National Textile University, Faisalabad

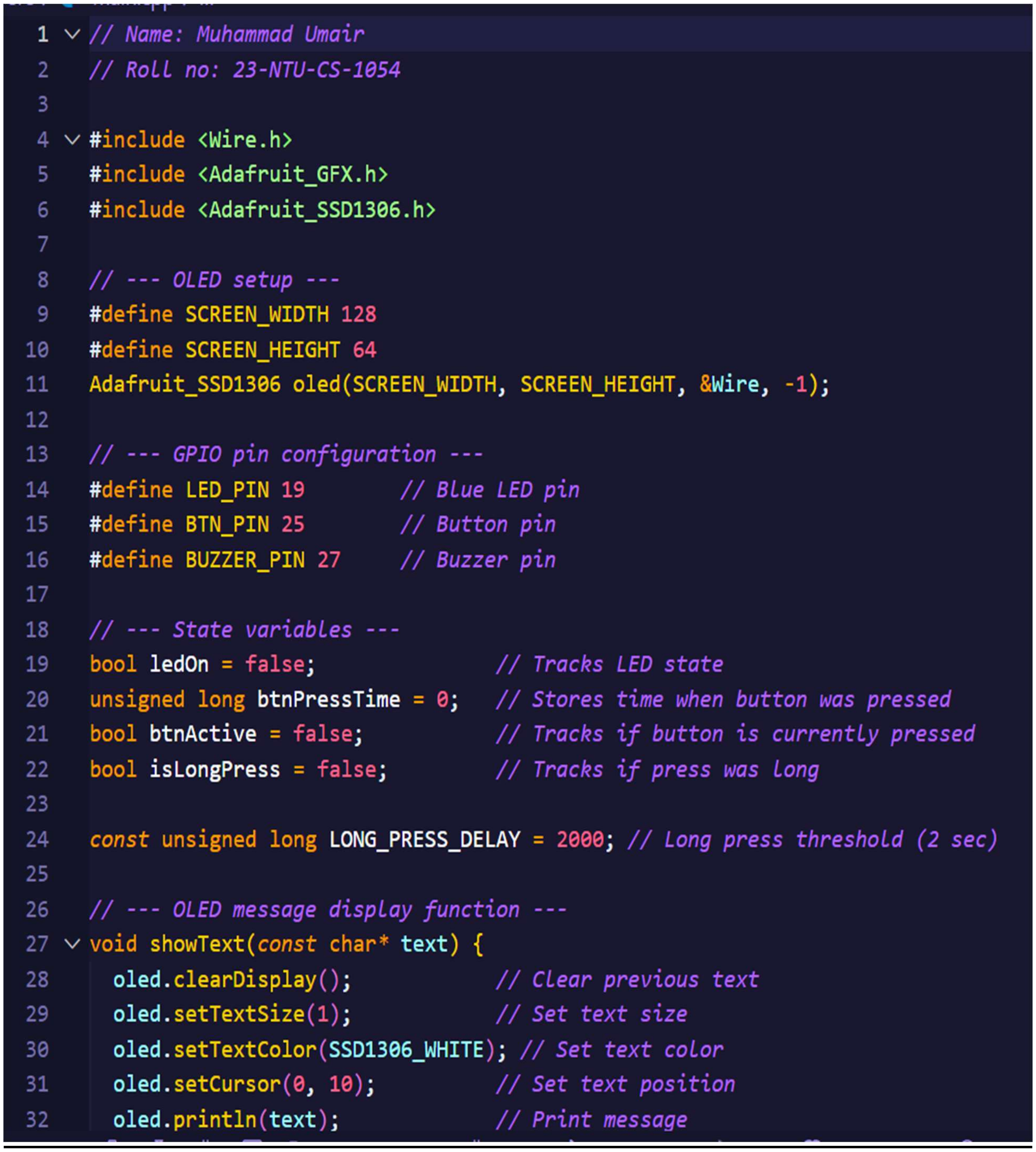
Department of Computer Science

Name:	MUHAMMAD UMAIR
Class:	BSCS 5th-B
Registration No:	23-NTU-CS-1054
Assignment No:	# 1 Task B
Course Name:	IoT Embedded Systems
Submitted To:	Nasir Mehmood
Submission Date:	26 - 10 - 2025

