Sri Lanka Institute of Information Technology

B sc.(Hons) in Information Technology

Object Oriented Concepts (IT1050)

Online photo editing system

<u>Group Assignment 2 -2019</u>

MLB_Group_7.1_4

Table of Content

1.Team members & details

Name	Registration No.
Rathnayaka R.M.M.P	IT19152974
Fernando K.D.A.B	IT19146898
Fernando B.D.C	IT19150680
Sathsarani M.W.A.R	IT19151120
Dharmarathne A.M.J.L.R	IT19145426

2.Use case scenarios

Assumptions- *we have assumed that user must register to the system to do the tasks of the system. So, we have written use case scenarios assuming that they have already registered to the system. If user wants to edit the photo user should send it to the system editor and pay for the editor. Payments will add to editor's account as editor's salary.*

Use case number: 01

Use case name: Upload Photo to system

Primary Actor: Registered user

Main flow:

1. User logs in to the system using user name and password of the account
2. System show menu
3. User select upload photo
4. User browse photo location
5. User select photo for upload
6. User enter editing details
7. User press on upload button
8. System show successful massage

Use case number: 02

Use case name: Edit the photo by editor

Primary Actor: Editor

Main flow:

1. Editor login to the system using editor's user name and password of account
2. System shows menu
3. Editor press on uploaded photos
4. System show the uploaded photos
5. Editor select photo
6. Editor press on send
7. System send photo and details to editor
8. Editor edit the photo according to the editing details
9. Editor upload the edited photo
10. System send the edited photo to the user

Use case number: 03

Use case name: Download photos

Primary Actor: Registered User

Main flow:

1. User login to the system
2. System show menu
3. User select gallery
4. System show gallery
5. User view photos and select photos to download
6. System ask to pay for download photos
7. User enter bank details
8. System check details and ask to confirm
9. User confirms it
10. System release photos to download
11. User download photos

Use case number: 04

Use case name: Buy a package

Primary Actor: Registered User

Main flow:

1. User login to system
2. System show menu
3. User clicks on packages
4. System show packages and details
5. User check packages
6. User select packages
7. System ask to add to cart
8. User add it to cart
9. System add package to user's cart
10. User goes to cart and press on buy
11. System show window to enter bank details
12. User enter bank details
13. System verify details and ask to confirm
14. User confirm payment
15. System show successful message

Use case number: 05

Use case name: Inquire information

Primary Actor: User

Main flow:

1. User login to system
2. System show menu
3. User selects on inquiry
4. System show inquiry page
5. User enter inquiry
6. User send inquiry
7. System show successful message

Use case number:06

Use case name: Delete photos

Primary Actor: Registered User

Main flow:

1. User login to system
2. System show menu
3. User select gallery
4. System show gallery
5. User view photos and select photos to delete
6. System delete photos
7. System send successful message

Use case number: 07

Use case name: Get salary

Primary Actor: Editor

Main flow:

1. Editor login to system
2. System show menu
3. Editor select on view account
4. System show account
5. Editor press on get salary
6. System send salary to editor’s bank account

Use case number: 08

Use case name: Update system details

Primary Actor: Admin

Main flow:

1. Admin login to system by using admin’s username and password
2. System show menu
3. Admin select update details
4. System show details of the system
5. Admin edit details and press on save
6. System save edited details
7. System update details

Use case number: 09

Use case name: Answer inquiries

Primary Actor: Admin

Main flow:

1. Admin login to system
2. System show menu
3. Admin go to inquiries section
4. System displays remaining inquiries to answer
5. Admin selects an inquiry
6. Admin writes answer to inquiry
7. Admin sends after writing
8. System send the answer as an e-mail
9. Admin go back to inquiries section and continue

Use case number: 10

Use case name: Send a feedback

Primary Actor: Registered User

Main flow:

1. User login to system
2. System show menu
3. User select feedback
4. System show feedback form
5. User enter feedback
6. System get it and check editor of that photo
7. System send that feedback to editor
8. Editor check notification
9. Editor check the feedback
10. Editor replies to the feedback
11. System send that reply to user

3.Noun verb analysis

Identifying nouns are highlighted in yellow color and verbs are in blue color.

