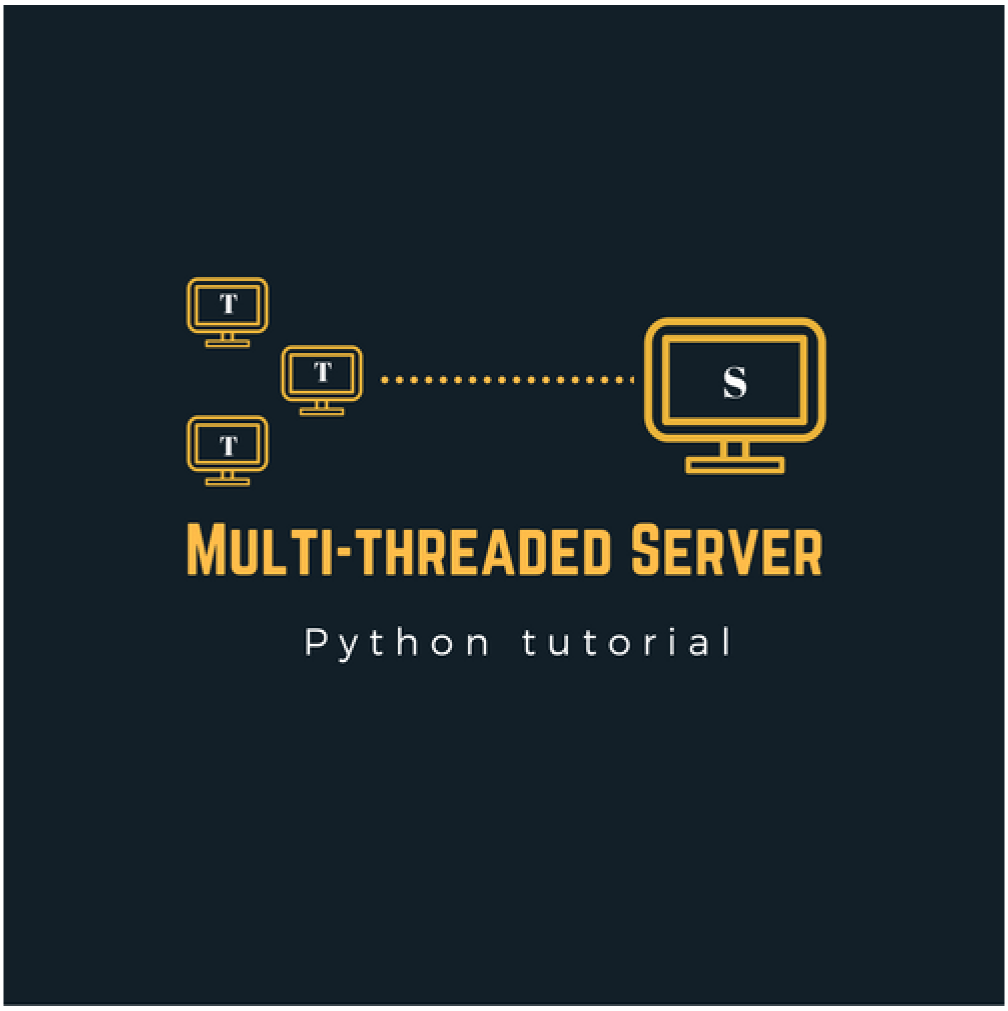
Multithreaded News Application/Server



- Semester: 02
- Group name: A14
- Course cose: ITNE352
- Section: 01
- Student names and ID: Husain Ali Husain - 202105828 / Ali Husain Ahmed - 202105543

Project Description:

Users can retrieve news headlines and sources from many online news providers by using the News API Application-Server System, a Python-based application. A Application script and a server script make up the system.

The server script responds to queries from Applications, interacts with the News API, and provides the requested information to the Applications. A user-friendly interface for communicating with the server and seeing the news data that has been retrieved is offered by the Application script.

Users can receive a list of news sources based on categories, nations, or languages, as well as search for news headlines based on keywords, categories, or countries. The News API is used by the system to retrieve real-time news information from various sources.

It's a handy tool for keeping up with current events because users can simply access and browse through a variety of news items and sources by executing the server script locally and connecting to it using the Application script.

Table of contents:

Requirements:

1. Python Installation: Ensure that Python is installed on your system. You can download the latest version of Python from the official Python website (https://www.python.org) and follow the installation instructions specific to your operating system.

