CEIS114
Final Project Deliverables

Akira Suain
Introduction to Digital Devices- 11882-May 2025
06/25/25

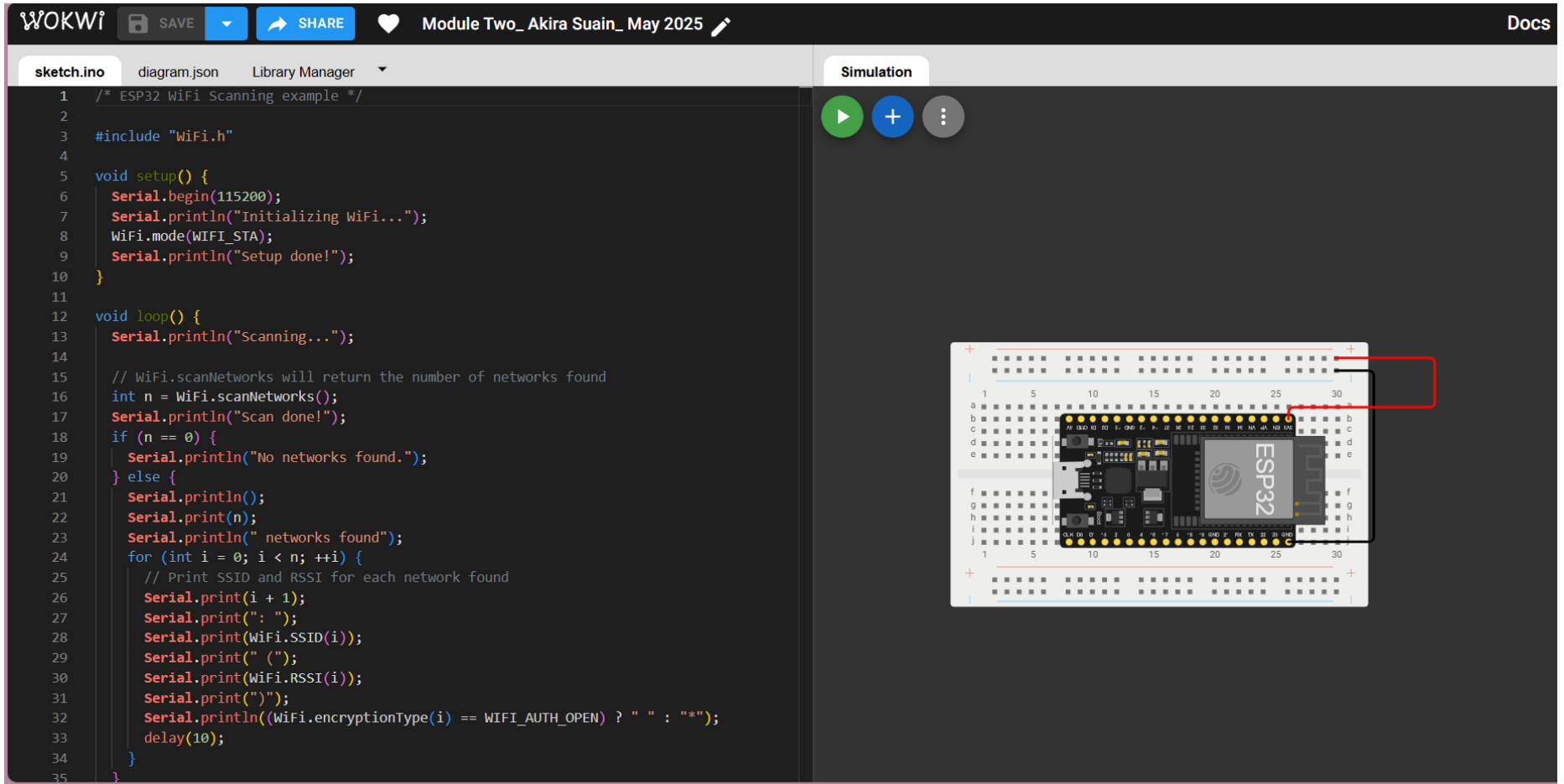
Rubric

EIS.CEIS114.W8COURSE_PROJECT.MAR24						
Criteria	**Ratings**					**Pts**
Apply basic programming to support IoT device (Smart Traffic Light Controller) threshold: 15.0 pts	**20 pts** The project has no programming concepts errors.	**15 pts** The project has 1 or more programming concepts errors.	**10 pts** The project has 2 or more programming concepts errors.	**5 pts** The project has 3 or more programming concepts errors.	**0 pts** No Programming Completed	20 pts
Classify the things or devices that make up the IoT (Smart Traffic Light Controller) threshold: 24.0 pts	**30 pts** The project was built using relevant technologies and IoT principles effectively and appropriately with no errors. All five circuit setup pictures were included: 1. Two sets of Traffic Lights 2. Push Button 3. LCD Panel 4. Buzzer 5. IOT Button or Motion Detector.	**24 pts** The project was built using relevant technologies and IoT principles effectively and appropriately with one of the following missing1. Two sets of Traffic Lights 2. Push Button 3. LCD Panel 4. Buzzer 5. IOT Button or Motion Detector.	**15 pts** The project was built using relevant technologies and IoT principles effectively and appropriately with two of the following missing1. Two sets of Traffic Lights 2. Push Button 3. LCD Panel 4. Buzzer 5. IOT Button or Motion Detector.	**7 pts** The project was built using relevant technologies and IoT principles effectively and appropriately with three of the following missing: 1. Two sets of Traffic Lights 2. Push Button 3. LCD Panel 4. Buzzer 5. IOT Button or Motion Detector.	**0 pts** No project was built.	30 pts
Design and produce an IoT system (Smart Traffic Light Controller) threshold: 15.0 pts	**20 pts** System was built, coded, and tested in a Virtual environment and/or IoT Kit with no errors.	**15 pts** System was built, coded, and tested in Virtual Environment and/or IoT Kit with one or more errors.	**10 pts** System was built, coded, and tested in Virtual Environment and/or IoT Kit with two or more errors.	**5 pts** System was built, coded, and tested in Virtual Environment and/or IoT Kit with three or more errors.	**0 pts** No Project was built.	20 pts
Explain the interconnectivity of devices in the IoT (Smart Traffic Light Controller) threshold: 10.0 pts	**15 pts** Final project submitted in the form of a professional presentation including transitions between modules projects. Other slides include title slide, introduction slide, challenges in the project slide, career skills slide, slides describing each module, and conclusion slide.	**10 pts** Final project submitted in the form of a professional presentation including transitions between modules. Missing one or two of the following: title slide, introduction slide, challenges in the project slide, career skills slide, slides describing each module, and conclusion slide.	**5 pts** Final project submitted in the form of a professional presentation including transitions between modules. Missing three to four of the following: title slide, introduction slide, challenges in the project slide, career skills slide, slides describing each module, and conclusion slide.	**3 pts** Final project submitted in the form of a professional presentation including transitions between modules. Missing all additional slides.	**0 pts** Final project presentation was not completed.	15 pts
						Total Points: 85

