Topic : Textile & Garment Management System

Group no : MLB_10.01_02

Campus : Malabe

Submission Date : 19th May 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**
IT21181856	Kandambige S.T.	0766239653
IT21181924	Sandeepani E.A.	0764296282
IT21181788	Madanayake P.C.S.	0763356331
IT21182396	A.P.Ranaweera	0712300379
IT21184208	M.G.S.D.Kumara	0778926506

Introduction

Customers can buy products, give orders, make payments, add or delete items in cart, give feed backs, request return and exchanges, give direct orders. Customers can make payments via cards or via bank deposits. customers can search and view products on the web site and contact customer service. Every customer has a name, customer id, email address, contact number and address. The system admin must be able to add and remove customers, add new products and view/ update existing products, make or remove orders and view existing orders, view and generate reports on the products. Designer can submit proposals and accept orders. Customer service officer can review those feedbacks and take actions and contact customers. The ware house administrator keeps and update the records of sales and stocks. The financial administrator can log in to the system and make reports and store them in the management system.

Classes

Customer

Feedback

Products

Stock

Cart

Payment

Sales

Bankcard

Paypal

Report

Order

Designer

Proposal

CRC cards

Customer class	
Responsibility	**Collaborators**
Update personal details	
Return and exchange	Product
Buy product	Product

Designer class	
Responsibility	**Collaborators**
Select designer	

Product class	
Responsibility	**Collaborators**
Store details of product	Customer,stock,cart

Order class	
Responsibility	**Collaborators**
Place order	Customer, Designer
Store order details	Report

Payment class	
Responsibility	**Collaborators**
Store payment details	Bank card, Paypal, cart
Validate	

Cart class	
Responsibility	**Collaborators**
Add product	Product
Delete product	Product
Buy cart items	Payement

Bank Cards class	
Responsibility	**Collaborators**
Store Bank card details	
Authorize	
Store bank card payment details	payment

Paypal class	
Responsibility	**Collaborators**
Store paypal payment details	payment

Feedbacks class	
Responsibility	**Collaborators**
Store feedback details	Customer
Give feedback	Customer

