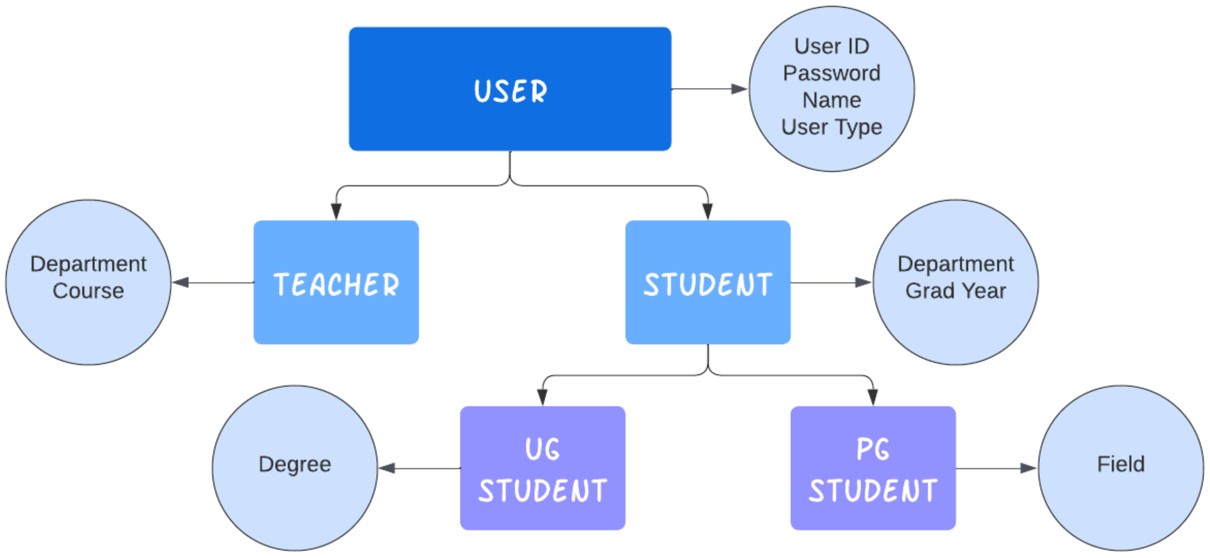
Software Application Development Using Python

Python Libraries Used:

- ➔ tkinter
- ➔ json
- ➔ re
- ➔ datetime

Class Hierarchy (`hierarchy.py`):

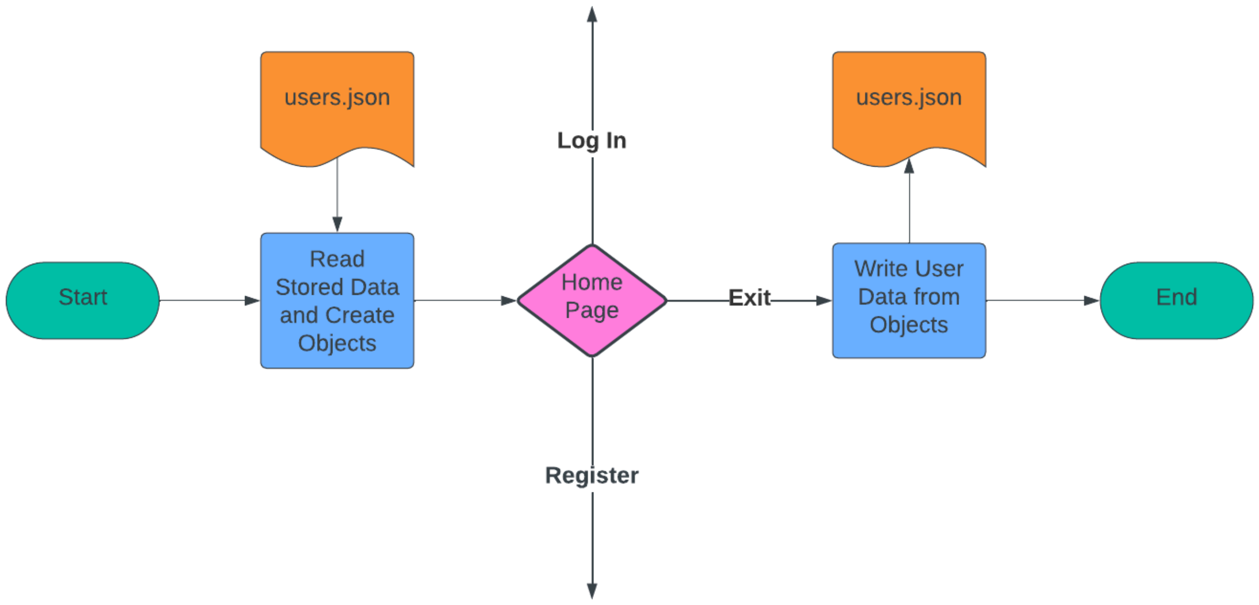


Classes and subclasses are implemented according to the above map.
In addition, the `User` class contains an instance attribute `attempts` (this is instantiated as 3 at the start of the application and reset after every login) to keep a count of login attempts, and every class contains a class attribute `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 resizeable) 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.

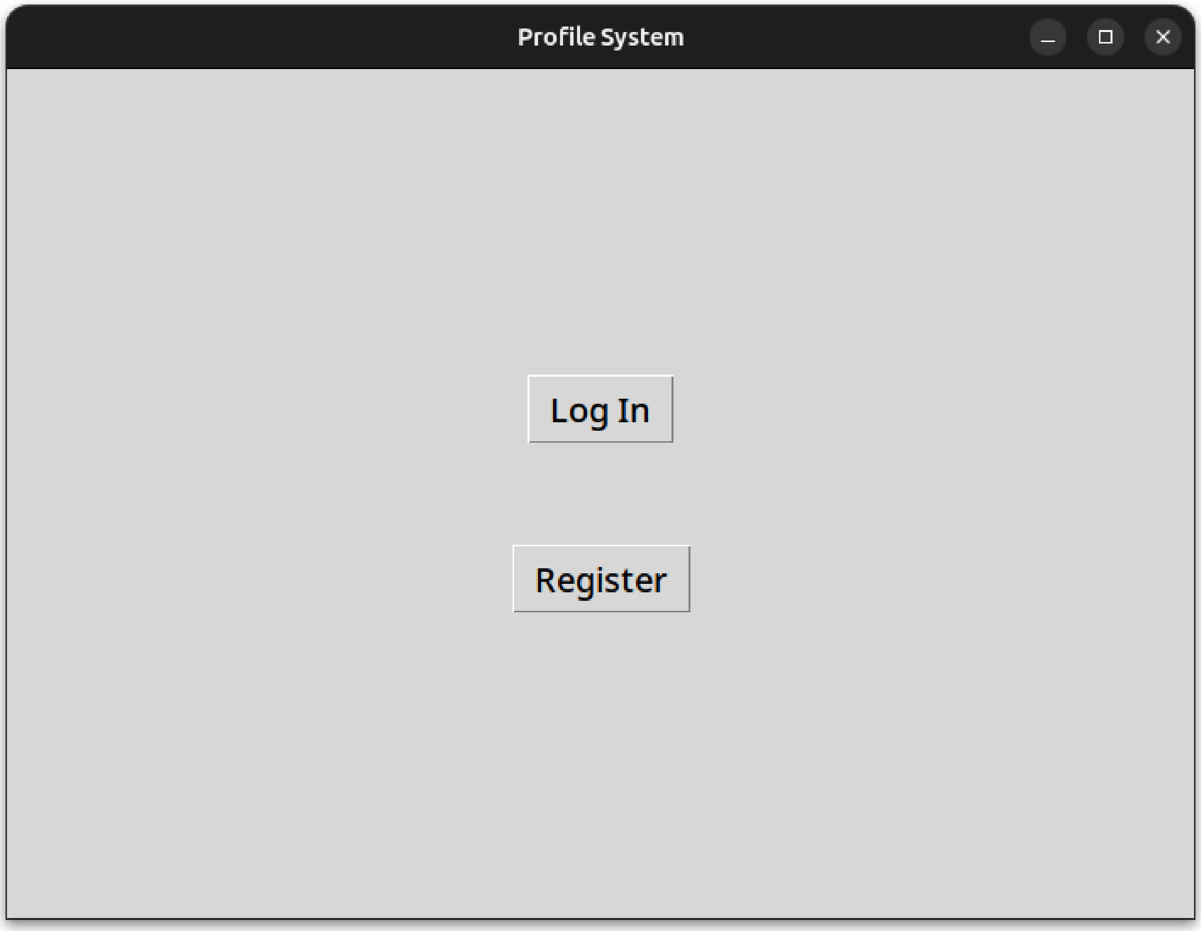
➔ Home Page:

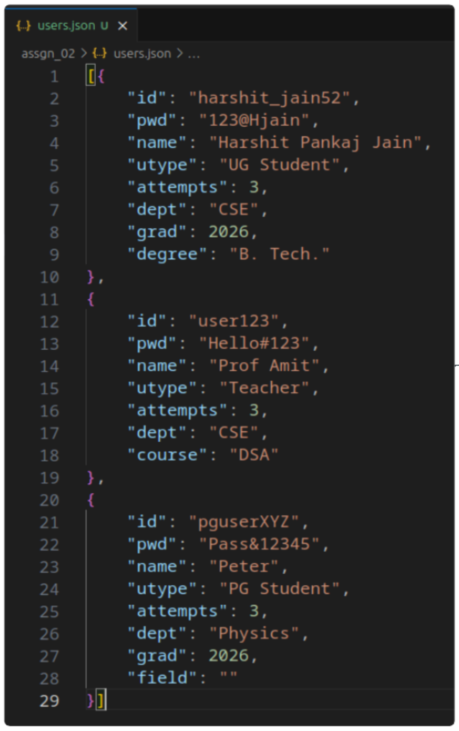

users.json
Log In
users.json
Start
Read Stored Data and Create Objects
Home Page
Exit
Write User Data from Objects
End
Register

Functions:

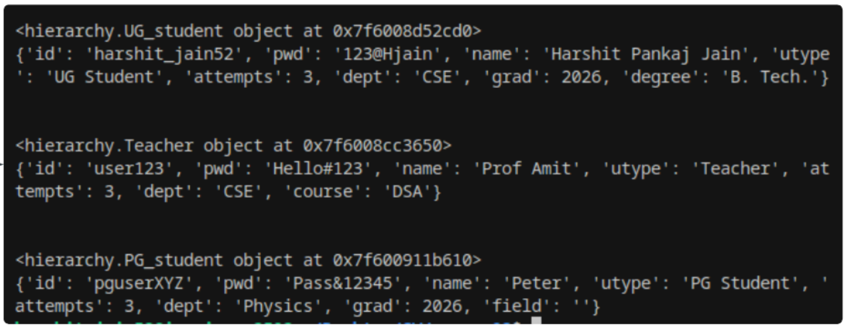
- `download_user_data()`
- `upload_user_data()`
- `home()`

Screenshots:





```
[{
    "id": "harshit_jain52",
    "pwd": "123@Hjain",
    "name": "Harshit Pankaj Jain",
    "utype": "UG Student",
    "attempts": 3,
    "dept": "CSE",
    "grad": 2026,
    "degree": "B. Tech."