Project Plan for IoT Traffic Controller

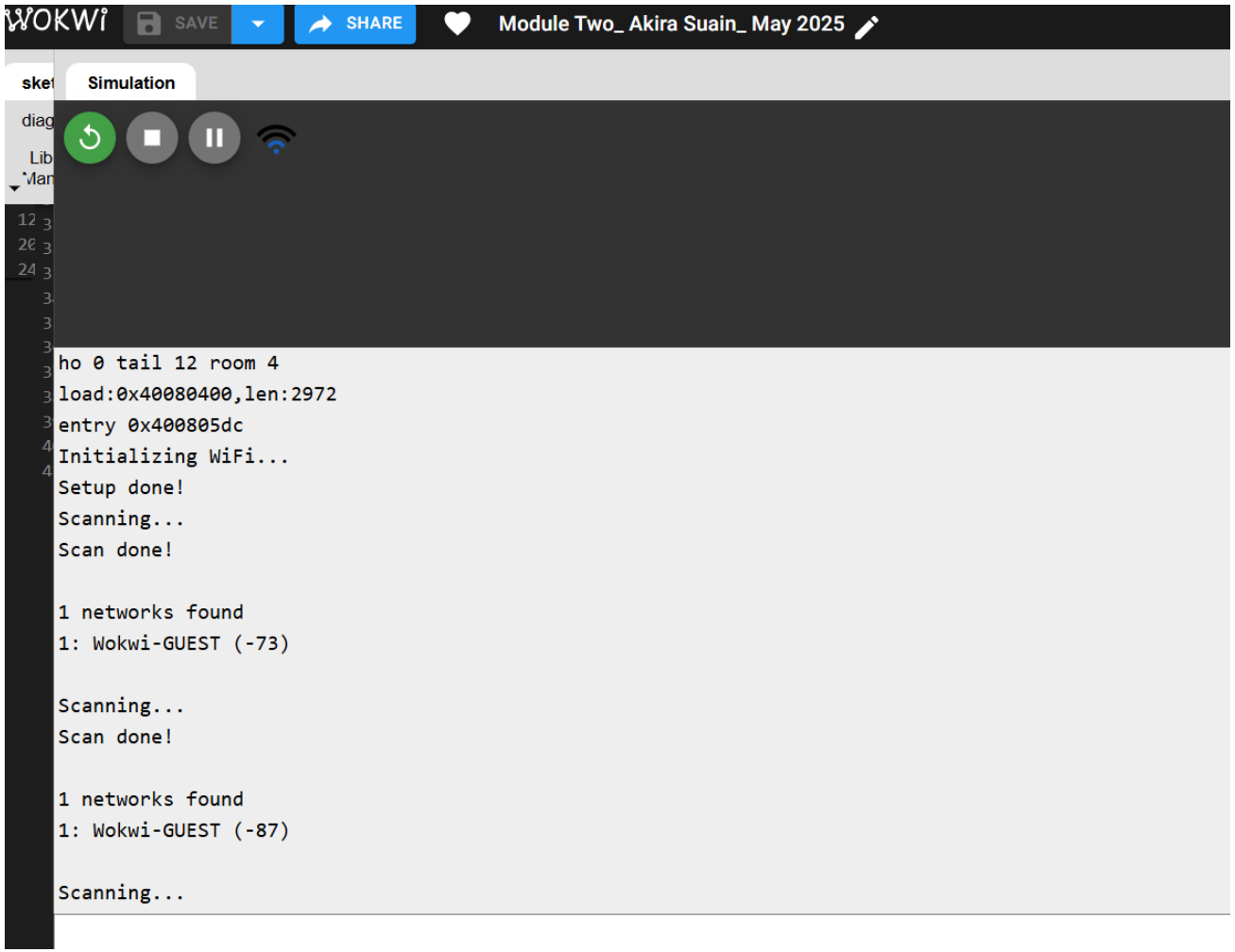
ESP32 (Screenshot)

Microcontroller mounted and powered ON



ESP32 WiFi Scan

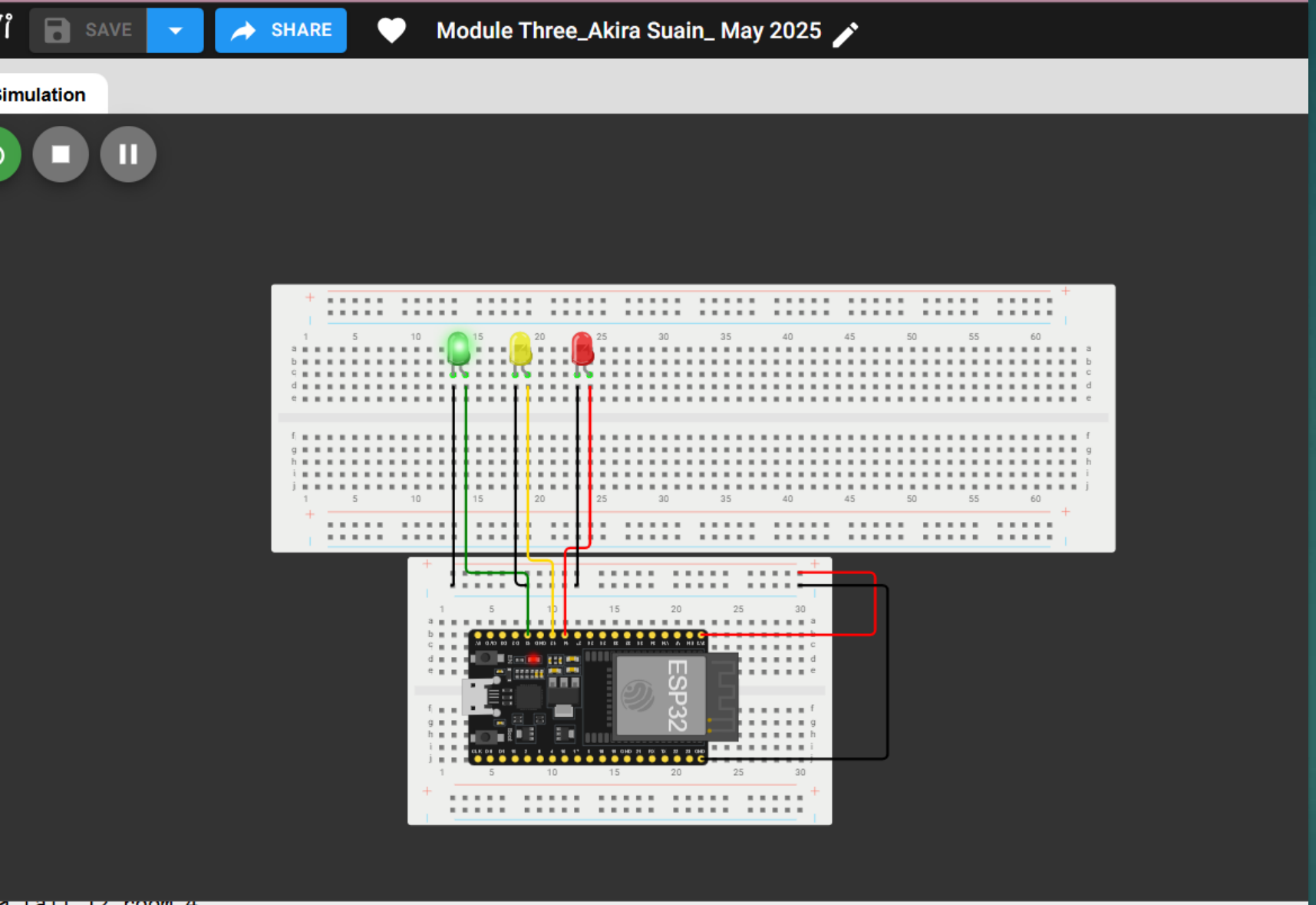
Screenshot of **Serial Monitor** showing the available networks



Creating the Traffic Controller

Rubric – Due at the end of Week 3

Activity	Requirement(s)	Points
Include a screenshot of your circuit	Picture of the breadboard with LEDs ON.	15
Screenshot of code showing your name in comment	Screenshot of code in the Code Editor showing your name in one of the comments.	15

