Topic : **Online Helpdesk for University Students**

Group no : **MLB_10.01_03**

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**
IT21181474	Hettiarachchi H.K.Y.K.	0705518090
IT21182914	Arachchi D.S.U.	0776048468
IT21184758	Liyanage P.P.	0778530266
IT21188572	Fernando S.D.K.	0713311649
IT21184130	Vidyani H.A.L.	0703265260

Table of Contents

Exercise 1:

USER REQUIREMENTS

1. In this online helpdesk system has many faculties.
2. One faculty consists of four types of users: student, lecturer, faculty-head, and guest.
3. Guest in the online helpdesk needs to first register providing related details (Example: name, E-mail address).
4. After register to the system, users can modify their accounts anytime.
5. A lecturer and a faculty-head publish new notices to the system and remove past notices from the system according to the degree.
6. A student must always check notices related to the degree.
7. A student can ask questions via FAQ.
8. A lecturer can answer FAQ questions easily.
9. A student can give ratings to the FAQ questions whether he/she satisfied or not.
10. Administrator and faculty-head should manage faculties.
11. The administrator helps the users with solving bugs and issues of the system.
12. The administrator updates the system after solving bugs.

(Noun – Verb)

Identified Classes using Noun-Verb Analysis

Identified Classes

- Guest
- Student
- Lecturer
- Faculty-head
- Notices
- FAQ
- Faculty
- Administrator

CRC CARDS FOR ONLINE HELPDESK

Guest	
Responsibilities:	**Collaborations:**
Register to the system	
Modify accounts	

Student	
Responsibilities:	**Collaborations:**
Register to the system	
Modify accounts	
Check notices	Notices
Ask questions	FAQ
Give ratings	FAQ

Lecturer	
Responsibilities:	**Collaborations:**
Register to the system	
Modify accounts	
Publish new notices	Notices
Remove past notices	Notices
Answer to the FAQ	FAQ

Faculty-head	
Responsibilities:	**Collaborations:**
Register to the system	
Modify accounts	
Publish new notices	Notices
Remove past notices	Notices
Manage faculties	Report

