

A Minor Project Report

On

## **MENTORING MANAGEMENT SYSTEM**

Submitted in partial fulfillment of the  
Requirements for the award of the degree

### **BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY**

Submitted By

**Karrolla Rohith Mithra (19H61A1229)**  
**M Raja Reddy (19H61A1238)**  
**S Nitya Sree (19H16A1250)**

Under the guidance of  
**Mr. G.Shekar Reddy**  
Assistant Professor



**Department of Information Technology**

**ANURAG GROUP OF INSTITUTIONS**

**(An Autonomous Institution)**

**(Affiliated to JNTU-HYD, Approved by AICTE and NBA Accredited)**

**Venkatapur (V), Ghatkesar (M), Medchal district, Hyderabad, Telangana, 500088**

**2019-2023**

# **ANURAG GROUP OF INSTITUTIONS**

**(An Autonomous Institution)**

**(Affiliated to JNTU-HYD, Approved by AICTE and NBA Accredited)**

**Venkatapur (V), Ghatkesar (M), Medchal district, Hyderabad, Telangana, 500088**

**Department of Information Technology**



## **CERTIFICATE**

This is to certify that the project report entitled “**MENTORING MANAGEMENT SYSTEM**” is a Bonafide work done and submitted by **Karrolla Rohith Mithra (19H61A1229)**, **M Raja Reddy (19H61A1238)** and **S Nitya Sree (19H61A1250)** in partial fulfillment of the requirements for the award of the degree of B.Tech in **Information Technology from Anurag Group of Institutions (An Autonomous Institution)**, Affiliated to Jawaharlal Nehru Technological University, Hyderabad during the academic year 2022-2023 and the Bonafide work has not been submitted elsewhere for the award of any other degree.

**Internal Guide**

**Mr. G. Sekhar Reddy**

**Assistant Professor**

**Department of IT**

**HOD**

**Dr. K. S. Reddy**

**Professor,**

**Department of IT**

**External Examiner**

## ACKNOWLEDGEMENT

It is our privilege and pleasure to express my profound sense of respect, gratitude and indebtedness to our guide **Mr.G.Sekhar Reddy, Assistant Professor**, Department of Information Technology, Anurag Group of Institutions, Ghatkesar, for his indefatigable inspiration, guidance, cogent, discussion, encouragement and valuable advice throughout the dissertation work.

We would like to express our sincere thanks to **Dr. K. Sudheer Reddy**, Professor and Head of the Department of Information Technology, Anurag Group of Institutions, Ghatkesar, whose motivation in the field of software development has made us to overcome all hardships during the course of study and successful completion of the project.

We extend our sincere thanks to **Dr. G.Vishnu Murthy**, Dean School of Engineering, Anurag University, Venkatapur(V), Ghatkesar(M), Medchal Dist., for their encouragement and constant help.

We would like to thank all the faculty of the IT department, for their support and encouragement.

Finally, we would like to express our heartfelt thanks to our parents who were very supportive both financially and mentally and for their encouragement to achieve our set goals.

**K Rohith Mithra (19H61A1229)**

**M Raja Reddy (19H61A1238)**

**S Nitya Sree (19H61A1250)**

## **DECLARATION**

This is to Certify that the project work entitled “**MENTORING MANAGEMENT SYSTEM**” submitted to Anurag Group of Institutions in partial fulfillment of the requirement for the award of the Degree of Bachelor of Technology (B-Tech), is an original work carried out by **Karrolla Rohith Mithra(19H61A1229)**, **M Raja Reddy(19H61A1238)**, **S Nitya Sree(19H61A1250)** under the guidance of **Mr.G.Sekhar Reddy**, Assistant Professor in the Department of Information Technology. This matter embodied in this project is a genuine work, done by the students and has not been submitted whether the university or to any other university/Institute for the fulfillment of the requirement of any course of study.

**K Rohith Mithra(19H61A1229)**

**M Raja Reddy(19H61A1238)**

**S Nitya Sree (19H61A1250)**

## **ABSTRACT**

Any functioning Organization requires a certain technology to ease their work. Specifically, when it comes to Educational Institutions where there exists a large number of students who can uniquely know their individual performance (marks, attendance). More importantly, Parents can view progress of their children using this Technological support.

This is where our project comes into play. We provide a platform that allows students to access their marks and attendance from anywhere at any time with just a click on our website. Parents can also view the progress of their children by simply logging in through their mobile phones or computers.

## **CONTENTS**

<b>CHAPTERS</b>	<b>PAGE NO</b>
<b>List of Figures</b>	
<b>1.INTRODUCTION</b>	
1.1 Overview	11
1.2 Objective	11
1.3 Problem Statement	11
1.4 Scope of the Project	11
1.5 System Requirements	12
1.5.1 Software Requirements	12
1.5.2 Hardware Requirements	12
1.6 Existing System	12
1.7 Proposed System	13
<b>2.REQUIREMENTS</b>	
2.1 Software Requirements and Specification	14
2.2 Modules	15
2.3 User Interfaces	15
<b>3.ANALYSIS</b>	
3.1 UML Diagrams	
3.1.1 Class Diagram	16
3.1.2 Use Case Diagram	17
3.1.3 Sequence Diagram	17
3.1.4 Component Diagram	20
3.1.5 Deployment Diagram	21
<b>4.DESIGN</b>	
4.1 System Architecture	23
4.2 E-R Diagram	24
4.3 Database Design	25



<b>5.IMPLEMENTATION</b>	
5.1 Code Snippets	27
<b>6.USER SCREENS</b>	46
<b>7.TESTING METHODOLOGY</b>	
7.1 Software Testing	71
7.1.1 Types of testing	71
7.2 Test Cases	73
<b>8.CONCLUSION</b>	74
<b>9.REFERENCE</b>	75



## LIST OF FIGURES

Figure Number	Name of Figure	Page Number
3.3.1	Class Diagram	15
3.3.2	Use case Diagram	16
3.3.3	Sequence Diagram	18
3.3.4	Component Diagram	19
3.3.5	Deployment Diagram	20
4.1	System Architecture	23
4.2	E-R diagram	24
4.3	Database Design	25
6.1	Home Page	46
6.2	Admin Login Page	47
6.2.1	Admin Index Page	48
6.2.2	Create accounts	49
6.2.3	View accounts	50
6.2.4	Assign students	51
6.2.5	Assigned students List	52
6.3	Mentor Login Page	53
6.3.1	Mentor Index page	54
6.3.2	Add student data	55
6.3.3	Update data	56
6.3.4	Delete data	57
6.3.5	View student list	58
6.3.6	View Query	59

6.3.7	Respond to Query	60
6.3.8	Add Comments	61
6.4	Student Login page	62
6.4.1	Student Index page	63
6.4.2	View Data	64
6.4.3	Create Query	65
6.4.4	View Query	66
6.5	Parent Login Page	67
6.5.1	Parent Index Page	68
6.5.2	View data	69
6.5.3	View Comments	70

## **1.1 INTRODUCTION**

A web application or web app is an application software that runs in a web browser, unlike software programs that run locally and natively on the operating system (OS) of the device. Web applications are delivered on the World Wide Web to users with an active network connection.

## **1.2 OVERVIEW**

The web application which we are developing is a dynamic web application which aims to retrieve data about every individual student regarding their marks and attendance and It can be accessed by Mentor, Parent and student. The Home Page of this web application consists of a Login Page where Mentor, Parent, Student can be logged in and perform their respective activities within.

## **1.3 OBJECTIVES OF PROJECT**

- To Track the Academics and Performance of Students by Parents.
- It makes possible to communicate with parents concerning the attendance of their children.
- It emphasizes continuous monitoring and it consumes less time.
- It ensures security but also reminds parents to be in constant touch, thus improving student learning outcomes.

## **1.4 PROBLEM STATEMENT**

To create a web application where Status of individual student is known by their respective mentors and parents.

## **1.5 SCOPE OF PROJECT**

Web application development has gotten a lot of focus in recent years in the attempt of improving the user Experience. Any modern web application development aims to design sites that are more intuitive and have the option of viewing on multiple platforms-

desktops, laptops, smartphones, etc., and a lot more. One such exciting upgrade to web application development happened through the creation of dynamic web application. Compare with a static web application, these are much more functional as they allow users to interact with the content mentioned on the page and it is easy to maintain and provides better user experience.

## **1.6 SOFTWARE AND HARDWARE REQUIREMENTS:**

### **1.6.1 SOFTWARE REQUIREMENTS:**

- |    |                         |                         |
|----|-------------------------|-------------------------|
| 1. | <b>Backend</b>          | : Python Language       |
| 2. | <b>Operating System</b> | : Windows XP/7/ 8/10/11 |
| 3. | <b>Database</b>         | : MySQL                 |
| 4. | <b>Frontend</b>         | : Tkinter               |

### **1.6.2 HARDWARE REQUIREMENTS:**

- |    |                          |                          |
|----|--------------------------|--------------------------|
| 1. | <b>Processor</b>         | : processor above 500MHz |
| 2. | <b>RAM Capacity</b>      | : 3 GB RAM               |
| 3. | <b>Screen Resolution</b> | : 190 x 200 MP           |

## **1.7 Existing System:**

The existing system consists of more manual work, and all the data and activities are stored in a paper-based format. This results in communication between mentors and mentees being delayed by closing universities and colleges. Only a short period of time is available for interaction during which there is no connection between mentors and mentees. This gap results in a lack of interaction between them, which means that it becomes difficult for the mentor to learn about the mentee's progress or understand their needs.

## **1.8 Proposed System:**

The proposed system takes up the whole existing system into a virtual platform. The main idea of this system is to cope up with today's technology and adapting into its culture which gives more security and more reliability. Time constraints is main part in it, where the students can interact with their mentors at any time they need. This builds a strong relationship between them which will be helpful for the mentors to upskill a student knowledge based on their interaction.

The proposed system has following benefits:

- It helps for better communication between the mentor and student;
- It helps for better interaction between the mentor and student;
- It helps for more efficient use of resources by parties (Mentor,parent and Student).

## **2.REQUIREMENTS**

### **2.1 Software Requirements and Specifications**

A software requirements specification (SRS) is a document that describes what the software will do and how it will be expected to perform. It also describes the functionality the product needs to fulfil all user's needs.

#### **Functional Requirements:**

These functional requirements define the external behavior of a system. In other words it illustrates what the system does and what the system must not do. The following are the functional requirements of this dashboard :

- It should be able to authenticate the user.
- It should help the users to access the data of the required category.
- It should be always accurate regarding the data.
- The uploaded excel files should be stored in the database.
- The user details such as their name, email and password must be stored and viewed by the admin when required.
- It must be helpful to view the past data also.

#### **Non Functional Requirements:**

A non-functional requirement is a goal that describes how a software system must work or perform. The non-functional requirements are:

- The performance should be dynamic and fast.
- It should be user friendly and visually appealing.
- The data provided should be accurate and clear.
- The users must not find any glitch when the data is being uploaded by the admin.

## 2.2 Modules :

- **Admin Module** : In this module the registration part is done, the admin registers the students and the faculty and also assigns faculty to students.
- **Student Module** : In this module the students view the faculty assigned to them and can communicate to the faculty through messaging them getting back responses.
- **Faculty Module** : In this module the faculties view the students assigned to them and can communicate with the students through messaging them.
- **Parent Module** : In this module , parents can view students attendance , marks and remarks of respective mentors.(track the status of students).

## 2.3 User Interface:

**Login page:** This the login page of Mentoring Management system. It has the login option for parent,student,mentor and admin from where the page redirects to their respective login pages

**Admin Homepage:** In this page, there are many options for Admin. The admin can add a new member such as mentor, student and parent. Admin can assign students to mentors.

**Mentor Homepage:** In this page, there are many options for Mentor. The mentor can add ,update and delete details of a students.

**Parent Homepage:** In this page, there are many options for parent. The parent can track the details of the student.

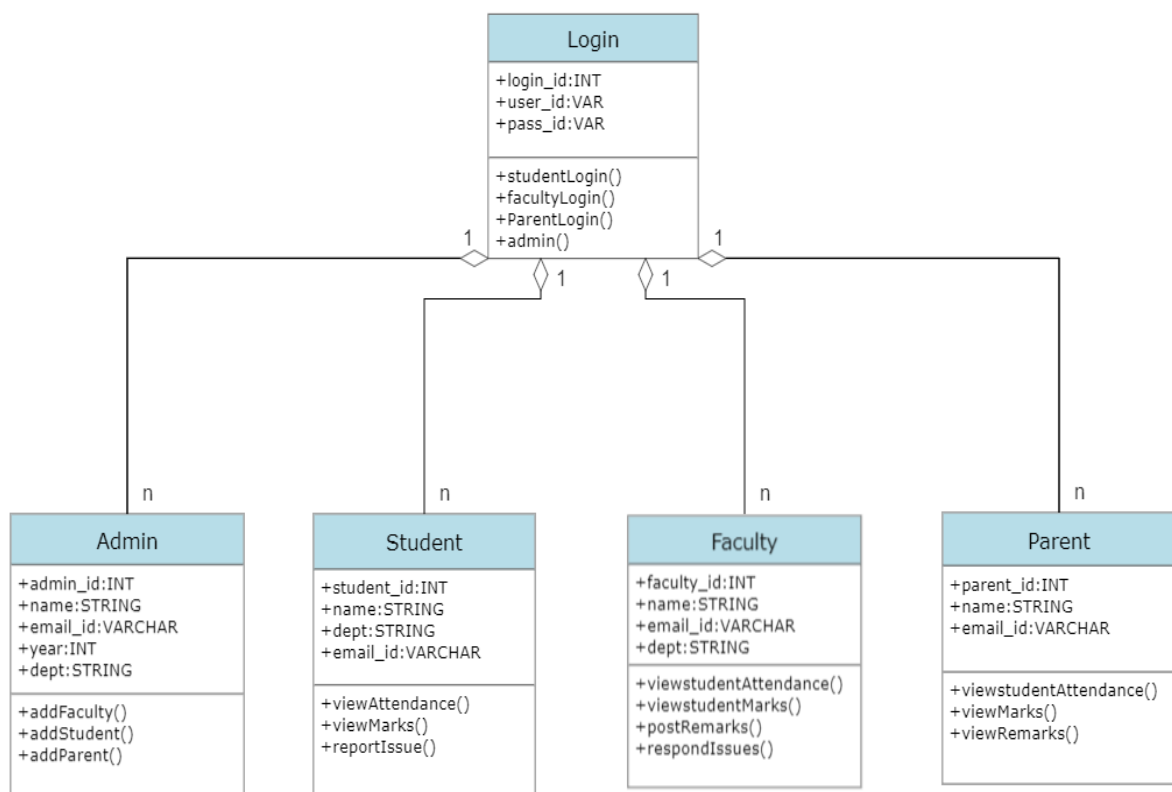
**Student Homepage:** In this page, there are many options for student. The admin can see their details and ask queries

## 3.ANALYSIS

### 3.1 UML Diagrams

#### 3.1.1 Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.





### 3.1.2 Use case diagram

In the Unified Modeling Language, a use case diagram can summarize the details of our system's users(also known as actors) and their interactions with the system. To build one, we will use a set of specialized symbols and connectors.

