To check the status of the subdomains and dispaly the results.

step 1:

we need to install required libraries

‘‘‘bash
pip install requests tabulate
‘‘‘

step2:

create a file

subdomains.py

‘‘‘bash
import requests
import time
from tabulate import tabulate

# Define the list of subdomains to check
subdomains = [
    ’awesomeweb1.com’,
    ’awesomeweb2.com’,
    ’awesomeweb3.com’,
    # Add more subdomains here
]

# Function to check the status of a subdomain
def check_status(subdomain):
    try:
        response = requests.get(f"http://{awesomeweb}", timeout=5)
        status = "Up" if response.status_code == 200 else "Down"
    except requests.ConnectionError:
        status = "Down"
    return subdomain, status

# Function to display the status in tabular format
def display_status(subdomain_status):
    headers = ["Subdomain", "Status"]
    print(tabulate(subdomain_status, headers=headers, tablefmt="grid"))

# Main loop to check status every minute
if __name__ == "__main__":
    while True:
        subdomain_status = []
        for subdomain in subdomains:
            result = check_status(subdomain)
            subdomain_status.append(result)

        display_status(subdomain_status)
        print("Checking again in 1 minute...")
        time.sleep(60)
‘‘‘

Step 3: Explanation of the script

* We import the necessary libraries: requests to perform HTTP requests and tabulate to format the status table.

* We define the list subdomains containing the subdomains to check. You can add more subdomains to th is list as needed.

* The check_status() function takes a subdomain as input and attempts to make an HTTP request to it. If t he request succeeds (status code 200), we consider the subdomain as "Up"; otherwise, it is "Down."

* The display_status() function takes a list of tuples containing the subdomain and its status (e.g., [('awes omeweb1.com', 'Up'), ('awesomeweb2.com', 'Down')]) and prints the status in a tabular format using tabul ate.

* The main loop runs indefinitely. It iterates through each subdomain, checks its status using check_statu s(), and stores the results in the subdomain_status list. Then, it calls display_status() to print the results in a tabular format.

* After displaying the status, the script sleeps for 60 seconds using time.sleep(60) before checking the sta tus again.

Step 4: Running the script

‘‘‘bash
python subdomain_checker.py
‘‘‘

The script will start checking the status of the subdomains every minute and display the results in tabular f ormat on the screen. It will continue running until you manually interrupt it