Open-Source Report: Websockets

Proof of knowing your stuff in CSE312

Guidelines

Provided below is a template you must use to write your reports for your project.

Here are some things to note when working on your report, specifically about the **General Information & Licensing** section for each technology.

- **Code Repository**: Please link the code and not the documentation. If you'd like to refer to the documentation in the **Magic** section, you're more than welcome to, but we need to see the code you're referring to as well.
- **License Type**: Three letter acronym is fine.
- **License Description**: No need for the entire license here, just what separates it from the rest.
- **License Restrictions**: What can you *not* do as a result of using this technology in your project? Some licenses prevent you from using the project for commercial use, for example.

Also, feel free to extend the cell of any section if you feel you need more room.

If there's anything we can clarify, please don't hesitate to reach out! You can reach us using the methods outlined on the course website or see us during our office hours.

Flask-SocketIO

General Information & Licensing

Code Repository	Flask-socket.io Repository
License Type	MIT License via source
License Description	• All code is free for commercial and private use • Code is free to be modified • Code is free to distribute • No citations or credits are required when publishing code that uses code from this repo

License Restrictions	• Flask-socket.io is not liable if we use their code and it breaks anything • Flask-socket.io does not provide a warranty for their code

Magic

SocketIO: Init and Handshake

Flask-SocketIO is a library that enables bi-directional communication between a Flask web application and clients through the use of WebSockets. The initial websocket request is sent from the client using the socket.io-client in javascript. Our code creates a socket that sends the websocket request to the server when the page is loaded and run with React.

Once you **Ctrl + Click(Command + Click)** the SocketIO() in our source code, you can access __init__.py of Flask-SocketIO. In the Flask application, you'll typically create an instance of the SocketIO class from the Flask-SocketIO library. This instance is responsible for managing the WebSocket connections between the server and clients. First, the __init__ function is ran where then the function init_app is called on our Flask app. Then the socketio middleware is attached to our socketio object. The _SocketIOMiddleware class is then initialized with the super init function from middleware.py of the socketio library. The super init function traces back to the WSGIApp class from middleware.py which then calls its own super init function from the WSGIApp of the middleware.py file in the engineio library. Finally, the WSGIApp of engineio calls its own __init__ function that sets the engineio 'app' and 'path' attribute to the passed app and path from socketio. Then the special function __call__ is called from the WSGIApp class. This function finally checks to see if the class's attribute 'engineio_path' is valid before returning the function handle_request from the Server class of the file server.py in the engineio library. The handle_request function handles HTTP requests from the client and in this case, the 101 switching protocols request.

SocketIO: Handle message

Flask-SocketIO provides a decorator, @socketio.on(), which you can use to define functions that handle incoming events. In our code, we use 'message' as an argument of the code. Our client emits a message over the websocket connection and then on the server, the message is received and a response is sent