Topic : Online Musical Instruments, Accessories and Equipment Store

Group no : KDY_10

Campus : Kandy

Submission Date : 20th 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
IT21303852	Ammaar M. A	0778596986
IT21303920	Mushtaq M. B. M	0777909315
IT21360114	Mohamed Z. M. N	0702254031
IT21296550	Ahlaan M. I	0777326274
IT21296796	Shabry S. M	0766440967

Description of the Requirements

Musical instruments can be used for various purposes such as, hobbies and for professional requirements. Buying a musical instrument can be very time consuming. To make it easier for the consumer to be able to purchase a musical instrument from the comfort of their homes this website is introduced. Through a web browser, a customer searches the website name. The search engine provides the relevant link. The user enters the website using that link. Then proceeds to search or browse for a particular item. Once the item is found, the user adds the item to the cart. Then the user enters the cart page and proceeds to buy the item. When buying the item, the user is required to create an account. To create an account, the user must enter a username and password and some personal details. Then the user is redirected to the purchase page, where the user must enter details such as credit card details and shipping address. Then the user is provided with a shipping date and an estimated delivery date. Then the user is redirected to page where the user can enter feedback.

- Users register to the system as a guest
- User logs into the site
- View musical instruments
- Search musical instruments
- View musical instruments categories
- User login to the site as a registered user
- Add products to the cart
- Registered customers can update profile details
- Purchase musical instruments
- Enter payment details
- Enter payment method
- Registered users can send feedback

Identified Classes

- User
- Customer
- Product
- Admin
- Payment
- Discount
- Order
- Cart
- Feedback

CRC Cards

User	
Responsibilities	Collaborations
Register to the system Search items	 Product

Customer	
Responsibilities	Collaborations
Store customer details Search product Purchase product Add to cart Make payments Provide feedback	 Product Product Cart Payments Feedback

Product	
Responsibilities	Collaborations
Store item details	

Payment	
Responsibilities	Collaborations
Store payment details Validate payment	

Order	
Responsibilities	Collaborations
Store order details Calculate total Remove orders Confirm order Display previous orders	 Payment

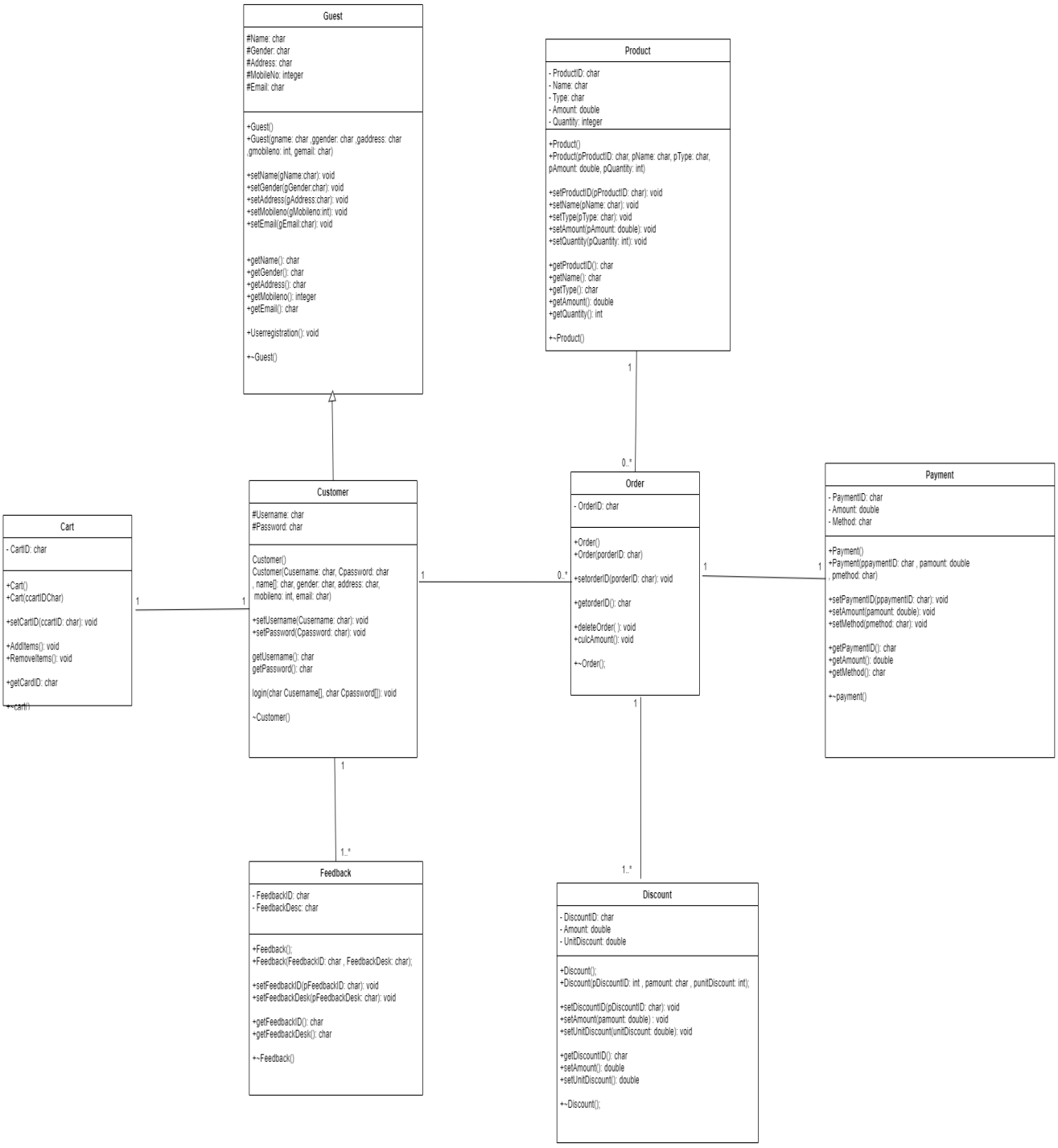
Feedback	
Responsibilities	Collaborations
Store feedback details Remove feedback	

