****

**UNIVERSITY OF SRI JAYEWARDENEPURA**

**Faculty of Technology**

**Department of Information and Communication Technology**

**ITS3552 – Software Engineering II**

**Assignment 02**

**Inventory Management System (IMS) Software Requirements Specification**

**GROUP MEMBERS:**

|  |  |  |
| --- | --- | --- |
| **Name** | **Index Number** | **Email** |
| **KATHRIARACHCHI D.A.K.K.** | **ICT/20/868** | **ict20868@fot.sjp.ac.lk** |
| **NUZHA M.W.W.** | **ICT/20/898** | **ict20898@fot.sjp.ac.lk** |
| **PREMACHANDRA Y.V.P.** | **ICT/20/907** | **ict20907@fot.sjp.ac.lk** |
| **RUSTHA M.Y.F.** | **ICT/20/920** | **ict20920@fot.sjp.ac.lk** |
| **WEERARATHNA T.D.** | **ICT/20/955** | **ict20955@fot.sjp.ac.lk** |

**Table of Contents**

**1. Introduction**

1.1. Purpose

1.2.Scope

1.3.System Overview

**2. System Requirements**

2.1.Functional Requirements

2.2. Non-Functional Requirements

2.3 User Roles and Permissions

2.4 Use Cases

**3. System Design**

3.1 Architecture pattern

3.2 Design Patterns

3.3 high-level architecture diagram and UML diagrams

**4. Interface Design**

4.1. User Interface Mockups

**5. development procedure**

5.1 Code Best Practices

5.2.Technologies

5.3.Collaboration

**6. Testing**

6.1 Testing Types

6.2 Test Cases and Test Results

**7. Security**

7.1 Code-Level Security Principles

**8. Maintenance**

8.1 Bug Tracking and Resolution

8.2 Software Updates and Enhancements

8.3 Software Configuration Management (SCM)

**9. Assumptions**

**10. Conclusion**

**11. Appendices**

11.1 Glossary

11.2 Acronyms

11.3 Document History

12. **References**

**ACKNOWLEDGEMENT**

As a team, we would like to express our deepest appreciation to all who allowed us to complete this IMS project successfully. Special gratitude, we would like to give Mrs. Sankhani Heenkenda, our lecturer for giving us valuable Guidance and advice, clearing all our doubts at any time without any hesitation, and for all the support shown throughout the period to complete our project, University, and the Department who provides academic environment and resources that are necessary to conduct the research about this assignment. Finally, we would like to take this opportunity to express our gratitude to all our group members, Kathriarachchi D.A.K.K, Nuzha M.W.W, Premachandra Y.V.P, Rustha M.Y.F, Weerarathna T.D. project would not have been successful without their cooperation and input.

1. **INTRODUCTION**

Inventory management involves tracking your production and sales to make sure your inventory always contains the products your customers want. The flow of commodities and products inside a company is properly managed by an IMS which is a crucial part of modern enterprises. In software engineering, an IMS is a comprehensive software solution that combines different modules and functionalities to simplify inventory-related tasks. Businesses may reduce stockouts, increase operational efficiency, and maintain correct inventory records with the use of these technologies. Businesses can utilize technology to acquire real-time visibility into their inventory and make decisions. Designing, and maintaining a reliable, approachable, and scalable inventory management system that is tailored to the requirements of the firm constitutes effective software engineering in this context.

**1.1 PURPOSE**

To keep a business well-organized and productive, inventory management is essential. The practice of inventory management is straightforward for some businesses. Every week or so, they conduct an inventory by going through a storage closet and determining what supplies they have left. However, other businesses must take inventory management very seriously, keeping track of each item as soon as it is delivered, moved, or used.

The basic goal of inventory management is to make it simple and effective for organizations to order, stock, store, and use inventory. You'll always be aware of the things you have on hand, their quantity, if you manage your inventory well.

**1.2 SCOPE**

Our IMS system will include a user-friendly interface that will make it possible for anyone, especially small business owners, to run it effectively without the need for training. A trustworthy and secure database will serve as the system's foundation, guaranteeing the correctness of real-time data, which is necessary for efficient inventory monitoring and control. Priority one will be given to performance improvement, allowing quick access to useful inventory data. The system will also be built to provide scalability, making it simple for administrators to integrate new software modules with little configuration and enabling the system to adapt to changing business needs.

**1.3.SYSTEM OVERVIEW**

IMS is a software solution tailored for small businesses to efficiently manage their inventory and meet customer demands. It is designed to provide:

1. **User-Friendly Interface**: The IMS offers easy-to-use interface that requires no advanced training, ensuring accessibility for all users.
2. **Reliable Database**: The system is built on a secure and reliable database that ensures the accuracy and real-time availability of inventory data.
3. **High Performance**: IMS delivers fast and actionable inventory monitoring and control capabilities, enhancing the efficiency of the system.
4. **Scalability**: Administrators can effortlessly expand the system's functionality by adding software modules with minimal configuration, allowing for future growth and adaptation to changing business needs.
5. **SYSTEM REQUIREMENTS**

**2.1.FUNCTIONAL REQUIREMENTS**

1. User Authentication and Authorization:
   * Users must log in with unique credentials.
   * Before logging in, the user should register into the system.
2. Product Management:
   * Add new products with details.
   * Update product information.
   * Delete or discontinue products.
3. Reporting and Analytics:
   * Generate reports when it is necessary.
4. Low Stock warnings:
   * Indicate low stock warnings, on the data grid view in the Dashboard clear and understandable manner.

**2.2. NON-FUNCTIONAL REQUIREMENTS**

1. Performance:
   * The system should provide fast response times, even with a large volume of data.
2. Scalability:
   * The system must support easy expansion by allowing administrators to add new features or modules.
3. Reliability:
   * The database should be highly reliable.
   * The system must prevent data corruption.
4. Security:
   * Implement robust security measures to protect user data and prevent unauthorized access.
   * Regularly update and patch the system to address security vulnerabilities.
   * Implement user authentication.
5. Usability:
   * The user interface should require minimal training.
   * Response in real-time
6. Availability:
   * Ensure that the system is available 24/7.

