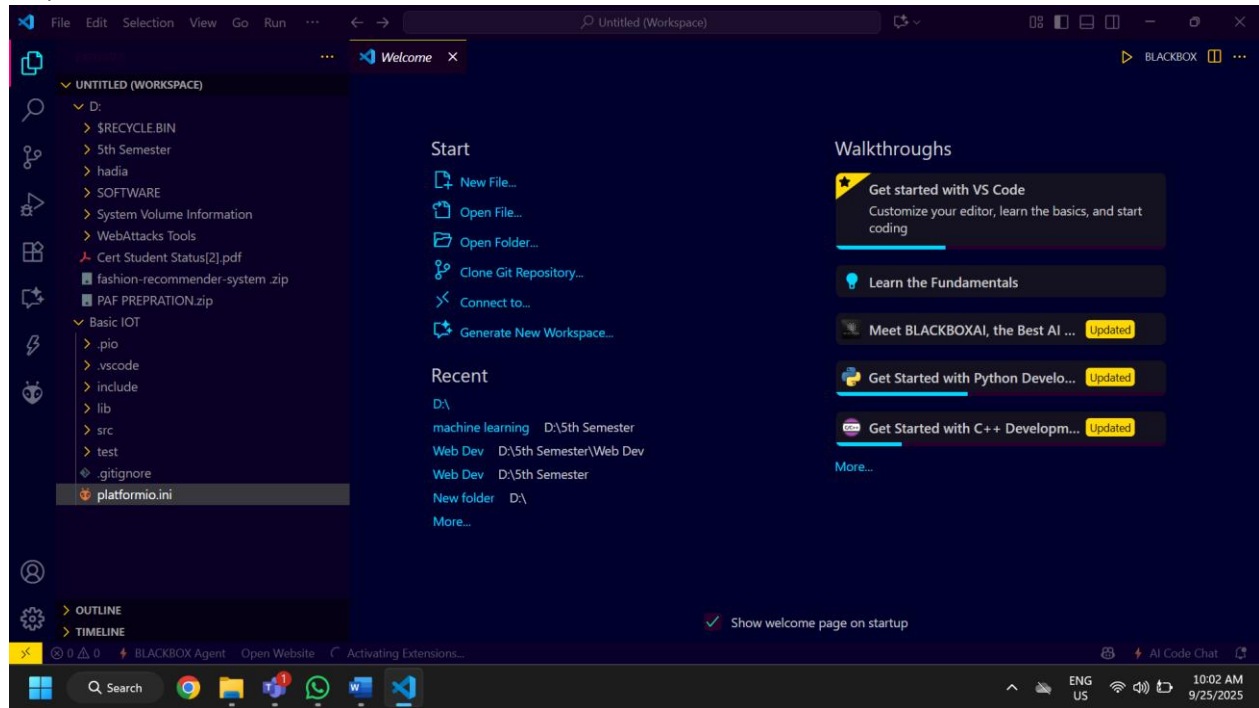




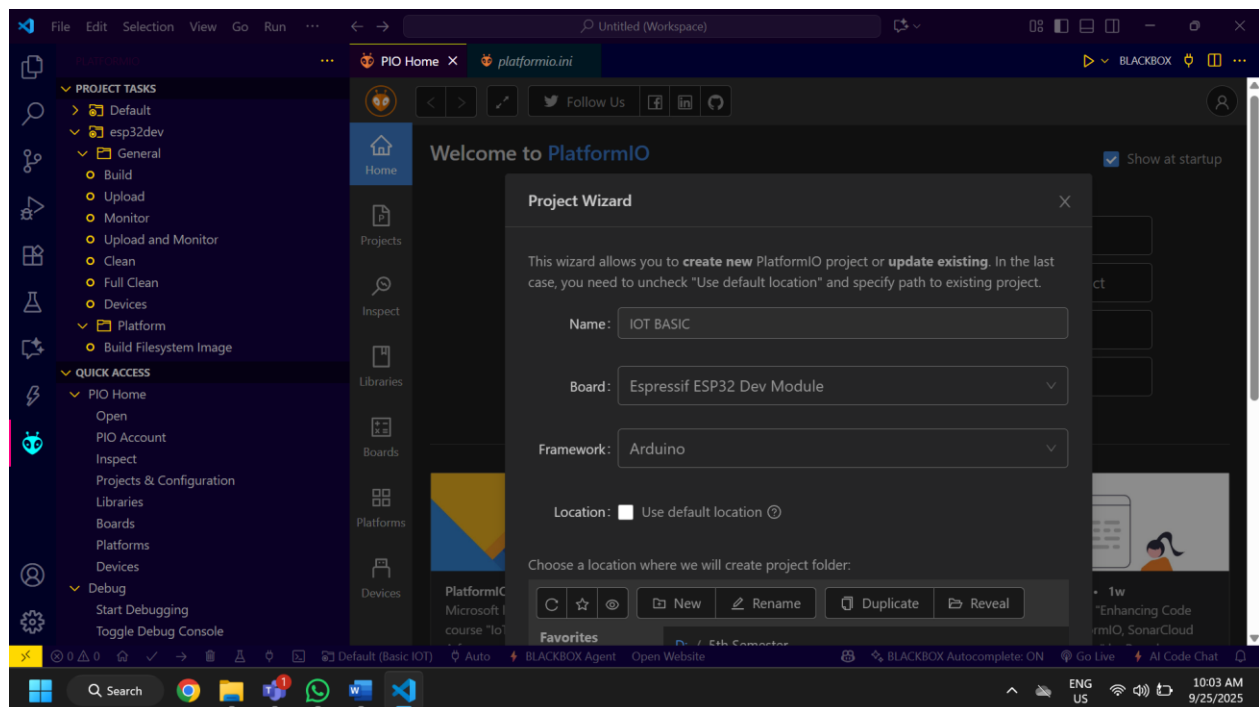
<b>Group Member:</b>	<b>Ibtisam Butt</b>
<b>Registration No:</b>	23-NTU-CS-1269
<b>Class:</b>	BSAI-5 <sup>th</sup>
<b>Course Name:</b>	Embedded IOT System
<b>Submitted To:</b>	<i>Nasir Mehmood</i>
<b>Submission Date:</b>	23-09-2025

