Topic : Car rental system

Group no : MLB_WE_01.02_01

Campus : Malabe

Submission Date : 15/05/2022

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
IT21507960	Wijayasiri B.M.G	0741067877
IT21487484	Devinuwara D.L.N.A	0763494831
IT21473524	Abeywickrama W.N.V	0701539773
IT21192746	Megasooriya M.M.S.I.B	0702184683
IT21048746	Nethsara H.F.A	0767728795

Requirement Analysis

(NOUN in RED & VERB in BLUE)

1. Initially, the user visits the online car rental system through an advertisement displayed on social media and scroll through the website.
2. User needs to fill-out the registration details such as name, email, contact no, address, bank account no, NIC to get registered on the website.
3. User can either rent a car or reserve a car once logged in and get the confirmation e-mail.
4. If the user chooses the 'rent a car' option, the renter is directed to a webpage where vehicle details has to be provided to receive a renter id.
5. Renter can rent one or more vehicles once the renter ID is received.
6. If the user chooses the 'reserve a car' option, the Customer receives a customer id.
7. Customer can reserve one or more vehicles once the Customer ID is received.
8. Customer grants an opportunity to search many cars according to the preference.
9. Once a car on the website suits the customer's desires, the customer decides to reserve the car.
10. Then the customer navigates to the payout page, the reservation and payment details has to be filled.
11. A driver is assigned according to the customer's request.
12. Driver is notified about customer's request through our mobile application using his DID.
13. An Agent gets notified once the driver starts the journey and the vehicle route is tracked by GPS tracking system.
14. At the end of journey, the agent pays the driver according to mileage.
15. Once the vehicle is returned ,if there are any damages, customers owes an additional payment
16. The renter is paid by the agent after the car is returned.
17. Agent inspect the car rental report.

Identified Classes

1. Renter
2. User
3. Customer
4. Reservation
5. Payment
6. Employee
7. Driver
8. Agent
9. Vehicle
10. Report
11. Vehicle route
12. Check damage

CRC

Renter	
Responsibility	**Collaborators**
Register to the system	
Rent out vehicle	vehicle
Gives feedback	Feedback

Customer	
Responsibility	**Collaborators**
Register to the system	
Search and Reserve vehicle	Vehicle, Reservation
Make payment for the reservation	Payment
Gives feedback	Feedback

Reservation	
Responsibility	**Collaborators**
Update reservation	Vehicle , Driver, customer
Add reservation	Vehicle, Driver, customer
Cancel reservation	Vehicle, Driver, customer

Payment	
Responsibility	**Collaborators**
Get payment details	Customer
Validate payment details	

Driver	
Responsibility	**Collaborators**
Check notifications	
Confirm customer request	Customer

Damage	
Responsibility	**Collaborators**
Charge additional cost	Payment

Agent	
Responsibility	**Collaborators**
Respond to feedback	Feedback
Pays the renter	Renter
Pays the driver	Driver
Tracks the route	Vehicle route, vehicle

Vehicle	
Responsibility	**Collaborators**
Store vehicle details	Renter
Availability of cars	Reservation

