SOFTWARE ENGINEERING LAB - 4

COLLEGE REGISTRATION PROJECT REPORT

By:-
B. Sri Chaitanya
22CS10018

Problem Statement:-

- The given problem is to create a system that acts as a portal for registration and other operations for people of a certain college. The main aim of this project is to develop the concept of OOPS and the use of 'tkinter' library of Python.

- There is a hierarchy based system with the root being a 'Person'. Teacher and student follow from it, again from the latter of which, UGStudent and PGStudent follow. Also, it must follow the follow functionalities:
-

1. User registration (new user enrolment)

- User registration includes setting a User ID and Password
  User ID: It should be the active email address of the user (or anything else the developer can decide)
- Password: A valid password should satisfy the following:
  a) It should be within 8-12 character long.
  b) It should contain at least one upper case, one digit, and one lower case.

c) It should contains one or more special character(s) from the list [! @ # $ % & *]
d) No blank space will be allowed

.

2. Sign-in to the system

The system should authenticate to check if a user is legitimate or not. A maximum of
three attempts will be allowed for the verification; after the three wrong attempts, the
account will be deactivated

.

3. Edit/ Update the user's profile

An authorised user will be allowed to fill in the user's data. Also, it allows to edit (modify,
update) user's data.

4. Deregistration request

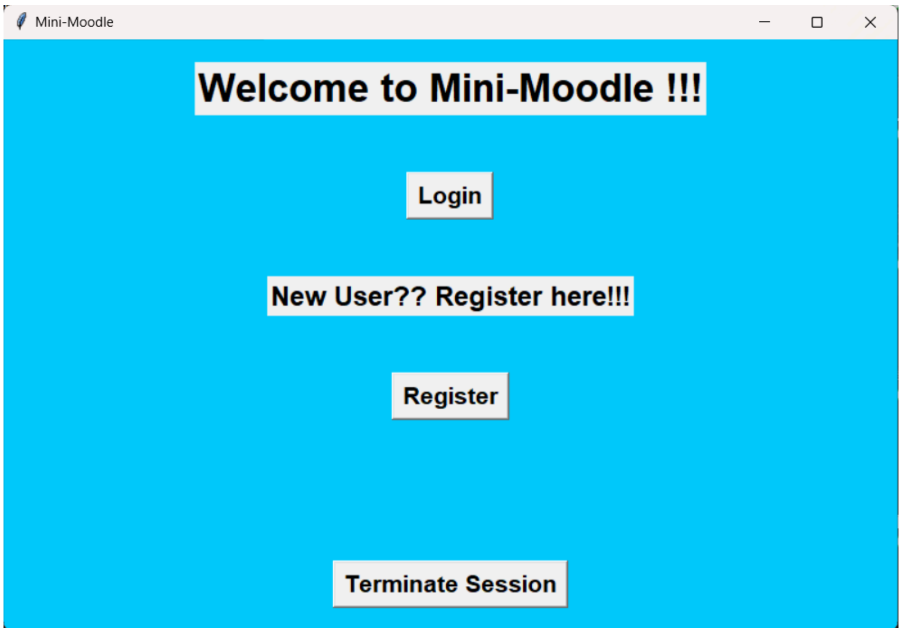
A user may be allowed to submit the deregistration request. On successful submission of
the request, the user's account will be inactive (logically deleted).

• Define classes as per the given class hierarchy.
• Decide the structure of the list, which can be used to store records of the different classes.
• Develop the user interfaces for the system with the use cases as mentioned.
• Link the GUI elements to the program at the back-end.