Use case number: 01

Use case name: Upload Photo to the system

Primary Actor: Registered user

Main flow:

1. User logs in to the system using user name and password of the account
2. System show menu
3. User select upload photo
4. User browse photo location
5. User select photo to upload
6. User enter editing details
7. User press on upload
8. System show successful message

Use case number: 02

Use case name: Edit the photo by editor

Primary Actor: Editor

Main flow:

1. Editor login to the system using editor's user name and password of account
2. System shows menu
3. Editor press on uploaded photos
4. System show the uploaded photos
5. Editor select photo
6. Editor press on send
7. System send photo and details to editor
8. Editor edit the photo according to the editing details
9. Editor upload the edited photo
10. System send the edited photo to the user

Use case number: 03

Use case name: Download photos

Primary Actor: Registered User

Main flow:

1. User login to the system
2. System show menu
3. User select gallery
4. System show gallery
5. User view photos and select photos to download
6. System ask to pay for download photos
7. User enter bank details
8. System validate details and ask to confirm
9. User confirms it
10. System release photos to download
11. User download photos

Use case number: 04

Use case name: Buy a package

Primary Actor: Registered User

Main flow:

1. User login to system
2. System show menu
3. User clicks on packages
4. System show packages and details
5. User check packages
6. User select packages
7. User press on buy
8. System show window to enter bank details
9. User enter bank details
10. System verify details and ask to confirm
11. User confirm payment
12. System show successful message

Use case number: 05

Use case name: Inquire information

Primary Actor: Registered User

Main flow:

1. User login to system
2. System show menu
3. User selects on inquiry
4. System show inquiry page
5. User enter inquiry
6. User send inquiry
7. System show successful message

Use case number:06

Use case name: Delete photos

Primary Actor: Registered User

Main flow:

1. User login to system
2. System show menu
3. User select gallery
4. System show gallery
5. User view photos and select photos to delete
6. System delete photos
7. System send successful message

Use case number: 07

Use case name: Get salary

Primary Actor: Editor

Main flow:

1. Editor login to system
2. System show menu
3. Editor select on view account
4. System show account
5. Editor press on get salary
6. System send salary to editor’s bank account

Use case number: 08

Use case name: Update system details

Primary Actor: Admin

Main flow:

1. Admin login to system by using admin’s username and password
2. System show menu
3. Admin select update details
4. System display details of the system
5. Admin edit details and press on save
6. System save edited details
7. System update details

Use case number: 09

Use case name: Answer inquiries

Primary Actor: Admin

Main flow:

1. Admin login to system
2. System show menu
3. Admin go to inquiries section
4. System displays remaining inquiries to answer
5. Admin selects an inquiry
6. Admin writes answer to inquiry
7. Admin sends after writing
8. System send the answer as an e-mail
9. Admin go back to inquiries section and continue

Use case number: 10

Use case name: Send a feedback

Primary Actor: Registered User

Main flow:

1. User login to system
2. System show menu
3. User select feedback
4. System show feedback form
5. User enter feedback
6. System get it and check editor of that photo
7. System send that feedback to editor
8. Editor check message
9. Editor view the feedback
10. Editor replies to the feedback
11. System send that reply to user

4.Nouns

Redundancy	**Outside scope**	**Attributes**
User	System	User name
Photo	Admin	Password
Editor	Bank account	Photo location
Registered user	Menu	Salary
Uploaded photos	Message	Email
Account	Window	
Gallery		
Packages		
Information		
Inquiry		
Details		
Answers		
Feedback		
Feedback form		

Classes

- Registered user
- Photo
- Edited photo
- Editor
- Package
- Payment
- Inquiry
- Feedback

5.CRC Cards & UML Class Diagram

Class Name: Registered User	
Responsibilities:	**Collaborations:**
Store and display user details	
Upload photo	Photo
Insert editing and bank account details	
Make payment to download photos	Payment
Buy package	Package, payment

Class Name: Photo	
Responsibilities:	**Collaborations:**
Store photo details	
Display photo details	
Download photos	
Delete photos	

Class Name: Editor	
Responsibilities:	**Collaborations:**
Store and display editor’s details	
Get details of the photos that need to be edited	photo
Edit the photo	photo
Upload the edited photo	Edited photo
Get the salary	
Reply to the feedbacks	Feedback

