Topic : Hotel Reservation System for tourists

Group no : MLB_WE_01.01_05

Campus : Malabe

Submission Date : 19/05/2022

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**
IT21014772	Chathuranga K.K.K.V	0711871796
IT21201196	Fernando P.W.D.A.M	0774299887
IT21030680	De Costa M. T. N.	0775367650
IT21210242	Bandaranayake C. N.	0775852054
IT21232718	Karunathilaka K. P. C. A	0763294781

Functional requirements

Users

- Users should be able to register to the system to reserve a hotel room, a table for dine in or a tour guide.
- Users should be able to log in to the system using their valid username and password. If the user forgets the password, he/she can reset it by confirming the identity by using the e-mail address.
- When a new user register to the system he/she should be able to enter first name, last name, country, address, date of birth, contact details, username, and password.
- Users should be able to change or update their profile details by login to their accounts.
- Unregistered users should be able to navigate through the site, but they will not be able to make any reservation until they register.
- Registered users should be able to confirm their reservation by paying a certain amount in advance using their preferred payment method.
- Users should be able to enter their check in and check out dates and find available hotels, select dine in options and select a tour guide before reserving.
- Users should be able to cancel their reservation as required and request a refund, but refund will only be done if the period of cancellation has not passed.
- Users should be able to write reviews about the site, rate it and access contact us page for any further inquiries.

System Manager

- System Manager should be able to login to system using a valid username and password.
- System Manager should be able to update and add new hotels to the system after validating them.
- System Manager should be able to generate reports.

System

- System should be able to store information about users.

- System should be able to store information about the hotels and tour guides.
- System should be able to validate the user when login to the system.
- System should be able to validate the user when a certain payment is done.
- System should be able to send confirmation email when registering to the system and when making a reservation.

Non-Functional requirements

Performance requirement

- Website must be available for anyone in any country.
- Website must be accessible from any device.
- Website must be available 24/7.
- All users must be able to access the system at the same time.
- Website loading speed should be 3 seconds or minimum.
- System should be able to access data from the database quickly and display them.
- System should be able to fix unexpected errors quickly.
- Login information should be verified by the system within 5 seconds.

Security requirements

- System admin's and system manager's access must be protected by the system.
- Payment details must be transmitted in encrypted form.
- Unauthorized users cannot access other user accounts, edit the details of other users' accounts, or make any reservation.
- Register confirmation e-mail must be sent to the new user when they register to the system.
- Refund will be done only for the validated user accounts.
- When registering users must provide a strong password.
- System should protect the website from viruses.
- System should backup data in order to prevent data loss.

Maintenance requirement

- System should be able to run security checks at least once a week.
- System should be able to analyze most popular and least popular hotels according to user reviews and ratings and display them accordingly.

- System should be able to check loading speed and fix the issues immediately.

Noun verb analysis

Nouns are highlighted in yellow colour; verbs are highlighted in blue colour.

'Fast Booking' is a website for 'Hotel Management System for Tourists'. Users can login to the website and do all the works like reserving a hotel, reserve a table to dine in, reserving a tour guide. Users can be registered or unregistered. All the users can view the website and get information they need about hotels, dine in options and details about tour guides. To reserve a hotel for vacation, reserve a table for dine in and book a tour guide, user must register to the system providing necessary information.

If the user wants to register to the register to the website, he/she should provide first name, last name, country, address, date of birth, contact details like e-mail and mobile number, an username and a password as required. When a user register as a member the system automatically generate an e-mail and sends it to the user to confirm registration with all details provided by the user. The registered member can access the website by providing the username and password.

Once registered member logs into the system he/she can reserve a hotel room, a table to dine in, a tour guide and confirm reservation by paying a certain amount in advance. When reserving a hotel room, the registered user has to enter check in and check out. Then system checks the available hotels and display them. If the user wants to reserve a table for dine in, he/she can choose dine in time (breakfast, lunch, dinner) and dine in options (vegetarian, non-vegetarian). The user can also reserve a tour guide if they need guidance while visiting different places.

