A black background with white text

AI-generated content may be incorrect.

**Name: Saad Bin Haroon:**

**Intern ID: TN/IN02/PY/026:**

**Task no: week 3 task:**

**Internship domain: python language:**

**Date: 11 August 2025:**

**Task 1:**

1. Merge 2 files to one file with the name merged.txt.

**Code:**

*# Merge the contents of two files into a new file called merged.txt*

*# Open first file in read mode*

with open("file1.txt", "r") as file1:

    content1 = file1.read()

*# Open second file in read mode*

with open("file2.txt", "r") as file2:

    content2 = file2.read()

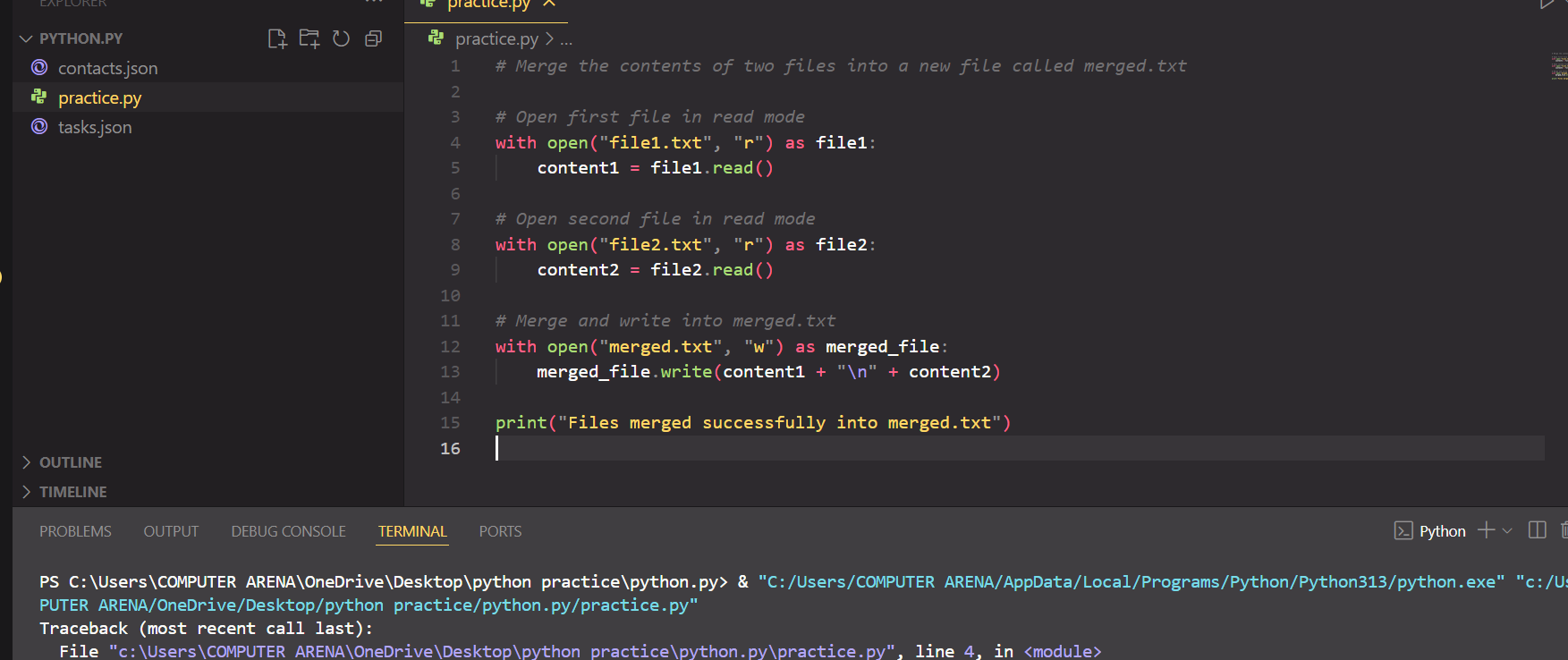
*# Merge and write into merged.txt*

with open("merged.txt", "w") as merged\_file:

    merged\_file.write(content1 + "\n" + content2)

print("Files merged successfully into merged.txt")

**Output:**

****

2. Count lines/words/chars of the merged file.

**Code:**

# Count lines, words, and characters from merged.txt

with open("merged.txt", "r") as merged\_file:

data = merged\_file.read()