**2.3 USER ROLES AND PERMISSIONS**

* Administrator (Manager):

The Administrator, often referred to as the Manager, is a user with the highest level of authority and control within the system.

Their primary responsibilities include overseeing and managing the entire system, making decisions, and ensuring the system functions correctly.

Administrator side typically include:

* User authentication: Login/Register into the system with unique credentials.
* Generating reports.
* Product management: Adding, Updating, or Deleting products from the inventory.
* System configuration: customizing the system to meet the organization's evolving needs.
* Security management: Configuring and maintaining security measures, including user authentication, authorization, and password policies.
* Employee:
* Employees are users with limited access and responsibilities within the inventory management system.
* Their role typically involves day-to-day tasks related to inventory management but does not include administrative functions.

Employee permissions often include:

* User authentication: Login/Register into the system with unique credentials.
* Generating reports .
* Product management (partial access): Some employees may have permission to update product quantity and price.

**2.4 USE CASES:**

* Create, delete and Update Products:
* Description
  + This use case involves the ability to create new product entries in the inventory management system, delete damaged and expired products and update existing product information as needed.
* Functionality
* Creating Products: administrator can add new products to the system. This includes providing details such as product name, unit price, Quantity, supplier name, supplier phone number, and total value.
* Delete product: administrator can delete products from the system.
* Updating Products: admin and employee(partial access) both Users can modify product information when necessary. This could involve changing product prices, name, or other attributes.
* Purpose: This use case ensures that the inventory database is up to date with the latest product information, enabling accurate tracking and management of inventory.

**3. SYSTEM DESIGN**

**3.1.** ARCHITECTURE PATTERN

We selected MVC architectural patterns for design our system. The following outlines ,the application of the MVC architectural pattern to our IMS. MVC is a well-established pattern that separates the concerns of internal representations of information from the ways it presented to and accepted from the user(break up the front end and the back end into separate components), contributing to better maintainability and scalability.

**1. Model (Data and Business Logic):**

Description: The Model component of the IMS is responsible for managing data, implementing business logic, and ensuring data integrity.

Responsibilities:

* Maintain a secure and reliable database for storing product information.
* Handle data validation, ensuring that data follows predefined business rules.

Purpose: The Model component ensures that data is managed consistently and provides the foundation for accurate inventory management.

**2. View (User Interface):**

Description: The View component of the IMS is responsible for presenting data to users through a user-friendly interface.

Responsibilities:

* Show product listing, reports, and real-time inventory information to users.
* Provide a user-friendly interface for creating, deleting, updating, and viewing products .
* Indicate low stock warnings, on the data grid view in the Dashboard clear and understandable manner.

Purpose: The View component ensures that users can interact with the system efficiently and receive information.

**3. Controller :**

Description: The Controller component acts as a bridge between the Model and the View, managing user input and business logic. It will handle all the interactions and inputs from the user view and update the DB using the user model.

Responsibilities:

* Receive and process user requests for actions such as adding new products, deleting, and updating products when necessary generating reports.
* Validate user input.
* Communicate with the View to display updated information to users.

Purpose: The Controller component separates the user interface from the business logic, promoting modularity and maintainability, and supporting multiple views for different user interfaces.

**Why have we used MVC?**

Implementing the MVC architectural pattern in our IMS is a strategic choice. This approach ensures a clear separation of concerns, allowing us to manage data, user interfaces, and business logic independently. With MVC, we can enhance the system's maintainability, scalability, and user experience. The Model handles data and business logic, the View delivers a user-friendly interface, and the Controller manages user interactions and data flow. This separation empowers us to adapt and extend the IMS efficiently as our inventory management needs evolve, ultimately providing a robust and user-centric solution.

**Advantages of Using the MVC Pattern:**

* Modularity: MVC divides the system into distinct components, making it easier to develop, test, and maintain each component independently.
* Scalability: The system can be scaled by adding more Views or Controllers without major changes to the underlying Model.
* Reusability: Components like the Model and Controller can be reused across different parts of the system or in future projects.
* Maintainability: Changes to one component don’t affect others, it will reduce the risk of introducing errors.
* Enhanced User Experience: The View can be customized to provide user-friendly interface, improving user satisfaction.

**Limitations:**

* Testing Challenges: Writing unit tests for MVC can be complex, especially when dealing with the user interface components in View.
* Duplication of code: there can be duplication codes between controller and views. Such as validation logic might be present in both a user side (view) and the server side (controller).

**3.2 DESIGN PATTERNS**

Design Patterns we used within the MVC Architecture of our IMS:

* Singleton Pattern :

We apply the Singleton pattern to our model component to ensure that there is only one instance of the Model throughout the application's lifespan. This guarantees data consistency and integrity by preventing multiple instances with potentially conflicting data.

It restricts the instantiation of the Model class to a single instance. Any part of the application that requires access to the Model interacts with this single instance. This ensures that all data and business logic are centralized, reducing the risk of data inconsistencies.

* Observer Pattern :

Within the View component, we use the Observer pattern to address the requirement for real-time updates of inventory information displayed to users.

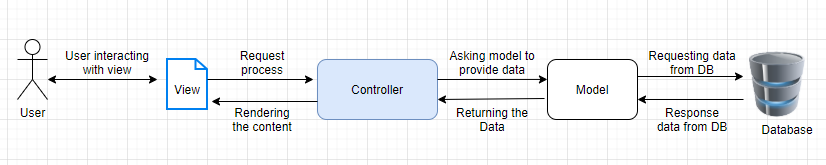
Views register themselves as observers of the Model. When changes occur in the Model's data, such as inventory updates, the Model notifies all registered observers (Views). This mechanism ensures that Views automatically update their displays to reflect the most current data, enhancing the user experience with real-time information.

