CEIS114 COURSE PROJECT

TRAFFIC CONTROLLER

PRESENTED BY: MARYAM ZISHAN

Introduction

First create and develop a two-way traffic controller with an option for pedestrian crossing as well as emergency motion detection sensors for a safer securer traffic experience. In order to create a smart IoT device a successful integration of hardware, software and networks is very important..

Secondly, the system could be controlled via a web browser remotely or through a non-internet connection using smart motion sensors that can slow the traffic flow every time a motion is detected on a major street..

Finally, in order to simulate a successful two-way traffic controller system a ESP32, microcontroller, web and a smart sensor will be utilized.

PROJECT PREPARATION

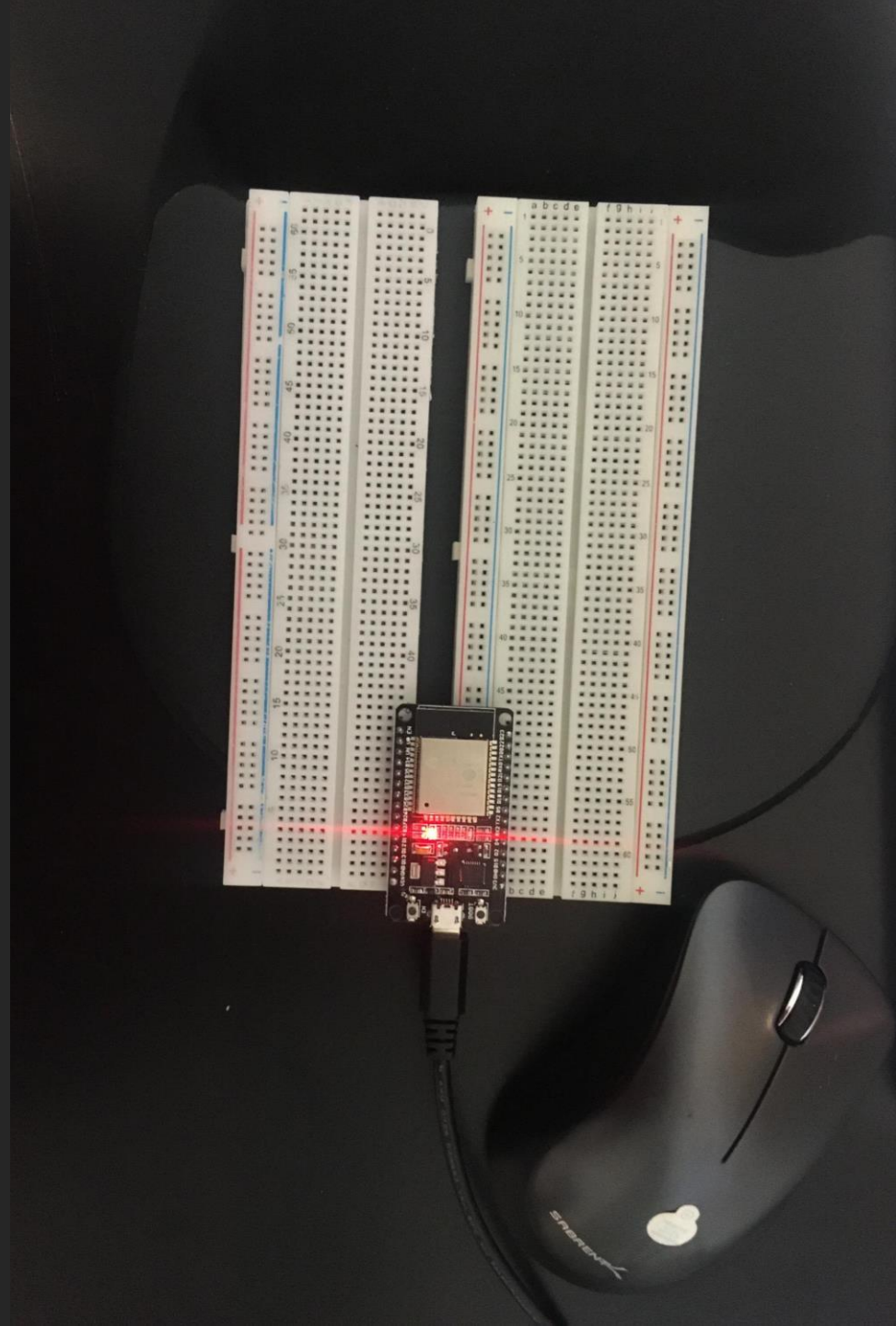
Inventory (Picture)

- ESP 32 Board
- Colored LEDs: Red, Yellow, Green, and Blu
- 220 Ohm Resistors (optional)
- Wires
- Breadboard(s)
- LCD Unit with I2C Adapter
- Active Buzzer
- Mini Router
- Push Button(s)
- PIR Motion Sensor

ESP32 (Picture)

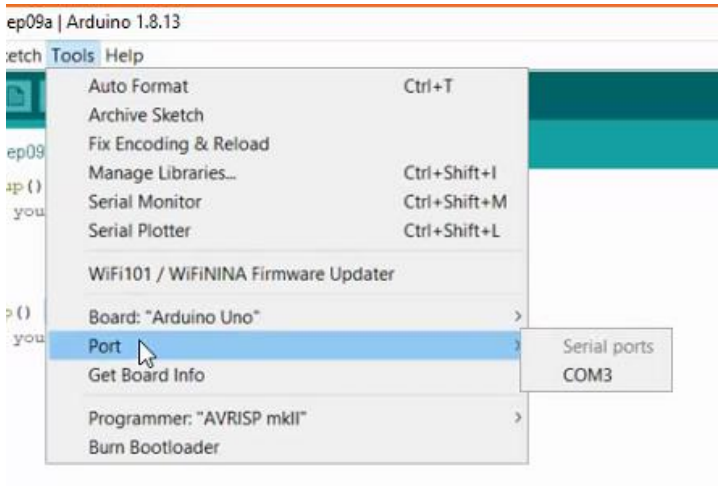
ESP32 (PICTURE)

