Topic : Online Market Store

Group no : MLB_PG_01.02_15_OnlineMarketStore

Campus : Malabe

Submission Date : 20/05/2022

We declare that this is our own work and this Assignment does not incorporate without acknowledgment any material previously submitted by anyone else in SLIIT or any other university/Institute. And we declare that each one of us equally contributed to the completion of this Assignment.

Registration No	**Name**	**Contact Number**
IT18139754	Ranathunga D.C	0716617324
IT18195408	Rupasinghe T.H.K	0768405205

Exercise 1

System Requirements for Online market store

1. **All the users can search an item in the store**
2. **The customer needs to register using their details such as name, address, and phone.**
3. **Once the user registers, the customer can view their profile, delete the profile, or edit profile details.**
4. **The administrator also needs to log in to the system using their credentials before doing the activities.**
5. **The online store administrator can add new items to the store, restock, and generate a list of items that need to be restocked.**
6. **A customer can place an order from the online store, and it consists of multiple items.**
7. **The customer can see the status of the orders placed, and get a list of previous orders made**
8. **The customer specifies a payment method (credit card, debit card, PayPal) for each order.**
9. **Once the customer confirms the order and the payment is validated the order is placed and items are updated.**
10. **Then the placed items are delivered to the address given by the customer.**

Noun and Verb Analysis

- Nouns in Red color
- Verbs in Blue color

1. **All the users can search for an item in the store**
2. **The customer needs to register using their details such as name, address, and phone.**
3. **Once the user registers, the customer can view their profile, delete the profile, or edit profile details.**
4. **The administrator also needs to log in to the system by providing their credentials before doing the activities.**
5. **The online store administrator can add new items or remove items in the store, restock, and generate a list of items that need to be restocked.**
6. **A customer can place an order from the online store, and it consists of multiple items.**
7. **The customer can see the status of the orders placed, and get a list of previous orders made**
8. **The customer specifies a payment method (credit card, debit card, PayPal) for each order.**
9. **Once the customer confirms the order and the payment is validated the order is placed and items are updated.**
10. **Then the placed items are delivered to the address given by the customer.**
11. **A list of previous orders and a list of item reports need to be generated.**

Identified classes using noun verb analysis

Nouns -:

Users

Item

Store

Customer

Profile

Name

Address

Phone

Administrator

System

Order

Status

List

Payment

Credit card

Debit card

Paypal

Identified classes -:

Customer

Administrator

Item

Order

Payment

Report

Reasons for Rejecting Other Nouns

Users - Redundant

Store - An event or an operation

Name - An attribute

Address - An attribute

Phone - An attribute

Profile - Outside scope of the system

System - Outside scope of the system

Credit card - An attribute

Debit card - An attribute

Paypal - An attribute

CRC cards for Online Market store

Customer	
Responsibilities:	Collaborations:
Register as a customer	
Log in to the system	
Search items	Item
View profile	
Delete profile	
Edit profile	
Place an order	Order
Make payments	Payment

Administrator	
Responsibilities:	Collaborations:
Log in to the system	
Remove items	Item
Check reports	Reports

