Topic : Event Photograph Management System

Group no : MLB_WD_CSNE_13_02

Campus : Malabe

Submission Date : 2022/05//20

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
IT21252372	Madhusankha W.V.S	0701352028
IT21264320	Rankothge K . A	0766367138
IT21315350	Wimaladharma A.G.D.A	0776540728
IT21359088	Perera M.R.D	0776929394
IT21242472	Gunawardhana K.A.S.H	0752041657

System Requirment

1. User can create an account as a customer , admin , staff or photographer.
2. Users can log in to the system anytime using his/her id as username and NIC as the password
3. User can reset the password using “Forgot password”.
4. Outsiders can give their feedback about the productivity of the website.
5. User can select the event and also select the payment method.
6. Customer can make a reservation by providing the location, date, time and package type.
7. The staff can assign photographers and appoint a team for each customers’ event.
8. The customer can give their feedback via the website and the administrator prioritize the feedbacks in the website and reply to the customer feedbacks.
9. Only registered users will be able to book online event venues.
10. User will get any instant message only through email address but not mobile numbers
11. Registered users can visit the system any number of times and also, they can browse any kind of packages
12. The sensitive information such as credit card numbers should be encrypted and then stored in the system to ensure the security of the customers’ data in the system

Exercise 1

1.Classes Identified

1. Staff
2. Photographer
3. Customer
4. Reservation
5. Delivery_item
6. Location
7. Payment
8. Feedback
9. Administrator
10. Admin_logn

2.CRC Cards

Class : Staff	
Responsibilities	**collaborations**
login to the system	
Store staff details	
Get photographer details	photographer
Send email to photographer	
Assign photographer	

Class : Photographer	
Responsibilities	**collaborations**
login to the system	
Store photographer details	
Respond to assigned work	

Class : Feedback	
Responsibilities	**Collaborations**
Store customer feedback	

Class : AdminLogin	
Responsibilities	**Collaborations**
Store login details	
check login credentials	

Class : Reservation	
Responsibilities	**collaborations**
Reservaton details of customer	Customer
Assign location	Location

