🧠 Embedded IoT Systems – Assignment 1

Name: Saad Faisal
Reg No: 23-NTU-CS-1281
Course: Embedded IoT Systems
Instructor: Sir Nasir Mehmood
Date: 26-Oct-2025

🧭 Overview

This repository contains **two ESP32-based IoT projects** created for the *Embedded IoT Systems* course.
Both were built and verified on the **Wokwi Simulator**, focusing on **hardware control**, **input handling**, and **OLED display integration**.

🔷 Task 1 – Smart Multi-Device Controller using ESP32

📘 Description

This project demonstrates control of multiple components — **LEDs, push buttons, buzzer, and OLED display** — through the ESP32.
Each button toggles devices and the OLED screen updates live to display the current system state.

🧩 Components

- ESP32 DevKitC V4
- OLED Display 128x64 (I2C)
- LEDs × 3
- Push Buttons × 2
- Buzzer × 1
- 420Ω Resistors × 3
- Jumper Wires

⚙️ Pin Mapping

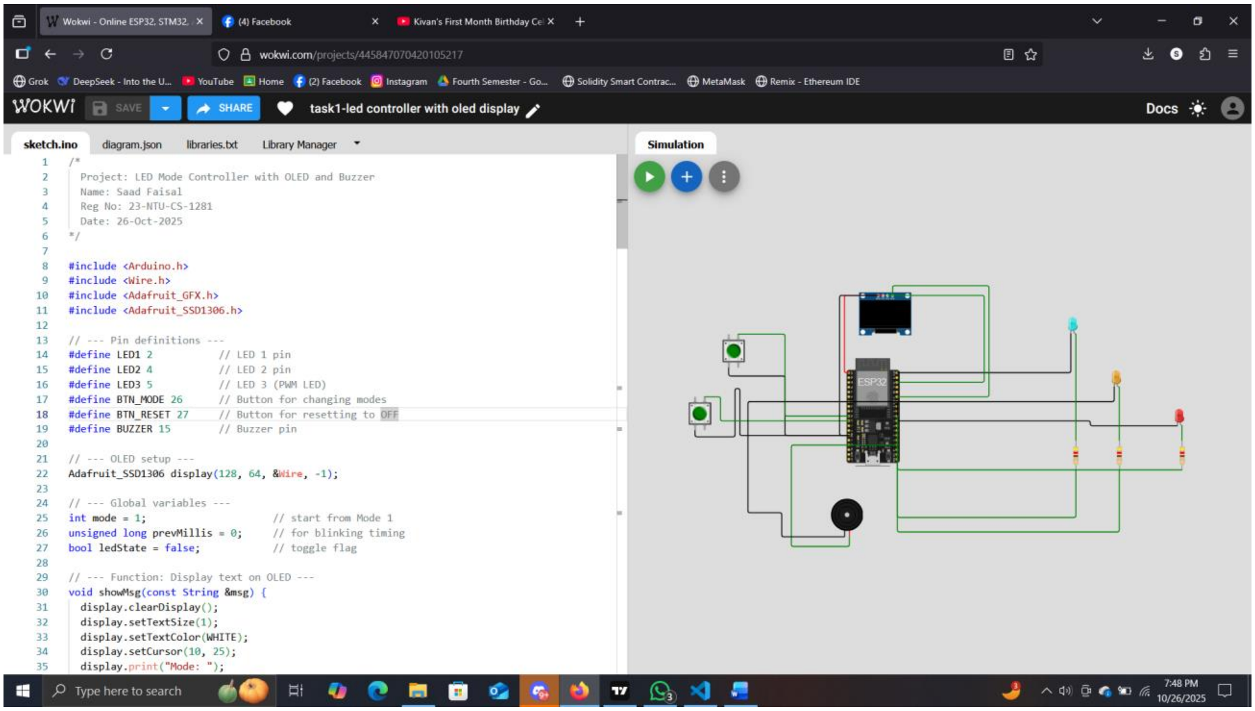
Component	Label	GPIO Pin
LED 1	D2	2
LED 2	D4	4
LED 3	D5	5
Button 1	D26	26
Button 2	D27	27
Buzzer	D15	15
OLED SDA	SDA	21
OLED SCL	SCL	22

▶️ Simulation Link

🔗 [View on Wokwi]

