Socket.IO/Flask-SocketIO Report

Socket.IO is a library that enables real-time, two-way communication between the server and the client. It uses the WebSocket protocol and consists of a javascript client library for the browser and a Node.js server. Since we are using the Flask framework to run our server, we used Flask-SocketIO which is Flask's implementation of the Socket.IO library. This keeps the same real-time, two-way communication between a client and a Python-compatible server side for fluid integration with new Flask built-in functions. Since the front end of this project is written in JavaScript, we used Socket.io-client, an extension of socket.IO, on the client side to parse socket events. Socket.io-client gives us compatibility in the front-end while providing communication with Flask-SocketIO. Both Flask-SocketIO and socket.io-client automatically finds and parses websockets when a socket is emitted, removing the extra steps of needing to read the HTTP request from the developers.

In this project, Flask-SocketIO allows the client to store and retrieve data from the server to present to the client. As the project is a game, the client would need to interact with the server to know the status of the game, the users involved, and other shared information. The server would also need to interact with the client to know when a button is pressed, when a user sent a chat message, and other interactive actions. Flask-SocketIO handles emitting and retrieving socket events that are visible on the server-side, while socket.IO-client does the same on the client-side.

Flask-SocketIO first initializes a socketIO class when the user connects with it, code being seen here:
https://github.com/miguelgrinberg/Flask-SocketIO/blob/2f1c322e687a8f7f080a081f042efdc6c8532123/flask_socketio/__init__.py#L56. When this socket server is created, there are two decorators assigned to this class that we use for in this project for retrieving and passing information, emit() and on(), seen here:
https://github.com/miguelgrinberg/Flask-SocketIO/blob/2f1c322e687a8f7f080a081f042efdc6c8532123/flask_socketio/__init__.py#L403, and here:
https://github.com/miguelgrinberg/Flask-SocketIO/blob/2f1c322e687a8f7f080a081f042efdc6c8532123/flask_socketio/__init__.py#L260 respectively. The on() decorator is used to register and listen for a socket event from the client specified in the making of the sockerIO class. The emit() decorator emits a socket event to the client specified. How these sockets validate handshakes is written here on the package arrival:
https://github.com/socketio/socket.io/blob/9fff03487c81f36ce5d4502547fa690623c10fae/client-dist/socket.io.js#L2860 and
https://github.com/socketio/socket.io/blob/9fff03487c81f36ce5d4502547fa690623c10fae/client-dist/socket.io.js#L2903. Since Flask-SocktIO is built from socket.IO, the code that allows socket-IO to process the parsed request coming in from the client is
https://github.com/socketio/socket.io/blob/9fff03487c81f36ce5d4502547fa690623c10fae/client-dist/socket.io.js#L138. The exact parsing function is seen here:
https://github.com/socketio/socket.io/blob/9fff03487c81f36ce5d4502547fa690623c10fae/client-dist/socket.io.js#L1621. This data is passed from the library of socket-IO to the parent library of

Flask-SocketIO, where it is given to the user as an argument in the on() decorator. When sending to the client, the emit() decorator gives the user given data here: https://github.com/socketio/socket.io/blob/9fff03487c81f36ce5d4502547fa690623c10fae/client-dist/socket.io.js#L2525 to be packaged for transport. This package is what is sent off to the client from the server with the user’s message.

The license attached to the usages of both Flask-SocketIO and socket.IO-client is the MIT License, granting access of these libraries to any person and allowing the distribution of any software incorporating copies of the software.