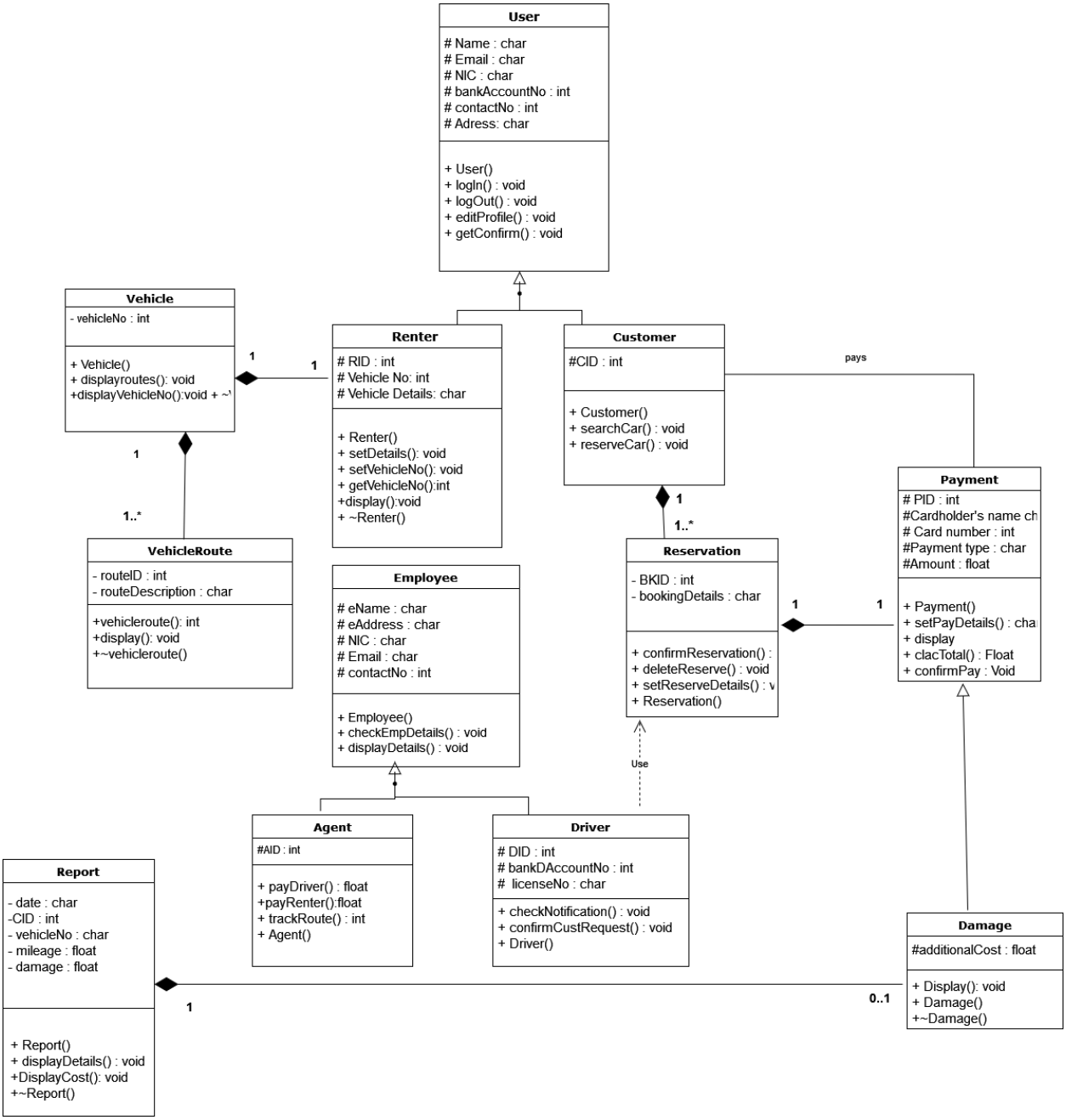
Report	
Responsibility	**Collaborators**
Generate customer details	Customer
Generate renter details	Renter
Generate payment details	Payment
Generate vehicle details	Vehicle

Vehicle route	
Responsibility	**Collaborators**
Add vehicle route	Vehicle, agent
Edit car routes	Vehicle, agent
Search routes	Vehicle, agent

User	
Responsibility	**Collaborators**
Login to the system	Renter, customer
Logout from the system	Renter, customer
Edit profile details	Renter, customer
Verify the account	Renter, customer

Employee	
Responsibility	**Collaborators**
Assign employee details	Agent , driver

Class diagram (UML notation)



Header files

User.h

```
#pragma once
class User
{
protected:
char Name[20];
char Email[30];
char NIC[12];
int bankAccountNo;
int contactNo;
char Address[50];

public:
User();
User(char pName[], char pEmail[], char pNIC[], int pBankAccountNo, int pContactNo, char pAddress[]);
void logIn();
void logOut();
void editProfile();
void getConfirm();
};
```

Renter.h

```
#include "User.h"
class Renter :public User{
private:
int RID;
int vehicleNo;
char vehicleDetails[20];
```

```
public:

Renter();

Renter(char pName[], char pEmail[], char pNIC[], int pBankAccountNo, int pContactNo, char pAddress[]);

void setDetails(int pRID, const char pdes[]);

void setVehicleNo(int pVehNo);

int getVehicleNo();

void display();

};
```

Customer.h

```
#include"User.h"

class Customer :public User

{

private:

int CID;

public:

Customer();

Customer(char pName[], char pEmail[], char pNIC[], int pBankAccountNo, int pContactNo, char pAddress[], int cCID);

void reserveCar();

void searchCar();

};
```