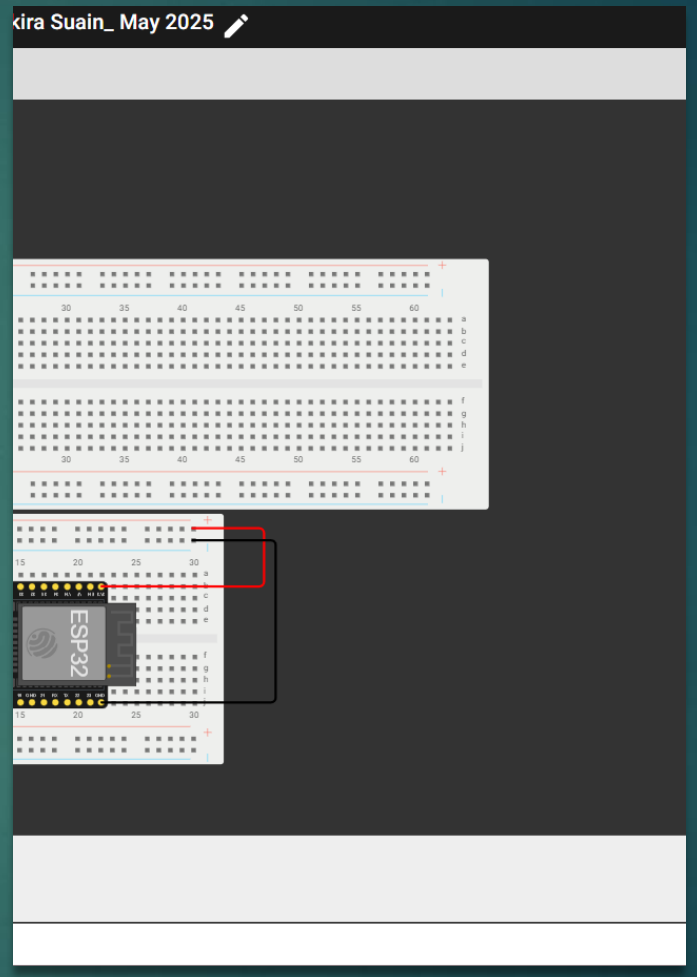


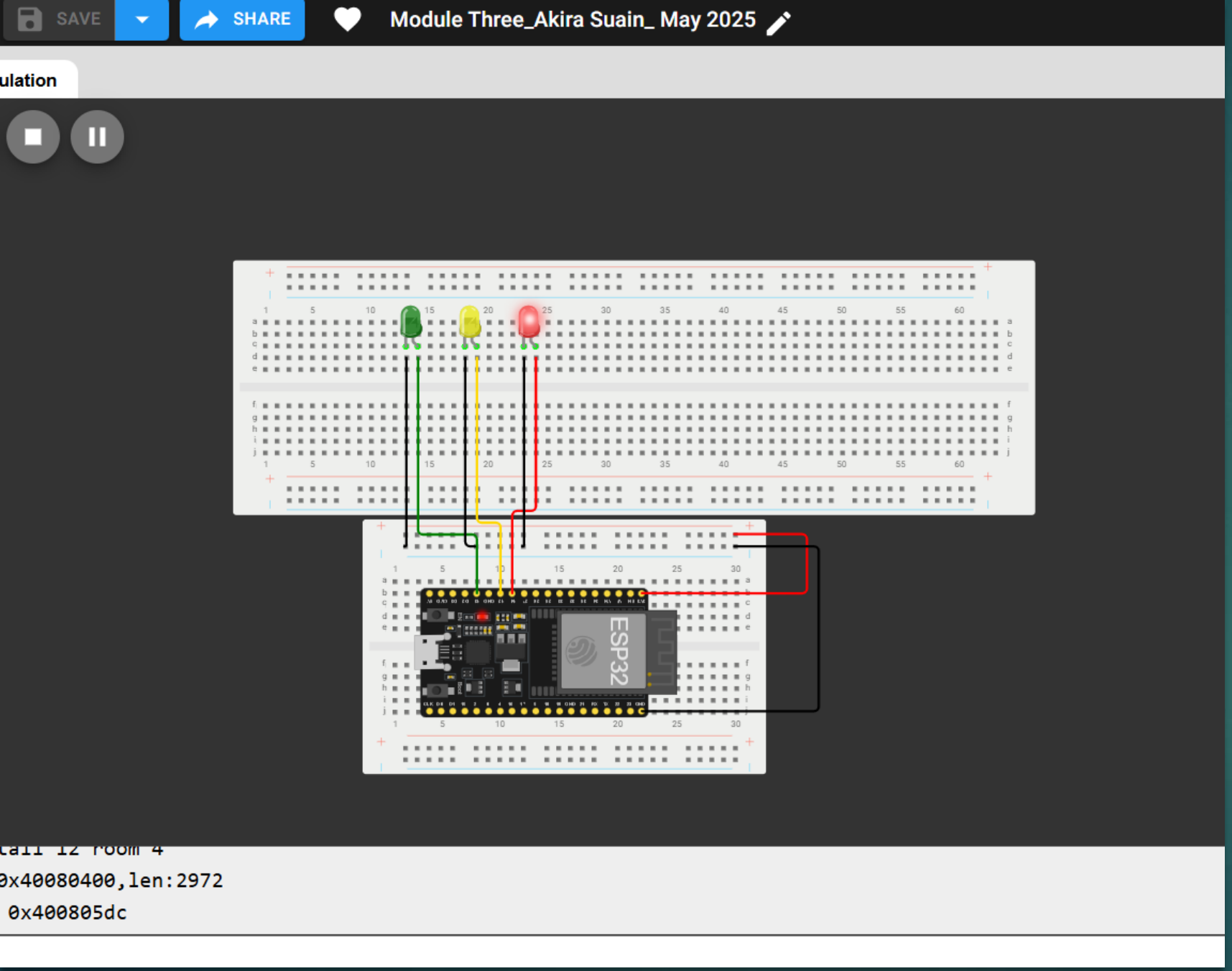
Picture of circuit with working LEDs

- ESP 32 Board
- Colored LEDs: Red, Yellow and Green
- Wires
- Breadboard

Picture of circuit with working LEDs

- ESP 32 Board
- Colored LEDs: Red, Yellow and Green
- Wires
- Breadboard



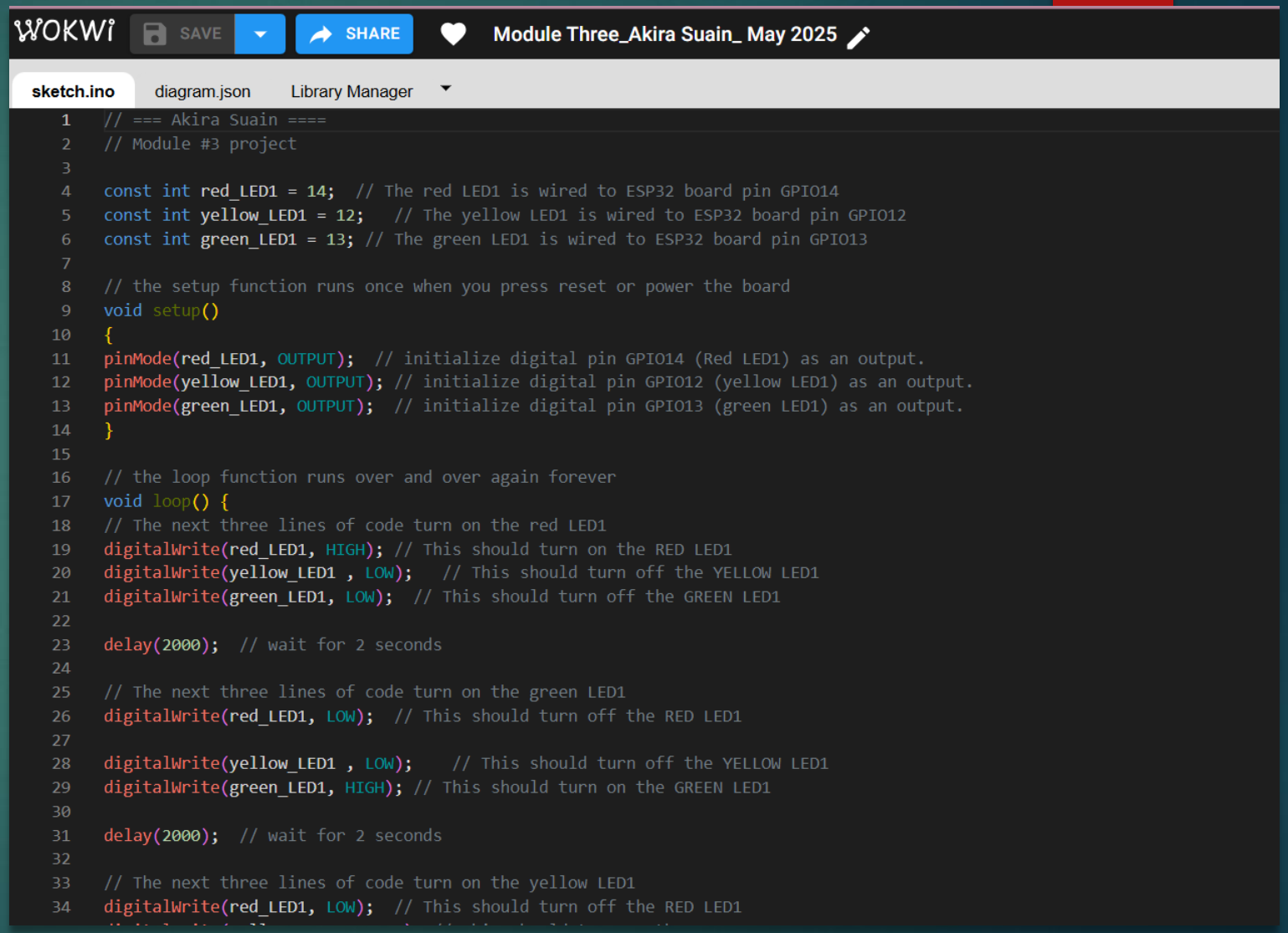


Picture of circuit with working LEDs

- ESP 32 Board
- Colored LEDs: Red, Yellow and Green
- Wires
- Breadboard

Screenshot of code in the Wokwi Code Editor showing **your name in the comment**

Screenshot of code in the Code Editor



CEIS 114 Module 4

CREATING A MULTIPLE TRAFFIC LIGHT CONTROLLER

AKIRA SUAIN

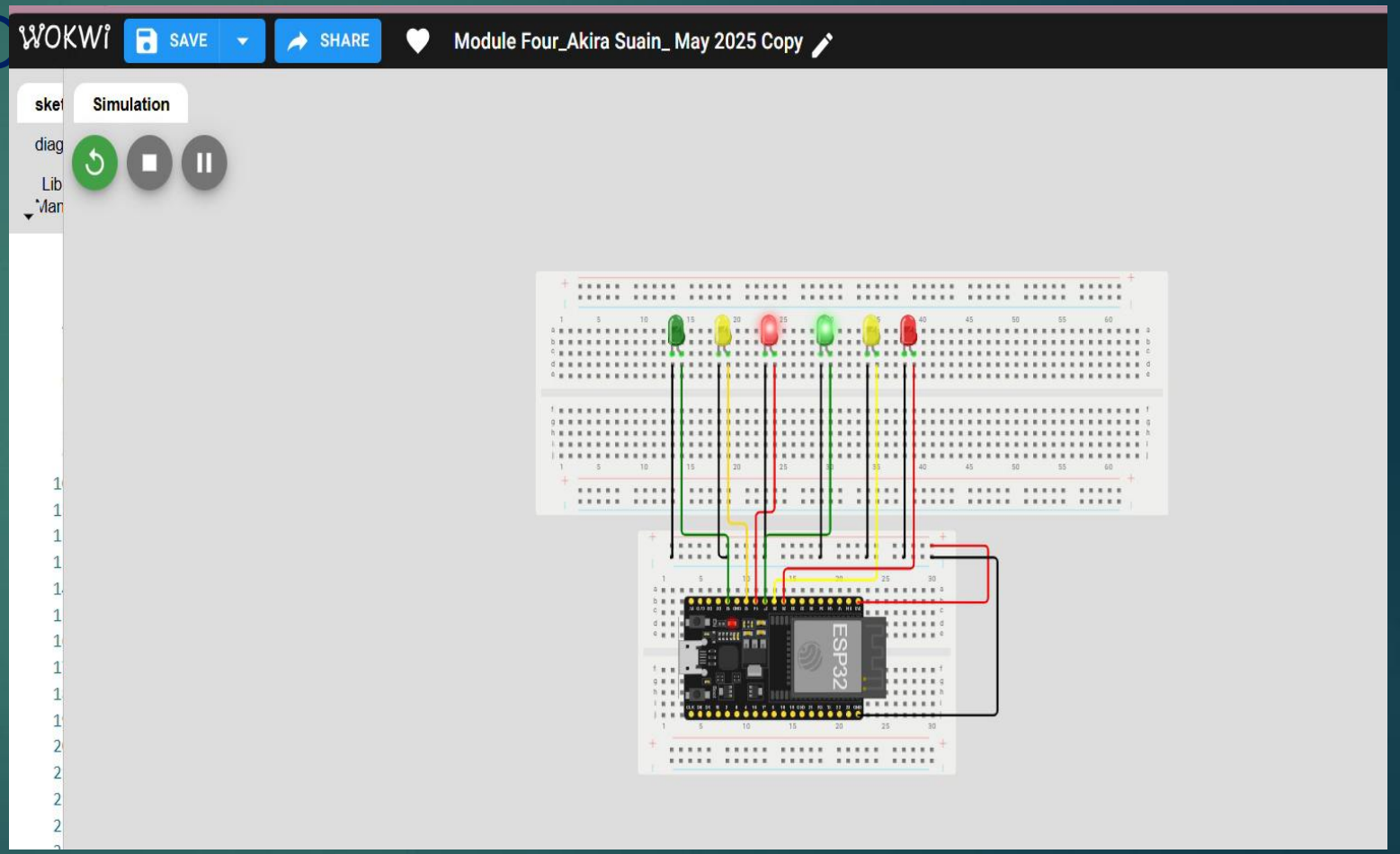
DUE DATE: JUNE 1ST 2025

Rubric – Due at the end of Week 4

Activity	Requirement(s)	Points
Include a screenshot of your circuit	Screenshot of the breadboard with LEDs ON.	15
Screenshot of code showing your name in comment	Screenshot of code in the Code Editor showing your name in one of the comments	15

Picture of circuit with working LED

- ESP 32 Board
- Colored LEDs: Red, Yellow and Green (two sets)
- Wires
- Breadboard



Screenshot of code Wokwi Code Editor showing **your name in the comment**

Screenshot of code in Wokwi

WOKWI SAVE SHARE Module Four_Akira Suain_ May 2025 Copy

sketch.ino ● diagram.json ● Library Manager

```
1  // === Akira Suain ====
2  // Module #4 project
3
4  // Define some labels
5  const int red_LED1 = 14; // The red LED1 is wired to ESP32 board pin GPIO14
6  const int yellow_LED1 =12; // The yellow LED1 is wired to ESP32 board pin GPIO12
7  const int green_LED1 = 13; // The green LED1 is wired to ESP32 board pin GPIO13