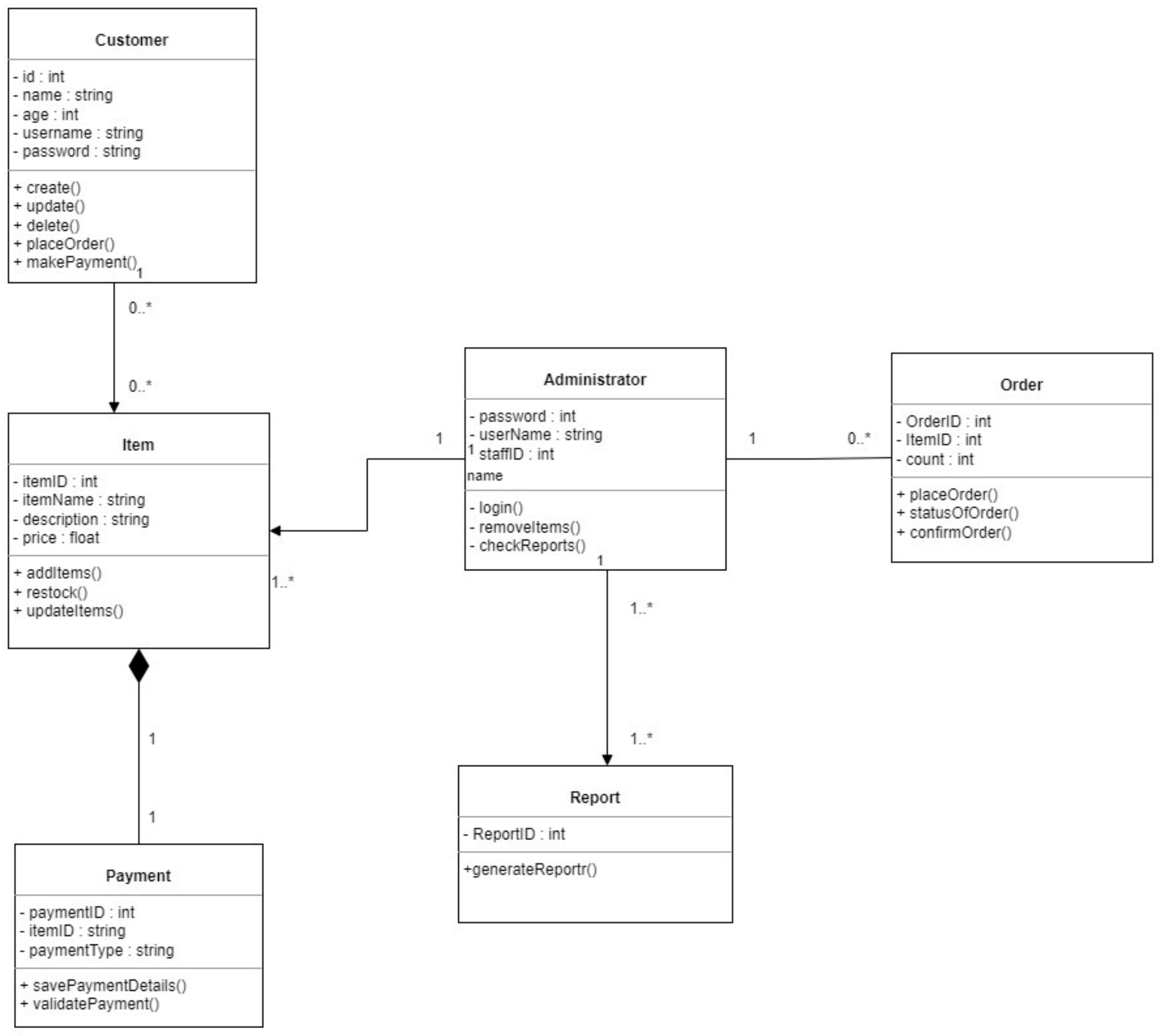
Item	
Responsibilities:	Collaborations:
Add items	
Restock	
Update items	

Order	
Responsibilities:	Collaborations:
Place orders	
Status of order	
Confirm order	Payment

