SLIIT

Discover Your Future

Topic : Event Photography Management System

Group no : MLB_04.02_10

Campus : Malabe

Submission Date : 30.05.2021

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**
IT20297640	Piyasinghe H.K.N.Y	0705606156
IT20628390	Mudannayaka M.S.V.M	0762838883
IT20630416	Premathilaka.R.G.U.K	0750855526
IT20628468	Padukka P.D.T.H	0719444615
IT20606756	Senanayake P.M.	0779915276

Exercise 1

System requirements for Event Photography Management System

1. All visitors can looking for our services without registration.

2. If user want to get our services, they should register in our system. Providing details such as name, e-mail address, contact number to buy services.

3. After becoming a registered user, they can view their profiles, edit their profile or if they are not satisfied with our service they can delete their profiles.

4. Then member can view our events.

5. A member can select category/categories according to their requirements.

6. We provide three types of packages, they are basic, standard and premium. Then member can choose a suitable package of these.

7. When they having a trouble to make decisions they can contact our service agents.

8. Service agent, Photographer, Editor are the staff members our system.

9. After selecting a package, member should fill up a form including date and time, location of his/her event.

8. Member can contact the photographer through our service agent.

11. A member agree to get our service he/she can make payment online through our website via using credit cards or cash.

12. After the event, our photographer can provide raw photos. Otherwise we will be able to provide edited photos from our editors.

13. After getting his/her service, member can provide feedbacks through our website.

Noun & Verb Analysis

- Nouns in Red color.
- Verbs in Blue color.

1. All visitors can looking for our services without registration.

2. If user want to get our services, they should register in our system. Providing details such as name, e-mail address, contact number to buy services.

3. After becoming a registered user, they can view their profiles, edit their profile or if they are not satisfied with our service they can delete their profiles.

4. Then member can view our events.

5. A member can select category/categories according to their requirements.

6. We provide three types of packages, they are basic, standard and premium. Then member can choose a suitable package of these.

7. When they having a trouble to make decisions they can contact our service agents.

8. Service agent, Photographer, Editor are the staff members our system.

8. After selecting a package, member should fill up a form including date and time, location of his/her event.

9. Member can contact the photographer through our service agent.

10. A member agree to get our service he/she can make payment online through our website via using credit cards or cash.

11. After the event, our photographer can provide raw photos. Otherwise we will be able to provide edited photos from our editors.

12. After getting his/her service, member can provide feedbacks through our website.

Identified Classes using Noun Verb Analysis

Identified Classes:

- Member
- Staff
- Service Agent
- Event
- Editor
- Photographer
- Package
- Payment
- Feedback

Nouns:

- Visitor
- User
- Registered user
- Staff
- Service
- System
- Name
- Email address
- Contact number
- Member
- Profile
- Event
- Category
- Make decision
- Package
- Basic
- Standard
- Premium
- Service Agent
- Form
- Date and time
- Location
- Photographer
- Payment
- Website
- Credit Card
- Cash
- Raw photo
- Edited photo

- Editor
- Feedback

Reasons for Rejecting Other Nouns

1. Redundant:
   - In our system visitor, user, member refers to the same person as “Member”.
   - Event and category are refers to the same type as “Event”.

2. An event or an operation:
   - Make decision is the operation of members.

3. Outside scope of system:
   - System, website, profile, form are outside scope of the Photography system.

4. Meta language:
   - In a system members who are using system can call as registered user.

5. An attribute:
   - Name
   - Email Address
   - Contact number
   - Basic
   - Standard
   - Premium
   - Credit card
   - Cash
   - Date and time
   - Location
   - Edited photos
   - Raw photos

Exercise 2

<u>CRC Cards for Event Photography Management System</u>

Class Name: Member	
Responsibilities:	**Collaborations:**
Register and login entering details.	
View profile	
Set member details	
Edit profile	
Select event	Event
Choose a package	Package
Contact service agent	Service Agent
Provide details in a form	
Requires a photographer	Service Agent, Photographer
Delete profile	
Make payment	Payment
Get an edited photos	Editor
Give a feedback	Feedback