8  const int red_LED2 = 25; // The red LED2 is wired to Mega board pin GPIO25
9  const int yellow_LED2 = 26; // The yellow LED2 is wired to Mega board pin GPIO 26
10 const int green_LED2 = 27; // The green LED2 is wired to Mega board pin GPIO 27
11
12 // the setup function runs once when you press reset or power the board
13 void setup() {
14 pinMode(red_LED1, OUTPUT); // initialize digital pin GPIO14 (Red LED1) as an output.
15 pinMode(yellow_LED1, OUTPUT); // initialize digital pin GPIO12 (yellow LED1) as an output.
16 pinMode(green_LED1, OUTPUT); // initialize digital pin GPIO13 (green LED1) as an output.
17 pinMode(red_LED2, OUTPUT); // initialize digital pin GPIO25(Red LED2) as an output.
18 pinMode(yellow_LED2, OUTPUT); // initialize digital pin GPIO26 (yellow LED2) as an output.
19 pinMode(green_LED2, OUTPUT); // initialize digital pin GPIO27 (green LED2) as an output.
20 }
21
22 // the loop function runs over and over again forever
23 void loop() {
24
25 // The next three lines of code turn on the red LED1
26 digitalWrite(red_LED1, HIGH); // This should turn on the RED LED1
27 digitalWrite(yellow_LED1 , LOW); // This should turn off the YELLOW LED1
28 digitalWrite(green_LED1, LOW); // This should turn off the GREEN LED1
29
30 delay(1000); //Extended time for Red light#1 before the Green of the other side turns ON
31
32 // The next three lines of code turn on the green LED2 for 2 seconds
33 digitalWrite(red_LED2, LOW); // This should turn off the RED LED2
34 digitalWrite(yellow_LED2 , LOW); // This should turn off the YELLOW LED2
35 digitalWrite(green_LED2 , HIGH); // This should turn on the GREEN LED2
```