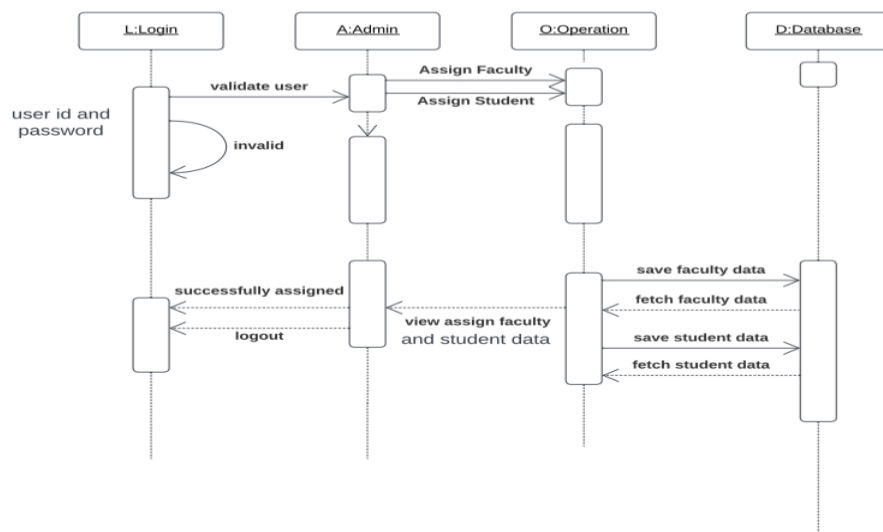


Use Case Diagram

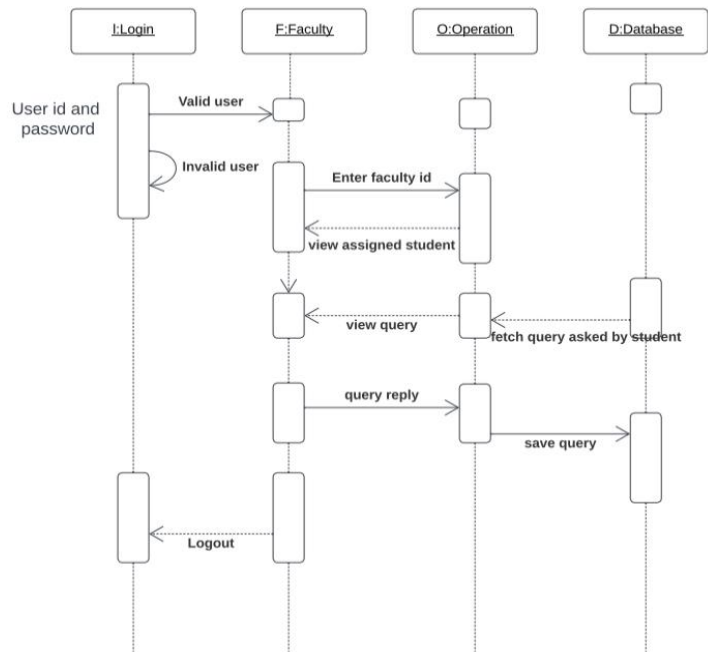
### 3.1.3 Sequence Diagram

A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

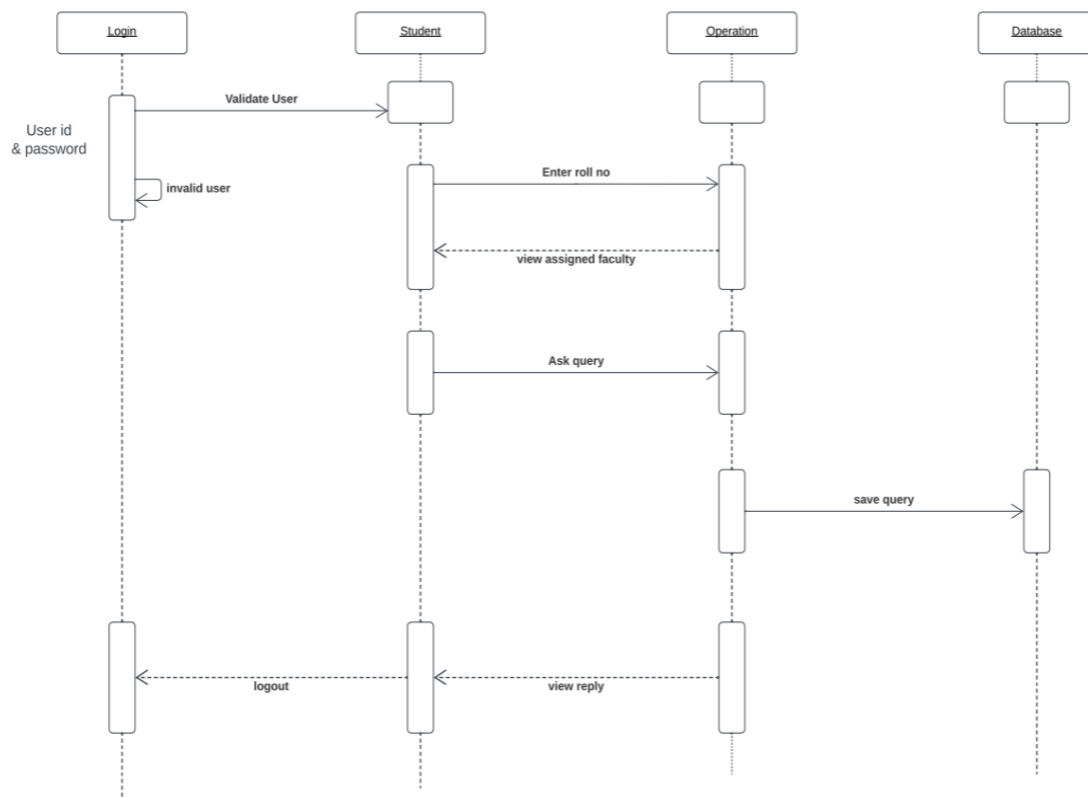
**a) Admin:**



**b)Faculty:**



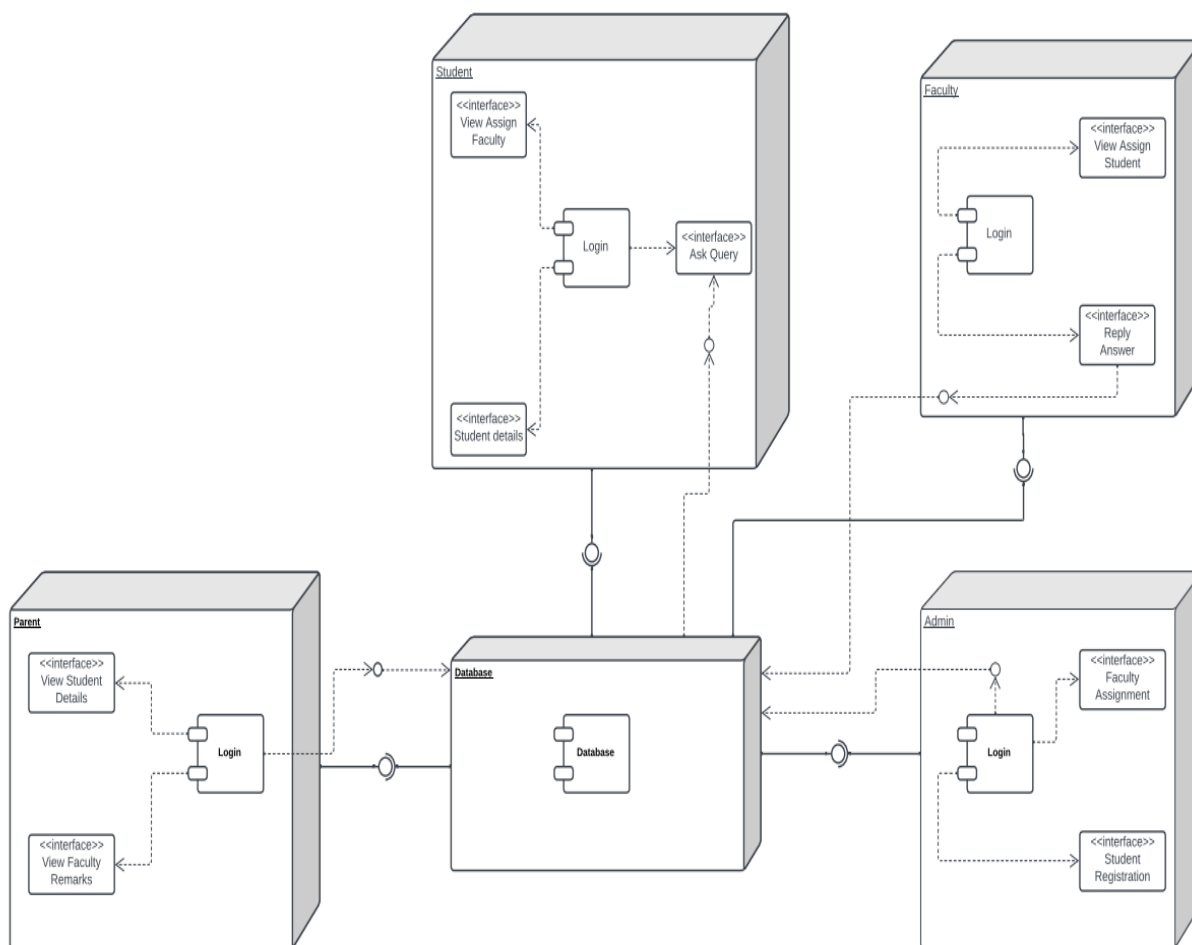
### c)Student:



#### **2.1.4 Component Diagram:**

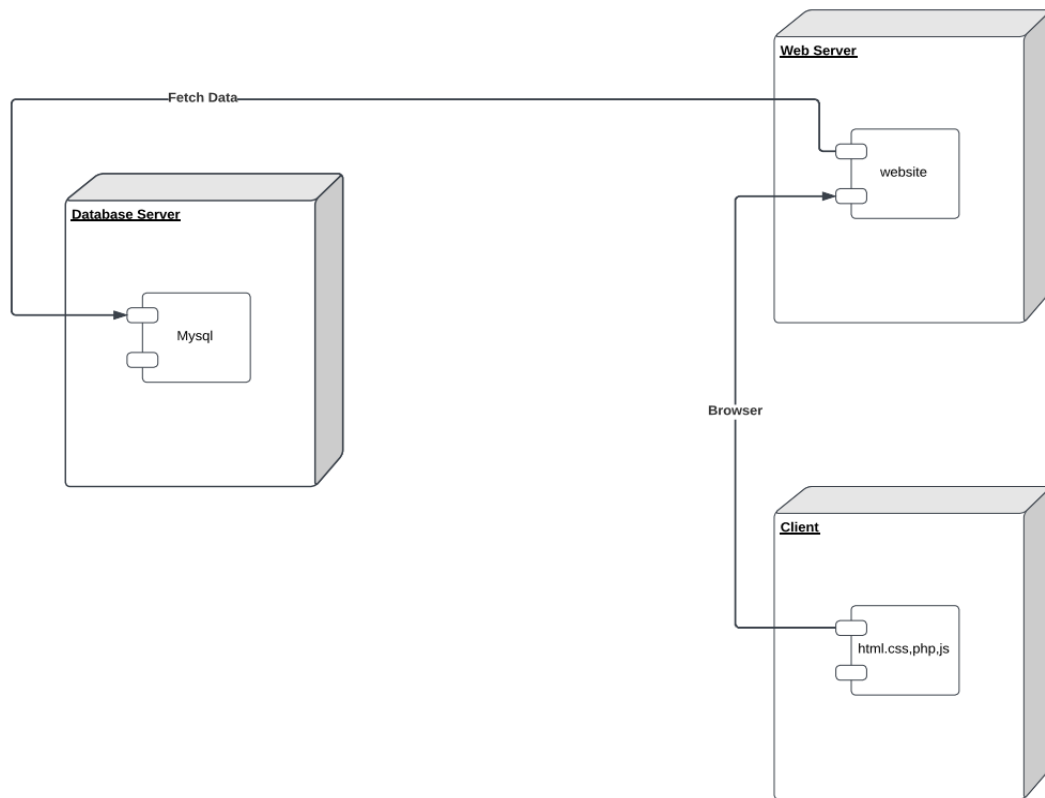
There are five main blocks in component diagram, the Student block, Faculty block, admin block, Parent block and Database block.

- **Student Block:** In Student there is a login component in which we have certain interfaces like ask query, student detail, view assigned faculty.
- **Faculty Block :** In faculty block there is a login component in which we have certain interfaces like view assigned student, reply answer.
- **Admin Block :** In admin block there is a login component in which we have certain interfaces like faculty assignment, student registration.
- **Parent Block :** In Parent Block there is a login component in which we have certain interfaces like view student details and view faculty remarks.
- **Database Block :** This block is in between all the blocks and it has a database component which is used for storing all the information.



### 2.3.4 Deployment Diagram:

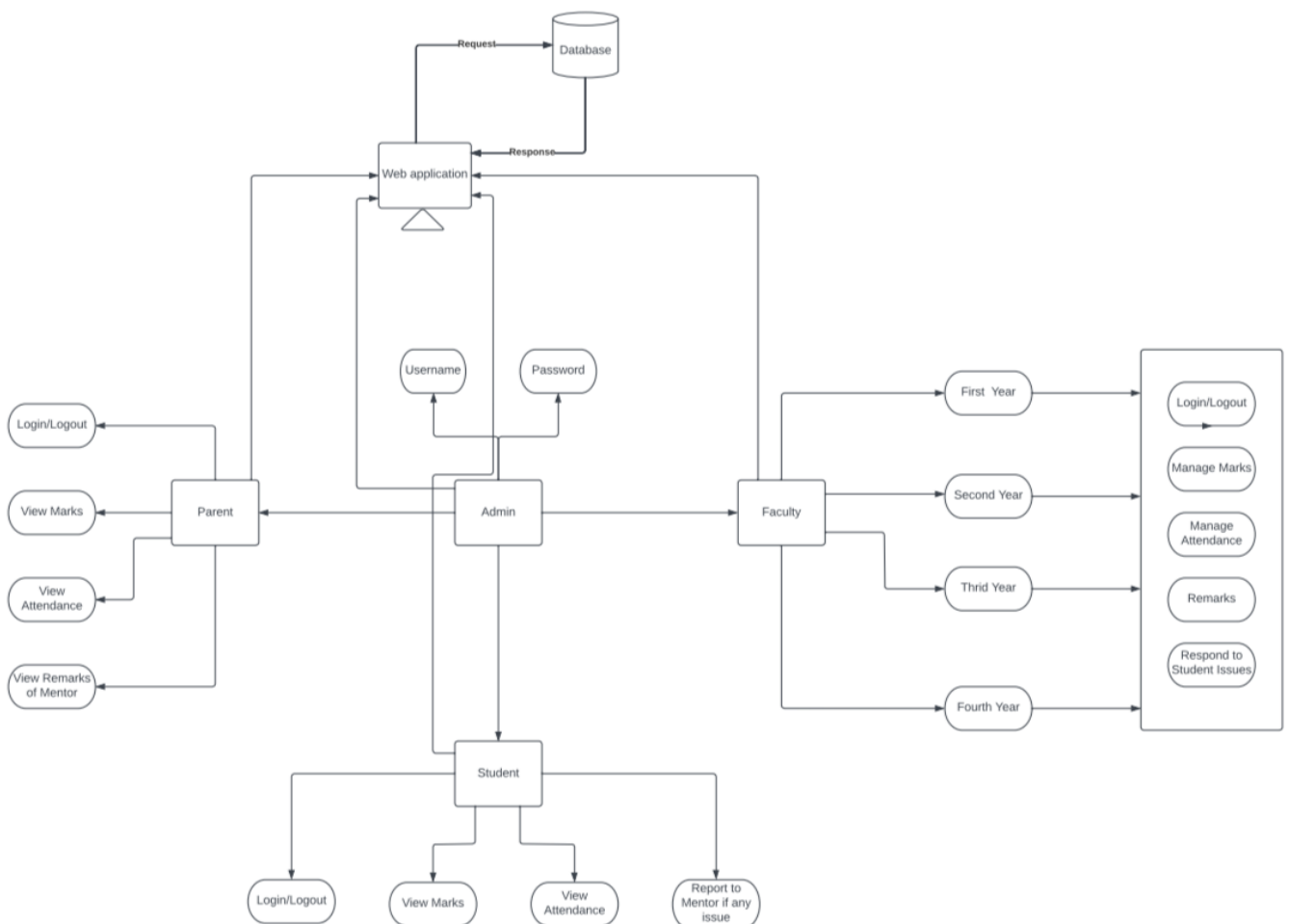
Deployment diagram is a structure diagram which shows architecture of our system as deployment(distribution) of software artifacts of deployment targets that is to the institute. In this diagram, institutes cloud server is there, which will be get connected to the web server so that end-user can interact with our website through there.