Class Name: Event	
Responsibilities:	**Collaborations:**
Set event details	
Update event details	Feedback
Get event details	
Add event	
Display previous event’s photos	
Generate booking date	

Class Name: Staff	
Responsibilities:	**Collaborations:**
Get staff details	

Class Name: Service Agent	
Responsibilities:	**Collaborations:**
Set agent details	
Get agent details	
Introduce a photographer	Photographer
Contact members	Member
Introduce event and package details	Event, Package

Class Name: Photographer	
Responsibilities:	**Collaborations:**
Set photographer details	
Get photographer details	
Contact members	Member, Service Agent
Accept the request	Member
Take photos	

Class Name: Editor	
Responsibilities:	**Collaborations:**
Set editor details	
Get editor details	
Contact members	Member, Service Agent
Edit photos	
Accept the request	Member

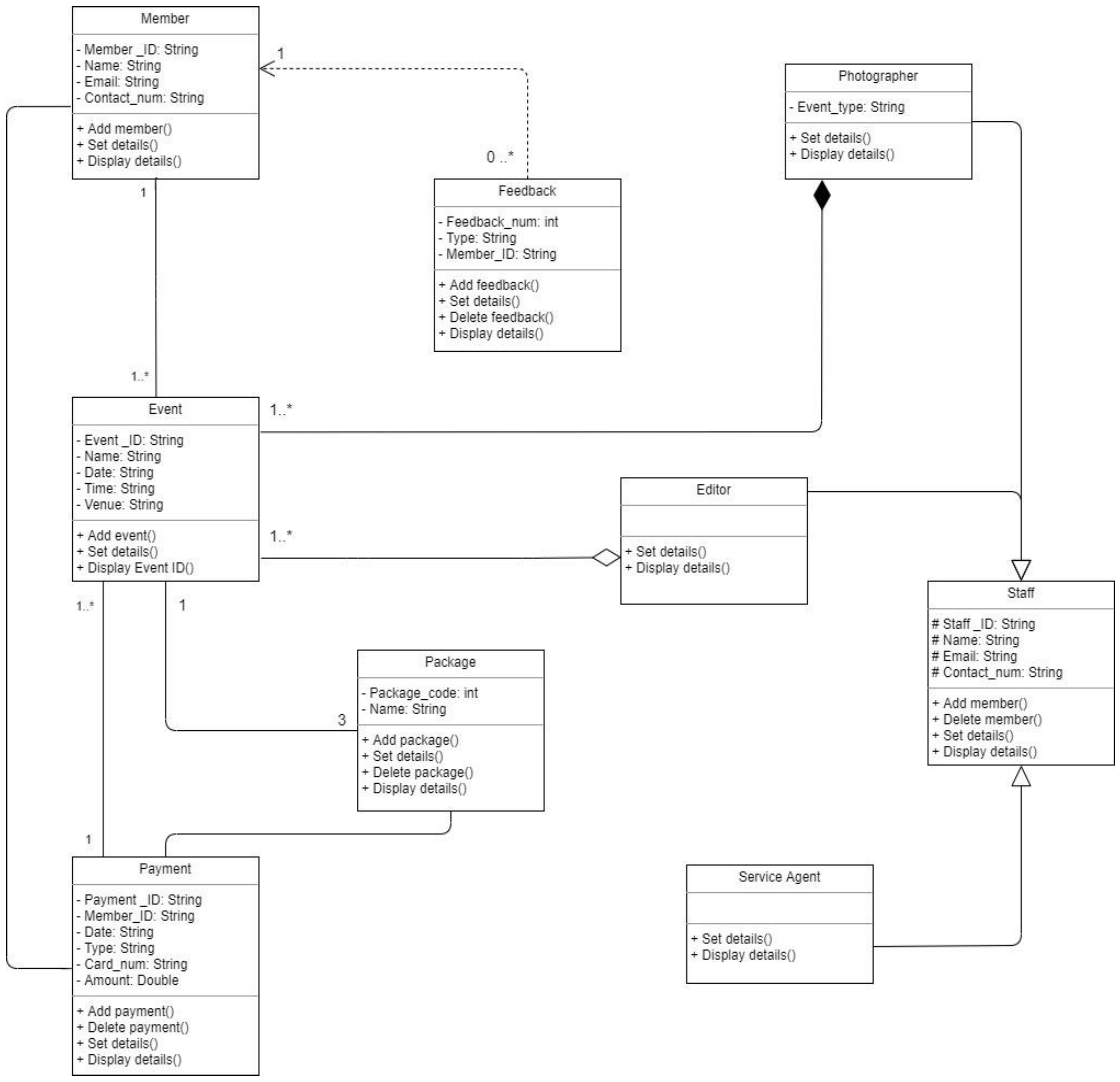
Class Name: Package	
Responsibilities:	**Collaborations:**
Set package details	
Update package details	Photographer
Add a new package	
Delete a package	