Employee.h

```
#pragma once
class Employee
{
	protected:
		char eName[35];
		char eAddress[50];
		char NIC[15];
		char Email[35];
		int contactNo;
	public:
		Employee();
		Employee(char pName[],char pAddress[],char pNIC[],char
   pEmail[],int pcontactNo);
		void checkEmpDetails();
		void displayDetails();

};
```

Agent.h

```
#include “Employee.h”
class Agent :public Employee
{
private:
	int AID;
public:
	Agent();
	Agent(char pName[],char pAddress[],char pNIC[],char pEmail[],int pcontactNo,int
pAID);
	float payRenter();
	float payDriver();
```

```
        int trackRoute();
};
```

Driver.h

```
#include “Employee.h”

class Driver :public Employee
{
private:
        int DID;
        int bankAccountNo;
        char licenceNo[15];



public:
        Driver();
        Driver(char pName[],char pAddress[],char pNIC[],char
  pEmail[],int pcontactNo,int pDID,int pbankAccountNo, char
  plicenceNo[]);
  void confirmCustRrequest(Reservation *R);
        void checkNotification();
};
```

Vehicle.h

```
#include “Renter.h”
#include “Vehicle.h”

class vehicle
{
        private:
```

```
        vehicleroute*route;
    public:
        vehicle(int no1);
        void displayroutes();
        ~vehicle();
};
```

Vehicle Route.h

```
class vehicleroute
{
    private:
        int routeID;
    public:
        vehicleroute(int no);
        void display();
        ~vehicleroute();
};
```

Payment.h

```
class Payment{
    private:
        int pID;
        int cardNumber;
        char name[8];
        char paymentType[4];
        float amount;
```

```
    public:
        Payment();
        void setPayDetails(int xpID, int xcardNumber, char xname[], char
xpaymentType[]);
        void display();
        float calcTotal();
        void confirmPay();

    protected:
        int pID;
        int cardNumber;
        char name[8];
        char paymentType[4];
        float amount;
};
```

Reservation .h

```
class Reservation {
    private:
        int bKID;
        char bookingDetails[20];
        double bookPrice;

    public:
        Reservation();
        Reservation(int bKID, char bookingDetails[20]);
        void setReservationDetails(int bKID, char bookingDetails[20]);
```

```
        void confirmReservation();
        void deleteReservation();

};
```

Report.h

```
class Report{
    private:
        Damage *dmg;//creating pointer
        char date[10];
        int CID;
        char vehicleno[10];
        float mileage;

    public:
        Report();
        Report(char pdate[],int CID,char pvehicleno[],float pmileage);
        Report(float Cost);
        void displayDetails();
        void DisplayCost();
        ~Report();
};
```

Damage.h

```
class Damage : public Payment{
    protected:
```

```
        float AdCost;
    public:
        Damage();
        Damage(float pAdCost);
        void Display();
        ~Damage();
};
```

.cpp files

User.cpp

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

User::User()
{
}
User::User(char pName[], char pEmail[], char pNIC[], int pBankAccountNo, int pContactNo, char pAddress[])
{
strcpy(Name, pName);
strcpy(Email, pEmail);
strcpy(NIC, pNIC);
bankAccountNo = pBankAccountNo;
contactNo = pContactNo;
strcpy(Address, pAddress);
}
```

```
void User::logIn()
{
cout<<"Welcome to Rent Me!"<<endl;
}
void User::logOut()
{
cout<<"logged out successfully"<<endl;
}

void User::editProfile()
{
}
void User::getConfirm()
{
}
```

Renter.cpp

```
#include<iostream>
#include<cstring>
#include "Renter.h"
using namespace std;
Renter::Renter()
{
RID = 0;
vehicleNo = 0;
strcpy(vehicleDetails, "");
}

Renter::Renter(char pName[], char pEmail[], char pNIC[], int pBankAccountNo, int
pContactNo, char pAddress[])
```

```
: User(pName, pEmail, pNIC, pBankAccountNo, pContactNo, pAddress)
{


}
void Renter::setDetails(int pRID, const char pdes[])
{
RID = pRID;
strcpy(vehicleDetails, pdes);
}
void Renter::setVehicleNo(int pVehNo)
{
vehicleNo = pVehNo;}
int Renter::getVehicleNo()
{
return vehicleNo;
}
void Renter::display()
{
cout<<"display vehicle"<<vehicleNo<<endl;
}
```

Customer.cpp

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


Customer::Customer()
{
CID = 0;
}
Customer::Customer(char pName[], char pEmail[], char pNIC[], int pBankAccountNo, int
pContactNo, char pAddress[], int cCID)
```

```
: User(pName, pEmail, pNIC, pBankAccountNo, pContactNo, pAddress)
{
CID = cCID;
}
void Customer::reserveCar()
{
}
void Customer::searchCar()
{

}
```

