Topic : Online Market Store

Group no : MLB_PRO_15

Campus : Malabe

Submission Date :17/05/2022

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**
IT19037066	De Silva W A D S	0769054835
IT19145112	Karunarathne K Y	0774308504
IT21046520	Nipuna H K	0771803088

1)

System Requirements for Online Market Store

- This system allows all users to look for online market store.
- Firstly, user can login to the system by providing their details such as user id and password.
- Any customer can sign up to become a member.
- User can view, update, and delete their user details.
- Admin can login by using admin name, contact number and email.
- Admin can view, update, and delete details.
- Once a customer filled the form, they can add their own information to the website.
- Customer can select any order available in the store.
- Customer can purchase and add to cart.
- Customer can update details.
- Customer can add items from the order list.
- Deliverers delivers orders.
- Customer can do their payments of order through the system as pay pal and credit card.
- User can add coupon details and change the total payment.
- Admin also needs to login to the system before do the activities.
- Allows account holders to track the order, view order details, get invoice in history.

Nouns/Verbs analysis

2)

Noun Analysis (Nouns are in red color)

- This system allows all users to look for online market store.
- Firstly, user can login to the system by providing their user details such as user id and password.
- Any customer can sign up and become a member.
- User can view, update, and delete their user details.
- Admin can login by using admin name, contact number and email.
- Admin can view, update, and delete details.
- Once a customer filled the form, they can add their own information to the website.
- Customer can select any order available in the store.
- Customer can purchase and add to cart.
- Customer can update details.
- Customer can add items from the order list.
- Deliverers delivers orders.
- Customer can do their payments of order through the system as pay pal and credit card.
- User can add coupon details and change the total payment.
- Admin also needs to login to the system before do the activities.
- Allows account holders to track the order, view order details, get invoice in history.

Verb Analysis (Verbs are in blue color)

- This system allows all users to look for online market store.
- Firstly, user can login to the system by providing their details such as user id and password.
- Any customer can sign up to become a member.
- User can view, update, and delete their user details.
- Admin can login by using admin name, contact number and email.
- Admin can view, update, and delete details.
- Once a customer filled the form, they can add their own information to the website.
- Customer can select any order available in the store.
- Customer can purchase and add to cart.
- Customer can update details.
- Customer can add items from the order list.
- Deliverers delivers orders.
- Customer can do their payments of order through the system as pay pal and credit card.
- User can add coupon details and change the total payment.
- Admin also needs to login to the system before do the activities.
- Allows account holders to track the order, view order details, get invoice in history.

Identified Classes using Noun Verb Analysis

Identified Classes:

- Admin
- User
- Customer
- Item
- Cart
- Order
- Payment
- History
- Delivery Information

Attributes:

- User ID
- Password
- Admin name
- Contact number
- Email

Out of scope:

- Pay pal
- Credit card
- Purchase

Redundant:

- User details
- Sign up

3)

CRC Cards for Online Market Store

Admin	
Responsibilities	Collaborations
login to the system	
Verify user accounts	User
View User Accounts	
Search User	
Delete User Accounts	
logout	

Customer	
Responsibilities	Collaborations
Pay Total Amount	Payment
Get Customer Details	
Update Profile	

User	
Responsibilities	Responsibilities
Account verification and Login to the system	Admin
Create User Account	
Account Update	
View Account	
Account Delete	

Order	
Responsibilities	Collaborations
Add Item to the cart	Cart
Calculate Total Price	

Cart	
Responsibilities	Collaborations
Add Item	Item
View Cart Details	
Update Quantity	
Delete Cart	
Checkout	

Payment	
Responsibilities	Collaborations
Set Payment ID	
Calculate Total Amount	
Pay Refund Amount	
Get Payment Details	Customer, History

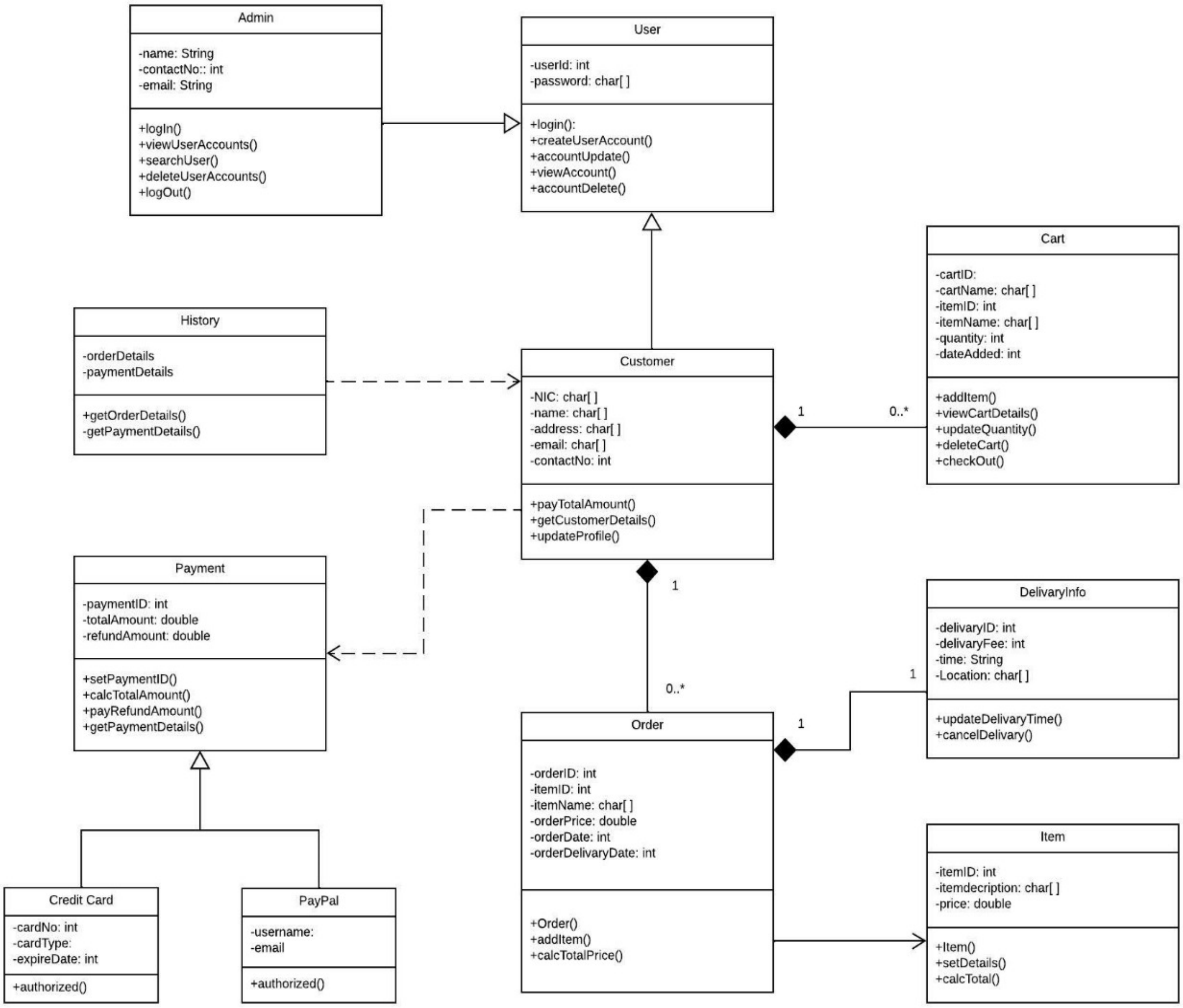
Delivery Information	
Responsibilities	Collaborations
Update Delivery time	
Cancel delivery	

History	
Responsibilities	Collaborations
Get Payment Details	Payment
Get Order Details	Order