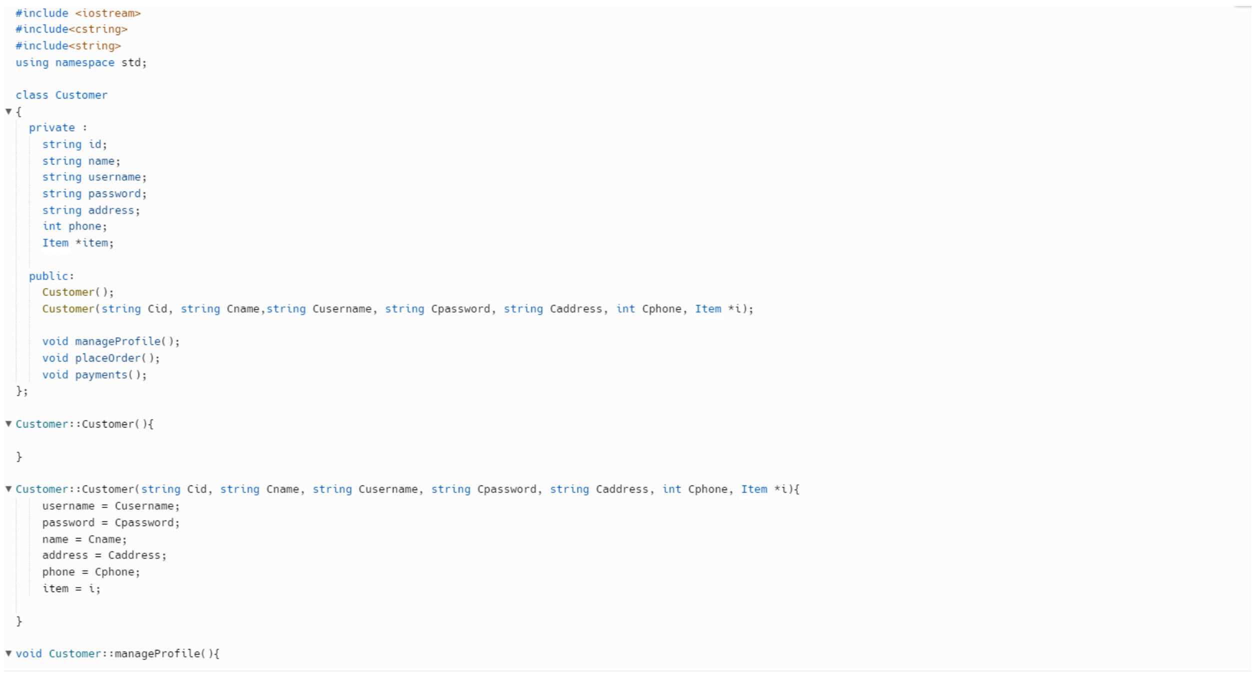
Payment	
Responsibilities:	Collaborations:
Save payment details	
Validate payments	

Report	
Responsibilities:	**Collaborations:**
List of restocking items	**Item**
List of the previous order	**Order, Customer**

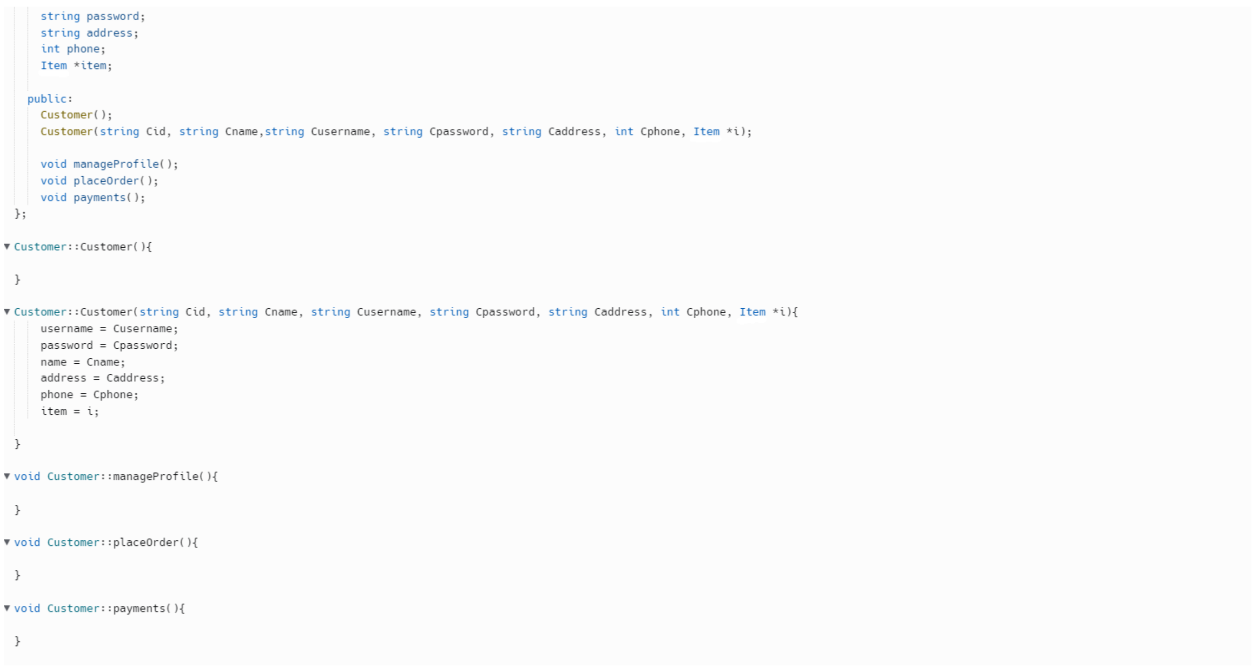
Class Diagram for the Online Land Sales System