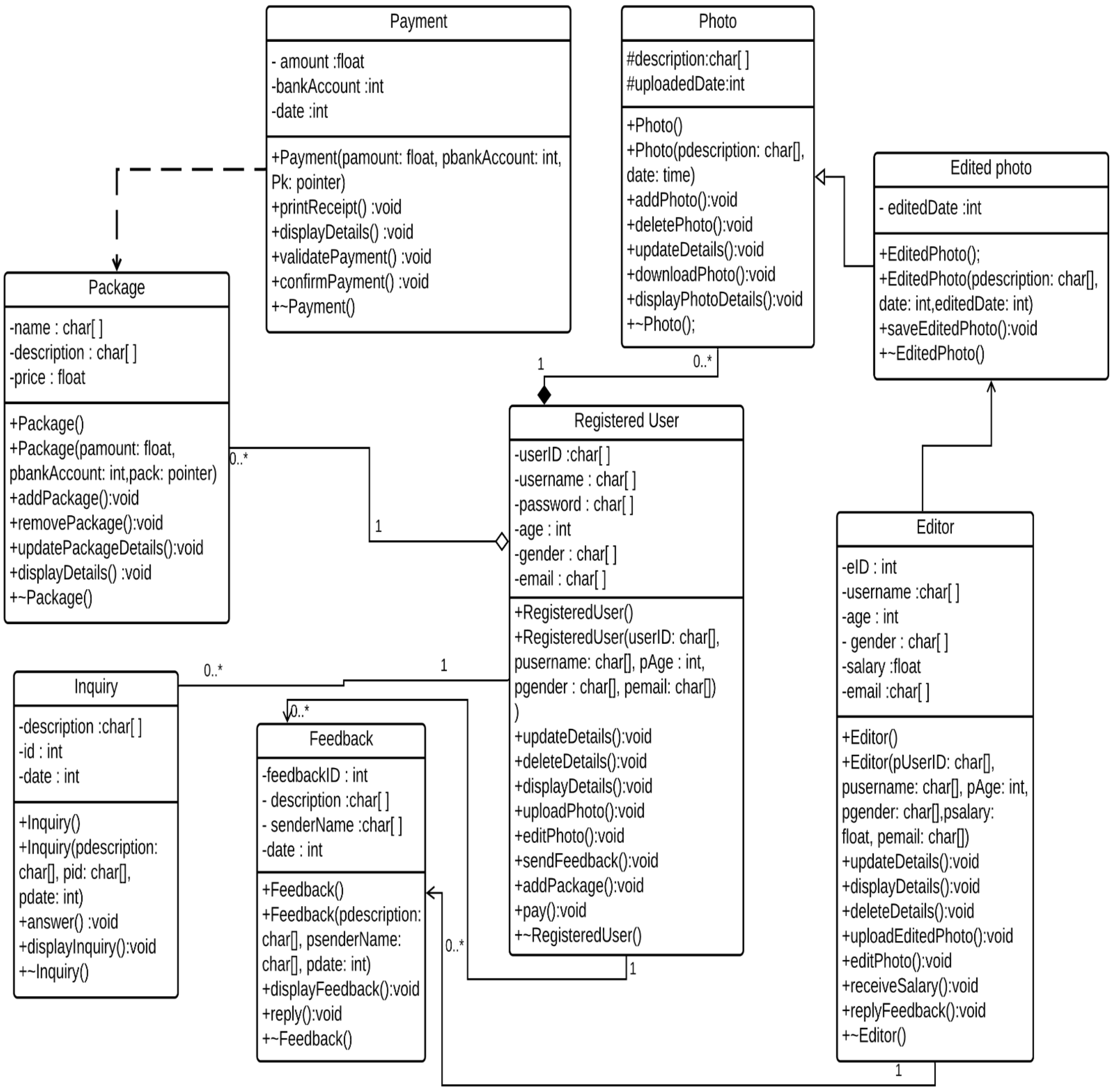
Class Name: Package	
Responsibilities:	**Collaborations:**
Store package details	
Display package details	
Install package to user’s account	Registered user

Class Name: Edited Photo	
Responsibilities:	**Collaborations:**
Store edited photo details	
Display edited photo details	
Download edited photo	
Delete edited photo	
Save edited photo to user’s account after payment	Registered user

Class Name: Inquiry	
Responsibilities:	**Collaborations:**
Store user's inquiries	Registered User
Display inquiries and details	
Send Inquiries to the admin	
Store admin's replies to inquiry	
Send admin's answer to the user	

Class Name: Payment	
Responsibilities:	**Collaborations:**
Store payment details	
Display payment methods	
Validate bank details	
Confirm payment details	Registered user
Print online payment receipt after completing payment	

Class Name: Feedback	
Responsibilities:	**Collaborations:**
Store user’s feedbacks	Registered user
Display user’s feedbacks	
Send the feedback to the editor	
Store editor’s replies to the feedback	Editor
Send editor’s reply to the user	