Item	
Responsibilities	Collaborations
Set Details	
Calculate total Amount	Order

Exercise 1

Class Diagram for the Online Market Store

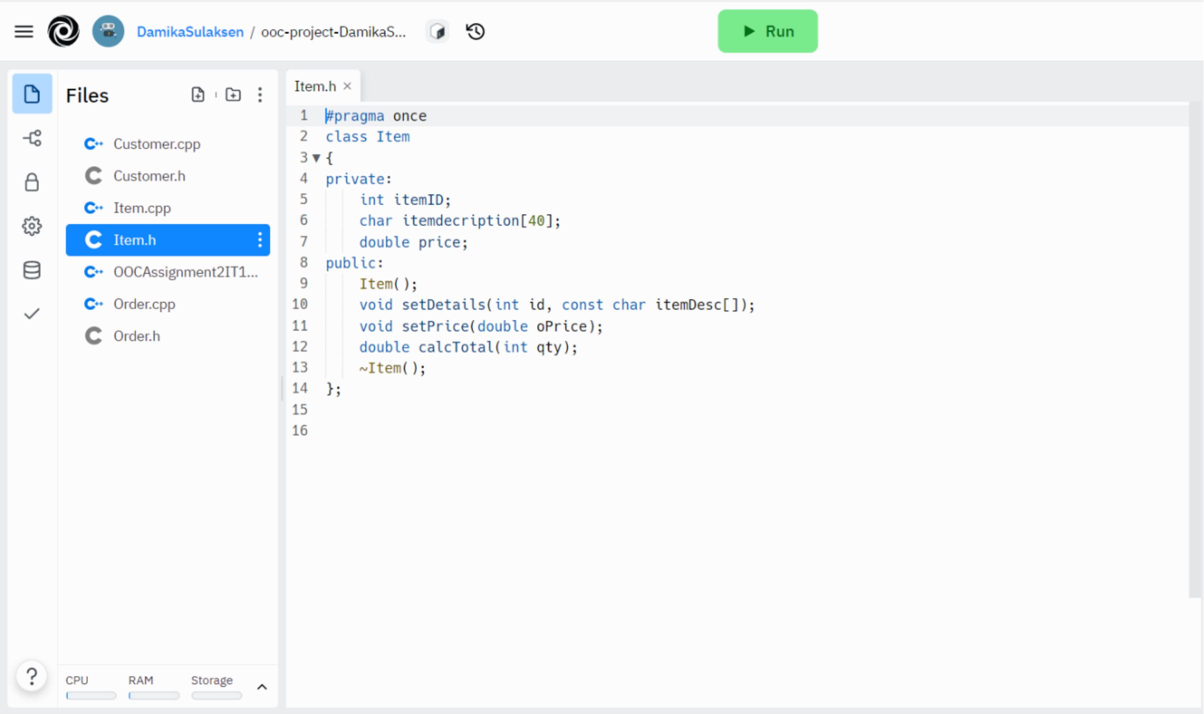


Exercise 2

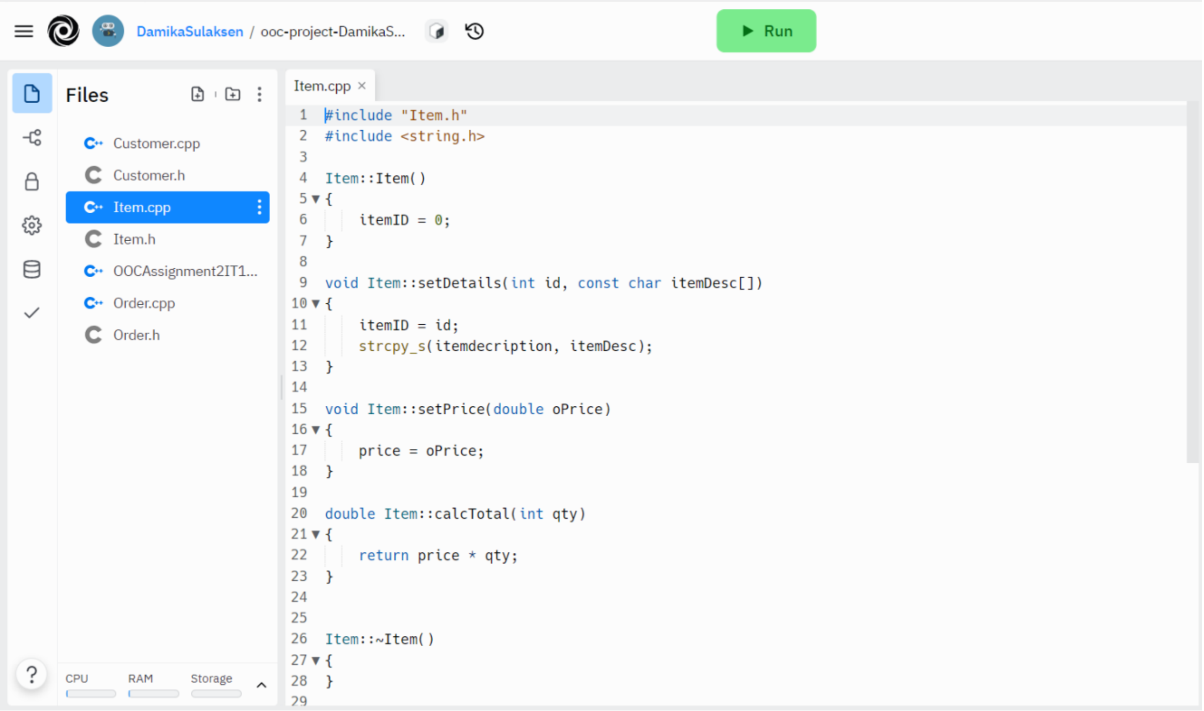
Coding for the Classes in Class Diagram

IT19037066 – De Silva W.A.D.S.

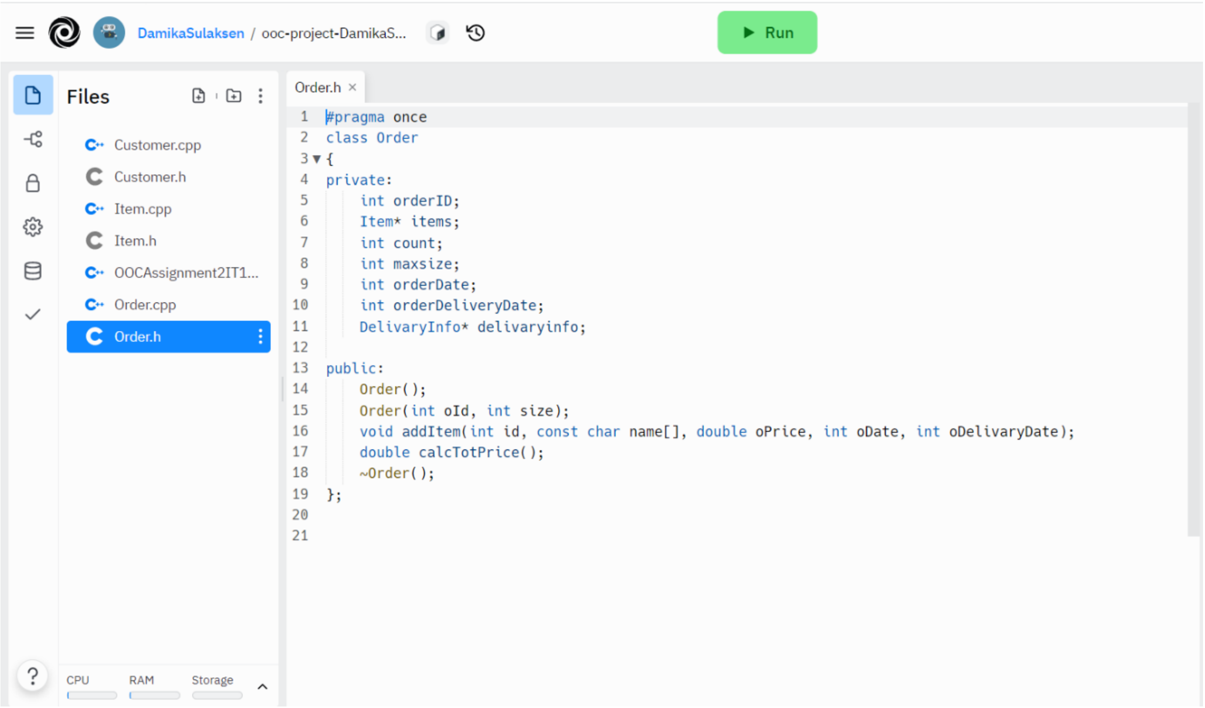
https://github.com/IT1050-2022-Feb/ooc-project-DamikaSulaksen



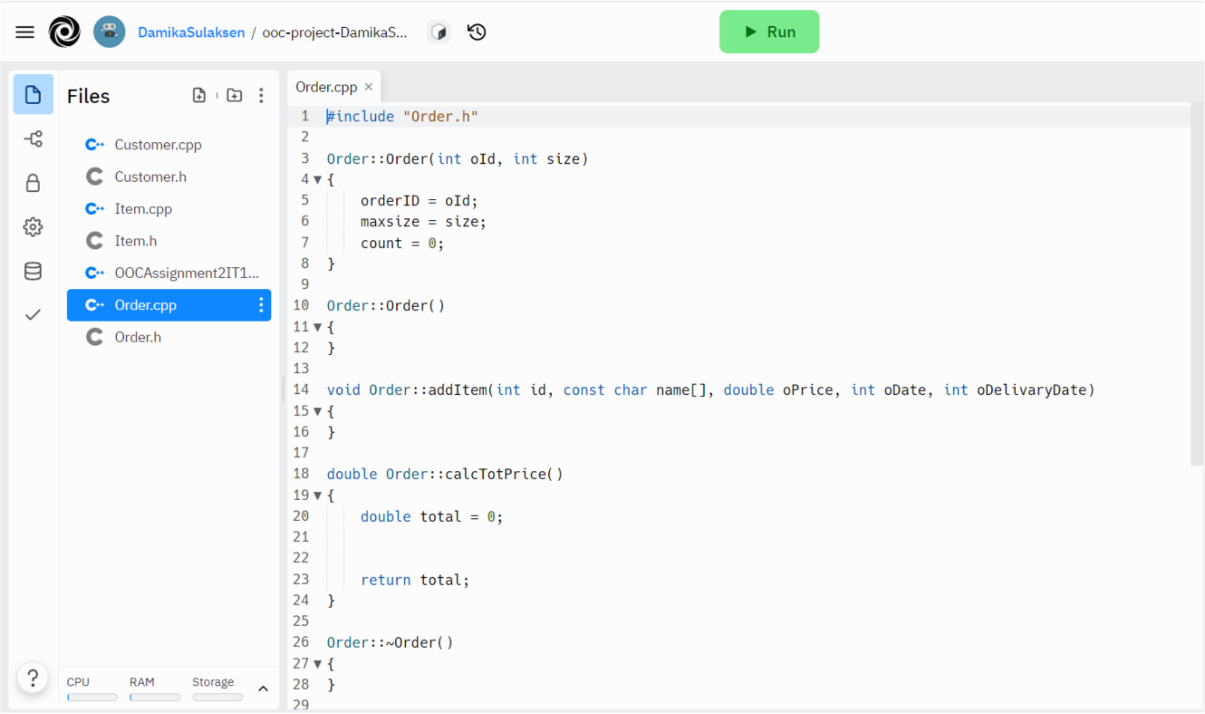
```cpp
#pragma once
class Item
{
private:
    int itemID;
    char itemdecription[40];
    double price;
public:
    Item();
    void setDetails(int id, const char itemDesc[]);
    void setPrice(double oPrice);
    double calcTotal(int qty);
    ~Item();
};


```



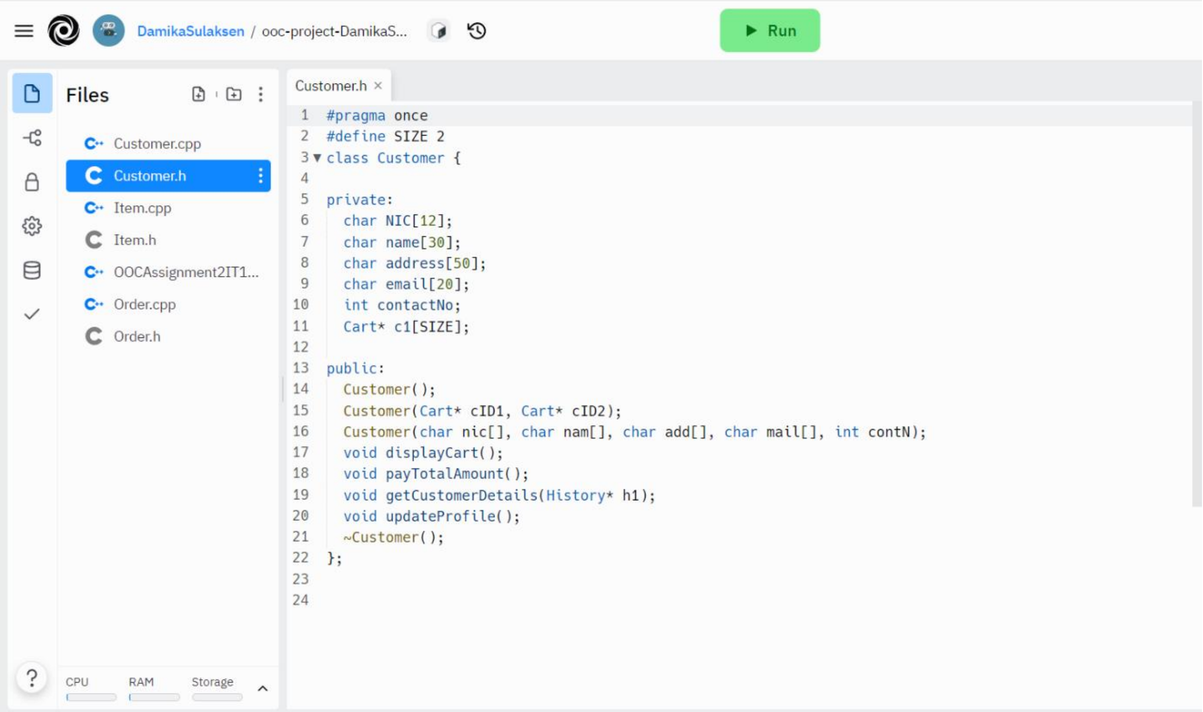
```cpp
#include "Item.h"
#include <string.h>

Item::Item()
{
    itemID = 0;
}

void Item::setDetails(int id, const char itemDesc[])
{
    itemID = id;
    strcpy_s(itemdecription, itemDesc);
}

void Item::setPrice(double oPrice)
{
    price = oPrice;
}

double Item::calcTotal(int qty)
{
    return price * qty;
}


Item::~Item()
{
}

