Project-4

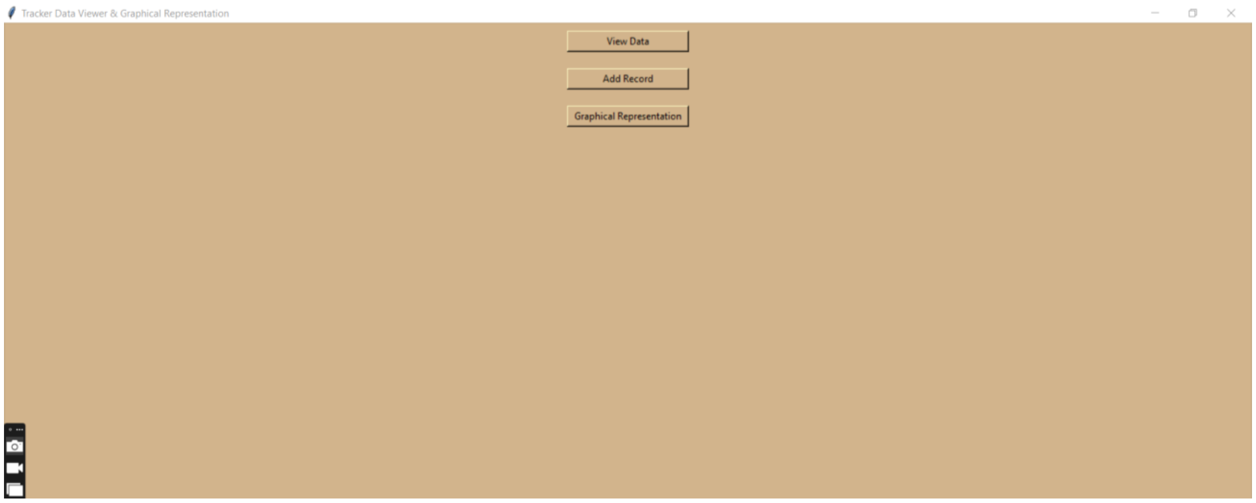
Name : V.Mahendra

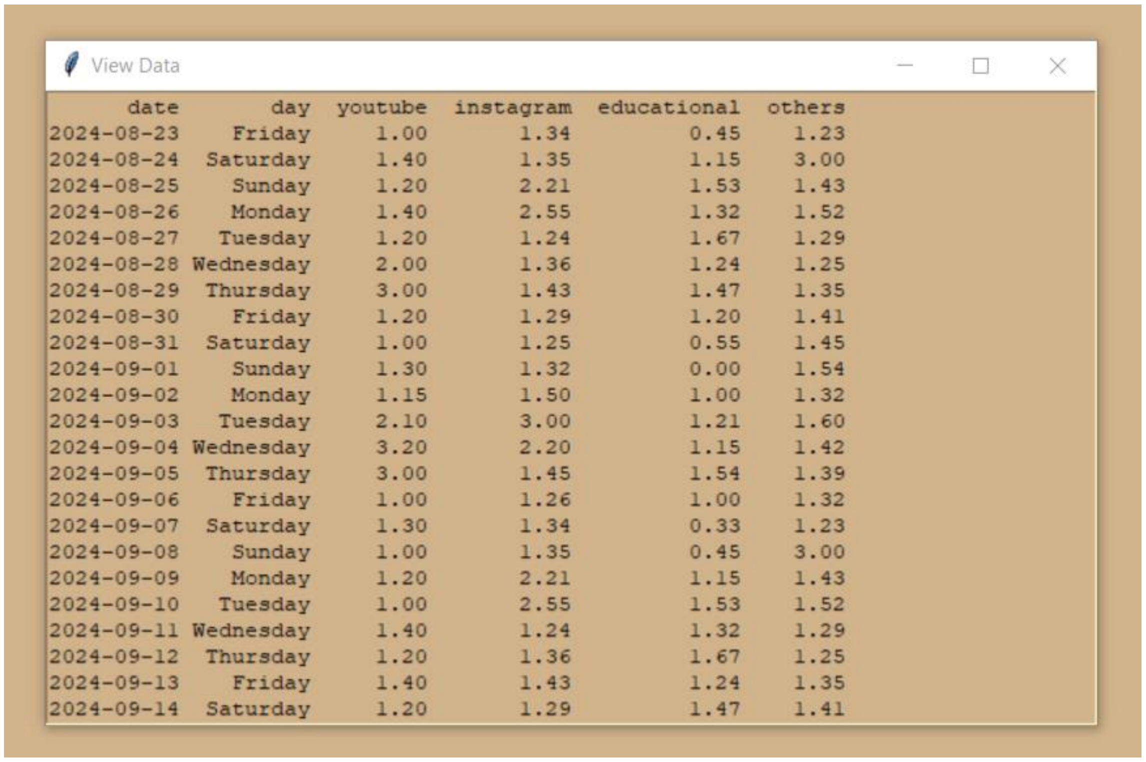
Reg no: 12305538

Roll no: 31

Submitted to : Dr. Kumar Vishal

