



Systems Analysis and Design

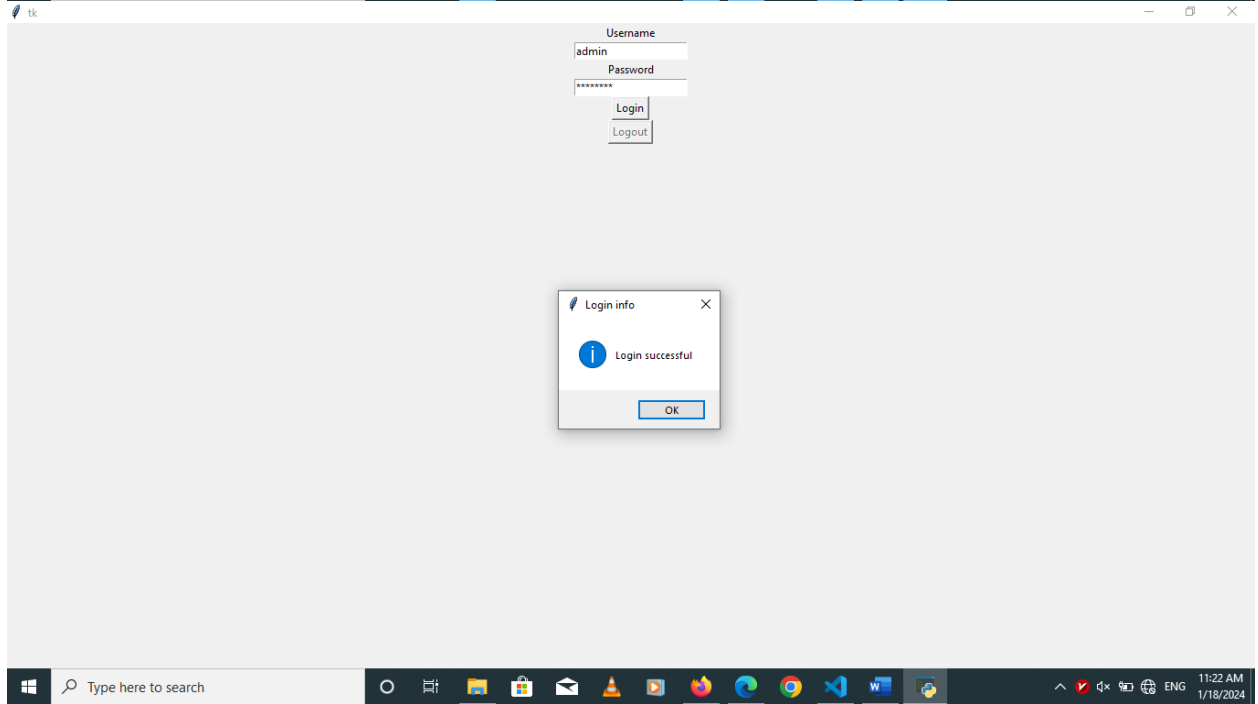
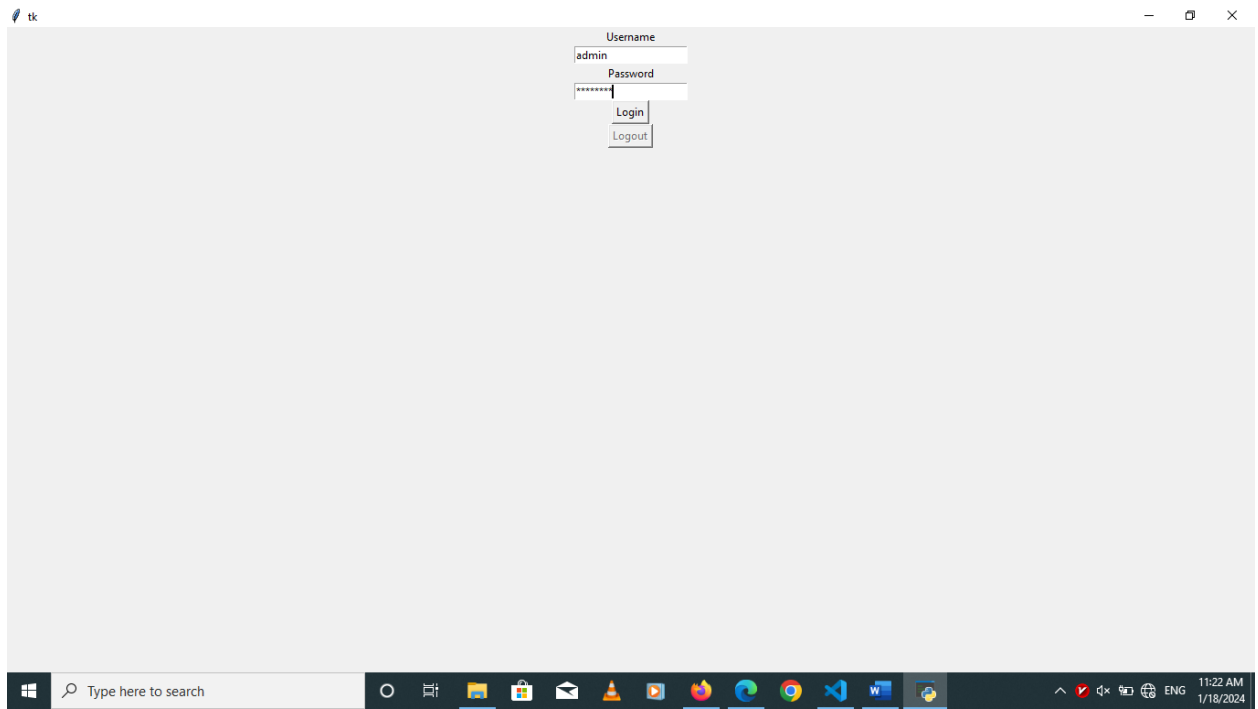
استاد مربوطه : دکتر شجاع