```



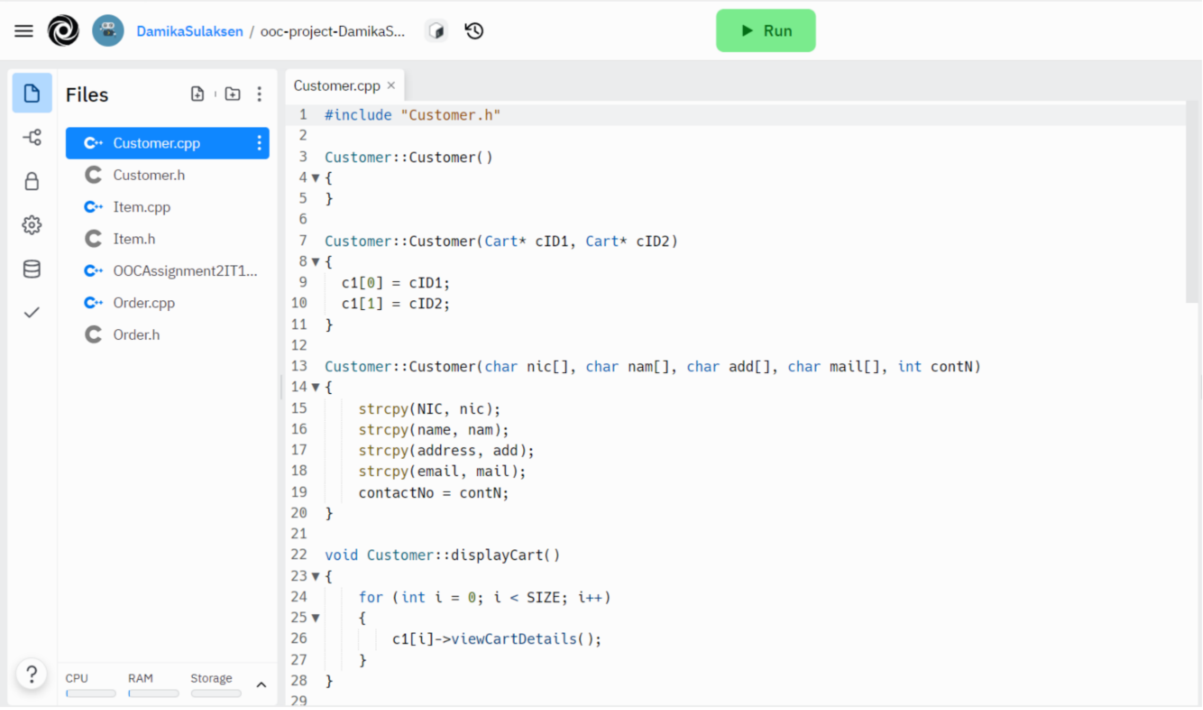
```
#pragma once
class Order
{
private:
    int orderID;
    Item* items;
    int count;
    int maxsize;
    int orderDate;
    int orderDeliveryDate;
    DelivaryInfo* delivaryinfo;

public:
    Order();
    Order(int oId, int size);
    void addItem(int id, const char name[], double oPrice, int oDate, int oDelivaryDate);
    double calcTotPrice();
    ~Order();
};


```



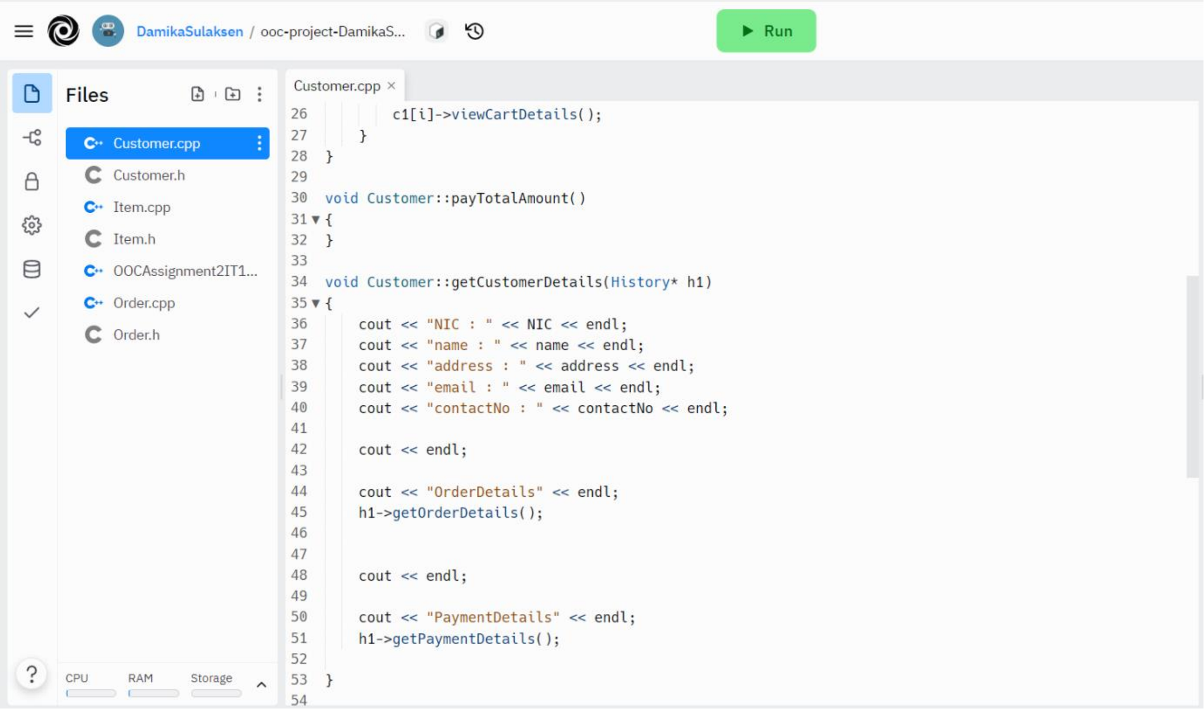
```
#include "Order.h"

Order::Order(int oId, int size)
{
    orderID = oId;
    maxsize = size;
    count = 0;
}

Order::Order()
{
}

void Order::addItem(int id, const char name[], double oPrice, int oDate, int oDelivaryDate)
{
}

double Order::calcTotPrice()
{
    double total = 0;


    return total;
}

Order::~Order()
{
}
```



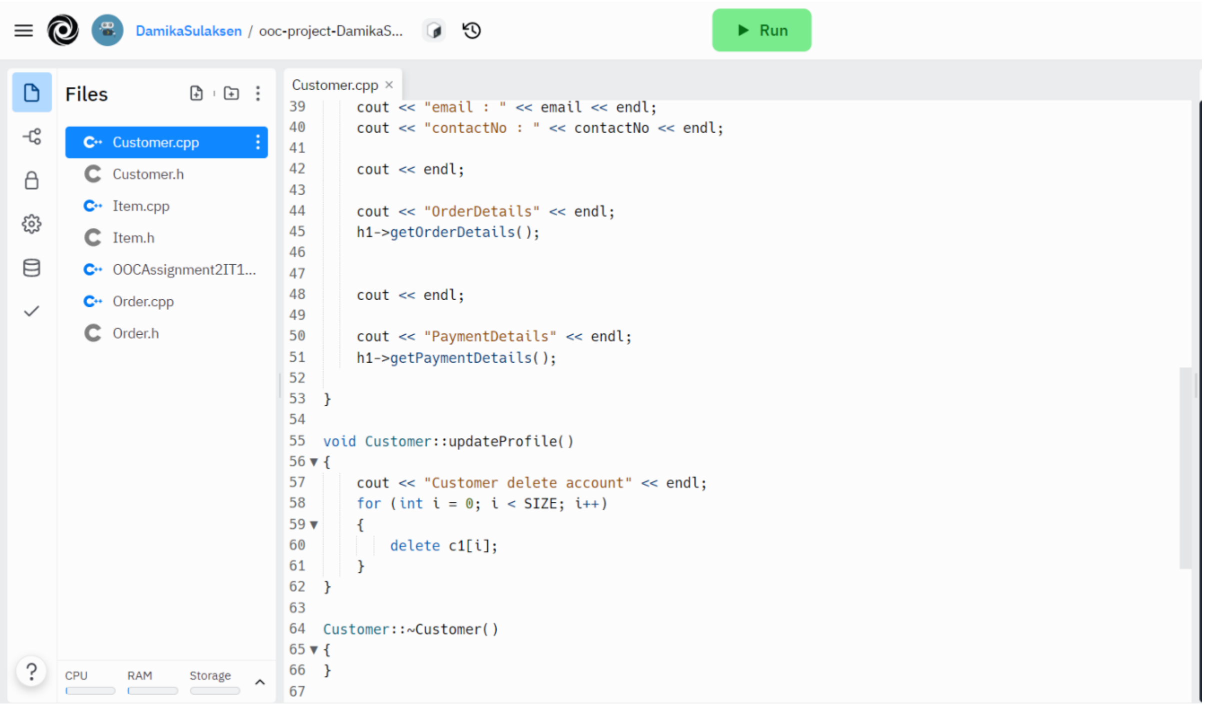
```cpp
#pragma once
#define SIZE 2
class Customer {

private:
  char NIC[12];
  char name[30];
  char address[50];
  char email[20];
  int contactNo;
  Cart* c1[SIZE];

public:
  Customer();
  Customer(Cart* cID1, Cart* cID2);
  Customer(char nic[], char nam[], char add[], char mail[], int contN);
  void displayCart();
  void payTotalAmount();
  void getCustomerDetails(History* h1);
  void updateProfile();
  ~Customer();
};


```



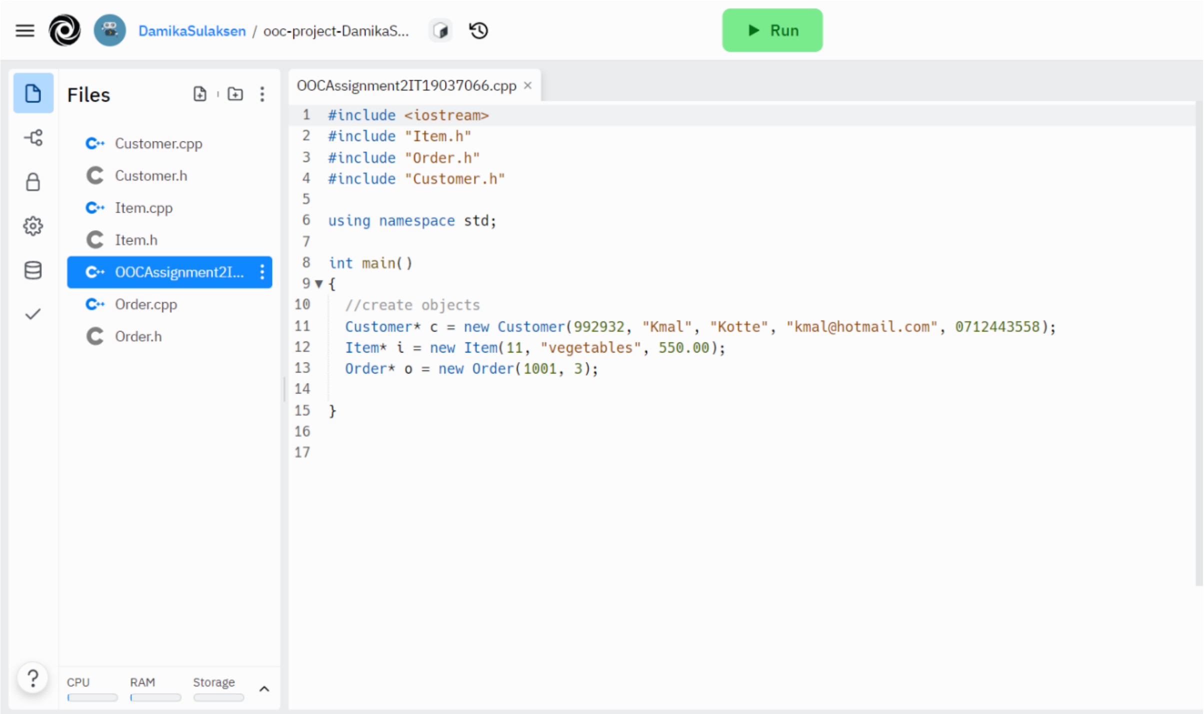
```cpp
#include "Customer.h"

Customer::Customer()
{
}

Customer::Customer(Cart* cID1, Cart* cID2)
{
  c1[0] = cID1;
  c1[1] = cID2;
}

Customer::Customer(char nic[], char nam[], char add[], char mail[], int contN)
{
    strcpy(NIC, nic);
    strcpy(name, nam);
    strcpy(address, add);
    strcpy(email, mail);
    contactNo = contN;
}

void Customer::displayCart()
{
    for (int i = 0; i < SIZE; i++)
    {
        c1[i]->viewCartDetails();
    }
}
```



```
        c1[i]->viewCartDetails();
    }
}

void Customer::payTotalAmount()
{
}

void Customer::getCustomerDetails(History* h1)
{
    cout << "NIC : " << NIC << endl;
    cout << "name : " << name << endl;
    cout << "address : " << address << endl;
    cout << "email : " << email << endl;
    cout << "contactNo : " << contactNo << endl;

    cout << endl;

    cout << "OrderDetails" << endl;
    h1->getOrderDetails();


    cout << endl;

    cout << "PaymentDetails" << endl;
    h1->getPaymentDetails();

}