* Factory Method Pattern :

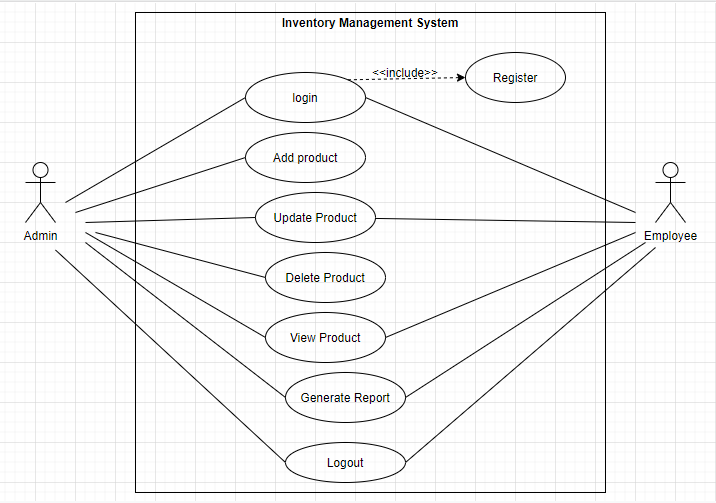
Within the Controller component, we use the Factory Method pattern to decouple the creation of specific controllers for different user interactions.

The Factory Method pattern defines an interface for creating Controllers, but the concrete subclasses of the Factory determine which specific Controller to instantiate based on the user's request. This promotes flexibility by allowing the application to create the appropriate Controller dynamically, based on the user's interaction, without tightly coupling the Controller creation logic to the user interface.

**3.3 HIGH-LEVEL ARCHITECTURE DIAGRAM AND UML DIAGRAMS**

****

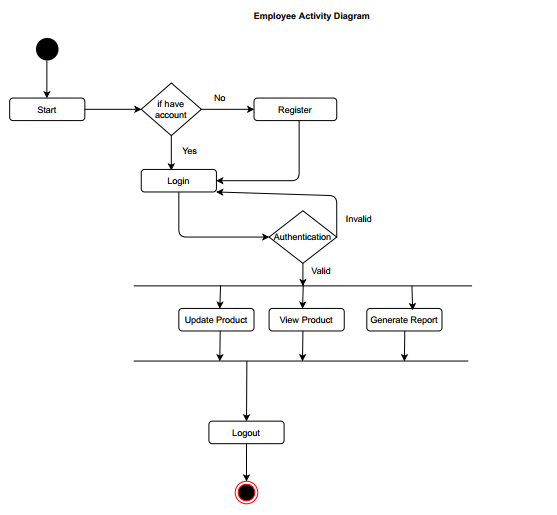
**Use case diagram.**



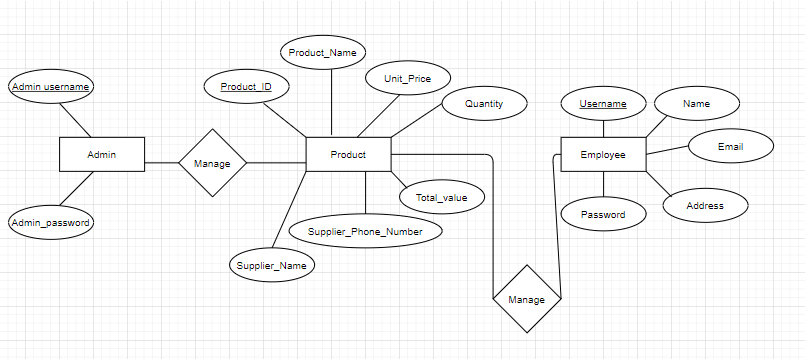
**Activity diagram.**

A diagram of a program

Description automatically generated



Class diagram.

ER diagram.

