Name: Muhammad Ahmad Raza Qadri

Intern ID: TN/IN02/PY/013

Task no:04

Question:

1. **Fetch GitHub info of user and print repos of your account**

Code:

Task01.py > ...

```python
import requests

def fetch_github_repos(username):
    url = f"https://api.github.com/users/{username}/repos"
    response = requests.get(url)

    if response.status_code == 200:
        repos = response.json()
        print(f"\nPublic Repositories of {username}:")
        for repo in repos:
            print(f"- {repo['name']}")
    else:
        print(f"Failed to fetch repos for {username}. Status code: {response.status_code}")

fetch_github_repos("MuhammadAhmadRazaOfficial")
```

Output

```
PS D:\Python projects\tasks\week 04 tasks>

PS D:\Python projects\tasks\week 04 tasks>
PS D:\Python projects\tasks\week 04 tasks> & C:/Users/Fame/AppData/Local/Microsoft/WindowsApps/python3.13.exe "
d:/Python projects/tasks/week 04 tasks/Task01.py"

Public Repositories of MuhammadAhmadRazaOfficial:
- exapmle
- internship-tasks-week-1
- internship-tasks-week-2
- internship-tasks-week-3
PS D:\Python projects\tasks\week 04 tasks>
```

2. Use joke api to fetch and print the jokes

Code:

Task02.py > ...

```python
import requests

def fetch_joke():
    url = "https://official-joke-api.appspot.com/random_joke"
    response = requests.get(url)

    if response.status_code == 200:
        joke = response.json()
        print("Here's a joke for you:\n")
        print(f"{joke['setup']}")
        print(f"{joke['punchline']}")
    else:
        print(f"Failed to fetch joke. Status code: {response.status_code}")

# Run the function
fetch_joke()

```

Output:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS Python

```
PS D:\Python projects\tasks\week 04 tasks> & C:/Users/Fame/AppData/Local/Microsoft/WindowsApps/python3.13.exe "d
:/Python projects/tasks/week 04 tasks/Task02.py"
Here's a joke for you:

Why did the tomato blush?
Because it saw the salad dressing.
PS D:\Python projects\tasks\week 04 tasks>
```

Question:

1. **Load students.csv that contain students data you have to clean it using pandas and then print the average marks for each subject of whole class.**

Code:

Task03.py > ...

```python
import pandas as pd
data = {
    "Name": ["Ali", "Sara", "Ahmed", "Ayesha"],
    "Math": [85, 90, 75, 95],
    "English": [78, None, 80, 85],
    "Science": [92, 88, None, 89]
}
df = pd.DataFrame(data)
df.to_csv("students.csv", index=False)
print("students.csv created successfully!\n")
df = pd.read_csv("students.csv")
print("Original Data:")
print(df)
df_cleaned = df.fillna(df.mean(numeric_only=True))
print("\nCleaned Data:")
print(df_cleaned)
averages = df_cleaned[['Math', 'English', 'Science']].mean()
print("\n Average Marks of the Whole Class:")
print(averages)
```

Output:

```
PS D:\Python projects\tasks\week 04 tasks> & C:/Users/Fame/AppData/Local/Microsoft/WindowsApps/python3.13.exe "d
:/Python projects/tasks/week 04 tasks/Task03.py"
students.csv created successfully!

Original Data:
     Name  Math  English  Science
0     Ali    85     78.0     92.0
1    Sara    90      NaN     88.0
2   Ahmed    75     80.0      NaN
3  Ayesha    95     85.0     89.0

Cleaned Data:
     Name  Math  English    Science
0     Ali    85     78.0  92.000000
1    Sara    90     81.0  88.000000
2   Ahmed    75     80.0  89.666667
3  Ayesha    95     85.0  89.000000

 Average Marks of the Whole Class:
Math       86.250000
English    81.000000
Science    89.666667
dtype: float64
PS D:\Python projects\tasks\week 04 tasks>
```