},
{
    "id": "user123",
    "pwd": "Hello#123",
    "name": "Prof Amit",
    "utype": "Teacher",
    "attempts": 3,
    "dept": "CSE",
    "course": "DSA"
},
{
    "id": "pguserXYZ",
    "pwd": "Pass&12345",
    "name": "Peter",
    "utype": "PG Student",
    "attempts": 3,
    "dept": "Physics",
    "grad": 2026,
    "field": ""
}]
```

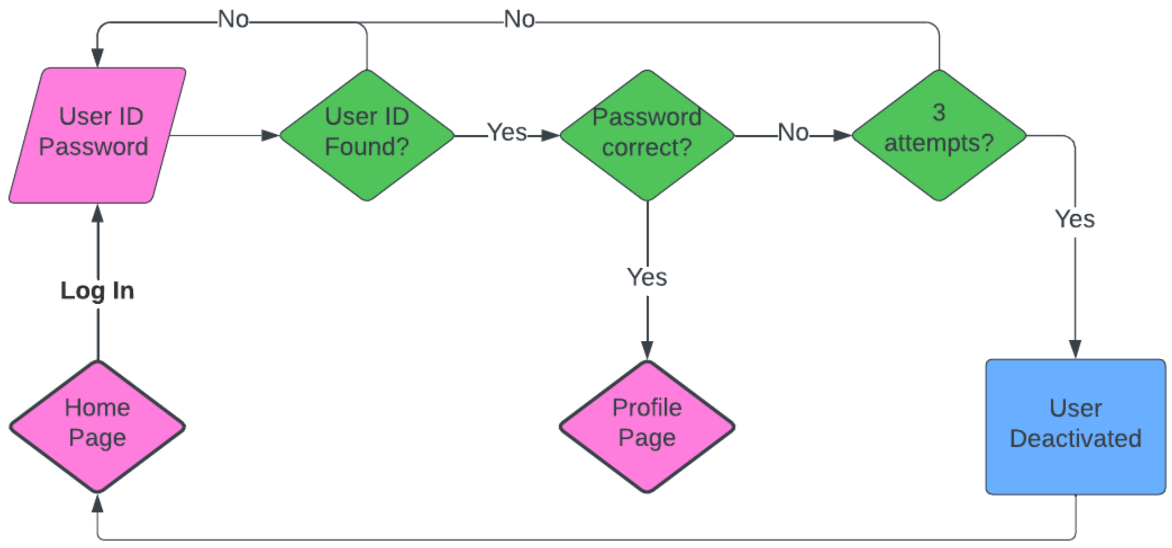


```
<hierarchy.UG_student object at 0x7f6008d52cd0>
{'id': 'harshit_jain52', 'pwd': '123@Hjain', 'name': 'Harshit Pankaj Jain', 'utype
': 'UG Student', 'attempts': 3, 'dept': 'CSE', 'grad': 2026, 'degree': 'B. Tech.'}


<hierarchy.Teacher object at 0x7f6008cc3650>
{'id': 'user123', 'pwd': 'Hello#123', 'name': 'Prof Amit', 'utype': 'Teacher', 'at
tempts': 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:

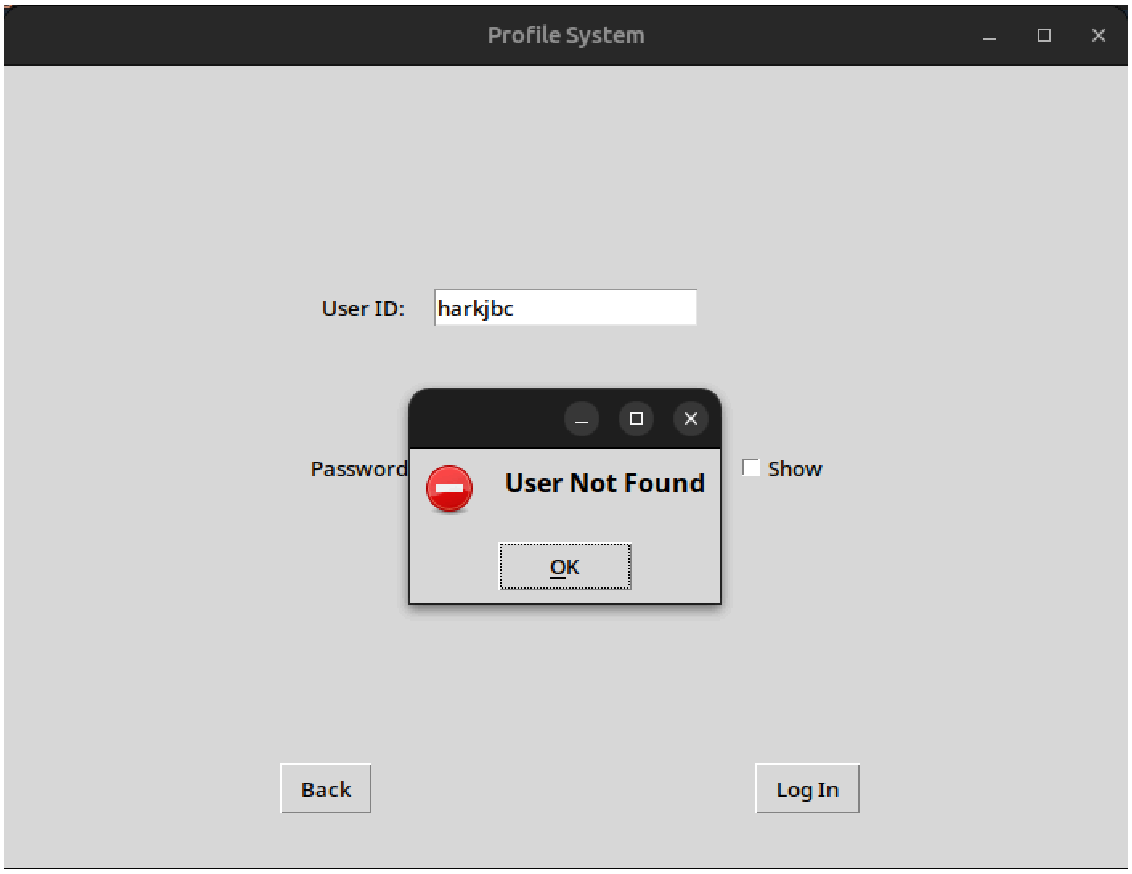


Entered User ID is searched for in the `User.all` list and the corresponding index is found out.
This index is used for password verification, for updating attempt count, displaying profile, and for deactivating the account.
Show/Hide password functionality is implemented using `Checkbutton` widget.
Errors and information are displayed using `messagebox`.

Functions:

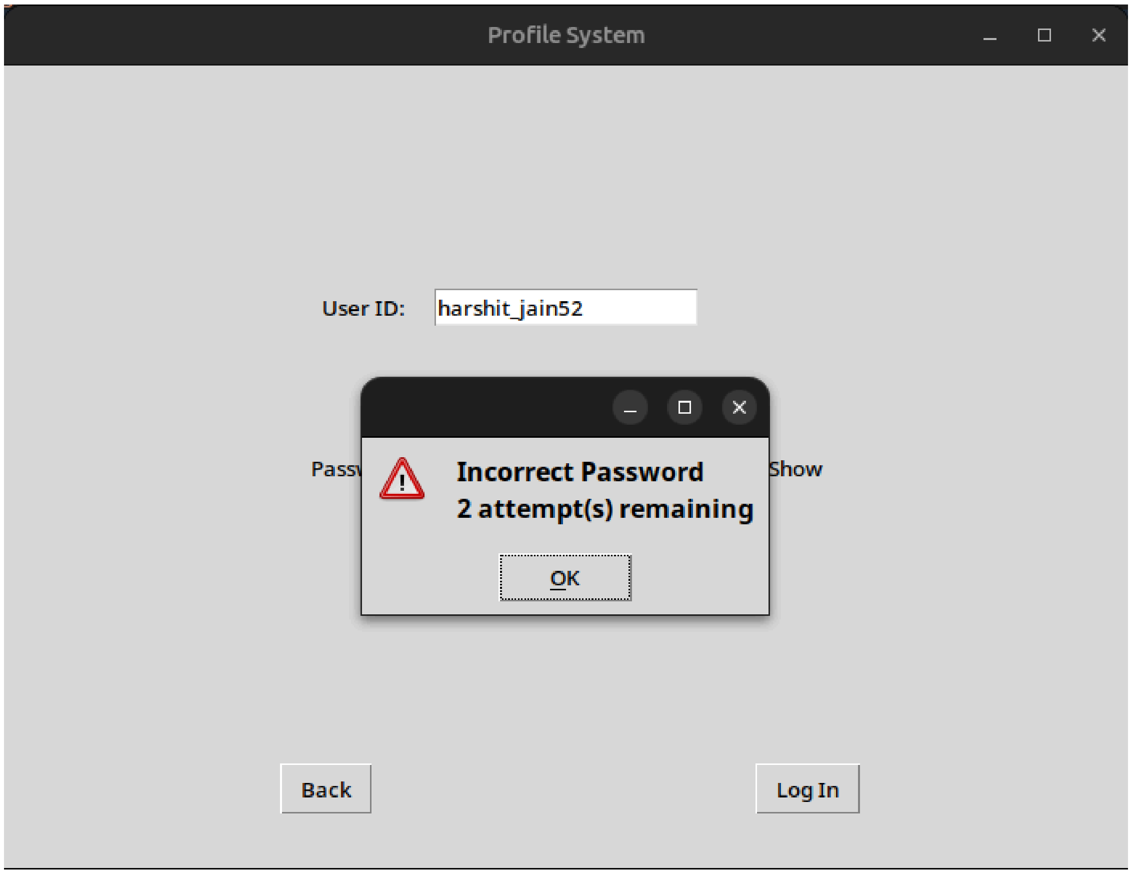
- `login()`
- `create_id_entry()`
- `create_pwd_entry()`
- `show_pwd()`
- `verify_login()`
- `back()`

Screenshots:



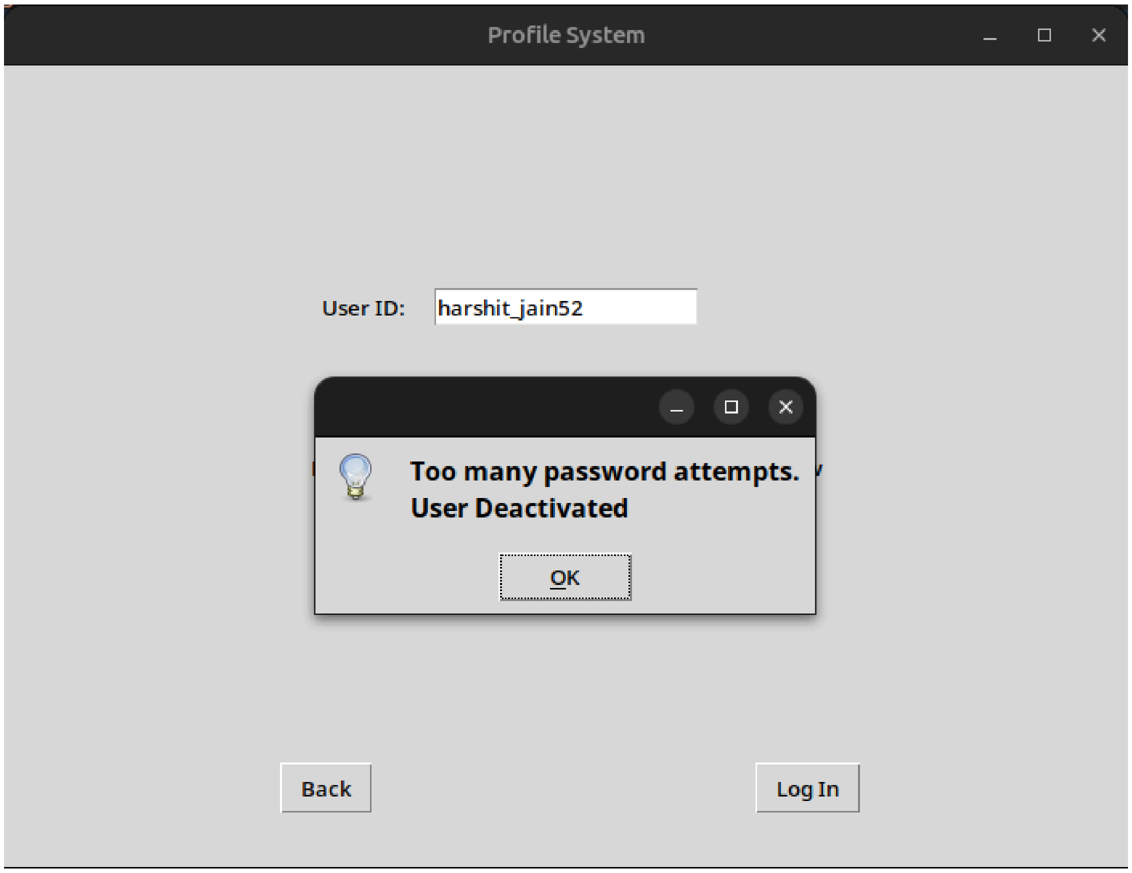
User Not Found

[user not found error]



Incorrect Password
2 attempt(s) remaining

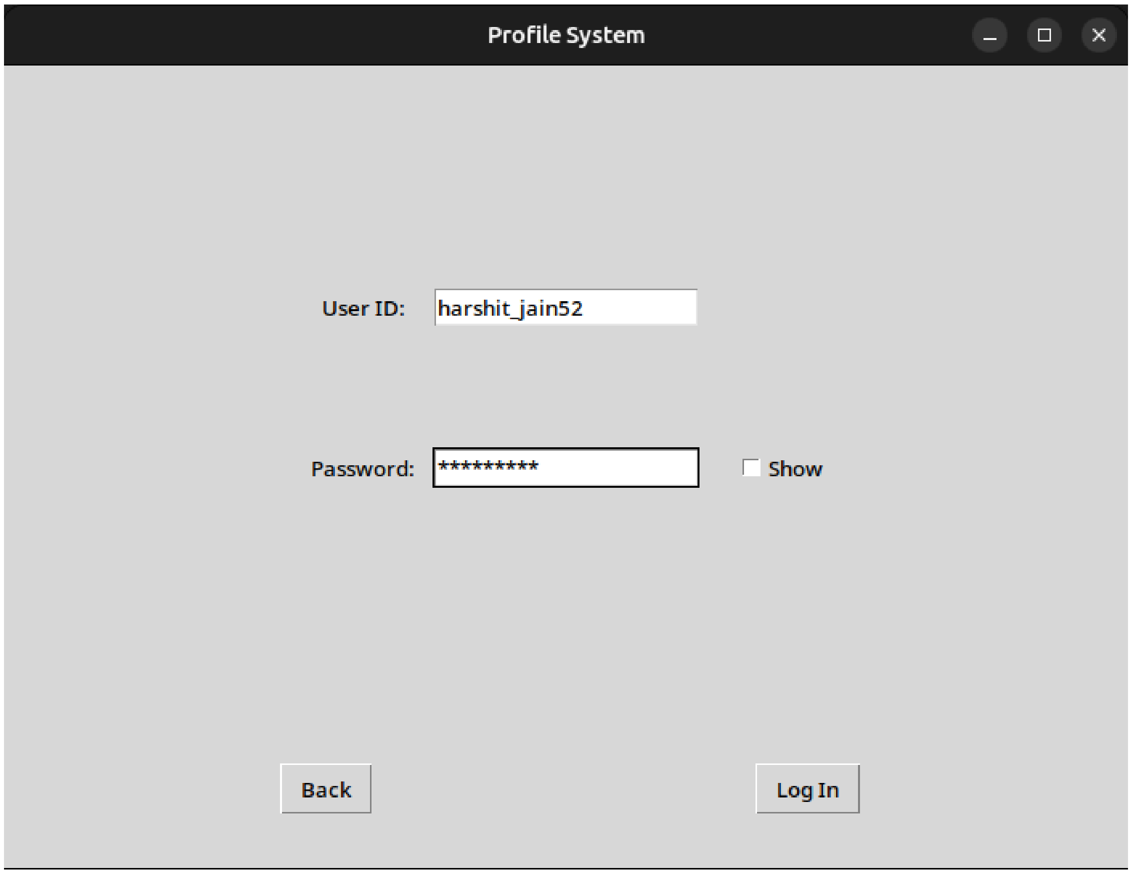
[incorrect password error]


Profile System

User ID: harshit_jain52

Too many password attempts.
