Topic: Vehicle Rental System

Group no: MLB 09 01 12

Campus: Malabe

Submission Date: 14/05/2022

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
IT21169212	M.A.T.M Wanshathilaka	0711348175
IT21168154	K.K.C Rangana	0772918325
IT21168390	U.N.H Madhuwantha	0714601823
IT21170720	K.D.S.P Jayawickrama	0778689051
IT21168086	V.N Munasinghe	0718995589

❖ Content

❖ System Requirements

- The system works 24 hours 365 days
- Guest users can overview the system, to book a vehicle they must register with the system by giving details such as Name, Address, NIC, Email, Mobile number. System identify user by username and password.
- User can edit their details in profile and not allow to edit user id.
- User can check offers.
- Before book a vehicle, customer should be able to allow current location in their device.
- Customer can book more than one vehicle at a time and cancel booking into 5 minutes.
- After successfully booked vehicle, system will automatically mark selected vehicle or vehicles are already booked.
- After system generate will date and time driver When the driver arrives at customer.
- If customer place booking vehicle with driver administration staff confirm booking and the relevant vehicle will be sent to customer's location.
- System generates 'Pay ID' to record payment details and transaction details for give offers and for member ranking.
- Customer can pay charge after journey.
- Payment will verify by bank or other trusted resources.
- Else customer come our company area and contact our customer service and sign to agreement and pay key money. After verification process customer can bring booked vehicle. Oder can place through online or come to our company.
- System generates 'Pay ID' to record payment details and transaction details for give offers and for member ranking.
- After journey user sends feedback to the system and system add points to relevant user profile.
- After journey, check vehicle condition, if do not have any faults vehicle marked as available in the system by administration staff. Else vehicle is sent to repair center and give information to insurance company.

❖ Noun Analysis

- The system works 24 hours 365 days
- User can overview the system, to book a vehicle they must register with the system by giving details such as Name, Address, NIC, Email, Mobile number. System identify user by username and password.
- User can edit their details in profile and not allow to edit user id.
- User can check offers
- Before book a vehicle, customer should be able to allow current location in their device
- Customer can book more than one vehicle at a time and cancel booking into 5 minutes
- After successfully booked vehicle, system will automatically mark selected vehicle or vehicles are already booked
- After system generate will date and time driver When the driver arrives at customer
- If customer place booking vehicle with driver administration staff confirm booking and the relevant vehicle will be sent to customer's location
- System generates 'Pay ID' to record payment details and transaction details for give offers and for member ranking
- Customer can pay charge after journey
- Payment will verify by bank or other trusted resources
- Else, customer come our company area and contact our customer service and sign to agreement and pay key money. After verification process customer can bring booked vehicle. Oder can place through online or come to our company
- System generates 'Pay ID' to record payment details and transaction details for give offers and for member ranking
- After journey user sends feedback to the system and system add points to relevant user profile.
- User can rate our website.
- After journey, check vehicle condition, if do not have any faults vehicle marked as available in the system by administration staff. Else vehicle is sent to repair center and give information to insurance company

❖ Identified Classes

- Customer
- Vehicle
- Report
- Payment
- Offer
- Order
- Insurance
- Category
- Rate
- Feedback

❖ Applied Rules

- **Outside scope of the system –** System, Bank, Staff, Company
- **Redundant –** User, Vehicle, Customer
- **An event or operation –**
- **Meta language –** Their
- **Attribute –** Customer (Name, Address, NIC, Email, Mobile number),
  Payment (PayID)

❖ Verb Analysis

- The system works 24 hours 365 days.
- Guest users can overview the system, to book a vehicle they must register with the system by giving details such as Name, Address, NIC, Email, Mobile number. System identify user by username and password.
- User can edit their details in profile and not allow to edit user id.
- User can check offers.
- Before book a vehicle, customer should be able to allow current location in their device.
- Customer can book more than one vehicle at a time and cancel booking into 5 minutes.
- After successfully booked vehicle, system will automatically mark selected vehicle or vehicles are already booked.
- After system generate will date and time driver When the driver arrives at customer.
- If customer place booking vehicle with driver administration staff confirm booking and the relevant vehicle will be sent to customer's location.
- System generates 'Pay ID' to record payment details and transaction details for give offers and for member ranking.
- Customer can pay charge after journey.
- Payment will verify by bank or other trusted resources.
- Else customer come our company area and contact our customer service and sign to agreement and pay key money. After verification process customer can bring booked vehicle. Oder can place through online or come to our company.
- System generates 'Pay ID' to record payment details and transaction details for give offers and for member ranking.
- After journey user sends feedback to the system and system add points to relevant user profile.
- After journey, check vehicle condition, if do not have any faults vehicle marked as available in the system by administration staff. Else vehicle is sent to repair center and give information to insurance company.