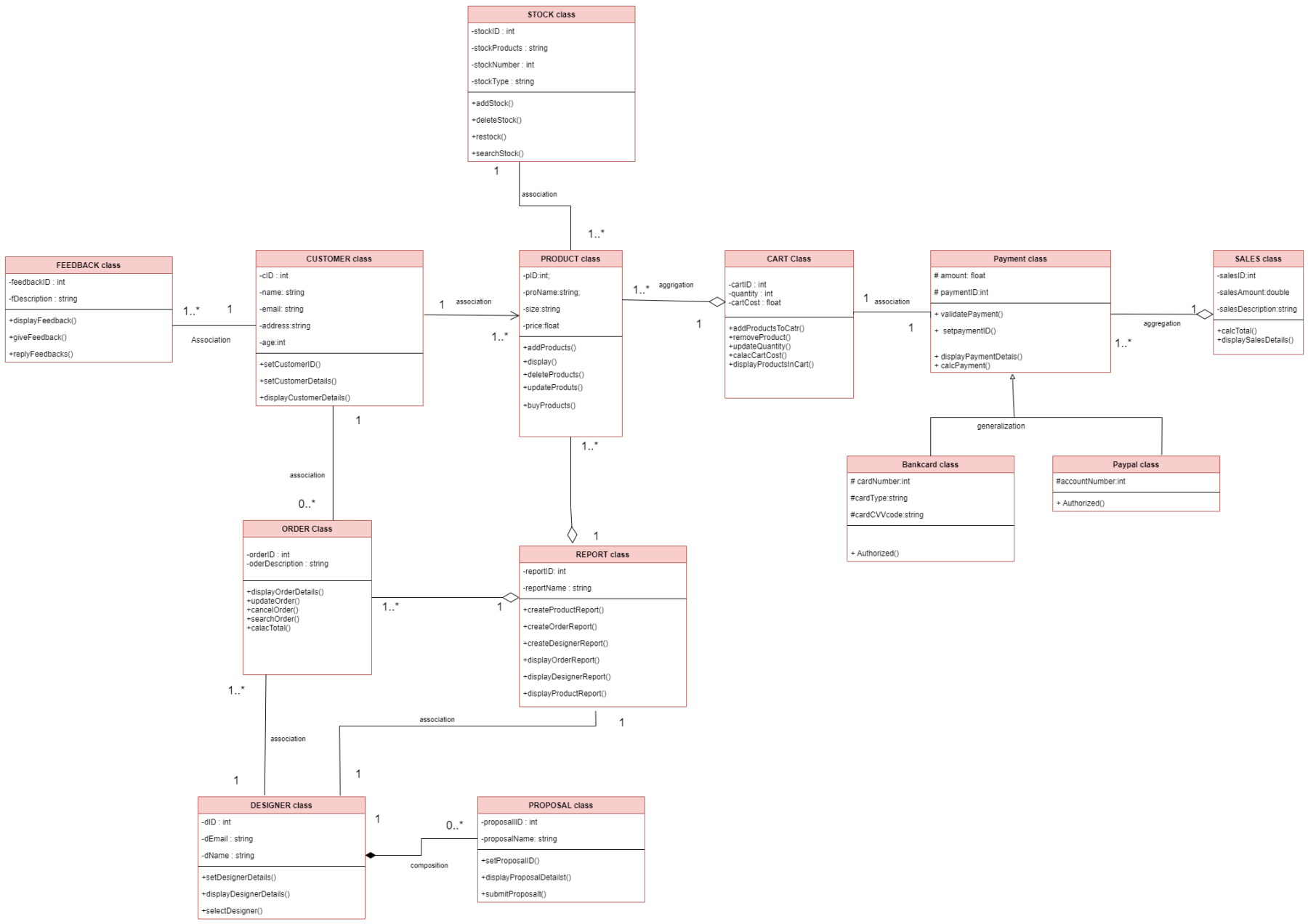
Reports class	
Responsibility	**Collaborators**
List of product	Product
List of previous order	Order
List of designer	Designer