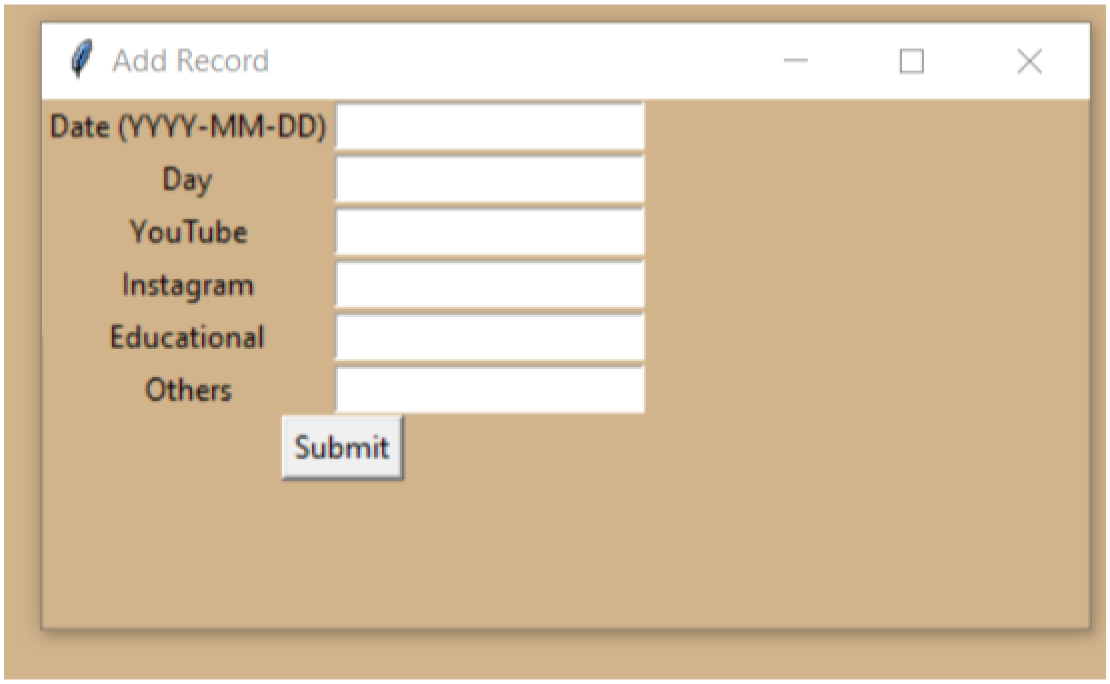
Q. Make GUI from dataset and make visualisation.



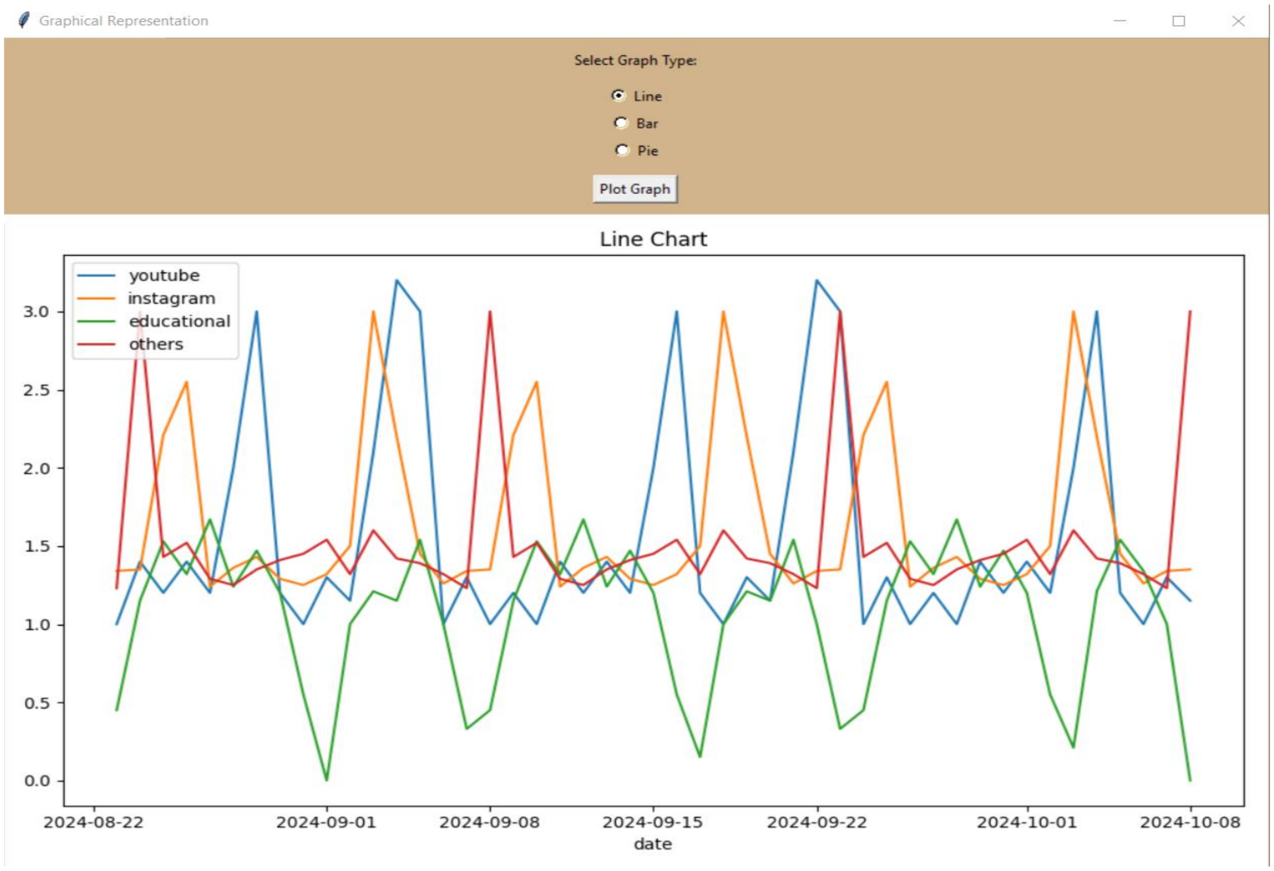


```
      date        day  youtube  instagram  educational  others
2024-08-23     Friday     1.00       1.34         0.45    1.23
2024-08-24   Saturday     1.40       1.35         1.15    3.00
2024-08-25     Sunday     1.20       2.21         1.53    1.43
2024-08-26     Monday     1.40       2.55         1.32    1.52
2024-08-27    Tuesday     1.20       1.24         1.67    1.29
2024-08-28  Wednesday     2.00       1.36         1.24    1.25
2024-08-29   Thursday     3.00       1.43         1.47    1.35
2024-08-30     Friday     1.20       1.29         1.20    1.41
2024-08-31   Saturday     1.00       1.25         0.55    1.45
2024-09-01     Sunday     1.30       1.32         0.00    1.54
2024-09-02     Monday     1.15       1.50         1.00    1.32
2024-09-03    Tuesday     2.10       3.00         1.21    1.60
2024-09-04  Wednesday     3.20       2.20         1.15    1.42
2024-09-05   Thursday     3.00       1.45         1.54    1.39
2024-09-06     Friday     1.00       1.26         1.00    1.32
2024-09-07   Saturday     1.30       1.34         0.33    1.23
2024-09-08     Sunday     1.00       1.35         0.45    3.00
2024-09-09     Monday     1.20       2.21         1.15    1.43
2024-09-10    Tuesday     1.00       2.55         1.53    1.52
2024-09-11  Wednesday     1.40       1.24         1.32    1.29
