Topic : **Car Rental System**

Group no: MLB_WD_CSNE_01.01_12

Campus : **Malabe**

Submission Date: 13/06/2023

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**
IT22253330	K.A.W.R.B. Iddamalgoda	0773959831
IT22339188	R.M.D.E.Rajapaksha	0774829636
IT22911858	S.A.H.D.M.Perera	0771581628
IT22312594	D.K.D.G.Perera	0725126980
IT22073082	E.M.B.Vihanga Bandara	0702874299

Contents

Page

1. Description of the requirements.

Car Rental System

Users should register to the system by providing their names, and contact information. The system has customers' names, contact information, and reservation history. Users can search
available cars based on location, date, and time. They can reserve a car for a specific duration andthey can cancel or modify their reservation at any time.

Also, every car should register in the system. The car has details like car ID, car type, availabilitystatus, and rental history. Users can reserve a car according to their requirements.

Every user should make a reservation including reservation ID, user ID, car ID, start date, enddate, and status.

All the users are provided with a rental invoice including invoice ID, user ID, car ID, rental duration, and charges. The invoice should generate rental invoices for users based on the rentalduration and car type.

Each and every user should make a payment based on their reservation of a car. It is the rental oftheir reservation. The system handles the payment processing for rental charges. It includes
information such as payment ID, invoice ID, payment amount, and payment status. The systemautomatically generates the payment.

The Administrator has the ability to handle the car rental system. He can access the system by using his username and password. The system should provide administrative functionality to manage the car fleet, including adding, removing, and updating car details. Also, it maintains arecord of rental history for reporting purposes.

2. Class Identified

1. User (inherited from the CarRentalSystem)
2. Car (inherited from the CarRentalSystem)
3. Reservation (inherited from the CarRentalSystem)
4. Invoice
5. Payment
6. Cash (inherited from the payment)
7. Card (inherited from the payment)
8. Admin (inherited from the CarRentalSystem)
9. CarRentalSystem

3. CRC Cards

Class: User	
Responsibilities	Collaborations
User details	
Search for availablecars	Car
Make a reservation	
Cancel or modify the reservation	Reservation

Class: Car	
Responsibilities	Collaborations
Car details	
Check availability andUpdate availability status	Reservation
Get rental history	
Reserve car	User

Class: Reservation	
Responsibilities	Collaborations
Create reservation	User
Update reservation details	
Get reservation details	Car

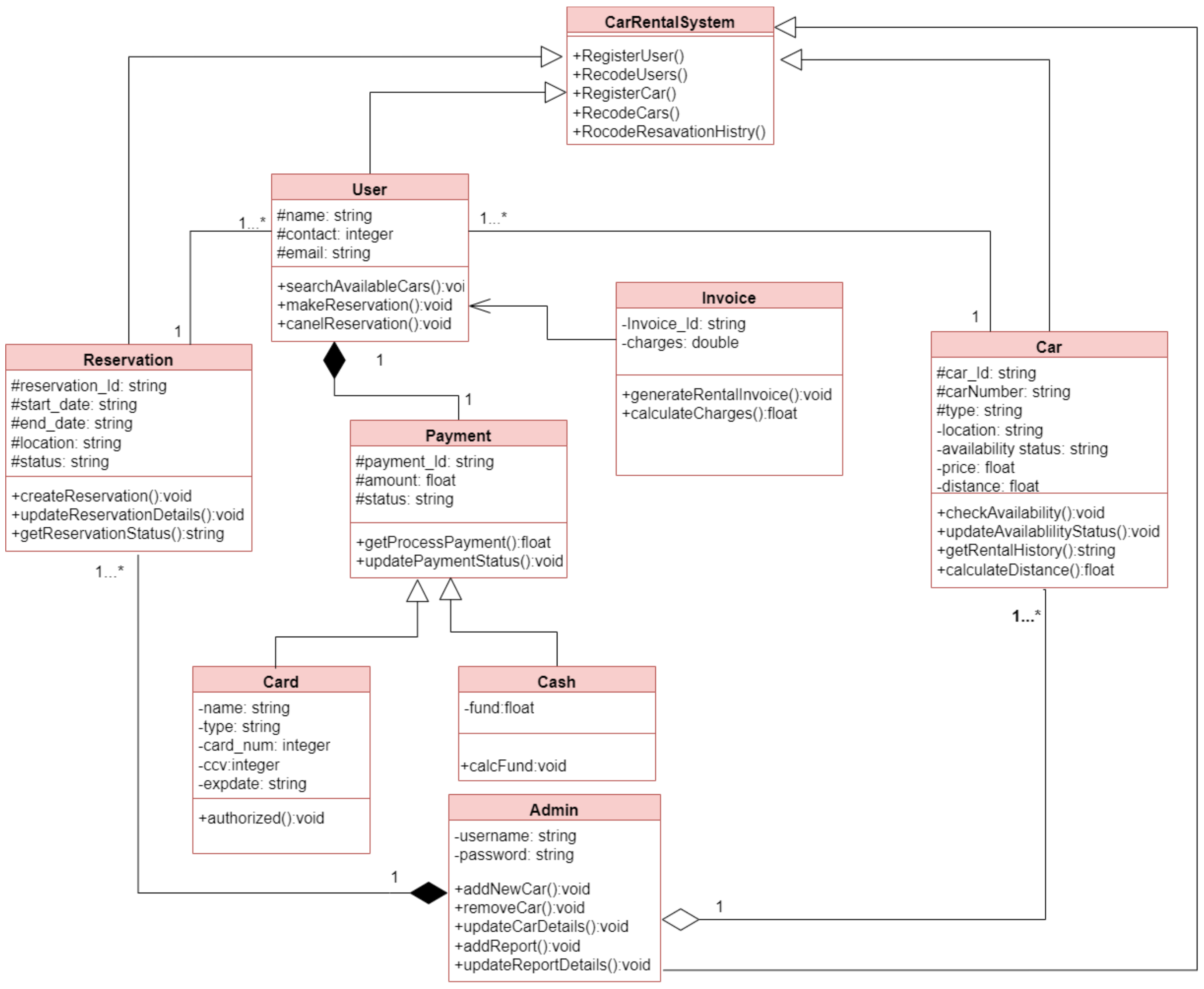
Class: Invoice	
Responsibilities	Collaborations
Generate rental invoice	User
Calculate charges	Car

