Topic : **Wild-life Safari Trip Management System**

Group no : **MLB_10.02_02**

Campus : **Malabe**

Submission Date:

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
IT21189944	Madusanka G.K. I	0766859740
IT21190216	Thisera W.N.M	0711442654
IT21379956	Hettiarachchi V. E	0754894667
IT21377280	Rajapaksha C. S	0767651004
IT21189630	Hewavitharana D. L	0783676104

Contents

1. Description of the requirements

- All the users can view the website, make donations, check availability, check packages, and check the FAQ section.
- Any visitor can register for the website by providing a name, username, address, password, date of birth, phone number, gender, email, and salutation.
- After registering, the system member can log in using his username and password.
- Registered users can make donations. Members can use different payment methods for donations, such as PayPal, credit card, debit card, and bank deposits.
- Registered users can check packages and make a reservation request. For reservation requests, the user must provide the name, address, country, date needed to be reserved, phone no, email, number of adults and children, and the selected package. If they want, customers can also cancel their reservation requests.
- Registered users can make inquiries by providing phone numbers and email.
- Registered users can give feedback about their wild trips.
- Registered users can view, delete, or edit their profiles.
- The tour guide can log in to the system by entering login credentials and can check/reply to customer inquiries.
- Administrator login into the website by providing a username and password
- The administrator can approve or decline the reservations.
- Administrators can activate and deactivate user accounts.
- The administrator can update safari details.
- The administrator can approve or reject feedback from members.
- Managers can generate Donation reports, Reservations reports and Hotel booking reports.
- Manager can manage Donations and salaries of employee

(The only payment on this site is donations. Payment for reservations is made after the confirmation. Those reservation transactions go directly to the bank)

2. Classes Identified

- User
- Donation
- Reservation
- Package
- Feedback
- Reports
- Administrator (inherited from the user)
- Manager (inherited from the user)
- Registered user (inherited from the user)
- Tour Guide (inherited from the user)
- Card (inherited from Donations)

3. CRC card

User	
Responsibilities:	**Collaboration:**
Log in to the system	
Validate user	
Donate	Donation
Check packages	Packages
Check FAQ	
Check availabilities	Package

Donation	
Responsibilities:	**Collaboration:**
Store donation details	
Validate payment	
Show donation details	

Reservation	
Responsibilities:	**Collaboration:**
Store reservation details	
Show reservation details	
Get package ID	Package

Packages	
Responsibilities:	**Collaboration:**
Store package details	
Show package details	
Show availability	

Inquiry	
Responsibilities:	**Collaboration:**
Store details about inquiry	
Show inquiry	

Feedback	
Responsibilities:	**Collaboration:**
Store details about Feedback	
Show approved feedbacks	

Report	
Responsibilities:	**Collaboration:**
Generate reports about donations	Donation
Generate reports of accepted reservations	Reservation
Generate reports of cancelled reservations	
Generate reports about packages	Packages

Registered User	
Responsibilities:	**Collaboration:**
Store customer details	
Make a reservation request	Reservation
Cancel reservation request	Reservation
Make inquires	Inquires
Give feedback	feedback
Manage profile	