## 4.DESIGN

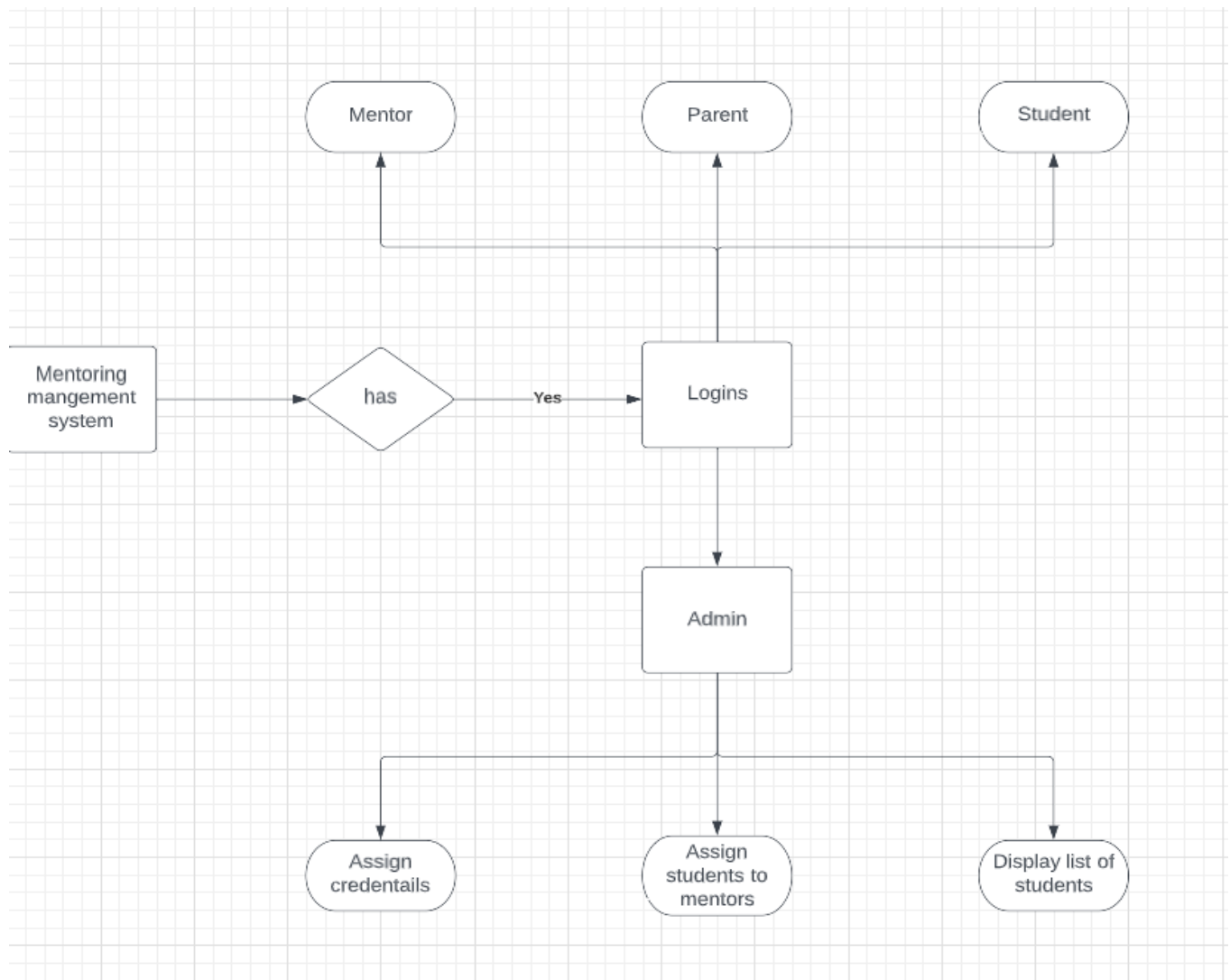
### 4.1 System Architecture

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. A system architecture can consist of system components and the sub-systems developed that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages (ADLs).



## 4.2 E-R Diagram

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships. ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships.





### 4.3 Database Design

#### Mentor table:

This table consists of four fields-id, name, email, password. These are the required Credentials for a mentor to login.

	id	name	email	password
►	I9Y14H	naveen	naveen@gmail.com	1234
	U96T87	lakshmi	lakshmi@gmail.com	lakshmi
	JL7ME5	asha	asha@gmail.com	asha
	ALUM0K	nitisha	nitisha@gmail.com	nitisha
	MQG6EO	sriram	sriram@gmail.com	sriram
	E9AHFB	hema	hema@gmail.com	hema
	0959Q1	hema	hema@gmail.com	hema
	STGBEW	naga	naga@gmail.com	naga

#### Student Table:

This table consists of four fields-id, name, email, password. These are the required Credentials for a Student to login.

	id	name	email	password
►	B048QL	nithya	nitya@gmail.com	1234
	8XVKM4	rohit	rohit@gmail.com	rohit
	NR8659	raja	raja@gmail.com	raja
	QRQ5LF	tanmai	tanmai@gmail.com	tanmai
	JFIR9V	chikku	chikku@gmail.com	chikku
	JFEABE	neha	neha@gmail.com	neha
	EBBKLW	nehan	nehan@gmail.com	nehan
	88DVNY	abhi	abhi@gmail.com	abhi

**Parent Table:**

This table consists of four fields-id, name, email, password. These are the required Credentials for a parent to login.

id	name	email	password
B048QL	srinivas	srinivas@gmail.com	1234
8XVKM4	kalyan	kalyan@gmail.com	kalyan
NR8659	shashi	shashi@gmail.com	shashi
QRQ5LF	shekar	shekar@gmail.com	shekar
JFEABE	kaushik	kaushik@gmail.com	kaushik
88DVNY	sudeer	sudeer@gmail.com	sudeer

## 5. IMPLEMENTATION

### 5.1 Code Snippets:

#### Database Code:

```
#create database mentormanagement;
#create table student(id varchar(250),name,email varchar(250),password varchar(250));
# create table mentor(id varchar(250),name varchar(250),email varchar(250),password varchar(250));
# create table parent(id varchar(250),name varchar(250),email varchar(250),password varchar(250));
#create table studentdata(id varchar(250),name varchar(250),cgpa varchar(25),attendance varchar(250));
# create table studentremarks(id varchar(250),comment varchar(250));
# create table studentquery(id varchar(250),query varchar(250));
#create table assign(studentid varchar(250),mentorid varchar(250));
import mysql.connector
from mysql.connector import Error
import string
import random
import base64
def db_connection():
    try:
        connection = mysql.connector.connect(host='localhost',
                                             database='mentormanagement',
                                             user='root',
                                             password='admin')
        if connection.is_connected():
            db_Info = connection.get_server_info()
            print("Connected to MySQL Server version ", db_Info)
            cur = connection.cursor()
            return connection,cur
    except Error as e:
        print(e)
#db_connection()
def studentregister(name,username ,password):
    N = 6

    res = ".join(random.choices(string.ascii_uppercase +
                                string.digits, k=N))
    connection,cur=db_connection()
    query=("insert into student(id ,name,email,password) values(%s,%s,%s,%s)" )
    cur.execute(query,(res ,name,username ,password))
    connection.commit()
#studentregister("naveen@gmail.com" ,"6305416114")
def parentregister(id,name,username ,password):
    connection,cur=db_connection()
    query=("insert into parent(id,name,email,password) values(%s,%s,%s,%s)" )
    cur.execute(query,(id,name,username ,password))
    connection.commit()
#parentregister('Q4T6UB',"rajeshrao@gmail.com" ,"987654")
```

```

def mentorregister(name,username ,password):
    N = 6

    res = ".join(random.choices(string.ascii_uppercase +
                                string.digits, k=N))
    connection,cur=db_connection()
    query=("insert into mentor(id ,name,email,password) values(%s,%s,%s,%s)" )
    cur.execute(query,(res ,name,username ,password))
    connection.commit()
#mentorregister("srinu@gmail.com" ,"456321")

def comments(id,comment):

    connection,cur=db_connection()
    query=("insert into studentremarks(id ,comment) values(%s,%s)" )
    cur.execute(query,(id,comment))
    connection.commit()
#comments("Q4T6UB" ,"your child have low attendance percentage")

def studentquery(id,q):

    connection,cur=db_connection()
    query=("insert into studentquery(id ,query) values(%s,%s)" )
    cur.execute(query,(id,q))
    connection.commit()
#studentquery("srinu@gmail.com" ,"wrong cgpa,please update it")

def student_data(id,name,cgpa,attendance):

    connection,cur=db_connection()
    query=("insert into studentdata(id,name,cgpa,attendance) values(%s,%s,%s,%s)" )
    cur.execute(query,(id,name,cgpa,attendance))
    connection.commit()
#student_data("Q4T6UB","naveen","6.3","89%")

def retrivestudentandmentor():
    connection,cur=db_connection()
    cur.execute("select id from student")
    data1=cur.fetchall()
    sid=[]
    for i in data1:
        sid.append(i[0])
    cur.execute("select id from mentor")
    data2=cur.fetchall()
    mid=[]
    for i in data2:
        mid.append(i[0])
    return sid,mid
#()
def assign(sid,mid):

```

```

#sid,mid=retrivestudentandmentor()
connection,cur=db_connection()
query=("insert into assign(studentid ,mentorid) values(%s,%s)" )
for i in sid:
    cur.execute(query,(i,mid))
    connection.commit()
#assign()
def sretrieve():
    connection,cur=db_connection()
    cur.execute("select studentid from assign")
    data=cur.fetchall()
    studentid=[]
    for i in data:
        studentid.append(i[0])
    return studentid
def rdata(id):
    connection,cur=db_connection()
    cur.execute("select * from studentdata where id=%s",(id,))
    data=cur.fetchall()
    n=[]
    c=[]
    a=[]
    for i in data:
        n.append(i[1])
        c.append(i[2])
        a.append(i[3])
    return n,c,a
def updatedata(name,cgpa,percentage,id):
    connection,cur=db_connection()
    cur.execute("update studentdata set name=%s,cgpa=%s,attendance=%s where id
=%s",(name,cgpa,percentage,id))
    connection.commit()

def deletedata(id):
    connection,cur=db_connection()
    cur.execute("delete from studentdata where id=%s",(id,))
    connection.commit()

def respquery(id,query):
    connection,cur=db_connection()
    cur.execute("insert into studentquery(id,query) values(%s,%s)",(id,query))
    connection.commit()

def sretrieve1():
    connection,cur=db_connection()
    cur.execute("select id from studentdata")
    data=cur.fetchall()
    studentid=[]
    for i in data:
        studentid.append(i[0])

```

```

    print(studentid)
    return studentid
#sretrieve1()
def sretrieve2():
    connection,cur=db_connection()
    cur.execute("select id from mentor")
    data=cur.fetchall()
    mentorid=[]
    for i in data:
        mentorid.append(i[0])
    return mentorid
def retrieveassign():
    connection,cur=db_connection()
    cur.execute("select * from assign")
    data=cur.fetchall()
    studentid=[]
    for i in data:
        studentid.append(i)

    #print(studentid)
    return studentid
#retrieveassign()
def retrieveassign2():
    connection,cur=db_connection()
    d=retrieveassign()
    d1=[list(ele) for ele in d]
    for k in d1:
        cur.execute("select name from student where id=%s",(k[0],))
        data=cur.fetchall()
        cur.execute("select name from mentor where id=%s",(k[1],))
        data1=cur.fetchall()

        for i,j in zip(data,data1):
            k.insert(1,i[0])
            k.insert(3,j[0])
    return d1
#retrieveassign2()

```

### **Main code:**

```

from cgitb import text

```

```

#from curses.textpad import Textbox
from tkinter import *
import tkinter as tk
from tkinter import ttk
from tkinter import filedialog as fd
from tkinter.messagebox import showinfo
from PIL import Image, ImageTk
from mentormanagement import *
from tkinter import messagebox

root = Tk()
root.title('Mentor Management')
root.resizable(True, True)
root.geometry('1500x750')
root.configure(bg='ghost white')
#root.wm_attributes('-transparentcolor', '#ab23ff')

titleframe = LabelFrame(root)
titleframe.grid(row=0,column=0)

image=Image.open("3.jpg")

img=image.resize((1500, 600))
my_img=ImageTk.PhotoImage(img)
    # Show image using label
label1 = Label( root, image = my_img)
label1.image = my_img
label1.place(x = 0, y = 100)

```

```

image=Image.open("image2.png")

img=image.resize((200, 50))
my_img=ImageTk.PhotoImage(img)
    # Show image using label
label1 = Label( root, image = my_img,bg='ghost white')
label1.image = my_img
label1.place(x = 0, y = 19)

f= ("Arial", 25, "bold")
label1      =      Label(      root,      text="MENTORING      MANAGEMENT
SYSTEM",font=f,bg="ghost white")
label1.place(x = 320, y = 10)

f= ("Times", 10, "bold")
label1 = Label( root, text="In recognition of the basic need for human
connection"+"\\n"+" a virtual mentorship program establishes deeper communication
between mentors, mentees and parents!!",font=f,bg="ghost white")
label1.place(x = 300, y =60)
def adminframe():
    def adminpage():
        def assign1():
            global lgn_frame2
            #lgn_frame3.destroy()
            #lgn_frame4.destroy()
            lgn_frame2 = Frame(lgn_frame1, bg='ghost white', width=1390, height=600)
            lgn_frame2.place(x=400, y=100)
            global sid

```



```
sid,mid=retrivestudentandmentor()
```

```
clicked1 = StringVar()
```

```
options=mid
```

```
clicked1.set( options[0] )
```

```
drop = OptionMenu( lgn_frame2 , clicked1 , *options )
```

```
drop.place(x=50, y=50)
```

```
clicked2 = StringVar()
```

```
options=["1","2","3","4","5","6","7","8","9","10","11","12","13","14","15","16","17","18","19","20"]
```

```
clicked2.set( options[0] )
```

```
drop = OptionMenu( lgn_frame2 , clicked2 , *options )
```

```
drop.place(x=200, y=50)
```

```
def assigned():
```

```
    sid1=sid[:int(clicked2.get())]
```

```
    assign(sid1,clicked1.get())
```

```
    del sid[:int(clicked2.get())]
```

```
    messagebox.showinfo("showinfo", "Successfully students assigned to mentor!")
```

```
lgn_button_label = Label(lgn_frame2, image=photo, bg='ghost white')
```

```
lgn_button_label.image = photo
```

```
lgn_button_label.place(x=100, y=100)
```

```
login = Button(lgn_button_label, text='assign', font=("yu gothic ui", 13, "bold"), width=25, bd=0,
```

```
                bg='#3047ff', cursor='hand2', activebackground='#3047ff', fg='white',command=assigned)
```

```
login.place(x=20, y=10)
```

```
def view():
```

```
    global lgn_frame3
```

```
    #lgn_frame4.destroy()
```

```
    #lgn_frame2.destroy()
```

```
    lgn_frame3 = Frame(lgn_frame1, bg='ghost white', width=1390,  
height=600)
```

```
    lgn_frame3.place(x=400, y=100)
```

```
    clicked=StringVar()
```

```
    options=["Student","Mentor","Parent"]
```

```
    clicked.set( "Student")
```

```
    drop = OptionMenu( lgn_frame3 , clicked , *options )
```

```
    drop.place(x=10, y=10)
```

```
    def clear_all():
```

```
        for item in tree.get_children():
```

```
            tree.delete(item)
```

```
    def view1():
```

```
        clear_all()
```

```
        connection,cur=db_connection()
```

```
        if clicked.get()=="Student":
```

```
            cur.execute("SELECT * FROM student")
```

```
            data=cur.fetchall()
```

```
            #data.insert(0,["ID","Name","Email","Password"])
```

```
            for student in data:
```

```
                tree.insert("", 'end', text="1", values=student)
```

```
                tree.place(x=10,y=150)
```

```

if clicked.get()=="Mentor":
    cur.execute("SELECT * FROM mentor")
    data=cur.fetchall()
    #data.insert(0,["ID","Name","Email","Password"])
    for student in data:
        tree.insert("", 'end', text="1", values=student)
        tree.place(x=10,y=150)
if clicked.get()=="Parent":
    cur.execute("SELECT * FROM parent")
    data=cur.fetchall()
    #data.insert(0,["ID","Name","Email","Password"])
    for student in data:
        tree.insert("", 'end', text="1", values=student)
        tree.place(x=10,y=150)
lgn_button_label = Label(lgn_frame3, image=photo, bg='ghost white')
lgn_button_label.image = photo
lgn_button_label.place(x=10, y=50)
login = Button(lgn_button_label, text='View', font=("yu gothic ui", 13,
"bold"), width=25, bd=0,
                bg='#3047ff', cursor='hand2', activebackground='#3047ff',
fg='white',command=view1)
login.place(x=20, y=10)

tree = ttk.Treeview(lgn_frame3, column=("ID", "Name",
"EMail","Password"), show='headings', height=600)
tree.column("# 1", anchor=CENTER)
tree.heading("# 1", text="ID")
tree.column("# 2", anchor=CENTER)
tree.heading("# 2", text="Name")

```

```

        tree.column("# 3", anchor=CENTER)
        tree.heading("# 3", text="EMail")
        tree.column("# 4", anchor=CENTER)
        tree.heading("# 4", text="Password")
def createaccounts():
    global lgn_frame4
    #lgn_frame2.destroy()
    #lgn_frame3.destroy()
    lgn_frame4 = Frame(lgn_frame1, bg='ghost white', width=1390, height=600)
    lgn_frame4.place(x=400, y=100)
    clicked = StringVar()
    name=StringVar()
    email=StringVar()
    password=StringVar()
    id=StringVar()
    def insertdata():
        if clicked.get()=='Student':
            studentregister(name.get(),email.get() ,password.get())
            messagebox.showinfo("showinfo", "Successfully account Created")
        if clicked.get()=='Mentor':
            mentorregister(name.get(),email.get() ,password.get())
            messagebox.showinfo("showinfo", "Successfully account Created")
        if clicked.get()=='Parent':
            parentregister(id.get(),name.get(),email.get() ,password.get())
            messagebox.showinfo("showinfo", "Successfully account Created")
            options=["Student","Mentor","Parent"]
    clicked.set( "Student" )

    drop = OptionMenu( lgn_frame4 , clicked , *options )