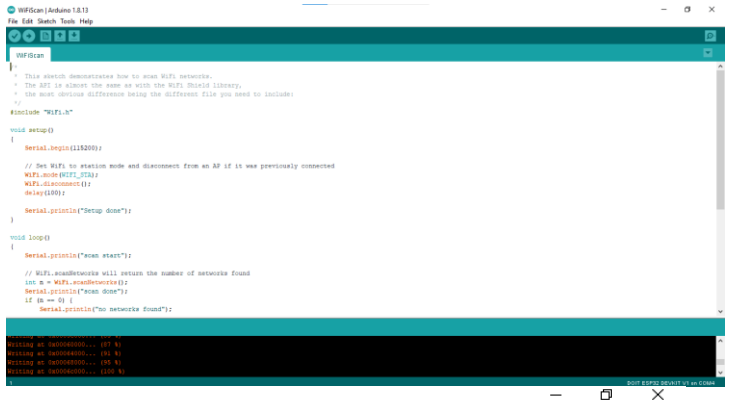
Microcontroller mounted and powered ON



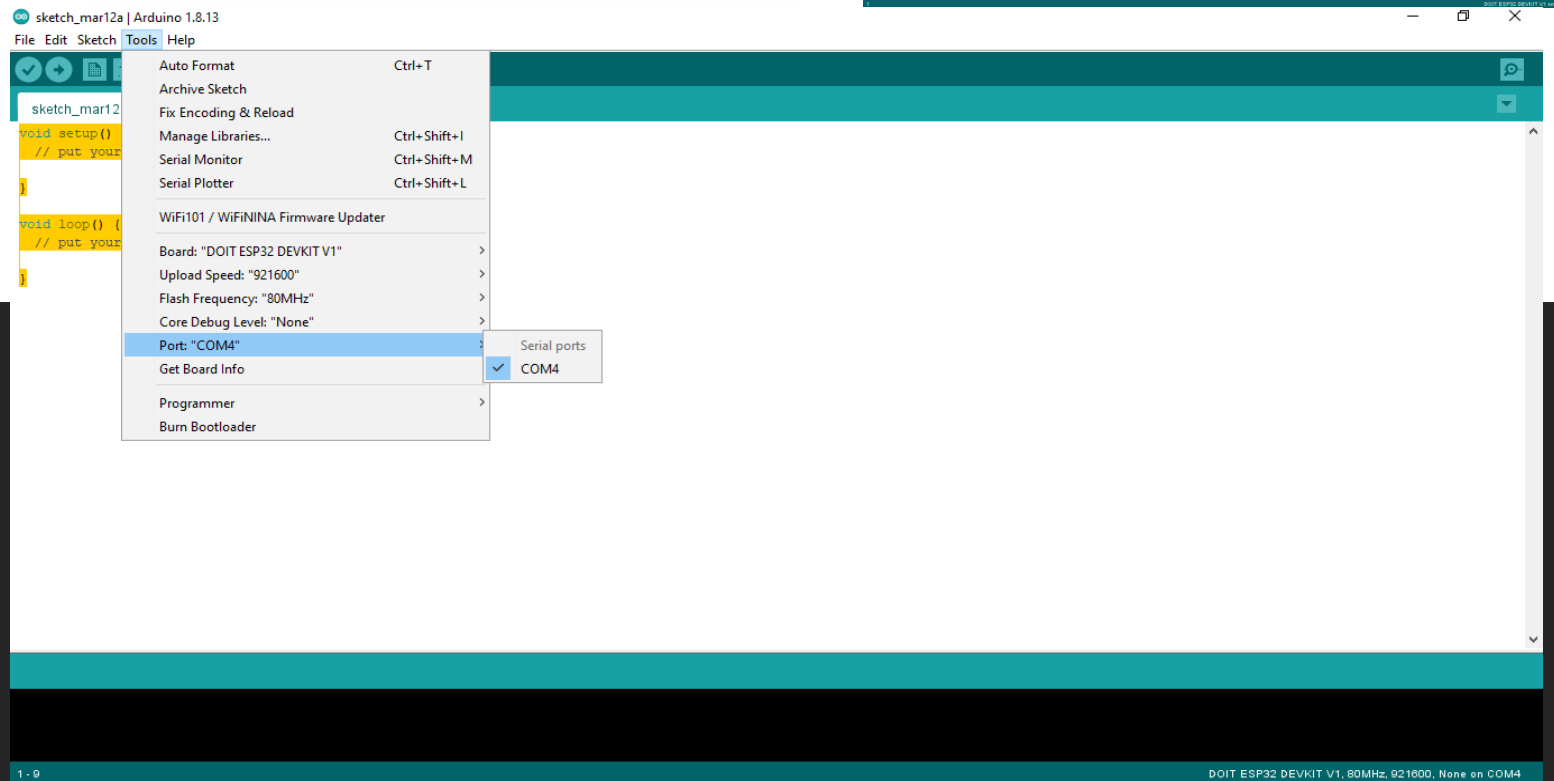
Screenshot of Arduino IDE with **Port** selected from Tools menu.

