Topic : Online Pharmacy Portal

Group no :MLB_01.01_03

Campus : Malabe

Submission Date :

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**
IT21068232	Wijesundara D.A.R	0767637175
IT21057410	Warushavithana T.M	775548379
IT21077524	Pathirana.A.P.C.E	768411387
IT21069390	Madarasinghege.S.U	773810055
IT21071652	Hewapathiranage T.K	702512947

Table of Contents

1.Worked Sections

IT Number	Name	Worked Section
IT21068232	Wijesundara.D.A.R	• Analysed and designed the user, customer, staff, pharmacist and admin classes. • Modeled CRC cards. • Modeled the relationships of class diagram. • I implemented the above mentioned classes.
IT21057410	Warushavithana T.M	• I have analysed the classes payment, cash, credit and discount individually. • I discussed with the other group members to how to draw our CRC cards and to build up our scenario. • I coded the above mentioned classes myself.
IT21077524	Pathirana.A.P.C.E	• I helped to identify cart and order classes. • As well as I implemented the cart class and order class and implement objects of that classes in main program.
IT21069390	Madarasinghege.S.U	• I implemented the Report class. • Helped to in identifying classes.
IT21071652	Hewapathiranage T.K	• Discovered errors in the class diagram. • I worked together with my team members to create CRC cards and class diagram. • I coded product class including medicine and medical equipment classes.

2.Description

Our system emphasize and focuses on registered customers, pharmacist and system admins as users. Unregistered customer can register to the system. After registration guest becomes a registered member. Registered member can login to the system and search for medicine, if desired he can add the items to the cart and purchase the items. Registered customers are eligible for discounts, registered customer can enter the promo code at the checkout to claim the discount, Customer can pay using credit or debit card or select cash on delivery option. Customer order will be updated to order list, and Head pharmacist assigns Assistant pharmacists to prepare the order. Once the order is prepared it is notified to the customer. Order is either delivered to the customer or kept till customer collects its from the pharmacy. Head Pharmacist can update stocks and update order list. System admin maintains user profiles and staff profiles, System admin manages feedbacks and reviews from customers.

PART 1

3.Functional Requirements

- Guest can register to the system by entering details.
- Registered customer should be authenticated before login to the system.
- Registered customer should able to add products to the cart.
- Registered Customer needs to validate their payment details.
- Customer should be able to rate the items and services.
- Customer should be able to update their profiles.
- Pharmacist must be able to update stocks.
- Pharmacist must be able to Add, replace and remove items.
- Pharmacist must be able to see orders and prescriptions.
- Pharmacist must be able to update order status.
- Administrator must be able manage user profiles.
- Administrator must able to manage customer reviews.

4.Classes

- Customer
- Pharmacist
- Admin
- Products
- Cart
- Payment
- Order
- Report
- Discount

5.Class Responsibility Collaboration Cards

Customer	
Responsibilities	**Collaborations**
Login to the system	
Update customer details	
Rate products	Products

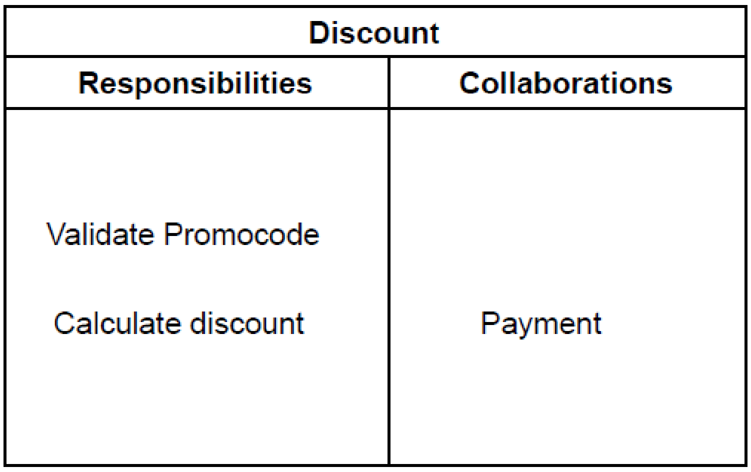
Products	
Responsibilities	**Collaborations**
Add products	
Replace products	
Update products	
Re stock	

Payment	
Responsibilities	**Collaborations**
Validate Payment	
Get payment details	
Calculate Total payment	Order,Products

