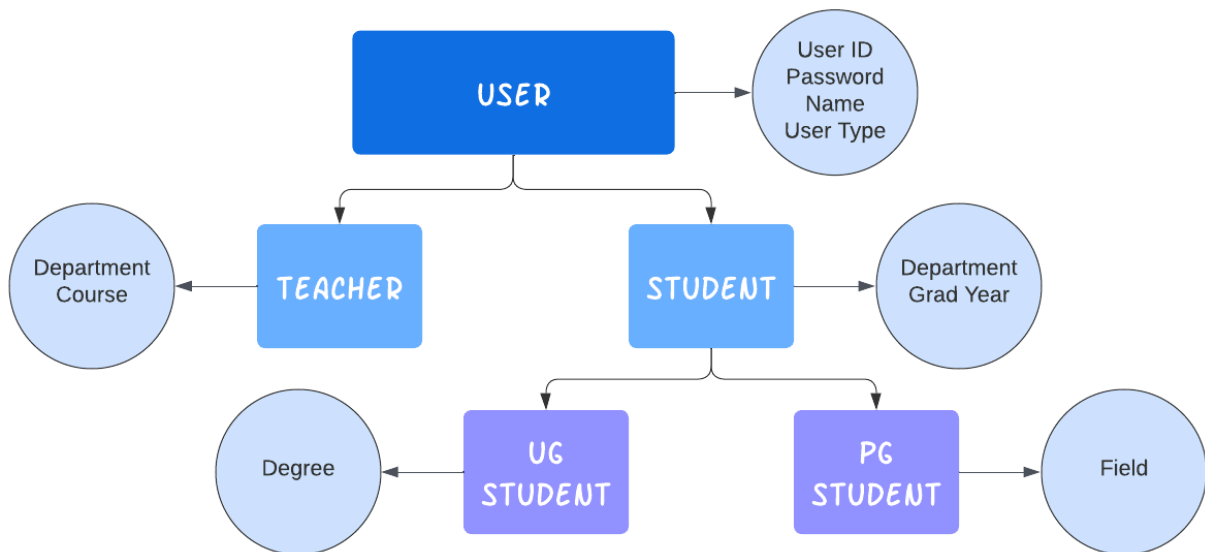


Software Application Development Using Python

Python Libraries Used:

- [tkinter](#)
- [json](#)
- [re](#)
- [datetime](#)

Class Hierarchy (hierarchy.py):



Classes and subclasses are implemented according to the above map.

In addition, the User class contains an instance attribute `attempts` (this is instantiated as 3 at the start of the application and reset after every login) to keep a count of login attempts, and every class contains a class attribute `all`, a list which stores objects of that particular class.

The User class also contains a class method, which is used to read data (stored in a JSON file `users.json`) and create corresponding objects.

Implementation & GUI (main.py):

Reading/Dumping data from/in JSON file is facilitated by the `json` library.

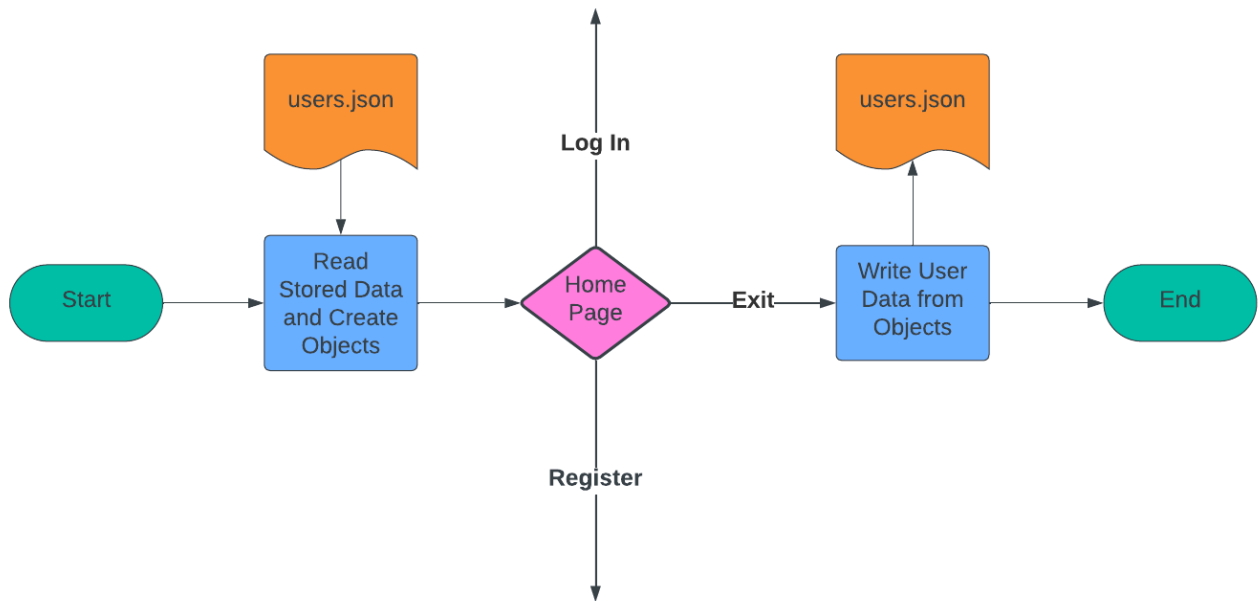
Using the `tkinter` library, a window is created (which opens at the center of the screen and is not resizable) and a **Canvas** is placed on it.

Every **widget** (Button, Entry, OptionMenu etc.) created is placed on (using windows on the canvas) or deleted from the canvas to traverse through different "pages".

Most button widgets have callback functions attached to them, which dictate the flow of the program.

In the next few pages, the flow and decisions are explained through flowcharts and screenshots.

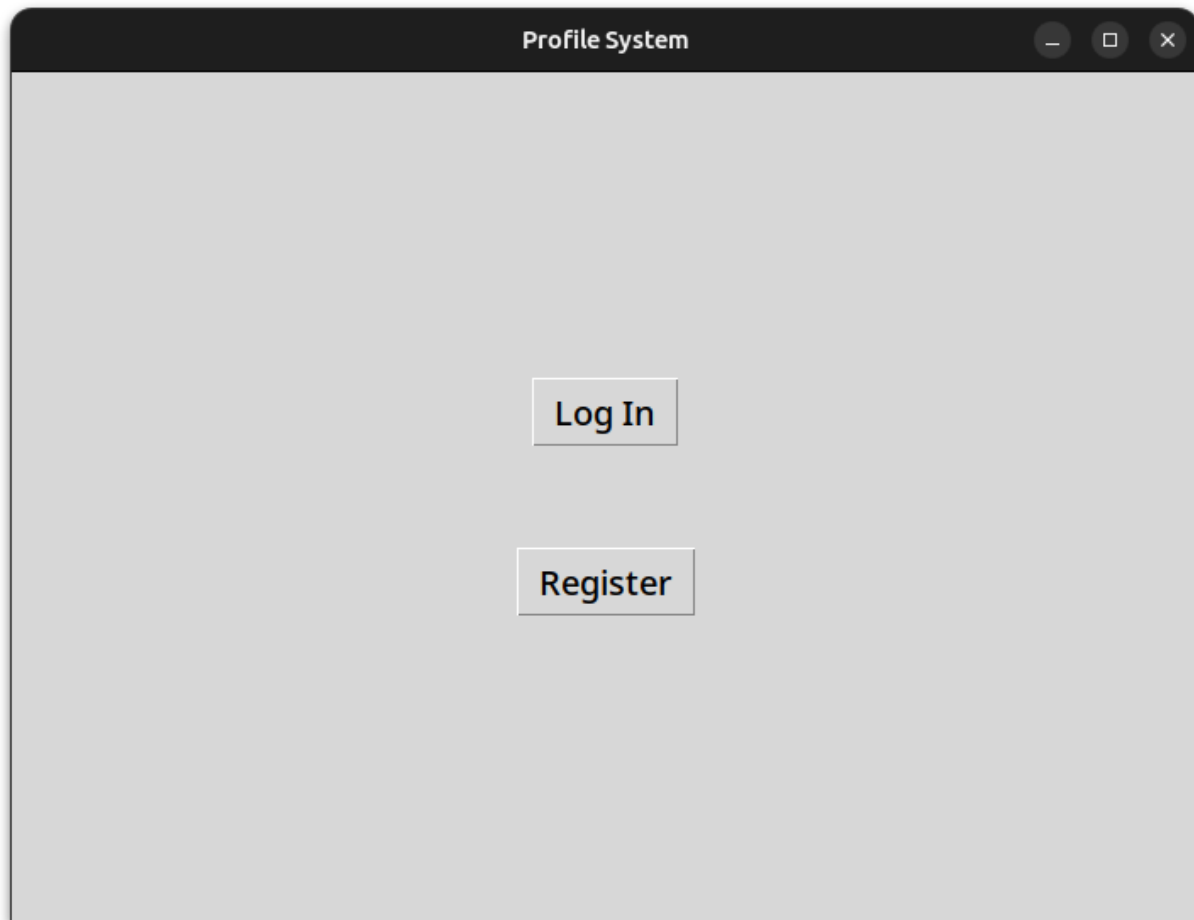
→ Home Page:



Functions:

- `download_user_data()`
- `upload_user_data()`
- `home()`

Screenshots:



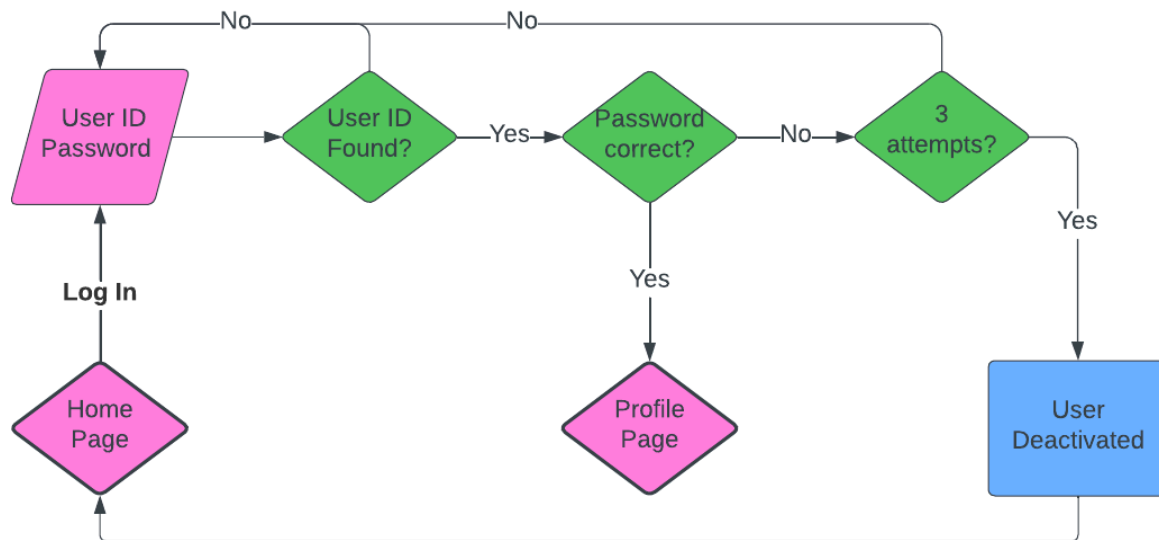
```
users.json U x
assign_02 > users.json > ...
1  {
2    "id": "harshit_jain52",
3    "pwd": "123@Hjain",
4    "name": "Harshit Pankaj Jain",
5    "utype": "UG Student",
6    "attempts": 3,
7    "dept": "CSE",
8    "grad": 2026,
9    "degree": "B. Tech."
10 },
11 {
12   "id": "user123",
13   "pwd": "Hello#123",
14   "name": "Prof Amit",
15   "utype": "Teacher",
16   "attempts": 3,
17   "dept": "CSE",
18   "course": "DSA"
19 },
20 {
21   "id": "pguserXYZ",
22   "pwd": "Pass&12345",
23   "name": "Peter",
24   "utype": "PG Student",
25   "attempts": 3,
26   "dept": "Physics",
27   "grad": 2026,
28   "field": ""
29 }
```

```
<hierarchy.UG_student object at 0x7f6008d52cd0>
{'id': 'harshit_jain52', 'pwd': '123@Hjain', 'name': 'Harshit Pankaj Jain', 'utype': 'UG Student', 'attempts': 3, 'dept': 'CSE', 'grad': 2026, 'degree': 'B. Tech.'}

<hierarchy.Teacher object at 0x7f6008cc3650>
{'id': 'user123', 'pwd': 'Hello#123', 'name': 'Prof Amit', 'utype': 'Teacher', 'attempts': 3, 'dept': 'CSE', 'course': 'DSA'}

<hierarchy.PG_student object at 0x7f600911b610>
{'id': 'pguserXYZ', 'pwd': 'Pass&12345', 'name': 'Peter', 'utype': 'PG Student', 'attempts': 3, 'dept': 'Physics', 'grad': 2026, 'field': ''}
```

→ Login Page:



Entered User ID is searched for in the User .all list and the corresponding index is found out.

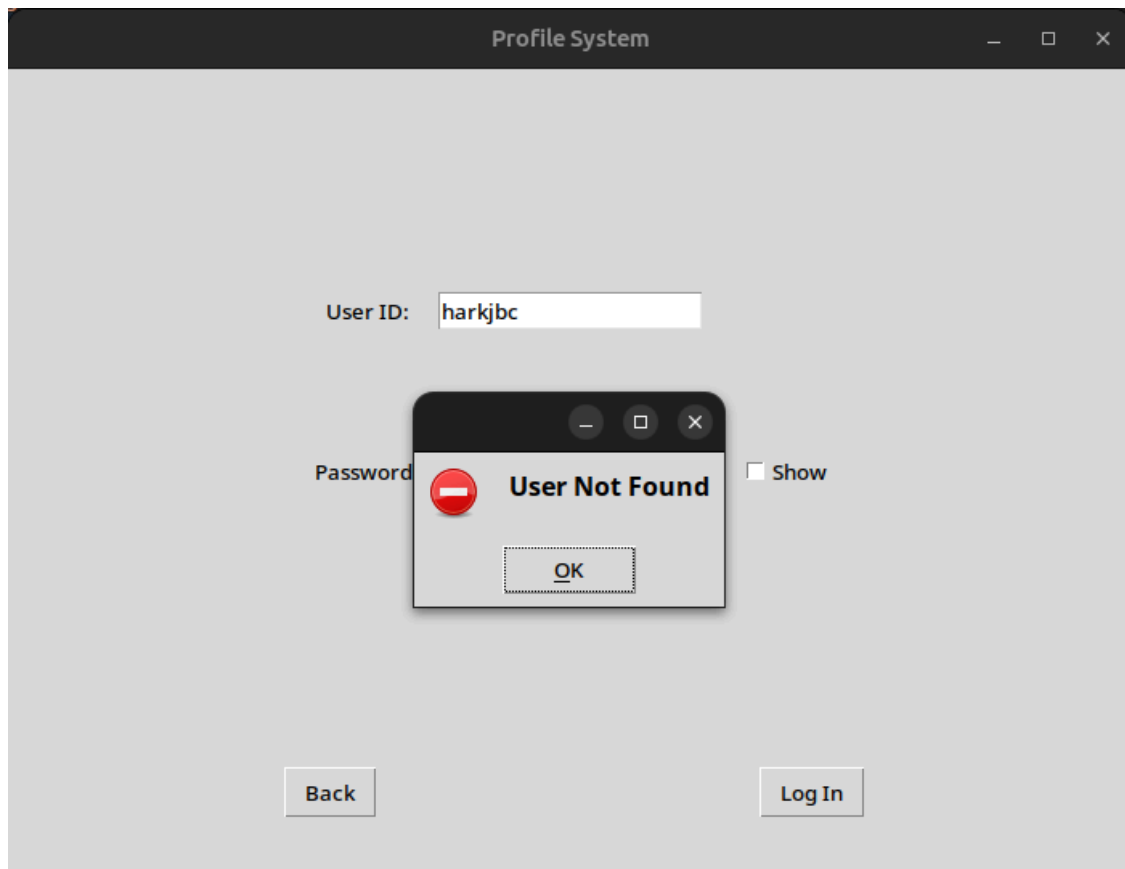
This index is used for password verification, for updating attempt count, displaying profile, and for deactivating the account.

Show/Hide password functionality is implemented using Checkbutton widget. Errors and information are displayed using messagebox.