```

```
drop.place(x=200, y=100)
```

```
username_label = Label(lgn_frame4, text="ID", bg="ghost white",  
fg="#4f4e4d",
```

```
font=("yu gothic ui", 13, "bold"))
```

```
username_label.place(x=50, y=150)
```

```
username_entry = Entry(lgn_frame4, highlightthickness=0, relief=FLAT,  
bg="white", fg="#6b6a69",
```

```
font=("yu gothic ui ", 12, "bold"),textvariable=id)
```

```
username_entry.place(x=200, y=150, width=270)
```

```
username_label = Label(lgn_frame4, text="*This field only for parent",  
bg="ghost white", fg="#4f4e4d",
```

```
font=("yu gothic ui", 13, "bold"))
```

```
username_label.place(x=300, y=150)
```

```
username_label = Label(lgn_frame4, text="Name", bg="ghost white",  
fg="#4f4e4d",
```

```
font=("yu gothic ui", 13, "bold"))
```

```
username_label.place(x=50, y=200)
```

```
username_entry = Entry(lgn_frame4, highlightthickness=0, relief=FLAT,  
bg="white", fg="#6b6a69",
```

```
font=("yu gothic ui ", 12, "bold"),textvariable=name)
```

```
username_entry.place(x=200, y=200, width=270)
```

```
username_label = Label(lgn_frame4, text="Email", bg="ghost white",  
fg="#4f4e4d",
```

```

        font=("yu gothic ui", 13, "bold"))
username_label.place(x=50, y=25👁️)

username_entry = Entry(lgn_frame4, highlightthickness=0, relief=FLAT,
bg="white", fg="#6b6a69",
        font=("yu gothic ui ", 12, "bold"),textvariable=email)
username_entry.place(x=200, y=258, width=270)

username_label = Label(lgn_frame4, text="Password", bg="ghost white",
fg="#4f4e4d",
        font=("yu gothic ui", 13, "bold"))
username_label.place(x=50, y=31👁️)

username_entry = Entry(lgn_frame4, highlightthickness=0, relief=FLAT,
bg="white", fg="#6b6a69",
        font=("yu gothic ui ", 12, "bold"),textvariable=password)
username_entry.place(x=200, y=318, width=270)

lgn_button_label = Label(lgn_frame4, image=photo, bg='ghost white')
lgn_button_label.image = photo
lgn_button_label.place(x=80, y=350)
login = Button(lgn_button_label, text='Create', font=("yu gothic ui", 13,
"bold"), width=25, bd=0,
        bg='#3047ff', cursor='hand2', activebackground='#3047ff',
fg='white',command=insertdata)
login.place(x=20, y=10)
def assignstudent():
    lgn_frame8 = Frame(lgn_frame1, bg='ghost white', width=1390, height=600)
    lgn_frame8.place(x=400, y=100)

```

```

tree = ttk.Treeview(lgn_frame8, column=("Student ID", "StudentName",
"Mentor ID","Mentor Name"), show='headings', height=600)

tree.column("# 1", anchor=CENTER)
tree.heading("# 1", text="Student ID")
tree.column("# 2", anchor=CENTER)
tree.heading("# 2", text="Student Name")
tree.column("# 3", anchor=CENTER)
tree.heading("# 3", text="Mentor ID")
tree.column("# 4", anchor=CENTER)
tree.heading("# 4", text="Mentor Name")

for i in retrieveassign2():

    tree.insert("", 'end', text="1", values=i)
    tree.place(x=10,y=50)

def quit():
    lgn_frame1.destroy()

lgn_frame1 = Frame(root, bg='ghost white', width=1390, height=600)
lgn_frame1.place(x=0, y=100)
lgn_button = Image.open('images\\btn1.png')
photo = ImageTk.PhotoImage(lgn_button)
lgn_button_label = Label(lgn_frame1, image=photo, bg='ghost white')
lgn_button_label.image = photo
lgn_button_label.place(x=30, y=50)
login = Button(lgn_button_label, text='Create accounts', font=("yu gothic ui", 13,
"bold"), width=25, bd=0,
                bg='#3047ff', cursor='hand2', activebackground='#3047ff',
fg='white',command=createaccounts)
login.place(x=20, y=10)

```

```
lgn_button_label = Label(lgn_frame1, image=photo, bg='ghost white')
lgn_button_label.image = photo
lgn_button_label.place(x=30, y=100)
login = Button(lgn_button_label, text='View accounts', font=("yu gothic ui", 13,
"bold"), width=25, bd=0,
                bg='#3047ff', cursor='hand2', activebackground='#3047ff',
fg='white',command=view)
login.place(x=20, y=10)
```

```
lgn_button_label = Label(lgn_frame1, image=photo, bg='ghost white')
lgn_button_label.image = photo
lgn_button_label.place(x=30, y=150)
login = Button(lgn_button_label, text='Assign Students', font=("yu gothic ui", 13,
"bold"), width=25, bd=0,
                bg='#3047ff', cursor='hand2', activebackground='#3047ff',
fg='white',command=assign1)
login.place(x=20, y=10)
```

```
lgn_button_label = Label(lgn_frame1, image=photo, bg='ghost white')
lgn_button_label.image = photo
lgn_button_label.place(x=30, y=200)
login = Button(lgn_button_label, text='Assigned Students list', font=("yu gothic
ui", 13, "bold"), width=25, bd=0,
                bg='#3047ff', cursor='hand2', activebackground='#3047ff',
fg='white',command=assignstudent)
login.place(x=20, y=10)
```

```
lgn_button_label = Label(lgn_frame1, image=photo, bg='ghost white')
```



```

lgn_button_label.image = photo
lgn_button_label.place(x=30, y=250)
login = Button(lgn_button_label, text='Logout', font=("yu gothic ui", 13, "bold"),
width=25, bd=0,
                bg='#3047ff',      cursor='hand2',      activebackground='#3047ff',
fg='white',command=quit)
login.place(x=20, y=10)
user=StringVar()
pwd=StringVar()
def adminlogin():
    if user.get()=="admin" and pwd.get()=="admin":
        adminpage()
bg_frame = Image.open('images\\background1.png')
photo = ImageTk.PhotoImage(bg_frame)
bg_panel = Label(root, image=photo)
bg_panel.image = photo
bg_panel.place(x=0,y=100)
# ===== Login Frame =====
lgn_frame = Frame(root, bg='ghost white', width=950, height=600)
lgn_frame.place(x=200, y=100)
txt = "Admin Login"
heading = Label(lgn_frame, text=txt, font=('yu gothic ui', 25, "bold"), bg="ghost
white",
                fg='black',
                bd=5,
                relief=FLAT)
heading.place(x=80, y=30, width=300, height=30)

side_image = Image.open('images\\vector.png')

```

```

photo = ImageTk.PhotoImage(side_image)
side_image_label = Label(lgn_frame, image=photo, bg='ghost white')
side_image_label.image = photo
side_image_label.place(x=5, y=150)
sign_in_image = Image.open('images\\hyy.png')
photo = ImageTk.PhotoImage(sign_in_image)
sign_in_image_label = Label(lgn_frame, image=photo, bg='ghost white')
sign_in_image_label.image = photo
sign_in_image_label.place(x=620, y=100)

#
username_label = Label(lgn_frame, text="Username", bg="ghost white",
fg="#4f4e4d",
                        font=("yu gothic ui", 13, "bold"))
username_label.place(x=550, y=300)

username_entry = Entry(lgn_frame, highlightthickness=0, relief=FLAT, bg="ghost
white", fg="#6b6a69",
                        font=("yu gothic ui ", 12, "bold"),textvariable=user)
username_entry.place(x=580, y=335, width=270)

username_line = Canvas(lgn_frame, width=300, height=2.0, bg="#bdb9b1",
highlightthickness=0)

username_line.place(x=550, y=359)
# ===== Username icon =====
username_icon = Image.open('images\\username_icon.png')
photo = ImageTk.PhotoImage(username_icon)
username_icon_label = Label(lgn_frame, image=photo, bg='ghost white')

```

```

username_icon_label.image = photo
username_icon_label.place(x=550, y=332)

lgn_button = Image.open('images\\btn1.png')
photo = ImageTk.PhotoImage(lgn_button)
lgn_button_label = Label(lgn_frame, image=photo, bg='ghost white')
lgn_button_label.image = photo
lgn_button_label.place(x=550, y=450)
login = Button(lgn_button_label, text='LOGIN', font=("yu gothic ui", 13, "bold"),
width=25, bd=0,
                bg='#3047ff',      cursor='hand2',      activebackground='#3047ff',
fg='white',command=adminlogin)
login.place(x=20, y=10)

def studentframe():
    def studentpage():
        student_frame=Frame(root, bg='ghost white', width=1390, height=600)
        student_frame.place(x=0, y=100)
        def view1():
            student_frame1=Frame(student_frame, bg='ghost white', width=1390,
height=600)
            student_frame1.place(x=500, y=100)
            clicked=StringVar()
            options=sretrieve1()
            clicked.set( sretrieve1()[0] )

            drop = OptionMenu( student_frame1 , clicked , *options )
            drop.place(x=100, y=50)
            def fetch():

```

```

textb=Text(student_frame1,height=50,width=100)
connection,cur=db_connection()
cur.execute("select * from studentdata where id=%s",(clicked.get(),))
data=cur.fetchall()
for i in data:

    tree.insert("", 'end', text="1", values=i)
    tree.place(x=0,y=200)
tree = ttk.Treeview(student_frame1, column=("Student ID", "Student Name",
"CGPA","Attendance Percentage"), show='headings', height=600)
tree.column("# 1", anchor=CENTER)
tree.heading("# 1", text="Student ID")
tree.column("# 2", anchor=CENTER)
tree.heading("# 2", text="Student Name")
tree.column("# 3", anchor=CENTER)
tree.heading("# 3", text="CGPA")
tree.column("# 4", anchor=CENTER)
tree.heading("# 4", text="Attendance Percentage")

lgn_button = Image.open('images\\btn1.png')
photo = ImageTk.PhotoImage(lgn_button)
lgn_button_label = Label(student_frame1, image=photo, bg='#040405')
lgn_button_label.image = photo
lgn_button_label.place(x=100, y=100)
login = Button(lgn_button_label, text='Submit', font=("yu gothic ui", 13,
label1.place(x = 0, y = 100)

b= ("Arial", 10, "bold")
open_button      =      Button(root,text='Home',command=home,bd=0,bg="ghost

```

```
white",font=b,fg='dark blue')
```

```
open_button.place(x=900,y=70)
```

```
open_button = Button(root,text='Admin',command=adminframe,bd=0,bg="ghost  
white",font=b,fg='dark blue')
```

```
open_button.place(x=1000,y=70)
```

```
open_button = Button(root,text='Student',command=studentframe,bd=0,bg="ghost  
white",font=b,fg='dark blue')
```

```
open_button.place(x=1100,y=70)
```

```
open_button = Button(root,text='Mentor',command=mentorframe,bd=0,bg="ghost  
white",font=b,fg='dark blue')
```

```
open_button.place(x=1200,y=70)
```

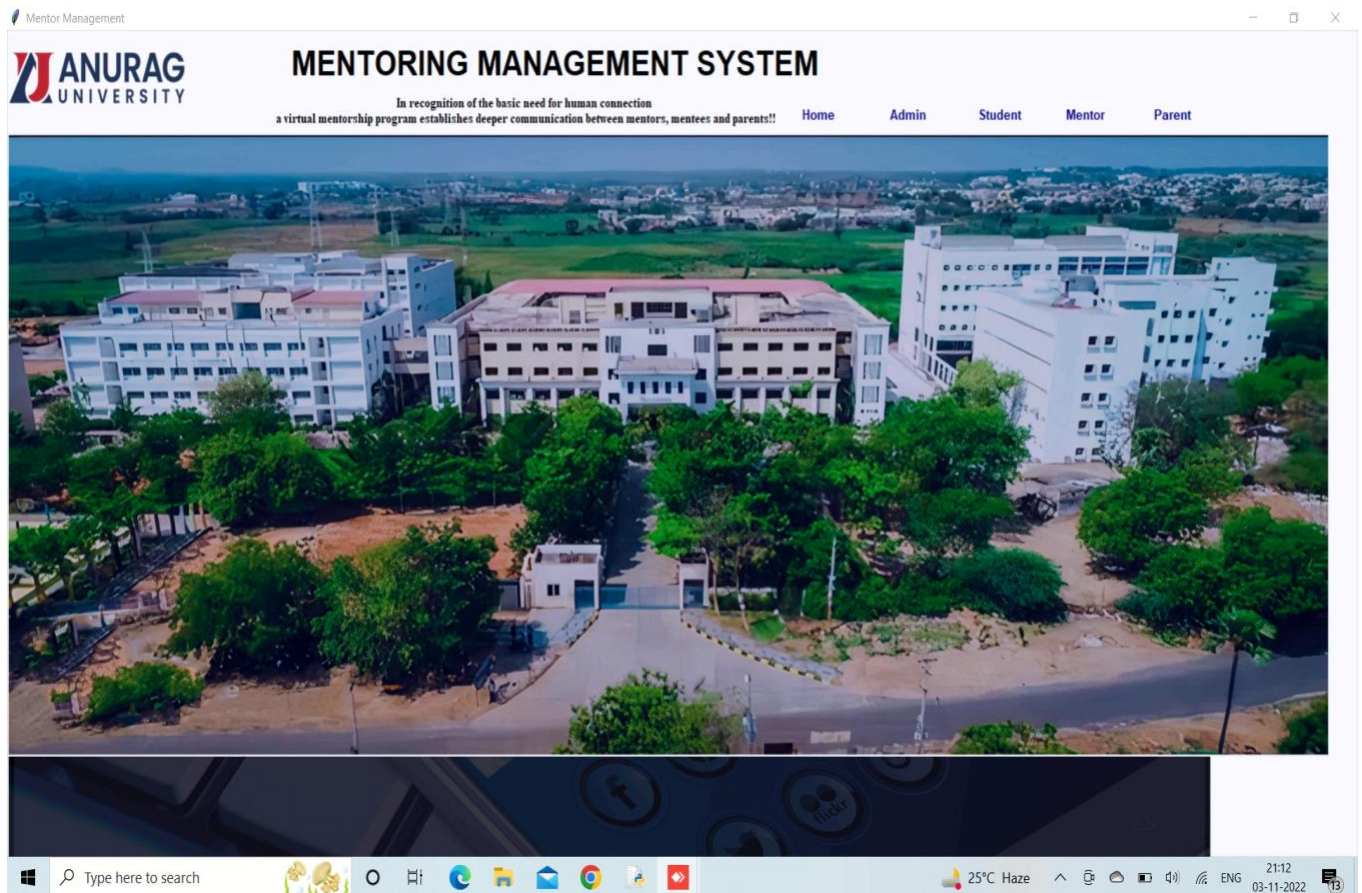
```
open_button = Button(root,text='Parent',command=parentframe,bd=0,bg="ghost  
white",font=b,fg='dark blue')
```

```
open_button.place(x=1300,y=70)
```

```
root.mainloop()
```