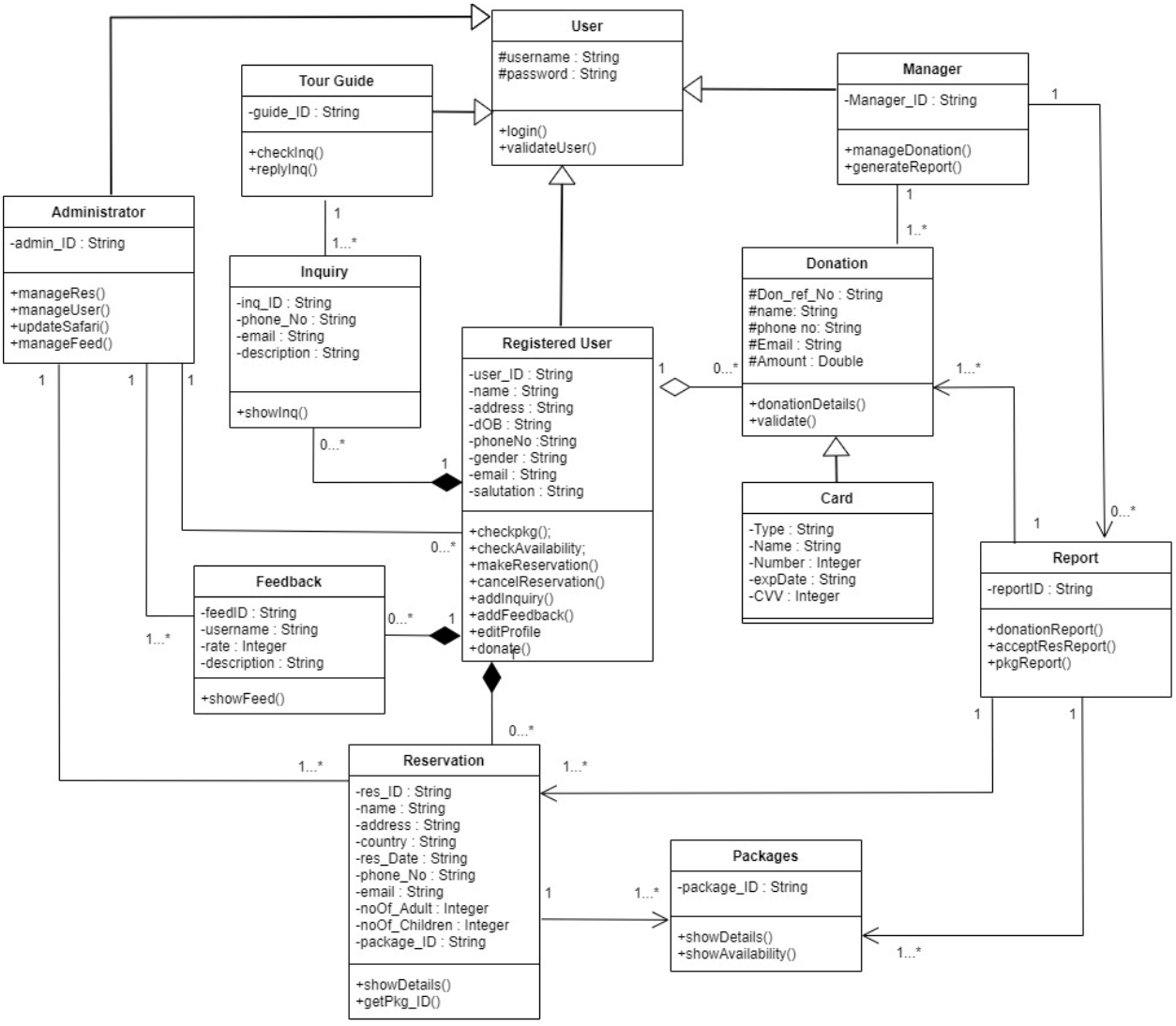
Administrator	
Responsibilities:	**Collaboration:**
Store details about administrator	
Approve reservation	Reservation
Decline reservation	Reservation
Manage member accounts	Registered User
Update safari details	
Manage feedbacks	Feedback

Manager	
Responsibilities:	**Collaboration:**
Store details about manager	
Generate reports	Report
Manage Donations	Donation

Tour Guide	
Responsibilities:	**Collaboration:**
Store details about tour guide	
Check inquires	Inquires
Reply to inquires	Inquires