User Deactivated

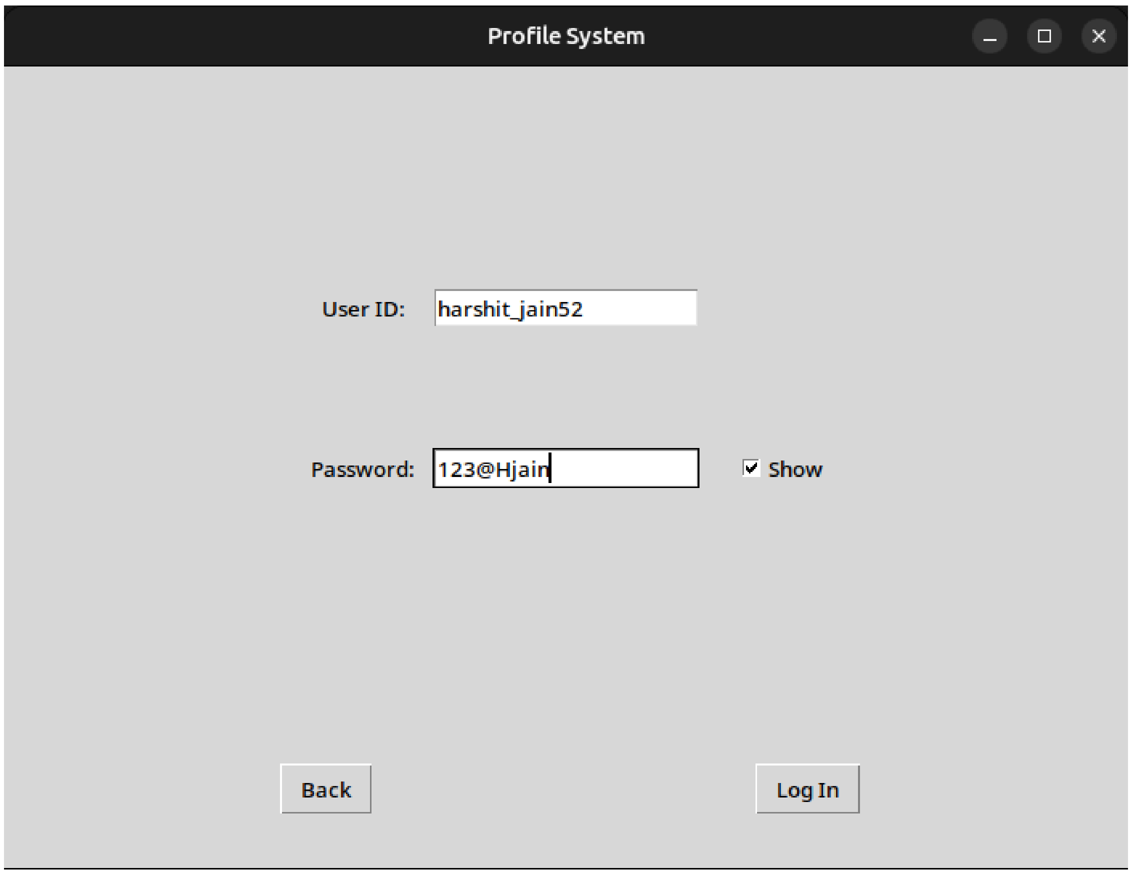
[deactivation due to 3 attempts]


Profile System

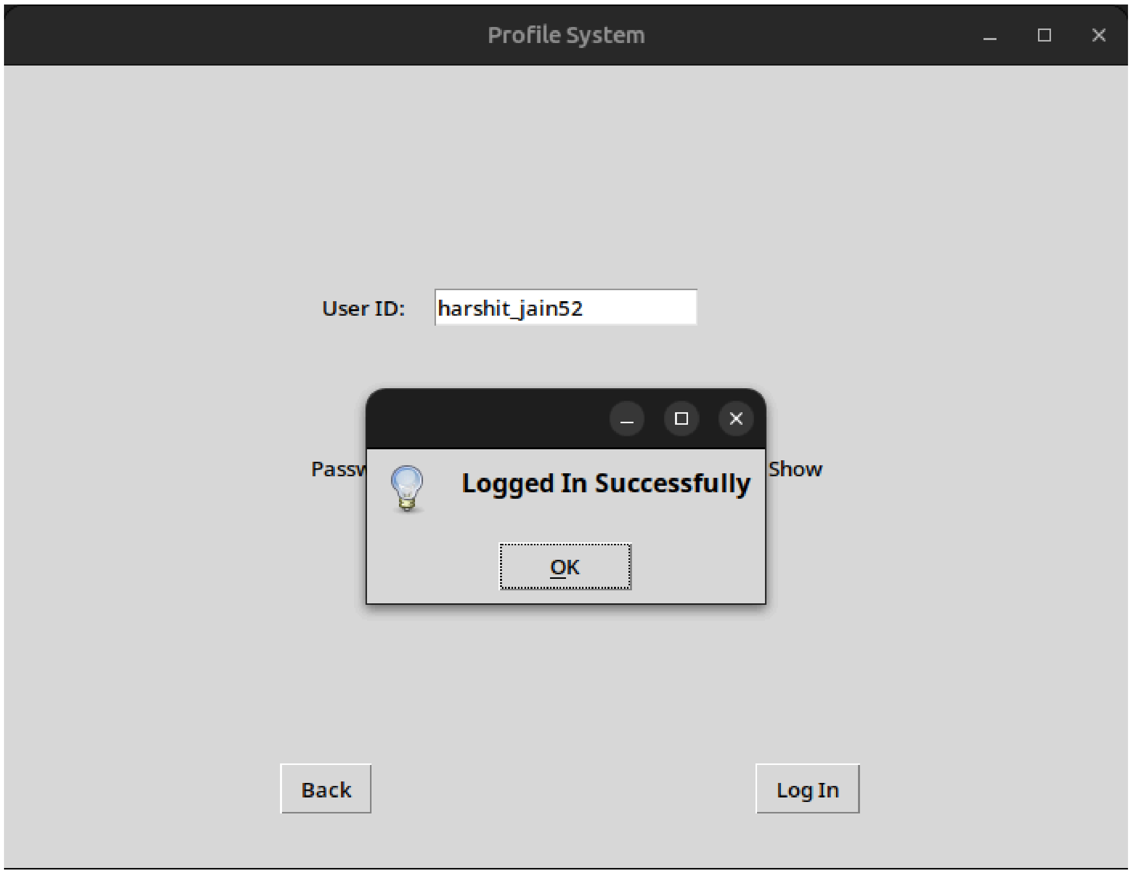
User ID: harshit_jain52

Password: *********

[hidden password]



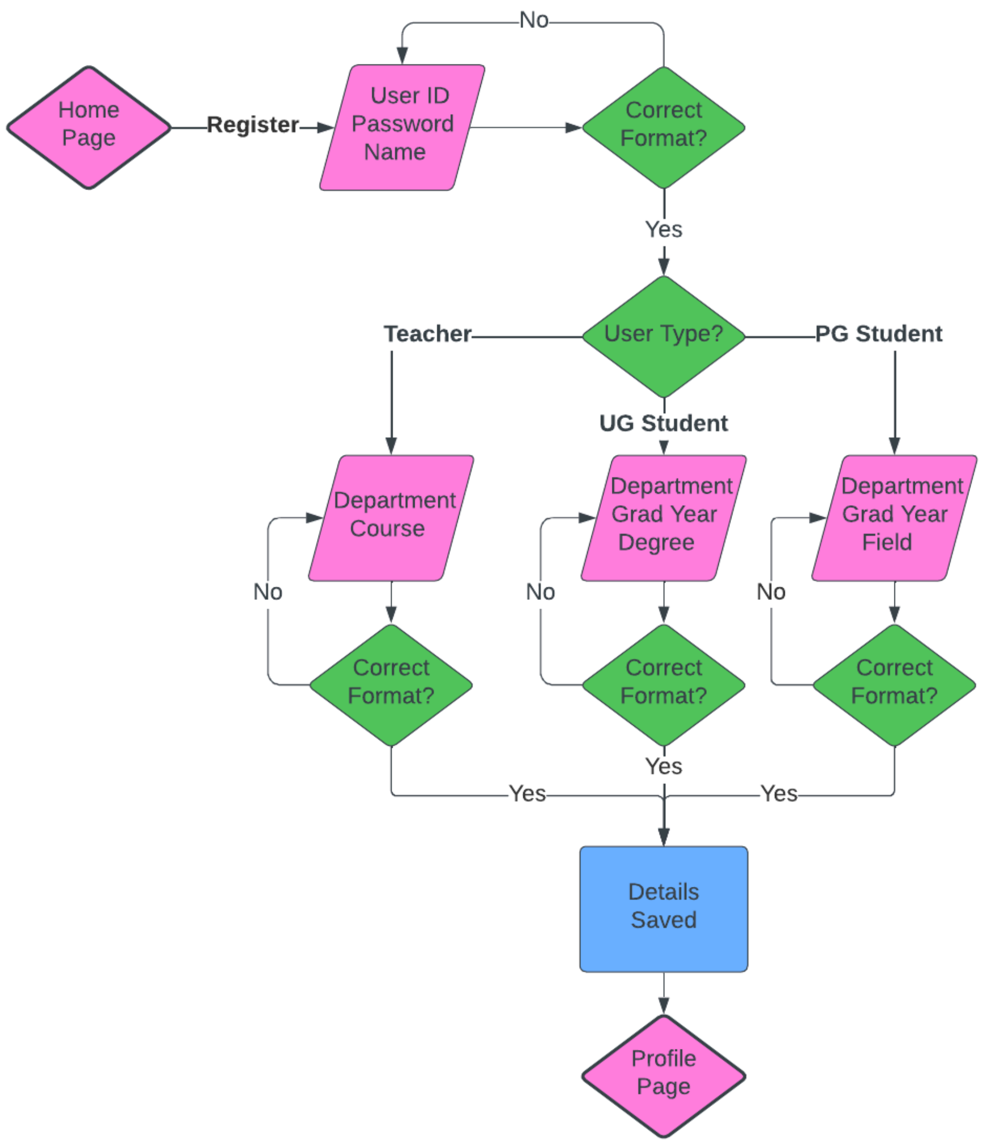
[unhidden password]



Logged In Successfully

[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 !@#$%&* 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 `OptionMenu` widget
Department	Can't be blank	Calculating length
Course	none	-
UG Grad Year	Must be from (current year) to (current year + 4)	Using `OptionMenu` widget, and the `datetime` library
PG Grad Year	Must be from (current year) to (current year + 2)	Using `OptionMenu` widget, and the `datetime` library
Degree	Must be among [B. Tech., Dual]	Using `Radiobutton` widget