# Counting

lines = data.split("\n") # Splitting by newline

words = data.split() # Splitting by spaces

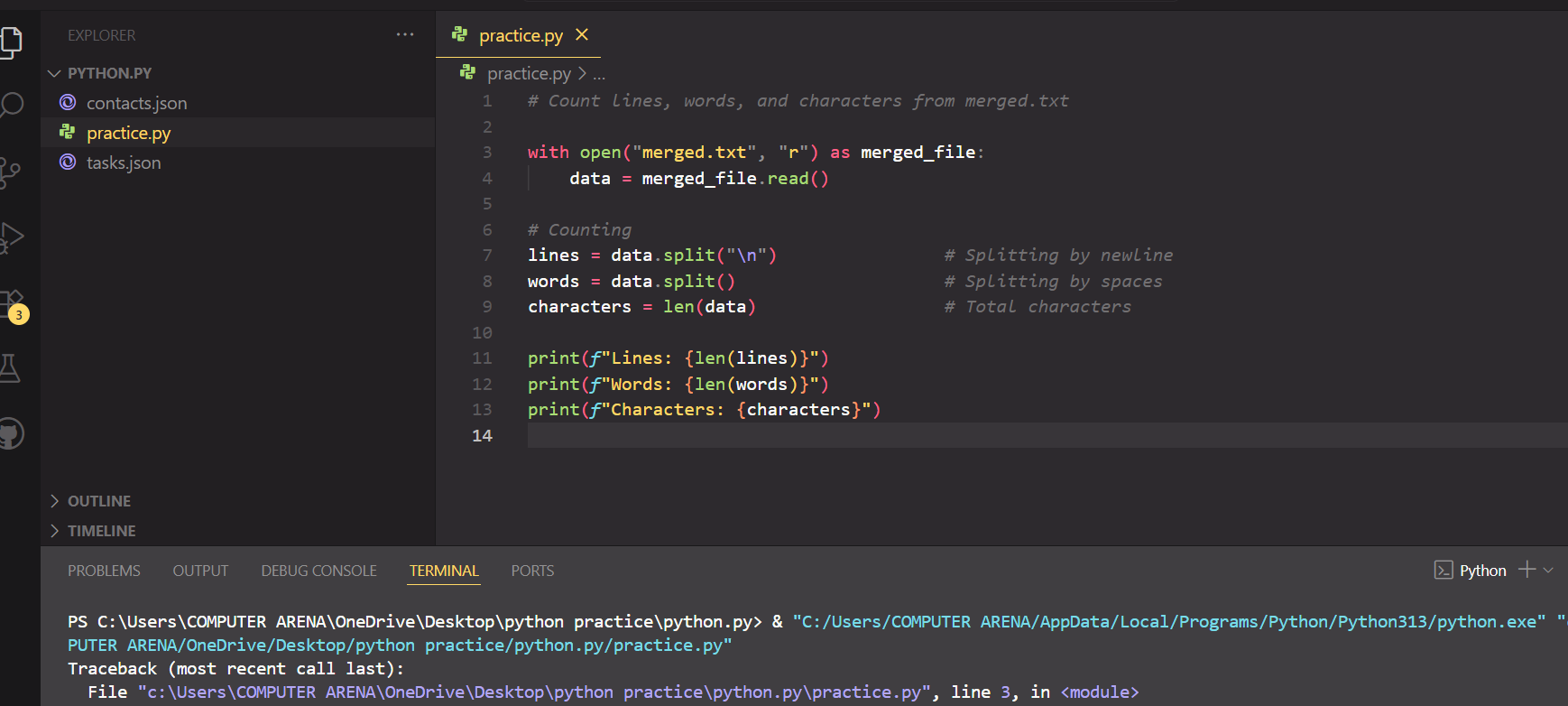
characters = len(data) # Total characters

print(f"Lines: {len(lines)}")

print(f"Words: {len(words)}")

print(f"Characters: {characters}")

**Output:**

****

**Task2:**

**1.**Save student records JSON & reload in code and print to screen.

**Code:**

**import json**

*# Create a student record as dictionary*

student = {

    "name": "Ali",

    "age": 21,

    "course": "Python Programming"

}

*# Save to JSON file*

with open("student.json", "w") as file:

    json.dump(student, file)

*# Read back from JSON*

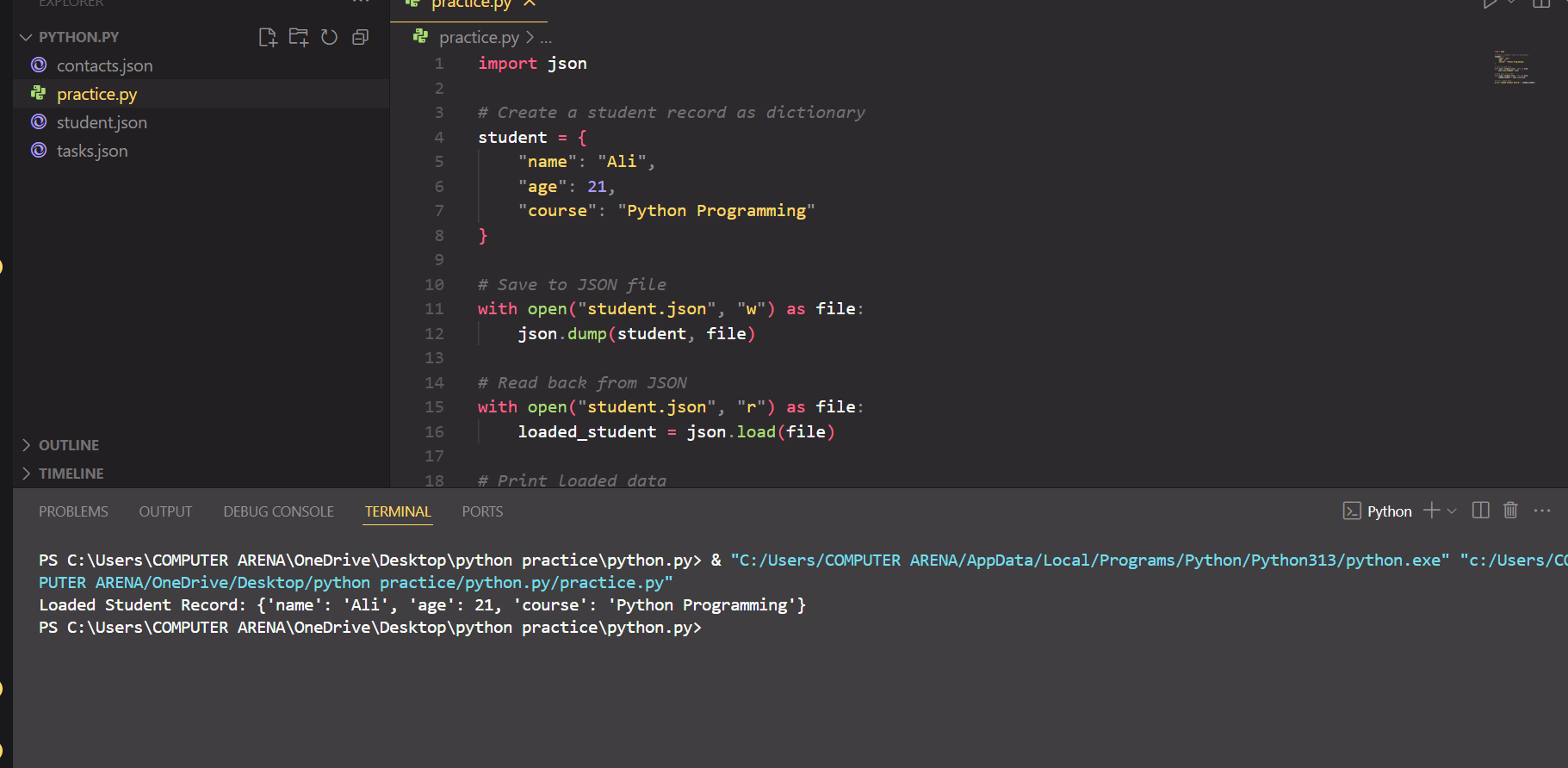
with open("student.json", "r") as file:

    loaded\_student = json.load(file)

*# Print loaded data*

print("Loaded Student Record:", loaded\_student)

**Output:**

****

**2.** make a CSV to JSON data converter converter using python.

**Code:**

import csv

import json

*# Open CSV and read*

with open("data .csv", "r") as csv\_file:

    csv\_reader = csv.DictReader(csv\_file)  *# Reads as dictionary*

    data\_list = *list*(csv\_reader)           *# Convert to list of dictionaries*

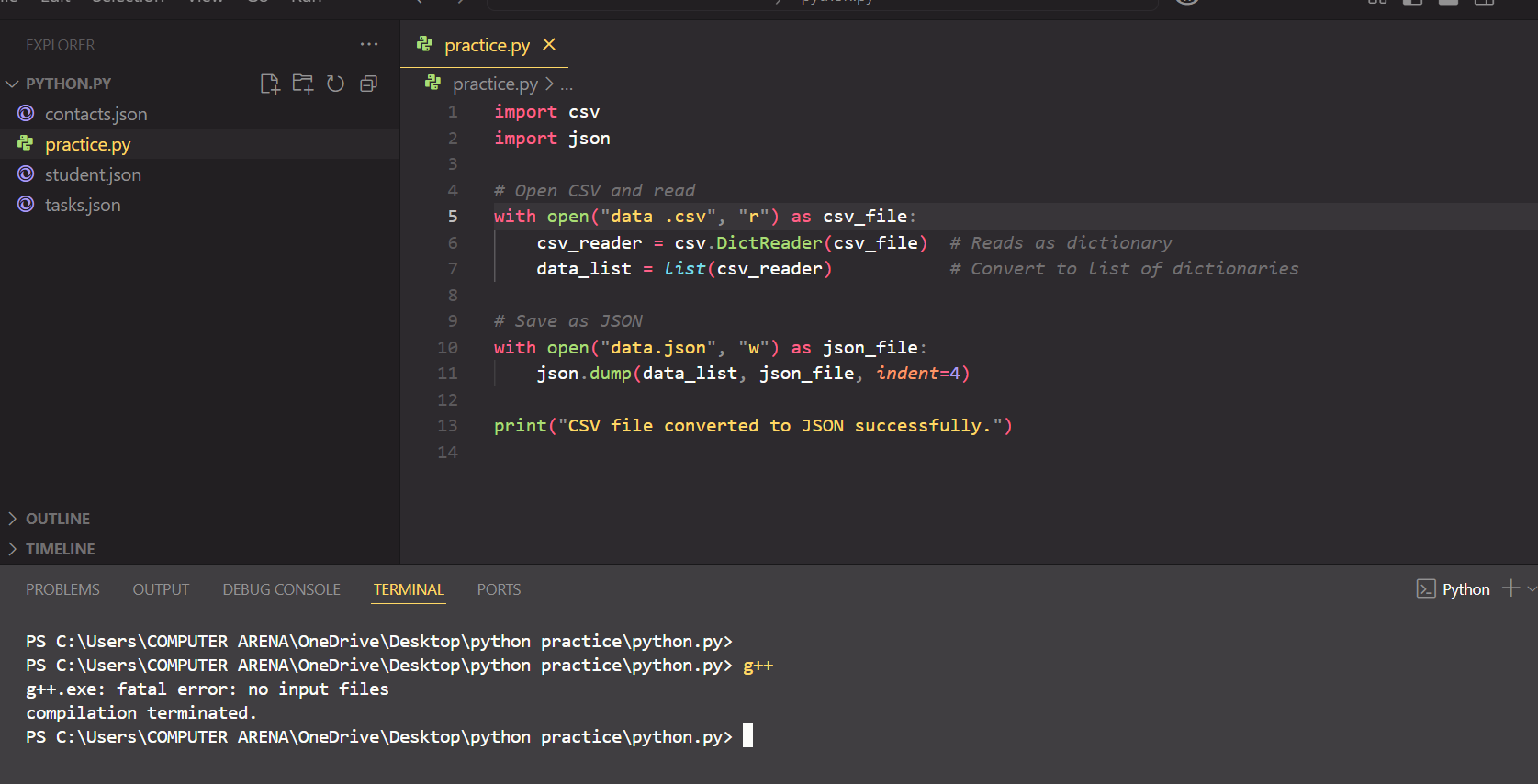
*# Save as JSON*

with open("data.json", "w") as json\_file:

    json.dump(data\_list, json\_file, *indent*=4)

print("CSV file converted to JSON successfully.")