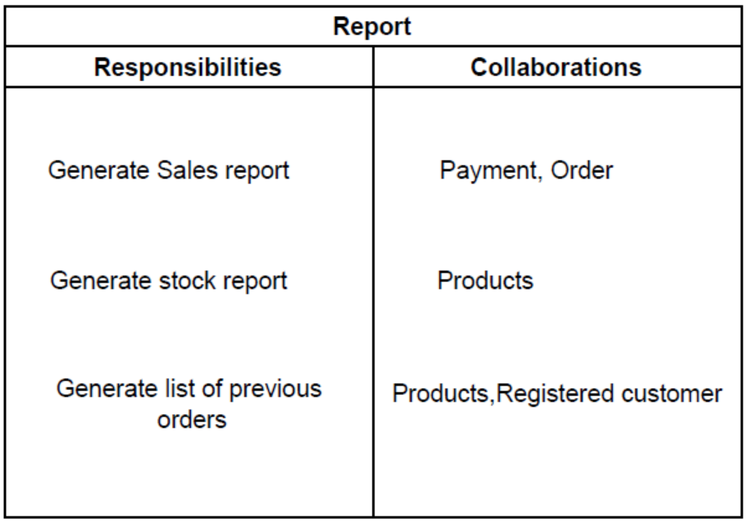
Cart	
Responsibilities	**Collaborations**
Display selected items	Products
Store selected items	Products
Display total	Payment

Pharmacist	
Responsibilities	**Collaborations**
Generate Invoice	Products
Store details of pharmacist	

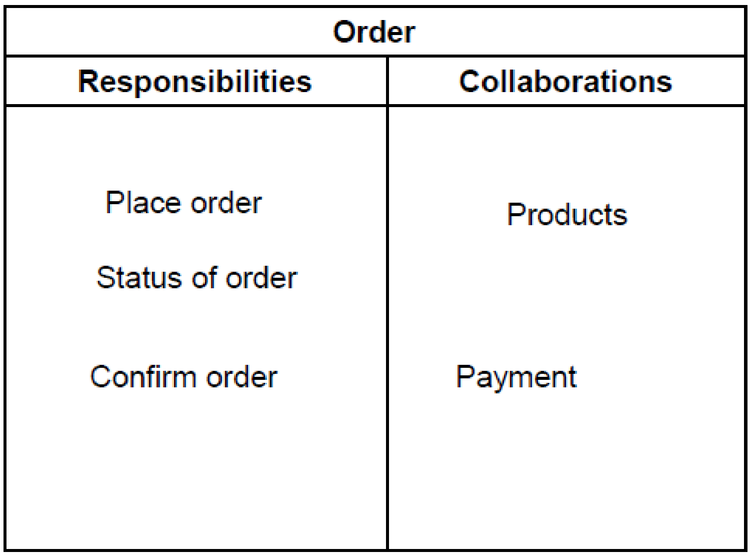
Admin	
Responsibilities	**Collaborations**
Manage user profiles	Registered customer
Store details of Admin	



Discount	
Responsibilities	**Collaborations**
Validate Promocode	
Calculate discount	Payment

