Web Server Project Documentation

Overview:

The Web Server Project is a basic implementation of server-client architecture in Python using the socket module. It demonstrates the creation of a TCP server that can handle HTTP GET requests and respond to them with appropriate HTTP responses. The corresponding HTTP client enables testing by sending requests to the server and displaying its responses.

Objective:

Develop a basic web server in Python that:

- Accepts and parses HTTP GET requests.
- Serves HTML files stored in the server directory.
- Sends a proper HTTP response including headers and file content.
- Returns a 404 error if the requested file is not found.

Technology Used:

- Language: Python 3
- Module:
    - socket from the Python standard library
    - sys to obtain server_host server_port filename from the command line

---

How It Works:

1. Server (server.py)

Below is the high-level flow of the web server:

1. Create a socket and bind it to the specified host and port.
2. Start listening for client connections using listen().

3. Accept incoming connections using accept() and process requests.
4. Parse incoming HTTP GET requests.
5. Locate the requested file:
    - If **found**, return the file content with an HTTP 200 response.
    - If **not found**, return a meaningful HTTP 404 error message.
6. Close the connection after responding.

2. Client (client.py)

Below is the high-level flow of the HTTP client:

1. Parse command-line arguments for the server address, port, and filename.
2. Create a TCP client socket and connect to the server.
3. Construct and send an HTTP GET request for the specified file.
4. Receive and display the HTTP response from the server.
5. Close the connection.

Error Handling:

1. **Server-Side**:

- Returns a 404 Not Found error for missing files or invalid paths.
- Handles unexpected errors gracefully with generic error messages.

2. **Client-Side**:

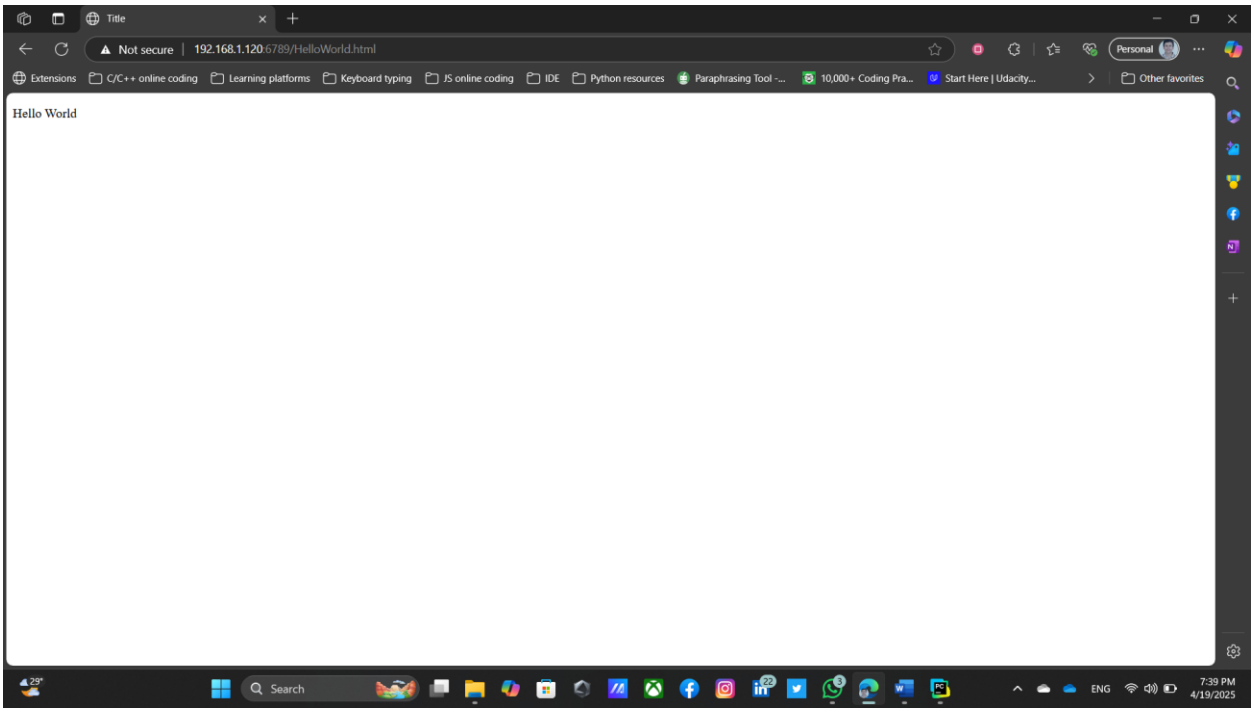
- Handles connection errors if the server is not running or unavailable.
- Validates command-line arguments to ensure proper format.