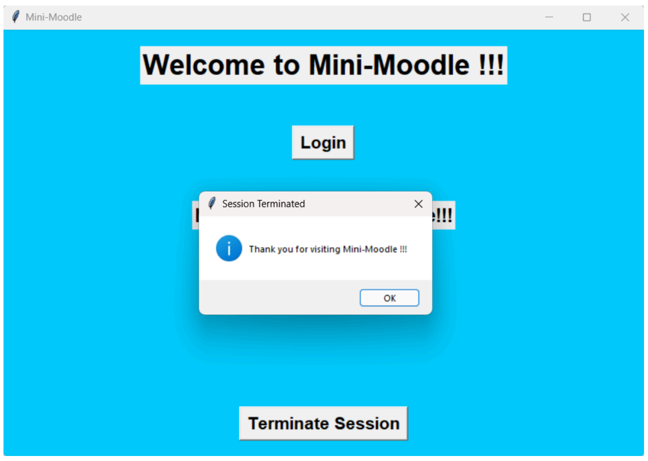
APPROACH :-

- Created classes based on the conditions imposed, with required initialisations done.
- All the data will be stored in a file "user_data.txt", so that the records can be accessed through multiple iterations of the program.
- Create a "Main Page", i.e the opening page of the software.

Main Page:-

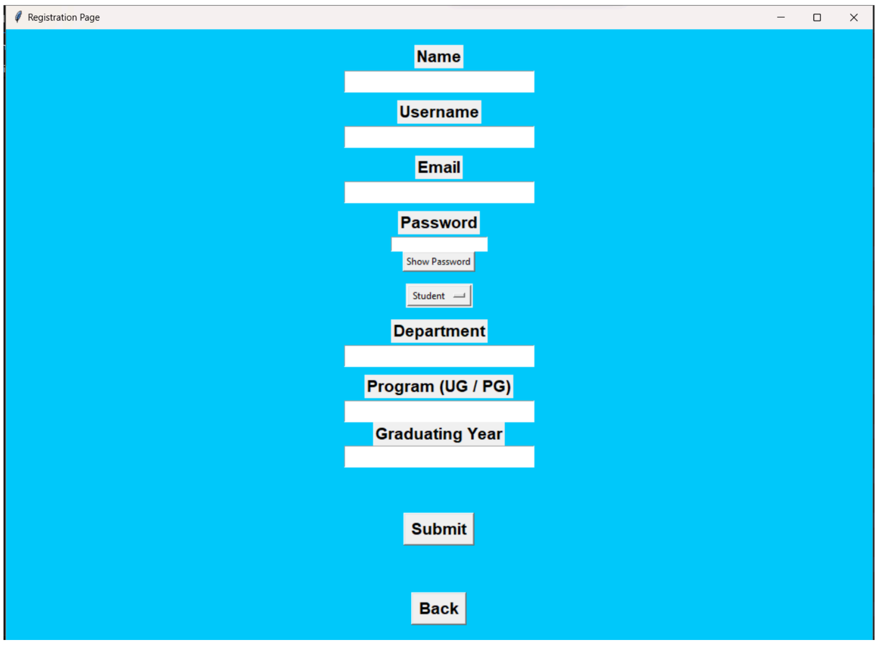


- The main page is as above. The main page, apart from the message labels, has 3 buttons.
- The **“Login”** button take’s the user to a login portal, with the **“Register”** button doing the same.
- The **“Terminate Session”** button closes the main page altogether and shows a message

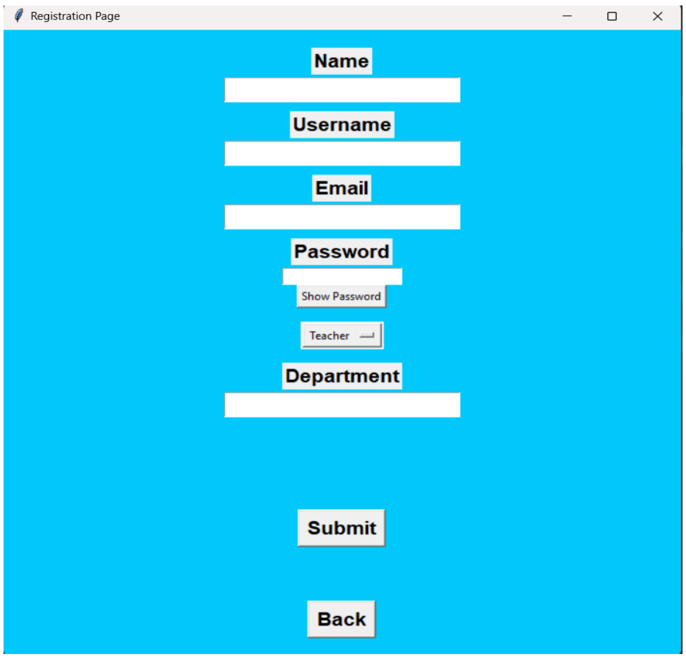


Thank you for visiting Mini-Moodle !!!