```



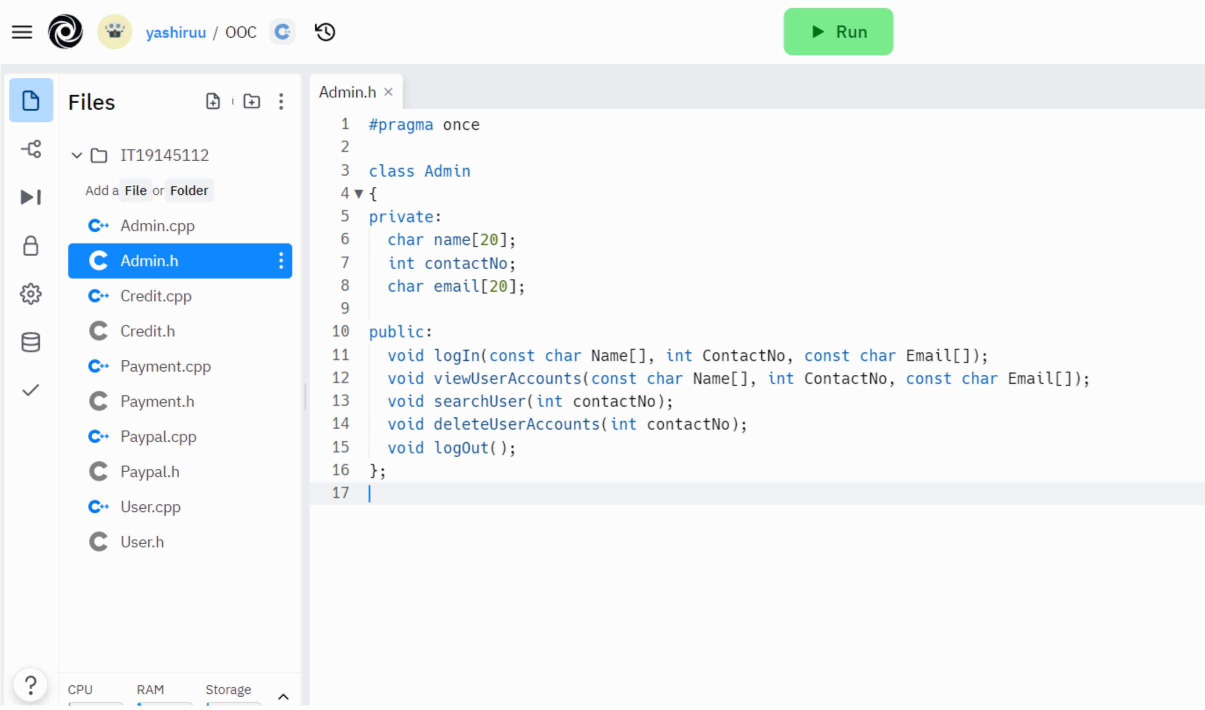
```
    cout << "email : " << email << endl;
    cout << "contactNo : " << contactNo << endl;

    cout << endl;

    cout << "OrderDetails" << endl;
    h1->getOrderDetails();


    cout << endl;

    cout << "PaymentDetails" << endl;
    h1->getPaymentDetails();

}

void Customer::updateProfile()
{
    cout << "Customer delete account" << endl;
    for (int i = 0; i < SIZE; i++)
    {
        delete c1[i];
    }
}

Customer::~Customer()
{
}

```



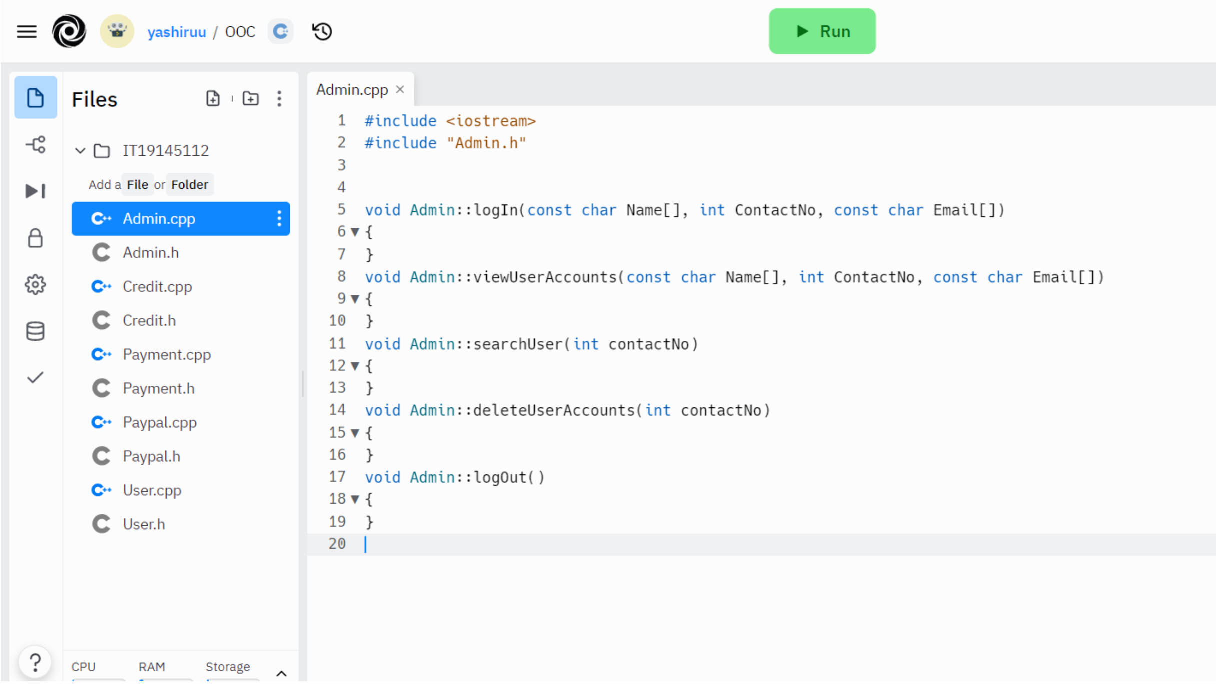
```cpp
#include <iostream>
#include "Item.h"
#include "Order.h"
#include "Customer.h"

using namespace std;

int main()
{
  //create objects
  Customer* c = new Customer(992932, "Kmal", "Kotte", "kmal@hotmail.com", 0712443558);
  Item* i = new Item(11, "vegetables", 550.00);
  Order* o = new Order(1001, 3);

}
```

IT19145112 – Karunarathne K Y

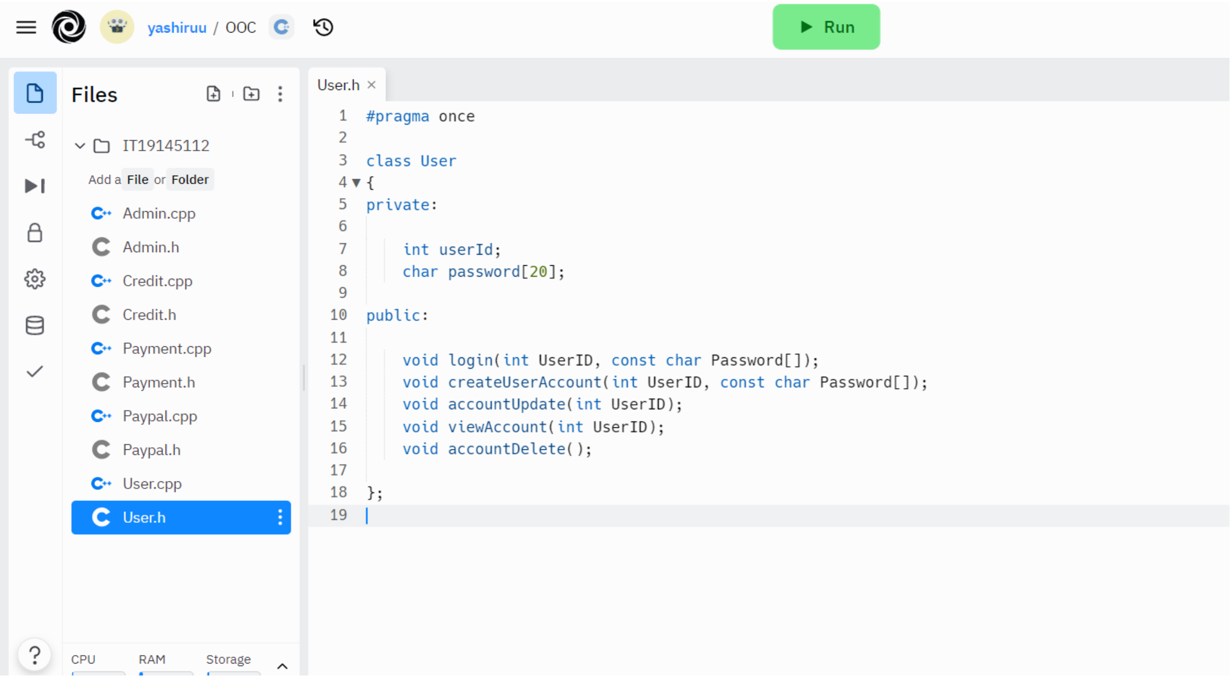
https://github.com/IT1050-2022-Feb/ooc-project-Kethmi1999