If the registered user wants to cancel a reservation, they must login to the system. Then they must cancel their reservation and request a refund. If the time period for cancellation has passed, the refund will not be done by the hotel. The time period for cancellation can be different from hotel to hotel.

To confirm a reservation registered user should pay a certain amount in advance. Payment can be done via debit/credit cards or fund transfer. Once the payment is done, a confirmation

of reservation e-mail will be sent to the user. This e-mail includes the reserved hotel name, reservation number, amount paid and time period to cancel the reservation.

System Manager and System Admin can login to the system by providing their username and password. Manager can update hotels/restaurant, add new hotels/restaurant. System Admin can generate cash flow statements, customer review reports, income reports. and change user details when they request to.

Outside the scope	Meta language	Event/operation	Attribute
System	Tourists	Login to the website	First name
Website	Member	Reserve a hotel	Last name
Vacation	Registered member	Reserve a table for dine in	Country
Places	Breakfast	Book a tour guide	Address
	Lunch	Register to the system providing necessary information	Date of birth
	Dinner	Generate an e-mail	e-mail
	Vegetarian	Send e-mail to user	Mobile number
	Non-vegetarian	Confirm registration	Username
	User details	Access the website	password
		Paying a certain amount	Amount in advance
		Enter check in and check out	Check in
		Display available hotels	Check out
		Choose to dine in time	Available hotels
		Cancel a reservation	Dine in time
		Request refund	Dine in option
		Confirm a reservation	Time period for cancellation
		Update hotels/restaurant	Debit cards
		Add new hotels/restaurant	Credit cards
		Generate cash flow statements, customer review reports and income reports	Fund transfers
		Change user details	Hotel name
		Request to change user details	Reservation number
			Amount paid

Classes

- User
- Hotel
- Rooms
- Table to dine in
- Tour guides
- Registered
- Unregistered
- Refund
- Payment
- Reservation
- System Manager
- System Admin
- Report

CRC Cards

Class name: Registered User	
Responsibilities:	**Collaborations:**
Reserve a hotel	Hotel
Reserve a table	Tables
Reserve a tour guide	Tour guides
Manage user profile	

Class name: Unregistered User	
Responsibilities:	**Collaborations:**
Browse website	
Registering to the website	

System Manager	
Responsibilities	**Collaboration**
Generate cash flow statements	report
Generate customer review reports	
Generate income reports	report

System Admin	
Responsibilities	**Collaboration**
Add new hotels	
Update hotel details	hotel
Delete hotels	hotel
Change user details	Registered user

Table	
Responsibilities	**Collaboration**
change status of the table whether reserved or not	reservation
Accommodate tables according to reservations	reservation
Display table status	
Display menu	
Display number of seats	

Report	
Responsibilities	**Collaboration**
Store cash in and cash out	Reservation

Hotel	
Responsibility	Collaborators
Provide available room details Provide available room details	Room Table

Tour-Guides	
Responsibility	Collaborators
Provide details on availability Update tour guide information	

Room	
Responsibility	Collaborators
Accommodate tourists Update availability status	Users