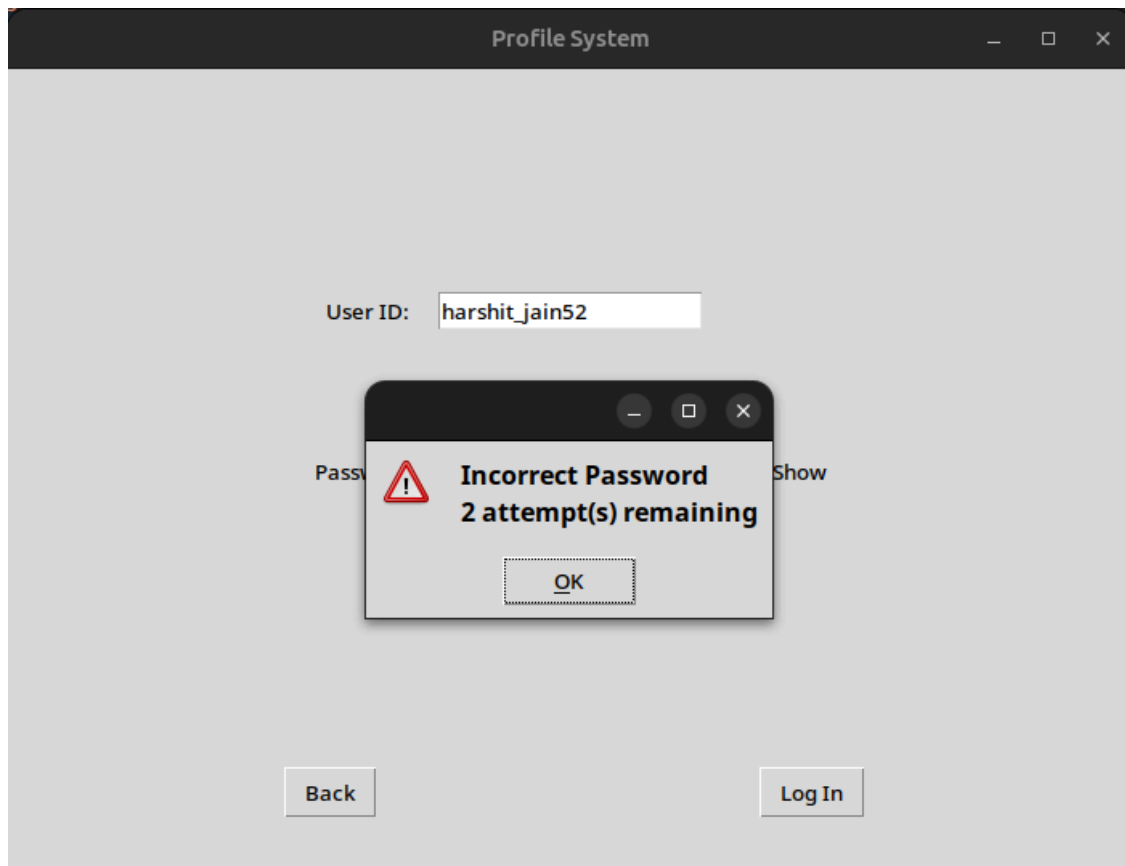
Functions:

- login()
- create_id_entry()
- create_pwd_entry()
- show_pwd()
- verify_login()
- back()

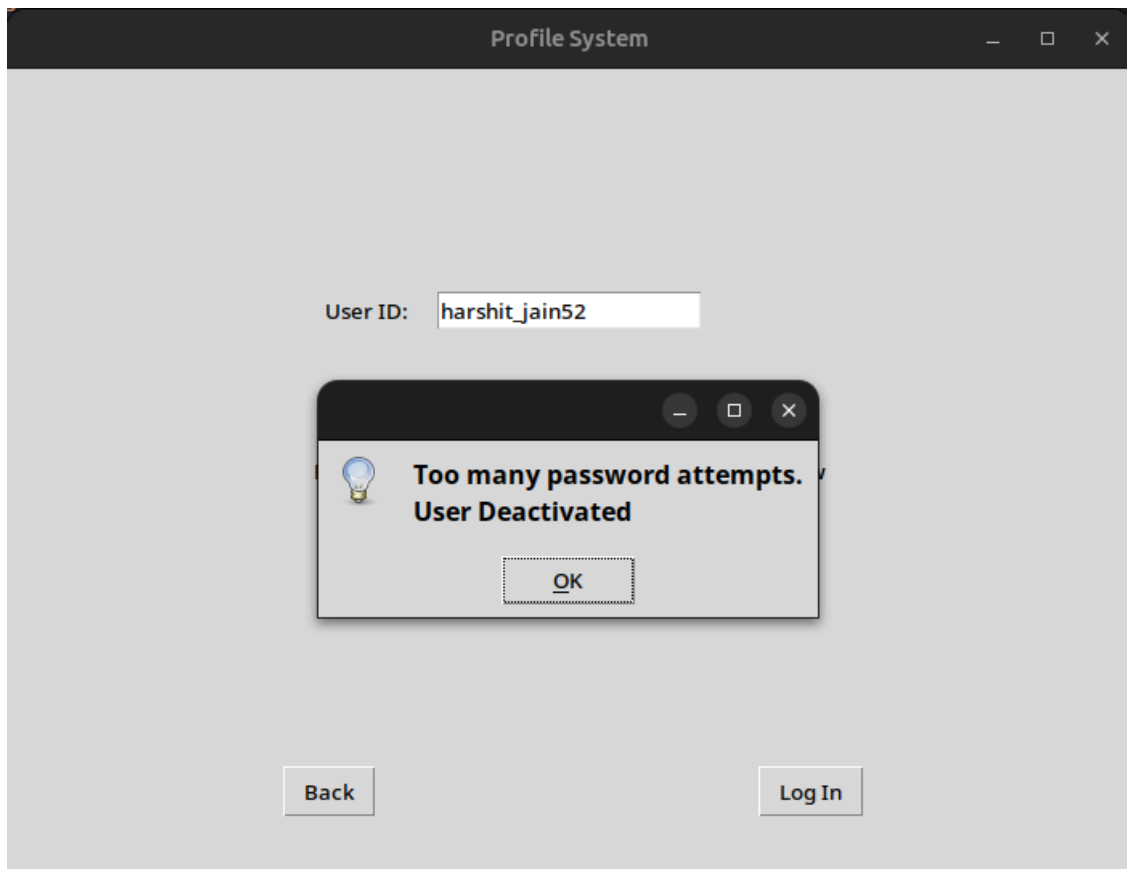
Screenshots:



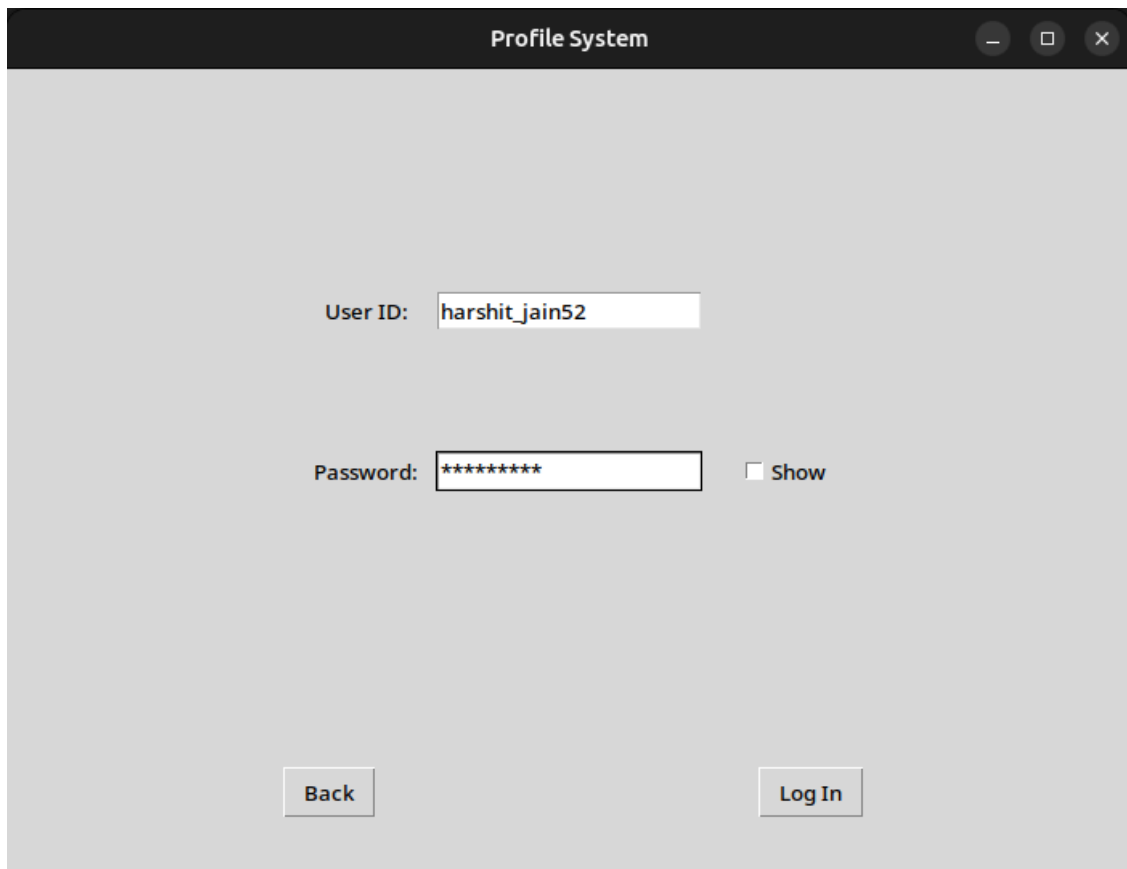
[user not found error]



[incorrect password error]



[deactivation due to 3 attempts]



[hidden password]

Profile System


User ID:

Password: ☒ Show

[unhidden password]