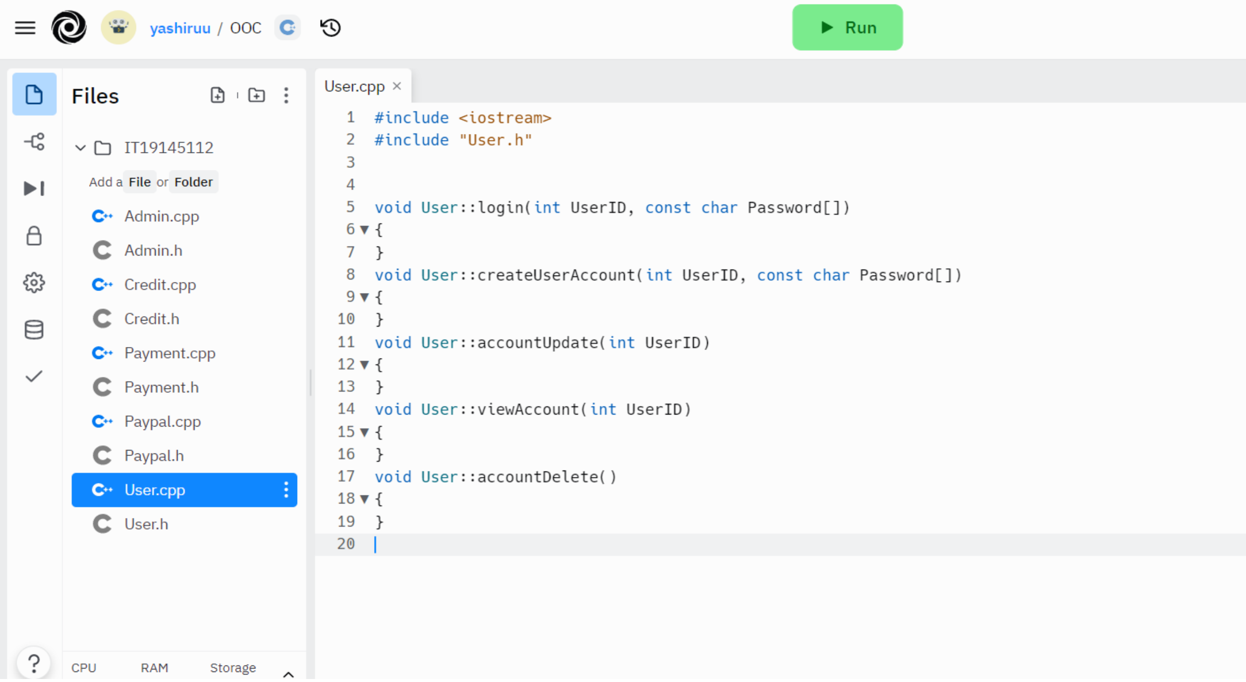
```cpp
#pragma once

class Admin
{
private:
  char name[20];
  int contactNo;
  char email[20];

public:
  void logIn(const char Name[], int ContactNo, const char Email[]);
  void viewUserAccounts(const char Name[], int ContactNo, const char Email[]);
  void searchUser(int contactNo);
  void deleteUserAccounts(int contactNo);
  void logOut();
};

```



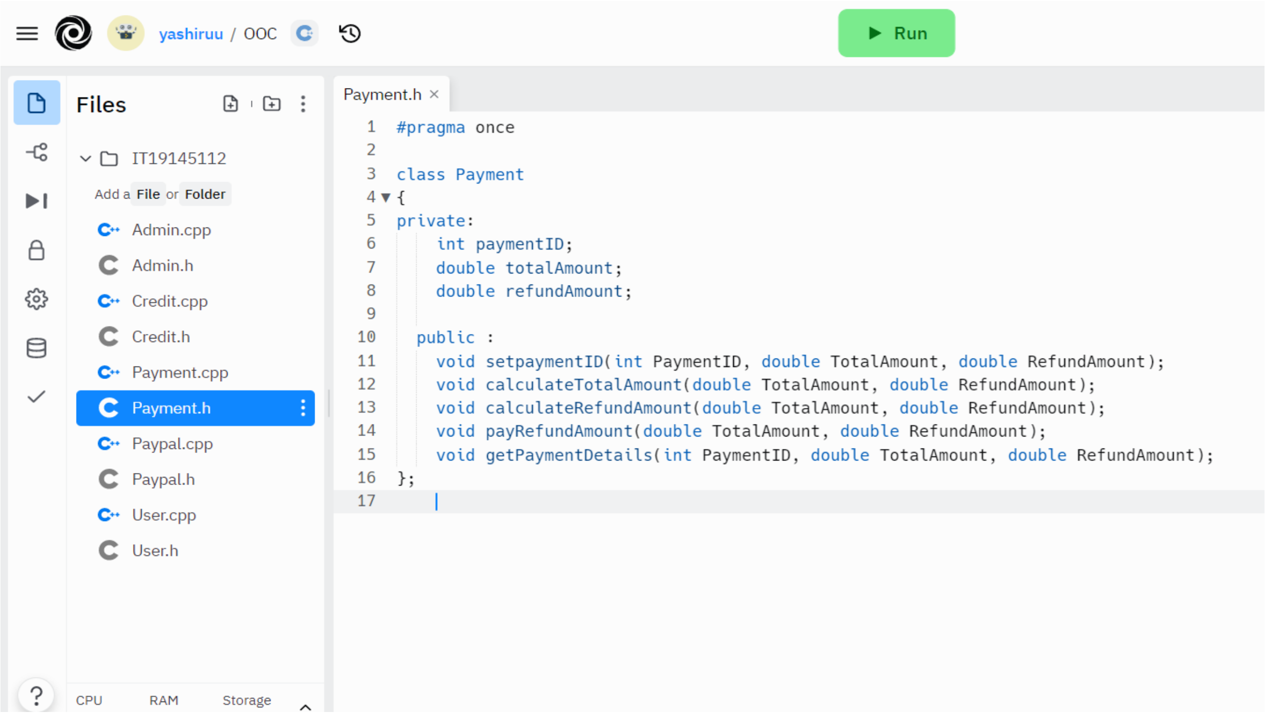
```cpp
#include <iostream>
#include "Admin.h"


void Admin::logIn(const char Name[], int ContactNo, const char Email[])
{
}
void Admin::viewUserAccounts(const char Name[], int ContactNo, const char Email[])
{
}
void Admin::searchUser(int contactNo)
{
}
void Admin::deleteUserAccounts(int contactNo)
{
}
void Admin::logOut()
{
}

```



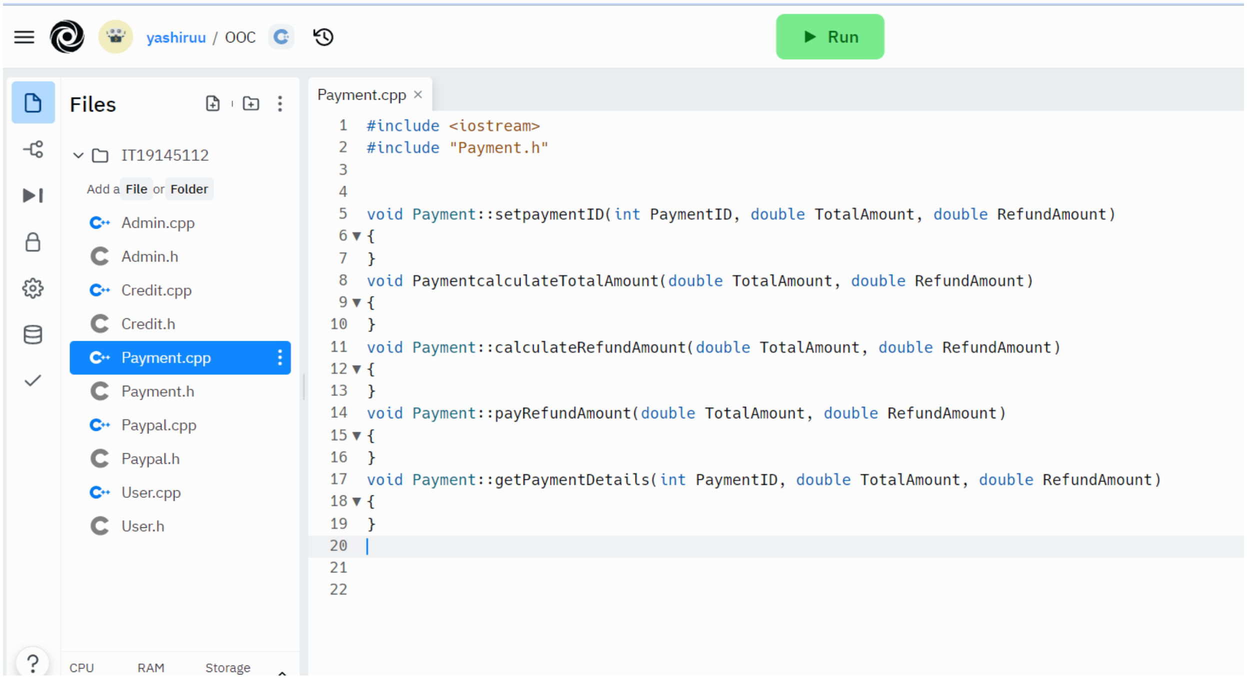
```cpp
#pragma once

class User
{
private:

    int userId;
    char password[20];

public:

    void login(int UserID, const char Password[]);
    void createUserAccount(int UserID, const char Password[]);
    void accountUpdate(int UserID);
    void viewAccount(int UserID);
    void accountDelete();

};
```



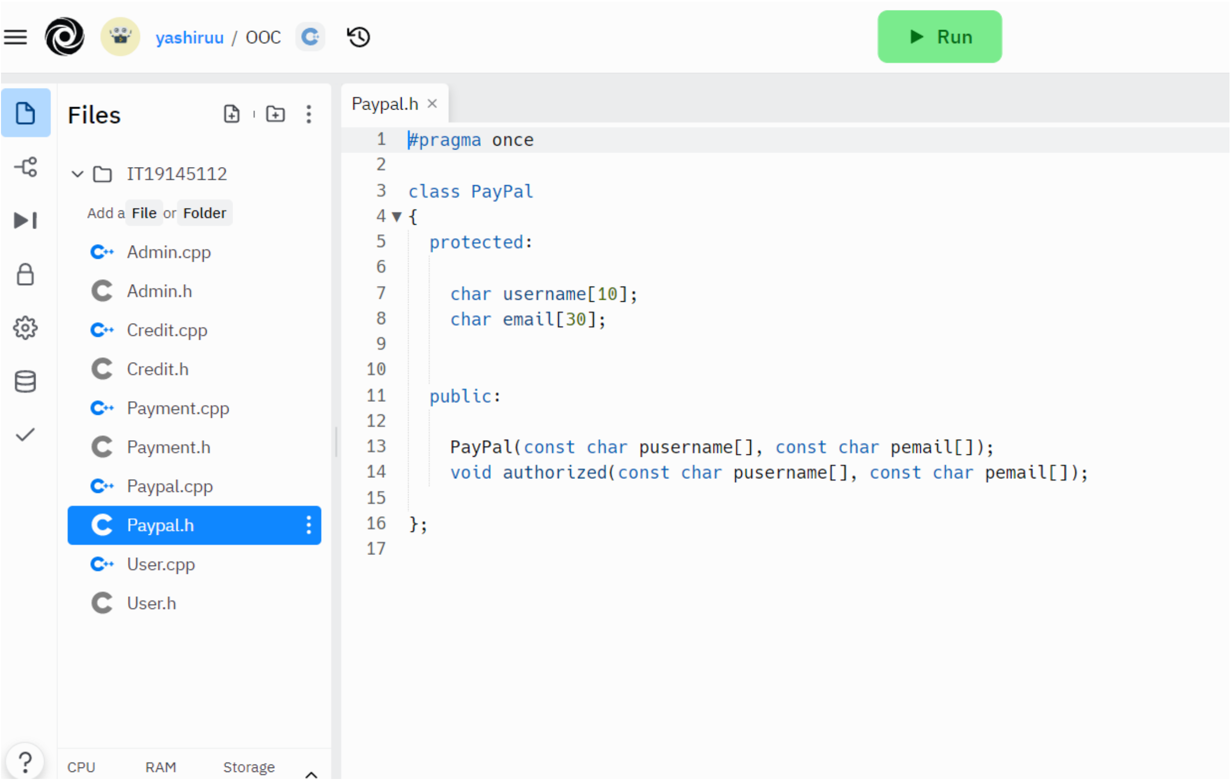
```cpp
#include <iostream>
#include "User.h"


void User::login(int UserID, const char Password[])
{
}
void User::createUserAccount(int UserID, const char Password[])
{
}
void User::accountUpdate(int UserID)
{
}
void User::viewAccount(int UserID)
{
}
void User::accountDelete()
{
}
```