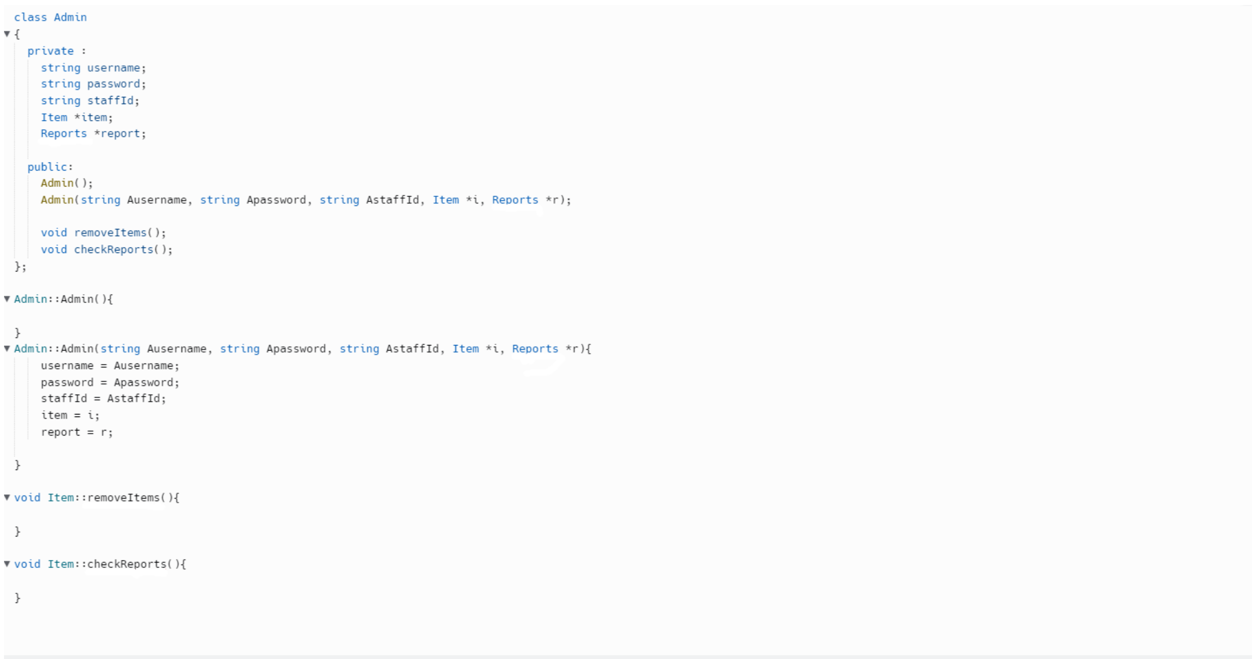
Coding for the Classes in Class Diagram



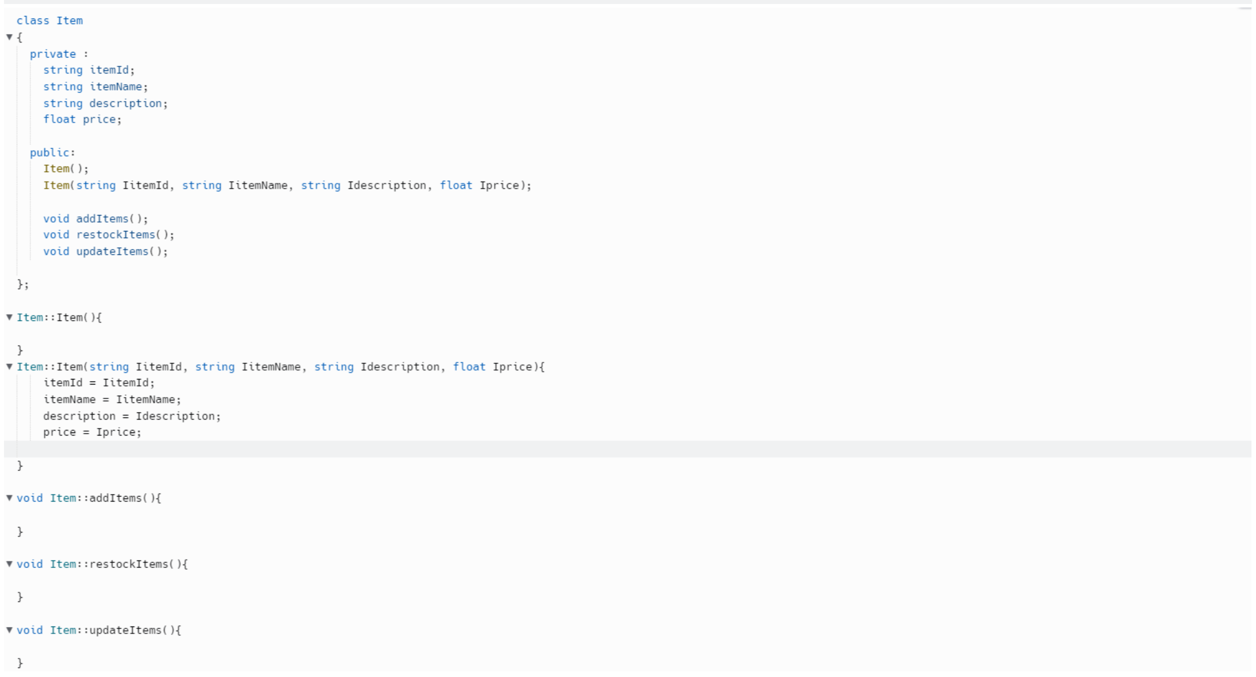
```cpp
#include <iostream>
#include<cstring>
#include<string>
using namespace std;

class Customer
{
  private :
    string id;
    string name;
    string username;
    string password;
    string address;
    int phone;
    Item *item;

  public:
    Customer();
    Customer(string Cid, string Cname,string Cusername, string Cpassword, string Caddress, int Cphone, Item *i);

    void manageProfile();
    void placeOrder();
    void payments();
};

Customer::Customer(){

}

Customer::Customer(string Cid, string Cname, string Cusername, string Cpassword, string Caddress, int Cphone, Item *i){
    username = Cusername;
    password = Cpassword;
    name = Cname;
    address = Caddress;
    phone = Cphone;
    item = i;

}

void Customer::manageProfile(){
```



