Topic : Wild-Life Safari Trip Management System.

Group no :MLB_06_02_06

Campus : Malabe

Submission Date : 20/05/2022

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**
IT21328534	D. Avishka Thenuwan	0772198385
IT21327094	Somarathne D. K.	0703534403
IT21333552	Abeysinghe I. U.	0704009080
IT21327230	Kaveesha Sankalpa Godage	0762346136
IT21191824	Dias D. T. M.	0762929330

Exercise 01:

System Requirements

1. A registered user/ guest user can visit online safari management system.
2. Guest user can overview the system, to use the system, they must register it with the system by entering details such as name, address, NIC, email, contact number.
3. The registered user can login to the system by entering the correct username and password.
4. The registered user/ guest user can search packages on the website.
5. Admin can add new/ remove packages, update the database, upload advertisements, manage bookings and FAQS, maintain the website.
6. A user can place a booking from website. A booking can consist of one or more packages.
7. System should generate a unique ID for the user after booking the tour.
8. Registered user must do a payment.
9. The user selects the payment method for each booking. (Credit, debit, visa, master)
10. User must enter their payment details such as CVC number, card holder name, and card number.
11. Once the customer finalizes the booking, payment is getting validated, and a booking ID is assigned to the particular booking.
12. After the payment is conform by bank or other trusted resources a report of the booking details for users and payment details are send by an email.
13. Financial records are updated by admin.
14. User is able to give feedbacks to the service received.
15. Manager check reports, manage tours and promotions.
16. Driver can check tours, manage routes, search directions and view customer statues.
17. System administrator has ability to add or remove user and staff members from the system.
18. System should function 24/7/365.

Noun & Verb Analysis

(Nouns are in RED colour and Verbs are in BLUE colour)

1. A registered user/ guest user can visit online safari management system.
2. Guest user can overview the system, to use the system, they must register it with the system by entering details such as name, address, NIC, email, contact number.
3. The registered user can login to the system by entering the correct username and password.
4. The registered user/ guest user can search packages on the website.
5. Admin can add new/ remove packages, update the database, upload advertisements, manage bookings and FAQS, maintain the website.
6. A user can place a booking from website. A booking can consist of one or more packages.
7. System should generate a unique ID for the user after booking the tour.
8. Registered user must do a payment.
9. The user selects the payment method for each booking. (Credit, debit, visa, master)
10. User must enter their payment details such as CVC number, card holder name, and card number.
11. Once the user finalizes the booking, payment is getting validated, and a booking ID is assigned to the particular booking.
12. After the payment is conform by bank or other trusted resources a report of the booking details for users and payment details are send by an email.
13. Financial records are updated by admin.
14. User is able to give feedbacks to the service received.
15. Manager check reports, manage tours and promotions.
16. Driver can check tours, manage routes, search directions and view user statues.
17. System administrator has ability to add or remove user and staff members from the system.
18. System should function 24/7/365.

Identified Classes

- Guest User
- Registered User
- Packages
- Admin
- Payment
- Manager
- Driver
- Financial Records

Reasons for rejecting other nouns

- **Redundant** – Tour, user, system administrator
- **An event or an operation** – Booking, feedbacks, service
- **Outside Scope of system** – System, website, database, bank, trusted resources, Report
- **Meta Language** – Staff member
- **An attribute** – Details, Name, address, NIC, email, contact number, Username, password, advertisements, Credit, debit, visa, master, FAQs, ID, payment details, CVC number, card holder name, card number, booking ID, email, promotions, routes, directions, user status

Methods

Guest User

- Visit
- Enter details
- Register
- Login

Registered User

- Login to the system by entering details
- Search packages
- Place a booking
- Do a payment
- Select
- Finalize
- Give feedbacks

Packages

- Generate package ID
- Calculate package price

Admin

- Add packages
- Remove packages
- Add user and staff members
- Remove user and staff members
- Update
- Upload
- Manage
- Maintain

Payment

- Generate payment ID
- Check payment details
- Confirm payment
- Send email

Manger

- Check
- Manage

Driver

- Check
- Manage
- Search
- View

Financial Records

- Update the system
- Calculate transactions

Exercise 02:

CRC Cards

Guest User	
Responsibility	**Collaborators**
Register to the system	
View the packages	Package

Registered User	
Responsibility	**Collaborators**
Login to the system	
Search packages	Packages
Book packages	Packages
Add and update user details	

Packages	
Responsibility	**Collaborators**
Generate ID	
Check availability of packages	Admin
Calculate package price	Financial records

Admin	
Responsibility	**Collaborators**
Login to the system	
Add packages details	Packages
Delete packages details	Packages
Update packages details	Packages

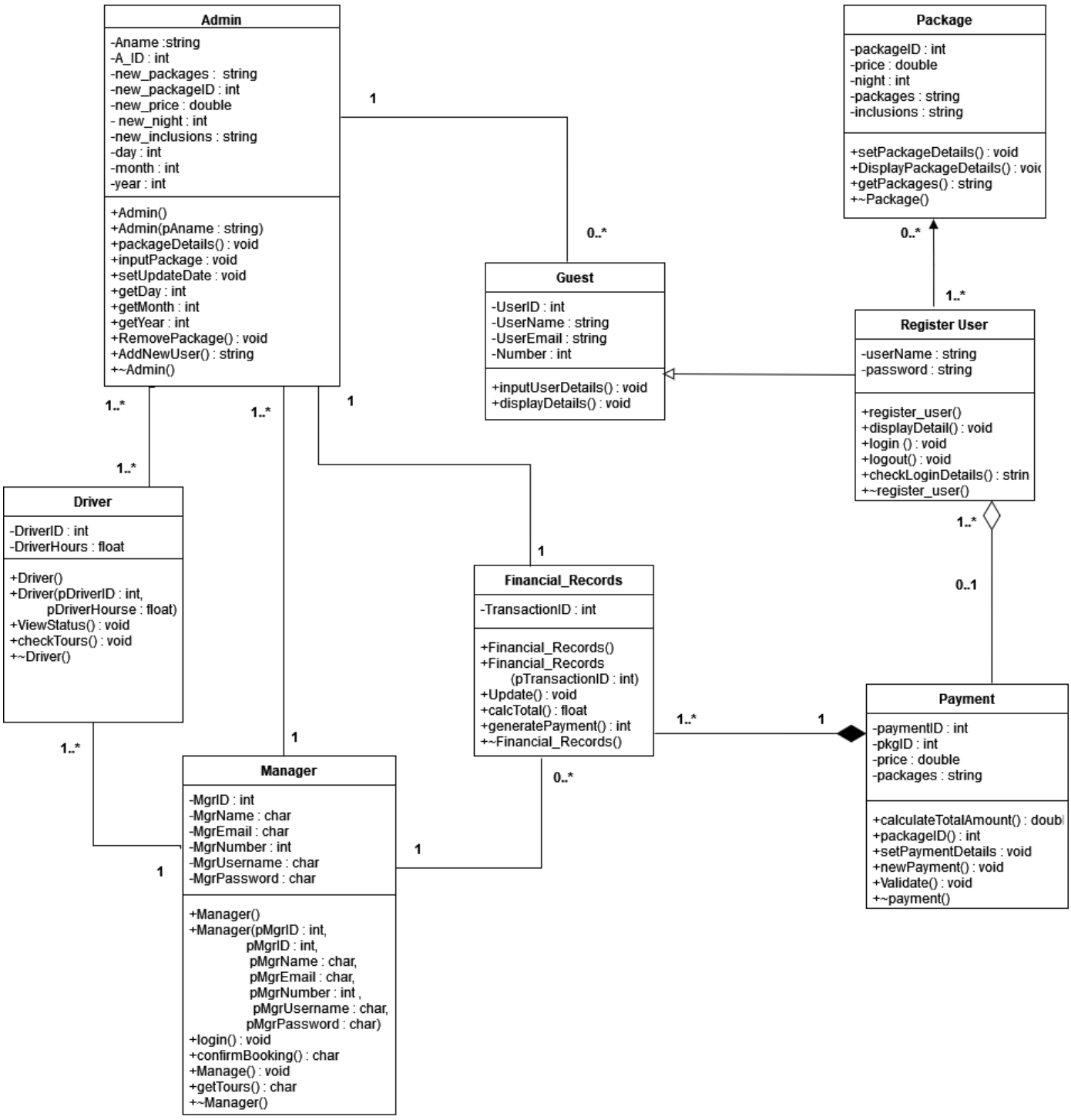
Payment	
Responsibility	**Collaborators**
Generate payment ID	Packages
Check payment details	Registered User
Confirm payment details	
Make a new payment	
Validate payment details	