Installation of Arduino IDE



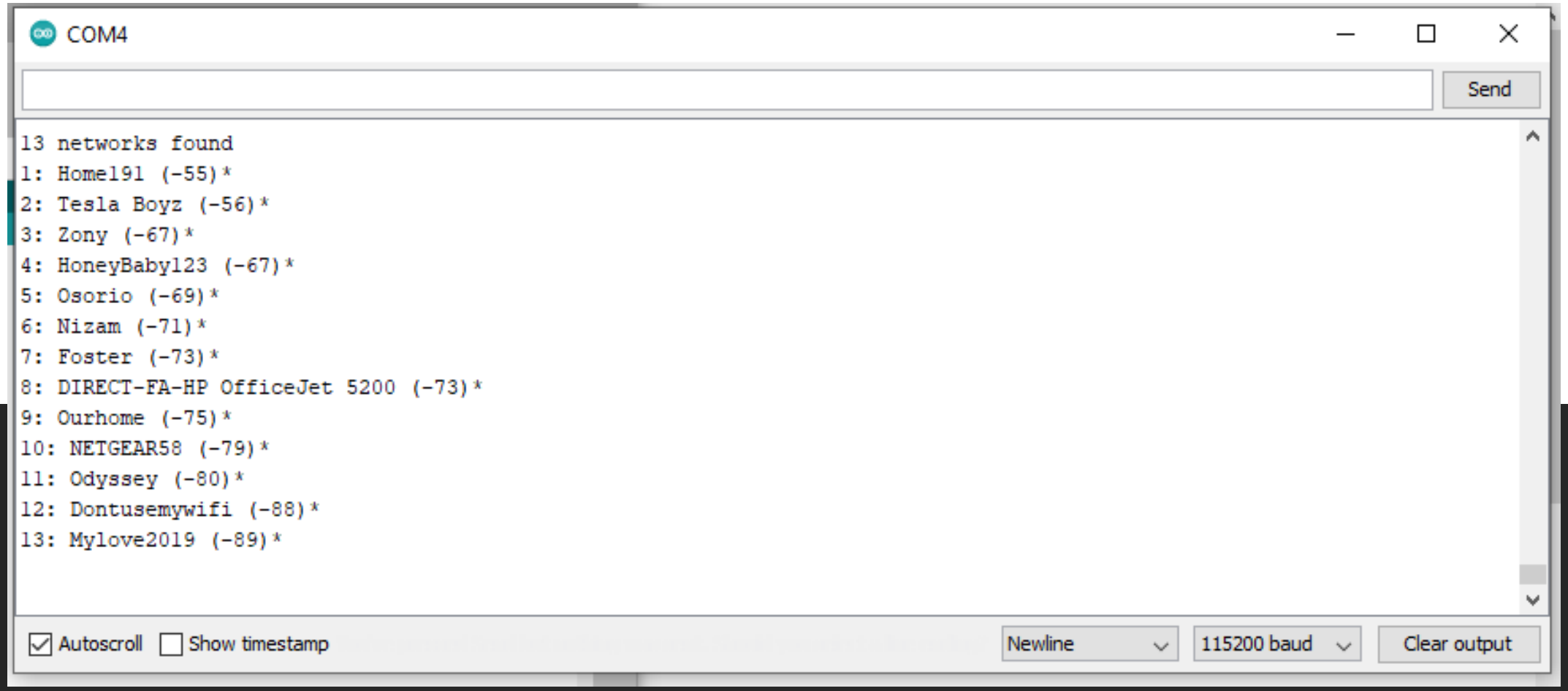


Example

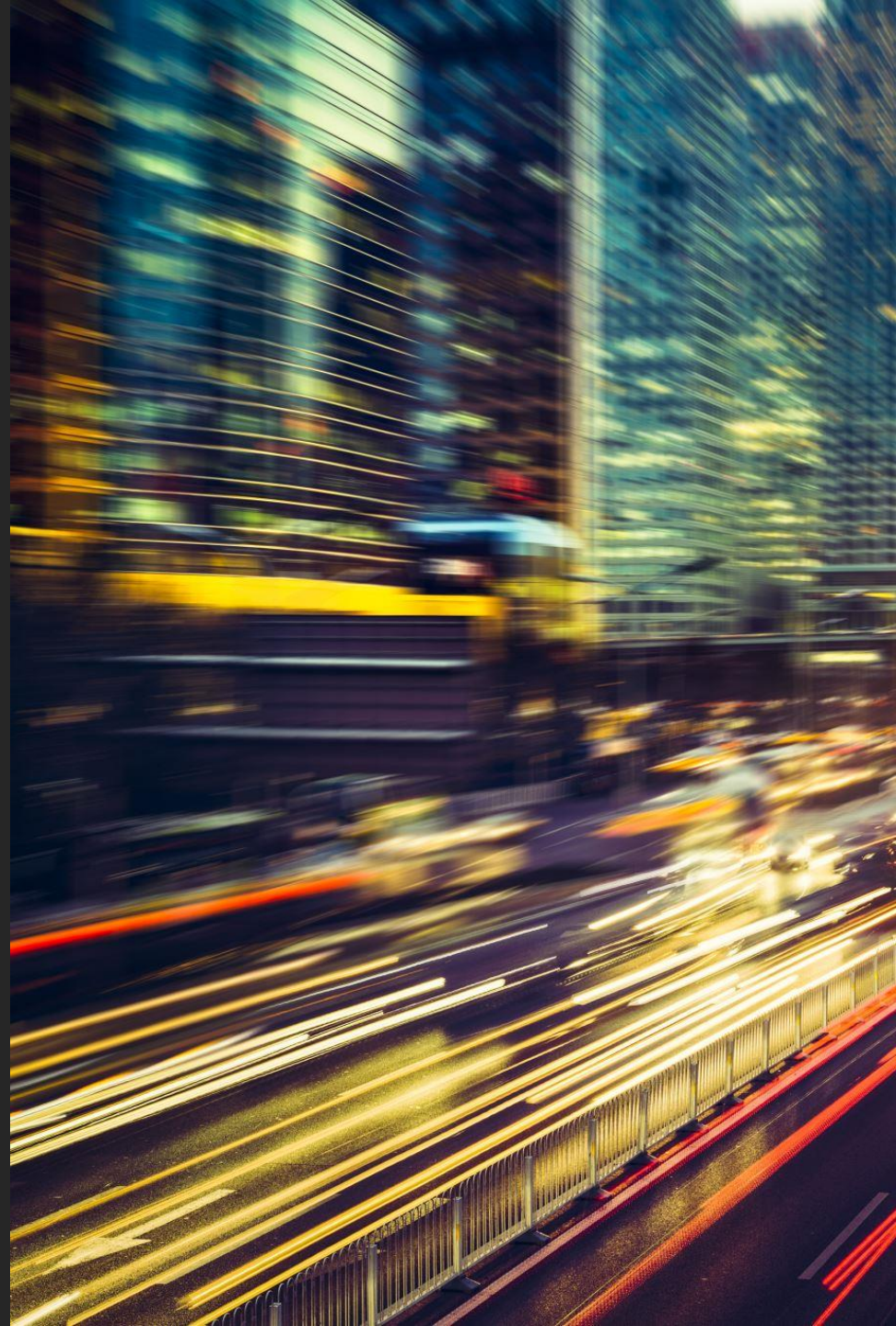


ESP32 WiFi Scan

Screenshot of **Serial Monitor** in Arduino IDE showing the available networks

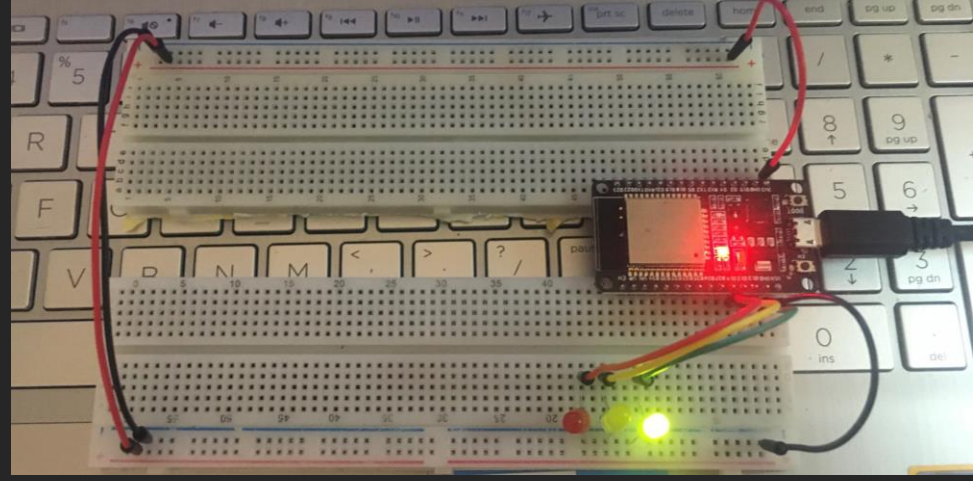


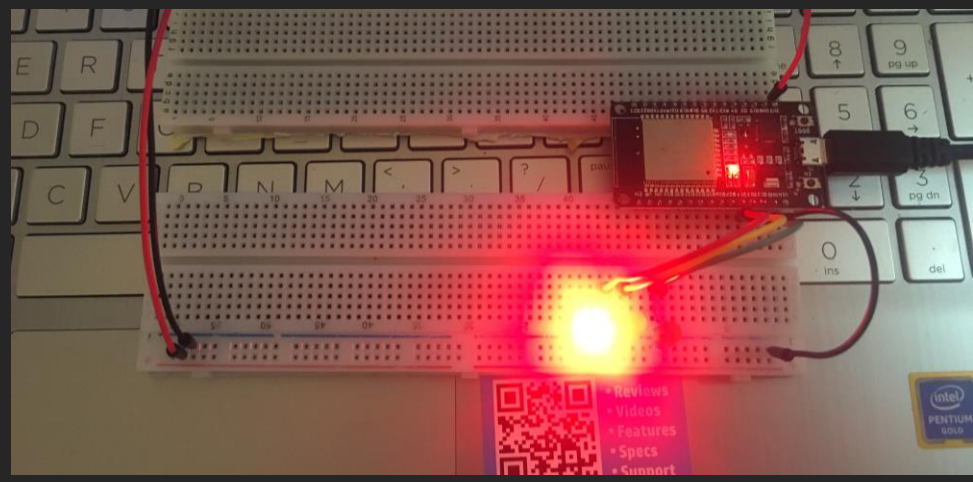
CREATING A BASIC TRAFFIC CONTROLLER



Picture of circuit with working LEDs

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





Screenshot of code in Arduino IDE

Screenshot of code in Arduino IDE showing **your name in the comment**

CEIS114_M3 | Arduino 1.8.13

File Edit Sketch Tools Help

CEIS114_M3

```
// === Maryam Zishan ====
// Module #3 project

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

// the setup function runs once when you press reset or power the board
void setup() {
 pinMode(red_LED1, OUTPUT);     // initialize digital pin GPIO14 (Red LED1) as an output.
pinMode(yellow_LED1, OUTPUT);   // initialize digital pin GPIO12 (yellow LED1) as an output.
pinMode(green_LED1, OUTPUT);      // initialize digital pin GPIO13 (green LED1) as an output.
}

// the loop function runs over and over again forever
void loop() {
  // The next three lines of code turn on the red LED1
  digitalWrite(red_LED1, HIGH);       // This should turn on the RED LED1
  digitalWrite(yellow_LED1 , LOW);    //  This should turn off the YELLOW LED1
  digitalWrite(green_LED1, LOW);      //  This should turn off the GREEN LED1

