Flask SocketIO

General Information & Licensing

Code Repository	https://github.com/miguelgrinberg/Flask-SocketIO/blob/main/LICENSE
License Type	MIT License
License Description	• A short and simple permissive license with conditions only requiring preservation of copyright and license notices. Licensed works, modifications, and larger works may be distributed under different terms and without source code.
License Restrictions	• Liability • Warranty
Who worked with this?	Richard, Kelly, Jieli

SocketIO Class

Purpose

- Using SocketIO along with Flasks allows us to set up a TCP connection between our client and server allowing for bidirectional communication.
- In line 33 of our code in app.py we initialise our socketio class.

Magic

- This technology sets up our original Flask server to become a Flask-SocketIO server allowing for bidirectional communication. Once it initialises the object, it listens for functions such as on() for TCP connection from users.
- The code that does the initialization of the Flask-SocketIO class is located on line 54 of the __init__.py file in Flask-SocketIO. (https://github.com/miguelgrinberg/Flask-SocketIO/blob/main/src/flask_socketio/__init__.py)
    - From there, the __init__ function on line 171 is called, and then the init_app function on line 191 is called. In the init_app function, it adds an extension called ['socketio'] to our Flask server object which then allows our Flask server to use the 'socketio' extension.
    - Now that our Flask app is initialized for socketIO connections, when we call socket.io.run() in line 390 of our app.py code, the following happens to establish a TCP connection:

@socketio.on() / on()

Purpose

- This decorator function allows us to receive information from our client in script.js and have our server handle the socket requests and return them to the user.
- This tech occurs in lines 11, 19, 61, 70, 134 in script.py and corresponds to the @socketio.on() decorators in lines 233, 243, 302, 320, 338.

Magic

- This technology does what it is supposed to do by using the socketIO Library in conjunction with our Flask app to successfully handle TCP sockets.
- The code that handles this tech is located in line 165 of server.py https://github.com/miguelgrinberg/python-socketio/blob/main/src/socketio/server.py
    - If the received event is “connect” the function _handle_connect on line 648 of server.py is called. This function will successfully set up a TCP socket for users.
    - If the received event is “disconnect” the function _handle_disconnect on line 691 will be called in server.py and it will successfully remove the TCP socket for the user.
    - If the received event is neither of the two, the function _handle_event on line 701 will be called in server.py and it will respond to the given TCP connection based on what our server is responding with. This could be sending a message, etc. The function sends the tcp connection the values we sent to the TCP socket with our event handler.

run()

Purpose

- Runs our Flask app, but uses the socketIO framework to allow for TCP socket connections. The app is being run on port 5000, but docker runs it on 5050.
- This tech is located on line 396 in our app.py file.

Magic

- This technology runs the SocketIO web server by taking in the parameters for our flask app object, hostname, and a port. This now allows us to use our Flask object to respond and create TCP sockets via the SocketIO library.
- This code is located on line 516 of __init__.py
    - It serves forever and is actively looking to establish new TCP sockets with any users that connect.