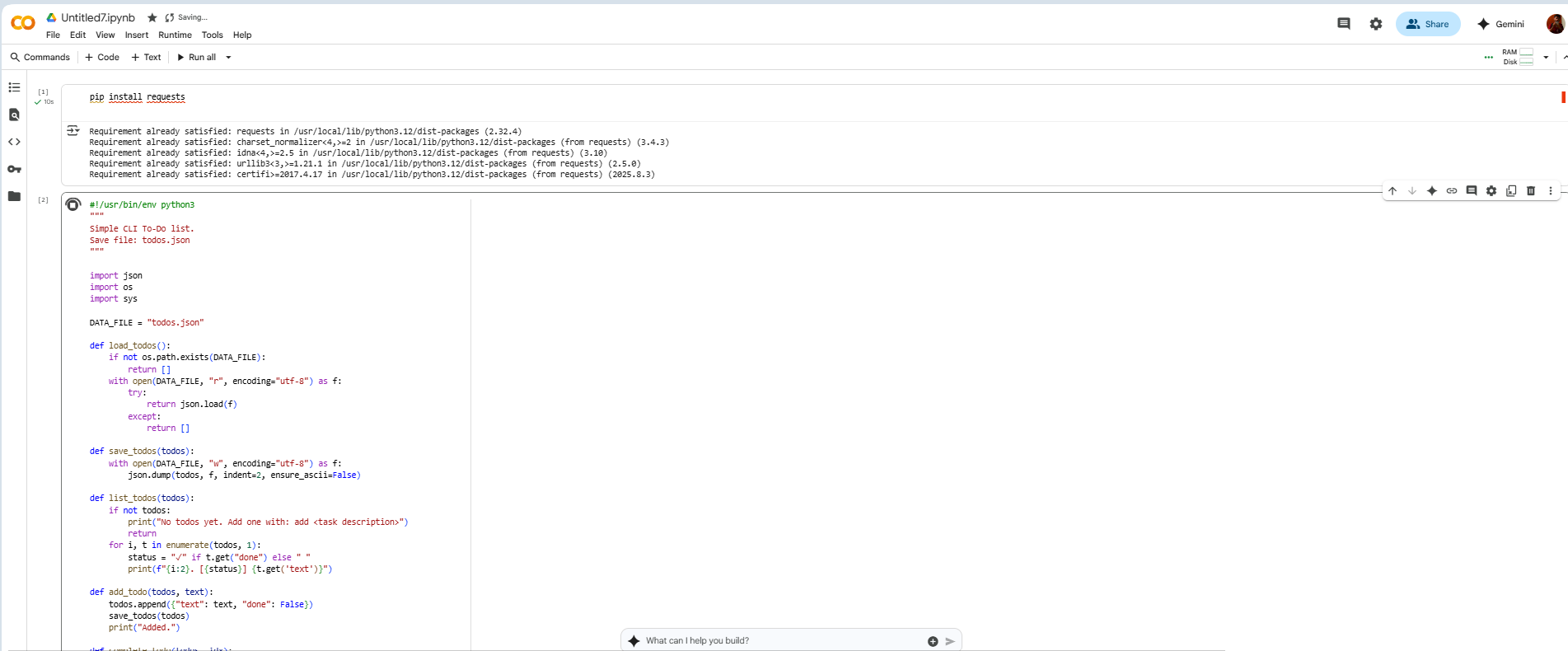