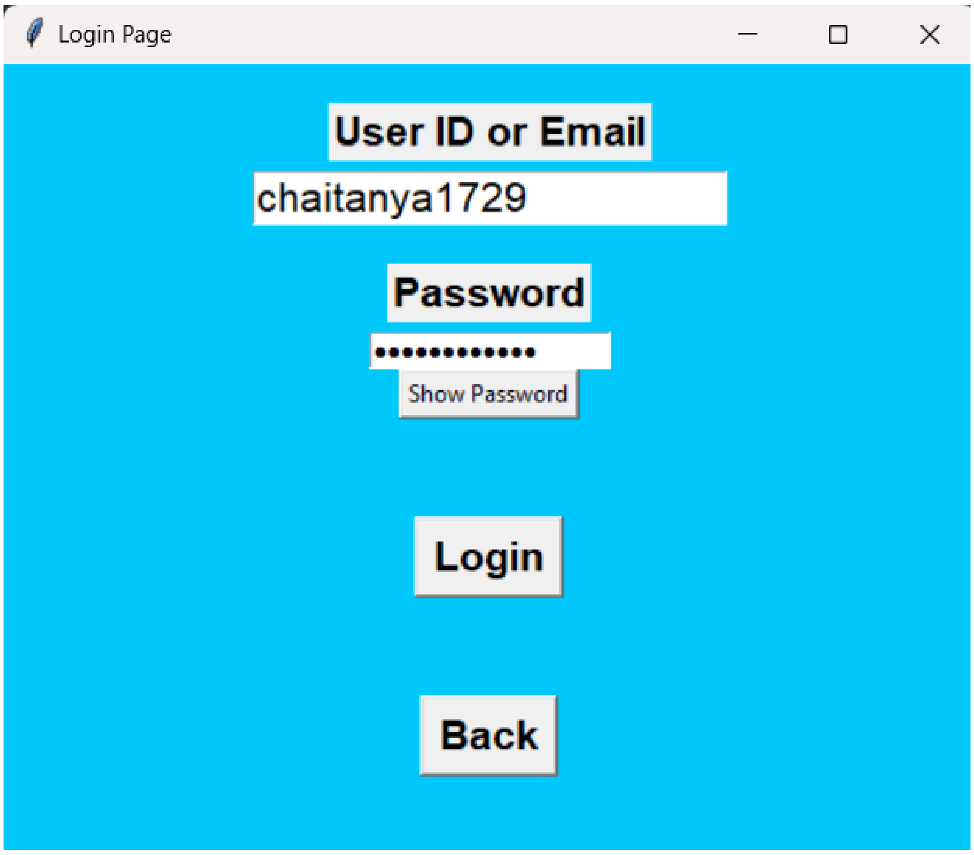
Registration Page:-



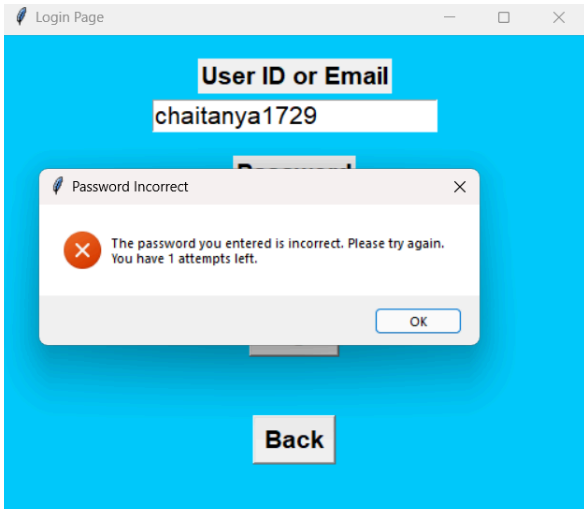
- The registration page contains fields that allow a user to create a new account and register himself.
- Each of the fields is designed and executed perfectly so that the required functionalities are met .
- The username and email are checked for uniqueness. Only a valid password is accepted.
- The user can alternatively use the teacher tab to get the fields for a teacher type account.