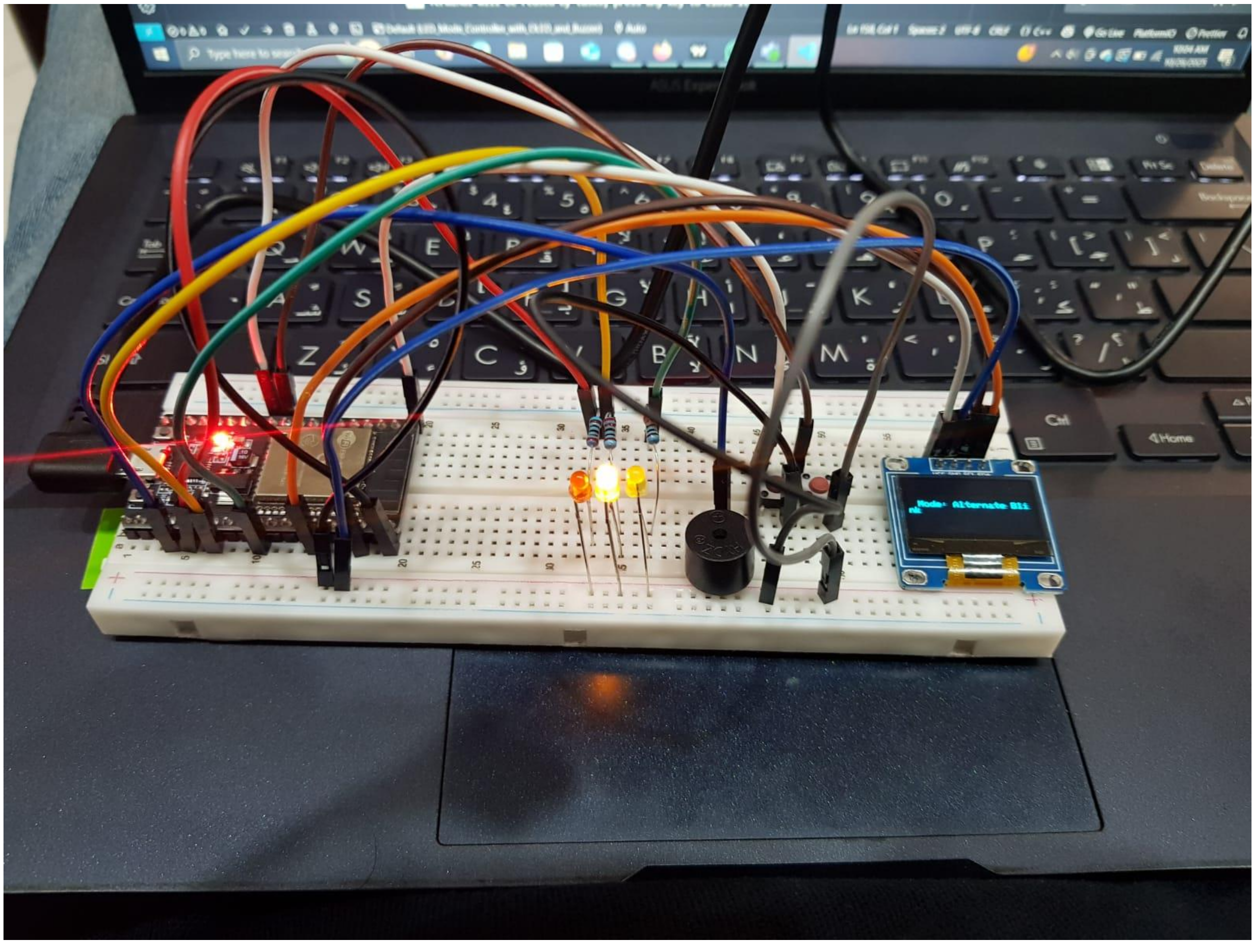
📷 Screenshots

WOKWI TASK 1:

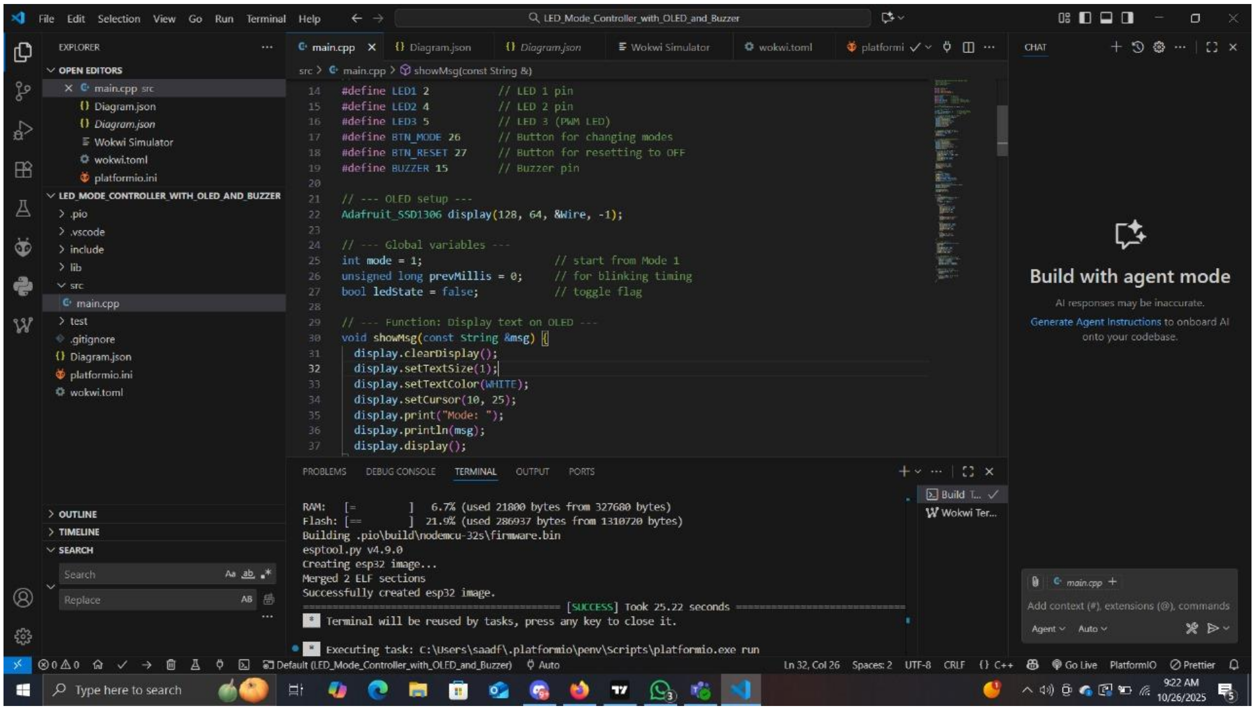


```
/*
  Project: LED Mode Controller with OLED and Buzzer
  Name: Saad Faisal
  Reg No: 23-NTU-CS-1281
  Date: 26-Oct-2025
*/

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

// --- Pin definitions ---
#define LED1 2          // LED 1 pin
#define LED2 4          // LED 2 pin
#define LED3 5          // LED 3 (PWM LED)
#define BTN_MODE 26     // Button for changing modes
#define BTN_RESET 27    // Button for resetting to OFF
#define BUZZER 15       // Buzzer pin

// --- OLED setup ---
Adafruit_SSD1306 display(128, 64, &Wire, -1);

// --- Global variables ---
int mode = 1;                   // start from Mode 1
unsigned long prevMillis = 0;   // for blinking timing
bool ledState = false;          // toggle flag

// --- Function: Display text on OLED ---
void showMsg(const String &msg) {
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(10, 25);
  display.print("Mode: ");
```

HARDWARE TASK 1:



OUTPUT TASK 1:



```cpp
#define LED1 2          // LED 1 pin
#define LED2 4          // LED 2 pin
#define LED3 5          // LED 3 (PWM LED)
#define BTN_MODE 26     // Button for changing modes
#define BTN_RESET 27    // Button for resetting to OFF
#define BUZZER 15       // Buzzer pin

// --- OLED setup ---
Adafruit_SSD1306 display(128, 64, &Wire, -1);

// --- Global variables ---
int mode = 1;                   // start from Mode 1
unsigned long prevMillis = 0;   // for blinking timing
bool ledState = false;          // toggle flag

// --- Function: Display text on OLED ---
void showMsg(const String &msg) {
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(10, 25);
  display.print("Mode: ");
  display.println(msg);
  display.display();
```

```
RAM:   [=         ]   6.7% (used 21800 bytes from 327680 bytes)
Flash: [==        ]  21.9% (used 286937 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 25.22 seconds ========================================
 *  Terminal will be reused by tasks, press any key to close it.

 *  Executing task: C:\Users\saadf\.platformio\penv\Scripts\platformio.exe run
```

AI responses may be inaccurate.