Employee.cpp

```
#include<iostream>
#include “Employee.h”
#include<cstring>
Employee::Employee()
{
        strcpy(eName, "");
        strcpy(eAddress, "");
        strcpy(NIC, "");
        strcpy(Email, "");
        contactNo = 0;
}

Employee::Employee(char pName[],char pAddress[],char pNIC[],char pEmail[], int pcontactNo)
{
        strcpy(eName, pName);
        strcpy(eAddress, pAddress);
```

```
	strcpy(NIC, pNIC);
	strcpy(Email, pEmail);
	contactNo = pcontactNo;
}

void Employee::checkEmpDetails()
{

}
void Employee::displayDetails()
{
}
```

Agent.cpp

```
#include<iostream>
#include “Agent.h”
#include <cstring>
#include “Employee.h”
Agent::Agent()
{
	AID = 0;
}

Agent::Agent(char pName[],char pAddress[],char pNIC[],char pEmail[], int pcontactNo,int cAID)
:Employee(pName, pAddress, pNIC, pEmail, pcontactNo)
{
	AID = cAID;
}

float Agent::payRenter()
```

```
{
	return 0.0;
}
float Agent::payDriver()
{
	return 0.0;
}

int Agent::trackRoute()
{
	return 0;
}
```

Driver.cpp

```
#include<iostream>
#include “Employee.h”
#include “Driver.h”
#include “Reservation.h”
#include <cstring>
Driver::Driver()
{
	DID = 0;
	bankAccountNo = 0;
	strcpy(licenceNo, "");

}
Driver::Driver(char pName[],char pAddress[],char pNIC[],char pEmail[], int pcontactNo, int pDID, int pbankAccountNo, char plicenceNo[] )
:Employee( pName, pAddress, pNIC, pEmail, pcontactNo)
```

```
{
        DID = pDID;
        bankAccountNo = pbankAccountNo;
        strcpy(licenceNo, plicenceNo);
}
void Driver::confirmCustRrequest(Reservation *R)
{
         R->confirmReservation();
}void Driver::checkNotification()
{
}
```

Vehicle Route.cpp

```
#include<iostream>
#include “Vehicle.h”
#include “Vehicleroute.h”

using namespace std;

vehicleroute::vehicleroute(int no){
routeID=no;
}
void vehicleroute::display(){
cout<<"route ID"<<routeID<<endl;
}
vehicleroute::~vehicleroute(){
cout<<"delete"<<routeID<<endl;
}
```

Vehicle.cpp

```
#include<iostream>
#include “Renter.h”
#include “Vechicle Route.h”
#include <cstring>

vehicle::vehicle(int no1)
{
rental=new Renter();

}
void vehicle::displayVehicleNo()
{
rental->display();
}
void vehicle::displayroutes()
{
route->display();
}
vehicle::~vehicle()
{
cout<<"delete vehicle"<<endl;
delete rental;
}
```

Payment.cpp

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

```
Payment::Payment()
{
	pID = 0;
	cardNumber = 0;
	strcpy(name,"");
	strcpy(paymentType,"");
}
	void Payment::setPayDetails(int xpID, int xcardNumber, char xname[], char
xpaymentType[])
{
	pID = xpID;
	cardNumber = xcardNumber;
	strcpy (name, xname);
	strcpy (paymentType, xpaymentType);
}

	void Payment::confirmPay()
{
}
void Payment::display()
{
		cout<<"PID: "<<pID<<endl;
			cout<<"Card holder's name: "<<name<<endl;
		cout<<"Card number: "<<cardNumber<<endl;
			cout<<"Payment Type: "<<paymentType<<endl;
}
		float calcTotal()
		{
			float amount;
```

```
                    return  amount;
}
```