Manager	
Responsibility	**Collaborators**
Login to the system	
Check reports	Financial records
Manage tours	Packages, Driver
Confirm bookings	Registered user

Driver	
Responsibility	**Collaborators**
Login to the system	
Check tours	Registered user, Admin
View user status	Registered user

Financial Records	
Responsibility	**Collaborators**
Store details of transaction	Payment
Update	Manger
Generate payment details	Payment

Class Diagram (UML Notation)



Class Header Files

Guest.h

```
// IT21191824
//Dias D. T. M.

class guest
{
private:
	int UserID;
	string UserName;
	string UserAddress;
	string UserEmail;
	int Number;

public:
	void inputUserDetails();
	void displayDetails();
};
```

Registered User.h

```
// IT21328534
//Thenuwan D. A.

class register_user :
	public guest
{
protected:
	string userName;
	string password;

public:
	register_user();
	register_user(string puserName, string ppassword, int
		pUserID, string pUserAddress, string
		pUserEmail);
	void displayDetails();
	void login();
	void logout();
	string checkLoginDetails();
	~register_user();
};
```

Packages.h

```
// IT21333552
// Abeysinghe I. U.

class Package {
private:
	int packageID;
	double price;
	int night;
	string packages;
	string inclusions;

public:
	void setPackageDetails(int SpackageID, double Sprice, int
		Snight, string Spackages, string Sinclusions);
	void DisplayPackageDetails();
	string getPackages();
	//desturctor
	~Package();
};
```

Admin.h

```
// IT21333552
// Abeysinghe I. U.

class Admin {
private:
	string Aname;
	int A_ID;
	string new_packages;
	int new_packageID;
	double new_price;
	int new_night;
	string new_inclusions;
	int day, month, year;
public:
	Admin();
	Admin(string pAname, int PA_ID);
	void AdminDetails();
	void packageDetails(string pNew_packages, int
		pnew_packageID, double pnew_price, int pnew_night,
		string pnew_inclusions);
```

```
        void inputPackage();
        void setUpdateDate();
        int getDay();
        int getMonth();
        int getYear();
        void RemovePackages();
        string AddNewUser();
        ~Admin();

};
```

Payment.h

```
// IT21328534
//Thenuwan D. A.

class payment {
private:
        int paymentID;
        int pkgID;
        double price;
        string packages;


public:
        double calculateTotalAmount();
        int packageID();
        void setPaymentDetails(int P_paymentID, double P_price,
                string P_packages, int P_pkgID);
        void newPayment();
        void Validate();
        ~payment();
};
```

Manager.h

```
// IT21327094
//Somarathne D. K.

class Manager {
private:
	int MgrID;
	char MgrName[30];
	char MgrEmail[30];
	int MgrNumber;
	char MgrUsername[20];
	char MgrPassword[20];
public:
	Manager();
	Manager(int pMgrID, const char pMgrName[], const char
		pMgrEmail[], int pMgrNumber, const char
		 pMgrUsername[], const char pMgrPassword[]);
	void login(const char L_MgrUsername[], const char
		  L_MgrPassword[]);
	char confirmBooking();
	void Manage();
	char getTours();
	~Manager();
};
```

Driver.h

```
// IT21327230
//K. S. Godage

class Driver {
private:
	int DriverID;
	float DriverHours;
public:
	Driver();
	Driver(int pDriverID, float pDriverHours);
	void ViewStatus();
	void checkTours();
	~Driver();
};
```

