### **Project 11: Multi-threaded Program**

#### **Objective:**

Build a multi-threaded file downloader that can handle multiple files simultaneously, displaying download progress for each file. This project will use the threading library to create a program that efficiently downloads files from a list of URLs.

### **Instructions**

#### **Step 1: Set Up the Environment**

1. Create a new Python file called multi\_threaded\_downloader.py.

Install the required libraries:  
Copy code  
pip install requests tqdm

#### **Step 2: Import Necessary Libraries**

Import the necessary libraries at the top of your script:  
python  
Copy code  
import threading

import requests

import os

from tqdm import tqdm

#### **Step 3: Create a List of File URLs**

Define a list of file URLs that you want to download:  
python  
Copy code  
file\_urls = [

"https://example.com/file1.zip",

"https://example.com/file2.zip",

"https://example.com/file3.zip"

]

#### **Step 4: Create a Function to Download a File**

1. Create a function called download\_file(url, folder) that:
   * Sends a request to download the file from the provided URL.
   * Uses tqdm to display a progress bar for the download.
   * Saves the file in the specified folder.

python  
Copy code  
def download\_file(url, folder):

file\_name = os.path.join(folder, url.split("/")[-1])

response = requests.get(url, stream=True)

total\_size = int(response.headers.get("content-length", 0))

block\_size = 1024 # 1 KB

# Display progress bar

tqdm\_bar = tqdm(total=total\_size, unit="B", unit\_scale=True, desc=file\_name)

with open(file\_name, "wb") as file:

for data in response.iter\_content(block\_size):

tqdm\_bar.update(len(data))

file.write(data)

tqdm\_bar.close()

print(f"{file\_name} downloaded successfully.")

#### **Step 5: Create a Function to Manage Downloads**

1. Create a function called download\_files(urls, folder) that:
   * Creates a thread for each file download.
   * Starts each thread and waits for all threads to complete.

python  
Copy code  
def download\_files(urls, folder="downloads"):

# Create folder if it doesn't exist

if not os.path.exists(folder):

os.makedirs(folder)

# Create a thread for each file download

threads = []

for url in urls:

thread = threading.Thread(target=download\_file, args=(url, folder))

threads.append(thread)

thread.start()

# Wait for all threads to complete

for thread in threads:

thread.join()

#### **Step 6: Create the Main Function**

Define a main() function that initializes the download process:  
python  
Copy code  
def main():

folder = "downloads"

print("Starting multi-threaded file downloads...")

download\_files(file\_urls, folder)

print("All downloads completed successfully.")

#### **Step 7: Run the Program**

Add a condition to run the main() function when the script is executed:  
python  
Copy code  
if \_\_name\_\_ == "\_\_main\_\_":

main()

#### **Step 8: Run and Test the Downloader**

1. Test the script with various URLs to ensure that it downloads files correctly.
2. Observe the progress bars for each file and confirm that downloads complete simultaneously.