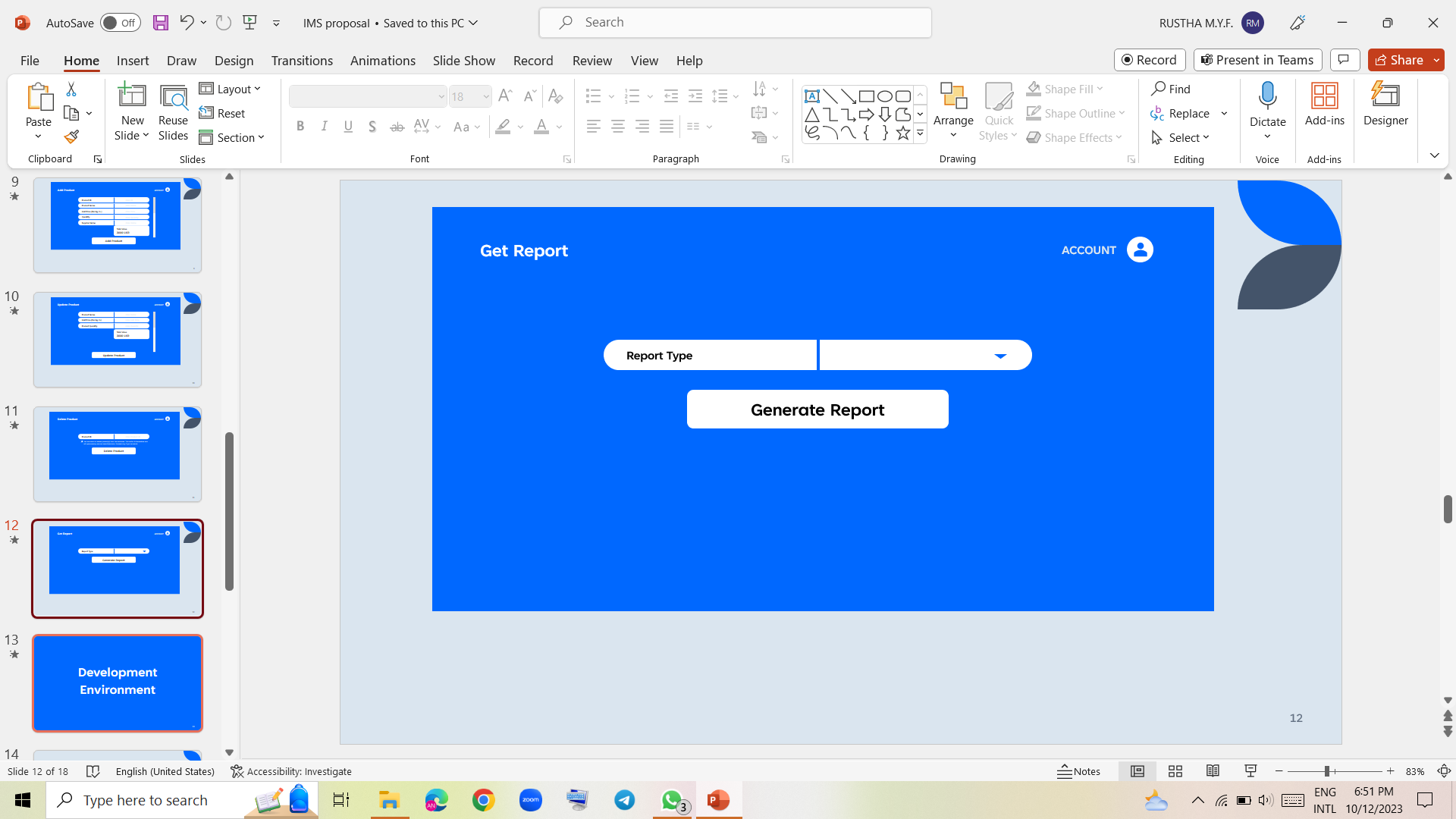
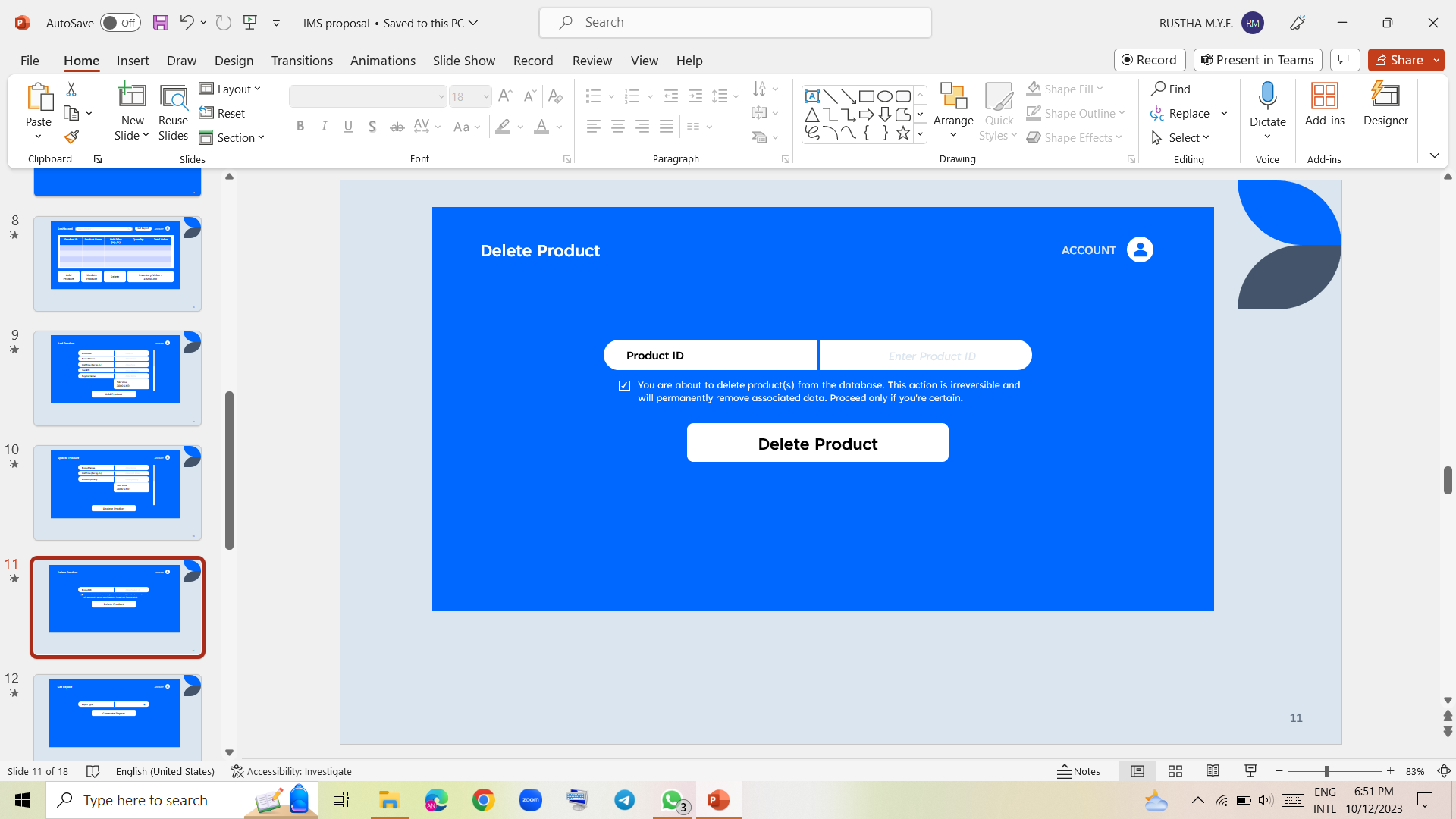
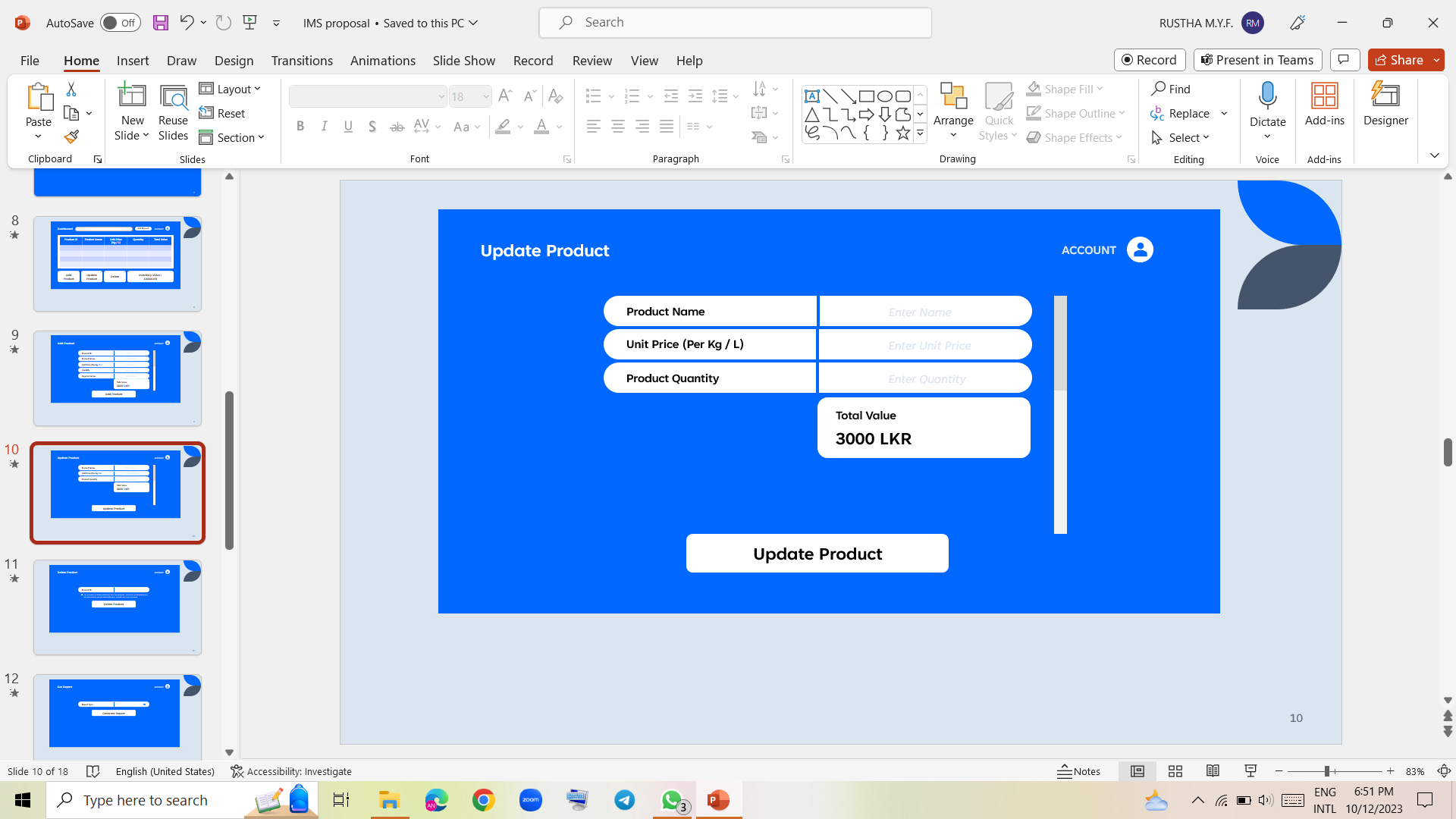