Field	none	-

*REGEX used:

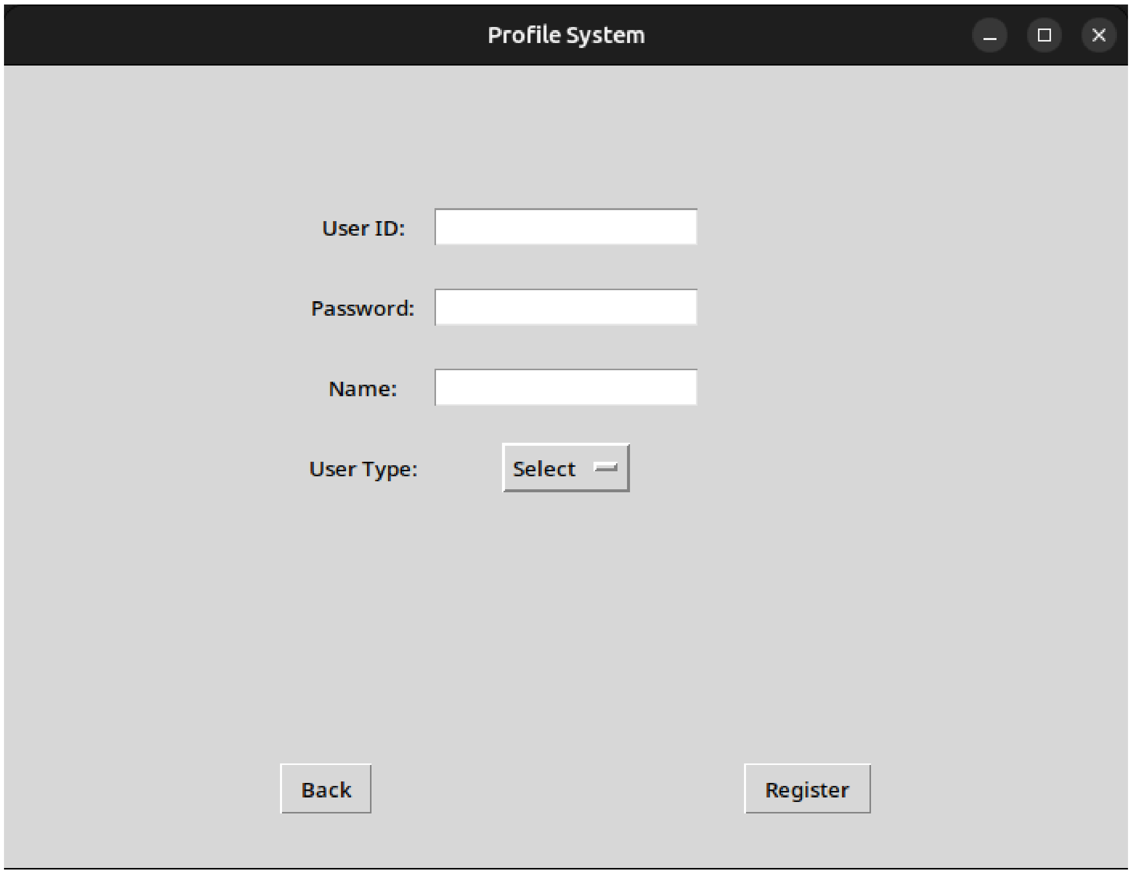
- password must match:

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

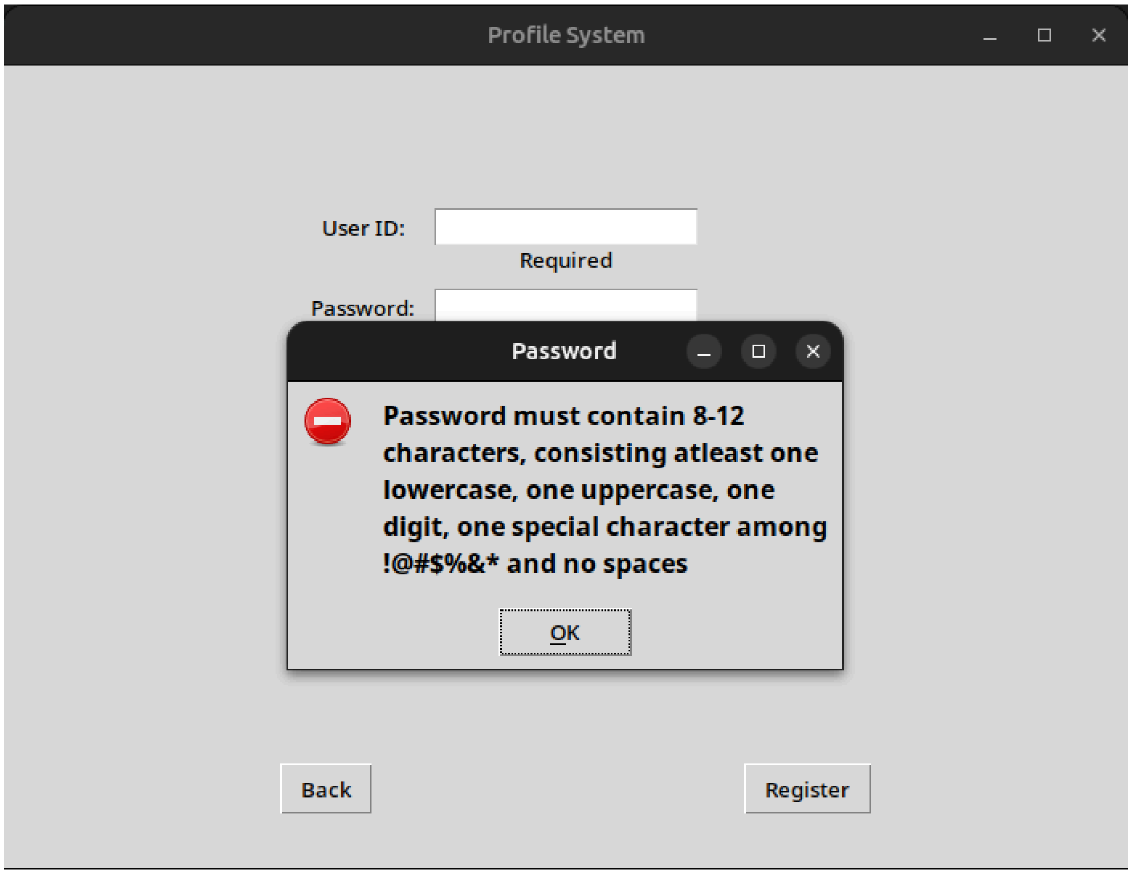
- and must not match:

```
[\s]
```

Screenshots:

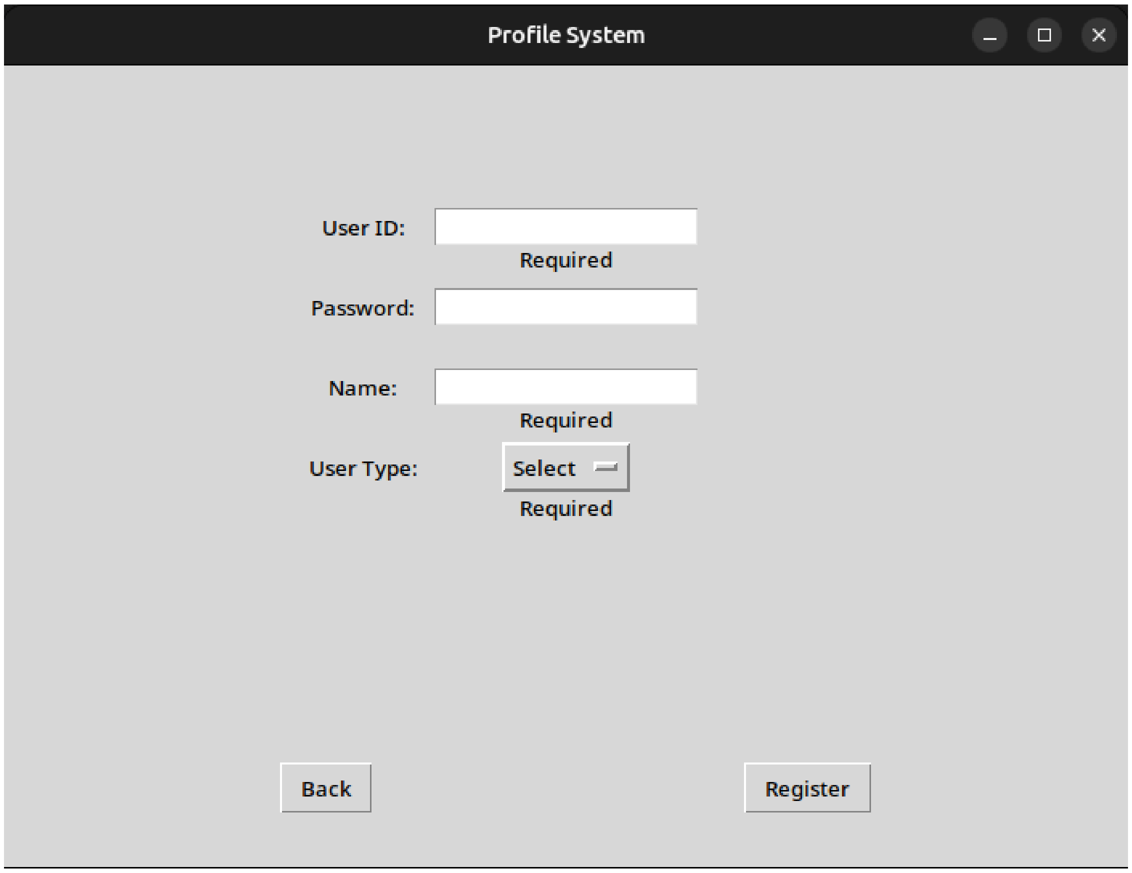


[registration page before selecting user type]



Password must contain 8-12 characters, consisting atleast one lowercase, one uppercase, one digit, one special character among !@#$%&* and no spaces

[password format]


Profile System

User ID:

Required

Password:

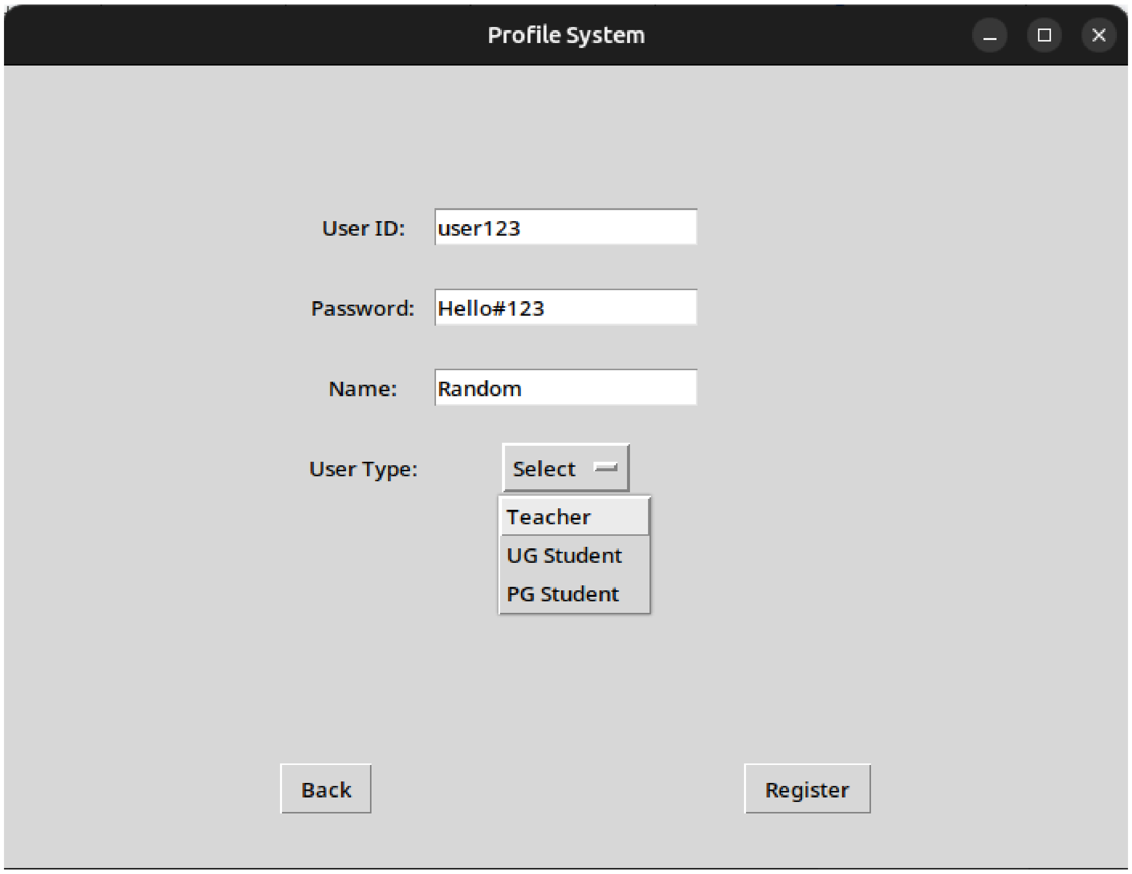
Name:

Required

User Type: Select

Required

[required fields are marked when user tries to register]


Profile System

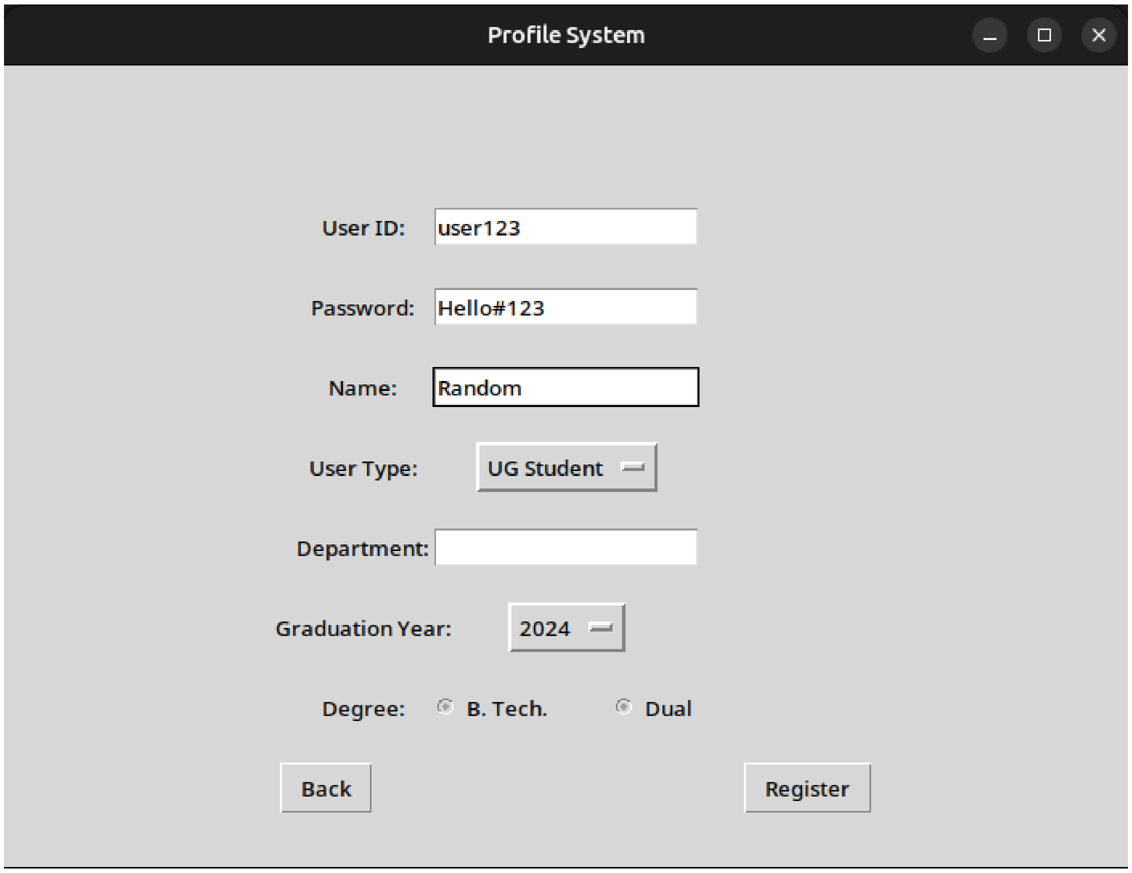
User ID: user123

Password: Hello#123

