CSE312 WebSockets Report

Web Framework: Flask / Python

General Information & Licensing

Code Repository	https://github.com/miguelgrinberg/Flask-SocketIO
License Type	MIT
License Description	● Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions: ○ The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
License Restrictions	● THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Magic ★★ ° ˙ ˚ ☽ ˚ ⌒ ˚ ★ ≡ ★ ≡

Create Websocket connection

- CSE312_GroupProject-/Server/project_server.py (Line 59)
- Link: https://github.com/XuyangLi-12088/CSE312_GroupProject-/blob/master/Server/project_server.py#L59
- SocketIO() will create a Flask-SocketIO Server with support for long-polling and WebSocket transports. (We are using “eventlet” for WebSocket transports)

```
socketio = SocketIO(app, cors_allowed_origins="*")
```

    - Call Class SocketIO(object) (Line 54)
    - Link: https://github.com/miguelgrinberg/Flask-SocketIO/blob/main/src/flask_socketio/__init__.py#L54
    - Class SocketIO(object) will “create a Flask-SocketIO Server”
    - Pass “app” as a parameter: The flask application instance. If the application instance isn't known at the time this class is instantiated, then call "socketio.init_app(app)" once the application instance is available.
    - Pass “cors_allowed_origins=`*`” as a parameter: Origin or list of origins that are allowed to connect to this server. Only the same origin is allowed by default. Set this argument to '*' to allow all origins, or to ``[]`` to disable CORS handling.
        - Call __init__() function
        - Link: https://github.com/miguelgrinberg/Flask-SocketIO/blob/main/src/flask_socketio/__init__.py#L171
        - Set all parameters to default value
            - Call init_app(app, **kwargs)
            - Set app.extension[‘socketio’] = self (set to a ‘socketIO’ object class)
            - Set server_options to kwargs
            - Set up client_manager
            - Self.server = socketio.Server(**self.server_options) (line 243)
            - Link: https://github.com/miguelgrinberg/Flask-SocketIO/blob/main/src/flask_socketio/__init__.py#L243
                - Pass all the options we set up in __init__() and init_app() functions into

socketio.Server(**self.server_options)
- Link: https://github.com/miguelgrinberg/python-socketio/blob/main/src/socketio/server.py#L13
- All parameters should already set up in in __init__() and init_app() functions
- Call __init__(self, …) (line 116)
- Link: https://github.com/miguelgrinberg/python-socketio/blob/main/src/socketio/server.py#L116
- self.eio = self._engineio_server_class() (line 134)
- Link: https://github.com/miguelgrinberg/python-socketio/blob/main/src/socketio/server.py#L134
- Call _engineio_server_class() function (line 811)
- Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L811
    - _engineio_server_class() function returns an engineio.Server
    - Call Class Server(object) (line 18)
    - Link: https://github.com/miguelgrinberg/python-engineio/blob/main/src/engineio/server.py#L18
    - Server class is an Engine.IO server. This class implements a fully compliant Engine.IO web server with support for websocket and long-polling transports.
    - if transport == 'polling' or transport == upgrade_header == 'websocket' (line 396)
    - Link: https://github.com/miguelgrinberg/python-engineio/blob/main/src/engineio/server.py#L396
    - Check If the transport options is “websocket” or not
    - Call _handle_connect() function (line 541)
    - Link:

https://github.com/miguelgrinberg/python-engineio/blob/6e93a2c19655b1a2f2dc4af858452505740eac3b/src/engineio/server.py#L541

