University Management System

End Semester Design Capstone I Report

by

Aneesha Mishra (UCSE21006)
Smrutirekha Panigrahi (UCSE21033)
Ranojay Sikdar (UCSE21048)

Under the supervision
of

Dr. Sourav Mondal

Design Capstone I: CSP501

School of Computer Science & Engineering
XIM University

1st December, 2023

1 Abstract

This project presents a university portal system that uses Python and Tkinter to create a user-friendly interface. The system allows users to log-in with their ID and password and then access a dashboard specific to their role (student, teacher, or admin). The student dashboard allows students to view their information, courses, grades, and notice boards. The teacher dashboard allows teachers to view their information, courses, grades, and notice boards. The admin dashboard allows admins to add students, add teachers, and add the notice board.

The system was implemented using MySQL as the database management system. The database stores user information, such as their ID, password, role, and other relevant data. The Python program uses Tkinter to create the login page and the dashboards. The program also uses the database to check the user's credentials and to determine which dashboard to open.

The system was tested with a small group of users and was found to be easy to use and functional. The system has the potential to be used by universities to provide a centralized platform for students, teachers, and admins to access their information and resources.

2 Introduction

A university portal system is a web-based application that provides students, teachers, and administrators with centralized access to university information and resources. The portal typically includes features such as:

- Login and registration
- Student information
- Course information
- Notice board

The university portal can be a valuable tool for students, teachers, and administrators. It can help to improve communication and collaboration, and it can make it easier to access information and resources. In this project, we will develop a university portal system using Python and Tkinter. The system will allow users to log-in with their ID and password and then access a dashboard specific to their role (student, teacher, or admin). The student dashboard will allow students to view their information, courses, grades, and notice boards. The teacher dashboard will allow teachers to view their information, courses, grades, and notice boards. The admin dashboard will allow admins to add students, add teachers, and view the notice board. The system will be implemented using MySQL as the database management system. The database will store user information, such as their ID, password, role, and other relevant data. The Python program will use Tkinter to create the login page and the dashboards. The program will also use the database to check the user's credentials and to determine which dashboard to open. The system will be tested with a small group of users and will be evaluated for its ease of use and functionality. The system will have the potential to be used by universities to provide a centralized platform for students, teachers, and admins to access their information and resources.

2.1 Project objectives

- To develop a user-friendly interface that is easy to use for students, teachers, and admins.
- To implement the system using MySQL as the database management system.
- To test the system with a small group of users and evaluate its ease of use and functionality.
- To deploy the system to a university and make it available to students, teachers, and admins.

2.2 Background of the project

The background of this project is the need for a centralized platform for students, teachers, and administrators to access university information and resources. Traditionally, students, teachers, and administrators have had to access university information and resources through a variety of different systems and websites. This can be inconvenient and time-consuming, and it can also lead to errors. A university portal system can provide a centralized platform for students, teachers, and administrators to access all of the information and resources they need. This can make it easier for them to stay organized and informed, and it can also help them to save time. In addition, a university portal system can help to improve communication and collaboration between students, teachers, and administrators. This can be beneficial for the learning process, and it can also help to create a more cohesive and supportive university community. The background of your project is also the fact that Python and Tkinter are good choices for developing a university portal system. Python is a powerful and versatile programming language that is easy to learn and use. Tkinter is a Python library that makes it easy to create graphical user interfaces (GUIs).

Together, Python and Tkinter can be used to create a user-friendly and functional university portal system.

The **solution** to the problem of a lack of a centralized platform for students, teachers, and administrators to access university information and resources is to develop a university portal system.

The **scope** of the project is to develop a university portal system using Python and Tkinter.

2.3 Operation Environment

- A database server (MySQL server)
- A programming language (Python)
- A graphical user interface (GUI) library (Tkinter)

3 System Analysis

3.1 Software requirement specification:

- PC Processor 10th Gen Intel(R) Core (TM) i3-10100F (8GB RAM)
- 32-bit OS and 64-bit OS

3.2 Software tools used:

- Visual Studio Code
- MySQL Workbench
- Figma

4 System Design

4.1 Table Design

- student: Stores information about students, such as their ID, name, department, email, phone number, and photo.