❖ Methods

- Customer
    - Register to the system providing details
    - Generate User id
    - Log in to the system
    - Search vehicle
    - Book vehicle
    - Select vehicle
    - Do payments
- Vehicle
    - Generate vehicle id
    - Delete/update vehicle details
- Offer
    - Generate offer id
    - Check customer type
- Payment
    - Generate payment id
    - Calculate payment
    - Record payment
- Category
    - Display Category
    - Add/Update/Delete vehicle
- Insurance
    - Check Insurance type
    - Calculate insurance amount
- Rate
    - Add ratings
    - Update rating
    - Delete ratings

- Feedback
    - Manage Feedbacks
    - View feedbacks
    - Reply feedbacks
- Order
    - Display Order
    - Calculate order
    - Get order details
- Report
    - Display Payment Detail
    - Display Booking Vehicle Detail

❖ CRC Cards

Reports	
Responsibilities	**Collaborations**
Generate Booking vehicle details	
Generate payment details	

Vehicles	
Responsibilities	**Collaborations**
Add vehicle details	Category
Delete vehicle details	Category
Update vehicle details	Category

Customer	
Responsibilities	**Collaborations**
Register as member	
Login to the sytem	
Search vehicle	Category
Book a vehicle	Category
Make payment	Payment

Feedback	
Responsibilities	**Collaborations**
Make feedback	Users
View feedback	Admin / Manager
Reply feedbacks	Admin / Manager

Rates	
Responsibilities	**Collaborations**
Add rates	
Update rates	
Delete rates	

Offers	
Responsibilities	**Collaborations**
Get new offers	
Check offers	
Calculate offers	
Check customer type	Customer

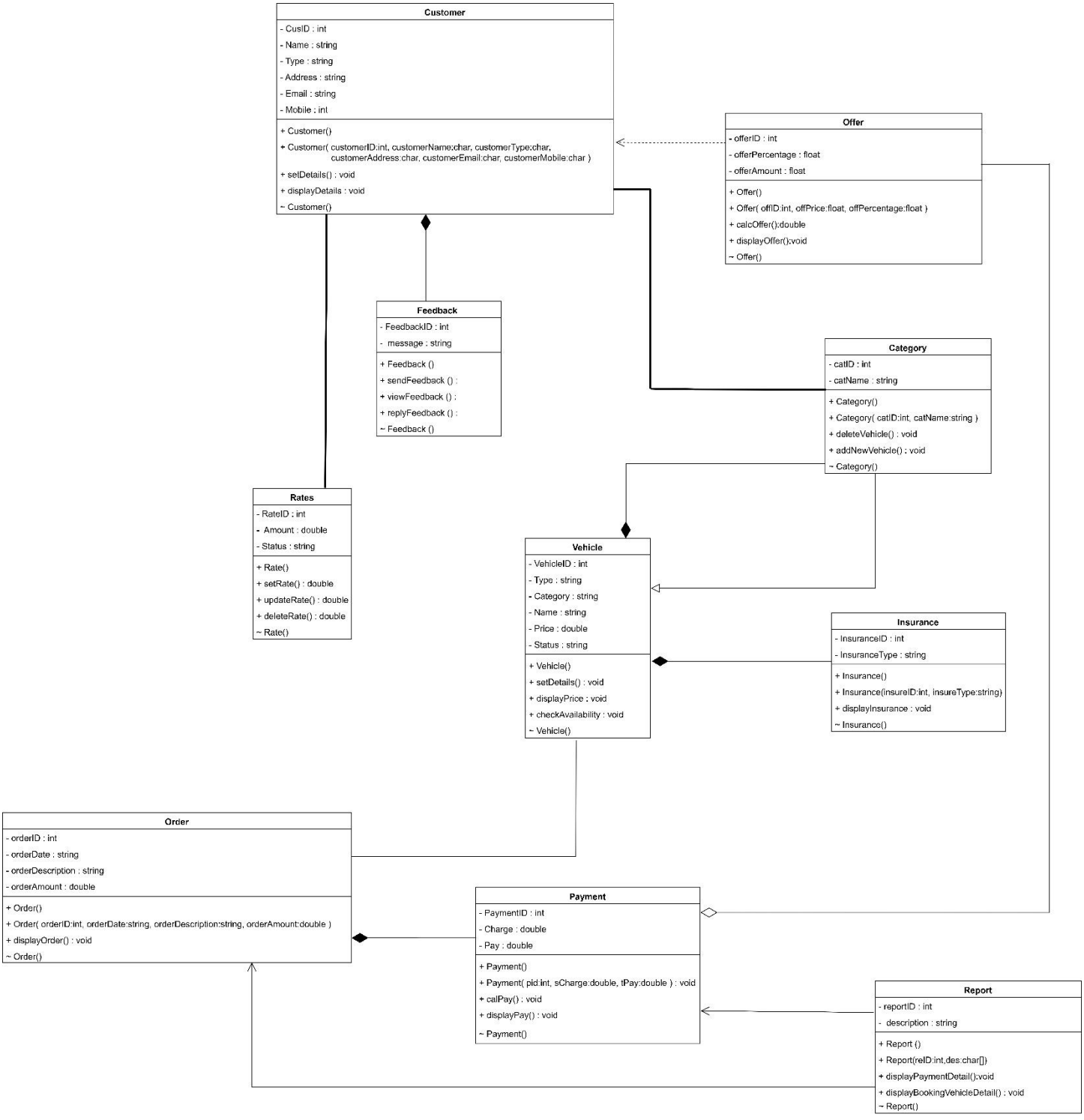
Payment	
Responsibilities	**Collaborations**
Make a payment	User
Generate payment ID	
Calculate payment	Order
Record payment in order	Order
Pay insurence	Insurence

