

**Object Oriented Programming**

Assignment 11

OOP Project Deliverable 2

**Online Shopping System**

|  |  |
| --- | --- |
| **Submitted by:** | **Enrollment NO:** |
| Muhammad Abdul Rahman | 01-135231-053 |

Semester/class:

BS-IT-2A

Date: 12-05-2024

Day: Sunday

Submitted to:

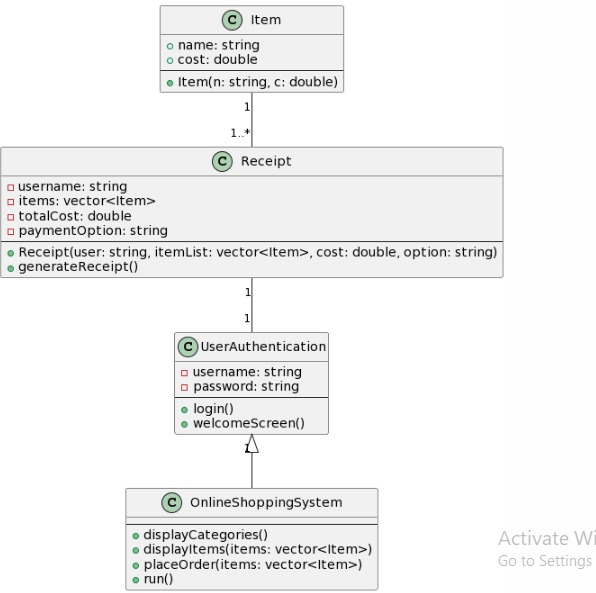
Mam NABIA KHALID

**Project: Online Shopping System**

**Introduction:**

In this project, we've developed a console-based Online Shopping System implemented in C++. The system provides users with an interface to browse various product categories, select items to purchase, and generate a receipt with the selected payment option. The project employs object-oriented programming (OOP) principles to create a modular and user-friendly shopping experience.

**Uml Diagram**:



**Mapping of the Functional Requirements with the Pillars of OOP:**

1. **Encapsulation**
   * **Classes and Data Encapsulation**: We encapsulated the shopping system logic within classes such as **Item**, **Receipt**, **UserAuthentication**, and **OnlineShoppingSystem**. Each class contains its own set of data members and member functions, hiding the internal details of how each operation is performed.
   * **Example**: The **Item** class encapsulates item-related functionality, including the item's name and cost. The **Receipt** class manages the receipt generation process, including storing the username, purchased items, total cost, and payment option.
2. **Inheritance**
   * **Class Hierarchy and Reuse**: We used inheritance to extend the functionality of the **UserAuthentication** class into the **OnlineShoppingSystem** class. This allows us to reuse the authentication functionality while adding additional features specific to the shopping system, such as displaying categories and handling orders.
   * **Example**: The **OnlineShoppingSystem** class inherits from **UserAuthentication**, enabling it to use the login functionality provided by the parent class while introducing new methods for displaying categories and processing purchases.
3. **Polymorphism**
   * **Dynamic Behavior Based on User Interaction**: Polymorphism is evident in the way we handle different product categories and items. The system dynamically displays different sets of items based on the user's choice of category and processes the selected items accordingly.
   * **Example**: The **placeOrder** function exhibits polymorphic behavior by handling the addition of different items to the cart based on user input, and then generating a receipt that reflects the user's choices.
4. **Abstraction:**
   * **Simplified User Interface**: Abstraction is demonstrated in the way we hide the implementation details of the shopping and authentication mechanics from the users. Users interact with the system through simple interfaces such as category selection, item addition, and payment option selection. The underlying complexities of item management, order processing, and receipt generation are abstracted away.
   * **Example**: The **displayCategories** and **displayItems** methods provide a simple interface for users to view available categories and items, while the complex logic of handling these operations is encapsulated within the respective classes.

**Conclusion:**

In conclusion, our Online Shopping System project showcases the principles of object-oriented programming (OOP) by implementing encapsulation, inheritance, polymorphism, and abstraction. Through effective use of classes and objects, we've created a modular and extensible system that allows for easy maintenance and future enhancements. Our system provides an efficient and enjoyable shopping experience for users while demonstrating the power and flexibility of OOP concepts in software development.