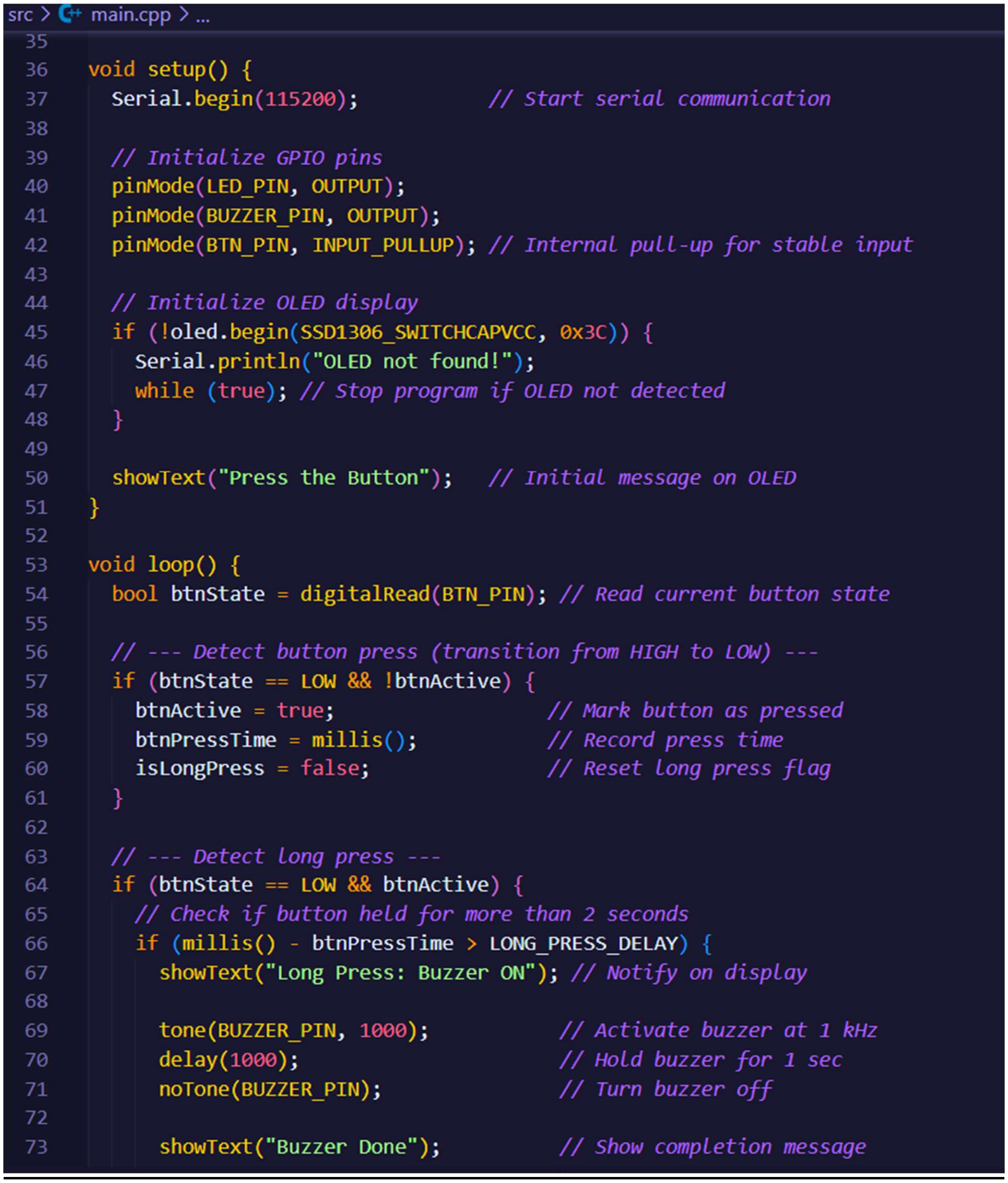
Assignment No 1

Task B

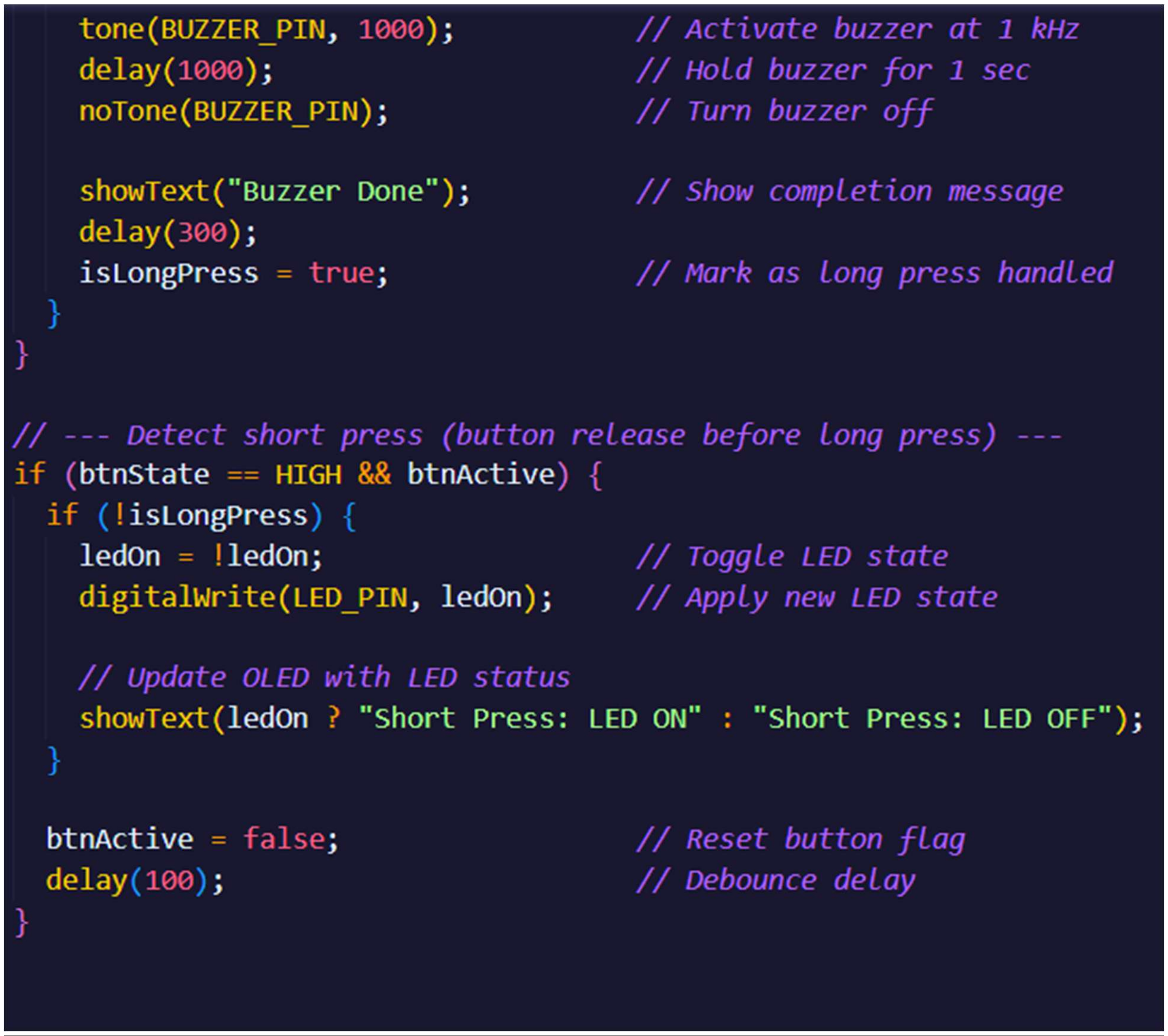
Code



```
// Name: Muhammad Umair
// Roll no: 23-NTU-CS-1054

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

// --- OLED setup ---
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 oled(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

// --- GPIO pin configuration ---
#define LED_PIN 19        // Blue LED pin
#define BTN_PIN 25        // Button pin
#define BUZZER_PIN 27     // Buzzer pin

// --- State variables ---
bool ledOn = false;               // Tracks LED state
unsigned long btnPressTime = 0;   // Stores time when button was pressed
bool btnActive = false;           // Tracks if button is currently pressed
bool isLongPress = false;         // Tracks if press was long

const unsigned long LONG_PRESS_DELAY = 2000; // Long press threshold (2 sec)

// --- OLED message display function ---
void showText(const char* text) {
  oled.clearDisplay();            // Clear previous text
  oled.setTextSize(1);            // Set text size
  oled.setTextColor(SSD1306_WHITE); // Set text color
  oled.setCursor(0, 10);          // Set text position
  oled.println(text);             // Print message
```