Payment
- amount :float
-bankAccount :int
-date :int
+Payment(pamount: float, pbankAccount: int, Pk: pointer)
+printReceipt() :void
+displayDetails() :void
+validatePayment() :void
+confirmPayment() :void
+~Payment()
Photo
#description:char[ ]
#uploadedDate:int
+Photo()
+Photo(pdescription: char[], date: time)
+addPhoto():void
+deletePhoto():void
+updateDetails():void
+downloadPhoto():void
+displayPhotoDetails():void
+~Photo();
Edited photo
- editedDate :int
+EditedPhoto();
+EditedPhoto(pdescription: char[], date: int,editedDate: int)
+saveEditedPhoto():void
+~EditedPhoto()
Package
-name : char[ ]
-description : char[ ]
-price : float
+Package()
+Package(pamount: float, pbankAccount: int,pack: pointer)
+addPackage():void
+removePackage():void
+updatePackageDetails():void
+displayDetails() :void
+~Package()
Registered User
-userID :char[ ]
-username : char[ ]
-password : char[ ]
-age : int
-gender : char[ ]
-email : char[ ]
+RegisteredUser()
+RegisteredUser(userID: char[], pusername: char[], pAge : int, pgender : char[], pemail: char[])
)
+updateDetails():void
+deleteDetails():void
+displayDetails():void
+uploadPhoto():void
+editPhoto():void
+sendFeedback():void
+addPackage():void
+pay():void
+~RegisteredUser()
Editor
-eID : int
-username :char[ ]
-age : int
- gender : char[ ]
-salary :float
-email :char[ ]
+Editor()
+Editor(pUserID: char[], pusername: char[], pAge: int, pgender: char[],psalary: float, pemail: char[])
+updateDetails():void
+displayDetails():void
+deleteDetails():void
+uploadEditedPhoto():void
+editPhoto():void
+receiveSalary():void
+replyFeedback():void
+~Editor()
Inquiry
-description :char[ ]
-id : int
-date : int
+Inquiry()
+Inquiry(pdescription: char[], pid: char[], pdate: int)
+answer() :void
+displayInquiry():void
+~Inquiry()
Feedback
-feedbackID : int
- description :char[ ]
- senderName :char[ ]
-date : int
+Feedback()
+Feedback(pdescription: char[], psenderName: char[], pdate: int)
+displayFeedback():void
+reply():void
+~Feedback()
1
0..*
0..*
1
0..*
1
0..*
0..*
1
1