• instructor: Stores information about instructors, such as their ID, name, department, email, and salary.

• department: Stores information about departments, such as their name, building, and floor.

• course: Stores information about courses, such as their ID, title, department, and credits.

• teaches: Stores information about the courses that an instructor teaches, such as the instructor's ID, the course ID, the year, and the semester.

• grades: Stores information about the grades that students have received in courses, such as the student's ID, the course ID, the mid-semester grade, the end-semester grade, the assignment grade, the practical grade, and the quiz grade.

• takes: Stores information about the courses that students have taken, such as the student's ID, the course ID, the semester, the year, and the grade.

s_id	f_name	l_name	dept_name	email	phone
xus19002	Benjamin	Garcia	Bachelor of Law	xus19002@stu.xim.edu.in	9606334774
xus19004	Grace	Brown	Bachelor of Law	xus19004@stu.xim.edu.in	9789119035
xus19006	Jessica	Garcia	Bachelor of Law	xus19006@stu.xim.edu.in	9809210860

Figure 1: Student Table

i_id	f_name	l_name	dept_name	email	salary
xuf004	Bryce	Jones	Bachelor of Law	xuf004@xim.edu.in	60000
xuf006	Theodore	Walker	Computer Science and Engineering	xuf006@xim.edu.in	65000
xuf008	Rachel	Gonzalez	Computer Science and Engineering	xuf008@xim.edu.in	70000

Figure 2: Instructor Table

a_id	f_name	l_name	dept_name	email	phone	salary
xua005	Luna	Williams	Bachelor of Law	xua005@xim.edu.in	9949981263	32000
xua006	Madison	Johnson	Bachelor of Arts in Economics	xua006@xim.edu.in	9687137328	29000
xua007	Giovanni	Jones	Computer Science and Engineering	xua007@xim.edu.in	9976756760	28000
xua008	Brooklyn	Miller	Bachelor Of Commerce	xua008@xim.edu.in	9942418176	24000

Figure 3: Admin Table

dept_name	building	floor
Bachelor of Arts in Economics	New Academic Block	2
Bachelor Of Commerce	New Academic Block	2
Bachelor of Law	Old Academic Block	1
Computer Science and Engineering	New Academic Block	3

Figure 4: Department Table

course_id	title	dept_name	credits
C1154	Legal Method	Bachelor of Law	3
C1174	Indian Economics	Bachelor of Arts in Economics	4
C1359	Public Administration	Bachelor of Law	4
C1541	Classic Shakesperean Plays	Bachelor of Arts in Economics	4

Figure 5: Course Table

t_id	i_id	course_id	year	semester
t001	xuf069	C5454	2001	1
t002	xuf006	C3682	2001	2
t003	xuf035	C4303	2002	3
t004	xuf006	C3545	2002	4

Figure 6: Teaches Table

	s_id	t_id	grades
▶	xus19002	t021	O
	xus19002	t022	A
	xus19002	t023	A
	xus19002	t024	C

Figure 7: Enrollment Table

	id	password	changing_permission
▶	xua005	25c2c9afdd83b8d34234aa2881cc341c09689aaa	1
	xua006	25c2c9afdd83b8d34234aa2881cc341c09689aaa	1
	xua007	25c2c9afdd83b8d34234aa2881cc341c09689aaa	1
	xua008	25c2c9afdd83b8d34234aa2881cc341c09689aaa	1

Figure 8: Login Table

	n_id	n_date	n_heading	n_content	n_from	n_designation
▶	1	15/11/2023	Inter-College Coding Competition	This is to inform all students that our school is organizing an Inter-College Coding C...	Ms. Giovanni Jones	Admin of School of Computer Science and Engin...
	2	30/10/2023	Inter-School Moot Court Competition	This is to inform all the students of our school that we are organizing an Inter-Scho...	Ms. Luna Williams	Admin of School of Law