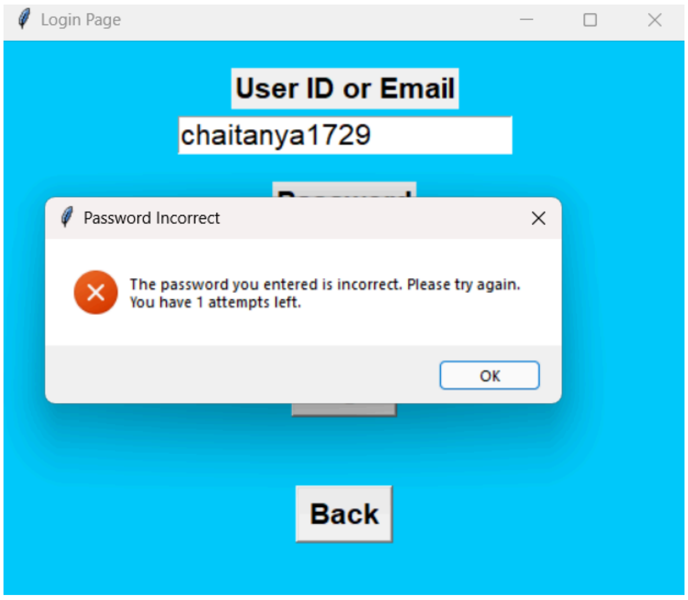
Login Page:-



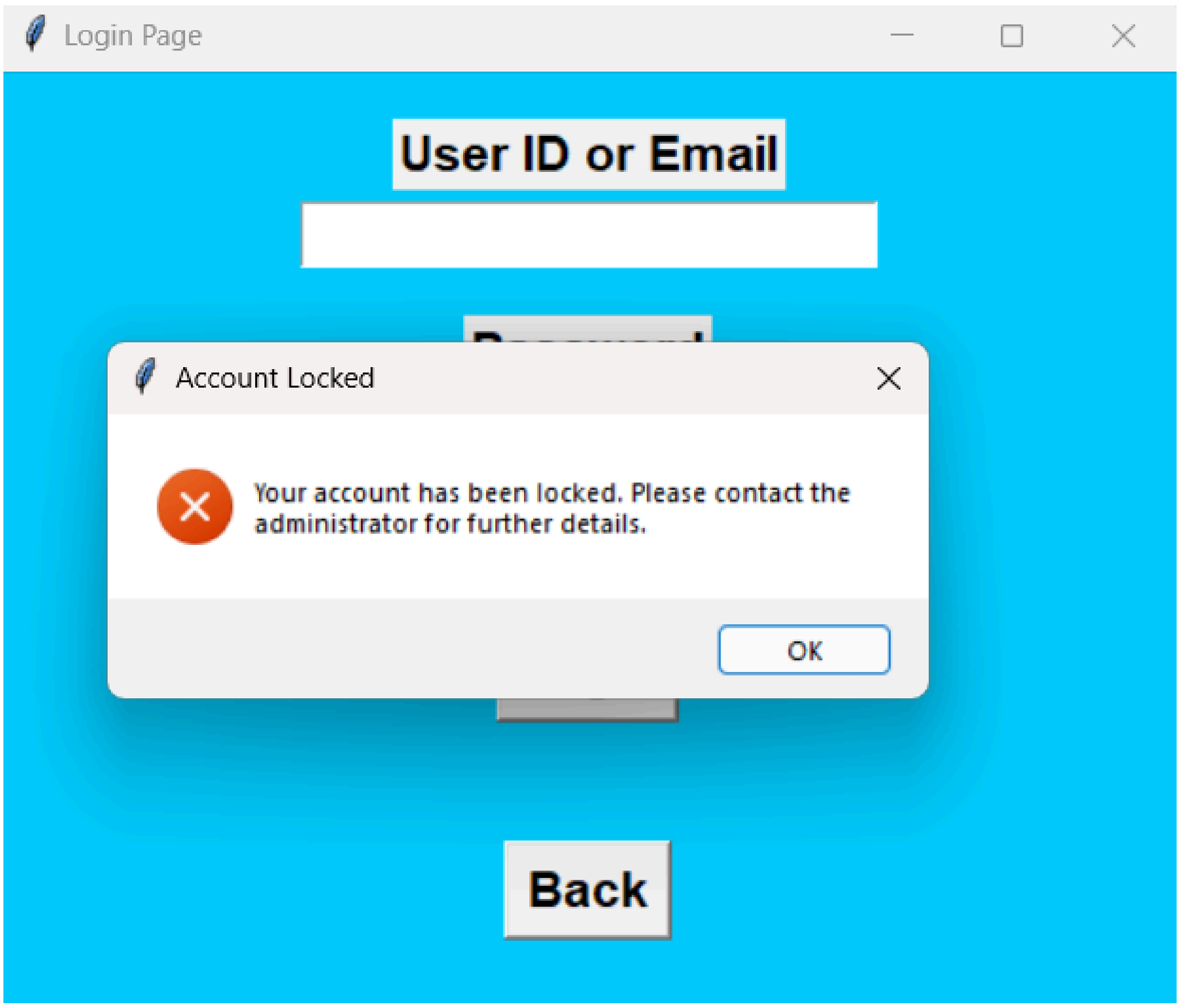
- The login page has two fields for username and password , which when correct , leads us into the personal page.
- However, if a user enters his password 3 times wrong consecutively in one session, his account will be deactivated, while being warned till the end of how many attempts he has left.