Class Name: Payment	
Responsibilities:	**Collaborations:**
Add payment	
Set payment details	
Get payment details	

Class Name: Feedback	
Responsibilities:	**Collaborations:**
Get feedback	
Set feedback details	
Delete a feedback	
An answer to feedback	Service Agent

Exercise 3

Class Diagram for the Event Photography Management System



- Assumption
    1. We made an assumption that event cant be held without a photographer.

Exercise 4

Coding for the Classes in Class Diagram

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

//CLASSES


//Staff class
class Staff{
  protected:
    string Staff_ID;
    string Name;
    string Email;
    string Contact_num;

  public:
    Staff();
    Staff(string StaffID);
    string getStaffID();
    void addMember(string StaffID, string name, string email, string Cnum);
    void displayDetails();
    void setDetails(string StaffID, string name, string email, string conNum);
    ~Staff();
};



//photographer
class Photographer : public Staff{
  protected:
    string Event_type;

  public:
  Photographer();
    Photographer(string Staff_ID) : Staff(Staff_ID){};
    void setDetails(string PStaff_ID, string PName, string PEmail, string
PCon_num);
    void displayDetails();
    ~Photographer();
};
```

```
//editor class
class Editor : public Staff{
  public:
  Editor();
    Editor(string Staff_ID) : Staff(Staff_ID){};
    void setDetails(string EStaff_ID, string EName, string EEmail, string
ECon_num);
    void displayDetails();
    ~Editor();
};

//event class
class Event
{
  private:
    string Event_ID;
    string Name;
    string Date;
    string Time;
    string Venue;

  public:
    Event();
    void setEventDetails(string EID, string Ename, string Edate, string Etime,
string Evenue);
    string getEventDetails();
    ~Event();
};

//member class
class Member{
  private:
    string Member_ID;
    string Name;
    string Email;
    string Contact_num;

  public:
    Member();
    void setMemberDetails(string MID, string Mname, string Memail, string
McNum);
    void displayMemberDetails();
    ~Member();
};
```

```
//payment class
class Payment
{
  private:
    string payment_ID;
    string member_ID;
    string date;
    string type;
    string card_num;
    double amount;

  public:
    Payment();
    void setPayment(string PayID, string MID, string Payd, string ty, string
PaycNum, double amnt);
    void displayDetails();
    ~Payment();
};
```

```
//Service agent class
class ServiceAgent : public Staff{
    public:
    ServiceAgent();
    ServiceAgent(string Staff_ID) : Staff(Staff_ID){};
    void setDetails(string AStaff_ID, string AName, string AEmail, string
ACon_num);
    void displayDetails();
    ~ServiceAgent();
};
```

```
//feedback class
 class Feedback{
   private:
   int num;
   char type[100];
   char Member_ID[100];
   public:
   Feedback();
   Feedback(int no,char tpy[],char id[]);
   ~Feedback();
   void setDetails();
   int getDetails();
 };
 //package class
 class Package{

     private:
     int code;
     char Name[];
     public:
     Package();
     Package(int c_id,char P_name[]);
     ~Package();
     void setDetails();
     int getDetails();