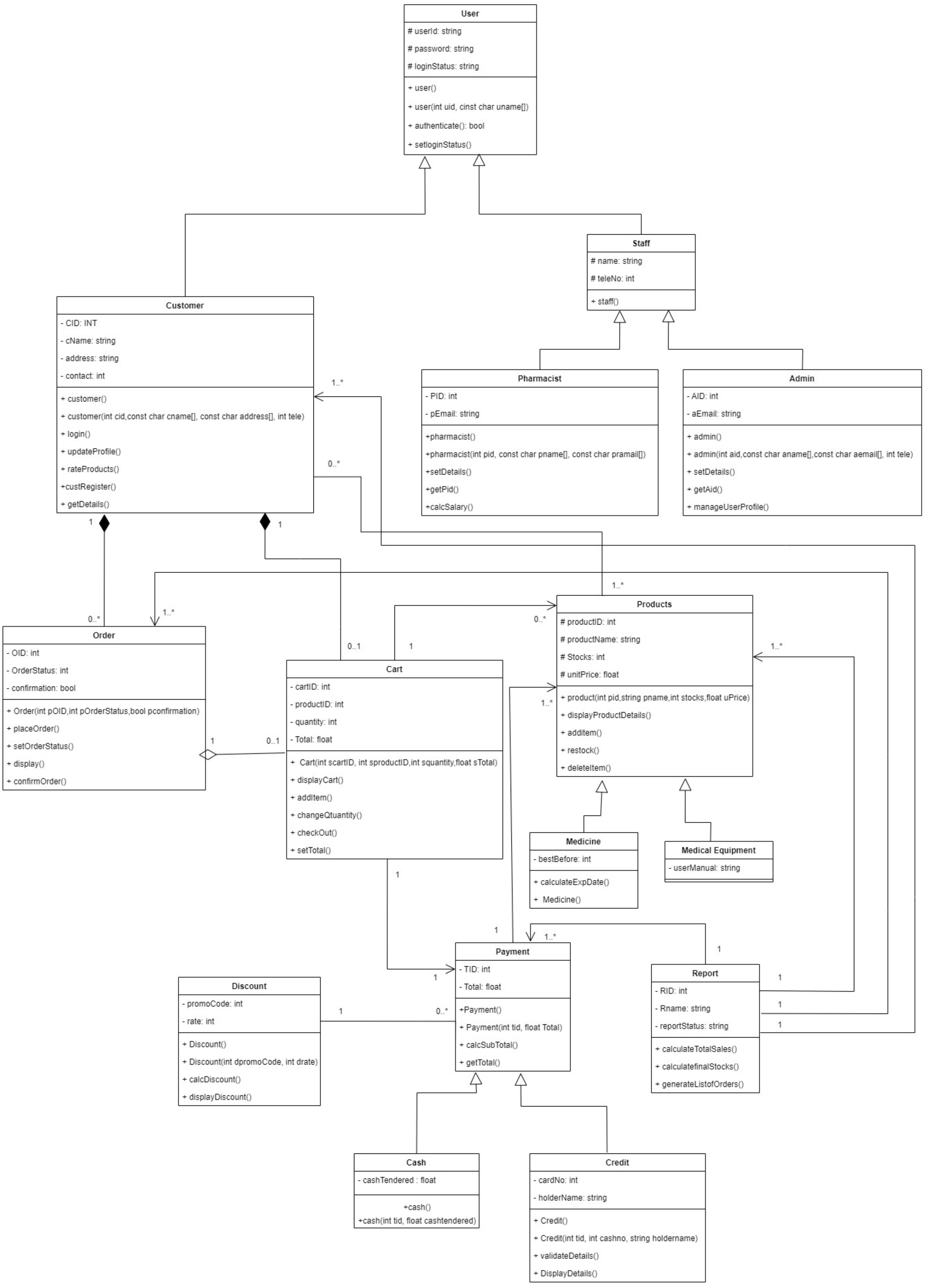


Report	
Responsibilities	**Collaborations**
Generate Sales report	Payment, Order
Generate stock report	Products
Generate list of previous orders	Products,Registered customer



Order	
Responsibilities	**Collaborations**
Place order	Products
Status of order	
Confirm order	Payment

6. Exercise 1


User
# userId: string
# password: string
# loginStatus: string
+ user()
+ user(int uid, cinst char uname[])
+ authenticate(): bool
+ setloginStatus()
Staff
# name: string
# teleNo: int
+ staff()
Pharmacist
- PID: int
- pEmail: string
+pharmacist()
+pharmacist(int pid, const char pname[], const char pramail[])
+setDetails()
+getPid()
+calcSalary()
Admin
- AID: int
- aEmail: string
+ admin()
+ admin(int aid,const char aname[],const char aemail[], int tele)
+ setDetails()
+ getAid()
+ manageUserProfile()
Customer
- CID: INT
- cName: string
- address: string
- contact: int
+ customer()
+ customer(int cid,const char cname[], const char address[], int tele)
+ login()
+ updateProfile()
+ rateProducts()
+custRegister()
+ getDetails()
1..*
0..*
1
1
0..*
1..*
Order
- OID: int
- OrderStatus: int
- confirmation: bool
+ Order(int pOID,int pOrderStatus,bool pconfirmation)
+ placeOrder()
+ setOrderStatus()
+ display()
+ confirmOrder()
1
0..1
0..1
1
Cart
- cartID: int
- productID: int
- quantity: int
- Total: float
+ Cart(int scartID, int sproductID,int squantity,float sTotal)
+ displayCart()
+ addItem()
+ changeQtuantity()
+ checkOut()
+ setTotal()
1..*
0..*
1..*
1..*
Products
# productID: int
# productName: string
# Stocks: int
# unitPrice: float
+ product(int pid,string pname,int stocks,float uPrice)
+ displayProductDetails()
+ additem()
+ restock()
+ deleteItem()
Medicine
- bestBefore: int
+ calculateExpDate()
+ Medicine()
Medical Equipment
- userManual: string
1
1
1
1..*
1
Payment
- TID: int
- Total: float
+Payment()
+ Payment(int tid, float Total)
+ calcSubTotal()
+ getTotal()
Discount
- promoCode: int
- rate: int
+ Discount()
+ Discount(int dpromoCode, int drate)
+ calcDiscount()
+ displayDiscount()
1
0..*
Report
- RID: int
- Rname: string
- reportStatus: string
+ calculateTotalSales()
+ calculatefinalStocks()
+ generateListofOrders()
1
1
1
Cash
- cashTendered : float
+cash()
+cash(int tid, float cashtendered)
Credit
- cardNo: int
- holderName: string
+ Credit()
+ Credit(int tid, int cashno, string holdername)
+ validateDetails()
+ DisplayDetails()