Name: Random

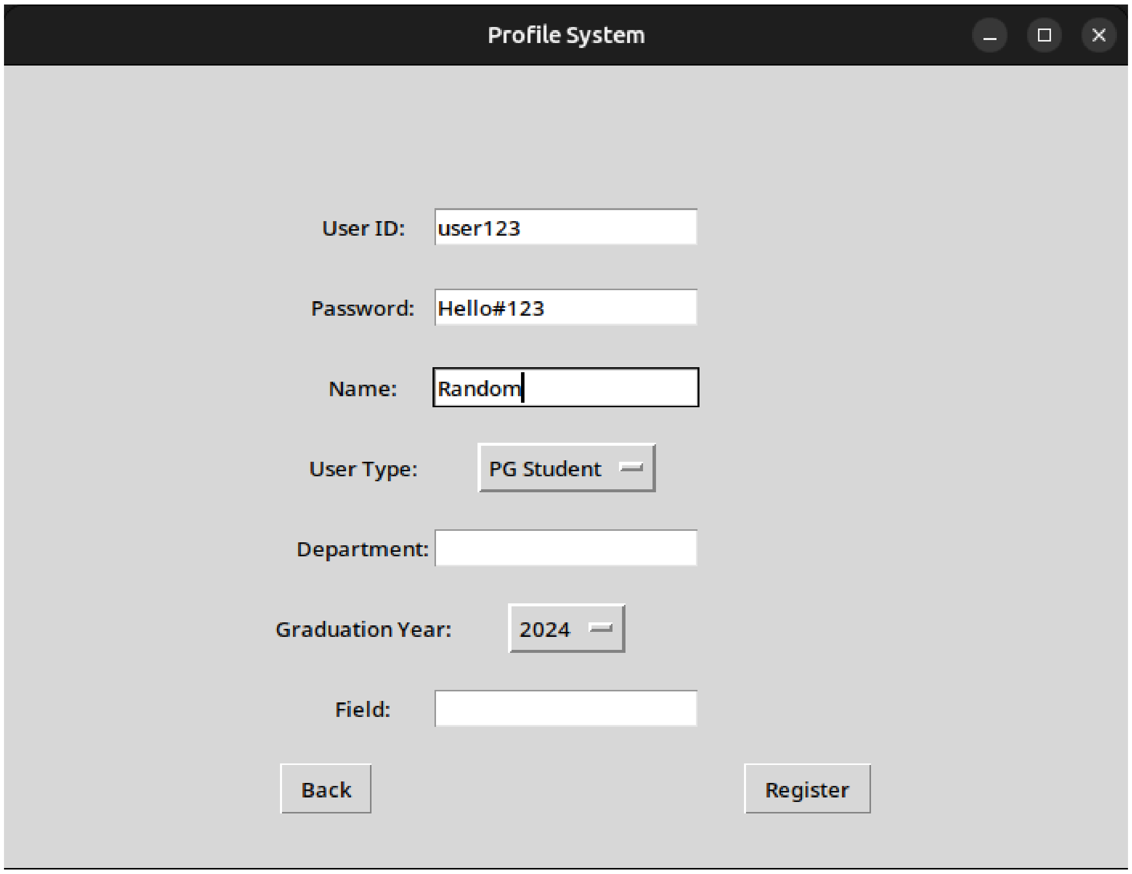
User Type: Select

[user type menu]


Profile System

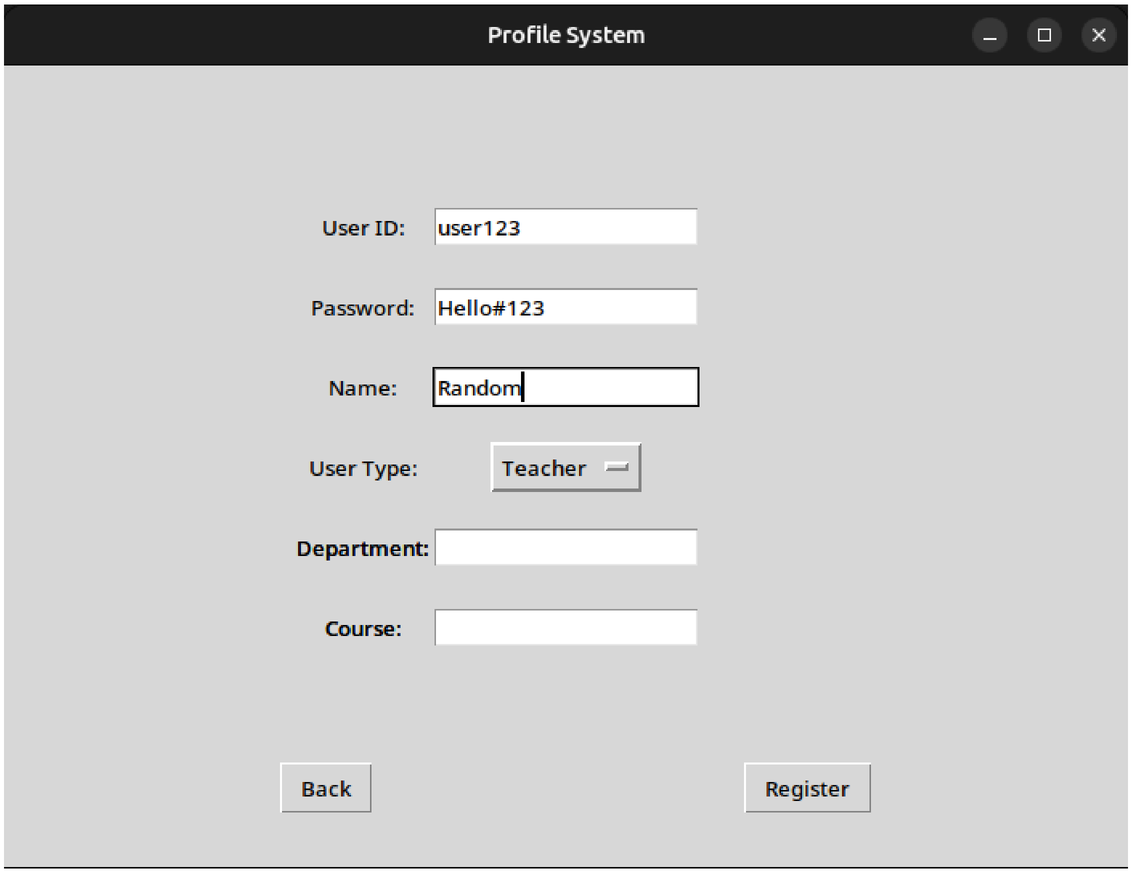
User ID:	user123
Password:	Hello#123
Name:	Random
User Type:	UG Student
Department:	
Graduation Year:	2024
Degree:	B. Tech. / Dual

[UG student registration]


Profile System

User ID:	user123
Password:	Hello#123
Name:	Random
User Type:	PG Student
Department:	
Graduation Year:	2024
Field:	

[PG student registration]


Profile System

User ID: user123

Password: Hello#123

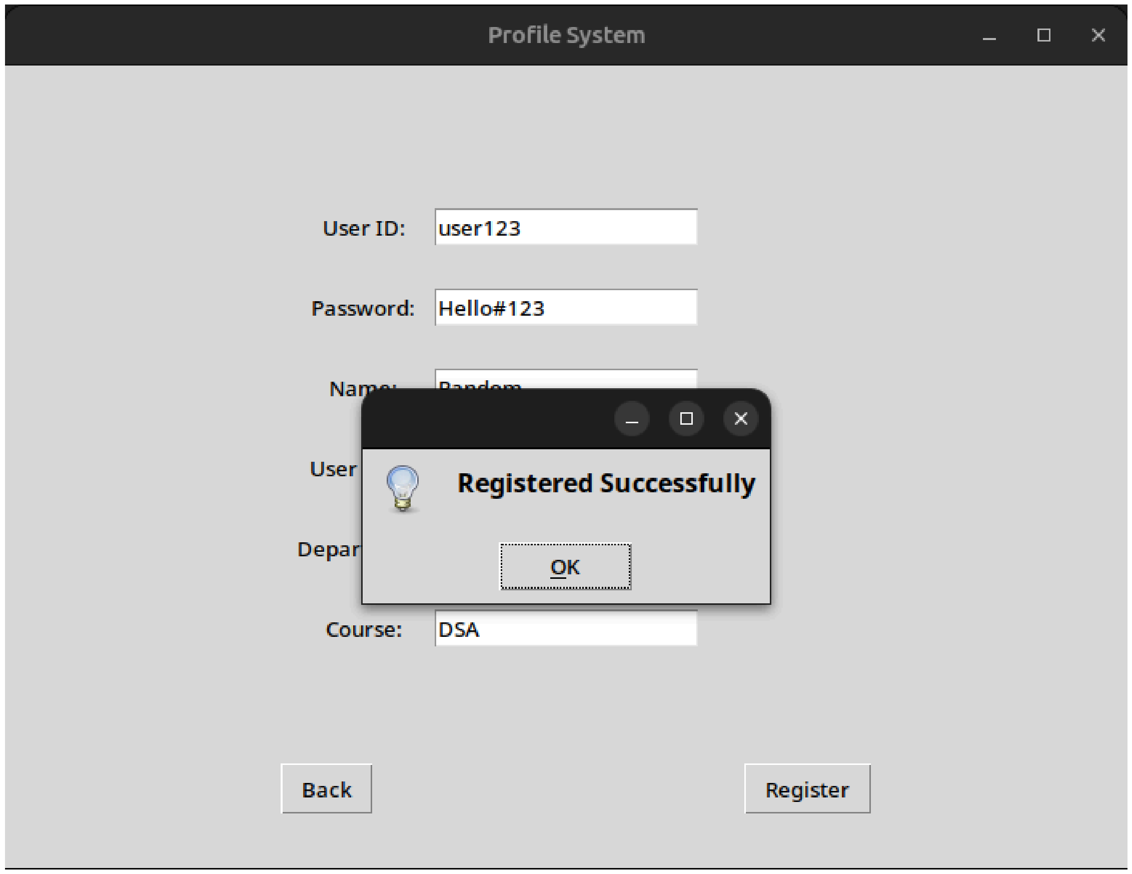
Name: Random

User Type: Teacher

Department:

Course:

[teacher registration]


Profile System

User ID: user123

Password: Hello#123

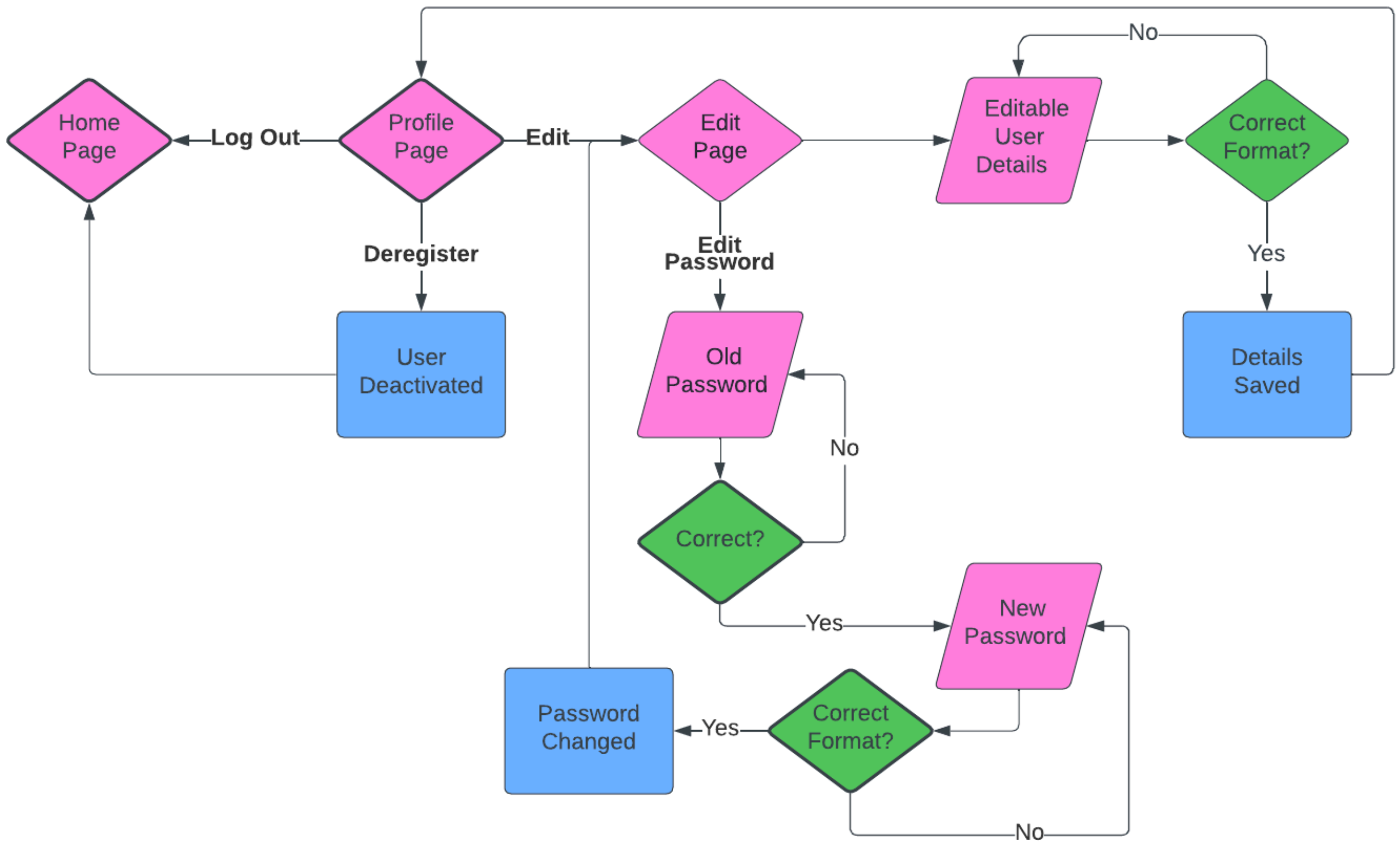
Name: Random

Course: DSA

