**IOT ASSIGNMENT DOCUMENTATION**
**NAME: SALMAN ASHRAF**
**ROLL NO: 1272**

**SECTION: BSAI**

-------------------------------------------------------------------------------------------

**There are two tasks in the assignment**

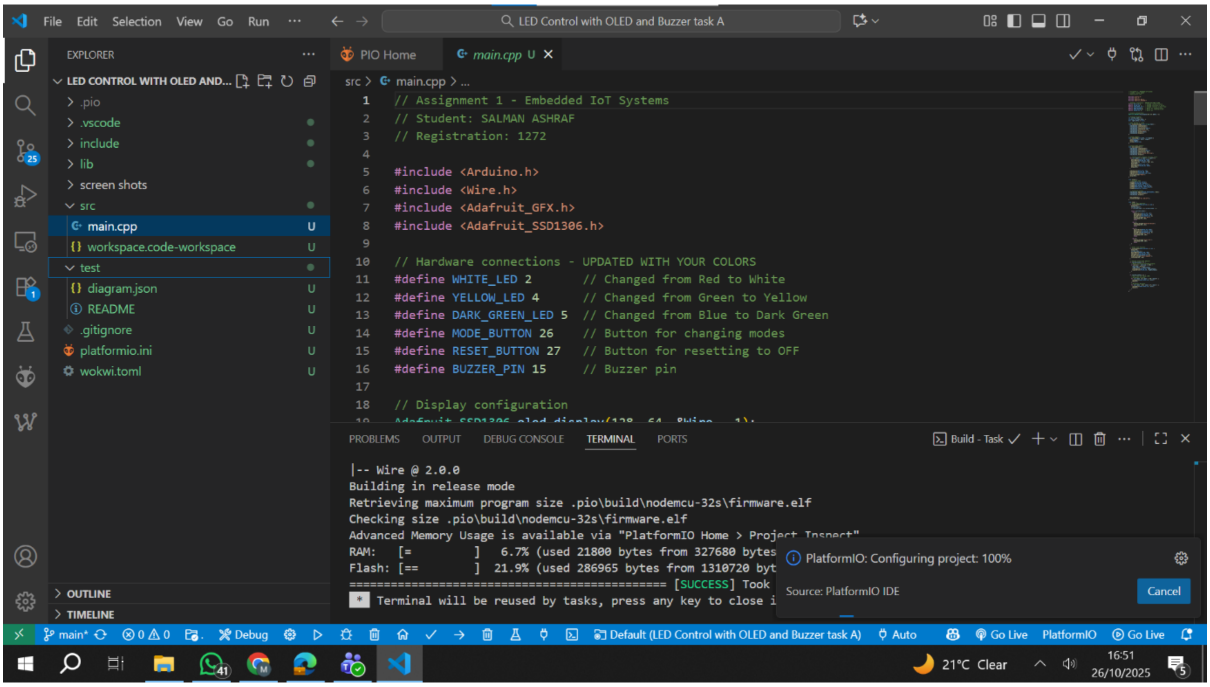
# TASK 1:

# Maincpp code:

```
// Assignment 1 - Embedded IoT Systems

// Student: SALMAN ASHRAF

// Registration: 1272


#include <Arduino.h>

#include <Wire.h>

#include <Adafruit_GFX.h>

#include <Adafruit_SSD1306.h>


// Hardware connections - UPDATED WITH YOUR COLORS

#define WHITE_LED 2       // Changed from Red to White

#define YELLOW_LED 4      // Changed from Green to Yellow

#define DARK_GREEN_LED 5  // Changed from Blue to Dark Green

#define MODE_BUTTON 26    // Button for changing modes

#define RESET_BUTTON 27   // Button for resetting to OFF

#define BUZZER_PIN 15     // Buzzer pin


// Display configuration

Adafruit_SSD1306 oled_display(128, 64, &Wire, -1);


// Program variables

int current_state = 1;
```

```
unsigned long last_time = 0;

bool light_status = false;


// Show message on screen

void display_message(const String &text) {

  oled_display.clearDisplay();

  oled_display.setTextSize(1);

  oled_display.setTextColor(WHITE);

  oled_display.setCursor(5, 28);

  oled_display.print("Status: ");

  oled_display.println(text);

  oled_display.display();

}


// Sound feedback

void sound_beep(int pitch, int length) {

  tone(BUZZER_PIN, pitch, length);

  delay(length);

  noTone(BUZZER_PIN);

}


// Initial boot sequence

void boot_sequence() {

  oled_display.clearDisplay();

  oled_display.setTextSize(1);

  oled_display.setTextColor(WHITE);

  oled_display.setCursor(10, 30);

  oled_display.print("System Starting");

  oled_display.display();


  // Flashing lights sequence with your colors
  for (int count = 0; count < 3; count++) {
    digitalWrite(WHITE_LED, HIGH);
    digitalWrite(YELLOW_LED, LOW);
    sound_beep(600 + (count * 300), 80);
    delay(100);
    digitalWrite(WHITE_LED, LOW);
    digitalWrite(YELLOW_LED, HIGH);
    delay(100);
  }

  digitalWrite(WHITE_LED, LOW);
  digitalWrite(YELLOW_LED, LOW);
  digitalWrite(DARK_GREEN_LED, LOW);
  display_message("Ready");
  delay(500);
}

void setup() {
  // Configure pins
  pinMode(WHITE_LED, OUTPUT);
  pinMode(YELLOW_LED, OUTPUT);
  pinMode(DARK_GREEN_LED, OUTPUT);
  pinMode(MODE_BUTTON, INPUT_PULLUP);
  pinMode(RESET_BUTTON, INPUT_PULLUP);
  pinMode(BUZZER_PIN, OUTPUT);

  // Initialize display
  oled_display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  oled_display.clearDisplay();
  oled_display.display();
  boot_sequence();
  display_message("All LEDs Off");
}

void loop() {
  // Handle mode selection
  if (digitalRead(MODE_BUTTON) == LOW) {
    delay(150);
    current_state++;
    if (current_state > 4) current_state = 1;

    switch (current_state) {
      case 1:
        digitalWrite(WHITE_LED, LOW);
        digitalWrite(YELLOW_LED, LOW);
        analogWrite(DARK_GREEN_LED, 0);
        display_message("All Off");
        sound_beep(750, 100);
        break;

      case 2:
        display_message("Alternate Blink");
        sound_beep(950, 100);
        break;

      case 3:
        digitalWrite(WHITE_LED, HIGH);
        digitalWrite(YELLOW_LED, HIGH);
        analogWrite(DARK_GREEN_LED, 0);
        display_message("All On");
        sound_beep(1150, 100);
        break;

      case 4:
        display_message("Green Fade");
        sound_beep(1400, 100);
        break;
    }
  }

  // Handle reset function
  if (digitalRead(RESET_BUTTON) == LOW) {
    delay(150);
    current_state = 1;
    digitalWrite(WHITE_LED, LOW);
    digitalWrite(YELLOW_LED, LOW);
    analogWrite(DARK_GREEN_LED, 0);
    display_message("Reset Complete");
    sound_beep(450, 150);
  }

  // Execute current mode behavior
  if (current_state == 2) {
    // White and Yellow alternate blinking
    if (millis() - last_time >= 600) {
      last_time = millis();
      light_status = !light_status;
      digitalWrite(WHITE_LED, light_status);
      digitalWrite(YELLOW_LED, !light_status);
    }
  }
```

```
  if (current_state == 4) {
    // Dark Green LED fading (PWM)
    // Brightness increase
    for (int level = 0; level <= 255; level++) {
      analogWrite(DARK_GREEN_LED, level);
      delay(5);
    }
    // Brightness decrease
    for (int level = 255; level >= 0; level--) {
      analogWrite(DARK_GREEN_LED, level);
      delay(5);
    }
  }
}
```