Notices	
Responsibilities:	**Collaborations:**
Display notices	
Update notices	

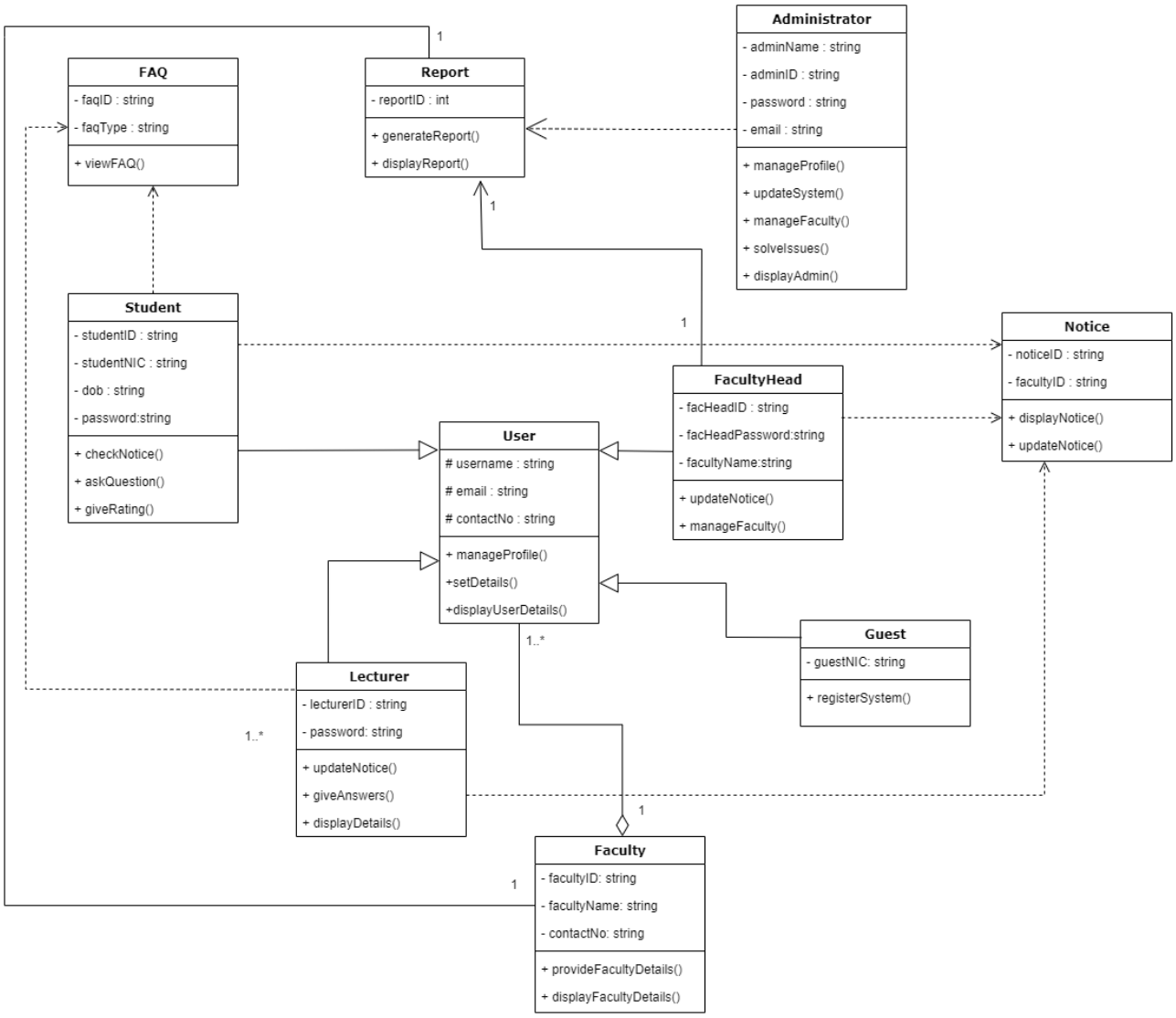
FAQ	
Responsibilities:	**Collaborations:**
View FAQ	

Faculty	
Responsibilities:	**Collaborations:**
Provide faculty details	

Administrator	
Responsibilities:	**Collaborations:**
Register to the system	
Modify accounts	
Manage faculties	Report
Solve bugs and issues	Report
Update the system	

Report	
Responsibilities:	**Collaborations:**
Generate faculty report	Faculty
Generate weekly report of updates	System administrator
Display reports	