};
```

```cpp
//Cpp files


//Staff.cpp
Staff::Staff()
{

}
Staff::Staff(string Staff_ID){
  this->Staff_ID = Staff_ID;
  this->Name = "Not set";
  this->Email = "Not set";
  this->Contact_num = "Not set";
}
Staff::~Staff(){
  cout << "DELETED STAFF MEMBER!"<<Staff_ID<< endl;
}
void Staff::addMember(string StaffID, string name, string email, string Cnum){
  this->Staff_ID = StaffID;
  this->Name = name;
  this->Email = email;
  this->Contact_num = Cnum;
cout << "NEW STAFF MEMBER: " << Staff_ID << " ADDED SUCCESSFULLY!!" << endl;
}
string Staff::getStaffID(){
  return Staff_ID;
}
void Staff::displayDetails(){
  cout << "STAFF ID : " << Staff_ID << endl;
  cout << "NAME : " << Name << endl;
  cout << "EMAIL : " << Email << endl;
  cout << "CONTACT NUMBER : " << Contact_num << endl<<endl;
}
void Staff::setDetails(string StaffID, string name, string email, string
conNum){
  this->Staff_ID = StaffID;
  this->Name = name;
  this->Email = email;
  this->Contact_num = conNum;
}
```

```
//Editor cpp
Editor::Editor()
{}
void Editor::setDetails(string EStaff_ID, string EName, string EEmail, string
ECon_num){
  this->Staff_ID = EStaff_ID;
  this->Name = EName;
  this->Email = EEmail;
  this->Contact_num = ECon_num;
}
void Editor::displayDetails(){
  cout << "STAFF ID : " << Staff_ID << endl;
  cout << "NAME : " << Name << endl;
  cout << "E-MAIL : " << Email << endl;
  cout << "CONTACT NO : " << Contact_num << endl<<endl<<endl;
}
Editor::~Editor(){
  cout<<"EDITOR DESTRUCTOR IS RUNNING!"<<endl;
}


//event cpp

Event::Event(){

}
void Event::setEventDetails(string EID, string Ename, string Edate, string
Etime, string Evenue){
  Event_ID = EID;
  Name = Ename;
  Date = Edate;
  Time = Etime;
  Venue = Evenue;
}
string Event::getEventDetails(){
  return Event_ID;
}
Event::~Event()
{
  cout<<"EVENT DESTRUCTOR IS RUNNING!"<<endl;
}
```

```cpp
//member.cpp
Member::Member(){

}
void Member::setMemberDetails(string MID, string Mname, string Memail, string
McNum){
  Member_ID = MID;
  Name = Mname;
  Email = Memail;
  Contact_num = McNum;
}
void Member::displayMemberDetails(){
   cout<<"MEMBER ID: "<< Member_ID<<endl;
   cout<<"NAME: "<<Name<<endl;
   cout<<"EMAIL: "<<Email<<endl;
   cout<<"CONTACT NO: "<<Contact_num<<endl<<endl;

}
Member::~Member()
{
  cout<<"MEMBER DESTRUCTOR IS RUNNING!"<<endl;
}


//payment cpp
Payment::Payment(){

}
void Payment::setPayment(string PayID, string MID, string Payd, string ty,
string PaycNum, double amnt){
    payment_ID = PayID;
    member_ID = MID;
    date = Payd;
    type = ty;
    card_num = PaycNum;
    amount = amnt;
}
void Payment::displayDetails(){
    cout << "PAYMENT ID: " << payment_ID << endl;
    cout << "MEMBER ID: " << member_ID << endl;
    cout << "DATE: " << date << endl;
    cout << "TYPE: " << type << endl;
    cout << "CARD NUMBER: " << card_num << endl;
    cout << "AMOUNT: " << amount << endl<<endl;
}
Payment::~Payment()
{
  cout << "PAYMENT DESTRUCOR IS RUNNING!" << endl;
}
```

```
//ServiceAgent cpp


ServiceAgent::ServiceAgent(){}
void ServiceAgent::setDetails(string AStaff_ID, string AName, string AEmail,
string ACon_num){
  this->Staff_ID = AStaff_ID;
  this->Name = AName;
  this->Email = AEmail;
  this->Contact_num = ACon_num;
}
void ServiceAgent::displayDetails(){
  cout << "STAFF ID : " << Staff_ID << endl;
  cout << "NAME : " << Name << endl;
  cout << "E-MAIL : " << Email << endl;
  cout << "CONTACT NUMBER : " << Contact_num << endl<<endl;
}
ServiceAgent::~ServiceAgent()
{
 cout<<"SERVICE AGENT DESTRUCTOR IS RUNNING!"<<endl;
}