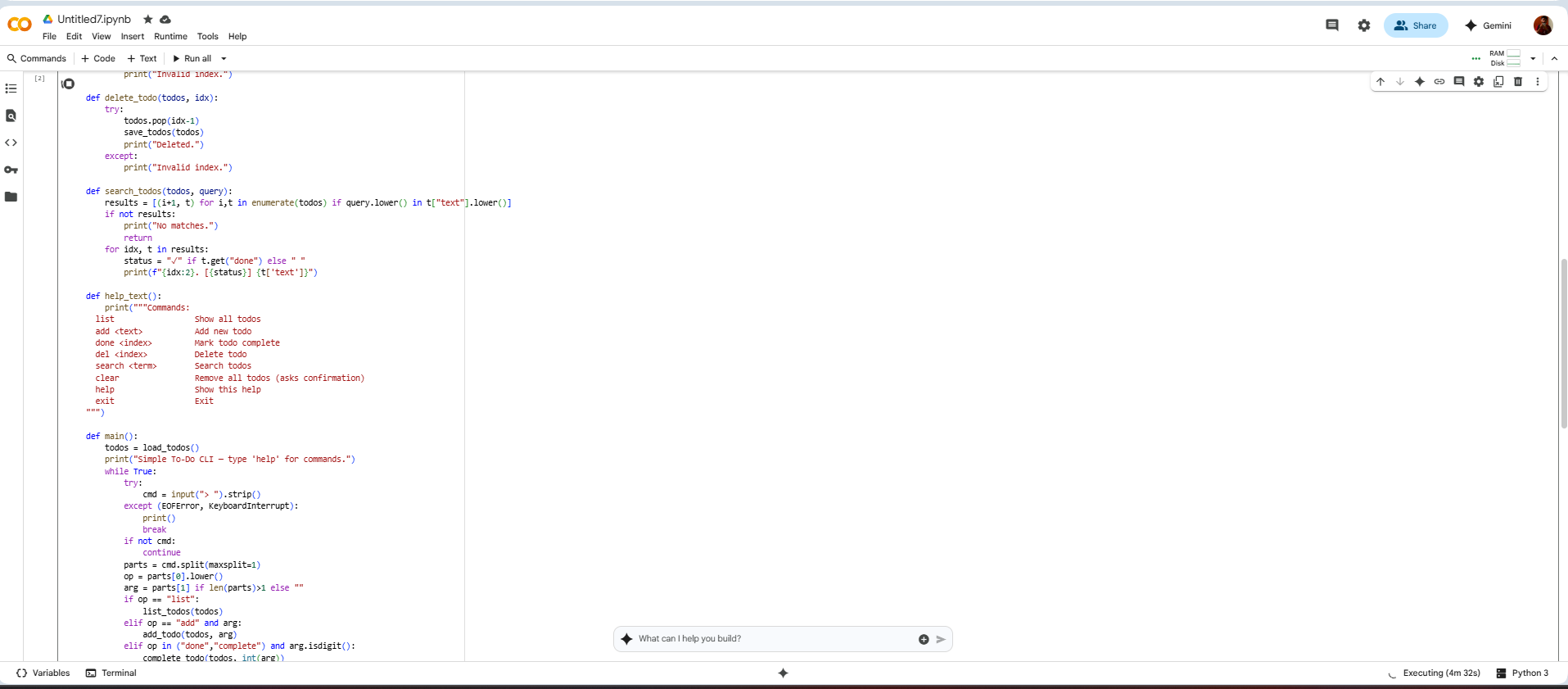
**Python Tasks**