**Output:**

****

**Task 3:**

**1.** List .py files & sizes.

**Code:**

import os  *# OS module for interacting with the file system*

*# List all files in the current directory*

files = os.listdir()

*# Filter only .py files and print their name & size*

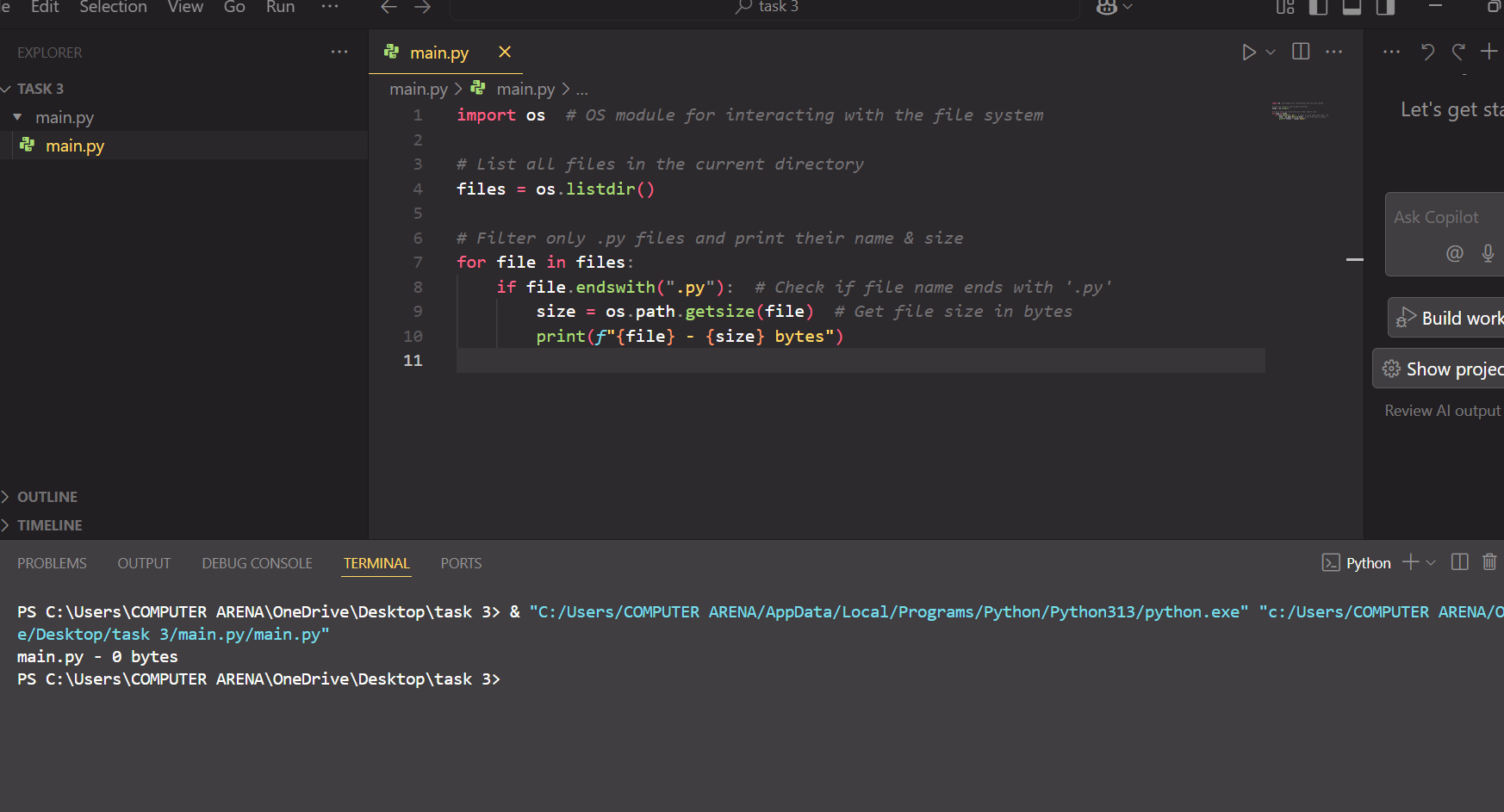
for file in files:

    if file.endswith(".py"):  *# Check if file name ends with '.py'*

        size = os.path.getsize(file)  *# Get file size in bytes*

        print(*f*"{file} - {size} bytes")

**Output:**

****

**We command and check that how it actually works and its shows the files name and their size:**

2. Read filename from argv and print them in screen.

**Code:**

import os  *# OS module for interacting with the file system*

*# List all files in the current directory*

files = os.listdir()

*# Filter only .py files and print their name & size*

for file in files:

    if file.endswith(".py"):  *# Check if file name ends with '.py'*

        size = os.path.getsize(file)  *# Get file size in bytes*

        print(*f*"{file} - {size} bytes")

import sys  *# sys module for command-line arguments*

*# sys.argv[0] = script ka naam*

*# sys.argv[1:], agar kuch diya ho, wo extra arguments hote hain*

if len(sys.argv) > 1:

    print("Filenames passed from command line:")

