Use as many of the sections below as needed, or create more, to explain every function, method, class, or object type you used from this library/framework.

websocket.serve

Purpose

- Creates the websocket server at a given IP address/port, and sends incoming messages to a customizable callback function
- This will be used to initialize the websocket server within websocket.py of our repo (line 34)

- an instance of serve is created by calling websockets.serve.
- Serve creates a WebSocketServer, which creates, holds, and destroys WebSocketServerProtocols.
- Serve also creates a WebSocketServerProtocol, which receives, processes, and sends out data to the server
- The handler for WebSocketServerProtocol waits for a handshake to commence, holding the entire handshake process in the handshake function
- The protocol created waits for an attempt at a valid handshake request, which is determined by reading the raw bytes and attempting to convert into a HTTP request using the read_http_request function.
- The proper HTTP handshake response is validated and processed by handshake, which after reading the HTTP request builds the response using build_response and writes it back to the client through write_http_response.

websocket.broadcast

Purpose

On a pre-existing websocket, send takes a string or bytes, formats it into a websocket frame, and then sends it to the client.
Broadcast is used to send messages to a variable amount of active users across multiple channels. Regardless of how they are chosen to be delivered, all usages of broadcast (lines 26 and 27 of websocket.py in our repo) echo messages sent over the channels back to other users listening into that channel, or to other channels (like the user's notification channel).

Magic

- The send coroutine (which WebSocketServerProtocol inherits from WebSocketCommonProtocol) is called with something that can be used as a payload (string, bytes)
- The payload is given to write_frame, who in turn calls write_frame_sync
- write_frame_sync creates an instance of Frame, which through instantiation and it's own write function constructions the frame to write, and then writes to the TCP
- broadcast takes a list of websockets and data, and for each websocket in the list calls its write_frame_sync.