**CS410 - Traffic Light System: Weekly Progress Report (Yonathan Temiess)**

**Project Role:** Backend Integration & Hardware Planning  
**Assigned GitHub Issue:** (1) Template for taking information from server to application

**Overview:**

Over the past week, I have been actively working on the core communication pipeline for our Traffic Light System, specifically handling the bridge between server-side processing and frontend application display. This is part of the larger objective to enable real-time synchronization between physical traffic light states (via Arduino) and the mobile interface (developed using Flutter).

My primary task revolved around issue (1) from the GitHub Kanban board: **"Template for taking information from server to application"**. This functionality makes sures that the state information (e.g., red/yellow/green and timer data) transmitted from the Arduino is received, processed, and then broadcasted to connected clients (applications) through a WebSocket.

**Technical Implementation:**

**1. Local Server Setup**

I began by pulling the latest server files from our group GitHub repository. These included server.js, package.json, and package-lock.json. I configured the environment locally by:

* Installing Node.js and necessary dependencies (npm install)
* Verifying and running the server (npm start) which launched a backend instance on port 3000

**2. WebSocket Integration & Testing**

The server uses WebSockets to push real-time updates to all connected frontend clients. To validate this:

* I connected to the server using two WebSocket client methods (web-based and script-based)
* I confirmed that the server broadcasts the initial traffic state ({"state": "red", "seconds": 0}) correctly
* Additionally, I resolved a critical bug in the server.js file where a variable latestState was undefined, preventing proper broadcasting. I corrected this to use the currentstate variable, restoring full functionality.

**3. Communication Confirmation**

Once connected, I received proper traffic light state messages through the WebSocket. This proved that the bridge between the Arduino input and the app was functional, even if the Arduino wasn't physically connected at the time.

This step is crucial to ensure seamless app integration later in the process.

**Physical Layout & 3D Modeling (Issue #6 & #7 Planning)**

Although not the focus of this week's task, I have proactively begun groundwork for my other assigned issues related to the **physical layout** and **3D modeling aspects** of our prototype (issues 6 and 7).

**1. Makerspace Visit**

I visited the UMass Boston Makerspace to:

* Consult with the staff about potential housing for the traffic light module
* Explore filament options for 3D printing enclosures
* Get assistance in preliminary sketches of a casing that would support Arduino and LEDs securely

**2. Design Considerations**

* I’ve started planning a modular casing, allowing us to insert or remove wires without damaging the body
* Requested access to a 3D printer slot for next week
* I’m currently sketching the prototype in TinkerCAD and Fusion 360

**3. Material Logistics**

* Contacted the Electrical Engineering lab to potentially borrow LED mounts
* Documented measurements of our Arduino Uno board and planned hole placements for wires and USB port

**Next Steps:**

* Continue preparing STL files for our first 3D printed prototype
* Finalize physical housing sketches and share with the group for feedback
* Assist in linking server output to the actual Flutter mobile UI (coordinate with mobile team)
* Begin testing with real Arduino data as soon as hardware is fully operational

**Summary:**

This week focused heavily on laying the technical foundation for real-time communication in our system. By resolving key backend errors and testing client connectivity, we now have a working server pipeline. In parallel, I have laid the foundation for the physical structure of our hardware system by initiating modeling and design conversations at the Makerspace.

**Yonathan Temiess**

Backend + Hardware Integration Lead  
CS410 - Spring 2025