Figure 9: Notice Table

5 Data Flow Diagram

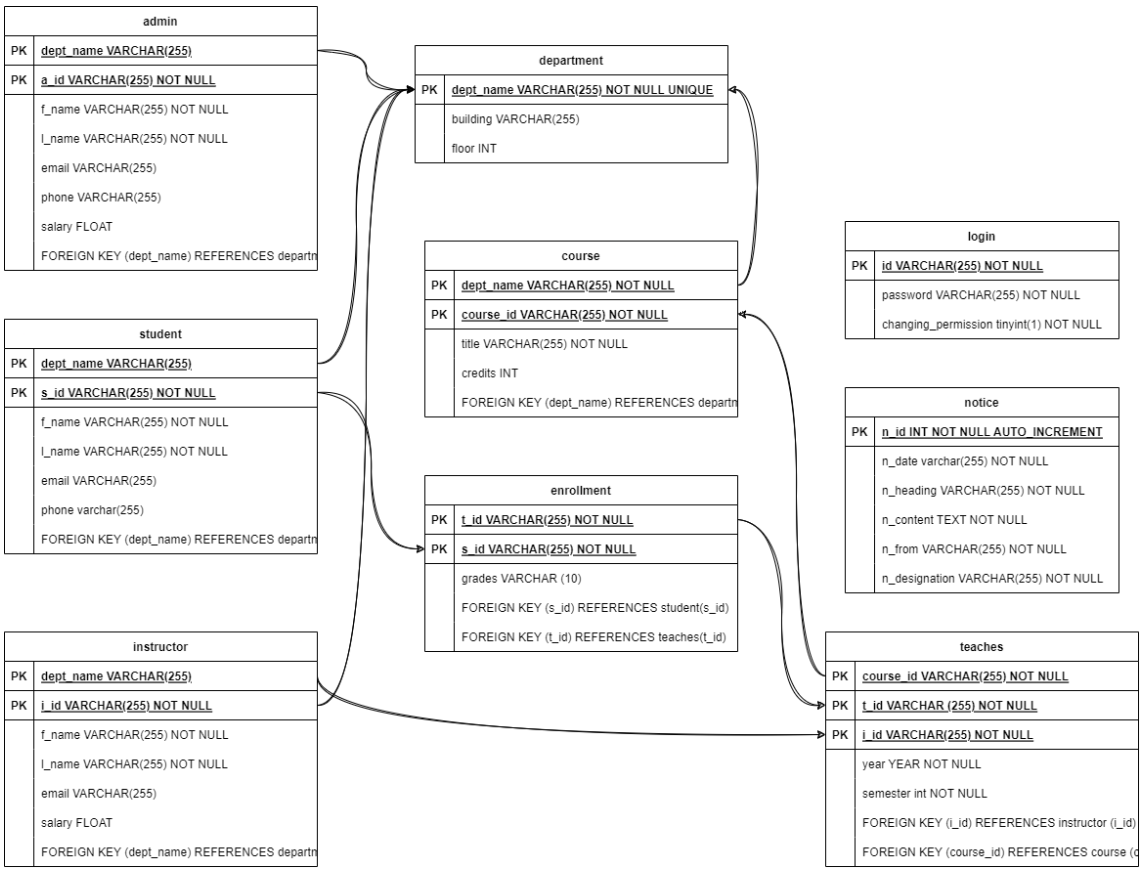


Figure 10: Data Flow Diagram

6 System Implementation

6.1 Module Description

Module 1: Introduction to the University Portal

This module will introduce the University Portal, its goals, and its scope. It will also provide an overview of the project's methodology and timeline.

Module 2: Design and Implementation of the GUI

This module will discuss the design and implementation of the GUI for the University Portal. It will cover the use of Python and Tkinter to create the GUI, as well as the design of the different dashboards for students, teachers, and administrators.

Module 3: Database Design

This module will discuss the database design for the University Portal. It will cover the creation of the database tables and the relationships between them.

Module 4: Data Entry and Validation

This module will discuss the data entry and validation procedures for the University Portal. It will cover the methods that will be used to enter data into the database, as well as the validation checks that will be performed to ensure the accuracy of the data.

Module 5: Security

This module will discuss the security measures that will be implemented in the University Portal. It will cover the use of passwords, encryption, and other security measures to protect the data in the portal.