Class : Location	
Responsibilities	**collaborations**
Store location details	

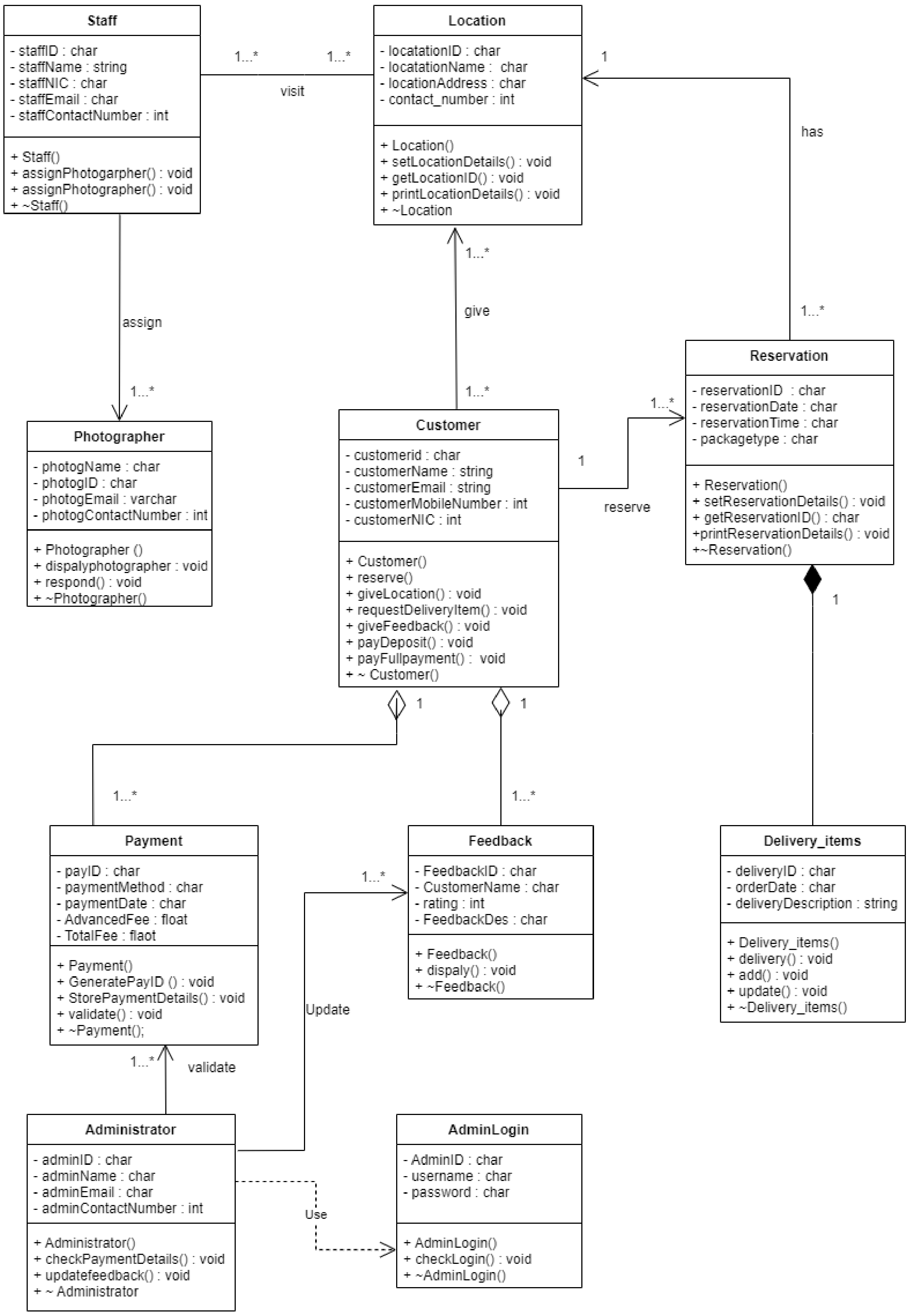
Class : Customer	
Responsibilities	**collaborations**
Reserve reservations	Reservation
Give the location	Location
Request delivery items	Delivery_item
Make a deposit	Payment
Make the full payment	Payment
Give feedback	Feedback

Class : Delivery_item	
Responsibilities	**collaborations**
Delivery the items	
Add new items	
Update the items	

Class : Payment	
Responsibilities	**collaborations**
Generate the payments	
Store the payment details	
Validate the payment	

Class : Administrator	
Responsibilities	**collaborations**
Check the payment methods	Payment
Update the feedback	Feedback

3.Uml Diagram



Exercise 2

1. Header files(.h files)

a)Staff.h

```
class staff
{
private:

	string StaffName;
	char StaffID[10];
	char S_NIC[10];
	char S_Email[50];
	int S_ContactNumber;

public:

	staff();
	staff(string Sname, const char SID[], const char SNIC[], const char
Smail[], int Scon);
	void displayStaff();
	void AssignPhotographer();
	~staff();

	};
```

b)Photographer.h

```
class photographer
{
private:

	char photog_Name[30];
	char photog_ID[10];
	char P_NIC[10];
	char P_Email[50];
	int P_ContactNumber;

public:

	photographer();
	photographer(const char Pname[], const char PID[], const char  PNIC[],
const char Pmail[], int Pcon);
	void displayPhotographer();
	void respond();
	~photographer();

	};
```

c)Reservation.h

```
class Reservation {
private:
	char resevationID[10];
	char reservationDate[10];
	char reservationTime[10];
	char package_type[20];
public:
	Reservation();
	Reservation(const char rID[], const char rdate[], const char rtime[],
const char packtype[]);
	void setReservationDetails(const char rID[], const char rdate[], const
char rtime[], const char packtype[]);
	char getReservationID();
	void printReservationDetails();
	~Reservation();
};
```

d)Location.h

```
class Location {
private:
	char locationID[10];
	char locationName[20];
	char locationAddress[100];
	int contact_number;

public:
	Location();
	Location(const char locationID[], const char locationName[], const char
locationAddress[], int contact_number);
	void setLocationdetails(const char lID[], const char lname[], const char
laddress[], int lcontactno);
	char getLocationID();
	void printLocationDetails();
	~Location();
	};
```