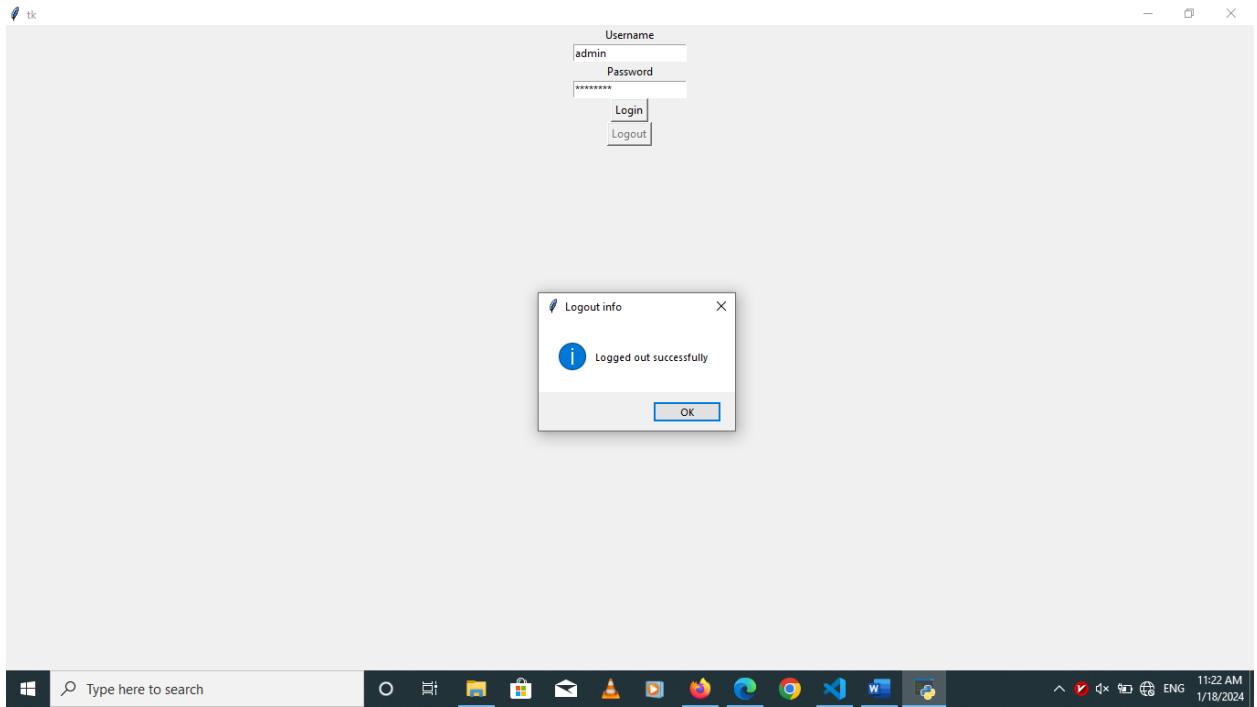
نام و نام خانوادگی : ترمه نجاراذری

شماره دانشجویی : ۱۴۰۰۴۴۲۱۴۵

پروژه نهایی

- `import tkinter as tk`: این خط کتابخانه `tkinter` را وارد می کند که برای ساخت رابط کاربری گرافیکی در پایتون استفاده می شود.
- `from tkinter import messagebox`: این خط `messagebox` را از کتابخانه `tkinter` وارد می کند که برای نمایش پیام های گرافیکی به کاربر استفاده می شود. ۳-۹. کلاس `User` تعریف می شود که شامل متدهای `login`, `__init__` و `logout` است. این کلاس برای مدیریت حساب کاربری در برنامه استفاده می شود. ۱۰-۳۱. کلاس `Application` تعریف می شود که از کلاس `tk.Frame` ارث بری می کند. این کلاس برای ساخت رابط کاربری گرافیکی استفاده می شود و شامل متدهای `__init__`, `login`, `create_widgets` و `logout` است.
- یک نمونه از کلاس `User` با نام کاربری `"admin"` و رمز عبور `"password"` ایجاد می شود.
- یک نمونه از کلاس `tk.Tk` ایجاد می شود که یک پنجره جدید برای برنامه ایجاد می کند.
- یک نمونه از کلاس `Application` با پنجره ایجاد شده در خط قبل ایجاد می شود.
- `app.mainloop()` فراخوانی می شود که حلقه رویدادهای گرافیکی را شروع می کند و باعث می شود پنجره برنامه نمایش داده شود و منتظر ورودی کاربر بماند.





- خطوط ۱-۲: کتابخانه های مورد نیاز برای ساخت رابط کاربری گرافیکی (tkinter) و پخش موسیقی (pygame) را وارد می کنیم.
- خطوط ۴-۳۴: کلاس MusicPlayer را تعریف می کنیم. این کلاس شامل متدهای __init__, open_file, playsong, stopsong, و pausesong است.
- خط ۳۶: یک نمونه از کلاس Tk.Tk ایجاد می کنیم که یک پنجره جدید برای برنامه ایجاد می کند.
- خط ۳۷: یک نمونه از کلاس MusicPlayer با پنجره ایجاد شده در خط قبل ایجاد می کنیم.
- خط ۳۸ root.mainloop() فراخوانی می شود که حلقه رویدادهای گرافیکی را شروع می کند و باعث می شود پنجره برنامه نمایش داده شود و منتظر ورودی کاربر بماند.

The screenshot shows a Python IDE with the following components:

- EXPLORER:** Shows a project named 'PROJ SW' containing files 'musicStream.py' and 'userAccount.py'.