CEIS 114 Module 5

CREATING A MULTIPLE TRAFFIC LIGHT CONTROLLER WITH A CROSS WALK

AKIRA SUAIN

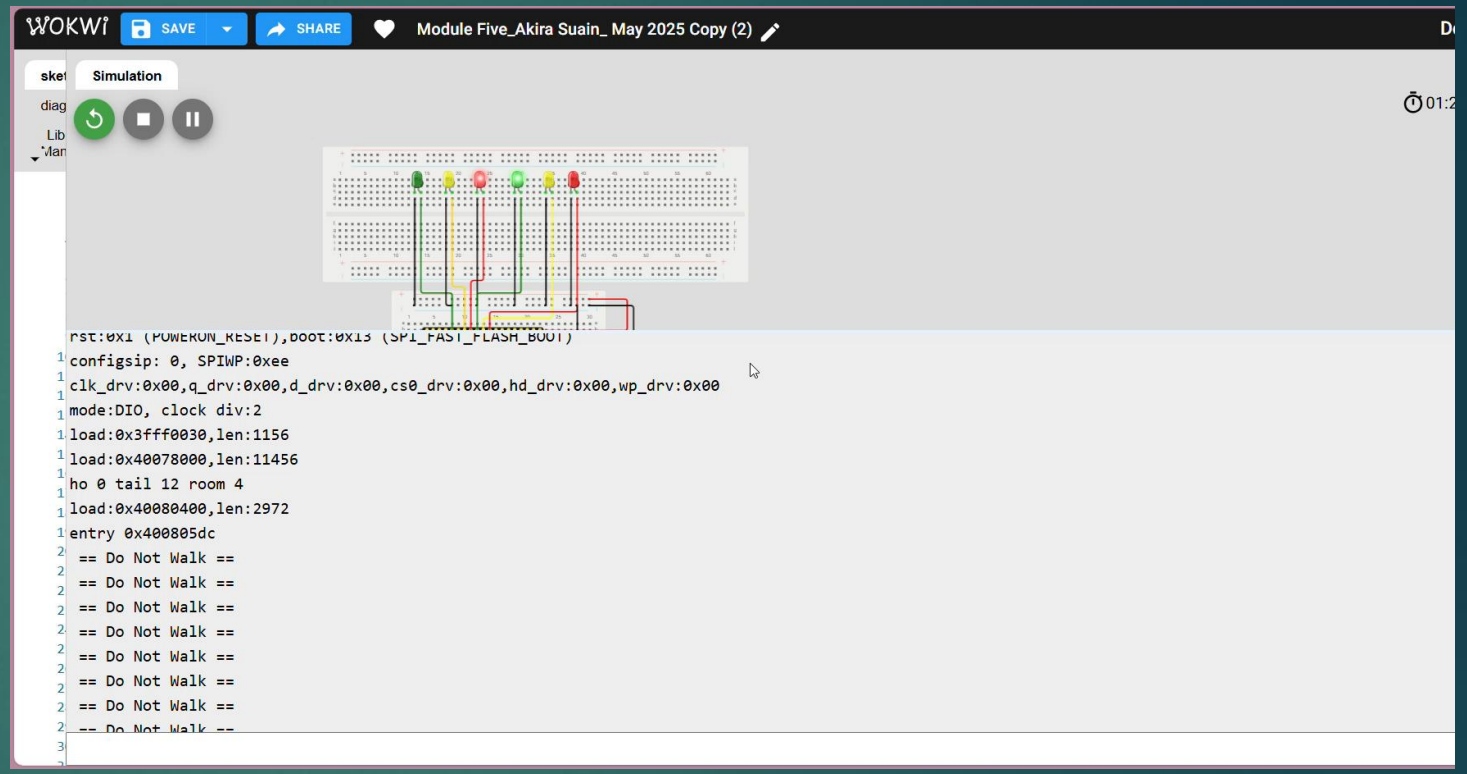
DUE: JUNE 8TH, 2025

Rubric – Due at the end of Week 5

Activity	Requirement(s)	Points
Include a picture of your circuit	Picture of the breadboard with LEDs ON	20
Screenshot of code showing your name in comment	Screenshot of code in Wokwi showing your name in one of the comments	20
Screenshot of output in Serial Monitor	Screenshot of Serial Monitor from Wokwi	20