Financial Records.h

```
// IT21191824
// Dias D. T. M.

class Financial_Records {
private:
	int TransactionID;

public:
	Financial_Records();
	Financial_Records(int pTransactionID);
	void Update();
	float calcTotal();
	int generatePayment();
	~Financial_Records();
};
```

Class cpp Files

Guest.cpp

```
// IT21191824
//Dias D. T. M.

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


void guest::inputUserDetails()
{
	cout << "Enter ID : ";
	cin >> UserID;

	cout << "Enter Name :";
	cin >> UserName;

	cout << "Enter Address  :";
	cin >> UserAddress;

	cout << "Enter Email  : ";
	cin >> UserEmail;

	cout << "Enter phone number :";
```

```
	cin >> Number;
}


void guest::displayDetails()
{
	cout << "ID  :" << UserID << endl
		<< "Name  :" << UserName << endl
		<< "Address  : " << UserAddress << endl
		<< "Email  : " << UserEmail << endl
		<< "Phone number  :" << Number << endl;
}
```

Registered User.cpp

```
// IT21328534
//Thenuwan D. A.

#include "Registered_User.h"
#include"Guest.h"
#include<iostream>
using namespace std;


register_user::register_user()
{
	userName = "";
	password = "";
}
register_user::register_user(string puserName, string ppassword,
int pUserID,
				string pUserAddress, string
pUserEmail)
{
	userName = puserName;
	password = ppassword;
	UserID = pUserID;
	UserAddress = pUserAddress;
	UserEmail = pUserEmail;
}
void register_user::displayDetails()
{

}
void register_user::login()
{

}
```

```
void register_user::logout()
{

}
string register_user::checkLoginDetails()
{
	return 0;
}
register_user::~register_user()
{
	//Destructor
}
```

Packages.cpp

```
// IT21333552
// Abeysinghe I. U.

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


void Package::setPackageDetails(int SpackageID, double Sprice,
int Snight, string
                                Spackages, string Sinclusions)
{
	packageID = SpackageID;
	price = Sprice;
	night = Snight;
	packages = Spackages;
	inclusions = Sinclusions;
}

void Package::DisplayPackageDetails()
{
	cout << "Package Number        : " << packageID << endl
		<< "Package name          : " << packages << endl
		<< "Price             : RS." << price << endl
		<< "Nigth             :  " << night << endl
		<< "Inclusions        :  " << inclusions << endl <<
endl;
}

string Package::getPackages()
{
	return packages;
}
```

```
Package :: ~Package() {

	cout << "Package deleted." << endl;
}
```

Admin.cpp

```
// IT21333552
// Abeysinghe I. U.

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


Admin::Admin() {
	Aname = "";
	A_ID = 0;
}

Admin::Admin(string pAname, int PA_ID) {

	Aname = pAname;
	A_ID = PA_ID;
}
void Admin::AdminDetails()
{
	cout << "Enter name : ";
	cin >> Aname;

	cout << "Enter ID : ";
	cin >> A_ID;
}
void Admin::packageDetails(string pNew_packages, int
pnew_packageID, double
				pnew_price, int pnew_night, string
pnew_inclusions)
{
	new_packages = pNew_packages;
	new_packageID = pnew_packageID;
	new_price = pnew_price;
	new_night = pnew_night;
	new_inclusions = pnew_inclusions;
}

void Admin::inputPackage() {
```

```
	cout << "Enter New Package  : ";
	cin >> new_packages;

	cout << "Enter package number :";
	cin >> new_packageID;

	cout << "Enter price  :";
	cin >> new_price;

	cout << "Enter how many nigths person can stay :";
	cin >> new_night;

	cout << "Enter inclusions : ";
	cin >> new_inclusions;
}
void Admin::setUpdateDate()
{
	cout << "Day :";
	cin >> day;
	cout << "Month :";
	cin >> month;
	cout << "Year :";
	cin >> year;
}
int Admin::getDay()
{
	return day;

}
int Admin::getMonth()
{
	return month;

}
int Admin::getYear()
{
	return year;

}
void Admin::RemovePackages() {}

string Admin::AddNewUser() {}

Admin::~Admin()
{
	cout << "new package deleted." << endl;
}
```