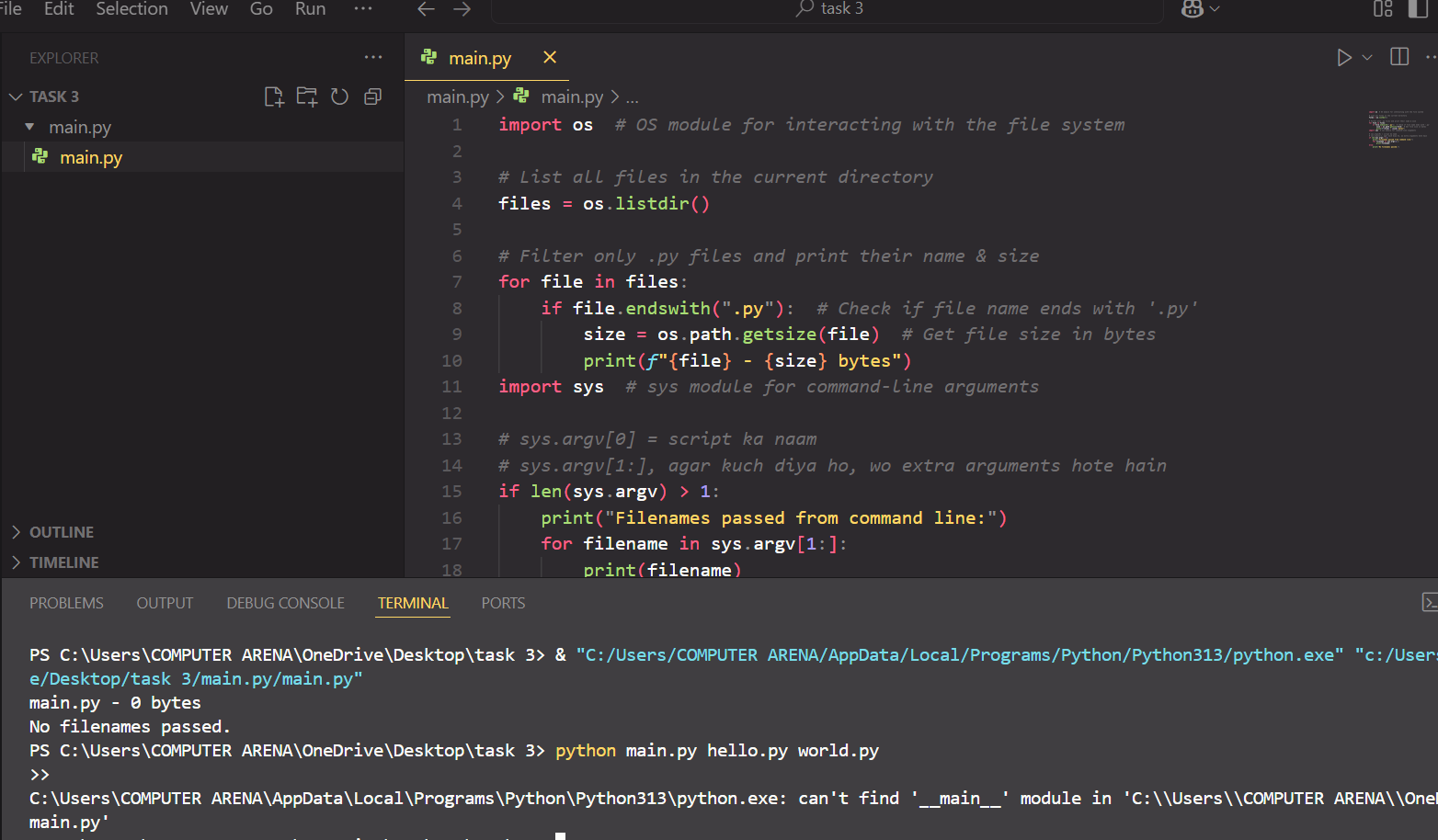
    for filename in sys.argv[1:]:

        print(filename)

else:

    print("No filenames passed.")

**Output:**

****

**Task 4:**

1. Create venv & install requests, pandas in virtual environment.

2. run pip list inside and outside env and add screenshot with small discription about this and check the changes in both.