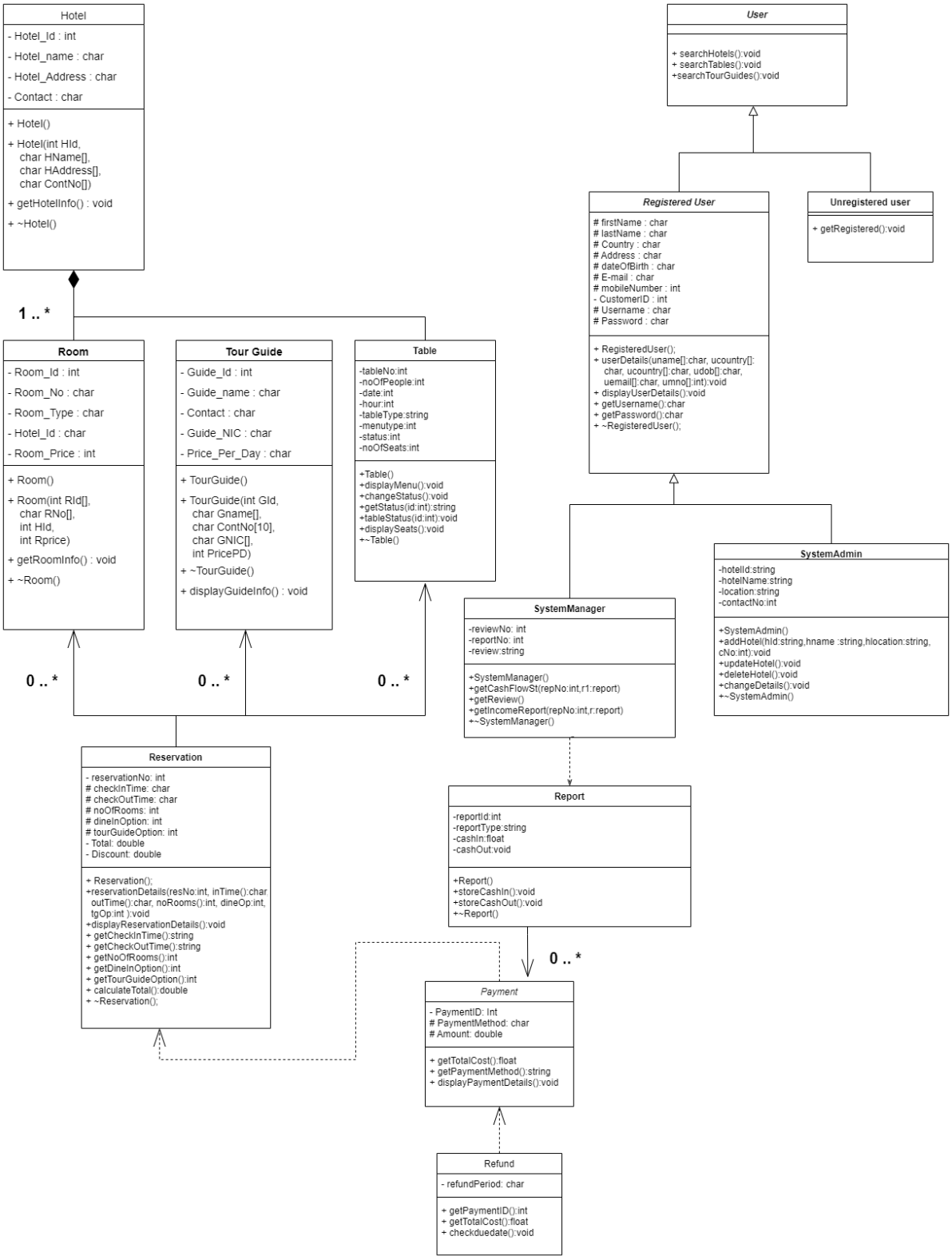
Class name = User	
Responsibilities	**Collaborations**
Register to system	
Login to system	
Reserve hotel	Hotel
Reserve table for dine in	Table to dine in
Book tour guides	Tour guides
Cancel reservation	
Enter check in, check out times	Hotel
Choose to dine in time	Table to dine in
Request refund	Refund
Request to change user details	System admin

Class name = Reservation	
Responsibilities	**Collaborations**
Get user information	User
Get hotel details	Hotel
Get dine in option details	Table to dine in
Get tour guide details	Tour guides
Calculate total cost of reservation	
Calculate discount received	

Class name=Payment	
responsibilities	**collaborations**
Validate payment details	
Get payment details	Customer
Store payment details	

Class name=Refund	
responsibilities	**collaborations**
Get refund details	Customer
Store refund details	

Class diagrams



User.h

```
#include "Rooms.h"
#include "Tables.h"
#include "Tourguide.h"

class user
{
private:
	Rooms* rooms;
	Tables* tables;
	TourGuide* tourguide;

public:
	void searchHotels();
	void searchTables();
	void searchTourGuides();
};
```