6.Coding for the identified classes

```
class EditedPhoto : public Photo
{
private:
	int editedDate;

public:
	EditedPhoto();
	EditedPhoto(const char pdescription[], int date, int editedDate) : 
Photo(pdescription, date) {};
	void saveEditedPhoto();
	~EditedPhoto();
};

#include "Feedback.h"
#include "EditedPhoto.h"
class Editor
{
private:
	char userID[20];
	char username[50];
	int age;
	char gender[10];
	float salary;
	char email[100];
	EditedPhoto* ep[100];
	Feedback* fd[100];

public:
	Editor();
	Editor(const char pUserID[], const char pusername[], int pAge, const char 
pgender[],float psalary,const char pemail[]);
	void displayDetail();
	void updateDetails();
	void uploadEditedPhoto(int i,EditedPhoto *ph);
	void displayFeedback(int i,Feedback* feed);
	void editPhoto();
	void recieveSalary();
	void replyFeedback();
	~Editor();
};

class Feedback
{
	private:
		char description[200];
		char senderName[50];
		int date;

	public:
		Feedback();
		Feedback(const char pdescription[], const char psenderName[], int pdate);
		void displayFeedback();
		void reply();
		~Feedback();
};
```

```
class Inquiry
{
private:
              char description[200];
              char id[20];
              int date;

public:
       Inquiry();
       Inquiry(const char pdescription[],const char pid[],int pdate);
       void answer();
       void displayInquiry();
       ~Inquiry();

};

class Package
{
       private:
              char name[20];
              char description[200];
              float price;

       public:
              Package();
              Package(const char pname[],const char pdescription[],float pprice);
              void displayDetails();
              void buyPackage();
              ~Package();

};

#include "Package.h"
class Payment
{
       private:
              float amount;
              int bankAccount;
              int date;
              Package* pk;

       public:
              Payment();
              Payment(float pamount, int pbankAccount, int pdate,Package *pack);
              void printReceipt();
              void displayDetails();
              void validatePayment();
              void confirmPayment();
              ~Payment();
};
```

```
class Photo
{
protected:
	char description[200];
	int uploadedDate;

public:
	Photo();
	Photo(const char pdescription[], int date);
	void updateDetails();
	void displayPhoto();
	void editPhoto();
	void download();
	~Photo();
};

#include "Photo.h"
#include "Feedback.h"
#include "Package.h"
#include "Inquiry.h"

class RegisteredUser
{
private:
	char userID[20];
	char username[50];
	int age;
	char gender[10];
	char email[50];
	Photo* ph[100];
	Feedback* fd[100];
	Package* pk[100];
	Inquiry* in;

public:
	RegisteredUser();
	RegisteredUser(const char pUserID[],const char pusername[],int pname,const char 
pgender[],const char pemail[]);
	void displayDetails();
	void editDetails();
	void deleteDetails();
	void uploadPhoto(int i,Photo *photo);
	void editPhoto();
	void pay();
	void sendFeedback(int i,Feedback *feed);
	void sendInquiry(Inquiry *inq);
	void addPackage(int i, Package* pack);
	~RegisteredUser();
};
```

```cpp
#include "EditedPhoto.h"
#include <iostream>
#include <cstring>

using namespace std;

/*EditedPhoto::EditedPhoto(const char pdescription[], int date, int peditedDate)
{
	strcpy_s(description,pdescription);
	uploadedDate = date;
	peditedDate = editedDate;
}*/

EditedPhoto::EditedPhoto()
{
}

void EditedPhoto::saveEditedPhoto()
{
}

EditedPhoto::~EditedPhoto()
{
}

#include "Editor.h"
#include <iostream>
#include <cstring>

using namespace std;


Editor::Editor()
{
}

Editor::Editor(const char pUserID[], const char pusername[], int pAge, const char
pgender[], float psalary, const char pemail[])
{
	strcpy_s(userID,pUserID);
	strcpy_s(username,pusername);
	age = pAge;
	strcpy_s(gender,pgender);
	salary = psalary;
	strcpy_s(email, pemail);
}

void Editor::displayDetail()
{
}

void Editor::updateDetails()
{
}

void Editor::uploadEditedPhoto(int i,EditedPhoto *ph)
{
	ep[i] = ph;
	cout << "Photo uploaded successfully" << endl;
}

void Editor::displayFeedback(int i, Feedback* feed)
{
}

void Editor::editPhoto()
{
}

void Editor::recieveSalary()
{
}

void Editor::replyFeedback()
{
}

Editor::~Editor()
{
	cout << "Destructor runs for Editor class" << endl;
}

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

using namespace std;

Feedback::Feedback()
{
}

Feedback::Feedback(const char pdescription[], const char psenderName[], int pdate)
{
	strcpy_s(description, pdescription);
	strcpy_s(senderName, psenderName);
	date = pdate;
	cout << "Constructor running for Feedback class" << endl;

}

void Feedback::displayFeedback()
{

}

void Feedback::reply()
{
}

Feedback::~Feedback()
{
	cout << "Desctructor running for Feedback class" << endl;

}
```

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

using namespace std;

Inquiry::Inquiry()
{
}

Inquiry::Inquiry(const char pdescription[], const char pid[], int pdate)
{
	strcpy_s(description,pdescription);
	strcpy_s(id, pid);
	date = pdate;

}

void Inquiry::answer()
{
}

void Inquiry::displayInquiry()
{
}

Inquiry::~Inquiry()
{
	cout << "Destructor running for Inquiry class" << endl;
}

#include "Package.h"
#include <iostream>
#include <cstring>

using namespace std;

Package::Package()
{
}

Package::Package(const char pname[], const char pdescription[], float pprice)
{
	strcpy_s(name,pname);
	strcpy_s(description,pdescription);
	price = pprice;

	cout << "Constructor running for Package class" << endl;
}

void Package::displayDetails()
{
}

void Package::buyPackage()
{
}
```

```
Package::~Package()
{
	cout << "Destructor running for package class" << endl;
}

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

Payment::Payment()
{
}

Payment::Payment(float pamount, int pbankAccount, int pdate,Package * pack)
{
	amount = pamount;
	bankAccount = pbankAccount;
	date = pdate;
	pk = pack;

	cout << "Constructor running for Payment class" << endl;;
}

void Payment::printReceipt()
{
}

void Payment::displayDetails()
{
}

void Payment::validatePayment()
{
}

void Payment::confirmPayment()
{
}

Payment::~Payment()
{
	cout << "Deconstructor running for Payment class" << endl;
}

