Topic : Online Help Desk for University Students.

Group no : MLB_06.01_03

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**
IT21311536	S.S.R.G SEETHAWAKA	0740658492
IT21327780	LIYANAGE L.D.P.D	0717656875
IT21326868	JAYASOORIYA J.M.D.T	0704642020
IT21308802	Minsara K.K.B	0710825074
IT21324024	Sooriyakumara S.M.H.D	0764261426

Table of Contents

System Requirements.

1. On this platform, an unregistered student can visit the website pages and search for relevant public information such as notices but, cannot raise a ticket regarding an issue.
2. An unregistered student can create an account by giving details such as student email, register number and contact details.
3. A registered student can issue a ticket regarding the problem he/she had to face, and the ticket raised would be delivered to a support agent by the system.
4. The student can monitor the status of the ticket raised.
5. The support agent resolves pending tickets and notify the student via email.
6. There is a system administrator present who has access to the system records, publishes notices, FAQs and can manage user and staff accounts.

Classes Identified (through noun/verb analysis and refinement)

1. Unregistered user
2. Registered user
3. Ticket
4. Support agent
5. Notice
6. User accounts
7. FAQ

CRC Cards

Unregistered student	
Responsibilities	**Collaborators**
View notices	Notice
create account	

Registered student	
Responsibilities	**Collaborators**
Provide Feedback	
Raise Ticket	ticket
View ticket status	ticket
View notice	notice

Ticket	
Responsibilities	**Collaborators**
show ticket status	
Notify support agent	
Store details of tickets raised	

Support Agent

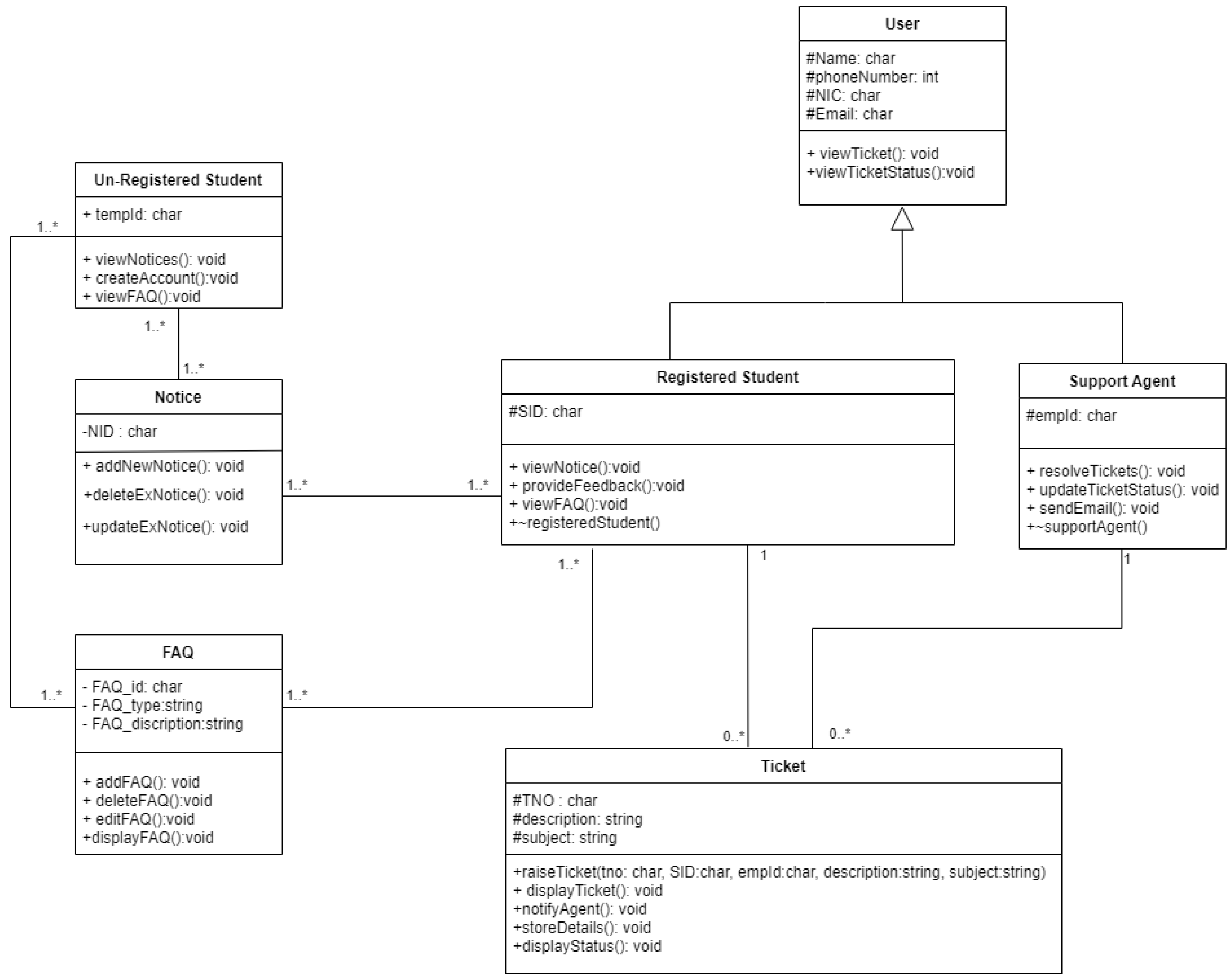
Support Agent	
Responsibilities	**Collaborators**
View pending tickets	
Resolve pending tickets	student
Update ticket status	ticket
Send email to notify student	