7. Exercise 2

User class

```
#pragma once
#include <cstring>
class user
{
protected:
	int userId;
	char userName[10];
	char loginStatus[10];
public:
	user() {
		userId = 0000;
		strcpy_s(userName, "NULL");
		strcpy_s(loginStatus, "NULL");
	};
	user(int uid, const char uname[]) {
		userId = uid;
		strcpy_s(userName, uname);
	};
	bool authenticate(int uid,const char uname[]);
	void setLoginStatus();
};
```

Customer class

```
#pragma once
#include "user.h"
#include <cstring>
class customer:public user
{
private:
	int CID;
	char cName[20];
	char Address[50];
	int contact;
	Products* pdt;
public:
	customer() {
		CID = 0000;
		strcpy_s(cName, "NULL");
		strcpy_s(Address, "NULL");
		contact = 0000000000;
	};
	customer(int cid, const char cname[], const char address[], int tele) {
		CID = cid;
		strcpy_s(cName, cname);
		strcpy_s(Address, address);
		contact = tele;
	};
	void login(int uid, const char uname);
	void updateProfile(int cid, const char cname[], const char address[], int
contact);
	void rateProducts();
	void custRegister(int cid, const char cname[], const char address[], int
contact);
	void getDetails();
};
```

Staff class

```
#include "user.h"
#include <cstring>
class staff:public user
{
protected:
	char name[20];
	int teleNo;
public:
	staff() {
		strcpy_s(name, "NULL");
		teleNo = 0000000000;
	};
};
```

Pharmacist class

```
#include "staff.h"
class pharmacist:public staff
{
private:
	int PID;
	char pEmail[10];
public:
	pharmacist() {
		PID = 0000;
		strcpy_s(name, "NULL");
		teleNo = 0000000000;
		strcpy_s(pEmail, "NULL");
	};
	pharmacist(int pid, const char pname[], const char pemail[], int tele) {
		PID = pid;
		strcpy_s(name, pname);
		strcpy_s(pEmail, pemail);
		teleNo = tele;
	};
	void setDetails(int pid, const char pname[], const char pemail[], int 
tele);
	int getPid();
	void calcSalary();
};
```

Admin class

```
#include "staff.h"

class admin:public staff
{
private:
	int AID;
	char aEmail[10];
public:
	admin() {
		AID = 0000;
		strcpy_s(name, "NULL");
		teleNo = 0000000000;
		strcpy_s(aEmail, "NULL");
	};
	admin(int aid, const char aname[], const char aemail[], int tele) {
		AID = aid;
		strcpy_s(name, aname);
		strcpy_s(aEmail, aemail);
		teleNo = tele;
	}
	void setDetails(int aid, const char aname[], const char aemail[], int 
tele);
	int getAid();
	void managerUserProfile();
};
```

Discount class

```
#include <iostream>
#include "Payment.h"
#include "Products.h"

class Discount{
private:
int promoCode;
int rate;
Products *Pr[SIZE];
Payment *Payment[];
public:
Discount(){
  promoCode = 0;
  rate = 0;
}
Discount(int dpromoCode, int drate){
  promoCode = dpromoCode;
  rate = drate;
}
void calcDiscount(int rate, int total);
void displayDiscount();
};
```