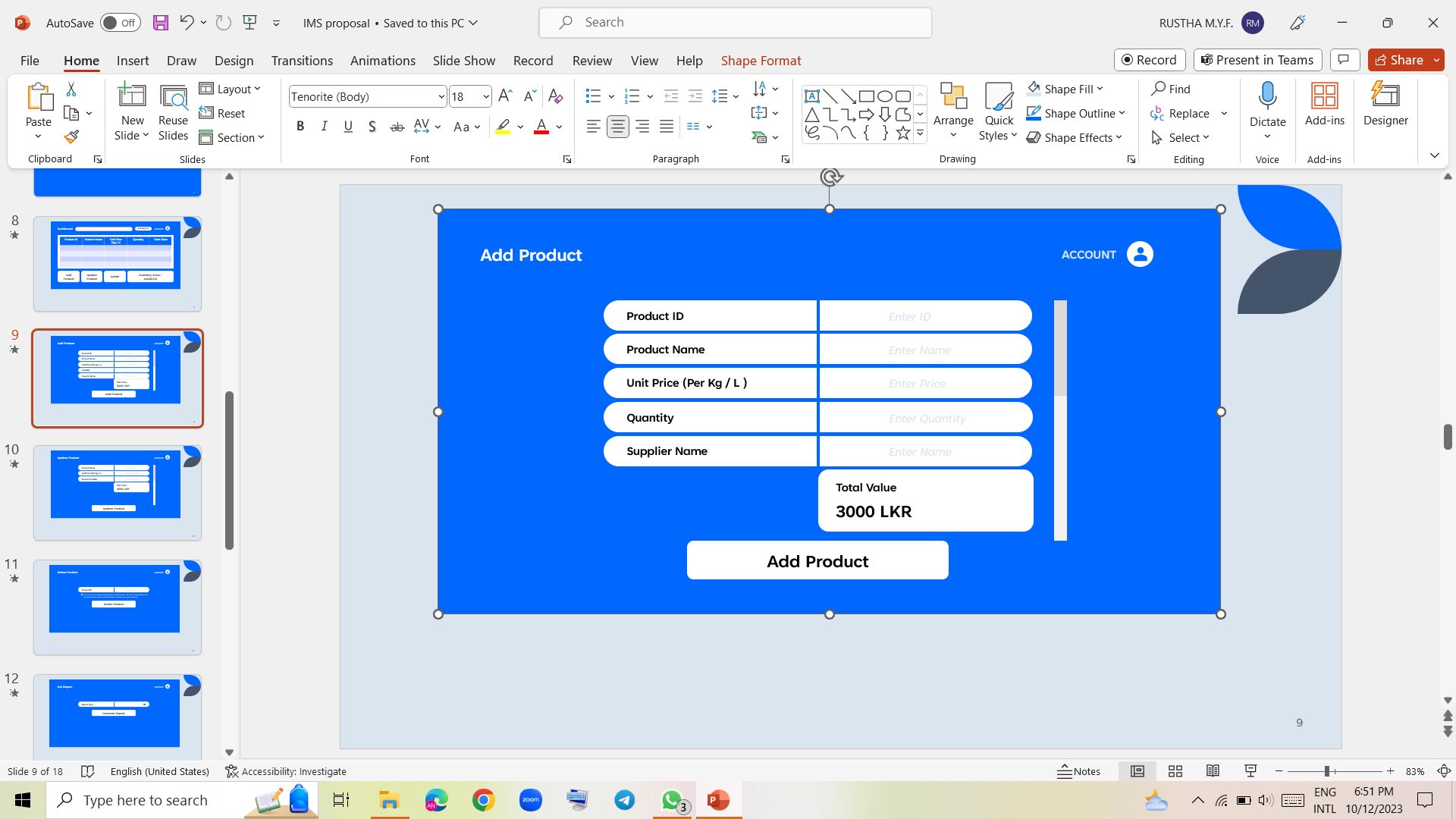
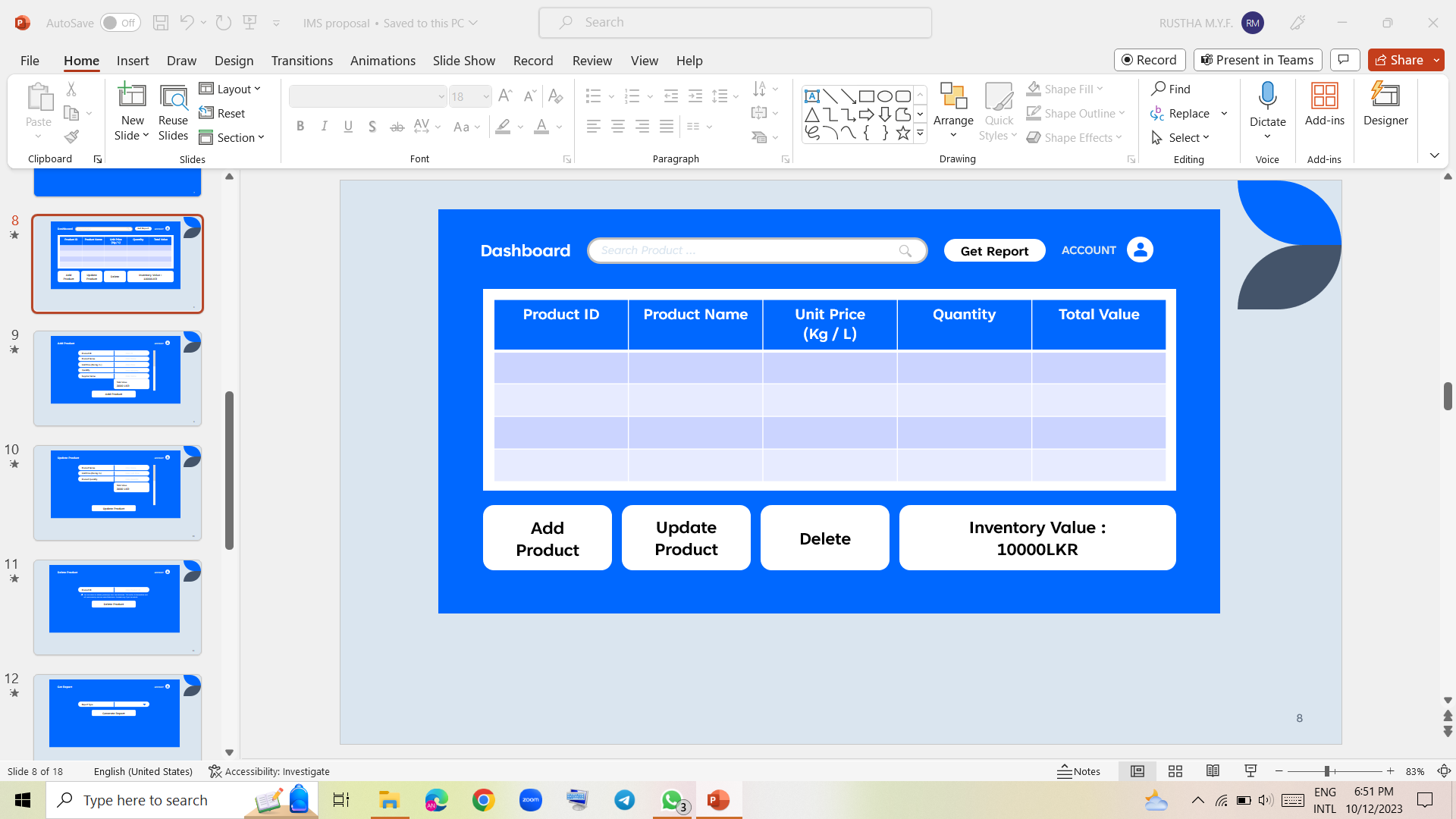
**Record in DB**