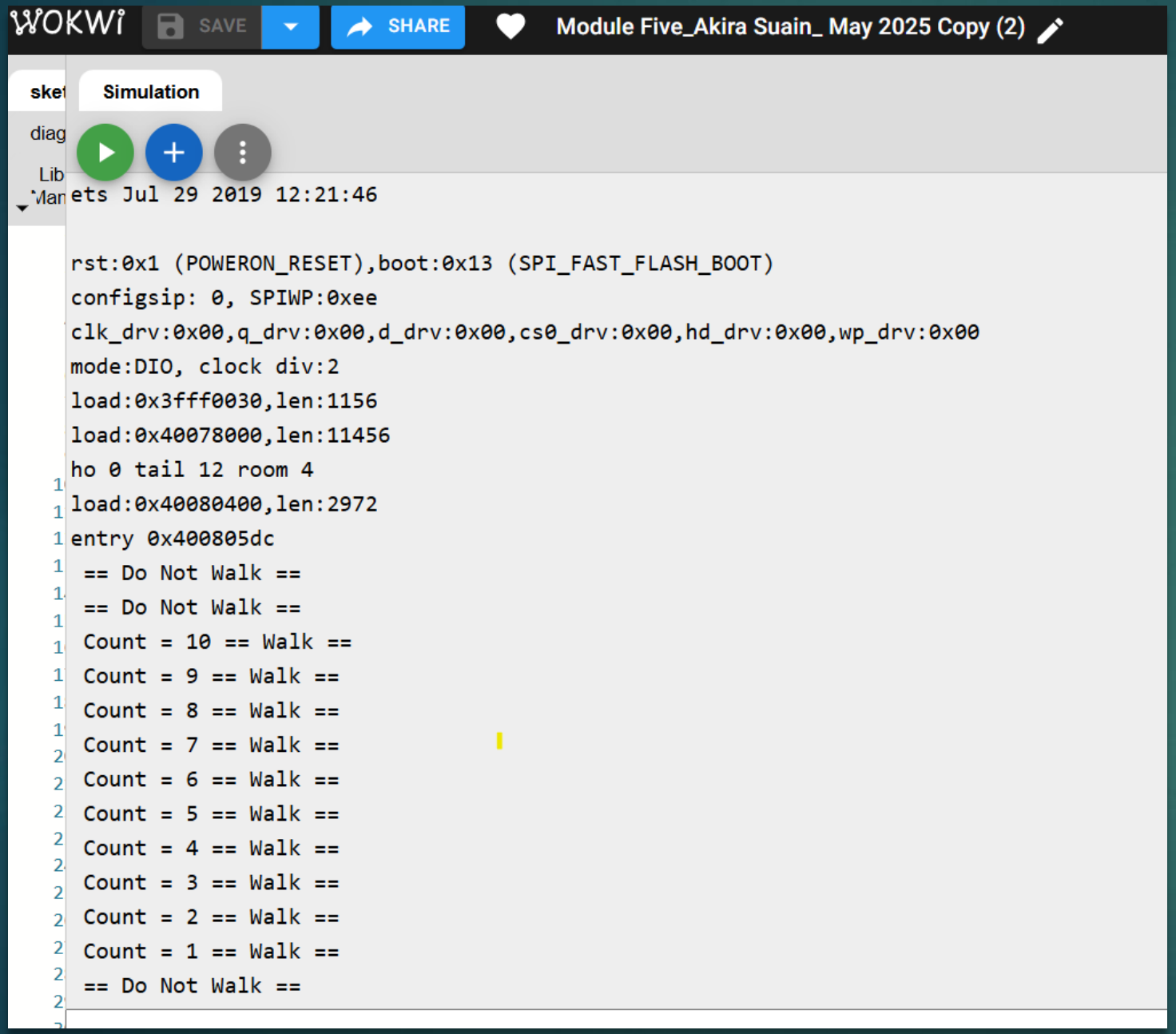
Screenshot of circuit with working LEDs

- ESP 32 Board
- Colored LEDs: Red, Yellow and Green (two sets)
- 220 Ohm Resistors (optional)
- Push Button
- Wires
- Breadboard



SCREENSHOT OF CODE IN WOKWI

Screenshot of code in Wokwi Code Editor showing **your name in the comment**



SCREENSHOT OF SERIAL MONITOR IN WOKWI

Screenshot of output in Serial Monitor

CEIS 114 Module 6

CREATING A MULTIPLE TRAFFIC LIGHT CONTROLLER WITH A CROSS WALK AND AN EMERGENCY BUZZER

AKIRA SUAIN

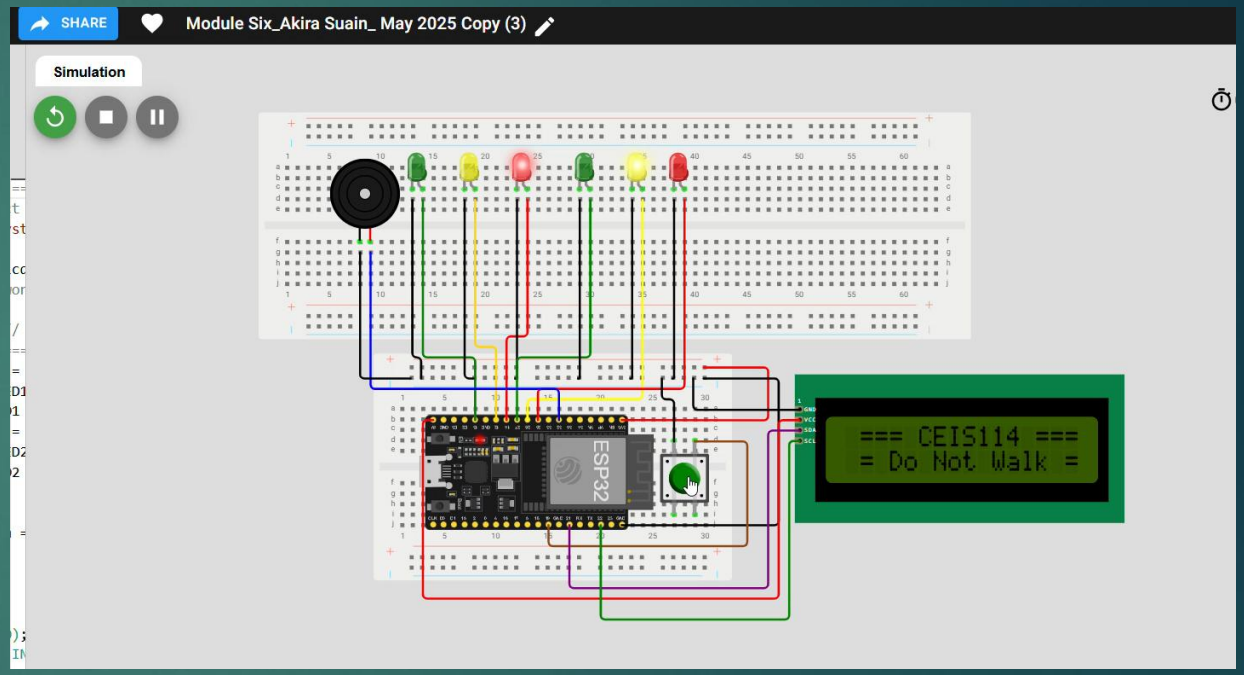
DUE JUNE 15TH 2025

Rubric – Due at the end of Week 6

Activity	Requirement(s)	Points
Include a picture of your circuit	Picture of the breadboard with LEDs ON and LCD displaying message.	20
Screenshot of code showing your name in comment	Screenshot of code showing your name in one of the comments	20
Screenshot of output in Serial Monitor	Screenshot of Serial Monitor	20

Picture of circuit with working LEDs and LCD display

- ESP 32 Board
- Colored LEDs: Red, Yellow and Green (two sets)
- 220 Ohm Resistors (optional)
- Push Button
- LCD Unit with Message Display
- Wires
- Breadboard



Screenshot of code in Code Editor

Screenshot of code in Code Editor showing **your name in the comment**

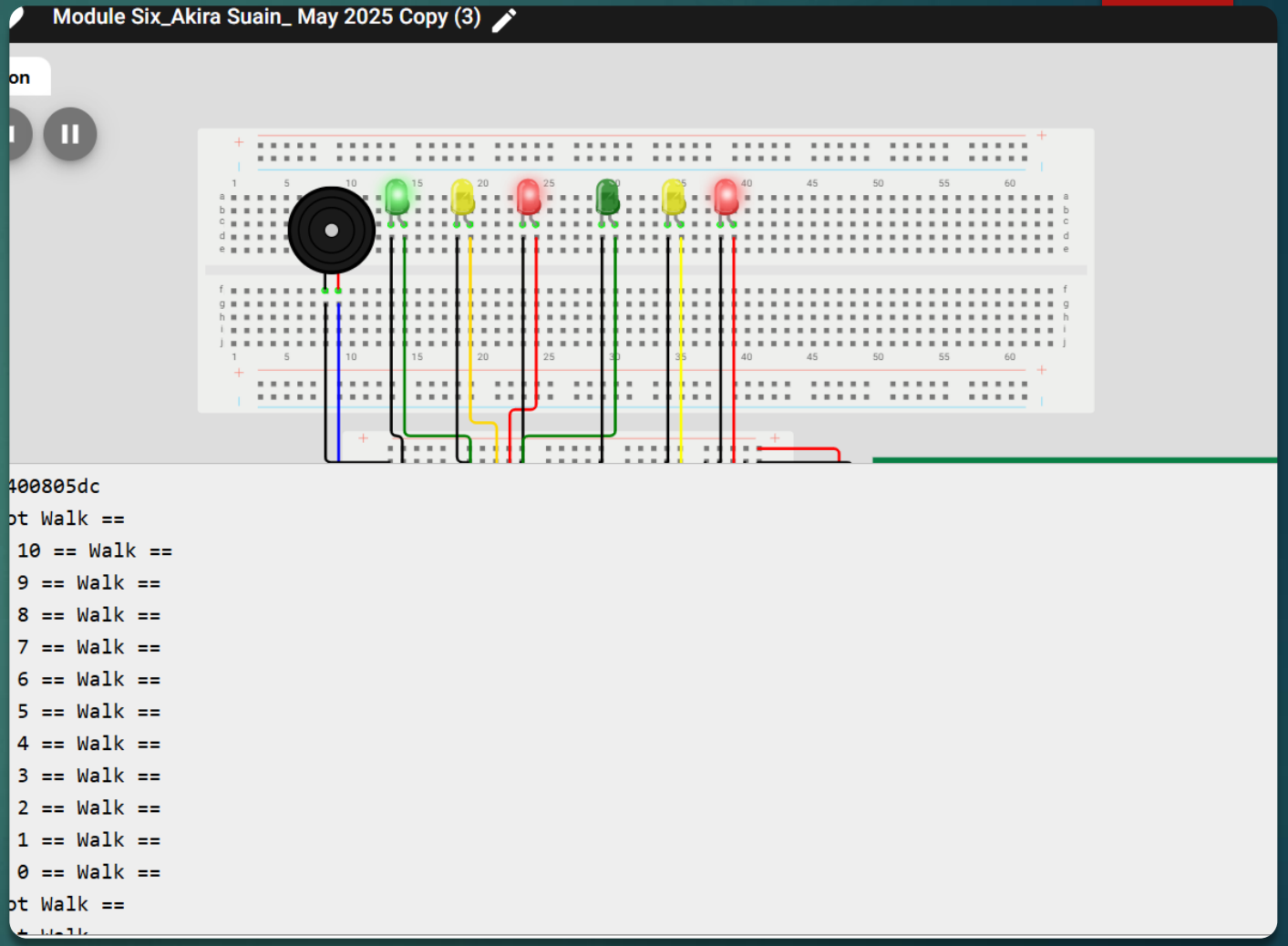
WOKWI SAVE SHARE Module Six_Akira Suain_ May 2025 Copy (3)

