News Client-Server Information System

Project Description

A multithreaded News Client/Server Information System, which exchanges information about current news. It allows multiple clients to request specific news from the server, which fetches the data from a RESTful API (NewsAPI) and sends it back to the clients.

Done by:

- Group A4 - Section 1 - ITNE352, Semester 1
- Reem Abdulla Aljalahma 202206512
- Najat Khalid Almalood 202207113

Table of Contents

1.Requirements

To run this system, you need to install the following dependencies:

- Python 3.x
- NewsAPI Python package

You can install the required dependencies using the following command:

```
pip install newsapi-python
```

2.How to Run the System

Prerequisites

Before running the system, ensure you have the following installed:

1. **Python 3.x**
2. **NewsAPI Python package**
3. **Code Editor**
    - You can use any code editor to open and edit the Python scripts (e.g., VSCode)

System Setup

1. **Download the Python Scripts**
    - Download the files and place them in a folder :
        - `server.py` - The server script that manages client requests and fetches news data.
        - `client.py` - The client script that sends requests to the server and receives the news.
        - `gui.py` - The graphical user interface (GUI) script for interacting with the client.

Running the System

1. **Run the Server Script**
    - Open your terminal/command prompt
    - Run the server script by typing the following command:

      ```
      python server.py
      ```

    - This will start the server and it will listen for client requests on the specified port.

2. **Run the Client Script**
    - Open another terminal/command prompt window.
    - Run the GUI client script by typing the following command:

      ```
      python gui.py
      ```

    - This will launch the graphical user interface, allowing you to input news topics and receive results from the server.

Interacting with the System

- In the GUI window that appears, you'll be prompted to choose an option (headlines , sources or quit) if you choose quit the gui window will close if you choose the two other options the qui will dicrect you to other gui page where you will have to choose other options until you get your desiered news (e.g., "technology", "sports", etc.).
- Once you enter a topic and click the "Get News" button, the client will send the request to the server.
- The server will fetch the relevant news from the NewsAPI and send the data back to the client, which will display it in the GUI.

3.The Scripts

1. server.py

The `server.py` script is responsible for running the server side of the client-server communication. It listens for incoming client requests, processes them, interacts with a RESTful API to gather news, and sends the results back to the client. **-fisrt step is setting up the socket (dine in `handle_server()` function)** , code snipper:

```
def handle_server():
print(30 * "-")
print("The server is running")
#creating the socket
ssocket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
ssocket.bind(("127.0.0.1", 49999))
ssocket.listen(3)
```

Main functionalities:

- Handles incoming client requests , code snippet:
- Queries the news API based on the request type (e.g., headlines, category, language, country), code snippet:
- Sends the results back to the client based on the chosen title/name .
- Supports multiple clients through threading, code snippet:

```
while True:
   sock, clientID = ssocket.accept()
   clientthread = threading.Thread(target=handle_client, args=(sock, clientID))
   clientthread.start()
       try:
```

Utilized packages:

- `socket` : For establishing communication between the client and server.
- `threading` : To handle multiple clients simultaneously.
- `newsapi` : To fetch the latest news via the NewsAPI.
- `json` : to use json features
- `time` : to let the server sleep (stop for seconds)

Key functions and classes:

- **`handle_server()`** : Starts the server and listens for client connections.
- **`handle_client(client_socket)`** : Handles a client connection by processing requests.
- **`def save_to_json(client_name, option, group_id, data)`** : saving the api results in Json file
- **`def get_headlines(param))`** : Queries the NewsAPI to fetch headlines news based on the user's request.
- **`def get_sources(params)`** : Queries the NewsAPI to fetch sources news based on the user's request.

2. GUI.py

The `gui.py` script provides the graphical user interface (GUI) for the News Client-Server system. This script allows users to input their usernames, select news categories, sources, or other filters, and interact with the server to retrieve news headlines or sources. It interacts with the server through functions provided by the `client.py` script.

Main Functionalities

- **User Login**: The GUI allows users to input their username.
- **Main Menu**: Once the user logs in, the main menu appears with options to view headlines, sources, or quit.
- **Request Handling**: Based on the user's choices, the script sends requests to the server for headlines or sources filtered by categories like country, language, etc.
- **Result Display**: Displays the received news articles or sources from the server.
- **Navigation**: Users can navigate back to the main menu or quit the application.

Utilized packages:

`tkinter` : to import the gui functions

Key functions and classes:

1. `enter_user_name()`

will attach a box widget to the Gui that allows the user to input its username when user click done main menu will be called and it provide the main three button headlines, sources ad quit

2. `headlines()`

Triggered when the user clicks the **Headlines** button. It presents the user with a submenu to choose from available options like categories, countries, languages, etc.

3. `sources()`

Activated when the user selects the **Sources** button. It provides a submenu for filtering sources based on categories, countries, and languages.