Discount	
Responsibilities	Collaborations
View payment details	Payment
Add discount	
Remove discount	
Update payment after discount	Payment

Cart	
Responsibilities	Collaborations
View product details	Product
Proceed to payment	Payment

Admin	
Responsibilities	Collaborations
Add product	Product
Remove product	Product
Update product details	Product

Class Diagram



Individual Contribution

Registration No	Name	Individual Contribution
IT21303852	Ammaar M. A	Product
IT21303920	Mushtaq M. B. M	User, Customer
IT21360114	Mohamed Z. M. N	Feedback, DIscount
IT21296550	Ahlaan M. I	Cart
IT21296796	Shabry S. M	Order, Payment

Codes

Main.cpp

```
#include <iostream>
#include <cstring>
#include "product.h"
#include "order.h"
#include "discount.h"
#include "Guest.h"
#include "customer.h"
#include "feedback.h"
#include "cart.h"
#include "payment.h"
using namespace std;

int main(){

  Guest *g1, *g2;
  g1 = new Guest();
  c2 = new Guest("M. B. M Mushtaq", "Male", "101, Anuradhapura", 0777909315,
"mushtaq123@gmail.com");

  Customer *c1, *c2;
  c1 = new Customer();
```

```
c2 = new Customer("Mushtaq", "Burhan123", "M. B. M Mushtaq", "Male", "101, Anuradhapura", 0777909315, "mushtaq123@gmail.com");

Product *p1, *p2;
p1 = new Product();
p2 = new Product("101", "Acoustic Guitar", "Guitar", 10000, 12);

Order *o1, *o2;
o1 = new Order();
o2 = new Order("1000");

Payment *p1, *p2;
o1 = new Payment();
o2 = new Payment("A200", 8000, "Credit card");

Cart *cart1, *cart2;
o1 = new Cart();
o2 = new Cart("103", "101");

Feedback *f1, *f2;
f1 = new Feedback();
f2 = new Feedback("A01" , char "Quality product but packaging can improve ");

Discount *d1, *d2;
d1 = new Discount();
d2 = new Discount("104", 200, 100);
```

```
}
```

Product.h

```
class Product{
//private attributes
  private:
     char productID[5];
     char name[20];
     char type[20];
          double amount;
     int quantity;

  public:
     Product(); //default constructor
     Product(char pProductID[], char pName[], char pType[], double pAmount, int pQuantity);
//overloaded constructor

// setters
     void setProductID(char pProductID[]);
     void setName(char pName[]);
     void setType(char pType[]);
     void setAmount(double pAmount);
     void setQuantity(int pQuantity);

//getters
     char getProductID();
```

```
    char getName();
    char getType();
    double getAmount();
    int getQuantity();
//destructor
    ~Product();
};
```

Product.cpp

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

//default constructor
Product::Product(){

    strcpy(productID, " ");
    strcpy(name, " ");
    strcpy(type, " ");
        amount = 0;
    quantity = 0;

}

//overloaded constructor
Product::Product(char pProductID[], char pName[], char pType[], double pAmount, int pQuantity){

    strcpy(productID, pProductID);
    strcpy(name, pName);
    strcpy(type, pType);
```

```
        amount = pAmount;
    quantity = pQuantity;