Notice	
Responsibilities	**Collaborators**
Add new notice	
Delete existing notice	
Update existing notice	

User account	
Responsibilities	**Collaborators**
Add new account to system	Unregistered student
Delete existing account	

FAQ	
Responsibilities	**Collaborators**
Add new FAQ	
Delete existing FAQ	
Update existing FAQ	
Display FAQ	

Class Diagram



Defining Classes

User Header File

```
#pragma once
/*User*/

#include<iostream>
using namespace std;
class User {

protected:
	char uName[20];
	int uPhoneNumber;
	char uNIC[12];
	char uEmail[30];

public:
	User();
	User(char name[], int phoneNumber, char NIC[], char email[]);
	void viewTicket();
	void viewTicketStatus();
	~User();
};
```

Registered Student Header File

```
#pragma once
/*Registered Student*/
#include<iostream>
#define SIZE 10
#include "User.h"
#include "Ticket.h"

class RegisteredStudent : public User
{

        protected:
                char SID[6];
                Ticket*ticket[SIZE];

        public:
                RegisteredStudent();
                RegisteredStudent(char name[], int phoneNumber, char NIC[], char email[], char sid[]);
                void viewNotice();
                void provideFeedback();
                ~RegisteredStudent();
};
```

Unregistered Student Header File

```
/* Unregistered Student */
#pragma once
#include <iostream>
```

```
class UnRegisteredStudent
{
protected:
        char utempId[6];

public:
        UnRegisteredStudent();
        UnRegisteredStudent(char tempId[]);
        void viewNotices();
        void createAccount();
        void viewFAQ();
        ~UnRegisteredStudent();
};
```

Support Agent Header File

```
#pragma once
/*SupportAgent*/
#include<iostream>
#include "User.h"
class SupportAgent :public User
{
        protected:
                char empId[6];

        public:
                SupportAgent();
                SupportAgent(char name[], int phoneNumber, char NIC[], char email[], char EID[]);
                void resolveTickets();
                void updateTicketStatus();
                void sendEmail();
                ~SupportAgent();
};
```

Ticket Header File

```
#pragma once
/*Ticket*/
#include<iostream>
#include "RegisteredStudent.h"
#include "SupportAgent.h"
using namespace std;

class Ticket:
{
protected:
   char TID[6];
   string Discription;
   string Subject;

public:
   Ticket();
   Ticket(char Tid[], string discription, string subject);
   void DisplayTicket();
   void NotifyAgent();
   void StoreDetails();
   void DisplayStatus();
   ~Ticket();
};
```

Notice Header File

```
/*Notice*/
#pragma once
#include<iostream>
#include "UnregisteredStudent.h"
#include "RegisteredStudent.h"



class Notice
{
private:
   char NID[6];
UnregistedStudent *unregistedstudent;
RegistedStudent*registedstudent;



public:
  void addNewNotice();
  void deleteExNotice();
  void updateExNotice();
};
```

FAQ Header File

```
#pragma once

#include<iostream>

#include<cstring>

#include<string>

#include "User.h"


#include "UnregisteredStudent.h"

#include "RegisteredStudent.h"


#include

using namespace std;



class FAQ

{


protected:

        char FAQ_id[20];

        string FAQ_type;

        string FAQ_discription;

        UnregistedStudent *unregistedstudent;

        RegistedStudent*registedstudent;


public:

        FAQ();

        FAQ(char Fid[], string Ftype, string Fdiscription);

        void addFAQ();
        void deleteFAQ();
        void editFAQ();
        void displayFAQ();
        ~FAQ();
};
```

```
}
```

Methods of Classes defined

Users

```
#include <iostream>

#include<iostream>
#include "User.h"
#include<string.h>

User::User()
{
	uPhoneNumber = 0;
	uEmail[0] = 0;
	uNIC[0] = 0;
	uName[0] = 0;
}
User::User(char name[], int phoneNumber, char NIC[], char email[])
{
	strcpy_s(uName, name);
	strcpy_s(uNIC, NIC);
	strcpy_s(uEmail, email);
	uPhoneNumber = phoneNumber;
}
void User::viewTicket()
{
}
void User::viewTicketStatus()
{
}
```

```
User::~User()
{
}
```

Unregistered Student

```
/* Unregistered Student */

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

UnRegisteredStudent::UnRegisteredStudent()
{
    utempId[0] = 0;
    cout << "Adding Unregistered student." << endl;
}

UnRegisteredStudent::UnRegisteredStudent(char tempId[])
{
    strcpy_s(utempId, tempId);
}

void UnRegisteredStudent::viewNotices()
{

}
```

```
void UnRegisteredStudent::createAccount()
{


}


void UnRegisteredStudent::viewFAQ()
{


}


UnRegisteredStudent::~UnRegisteredStudent()
{


}
```