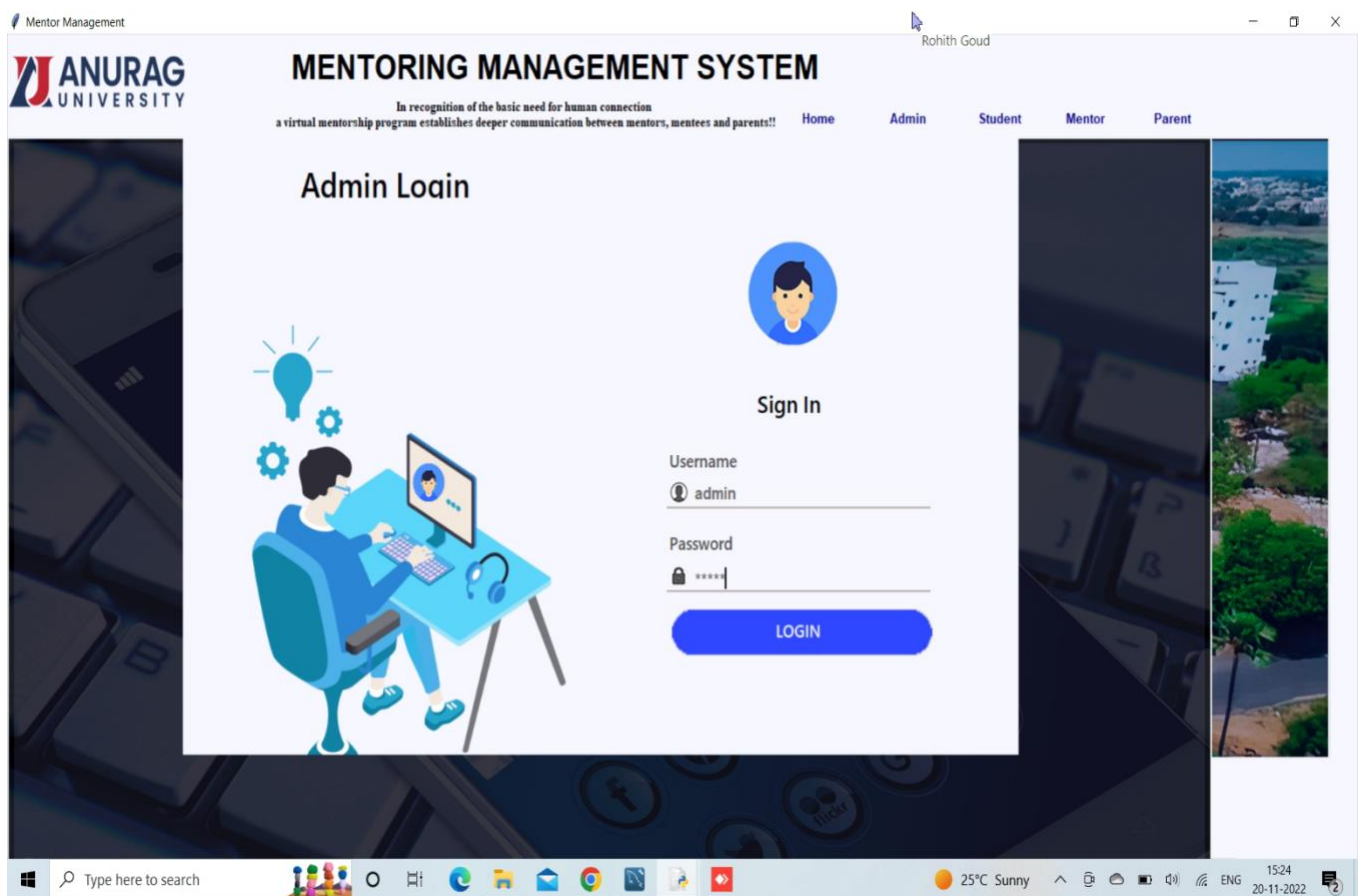
## 6. USER SCREENS

### 6.1 Main Page



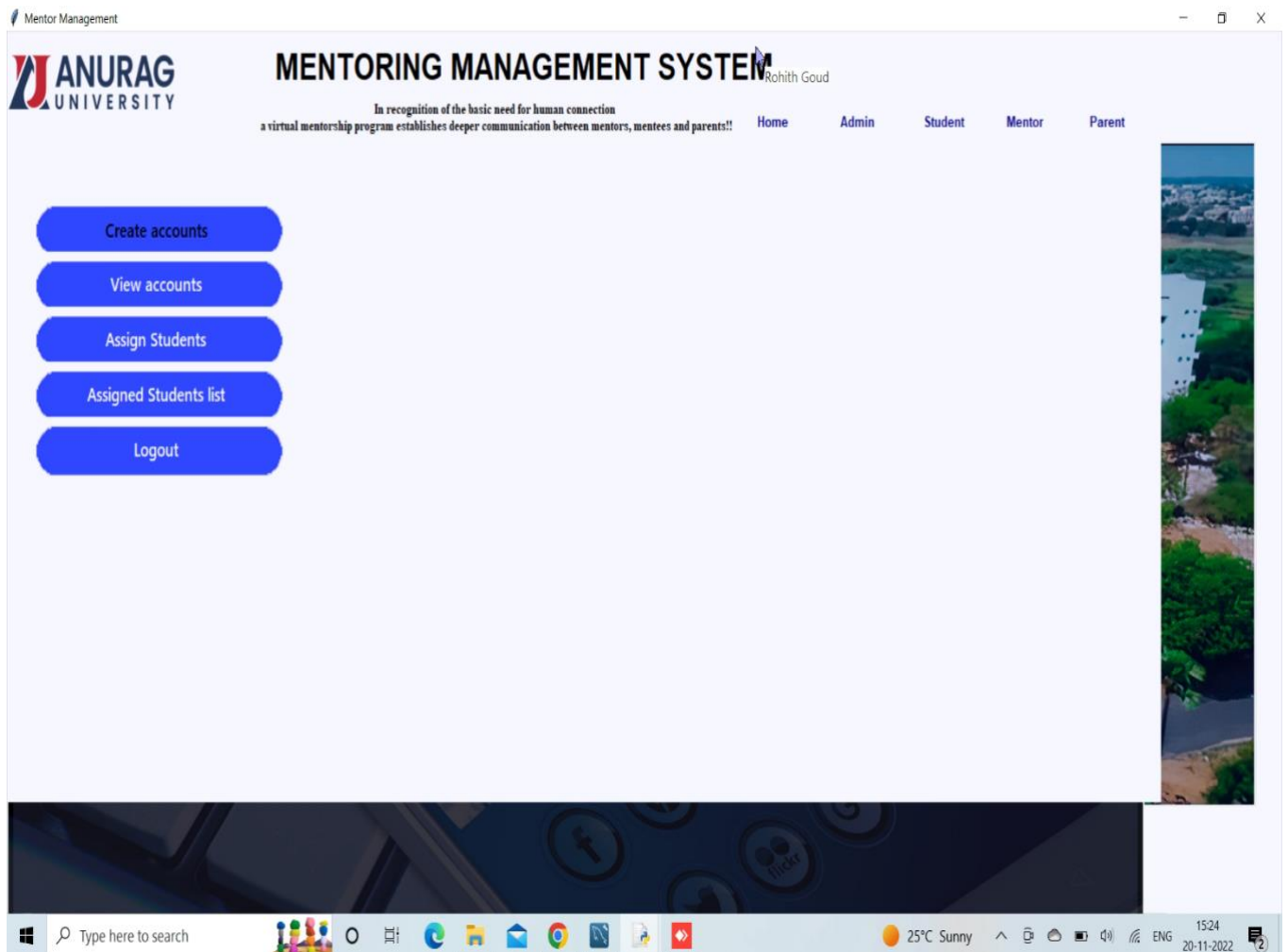
The above image shows the main page of the Mentor Management System where a student, admin, parent and mentor can login.

## 6.2 admin page:



The above image shows the login page of the admin.

### 6.3 Admin Index page:



The above image shows the admin index page.



## Create Accounts:

The screenshot displays the 'Create accounts' page of the ANURAG UNIVERSITY MENTORING MANAGEMENT SYSTEM. The page features a sidebar with navigation buttons: 'Create accounts', 'View accounts', 'Assign Students', 'Assigned Students list', and 'Logout'. The main content area includes a dropdown menu for user roles (Student, Mentor, Parent) and a form with fields for ID, Name, Email, and Password. A note states '\*This field only for parent' next to the ID field. A 'Create' button is positioned below the form. The page is titled 'MENTORING MANAGEMENT SYSTEM' and includes a tagline: 'In recognition of the basic need for human connection a virtual mentorship program establishes deeper communication between mentors, mentees and parents!!'. The top navigation bar contains links for 'Home', 'Admin', 'Student', 'Mentor', and 'Parent'. The bottom of the image shows a Windows taskbar with the date 20-11-2022 and time 15:24.

Mentor Management

**ANURAG UNIVERSITY**

**MENTORING MANAGEMENT SYSTEM**

In recognition of the basic need for human connection  
a virtual mentorship program establishes deeper communication between mentors, mentees and parents!!

Home Admin Student Mentor Parent

Create accounts

View accounts

Assign Students

Assigned Students list

Logout

ID \*This field only for parent

Name

Email

Password

Create

Student

Student

Mentor

Parent

Type here to search

25°C Sunny

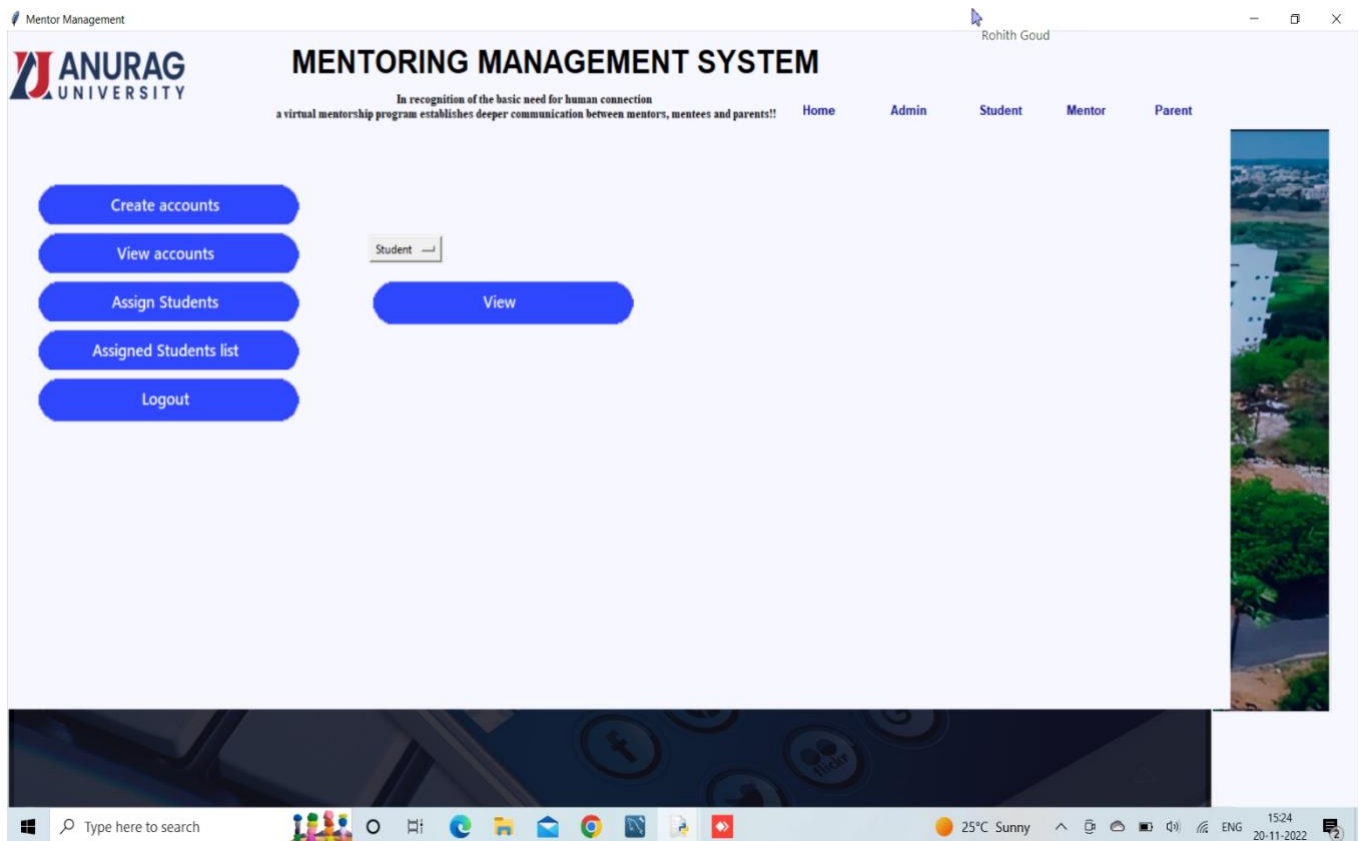
ENG

15:24

20-11-2022

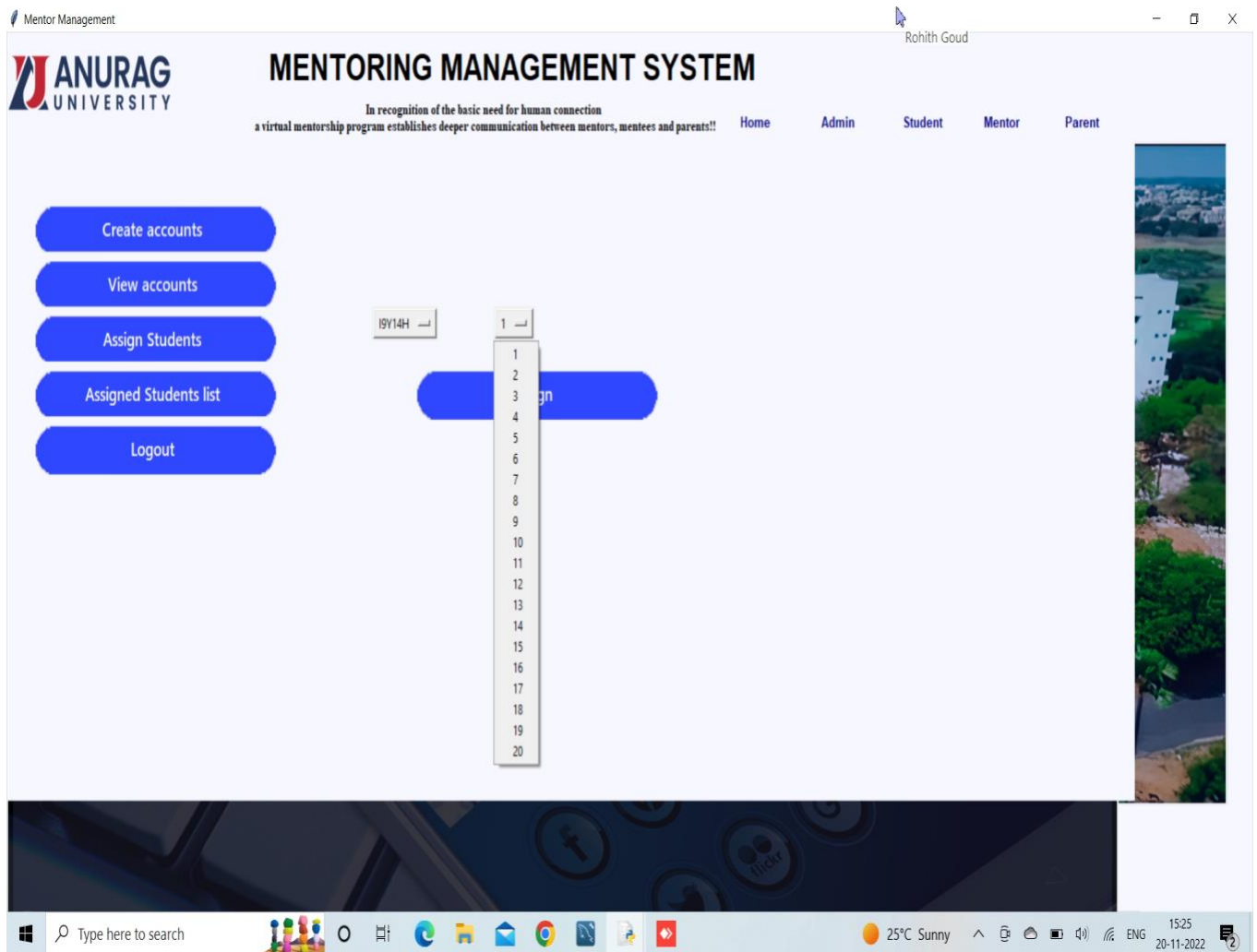
The above image shows the Create accounts interface.

## View accounts:



The above image shows the accounts created by the admin.

## Assign Students:



The above image shows how to assign students to mentors.