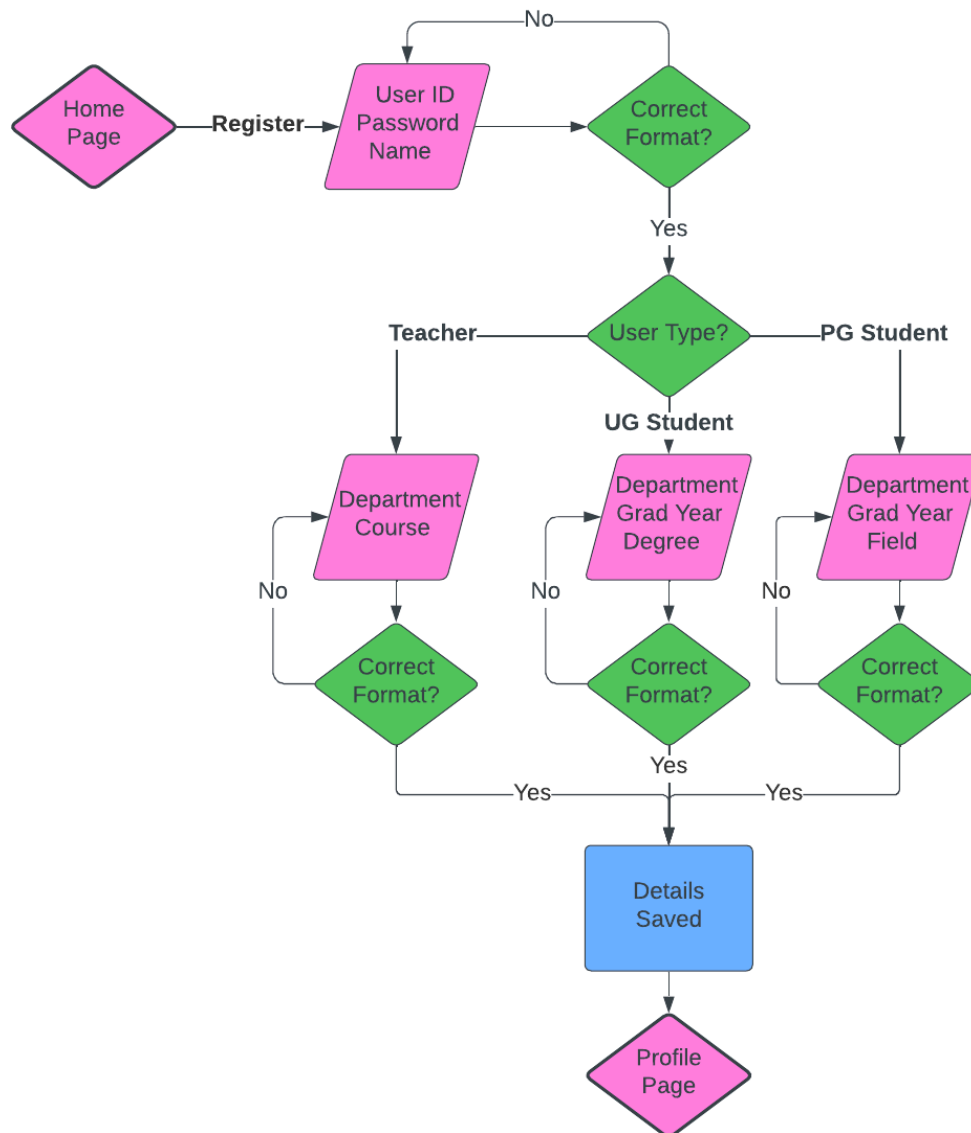
Profile System

User ID:

Passv  **Logged In Successfully** Show

[login success message]

→ Registration Page:



When a user registers, a corresponding object is created and appended (this is handled by the constructor of the corresponding class) to the `User.all` list.

Functions:

- `register()`
- `create_id_entry()`
- `create_pwd_entry()`
- `create_name_entry()`
- `create_dept_entry()`
- `create_course_entry()`
- `create_degree_entry()`
- `create_grad_menu()`
- `create_field_entry()`
- `create_type_menu()`

- `utype_select()`
- `teacher_register()`
- `ug_student_register()`
- `pg_student_register()`
- `verify_registration()`
- `verify_pwd_pattern()`
- `back()`

Profile Detail	Format Constraints	Handled by
User ID	Must not match an existing one	Traversing through the list of objects
	Can't be blank, can't have spaces	Checking length and for existence of whitespace
Password	8<=length<=12	Calculating length
	Must contain at least one lowercase, one uppercase, one digit, one special character among <code>!@#\$%&*</code> and no spaces	Using REGEX* (the re library)
Name	Can't be blank	Calculating length
User Type	Must be among [Teacher, UG Student, PG Student]	Using <code>OptionMenu</code> widget
Department	Can't be blank	Calculating length
Course	none	-
UG Grad Year	Must be from (current year) to (current year + 4)	Using <code>OptionMenu</code> widget, and the <code>datetime</code> library
PG Grad Year	Must be from (current year) to (current year + 2)	Using <code>OptionMenu</code> widget, and the <code>datetime</code> library
Degree	Must be among [B. Tech., Dual]	Using <code>Radiobutton</code> widget
Field	none	-

*REGEX used:

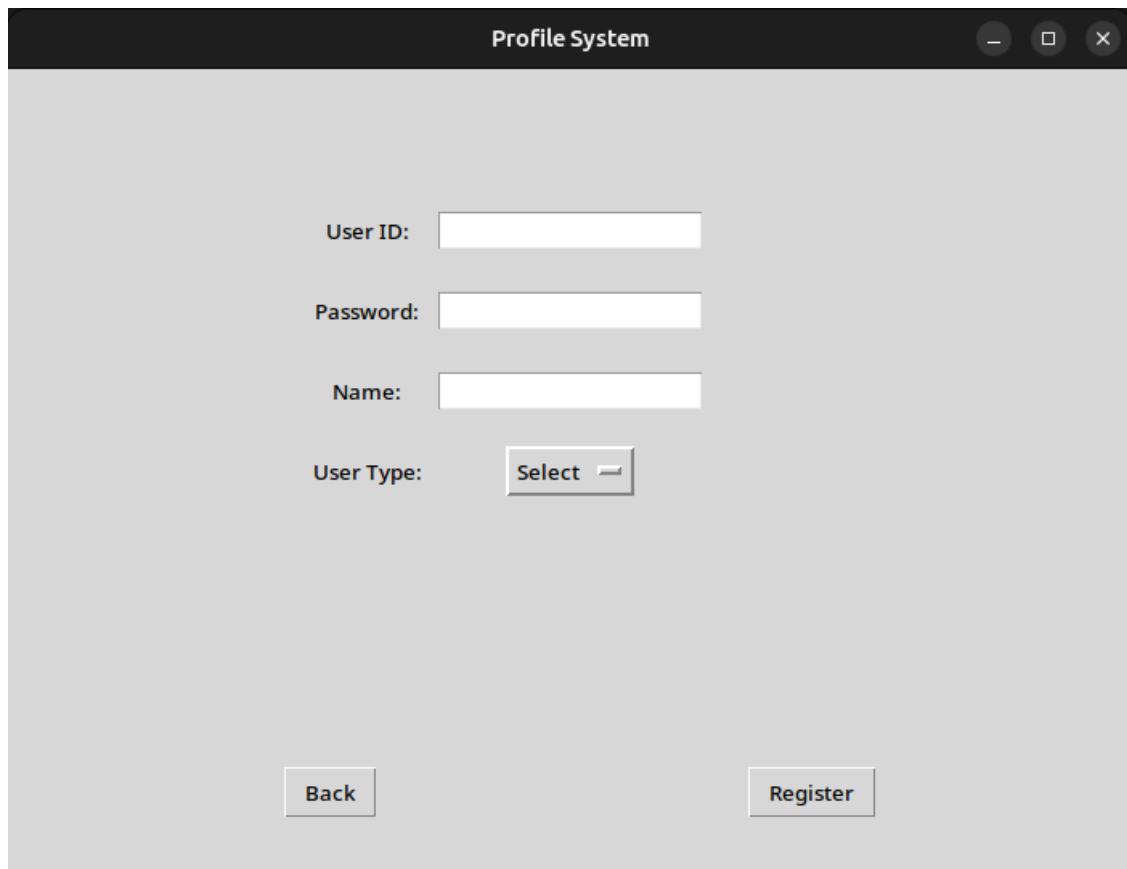
- password must match:

```
^(?=.*[a-z])(?=.*[A-Z])(?=.*[!@#$%&*])(?=.*[0-9]).*$
```

- and must not match:

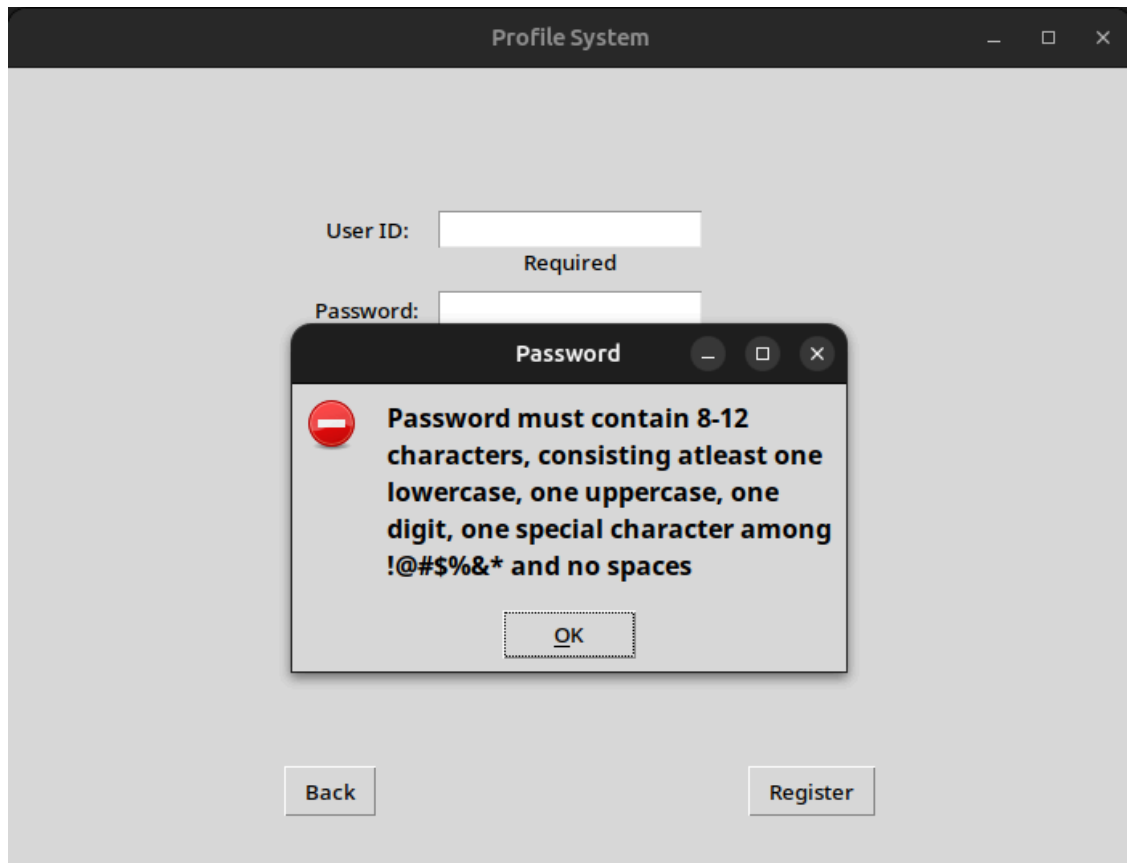
```
[\s]
```

Screenshots:



The screenshot shows a window titled "Profile System" with a light gray background. It contains four input fields: "User ID:", "Password:", "Name:", and "User Type:". The "User Type:" field is a dropdown menu with "Select" and a downward arrow. At the bottom, there are two buttons: "Back" and "Register".

[registration page before selecting user type]



The screenshot shows the same "Profile System" window, but with a modal dialog box open. The dialog is titled "Password" and has a red error icon. The text inside the dialog reads: "Password must contain 8-12 characters, consisting atleast one lowercase, one uppercase, one digit, one special character among !@#\$%&* and no spaces". Below the text is an "OK" button. The background form is partially obscured by the dialog.

[password format]

Profile System

User ID: Required

Password:

Name: Required

User Type: Required

[required fields are marked when user tries to register]

Profile System

User ID:

Password:

Name:

User Type:

[user type menu]

Profile System

User ID:

user123

Password:

Hello#123

Name:

Random

User Type:

UG Student

Department:

Graduation Year:

2024

Degree:

☒ B. Tech.

☐ Dual

Back

Register

[UG student registration]

Profile System

User ID:

user123

Password:

Hello#123

Name:

Random

User Type:

PG Student

Department:

Graduation Year:

2024

Field:

Back

Register

[PG student registration]

Profile System

User ID:

Password:

Name:

User Type:

Department:

Course:

[teacher registration]

Profile System

User ID:


Password:

Name:

User:

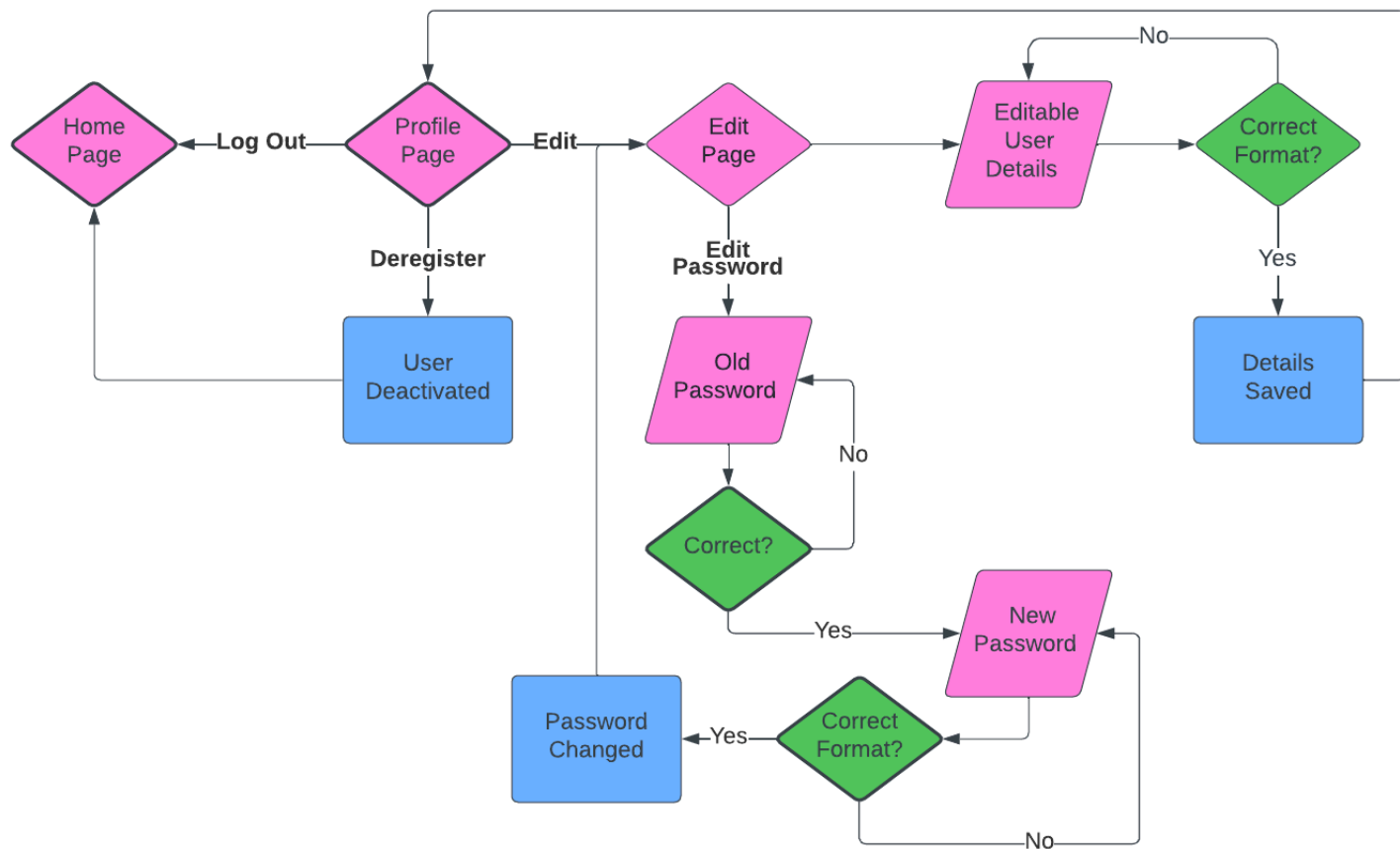
Depart:

Course:

 **Registered Successfully**

[registration success message]

→ Profile Page



Profile details are displayed using the index (position) of the object in `User.all` list.