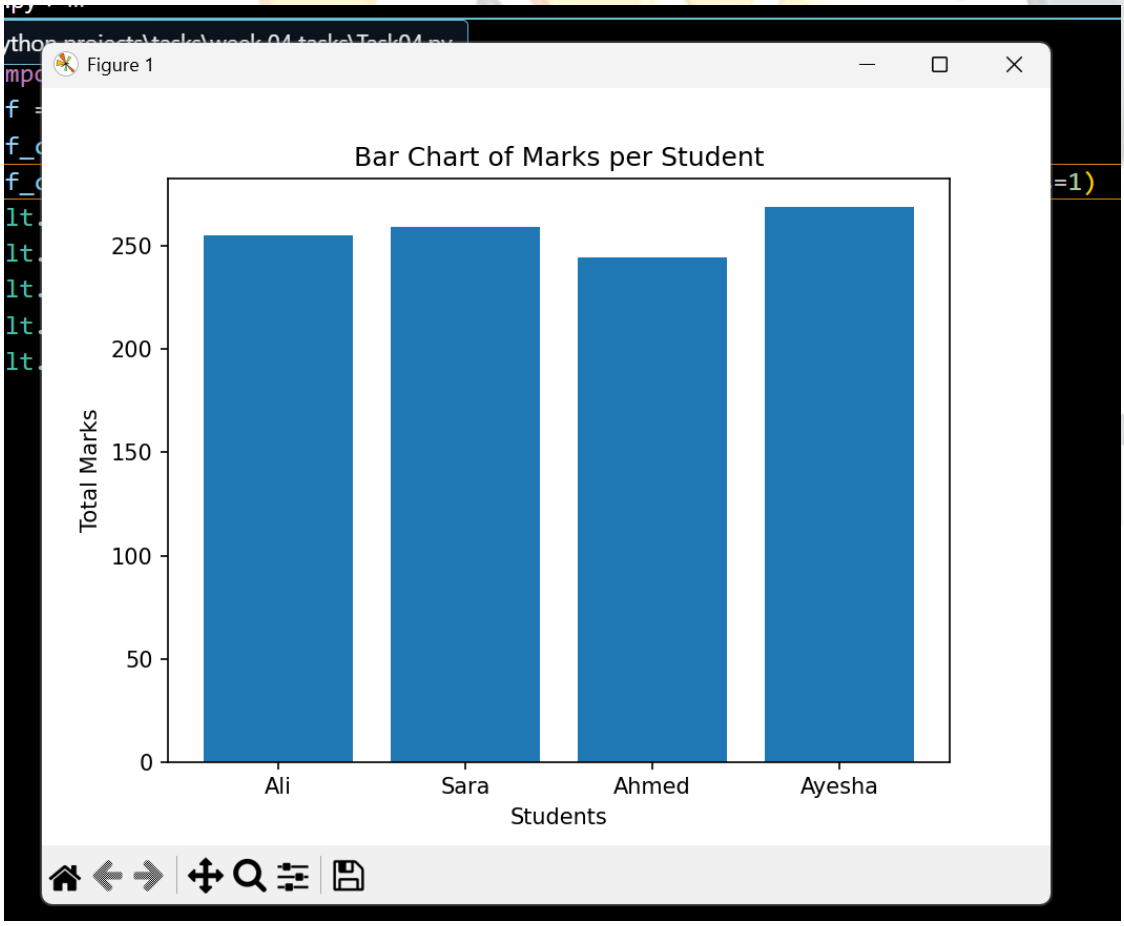
Question:

1. **Bar chart of marks per student from students.csv**

Code:

```python
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv("students.csv")
df_cleaned = df.fillna(df.mean(numeric_only=True))
df_cleaned["Total"] = df_cleaned[["Math", "English", "Science"]].sum(axis=1)
plt.bar(df_cleaned["Name"], df_cleaned["Total"])
plt.xlabel("Students")
plt.ylabel("Total Marks")
plt.title("Bar Chart of Marks per Student")
plt.show()
```

Output:

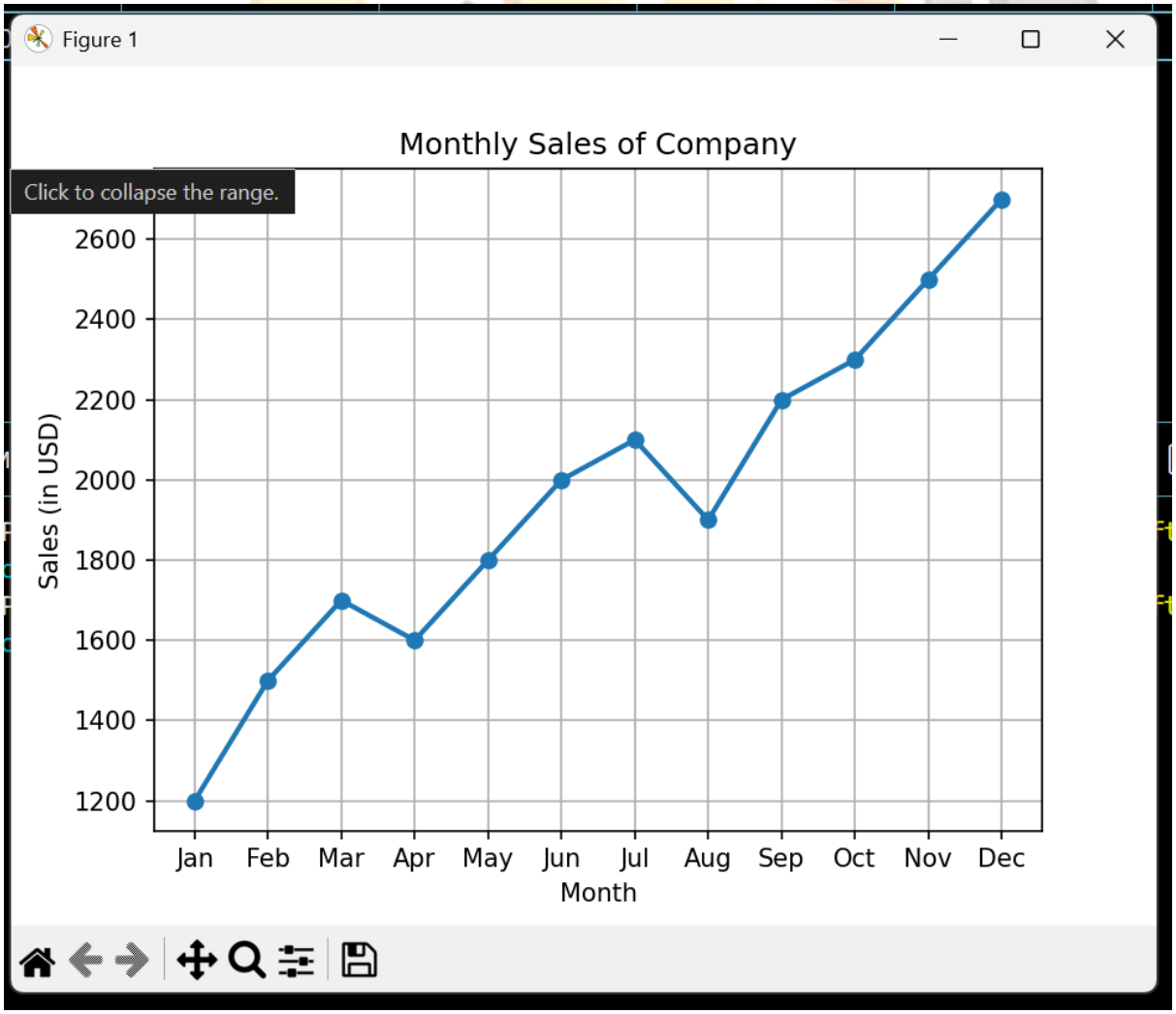


2. Line chart monthly sales of company on dummy data.

Code:

```python
import pandas as pd
import matplotlib.pyplot as plt
data = {
    "Month": ["Jan", "Feb", "Mar", "Apr", "May", "Jun",
              "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"],
    "Sales": [1200, 1500, 1700, 1600, 1800, 2000,
              2100, 1900, 2200, 2300, 2500, 2700]
}
df = pd.DataFrame(data)
plt.plot(df["Month"], df["Sales"], marker='o', linestyle='-', linewidth=2)
plt.title("Monthly Sales of Company")
plt.xlabel("Month")
plt.ylabel("Sales (in USD)")
plt.grid(True)
plt.show()
```

Output:



Question:

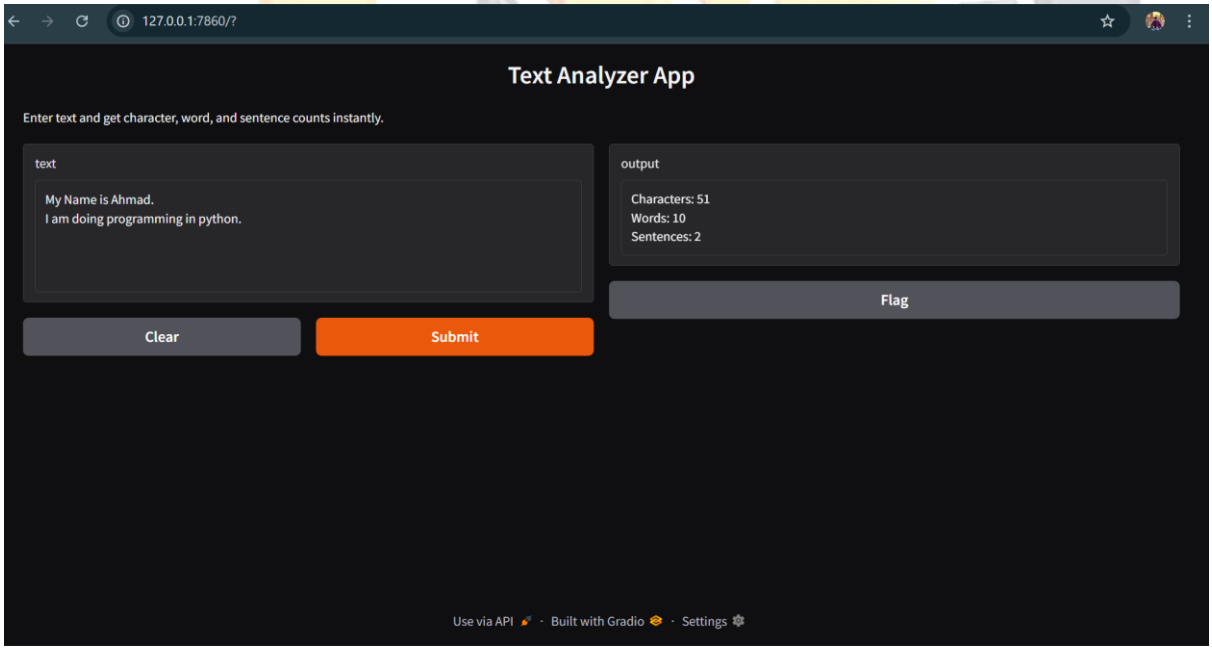
1. **Text analyzer character count app using gradio.**

Code:

Task06.py > ...

```python
import gradio as gr
def text_analyzer(text):
    char_count = len(text)
    word_count = len(text.split())
    sentence_count = text.count('.') + text.count('!') + text.count('?')
    return f"Characters: {char_count}\nWords: {word_count}\nSentences: {sentence_count}"
app = gr.Interface(
    fn=text_analyzer,
    inputs=gr.Textbox(lines=5, placeholder="Enter your text here..."),
    outputs="text",
    title="Text Analyzer App",
    description="Enter text and get character, word, and sentence counts instantly."
)
app.launch()
```

Output:



Reflection on Week 04 Python Tasks

During these tasks, I learned how to apply Python libraries like **pandas**, **matplotlib**, and **gradio** to solve real-world problems.

Data Handling with Pandas

I practiced loading data from a CSV file, cleaning it by handling missing values, and calculating averages.

I understood the importance of data preprocessing before performing analysis.

Data Visualization

By creating **bar charts** and **line charts**, I learned how to represent data visually using **matplotlib**.

The bar chart helped me see students' marks distribution, while the line chart showed sales trends across months.

This improved my skills in making data more meaningful and easy to understand.

Building Interactive Apps with Gradio

I created a simple **Text Analyzer App** where I learned to design an interface and connect Python functions with a web UI.

I understood how Python code can be turned into a usable app for end-users without extra web development.

Error Solving & Debugging

While working, I faced errors like FileNotFoundError, EmptyDataError, and ModuleNotFoundError.

Fixing these taught me how to check CSV files, install missing libraries, and debug step-by-step.

Learnings

These tasks gave me **hands-on experience** in:

Data cleaning and analysis with pandas.

Visualizing information with charts.

Creating interactive Python apps using Gradio.

Debugging and managing Python environments.

I realized how **Python can handle data, visualization, and even web apps** all in one language, which will help me in both academic projects and future professional work.

TECHNIK NEST