## Assigned students list:

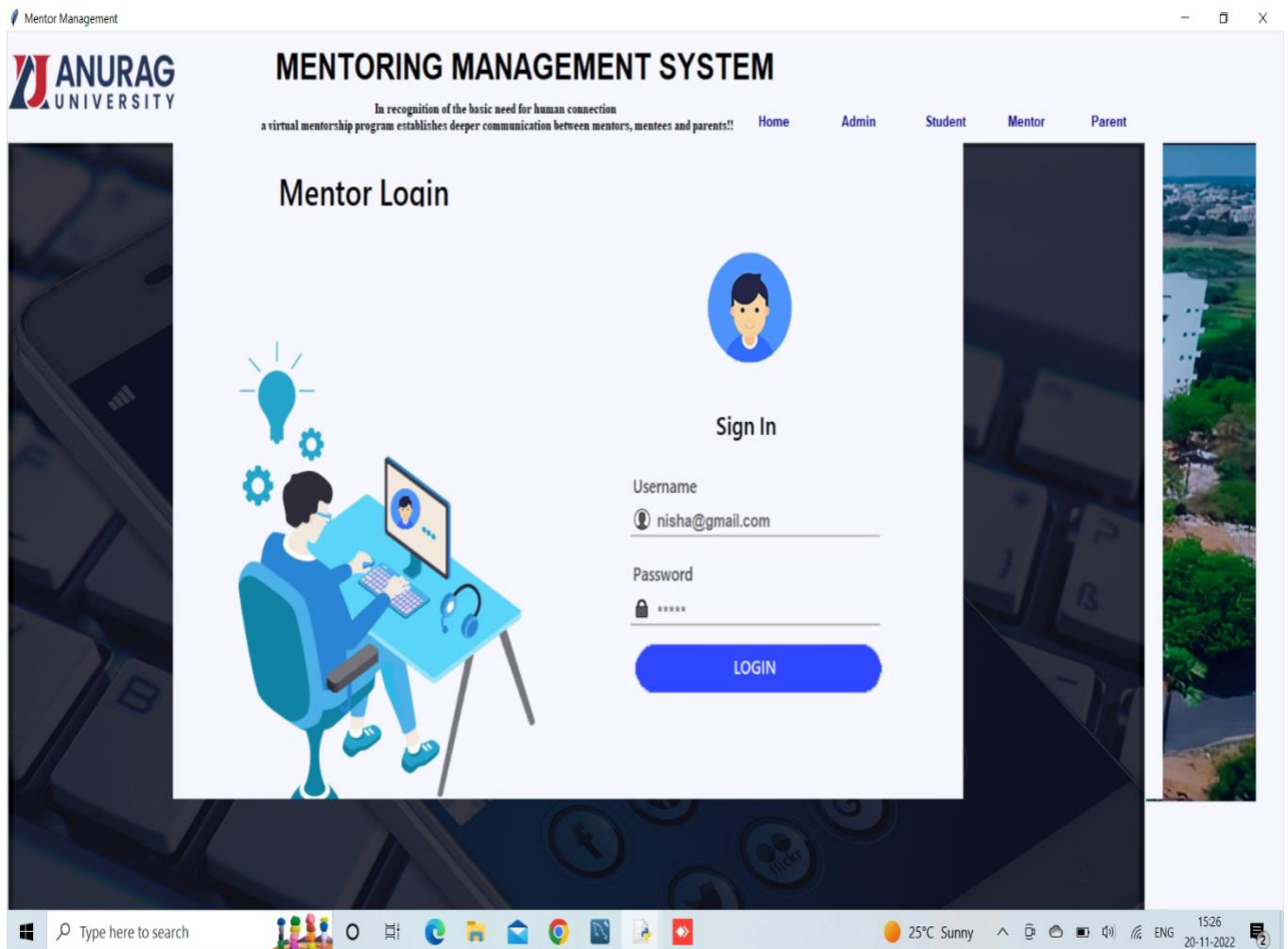
The screenshot displays the 'MENTORING MANAGEMENT SYSTEM' interface for ANURAG UNIVERSITY. The page is titled 'Assigned Students list' and shows a table of assigned students. The table has four columns: Student ID, Student Name, Mentor ID, and Mentor Name. The data is as follows:

Student ID	Student Name	Mentor ID	Mentor Name
B048QL	nithya	I9Y14H	naveen
B048QL	nithya	U96T87	lakshmi
8XVKM4	rohit	U96T87	lakshmi
B048QL	nithya	U96T87	lakshmi
8XVKM4	rohit	U96T87	lakshmi
B048QL	nithya	JL7ME5	asha
8XVKM4	rohit	JL7ME5	asha
NR8659	raja	JL7ME5	asha
B048QL	nithya	ALUMOK	nitisha
8XVKM4	rohit	ALUMOK	nitisha
NR8659	raja	ALUMOK	nitisha
QRQ5LF	tanmai	ALUMOK	nitisha
B048QL	nithya	ALUMOK	nitisha
8XVKM4	rohit	ALUMOK	nitisha
NR8659	raja	ALUMOK	nitisha
QRQ5LF	tanmai	ALUMOK	nitisha
B048QL	nithya	MQG6EO	siiram
8XVKM4	rohit	MQG6EO	siiram
NR8659	raja	MQG6EO	siiram
QRQ5LF	tanmai	MQG6EO	siiram
JFIR9V	chikku	MQG6EO	siiram

The interface includes a sidebar with navigation buttons: 'Create accounts', 'View accounts', 'Assign Students', 'Assigned Students list' (selected), and 'Logout'. The top navigation bar includes 'Home', 'Admin', 'Student', 'Mentor', and 'Parent'. The bottom of the screenshot shows a Windows taskbar with the search bar, application icons, and system tray information (25°C Sunny, 15:25, 20-11-2022).

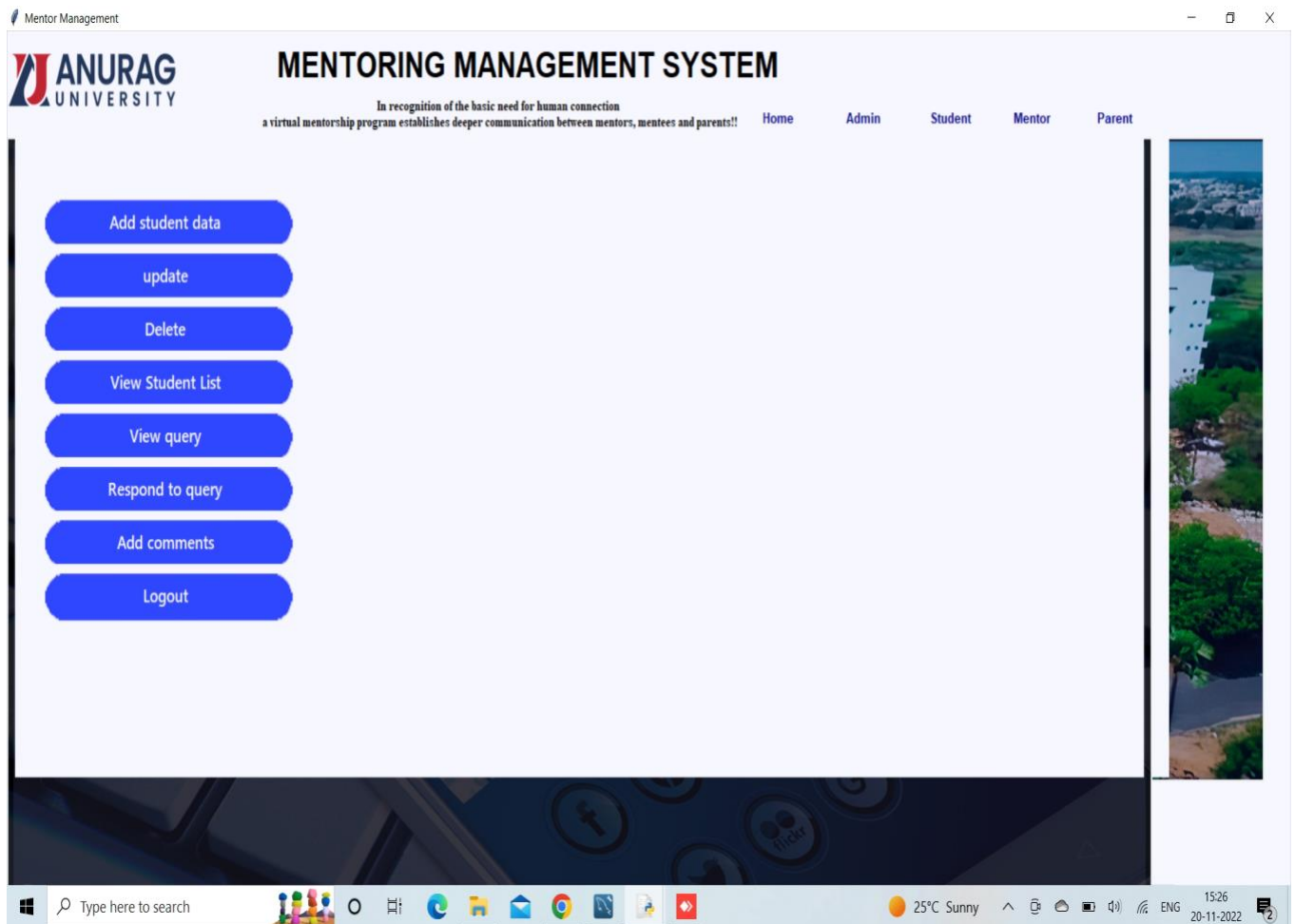
The above image shows the assigned students list.

## Mentor Login:



The above image shows the login page of the mentor.

## Mentor page:



The above image shows the index page of the mentor.

## Add student data:

The screenshot displays the 'MENTORING MANAGEMENT SYSTEM' interface for ANURAG UNIVERSITY. The page is titled 'Add student data' and features a sidebar with navigation buttons: 'Add student data', 'update', 'Delete', 'View Student List', 'View query', 'Respond to query', 'Add comments', and 'Logout'. The main content area includes a form with fields for 'Name', 'CGPA', and 'Attendance', each with a corresponding input box. A blue 'Add' button is positioned below the form. The top navigation bar includes links for 'Home', 'Admin', 'Student', 'Mentor', and 'Parent'. The user 'Rohith Goud' is logged in. The bottom of the image shows a Windows taskbar with the search bar, application icons, and system tray information indicating 25°C Sunny weather on 20-11-2022 at 15:27.

Mentor Management

ANURAG UNIVERSITY

MENTORING MANAGEMENT SYSTEM

In recognition of the basic need for human connection  
a virtual mentorship program establishes deeper communication between mentors, mentees and parents!!

Home Admin **Student** Mentor Parent

Add student data

update

Delete

View Student List

View query

Respond to query

Add comments

Logout

B048QL

Name

CGPA

Attendance

Add

Type here to search

25°C Sunny

15:27  
20-11-2022

The above image shows how to add student data page.



## Update page:

Mentor Management

**ANURAG UNIVERSITY**

# MENTORING MANAGEMENT SYSTEM

In recognition of the basic need for human connection  
a virtual mentorship program establishes deeper communication between mentors, mentees and parents!!

Home Admin Student Mentor Parent

Rohith Goud

Add student data

update

Delete

View Student List

View query

Respond to query

Add comments

Logout

B048QJL

Name

CGPA

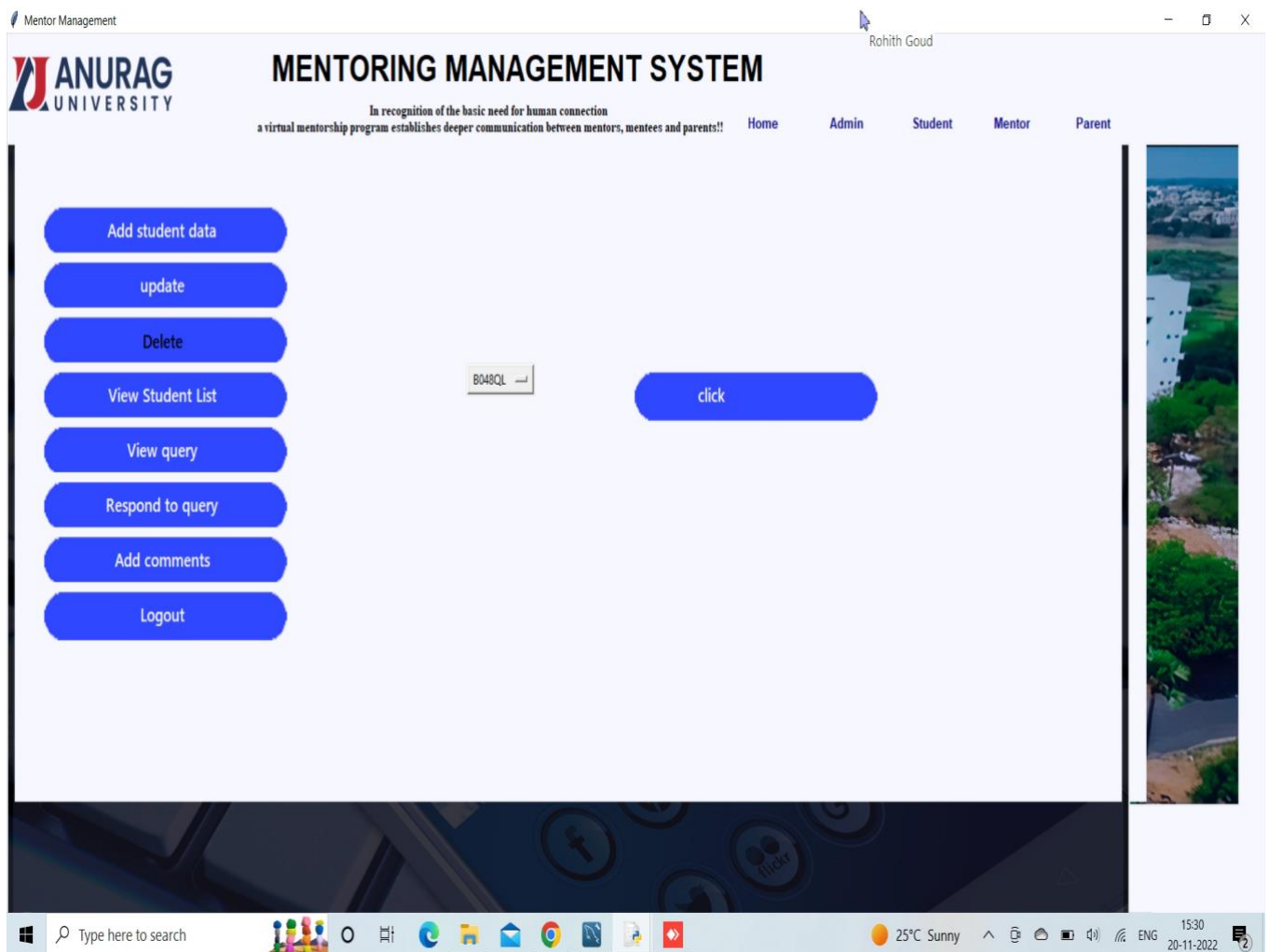
Attendance

Add

The above image shows how to update the data.



## Delete page:



The above image shows how to delete a student data.