  delay(2000);                    // wait for 2 seconds

  // The next three lines of code turn on the green LED1
  digitalWrite(red_LED1, LOW);        // This should turn off the RED LED1
```

Done uploading.

```
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.0 seconds (effective 1445.7 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

14 DOIT ESP32 DEVKIT V1 on COM4

Creating a Multiple Traffic Light Controller

Picture of circuit with working LEDs

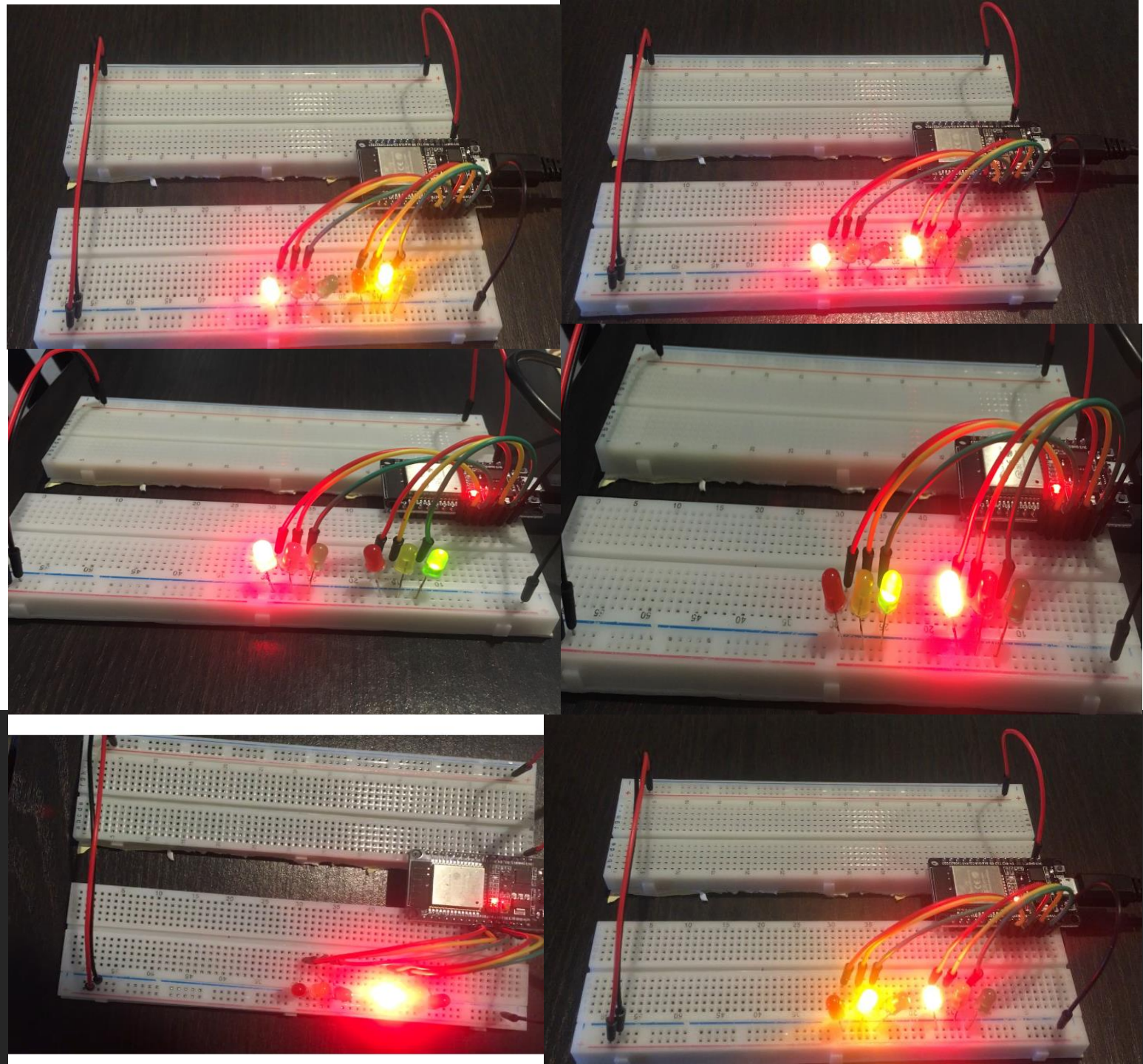
ESP 32 Board

Colored LEDs: Red, Yellow and Green (two sets)

220 Ohm Resistors (optional)

Wires

Breadboard



CEIS114_M4 | Arduino 1.8.13

File Edit Sketch Tools Help

CEIS114_M4

```
// === Maryam Zishan ====
// Module #4 project

// Define some labels
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

// the setup function runs once when you press reset or power the board
void setup() {
 pinMode(red_LED1, OUTPUT);  // initialize digital pin GPIO14 (Red LED1) as an output.
 pinMode(yellow_LED1, OUTPUT);  // initialize digital pin GPIO12 (yellow LED1) as an output.
 pinMode(green_LED1, OUTPUT);    // initialize digital pin GPIO13 (green LED1) as an output.
 pinMode(red_LED2, OUTPUT);  // initialize digital pin GPIO25(Red LED2) as an output.
 pinMode(yellow_LED2, OUTPUT); // initialize digital pin GPIO26 (yellow LED2) as an output.
 pinMode(green_LED2, OUTPUT); // initialize digital pin GPIO27 (green LED2) as an output.
}
```

Done uploading.