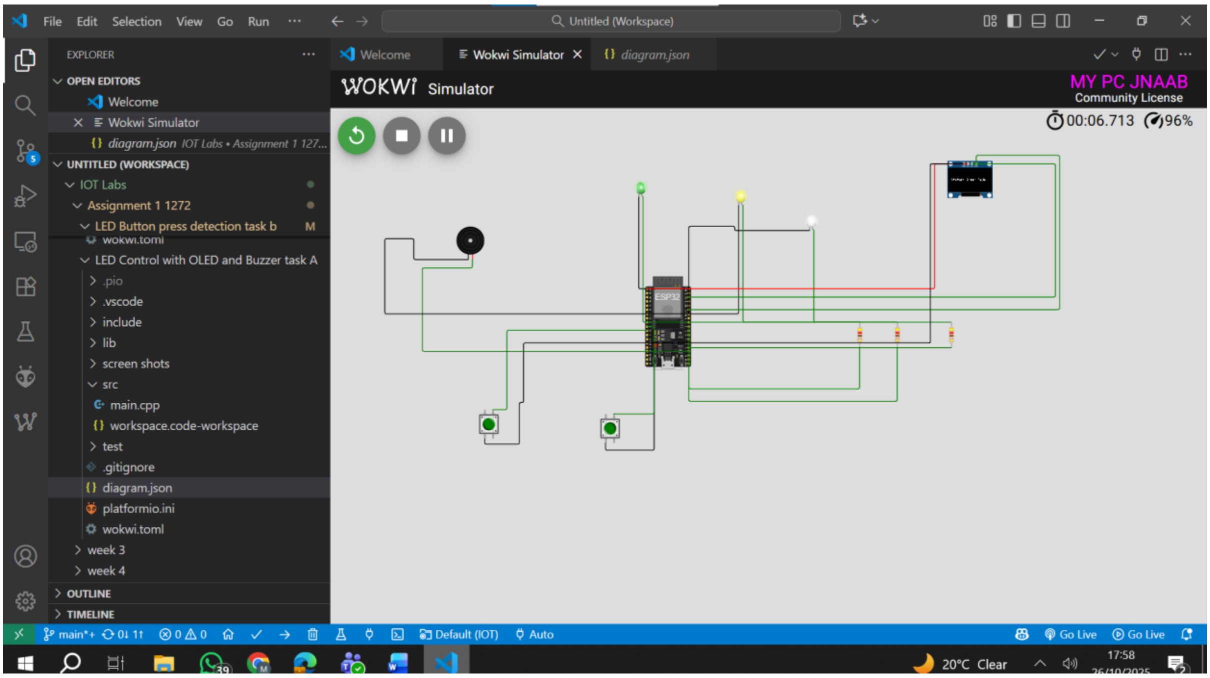
```
// Assignment 1 - Embedded IoT Systems
// Student: SALMAN ASHRAF
// Registration: 1272

#include <Arduino.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

// Hardware connections - UPDATED WITH YOUR COLORS
#define WHITE_LED 2       // Changed from Red to White
#define YELLOW_LED 4      // Changed from Green to Yellow
#define DARK_GREEN_LED 5  // Changed from Blue to Dark Green
#define MODE_BUTTON 26    // Button for changing modes
#define RESET_BUTTON 27   // Button for resetting to OFF
#define BUZZER_PIN 15     // Buzzer pin

// Display configuration
```

```
|-- Wire @ 2.0.0
Building in release mode
Retrieving maximum program size .pio\build\nodemcu-32s\firmware.elf
Checking size .pio\build\nodemcu-32s\firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM:   [=         ]   6.7% (used 21800 bytes from 327680 bytes
Flash: [==        ]  21.9% (used 286965 bytes from 1310720 byt
============================================= [SUCCESS] Took
 *  Terminal will be reused by tasks, press any key to close i
```