Card	
Responsibilities:	**Collaboration:**
Authorized the card	

4. Class diagram



5. Coding for the classes

1.main.cpp

```
//main.cpp

#include<iostream>
#include "Registered_user.h"
#include "Manager.h"
#include "Tour_Guide.h"
#include "Administrator.h"
#include "Inquiry.h"
#include "Donation.h"
#include "Feedback.h"
#include "Reservation.h"
using namespace std;


int main() {
  //data insert to registered user
        Registered_user* regUser = new Registered_user("chillbroh", "1234",
"U0001", "Ishara", "Galle", "1998/06/14", "0766859740", "male",
"ishara410@gmail.com", "mr");
  cout << "Details of User : " << endl;
  regUser->display(); //display registered user details
  cout << endl;

 //data insert to manager
  Manager *manage = new Manager("Ehani01","Ehani123","M0001");
  cout << "Details of Manager : " << endl;
  manage->display(); //display manager details
  cout << endl << endl;

  //date insert to tour guide
  Tour_Guide *tG = new Tour_Guide("Dileep", "Dileep123","TG0001");
  cout << "Details of Tour_Guide : " << endl;
  tG->display(); //display tour guide details
  cout << endl << endl ;

  //data insert to administrator
  Administrator *admin = new Administrator("Chala","Chalani123","A0001");
  cout << "Details of Administrator : " << endl;
  admin->display(); //display admin details
  cout << endl << endl;

  //data insert to donation
  Donation *donation = new
Donation("D0001","Nimadi","0758469856","nimadi41@gmail.com",1000.0);
  cout << "Details of Donation : " << endl;
  Registered_user *Donate = new Registered_user(); //display donation details
  Donate->addDonation(donation);
  Donate->displayDonation();
```

```
  cout << endl << endl;

  //data insert to feed
  Registered_user *feed;
  feed = new Registered_user;
  feed-> addFeed("F0001","chalan","safadfakdhfuaef adfjkad ffkajdlfaieu
fajdlfaei adfjlajfaiej alkdjfl adhfaehf aljkdfa",5);
  feed-> displayFeed(); //display feed details
  cout << endl << endl;

  //data insert to inquriy
  Registered_user *inq;
  inq = new Registered_user;
  inq->addInq("Inq001","076453743","shanperera@gmail.com","can I know about
asldjlsaiiueualn?");
  inq->displayinq(); //display Inquiry details
  cout << endl << endl;

  //data insert to reservation
  Registered_user *reserve;
  reserve = new Registered_user;
  reserve->addReservation("R0001","Ajantha","colombo","Sri
Lanka","2022/05/30","0768957420","ajantha@gmail.com",6,3);
  reserve->displayReservation(); //display reservation details


  return 0;

  }
```

User.h

```
//Class User
#pragma once
class User
{
protected:
	char userName[30];
	char password[15];

public:
  User();
	User(char pusername[], char pass[]);
  void display();
	void login();
	void validateUser();
	~User();
};
```

User.cpp

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

User::User(){
  strcpy(userName, "");
	strcpy(password, "");
}
User::User(char pusername[], char pass[]) {
	strcpy(userName, pusername);
	strcpy(password, pass);
}
void User::display (){
  cout << userName << endl << password << endl;
}
void User::login() {

}
void User::validateUser() {

}
User::~User() {

}
```

Registered_user.h

```
#pragma once
#include "User.h"
#include "Inquiry.h"
#include "Feedback.h"
#include "Reservation.h"
#include "Donation.h"
#define SIZE 2

class Registered_user : public User
{
private:
        char user_ID[10];
        char name[30];
        char address[100];
        char dob[20];
        char phone[20];
        char gender[10];
        char email[30];
        char salutation[10];
  Inquiry *inq[SIZE];
  Feedback *feed[SIZE];
  Reservation *reservation[SIZE];
  Donation *donate[SIZE];
public:
  Registered_user();
        Registered_user(char u_username[], char u_pass[], char u_id[], char
u_name[], char u_address[], char u_dob[], char u_phone[], char u_gender[],
char u_email[], char u_salutation[]);
  void display();
        void checkpkg();
        void checkAvailability();
        void makeReservation();
        void cancelReservation();
        void addInquiry();
        void addFeedback();
        void editProfile();
        void addDonation(Donation *d1);
  void displayDonation();
  void addInq(char Inq_id[],char Inq_phone[],char Inq_mail[],char
Inq_descript[]);
  void displayinq();
  void addFeed(char feed_id[],char feed_username[],char
feed_description[],int feed_rate);
  void displayFeed();
  void addReservation(char res_id[],char res_name[],char res_address[],char
res_country[],char res_date[],char res_phone_no[],char res_email[],int
res_adult, int res_children);
  void displayReservation();
  ~Registered_user();
};
```

Registered_user.cpp