2024-09-12   Thursday     1.20       1.36         1.67    1.25
2024-09-13     Friday     1.40       1.43         1.24    1.35
2024-09-14   Saturday     1.20       1.29         1.47    1.41
```

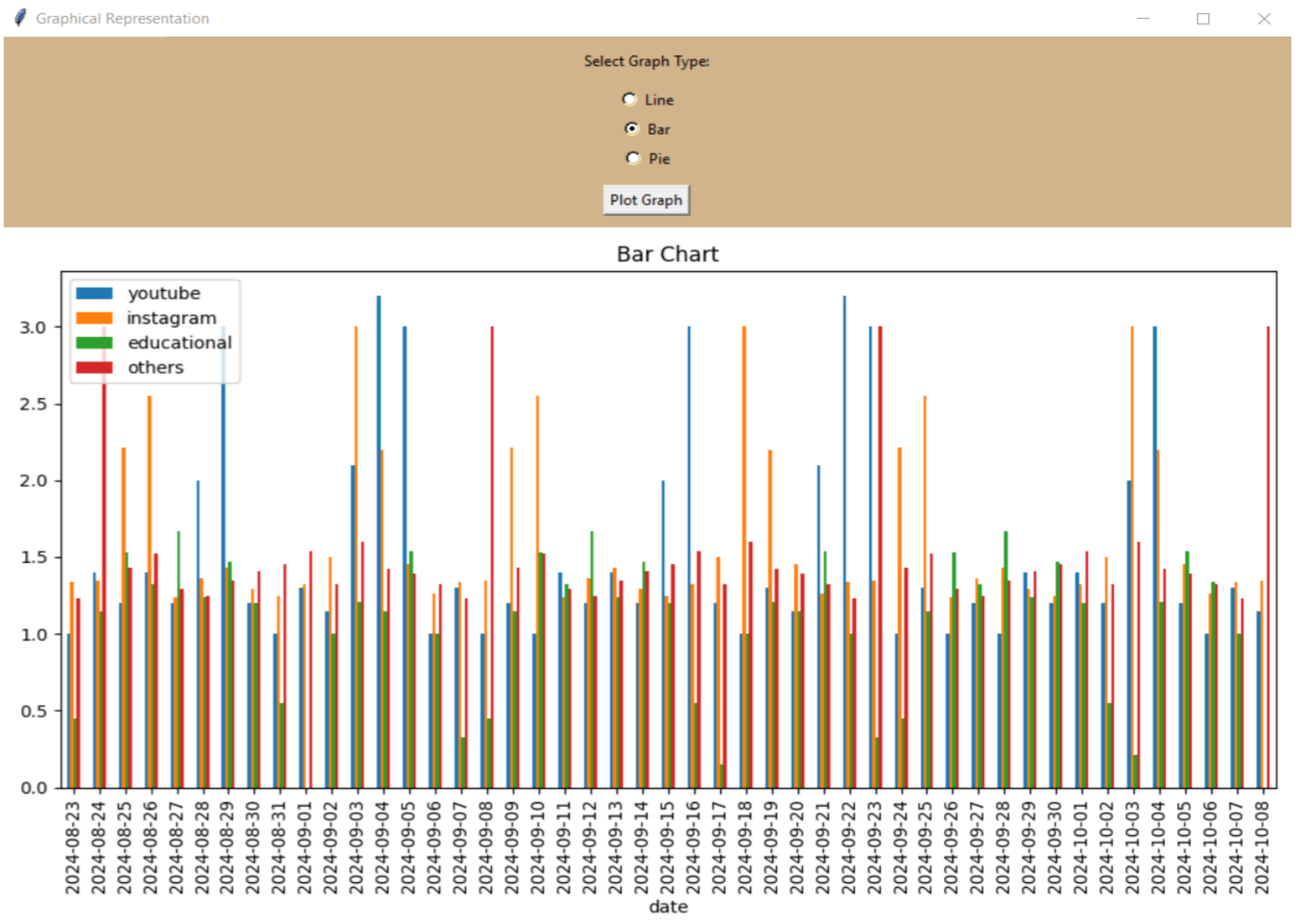
View data:



Date (YYYY-MM-DD)	
Day	
YouTube	
Instagram	
Educational	
Others	



Select Graph Type: [✓ Line] Bar Pie



```
import mysql.connector

import tkinter as tk

from tkinter import messagebox

import pandas as pd

import matplotlib.pyplot as plt

from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg


# MySQL database connection

def connect_db():

    conn = mysql.connector.connect(
        host="127.0.0.1",
        port=3306,
        user="root",
        passwd="",  # Replace with your MySQL root password
        database="data_tracker"  # Use the data_tracker database
    )
    return conn

# Create the database if it doesn't exist
def create_database():
    conn = mysql.connector.connect(
        host="127.0.0.1",
        port=3306,
        user="root",
        passwd=""  # Replace with your MySQL root password
    )
    cursor = conn.cursor()
    cursor.execute("CREATE DATABASE IF NOT EXISTS data_tracker")
    conn.close()

# Create table in the database if it doesn't exist
def create_table():
    conn = connect_db()
    cursor = conn.cursor()
    cursor.execute('''
        CREATE TABLE IF NOT EXISTS screen_usage (
            date DATE,
            day VARCHAR(10),
            youtube FLOAT,