CLASS DIAGRAMS



Exercise 2

CODES

```
class Administrator // Administrator Class
{
private:
string adminName;
string adminID;
string password;
string email;
public:
Administrator(string aName, string aID, string pw, string mail);
void manageProfile();
void updateSystem();
void manageFaculty();
void solveIssues(int type, Report *r2);
void displayAdmin();
};
#include "Administrator.h"

//constructor with parameters
Administrator::Administrator(string aName, string aID, string pw, string mail){
adminName = aName;
adminID = aID;
password = pw;
email = mail;
}
```

```
void Administrator::manageProfile() {
}
void Administrator::updateSystem() {
}
void Administrator::manageFaculty() {
}
void Administrator::solveIssues(int type, Report *r2) {
}
void Administrator::displayAdmin(){
cout << "Administrator Name  :" << adminName << endl;
    cout << "Administrator ID    :" << adminID << endl;
}
```

```
// Faculty Class
class Faculty{
private:
  string facultyID;
  string facultyName;
  string contactNo;
  Report *repo;
  User *usr[SIZE1];
public:
  Faculty();
  Faculty(string facID, string facName, string num);
  void addUser(User *s, User *l, User *u, User *f);
  void provideFacultyDetails();
  void displayFacultyDetails();
  ~Faculty();
};




Faculty::Faculty(){


  }
//constructor with parameters
Faculty :: Faculty(string facID, string facName, string num){
  facultyID = facID;
  facultyName = facName;
  contactNo = num;
```

```
}
void Faculty::addUser(User *s, User *l, User *u, User *f){
      usr[0] = s;
      usr[1] = l;
      usr[2] = u;
      usr[3] = f;
  }
void Faculty::provideFacultyDetails(){
  }
void Faculty::displayFacultyDetails(){
    cout<<"Faculty ID         :"<<facultyID<<endl;
    cout<<"Faculty Name       :"<<facultyName<<endl;
    cout<<"Faculty Contact No :"<<contactNo<<endl<<endl;
  }
Faculty::~Faculty(){
  cout << "Deleting faculty" << facultyID << endl;
}
```

```
//FacultyHead class
class FacultyHead : public User{
private:
        string facHeadID;
        string facHeadPassword;
        string facultyName;
        Report *rpt2;
public:
        FacultyHead();
        FacultyHead (string pfacHeadID, string pfacHeadPassword, string pfacultyName);
        void updateNotice(int qty, Notice *n3);
   void manageFaculty();
        void setDetails();
        void displayDetails();
        ~FacultyHead();
};
```

```
//default constructor
FacultyHead :: FacultyHead(){}

//constructor with parameters
FacultyHead :: FacultyHead (string pfacHeadID, string pfacHeadPassword, string pfacultyName)
```

```
{
facHeadID = pfacHeadID;
facHeadPassword = pfacHeadPassword;
facultyName = pfacultyName;
}void FacultyHead :: updateNotice(int qty, Notice *n3){

}void FacultyHead :: manageFaculty(){

}
void FacultyHead :: setDetails(){
}
void FacultyHead :: displayDetails(){
cout<<"Faculty Head ID\t:"<<facHeadID<<endl;
cout<<"Password\t:"<<facHeadPassword<<endl;
cout<<"Faculty Name\t:"<<facultyName<<endl<<endl;
}

FacultyHead :: ~FacultyHead(){

}
```

```
//FAQ Class
class FAQ{
private:
        string faqID;
        string faqType;

public:
        FAQ();
        FAQ(string pfaqID, string pfaqType);
        void viewFAQ();
        void setDetails();
        void displayDetails();
        ~FAQ();
};


//default constructor
FAQ :: FAQ(){
}
//constructor with parameters
FAQ :: FAQ(string pfaqID, string pfaqType)
{
faqID = pfaqID;
faqType = pfaqType;
}
void FAQ :: viewFAQ(){
}
void FAQ:: setDetails(){
```

```
}

void FAQ :: displayDetails()
{
cout<<"FAQ ID\t:"<<faqID<<endl;
cout<<"FAQ Type:"<<faqType<<endl<<endl;
}

FAQ :: ~FAQ(){

}
```

```
//Guest class
class Guest:public User
{
private:

string guestNIC;



public:
Guest();
Guest(string name, string gNIC,string Email,  string phone);
  void setDetails();
void registerSystem();



};



//default constructor
Guest::Guest(){
}
//constructor with parameters
Guest::Guest(string name, string gNIC,string gEmail,  string phone) {