}

void Product::setProductID(char pProductID[]){

    //method implementation

}

void Product::setName(char pName[]){

    //method implementation

}

void Product::setType(char pType[]){

     //method implementation

}

void Product::setAmount(double pAmount){
```

```
    //method implementation

}

void Product::setQuantity(int pQuantity){

    //method implementation

}

char Product::getProductID(){

    //method implementation

}

char Product::getName(){

    //method implementation

}

char Product::getType(){

    //method implementation
```

```
}

double Product::getAmount(){

    //method implementation

}

int Product::getQuantity(){

    //method implementation

}

Product::~Product(){

    //method implementation

}
```

Guest.h

```
#pragma once

class Guest{
protected:
        char Name[50];
        char Gender[10];
        char Address[100];
        int MobileNo;
        char Email[50];

public:
//Default Constructor
        void guest();

//Overloaded Constructor
        void guest(char gname[],char ggender[],char gaddress[], int gmobileno, char gemail[]);

//Setters
        void setName(char gName[]);
        void setGender(char gGender[]);
        void setAddress(char gAddress[]);
        void setMobileno(int gMobileno);
```

```
        void setEmail(char gEmail[]);


//Getters
        char getName();
        char getGender();
        char getAddress();
        int getMobileno();
        char getEmail();


        void Userregistration();


//Destructors
        ~Guest();
};
```

Guest.cpp

```
#include <iostream>
#include "Guest.h"
#include <cstring>

using namespace std;

//Default Constructor
void Guest::guest() {
	strcpy(Name, "");
	strcpy(Gender, "");
	strcpy(Address, "");
	MobileNo = 0;
	strcpy(Email, "");
}
//Overloaded Constructor
void Guest::guest(char gname[],char ggender[], char gaddress[], int gmobileno, char gemail[]) {
	strcpy(Name, gname);
	strcpy(Gender, ggender);
	strcpy(Address, gaddress);
	MobileNo = gmobileno;
	strcpy(Email, gemail);
```

```
}
void Guest::setName(char gName[]) {
        //Implementations
}
void Guest:: setGender(char gGender[]) {
        //Implementations
}
void Guest:: setAddress(char gAddress[]) {
        //Implementations

}
void Guest:: setMobileno(int gMobileno) {
        //Implementations

}
void Guest:: setEmail(char gEmail[]){
        //Implementations

}

char Guest:: getName(){
        //Implementations
}

char Guest:: getGender() {
```

```
	//Implementations
}

char Guest:: getAddress() {
	//Implementations
}

int Guest:: getMobileno() {
	//Implementations
}

char Guest:: getEmail(){
	//Implementations
}

void Guest::Userregistration() {
	//Implementations
}

Guest:: ~Guest() {

}
```

Customer.h

```
#pragma once

#include "Guest.h"


class Customer : public Guest{


protected:

	char Username[15];

	char Password[15];


public:

	//Default Constructor

	void customer();


	//Overloaded Constructor

	void customer(char Cusername[], char Cpassword[], char name[], char gender[], char address[], int mobileno, char email[]);


	//Setters

	void setUsername(char Cusername[]);

	void setPassword(char Cpassword[]);


	//Getters

	char getUsername();

	char getPassword();
```

```
        void login(char Cusername[], char Cpassword[]);


        //Destructors

        ~Customer();


};
```

Customer.cpp

```
#include <iostream>
#include "customer.h"
#include <cstring>

using namespace std;

void Customer :: Customer() {
        strcpy(Username, "");
        strcpy(Password, "");

}
void Customer :: Customer(char Cusername[], char Cpassword[], char name[], char gender[],
char address[], int mobileno, char email[]){

  strcpy(Username, Cusername);
        strcpy(Password, Cpassword);
        strcpy(Name, name);
        strcpy(Gender, gender);
        strcpy(Address, address);
        MobileNo = mobileno;
        strcpy(Email, email);

}
```

```
void Customer::setUsername(char uname[]) {
        //Implementations
}

void Customer::setPassword(char upass[]) {
        //Implementations
}

char Customer::getUsername() {
        //Implementations
}

char Customer::getPassword() {
        //Implementations
}

void login(char uname[], char upass[]) {
        //Implementations
}

Customer :: ~Customer() {

}
```

Cart.h

```
#include<iostream>
#include<cstring>

using namespace std;

///class cart implementation
class cart
{
private:
        char CartID[10];
        char ProductID[10];

public:

        // default constructor
         cart();

         // overloaded constructor
         cart(char ccartID[], char pproductID[]);

         //setters
         void setCartID(char ccartID);
         void setProductID(char pproductId);
```

```
        //getters
        char AddItems();
        char RemoveItems();

        //destructor
        ~cart();

};
```

Cart.cpp

```
#include "cart.h"
#include<iostream>
#include<cstring>

using namespace std;

// default constructor implementation
cart::cart()
{
	strcpy(CartID, "");
	strcpy(ProductID, "");
}