```
#include "Registered_user.h"
#include "Inquiry.h"
#include "Feedback.h"
#include "Reservation.h"
#include<iostream>
#include<cstring>
using namespace std;
//default constructer for registered user
Registered_user::Registered_user(){
  strcpy(user_ID, "");
        strcpy(name, "");
        strcpy(address, "");
        strcpy(dob, "");
        strcpy(phone, "");
        strcpy(gender, "");
        strcpy(email, "");
        strcpy(salutation, "");
}
//constructer with parameters
Registered_user::Registered_user(char u_username[], char u_pass[], char
u_id[], char u_name[], char u_address[], char u_dob[], char u_phone[], char
u_gender[], char u_email[], char u_salutation[])
        :User(u_username, u_pass) {
        strcpy(user_ID, u_id);
        strcpy(name, u_name);
        strcpy(address, u_address);
        strcpy(dob, u_dob);
        strcpy(phone, u_phone);
        strcpy(gender, u_gender);
        strcpy(email, u_email);
        strcpy(salutation, u_salutation);

}
void Registered_user::display(){
  User::display();
  cout << user_ID << endl << name << endl << address << endl << dob << endl
<< phone << endl << gender << endl << email << endl << salutation << endl;
}
void Registered_user::checkpkg() {

}
void Registered_user::checkAvailability() {

}
void Registered_user::makeReservation() {

}
void Registered_user::cancelReservation() {

}
```

```
void Registered_user::addInquiry() {

}
void Registered_user::addFeedback() {

}
void Registered_user::editProfile() {

}

//donation part in registered user (Aggregation)
void Registered_user::addDonation(Donation *d1) {
          donate[0] = d1;

}

void Registered_user::displayDonation(){
    donate[0]->donationDetails();
}

//inquiry part in registered user (composition)
void Registered_user::addInq(char Inq_id[],char Inq_phone[],char
Inq_mail[],char Inq_descript[]){
          inq[0] = new Inquiry(Inq_id,Inq_phone,Inq_mail, Inq_descript);
}
void Registered_user::displayinq(){
      inq[0]->showInq();
}

// feedback part in registered user (composition)
void Registered_user::addFeed(char feed_id[],char feed_username[],char
feed_description[],int feed_rate){
  feed[0] = new Feedback(feed_id,feed_username,feed_description,feed_rate);
}
void Registered_user::displayFeed(){
  feed[0]->showFeed();
}
//reservation part in registered user (composition)
void Registered_user::addReservation(char res_id[],char res_name[],char
res_address[],char res_country[],char res_date[],char res_phone_no[],char
res_email[],int res_adult, int res_children){

 reservation[0] = new Reservation (res_id, res_name, res_address,
res_country,res_date, res_phone_no,res_email,res_adult, res_children);

}
void Registered_user::displayReservation(){
  reservation[0]->showDetails();
}
Registered_user::~Registered_user(){

}
```

Administrator.h

```
#pragma once
#include "User.h"
#include "Feedback.h"
#include "Registered_user.h"
#include "Reservation.h"

class Administrator : public User{
    private:
      char Admin_ID[10];
    public:
      Administrator();
      Administrator(char M_username[],char M_pass[], char A_ID[]);
      void display();
      void ManageRes();
      void manageUser();
      void manageSafari();
      void manageFeed();
      ~Administrator();
};
```

Administrator.cpp

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

Administrator:: Administrator(){
  strcpy(Admin_ID,"");
}
Administrator::Administrator(char a_username[], char a_pass[], char a_ID[])
    :User(a_username,a_pass){
      strcpy(Admin_ID,a_ID);
    }
void Administrator::display(){
      User::display();
      cout << Admin_ID << endl;
}
void Administrator::ManageRes(){

}
void Administrator::manageUser(){

}
void Administrator::manageSafari(){

}
void Administrator::manageFeed(){
```

```
}
Administrator::~Administrator(){

}
```