```
Writing at 0x00018000... (42 %)
Writing at 0x0001c000... (57 %)
Writing at 0x00020000... (71 %)
Writing at 0x00024000... (85 %)
Writing at 0x00028000... (100 %)
Wrote 199184 bytes (104391 compressed) at 0x00010000 in 1.6 seconds (effective 1023.4 kbit/s)...
Hash of data verified.
Compressed 3072 bytes to 128...
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.0 seconds (effective 1536.0 kbit/s)...
Hash of data verified.

Leaving...
```

1 DOIT ESP32 DEVKIT V1 on COM4

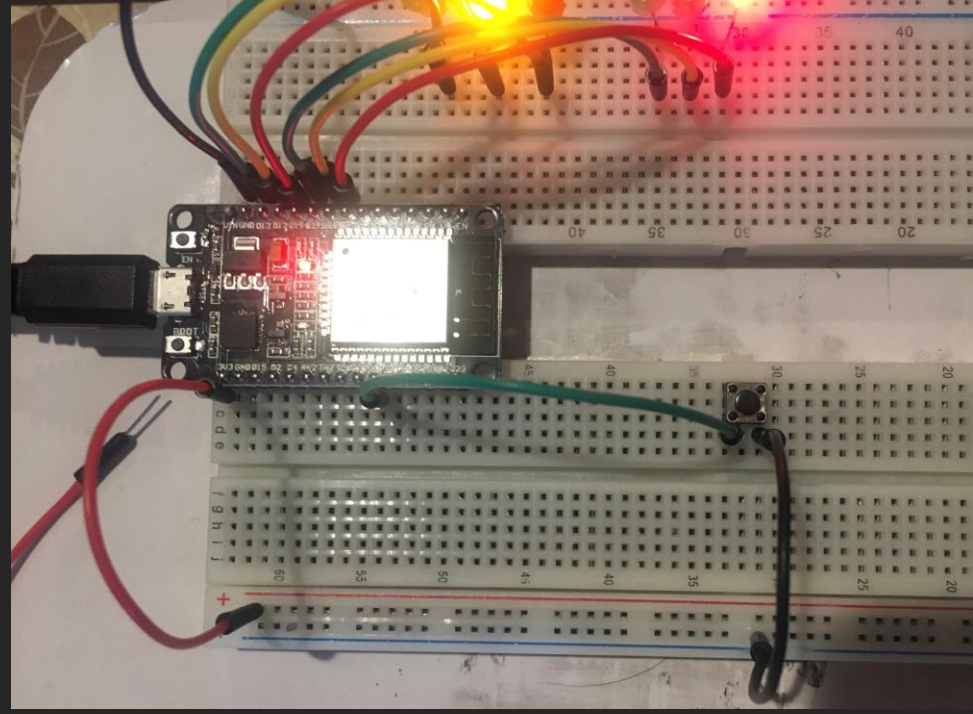
Screenshot of code in Arduino IDE

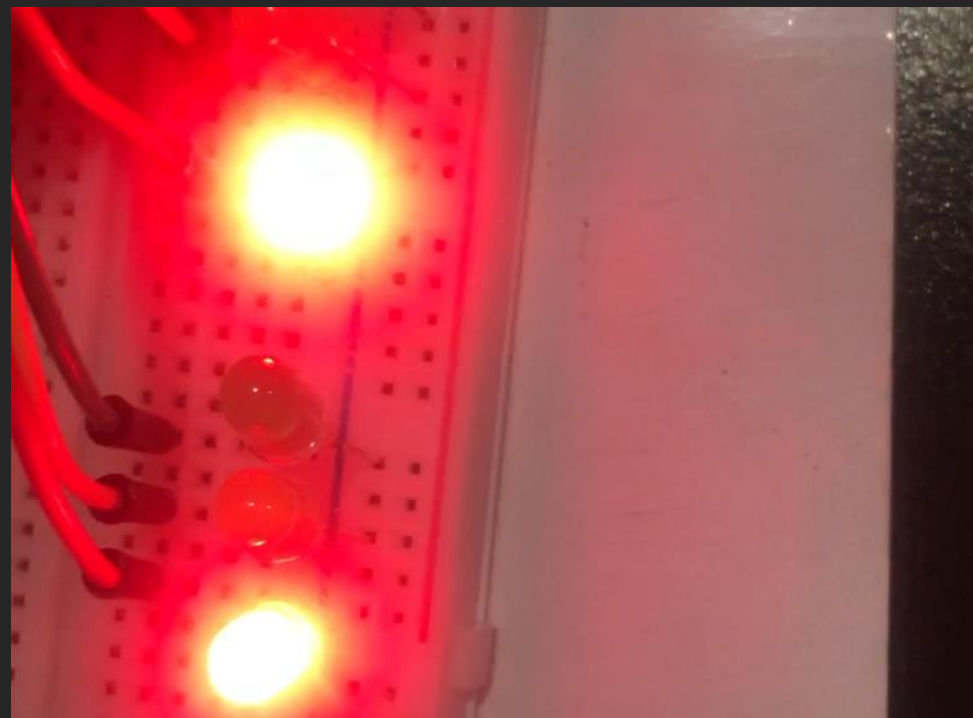
Screenshot of code in Arduino IDE showing **your name in the comment**

Creating a Multiple Traffic Light Controller with a Cross Walk

Picture of circuit with working LEDs

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





CEIS114_M5 | Arduino 1.8.13

File Edit Sketch Tools Help

CEIS114_M5

```
// === Maryam Zishan ====
// Module #5 project
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

// the setup function runs once when you press reset or power the board
void setup() {

 pinMode(Xw_button, INPUT_PULLUP); // 0=pressed, 1 = unpressed button
 Serial.begin(115200);
 pinMode(red_LED1, OUTPUT);  // initialize digital pin 14 (Red LED1) as an output.
 pinMode(yellow_LED1, OUTPUT);  // initialize digital pin 12 (yellow LED1) as an output.
 pinMode(green_LED1, OUTPUT);    // initialize digital pin 13 (green LED1) as an output.

 pinMode(red_LED2, OUTPUT);  // initialize digital pin 25(Red LED2) as an output.
 pinMode(yellow_LED2, OUTPUT); // initialize digital pin 26 (yellow LED2) as an output.
 pinMode(green_LED2, OUTPUT); // initialize digital pin 27 (green LED2) as an output.
}
```

Done uploading.