- _handle_connect() function will Handle a client connection request
  - s = socket.Socket(self, sid) (line 550) This returns a Socket class.
  - Link: https://github.com/miguelgrinberg/python-engineio/blob/6e93a2c19655b1a2f2dc4af858452505740eac3b/src/engineio/server.py#L550
  - Call class Socket(object) (line 10)
  - Socket class is an Engine.IO socket.
    - Set upgrade_protocols = ['websocket'] (line 12)
    - Link: https://github.com/Qdigital/python-engineio/blob/d6ae500a9c2a33bafdddc27a5ff30e663064a1bd/engineio/socket.py#L12
    - _upgrade_websocket() (line 144)
    - Link: https://github.com/Qdigital/python-engineio/blob/d6ae500a9c2a33bafdddc27a5ff30e663064a1bd/engineio/socket.py#L144
    - _upgrade_websocket() function will upgrade the connection from polling to websocket.
    - Call _websocket_handler() function (line 154)
    - Link:
    - https://github.com/Qdigital/python-engineio/blob/d6ae500a9c2a33bafdddc27a5ff30e663

064a1bd/engineio/socket.py#L154
    - _websocket_handler() function is an Engine.IO handler for websocket transport.
    - set self.connected = True, self.upgraded = True
- self.sockets[sid] = s (set self.sockets[sid] to a Socket class) (line 551)
- Link: https://github.com/miguelgrinberg/python-engineio/blob/6e93a2c19655b1a2f2dc4af858452505740eac3b/src/engineio/server.py#L551
- 'upgrades': self._upgrades(sid, transport) (line 555)
- Link: https://github.com/miguelgrinberg/python-engineio/blob/6e93a2c19655b1a2f2dc4af858452505740eac3b/src/engineio/server.py#L555
- Call _upgrades() function (line 600)
    - Link: https://github.com/miguelgrinberg/python-engineio/blob/6e93a2c19655b1a2f2dc4af858452505740eac3b/src/engineio/server.py#L600
    - _upgrades() function will return the list of possible upgrades for a client connection. And check if websocket transport is available.
- In transport(self, sid) function (line 314), it will return the name of the transport used by the client
- Link: https://github.com/miguelgrinberg/python-engineio/blob/6e93a2c19655

b1a2f2dc4af858452505740eac3b/src/engineio/server.py#L314

- Call _get_socket(self, sid) (line 630), which will return the socket object for a given session.
- Link: https://github.com/miguelgrinberg/python-engineio/blob/6e93a2c19655b1a2f2dc4af858452505740eac3b/src/engineio/server.py#L630
- In this case, it will return self.sockets[sid].

Parsing Frames

- CSE312_GroupProject-/Server/project_server.py (Line 59)
- Link: https://github.com/XuyangLi-12088/CSE312_GroupProject-/blob/master/Server/project_server.py#L59
- SocketIO() will create a Flask-SocketIO Server with support for long-polling and WebSocket transports. (We are using “eventlet” for WebSocket transports)
- `socketio = SocketIO(app, cors_allowed_origins="*")`
  - Call Class SocketIO(object) (Line 54)
  - Link: https://github.com/miguelgrinberg/Flask-SocketIO/blob/main/src/flask_socketio/__init__.py#L54
  - Class SocketIO(object) will “create a Flask-SocketIO Server”
  - Pass “app” as a parameter: The flask application instance. If the application instance isn't known at the time this class is instantiated, then call "socketio.init_app(app)" once the application instance is available.
  - Pass “cors_allowed_origins=`*`” as a parameter: Origin or list of origins that are allowed to connect to this server. Only the same origin is allowed by default. Set this argument to '*' to allow all origins, or to ``[]`` to disable CORS handling.
    - Call __init__() function

- Link: https://github.com/miguelgrinberg/Flask-SocketIO/blob/main/src/flask_socketio/__init__.py#L171
- Set all parameters to default value
  - Call init_app(app, **kwargs)
  - Set app.extension['socketio'] = self (set to a 'socketIO' object class)
  - Set server_options to kwargs
  - Set up client_manager
  - Self.server = socketio.Server(**self.server_options) (line 243)
  - Link: https://github.com/miguelgrinberg/Flask-SocketIO/blob/main/src/flask_socketio/__init__.py#L243
    - Pass all the options we set up in __init__() and init_app() functions into self.eio.on('connect', self._handle_eio_connect) (line 135)
    - Link: https://github.com/miguelgrinberg/python-socketio/blob/main/src/socketio/server.py#L135
      - Call _handle_eio_connect() function (line 768)
      - Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L768
      - _handle_eio_connect() will handle the Engine.IO connection event
    - self.eio.on('message', self._handle_eio_message) (line 136)
    - Link: https://github.com/miguelgrinberg/python-socketio/blob/main/src/socketio/server.py#L136
      - Call _handle_eio_message() function (line 775)
      - Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L775
      - _handle_eio_message() function will