Proposal class	
Responsibility	**Collaborators**
Submit proposal	designer

Sales class	
Responsibility	**Collaborators**
Store sales details	payment

Stocks class	
Responsibility	**Collaborators**
Add stock	product
Restock	

Class diagram



Codes

```
#define SIZE 2

//Stock class

class Stock

{

   private:

      int stockID;

      int stockNumber;

      string stockType;

      Product *prod[SIZE];

      int productNo;


public:

    Stock(){};

    Stock(int id, int num, string type );

    void addProduct(Product *pro);

    void displayStockDetails();

};


//Customer class

class customer

{

  private :

    int customerID;

    string name;

    string email;

    string address;
```

```
    int age;
    feedback *Feedback[SIZE];
    int noOfFeedback;
    Order *order[SIZE];
    Product *product[SIZE];

  public  :
    customer();
    customer(string cname, string cemail, string caddress,int cage);
    void setCustomerID(int customerID );
    void addFeedback(feedback *o);
    void customerDetails();
    void displayCustomerAndFeedbackDetails ();
    ~customer();
};



//Feedback class
class feedback
{
  private :
    string feedbackID;
    string fDescription[200];
    customer *cus;

  public  :
    feedback(string fID, customer *fCus);
```

```
    string getGiveFeedback();
    void setReplyFeedback(string fReply);
    void displayFeedback();
    ~feedback();
};

//Product class
class Product{
private:
int pID;
char proName[20];
char size[3];
float price;
Stock *stock;//bidirectional relationship to Stock class

public:
Product();
Product(int id, const char Name[],const char pSize[],float pPrice,Stock *st);
void display();
void addProduct();
void deleteProduct();
void updateProduct();
void buyProduct();
~Product();
};

//Report class
```

```
class Report{
private:
int reportID;
char reportName[20];
Product *product[SIZE];//aggregation relationship to Product class
Order *order[SIZE];//aggregation relationship to Order class
DESIGNER *d;//bi derectional association relationship to Designer class
public:
Report();
void createProductReport(Product *p1, Product *p2);
void displayProductReport();
void createOrdertReport(Order *o1, Order *o2) ;
void displayOrderReport() ;
void createDesignerReport(DESIGNER *d1);
void displayDesignerReport();
~Report();
};
```

```
//Cart class
#include"Product.h"
#include"Payment.h"

class Cart{
        private:
                int cartID ;
                int quantity ;
                float cartCost ;
```

```
        Product *product[SIZE];//aggregation relationship to product
        Payment *pay;//association relationship to payment class
    public:
        Cart(){};
        Cart(int id, int qty, float cost)
        {
            cartID=id;
            quantity=qty;
            cartCost=cost;
        }
        void addProductsToCart(Product *p1, Product *p2);
        void dispalyProductsInCart();
        void removeProductsInCart();
        void updateQuantity();
        float calcCartCost();
        ~Cart();
};
//Order class
#include"Customer.h"
#include"DESIGNER.h"

class Order{
    private:
        int orderID;
        char orderDescription[50];
    customer *cus;//association relationship to Customer
        DESIGNER *designer;//association relationship to Designer class
```

```
        public:
                Order();
                Order(int oID);
                void displayOrderDetails(const char desc[], customer*cus1);
                ~Order();
};

//Proposal class
class PROPOSAL{
        private:
                int proposalID;
                char proposalName;
        public:
                PROPOSAL(){};
                PROPOSAL(int pno,const char pname[] ){
                        proposalID=pno;
                        strcpy(proposalName,pname);
                }
                void setProposalID();
                void submitProposal();
                void displayProposalDetailst();
                ~PROPOSAL();
};

//Designer class
class DESIGNER{
```

```
private:
        int dID;
        string dName;
        string dEmail;
        PROPOSAL *propsal[SIZE];
 Order *order[SIZE];

public:
        void setDesignerDetails(int di,string dnam,string mail);
        void displayDesignerDetails();
        void selectDesigner();
 DESIGNER();
        DESIGNER(int p1,int p2,string p1n,string p2n);
        void DisplayProposals();