2. Install Required Packages: The project relies on several packages that need to be installed. Open a terminal or command prompt and execute the following command to install the required packages using pip, the Python package installer:

```
pip install newsapi
```

3. API Key: The project utilizes the News API to retrieve news data. You will need to obtain an API key from the News API website (https://newsapi.org) by signing up for an account. Once you have the API key, make sure to keep it secure and confidential.

How to:

1. Run the Server:

- Open a terminal or command prompt.
- Navigate to the directory containing the server script (server.py).
- Execute the following command to start the server:

```
python server.py
```

2. Run the Application: Open a terminal or command prompt, navigate to the directory containing your GUI application script, and execute the following command to start the application:

```
python gui.py
```

3. Interacting with the System (Application):

- Once the Application is running, you will see a menu with different options.
- Use the provided menu options to interact with the system. For example, you can select options to search for news headlines based on keywords, categories, or countries, or retrieve a list of news sources based on categories, countries, or languages.
- Follow the on-screen instructions to input the desired criteria and retrieve the corresponding news data.
- The Application will communicate with the server, which will interact with the News API to fetch the requested data.
- The retrieved data will be displayed in the Application's terminal.

The scripts:

Server:

- The imported packages that are used by the server are:-

```
import socket
import threading
import json
import os
from newsapi import NewsApiClient
```

- The following server code will firstly checks if the current module is the main module being executed directly if it is true it will execute the StartupOfServer() function.The function will assign an IP address and port number to the server and bind them with the client and it will listen to at maximum of 3 different clients at a time then it will print a message containing its IP address and port number then while there is no que on the clients it will accept the client and start threading and go to the client_handling function.

```
if __name__ == "__main__":
    StartupOfServer()

def StartupOfServer():
    ip_address = "127.0.0.1"
    port_number = 4926
    ss = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    ss.bind((ip_address, port_number))
    ss.listen(3)
    print(f"Server is listening on...\n IP address: {ip_address} \n  Port number: 
{port_number}")
    while True:
        cs, Client_IP = ss.accept()
        client_handler = threading.Thread(target=client_handling, args=(cs, 
Client_IP))
        client_handler.start()
```

- In the following function it will print that the client is connected with its address and receives its username and print the username, then while the client is connected the server will be receiving the message from the client if it didnt receive it the client will be disconnected from the server else the message will be splited from for ex. 1.1.msg to 1 1 msg then if the first option was 1 (headlines) it will open 2 functions (get_headlines and data_extractionAndSending) if it was 2 (sources) it will open (get_sources and data_extractionAndSending).

```
def client_handling(cs, address):
    print(f"Host {address} connected")
    username = cs.recv(8192).decode()
    print(f"Client {username} has connected to the server")
    while True:
        try:
            user_message = cs.recv(8192).decode()
            if not user_message:
                break
            upper_option, lower_option,request = user_message.split(".")
            if upper_option == "1":
                get_headlines(request, lower_option, username)
                data_extractionAndSending(upper_option, lower_option, username,
cs)
            elif upper_option == "2":
                get_sources(request, lower_option, username)
                data_extractionAndSending(upper_option, lower_option, username,
cs)

        except Exception as e:
            print(f"Error handling client {username}: {e}")
            break
    cs.close()
    print(f" {address} disconnected")
```

- In the following function it will extract the information from the api and send it to the client and prints that it was sent successfully.

```
def data_extractionAndSending(upper_option, lower_option, username, cs):
    with
open(os.path.join(os.path.join(os.path.dirname(os.path.abspath(__file__)),
"database"),
                           f'A14_{username}_{upper_option}.{lower_option}.json'))
as f:
        records = json.load(f)
        cs.send(json.dumps(records).encode())
        print(f"A14_{username}_{upper_option}.{lower_option} has been delivered to
{username} successfully")
```

- The following functions are the same but one for the headline and the other is for the sources they will check the second option that has been choosen and get the data from the api then it will limit the number of records returned to 15 records.

```
def get_sources(request, lower_option, client_name):
    if lower_option == '1':
        data = api.get_sources(category=request)
    elif lower_option == '2':
        data = api.get_sources(country=request)
    elif lower_option == '3':
        data = api.get_sources(language=request)
    elif lower_option == '4':
        data = api.get_sources()
    data["sources"] = data["sources"][0:15]
    with
open(os.path.join(os.path.join(os.path.dirname(os.path.abspath(__file__)),
"database"),
                            f'A14_{client_name}_2.{lower_option}.json'), 'w') as
outfile:
        json.dump(data, outfile)

def get_headlines(request, lower_option, client_name):
    if lower_option == '1':
        data = api.get_top_headlines(q=request)
    elif lower_option == '2':
        data = api.get_top_headlines(category=request)
    elif lower_option == '3':
        data = api.get_top_headlines(country=request)
    elif lower_option == '4':
        data = api.get_top_headlines()
    data["articles"] = data["articles"][0:15]
    with
open(os.path.join(os.path.join(os.path.dirname(os.path.abspath(__file__)),
"database"),
                            f'A14_{client_name}_1.{lower_option}.json'), 'w') as
outfile:
        json.dump(data, outfile)
```

client:

- The imported packages that are used by the client are:-

```
import socket
import json
```

- The following client code will firstly checks if the current module is the main module being executed directly if it is true it will execute the mainmenu() function. The function will have the server IP address and port number and will connect to the server and prints the connection then it will ask the user to enter their username and send it to the server using the send_client() function then while the client enters a username we will print 3 options 1- Search for headline 2- Search for sources and 3- Exit the program the first one will opens headline_menu(cs) function the second will open source_menu(cs) function the third will exit the client else if the client entered something else it will print Invalid option and run the code again.

```
if __name__ == "__main__":
    #starting the client
    main_menu()
def main_menu():
    server_ip = "127.0.0.1"
    server_port = 4926
    cs = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    cs.connect((server_ip, server_port))
    print(f"Connected to server {server_ip}:{server_port}")
    username = input("Enter your username: ")
    send_client(username, cs)
    while True:
        print("####################################################\n")
        print(f" welcome {username} please choose the desired option:  \n")
        print("1- search for a specific headlines")
        print("----------------------------------------------------")
        print("2- search for a specific list of sources")
        print("----------------------------------------------------")
        print("3- Quit the program")
        print("----------------------------------------------------")
        option = int(input("Enter your option: "))
        print("####################################################\n")
        if option == 1:
            headline_menu(cs)
        elif option == 2:
            sources_menu(cs)
        elif option == 3:
            print("goodbye, you are Quitting the program...")
            cs.close()
            break
        else:
            print("Invalid option. Please chooce and option from the list above")
```

- In the following function it will print the options for the headline and will ask the user to choose one from the five options and for each option it will opens send_client() function and headline() function exept the last option will return it to the main_menu() function any other options will print an invalid option and run it again.

```
def headline_menu(cs):
    #printing headline options and asking user to pick one.
    print("1- Search for Keywords")
    print("----------------------------------------------------")
    print("2- Search by Category")
    print("----------------------------------------------------")
    print("3- Search by Country")
    print("----------------------------------------------------")
    print("4- List all New Headlines")
    print("----------------------------------------------------")
    print("5- Back to Main Menu")
    print("----------------------------------------------------")
    lower_option = int(input("Enter your option: "))
    print("##################################################\n")
    if lower_option == 1:
        request = input("Enter a pecific keword you want to search about: ")
        option = f"1.{lower_option}.{request}"
        send_client(option, cs)
        headlines(cs)
    elif lower_option == 2:
        print(["entertainment","business" , "general", "technology","sports",
"science" ,"health" ])
        request = input(f"\nEnter the desired category: ")
        if request not in cat:
            print("sorry, Invalid category choose from the list above")
            return
        option = f"1.{lower_option}.{request}"
        send_client(option, cs)
        headlines(cs)
    elif lower_option == 3:
        request = input(f"Enter country: ['eg', 'nz', 'ca', 'ae', 'sa', 'gb',
'us','au' , 'ma']")
        if request not in regions:
            print("sorry, Invalid country choose from the list above")
            return
        option = f"1.{lower_option}.{request}"
        send_client(option, cs)
        headlines(cs)
    elif lower_option == 4:
        option = f"1.{lower_option}.null"
        send_client(option, cs)
        headlines(cs)
    elif lower_option == 5:
        return
    else:
        print("Invalid option. Please chooce and option from the list above")
        return
```

