TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PULCHOWK CAMPUS

A PROJECT REPORT ON

APPLICATION OF OBJECT-ORIENTED PROGRAMMING USING C++ "STOCKSYNC"

Submitted To:

Department of Computer and Electronics Engineering

Submitted By:

Ashok Prasad Neupane (078BCT021)

Asok Bk (078BCT022)

Anil Shrestha (078BCT009)

1. Acknowledgement

We would like to express our heartfelt appreciation to our professor Er. Daya Sagar Baral and the Department of Electronics and Computer Engineering at the Institute of Engineering, Pulchowk Campus. They have provided us with an opportunity to develop this project, which will not only strengthen our understanding of object-oriented programming but also enhance our teamwork skills.

This project will enable us to apply the theoretical knowledge we have acquired in our Object Oriented Programming course. We would like to extend our gratitude to our professor for continuously motivating us to explore new topics related to our field of interest.

Throughout the duration of this project, we look forward to receiving your valuable guidance and feedback. Any suggestions you provide will be highly appreciated.

Authors:

- **Ashok Prasad Neupane (078BCT021)**
- **Asok Bk (078BCT022)**
- **Anil Shrestha (078BCT009)**

2. Abstract

Programming (OOP) project focused on an Inventory Management System. The system incorporates robust authentication and authorization features, including secure password hashing. It encompasses three distinct roles: administrator, buyer, and seller. The project exemplifies the application of fundamental software engineering principles in building a comprehensive and efficient inventory management solution.

Keywords: C++, oop, inventory management, authentication, hashing

3. Table of Contents

4. Objectives

The main objectives of our project are as follows:

- To apply the concepts of Object Oriented Programming in practical application,
- To explore different features of C++ programming language,
- To implement external libraries for UI,
- To understand the dynamics of working in a group,
- To be familiar with git and Github for collaborating on a project in a team,
- To solve a real-world problem.

5. Introduction

As a dedicated group of computer engineering students, we are currently engaged in an exciting software project aimed at assisting companies, such as Amazon, in effectively managing their supplies, stocks, and orders, while ensuring seamless billing processes. By harnessing our in-depth understanding of authentication and authorization protocols, we are crafting a comprehensive solution that not only addresses these critical business needs but also showcases our expertise in object-oriented programming (OOP).

Our project is poised to revolutionize inventory management by providing a centralized platform that enables real-time tracking of supplies, efficient stock management, streamlined order processing, and accurate billing generation. Through the implementation of robust authentication and authorization mechanisms, we are committed to ensuring secure access and data integrity, safeguarding the vital information of both the company and its customers. Emphasizing our proficiency in OOP, we are employing industry best practices to create a modular and extensible software architecture. This approach allows for scalability, making it easy to adapt and integrate new features as business requirements evolve.

By leveraging our technical skills and collaborative efforts, we aim to deliver a cutting-edge solution that optimizes supply chain operations, enhances overall efficiency, and ultimately empowers businesses to thrive in the dynamic world of e-commerce. We are excited about the potential impact of our project and look forward to making a meaningful contribution to the industry.

6. Application

The Inventory Management System we are developing holds significant potential for various industries and businesses that rely on efficient inventory control and seamless order management. Some key scopes and applications of this solution include:

E-commerce Platforms:

- Online retailers can use this system to manage their vast product catalogs, track stock levels, and process orders swiftly and accurately. The role-based access ensures that administrators, buyers, and sellers have tailored functionalities.

Retail Chains:

- Large retail chains can utilize the system to centralize inventory data across multiple locations. This enables them to optimize stock distribution, reduce overstocking or understocking, and streamline replenishment processes.

Wholesale Distributors:

- Distributors dealing with bulk quantities can benefit from real-time tracking of inventory. The system assists in managing supplies, fulfilling large orders efficiently, and generating precise invoices.

Manufacturing Companies:

- Manufacturers can use the system to monitor raw material levels, work-in-progress, and finished goods inventory. This helps in ensuring continuous production and timely delivery.

Hospitality Industry:

- Hotels and restaurants can employ the system to manage their kitchen supplies, track usage of ingredients, and place timely orders to replenish stocks.

Medical Facilities:

- Healthcare institutions can benefit from inventory control for medical supplies, ensuring availability of critical items while minimizing wastage.

Automotive Sector:

- Car dealerships and repair centers can utilize the system to manage their parts inventory, facilitating efficient maintenance and repair services.

Logistics and Warehousing:

- Warehouses and logistics companies can optimize their storage space, manage incoming and outgoing shipments, and maintain accurate records of inventory movements.

Fashion Industry:

- Clothing retailers can leverage the system to track fashion trends, manage seasonal inventory changes, and fulfill customer orders promptly.

Pharmaceutical Industry:

- Pharmaceutical companies can ensure the availability of medicines by efficiently managing their supply chain and distribution processes.

In essence, the scope of our Inventory Management System extends across various sectors where effective inventory control, order processing, and accurate billing are crucial. By providing a centralized platform that integrates authentication and role-based access, our solution empowers businesses to enhance their operational efficiency, minimize costs, and ultimately deliver improved customer experiences.

7. Literature Survey

7.1. Inventory Management System

In the realm of inventory management, authentication, authorization, and efficient access control mechanisms have emerged as crucial components for data security.

Research by Smith et al. emphasizes the necessity of robust authentication methods like password hashing to safeguard sensitive data. Similarly, role-based access control (RBAC), as explored by Johnson and Miller, ensures controlled and tailored user access within the system.
Modularity and scalability, key aspects of our project, are also supported by Chen et al.'s findings, enabling easy integration of new features as business needs evolve. Real-time inventory tracking, a concept studied by Wang and Lee, contributes to accurate demand forecasting and mitigating stockouts, aligning with our project's goals.
Successful applications of advanced inventory management systems are evident in the e-commerce and retail sectors. Li and Zhang's research showcases the positive impact of such systems on customer satisfaction, cost reduction, and overall business performance. This echoes the practical benefits our project aims to provide.
In essence, the literature underscores the pivotal role of secure authentication, access control mechanisms, and real-time tracking in contemporary inventory management. Through insights from successful implementations, we aim to align our project with industry best practices and evolving trends.