sketch.ino ● diagram.json ● libraries.txt ● Library Manager

```
// === Akira Suain ====
// Module #6 project #include <Wire.h> //lcd
#include <LiquidCrystal_I2C.h> //lcd

LiquidCrystal_I2C lcd(0x27,16,2); //set the LCD address to 0x3F for a 16 chars and 2-line display
// if it does not work then try 0x3F, if both addresses do not work then run the scan code below

const int bzr=32; // GPIO32 to connect the Buzzer
//=================== LCD ====================
const int red_LED1 = 14;  // The red LED1 is wired to ESP32 board pin GPIO14
const int yellow_LED1 =12;  // The yellow LED1 is wired to ESP32 board pin GPIO12
const int green_LED1 = 13; // The green LED1 is wired to ESP32 board pin GPIO13
const int red_LED2 = 25;  // The red LED2 is wired to Mega board pin GPIO25
const int yellow_LED2 = 26; // The yellow LED2 is wired to Mega board pin GPIO 26
const int green_LED2 = 27; // The green LED2 is wired to Mega board pin GPIO 27

int Xw_value;
const int Xw_button = 19; //Cross Walk button

void setup()
{

Serial.begin(115200);
pinMode(Xw_button, INPUT_PULLUP); // 0=pressed, 1 = unpressed button

lcd.init(); // initialize the lcd lcd.backlight();
lcd.setCursor(0,0); // column#4 and Row #1
lcd.print(" === CEIS114 ===");
pinMode(bzr,OUTPUT);

pinMode(red_LED1, OUTPUT); // initialize digital pin 14 (Red LED1) as an output.
pinMode(yellow_LED1, OUTPUT); // initialize digital pin12 (yellow LED1) as an output.
pinMode(green_LED1, OUTPUT);  // initialize digital pin 13 (green LED1) as an output.
```

Screenshot of Serial Monitor

Screenshot of output in Serial Monitor



CEIS 114
Week 7 Project

CREATING A MULTIPLE TRAFFIC LIGHT CONTROLLER WITH A CROSS WALK AND AN EMERGENCY BUZZER WITH SECURED IOT CONTROL VIA WEB

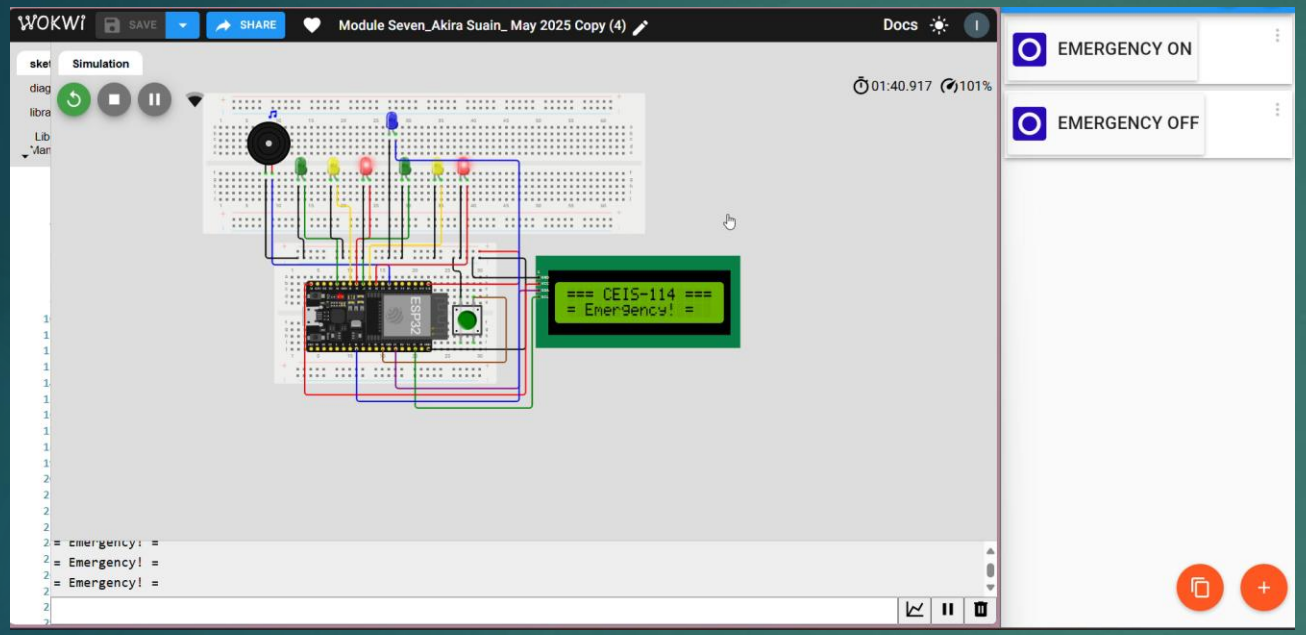
AKIRA SUAIN

DUE JUNE 22ND 2025

Rubric – Due at the end of Week 7

Activity	Requirement(s)	Points
Building/Operation	Screenshot of circuit **with working LEDs and LCD display**	20
Testing	Screenshot of **code in Code Editor**	20
Testing	Screenshot **of Serial Monitor**	20

Screenshot of circuit with working LEDs and LCD display (Building/Operation)



- ESP 32 Board
- Colored LEDs: Red, Yellow and Green (two sets)
- One Blue LED – Emergency Light
- Push Button
- LCD Unit
- Buzzer
- Wires
- Breadboard

WOKWI SAVE SHARE Module Seven_Akira Suain_ May 2025 Copy (4)