            instagram FLOAT,
            educational FLOAT,
            others FLOAT
        )
    ''')
    conn.commit()
    conn.close()

# Load data from the MySQL database into a DataFrame
def load_data():
    conn = connect_db()
    query = "SELECT * FROM screen_usage"  # Updated to use screen_usage
    df = pd.read_sql_query(query, conn)
    conn.close()
    return df

# Save DataFrame data to the MySQL database
def save_data():
    conn = connect_db()
    cursor = conn.cursor()
    for _, row in df.iterrows():
        cursor.execute('''
            INSERT INTO screen_usage (date, day, youtube, instagram, educational, others)  # Updated
to use screen_usage
            VALUES (%s, %s, %s, %s, %s, %s)
        ''', (row['date'], row['day'], row['youtube'], row['instagram'], row['educational'], row['others']))
    conn.commit()
    conn.close()

# Initialize the database and table
create_database()
```

```
create_table()

# Load data into a DataFrame
try:
    df = load_data()
except:
    df = pd.DataFrame(columns=["date", "day", "youtube", "instagram", "educational", "others"])

# Tkinter GUI setup
root = tk.Tk()
root.title("Tracker Data Viewer & Graphical Representation")
root.configure(bg='#D2B48C')

# View Data Function
def view_data():
    top = tk.Toplevel(root)
    top.title("View Data")
    top.configure(bg='#D2B48C')
    text = tk.Text(top, wrap="word", bg='#D2B48C', fg='black')
    text.pack(fill=tk.BOTH, expand=True)
    text.insert(tk.END, df.to_string(index=False))
    text.config(state=tk.DISABLED)

# Add Record Function
def add_record():
    def submit_record():
        try:
            date = date_entry.get()
            day = day_entry.get()
            youtube = float(youtube_entry.get())
            instagram = float(instagram_entry.get())
            educational = float(educational_entry.get())
            others = float(others_entry.get())

            new_data = {
                "date": date,
                "day": day,
                "youtube": youtube,
                "instagram": instagram,
                "educational": educational,
                "others": others
            }

            global df
            df = pd.concat([df, pd.DataFrame([new_data])], ignore_index=True)

            # Save the new data to the MySQL database
            conn = connect_db()
            cursor = conn.cursor()
            cursor.execute('''
                INSERT INTO screen_usage (date, day, youtube, instagram, educational, others)  #
Updated to use screen_usage
                VALUES (%s, %s, %s, %s, %s, %s)
            ''', (date, day, youtube, instagram, educational, others))
            conn.commit()
            conn.close()

            messagebox.showinfo("Success", "Record added successfully!")
            add_window.destroy()
        except ValueError:
```

```
        messagebox.showerror("Input Error", "Please enter valid numerical values.")