#include "Photo.h"
#include <iostream>
#include <cstring>

using namespace std;


Photo::Photo()
{
}

Photo::Photo(const char pdescription[], int date)
```

```
{
        strcpy_s(description, pdescription);
        uploadedDate = date;

        cout << "Constructor running for Photo class" << endl;
}

void Photo::updateDetails()
{
}

void Photo::displayPhoto()
{
}

void Photo::editPhoto()
{
}

void Photo::download()
{
}

Photo::~Photo()
{
        cout << "Destructor running for Photo class" << endl;
}

#include "RegisteredUser.h"
#include <iostream>
#include <cstring>

using namespace std;

RegisteredUser::RegisteredUser()
{
}

RegisteredUser::RegisteredUser(const char pUserID[], const char pusername[], int pAge,
const char pgender[], const char pemail[])
{
        strcpy_s(userID, pUserID);
        strcpy_s(username, pusername);
        age = pAge;
        strcpy_s(gender,pgender);
        strcpy_s(email, pemail);

        cout << "Constructor running for Registered User class" << endl;
}

void RegisteredUser::displayDetails()
{
}

void RegisteredUser::editDetails()
{
}
```

```
void RegisteredUser::deleteDetails()
{
}

void RegisteredUser::uploadPhoto(int i, Photo* photo)
{
	ph[i] = photo;
	cout << "Photo Uploaded successfully" << endl;
}

void RegisteredUser::editPhoto()
{
}

void RegisteredUser::pay()
{
}

void RegisteredUser::sendFeedback(int i,Feedback* feed)
{
	fd[i] = feed;
	cout << "Feedback sended successfully" << endl;
}

void RegisteredUser::sendInquiry(Inquiry* inq)
{
	in = inq;
}

void RegisteredUser::addPackage(int i, Package* pack)
{
	pk[i] = pack;
	cout << "Package added successfully" << endl;
}

RegisteredUser::~RegisteredUser()
{
	cout << "Deconstructor running for Registered User class" << endl;
}
```

Main program

```cpp
#include <iostream>
#include "RegisteredUser.h"
#include "Photo.h"
#include "Feedback.h"
#include "Package.h"
#include "Editor.h"
#include "Payment.h"
#include "Inquiry.h"
#include "EditedPhoto.h"

using namespace std;

int main() {
	RegisteredUser *reg;
	reg = new RegisteredUser("123", "hello world", 12, "male","name@mail.com");

	Photo* ph;
	ph = new Photo("Hello",12);

	Feedback* fd;
	fd = new Feedback("Hello World","Malindu",12);

	Package* pk;
	pk = new Package("package 1", "Hello World", (float)19.99);

	Payment* py;
	py = new Payment((float)19.99, 99112345, 1231, pk);

	Editor* ed;
	ed = new Editor("1234", "Steve", 28, "Male", (float)50.00, "name@mail.com");

	EditedPhoto* edtp;
	edtp = new EditedPhoto("Hello world", 0425, 0501);

	Inquiry* inq;
	inq = new Inquiry("hello world", "1234", 0115);


	//reg user methods
	reg->addPackage(1,pk);
	reg->deleteDetails();
	reg->displayDetails();
	reg->editDetails();
	reg->pay();
	reg->sendFeedback(1,fd);
	reg->sendInquiry(inq);
	reg->uploadPhoto(1,ph);


	//photos methods
	ph->displayPhoto();
	ph->download();
	ph->editPhoto();
	ph->updateDetails();
```

```
        //feedback methods
        fd->displayFeedback();
        fd->reply();

        //package methods
        pk->buyPackage();
        pk->displayDetails();

        //payment methods
        py->confirmPayment();
        py->displayDetails();
        py->printReceipt();
        py->validatePayment();

        //editor methods
        ed->displayDetail();
        ed->displayFeedback(1,fd);
        ed->editPhoto();
        ed->recieveSalary();
        ed->replyFeedback();
        ed->updateDetails();
        ed->uploadEditedPhoto(1,edtp);

        //edited photo method
        edtp->displayPhoto();
        edtp->download();
        edtp->editPhoto();
        edtp->saveEditedPhoto();
        edtp->updateDetails();

        //Inquiry methods
        inq->answer();
        inq->displayInquiry();

        delete reg;
        delete ph;
        delete fd;
        delete pk;
        delete py;
        delete ed;
        delete edtp;
        delete inq;
}
```