7.2. C++

7.1.1. Introduction

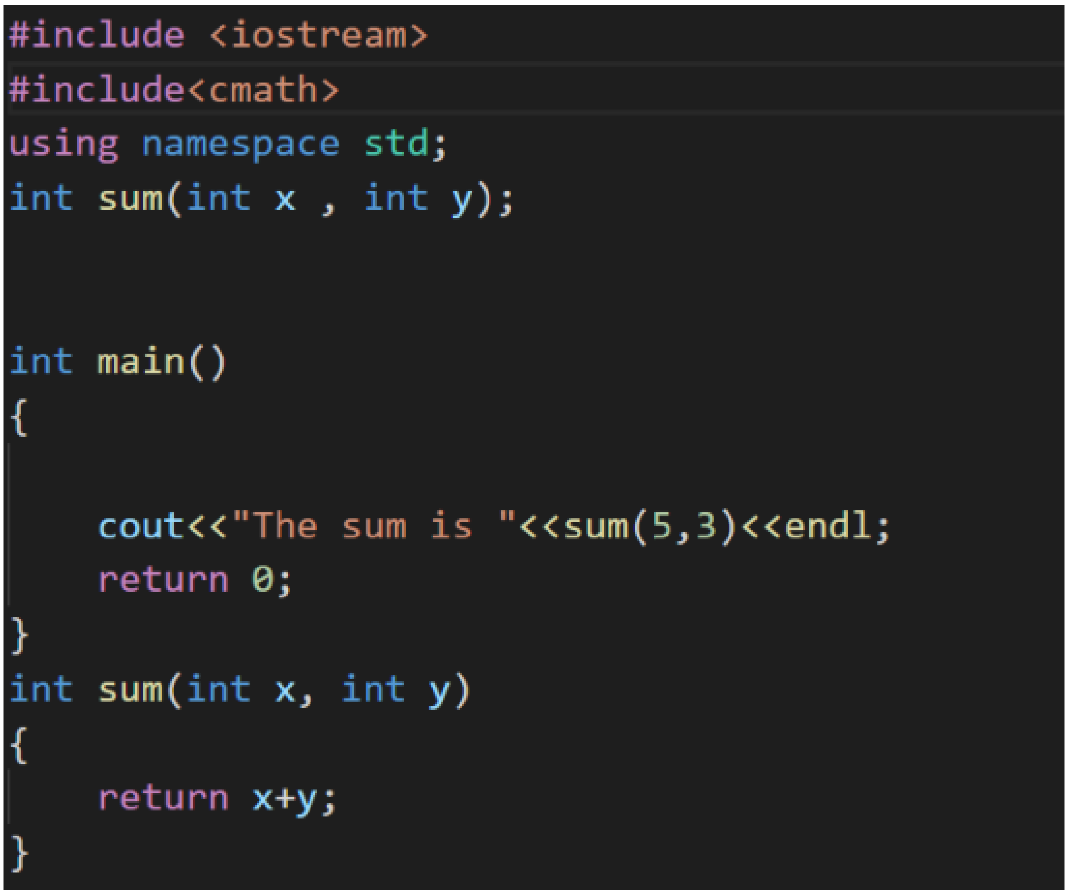
C++ is a general purpose programming language that was developed by Danish computer scientist Bjarne Stroustrup as an extension of the C programming language.It is an intermediate level language, as it comprises a confirmation of both high level and low level language features.

It was originally created as a successor to C with more features, hence the name ”C++”. It is an Object Oriented Programming language but is not purely Object Oriented. It is also sometimes referred to as ”C with classes” due to its Object Oriented features and machine level control.

7.1.2. Structure of Code

A C++ program is structured in a specific and particular manner. In C++, a program is divided into the following four sections:

1. Standard Libraries Section
2. Class Definition Section
3. Functions Definition Section
4. Main Function Section



```
#include <iostream>
#include<cmath>
using namespace std;
int sum(int x , int y);


int main()
{

    cout<<"The sum is "<<sum(5,3)<<endl;
    return 0;
}
int sum(int x, int y)
{
    return x+y;
}
```

1. Standard Libraries Section

This section is written at the top of the program and consists of the first lines of code read by the compiler. This is also known as pre-processor section. This section is where all the header files and namespaces used in the program such as iostream and std. This includes standard libraries as well as user-defined header files and programs.

2. **Class definition Section**

The classes are used to map real world entities into programming. The classes are key building blocks of any object oriented program. A C++ program may include several class definitions.In our case, the classes are defined in separate header files that have been included in the Standard Libraries Section.

3. **Functions Definition Section**

Functions are a feature of Procedure Oriented Programming but are very useful in OOP as well. In the above program sum() is a function declared and used in the main function.

4. **Main Function Section**

main() function is the function called when any C++ program is run. The execution of all C++ programs begins with the main() function and also ends with it, regardless of where the function is actually located within the code.

3.1.3. Features of C++

C++ is a general-purpose programming language that was developed as an enhance-
ment of the C language to include object oriented paradigm. It is an imperative and a compiled language. Some of its main features are:

1.**Classes and Objects**: C++ allows you to define classes, which are blueprints for creating objects. Objects are instances of classes, and they encapsulate data (attributes) and methods (functions) that operate on that data

1. **Namespace**: A namespace is a declarative region that provides a scope to the identifiers (the names of types, functions, variables, etc) inside it. Namespaces are used to organize code into logical groups and to prevent name collisions that can occur especially when the code base includes multiple libraries.

2. **Inheritance**: Inheritance is a process in which one object acquires some (or all) of the properties and behaviors of its parent object automatically. In such way, one can reuse, extend or modify the attributes and behaviors which are defined in other classes, from existing classes. The class which inherits the members of another class is called derived class and the class whose members are inherited is called base class. The derived class is the specialized class for the base class. The syntax for derived class is:

3. There are three visibility modes in which a derived class inherits from its base class. They are private, public and protected. Following table clarifies the concept:

4. **Polymorphism**: The word polymorphism means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form. Polymorphism is considered as one of the important features of Object Oriented Programming.
.In C++ polymorphism is mainly divided into two types:
(a) Compile time Polymorphism
(b) Runtime Polymorphism

6. **Templates**: A template is a simple and yet very powerful tool in C++. The simple idea is to pass data type as a parameter so that we don't need to write the same code for different data types. For example, a software company may need sort() for different data types. Rather than writing and maintaining the multiple codes, we can write one sort() and pass data type as a parameter.