The password you entered is incorrect. Please try again.
You have 1 attempts left.


Password Incorrect

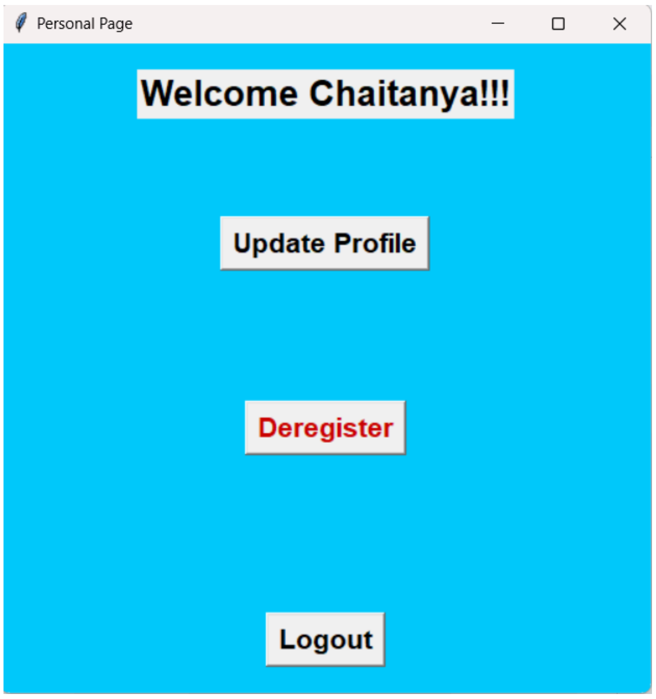
The password you entered is incorrect. Please try again.
You have 1 attempts left.


Account Locked

Your account has been locked. Please contact the administrator for further details.

- Upon successful login, we go into the personal page.