Class: Payment	
Responsibilities	Collaborations
Process payment	User
payment status	User

Class: Admin	
Responsibilities	Collaborations
Handle the car rentalsystem	
Mange car fleet	Car
Maintain recode ofrental history	Reservation

Class: CarRentalSystem	
Responsibilities	Collaborations
Register user	User
Register car	Car
Recode Reservation history	Reservation

4. Class Diagram



5. Coding for the Classes

main.cpp

```
#include<iostream>
#include<string>
#include"User.h"
#include"Car.h"
#include"Reservation.h"
#include"Invoice.h"
#include"Payment.h"
#include"Cash.h"
#include"Card.h"
#include"Admin.h"
#include"CarRentalSystem.h"

using namespace std;

int main()
{
	//Data insert to user
	User* usr = new User("Rashminda", "0773959831", "warnarashminda@gmail.com");
	cout << "Details of User" << endl;
	usr->DisplayRegisterUser();//display registered user
	cout << endl;

	//Data insert to Car
	Car* Cr = new Car("1", "AAA-1020", "Petrol", "Rathnapura", "Available", 100000, 500000);
	cout << "Details of Car" << endl;
	Cr->DisplayRegisterCar();//display registered car
	cout << endl;

	//Data insert to reservation
	Reservation* Rv = new Reservation("1", "2023/6/12", "2023/6/13", "Rathnapura", "One-Day");
	cout << "Details of Reservation" << endl;
	Rv->DisplayCreateReservation();//display create reservation
	cout << endl;

	//Data insert to Invoice
	Invoice* Iv = new Invoice("1", 5000);
	Iv->DisplayCalculateCharges();//display calculate chargers
	cout << endl;

	//Data insert to payment
	Payment* Pmt = new Payment("1", 5000, "Card");
	Pmt->DisplayPayment();//display Payment
	cout << endl;

	//Data insert to card
	Card* Crd = new Card("K.A.W.R.B.Iddamalgoda", "Debit Card", "1642001089718451", 115,
"05/24");


//Data insert to cash
	Cash* Csh = new Cash(5000);

	//Data insert to Admin
	Admin* Adm = new Admin("Admin", "Admin123");

	//CarRentalSystem handle the system and store recodes
	CarRentalSystem* CRS = new CarRentalSystem();


	return 0;


}
```

User.h

```
#pragma once
#include"CarRentalSystem.h"
#include"Payment.h"
#include"Reservation.h"
#include"Car.h"
#define SIZE 2

class User : public CarRentalSystem
{
private:
	string name;
	string contact;
	string  email;
	Payment* Pm[SIZE];
	Reservation* Rv[1];
	Car* Cr[1];

public:
	User();
	User(string Name, string Contact, string Email);
	void SearchAvailableCars();
	void makeReservation();
	void cancelReservation();
	~User();
};
```

