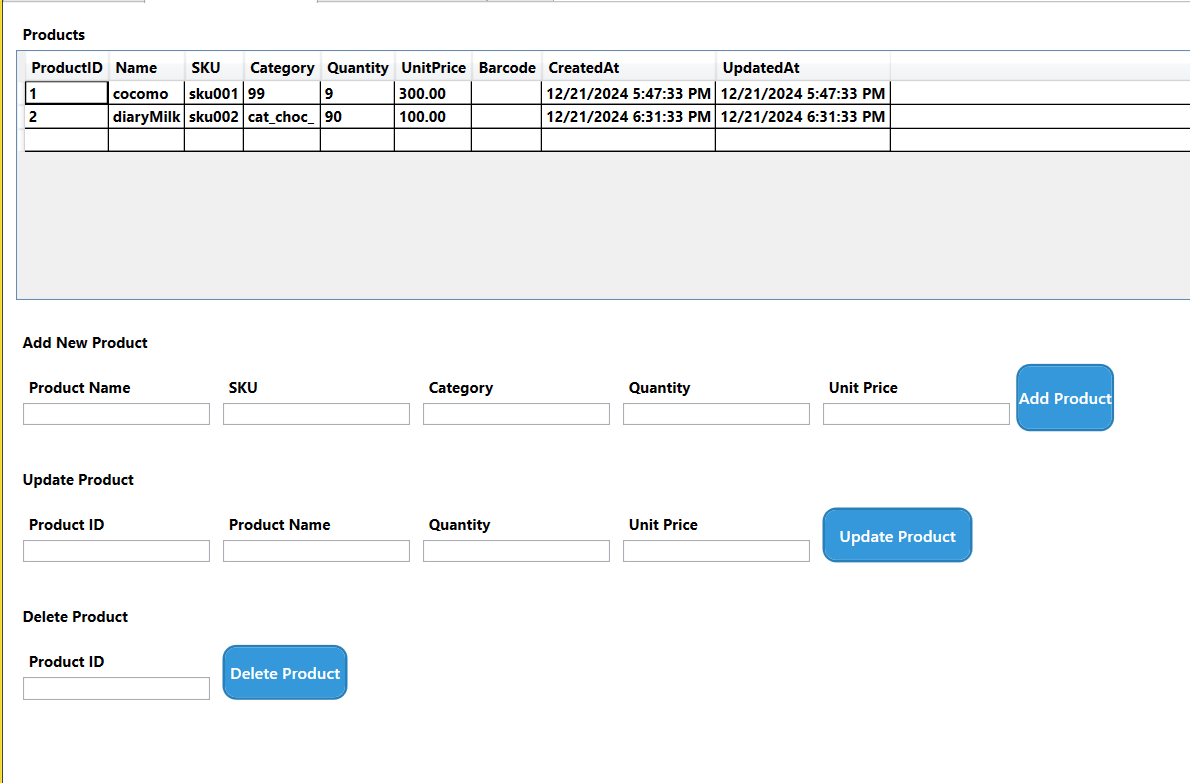
## 

### Admin Interfaces

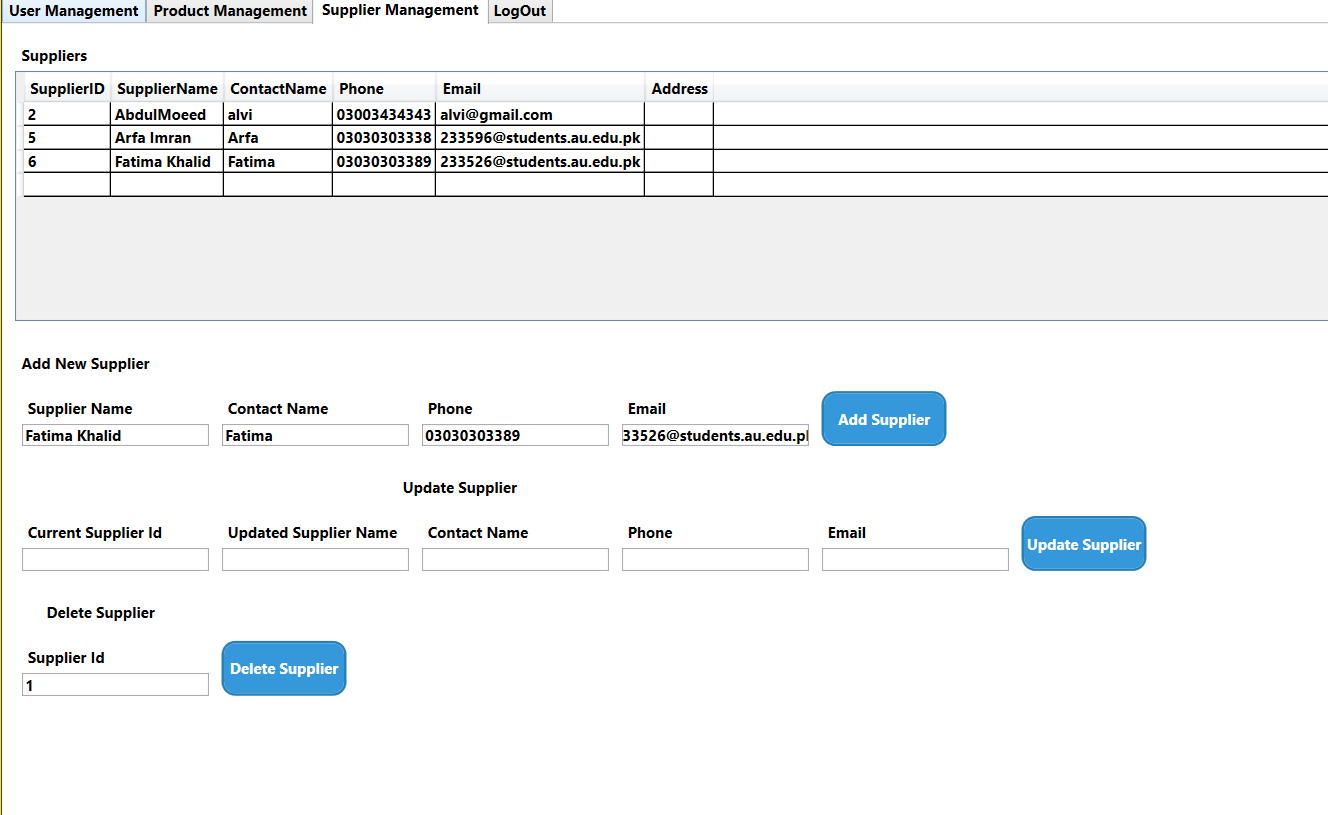
* User Management



* Product Management



* Supplier Management



## Logout page:

## 

### C:\Users\zain\Desktop\manager\dsa.PNGManager Interfaces:

### Orders Page

## C:\Users\zain\Desktop\manager\Capture.PNG

## Products Page

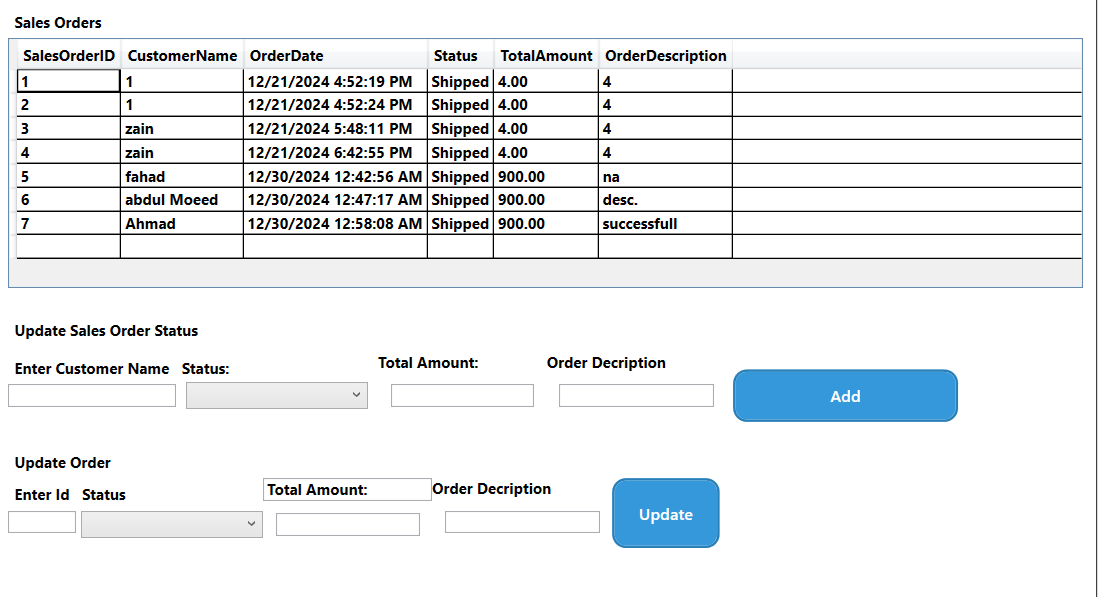
## C:\Users\zain\Desktop\manager\dssd.PNG

## Stock Movements Page

## C:\Users\zain\Desktop\manager\ds.PNG

## Sales Persons Page

### Staff Interfaces

* Sales Orders
* Stock Movements

## 

## 8.3 Tools and Technologies Used

* **Frontend:** XAML
* **Backend:** C#
* **Database:** Microsoft SQL Server
* **Framework:** .NET Framework
* **Libraries:** Entity Framework for data access
* **Development Tools:** Visual Studio, SQL Server Management
* **Hashing :** BCrypt

## 8.4 Challenges and Solutions

During the development of the Inventory Management System (IMS), several challenges were encountered. These challenges were addressed through various solutions that enhanced the overall functionality and usability of the system.