- EDITOR:** Displays the code for 'musicStream.py'. The code defines a 'MusicPlayer' class with an '__init__' method. The method creates a Tkinter window titled 'Music Player', initializes pygame mixer, and sets up a 'Song Track' label and a status label. The window is resizable and has a blue background.
- TERMINAL:** Shows the output of running the program. It displays the pygame version (2.5.2), the Python version (3.11.4), and a message from the pygame community. The program exits with code 0 in 3.087 seconds.

```

1 import tkinter as tk
2 import pygame
3 from tkinter import filedialog
4
5 class MusicPlayer:
6     def __init__(self, window):
7         window.geometry('600x400'); window.resizable(1,1)
8         pygame.mixer.init()
9         self.track = ''
10        self.status = ''
11
12        # Creating Song Track Label
13        trackframe = tk.LabelFrame(window, text="Song Track", font=("times new roman",15,"bold"), bg="Navyblue",
14                                    trackframe.place(x=0, y=0, width=600, height=100)
15        # Inserting Song Track Label
16        songtrack = tk.Label(trackframe, textvariable=self.track, width=20, font=("times new roman", 24, "bold"))
17        # Inserting Status Label
18        trackstatus = tk.Label(trackframe, textvariable=self.status, font=("times new roman", 24, "bold"), bg="c

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS SQL CONSOLE

[Done] exited with code=0 in 4.117 seconds

[Running] python -u "c:\Users\Asus\Desktop\proj sw\musicStream.py"

pygame 2.5.2 (SDL 2.28.3, Python 3.11.4)

Hello from the pygame community. <https://www.pygame.org/contribute.html>

[Done] exited with code=0 in 3.087 seconds

[Running] python -u "c:\Users\Asus\Desktop\proj sw\musicStream.py"

pygame 2.5.2 (SDL 2.28.3, Python 3.11.4)

Hello from the pygame community. <https://www.pygame.org/contribute.html>

Ln 7, Col 28 Spaces: 4 UTF-8 CRLF Python 3.11.4 64-bit

11:30 AM 1/18/2024