Project Title

News Service System – Client/Server Project

semester

first semester:2025-2026

Group

- Course: ITNE352 – Network Programming
- Sec:2
- Group Name:GB3
- Student 1:Fatema Salem Rashed 202302468(serever side)
- Student 2: haleema khamis 202104099 (client side)

Content of Table

table:

Introduction

This project is a Python-based client-server application that provides access to online news The server communicates with NewsAPI and supports multiple client connections Users can request and view news headlines and sources using various search options.

Decsciption of the project

This initiative focuses on developing a complete client-server system for delivering news services. The server is structured to manage several client connections simultaneously through the use of socket programming methods. It retrieves current news information from NewsAPI, processes that data, and saves the resulting content in . JSON format, making it easy to access and manage. On the client end, users can engage with a user-friendly, menu-driven interface. This feature enables them to search for news articles and sources using different criteria, such as keywords, country, and category. The project demonstrates a practical application of fundamental principles including networking, API integration, file handling, and organized data exchange among distributed systems.

Requitements

To set up and execute this project locally, you need to install the necessary dependencies using the built-in pip package manager. Ensure you have an active internet connection and a valid NewsAPI key, which can be obtained from newsapi.org. Once the project files are downloaded or cloned, navigate to the project directory and use pip to install the required dependency (requests), keeping in mind that all other modules used are part of Python's standard library. For security purposes, the NewsAPI key can be configured either directly in the server code or via an environment variable. The system operates on a client-server architecture. The server is initialized first (using python server.py), enabling it to handle multiple client connections concurrently through TCP. The client application (executed with python client.py) then connects to the server, allowing users to interact via a menu-driven interface. The server manages incoming requests, communicates with the NewsAPI service to retrieve data, efficiently stores full JSON responses locally, and returns structured summaries and detailed results to the client in real-time, ensuring seamless coordination between the two components.

HOW TO

To set up and run this project locally, ensure that Python 3.10 or higher is installed on your system along with the built-in pip package manager. The project requires an active internet connection and a valid NewsAPI key, which can be obtained by creating a free developer account at newsapi.org. After downloading or cloning the project files, navigate to the project directory using a terminal or command prompt. It is recommended to create and activate a virtual environment to isolate dependencies, then install the required external library using pip install requests, as the remaining modules (socket, threading, json, and os) are part of Python's standard library. Next, configure the NewsAPI key either by directly assigning it to the API_KEY variable in the server script or by setting it as an environment variable and retrieving it within the code. Once the configuration is complete, start the server by running python server.py, which will listen for incoming client connections on the specified port. In a separate terminal window, run the client application using python client.py, enter a username when prompted, and interact with the system through the menu-based interface to request news headlines or sources. The server processes each request, retrieves data from NewsAPI, and stores the full JSON responses in the local data directory while returning summarized results and detailed information to the client in real time.

the Scripts

server

The server script implements a multi-client TCP server that accepts concurrent client connections using threads. For each connected client, the server receives a username, then repeatedly accepts menu requests (headlines by country/category/keywords or sources by country/category/language), calls the NewsAPI endpoint (top-headlines or sources), saves the full API response as a JSON file in the data/ directory, and returns a summarized list of up to 15 results and a full detailed record for a user-selected index. The server keeps running until the client sends a quit command.

Standard library: socket, threading, json, os Third-party: requests (HTTP requests to NewsAPI)

fetch_news (endpoint, params)

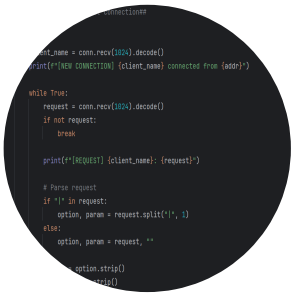
TaskedwithinvokingNewsAPI anddeliveringthe JSONreply. Itincorporatesthe APIcredentialwitheachsolicitation,managestheHTTPfailures, andyieldseithersoundJSONcontentorafaultobject.

JSON saving

Implements reliable message sending to the client by usingFor every request, the server stores the full API response in the data/ folder using a structured

handle_client

connected client (runs in its own thread). It:Reads the client username. Receives requests in the format option param. Routes the request to the correct NewsAPI endpoint.Saves the full JSON response to disk using the required naming format.Sends a summarized list()Receives an index and returns full details.



start_server()

Starts the TCP server, listens for incoming connections, and spawns a new thread per client.

Client-Side

1-Main Functionalities 2-Utilized Packages 3-Important Functions

1. main() – connecting to the server and showing menu

1. main() – connecting to the server and showing menu client = socket.socket(socket.AF_INET, socket.SOCK_STREAM) client.connect((SERVER_HOST, SERVER_PORT)) client.send(client_name.encode())

2-headlines_menu() – sending request and receiving data

request = options_map.get(choice) + '|' + param client.send(request.encode()) data = client.recv(100000).decode() data_list = json.loads(data)

3. sources_menu() – receiving details for selected source

client.send(str(idx).encode()) details_data = client.recv(100000).decode() details = json.loads(details_data)

Additional concept

Additional Concepts Used Including: | table: | |----------| |Multithreading| |API Integration| |JSON Data Handling| |Input Validation| |File Storage| |Custom Communication Protocol|

1- Multithreading Example

thread = threading.Thread(target=handle_client, args=(conn, addr)) thread.start()

2-Custom Protocol Example

request = "headlines_country|us" client.send(request.encode())

3-JSON Data Handling Example

Sending JSON

conn.send(json.dumps(send_list).encode())

Receiving JSON

details = json.loads(details_data)

API Integration Example

response = requests.get(BASE_URL + endpoint, params=params) data = response.json()

Object-Oriented Programming is the key concept implemented in our code. This paradigm focuses on organizing code around objects, combining data and behavior. It enhances scalability and maintainability of the code. achieves this through the use of methods, objects, classes, and more. By leveraging these elements, we

ensure our code remains structured and adaptable. In the following discussion, we will delve deeper into this concept with the help of provided code snippets to better illustrate its application.

Acknowledgments

we would like to express our sincere gratitude to our course instructor for the guidance and support provided throughout this project. We also thank the university for providing the necessary resources and learning environment that enabled us to .complete this work successfully

conclusion

In this project, we developed a client-server-based news retrieval system using Python. Object-oriented programming was utilized to manage client connections for the server. The server communicates with an external NewsAPI service to fetch real-time news data and provide responses to connected clients. Multithreading was incorporated to efficiently handle multiple client connections simultaneously. We relied on GitHub for script management and troubleshooting, ensuring seamless collaboration and version control. Overall, this project offered valuable experience in network programming, API integration, and troubleshooting. It enhanced our coding skills and deepened our understanding of various concepts in programming and system design.

Reference

- onecompiler (nd-b). Python Tutorials Real Python https://onecompiler.com/python/3yxmxq8xd (https://onecompiler.com/python/3yxmxq8xd)
- Stack Overflow. (nd) Newest questions https://stackoverflow.com/questions (https://stackoverflow.com/questions)
- newsapi.https://newsapi.org/