A screenshot of a computer

Description automatically generated

**4.SYSTEM DESIGN**

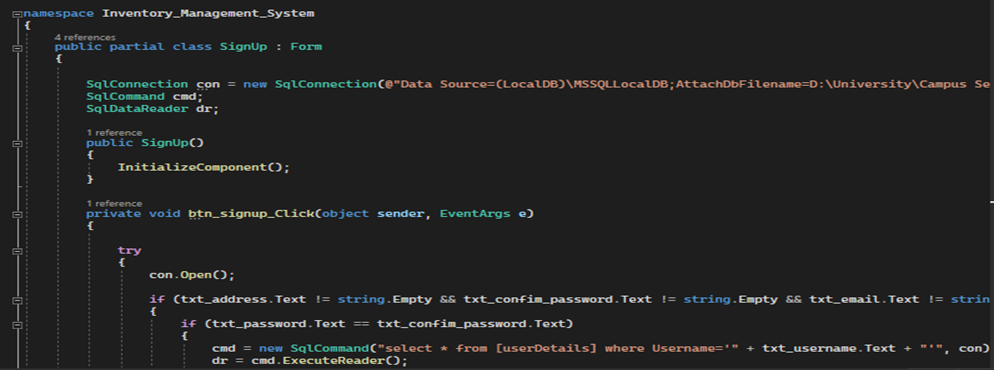
4.1 USER INTERFACE MOCKUPS



**5.DEVELOPMENT PROCEDURE**

**5.1 CODE BEST PRACTICES**

* Use Pascal casing for Class names.