Editing: The attributes of the object at the given index are changed

Following profile details are editable:

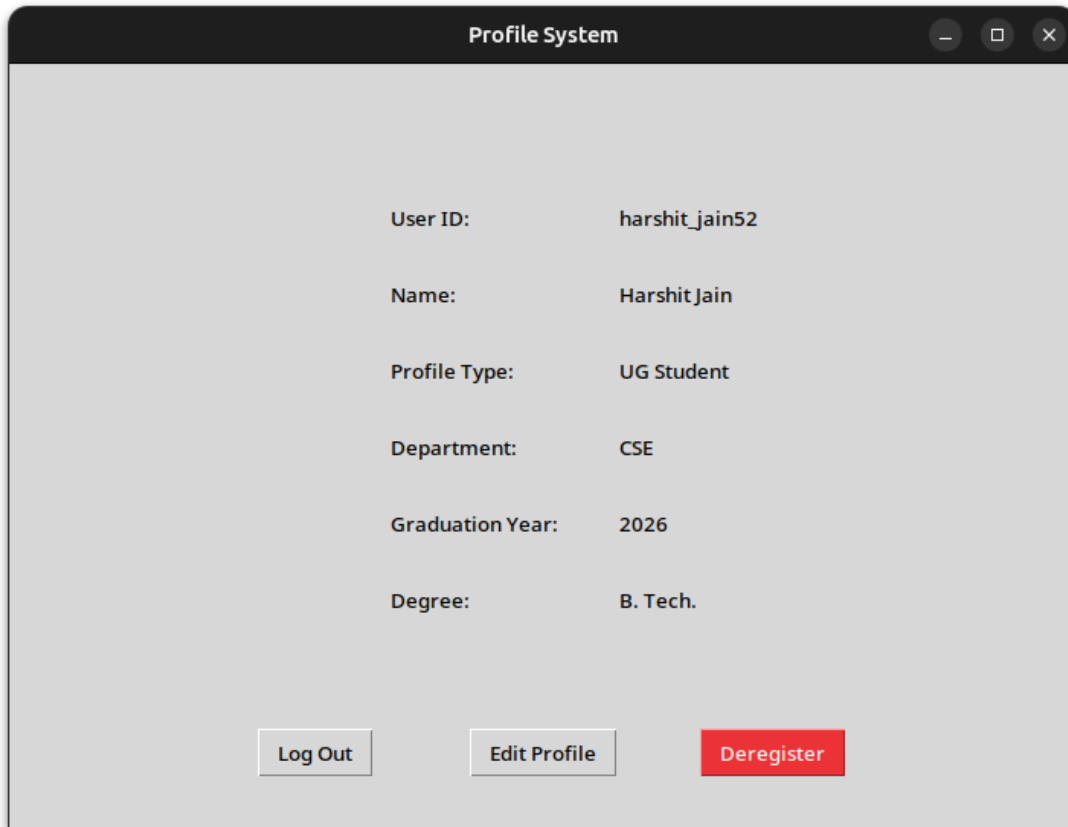
- Password
- Name
- Department
- Course
- Grad Year
- Degree
- Field

Deregistration: The object at given index is popped from the list

Functions:

- `profile()`
- `edit_profile()`
- `create_name_entry()`
- `create_dept_entry()`
- `create_course_entry()`
- `create_grad_menu()`
- `create_degree_entry()`
- `create_field_entry()`
- `create_pwd_entry()`
- `change_pwd()`
- `get_new_pwd()`
- `verify_pwd_pattern()`
- `save_new_pwd()`
- `save_edits()`
- `dereg()`
- `logout()`
- `back()`

Screenshots:

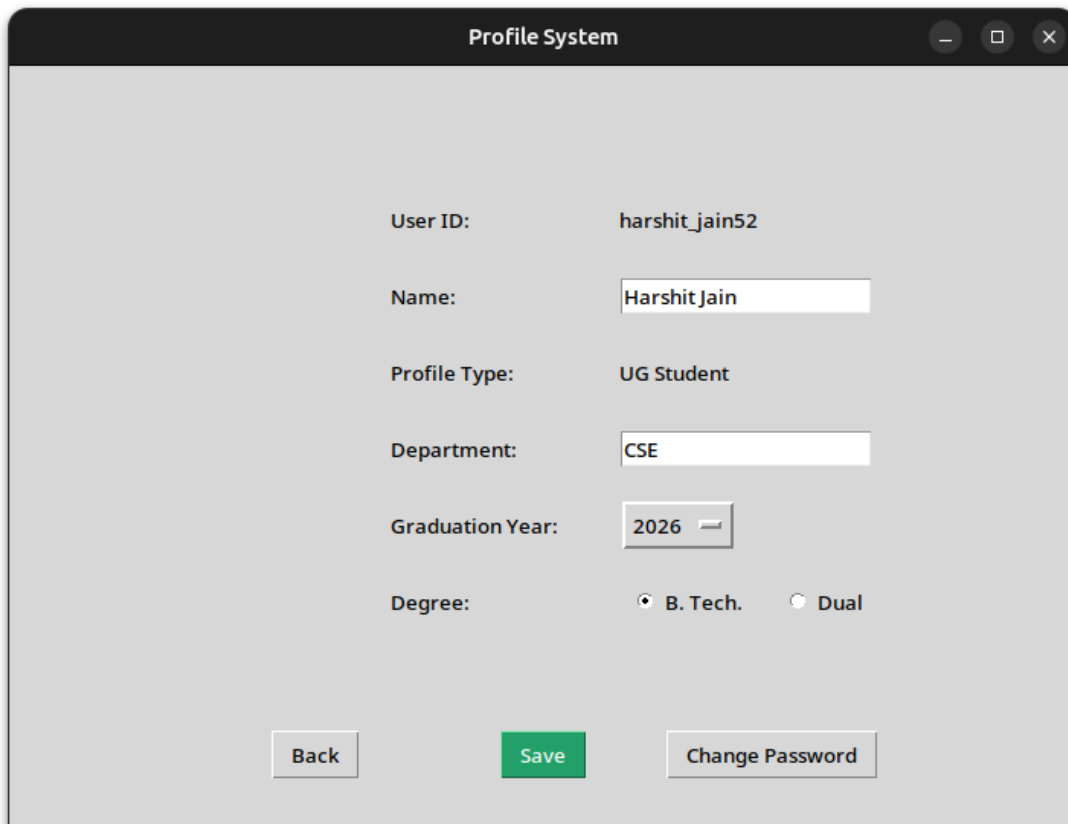


The screenshot shows a window titled "Profile System" with a dark header bar containing standard window controls. The main content area is light gray and displays a profile for a UG student. The profile information is organized into two columns: labels on the left and values on the right. The labels are "User ID:", "Name:", "Profile Type:", "Department:", "Graduation Year:", and "Degree:". The corresponding values are "harshit_jain52", "Harshit Jain", "UG Student", "CSE", "2026", and "B. Tech.". At the bottom of the window, there are three buttons: "Log Out" (light gray), "Edit Profile" (light gray), and "Deregister" (red).

User ID:	harshit_jain52
Name:	Harshit Jain
Profile Type:	UG Student
Department:	CSE
Graduation Year:	2026
Degree:	B. Tech.

[Log Out](#) [Edit Profile](#) [Deregister](#)

[UG student profile]

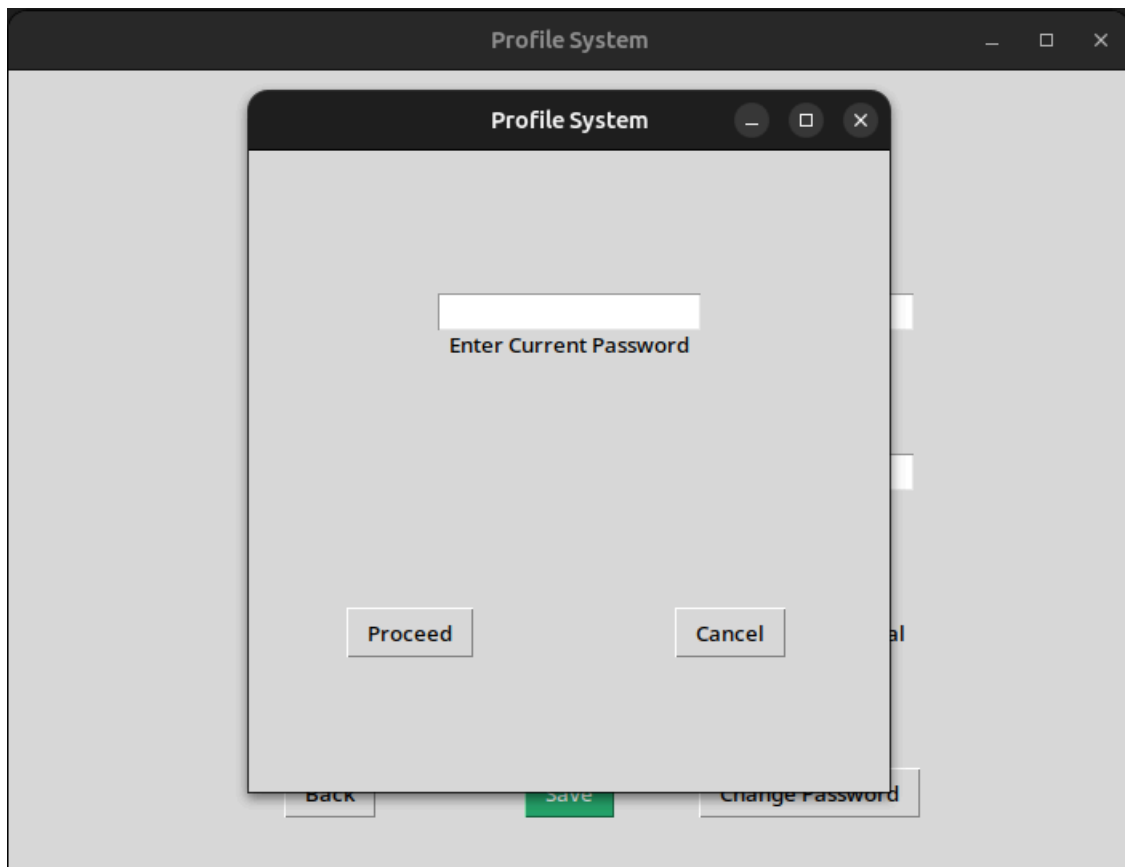


The screenshot shows the same "Profile System" window, but in edit mode. The profile information is displayed with input fields for "Name:", "Department:", and "Graduation Year:". The "Name:" field contains "Harshit Jain", "Department:" contains "CSE", and "Graduation Year:" contains "2026". The "Degree:" field has two radio button options: "B. Tech." (which is selected) and "Dual". At the bottom, there are three buttons: "Back" (light gray), "Save" (green), and "Change Password" (light gray).