Manager.h

```
#pragma once
#include "User.h"
#include "Report.h"
class Manager : public User{
    private:
      char Manager_ID[10];
    public:
      Manager();
      Manager(char M_username[],char M_pass[], char M_ID[]);
      void display();
      void ManageDonation();
      void generateReport();
      ~Manager();
};
```

Manager.cpp

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

Manager::Manager(){
  strcpy(Manager_ID,"");
}
Manager::Manager(char M_username[],char M_pass[], char M_ID[] )
:User(M_username, M_pass){
  strcpy(Manager_ID, M_ID);
}
void Manager::display(){
  User::display();
  cout << Manager_ID;
}
void Manager::ManageDonation(){

}
void Manager::generateReport(){

}
Manager::~Manager(){

}
```

Tour_Guide.h

```
#pragma once
#include "User.h"
#include "Inquiry.h"
class Tour_Guide : public User{
    private:
        char guide_ID[10];
    public:
        Tour_Guide();
        Tour_Guide(char g_username[], char g_pass[], char G_ID[]);
        void display();
        void checkInq();
        void replyInq();
        ~Tour_Guide();
};
```

Tour_Guide.cpp

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

Tour_Guide::Tour_Guide(){
   strcpy(guide_ID,"");
}
Tour_Guide::Tour_Guide(char g_username[], char g_pass[], char G_ID[])
    :User(g_username,g_pass){
      strcpy(guide_ID,G_ID);
    }
void Tour_Guide::display(){
      User::display();
      cout << guide_ID << endl;
}
void Tour_Guide::checkInq(){

}
void Tour_Guide::replyInq(){

}
Tour_Guide::~Tour_Guide(){

}
```

Donation.h

```
#pragma once

#include "Report.h"

class Donation{
    protected:
        char Don_ref_No[10];
        char name[15];
        char phoneNo[20];
        char email[20];
        double amount;
    public:
        Donation();
        Donation(char ref_no[],char d_name[],char d_phone[],char
d_email[],double d_amount);
        void donationDetails();
        void validate();
        ~Donation();
};
```

Donation.cpp

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

Donation::Donation(){
        strcpy(Don_ref_No,"");
        strcpy(name,"");
        strcpy(phoneNo,"");
        strcpy(email,"");
        amount = 0;
}
Donation::Donation(char ref_no[],char d_name[],char d_phone[],char
d_email[],double d_amount){
        strcpy(Don_ref_No,ref_no);
        strcpy(name,d_name);
        strcpy(phoneNo,d_phone);
        strcpy(email,d_email);
        amount = d_amount;
}
void Donation::donationDetails(){
      cout << Don_ref_No << endl << name << endl << phoneNo << endl << email
<< amount << endl;
}
void Donation::validate(){

}
```

```
Donation::~Donation(){

}
```

Card.h

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

class Card : public Donation {
      private:
        char type[20];
        char name[20];
        int number;
        char expDate[15];
        int CVV;
}; //not going to store these details
```

Reservation.h

```
#pragma once
#include "Package.h"
#include "Report.h"
#define SIZE 2
class Reservation{
    private:
        char res_ID[10];
        char name[20];
        char address[100];
        char country[20];
        char res_date[15];
        char phone_no[20];
        char email[50];
        int noOfAdult;
        int noOfChildren;

    public :
      Reservation();
      Reservation(char r_id[],char r_name[],char r_address[],char
r_country[],char r_date[],char r_phone_no[],char r_email[],int adult, int
children);
      void showDetails();
      void setpkg_ID();
      void showpkg();
      int getPkg_ID();
      ~Reservation();
};
```