Module 6: Testing and Deployment

This module will discuss the testing and deployment procedures for the University Portal. It will cover the methods that will be used to test the portal for functionality and errors, as well as the procedures that will be used to deploy the portal to production.

Module 7: Conclusion

This module will conclude the project by summarizing the key findings and recommendations. It will also discuss the limitations of the project and the future work that could be done. This is a possible module description, and the specific modules included in your project will depend on the nature of the project. However, these are some of the key modules that are often included in projects that involve the development of a GUI-based application.

7 Screenshot

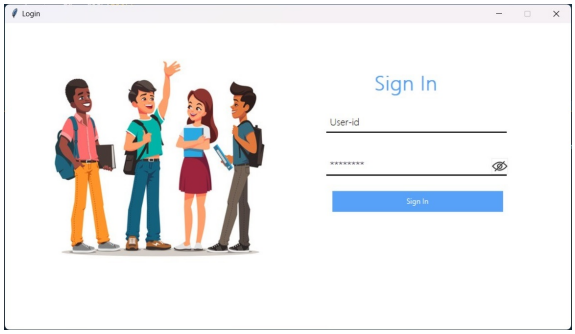


Figure 11: Login Window

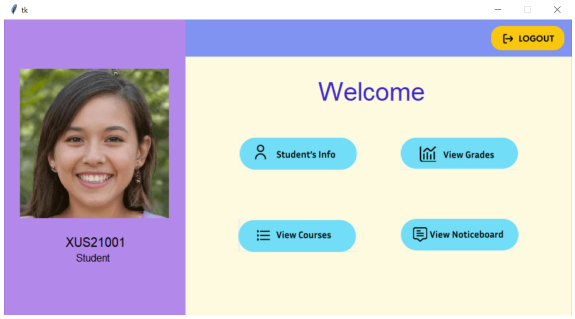


Figure 12: Student's Dashboard

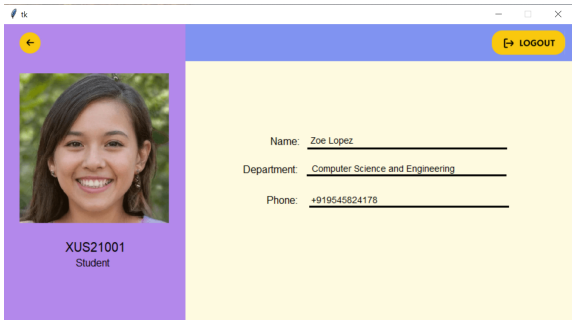


Figure 13: Student's Information

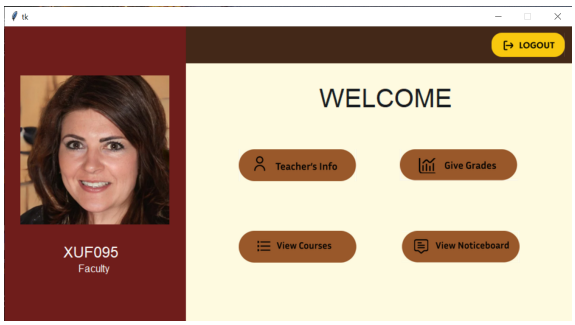


Figure 14: Teacher's Dashboard

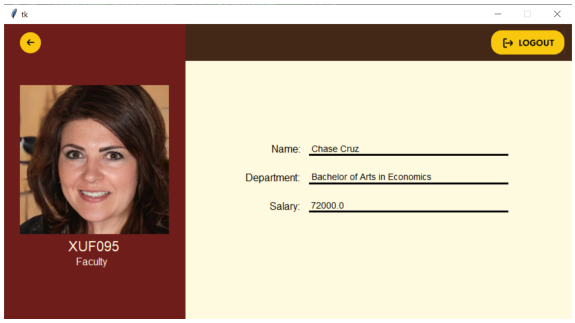


Figure 15: Teacher's Information

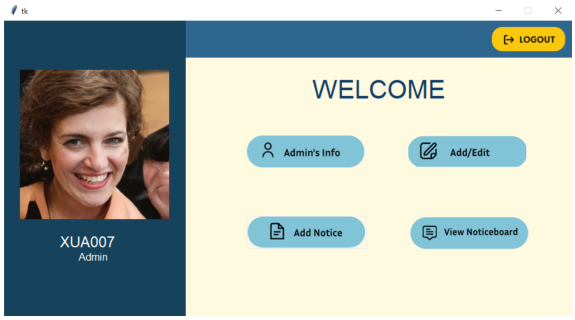


Figure 16: Admin's Dashboard

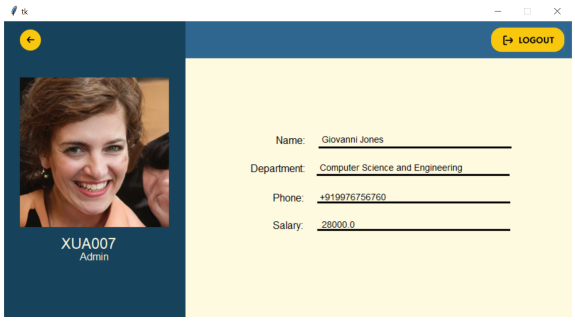