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:

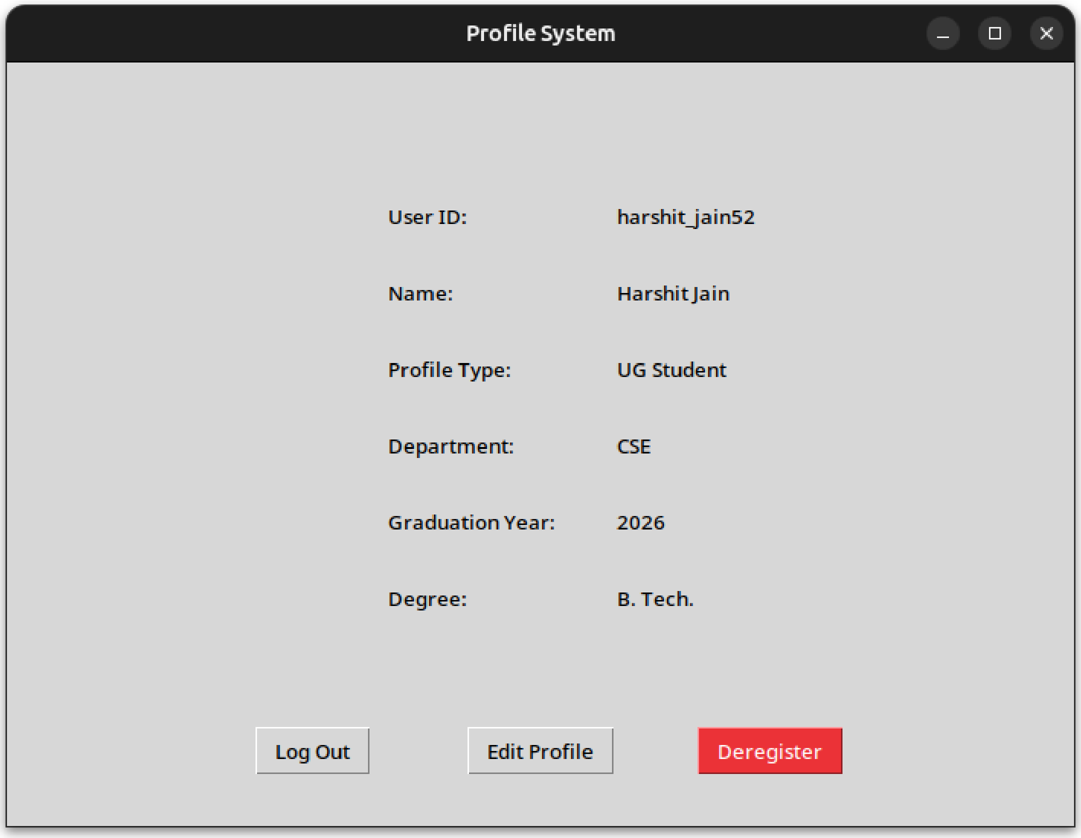
- Password
- Name
- Department
- Course
- Grad Year
- Degree
- Field

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

Functions:

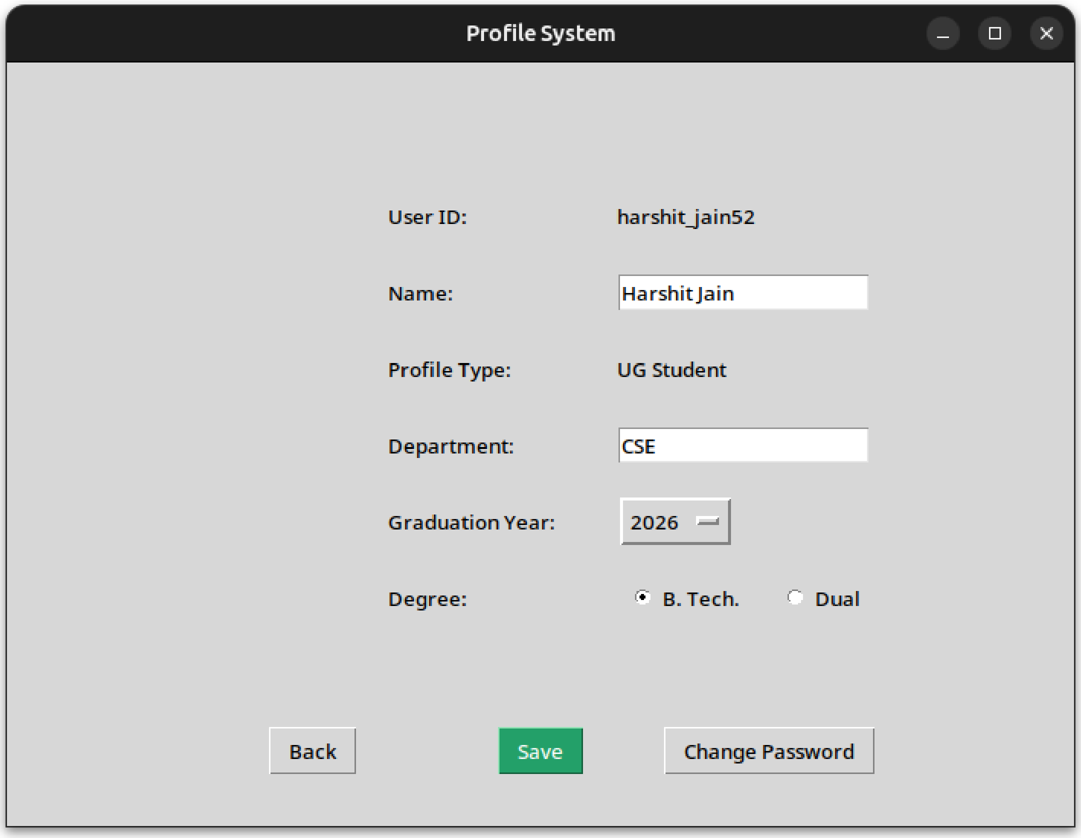
- `profile()`
- `edit_profile()`
- `create_name_entry()`
- `create_dept_entry()`
- `create_course_entry()`
- `create_grad_menu()`
- `create_degree_entry()`
- `create_field_entry()`
- `create_pwd_entry()`
- `change_pwd()`
- `get_new_pwd()`
- `verify_pwd_pattern()`
- `save_new_pwd()`
- `save_edits()`
-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

Screenshots:



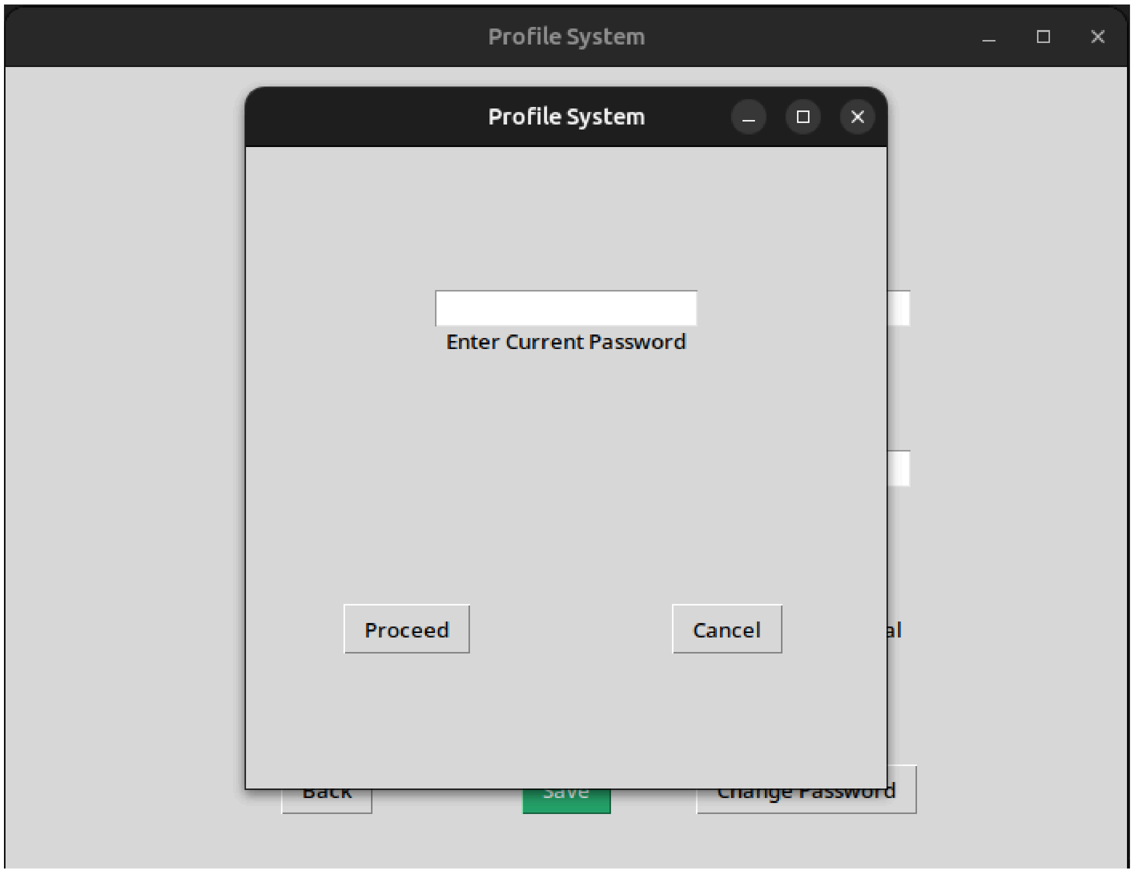
User ID:	harshit_jain52
Name:	Harshit Jain
Profile Type:	UG Student
Department:	CSE
Graduation Year:	2026
Degree:	B. Tech.