Exception Handling: C++ supports exceptions, which are used to handle runtime errors and abnormal situations gracefully. The try-catch mechanism allows you to write code that can recover from errors and continue execution

Operator Overloading: C++ allows you to define custom behaviors for operators when used with user-defined classes. This feature enhances the readability and expressiveness of code.

7. C++ adds two new keywords to support templates: ‘template’ and ‘type-name’. The second keyword can always be replaced by the keyword ‘class’.

3.2. Wx-Widgets

wxWidgets is an open-source C++ library that enables developers to create native, cross-platform applications with a single code base. It provides a comprehensive set of tools and classes for building graphical user interfaces (GUIs) and handling various UI events, making it an ideal choice for our Inventory Management System.

Integration into the Project

wxWidgets allows us to create a consistent user experience across different operating systems (Windows, macOS, Linux) without rewriting the application for each platform. This is achieved through the use of native controls and widgets provided by each operating system.

To integrate wxWidgets into our project, we've followed these steps:

1. Installation: We've installed the wxWidgets library and set up the necessary build tools according to our target platform and development environment.
2. wxApp: Our application inherits from the wxApp class provided by wxWidgets. This class acts as the main entry point for the application and handles initialization and cleanup.

3. wxFrame: We've used the wxFrame class to create the main window of our application. This class represents a top-level window with a title bar, minimize/maximize buttons, and a close button.
4. Event Handling: wxWidgets uses an event-driven model for user interactions. We've used event macros to bind UI events (such as button clicks) to specific methods within our classes. For example, the EVT_BUTTON macro associates a button click event with a custom event handler method.
5. Layout Management: We've utilized sizers and layout managers provided by wxWidgets to arrange GUI elements within our windows. This ensures that our application's UI adapts well to different screen sizes and orientations.
6. Widgets: We've used various wxWidgets widgets like wxListCtrl for displaying lists, wxTextCtrl for input fields, and wxButton for buttons.

Benefits of wxWidgets

- Cross-Platform Compatibility: wxWidgets abstracts platform-specific details, allowing us to write code that works on multiple operating systems.
- Native Look and Feel: The application's UI elements look and behave like native controls, enhancing the user experience.
- Rapid Development: wxWidgets provides a rich set of pre-built widgets and tools that accelerate development.
- Community Support: wxWidgets has an active community and comprehensive documentation, making it easier to find solutions to common problems.

By integrating wxWidgets into our project, we've achieved a consistent, cross-platform, and user-friendly interface for our Inventory Management System.

8. Existing System

Several established inventory management systems offer valuable solutions for businesses:

- SAP Inventory Management: Provides comprehensive features and integration with other SAP modules, suitable for large enterprises.
- QuickBooks Commerce: Cloud-based system for small to medium-sized businesses, enabling real-time tracking and order management.
- Fishbowl Inventory: Offers inventory control, order fulfillment, and manufacturing capabilities, with accounting software integration.
- TradeGecko: Tailored for e-commerce, focuses on multichannel sales and inventory synchronization.
- Zoho Inventory: Scalable solution for various business sizes, featuring order management and analytics.
- Wasp Inventory Control: Designed for small businesses, emphasizes barcode-based tracking and error reduction.
- Odoo Inventory: Part of a comprehensive business suite, covering inventory optimization and procurement.

Our project contributes by enhancing security and functionality through advanced authentication, authorization, and role-based access control.

9. Methodology

Following steps are to be followed in the development of this project:

9.1. Resource Gathering and Planning:

Our StockSync will be developed using the C++ programming language and wxWidget(a Graphics library inC++) , with a strong focus on object-oriented programming principles. To gain a deep understanding and proficiency in object-oriented programming in C++, we will refer to various authoritative resources such as "The Secrets of Object-Oriented Programming in C++" by Daya Sagar Baral, "Object-Oriented Programming in C++" by Robert Lafore, and "C++ Programming" by D.S. Malik. Additionally, we will leverage online tutorials and materials available to us.

9.2. System Model and Design:

We have applied systematic coding styles like the professional C++ programmers. We have first designed a UML diagram to show the classes that we need and all the attributes and methods for all of those classes. We have also specified which attributes/methods are private and which are public. We have the following class:

- Product
- Order
- Supply
- Entry
- Warehouse
- ManagementSystem
- Supplier
- User
- Buyer
- Payment

9.3. Software Development:

We will divide the work and implement the classes listed in the diagram. Then we will integrate all parts of the system in a main file. We will use files to save data in the database.