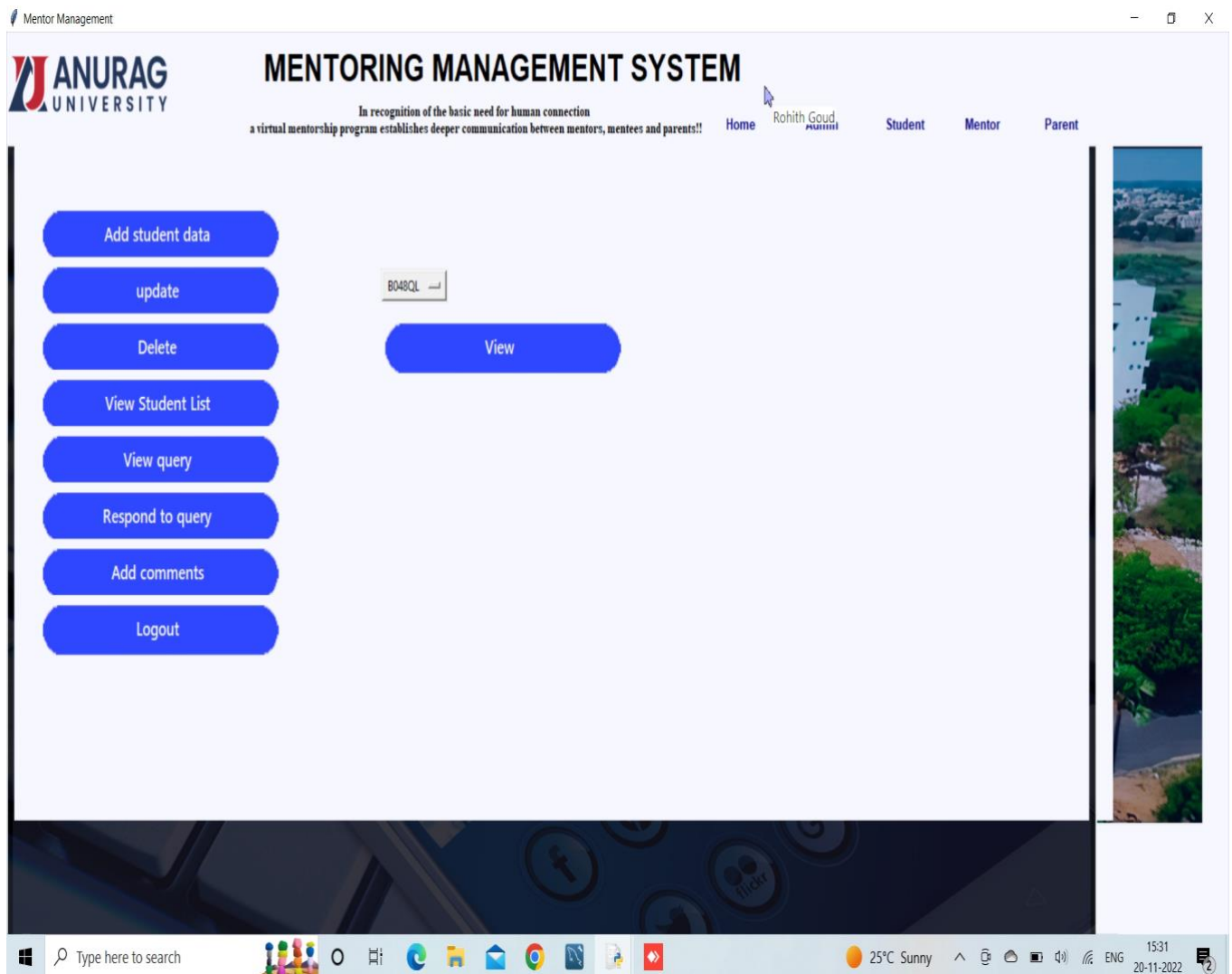
## View student list:

The screenshot displays the 'MENTORING MANAGEMENT SYSTEM' interface for ANURAG UNIVERSITY. The system is designed to facilitate communication between mentors, mentees, and parents. The interface includes a sidebar with navigation buttons: 'Add student data', 'update', 'Delete', 'View Student List', 'View query', 'Respond to query', 'Add comments', and 'Logout'. The main content area shows a table of student data with columns for ID, Name, CGPA, and Attendance Percentage. The table lists eight students with their respective IDs, names, CGPAs, and attendance percentages. The interface also features a top navigation bar with links for 'Home', 'Admin', 'Student', 'Mentor', and 'Parent'. The user 'Rohith Goud' is logged in. The bottom of the screenshot shows a Windows taskbar with various application icons and system information.

ID	Name	CGPA	Attendance Percentage
8XVKM4	rohit	9	80
NR8659	raja	9	90
QRQSLF	tanmai	8	89
JFIR9V	neha	8	90
EBBKLW	nehan	8	90
88DVNY	abhi	8.5	80
S87VZO	teja	8	60

The above image shows the student list.

## View Query:



The above image shows the query page.

## Respond to Query:

Mentor Management

Rohith Goud

**ANURAG UNIVERSITY**

# MENTORING MANAGEMENT SYSTEM

In recognition of the basic need for human connection  
a virtual mentorship program establishes deeper communication between mentors, mentees and parents!!

Home Admin Student Mentor Parent

Add student data

update

Delete

View Student List

View query

Respond to query


Add comments

Logout

Enter your ID

Answer to query

submit



Windows taskbar: Type here to search, 25°C Sunny, 15:31, 20-11-2022

The above image shows respond to query page.

## Add Comments:

Mentor Management

**ANURAG UNIVERSITY**

# MENTORING MANAGEMENT SYSTEM

In recognition of the basic need for human connection  
a virtual mentorship program establishes deeper communication between mentors, mentees and parents!!

Home Admin Student Mentor Parent

Add student data

update

Delete

View Student List

View query

Respond to query

Add comments

Logout

Enter student id

Enter your comment

submit

25°C Sunny

15:31  
20-11-2022

The above image shows add comments page.

## Student Login:

The screenshot displays a web browser window titled "Mentoring Management System" for ANURAG UNIVERSITY. The page is for "Student Login". It features a navigation bar with links for Home, Admin, Student, Mentor, and Parent. The main content area includes a "Sign In" section with fields for Username (nani@gmail.com) and Password (masked with asterisks), and a blue "LOGIN" button. To the left of the login form is an illustration of a person at a desk with a laptop, a lightbulb, and gears. The browser's address bar shows "Mentor Management" and the user's name "Rohith Goud". The Windows taskbar at the bottom shows the search bar, various application icons, and system status information including "25°C Sunny" and the date "20-11-2022".

Mentor Management

ANURAG UNIVERSITY

MENTORING MANAGEMENT SYSTEM

In recognition of the basic need for human connection  
a virtual mentorship program establishes deeper communication between mentors, mentees and parents!!

Home Admin Student Mentor Parent

Student Login

Sign In

Username  
nani@gmail.com

Password  
\*\*\*\*

LOGIN

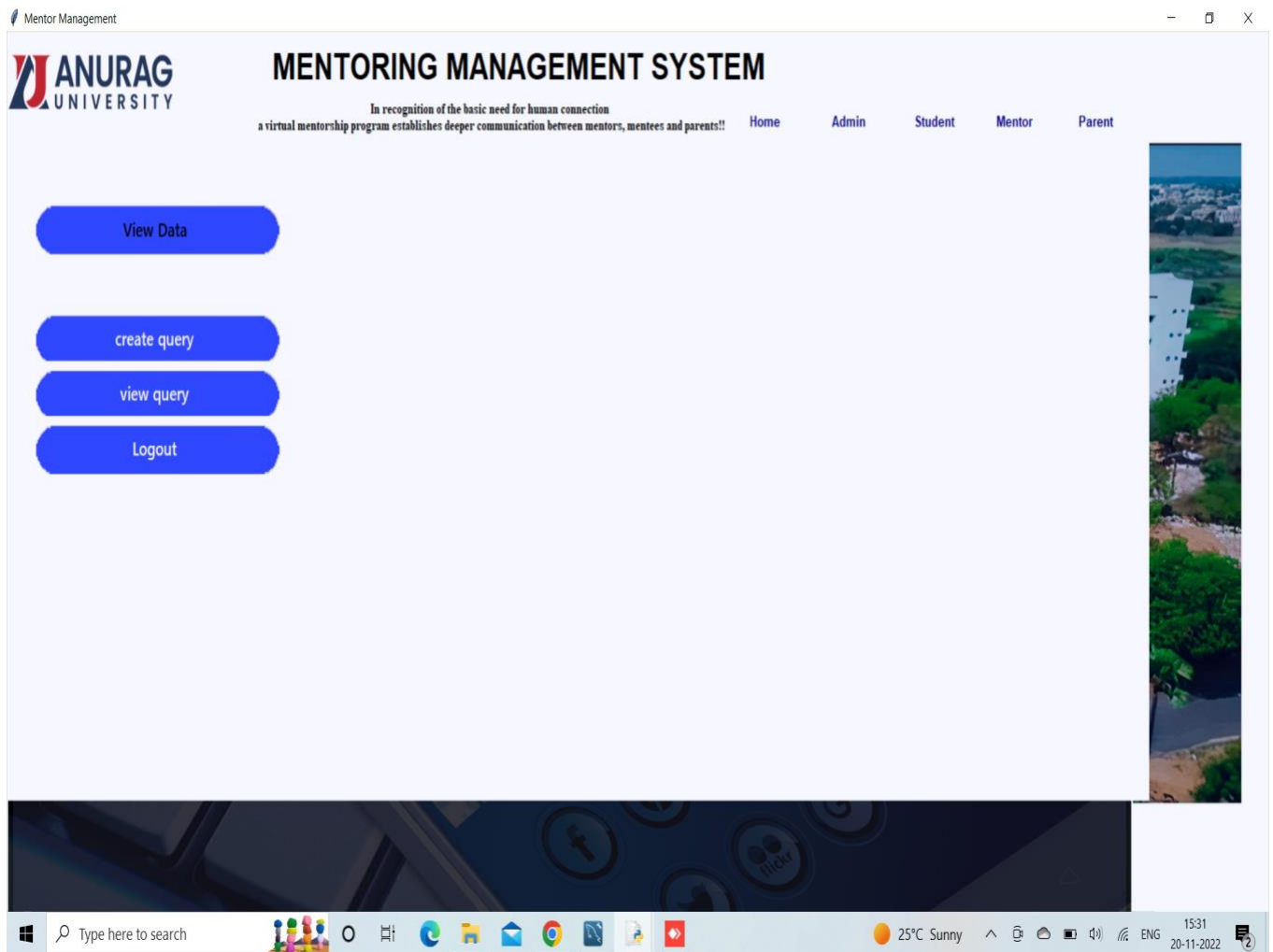
Type here to search

25°C Sunny

15:27  
20-11-2022

The above image shows the login page of the Student.

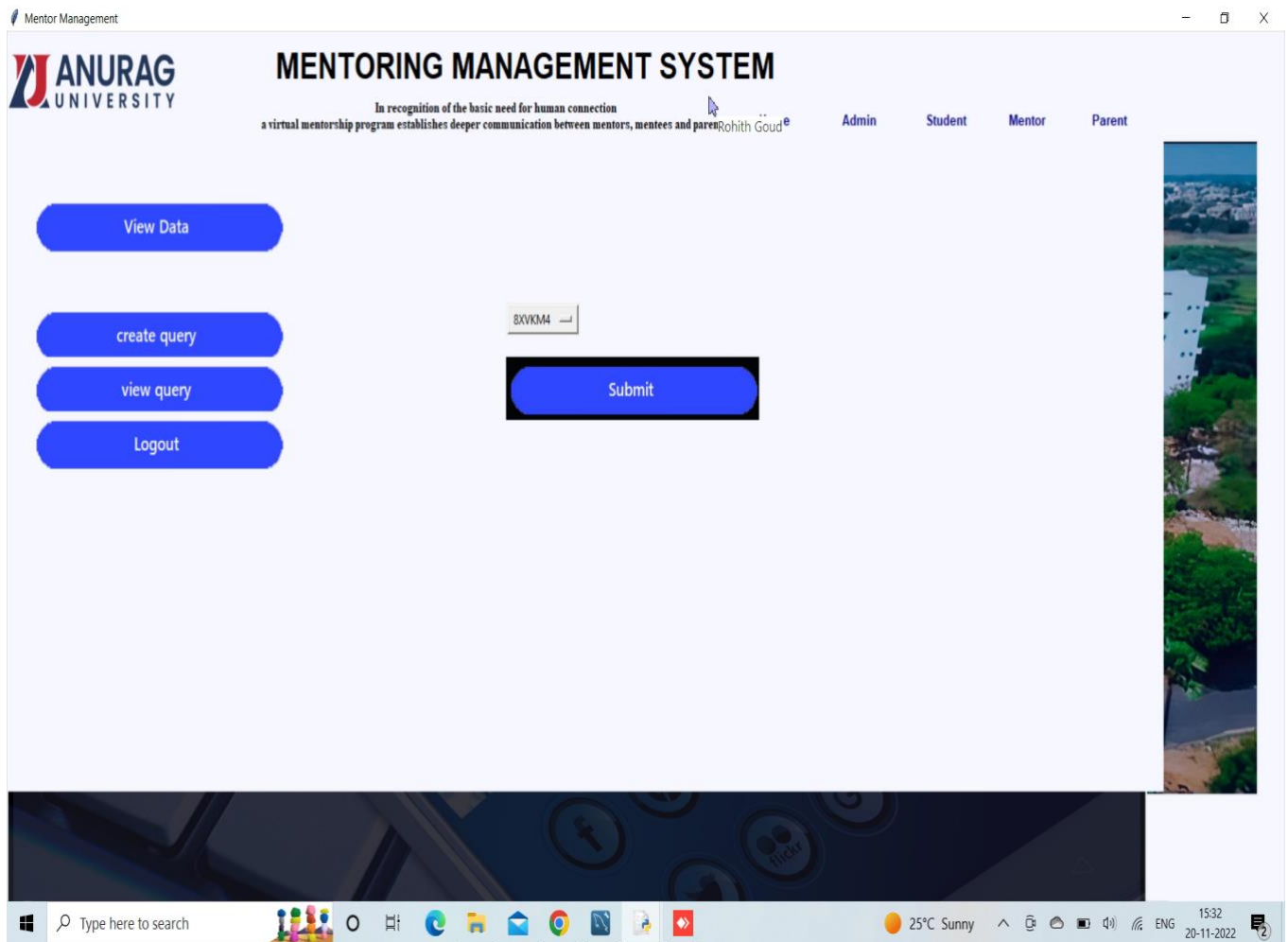
## Student Index Page:



The above image shows the index page of student.



## View Data:



The above image views the student data.



## View Data:

Mentor Management

**ANURAG UNIVERSITY**

# MENTORING MANAGEMENT SYSTEM

In recognition of the basic need for human connection  
a virtual mentorship program establishes deeper communication between mentors, mentees and parents!!

[Home](#) [Admin](#) [Student](#) [Mentor](#) [Parent](#)

Rohith Goud

View Data

create query



view query

Logout

Enter your id

Enter your query

Submit



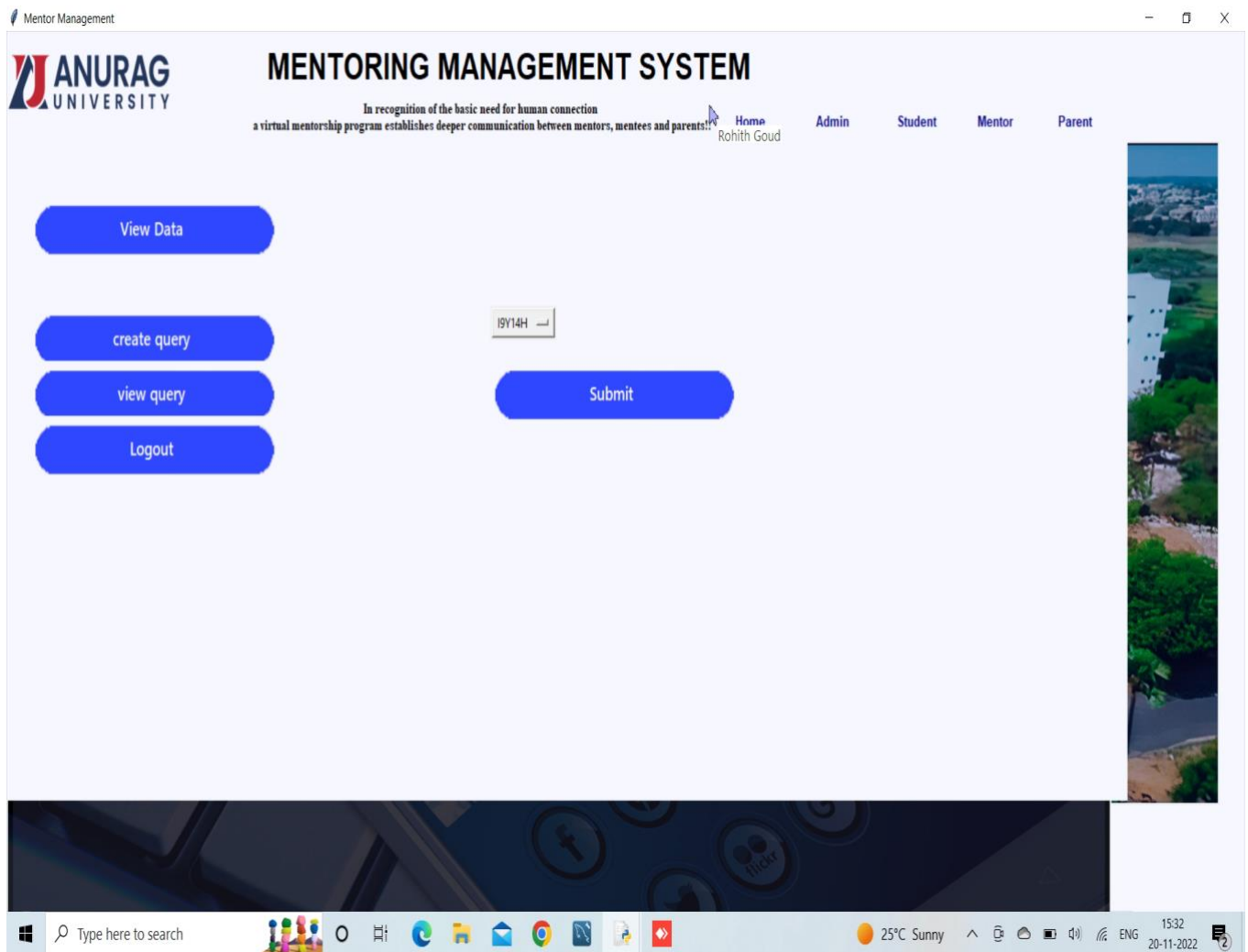
Type here to search

25°C Sunny

15:32  
20-11-2022

The above image shows how to create a query.