[UG student profile]

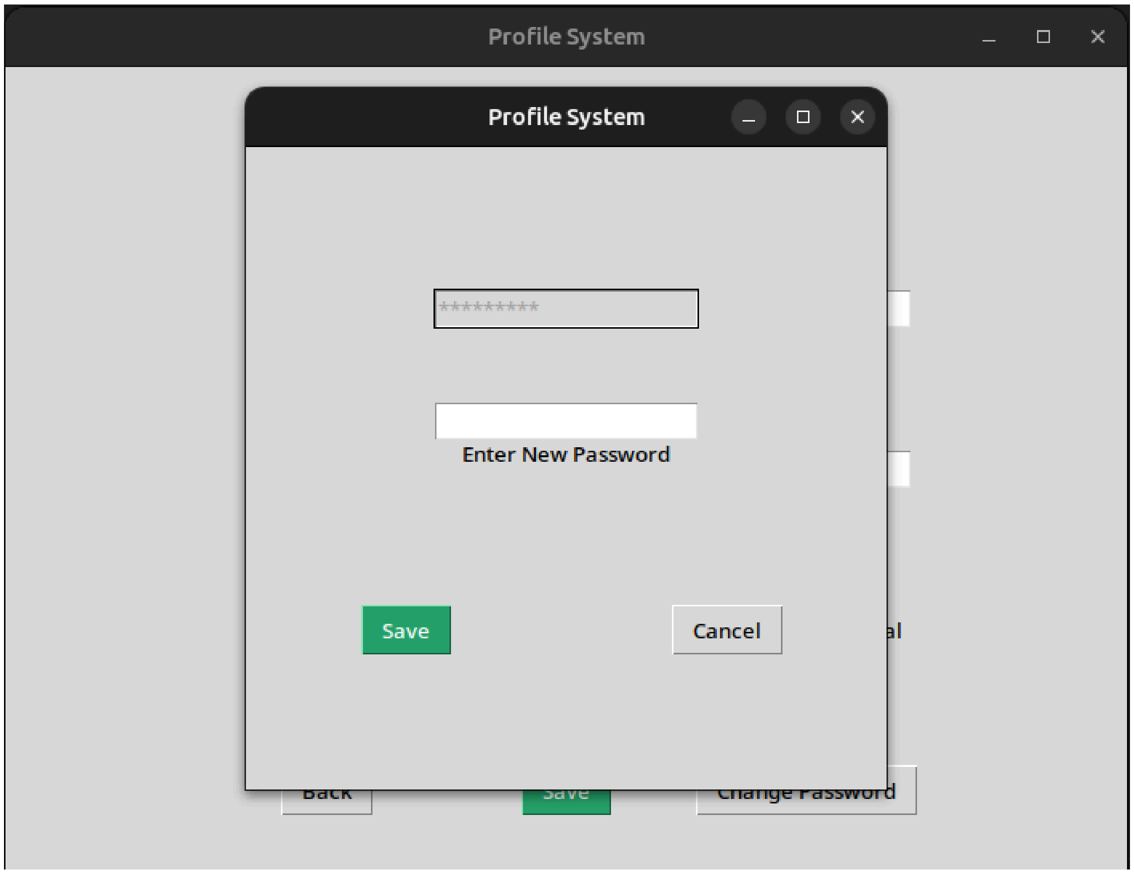


User ID:	harshit_jain52
Name:	Harshit Jain
Profile Type:	UG Student
Department:	CSE
Graduation Year:	2026
Degree:	[✓ B. Tech.] Dual

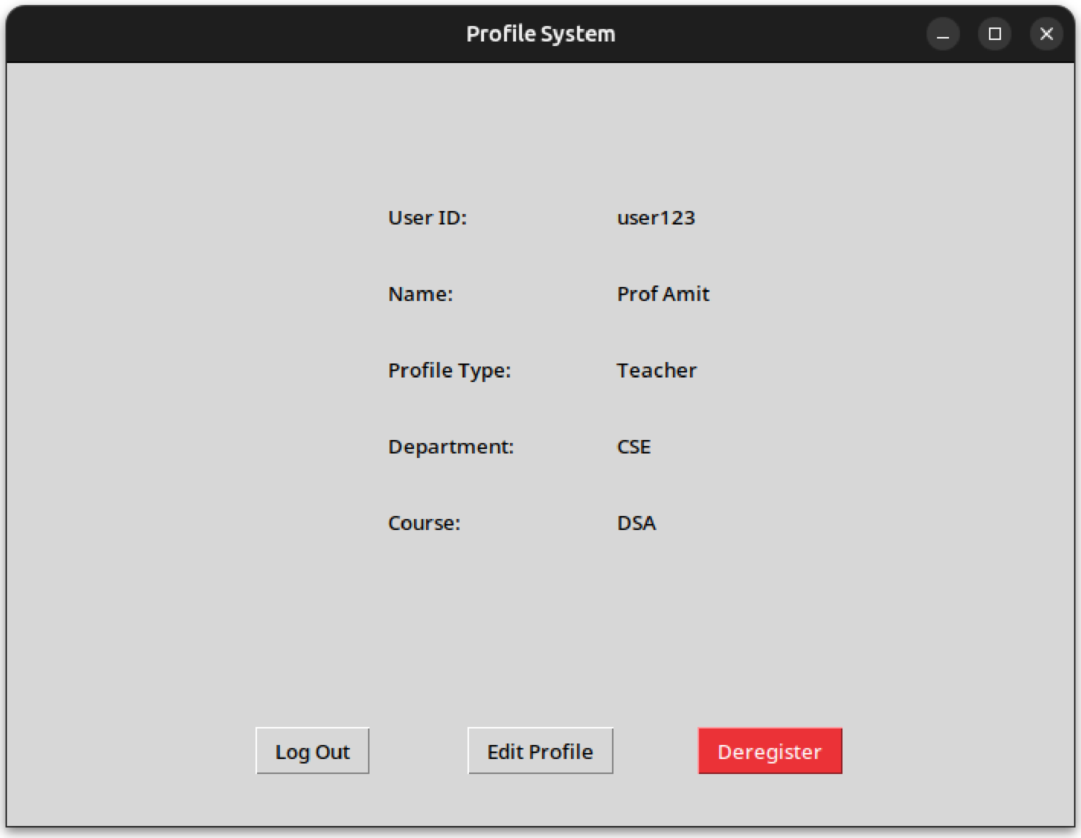
[UG student profile editing]



[change password: asking current password first]

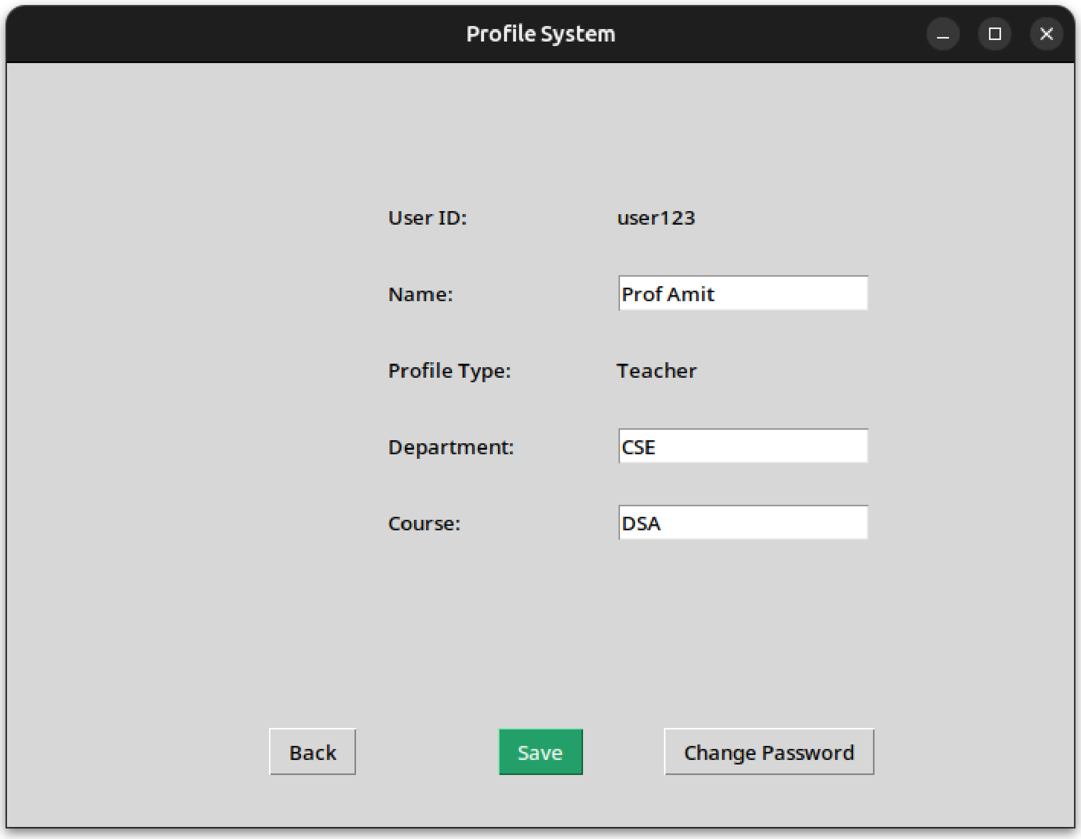


[change password: new password]


Profile System

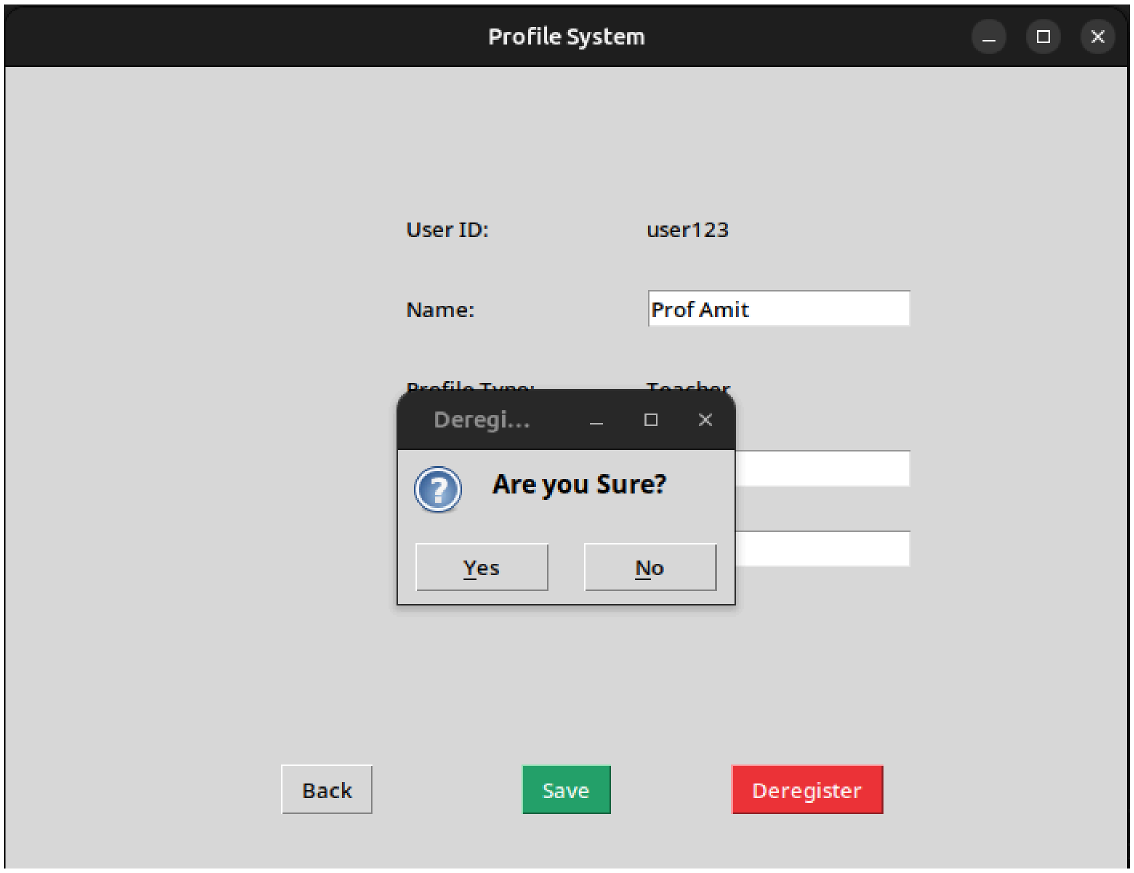
User ID:	user123
Name:	Prof Amit
Profile Type:	Teacher
Department:	CSE
Course:	DSA

[teacher profile]

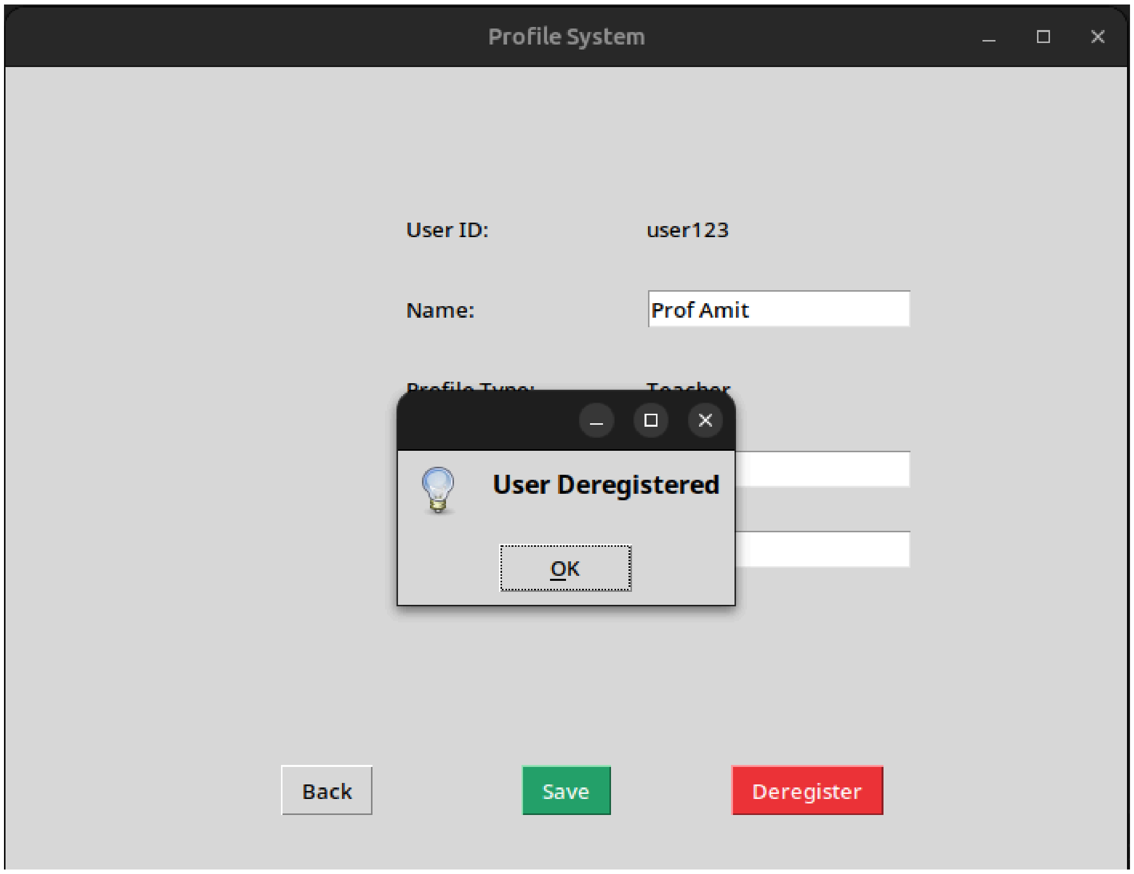

Profile System

User ID:	user123
Name:	Prof Amit
Profile Type:	Teacher
Department:	CSE
Course:	DSA

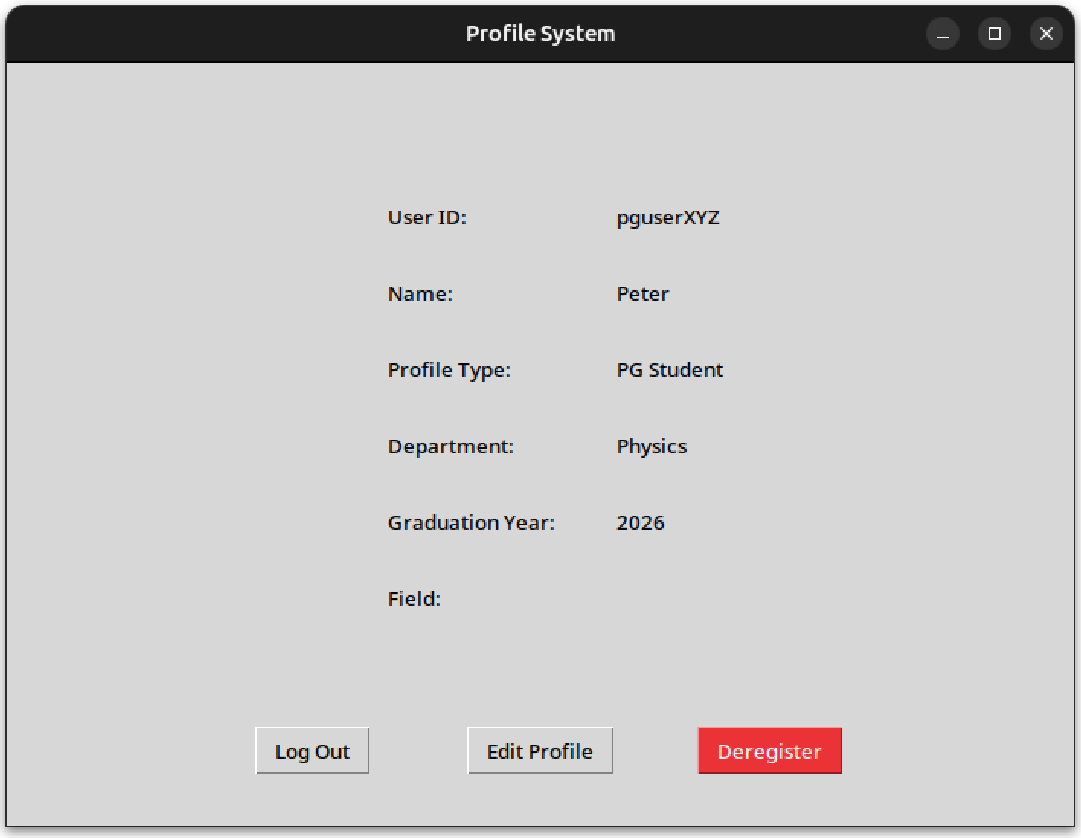