Figure 17: Admin's Information

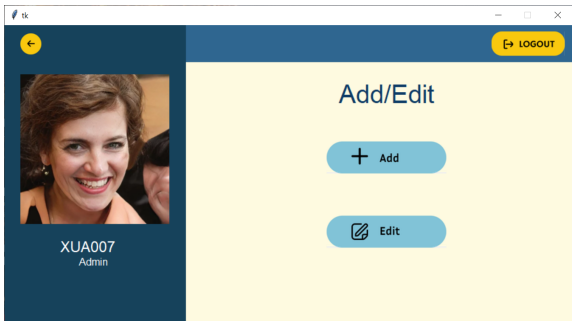


Figure 18: Add/Edit

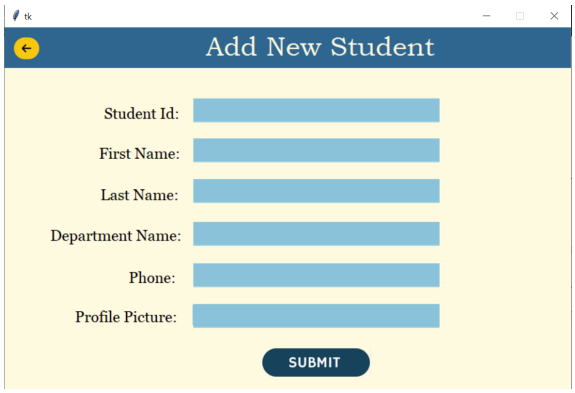


Figure 19: Add

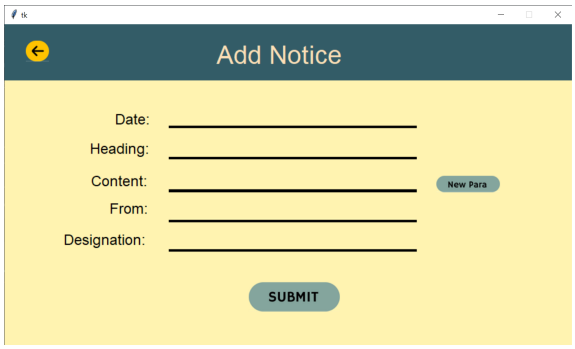


Figure 20: Add Notice

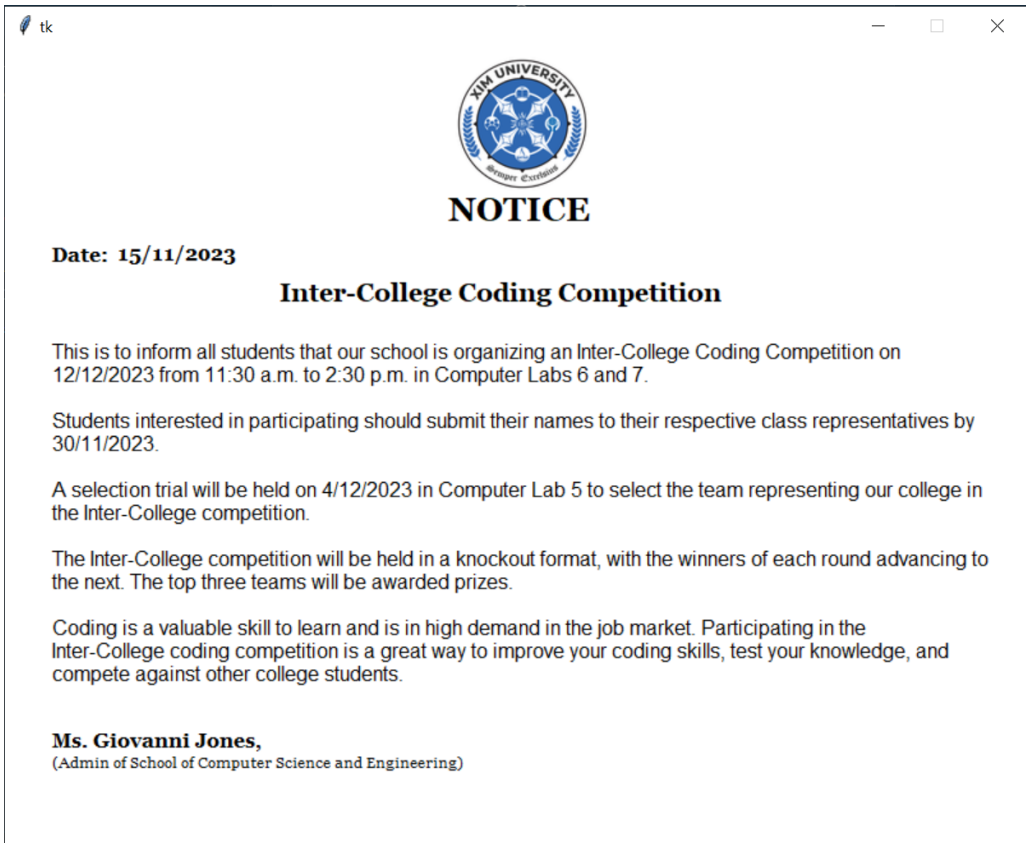


Figure 21: Noticeboard

8 Work to be done

- View notice board is not yet functional.
- View grades and view courses for students is also not yet functional.
- Searching feature is not added yet such as search by student name, course name, teacher name, student's phone number, etc.
- Dynamic button concept not yet implemented in the view notice board section.