User.cpp

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

void user::searchHotels()
{
	int id;
	cout << "Enter room ID: ";
	cin >> id;

	rooms.GetRoomInfo(id);
}

void user::searchTables()
{
	int id;
	cout << "Enter table ID: ";
	cin >> id;

	tables.tableStatus(id);
}

void user::searchTourGuides()
{
	int id;
	cout << "Enter Tour Guide ID: ";
	cin >> id;

	tourguide.displayGuideInfo(id);
}
```

Reservation.h

```
#pragma once
#include "Rooms.h"
class Reservation
{
protected:
	int resNo;
	char checkin[6];
```

```
	char checkout[6];
	int noOfRooms;
	int dineOption;
	int tourGuideOption;

	double price[noOfRooms];

private:
	double total;
	double discount;
	Rooms* rooms;

public:
	Reservation();
	void resdetails(int rno, char cin[], char cout[], int nrooms, int dop, int
tgop);
	void displayDetails();
	char getcheckIn();
	char getcheckOut();
	int getnoOfRooms();
	int getdineOption();
	int gettourGuideOption();
	double calculateTotal();
	~Reservation();
};
```

Reservation.cpp

```
#include "Reservation.h"
#include <cstring>

Reservation::Reservation()
{
	resNo = 0;
	strcpy_s(checkin, "");
	strcpy_s(checkout, "");
	noOfRooms = 0;
	dineOption = 0;
	tourGuideOption = 0;
}

void Reservation::resdetails(int rno, char cin[], char cout[], int nrooms, int
dop, int tgop)
{
	resNo = rno;
	strcpy_s(checkin, cin);
	strcpy_s(checkout, cout);
	noOfRooms = nrooms;
	dineOption = dop;
	tourGuideOption = tgop;
	rooms = new Rooms[nrooms];
	noOfRooms = nrooms;
}

void Reservation::displayDetails()
{
	int i = 0;

	for (i = 0; i < noOfRooms; i++)
	{
		price[i] = Rooms[i].setPrice(price);
```

```
	}

	cout << "Reservation: " << endl << "Reservation Number: " << resNo <<
endl;
	if (noOfRooms != 0) cout << "No of Rooms: " << noOfRooms << endl << "Check
in time: " << checkin << endl << "Check out time: " << checkout << endl;
	else if (dineOption != 0) cout << "Dine in Option: " << dineOption <<
endl;
	else if (tourGuideOption != 0) cout << "Tour Guide Options: " <<
tourGuideOption << endl;
}

char Reservation::getcheckIn()
{
	return checkIn[6];
}

char Reservation::getcheckOut()
{
	return checkOut[6];
}

int Reservation::getnoOfRooms()
{
	return noOfRooms;
}

int Reservation::getdineOption()
{
	return dineOption;
}

int Reservation::gettourGuideOption()
{
	return tourGuideOption;
}

double Reservation::calculateTotal()
{
	double total = 0;
	int i;

	for (i = 0; i < noOfRooms; i++)
	{
		total += price[i];
	}
	return total;
}

Reservation::~Reservation()
{
	cout << "Objected deleted";
}
```

RegisteredUser

```
class RegisteredUser
{
	protected:
		char firstName[20];
		char lastName[20];
```

```
		char country[10];
		char address[50];
		char dateOfBirth[10];
		char email[20];
		int mobilenumber;
		char username[10];
		char password[20];

	private:
		int customerID;