Payment.cpp

```
// IT21328534
//Thenuwan D. A.

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


void payment::setPaymentDetails(int P_paymentID, double P_price,
string
                                    P_packages, int P_pkgID)
{
     paymentID = P_paymentID;
     price = P_price;
     packages = P_packages;
     pkgID = P_pkgID;
}
double payment::calculateTotalAmount()
{
     return price;
}

int payment::packageID()
{
     cout << "Enter package ID  : ";
     cin >> pkgID;
     return pkgID;
}

void payment::newPayment()
{                              }

void payment::Validate()
{                              }

payment::~payment()
{
     cout << "Payment deleted.";
}
```

Manager.cpp

```
// IT21327094
//Somarathne D. K.

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


Manager::Manager()
{
	MgrID = 0;
	strcpy(MgrName, "");
	strcpy(MgrEmail, "");
	strcpy(MgrPassword, "");
	strcpy(MgrUsername, "");
	MgrNumber = 0000000000;
}
Manager::Manager(int pMgrID, const char pMgrName[], const char
pMgrEmail[], int
		pMgrNumber, const char pMgrUsername[], const
char
		 pMgrPassword[])
{
	MgrID = pMgrID;
	MgrNumber = pMgrNumber;
	strcpy(MgrName, pMgrName);
	strcpy(MgrEmail, pMgrEmail);
	strcpy(MgrPassword, pMgrPassword);
	strcpy(MgrUsername, pMgrUsername);

}
void Manager::login(const char L_MgrUsername[], const char
L_MgrPassword[])
{
	strcpy(MgrPassword, L_MgrPassword);
	strcpy(MgrUsername, L_MgrUsername);
}
char Manager::confirmBooking()
{

}
void Manager::Manage()
{

}
char Manager::getTours()
{
```

```
	return 0;
}
Manager::~Manager()
{
	//Destructor
}
```

Driver.cpp

```
// IT21327230
//K. S. Godage

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


Driver::Driver()
{
	DriverID = 0;
	DriverHours = 0.0;
}

Driver::Driver(int pDriverID, float pDriverHours)
{
	DriverID = pDriverID;
	DriverHours = pDriverHours;
}
void Driver::ViewStatus()
{

}
void Driver::checkTours()
{

}
Driver::~Driver()
{
	cout << "Tour deleted." << endl;
}
```

Financial Records.cpp

```
// IT21191824
// Dias D. T. M.

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

Financial_Records::Financial_Records()
{
	TransactionID = 0;
}
Financial_Records::Financial_Records(int pTransactionID)
{
	TransactionID = pTransactionID;
}
void Financial_Records::Update() {}

int Financial_Records::generatePayment() {}

float Financial_Records::calcTotal() {}

Financial_Records::~Financial_Records()
{
	cout << "Record is deleted.";
}
```