## DIAGRAM.JSON code

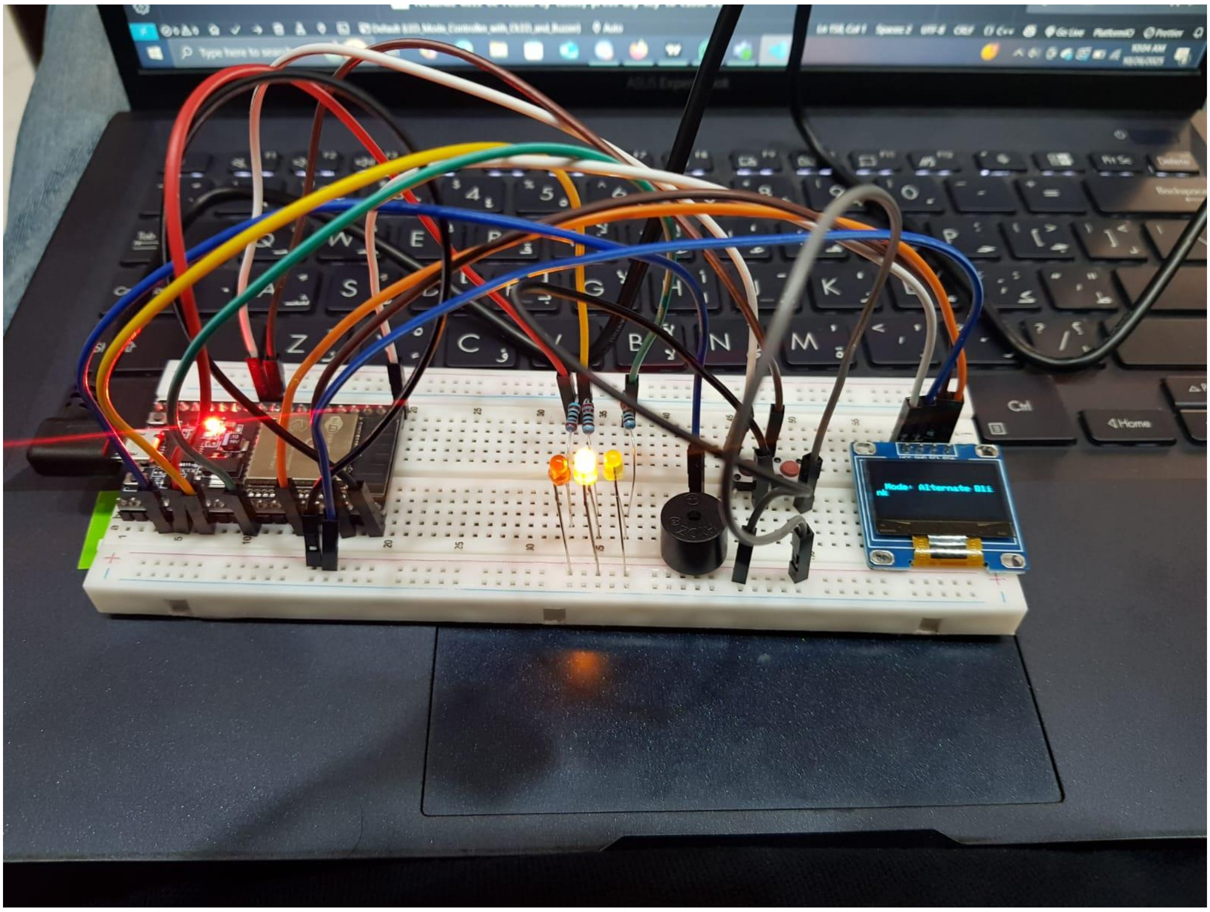
```
{
  "version": 1,
  "author": "SALMAN ASHRAF",
"editor": "wokwi",
"parts": [
  { "type": "board-esp32-devkit-c-v4", "id": "esp", "top": 19.2, "left": -62.36, "attrs": {} },
  {
    "type": "wokwi-pushbutton",
    "id": "btn1",
    "top": 345.2,
    "left": -175.6,
    "rotate": 270,
    "attrs": { "color": "green", "xray": "1" }
  },
  {
    "type": "wokwi-pushbutton",
    "id": "btn2",
    "top": 335.4,
    "left": -457.8,
    "rotate": 90,
    "attrs": { "color": "green", "xray": "1" }
  },
  {
    "type": "wokwi-led",
    "id": "led1",
    "top": -128.4,
    "left": 301.4,
    "attrs": { "color": "white" }
  },
  {
    "type": "wokwi-led",
    "id": "led2",
    "top": -186,
    "left": 138.2,
    "attrs": { "color": "yellow" }
  },
{
  "type": "wokwi-led",
  "id": "led3",
  "top": -205.2,
  "left": -92.2,
  "attrs": { "color": "green" }
},
{
  "type": "board-ssd1306",
  "id": "oled1",
  "top": -246.46,
  "left": 633.83,
  "attrs": { "i2cAddress": "0x3c" }
},
{
  "type": "wokwi-buzzer",
  "id": "bz1",
  "top": -103.2,
  "left": -497.4,
  "attrs": { "volume": "0.1" }
},
{
  "type": "wokwi-resistor",
  "id": "r1",
  "top": 148.8,
  "left": 613.85,
  "rotate": 90,
  "attrs": { "value": "420" }
},
{
  "type": "wokwi-resistor",
  "id": "r2",
  "top": 148.8,
    "left": 489.05,
    "rotate": 90,
    "attrs": { "value": "420" }
  },
  {
    "type": "wokwi-resistor",
    "id": "r3",
    "top": 148.8,
    "left": 402.65,
    "rotate": 90,
    "attrs": { "value": "420" }
  }
],
"connections": [
  [ "esp:TX", "$serialMonitor:RX", "", [] ],
  [ "esp:RX", "$serialMonitor:TX", "", [] ],
  [ "led3:C", "esp:GND.2", "black", [ "v0" ] ],
  [ "led2:C", "esp:GND.3", "black", [ "v0" ] ],
  [ "led1:C", "esp:GND.3", "black", [ "h-172.4", "v-9.6", "h-76.8" ] ],
  [ "led3:A", "r3:1", "green", [ "v0" ] ],
  [ "led2:A", "r2:1", "green", [ "v0" ] ],
  [ "led1:A", "r1:1", "green", [ "v0" ] ],
  [ "btn2:2.r", "esp:GND.1", "black", [ "v12.6", "h79.6", "v-96", "h9.6", "v-28.8" ] ],
  [ "btn1:1.l", "esp:GND.1", "black", [ "v16.4", "h112.2", "v-19.2" ] ],
  [ "bz1:2", "esp:15", "green", [ "v19.2", "h-115.6", "v-105.6" ] ],
  [ "r1:2", "esp:2", "green", [ "h0" ] ],
  [ "r3:2", "esp:5", "green", [ "v94.8", "h-134.4" ] ],
  [ "r2:2", "esp:4", "green", [ "v123.6", "h-211.2" ] ],
  [ "oled1:VCC", "esp:3V3", "red", [ "h-76.65", "v153.6" ] ],
  [ "oled1:SDA", "esp:21", "green", [ "v-19.2", "h192.07", "v192" ] ],
  [ "oled1:SCL", "esp:22", "green", [ "h192.3", "v144" ] ],
  [ "btn1:2.r", "esp:26", "green", [ "h93.2", "v25.8" ] ],
  [ "btn2:1.l", "esp:27", "green", [ "h31.8", "v-74" ] ],
    [ "bz1:1", "esp:GND.3", "black", [ "h-124.8", "v-48", "h-67.2", "v-124.8" ] ],
    [ "oled1:GND", "esp:GND.1", "black", [ "h-76.8", "v249.6" ] ]
  ],
  "dependencies": {}
}
```