## 1. ****Challenge: User Authentication and Role-Based Access Control (RBAC)****

**Problem:** Implementing role-based access control (RBAC) to differentiate between Admin, Manager, and Staff users posed challenges in terms of ensuring secure and efficient access management. Additionally, the system needed to ensure that only authorized users could perform certain operations.

**Solution:** The solution involved creating a user authentication system with password hashing using BCrypt to enhance security. Role-based access was implemented using conditional checks in the backend to restrict access to specific features based on the user's role. This ensured that each user could only perform tasks relevant to their assigned role.

## 2. ****Challenge: Real-Time Data Synchronization****

**Problem:** Keeping inventory data up-to-date in real-time was crucial for the system's effectiveness, particularly when multiple users were interacting with the database simultaneously. This posed the challenge of ensuring data consistency without any lag or delays.

**Solution:** The database was configured for real-time updates, and data operations (such as stock movements, orders, and product management) were designed to instantly reflect changes on the user interface. This was achieved by using efficient queries and leveraging the capabilities of Entity Framework to handle transactions in a way that minimized delays.

## 3. ****Challenge: User Interface (UI) Design****

**Problem:** Designing a user-friendly and intuitive interface for users with varying technical expertise (Admin, Manager, and Staff) was a key challenge. The goal was to make the application accessible and easy to navigate for users while maintaining robust functionality.

**Solution:** A WPF application was developed with a modern, user-centric design that used role-specific dashboards. Each role had a tailored interface, ensuring that users only saw relevant information and actions. The UI was designed using XAML, and various controls were used to optimize the layout for usability, including buttons, dropdown menus, and data grids for displaying inventory information.

## 4. ****Challenge: Data Security****

**Problem:** Given the sensitivity of the data, particularly regarding sales orders and inventory management, ensuring the security of the data against unauthorized access was a critical challenge.

**Solution:** The application implemented several layers of security, including user authentication, role-based access control, and data validation mechanisms. Passwords were hashed using BCrypt, and access to sensitive features was restricted based on user roles. This ensured that only authorized personnel could perform critical tasks.

## 

# 9. Conclusion and Results

9.1 Project Outcomes Summary  
The Inventory Management System (IMS) successfully meets its primary goal of streamlining inventory-related tasks within an organization. Key accomplishments include:

* Implementation of a robust relational database for managing products, suppliers, orders, and stock movements.
* Development of a user-friendly WPF application with role-based dashboards for Admin, Manager, and Staff users.
* Integration of essential features such as user management, stock monitoring, and report generation, ensuring operational efficiency.

The system enhances data accuracy, reduces manual errors, and provides real-time insights for better decision-making.

9.2 Comparison of Objectives and Results

| **Objective** | **Achieved Results** |
| --- | --- |
| Develop a relational database for inventory | Fully implemented with efficient data handling. |
| Create a WPF application with role-based access | Designed and tested with Admin, Manager, and Staff roles. |
| Ensure data integrity | Role-based access and input validation implemented. |
| Provide accurate stock tracking | Real-time stock monitoring. |
| Ensure data security | Enhanced password Security using hashing. |

9.3 Future Enhancements  
to further improve the IMS, the following enhancements are recommended:

1. **Integration with Cloud Services**:
   * Enable cloud-based data storage and synchronization for multi-location organizations.
2. **Mobile Application Support**:
   * Develop a mobile version of the IMS to provide accessibility for users on the go.
3. **Enhanced Forecasting**:
   * Implement AI-driven predictive analytics for more accurate demand forecasting.
4. **Barcode and QR Code Support**:
   * Integrate barcode and QR code scanning for faster inventory management and stock updates.
5. **Automated Alerts and Notifications**:
   * Add automated email or SMS alerts for stock thresholds, order deadlines, and supplier updates.
6. **Third-Party System Integration**:
   * Connect the IMS with external systems like accounting or ERP software to streamline workflows.

By addressing these enhancements, the IMS can provide even greater value and scalability to meet evolving organizational needs.

## 

# 10. References

## 10.1 Cited Resources

1. "Pro WPF in C# 2010: Windows Presentation Foundation in .NET 4" by Matthew MacDonald. Après, 2010.
2. Microsoft Learn: **"Entity Framework Core Documentation"** – Comprehensive guide on using Entity Framework for database management. Available at: <https://learn.microsoft.com>.

## 10.2 Further Reading

1. "Design Patterns: Elements of Reusable Object-Oriented Software" by Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. Addison-Wesley, 1994.
2. Microsoft Learn: **"WPF Documentation"** – In-depth resources on creating WPF applications. Available at: <https://learn.microsoft.com>.

## 

# 11. Appendices

## 11.1 Code Snippets

## Create Account

## 

## Login Account

## 

## Add Product

## 

## Delete Product

## 

## Update Product

## 

## Load Data for Manager UI

## 

## Add Order (Staff)

## 

## 

## Stock Movement (STAFF)

## 

## 