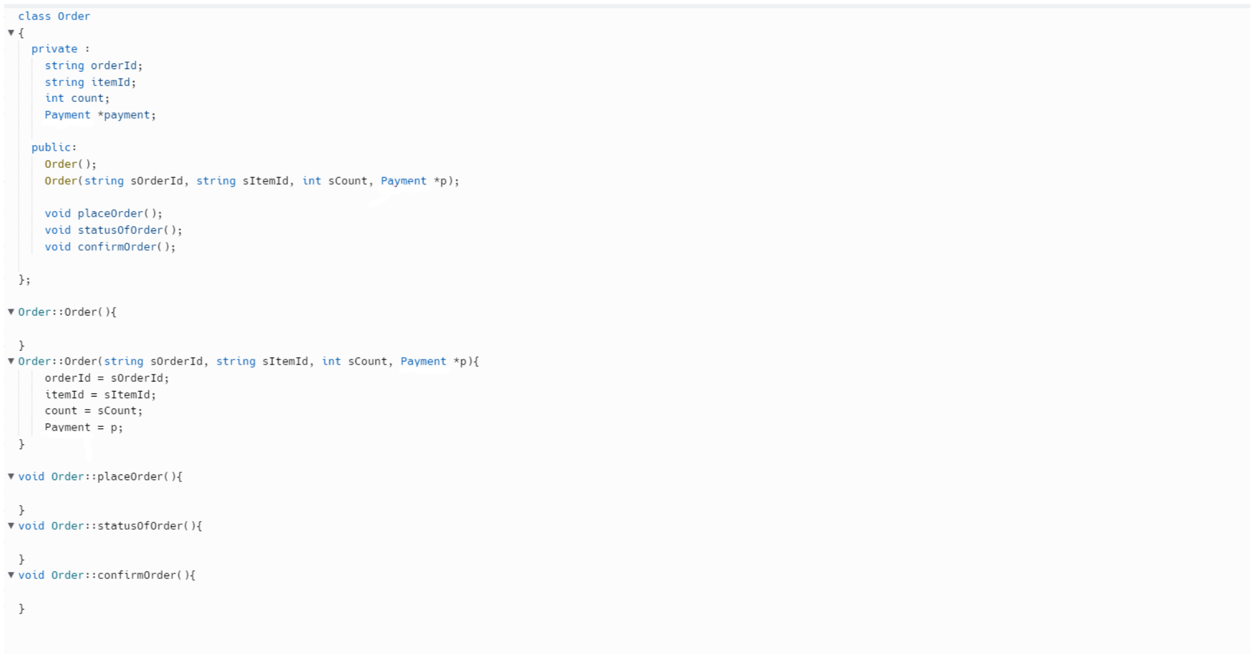
```cpp
    string password;
    string address;
    int phone;
    Item *item;

  public:
    Customer();
    Customer(string Cid, string Cname,string Cusername, string Cpassword, string Caddress, int Cphone, Item *i);

    void manageProfile();
    void placeOrder();
    void payments();
};

Customer::Customer(){

}

Customer::Customer(string Cid, string Cname, string Cusername, string Cpassword, string Caddress, int Cphone, Item *i){
    username = Cusername;
    password = Cpassword;
    name = Cname;
    address = Caddress;
    phone = Cphone;
    item = i;

}

void Customer::manageProfile(){

}

void Customer::placeOrder(){