src > main.cpp > ...

```cpp

void setup() {
  Serial.begin(115200);           // Start serial communication

  // Initialize GPIO pins
  pinMode(LED_PIN, OUTPUT);
  pinMode(BUZZER_PIN, OUTPUT);
  pinMode(BTN_PIN, INPUT_PULLUP); // Internal pull-up for stable input

  // Initialize OLED display
  if (!oled.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println("OLED not found!");
    while (true); // Stop program if OLED not detected
  }

  showText("Press the Button");   // Initial message on OLED
}

void loop() {
  bool btnState = digitalRead(BTN_PIN); // Read current button state

  // --- Detect button press (transition from HIGH to LOW) ---
  if (btnState == LOW && !btnActive) {
    btnActive = true;                 // Mark button as pressed
    btnPressTime = millis();          // Record press time
    isLongPress = false;              // Reset long press flag
  }

  // --- Detect long press ---
  if (btnState == LOW && btnActive) {
    // Check if button held for more than 2 seconds
    if (millis() - btnPressTime > LONG_PRESS_DELAY) {
      showText("Long Press: Buzzer ON"); // Notify on display

      tone(BUZZER_PIN, 1000);           // Activate buzzer at 1 kHz
      delay(1000);                      // Hold buzzer for 1 sec
      noTone(BUZZER_PIN);               // Turn buzzer off

      showText("Buzzer Done");          // Show completion message
```



```
    tone(BUZZER_PIN, 1000);           // Activate buzzer at 1 kHz
    delay(1000);                      // Hold buzzer for 1 sec
    noTone(BUZZER_PIN);               // Turn buzzer off

    showText("Buzzer Done");          // Show completion message
    delay(300);
    isLongPress = true;               // Mark as long press handled
  }
}

// --- Detect short press (button release before long press) ---
if (btnState == HIGH && btnActive) {
  if (!isLongPress) {
    ledOn = !ledOn;                   // Toggle LED state
    digitalWrite(LED_PIN, ledOn);     // Apply new LED state

    // Update OLED with LED status
    showText(ledOn ? "Short Press: LED ON" : "Short Press: LED OFF");
  }

  btnActive = false;                  // Reset button flag
  delay(100);                         // Debounce delay
}
```

Brief Explanation about Code:

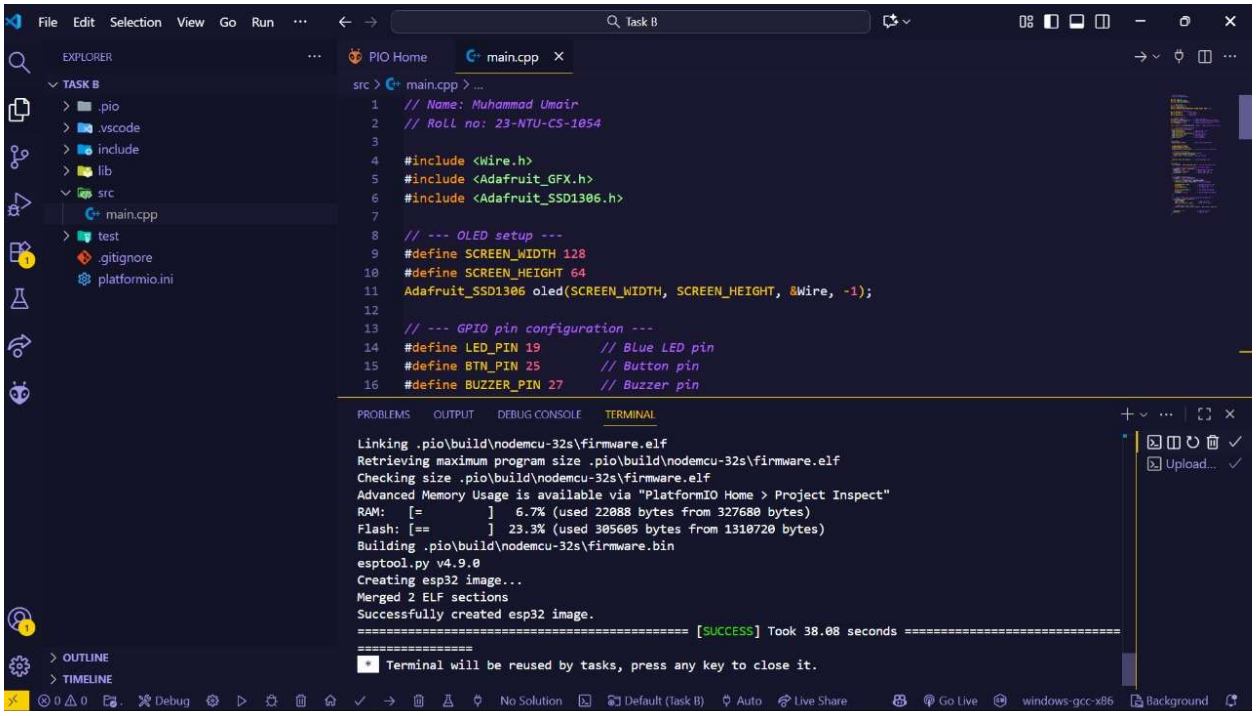
This program is made for an ESP32 (or Arduino) that uses a yellow LED, a buzzer, and an OLED display. When you power it on, the OLED screen shows a “Ready” message, meaning the system is waiting for your input.

There’s a blue button that controls everything:

- If you press it briefly, the LED switches between ON and OFF, and the display updates to show its current state.
- But if you hold the button for more than two seconds, it’s detected as a long press — the buzzer will turn on for one second while the screen shows “Buzzer ON.” After that, the display updates again to confirm that the buzzer has turned off.

In short, the program gives you both visual and sound feedback depending on how long you press the button.