        ~DESIGNER();
};
//Payment class
class  Payment{
   protected:
      float amount;
      int paymentID;
      float paymentDate;
      Cart *cart;

   public:
      Payment();
      Payment(float amt);
```

```
        void validatePayment();
        void setPaymentID();
        void setPaymentData();
        void displayPaymentDetails();
        void calcPayments();
        void displayAmount();

    };
//Bankcard class
class Bankcard: public Payment{
     protected:
        int cardNum;
        string cardType;
        int cardCVVcode;
     public:
        void authorized();
        Bankcard();
        Bankcard (float amt,int number, string type, int code): Payment(amt);
        void displayBankcard();
};
//paypal class
class paypal: public Payment{
    protected:
      int paypalID;
      int accountNumber;

   public:
```

```
        void authorized();
        paypal();
        paypal(float amt,float pay, int aacNo):Payment(amt);
        void displayPaypal();
};

//Sales class
class Sales{
private:
int salesID;
double salesAmount;
char salesDescription[100];
Payment *pay[SIZE];

public:
Sales();
int calcTotal(Payment *p1, Payment *p2);
void displaySalesDetails();
};

//Main .cpp

#include<iostream>
#include<cstring>
#include"Stock.h"
#include"Feedback.h"
#include"Customer.h"
```

```
#include"Stock.h"



using namespace std;


Stock::Stock(int id, int num, string type){
            stockID=id;
            stockNumber=num;
            stockType=type;


      }


void Stock:: addProduct(Product *pro)
   {
       if(productNo < SIZE) {
         prod[productNo]=pro;
         productNo++;
         }


   }


void Stock::displayStockDetails()
   {
            cout<<"Stock id: "<<stockID<<endl;
            cout<<"Stock Number: "<<stockNumber<<endl;
            cout<<"Stock type: "<<stockType<<endl<<endl;
```

```
     for(int i=0; i< productNo;i++)
          prod[i] -> productDisplay();
     }

feedback::feedback (string fID , customer *fCus)
{
  feedbackID = fID;
  cus = fCus;
  cus ->addFeedback(this);
}

void feedback::displayFeedback()
 {
   cout<< "feedback ID  :"<< feedbackID << endl;
 }

customer::customer(string cname, string cemail, string caddress,int cage)
 {
   name = cname;
   email = cemail;
   address = caddress;
   age = cage;
   noOfFeedback=0;
 }
```

```
void customer::addFeedback(feedback *o)
{
  if (noOfFeedback < SIZE)
      Feedback[noOfFeedback]=o;
      noOfFeedback++;
}



 void customer:: displayCustomerAndFeedbackDetails ()
  {
        cout<<"Customer name :"<< name<< endl;
        cout<<"Customer email :"<< email<< endl;
        cout<<"Customer address :"<< address<< endl;
        cout<<"Customer age :"<< age<< endl;

    for(int i=0; i<noOfFeedback; i++)
      Feedback[i] -> displayFeedback();
  }

Product::Product(){};
Product::Product(int id, const char Name[],const char pSize[],float pPrice ,Stock *st){
  pID=id;
  strcpy(proName,Name);
  strcpy(size,pSize);
  price=pPrice;
  st=stock;
}
```

```
void Product:: display(){
  cout<<"Product ID: "<<pID<<endl;
  cout<<"Product name: "<<proName<<endl;
  cout<<"Product size: "<<size<<endl;
  cout<<"Product price: "<<price<<endl;
}

Report::Report(){};
void Report:: createProductReport(Product *p1, Product *p2) {
        product[0] = p1;
        product[1] = p2;
    }
void Report:: displayProductReport() {
          product[0]->display();
          product[1]->display();
    }

void Report:: createOrdertReport(Order *o1, Order *o2) {
        order[0] = o1;
        order[1] = o2;
    }
void Report:: displayOrderReport() {
          order[0]->displayOrderDetails();
          order[1]->displayOrderDetails();
    }
void Report:: createDesignerReport(DESIGNER *d1) {
    d=d1;
```

```
    }

void Report:: displayDesignerReport() {
        order[0]->setDesignerDetails();
        order[1]->displayDesignerDetails();
    }

void Cart:: addProductsToCart(Product *p1, Product *p2){
  product[0]=p1;
  product[1]=p2;
}

void Cart:: dispalyProductsInCart(){
product[0]->display();
  product[2]->display();
  }

void Order:: displayOrderDetails(const char desc[],customer *cus1){
                        strcpy(orderDescription,desc);
    cus=cus1;
    cout<<"Order id:"<<orderID<<endl;
    cout<<"Order description: "<<orderDescription<<endl;

                    }

void PROPOSAL:: displayProposalDetailst(){
                        cout<<"Propoal ID:"<<proposalID<<endl;