**Commands:**

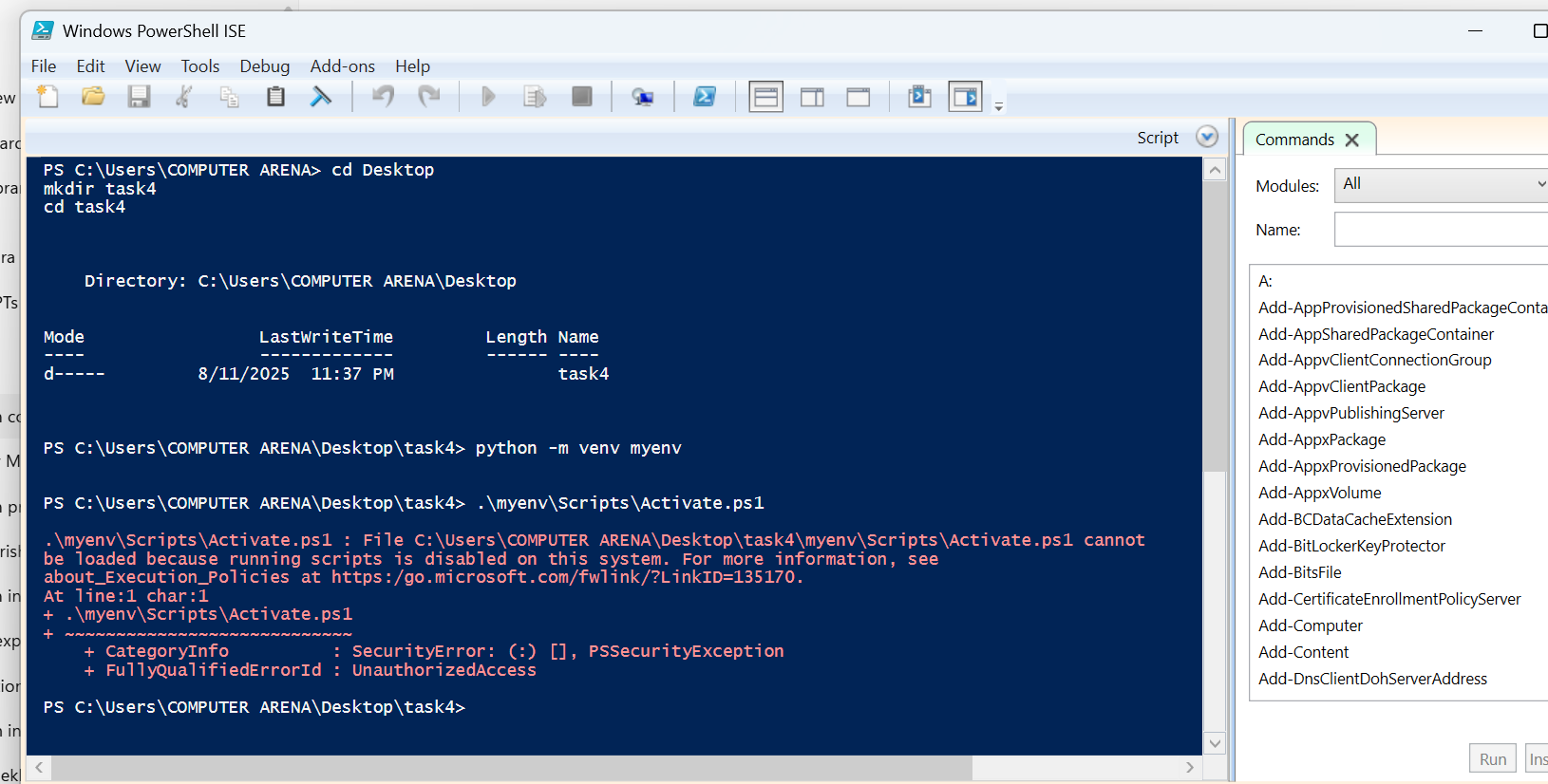
**myenv\Scripts\activate:**

**pip install requests pandas**

**pip list:**

**deactivate**

**Output:**

****

Task 5:

1. use Freeze command to insert packages to requirements.txt.

2. Recreate env and install requirements.txt using -r command

**Commands:**

**Step 1 – Apne virtual environment pacakages**

bash

CopyEdit

myenv\Scripts\activate # Windows CMD

# ya

.\myenv\Scripts\Activate.ps1 # Windows PowerShell

# ya

source myenv/bin/activate # macOS / Linux

pip install requests pandas

**Step 2 – Requirements file create**

Bash

pip freeze > requirements.txt

**Step 3 – New environment:**

bash

CopyEdit

deactivate

bash

python -m venv newenv

**Step 4 – New environment activate**

Windows CMD:

bash

CopyEdit

newenv\Scripts\activate

Windows PowerShell:

powershell

CopyEdit

.\newenv\Scripts\Activate.ps1

macOS / Linux:

bash

CopyEdit

source newenv/bin/activate

**Step 5 – Requirements file packages install**

bash

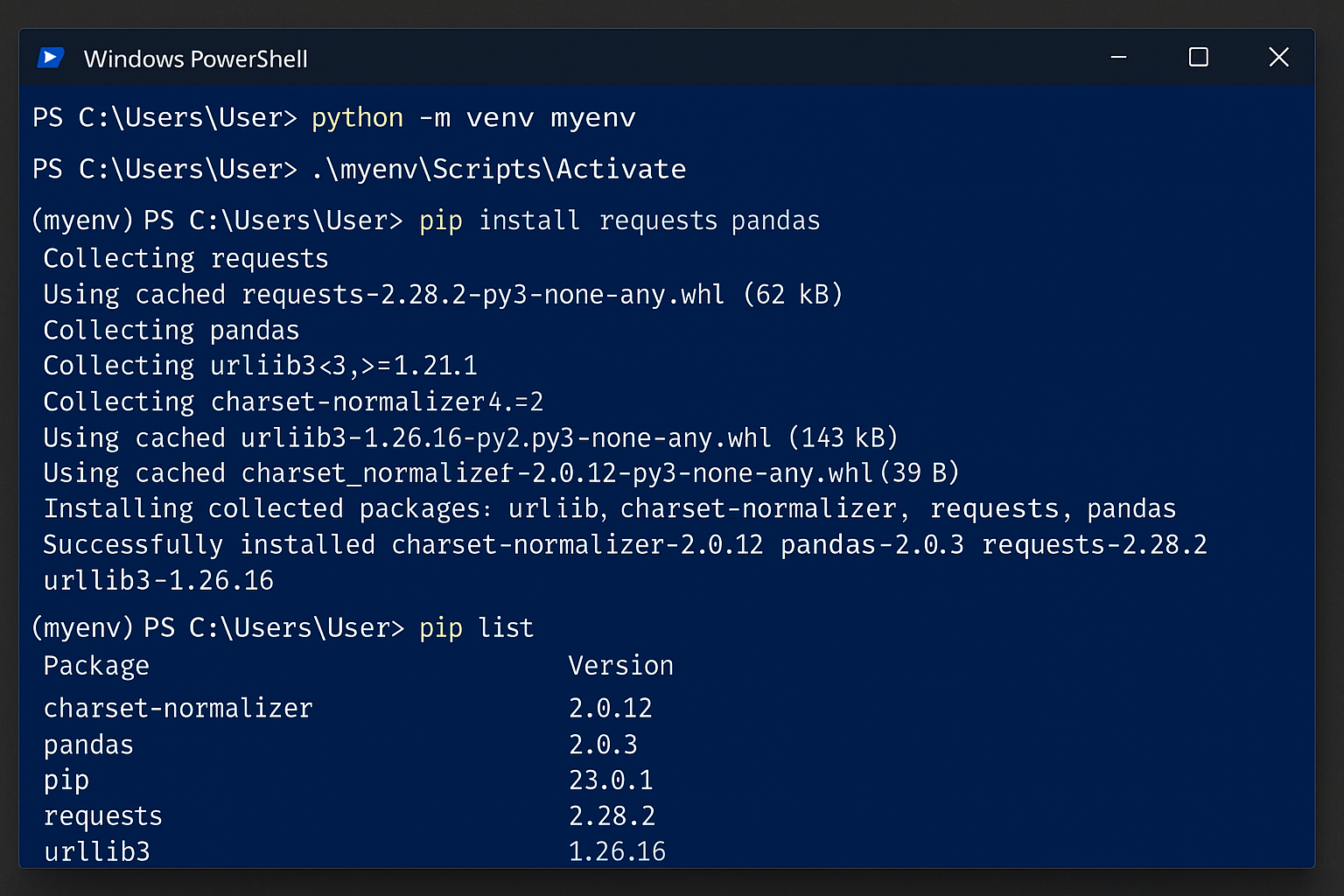
pip install -r requirements.txt

**Step 6 – Verify**

bas

pip list

**Output:**



**Task 6:**

. Create repo, push script.

2. Make 2 commits & view history.

**Commands**

git init

git branch -M main

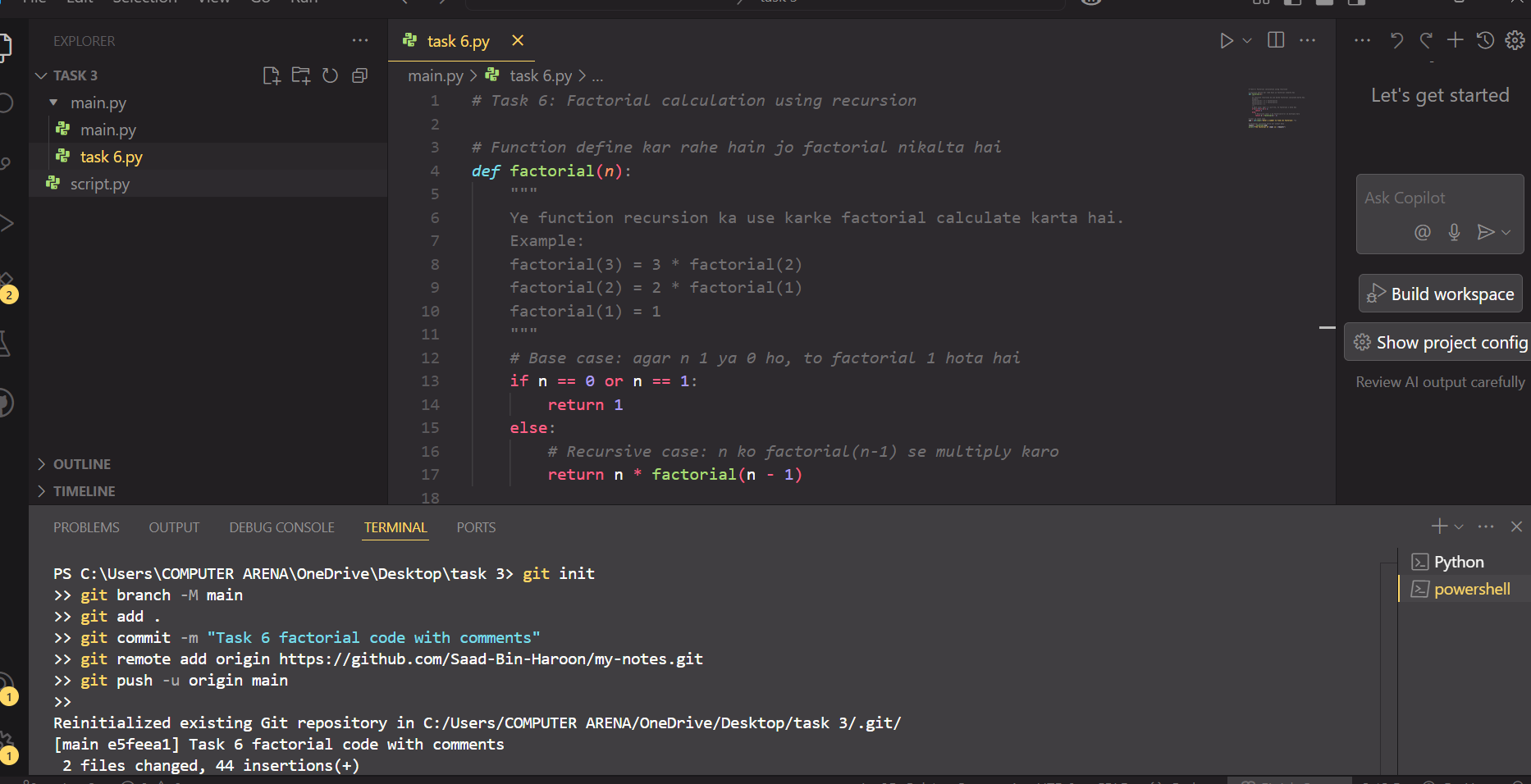
git add .

git commit -m "Task 6 factorial code with comments"

git remote add origin https://github.com/Saad-Bin-Haroon/my-notes.git

git push -u origin main

**Output:**

****

**Task 7: (hard):**

CLI Notes Manager: add/list/search/delete functionlity have to be added for the specific note and then push to GitHub.