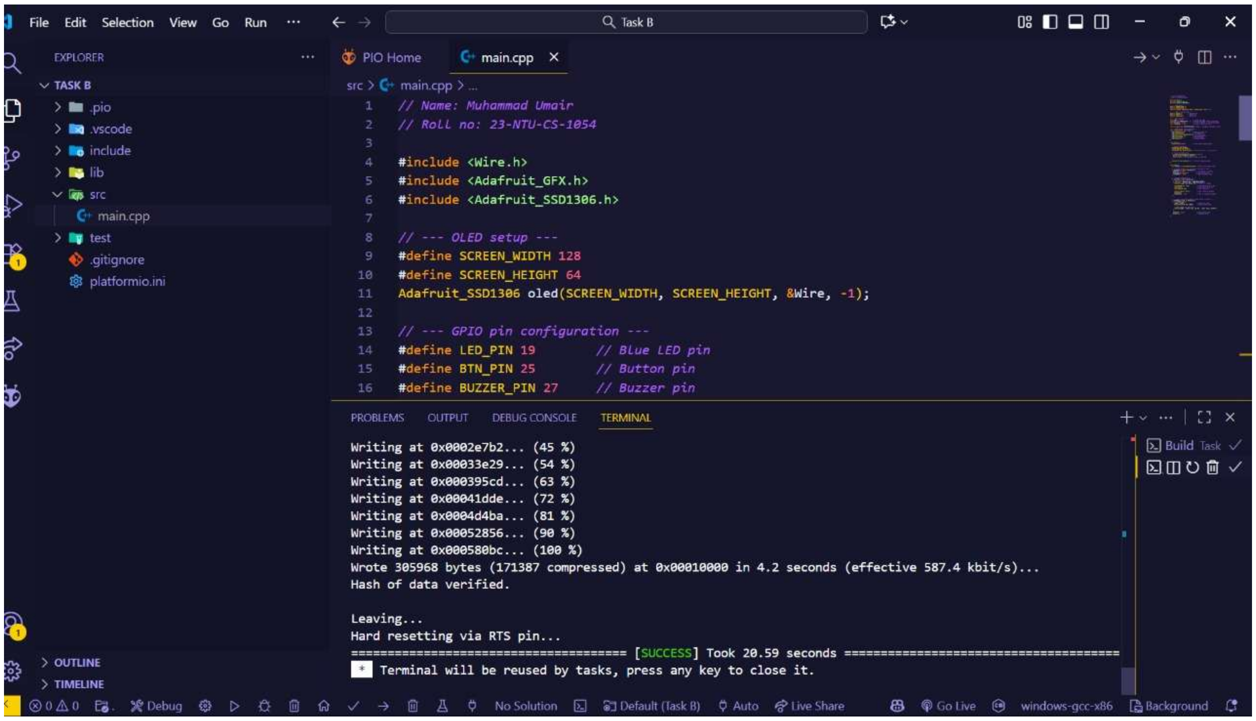
Build Output



```cpp
// Name: Muhammad Umair
// Roll no: 23-NTU-CS-1054

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

// --- OLED setup ---
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 oled(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

// --- GPIO pin configuration ---
#define LED_PIN 19        // Blue LED pin
#define BTN_PIN 25        // Button pin
#define BUZZER_PIN 27     // Buzzer pin
```

```
Linking .pio\build\nodemcu-32s\firmware.elf
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 22088 bytes from 327680 bytes)
Flash: [==        ]  23.3% (used 305605 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 38.08 seconds ============================
================
 *  Terminal will be reused by tasks, press any key to close it.
```