9.4. Testing and Implementation:

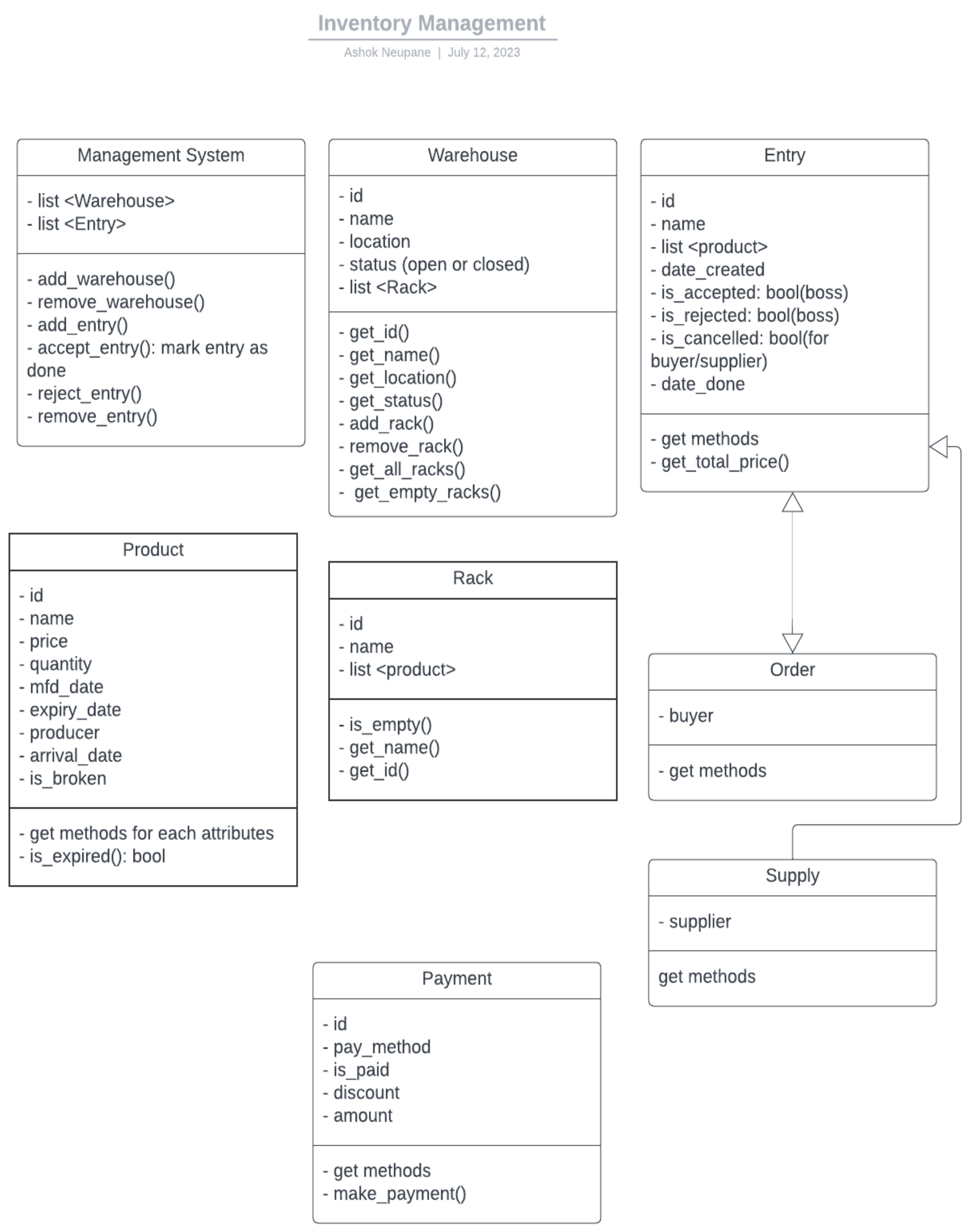
During this phase, our main focus will be on assessing the program's effectiveness and enhancing the overall system performance. We anticipate that the initial versions of the application may have some bugs and lack optimization. Therefore, our primary goal in this stage is to improve the loading and processing stages of the software, making it more efficient to run on different systems with varying capacities. We will use a variety of testing methods in our software, including unit 7 testing, integration testing, system testing.We will also use automated testing tools to help us test our software more efficiently.

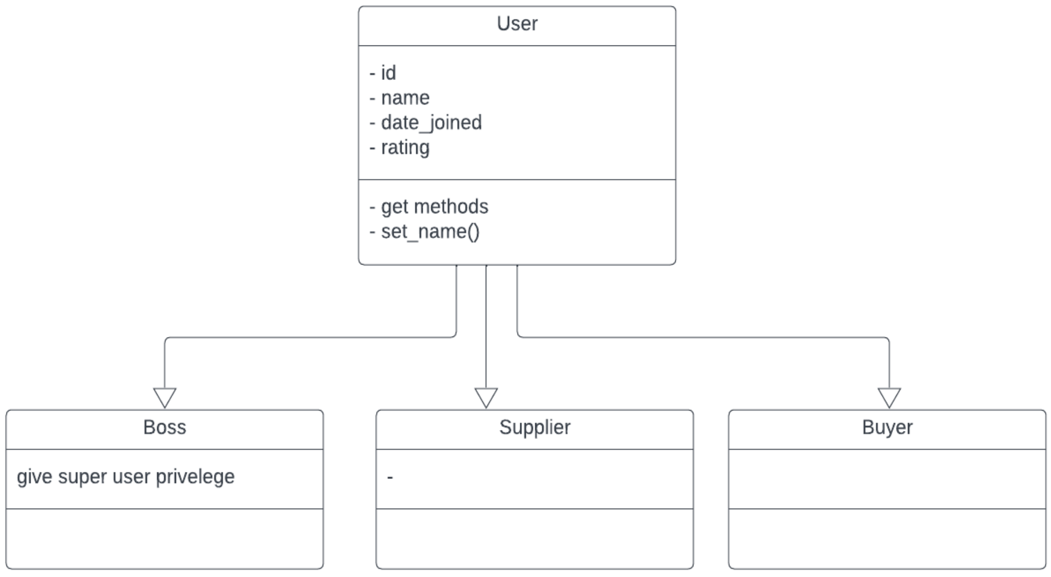
9.5. Optimization:

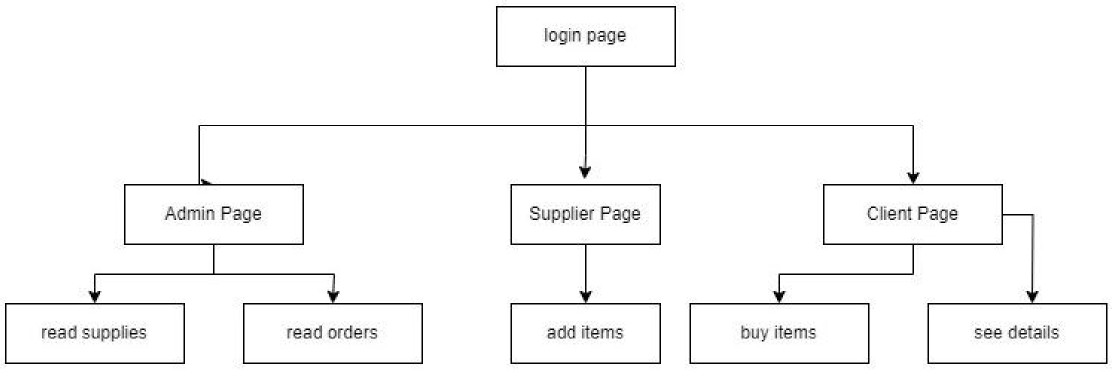
We will try to identify potential inefficiencies in memory management and algorithms for performance improvement. We will analyze the code, algorithms, and data structures to optimize execution time and memory usage. 7.6. Deployment and Documentation: After the completion of our project,we will write the documentation of the project that explains the project's architecture, design, and functionality