e)Customer.h

```
class Customer
{
private:
	char customerId[10];
	char customerName[20];
	char customerEmail[20];
	int customerMobileNumber[10];
	int customerNIC;

public:
	Customer();
	Customer(char cId[], char cName[], char cEmail[])

		void reserve();
	void giveLocation();
	void requestDeliveryItem();
	void giveFeedback();
	void payDeposit();
	void payFullPayment();
	~Customer();

};
```

f)Delivery_item.h

```
class Delivery_items
{
private:
	char deliveryId;
	char orderDate;
	string deliveryDescription;

public:
	Delivery_items();
	Delivery_items(char dId[], char dDate[], string dDescription[])
		void delivery();
	void add();
	void update();
	~Delivery_items();
	};
```

g)Payment.h

```
class Payment
{
private:
	char payId[10];
	char paymentMethod[10];
	char paymentDate[10];
	float advancedFee;
	float totalFee;

public:
	Payment();
	Payment(char pId[], char pDate[], float ptotalFee);
	void generatePayId();
	void storePaymentDetails();
	void validate();
	~Payment();
	};
```

h)Feedback.h

```
class Feedback {

private:

	char FeedbackID[10];
	char CustomerName[50];
	int rating;
	char FeedbackDes[150];
	Customer* id;

public:

	Feedback();
	Feedback(const char ID[], const char name[], int rt, const char des[]);
	void display();
	void addID(Customer* id);
	~Feedback();

	};
```

i)Administrator.h

```
class Administrator
{
private:
	char adminId;
	char adminName;
	char adminEmail;
	int adminContactNumber;

public:
	Administrator();
	Administrator(char adminId[], char adminName[], char "adminEmail");
	void chackPaymentDetails();
	void updateFeedback();
	~Administrator
};
```

j)AdminLogin.h

```
class AdminLogin {

private:

	char AdminID[10];
	char username[50];
	char password[50];

public:

	AdminLogin();
	AdminLogin(const char id[], const char usrnm[], const char psswd[]);
	void checkLogin();
	~AdminLogin();

	};
```

2 .cpp files

a) Staff.cpp

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

staff::staff()
{
	StaffName = "";
	strcpy(StaffID, "");
	strcpy(S_NIC, "");
	strcpy(S_Email, "");
	S_ContactNumber = 0;

}

staff::staff(string Sname, const char SID[], const char SNIC[], const char
Smail[], int Scon)
{
	StaffName = Sname;
	strcpy(StaffID, SID);
	strcpy(S_NIC, SNIC);
	strcpy(S_Email, Smail);
	S_ContactNumber = Scon;
}

void staff::displayStaff()
{
	cout << "Staff  Nmae =" << StaffName << endl;
	cout << "Staff ID =" << StaffID << endl;
	cout << "Staff NIC =" << S_NIC << endl;
	cout << "Staff Email =" << S_Email << endl;
	cout << "Staff Contact Number =" << S_ContactNumber << endl;