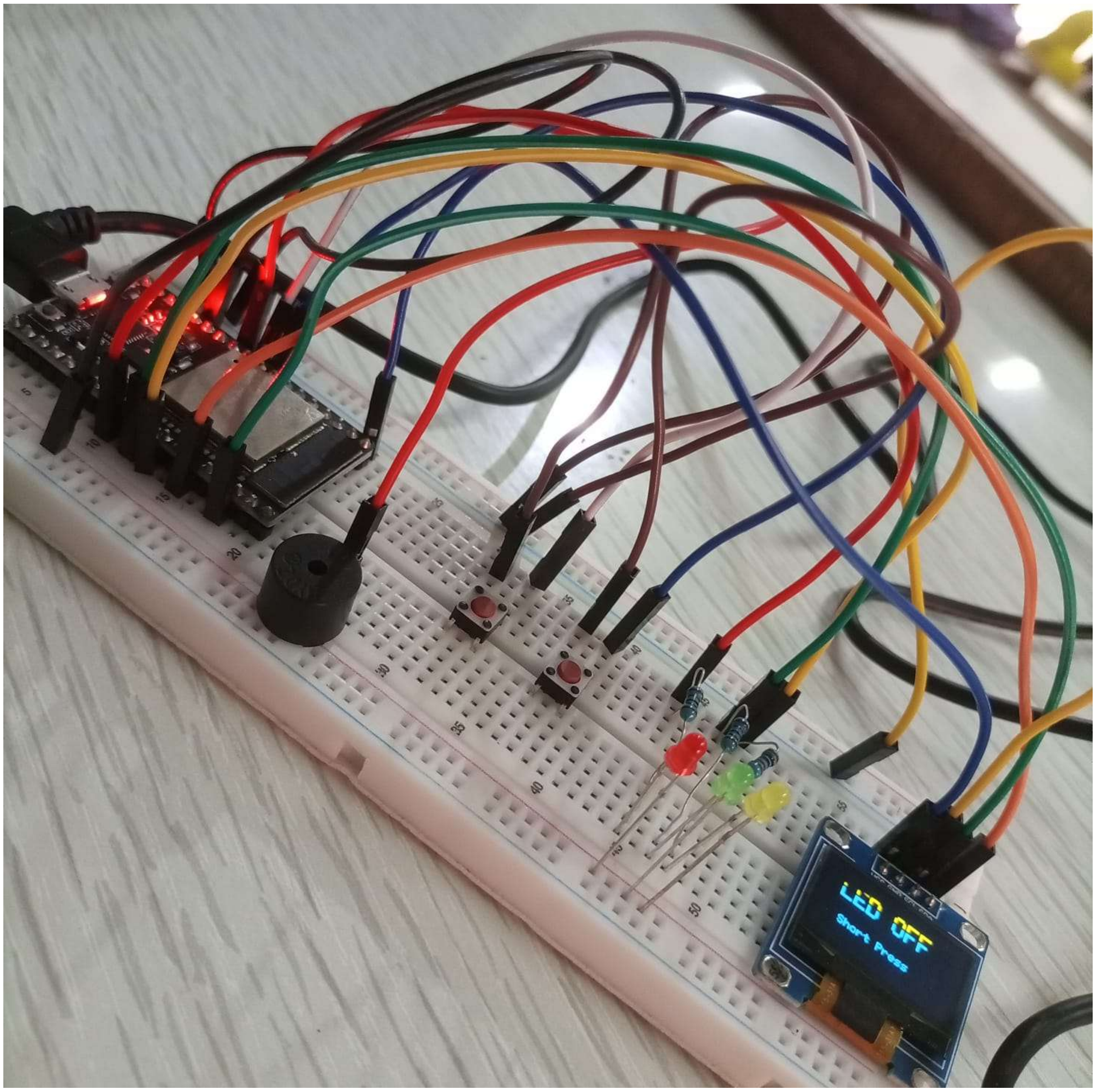
Upload Output



```cpp
// Name: Muhammad Umair
// Roll no: 23-NTU-CS-1054

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

// --- OLED setup ---
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 oled(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

// --- GPIO pin configuration ---
#define LED_PIN 19        // Blue LED pin
#define BTN_PIN 25        // Button pin
#define BUZZER_PIN 27     // Buzzer pin
```

```
Writing at 0x0002e7b2... (45 %)
Writing at 0x00033e29... (54 %)
Writing at 0x000395cd... (63 %)
Writing at 0x00041dde... (72 %)
Writing at 0x0004d4ba... (81 %)
Writing at 0x00052856... (90 %)
Writing at 0x000580bc... (100 %)
Wrote 305968 bytes (171387 compressed) at 0x00010000 in 4.2 seconds (effective 587.4 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
====================================== [SUCCESS] Took 20.59 seconds ======================================
 *  Terminal will be reused by tasks, press any key to close it.
```