    add_window = tk.Toplevel(root)
    add_window.title("Add Record")
    add_window.configure(bg='#D2B48C')

    # Input fields
    tk.Label(add_window, text="Date (YYYY-MM-DD)", bg='#D2B48C').grid(row=0, column=0)
    tk.Label(add_window, text="Day", bg='#D2B48C').grid(row=1, column=0)
    tk.Label(add_window, text="YouTube", bg='#D2B48C').grid(row=2, column=0)
    tk.Label(add_window, text="Instagram", bg='#D2B48C').grid(row=3, column=0)
    tk.Label(add_window, text="Educational", bg='#D2B48C').grid(row=4, column=0)
    tk.Label(add_window, text="Others", bg='#D2B48C').grid(row=5, column=0)

    date_entry = tk.Entry(add_window)
    day_entry = tk.Entry(add_window)
    youtube_entry = tk.Entry(add_window)
    instagram_entry = tk.Entry(add_window)
    educational_entry = tk.Entry(add_window)
    others_entry = tk.Entry(add_window)

    date_entry.grid(row=0, column=1)
    day_entry.grid(row=1, column=1)
    youtube_entry.grid(row=2, column=1)
    instagram_entry.grid(row=3, column=1)
    educational_entry.grid(row=4, column=1)
    others_entry.grid(row=5, column=1)

    submit_button = tk.Button(add_window, text="Submit", command=submit_record)
    submit_button.grid(row=6, column=0, columnspan=2)
```

```
# Graphical Representation Function
def graphical_representation():
    def plot_graph():
        selected_graph = graph_type_var.get()
        df_sorted = df.dropna(subset=["date"]).sort_values("date")

        fig, ax = plt.subplots(figsize=(10, 6))

        if selected_graph == "Line":
            df_sorted.plot(x="date", y=["youtube", "instagram", "educational", "others"], kind="line",
ax=ax)
        elif selected_graph == "Bar":
            df_sorted.plot(x="date", y=["youtube", "instagram", "educational", "others"], kind="bar",
ax=ax)
        elif selected_graph == "Pie":
            aggregate_data = df_sorted[['youtube', 'instagram', 'educational', 'others']].sum()
            aggregate_data.plot(kind="pie", autopct='%1.1f%%', ax=ax)
            ax.set_ylabel('')
        else:
            messagebox.showerror("Error", "Please select a valid graph type.")
            return

        ax.set_title(f"{selected_graph} Chart")
        plt.tight_layout()

        canvas = FigureCanvasTkAgg(fig, master=graph_window)
        canvas.get_tk_widget().pack(fill=tk.BOTH, expand=True)
        canvas.draw()

    graph_window = tk.Toplevel(root)
    graph_window.title("Graphical Representation")

    graph_window.configure(bg='#D2B48C')


    tk.Label(graph_window, text="Select Graph Type:", bg='#D2B48C').pack(pady=10)

    graph_type_var = tk.StringVar(value="Line")

    graph_types = ["Line", "Bar", "Pie"]

    for graph in graph_types:

        tk.Radiobutton(graph_window, text=graph, variable=graph_type_var, value=graph,
bg='#D2B48C').pack()


    plot_button = tk.Button(graph_window, text="Plot Graph", command=plot_graph)

    plot_button.pack(pady=10)


# Main Buttons

view_button = tk.Button(root, text="View Data", width=20, command=view_data, bg='#D2B48C')

view_button.pack(pady=10)


add_button = tk.Button(root, text="Add Record", width=20, command=add_record,
bg='#D2B48C')

add_button.pack(pady=10)


graph_button = tk.Button(root, text="Graphical Representation", width=20,
command=graphical_representation, bg='#D2B48C')

graph_button.pack(pady=10)


root.mainloop()
```