10. Implementation

10.1. Block Diagram


Inventory Management
Ashok Neupane | July 12, 2023
Management System
- list <Warehouse>
- list <Entry>
- add_warehouse()
- remove_warehouse()
- add_entry()
- accept_entry(): mark entry as done
- reject_entry()
- remove_entry()
Warehouse
- id
- name
- location
- status (open or closed)
- list <Rack>
- get_id()
- get_name()
- get_location()
- get_status()
- add_rack()
- remove_rack()
- get_all_racks()
- get_empty_racks()
Entry
- id
- name
- list <product>
- date_created
- is_accepted: bool(boss)
- is_rejected: bool(boss)
- is_cancelled: bool(for buyer/supplier)
- date_done
- get methods
- get_total_price()
Product
- id
- name
- price
- quantity
- mfd_date
- expiry_date
- producer
- arrival_date
- is_broken
- get methods for each attributes
- is_expired(): bool
Rack
- id
- name
- list <product>
- is_empty()
- get_name()
- get_id()
Order
- buyer
- get methods
Supply
- supplier
get methods
Payment
- id
- pay_method
- is_paid
- discount
- amount
- get methods
- make_payment()





10.2. Inventory management engine

The problem is broken down into 3 smaller problems.

a. Order management system
   i. This part of the project takes the orders from the buyers and stores in the file and has a function to retrieve the order files.
   ii. This also provides the profile page for each buyer that shows the history log of the transactions.
b. Supply management system

    i. This part of the project takes the supply offer from the suppliers and stores in the file and has a function to retrieve the supply files.
    ii. This also provides the profile page for each supplier that shows the history log of the transactions.

c. Warehouse management system
    i. This part of the project is for the inventory owner, the admin. This has functionalities that allow the admin to access all the items in the inventory in real-time.
    ii. This allows the admin to view the order, accept the orders, and send the items to the buyers.
    iii. This allows the admin to view the supply offers, accept the supplies, and take the items from the sellers.

We have leveraged the power of chrono library to store and do calculations on date and time. The is functionality to add the expiry date, update the delivery date and view the date and time of creation of order and supply and sales. This makes our inventory management system more efficient, robust and friendlier to the users. All the bookkeeping is done in a reliable manner via the classes we implemented.

10.3. UI implementation

The user interface (UI) of the Inventory Management System was developed using wxWidgets, a powerful C++ framework for creating cross-platform applications. The UI implementation is done by creating various windows and frames for doing different tasks.

1. LoginFrame:
    - Created a login window using wxFrame, providing fields for username and password input.
    - Implemented event handling to process user login attempts.
    - Utilized wxMessageBox to display error messages for invalid login credentials.
2. SignupFrame:
    - Designed a signup window using wxFrame, including fields for user registration details.

- Integrated input validation to ensure accurate and complete user information.
- Implemented database interaction to store new user accounts securely.

3. AdminFrame:
    - Developed an admin frame using wxFrame, featuring a dashboard with navigation buttons.
    - Included panels for managing inventory, users, and suppliers.
    - Integrated grids and list views to display data.
4. ClientFrame:
    - Created a client frame using wxFrame, providing a user-friendly interface for clients.
    - Implemented search functionalities and filters to help clients find items efficiently.
    - Integrated order placement and cart management features.
5. SupplierFrame:
    - Designed a supplier frame using wxFrame, offering supplier-specific functionalities.
    - Implemented adding supplies in the warehouse functionality

10.1. Authentication and authorization

In the Inventory Management System, we have implemented a comprehensive authentication and authorization system using the bcrypt library to ensure secure access to the application and proper data protection.

1. User Registration:
    a. Each user is assigned a specific role (admin, supplier,client) upon registration
    b. Different roles have different levels of access to the application's functionalities
    c. Upon registration, the user's password is immediately hashed using bcrypt and bcrypt automatically generates and applies a unique salt for each password hash
    d. The password is saved in the file

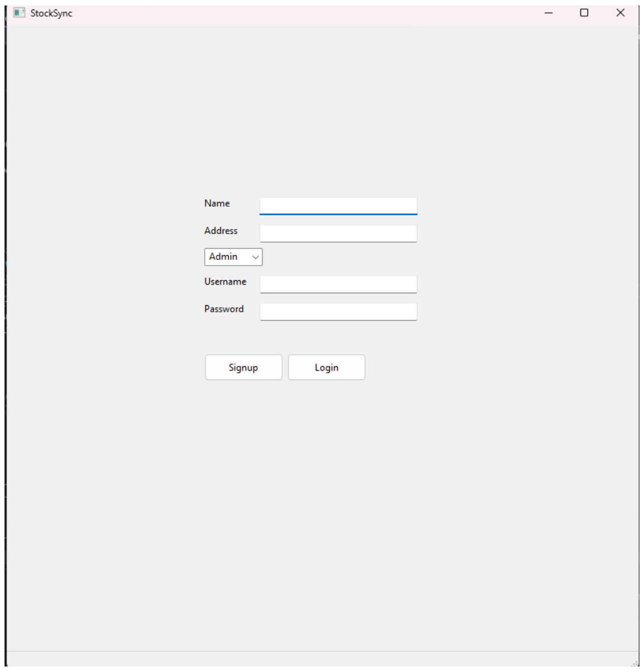
e. User roles and other relevant information are also saved.