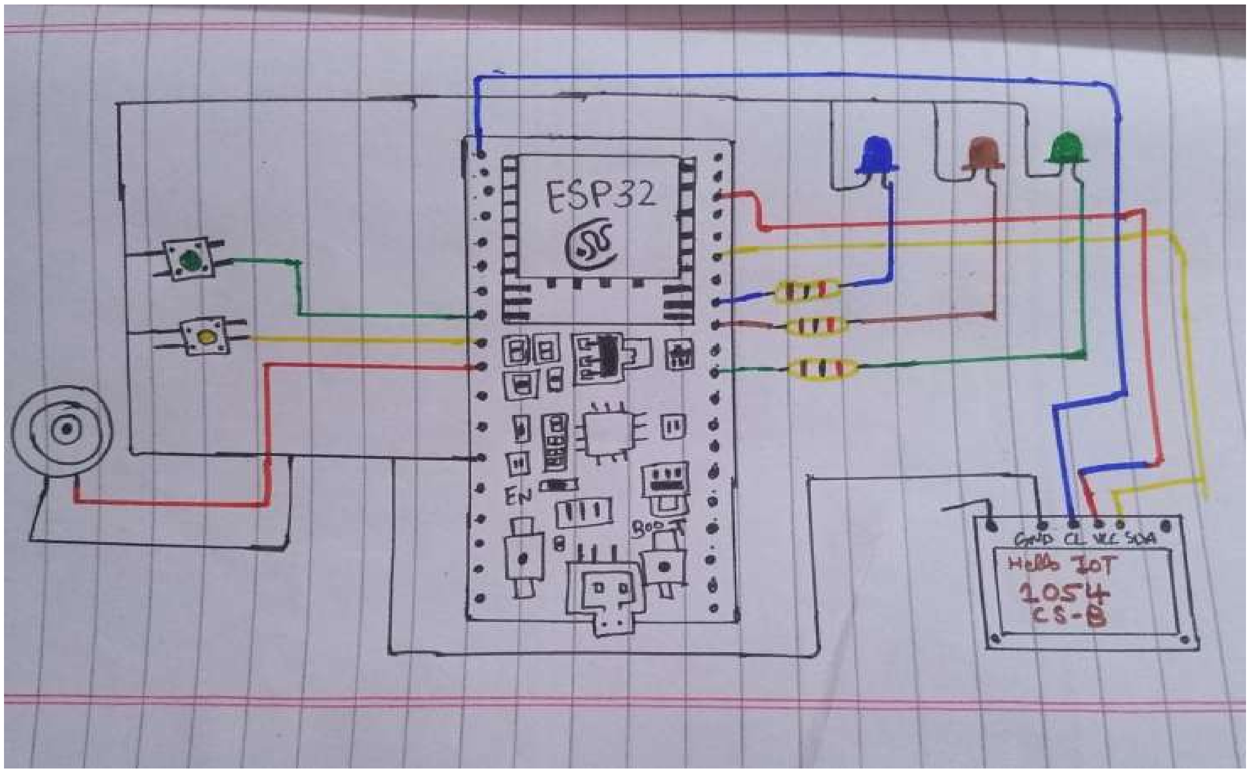
Hardware Picture



Pin Map

Pin No	Name	Function	Use Case
GND .2	Ground	Common Ground	For all LEDs, Buzzer, Buttons, OLED
25	GPIO 25	Pin for Blue Button	Output for Blue Button (Modebtn)
26	GPIO 26	Pin for White Button	Output for White Button (Resetbtn)
27	GPIO 27	Pin for Buzzer	Output for Buzzer
3v3	Power	3.3V output power	OLED VCC
22	GPIO 22	I2C SCL	OLED SCL
21	GPIO 21	I2C SDA	OLED SDA
19	GPIO 19	Pin for Yellow LED	Output for Yellow LED
18	GPIO 18	Pin for Green LED	Output for Green LED
17	GPIO 17	Pin for Red LED	Output for Red LED

Sketch


ESP32
EN
BOOT
GND CL VCC SDA
Hello IOT
1054
CS-B