User.cpp

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

using namespace std;

User::User()
{
	name = "";
	contact = "0";
	email = "";

}

User::User(string Name, string Contact, string Email)
{
	name = Name;
	contact = Contact;
	email = Email;
}

void User::SearchAvailableCars()
{
}


void User::makeReservation()
{
}

void User::cancelReservation()
{
}

User::~User()
{
	cout << "Delete User"<<endl;

}
```

Car.h

```
#pragma once
#include"CarRentalSystem.h"
#include"User.h"

class Car : public CarRentalSystem
{
protected:
	string carID;
	string carNumber;
	string type;
private:

	string location;
	string availabilityStatus;
	float price;
	float distance;
	User* Usr;
public:
	Car();
	Car(string cID, string cNum, string cType, string cLoaction, string aStatus, float cPrice, float
cdistance); // Constructor
	void checkAvailability();
	void updateAvailabilityStatus();
	string getRentalHistory();
	float getCalculateDistance();
	~Car();

};
```

Car.cpp

```
#include<iostream>
#include<string>
#include "Car.h"

using namespace std;

Car::Car()
{
	carID = "";
	carNumber = "";
	type = "";
	location = "";
	availabilityStatus = "";
	price = 0;
	distance = 0;

}

Car::Car(string cID, string cNum, string cType, string cLocation, string aStatus, float cPrice, float
cDistance)
{
	carID = cID;
	carNumber = cNum;
	type = cType;
	location = cLocation;
	availabilityStatus = aStatus;
	price = cPrice;
	distance = cDistance;
}

void Car::checkAvailability()
{
}

void Car::updateAvailabilityStatus()
{
}

string Car::getRentalHistory()
{
	return string();
}

float Car::getCalculateDistance()
```

```
{
	return 0.0f;
}

Car::~Car()
{
	cout << "Car Deleted" << endl;
}
```

Reservation.h

```
#pragma once
#include"CarRentalSystem.h"
#include"User.h"

class Reservation : public CarRentalSystem
{
protected:
	string reservationID;
	string startDate;
	string endDate;
	string location;
	string status;
private:
	User* usr[];
public:
	Reservation();
	Reservation(string rID, string sDate, string eDate, string rlocation, string rStatus);
	void createReservation();
	void DisplayCreateReservation();
	void updateReservationDetails();
	string getReservationStatus();
	~Reservation();
};
```

Reservation.cpp

```
#include<iostream>
#include<string>
#include "Reservation.h"

using namespace std;

Reservation::Reservation()
{
	reservationID = "";
	startDate = "";
	endDate = "";
	location = "";
	status = "";
}

Reservation::Reservation(string rID, string sDate, string eDate, string rLocation, string rStatus)
{
	reservationID = rID;
	startDate = sDate;
	endDate = eDate;
	location = rLocation;
	status = rStatus;
}

void Reservation::createReservation()
{
}

void Reservation::DisplayCreateReservation()
{
}

void Reservation::updateReservationDetails()
{
}

string Reservation::getReservationStatus()
{
	return string();
}

Reservation::~Reservation()
{
	cout << "Reservation Delete" << endl;}
```

Invoice.h

```
#pragma once
#include"User.h"

class Invoice
{
private:
	string InvoiceID;
	double charges;
	User* Usr;

public:
	Invoice();
	Invoice(string iID, double charge);
	void generateRentalInvoice();
	float calculateCharges();
	void DisplayCalculateCharges();
	~Invoice();

};
```

Invoice.cpp

```
#include<iostream>
#include<string>
#include "Invoice.h"

using namespace std;

Invoice::Invoice()
{
	InvoiceID = "";
	charges = 0;

}

Invoice::Invoice(string iID, double charge)
{
	InvoiceID = iID;
	charges = charge;
}
void Invoice::generateRentalInvoice()
{


}
float Invoice::calculateCharges()
{

}
void Invoice::DisplayCalculateCharges()
{
}
Invoice::~Invoice()
{
	cout << "Delete nvoice"<<endl;
}
```

Payment.h

```
#pragma once

class Payment
{
protected:
	string paymentId;
	float amount;
	string status;
public:
	Payment();
	Payment(string pID, float pAmount, string pStatus);
	void DisplayPayment();
	float getProcessPayment();
	void updatePaymentStatus();
	~Payment();
};
```

Payment.cpp

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

using namespace std;

Payment::Payment()
{
	paymentId = "";
	amount = 0;
	status = "";

}
Payment::Payment(string pID, float pAmount, string pStatus)
{
	paymentId = pID;
	amount = pAmount;
	status = pStatus;
}
void Payment::DisplayPayment()
{
}
float Payment::getProcessPayment()
{
	return 0.0f;
}
void Payment::updatePaymentStatus()
{
}
Payment::~Payment()
{
	cout << "Payment Delete" << endl;
}
```