```
Changed.
Configuring flash size...
Auto-detected Flash size: 4MB
Compressed 8192 bytes to 47...
Writing at 0x0000e000... (100 %)
Wrote 8192 bytes (47 compressed) at 0x0000e000 in 0.0 seconds (effective 4681.2 kbit/s)...
Hash of data verified.
```

105 DOIT ESP32 DEVKIT V1 on COM4

Screenshot of code in Arduino IDE

Screenshot of code in Arduino IDE showing **your name in the comment**

CEIS114_M5 | Arduino 1.8.13

File Edit Sketch Tools Help

CEIS114_M5

```
// === Maryam Zishan ====
// Module #5 project
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

// the setup function runs once when you press reset or power the board
void setup() {

 pinMode(Xw_button, INPUT_PULLUP); // 0=pressed, 1 = unpressed button
 Serial.begin(115200);
 pinMode(red_LED1, OUTPUT);  // initialize digital pin 14 (Red LED1) as an output.
 pinMode(yellow_LED1, OUTPUT);  // initialize digital pin 12 (yellow LED1) as an output.
 pinMode(green_LED1, OUTPUT);    // initialize digital pin 13 (green LED1) as an output.

 pinMode(red_LED2, OUTPUT);  // initialize digital pin 25(Red LED2) as an output.
 pinMode(yellow_LED2, OUTPUT); // initialize digital pin 26 (yellow LED2) as an output.
 pinMode(green_LED2, OUTPUT); // initialize digital pin 27 (green LED2) as an output.
}

// the loop function runs over and over again forever
void loop() {
```

COM4

Send

```
Count =  1  == Walk ==
 == Do Not Walk ==
 == Do Not Walk ==
 == Do Not Walk ==
Count =  10  == Walk ==
Count =  9  == Walk ==
Count =  8  == Walk ==
Count =  7  == Walk ==
Count =  6  == Walk ==
Count =  5  == Walk ==
Count =  4  == Walk ==
Count =  3  == Walk ==
Count =  2  == Walk ==
Count =  1  == Walk ==
 == Do Not Walk ==
 == Do Not Walk ==
```

Autoscroll Show timestamp Newline 115200 baud Clear output

```
Leaving...
Hard resetting via RTS pin...
```

DOIT ESP32 DEVKIT V1 on COM4

Type here to search

6:00 PM
4/1/2021

Screenshot of Serial Monitor in Arduino IDE

Screenshot of output in Serial Monitor

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

Picture of circuit with working LEDs and LCD display

ESP 32 Board

Colored LEDs: Red, Yellow and Green (two sets)

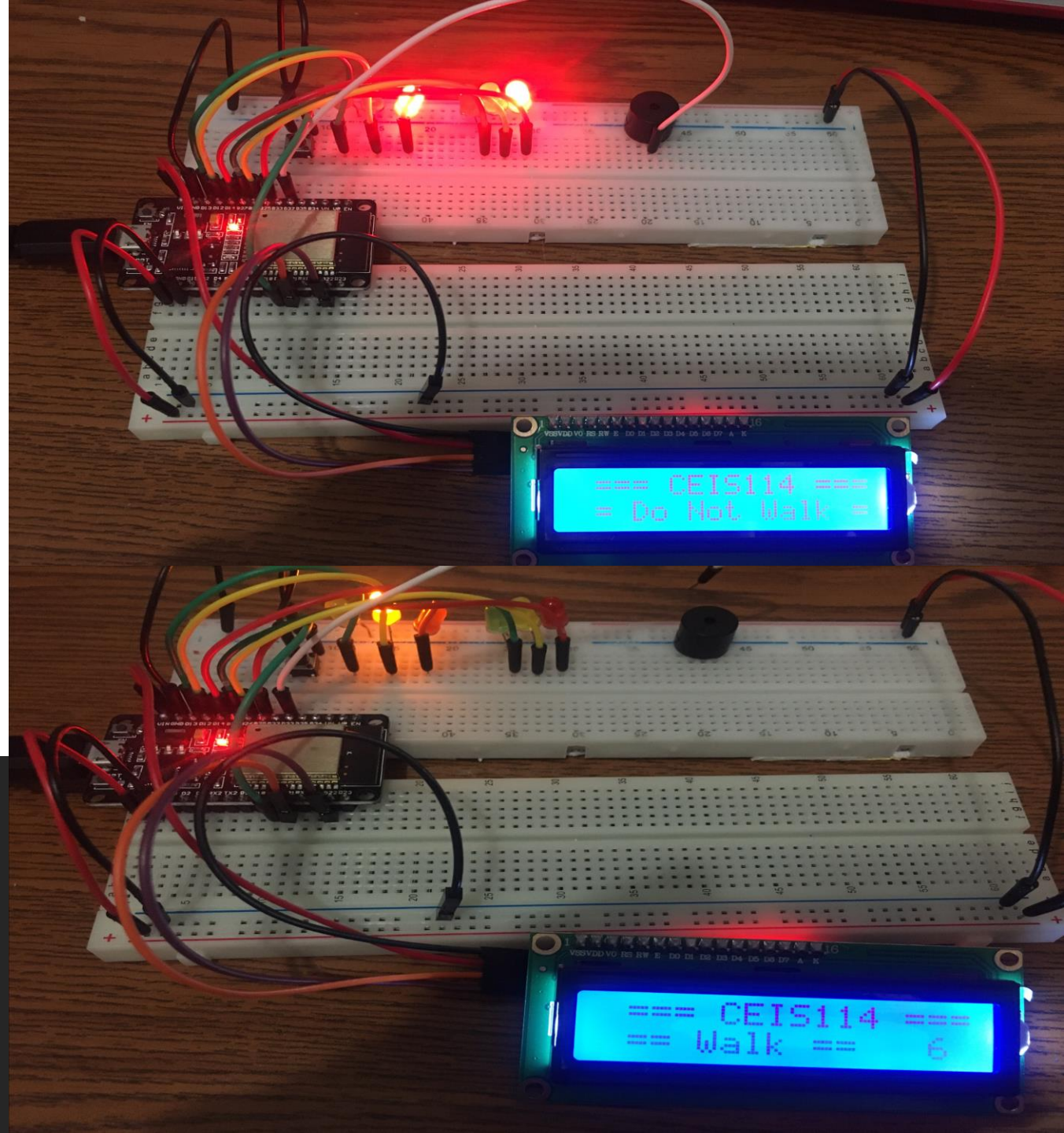
220 Ohm Resistors (optional)

Push Button

LCD Unit with Message Display

Wires

Breadboard



Screenshot of code in Arduino IDE showing **your name in the comment**

Screenshot of code in Arduino IDE

CEIS114_M6 | Arduino 1.8.13

File Edit Sketch Tools Help

CEIS114_M6

```
#include <LiquidCrystal_I2C.h>

// === Maryam Zishan ====
// 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
```

```
Hash of data verified.
Compressed 3072 bytes to 128...
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.0 seconds (effective 1536.0 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