Generate Agent Instructions to onboard AI onto your codebase.

🔷 Task 2 – Button Press Duration Detection (Short vs Long Press)

📘 Description

This project detects **short and long button presses** using the ESP32.
A **short press** toggles an LED, while a **long press** triggers a buzzer. The OLED screen displays the detected press type.

🧩 Components

- ESP32 DevKitC V4
- OLED Display 128x64 (I2C)
- LED × 1
- Push Button × 1
- Buzzer × 1
- Resistor × 1

⚙️ Pin Mapping

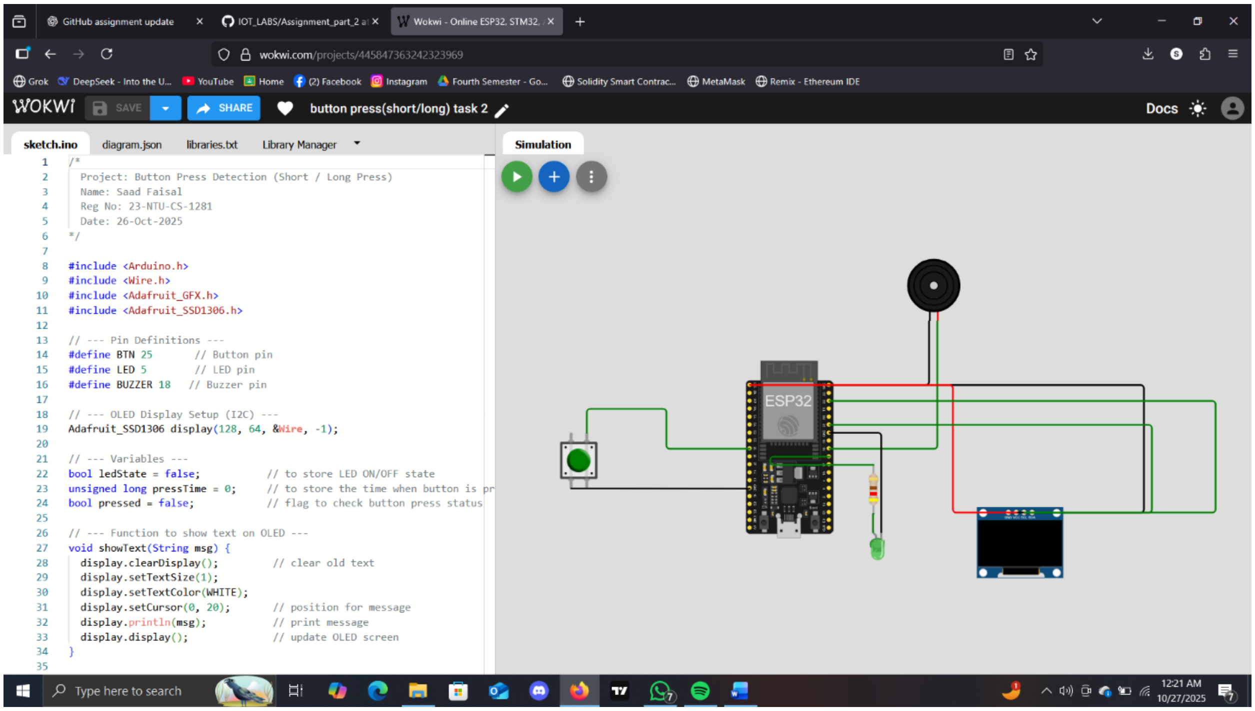
Component	Label	GPIO Pin
LED	D5	5
Button	D25	25
Buzzer	D18	18
OLED SDA	SDA	21
OLED SCL	SCL	22

▶️ Simulation Link

🔗 [View on Wokwi]