email=gEmail;
guestNIC=gNIC;
username = name;
}
void Guest::setDetails() {
```

```
   cout<<username<<"\t\t\t"<<guestNIC<<endl;
}

void Guest::registerSystem() {
}
```

```
//Lecturer class
class Lecturer:public User{


  private:
    string lecturerID;
    string password;



  public:
  Lecturer();
  Lecturer(string aLectureID,string name,string uPassword,string email);
  void updateNotice(int qty, Notice *n2);
  void giveAnswers (int type, FAQ *a1);
  void displayDetails();
  void manageProfile();
  ~Lecturer();




};


//constructor with parameters
 Lecturer::Lecturer(string aLectureID,string lname,string uPassword,string lemail)
  {
      lecturerID = aLectureID;
      username = lname;
    password = uPassword;
      email = lemail;
```

```
}
void   Lecturer::updateNotice(int qty, Notice *n2){
}

void  Lecturer::giveAnswers(int type, FAQ *a1){

}
void Lecturer::displayDetails(){
  cout<<"Lecturer ID   :"<<lecturerID<<endl;
  cout<<"Lecturer Name :"<<username<<endl<<endl;
}

void Lecturer::manageProfile(){

}

Lecturer::~Lecturer(){

}
```

```
//Notice class
class Notice{
  private:
      string noticeID;
      string facultyID;
  public:
      Notice();
      Notice(string nID,string fID);
      void displayNotice();
      void updateNotice();
};

#include "Notice.h"
using namespace std;



//constructor with parameters
Notice::Notice(string nID,string fID){
        noticeID=nID;
        facultyID=fID;
}

void Notice::displayNotice(){
        cout<<"Notice ID  :"<<noticeID<<endl;
        cout<<"Faculty ID :"<<facultyID<<endl;
}

void Notice::updateNotice(){
```

```
}
```

```
class Report   //Report Class
{
private:
int reportID;
Faculty *fac;
public:
  Report();
Report(int preportID);
void genarateReport();
void displayReport();
};#include "Report.h"

//default constructor
Report::Report(){

}

//constructor with parameters
Report::Report(int preportID){
  reportID = preportID;
}

void Report::genarateReport() {

}

void Report::displayReport() {
cout << "Report ID :" << reportID << endl;
```

```
}
```

```
//Student class
class Student : public User
{
private:
string studentID;
string studentNIC;
string dob;
string password;
public:
Student();
Student(string stID,string uPassword,string stNIC,string sDOB,string name);
void askQuestion();
void manageProfile();
void checkNotice(int format , Notice *n1);
void giveRating(int rate, FAQ *r1);

};

#include"Student.h"
using namespace std;


//default constructor
Student::Student(){
}

//constructor with parameters
Student::Student(string name,string uPassword,string stID,string stNIC,string sDOB ) {
```

```
username=name;
password=uPassword;
studentID=stID;
studentNIC=stNIC;
dob=sDOB;
}

void Student::askQuestion() {
}

void Student::manageProfile(){
cout<<username<<"\t\t"<<studentID<<"\t"<<studentNIC<<endl;
}

void Student::checkNotice(int format , Notice *n1) {
}

void Student::giveRating(int rate, FAQ *r1) {
}
```

```
//User class
class User
{
protected:
string username;
string email;
  string contactNo;
public:
User();
User(string name,string uemail,string phone);
virtual void setDetails();
void displayUserDetails();
void manageProfile();



};


#include"User.h"
//default constructor
User::User() {
}

//constructor with parameters
User::User(string name,string uemail,string phone) {
username=name;
email = uemail;
contactNo = phone;
}
```

```
void User::setDetails() {

}

void User::manageProfile() {

}


void User::displayUserDetails() {

}
```

```
//Main

#include <iostream>
#include <string>

//Linking .cpp files
#include "Administrator.cpp"
#include "Faculty.cpp"
#include "FacultyHead.cpp"
#include "FAQ.cpp"
#include "Guest.cpp"
#include "Lecturer.cpp"
#include "Notice.cpp"
#include "Report.cpp"
#include "Student.cpp"
#include "User.cpp"

using namespace std;

//client program
int main()
{
cout << endl << "**********Online Helpdesk for Univercity
Students**********" << endl << endl << endl;

