**Event Listener (Line 1):**

The first line of script code would make sure of the DOM Content being fully loaded. After the DOM gets loaded further instructions would get executed. This makes the execution process more effective and efficient.

document.addEventListener('DOMContentLoaded', function () {

//Code Explanation

Listens for the 'DOMContentLoaded' event, ensuring the DOM is fully loaded before executing the code.

**WebSocket Connection (Line 2):**

After DOM Content has been loaded and ensured the second line of code will establish the client side websocket connection with the server to establish a communication layer.

const socket = io.connect('http://localhost:5500');

//Code Explanation

Establishes a WebSocket connection to the server at 'http://localhost:5500' using the io.connect method.

**toggleBulb Function & Socket Event Listeners (Lines 4-30):**

This function will simply toggle the current state to the opposite one like ON to OFF or OFF to ON and on state change it’ll also update the bordercolor of the div elements for visual representation to either Red or Green. After that the changed state (newState) will get sent via socket.emit. It’ll contain the index information and the changed state (newState).

Socket.on(toggle,data) is the receiver end that will get the information of the index and changed state. On receiving the data the receiving side will change the border colour to match with the sending client.

**Point to be noted here is that the IOT device function to actually turn on or off the device will also be added here in this section.**

window.toggleBulb = function (index) {

const bulb = document.getElementById(`bulb${index}`);

const currentState = bulb.classList.contains('on');

const newState = !currentState ? 'on' : 'off';

bulb.classList.toggle('on', !currentState);

bulb.style.borderColor = currentState ? 'red' : 'green';

socket.emit('toggle', { index, state: newState });

socket.emit('broadcastToggle', { index, state: newState });

updateButtonInnerHTML();

};

socket.on('toggle', (data) => {

const bulb = document.getElementById(`bulb${data.index}`);

bulb.style.borderColor = data.state === 'on' ? 'green' : 'red';

console.log(`Received toggle event: Bulb ${data.index} is now ${data.state}`);

updateButtonInnerHTML();

});

//Algorithmic explanation

Defines a function toggleBulb in the global scope.

Toggles the state of a bulb based on its current state.

Sends 'toggle' and 'broadcastToggle' events to the server with the bulb index and new state.

Calls the updateButtonInnerHTML function to update the button's inner HTML.

Listens for 'toggle' and 'broadcastToggle' events from the server.

Updates the border colour of the corresponding bulb based on the received state.

Logs messages to the console.

**updateButtonInnerHTML (Lines 31-44):**

This function is used to dynamically update the text content of a button based on the current state of the bulbs, providing a convenient way to reflect the overall state of the IoT system.

function updateButtonInnerHTML() {

const bulbs = document.getElementsByClassName('bulb');

const button = document.querySelector('.btn');

const anyBulbOn = Array.from(bulbs).some(bulb => bulb.classList.contains('on'));

button.innerHTML = anyBulbOn ? 'OFF' : 'ON';

}

});

//Algorithmic Explanation

const bulbs = document.getElementsByClassName('bulb');:

Retrieves all elements in the document with the class 'bulb' and stores them in the bulbs variable. This assumes the bulbs have the 'bulb' class.

const button = document.querySelector('.btn');:

Selects the first element with the class 'btn' and stores it in the button variable. This assumes there is only one button with the 'btn' class.

const anyBulbOn = Array.from(bulbs).some(bulb => bulb.classList.contains('on'));:

Converts the bulbs NodeList to an array using Array.from().

Check if at least one bulb has the class 'on' using the some method.

button.innerHTML = anyBulbOn ? 'OFF' : 'ON';:

Updates the inner HTML of the selected button based on the state of the bulbs.

If any bulb is on (anyBulbOn is true), set the button text to 'OFF'; otherwise, set it to 'ON'.