[teacher profile editing]



[deregistration confirmation]



[deregistration success message]



User ID: pguserXYZ

Name: Peter

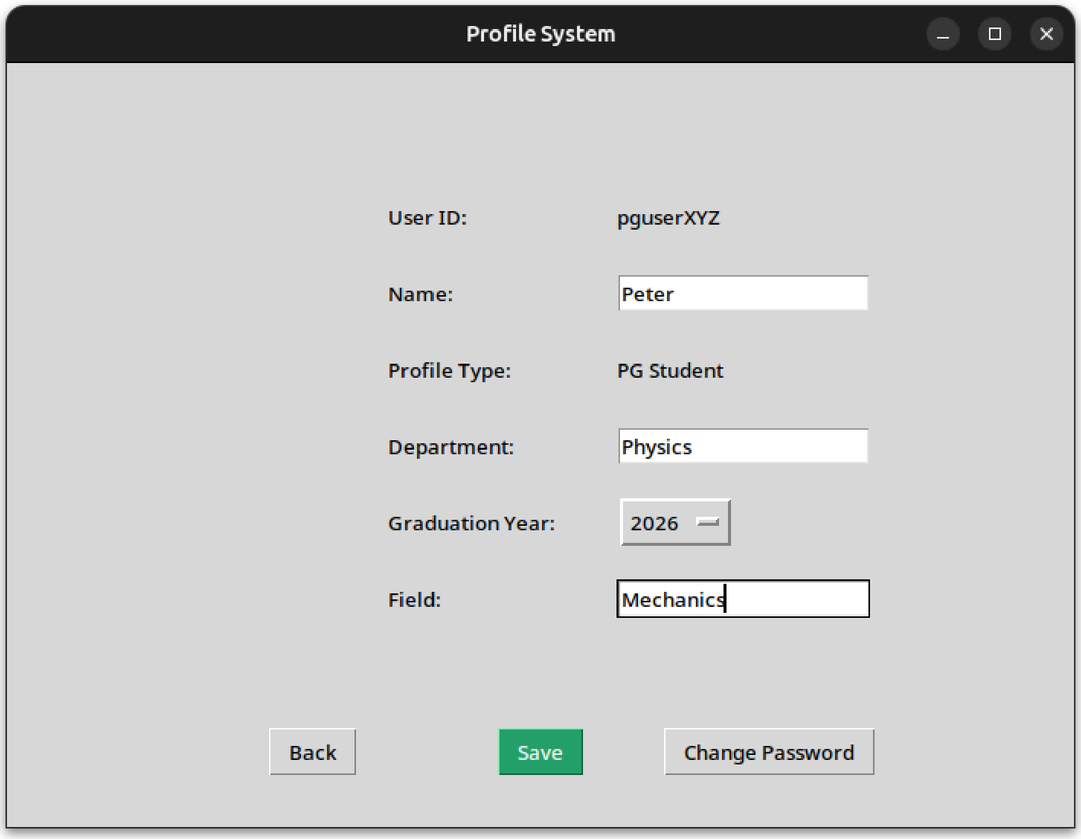
Profile Type: PG Student

Department: Physics

Graduation Year: 2026

Field:

[PG student profile]



User ID: pguserXYZ

Name: Peter

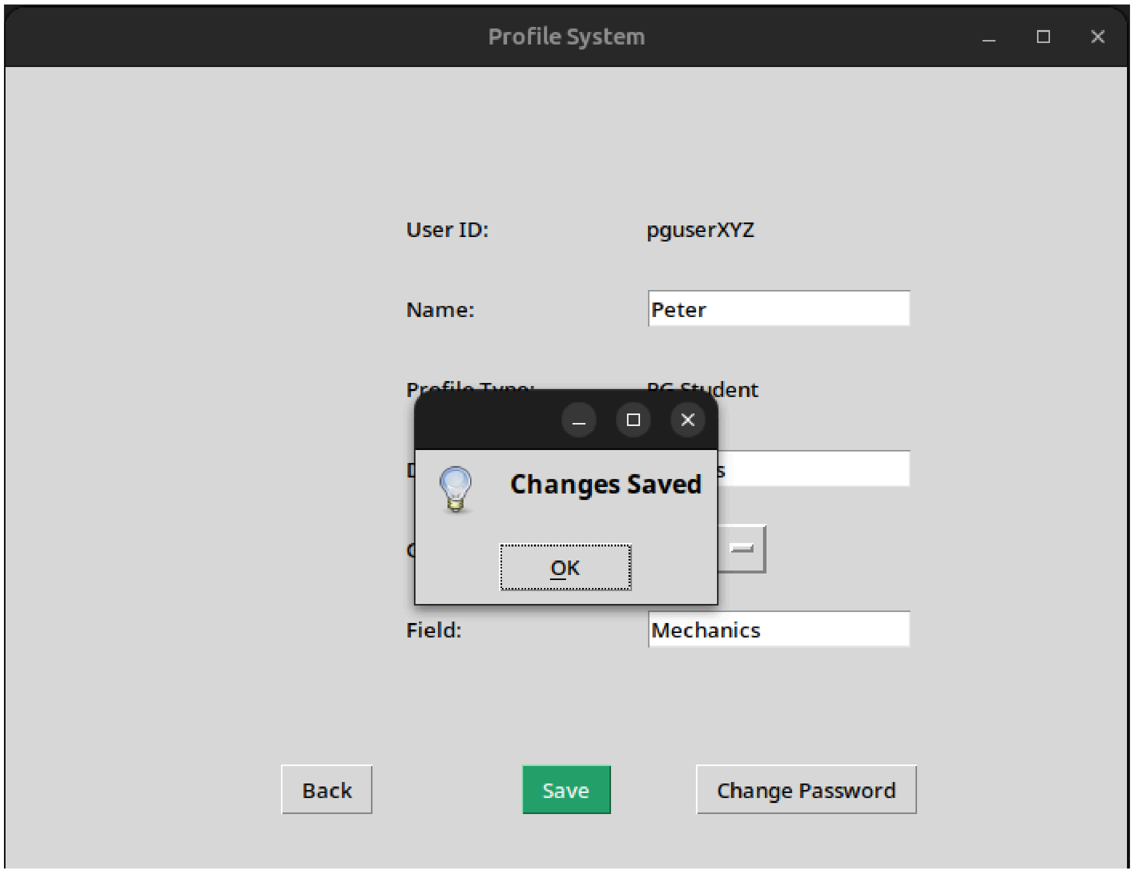
Profile Type: PG Student

Department: Physics

Graduation Year: 2026

Field: Mechanics

[PG student profile editing]


Profile System

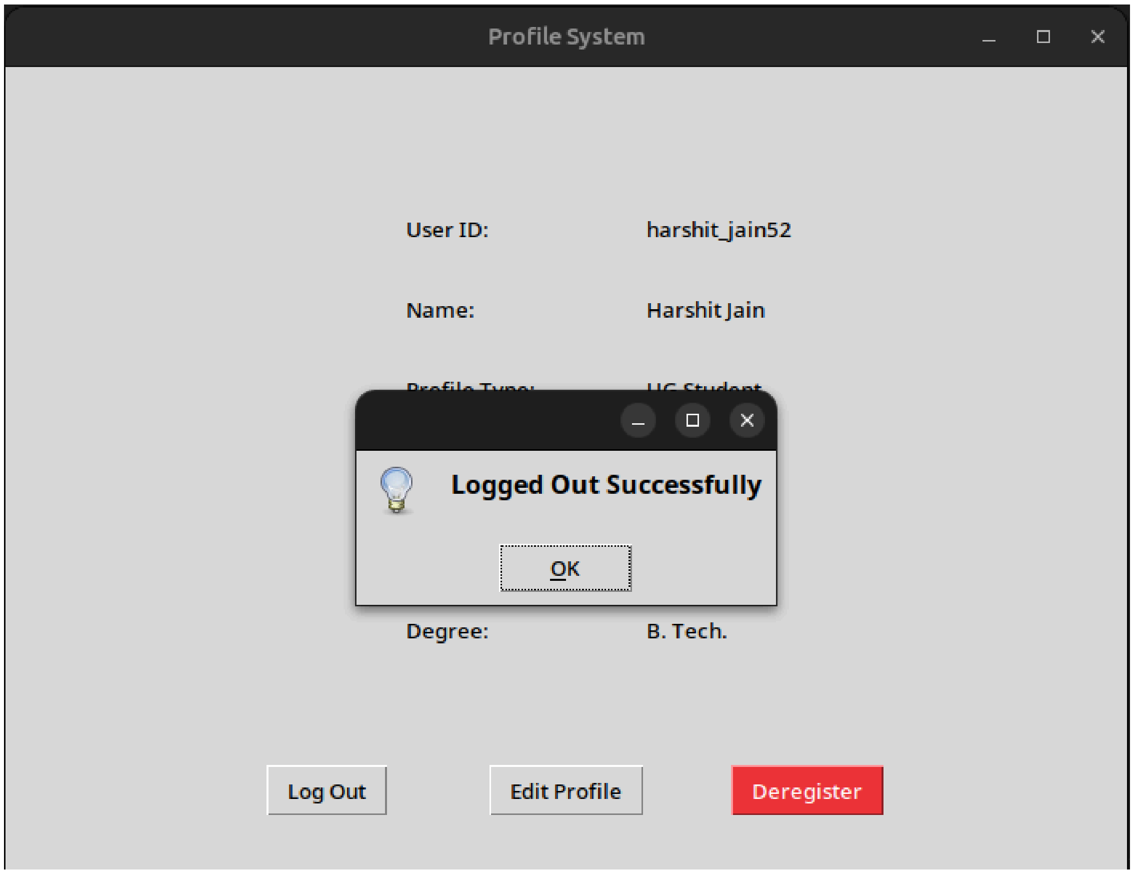
User ID: pguserXYZ

Name: Peter

Field: Mechanics

Changes Saved

[edits saved message]


Profile System

User ID: harshit_jain52

Name: Harshit Jain

Degree: B. Tech.

Logged Out Successfully

[logout success message]