	public:
		Registereduser();
		void userDetails(char fname[], char lname[], char cntry[], char
addr[], char dob[], char mail[], int mno);
		void displayUserDetails();
		~Registereduser();
};
```

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

RegisteredUser::RegisteredUser(){
	strcpy_s(firstName, "");
	strcpy_s(lastName, "");
	strcpy_s(country, "");
	strcpy_s(address, "");
	strcpy_s(dateOfBirth, "");
	strcpy_s(email, "");
	mobilenumber = 0;
	strcpy_s(username, "");
	strcpy_s(password, "");
	customerID = 0;
}

void RegisteredUser::userDetails(char fname[], char lname[], char cntry[], char
addr[], char dob[], char mail[], int mno){
	strcpy_s(firstName, fname);
	strcpy_s(lastName, lname);
	strcpy_s(country, cntry);
	strcpy_s(address, addr);
	strcpy_s(dateOfBirth, dob);
	strcpy_s(email, mail);
	mobilenumber = mno;
}

void RegisteredUser::displayUserDetails(){
	cout << "User Details: " << endl << "First Name: " << firstName << endl <<
"Last Name: " << lastName << endl;
	cout << "Country: " << country << endl << "Address: " << address << endl
<< "Date of Birth: " << dateOfBirth << endl;
	cout << "E-mail: " << email << endl << "Mobile Number: " << mobilenumber
<< endl;
}

char RegisteredUser::getUsername(){
	return username[9];
}

char RegisteredUser::getPassword(){
	return password[19];
```

```
}

RegisteredUser::~RegisteredUser()
{
	cout << "object deleted";
}
```

UnregisteredUser

```
#include "RegisteredUser.h"
class UnregisteredUser
{
	private:
		RegisteredUser* user;

	public:
		void getRegistered();
};
```

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

void UnregisteredUser::getRegistered()
{
	user = new RegisteredUser[];
}
```

Hotel.h

```
#pragma once
class Hotel
{
private:
	int Hotel_Id;
	char Hotel_name[30];
	char Hotel_Address[80];
	char Contact[10];
	Rooms* room[5];

public:
	Hotel()
	{
		room[0] = new Rooms();
		room[1] = new Rooms();
	};
	Hotel(int HId, char HName[], char HAddress[], char ContNo[]);
	void getHotelInfo();
	~Hotel();
};
```

Hotel.cpp

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

```
Hotel::Hotel()
{
	Hotel_Id = 0;
	strcpy(Hotel_name , "Empty");
	strcpy(Hotel_Address, "Empty");
	strcpy(Contact, "Empty");
}
Hotel::Hotel(int HId, char HName[], char HAddress[],char ContNo[])
{
	Hotel_Id = HId;
	strcpy(Hotel_name, HName);
	strcpy(Hotel_Address, HAddress);
	strcpy(Contact, ContNo);

}
void Hotel::getHotelInfo()
{
}

Hotel::~Hotel()
{
	cout << "Destructor called : Hotel";
}
```

TourGuide.h

```
#pragma once
class TourGuide
{
private:
	int Guide_Id;
	char Guide_name[30];
	char Contact[10];
	char Guide_NIC[15];
	int Price_Per_Day;
public:
	TourGuide();
	TourGuide(int GId, char Gname[], char ContNo[10], char GNIC[], int
PricePD);
	~TourGuide();
	void displayGuideInfo();
};
```

TourGuide.cpp

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

TourGuide::TourGuide()
{
	Guide_Id = 0;
	strcpy(Guide_name, "Empty");
	strcpy(Contact, "Empty");
	strcpy(Guide_NIC, "Empty");
	Price_Per_Day = 0;
}
```

```
TourGuide::TourGuide(int GId, char Gname[], char ContNo[10], char GNIC[], int
PricePD)
{
	Guide_Id = GId;
	strcpy(Guide_name, Gname);
	strcpy(Contact, ContNo);
	strcpy(Guide_NIC, GNIC);
	Price_Per_Day = 0;
}

TourGuide::~TourGuide()
{
	cout << "Destructor called : Tuor Guide";
}

void TourGuide::displayGuideInfo()
{

}
```

Room.h

```
#pragma once
class Rooms
{
private:
	int Room_Id;
	char Room_No[6];
	char Room_Type[20];
	int Hotel_Id;
	int Room_Price;

public:
	Rooms();
	Rooms(int RId, char RNo[],char RType[], int HId, int Rprice);
	~Rooms();
	void GetRoomInfo();
};
```

Rooms.cpp

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

Rooms::Rooms()
{
	Room_Id = 0;
	strcpy(Room_No,"Empty");
	strcpy(Room_Type, "Empty");
	Hotel_Id = 0;
	Room_Price = 0;
}