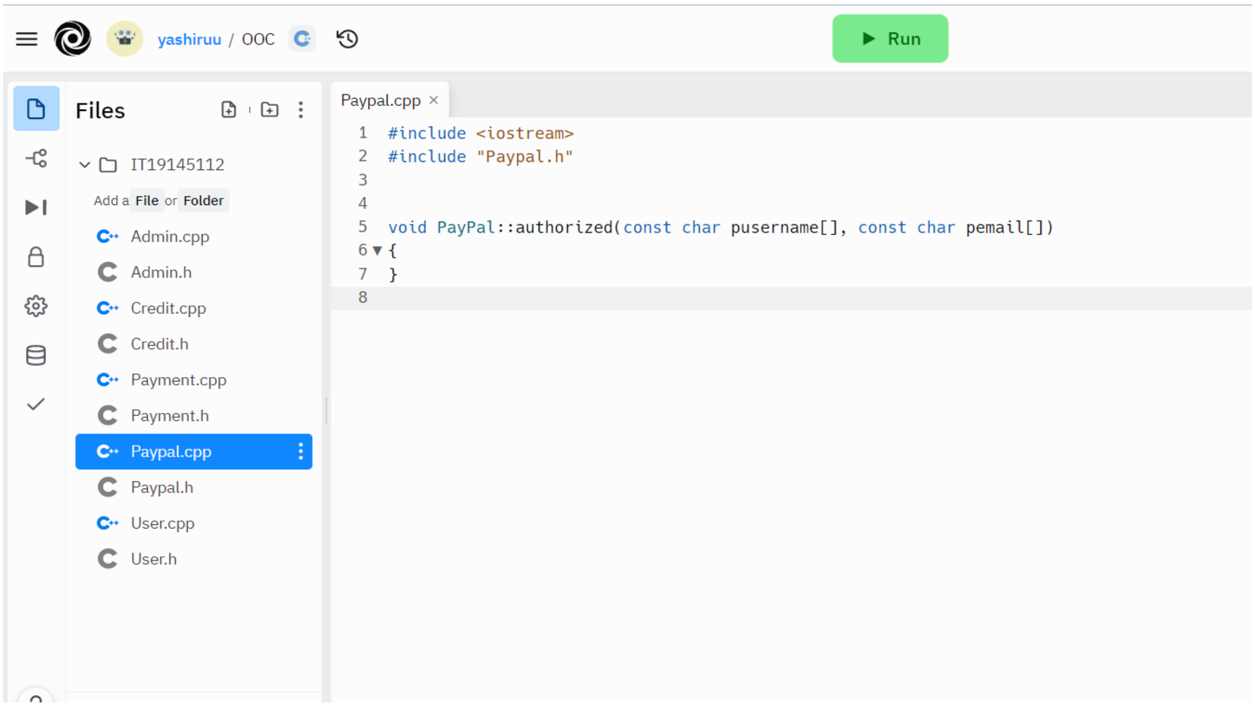
```cpp
#pragma once

class Payment
{
private:
    int paymentID;
    double totalAmount;
    double refundAmount;

  public :
    void setpaymentID(int PaymentID, double TotalAmount, double RefundAmount);
    void calculateTotalAmount(double TotalAmount, double RefundAmount);
    void calculateRefundAmount(double TotalAmount, double RefundAmount);
    void payRefundAmount(double TotalAmount, double RefundAmount);
    void getPaymentDetails(int PaymentID, double TotalAmount, double RefundAmount);
};

```



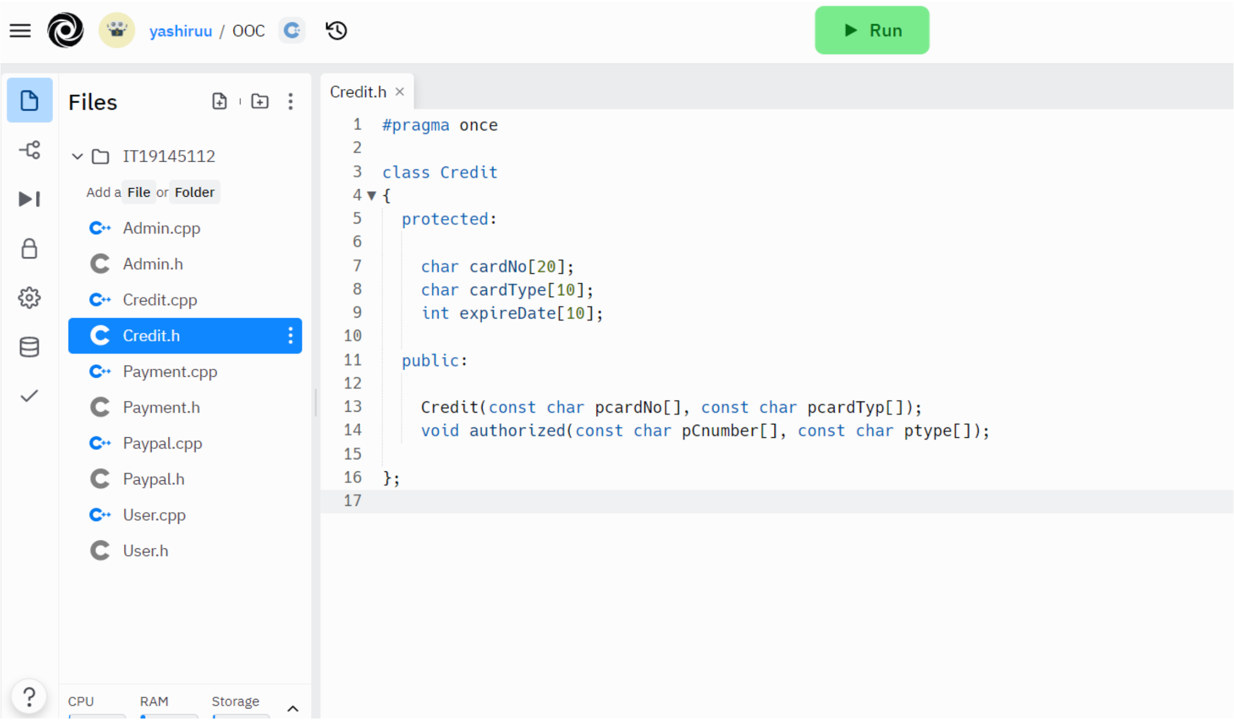
```cpp
#include <iostream>
#include "Payment.h"


void Payment::setpaymentID(int PaymentID, double TotalAmount, double RefundAmount)
{
}
void PaymentcalculateTotalAmount(double TotalAmount, double RefundAmount)
{
}
void Payment::calculateRefundAmount(double TotalAmount, double RefundAmount)
{
}
void Payment::payRefundAmount(double TotalAmount, double RefundAmount)
{
}
void Payment::getPaymentDetails(int PaymentID, double TotalAmount, double RefundAmount)
{
}



```



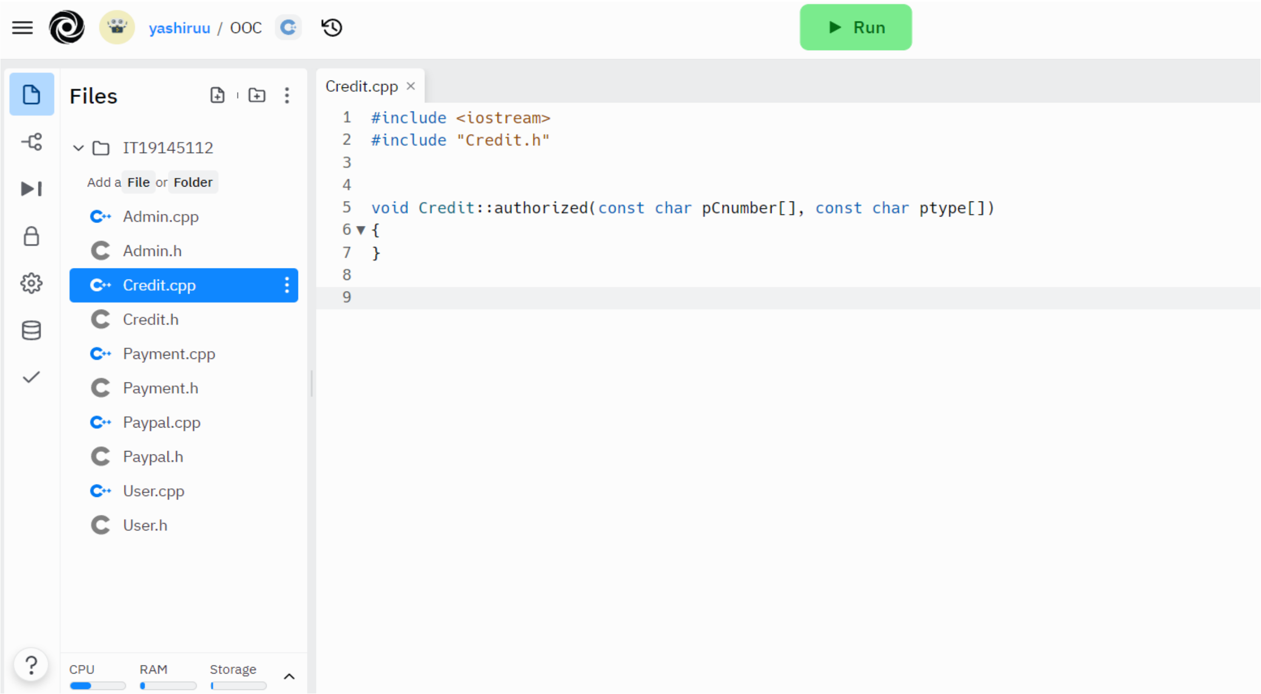
```
#pragma once

class PayPal
{
  protected:

    char username[10];
    char email[30];


  public:

    PayPal(const char pusername[], const char pemail[]);
    void authorized(const char pusername[], const char pemail[]);

};

```



```
#include <iostream>
#include "Paypal.h"


void PayPal::authorized(const char pusername[], const char pemail[])
{
}