**Task no 1: TO DO LIST (CLI)**







Code:

pip install requests

#!/usr/bin/env python3

"""

Simple CLI To-Do list.

Save file: todos.json

"""

import json

import os

import sys

DATA\_FILE = "todos.json"

def load\_todos():

if not os.path.exists(DATA\_FILE):

return []

with open(DATA\_FILE, "r", encoding="utf-8") as f:

try:

return json.load(f)

except:

return []

def save\_todos(todos):

with open(DATA\_FILE, "w", encoding="utf-8") as f:

json.dump(todos, f, indent=2, ensure\_ascii=False)

def list\_todos(todos):

if not todos:

print("No todos yet. Add one with: add <task description>")

return

for i, t in enumerate(todos, 1):

status = "✓" if t.get("done") else " "

print(f"{i:2}. [{status}] {t.get('text')}")

def add\_todo(todos, text):

todos.append({"text": text, "done": False})

save\_todos(todos)

print("Added.")

def complete\_todo(todos, idx):

try:

todos[idx-1]["done"] = True

save\_todos(todos)

print("Marked complete.")

except:

print("Invalid index.")

def delete\_todo(todos, idx):

try:

todos.pop(idx-1)

save\_todos(todos)

print("Deleted.")

except:

print("Invalid index.")

def search\_todos(todos, query):

results = [(i+1, t) for i,t in enumerate(todos) if query.lower() in t["text"].lower()]

if not results:

print("No matches.")

return

for idx, t in results:

status = "✓" if t.get("done") else " "

print(f"{idx:2}. [{status}] {t['text']}")

def help\_text():

print("""Commands:

list Show all todos

add <text> Add new todo

done <index> Mark todo complete

del <index> Delete todo

search <term> Search todos

clear Remove all todos (asks confirmation)

help Show this help

exit Exit

""")

def main():

todos = load\_todos()

print("Simple To-Do CLI — type 'help' for commands.")

while True:

try:

cmd = input("> ").strip()

except (EOFError, KeyboardInterrupt):

print()

break

if not cmd:

continue

parts = cmd.split(maxsplit=1)

op = parts[0].lower()

arg = parts[1] if len(parts)>1 else ""

if op == "list":

list\_todos(todos)

elif op == "add" and arg:

add\_todo(todos, arg)

elif op in ("done","complete") and arg.isdigit():

complete\_todo(todos, int(arg))

elif op in ("del","delete") and arg.isdigit():

delete\_todo(todos, int(arg))

elif op == "search" and arg:

search\_todos(todos, arg)

elif op == "clear":

confirm = input("Really clear all todos? (y/N) ").lower()

if confirm == "y":

todos.clear()

save\_todos(todos)

print("Cleared.")

elif op in ("help","h","?"):

help\_text()

elif op in ("exit","quit"):

break

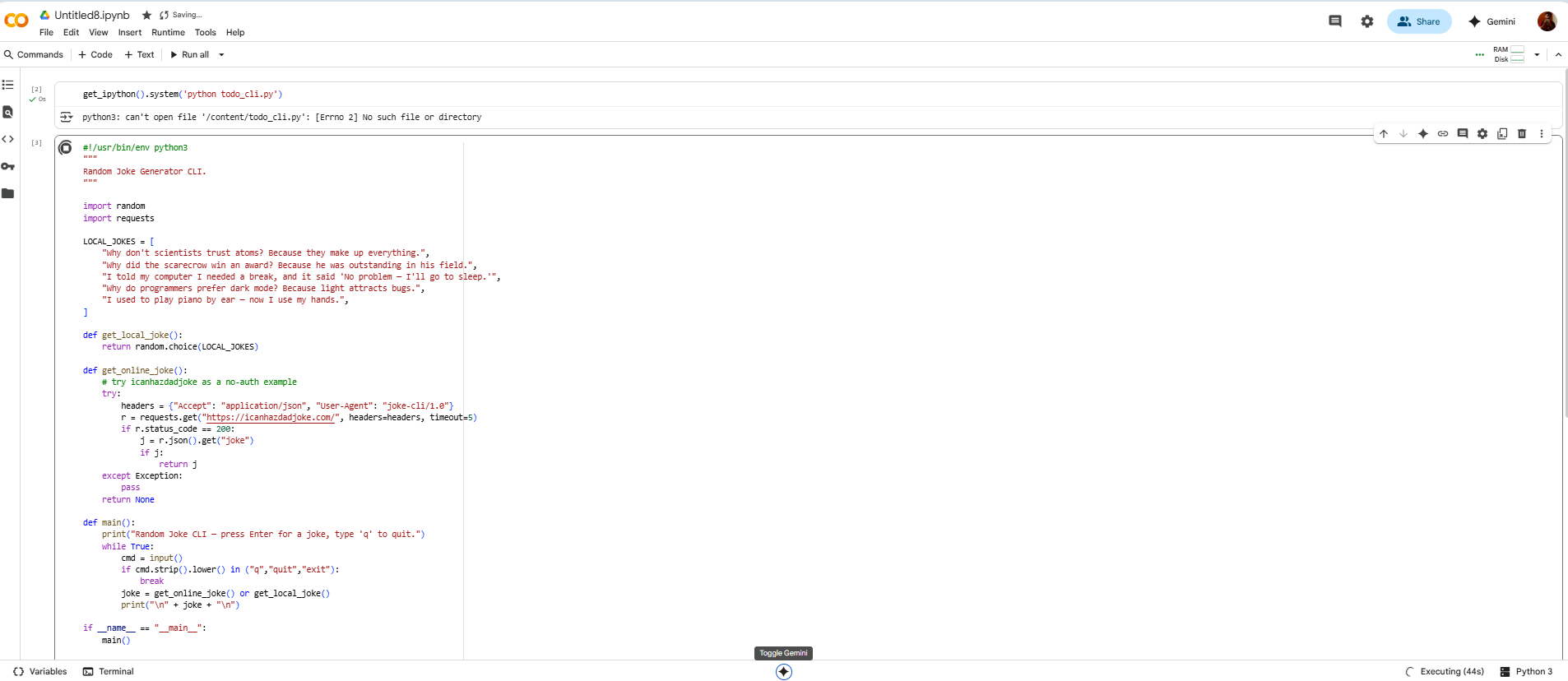
else:

print("Unknown command. Type 'help'.")

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Task no 2: Random Joke Generator**





Code:

#!/usr/bin/env python3

"""

Random Joke Generator CLI.

"""

import random

import requests

LOCAL\_JOKES = [

"Why don't scientists trust atoms? Because they make up everything.",

"Why did the scarecrow win an award? Because he was outstanding in his field.",

"I told my computer I needed a break, and it said 'No problem — I'll go to sleep.'",

"Why do programmers prefer dark mode? Because light attracts bugs.",

"I used to play piano by ear — now I use my hands.",

]

def get\_local\_joke():

return random.choice(LOCAL\_JOKES)

def get\_online\_joke():

# try icanhazdadjoke as a no-auth example

try:

headers = {"Accept": "application/json", "User-Agent": "joke-cli/1.0"}

r = requests.get("https://icanhazdadjoke.com/", headers=headers, timeout=5)

if r.status\_code == 200:

j = r.json().get("joke")

if j:

return j

except Exception:

pass

return None

def main():

print("Random Joke CLI — press Enter for a joke, type 'q' to quit.")

while True:

cmd = input()

if cmd.strip().lower() in ("q","quit","exit"):

break

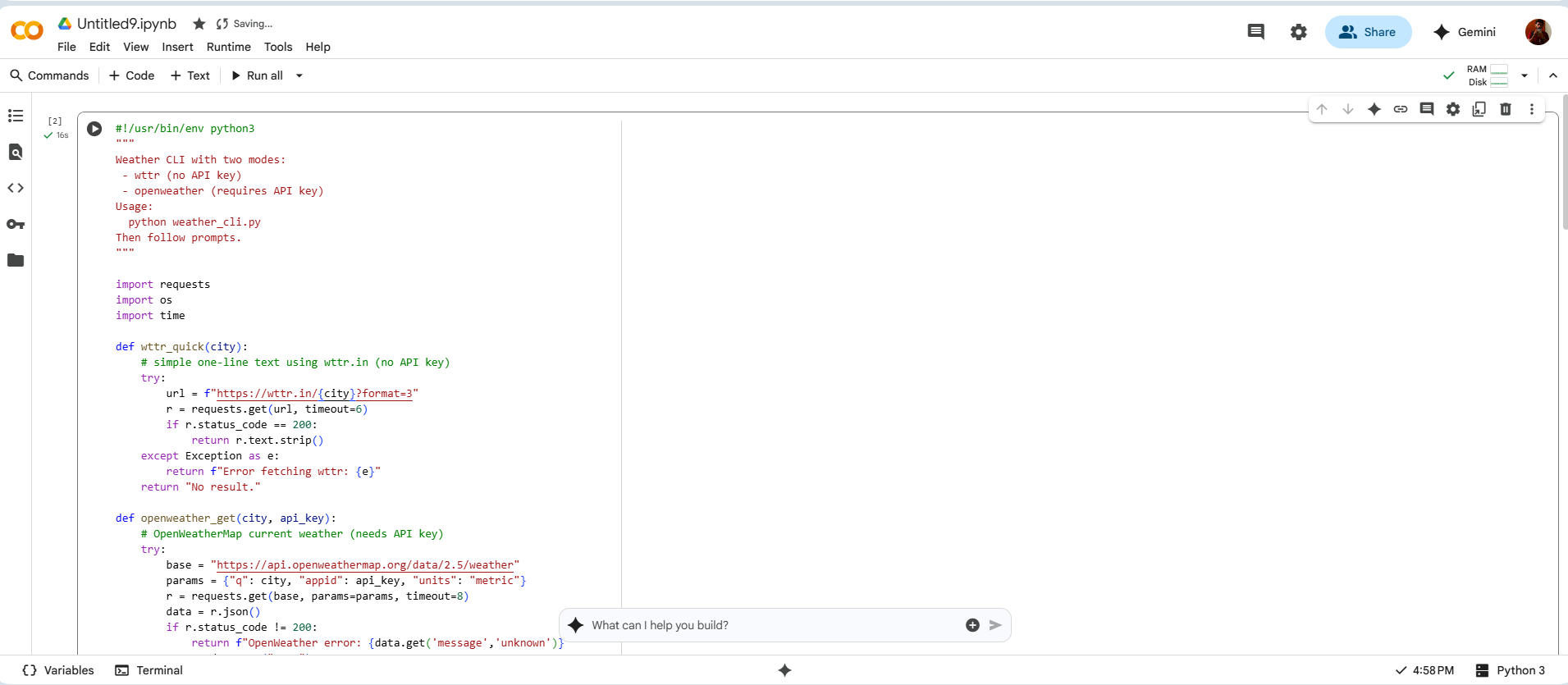
joke = get\_online\_joke() or get\_local\_joke()

print("\n" + joke + "\n")

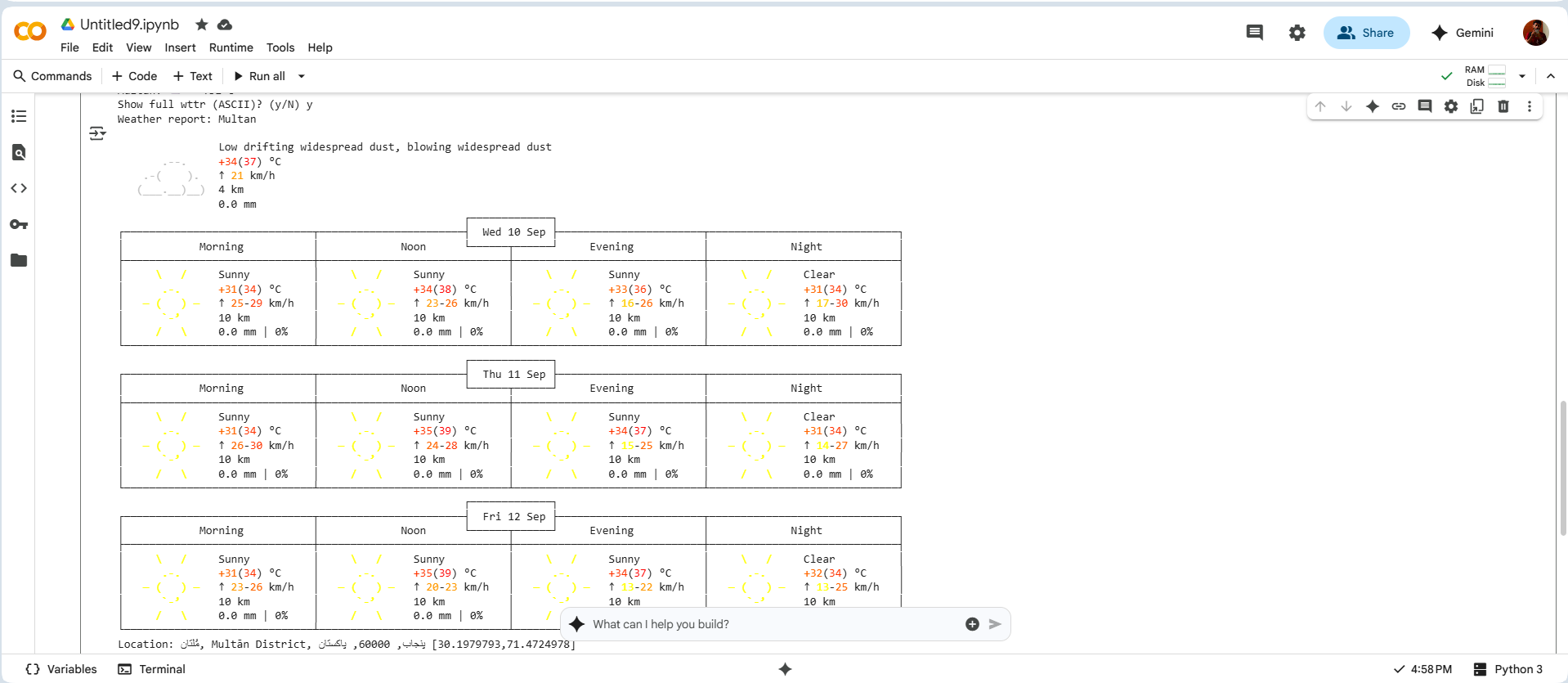
if \_\_name\_\_ == "\_\_main\_\_":