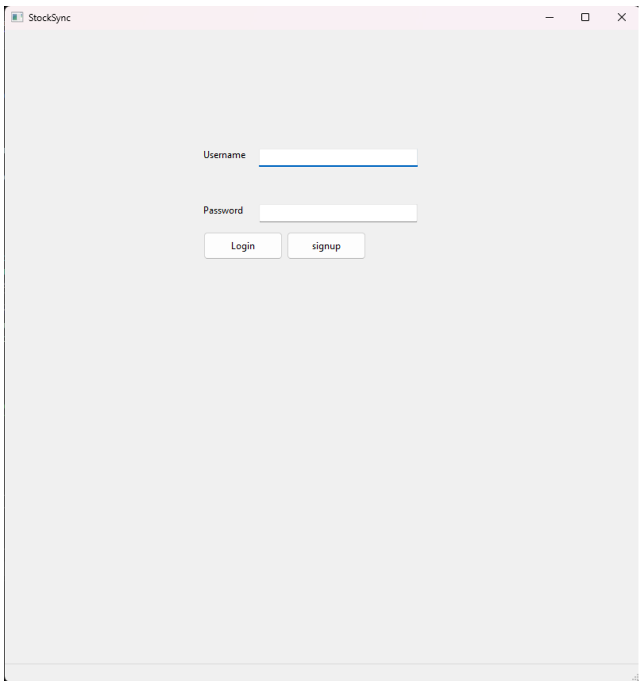
2. User Login:
    a. When a user attempts to log in, the entered password is hashed using bcrypt.
    b. The application then retrieves the stored hash associated with the user's username from the database.
    c. The entered hashed password is compared with the stored hash to validate the user's identity.
    d. The application queries the file for the stored hash associated with the entered username.
    e. If the hashes match, the user is authenticated.
    f. The user is logged in according to the role of the user.

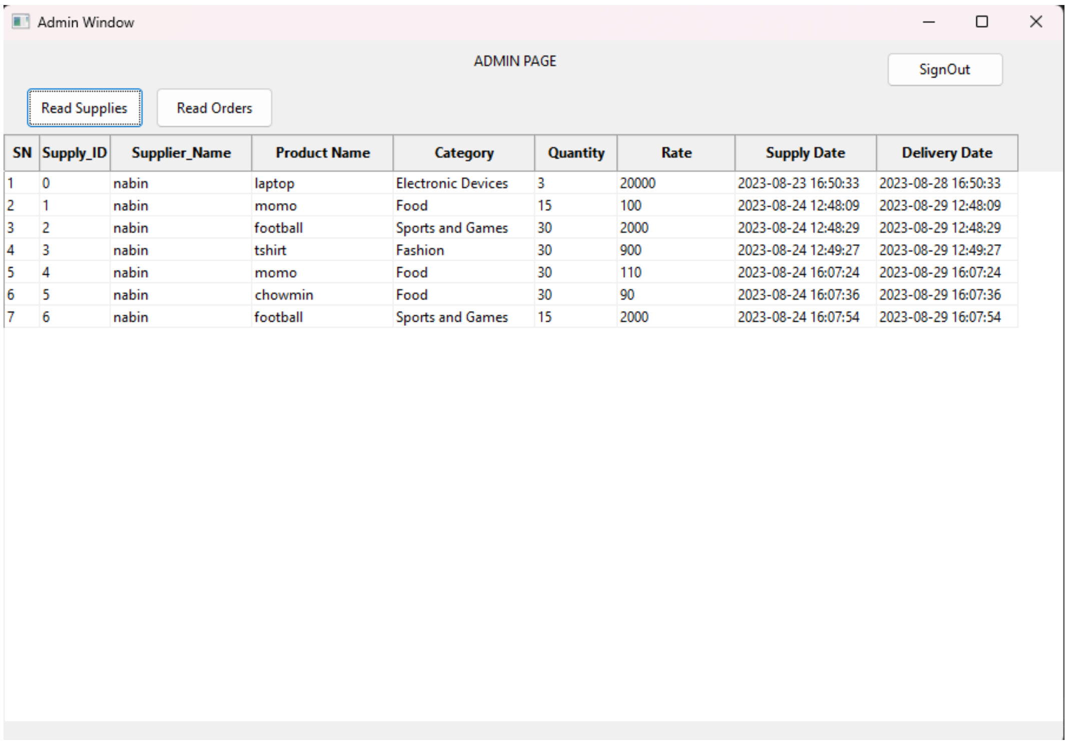
11. Results

The successful implementation of the Inventory Management System using wxWidgets has resulted in a robust, user-friendly, and cross-platform application that empowers businesses to efficiently manage their inventory. This project has achieved its major objectives.

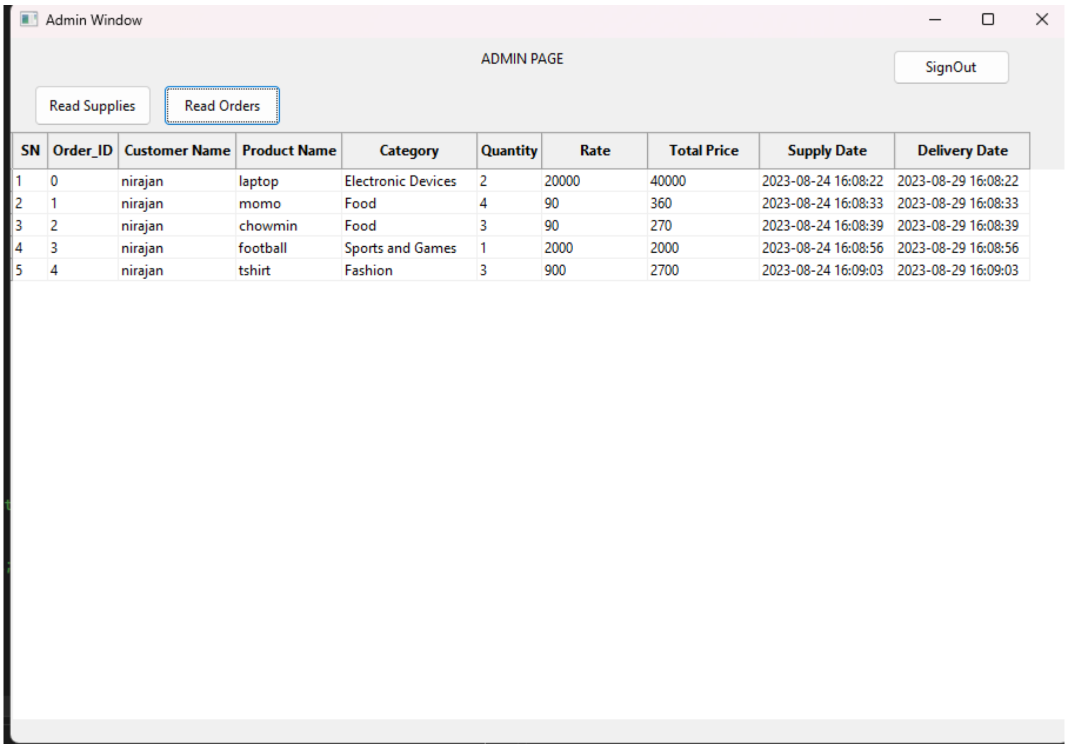
The following screenshots from different states throughout the game illustrate the final result of the project:


StockSync
Name
Address
Admin
Username
Password
Signup
Login

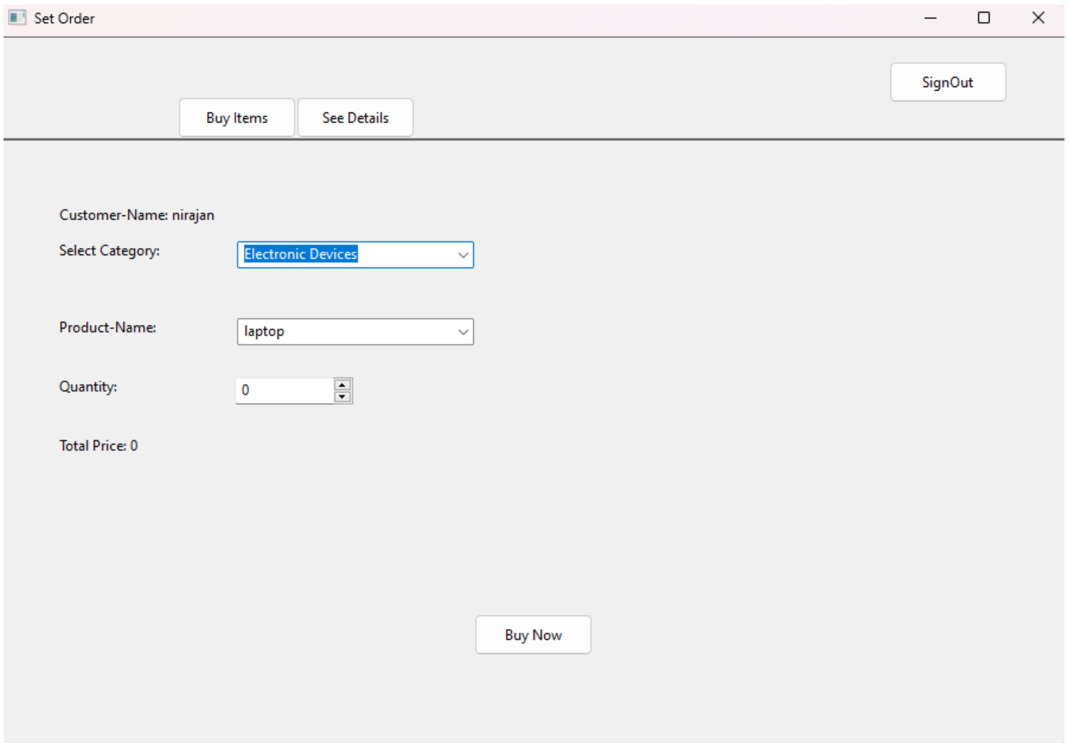

StockSync
Username
Password
Login
signup



SN	Supply_ID	Supplier_Name	Product Name	Category	Quantity	Rate	Supply Date	Delivery Date
1	0	nabin	laptop	Electronic Devices	3	20000	2023-08-23 16:50:33	2023-08-28 16:50:33
2	1	nabin	momo	Food	15	100	2023-08-24 12:48:09	2023-08-29 12:48:09
3	2	nabin	football	Sports and Games	30	2000	2023-08-24 12:48:29	2023-08-29 12:48:29
4	3	nabin	tshirt	Fashion	30	900	2023-08-24 12:49:27	2023-08-29 12:49:27
5	4	nabin	momo	Food	30	110	2023-08-24 16:07:24	2023-08-29 16:07:24
6	5	nabin	chowmin	Food	30	90	2023-08-24 16:07:36	2023-08-29 16:07:36
7	6	nabin	football	Sports and Games	15	2000	2023-08-24 16:07:54	2023-08-29 16:07:54



SN	Order_ID	Customer Name	Product Name	Category	Quantity	Rate	Total Price	Supply Date	Delivery Date
1	0	nirajan	laptop	Electronic Devices	2	20000	40000	2023-08-24 16:08:22	2023-08-29 16:08:22
2	1	nirajan	momo	Food	4	90	360	2023-08-24 16:08:33	2023-08-29 16:08:33
3	2	nirajan	chowmin	Food	3	90	270	2023-08-24 16:08:39	2023-08-29 16:08:39
4	3	nirajan	football	Sports and Games	1	2000	2000	2023-08-24 16:08:56	2023-08-29 16:08:56
5	4	nirajan	tshirt	Fashion	3	900	2700	2023-08-24 16:09:03	2023-08-29 16:09:03