9 Appendix

9.1 Github Link

https://github.com/ranojay-07/Capstone-1

9.2 Code Snippet

main.py

```
import login, importlib
while True:
    importlib.reload(login)
```

login.py

```
from tkinter import *
from PIL import Image, ImageTk
import mysql.connector, pickle, os, sys, importlib

def destroy_info():
    if os.path.exists("id.pickle"):
        os.remove("id.pickle")
    if os.path.exists(f"temporary/temp{id}.png"):
        os.remove(f"temporary/temp{id}.png")

def on_closing():
    destroy_info()
    root.destroy()
    sys.exit()

def loginuser():
    id = user.get()
    with open("id.pickle", "wb") as uid:
        pickle.dump(id, uid)
    password = code.get()
    connection = mysql.connector.connect(
        host='localhost',
        user='root',
        password='rootroot',
        database='xim'
    )
    cursor = connection.cursor()
    query = "SELECT * FROM login WHERE id = %s AND password = SHA(%s)"
```

```
    cursor.execute(query, (id, password))
    results = cursor.fetchone()

    if results is not None:
        root.destroy()
        if (id[2] == "s"):
            import student
            while True:
                if os.path.exists("close.pickle"):
                    with open("close.pickle","rb") as closeid:
                        close = pickle.load(closeid)
                    if (close == "logout"):
                        if os.path.exists("close.pickle"):
                            os.remove("close.pickle")
                        break
                elif not os.path.exists("id.pickle"):
                    break
                else:
                    importlib.reload(student)
        elif (id[2] == "f"):
            import faculty
            while True:
                if os.path.exists("close.pickle"):
                    with open("close.pickle","rb") as closeid:
                        close = pickle.load(closeid)
                    if (close == "logout"):
                        if os.path.exists("close.pickle"):
                            os.remove("close.pickle")
                        break
                elif not os.path.exists("id.pickle"):
                    break
                else:
                    importlib.reload(faculty)
        else:
            import admin
            while True:
                if os.path.exists("close.pickle"):
                    with open("close.pickle","rb") as closeid:
                        close = pickle.load(closeid)
                    if (close == "logout"):
                        if os.path.exists("close.pickle"):
                            os.remove("close.pickle")
                        break
                elif not os.path.exists("id.pickle"):
                    break
                else:
                    importlib.reload(admin)
    else:
        label = Label(frame, text="Invalid Id or Password", fg='red', bg='white',
        font=('Microsoft YaHei UI Light', 10))
        label.place(x=110, y = 250)

root=Tk()
root.title('Login')
root.geometry('925x500+300+200')
root.configure(bg="#fff")
root.resizable(False, False)
```

```
img = PhotoImage(file='images/login.png')

Label(root, image=img, bg='white').place(x=50,y=50)

frame = Frame(root, width=350, height=350,bg='white')
frame.place(x=500, y=70)

heading= Label(frame, text='Sign In', fg='#57a1f8',bg='white',
font=('Microsoft YaHei UI Light', 23, 'bold'))
heading.place(x=100,y=5)


""" id ENTRY """
def user_focus_in(e):
    user.delete(0, 'end')

def user_focus_out(e):
    name = user.get()
    if name =='':
        user.insert(0, 'id')

user = Entry(frame, width=25, fg='black', border = 0, bg='white',
font=('Microsoft YaHei UI Light', 11))
user.place(x=30, y=80)
user.insert(0, 'User-id')
user.bind("<FocusIn>", user_focus_in)
user.bind("<FocusOut>",user_focus_out)

Frame(frame, width=295, height=2, bg='black').place(x=25, y=107)

""" PW Entry """
def code_focus_in(e):
    code.delete(0, 'end')

def code_focus_out(e):
    name = code.get()
    if name =='':
       code.insert(0, 'Password')

code = Entry(frame, width=25, fg='black', border = 0, bg='white',font=
('Microsoft YaHei UI Light',11), show = "*")
code.place(x=30, y=150)
code.insert(0, 'Password')
code.bind("<FocusIn>", code_focus_in)
code.bind("<FocusOut>",code_focus_out)

Frame(frame, width=295, height=2, bg='black').place(x=25, y=177)

""" Submit button """
Button(frame, width=39, pady=7, text='Sign In',bg='#57a1f8', fg='white', border=0,
command=loginuser).place(x=35, y=204)

button_mode = True

def hide():
    global button_mode
    if not button_mode:
        eyeButton.config(image = closeeye, activebackground="white")
```

```
        code.config(show = "*")
        button_mode = True
    else:
        eyeButton.config(image = openeye, activebackground="white")
        code.config(show="")
        button_mode = False

openi = Image.open("images\show.png")
openi = openi.resize((25, 25))
openeye = ImageTk.PhotoImage(openi)

closei = Image.open("images\hide.png")
closei = closei.resize((25, 25))
closeeye = ImageTk.PhotoImage(closei)

eyeButton = Button(image = closeeye, bd = 0, command = hide)
eyeButton.place(x = 795, y = 220)

loginButton = Button(root, text = "LOGIN", width = 10, height = 1, font =
("arial", 20, "bold"), bd = 0, fg = "black",  command = loginuser)
loginButton.place (x = 570, y = 600)

root.protocol("WM_DELETE_WINDOW", on_closing)
root.mainloop()
```