}

void staff::AssignPhotographer()
{
	cout << "Photographer assigned" << endl;
}

staff :: ~staff()
{
	cout << "Destructor callled" << endl;
}
```

b) Photographer.cpp

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

photographer::photographer()
{
	strcpy(photog_Name, "");
	strcpy(photog_ID, "");
	strcpy(P_NIC, "");
	strcpy(P_Email, "");
	P_ContactNumber = 0;

}

photographer::photographer(const char Pname[], const char PID[], const char
PNIC[], const char Pmail[], int Pcon)
{
	strcpy(photog_Name, Pname);
	strcpy(photog_ID, PID);
	strcpy(P_NIC, PNIC);
	strcpy(P_Email, Pmail);
	P_ContactNumber = Pcon;

}

void photographer::displayPhotographer()
{
	cout << "Photographer Nmae =" << photog_Name << endl;
	cout << "Photographer ID =" << photog_ID << endl;
	cout << "Photographer NIC =" << P_NIC << endl;
	cout << "Photographer Email =" << P_Email << endl;
	cout << "Contact Number =" << P_ContactNumber << endl;
}

void photographer::respond()
{
	cout << "Respond sent" << endl;
}


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

c) Reservation.cpp

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

Reservation::Reservation()
{
	strcpy(resevationID, "");
	strcpy(reservationDate, "");
	strcpy(reservationTime, "");
	strcpy(package_type, "");
}
Reservation::Reservation(const char rID[], const char rdate[], const char
rtime[], const char packtype[])
{
	strcpy(resevationID, rID);
	strcpy(reservationDate, rdate);
	strcpy(reservationTime, rtime);
	strcpy(package_type, packtype);
}
void Reservation::setReservationDetails(const char rID[], const char rdate[],
const char rtime[], const char packtype[])
{
	strcpy(resevationID, rID);
	strcpy(reservationDate, rdate);
	strcpy(reservationTime, rtime);
	strcpy(package_type, packtype);
}
char Reservation::getReservationID()
{
	return reservationID;
}
void Reservation::printReservationDetails()
{
	cout << "Reservation ID:  " << resevationID << endl;
	cout << "Reservation date:  " << reservationDate << endl;
	cout << "Reservation time:  " << reservationTime << endl;
	cout << "Package type:  " << package_type << endl;
}
Reservation::~Reservation()
{
	cout << "Remove reservation";
}
```

d) Location.cpp

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

Location::Location()
{
	strcpy(locationID, "");
	strcpy(locationName, "");
	strcpy(locationAddress, "");
	contact_number = 0;
}
Location::Location(const char lID[], const char lname[], const char laddress[],
int lcontactno)
{
	strcpy(locationID, lID);
	strcpy(locationName, lname);
	strcpy(locationAddress, laddress);
	contact_number = lcontactno;
}
void Location::setLocationdetails(const char lID[], const char lname[], const
char laddress[], int lcontactno)
{
	strcpy(locationID, lID);
	strcpy(locationName, lname);
	strcpy(locationAddress, laddress);
	contact_number = lcontactno;
}
char Location::getLocationID()
{
	return locationID;
}
void Location::printLocationDetails()
{
	cout << "Location ID:  " << locationID << endl;
	cout << "Location name:  " << locationName << endl;
	cout << "Location address:  " << locationAddress << endl;
	cout << "Location contact number:  " << contact_number << endl;
}
Location::~Location()
{
	cout << "Location removed";
}
```

e) Customer.cpp

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


Customer::Customer()
```

```cpp
{

}
Customer::Customer()
{
	strcpy(cId, " ");
	strcpy(cName, " ");
	strcpy(cEmail, " ");
	customerMobileNumber = 0;
	customerNIC = 0;
}
Customer(char cId[], char cName[], char cEmail[])
{
	customerId = cId;
	customerName = cName;
	customerEmail = cemail;
}
void Customer::reserve()
{

}
void Customer::giveLocation()
{

}
void Customer::requestDeliveryItem()
{

}
void Customer::giveFeedback()
{

}
void Customer::payDeposit()
{

}
void Customer::payFullPayment()
{

}
Customer :: ~Customer()
{
	//Destructor

	}
```

f) Delivery_item.cpp

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

Delivery_items :: Delivery_items()
{

}
```

```
Delivery_items::Delivery_items()
{
	strcpy(deliveryId, " ");
	strcpy(orderDate, " ");
	deliveryDescription = 0;
}
Delivery_items(char dId(), char dDate(), string dDescription) {
	deliveryId = dId;
	orderDate = dDate;
	deliveryDescription = dDescription;
}
void Delivery_items::delivery()
{

}
void Delivery_items::add()
{

}
void Delivery_items::update()
{

}

Delivery_items :: ~Delivery_items()
{
	//Destructor

}
```

g) Payment.cpp

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

//.cpp
Payment::Payment()
{

}
Payment::Payment()
{
	strcpy(payId, " ");
	strcpy(paymentMethod, " ");
	strcpy(paymentDate, " ");
	advancedFee = 0.0;
	totalFee = 0.0;
}
Payment(char pId[], char pDate[], float ptotalFee)
{
	payId = pId;
	paymentDate = pDate;
	totalFee = ptotalFee;
}
void Payment::generatePayId()
```

```
{

}
void Payment::storePaymentDetails()
{

}
void Payment::validate()
{

}
Payment :: ~Payment()
{
	//Destructor
}
```

h) Feedback.cpp

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

using namespace std;