## JSON diagram on VS

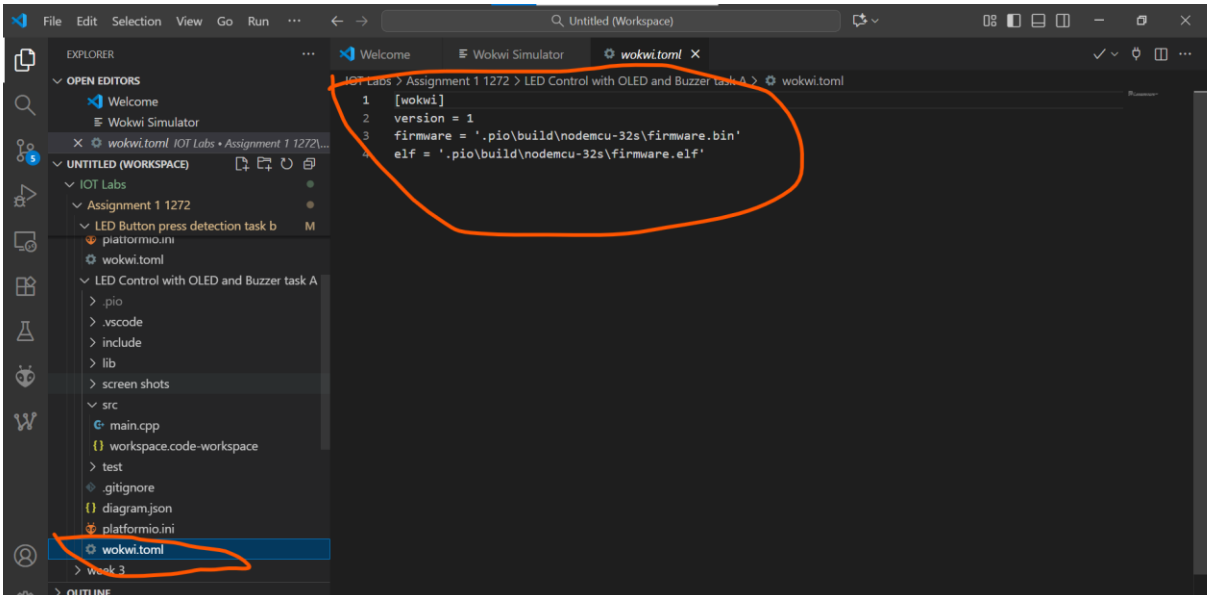


## PRACTICAL WORK

task1 running.mp4



## Wokwi.toml file



```
[wokwi]
version = 1
firmware = '.pio\build\nodemcu-32s\firmware.bin'
elf = '.pio\build\nodemcu-32s\firmware.elf'
```