16 DOIT ESP32 DEVKIT V1 on COM4

Type here to search 3:05 AM 4/16/2021

CEIS114_M6 | Arduino 1.8.13

File Edit Sketch Tools Help

CEIS114_M6

```
#include <LiquidCrystal_I2C.h>

// === Maryam Zishan ====
// Module #6 project
#include <Wire.h>  //lcd
#include <LiquidCrystal_I2C.h> //lcd
LiquidCrystal_I2C lcd(0x27,16,2); //set the LCD address to 0x3F
// if it does not work then try 0x3F, if both addresses do not w

const int bzr=32;      // GPIO32 to connect the Buzzer
//==================== LCD ====================
const int red_LED1  = 14;   // The red LED1 is wired to ESP32 bo
const int yellow_LED1  =12;   // The yellow LED1 is wired to ESI
const int green_LED1 = 13; // The green LED1 is wired to ESP32 b
const int red_LED2  = 25;   // The red LED2 is wired to Mega boa
const int yellow_LED2  = 26;   // The yellow LED2 is wired to Me
const int green_LED2 = 27; // The green LED2 is wired to Mega bo

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

void setup() {
  Serial.begin(115200);
```

COM4

Send

```
 == Do Not Walk ==
Count =  10  == Walk ==
Count =  9  == Walk ==
Count =  8  == Walk ==
Count =  7  == Walk ==
Count =  6  == Walk ==
Count =  5  == Walk ==
Count =  4  == Walk ==
Count =  3  == Walk ==
Count =  2  == Walk ==
Count =  1  == Walk ==
Count =  0  == Walk ==
 == Do Not Walk ==
 == Do Not Walk ==
 == Do Not Walk ==
 == Do Not Walk ==
```

Autoscroll Show timestamp Newline 115200 baud Clear output

Screenshot of Serial Monitor in Arduino IDE

SCREENSHOT OF OUTPUT IN SERIAL MONITOR

Adding remote emergency control IoT Controller- option 1

Picture of circuit with working LEDs and LCD display

ESP 32 Board

Colored LEDs: Red, Yellow and Green (two sets)