dispatch Engine.IO message

- If packet_type is Connect Call _handle_connect() function (line 656)
- Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L656
- _handle_connect() function will handle a client connection request
  - Call connect() function (line 51)
  - Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/base_manager.py#L51
  - connect() function will register a client to a namespace
    - Call enter_room() function (line 116)
    - Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/base_manager.py#L116
    - enter_room() function will add the client into a room with the corresponding namespace
- If packet_type is Event Call _handle_event() function (line 712)
- Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L712
- _handle_event() function will handle an incoming client event
  - Call handle_event_internal() function (line 729)
  - Link:

https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L729

- Call _trigger_event() function (line 751)
- Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L751
- _trigger_event() function will Invoke an application event handler
- Call _send_packet() function (line 647)
- Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L647
- _send_packet() function will send a socket.IO packet to a client
  - self.eio.send() function will send a message to one or more connected clients.
  - Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L322
  - self.emit() function will Emit a custom event to one or more connected clients
  - Link:

https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L271
        - self.manager.emit() will send the data to all connected clients
        - Call Class BaseManger()
        - Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/base_manager.py#L9
        - Class BaseManger() will manage all connected client connections
    - self.eio.on('disconnect', self._handle_eio_disconnect) (line 137)
    - Link: https://github.com/miguelgrinberg/python-socketio/blob/main/src/socketio/server.py#L137
        - Call _handle_disconnect() function
        - Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/server.py#L702
        - Self.manger.disconnect()
        - Call disconnect() function
        - disconnect() function will register a client disconnect from a namespace
        - Link: https://github.com/miguelgrinberg/python-socketio/blob/55db7458900a179a9363294cc4fc91eb9c775f54/src/socketio/base_

manager.py#L98
- socketio.Server(**self.server_options)
- Link: https://github.com/miguelgrinberg/python-socketio/blob/main/src/socketio/server.py#L13
- All parameters should already set up in in __init__() and init_app() functions
- Call __init__(self, …) (line 116)
- Link: https://github.com/miguelgrinberg/python-socketio/blob/main/src/socketio/server.py#L116
    - self.packet_class = packet.Packet (line 124)
    - Link: https://github.com/miguelgrinberg/python-socketio/blob/main/src/socketio/server.py#L124
    - Call class Packet(object) (line 12)
    - Link: https://github.com/csulliv9/python_socketio/blob/1600aa424c334eb9328dc8ae633ada68f65c3923/socketio/packet.py#L12
    - Packet class is a Socket.IO packet, which contains data: JSON dump of data payload.
    - In _inti_() (line 27)
    - Link: https://github.com/csulliv9/python_socketio/blob/1600aa424c334eb9328dc8ae633ada68f65c3923/socketio/packet.py#L27
        - If there is an encoded_packet(), call self.decode(encoded_packet) (line 43)
        - Link: https://github.com/csulliv9/python_socketio/blob/1600aa424c334eb9328dc8ae633ada

68f65c3923/socketio/packet.py#L43

- Call decode(self, encoded_packet) function (line 76)
- Link: https://github.com/csulliv9/python_socketio/blob/1600aa424c334eb9328dc8ae633ada68f65c3923/socketio/packet.py#L76
- decode() function will decode a transmitted package. It get the “encoded_packet”, then parsing it get the packet_type, dash, attachment_count, namespace, id. Most importantly, it will json.loads(ep), get the data.