USER MANUAL: WEBSOCKET-BASED WEB SERVER

This manual provides step-by-step instructions to set up and run the WebSocket-based web server project on Windows and Mac machines.

1. Prerequisites

Before running the project, ensure the following are installed on your system:

For Windows and Mac:

1. **Python 3.x:**
   - Download and install Python from python.org.
   - During installation, ensure you check the box to Add Python to PATH.
2. **Web Browser:**
   - Use a modern browser like Google Chrome, Mozilla Firefox, or Microsoft Edge.
3. **Text Editor or IDE:**
   - Use VS Code, Sublime Text, or any text editor of your choice.

2. Setting Up the Project

Step 1: Download the Project Files

- Download the project files (server.py, socket.html, and server.js) from your source (e.g., GitHub or shared folder).
- Save them in a folder named websocket_server.

Step 2: Install Required Libraries

1. Open a terminal (Command Prompt on Windows or Terminal on Mac).
2. Navigate to the project folder:

```
cd path/to/websocket_server
```

3. Install the websockets library:

```
pip install websockets
```

3. Running the Project

For Windows:

Step 1: Start the Server

1. Open **Command Prompt**.
2. Navigate to the project folder:

```
cd path\to\websocket_server
```

3. Run the server:

```
python main.py
```

Step 2: Access the Web Interface

1. Open a web browser (e.g., Chrome, Firefox, or Edge).
2. Go to the following URL:

```
http://localhost:8000
```

3. You should see the WebSocket Server Monitor interface.

Step 3: Monitor and Control the Server

- The interface displays:
  - Server Status: Indicates if the server is running.
  - Uptime: Shows how long the server has been running.
  - Active Connections: Displays the number of connected clients.
- Use the Stop Server button to shut down the server remotely.

For Mac:

Step 1: Start the Server

1. Open Terminal.
2. Navigate to the project folder:

```
cd path/to/websocket_server
```

3. Run the server:

```
python3 main.py
```

Step 2: Access the Web Interface

1. Open a web browser (e.g., Chrome, Firefox, or Safari).
2. Go to the following URL:

```
http://localhost:8000
```

3. You should see the WebSocket Server Monitor interface.

Step 3: Monitor and Control the Server

- The interface displays:
  - **Server Status:** Indicates if the server is running.
  - **Uptime:** Shows how long the server has been running.
  - **Active Connections:** Displays the number of connected clients.
- Use the **Stop Server** button to shut down the server remotely.

4. Testing the Project

Simulate Multiple Clients

- Open multiple browser tabs or windows and navigate to http://localhost:8000.
- Observe how the Active Connections count updates in real-time.

Stop the Server

- Click the Stop Server button to safely shut down the server.
- The server will terminate, and the browser will display a confirmation message.

---

5. Troubleshooting

Common Issues and Solutions

1. Port Already in Use

- If you see an error like Address already in use, it means another process is using port 8000 or 8765.
- Solution:
    - Stop the conflicting process or change the port in main.py (e.g., replace 8000 with 8080).

2. Python Not Recognized

- If you see python: command not found, ensure Python is added to your system's PATH.
- Solution:
    - Reinstall Python and check the Add Python to PATH option during installation.

3. WebSocket Connection Failed

- If the browser cannot connect to the WebSocket server, ensure:
    - The server is running (python main.py).
    - You are accessing http://localhost:8000 (not https).

---

6. Conclusion

This project demonstrates how to build a WebSocket-based web server using Python and JavaScript. It provides real-time monitoring of server uptime and active connections, along with a remote shutdown feature. Follow the steps in this manual to set up and run the project on Windows or Mac.

Additional Notes

- The server is restricted to localhost for security. To allow external connections, replace localhost with 0.0.0.0 in main.py.
- You can modify the project to add more features, such as logging or additional commands.

Enjoy exploring the project! 🚀

Contributed by :

M. Anas Yousaf
Abdullah Mushtaq
Abdul Rehman Khalid