// overloaded constructor implementation
cart::cart(char ccartID[], char pproductID[])
{
	strcpy(CartID, ccartID);
	strcpy(ProductID, pproductID);
}

void cart::setCartID(char ccartID[])
{
```

```
 //Implementations
}

void cart::setProductID(char pproductId[])
{
 //Implementations
}

char cart::AddItems()
{
 //Implementations
}

char cart::RemoveItems()
{
 //Implementations
}
```

Payment.h

```
class Payment
 {
   private:
     char paymentID[10];
     double amount;
     char method[10];

   public:

     // default constructor
     Payment();

     // overloaded constructor
     Payment(char ppaymentID[10] , double pamount , char pmethod[10]);

     //setters
     void setpaymentID(char ppaymentID[10]) ;
     void setamount(double pamount) ;
     void setmethod(char pmethod[10]) ;

     //getters
     char getpaymentID() ;
```

```
    double getamount();
    char getmethod();

    //destructor
    ~payment();

};
```

Payment.cpp

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

using namespace std;

Payment :: Payment()
{
 strcpy("paymentID","");
 amount = 0 ;
 strcpy("method","");
}

Payment :: Payment(char ppaymentID[10] , double pamount , char pmethod[10])
{
 strcpy("paymentID","ppaymentID");
 pamount = amount;
 strcpy("method","pmethod");
}

void Payment :: setpaymentID(char ppaymentID[10])
{
```

```
}

void Payment :: setamount(double pamount)

{


}


void Payment :: setmethod(char pmethod[10])

{


}


char Payment :: getpaymentID()

{
```

Order.h

```
class Order
 {
   private:
     char orderID[10];



   public:
     // default constructor
     Order();

     // overloaded constructor
     Order( char porderID[] );

     //setters
     void setorderID(char porderID);

     //getters
     char getorderID();

     void deleteOrder(  );
     void culcAmount();
```

```
    //destructor
    ~Order();

};
```

Order.cpp

```
#include "Order.h"
#include<iostream>
#include<cstring>

using namespace std;

Order :: order()
{
  strcpy("orderID" , "") ;
}
Order :: order( char porderID[] )
{
  strcpy("orderID" , "porderID") ;
}

void Order :: setorderID(char porderID)
{


}

char Order :: getorderID()
{
```

```
}

void Order :: deleteOrder()
{

}

void Order:: culcAmount()
{

}

Order :: ~order()
{

}
```

Feedback.h

```
#include <iostream>
using namespace std;

Class Feedback{
private:
  char FeedbackID[10];
  char FeedbackDesc[100];

public:
  Feedback();

  Feedback(char FeedbackID[] , char FeedbackDesk[]);

  void setFeedbackID(char pFeedbackID[]);
  void setFeedbackDesk(char pFeedbackDesk[]);

  char getFeedbackID();
  char getFeedbackDesk();

  ~Feedback();
};
```

Feedback.cpp

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

 Feedback()
 {
  FeedbackID = " ";
  FeedbackDesk = " ";
 }

 Feedback(char pFeedbackID[] , char pFeedbackDesk[])
 {
  strcpy(FeedbackID, pFeedbackID) ;
  strcpy(FeedbackDesk , pFeedbackDesk);
 }

 void setFeedbackID(char pFeedbackID[])
 {
  //method implementation
 }
 void setFeedbackDesk(char pFeedbackDesk[])
```

```
{
  //method implementation
}

char getFeedbackID()
{
  //method implementation
}
char getFeedbackDesk()
{
  //method implementation
}

~Feedback()
{
  //method implementation
}
```

Discount.h

```
Class Discount{
 private:
      char DiscountID[10];
      double Amount;
      double UnitDiscount;
public:
 Discount();

 Discount(int pDiscountID , char pamount , int punitDiscount);

 void setDiscountID( char pDiscountID );
 void setAmount(double pamount) ;
 void setUnitDiscount(double unitDiscount) ;

 char getDiscountID();
 double setAmount();
 double setUnitDiscount();

 ~Discount();
};
```

Discount.cpp

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

Discount(){
 strcpy("DiscountID","");
 strcpy("Amount" , "");
 strcpy("UnitDiscount" , "");
}

 Discount(int pDiscountID , char pamount , int punitDiscount)
{
 strcpy("DiscountID","pDiscountID");
 strcpy("Amount" , "pamount");
 strcpy("UnitDiscount" , "punitDiscount");
}

 void setDiscountID( int pDiscountID )
{

}
```

```
 void setAmount(char pamount)
{


}
 void setUnitDiscount(ppudiscount)
{


}

 int getDiscountID()
{


}
 char setAmount()
{


}
 int setUnitDiscount()
{


}

 ~Discount()
{
```

```
}
```