Rooms::Rooms(int RId, char RNo[], char RType[], int HId, int Rprice)
{
	Room_Id = RId;
	strcpy(Room_No, RNo);
```

```
	strcpy(Room_Type, RType);
	Hotel_Id = HId;
	Room_Price = Rprice;
}

Rooms::~Rooms()
{
	cout << "Destructor called";
}

void Rooms::GetRoomInfo()
{
}
```

Payment.h

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

class Payment {
private:
	int paymentId;
	string paymentMethod;
	double paymentAmount;

public:
	Payment();
	float getTotalCost();
	string getPaymentMethod();
	displayPaymentDetails();
	~Payment();
};
```

Refund.h

```
class Refund {
private:
	string refundPeriod;

public:
	Refund();
	int getPaymentID();
	float getTotalCost();
	checkDueDate();
	~Refund();
};
```

Systemadmin

```
//code system admin
#include<iostream>
#include<cstring>
#define SIZE
using namespace std;

//class
class systemAdmin :public registeredUser {
private:
	string hotelId;
	string hotelName;
	string location;
	int contactNo;

public:
	systemAdmin();
	void addHotel(string hId, string hName, string hLocation, int cNo);
	void updateHotel();
	void deleteHotel();
	void changedetails();
	~systemAdmin();
};
```

SystemManager

```
//code for system manager

// classes
class systemManager :public registeredUser {
private:
	int reviewNo;
	int reportNo;
	string review;
public:
	systemManager();
	void getCashFlowSt(int repNo, report* r1);
	void enterReview();
	void getIncomeReport(int repNo, report* r);
	~systemManager();
};
```

Table

```
//code for table class
class table {
private:
	int tableNo;
	int noOfPeople;
	int date;
	string time;
	string tableType;
	int menuNo;
	int status;
	int noOfSeats;

public:
	table();
```

```
        void diplayMenu();
        void tableStatus(int id);
        string getStatus(int id);
        void changeStatus();
        void displaySeats();
        ~table();

};
```

Report

```
//code for report class
class report {
private:
        payment* p1
                int reportID;
        string reportType;
        float cashIn;
        void cashOut;

public:
        report(payment* p);
        void storeCashIn();
        void storeCashOut();
        float CashOut();
        ~report();
};
```

Main

```
#include <iostream>
#include <cstring>
#include "user.h"
#include "Reservation.h"
#include "Hotel.h";
#include "Rooms.h";
#include "TourGuide.h";
#include "RegisteredUser.h"
#include "payment.h"
#include "report.h"
#include "table.h"
#include "systemManager.h"
#include "systemAdmin.h"
#define SIZE
using namespace std;

int main()
{
    RegisteredUser* user1;

    user1 = new RegisteredUser();

    delete user1;

    systemAdmin ad1;
    systemManager man1;
    Hotel* hotel1;
    hotel1 = new hotel();
    payment* p;
```

```
    p = new payment();
    report* r1;
    r1 = new report(p);


    delete hotel1;
    delete p;
    delete r1;

    Hotel* Hotel1;
    Hotel1 = new Hotel();

    TourGuide* Guide1;
    Guide1 = new TourGuide();

    delete Hotel1;
    delete Guide1;

    Reservation* Res1;

    Res1 = new Reservation();
    delete Res1;

    Payment* pay;
    Refund* fund;

    pay = new Payment();
    fund = new Refund();

    delete pay;
    delete fund;

    return 0;
}
```

Individual Contribution

Student ID	Contribution
IT21014772	CRC Cards, Class diagram and Code on RegisteredUser and UnregisteredUser
IT21201196	CRC Cards, Class diagram and Code on systemAdmin, systemManager, Table, Report
IT21030680	CRC Cards, Class diagram and Code on Hotel, Rooms, Tourguide
IT21210242	CRC Cards, Class diagram and Code on User, Reservation
IT21232718	CRC Cards, Class diagram and Code on Payment, Refund

The functional requirements, noun verb analysis and relationship diagram were completed by the contribution of all group members.