Reservation.cpp

```
#include<iostream>
#include "Reservation.h"
#include<cstring>
using namespace std;
Reservation::Reservation()
{
        strcpy(bKID, "");
        strcpy(bookingDetails, "");
        int count = 0;
        bookPrice = 0;
        payment* payment[SIZE];
}
        Reservation::Reservation(int bKID, char bookingDetails)
{
        void Booking::calculateBookPrice(int pID, int cardNumber, char paymentType, flaot
amount)
{
if (count < SIZE)
{
        payment[count] = new Payment(pID, cardNumber, paymentType, amount);
        count++;
}
}
void Reservation::displayBookPrice()
{
}
void Reservation::confirmReservation()
```

```
{
}
Reservation::~Reservation()
{

for (int i = 0; i < SIZE; i++)
{
delete payment[i];
}
}
```

Damage.cpp

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

//Implementation of part class
Damage::Damage()
        {
                AdCost=0.0;
        }
Damage::Damage(float pAdCost)
        {
                AdCost=pAdCost;
        }
void Damage::Display()
        {
                        cout<<"Cost Damage: " <<AdCost<<endl;
        }
Damage::~Damage()
```

```
	{
		cout<<"delete "<<AdCost<<endl;
	}
```

Report.cpp

```
#include<iostream>
#include<cstring>
#include"damage.h"
#include "report.h"

using namespace std;
Report::Report()
	{
		strcpy(date, "");
		CID=0;
		strcpy(vehicleno, "");
		mileage=0.0;
	}
Report::Report(char pdate[],int pCID,char pvehicleno[],float pmileage)
{
		strcpy(date,pdate);
		CID=pCID;
		strcpy(vehicleno,pvehicleno);
		mileage=pmileage;
}

Report::Report(float Cost)
	{
		dmg= new Damage(Cost);
	}
```

```
void Report::displayDetails()
	{
		cout<<"Date		: " << date <<endl;
		cout<<"Customer's id	: " <<CID<<endl;
		cout<<"Vehicle No	: " <<vehicleno<<endl;
		cout<<"Mileage		: " << mileage <<endl;
	}
void Report::DisplayCost()
	{
		dmg->Display();
	}
Report::~Report()
	{
		cout<<endl;
			cout<<"Report deleting"<<endl;
			delete dmg;
	}
```

Main Program

```
#include "User.h"
#include "Renter.h"
#include "Customer.h"
#include “Employee.h”
#include “Agent.h”
#include “Driver.h”
#include “Reservation.h”
```

```
#include “Vehicle.h”
#include “Vehicleroute.h”
#include "Payment.h"
#include "damage.h"
#include "report.h"
#include<iostream>
using namespace std;
Int main()
{
//object creation
 Customer cus; // Object -
Renter ren; // Object -
 Employee* E1 = new Employee();
 Driver* D1 = new Driver();
Agent* A1 = new Agent();
 vehicle*myvehicle=new vehicle();
vehicle*myrent=new vehicle();
Report *myReport=new Report("2022-02-19",40001,"CAY-3261",520.0);
Payment cus1;
 myReport=new Report(8000.00);



//method calling
 cus.logIn();
```

```
cus.logOut();

cus.getConfirm();

cus.editProfile();

cus.reserveCar();

cus.searchCar();


ren.logIn();

ren.logOut();

ren.getConfirm();

ren.editProfile();

ren.setVehicleNo(3325);

ren.getVehicleNo();

myvehicle->displayroutes();


myrent->displayVehicleNo();

myReport ->displayDetails();

cus1.setPayDetails(2015,7854126, "Kamal", "Visa");

cus1.display();

myReport ->DisplayCost();
```

```
delete cus;
delete ren;
delete E1;
delete D1;
delete A1;
delete myvehicle;
delete myrent;
delete myReport;
delete cus1;
return 0;
}
```

Student contribution

Student ID	**Contribution**
IT21507960	Identified the requirements analysis Identified the classes Drew the class diagram Coded the vehicle, vehicle route and vehicle composition relationships.
IT21487484	Identified the requirements analysis Identified the classes Drew the class diagram Coded Inheritance relationship of user, customer and renter
IT21473524	Identified the requirements analysis Identified the classes Drew the class diagram Composition relationship between report and damage class. Coded Inheritance relationship of payment and damage.
IT21192746	Identified the requirements analysis Identified the classes Drew the class diagram Coded Inheritance relationship of employee, driver and agent. Coded dependency between driver and reservation.
IT21048746	Identified the requirements analysis Identified the classes Drew the class diagram Coded composition relationship between payment and reservation