//Admin create objects
Administrator *A1 = new Administrator("John", "001", "John12345","john@gmail.com");
Administrator *A2 = new Administrator("Smith", "002", "Smith12345","smith@gmail.com");
```

```
A1->displayAdmin();

A2->displayAdmin();


cout << endl <<
"*********************************************************************************
*****" << endl << endl;


//Faculty create objects

Faculty *FA1 = new Faculty("FA001", "Faculty of Computing", "0111234567");

Faculty *FA2 = new Faculty("FA002", "Faculty of Management", "0112223456");


FA1->displayFacultyDetails();

FA2->displayFacultyDetails();


cout << endl <<
"*********************************************************************************
*****" << endl << endl;


//Faculty Head create objects

FacultyHead *F1 = new FacultyHead("FBA2100001", "Perera123@", "Faculty of Business
Administration");

FacultyHead *F2 = new FacultyHead("FOC2100016", "Fdo456!", "Faculty of Computing");


F1->displayDetails();

F2->displayDetails();


cout << endl <<
"*********************************************************************************
*****" << endl << endl;


//FAQ create objects
```

```
FAQ *faq1 = new FAQ("FAQ00001","Type 02");

FAQ *faq2 = new FAQ("FAQ00100","Type 03");


faq1->displayDetails();

faq2->displayDetails();


cout << endl <<
"*********************************************************************************
*****" << endl << endl;


//Guest create objects

Guest *g1 = new Guest("Jon","965530835V","Jon1999@gmail.com", "0715342001");

Guest *g2 = new Guest("Ben","988870835V","Ben98@gmail.com", "0715243010");


cout<<"Guest Name"<<"\t\t"<<"NIC"<<endl<<endl;


g1->setDetails();

g2->setDetails();


 cout << endl <<
"*********************************************************************************
*****" << endl << endl;


//Lecturer create objects

Lecturer *L1 = new Lecturer("LID001", "Yapa A.B.", "Yapa12345","yapa@gmail.com");

Lecturer *L2 = new Lecturer("LID002", "Gunapala C.C.", "Cc12345","gunapala@gmail.com");


L1->displayDetails();

L2->displayDetails();
```

```
cout << endl <<
"*********************************************************************************
*****" << endl << endl;


//Notice create objects

Notice*N1=new Notice("N001","F002");

Notice*N2=new Notice("N002","F005");


N1->displayNotice();

N2->displayNotice();


cout << endl <<
"*********************************************************************************
*****" << endl << endl;


//Report create objects

Report *R1 = new Report(1);

Report *R2 = new Report(2);


R1->displayReport();

R2->displayReport();


cout << endl <<
"*********************************************************************************
*****" << endl << endl;


//Student create objects

Student *s1 = new Student("Ben","Ben@win1","SIT105278","988530835V","1998.12.1");

Student *s2 = new Student("Ron","Ron@campus","SIT105267","988598835V","1998.2.1");


cout<<"Username"<<"\t"<<"Student ID"<<"\t"<<"Student NIC"<<endl<<endl;
```

```
s1->manageProfile();

s2->manageProfile();


 cout << endl <<
"*********************************************************************************
*****" << endl << endl;


//User create objects

User *U1 = new User("Jon","Jon1999@gmail.com","0715342001");

User *U2 = new User("Ben","Ben98@gmail.com","0715243010");


U1->displayUserDetails();

U2->displayUserDetails();




return 0;

}//end main function
```

CONTRIBUTION OF CODES

(1) Student Name: Hettiarachchi H.K.Y.K.

Student ID: IT21181474

- Student and User codes

(2) Student Name: Arachchi D.S.U

Student ID: IT21182914

- Lecturer and Guest codes

(3) Student Name: Vidyani H.A.L

Student ID: IT21184130

- Notice and Faculty codes

(4) Student Name: Fernando S.D.K

Student ID: IT21188572

- Faculty-head and FAQ codes

(5) Student Name: Liyanage P.P

Student ID: IT21184758

- Administrator and Report codes

- All our group members do our level best to complete our assignment.