```

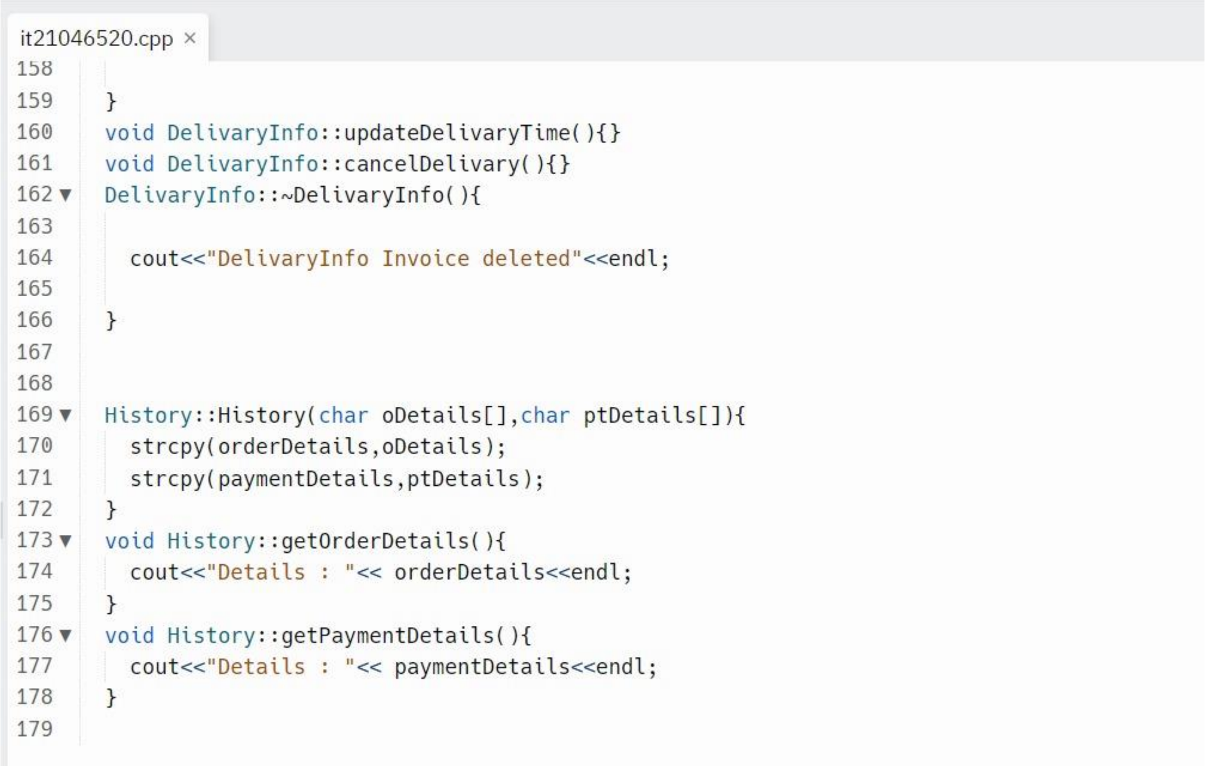


```
#pragma once

class Credit
{
  protected:

    char cardNo[20];
    char cardType[10];
    int expireDate[10];

  public:

    Credit(const char pcardNo[], const char pcardTyp[]);
    void authorized(const char pCnumber[], const char ptype[]);

};
```

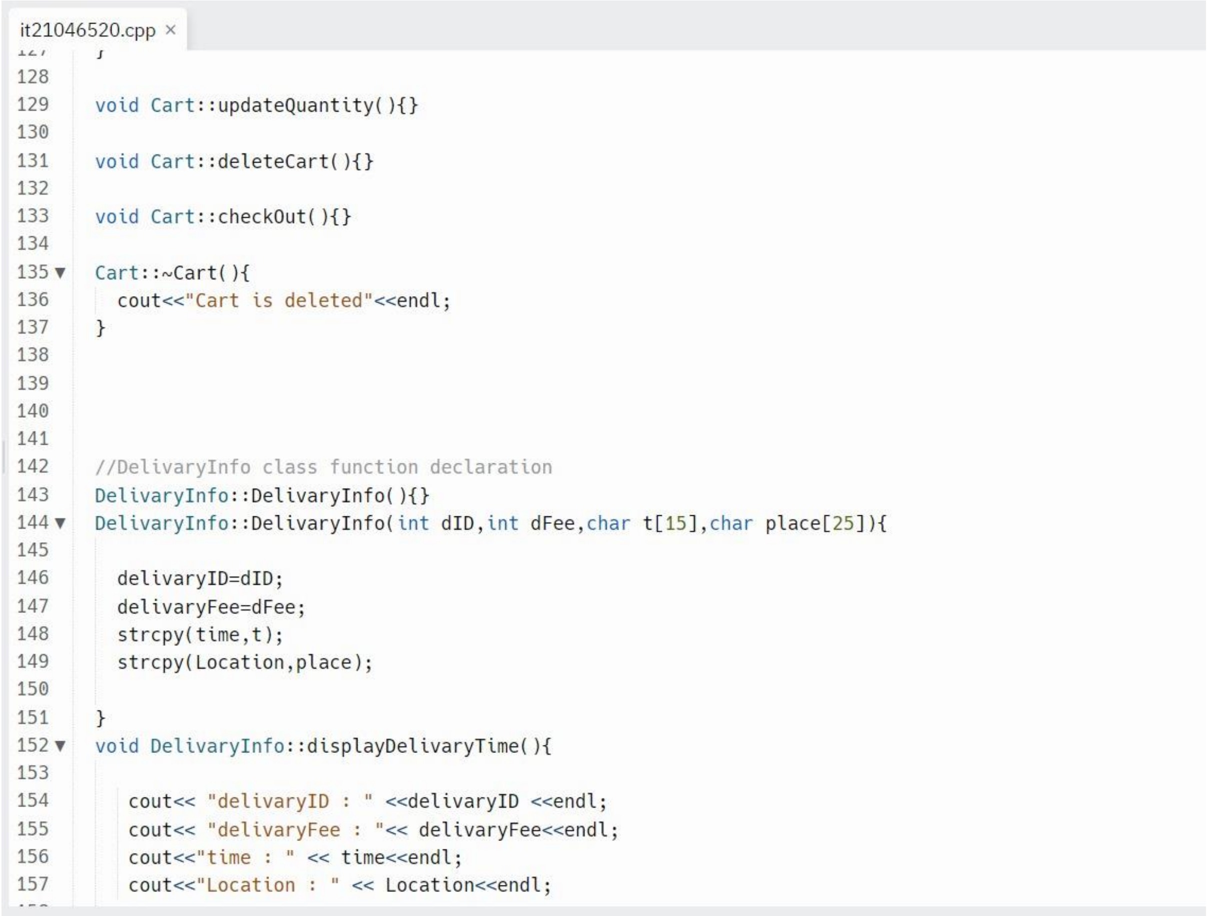


```
#include <iostream>
#include "Credit.h"


void Credit::authorized(const char pCnumber[], const char ptype[])
{
}
```

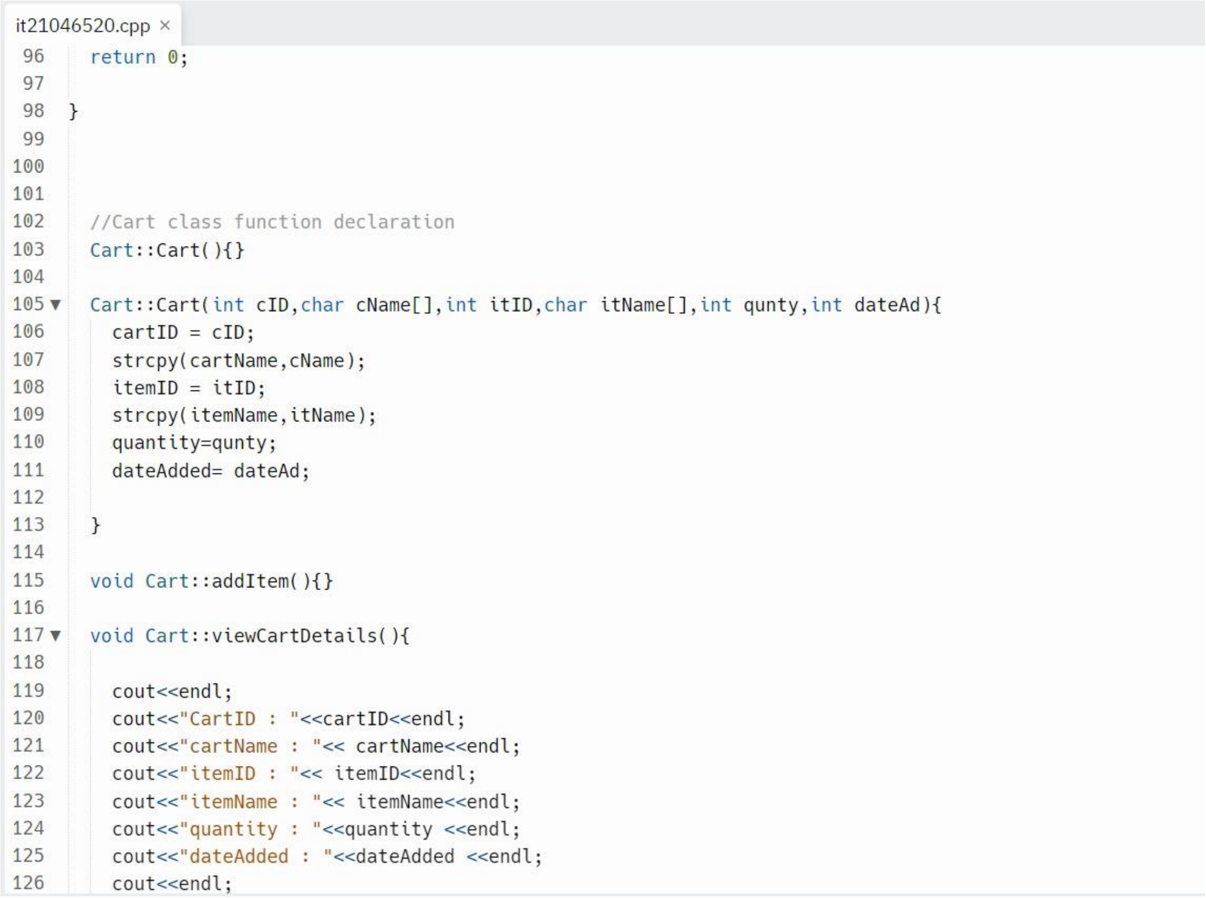
IT21046520 – Nipuna H K
https://github.com/IT1050-2022-Feb/ooc-project-it21046520