main()

**Task no 3: Weather app (CLI)**







Code:

#!/usr/bin/env python3

"""

Weather CLI with two modes:

- wttr (no API key)

- openweather (requires API key)

Usage:

python weather\_cli.py

Then follow prompts.

"""

import requests

import os

import time

def wttr\_quick(city):

# simple one-line text using wttr.in (no API key)

try:

url = f"https://wttr.in/{city}?format=3"

r = requests.get(url, timeout=6)

if r.status\_code == 200:

return r.text.strip()

except Exception as e:

return f"Error fetching wttr: {e}"

return "No result."

def openweather\_get(city, api\_key):

# OpenWeatherMap current weather (needs API key)

try:

base = "https://api.openweathermap.org/data/2.5/weather"

params = {"q": city, "appid": api\_key, "units": "metric"}

r = requests.get(base, params=params, timeout=8)

data = r.json()

if r.status\_code != 200:

return f"OpenWeather error: {data.get('message','unknown')}"

name = data.get("name")

sys = data.get("sys",{})

country = sys.get("country","")

main = data.get("main",{})

weather = (data.get("weather") or [{}])[0]

s = (

f"{name},{country} — {weather.get('main','')} ({weather.get('description','')})\n"

f"Temp: {main.get('temp')}°C | Feels: {main.get('feels\_like')}°C | "

f"Humidity: {main.get('humidity')}%\n"

)

return s

except Exception as e:

return f"Error calling OpenWeather: {e}"

def main():

print("Weather CLI")

city = input("City (e.g. Multan): ").strip() or "Multan"

print("Choose mode: 1) wttr.in (no key) 2) OpenWeatherMap (API key required)")

mode = input("Mode [1]: ").strip() or "1"

if mode == "2":

api\_key = os.environ.get("OPENWEATHER\_API\_KEY") or input("Enter OpenWeatherMap API key: ").strip()

print("Fetching from OpenWeatherMap...")

print(openweather\_get(city, api\_key))

else:

print("Fetching quick result from wttr.in...")

print(wttr\_quick(city))

# optionally show detailed ascii:

more = input("Show full wttr (ASCII)? (y/N) ").strip().lower()

if more == "y":

try:

r = requests.get(f"https://wttr.in/{city}", timeout=8)

print(r.text)

except Exception as e:

print("Error fetching detailed wttr:", e)

if \_\_name\_\_ == "\_\_main\_\_":

main()