Orders	
Responsibilities	**Collaborations**
Place order	
Display category	Category
Calculate rental charge	Payment

Insurance	
Responsibilities	**Collaborations**
Check insurence type	
Claim insurence	

Category	
Responsibilities	**Collaborations**
Delete vehicle	Vehicle
Add new Vehicle	Vehicle

❖ Class Diagram



Customer.h

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

class Customer{
  private:
    int cusID;
    string name;
    string type;
    string address;
    string email;
    int mobile;

  public:
    Customer();
    Customer(int customerID, string customerName, string customerType, string customerAddress, string customerEmail, int customerMobile);
    void setDetails();
    void displayDetails();
    ~Customer();
};
```

Customer.cpp

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

Customer::Customer(){
  cusID = 0;
  name = "";
  type = "";
  address = "";
  email = "";
  mobile = 0000000000;
}
Customer::Customer(int customerID, string customerName, string customerType, string customerAddress, string customerEmail, int customerMobile){
  cusID = customerID;
  name = customerName;
  type = customerType;
  address = customerAddress;
  email = customerEmail;
  mobile = customerMobile;
}
void Customer::setDetails(){

}
void Customer::displayDetails(){

}
Customer::~Customer(){

}
```

Category.h

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

class Category{

private:
  int catID;
  string catName;

public:
  Category();
  Category(int categoryID, string categoryName);
  void addNewVehicle();
  void deleteVehicle();
  ~Category();
};
```

Category.cpp

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

Category::Category(){
  catID = 0;
  catName = "";
}
Category::Category(int categoryID, string categoryName){
  catID = categoryID;
  catName = categoryName;
}
void Category::addNewVehicle(){

}
void Category::deleteVehicle(){

}
Category::~Category(){

}
```

Vehicle.h

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

class Vehicle{

private:
  int vehicleID;
  string name;
  string type;
  string category;
  string status;
  double price;

public:
  Vehicle();
  Vehicle(int vID, string vName, string vType, string vCat, string vStatus, double vPrice);
  void setDetails();
  double displayPrice();
  void checkAvailability();
  ~Vehicle();
};
```

Vehicle.cpp

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

Vehicle::Vehicle(){
  vehicleID = 1;
  name = "";
  type = "";
  category = "";
  status = "";
  price = 0.0;
}

Vehicle::Vehicle(int vID, string vName, string vType, string vCat, string vStatus, double vPrice){
  vehicleID = vID;
  name = vName;
  type = vType;
  category = vCat;
  status = vStatus;
}
void Vehicle::setDetails(){
}
double Vehicle::displayPrice(){
}
void Vehicle::checkAvailability(){
}
Vehicle::~Vehicle(){
}
```

Insurance.h

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

class Insurance{
private:
  int insuranceID;
  string insuranceType;

public:
  Insurance();
  Insurance(int insureID, string insureType);
  void displayInsurance();
  ~Insurance();
};
```

Insurance.cpp

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

Insurance::Insurance(){
  insuranceID = 0;
  insuranceType = "";
}

Insurance::Insurance(int insureID, string insureType){
  insuranceID = insureID;
  insuranceType = insureType;
}

void Insurance::displayInsurance(){

}

Insurance::~Insurance(){

}
```

Offer.h

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

class Offer{

private:
  int offerID;
  float offerPercentage;
  float offerAmount;

public:
  Offer();
  Offer(int offID, float offPrice, float offPercentage);
  double calcOffer();
  void displayOffer();
  ~Offer();
};
```

Offer.cpp

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

  Offer::Offer(){
    offerID = 0;
    offerPercentage = 0;
    offerAmount = 0;
  }
  Offer::Offer(int offID, float offPrice, float offPercentage){
    offerID = offID;
    offerPercentage = offPrice;
    offerAmount = offPercentage;
  }
  double Offer::calcOffer(){

  }
  void Offer::displayOffer(){

  }
  Offer::~Offer(){

  }
```