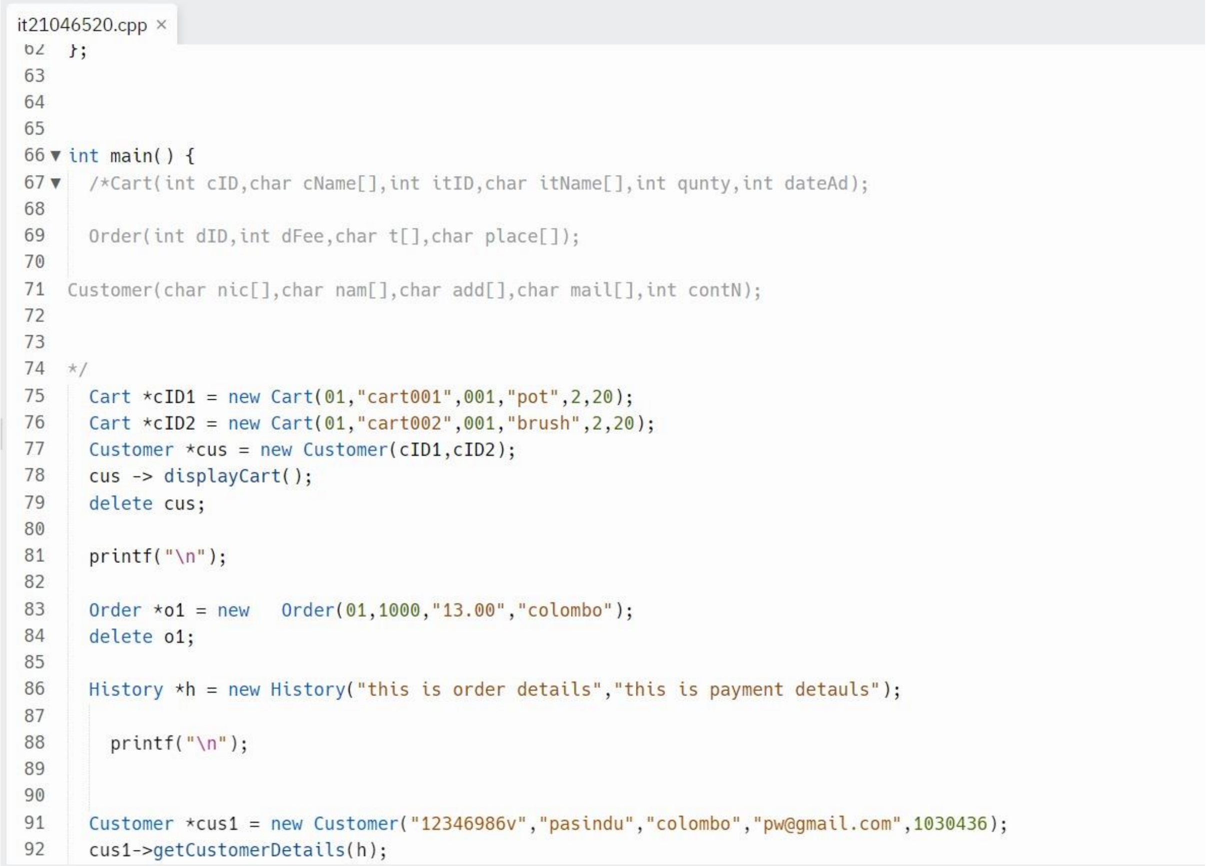
```
}
void DelivaryInfo::updateDelivaryTime(){}
void DelivaryInfo::cancelDelivary(){}
DelivaryInfo::~DelivaryInfo(){

  cout<<"DelivaryInfo Invoice deleted"<<endl;

}


History::History(char oDetails[],char ptDetails[]){
  strcpy(orderDetails,oDetails);
  strcpy(paymentDetails,ptDetails);
}
void History::getOrderDetails(){
  cout<<"Details : "<< orderDetails<<endl;
}
void History::getPaymentDetails(){
  cout<<"Details : "<< paymentDetails<<endl;
}

```



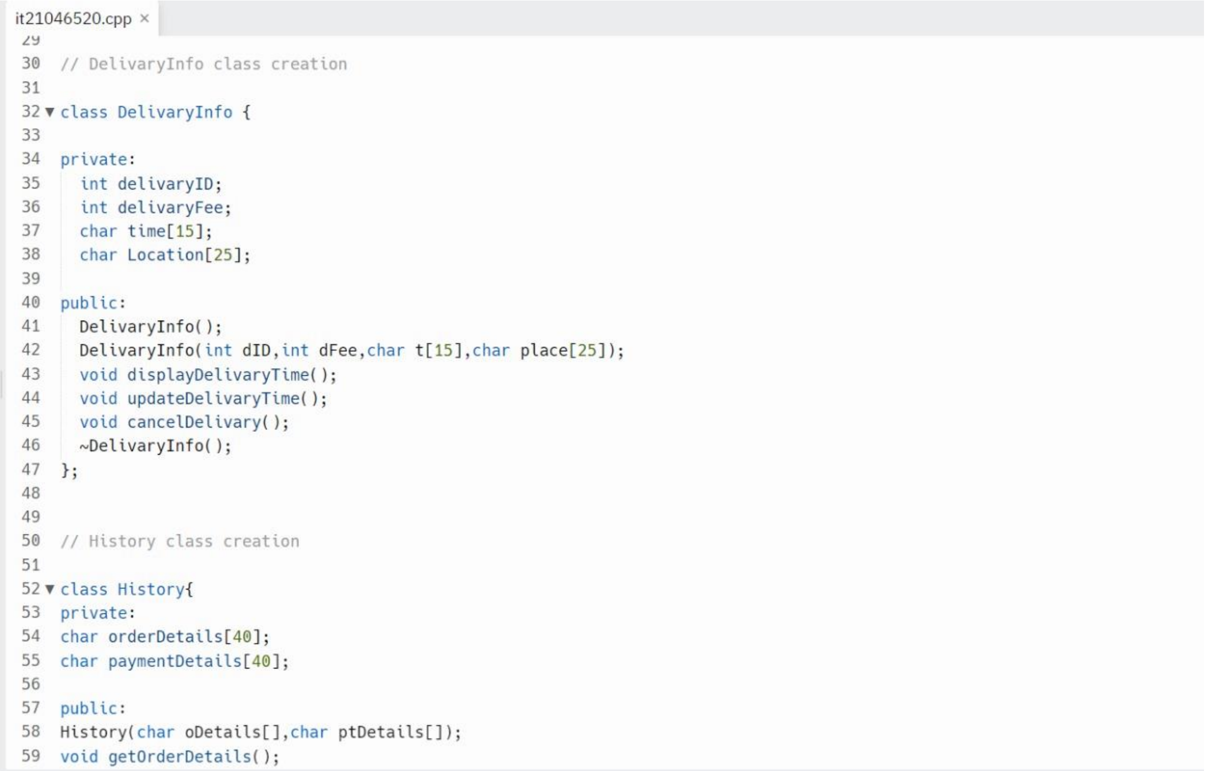
```
}

void Cart::updateQuantity(){}

void Cart::deleteCart(){}

void Cart::checkOut(){}

Cart::~Cart(){
  cout<<"Cart is deleted"<<endl;
}




//DelivaryInfo class function declaration
DelivaryInfo::DelivaryInfo(){}
DelivaryInfo::DelivaryInfo(int dID,int dFee,char t[15],char place[25]){

  delivaryID=dID;
  delivaryFee=dFee;
  strcpy(time,t);
  strcpy(Location,place);

}
void DelivaryInfo::displayDelivaryTime(){

   cout<< "delivaryID : " <<delivaryID <<endl;
   cout<< "delivaryFee : "<< delivaryFee<<endl;
   cout<<"time : " << time<<endl;
   cout<<"Location : " << Location<<endl;
```



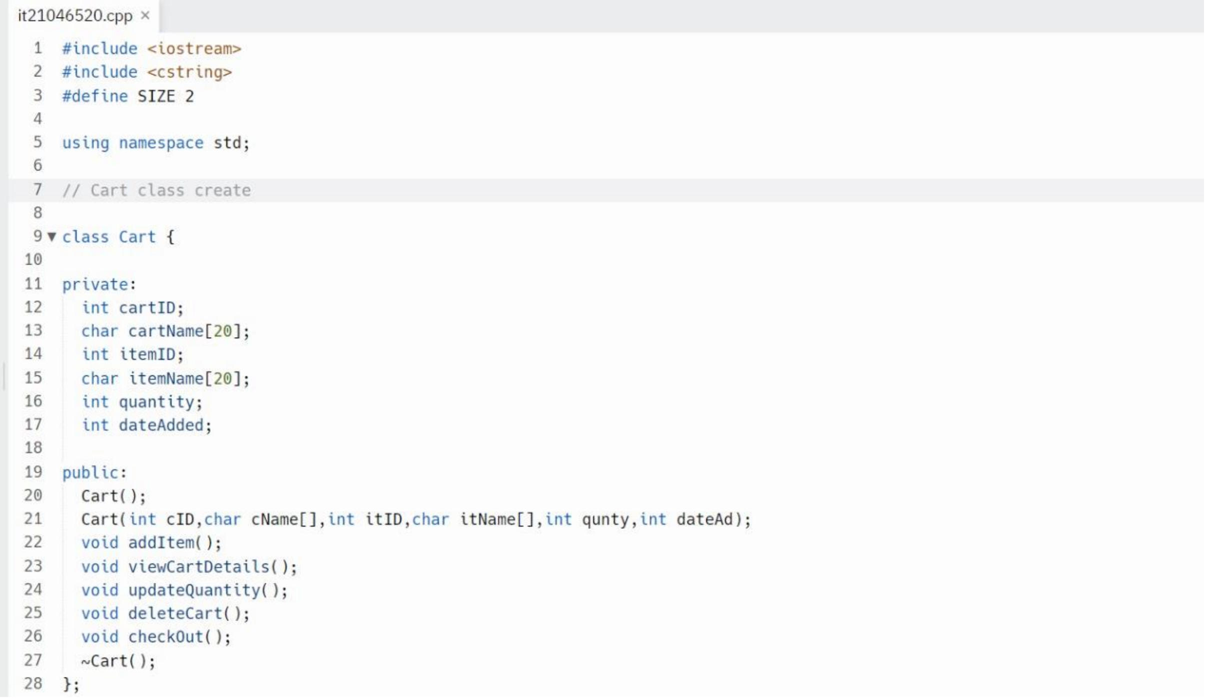
```cpp
  return 0;

}



  //Cart class function declaration
  Cart::Cart(){}

  Cart::Cart(int cID,char cName[],int itID,char itName[],int qunty,int dateAd){
    cartID = cID;
    strcpy(cartName,cName);
    itemID = itID;
    strcpy(itemName,itName);
    quantity=qunty;
    dateAdded= dateAd;

  }

  void Cart::addItem(){}

  void Cart::viewCartDetails(){

    cout<<endl;
    cout<<"CartID : "<<cartID<<endl;
    cout<<"cartName : "<< cartName<<endl;
    cout<<"itemID : "<< itemID<<endl;
    cout<<"itemName : "<< itemName<<endl;
    cout<<"quantity : "<<quantity <<endl;
    cout<<"dateAdded : "<<dateAdded <<endl;
    cout<<endl;
```



```cpp
};



int main() {
  /*Cart(int cID,char cName[],int itID,char itName[],int qunty,int dateAd);

  Order(int dID,int dFee,char t[],char place[]);

Customer(char nic[],char nam[],char add[],char mail[],int contN);


*/
  Cart *cID1 = new Cart(01,"cart001",001,"pot",2,20);
  Cart *cID2 = new Cart(01,"cart002",001,"brush",2,20);
  Customer *cus = new Customer(cID1,cID2);
  cus -> displayCart();
  delete cus;

  printf("\n");

  Order *o1 = new   Order(01,1000,"13.00","colombo");
  delete o1;

  History *h = new History("this is order details","this is payment detauls");

    printf("\n");


  Customer *cus1 = new Customer("12346986v","pasindu","colombo","pw@gmail.com",1030436);
  cus1->getCustomerDetails(h);
```



```cpp

// DelivaryInfo class creation

class DelivaryInfo {

private:
  int delivaryID;
  int delivaryFee;
  char time[15];
  char Location[25];

public:
  DelivaryInfo();
  DelivaryInfo(int dID,int dFee,char t[15],char place[25]);
  void displayDelivaryTime();
  void updateDelivaryTime();
  void cancelDelivary();
  ~DelivaryInfo();
};


// History class creation

class History{
private:
char orderDetails[40];
char paymentDetails[40];

public:
History(char oDetails[],char ptDetails[]);
void getOrderDetails();
```



```cpp
#include <iostream>
#include <cstring>
#define SIZE 2

using namespace std;

// Cart class create

class Cart {

private:
  int cartID;
  char cartName[20];
  int itemID;
  char itemName[20];
  int quantity;
  int dateAdded;

public:
  Cart();
  Cart(int cID,char cName[],int itID,char itName[],int qunty,int dateAd);
  void addItem();
  void viewCartDetails();
  void updateQuantity();
  void deleteCart();
  void checkOut();
  ~Cart();
};
```