Cash.h

```
#pragma once
#include"Payment.h"

class Cash : public Payment
{
private:
	float fund;
public:
	Cash();
	Cash(float pCash);
	double calcFund();
	~Cash();

};
```

Cash.cpp

```
#include<iostream>
#include<string>
#include "Cash.h"

using namespace std;

Cash::Cash()
{
	fund = 0;
}

Cash::Cash(float pCash)
{
	fund = pCash;
}

double Cash::calcFund()
{
	return 0.0;
}

Cash::~Cash()
{
	cout << "Cash deleted" << endl;
}
```

Card.h

```
#pragma once
#include"Payment.h"
class Card : public Payment
{
private:
	string name;
	string type;
	int CardNumber;
	int CCV;
	string expDate;
public:
	Card();
	Card(string cName, string cType, string cNumber, int cCCV, string cExpDate);
	void authorized();
	~Card();
};
```

Card.cpp

```
#include<iostream>
#include<string>
#include "Card.h"

using namespace std;

Card::Card()
{
	name = "";
	type = "";
	CardNumber = 0;
	CCV = 0;
	expDate = "";
}
Card::Card(string cName, string cType, string cNumber, int cCCV, string cExpDate)
{
	name = cName;
	type = cType;
	CardNumber = cNumber;
	CCV = cCCV;
	expDate = cExpDate;
}
void Card::authorized()
{}
Card::~Card()
{	cout << "Card Deleted" << endl; }
```

Admin.h

```
#pragma once
#include"CarRentalSystem.h"
#include"Reservation.h"
#include"Car.h"
#define SIZE 2

class Admin : public CarRentalSystem
{
private:
	string UserName;
	string Password;
	Reservation* Rv[SIZE];
	Car* Cr[SIZE];
public:
	Admin();
	Admin(string pusrnme, string pwd);
	bool login(string& user, string& pw);
	void addNewCar();
	void removeCar();
	void updateCarDetails();
	void addReport();
	void updateReportDetails();
	~Admin();
};
```

Admin.cpp

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

using namespace std;

Admin::Admin()
{
	string UserName = "";
	string Password = "";
}

Admin::Admin(string pusrnme, string pwd) // Constructor
{
	UserName = pusrnme;
	Password = pwd;
}

bool Admin::login(string& user, string& pw) // login
{
	return (UserName == user && Password == pw);
}

void Admin::addNewCar()
{
}

void Admin::removeCar()
{
}

void Admin::updateCarDetails()
{
}
void Admin::addReport()
{
}
void Admin::updateReportDetails()
{
}
Admin::~Admin()
{
	cout << "Delete Admin" << endl;}
```

CarRentalSystem.h

```
#pragma once

class CarRentalSystem
{
private:


public:

	CarRentalSystem();
	void RegisterUser();
	void DisplayRegisterUser();
	void RecodeUsers();
	void RegisterCar();
	void DisplayRegisterCar();
	void RecodeCar();
	void RecodeReservationHistry();
	~CarRentalSystem();

};
```

}

CarRentalSystem.cpp

```
#include<iostream>
#include<string>
#include "CarRentalSystem.h"

using namespace std;

CarRentalSystem::CarRentalSystem()
{

}

void CarRentalSystem::RegisterUser()
{
}

void CarRentalSystem::DisplayRegisterUser()
{
}

void CarRentalSystem::RecodeUsers()
{
}

void CarRentalSystem::RegisterCar()
{
}

void CarRentalSystem::DisplayRegisterCar()
{
}

void CarRentalSystem::RecodeCar()
{
}
void CarRentalSystem::RecodeReservationHistry()
{
}
CarRentalSystem::~CarRentalSystem()
{
	cout << "Delete CarRentalSystem" << endl;
}
```

6. Contribution

ID Number	Name	Contribution
IT22253330	K.A.W.R.B. Iddamalgoda	Car, Admin, CarRentalSystem
IT22339188	R.M.D.E.Rajapaksha	Payment, Cash, Card
IT22911858	S.A.H.D.M.Perera	Reservation
IT22312594	D.K.D.G.Perera	User
IT22073082	E.M.B.Vihanga Bandara	Invoice