**Student Management System**

This project is an implementation of a Student Management System using the Tkinter library in Python. The system allows you to perform various operations related to managing student information, such as adding students, searching for students, updating student records, deleting students, displaying student information, and exporting data to a CSV file.

**Overview:**

This code snippet implements a simple login page GUI using Tkinter. It provides a login form with username and password fields, along with a login button. Upon successful login, it displays a success message and launches another script named "sms.py".

**Installation and Setup:**

To run this code, you need to have Tkinter and PIL (Python Imaging Library) installed. You can install PIL using pip install pillow.

**File Structure:**

bg.jpg: Background image file used in the GUI.

logo.png: Logo image file used in the GUI.

user.png: Username image file used in the GUI.

password.png: Password image file used in the GUI.

sms.py: Another script that will be launched upon successful login.

**Functionality of main.py:**

The code defines a login function to handle the login action. It checks if the username and password fields are empty and displays an error message if they are. If the entered credentials are correct **(username: "mahmoud", password: "1234"),** it displays a success message and launches the **"sms.py"** script using subprocess.run().

**User Interface of login page:**

The GUI consists of a main window (window) with a fixed size of 1280x700. It displays a background image (bg.jpg). The login form is contained within a frame (loginFrame) positioned at (400, 150). The frame includes a logo image (logo.png), username and password labels with corresponding images (user.png and password.png), entry fields for username and password, and a login button.

**Code Structure and Organization:**

The code follows a modular structure. It imports necessary modules, defines the login function, creates the main window, and sets up the user interface elements.

**Usage and Examples:**

1. Run the script using python main.py.
2. Enter "mahmoud" as the username and "1234" as the password.
3. Click the "Login" button.
4. If the credentials are correct, a success message will be displayed, and the "sms.py" script will be launched.

**Dependencies**

The code uses the following libraries:

* tkinter: Provides the GUI (Graphical User Interface) components and functionality for creating windows, buttons, labels, etc.
* ttk: Provides additional themed widgets and styling options for tkinter.
* messagebox: Provides a dialog box for displaying messages or asking for user confirmation.
* filedialog: Allows the user to choose a file or directory.
* pandas: Provides data manipulation and analysis tools for handling data in a tabular format.
* psycopg2: A PostgreSQL adapter for Python, used for connecting to a PostgreSQL database.

**Functionality**

The code includes the following functionality:

* Database Connection: The connect\_database() function opens a window for connecting to a PostgreSQL database. It prompts the user to enter the hostname, username, and password for the database. After successful connection, the student management system functionalities are enabled.

A screenshot of a computer

Description automatically generated

* Add Student: The add\_data() function is called when the "Add Student" button is clicked. It opens a new window (toplevel\_data()) for entering student details such as name, faculty ID, academic year, degree type, department, and courses. The data is then inserted into the database table students. The user can also select multiple courses for a student. The GPA (Grade Point Average) and letter grade are calculated based on the total marks obtained in the selected courses.
* Search Student: The search\_data() function is called when the "Search Student" button is clicked. It prompts the user to enter a faculty ID and retrieves the corresponding student details from the students table in the database. The student details are displayed in the table.
* Delete Student: The delete\_student() function is called when the "Delete Student" button is clicked. It deletes the selected student from the student\_courses table based on the faculty ID.
* Update Student: The update\_data() function is called when the "Update Student" button is clicked. However, this function is currently commented out in the provided code and needs to be implemented. It would allow updating the student details such as name, faculty ID, academic year, degree type, department, and courses.
* Show Student: The show\_student() function is called when the "Show Student" button is clicked. It retrieves all student details from the students table and displays them in the table.
* Export Data: The export\_data() function is called when the "Export Data" button is clicked. It allows the user to choose a location to save the student data in CSV format.
* Exit: The iexit() function is called when the "Exit" button is clicked. It prompts the user for confirmation before closing the application.

**GUI Layout**

A screenshot of a computer

Description automatically generated

* The GUI layout is divided into two frames: the left frame and the right frame.
* The left frame contains buttons for various operations, including adding a student, searching for a student, deleting a student, updating a student's record, displaying all students, exporting data, and exiting the application.
* The right frame contains a Treeview widget that displays the student information in a tabular format. The Treeview is configured with columns for the student's name, faculty ID, academic year, degree type, department, course name, midterm mark, final mark, total mark, GPA, and letter grade.
* The scrollBarX and scrollBarY objects are used to provide horizontal and vertical scrolling functionality to the Treeview widget.

**Additional Features**

The code includes a slider at the top of the window that displays the message "Shoubra Faculty Of Engineering Registration System" in an animated manner.

The code also includes a clock that displays the current date and time at the top left corner of the window.

Please note that some parts of the code are commented out or not fully implemented, such as the update\_data() function. You would need to complete those parts based on your requirements.

**Acknowledgments:**

The code snippet uses Tkinter and PIL libraries, which are commonly used for GUI development in Python.