Main Program

main.cpp

```
#include "Guest.h"
#include "Packages.h"
#include "Payment.h"
#include "Admin.h"
#include "Registered_User.h"
#include "Manager.h"
#include "Driver.h"
#include "Financial_Records.h"
#include <iostream>
using namespace std;

int main()
{
//-------------------------start Guest class----------------
     guest g1;
     string letter;

     cout << "Do you want register now (yes-Y or y/no-N or n).
          : ";
     cin >> letter;
     cout << endl << endl;

     if (letter == "Y" || letter == "y")
     {
          g1.inputUserDetails();
          cout << endl << endl;
          g1.displayDetails();
     }
     else
     {
          cout << "Okay, Have a good day.";
     }
//--------------------end Guest class--------------------------

//---------------------------start Package class--------------

     Package p1, p2, p3, p4;

     p1.setPackageDetails(5001, 20000.00, 2, "Beach Package",
"Meals , Flights , Accommodation , Transfer , Sightseeing ");

     p2.setPackageDetails(5002, 50000.00, 3, "Luxury Package",
"Meals , Accommodation , Transfer , Sightseeing ");
```

```
     p3.setPackageDetails(5003, 80000.00, 5, "WildLife Package",
"Meals , Flights , Accommodation , Sightseeing ");

     p4.setPackageDetails(5004, 30000.00, 2, "Family Package", "
Flights, Accommodation, Transfer, Sightseeing");


     p1.DisplayPackageDetails();
     cout <<
"*************************************************************
************************************" << endl;
     p2.DisplayPackageDetails();
     cout <<
"*************************************************************
************************************" << endl;
     p3.DisplayPackageDetails();
     cout <<
"*************************************************************
************************************" << endl;
     p4.DisplayPackageDetails();

//----------------------------end Package class----------------

//----------------------------start Payment class--------------

     int i, a, b;
     double total = 0.0;
     payment py1;
     cout << "Enter Package quantity :";
     cin >> i;

     for (a = 1; a <= i; a++)
     {
          py1.calculateTotalAmount() == 0.0;
          {
               py1.packageID() == b;
               if (b == 50001)
               {
                    py1.calculateTotalAmount() ==
                  py1.calculateTotalAmount() + 20000;
                    return py1.calculateTotalAmount();
               }
               else if (b == 50002)
               {
                    py1.calculateTotalAmount() ==
                    py1.calculateTotalAmount() + 30000;
                    return py1.calculateTotalAmount();
               }
               else if (b == 50003)
               {
```

```
                    py1.calculateTotalAmount() ==
                    py1.calculateTotalAmount() + 50000;
                    return py1.calculateTotalAmount();
               }
               else if (b == 50004)
               {
                    py1.calculateTotalAmount() ==
                    py1.calculateTotalAmount() + 80000;
                    return py1.calculateTotalAmount();
               }
               else
                    break;
               total = total + py1.calculateTotalAmount();
          }
     }
     cout << "TOTAL AMOUNT OF BOOKING PACKAGE  : " << total <<
               endl << endl;

     py1.newPayment();
     py1.Validate();

//----------------------------end Payment class-----------------

//----------------------------start Admin class-----------------

     Admin* pkg1;
     string decision;
     string letter;
     int i = 1;
     while (i <= 200)
     {
          cout << "Do you want update the system (YES-Y/NO-N).
                    :  ";
          cin >> decision;

          if (decision == "Y")
          {
               pkg1 = new Admin();
               pkg1->AdminDetails();
               pkg1->inputPackage();
               pkg1->setUpdateDate();
               cout << "Updated date  :" << pkg1->getDay() <<
                    "/" << pkg1->getMonth() << "/"
                     << pkg1->getYear() << endl;
               delete pkg1;

               cout << "Do you want remove package or add new
                      user to the system(YES-Y/NO-N).   :  ";
               cin >> letter;
```

```
                if (letter == "Y")
                {
                     pkg1->RemovePackages();
                     pkg1->AddNewUser();
                }
                else
                {
                     continue;
                }

           }
           else {
                break;
           }
           i++;
           cout <<
"*************************************************************
************************************" << endl;
     }

//----------------------------end Admin class-------------------

//----------------------------start Resitered user class--------

     register_user RG1, RG2;

     RG1.login();
     RG1.displayDetails();

     RG2.login();
     RG2.displayDetails();

//----------------------------end Resitered user class----------

//----------------------------start Manager class---------------

     Manager Mgr1, Mgr2;

     Mgr1.login();
     Mgr1.getTours();
     Mgr1.confirmBooking();
     Mgr1.Manage();

     Mgr2.login();
     Mgr2.getTours();
     Mgr2.confirmBooking();
     Mgr2.Manage();

//----------------------------end Manager class---------------
```

```
//----------------------------start Driver class--------------

	Driver* D1;
	Driver* D2;

	D1 = new Driver();
	D2 = new Driver();

	D1->checkTours();
	D1->ViewStatus();

	D2->checkTours();
	D2->ViewStatus();

	delete D1;
	delete D2;

//----------------------------end Driver class----------------

//------------------start Financial_Records class--------------

	Financial_Records F1, F2;

	F1.Update();
	F1.generatePayment();
	F1.calcTotal();

	F2.Update();
	F2.generatePayment();
	F2.calcTotal();

//----------------------end Financial_Records class------------

	return 0;
}
```