//photograper cpp
Photographer::Photographer()
{

}
Photographer:: ~Photographer(){
 cout<<"PHOTOGRAPHER DESTRUCORT IS RUNNING!"<<endl;
}
void Photographer::setDetails(string PStaff_ID, string PName, string PEmail,
string PCon_num){
  this->Staff_ID = PStaff_ID;
  this->Name = PName;
  this->Email = PEmail;
  this->Contact_num = PCon_num;
}
void Photographer::displayDetails(){
  cout << "STAFF ID : " << Staff_ID << endl;
  cout << "NAME : " << Name << endl;
  cout << "E-MAIL : " << Email << endl;
  cout << "CONTACT NO : " << Contact_num<<endl<<endl;
}
```

```cpp
//feedback.cpp
Feedback::Feedback()
{
  num=0;
}
Feedback::Feedback(int no,char tpy[],char id[])
{
  num=no;
  strcpy(type,tpy);
  strcpy(Member_ID,id);
}
void Feedback::setDetails()
{
  cout<<"ENTER THE FEEDBACK ID: ";
  cin>>num;
  cout<<"ENTER THE TYPE OF FEEDBACK: ";
  cin>>type;
  cout<<"ENTER THE MEMBER ID: ";
  cin>>Member_ID;
}

int Feedback::getDetails()
{
  cout<<"FEEDBACK_ID: "<<num<<endl;
  cout<<"TYPE OF FEEDBACK "<<type<<endl;
  cout<<"MEMBER ID: "<<Member_ID<<endl<<endl;
  return 0;
}

Feedback::~Feedback()
{
  cout<<"FEEDBACK DESTRUCTOR IS RUNNING!"<<endl;
}
```

```cpp
//package.cpp
Package::Package()
{
  code=0;
}
Package::Package(int c_id,char P_name[])
{
  code=c_id;
  strcpy(Name,P_name);
}

void Package::setDetails()
{
  cout<<"ENTER THE PACKAGE CODE: ";
  cin>>code;
  cout<<"ENTER THE PACKAGE NAME: ";
  cin>>Name;

}
int Package::getDetails()
{
  cout<<"PACKAGE CODE IS: "<< code<<endl;
  cout<<"PACKAGE NAME IS: "<< Name<<endl<<endl;
  return 0;

}

Package::~Package()
{
  cout<<"PACKAGE DESTRUTOR IS RUNNING!"<<endl;
}
```

```
//main function
int main()
{
  Staff *S1= new Staff();
  Feedback *F1=new Feedback();
  Package *P1 = new Package();
  Payment *PA1= new Payment();
  Member *M1=new Member();
  Event *E1=new Event();
 ServiceAgent *SA1= new ServiceAgent();
 Photographer *Ph1= new Photographer();
 Editor *Ed1= new Editor();


  F1->setDetails();
  F1->getDetails();
  P1->setDetails();
  P1->getDetails();
  S1->setDetails("SID20637489","Navodya","navodya@gmail.com","0763656444");
  S1->displayDetails();
  PA1-
>setPayment("PID2062","ID20637489","2020/03/23","Cash","30763647775",500000.00
);
  PA1->displayDetails();
  M1-
>setMemberDetails("ID20456567","Kithmin","kithmin@gmail.com","0783456342");
  M1->displayMemberDetails();
  E1->setEventDetails("EID1212","Wedding","2022/12/22","00:34:00
a.m","SLIIT");
  E1->getEventDetails();
  SA1->setDetails("SID20634569","Pasindu","pasindu@gmail.com","0752341112");
  SA1->displayDetails();
  Ph1->setDetails("PHID20634534","Sachin","sachin@gmail.com","0783647774");
  Ph1->displayDetails();
  Ed1->setDetails("EDID234512356","Thejana","thejana@gmail.com","0704344441");
  Ed1->displayDetails();

delete F1;
delete P1;
delete PA1;
delete M1;
delete E1;
delete SA1;
delete Ph1;
delete Ed1;
}
```