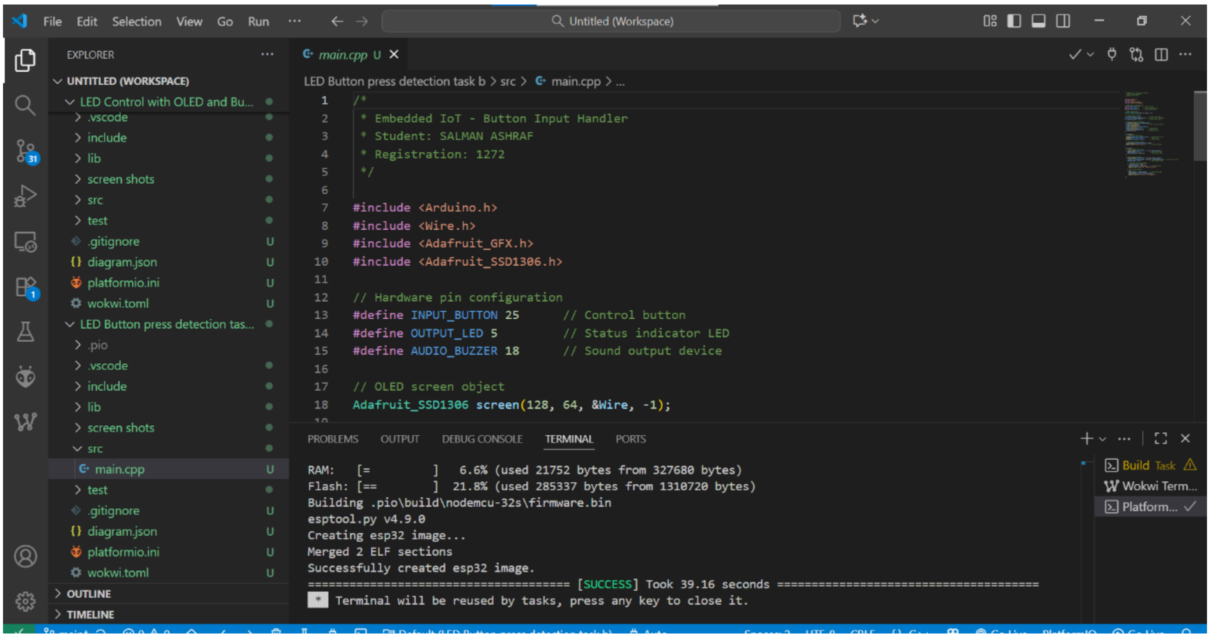
## Its is necessary for run the diagram

# TASK B (BUTTON PRESS DETECTION)

## Maincpp code

```
/*
 * Embedded IoT - Button Input Handler
 * Student: SALMAN ASHRAF
 * Registration: 1272
 */

#include <Arduino.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

// Hardware pin configuration
#define INPUT_BUTTON 25      // Control button
#define OUTPUT_LED 5         // Status indicator LED
#define AUDIO_BUZZER 18      // Sound output device

// OLED screen object
Adafruit_SSD1306 screen(128, 64, &Wire, -1);

// System state variables
bool light_status = false;          // Current LED state
unsigned long button_down_time = 0;  // Button press timestamp
bool button_active = false;         // Button press flag

// Update display with message
void update_screen(String display_text) {
  screen.clearDisplay();            // Clear previous content
  screen.setTextSize(1);
  screen.setTextColor(SSD1306_WHITE);
  screen.setCursor(5, 25);          // Text position
  screen.println(display_text);      // Output text
  screen.display();                  // Refresh screen
}

void setup() {
  // Configure hardware pins
  pinMode(INPUT_BUTTON, INPUT_PULLUP);  // Button with pull-up
  pinMode(OUTPUT_LED, OUTPUT);          // LED output
  pinMode(AUDIO_BUZZER, OUTPUT);        // Buzzer output

  // Initialize display system
  screen.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  update_screen("System Active");       // Initial message
}

void loop() {
  // Detect button press start
  if (digitalRead(INPUT_BUTTON) == LOW && !button_active) {
    button_active = true;               // Set button state
    button_down_time = millis();        // Record press time
  }

  // Detect button release
  if (digitalRead(INPUT_BUTTON) == HIGH && button_active) {
    unsigned long hold_duration = millis() - button_down_time; // Calculate hold time
    button_active = false;              // Reset button state

    // Long press handling (>1500ms)
    if (hold_duration > 1500) {
      tone(AUDIO_BUZZER, 1200, 600);    // Activate buzzer
      update_screen("Long Press - Buzzer Active");
    }
    // Short press handling
    else {
      light_status = !light_status;     // Toggle LED state
      digitalWrite(OUTPUT_LED, light_status);
      update_screen("Short Press - LED Toggled");
    }
  }
}
```

## SUCCESSFULLY BUILD



```
/*
 * Embedded IoT - Button Input Handler
 * Student: SALMAN ASHRAF
 * Registration: 1272
 */

#include <Arduino.h>
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>

// Hardware pin configuration
#define INPUT_BUTTON 25       // Control button
#define OUTPUT_LED 5          // Status indicator LED
#define AUDIO_BUZZER 18       // Sound output device

// OLED screen object
Adafruit_SSD1306 screen(128, 64, &Wire, -1);
```