4. `quit()`

When the user clicks the **Quit** button, this function terminates the connection with the server, closes the GUI, and shuts down the client socket.

function 2,3,4 will show the sub menus as button each button of sub menu will call another function. Those functions contain radio buttons with all choices that server provides .user need to choose one radio button and click send The request will be sent, including:

- `Username`
- `Request type (headline, sources).`
- `Submenu choice (category, country, language, etc.)` the request will be send using the function bellow

5. `send_request()`

After sending the request, a message will appear telling the user that the request is sent. -then a button will show, called "view received titles" clickling on it will call:

6. `view_received_titles()` function

then we will have Two cases 1. There is data from API: • If there are articles or sources in the API for the user's request, the GUI will show at least one or more (titles or names) received from the server. • The user can choose one and click `send choice` then click on `view full data` and all the (article or source) details will appear.

```
There is no data from API:
```

• If there is no data about the user's request, a message will appear to the user indicating that there is no data for their request.

7. `back_to_main_menu()`

Allows the user to return to the main menu after receiving data to quit or if they wish to make a new request

8. `quit_connection()`

When the user selects **Quit**, this function will:

- Send a message to the server to close the connection.
- Terminate the GUI and close the client socket

Summary Workflow:

1. **`enter_user_name()`** : Collects the username and displays the main menu.
2. **`Main Menu`** : Provides buttons for Headlines, Sources, and Quit.
3. **`headlines()`** or **`sources()`** : Displays submenu choices based on the selected option.
4. **`send_request()`** : Sends the user request to the server.
5. **`view_received_titles()`** : Displays the titles or sources returned by the server.
6. **`back_to_main_menu()`** : Allows the user to return to the main menu or quit.
7. **`quit_connection()`** : Closes the connection and terminates the client.
8.

3. client.py

The client script is a simple Python file that set up the client socket and contains four main functions. -set up the socket (code snippet):

```
import socket
server_add=("127.0.0.1",49999)
username_sent=False
clinet_sock=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
clinet_sock.connect(("127.0.0.1",49999))
```

This file is imported inside the GUI script as `client`, and all four functions are called using `client.<function_name>`. The functions are as follows:

Functions:

1. `send_choice(choice)`

- **Description**: This function is called within the `send_gui_choice` method in the GUI script. The `send_choice` function sends the user's chosen title or name from the list to the server and receives the full details about the chosen title.
- **Purpose**: This function is crucial for allowing users to select a news title and request more details.
- code snippet:

```
def send_choice(message):
    try:
        print(f"Sending choice to server: {message}")
        clinet_sock.send(message.encode('utf-8'))

    except Exception as e:
        print("Error in sending:", e)
```

2. `client_close()`

- **Description**: This function is used to close the client socket.
- **Purpose**: It ensures that the client properly closes the connection to the server when the user is done interacting with the application.

3. `send_username_request()`

- **Description**: This function is called within the `send_request` method in the GUI. It sends the username and request type (e.g., headlines or sources) to the server, along with the user's sub-choice (e.g., by category, country, language). The function also sends a message (e.g., general, Arabic, sport).
- **Purpose**: This function is responsible for sending the request with user input to the server. code snippet: ```python def send_username_request(username,request_type,sub_menue_choise,msg):

  try: message = "|".join([username,request_type,sub_menue_choise,msg]) clinet_sock.send(message.encode("ascii")) print(" user name and request is sent from najat client ") print(" the message is ",message) except Exception as e: print(" error at sending username and request from najat client ")

4. `recv()`

- **Description**: This function is also called within the `send_request` method. It is used to receive data from the server after sending the request. Specifically, this function retrieves the titles or sources returned by the server based on the user's request.
- **Purpose**: It is essential for the client to receive the server's response, which contains either the titles of articles or sources related to the user's query.

4.Additional Concepts

As an additional concept, As we mentioded before we have added a graphical user interface (GUI) to the system in order to help the client interact with the server more easily. The GUI is saved in the `gui.py` file, which takes the `client.py` as an import to allow easy communication between server and client. When the GUI starts, a window will appear. The user must write their username once, and then can send and receive multiple data to and from the server using the GUI window.

5.Acknowledgments

We would like to express our gratitude to the following individuals and resources that helped us throughout this project: - youtube channel: - Codemy.com, for their invaluable guidance/videos - Libraries & Tools: - NewsAPI - For providing access to current news data through their RESTful API - Python - For being the core programming language used in this project

6.Conclusion

This project shows how the client-server model works with the API. And how multithreading allows multiple users to communicate simultaneously with the server. Throughout this project, we learned the importance of ensuring harmony between client and server in order to complete the work to its best ability. Through this program, I learned that Python allows us to accomplish a lot and is easy to use, except for the fact that any indentation problem can cause a huge mess.

7.Resources

- **NewsAPI Documentation**: The primary source for news data in the project, NewsAPI provides a comprehensive API for retrieving articles, headlines, and sources.
    - Link: NewsAPI Documentation
- **GitHub Resources**:
  GitHub repositories and tutorials that helped us with version control and project collaboration.
    - Link: GitHub