Ex. No.:1 **Date:**

Register No.:241701013 **Name: S.K.DHANESH**

Create a Study Focus Tracker Application using a Timer and Data Storage

AIM: To develop a simple **Study Focus Tracker application** that utilizes a **GUI (Graphical User Interface)** with:

- A **timer** to set and run focused study sessions.
- The ability to **record and save session data** (subject and duration) to a JSON file.
- A **stats display** to show the total study time for the current day.

Procedure:

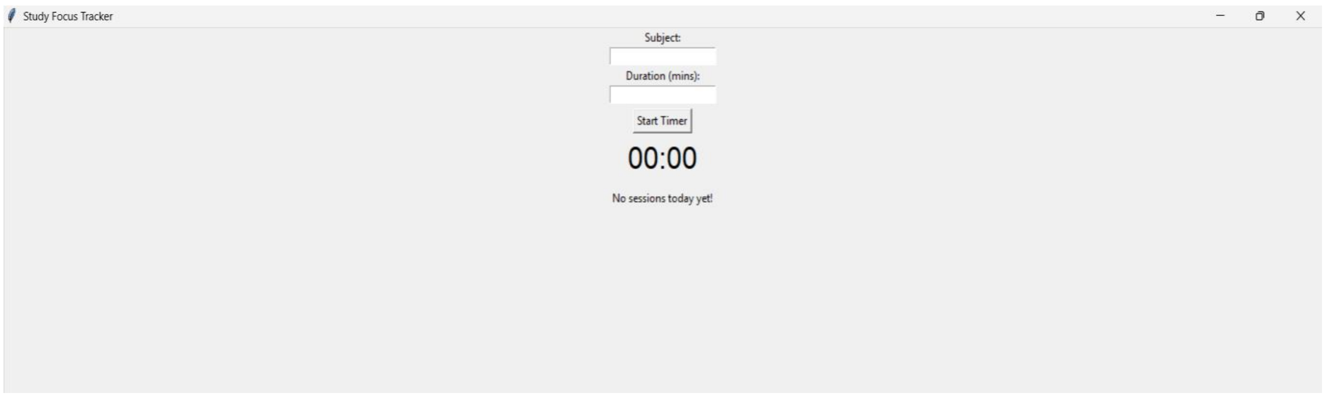
1. Use the **Python GUI framework tkinter** for the user interface.
2. Implement a function to **read and write session data** to a JSON file (study_data.json) using the json module.
3. Create a countdown function to handle the **timer logic** and update the display every second using root.after().
4. Design the GUI with input fields for **Subject** and **Duration (in minutes)**.
5. Include a **"Start Timer" button** to initiate the session.
6. Implement a show_stats function to calculate and display the **total study time for the current day**.

Program:

```
import tkinter as tk

from tkinter import messagebox

import time, json, math

from datetime import date

try:

    with open("study_data.json", "r") as f:

        study_data = json.load(f)

except:

    study_data = {}
```

```
def save_session(subject, duration):
    today = str(date.today())
    if today not in study_data:
        study_data[today] = []
    study_data[today].append({"subject": subject, "duration": duration})
    with open("study_data.json", "w") as f:
        json.dump(study_data, f, indent=4)


def start_timer():
    subject = subject_entry.get()
    try:
        duration = int(duration_entry.get())
        countdown(duration * 60, subject)
    except ValueError:
        messagebox.showerror("Input Error", "Duration must be an integer!")


def countdown(seconds, subject):
    if seconds > 0:
        mins = seconds // 60
        secs = seconds % 60
        timer_label.config(text=f"{mins:02d}:{secs:02d}")
        root.after(1000, countdown, seconds - 1, subject)
    else:
        messagebox.showinfo("Done!", "Session Complete! Take a break 😊")
        save_session(subject, int(duration_entry.get()))
        show_stats()


def show_stats():
    today = str(date.today())
    if today in study_data:
        total = sum([x['duration'] for x in study_data[today]])
        hrs = math.floor(total / 60)
        mins = total % 60
        stats_label.config(text=f"Total Study Today: {hrs}h {mins}m")
    else:
        stats_label.config(text="No sessions today yet!")
root = tk.Tk()
root.title("Study Focus Tracker")

tk.Label(root, text="Subject:").pack()
subject_entry = tk.Entry(root)
subject_entry.pack()

tk.Label(root, text="Duration (mins):").pack()
duration_entry = tk.Entry(root)
duration_entry.pack()

tk.Button(root, text="Start Timer", command=start_timer).pack(pady=5)
timer_label = tk.Label(root, text="00:00", font=("Helvetica", 24))
timer_label.pack()

stats_label = tk.Label(root, text="Total Study Today: 0h 0m")
stats_label.pack(pady=10)

show_stats() # Load initial stats
root.mainloop()
```

Output:



Result:

A **Study Focus Tracker application** was successfully created using Python's tkinter library. It includes a functional timer for focused work sessions, saves session details to a JSON file, and displays the total study time accumulated for the current day.