Order.h

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

class Order{

private:
  int orderID;
  string orderDate;
  string orderDescription;
  double orderAmount;

public:
  Order();
  Order(int oID, string oDate, string oDescription, double oAmount);
  void displayOrder();
  ~Order();
};
```

Order.cpp

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

Order::Order(){
  orderID = 0;
  orderDate = "";
  orderDescription = "";
  orderAmount = 0;
}
Order::Order(int oID, string oDate, string oDescription, double oAmount){
  orderID = oID;
  orderDate = oDate;
  orderDescription = oDescription;
  orderAmount = oAmount;
}
void Order::displayOrder(){
}

Order::~Order(){

}
```

Rate.h

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

class Rate{
  private:
    int rateID;
    double amount;
    string status;

  public:
    Rate();
    Rate(int rate_ID, double rate_amount, string rate_status);
    int setRate();
    double updateRate();
    double deleteRate();
    ~Rate();
};
```

Rate.cpp

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

Rate::Rate(){
  rateID = 0;
  amount = 0.0;
  status = "";
}
Rate::Rate(int rate_ID, double rate_amount, string rate_status){
  rateID = rate_ID;
  amount = rate_amount;
  status = rate_status;
}
int Rate::setRate(){

}
double Rate::updateRate(){

}
double Rate::deleteRate(){

}
Rate::~Rate(){

}
```

Payment.h

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

class Payment
{
   private:
     int Payment_ID;
     double Charge;
     double Pay;

   public:
     Payment();
     Payment(int payID, double payCharge, double paying);
     void calcPay();
     void displayPay();
     ~Payment();
};
```

Payment.cpp

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

Payment::Payment(){
  Payment_ID = 0;
  Charge = 0;
  Pay = 0;
}
Payment::Payment(int payID, double payCharge, double paying){
  Payment_ID = payID;
  Charge = payCharge;
  Pay = paying;
}
void Payment::calcPay(){

}
void Payment::displayPay(){

}
Payment::~Payment(){

}
```

Report.h

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

class Report{
  private:
    int reportID;
    string description;

  public:
    Report();
    Report(int repID, string repDesc);
    void displayPaymentDetail();
    void displayBookingVehicleDetail();
    ~Report();
};
```

Report.cpp

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

Report::Report(){
  reportID = 0;
  description = "";
}

Report::Report(int repID, string repDesc){
  reportID = repID;
  description = repDesc;
}

void Report::displayPaymentDetail(){

}
void Report::displayBookingVehicleDetail(){

}

Report::~Report(){

}
```

Feedback.h

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

class Feedback
{
   private:
     int Feedback_ID;
     string message;

   public:
     Feedback();
     Feedback(int Fb_Id, string mess);
     void sendFeedback();
     void viewFeedback();
     void replyFeedback();
     ~Feedback();
};
```

Feedback.cpp

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

Feedback::Feedback(){
   Feedback_ID = 0;
   message = "";
}
Feedback::Feedback(int Fb_Id, string mess){
  Feedback_ID = Fb_Id;
  message = mess;
}
void Feedback::sendFeedback(){

}
void Feedback::viewFeedback(){

}
void Feedback::replyFeedback(){

}
Feedback::~Feedback(){

}
```

main.cpp

```
#include "Category.h"
#include "Customer.h"
#include "Feedback.h"
#include "Insurance.h"
#include "Offer.h"
#include "Order.h"
#include "Payment.h"
#include "Rate.h"
#include "Report.h"
#include "Vehicle.h"

#include <iostream>
using namespace std;

int main(){
  Category* cat = new Category();
  Customer* cust = new Customer();
  Feedback* fdBack = new Feedback();
  Insurance* insure = new Insurance();
  Offer* off = new Offer();
  Order* booking = new Order();
  Payment* pay = new Payment();
  Rate* rates = new Rate();
  Report* rep = new Report();
  Vehicle* vehi = new Vehicle();

  cat->addNewVehicle();
  cat->deleteVehicle();

  cust->setDetails();
  cust->displayDetails();

  fdBack->sendFeedback();
  fdBack->replyFeedback();
  fdBack->viewFeedback();

  insure->displayInsurance();

  off->calcOffer();
  off->displayOffer();

  booking->displayOrder();

  pay->calcPay();
  pay->displayPay();
```

```
    rates->setRate();
    rates->updateRate();
    rates->deleteRate();

    rep->displayPaymentDetail();
    rep->displayBookingVehicleDetail();

    vehi->setDetails();
    vehi->checkAvailability();
    vehi->displayPrice();


    delete cat;
    delete cust;
    delete fdBack;
    delete insure;
    delete off;
    delete booking;
    delete pay;
    delete rates;
    delete rep;
    delete vehi;

    return 0;
}
```