}

void Customer::payments(){

}
```



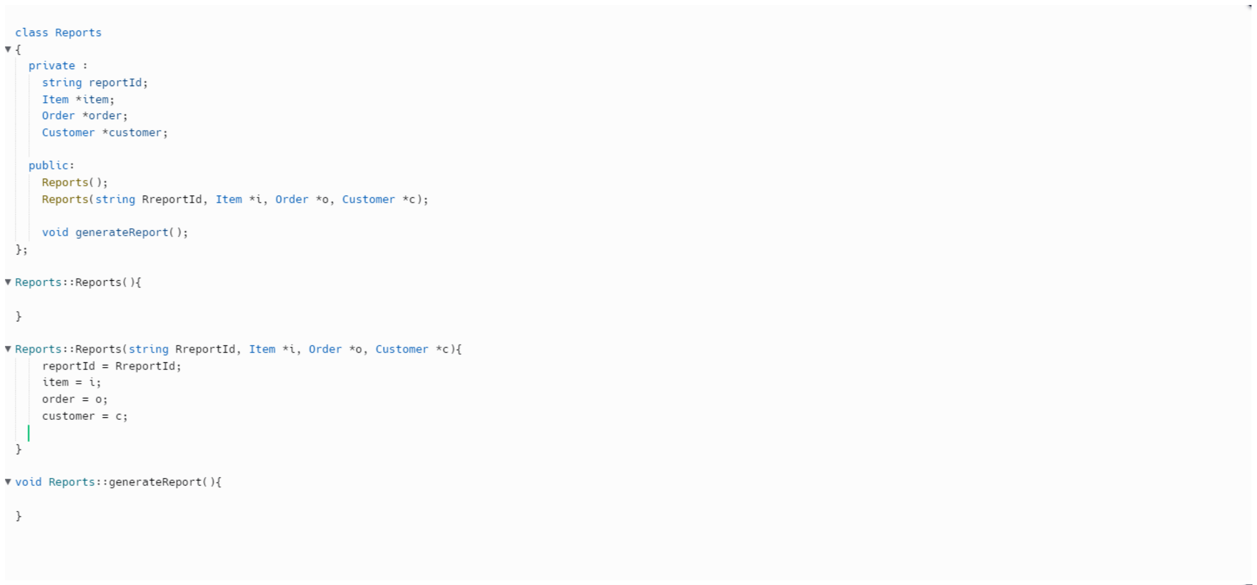
```
class Admin
{
  private :
    string username;
    string password;
    string staffId;
    Item *item;
    Reports *report;

  public:
    Admin();
    Admin(string Ausername, string Apassword, string AstaffId, Item *i, Reports *r);

    void removeItems();
    void checkReports();
};

Admin::Admin(){

}
Admin::Admin(string Ausername, string Apassword, string AstaffId, Item *i, Reports *r){
    username = Ausername;
    password = Apassword;
    staffId = AstaffId;
    item = i;
    report = r;

}

void Item::removeItems(){

}

void Item::checkReports(){

}
```