Personal Page:-

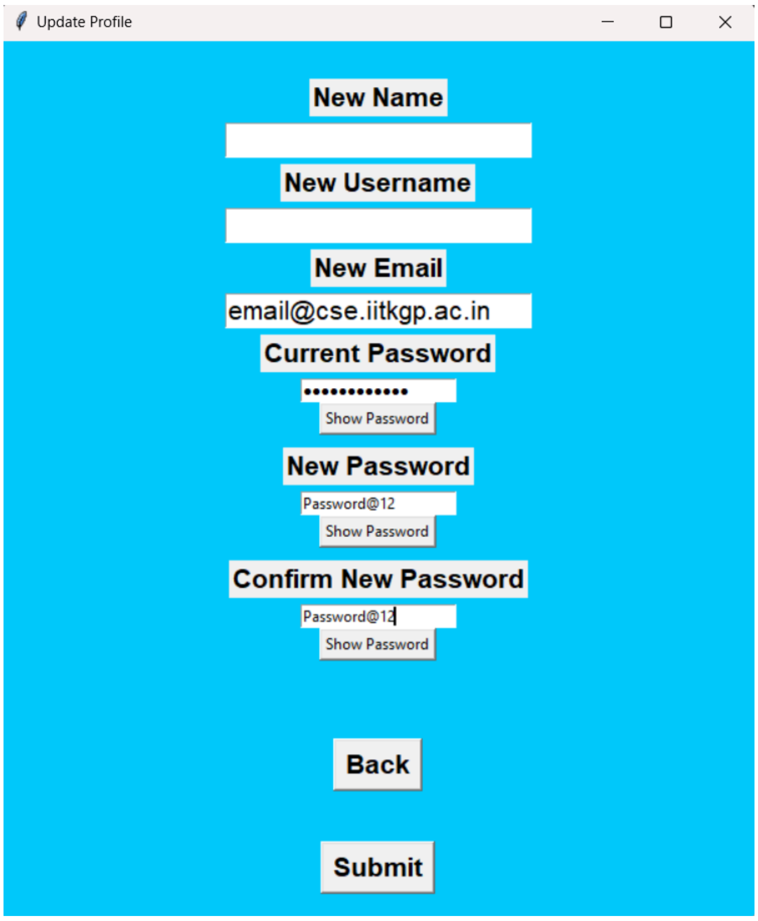


- The personal page is the main kernel, which acts as the portal to perform to operations.
- It has a welcome message to the user, calling him/her by their name.
- It consists of 3 buttons. The **"Update Profile"** button takes us to the update profile window, where the user can update the existing details. **"Deregister"** button, which after asking for confirmation, completely deletes the user's account.
- **"Logout"** logs out of the account and takes the user back to the main page

a) Update Profile:-

- It contains fields that allow the user to modify his basic data such as email,name,etc.
- We do not allow the user to change his profession , department, etc.

- If the user doesn't want to change a particular attribute, the field for it can be left completely blank to ignore any changes there.