addnotice.py

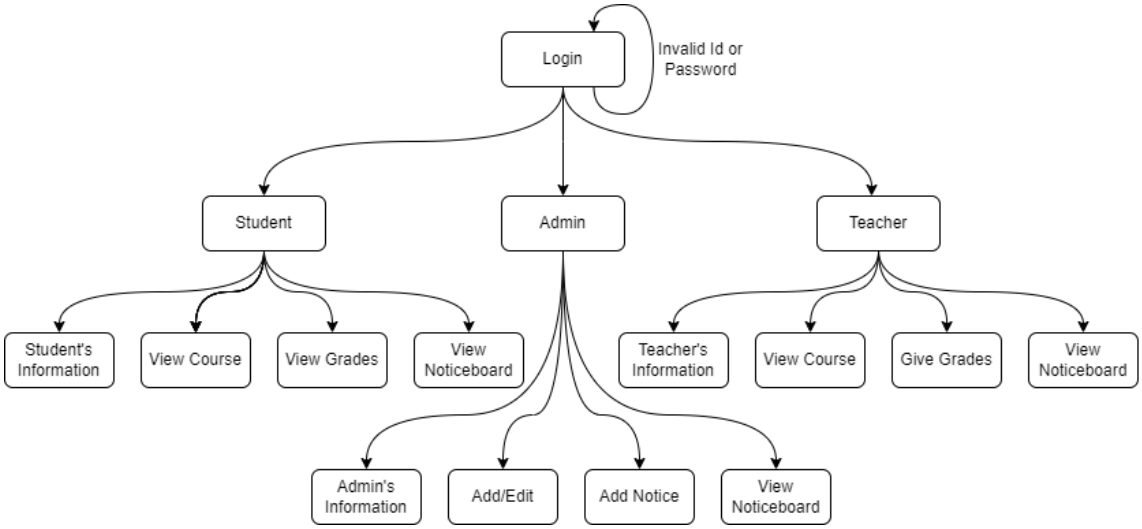
```
...
def seperator (strings):
    final_string = ""
    clear_content()
    previous_sentence = ""
    sentence = ""

    word = ""
    c = 0

    for i in range ( len ( strings ) ) :
        if ( strings[i] != " " ):
            word = word + strings[i]
            c = c + 1

        else:
            if c > 107:
                if (final_string == "" ):
                    final_string = previous_sentence
                else:
                    final_string = final_string + "n̤" + previous_sentence
                sentence = ""
                c = len (word)

            if (sentence == "" ):
                sentence = word

            else:
                sentence = sentence + " " + word
                c = c + 1
```

```
            word = ""
            previous_sentence = sentence

    if ( (len(sentence) + len (word) + 1) > 108 ):
        final_string = final_string + "n̤" + sentence + "n̤" + word
    else:
        final_string = final_string + "n̤" + sentence + " " + word

    if final_string[0] == "n̤":
        final_string = final_string [1:]

    if not os.path.exists("notice.pickle"):
        with open("notice.pickle", "wb") as ns:
            pickle.dump(final_string, ns)
    else:
        with open("notice.pickle","rb") as ns:
            newFinalString = pickle.load(ns)
        newFinalString = newFinalString + "n̤n̤" +  final_string
        os.remove("notice.pickle")
        with open("notice.pickle", "wb") as ns:
            pickle.dump(newFinalString, ns)
...
```

Took help from:

1. Tkinter Designer:
   (a) https://github.com/ParthJadhav/Tkinter-Designer
   (b) https://www.youtube.com/watch?v=Qd-jJjduWeQ
2. Stackoverflow Doubt link
   (a) https://stackoverflow.com/questions/77183997/i-cant-go-from-one-tkinter-python-file-to-another-tkinter-python-file-again-and
3. Tkinter basics:
   (a) https://docs.python.org/3/library/tkinter.html
   (b) https://www.geeksforgeeks.org/python-gui-tkinter/
   (c) https://www.tutorialspoint.com/python/python-gui-programming.htm
4. Login Window design and mechanism:
   (a) https://www.youtube.com/watch?v=X9reTIMckk

9.3 Flow Chart



Here, the login window is the first window that the user encounters. Here, he or she has to enter their ID with its password to gain access to the respective dashboard. For a student, their privilege is to see their information, including the courses in which they were enrolled or are currently enrolled, to check their grades, and finally to see the noticeboard.

How will it work?

When a user makes a successful login, then from their ID, it will detect whether the user is a student, a teacher, or an admin. Then, it will store the ID, and it will just open whatever the user wants to do. If the user wants to see his or her information, then all the information that comes under that ID will be displayed; likewise, the other options will also show information under that ID in that table of the database.