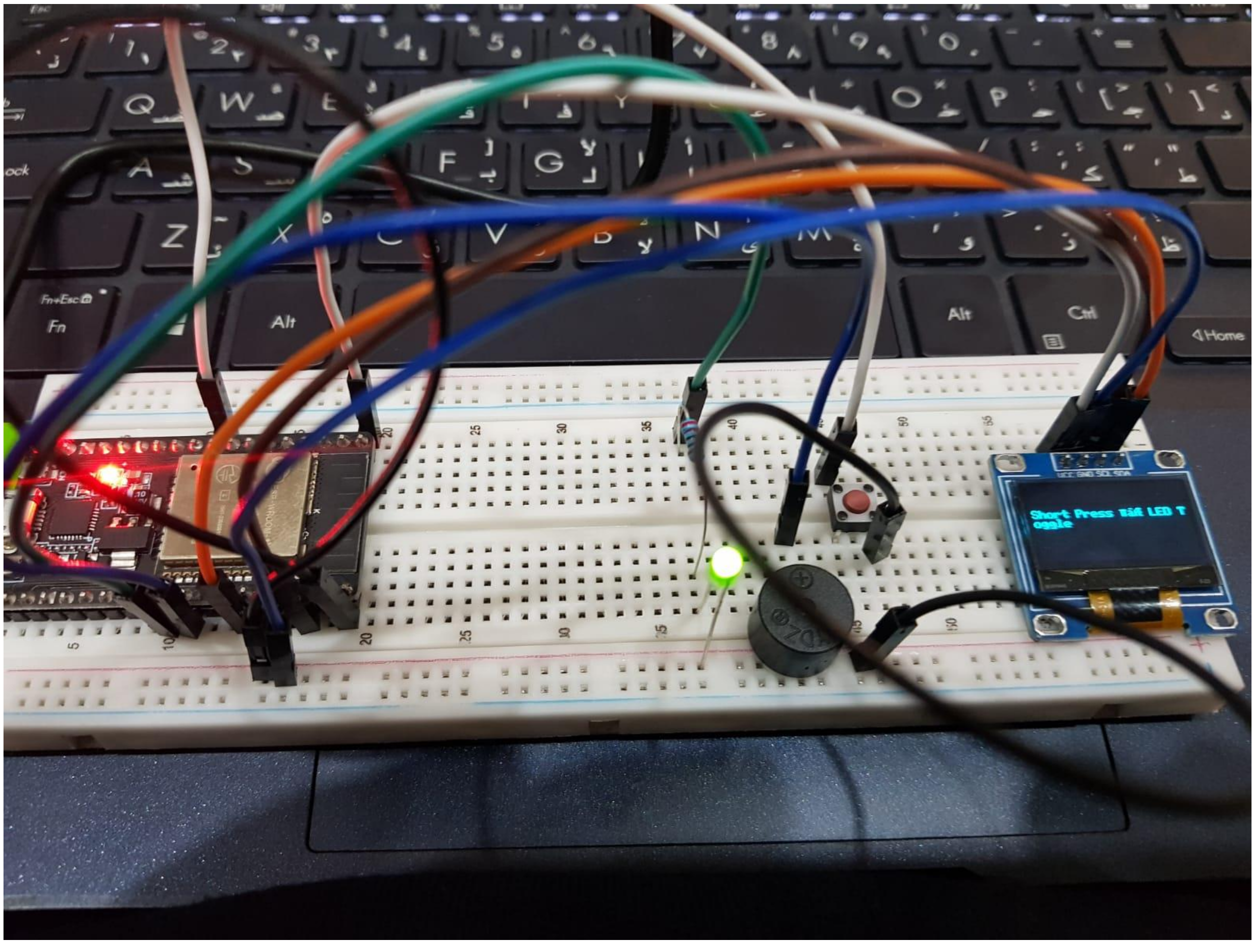
📷 Screenshots

WOKWI TASK 2:

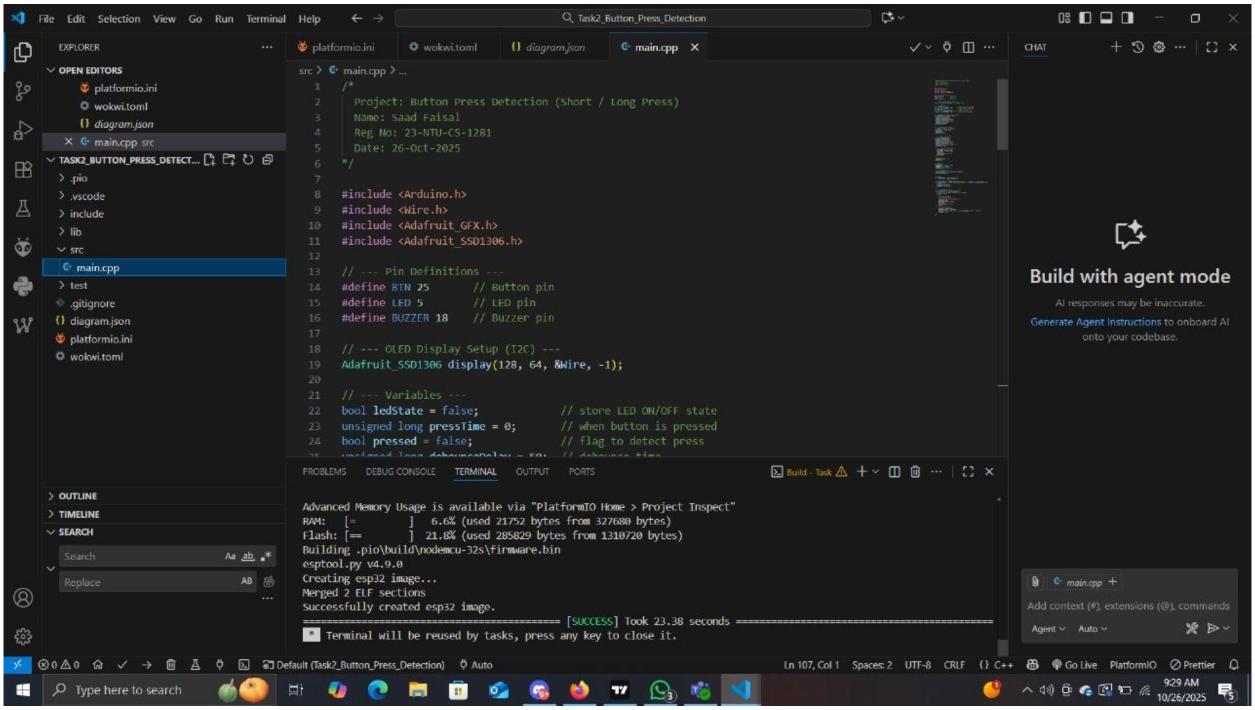


```
/*
  Project: Button Press Detection (Short / Long Press)
  Name: Saad Faisal
  Reg No: 23-NTU-CS-1281
  Date: 26-Oct-2025
*/

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

// --- Pin Definitions ---
#define BTN 25       // Button pin
#define LED 5        // LED pin
#define BUZZER 18   // Buzzer pin

// --- OLED Display Setup (I2C) ---
Adafruit_SSD1306 display(128, 64, &Wire, -1);

// --- Variables ---
bool ledState = false;            // to store LED ON/OFF state
unsigned long pressTime = 0;      // to store the time when button is pr
bool pressed = false;             // flag to check button press status

// --- Function to show text on OLED ---
void showText(String msg) {
  display.clearDisplay();          // clear old text
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(0, 20);        // position for message
  display.println(msg);            // print message
  display.display();               // update OLED screen
}

```

HARDWARE TASK 2:



OUTPUT TASK 2:



```cpp
/*
  Project: Button Press Detection (Short / Long Press)
  Name: Saad Faisal
  Reg No: 23-NTU-CS-1281
  Date: 26-Oct-2025
*/

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

// --- Pin Definitions ---
#define BTN 25       // Button pin
#define LED 5        // LED pin
#define BUZZER 18    // Buzzer pin

// --- OLED Display Setup (I2C) ---
Adafruit_SSD1306 display(128, 64, &Wire, -1);

// --- Variables ---
bool ledState = false;              // store LED ON/OFF state
unsigned long pressTime = 0;        // when button is pressed
bool pressed = false;               // flag to detect press
unsigned long debounceDelay = 50;   // debounce time
```

```
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM:   [=         ]   6.6% (used 21752 bytes from 327680 bytes)
Flash: [==        ]  21.8% (used 285829 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 23.38 seconds ===========================================
 *  Terminal will be reused by tasks, press any key to close it.
```

⚙️ Features

✅ Real-time OLED display feedback

✅ Button-based LED and buzzer control

✅ Detection of both short and long press durations

✅ Fully simulated on **Wokwi IoT Simulator**

🎯 Learning Outcomes

- Interfaced ESP32 with **I2C-based OLED**
- Implemented **GPIO handling and debounce logic**
- Detected button press duration using **millis()**
- Strengthened understanding of **embedded hardware integration**

📁 Repository Structure

🧠 Author

Saad Faisal

📧 saadfaisal065@gmail.com

💻 *Computer Science Student – NTU Faisalabad*

⭐ *If you found this project helpful, consider starring the repository!*