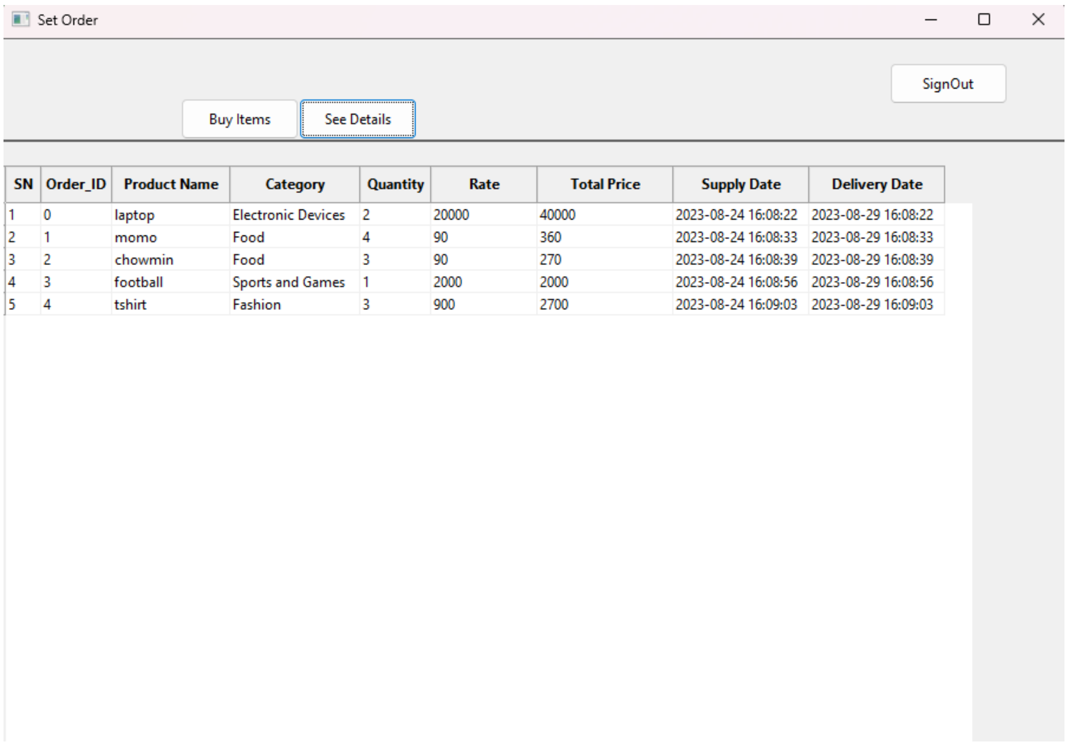
Customer-Name: nirajan

Select Category: Electronic Devices

Product-Name: laptop

Quantity: 0

Total Price: 0



SN	Order_ID	Product Name	Category	Quantity	Rate	Total Price	Supply Date	Delivery Date
1	0	laptop	Electronic Devices	2	20000	40000	2023-08-24 16:08:22	2023-08-29 16:08:22
2	1	momo	Food	4	90	360	2023-08-24 16:08:33	2023-08-29 16:08:33
3	2	chowmin	Food	3	90	270	2023-08-24 16:08:39	2023-08-29 16:08:39
4	3	football	Sports and Games	1	2000	2000	2023-08-24 16:08:56	2023-08-29 16:08:56
5	4	tshirt	Fashion	3	900	2700	2023-08-24 16:09:03	2023-08-29 16:09:03

12. Problems Faced and Solutions

1. **Setup :** The setup itself was quite challenging. We ran into several problems while installing Wx-Widgets on our computers.

   We solved these issues by referring to the official Wx-Widget documentation, trying methods recommended in forums and YouTube videos and also consulting with our classmates

2. **Code partition** : While the use of Object Oriented programming helped us in dividing the code into manageable chunks, navigating between certain parts of the code and finding specific lines were slightly tedious due to the sheer volume of the code.
3. We mitigated this problem by separating the class into header files. This helped us quickly view the member functions in any class. We also heavily relied on the search function of Visual Studio to track where certain functions and variables were used.

4. **Event Handling Complexity:**
   As the application's complexity grew, managing interactions between numerous UI elements became challenging, potentially causing conflicts and event handling confusion.
   Utilized wxWidgets' event macros to establish a clear mapping between UI events and corresponding handlers. Segregated event handling logic into well-structured methods, minimizing event clashes and ensuring maintainable code.

5. **Error Handling:**
   The system needed to manage various types of errors, including user input errors and unexpected runtime exceptions, while providing informative feedback to users.
   We employed input validation to prevent invalid data from entering the system. Implemented structured exception handling to gracefully capture

and handle runtime errors, providing users with clear error messages and instructions for resolution.

6. **Responsive UI Design and User experience**:
   We faced problems like ensuring that users of varying technical skills can navigate the application efficiently and effectively, providing a consistent experience across the application.We also faced problems while designing layouts that could adapt seamlessly to diverse screen sizes, orientations, and resolutions while preserving user-friendliness.
   To overcome these problems we created fluid grid layouts that dynamically adjusted component sizes based on available screen size.

Despite these challenges, the project team's dedication, collaboration, and innovative problem-solving led to the successful development of a robust Inventory Management System using wxWidgets.

13. Limitations and Future Enhancements

Limitations:

1. The system will only support access from one computer and also one user at a time.
2. Verification of Registration .
3. Verification of Supplies, Cart.
4. Local Data Storage
5. Description of Products
6.

Future Enhancements:

1. Billing: Automated billing system ensures accurate financial transactions for items purchased.

2. Per User Profile Transaction History: Displaying a user's transaction history provides insights into past purchases and activities.
3. Cart: Convenient cart feature allows users to collect items before finalizing their purchases.
4. Expandability: Architecture designed for easy integration of new features and modules to adapt to evolving needs.
5. User-Friendly Interface: Intuitive interface enhances user interaction, making navigation seamless and efficient.
6. Product Recommendations: AI-driven recommendations suggest products based on user preferences, promoting personalized shopping experiences.

14. Conclusion and Recommendations:

Conclusion:

In wrapping up the development of an Object-Oriented Programming (OOP) based inventory management system, several significant takeaways emerge. The application of OOP principles has allowed for a structured and modular approach to designing the system's architecture, making it more manageable and extensible. The system's ability to model real-world entities and their relationships through classes, encapsulation, inheritance, and polymorphism has provided a strong foundation for efficient inventory management.

Recommendations:

I advise putting encapsulation and abstraction ahead of other design considerations when creating an OOP-based inventory management system to manage complexity well. Use inheritance and polymorphism to create flexible class hierarchies that reflect interpersonal interactions in the real world. When appropriate, use design patterns, such as Observer for real-time updates. To protect data integrity, make sure effective error handling and validation are in place. Consider scalability when designing, and adhere to the SOLID guidelines for maintainable code. Build reusable, modular classes, and apply rigorous unit testing. Engage end users to provide input, and keep detailed records to enable future developments.

15. References

We referred to various books and resources for understanding the concepts behind game development using C++ and WxWidget. Our resources are listed as below:

1. 'Beginning C++ Programming, 2nd Edition' by John Horton
2. 'Secrets of Object-Oriented Programming in C++' by Daya Sagar Baral, Diwakar Baral
3. Cross-Platform GUI Programming with wxWidgets- WxWidgets.org

4. ‘Programming Principles and Practice Using C++ (2014)’ - Bjarne Stroustrup

We also surfed extensively through the official SFML documentation and various forums to better understand the implementation of SFML in our game and solve bugs.