****

* A screen shot of a computer program

  Description automatically generatedUse Pascal casing for Method names
* A screenshot of a computer program

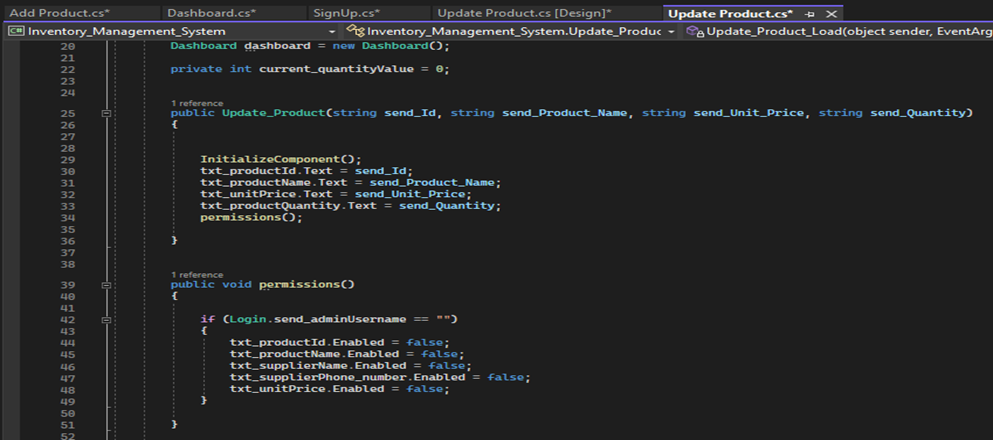
  Description automatically generatedUse Pascal Case for file names
* Use appropriate prefixes for the UI elements so that you can identify them from the rest of the variable**s.**

****

**A screen shot of a computer

Description automatically generated**

* Curly braces ( {} ) should be at the same level as the code outside the braces.

****

* A computer screen shot of a program

  Description automatically generatedThe curly braces should be on a separate line and not in the same line as if, for etc.
* Use a single space before and after each operator and brackets.

**A screen shot of a computer program

Description automatically generated**

* Comments should be on the same level as the code .

**5.2.TECHNOLOGIES**

* C# (Programming Language):

We have chosen C# as the primary programming language for building our IMS. C# is a versatile and widely adopted language known for its strong typing, modern features, and compatibility with various development platforms, making it an excellent choice for creating scalable and maintainable software solutions. C# is used to implement business logic, data processing, user interface, and various functionalities within the IMS.

* .NET Framework:

provides a comprehensive set of tools and libraries for building Windows desktop applications with a rich user interface.it is best for Windows-based applications. WinForms is a graphical user interface (GUI) library that allows you to create Windows-based applications with a traditional, native Windows look and feel.

* MySQL (Database Management System):

MySQL is our chosen database management system (DBMS) for storing and managing the IMS's critical data, including product information. MySQL is an open-source, relational database that offers excellent performance and scalability. MySQL serves as the backend database for our IMS, ensuring data reliability, integrity, and efficient data retrieval and storage.

In Summary, these technologies, C#, and MySQL, complement each other in our IMS, enabling us to build a user-friendly and scalable inventory management solution while ensuring data accuracy and accessibility. Our choice of these technologies reflects our commitment to delivering a reliable IMS that can adapt to the evolving needs of our business.

**5.3.COLLABORATION**

Collaboration is essential in the development of our IMS. Effective communication and the use of collaboration tools are key to the success of this project. Here's how we facilitate collaboration within our development team:

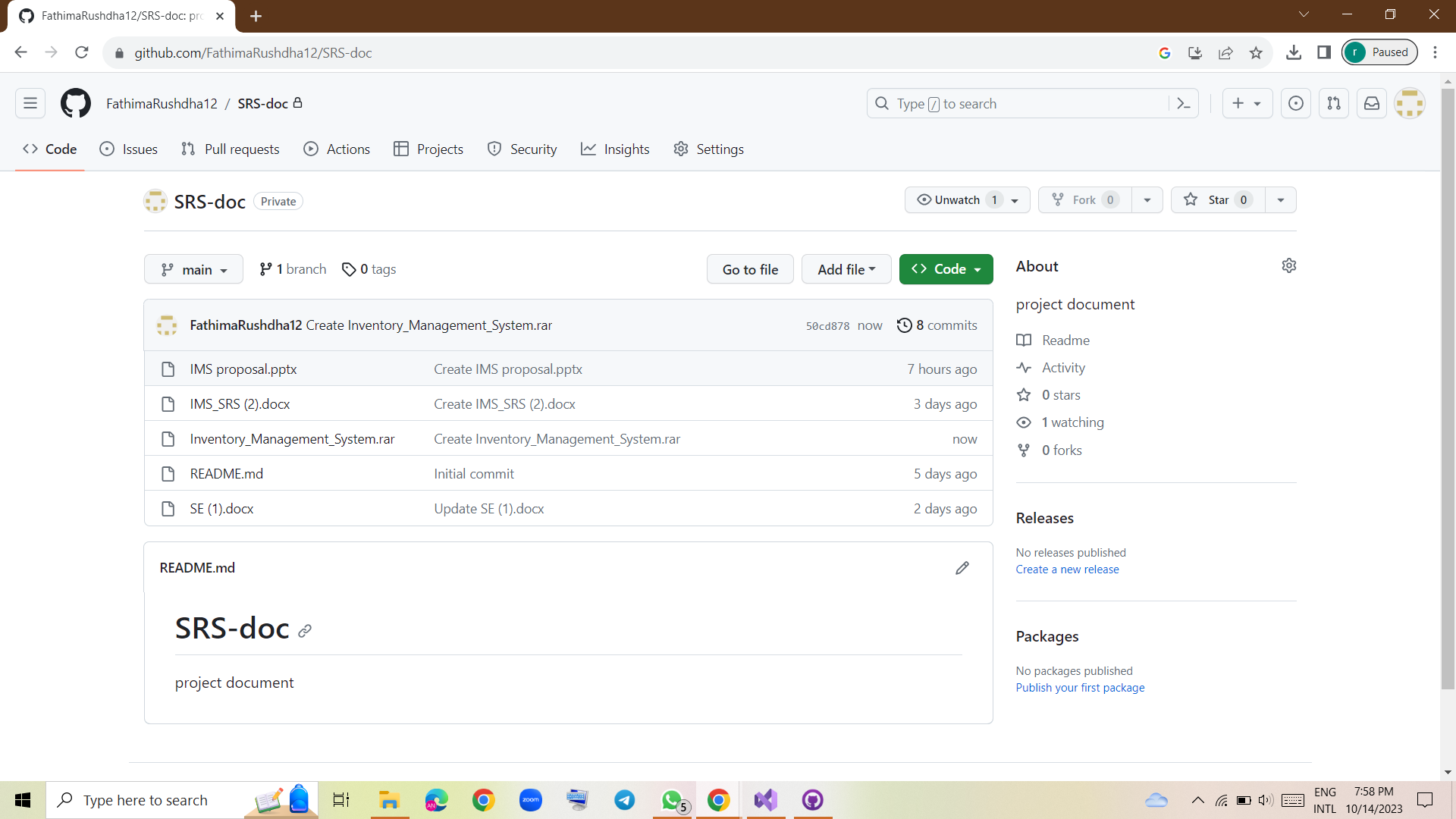
**Version Control :** It indicates the practice of managing the changes in our software coding and documentations. Utilize GitHub as our version control system, enabling us to track changes, and collaborate on code development efficiently. Our Git repository is hosted on a platform that facilitates collaboration and code review.