Payment class

```
#include <iostream>
#include "Discount.h"
using namespace std;

class Payment{
protected:
float total;
int TID;
private:
Products *Pr[SIZE];
Discount *Dis;
public:
Payment(){
  total = 0;
  TID = 0;
}
Payment(int tid, float TOTAL, Discount *pdis){
  TID = tid;
  total = TOTAL;
  Dis[SIZE] = pdis[SIZE];}
float getTotal(){
  return total;
}
void calcSubTotal(float total,Discount *Dis);
};
```

Cash class

```
class Cash:public Payment{
private:
float cashTendered;
public:
Cash(){
  cashTendered = 0;
}
Cash(int tid, float cashtendered):Payment(tid){
  TID = tid;
  cashTendered = cashtendered;
}
};
```

Credit class

```
class Credit:public Payment{
private:
int cashNo;
string holderName;
public:
Credit(){
  cashNo = 0;
  holderName = "";
}
Credit(int tid,int cashno, string holdername):Payment(tid){
  TID = tid;
  cashNo = cashno;
  holderName = holdername;
};
void validateDetails();
void DisplayDetails();
};
```

Cart class

```
class Cart
{
    private:
        int cartID;
        int productID;
        int quantity;
        float Total;
    public:
        Cart(int scartID, int sproductID, int squantity, float sTotal)
        {
            scartID = cartID;
            sproductID = productID;
            squantity = quantity;
            sTotal = Total;
        }
        void displayCart();
        void addItem();
        void changeQuantity();
        void checkOut();
        void setTotal();
};
```

Order class

```
using namespace std;
class Order {
private:
    int OID;
    int OrderStatus;
    bool confirmation;

public:
    Order(int pOID, int pOrderStatus, bool pconfirmation);
    {
        pOID = OID;
        pOrderStatus = OrderStatus;
        pconfirmation = confirmation;
    }
    void setOrderStatus();
    void displayOrder();
    bool confirmOrder();
};
```

Product class

```
class Product {

protected:
    int PID;
    string PName;
    int Stocks;
    float UPrice;
    Payment* paymentP;
    Cart* cartP;
    Pharmacist* pharmacistP;
    Report* reportP;

public:
    Product(int pID, string pName, int stocks, float uPrice)
    {

        PID = pID;
        PName = pName;
        Stocks = stocks;
        UPrice = uPrice;
    }
    void displayProductDetails();
    virtual int calculateExpDate();
    void addItem(int pID, string pName, int stocks, float uPrice);
    void deleteItem();
    void Restock();
};
```

Medicine class

```
class Medicine :public Product {

private:
    int bestBefore;
    int currentDate;


public:
    Medicine();
    Medicine(int bBefore, int CDate, int pID, string pName, int stocks, double
uPrice) : Product(int pID, string pName, int stocks, double uPrice) {
        bestBefore = bBefore;
        currentDate = CDate;
    }

    int calculateExpDate(int bestBefore, int currentDate);
};
```

Medical Equipment class

```
class MedicalEquipment :public Product {

private:
    string userManual;

public:
    MedicalEquipment();

};
```

Report class

```
class Report
{
private:
    int rid;
    char rname[];
    char reportStatus[];

    Payment* pay;
    Product* pro;
    Order* odr;
    Customer* cus;

public:
    Report();
    int calculateTotalSales();
    int calculateFinalStocks();
    void generateListofOrders();

};
```

Main.cpp file

```cpp
#include <iostream>
#include <cstring>
#include "user.h"
#include "customer.h"
#include "staff.h"
#include "admin.h"
#include "pharmacist.h"
#include "Payment.h"
#include "Discount.h"
#include "Products.h"
#include "Order.h"
#include "Report.h"

using namespace std;

int main()
{
    user u1;
    customer c1;
    staff s1;
    admin a1;
    pharmacist p1;
    Payment* P1;
    Cash c1;
    Credit C1;
    Discount* d1;
    Product* P[2];
    P[0] = new Medicine("2025", "2022", "001", "uy", "200", "250.00");
    P[1] = new MedicalEquipment("002", "iuy", "1200", "260.00");
    Order* o1 = new Order();
    Cart* c1 = new Cart();
    Report r1;

    return 0;
}
```