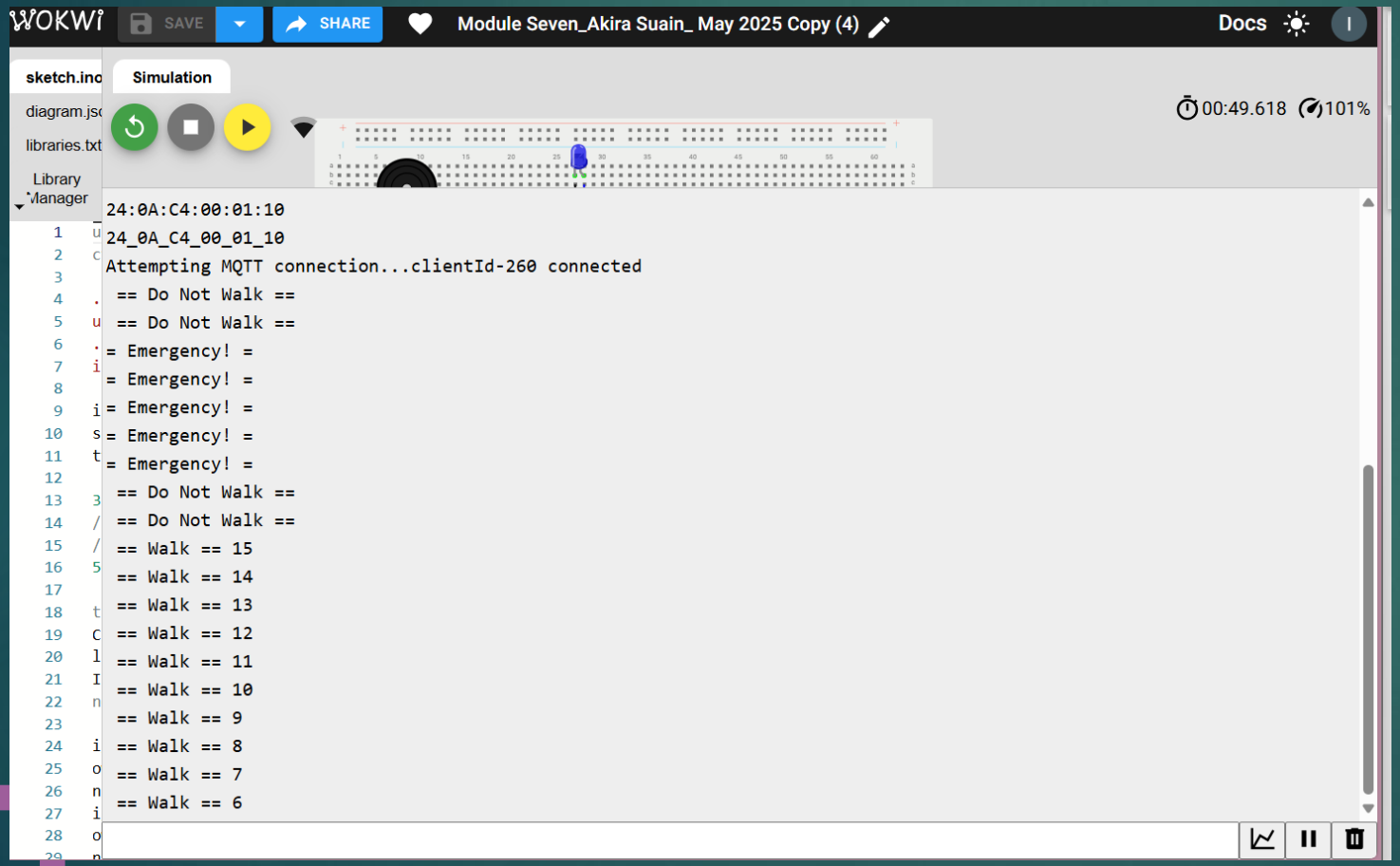
sketch.ino | diagram.json | libraries.txt | Library Manager

```
// === Akira Suain ====
// Final Project Component, Option 1

#include <WiFi.h> // WiFi header file
#include <PubSubClient.h> // MQTT publish and subscribe header file
#include <Wire.h> // I2C header file
#include <LiquidCrystal_I2C.h> // I2C lcd header file

const char* ssid = "Wokwi-GUEST"; // This is the access point to your wireless network.
const char* password = ""; // This is the password to the SSID. For the smart mini rout
const char* mqttServer = "test.mosquitto.org"; // This is the free MQTT broker we will

int port = 1883; // MQTT brokers listen to port 1883 by default
String stMac; // C string used for convenience of comparisons.
char mac[50]; // C char array used to hold the MAC address of your ESP32 microconroller
char clientId[50]; // This client ID is used to identify the user accessing the MQTT br

// For our test.mosquitto.org broker, we just generate a random user client ID
WiFiClient espClient; // instantiate the WiFi client object
PubSubClient client(espClient); // instantiate the publish subscribe client object
LiquidCrystal_I2C lcd(0x27,16,2); //set the LCD address to 0x27 for a 16 chars and 2-li
// if it does not work then try 0x3F, if both addresses do not work then run the scan c

const int redLightNorthSouth = 14; // The red LED NS is wired to ESP32 board pin GPIO 1
const int yellowLightNorthSouth = 12; // The yellow LED NS is wired to ESP32 board pin
const int greenLightNorthSouth = 13; // The green LED NS is wired to ESP32 board pin GP
const int redLightEastWest = 25; // The red LED EW is wired to ESP32 pin GPIO 25
const int yellowLightEastWest = 26; // The yellow LED EW is wired to ESP32 board pin GP
const int greenLightEastWest = 27; // The green LED EW is wired to ESP32 board pin GPIO

int crossWalkButtonState = 1 ; // Variable will store the state of the crosswalk buttor
const int crossWalkButton = 19; // Cross Walk button pin is GPIO 19
const int emergencyBlueLED = 16; // The blue LED is wired to ESP32 board pin GPIO 16
```

Screenshot of code in Code Editor (Testing)

Screenshot of code in Code Editor showing **your name in the comment**

Screenshot of Serial Monitor (Testing)



Screenshot of output in Serial Monitor

Challenges

- Website Functionality: Working in Wokwi.com came with a few technical headaches. The interface was often glitchy, especially the mouse alignment. When dragging components like LEDs onto the screen, the delay made it hard to place them precisely, leading to a lot of frustrating trial and error.
- Wiring Overload: Although the individual wiring steps weren't too difficult, the sheer number of connections became overwhelming by the end. The circuit was so cluttered that I ended up having to rebuild large portions of it just to troubleshoot and clean up the layout.
- Coding from Scratch: Instead of copying and pasting code, I challenged myself to write everything from the ground up. It took more time, but it taught me how to properly debug—reading error codes and tracing them to the right line, instead of giving up and restarting the whole script.
- Module Five Button Bug: The module five button had a five-second delay that wasn't explained in the documentation or videos. I kept redoing it, thinking something was broken, until I realized it was a built-in delay. That experience taught me to be more observant about how modules behave beyond what's explicitly taught.

Career Skills

Problem Solving & Debugging: I learned to approach errors methodically, especially while coding from scratch. This helped me build confidence in debugging and taught me how to read error messages to fix issues efficiently, a valuable skill in any tech role.

Attention to Detail: From aligning glitchy components in Wokwi to untangling messy wiring, I had to stay focused and precise. That level of attention will be useful in both hardware design and cybersecurity work, where small mistakes can have big consequences.

Resilience & Adaptability: When things didn't work as expected—like the unexpected delay in the module five button—I kept experimenting until I figured it out. This built my patience and adaptability, which are key in fast-paced IT environments.

Independent Learning & Self-Motivation: Choosing to write my own code rather than copy it showed me how much I enjoy solving problems on my own. That kind of initiative is something I hope to bring to internships and future job roles

Conclusion

Building a smart traffic control system in Wokwi pushed me beyond just connecting wires and uploading code—it challenged my patience, sharpened my problem-solving skills, and reminded me that progress often comes through trial and error.

Despite the technical glitches, confusing wiring, and coding setbacks, I stuck with it. I walked away with more than just a functioning project—I gained real experience in how to think like a developer and troubleshoot like a professional.

This project gave me a glimpse into what it's like to build and improve complex systems, which ties perfectly into the career I'm pursuing in cybersecurity and IT. I'm proud of how far I've come and excited for the challenges I'll face next.

Wix: Blog | Cybersercuity Projec