**Team Collaboration Tools:** We went through the various collaboration tools to communicate with our group members to make changes, discussions and review the code.

* **Git and GitHub:**

Git is a version control system, while GitHub is web-based platform for hosting Git repositories and managing collaborative development.

Git and GitHub are essential for version control, code collaboration, and code review. They enable us to work on the IMS codebase, manage code changes, and conduct code reviews, also documentation.

A screenshot of a computer

Description automatically generated

* **Zoom (for Meetings):**

These platforms provide video conferencing, screen sharing, and virtual meeting capabilities.

It helps us for team meetings and discussions ,especially when team members are geographically apart from each other.

**Collaboration Workflow:**

Code Reviews: We conduct thorough code reviews to maintain code quality, identify errors, and share knowledge within the team.

Agile Practices: We follow agile methodologies, conducting regular sprint planning, meetings, and retrospectives to ensure that development aligns with project goals.

Task Assignment: Tasks are assigned based on team members’ expertise.

By employing these collaboration practices and tools, we create an efficient development environment that promotes teamwork, and the successful development of our IMS.

**6.TESTING**

6.1 TESTING TYPES

Unit Testing

Integration Testing

Functional Testing

Performance Testing

User Acceptance Testing

6.2 TEST CASES

6.3TEST RESULTS

**7. SECURITY**

7.1 CODE-LEVEL SECURITY PRINCIPLES

1. **Input Validation:**
   * Always validate user inputs to prevent data corruption ,for example ,unit price and quantity can’t be string ,supplier phone number should be within 10 digits.
2. **Authentication and Authorization:**
   * Implement strong authentication mechanisms to verify the identity of users.
   * authorization mechanisms to ensure that only authorized users can access and modify inventory data.
3. **Password Security:**

* Enforce password length requirements.
* notify unsuccessful login attempts.
* Log out functionality should be available from all pages.
* Log out functionality should fully terminate the associated session or connection.

**8. MAINTENANCE**

Maintenance is an essential aspect of the IMS project's lifecycle, ensuring that the system continues to operate effectively and meets evolving business needs.

8.1 BUG TRACKING AND RESOLUTION

* **Issue Identification**: Users report issues encountered in the system, including errors, unexpected behaviors, or usability problems.
* **Testing**: Bug fixes undergo testing to ensure they do not introduce new issues.
* **Documentation**: Updates to the system, including bug fixes and their release notes, are documented.

8.2 SOFTWARE UPDATES AND ENHANCEMENTS

* **Version Control**: All software changes and documentation, including updates and enhancements, are managed through the version control system.
* **Feature Enhancements**: new features are developed to meet user requirements and business objectives.
* **Scalability**: As the business grows, the system should be capable of scaling to accommodate increased demand.
* **Security Updates**: Security vulnerabilities are addressed with regular updates, patches, and security best practices.
* **User Training**: When new features are introduced, user training and documentation are provided.

8.3 **SOFTWARE CONFIGURATION MANAGEMENT (SCM)**

* **Version Control**: A version control system, Git and GitHub, is used to manage and track changes to the software code and documentation. It ensures that a record of every change is maintained, including who made the change.
* **Documentation Management**: Ensuring that documentation, including design specifications are kept up to date and accurately reflects the system's current state.
* **Traceability**: Establishing traceability between software requirements, design, and implementation to ensure that all requirements are met, and changes can be tracked throughout the development and maintenance phases.

**9. ASSUMPTIONS**

1. **User Availability**: We assume that end users (administrator and employee) will be available for user acceptance testing when needed.
2. **Data Accuracy**: We assume that data entered into the system is accurate and valid.
3. **Scalability**: We assume that the business's growth aligns with the IMS's scalability plans.
4. **User Authentication**: We assume that user authentication details, such as usernames and passwords, are managed securely.
5. **System Maintenance**: We assume that system maintenance, including software updates and enhancements, will be carried out to address the maintenance of system performance.

**10. CONCLUSION**

IMS represents a necessity for modern businesses, streamlining the complex task of inventory management. With a user-friendly interface, and scalability, the IMS aims to empower organizations to maintain precise inventory records and operational efficiency. The project incorporates version control and collaboration tools, to ensure efficient code management, documentation, and continuous improvement. Furthermore, by addressing regular maintenance, updates, and software configuration management, the IMS will remain a reliable asset for businesses, aligning with their evolving needs and promoting growth.

**11. APPENDICES**

**11.1 GLOSSARY**

1. Inventory Management System (IMS): involves tracking our production to make sure our inventory always contains the products our customers want.

2. User Roles: Distinct roles within the system, such as Administrator and Employee, with specific permissions and responsibilities.

3. Low Stock warnings: Indicate low stock warnings, on the data grid view in the Dashboard clear and understandable manner.

4. Version Control: A system for tracking and managing changes to the project's source code and documentation.

5.Software Configuration Management (SCM)-methodology for software problems request initiation and tracking, changing impact analysis, version control, security administration of software assets.

6..NET Framework-A software framework for build windows applications.

**1.2 ACRONYMS**

1. IMS: Inventory Management System

2. DBMS: Database Management System

3. SQL: Structured Query Language

4. MVC: Model-View-Controller

6. UI: User Interface

7. Sprint: A time-boxed development iteration in Agile methodologies

8. SCM-Software Configuration Management

12.**References**