- In the following function it will print the options for the sources and will ask the user to choose one from the five options and for each option it will opens send_client() function and resources() function exept the last option will return it to the main_menu() function any other options will print an invalid option and run it again.

```
def sources_menu(cs):
    print("Select the desired option: ")
    print("1- Search by category")
    print("----------------------------------------------------")
    print("2- Search by country")
    print("----------------------------------------------------")
    print("3- Search by languages")
    print("----------------------------------------------------")
    print("4- List all the available sources")
    print("----------------------------------------------------")
    print("5- Back to Main Menu")
    print("----------------------------------------------------")
    lower_option = int(input("Enter your option: "))
    print("###################################################\n")
    if lower_option == 1:
        request = input(f"Enter category: ['entertainment','business' , 'general',
'technology','sports', 'science' ,'health' ]")
        if request not in cat:
            print("sorry, Invalid category choose from the list above")
            return
        option = f"2.{lower_option}.{request}"
        send_client(option, cs)
        resources(cs)
    elif lower_option == 2:
        request = input(f"Enter country: ['eg', 'nz', 'ca', 'ae', 'sa', 'gb',
'us','au' , 'ma']")
        if request not in regions:
            print("sorry, Invalid country choose from the list above")
            return
        option = f"2.{lower_option}.{request}"
        send_client(option, cs)
        resources(cs)
    elif lower_option == 3:
        request = input(f"Enter language:['ar', 'en']")
        if request not in languages:
            print("sorry, Invalid language choose from the list above")
            return
        option = f"2.{lower_option}.{request}"
        send_client(option, cs)
        resources(cs)
    elif lower_option == 4:
        option = f"2.{lower_option}.null"
        send_client(option, cs)
        resources(cs)
    if lower_option == 5:
        return
    else:
        print("Invalid option. Please chooce and option from the list above")
        return
```

- The following function will recieve the headline data requested from the client from the server and print 15 articles as requested and ask the client to choose an article from them and print there information.

```
def headlines(cs):
    print("please wait we are processing you request:\n")
    headlines_data = cs.recv(200000).decode()
    list = json.loads(headlines_data)
    print("the reached headlines:\n")
    print("----------------------------------------------------")
    for index, item in enumerate(list['articles']):
        print(f"{index + 1}. {item['title']} - {item['description']}\n \n")
    headNo = int(input("Enter the article number that you want to its details: "))
    print("###################################################################")
    desired_headline = list['articles'][headNo - 1]
    print(f"""Article {headNo} Details:
        Source: {desired_headline['source']['name']}
        ----------------------------------------------------
        Author: {desired_headline['author']}
        ----------------------------------------------------
        Title: {desired_headline['title']}
        ----------------------------------------------------
        URL: {desired_headline['url']}
        ----------------------------------------------------
        Description: {desired_headline['description']}
        ----------------------------------------------------
        Published At: {desired_headline['publishedAt']}
        """)
```

- The following function will recieve the sources data requested from the client from the server and print 15 articles as requested and ask the client to choose an article from them and print there information.

```
def resources(cs):
    print("please wait we are processing you request:\n")
    sources_data = cs.recv(200000)
    sources_data.decode()
    list = json.loads(sources_data)
    print("the reached sources:\n")
    print("----------------------------------------------------")
    for index, item in enumerate(list['sources']):
        print(f"{index + 1}. {item['name']}")
    sourceNO = int(input("Enter the source number that you want to access its 
details:  "))
    print("###################################################################")
    desired_source = list['sources'][sourceNO - 1]
    print(f"""Source {sourceNO} Details:
        ----------------------------------------------------
        Source: {desired_source['name']}
        ----------------------------------------------------
        Country: {desired_source['country']}
        ----------------------------------------------------
        Description: {desired_source['description']}
        ----------------------------------------------------
        URL: {desired_source['url']}
        ----------------------------------------------------
        Category: {desired_source['category']}
        ----------------------------------------------------
        Language: {desired_source['language']}
        """)
```

- The following code contains the send function

```
def send_client(option, cs):
    cs.send(option.encode())
```

Additional concept:

- Grafical user interface (graphical_user_interface.py):

The imported pakages used in the GUI are:-

```
import socket
import json
import tkinter as tk
from tkinter import ttk, messagebox, simpledialog
```

The GUI in the provided code creates a news client interface. Upon launching the program, it establishes a socket connection with a server. The tkinter window appears, prompting the user to enter a username. Once the username is provided, it is sent to the server. The main menu is then displayed, offering different options for the user. Depending on the selected option, the GUI updates accordingly by clearing the window and presenting relevant labels and buttons. The user's input, captured through button clicks or dialog boxes, triggers the corresponding function to handle the choice. The option is sent to the server via the socket connection. The server processes the request and sends back the requested data, which is then displayed in the GUI. Additional features include selecting categories, countries, and languages, as well as viewing details of specific items. "Back" buttons allow the user to navigate through different menus and options. The program runs until the user chooses to quit, closing the socket connection. Overall, the GUI provides a user-friendly interface for interacting with the news client, allowing seamless searching and exploration of headlines and sources.

Acknowledgments:

- Mohamed A. Almeer
- Matt Lisivick

Conclusion:

The Multithreaded News Application/Server project efficiently provides real-time news headlines and sources using the News API. By separating the server and application scripts, it allows users to search for news based on various criteria and browse multiple sources effortlessly. This project showcases the practical use of Python and APIs, offering a convenient tool for staying informed on current events.