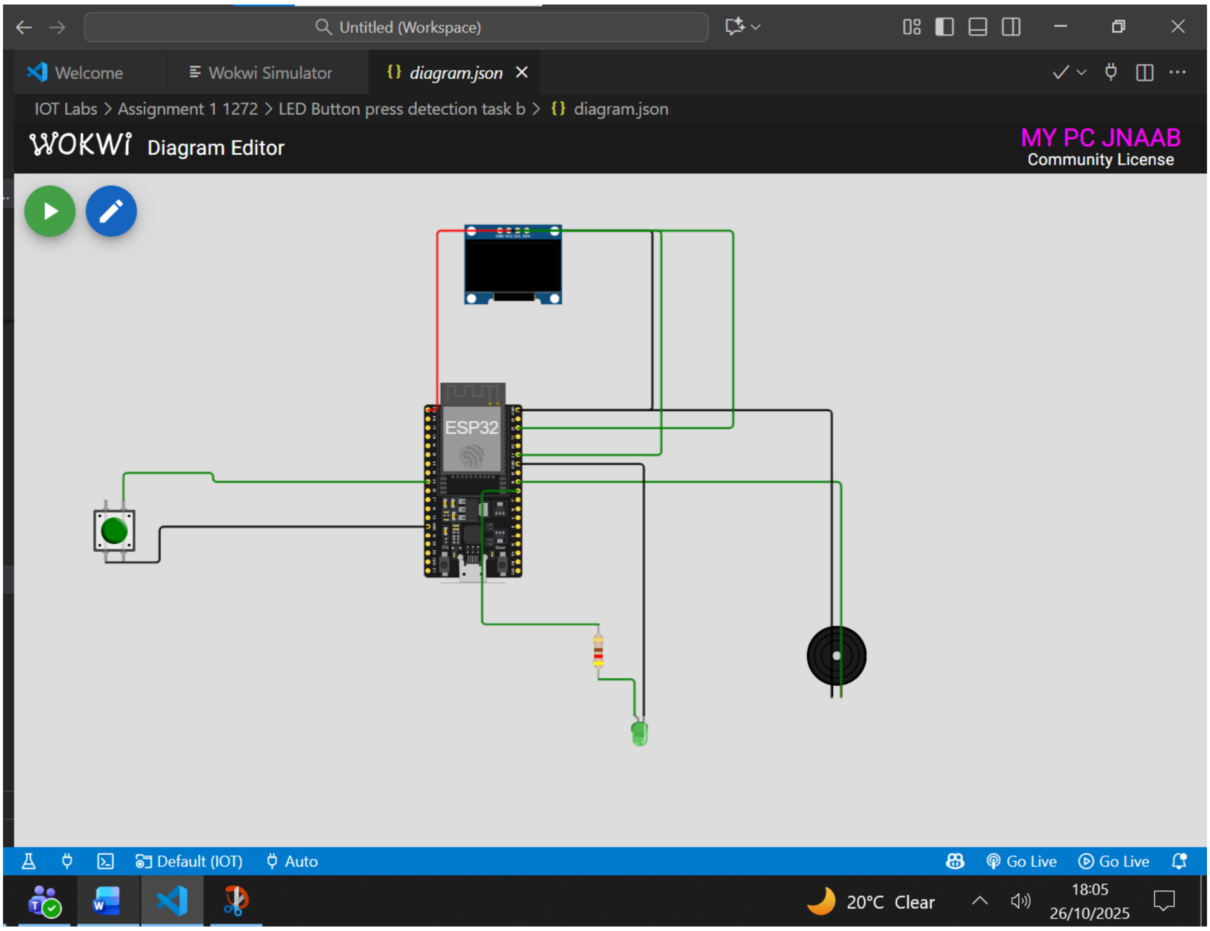
```
RAM:   [=         ]   6.6% (used 21752 bytes from 327680 bytes)
Flash: [==        ]  21.8% (used 285337 bytes from 1310720 bytes)
Building .pio\build\nodemcu-32s\firmware.bin
esptool.py v4.9.0
Creating esp32 image...
Merged 2 ELF sections
Successfully created esp32 image.
======================================== [SUCCESS] Took 39.16 seconds ========================================
 *  Terminal will be reused by tasks, press any key to close it.
```