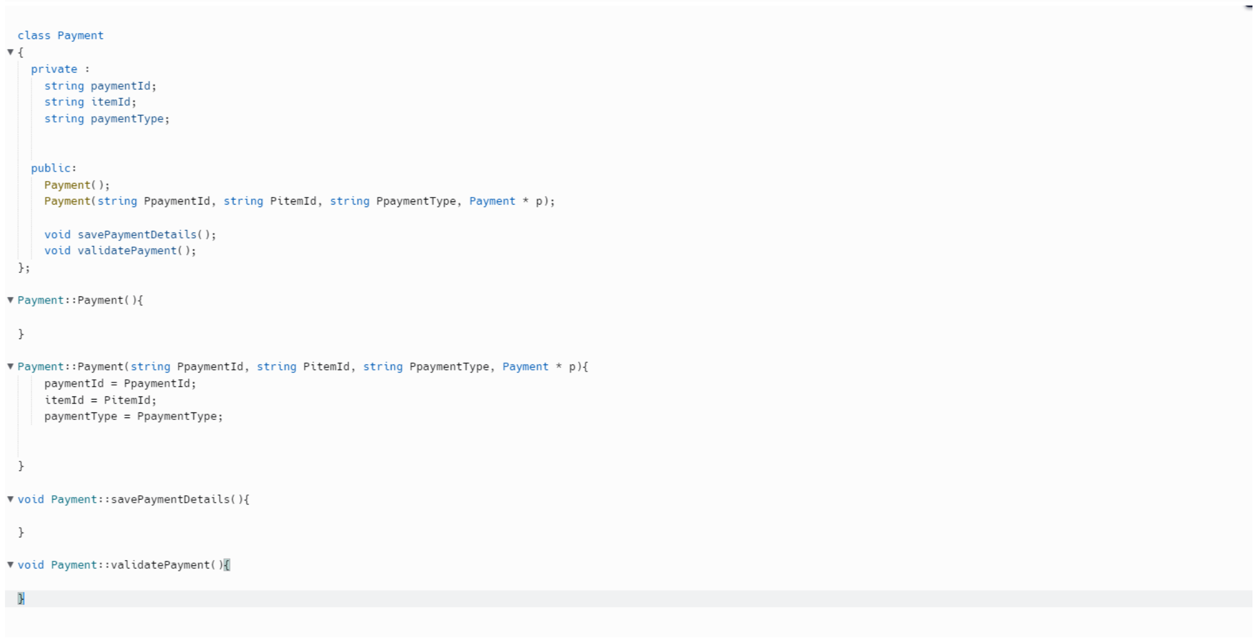
```
class Item
{
  private :
    string itemId;
    string itemName;
    string description;
    float price;

  public:
    Item();
    Item(string IitemId, string IitemName, string Idescription, float Iprice);

    void addItems();
    void restockItems();
    void updateItems();

};

Item::Item(){

}
Item::Item(string IitemId, string IitemName, string Idescription, float Iprice){
    itemId = IitemId;
    itemName = IitemName;
    description = Idescription;
    price = Iprice;

}

void Item::addItems(){

}

void Item::restockItems(){

}

void Item::updateItems(){

}
```