Feedback::Feedback() {

	strcpy(FeedbackID, "");
	strcpy(CustomerName, "");
	rating = 0;
	strcpy(FeedbackDes, "");

}

Feedback::Feedback(const char ID[], const char name[], int rt, const char des[])
{

	strcpy(FeedbackID, ID);
	strcpy(CustomerName, name);
	rating = rt;
	strcpy(FeedbackDes, des);
}

void ::Feedback::display() {

	cout << "FeedbackID   : " << FeedbackID << endl;
	cout << "Customer Name : " << CustomerName << endl;
	cout << "Rating : " << rating << endl;
	cout << "Feedback : " << FeedbackDes << endl;

}

void Feedback::addID(Customer* id) {


	id = new Customer()
}
```

```
Feedback :: ~Feedback() {

       cout << "Destructor runs" << endl;
}
```

i) Administrator.cpp

```
#include <iostream>
#include <cstring>
#include "Administrator.h"
using namespace std;
//.cpp file
Administrator::Administrator
{

}
Administrator::Administrator
{
       strcpy(adminId, " ");
       strcpy(adminName, " ");
       strcpy(adminEmail, " ");
       adminContactNumber = 0;
}
Administrator(char adminId[], char adminName[], char adminEmail[])
{
       adminId = aId;
       adminName = aName;
       adminEmail = aEmail;
       adminContactNumber = 0;
}
void Administrator::chackPaymentDetails()
{

}
void Administrator::updateFeedback()
{

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

j) AdminLogin.cpp

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

using namespace std;
```

```
AdminLogin::AdminLogin() {

	strcpy(AdminID, "");
	strcpy(username, "");
	strcpy(password, "");

}

AdminLogin::AdminLogin(const char id[], const char usrnm[], const char psswd[]) {

	strcpy(AdminID, id);
	strcpy(username, usrnm);
	strcpy(password, psswd);

}

void AdminLogin::checkLogin() {


}

AdminLogin :: ~AdminLogin() {

	cout << "Destructor runs" << endl;

}
```

3.Main.cpp

```
#include<iostream>
#include<cstring>

#include "Feedback.h"
#include "AdminLogin.h"
#include "Location.h"
#include "Reservation.h"
#include "Admin.h"
#include "Customer.h"
#include "Item_details.h"
#include "Payment.h"
#include "Photographer.h"
#include "Staff.h"


using namespace std;

int main() {

	Customer* c1 = new Customer("cs1234", "Kasun Kalhara",
"kasunkalhara@gmail.com");
	Reservation* r1 = new Reservation("RS1234", "2022/10/23", "10.00Am",
"Wedding");
	Location* l1 = new Location("LC1234", "Hilton Hotel", "Gall rd , Colombo",
0112324345);
	Photographer* ph1 = new Photographer("Janith Perera", "PH1234", "2000234v"
"janithperera@gmail.com", 0771231235);
	Staff* s1 = new Staff("Kusal Shanaka", "st1234", "123753v",
"kusalshanaka@gmail.com", 0753421134);
```

```
	Payment* p1 = new Payment("PY1234", "MASTER", "2022/05/24", 15000.00, 
75000.00);
	Delivery_items* d1 = new Delivery_items("dl1234", "2022/11/13", "album");
	Feedback* f1 = new Feedback("FB1234", "Kasun Kalhara", 8, "Very good, 
recommended!");
	Administrator* a1 = new Administrator("AD1234", "Sasindu Perera", 
"sasindu@gmail.com");
	AdminLogin* al1 = new AdminLogin("AD1234", "sasindu144438", 
"colombo1234@");

	delete c1, r1, l1, ph1, s1, p1, d1, f1, a1, al1;

	return 0;
}
```