# JSON CODE :

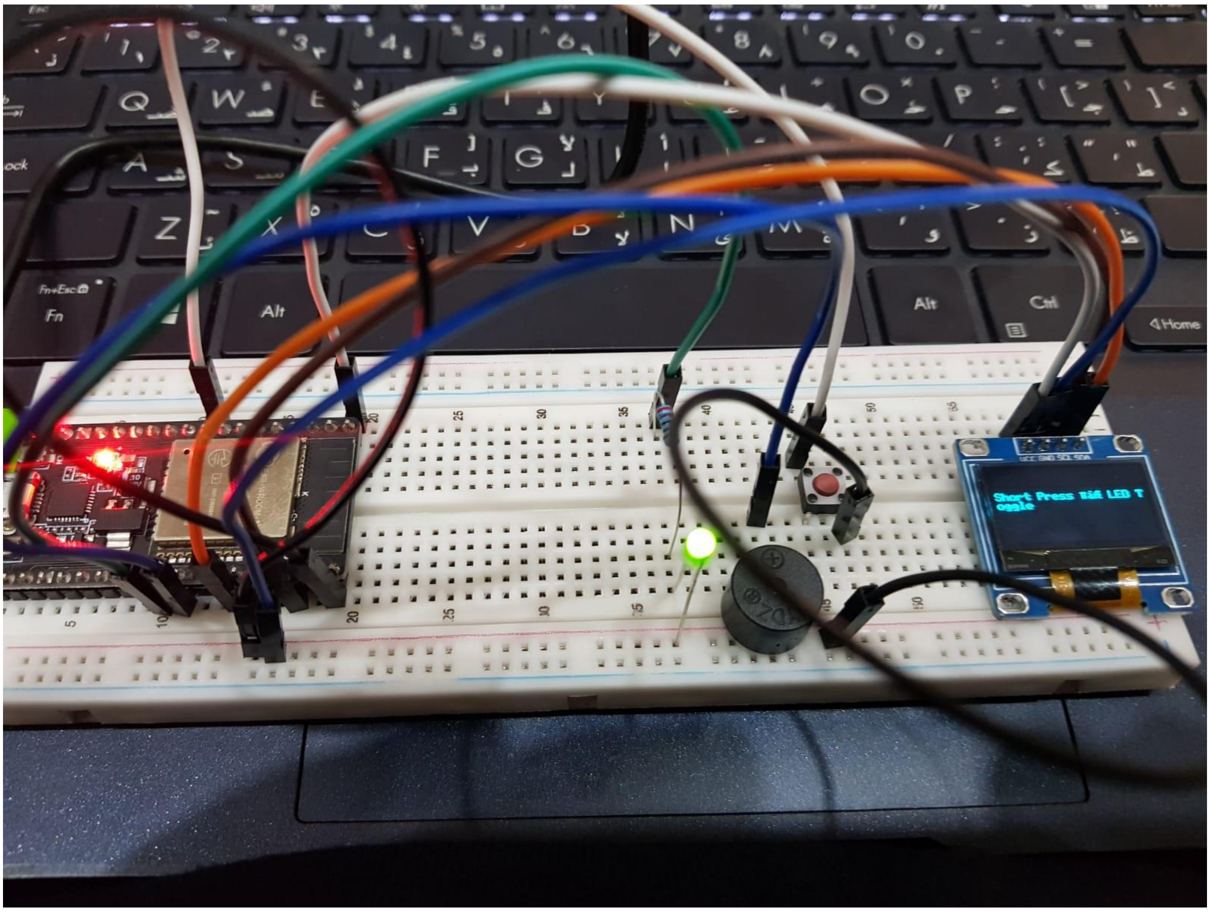
```
{
  "version": 1,
  "author": "SALMAN ASHRAF",
  "editor": "wokwi",
  "parts": [
    { "type": "board-esp32-devkit-c-v4", "id": "esp", "top": -134.4, "left": 168.04, "attrs": {} },
    {
      "type": "wokwi-led",
      "id": "led1",
      "top": 164.8,
      "left": -82.2,
      "rotate": 180,
      "attrs": { "color": "magenta" }
    },
    {
      "type": "wokwi-resistor",
      "id": "r1",
      "top": 147.4,
      "left": 325.55,
      "rotate": 270,
      "attrs": { "value": "420" }
    },
    {
      "type": "wokwi-pushbutton",
      "id": "btn1",
      "top": -0.6,
      "left": -198.6,
      "rotate": 90,
      "attrs": { "color": "green", "xray": "1" }
    },
    {
      "type": "wokwi-buzzer",
      "id": "bz1",
      "top": 117.6,
      "left": 577.8,
      "attrs": { "volume": "0.1" }
    },
    {
      "type": "board-ssd1306",
      "id": "oled1",
      "top": -304.06,
      "left": 211.43,
      "attrs": { "i2cAddress": "0x3c" }
    }
```

```
  ],
  "connections": [
    [ "esp:TX", "$serialMonitor:RX", "", [] ],
    [ "esp:RX", "$serialMonitor:TX", "", [] ],
    [ "btn1:1.l", "esp:25", "green", [ "v-28.8", "h96", "v-28.8" ] ],
    [ "led1:A", "r1:1", "green", [ "v0" ] ],
    [ "btn1:2.r", "esp:GND.1", "black", [ "h57.8", "v-28.6" ] ],
    [ "bz1:2", "esp:18", "green", [ "v0" ] ],
    [ "bz1:1", "esp:GND.2", "black", [ "v0" ] ],
    [ "led1:C", "esp:GND.3", "black", [ "v0" ] ],
    [ "r1:2", "esp:5", "green", [ "h-124.8", "v-30" ] ],
    [ "oled1:GND", "esp:GND.2", "black", [ "h163.2", "v134.4" ] ],
    [ "oled1:VCC", "esp:3V3", "red", [ "h-76.65", "v134.4" ] ],
    [ "oled1:SCL", "esp:22", "green", [ "h230.7", "v153.6" ] ],
    [ "oled1:SDA", "esp:21", "green", [ "h144.07", "v182.4" ] ]
  ],
  "dependencies": {}
}
```

## JSON DIAGRAM



## PRACTICAL WORK



task2 running.mp4

IMPORTANT NOTE:

- **We need to add this library for running our desired code successfully**

```
•  lib_deps =
•    adafruit/Adafruit GFX Library@^1.12.3
•    adafruit/Adafruit SSD1306@^2.5.15
```

- **Also we need a wokwi.toml so that we can run our JSON diagram on VS code**