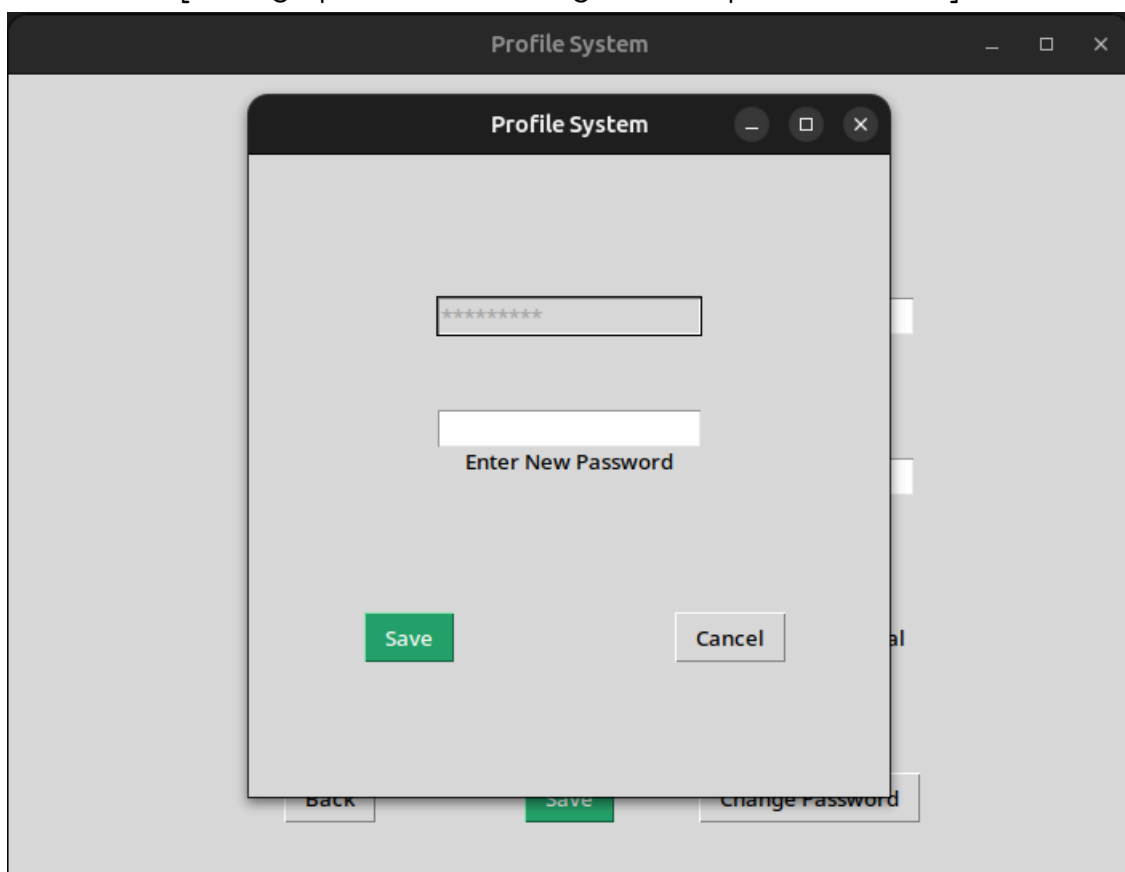
User ID:	harshit_jain52
Name:	<input type="text" value="Harshit Jain"/>
Profile Type:	UG Student
Department:	<input type="text" value="CSE"/>
Graduation Year:	<input type="text" value="2026"/>
Degree:	<input checked="" type="radio"/> B. Tech. <input type="radio"/> Dual

[Back](#) [Save](#) [Change Password](#)

[UG student profile editing]



[change password: asking current password first]



[change password: new password]

Profile System

User ID:

user123

Name:

Prof Amit

Profile Type:

Teacher

Department:

CSE

Course:

DSA

Log Out

Edit Profile

Deregister

[teacher profile]

Profile System

User ID:

user123

Name:

Prof Amit

Profile Type:

Teacher

Department:

CSE

Course:

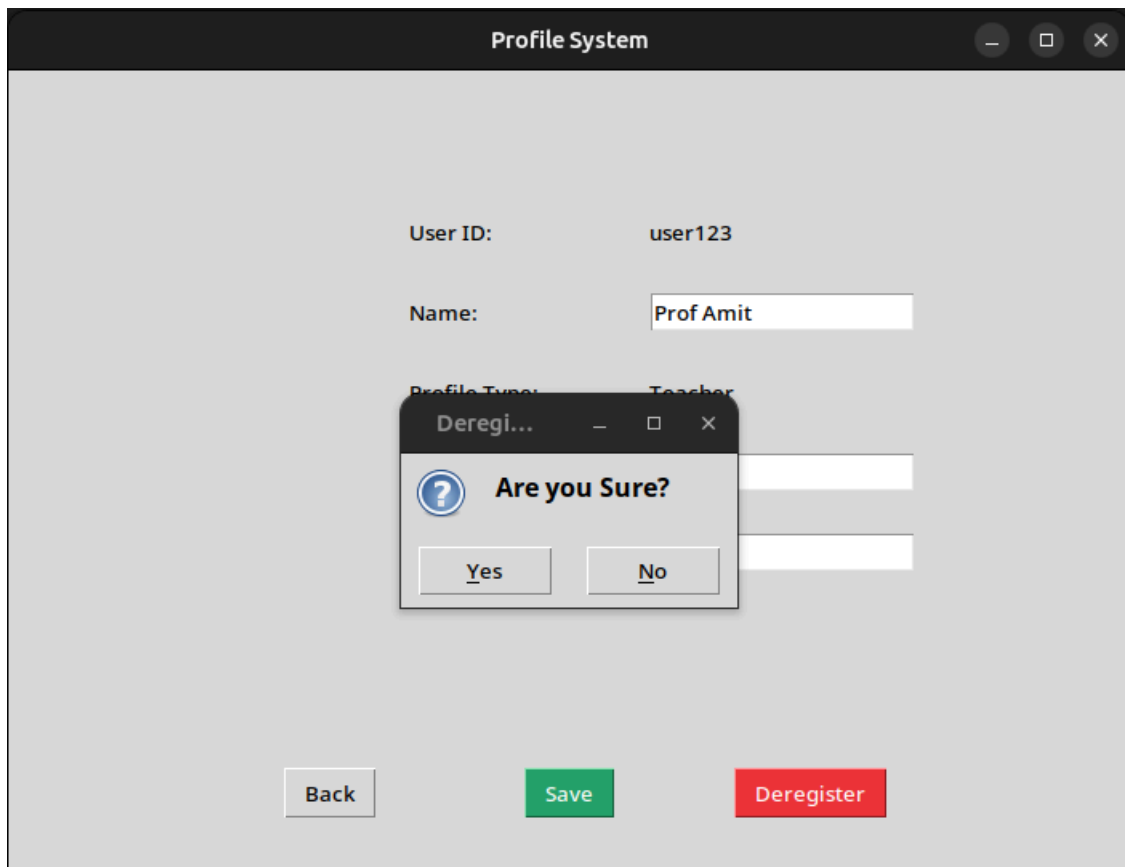
DSA

Back

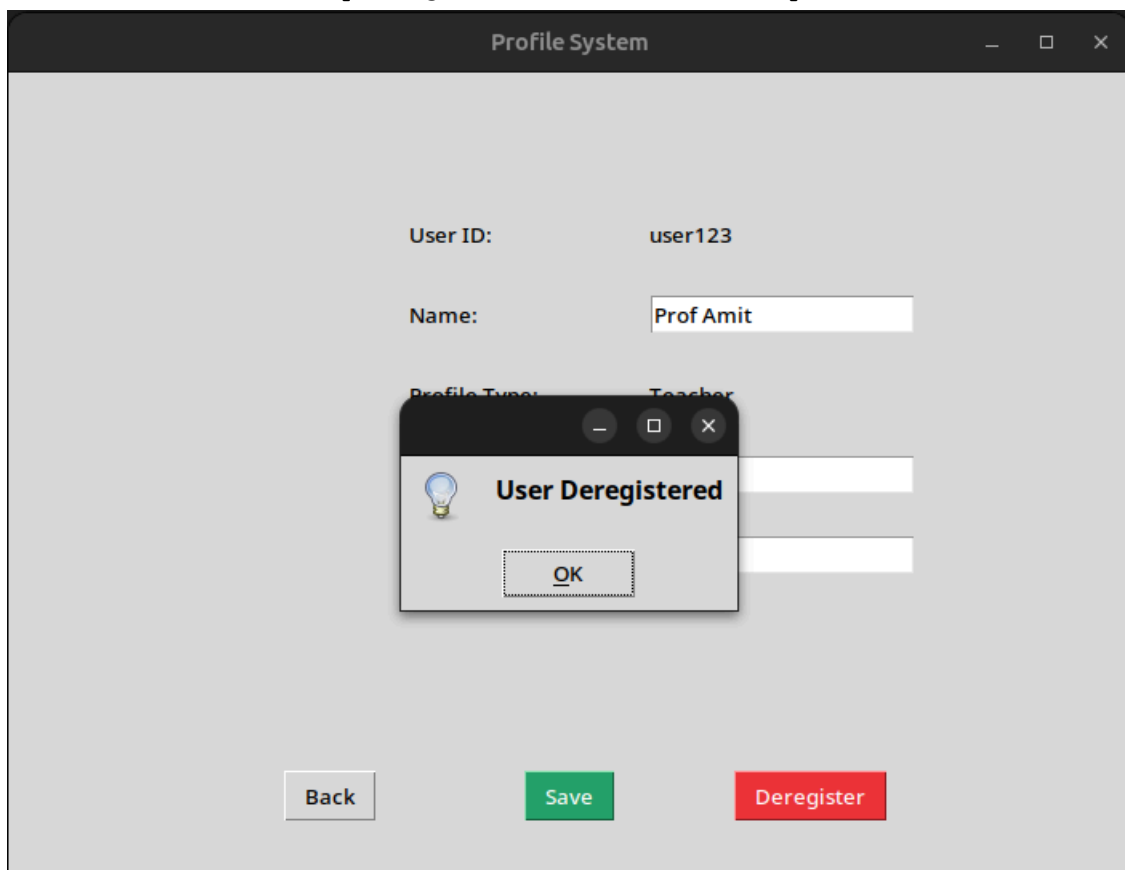
Save

Change Password

[teacher profile editing]



[deregistration confirmation]



[deregistration success message]

Profile System

User ID: pguserXYZ

Name: Peter

Profile Type: PG Student

Department: Physics

Graduation Year: 2026

Field:

Log Out Edit Profile Deregister

[PG student profile]

Profile System

User ID: pguserXYZ

Name:

Profile Type: PG Student

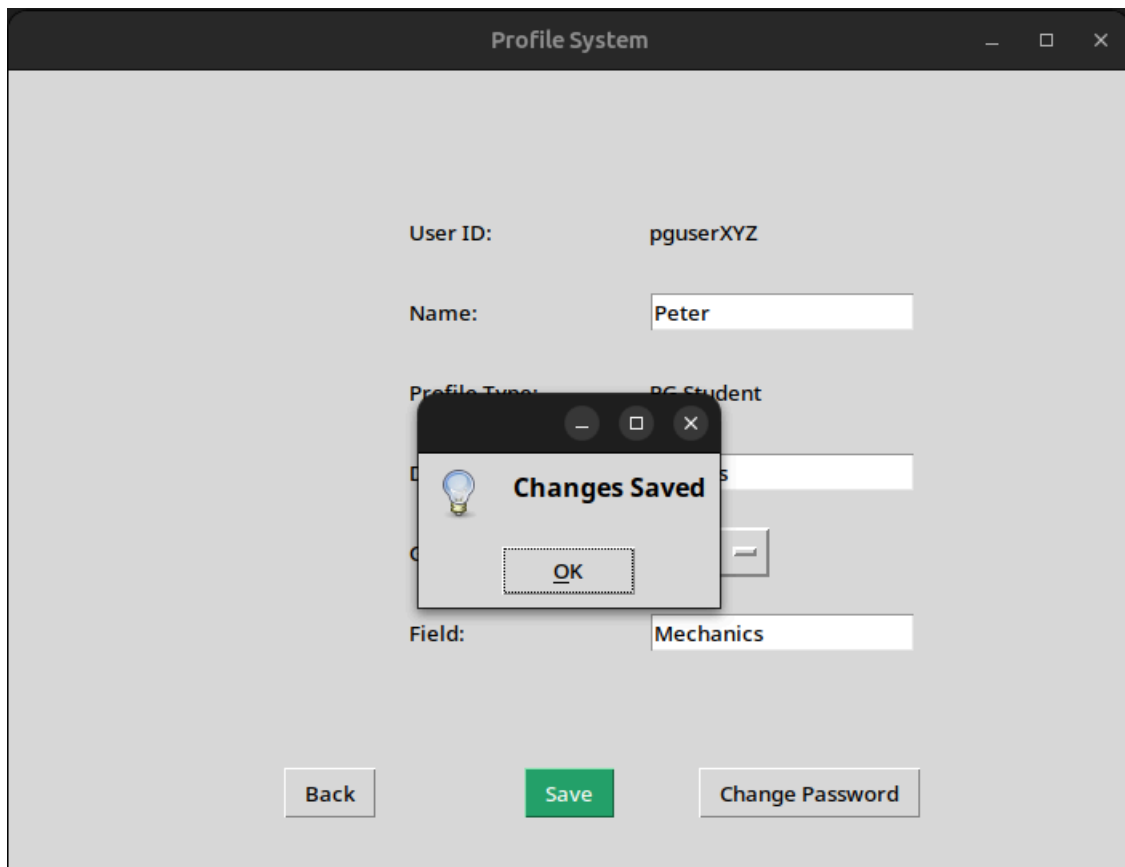
Department:

Graduation Year:

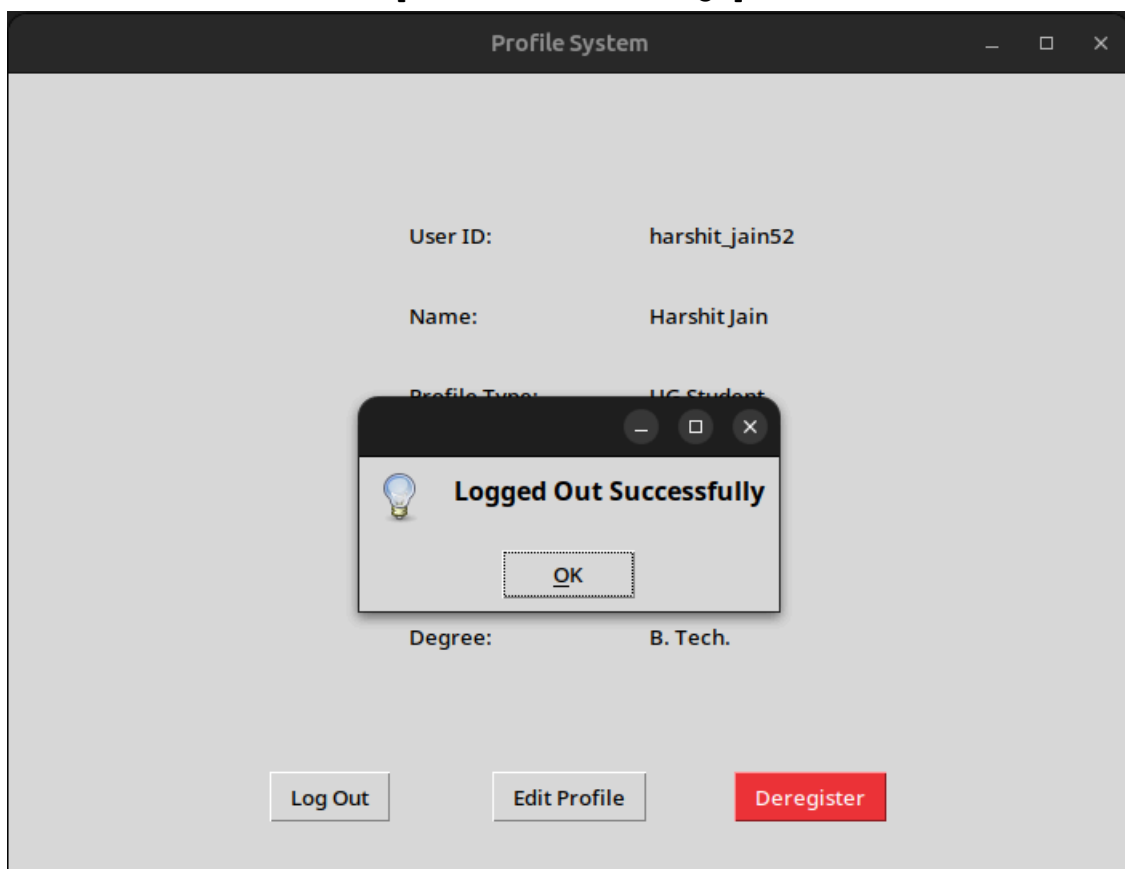
Field:

Back Save Change Password

[PG student profile editing]



[edits saved message]



[logout success message]