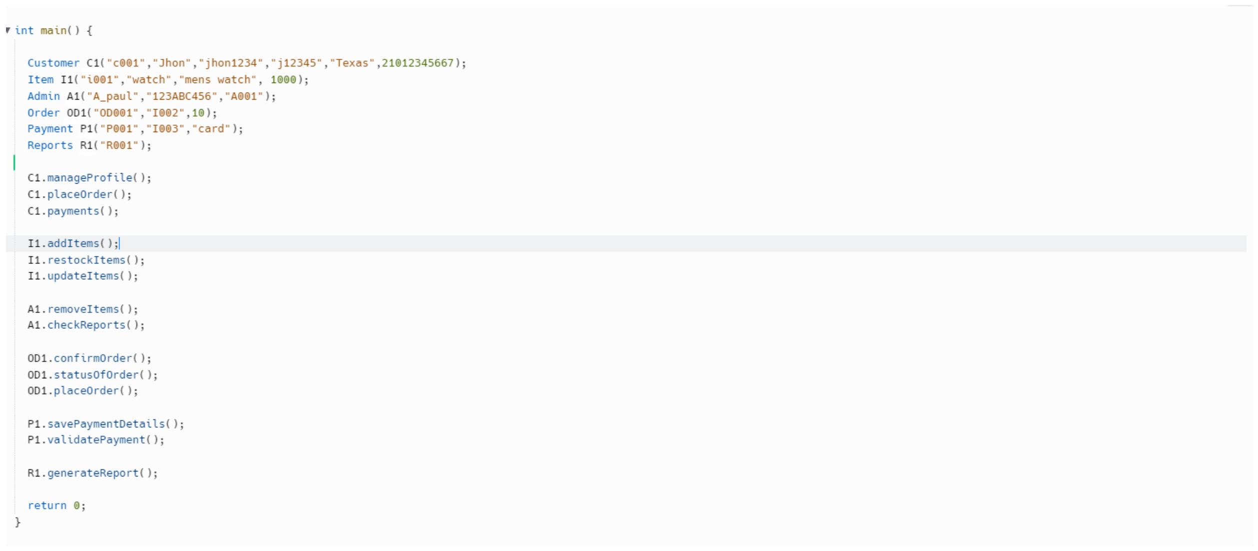
```cpp
class Order
{
  private :
    string orderId;
    string itemId;
    int count;
    Payment *payment;

  public:
    Order();
    Order(string sOrderId, string sItemId, int sCount, Payment *p);

    void placeOrder();
    void statusOfOrder();
    void confirmOrder();

};

Order::Order(){

}
Order::Order(string sOrderId, string sItemId, int sCount, Payment *p){
    orderId = sOrderId;
    itemId = sItemId;
    count = sCount;
    Payment = p;
}

void Order::placeOrder(){

}
void Order::statusOfOrder(){

}
void Order::confirmOrder(){

}
```



```
class Reports
{
  private :
    string reportId;
    Item *item;
    Order *order;
    Customer *customer;

  public:
    Reports();
    Reports(string RreportId, Item *i, Order *o, Customer *c);

    void generateReport();
};

Reports::Reports(){

}

Reports::Reports(string RreportId, Item *i, Order *o, Customer *c){
    reportId = RreportId;
    item = i;
    order = o;
    customer = c;

}

void Reports::generateReport(){

}
```



```
class Payment
{
  private :
    string paymentId;
    string itemId;
    string paymentType;


  public:
    Payment();
    Payment(string PpaymentId, string PitemId, string PpaymentType, Payment * p);

    void savePaymentDetails();
    void validatePayment();
};

Payment::Payment(){

}

Payment::Payment(string PpaymentId, string PitemId, string PpaymentType, Payment * p){
    paymentId = PpaymentId;
    itemId = PitemId;
    paymentType = PpaymentType;


}

void Payment::savePaymentDetails(){

}

void Payment::validatePayment(){

}
```



```cpp
int main() {

    Customer C1("c001","Jhon","jhon1234","j12345","Texas",21012345667);
    Item I1("i001","watch","mens watch", 1000);
    Admin A1("A_paul","123ABC456","A001");
    Order OD1("OD001","I002",10);
    Payment P1("P001","I003","card");
    Reports R1("R001");

    C1.manageProfile();
    C1.placeOrder();
    C1.payments();

    I1.addItems();
    I1.restockItems();
    I1.updateItems();

    A1.removeItems();
    A1.checkReports();

    OD1.confirmOrder();
    OD1.statusOfOrder();
    OD1.placeOrder();

    P1.savePaymentDetails();
    P1.validatePayment();

    R1.generateReport();

    return 0;
}
```