One Blue LED – Emergency Light

220 Ohm Resistors (optional)

Push Buttons - 2

LCD Unit

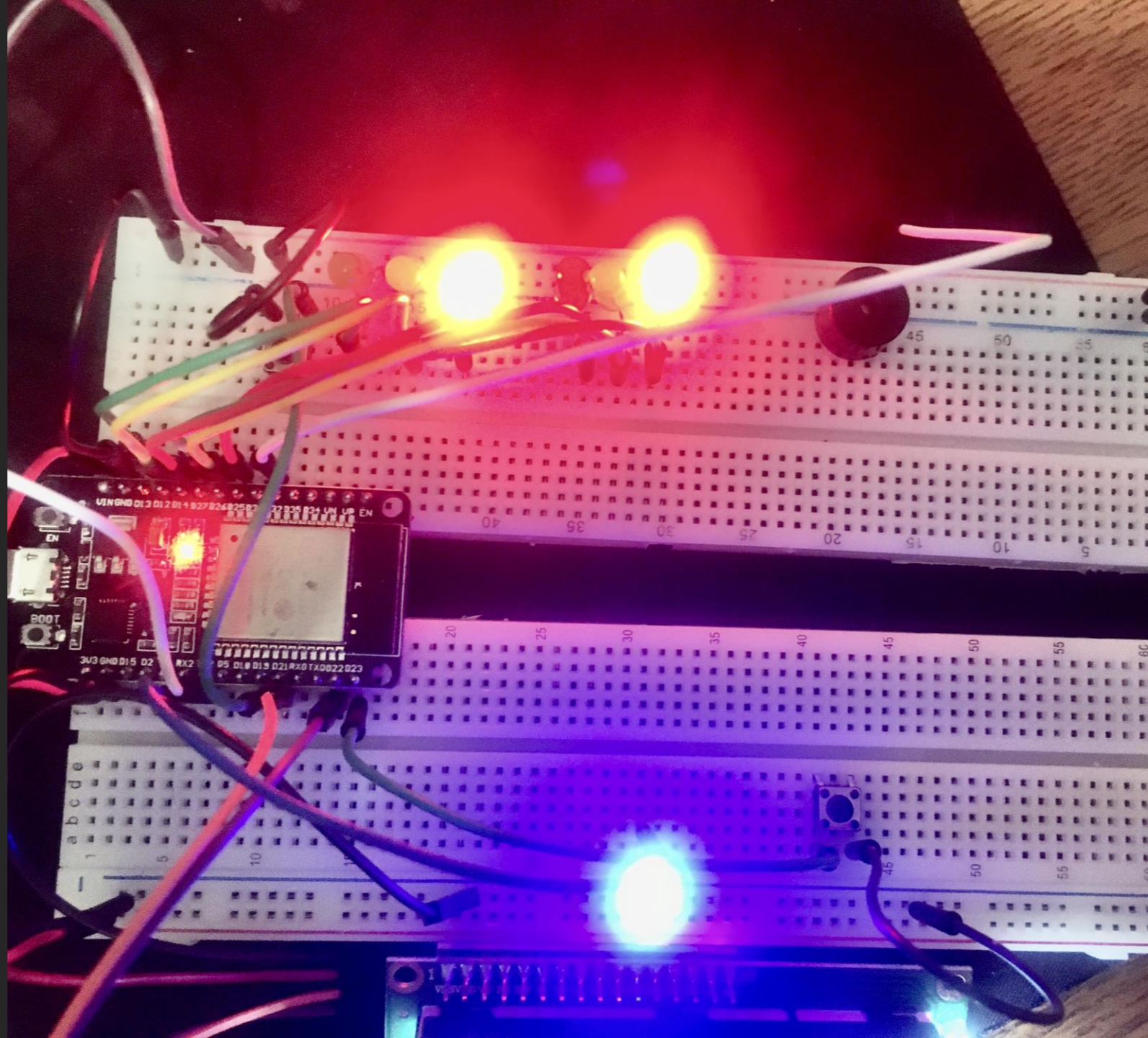
Buzzer

PIR Motion Sensor

Router

Wires

Breadboard



CEIS114_M7 | Arduino 1.8.13

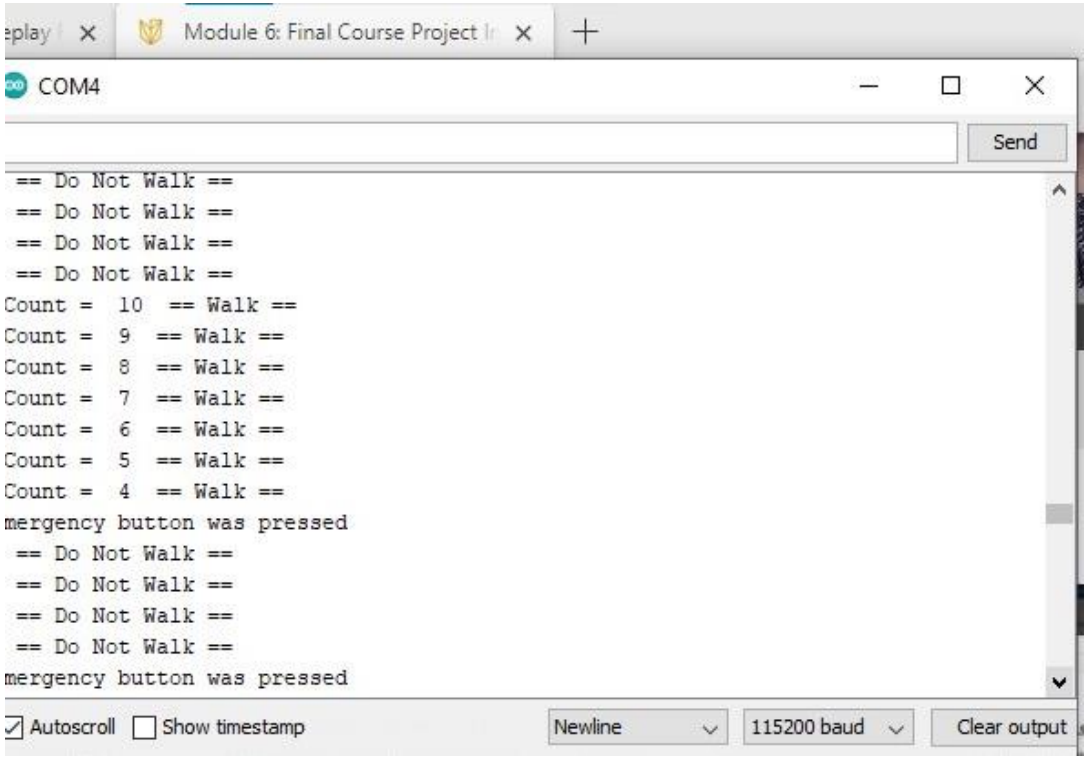
File Edit Sketch Tools Help

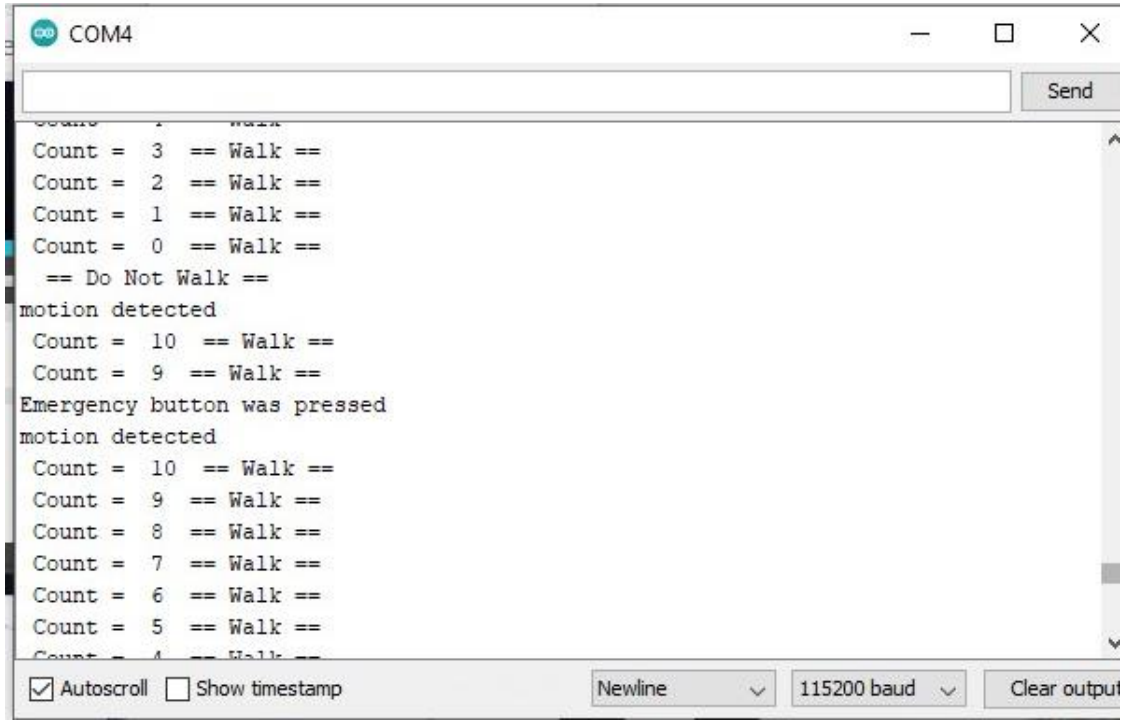
CEIS114_M7

```
// === Maryam Zishan ====
// Final Project Component - Option 2

#include <Wire.h>  //lcd
#include <LiquidCrystal_I2C.h> //lcd
LiquidCrystal_I2C lcd(0x3F,16,2); //set the LCD address to 0x27 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

// Set GPIOs for LED and PIR Motion Sensor
const int led = 16; // Flashing White (Blue) Led
const int motionSensor = 17;
int pirState = 0 ;
int j,Em_value,Xw_value;

const int Em_button = 23;  // Emergency button
const int Xw_button = 19; //Cross Walk button

//==================== LCD ====================
const int red_LED1  = 14;   // The red LED1 is wired to ESP32 board pin GPIO04
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
```

Screenshot of code in Arduino IDE

SCREENSHOT OF CODE IN ARDUINO IDE SHOWING **YOUR NAME IN THE COMMENT**





Screenshot of Serial Monitor in Arduino IDE

SCREENSHOT OF OUTPUT IN SERIAL MONITOR

Challenges

Below is the List of some of the challenges that I had to face:

- The LED won't turn on.
- Reading pin numbers on board could be quite challenging
- Sometimes I would miss to copy the last lines of the code
- While I fixing the LCD I pushed too hard with the screw driver and damaged it! Realizing that it should be twisted very gently.
- Extra wires were attached to make sure the current in the breadboard flows as desired.
- Adding zipped files in libraries.

Career Skills

Basic skills that I acquired through this project are listed below:

- Basic understanding of Hardware and Software
- Network connectivity and Devices
- Code/ programming/Python
- Importance of Security and it's knowledge at basic level
- Facts on IoT
- Business Analytics and Fundaments of Operating System
- Learned about P2D2 and M2M
- HOW ESP32 works along with the motion detector as well as web and iCloud connectivity!
- Buzzer alerts
- Understanding of traffic signals operability as well as use of SMART IoT devices in order to enhance the quality of traffic flow with increased passenger safety

Conclusion

With basic understanding of Arduino Mega Board and effective use of it's component I feel more confident about my future in the world of IoT.

Hopefully, this will give me some confidence in terms of how devices are created to benefit people in the best way possible. Not only aiming towards a better quality of life but also a more secure and safer experience.

By creating this traffic signal controller and motion alerts we can help a lot of people by reducing the number of accidents. This will help a lot of lives and give people a less stressful driving experience. Indeed, the IoT is here to make this world a better place and I am grateful to be a part of it!