Registered Student

```
#pragma once
/*Registered Student*/
#include<iostream>
#include "User.h"
#include "Ticket.h"


class RegisteredStudent : public User
{


	protected:
		char SID[6];
		 Ticket * ticket[0];


	public:
			RegisteredStudent();
			RegisteredStudent(char name[], int phoneNumber, char NIC[], char email[], char sid[]);
			void viewNotice();
			void provideFeedback();
			~RegisteredStudent();
};
```

<u>Support Agent</u>

```
#include<iostream>
#include<string.h>
#include "SupportAgent.h"



using namespace std;

SupportAgent::SupportAgent()
{
	empId[0] = 0;
	cout << "Adding new Support Agent." << endl;
}
SupportAgent::SupportAgent(char name[], int phoneNumber, char NIC[], char email[], char EID[]) :
User(name, phoneNumber, NIC, email)
{
	strcpy_s(empId, EID);
}
void SupportAgent::resolveTickets()
{


}
void SupportAgent::updateTicketStatus()
```

```
{


}
void SupportAgent::sendEmail()
{


}
SupportAgent::~SupportAgent()
{
            cout << "Deleting Support agent details" << endl;
}
```

Ticket

```
#pragma once
/*Ticket*/
#include<iostream>
#include "RegisteredStudent.h"
#include "SupportAgent.h"
using namespace std;

class Ticket
{
protected:

    string Discription;
    string Subject;
    char TID[6];

public:
    Ticket();
    Ticket(char Tid[], string discription, string subject);
    void DisplayTicket();
    void NotifyAgent();
    void StoreDetails();
    void DisplayStatus();
    ~Ticket();
};
```

<u>Notice</u>

```
/*Notice*/
#include <iostream>
#include<string.h>
#include"Notice.h"
using namespace std;

Notice :: Notice()
{
  NID[0] = 0;
  cout << "Notice."<< endl;
}
Notice :: Notice(char NID)
{
  strcpy_s(NID);
}
void Notice :: addNewNotice()
{


}
```

```
void Notice :: deleteExNotice()
{

}
void Notice :: updateExNotice()
{

}
```

FAQ

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



FAQ::FAQ() {
	cout << "Adding new FAQ " << endl;
	strcpy_s(FAQ_id, " ");
	FAQ_type = "";
	FAQ_discription = "";


};

FAQ::FAQ(char Fid[], string Ftype, string Fdiscription)
{
        strcpy_s(FAQ_id, Fid);
        FAQ_type = Ftype;
        FAQ_discription = Fdiscription;

};
void FAQ::addFAQ() {

};
void FAQ::deleteFAQ() {

};
void FAQ::editFAQ() {

};

void FAQ::displayFAQ() {

};

FAQ::~FAQ() {

        cout << "Deleting FAQ " << endl;
};
```

Main.cpp

```
#include<iostream>

#include<string.h>

#include "UnregisteredStudent.h"

#include "RegisteredStudent.h"

#include "SupportAgent.h"

#include "User.h"

#include "Notice.h"

#include "FAQ.h"

#include "Ticket.h"


using namespace std;


int main()

{

	RegisteredStudent * regstudent = new RegisteredStudent();

	SupportAgent * supagent = new SupportAgent();

	UnRegisteredStudent * unregstuent = new UnRegisteredStudent();

	Notice * notice = new Notice();

	FAQ * newFAQ = new FAQ();

	Ticket * newTicket = new Ticket();


	delete regstudent;
```

```
        delete supagent;
        delete unregstuent;
        delete notice;
        delete FAQ;
        delete Ticket;
}
```

Contributions

REGISTER NUMBER	CONTRIBUTIONS
IT21311536	REQUIREMENTS IDENTIFIED: No. 03 and 02 CLASSES IDENTIFIED : 1. User class 2. Registered Student 3. Support Agent CLASS DIAGRAM: 1. User class 2. Registered Student 3. Support Agent CLASSES CREATED: 1. User class 2. Registered Student 3. Support Agent

IT21327780	REQUIREMENTS IDENTIFIED: No. 01 CLASSES IDENTIFIED : 1. Unregistered user CLASS DIAGRAM: 1. Un – registered student CLASSES CREATED: 1. Un – registered student
IT21326868	REQUIREMENTS IDENTIFIED: No. 06 CLASSES IDENTIFIED : 1. Notice CLASS DIAGRAM: 1. Notice CLASSES CREATED: 1. Notice
IT21324024	REQUIREMENTS IDENTIFIED: No. 04 CLASSES IDENTIFIED : 2. Ticket CLASS DIAGRAM: 2. Ticket CLASSES CREATED: 2. Ticket
IT21308802	REQUIREMENTS IDENTIFIED: No. 07 CLASSES IDENTIFIED : 3. FAQ

	CLASS DIAGRAM: 3. FAQ CLASSES CREATED: 3. FAQ