How to Run from a Browser:

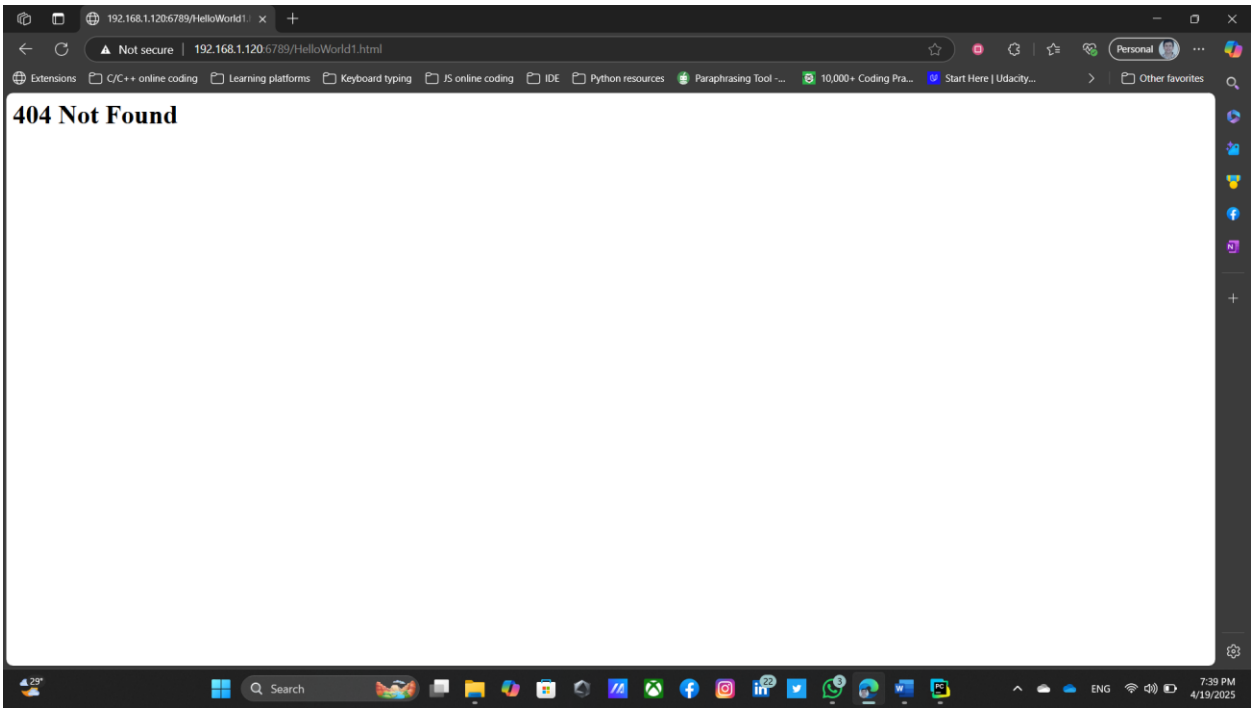
1. Save the Python server code as web_server.py.
2. Place an HTML file (e.g., HelloWorld.html) in the same directory.
3. Run the script using: **python web_server.py**
4. Open a browser on another device in the same network and visit:
   http://< Server-IP>:6789/HelloWorld.html
5. To test the 404 error, try accessing a file that does not exist.

Sample Screenshots on localhost:

Browser successfully displaying HelloWorld.html



Browser showing 404 error for a missing file



Terminal shows the request message:

```
Ready to serve...

Ready to serve...
Received request from ('127.0.0.1', 61633):
GET /Helloworld.html HTTP/1.1
Host: localhost:6789
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="136", "Microsoft Edge";v="136", "Not.A/Brand";v="99"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Safari/537.36 Edg/136.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br, zstd
Accept-Language: en-US,en;q=0.9,fr;q=0.8
Cookie: Idea-1b57baf8=89b14143-871c-4027-aa67-bdf8a3fc3d56; Pycharm-eee732e=b681009a-8f4d-4a46-9f0f-1ceebda966da
```

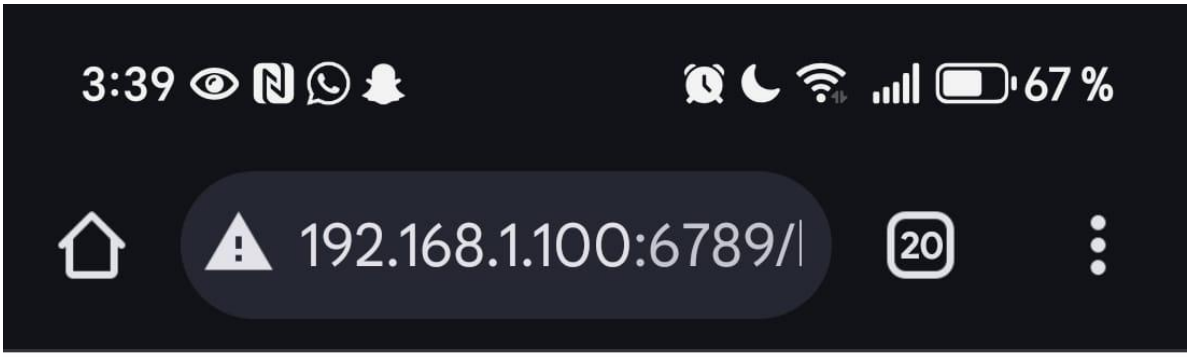
Terminal shows Error message:

```
Ready to serve...
HTTP/1.1 404 Not Found

<html><body><h1>404 Not Found</h1></body></html>
Connection with ('127.0.0.1', 49361) closed.
```

Sample Screenshots from mobile:

Browser successfully displaying HelloWorld.html



Hello World

Terminal shows the request message:

```
Received request from ('192.168.1.143', 48252):
GET /helloworld.html HTTP/1.1
Host: 192.168.1.100:6789
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Linux; Android 10; K) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/136.0.0.0 Mobile Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate
Accept-Language: de-EG,de;q=0.9,en-EG;q=0.8,en;q=0.7,ar-EG;q=0.6,ar;q=0.5,de-DE;q=0.4,en-US;q=0.3


Connection with ('192.168.1.143', 48252) closed.
```