Reservation.cpp

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

Reservation::Reservation(){
        strcpy(res_ID,"");
        strcpy(name,"");
        strcpy(address,"");
        strcpy(country,"");
        strcpy(res_date,"");
        strcpy(phone_no,"");
        strcpy(email,"");
        noOfAdult = 0;
        noOfChildren = 0;
}
Reservation::Reservation(char r_id[],char r_name[],char r_address[],char
r_country[],char r_date[],char r_phone_no[],char r_email[],int adult, int
children){
        strcpy(res_ID,r_id);
        strcpy(name,r_name);
        strcpy(address,r_address);
        strcpy(country,r_country);
        strcpy(res_date,r_date);
        strcpy(phone_no,r_phone_no);
        strcpy(email,r_email);
        noOfAdult = adult;
        noOfChildren = children;

}
void Reservation::showDetails(){
  cout << name << endl << address << endl << country << res_date << endl <<
phone_no << endl << email << endl << "No of adults : " << noOfAdult << endl
<< "No of Children :" << noOfChildren <<  endl;
}
void Reservation::setpkg_ID(){

}
void Reservation::showpkg(){

}
int Reservation::getPkg_ID(){

}
Reservation::~Reservation(){

}
```

Feedback.h

```
#pragma once

class Feedback{
    private:
      char feed_ID[10];
      char username[20];
      char description[500];
      int rate;
    public:
     Feedback();
      Feedback(char f_id[],char f_username[],char f_description[],int
f_rate);
      void showFeed();
      ~Feedback();
};
```

Feedback.cpp

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

Feedback::Feedback(){
      strcpy(feed_ID,"");
      strcpy(username,"");
      strcpy(description,"");
      rate = 0;
}
Feedback::Feedback(char f_id[],char f_username[],char f_description[],int
f_rate){
      strcpy(feed_ID,f_id);
      strcpy(username,f_username);
      strcpy(description,f_description);
      rate = f_rate;
}
void Feedback::showFeed(){
   cout << username << endl << description << endl << rate << endl;
}
Feedback::~Feedback(){

}
```

Inquiry.h

```
#pragma once
#include "Tour_Guide.h"
class Inquiry{
    private:
      char inqID[10];
      char phoneNo[20];
      char email[50];
      char description[500];
    public:
       Inquiry();
       Inquiry(char inq_ID[],char phone[], char mail[],char descript[]);
      void showInq();
      ~Inquiry();
};
```

Inquiry.cpp

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

Inquiry::Inquiry(){
      strcpy(inqID,"");
      strcpy(phoneNo,"");
      strcpy(email,"");
      strcpy(description,"");
}
Inquiry::Inquiry(char inq[],char phone[], char mail[],char descript[]){
      strcpy(inqID,inq);
      strcpy(phoneNo,phone);
      strcpy(email,mail);
      strcpy(description,descript);
}
void Inquiry::showInq(){
      cout << inqID << endl << phoneNo << endl << email << endl <<
description << endl;
}
Inquiry::~Inquiry(){

}
```

Package.h

```
#pragma once
#include "Report.h"
class Package{
  private:
      char name[20];
      float price;
      int nights;
  public:
      Package(char pname[],float pPrice,int pnights);
      void showpackagedetails();
      void showAvailability();
      ~Package();
  };
```

Package.cpp

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

Package::Package(char pname[],float pPrice,int pnights){
    strcpy(name,pname);
    price = pPrice;
    nights = pnights;
}
void Package::showpackagedetails(){
      cout << "name :" << name << endl << "Price : " << price << endl <<
"Nights :" << nights;
}
void Package::showAvailability(){

}
Package::~Package(){

}
```

Report.h

```
#pragma once
#include "Donation.h"
#include "Reservation.h"
#include "Package.h"

class Report{
    private:
      char report_ID[10];
    public:
      Report();
      Report(char id[]);
      void donationReport();
      void ResReport();
      void pkgReport();
      ~Report();
};
```

Report.cpp

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

Report::Report(char id[]){
  strcpy(report_ID,id);
}
void Report::donationReport(){


}
void Report::ResReport(){

}
void Report::pkgReport(){

}
Report::~Report(){

}
```