## View query:



The above image shows Queries sent by the mentor.

## Parent Login:

The screenshot displays the 'Parent Login' interface of the 'MENTORING MANAGEMENT SYSTEM' at Anurag University. The page features a navigation bar with links for Home, Admin, Student, Mentor, and Parent. The main content area includes a large illustration of a person working at a computer with a lightbulb icon, symbolizing ideas and mentorship. To the right, there is a 'Sign In' section with input fields for 'Username' (containing 'ramesh@gmail.com') and 'Password' (masked with '\*\*\*\*'), followed by a blue 'LOGIN' button. The page is framed by a dark blue sidebar on the left and a vertical image of a building on the right. The Windows taskbar at the bottom shows the date as 20-11-2022 and the time as 15:32.

Mentor Management

ANURAG UNIVERSITY

# MENTORING MANAGEMENT SYSTEM

In recognition of the basic need for human connection  
a virtual mentorship program establishes deeper communication between mentors, mentees and parents!!

Home Admin Student Mentor Parent

## Parent Login

Sign In

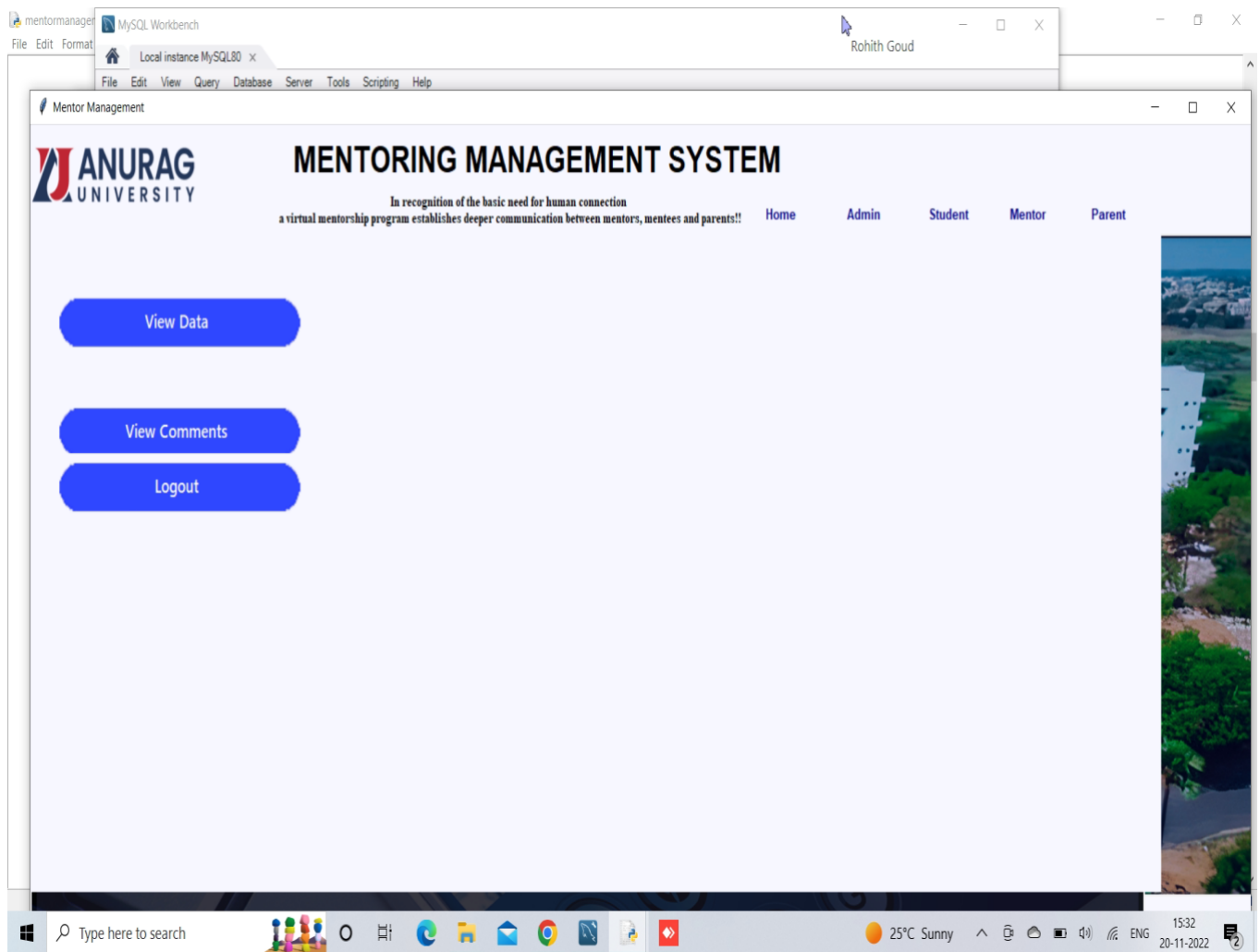
Username  
ramesh@gmail.com

Password  
\*\*\*\*

LOGIN

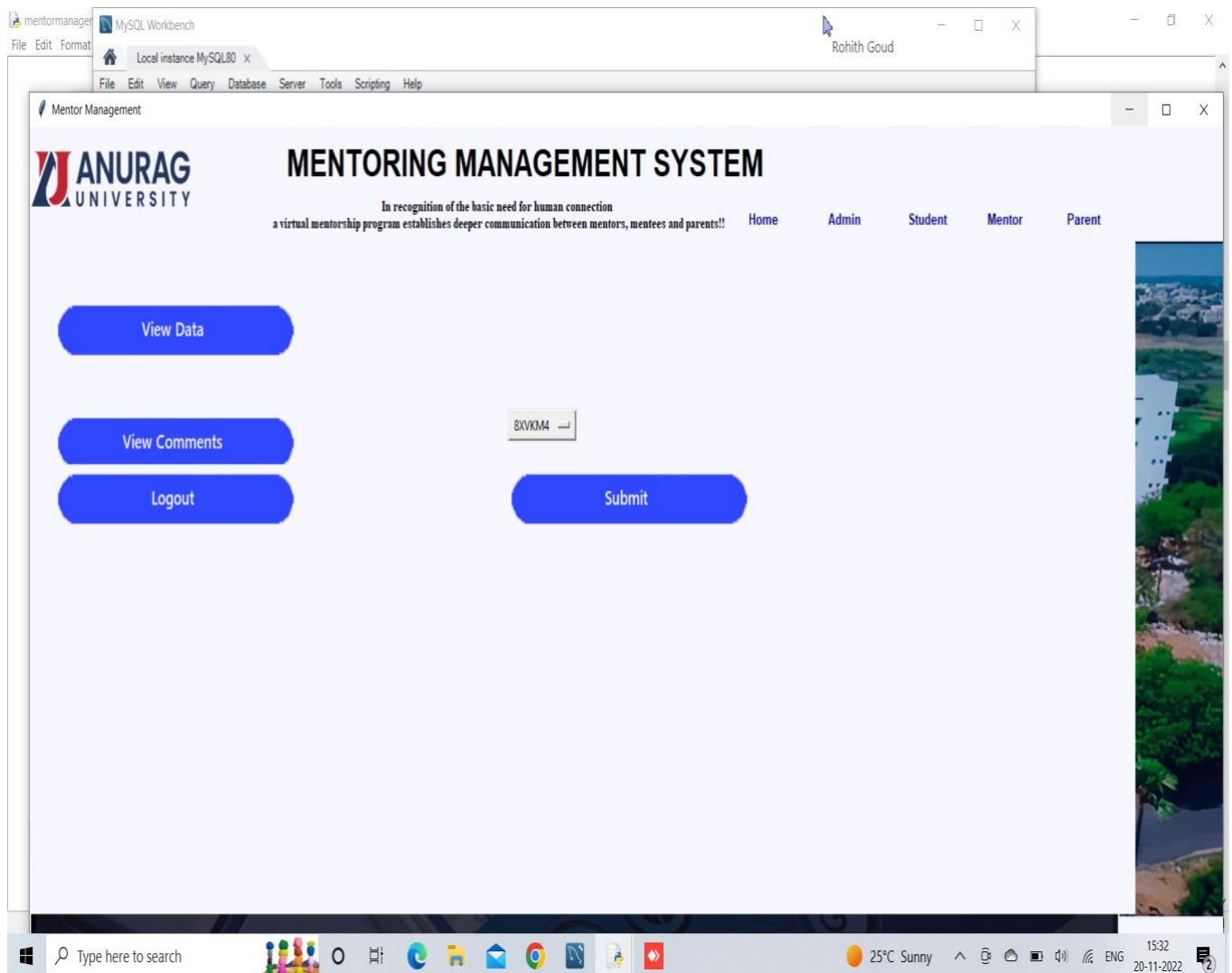
The above image shows the login page of the Parent.

## Parent Index Page:



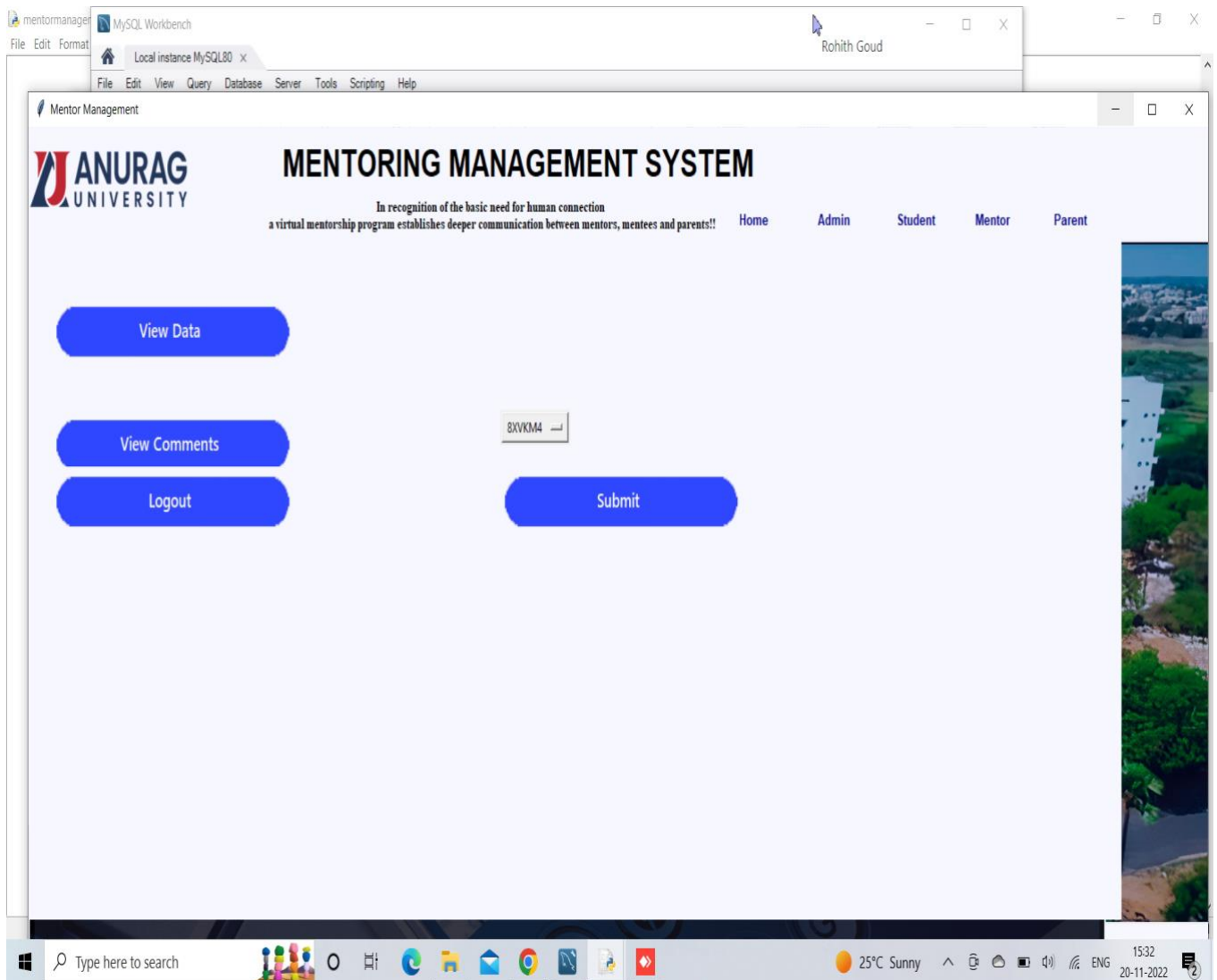
The above image shows the index page of the parent.

## View Data:



The above image shows the details of student.

## View Comments:



The above image shows the Comments.

## 4. TESTING METHODOLOGY

In this system we are using waterfall model to apply these ideas. In this each step is separated and when we finish one phase the output it is given input to the next phase. Also, backward approach can be implemented if there is a new requirement or to apply any update.

### 7.1. SOFTWARE TESTING

Software testing is the process of evaluating a software item to detect differences between given input and expected output. Testing assesses the quality of the product. Software testing is a process that should be done during the development process. In other words, software testing is a verification and validation process.

**Verification** is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

**Validation** is the process to make sure the product satisfies the specified requirements at the end of the development phase. In other words, to make sure the product is built as per customer requirements.

#### Basics of software testing

There are two basics of software testing: Black box testing and white box testing.

##### Black box Testing

Black box testing is a testing technique that ignores the internal mechanism of the system and focuses on the output generated against any input and execution of the system. It is also called functional testing.

##### White box Testing

White box testing is a testing technique that takes into account the internal mechanism of a system. It is also called structural testing and glass box testing. Black box testing is often used for validation and white box testing is often used for verification.

#### 7.1.1 TYPES OF TESTING

There are different types of testing

- Unit Testing
- Integration Testing

- Functional Testing
- System Testing
- Regression Testing etc.

### **Unit Testing**

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

### **Integration Testing**

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

### **Functional Testing**

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

### **System Testing**

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

### **Regression Testing**

Regression testing is the testing after modification of a system, component, or a group of related units to ensure that the modification is working correctly and is not damaging or imposing other modules to produce unexpected results. It falls under the class of black box testing.



## 7.2 TEST CASES

SNO	Test Case	Pass	Fail
1	Admin should enter correct login credentials.	Enters into Admin Index Page.	Redirects to Admin login page.
2	Mentor should enter correct login credentials.	Enters into Mentor Index page.	Redirects to Mentor login page.
3	Enter email, password and press submit button.	It should navigate to respective home pages based on credentials entered.	It will redirect to the same page if given wrong.
4	Selecting a Category in Index Pages	Visualizations will be generated based on the category.	Category must be selected correctly for viewing the data.
5	Data should be added manually before clicking submit button.	Data will be uploaded.	If not it will ask to check the data
6	After clicking on submit button in add user section.	An screen is displayed saying 'New record added!'	Redirects to the same page until you enter correct details.
7	In the bottom of every index page there is an option to logout .	It goes back to the login page .	If not remains in the same page .
8	Once you add the add details and submit. Click on view data.	You will observe the data just now entered in the table	If not added correctly it won't be displayed.

## 8. CONCLUSION

Mentoring is an important aspect in everyone's life. It helps one talk to someone about their problems as well as get a guide on how to go about a situation. However, some people shy off speaking about their issues to someone in person. The system that has been developed will give mentees the chance to be in touch with their mentors regardless of their location. It is also supposed to encourage students through the blogs that mentors post. This system will however need the user to have a stable internet connection in order to have their sessions.

### **Future Enhancements:**

Our further enhancement is develop this application in IOS and Android platform and planning to accessed as a offline app. We are also planning to add more features like video / audio communication through our application and sending student details to parents via Email and messages frequently.

## 9. REFERENCES

- [1] Database System Concepts, Tata Mc Graw- Hill.
- [2] Database Systems Design, Implementation and Management.
- [3] Database Systems Design, Implementation and Management.
- [4] <https://www.w3schools.com/>
- [5] Soharuni.edu.om.(2018).[online]Available  
at: <https://www.soharuni.edu.om/sualumni/>.
- [6] User, S. (2019). Alumni - Majan University College. [online] Majancollege.edu.om. Available at:  
<https://www.majancollege.edu.om/the-alumni>
- [7] Python-<https://blog.bitsrc.io/using-python-for-frontend-f90a0f8d51ae>
- [8] [javatpoint.com/python-tkinter](http://javatpoint.com/python-tkinter)