```

```
                cout<<"Propoal Name:"<<proposalName<<endl;
        }

PROPOSAL::~PROPOSAL(){
                cout<<"Deleting Proposal ID:"<<proposalID<<endl;
        }

DESIGNER::DESIGNER(){
    propsal[0]=new PROPOSAL(123,"abc");
    propsal[1]=new PROPOSAL(456,"def");
    }

DESIGNER::DESIGNER(int p1,int p2,string p1n,string p2n){

        propsal[0]=new PROPOSAL(p1,p1n);
    propsal[1]=new PROPOSAL(p2,p2n);

    }

void DESIGNER:: DisplayProposals(){
        for(int i=0;i<SIZE;i++){
                propsal[i]->displayProposalDetailst();
        }

    DESIGNER::~DESIGNER(){
```

```
                cout<<"designer resign"<<endl;
                for(int r=0;r<SIZE;r++){

                        delete propsal[r];
                }
                cout<<"Everthing is deleted"<<endl;
            }

Payment::Payment(float amt){
            amount=amt;
    }
void Payment:: displayAmount(){
        cout<<"Amount :"<<amount<<endl<<endl;
    }

Bankcard::Bankcard (float amt,int number, string type, int code):Payment( amt ){

          cardNum=number;
          cardType=type;
          cardCVVcode=code;
      }

void Bankcard:: displayBankcard(){
        cout<<"Bank Card Details :"<<endl<<endl;
        cout<<"Card Number :"<<cardNum<< endl;
        cout<<"Card Type :"<<cardType<<endl;
        cout<<"Card Code :"<<cardCVVcode<<endl;
```

```
        displayAmount();
      }

paypal::paypal(float amt,float pay, int aacNo):Payment(amt){
          paypalID=pay;
          accountNumber=aacNo;
       }

void paypal:: displayPaypal(){
        cout<<"Paypal Details :"<<endl<<endl;

        cout<<"Paypal ID :"<<paypalID<< endl;
        cout<<"Account Number :"<<accountNumber<<endl;
        displayAmount();
      }

int Sales::calcTotal(Payment *p1, Payment *p2){
 salesAmount=p1+p2;
}

int main(){

 designer1=new DESIGNER(111,222,"aaa","bbb");
 Payment *P1 = new Payment(20000);
 Bankcard *B1 = new Bankcard(20000,4555,"visa",122);
 Payment *P2 = new Payment(10000);
 paypal *pay1 = new paypal(20000,17812,1111);
```

```
  Stock *S1= new Stock(666,45,"good");

  Stock *S2= new Stock(566,55,"good");

  customer *c1 = new customer ("Sandaru Vishvajith","sandaru@gmail.com","No 5, Madanayaka
road,Galle",35);

  customer *c2 = new customer ("raveen Ariyarathne","ravee11@gmail.com","No 3 Maddegoda
road",40);

  feedback *f1 = new feedback ("0011",c1);

  feedback *f2 = new feedback ("0018",c1);

  Product *P1= new Product(001,"T shirt", "L", 4000,S1);

  Product *P2= new Product(002,"Frock", "M", 5000,S1);


 DESIGNER *designer1;

designer1->DisplayProposals();


  B1->displayBankcard();

  pay1->displayPaypal();


  Sales *sale=new Sales();

  sale->calcTotal(P1,P2);


  Order *O1= new Order();

  Order *O2= new Order();


  Designer *D= new Designer();


  Report *r1= new Report();

  r1->createProductReport(P1, P2);

  r1->displayProductReport();
```

```
 Report *r2=new Report();

 r2->createOrderReport(o1,o2);

 r2->displayOrderReport();

 Report *r3=new Report();

 r3->createDesignerReport(d1);

 r3->displayDesignerReport();


 Cart *c=new Cart();

 c->addProductsToCart(P1, P2);

 c->dispalyProductsInCart();

 c1 -> displayCustomerAndFeedbackDetails();

 S1-> displayStockDetails();

 S2-> displayStockDetails();


delete designer1;


}
```

Individual contribution

	Student ID	Student Name	Individual Contribution
1	IT21181788	Madanayake P.C.S.	• Create CRC card for Payment, Bank card and Paypal classes. • Create the class diagram for Payment, Bank card and Paypal classes. • Implement the coding for Payment, Bank card and Paypal classes.

2	IT21181924	Sandeepani E.A.	• Create CRC card for Order and Cart classes. • Create the class diagram for Order and Cart classes • Implement the coding for Order and Cart classes
3	IT21181856	Kandambige S.T.	• Create CRC card for Report, Product and Sales. • Create the class diagram for Report, Product and Sales. • Implement the coding for Report, Product and Sales.
4	IT21184208	Kumara M.G.S.D	• Create CRC card for Designer, Proposals and Stock. • Create the class diagram for Designer, Proposals and Stock. • Implement the coding for Designer, Proposals and Stock.
5	IT21182396	Ranaweera A.P.	• Create CRC card for Customer and Feedback classes. • Create the class diagram for Customer and Feedback classes. • Implement the coding for Customer and Feedback classes.