Update Profile

New Name

New Username

New Email

email@cse.iitkgp.ac.in

Current Password

•••••••••••••

Show Password

New Password

Password@12

Show Password

Confirm New Password

Password@12

Show Password

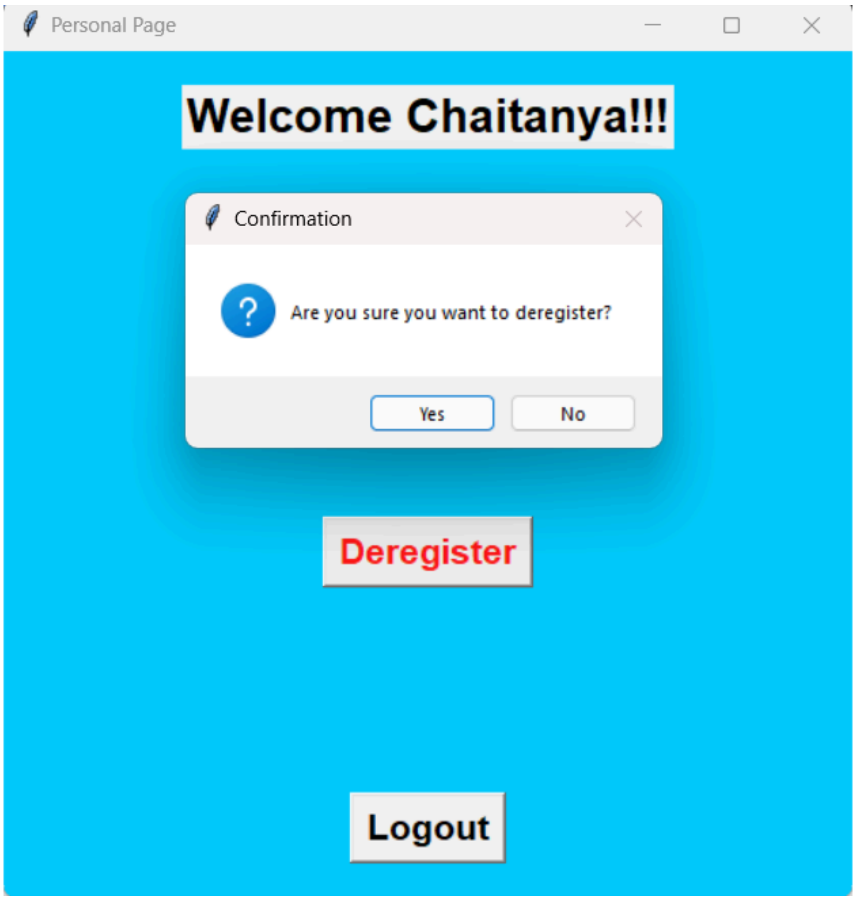
Back

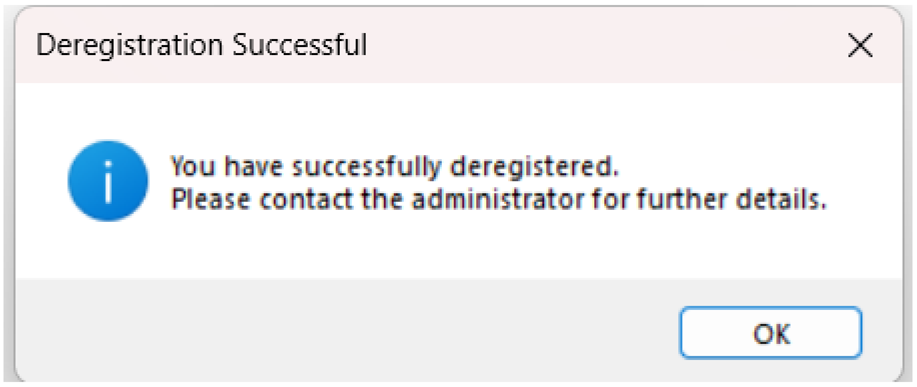
Submit

- To make any change possible, the current password is taken again and checked for security measures. Any change is thereby made only after the password matches
- The **"Back"** button takes us back to the personal page, while the **"Submit"** button submits the changes.
- These changes are then written in the file through functions at the back end.

b) **<u>Deregister:-</u>**

- The deregister button, when pressed, asks the user for confirmation to deregister the user, and if yes, deletes the user's data from the records(file)





You have successfully deregistered.
Please contact the administrator for further details.

- After the deregistration is complete, the user goes back to the main page.

<u>BACK-END Details :-</u>

- All the user(s) details are written into a text file named “user_data.txt”.
- The passwords are encrypted and written into the file for security issues. The function **encrypt_password(password)** encrypts the given password using the “Fernet” library.
- Although the encryption key must be kept securely, it is fixed here and declared as a global variable for demonstration purposes.
- A function **decrypt_password** also exists to decrypt the encrypted passwords. Care must be taken while handling the passwords because the encryption of the same password twice wouldn’t be same (although the decryption would be, which is exploited here).
- The encrypted passwords are of type bytestrings, which should be taken care of while using.
- At the beginning of the program, all the user_name and email are added into a global list to keep track of the number of attempts during login.
- **Show Password** functionality has been added so that there is an option for the user to see what password they’ve typed.
- There are other back end functions such as **validate_password,validate_identifier,delete_from_file,** etc that perform the required operations in accordance with the GUI.

- In this way, the functionality has been achieved for the given problem.

- A read through the code, which is well commented, would give you insight regarding what is the exact purpose of each function, and where they are used.