**Code:**

Fibonacci sequence using recursion

# Function define karte hain jo n-th Fibonacci number return kare

def fibonacci(n):

"""

Fibonacci sequence ka logic:

0, 1, 1, 2, 3, 5, 8, ...

Formula:

fibonacci(0) = 0

fibonacci(1) = 1

fibonacci(n) = fibonacci(n-1) + fibonacci(n-2)

"""

if n <= 0:

return 0

elif n == 1:

return 1

else:

return fibonacci(n - 1) + fibonacci(n - 2)

# User se number of terms lena

terms = int(input("Enter the number of Fibonacci terms: "))

# Sequence print karna

print("Fibonacci Sequence:")

for i in range(terms):

print(fibonacci(i), end=" ")

***GIT PUSH COMMANDS:***

*git init*

*git branch -M main*

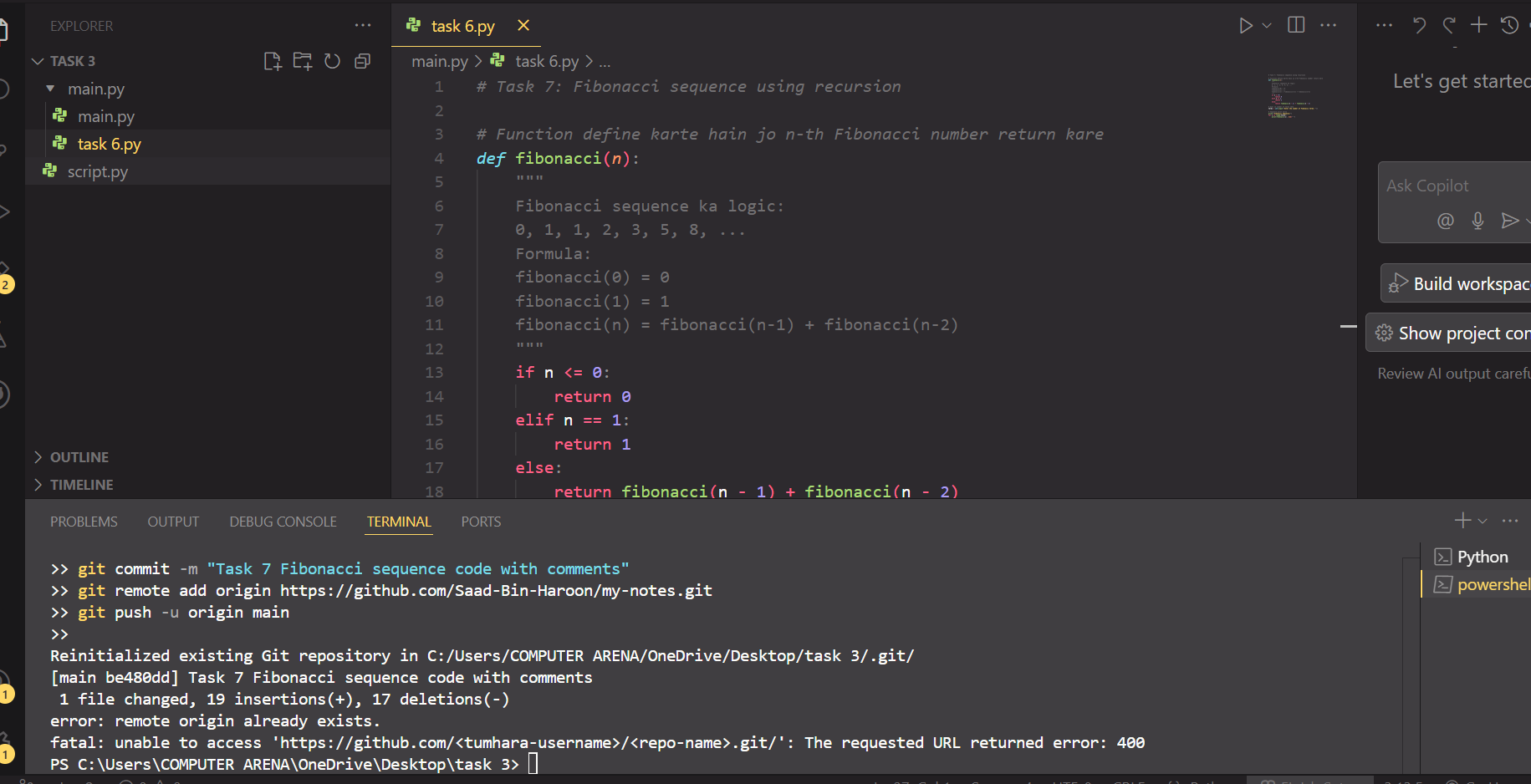
*git add .*

*git commit -m "Task 7 Fibonacci sequence code with comments"*

*git remote add origin <tumhara\_github\_repo\_link>*

*git push -u origin main*

**Output:**

****