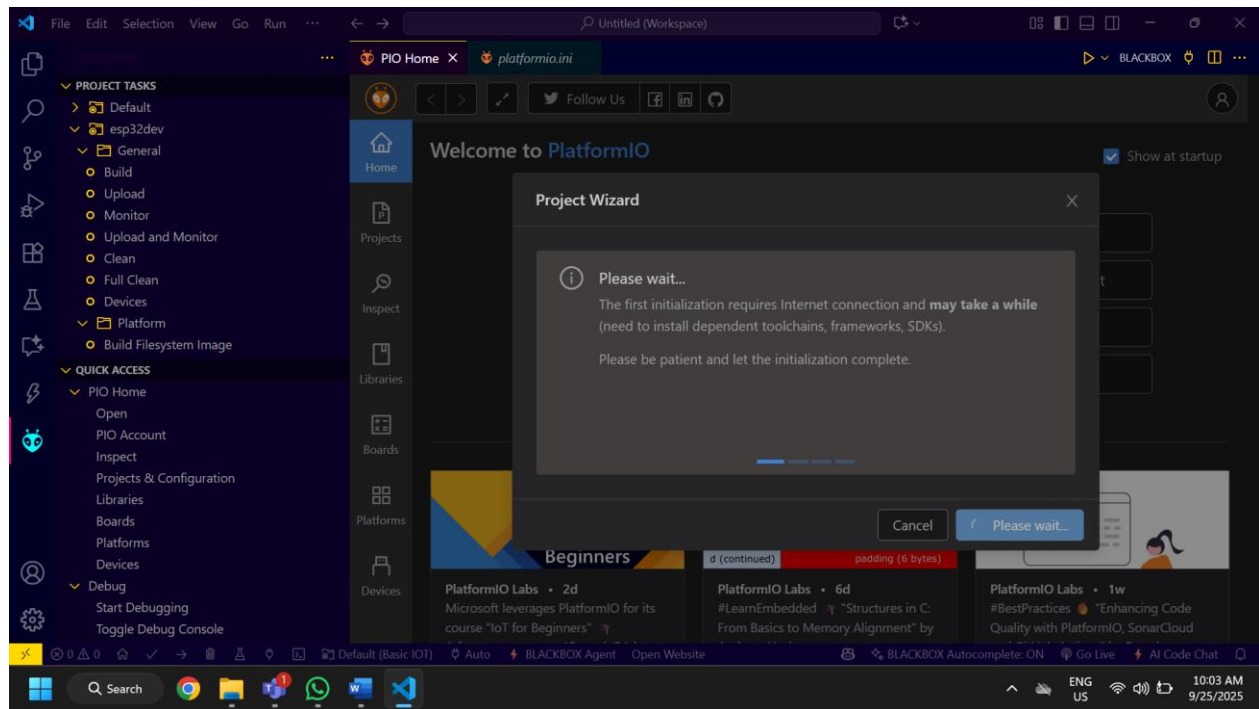
# Embedded IOT System

## 1) Open VS Code

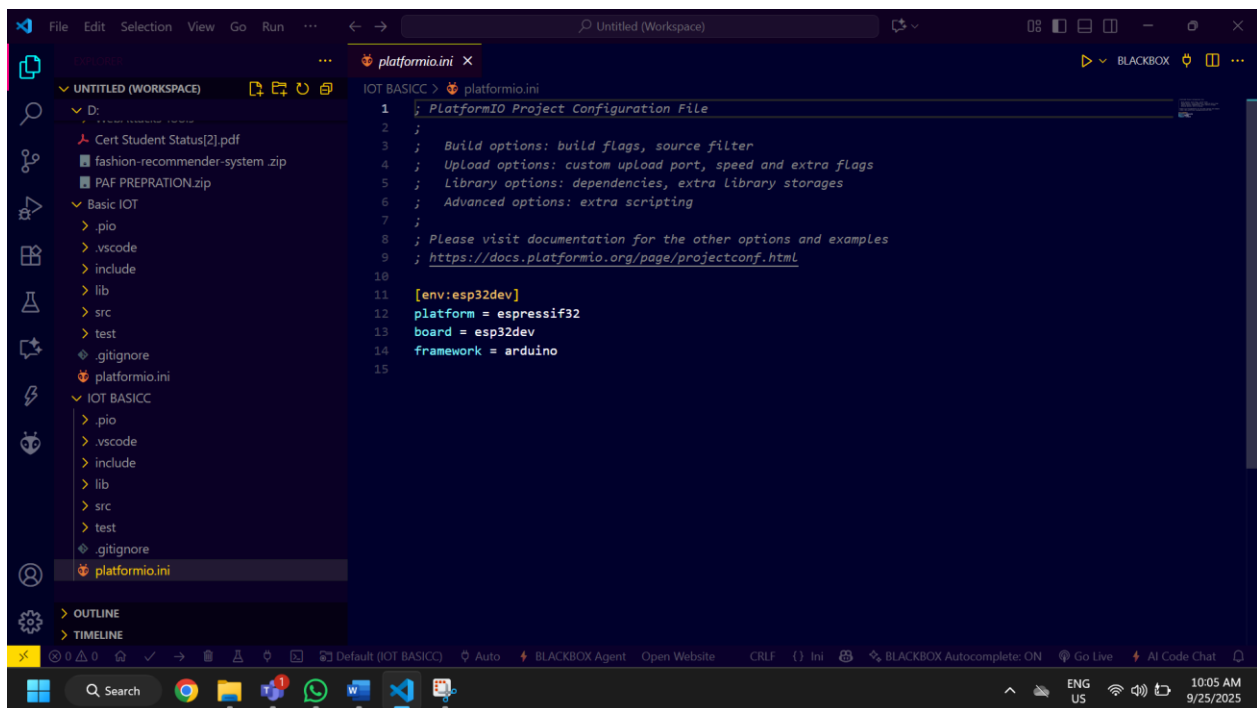


## 2) Create the project





### 3) platformio.ini (verify this)



```
platformio.ini X
IOT BASICCC > platformio.ini
1 ; PlatformIO Project Configuration File
2 ;
3 ; Build options: build flags, source filter
4 ; Upload options: custom upload port, speed and extra flags
5 ; Library options: dependencies, extra library storages
6 ; Advanced options: extra scripting
7 ;
8 ; Please visit documentation for the other options and examples
9 ; https://docs.platformio.org/page/projectconf.html
10
11 [env:esp32dev]
12 platform = espressif32
13 board = esp32dev
14 framework = arduino
15
```

#### 4) Blink code (Arduino framework)

```
File Edit Selection View Go Run ...
Untitled (Workspace)
platformio.ini X main.cpp X
Basic IOT > src > main.cpp > loop()
1 #include <Arduino.h>
2 // Prefer LED_BUILTIN if your board defines it.
3 // If your board doesn't, uncomment the correct pin below.
4 // Common pins:
5 //
6 //
7 //
8 // - ESP32-DevKitC/V1: 2 - Some ESP32-S2/S3: 13 or 48 (varies) - ESP32-C3 DevKitM-1: 8 (often active-
9 // #define LED_PIN 2
10 #ifndef LED_BUILTIN
11 #define LED_BUILTIN 2 // fallback; adjust if your board uses another pin
12 #endif
13 // Set this to true if your LED is active-low (many ESP32-C3 boards)
14 const bool ACTIVE_LOW = false;
15 void setup() {
16 pinMode(LED_BUILTIN, OUTPUT);
17 // Start with LED off (respect active level)
18 digitalWrite(LED_BUILTIN, ACTIVE_LOW ? HIGH : LOW);
19 }
20 void loop() {
21 // toggle
22 static bool on = false;
23 on = !on;
24 digitalWrite(LED_BUILTIN, (on ^ ACTIVE_LOW) ? HIGH : LOW);
25 delay(500); // 0.5s on, 0.5s off
26 }
```

The image shows a VS Code editor window with a project named 'Basic IOT'. The file explorer on the left shows the project structure, including a 'src' directory with 'main.cpp'. The main.cpp file is open in the editor, showing code for an LED. The terminal at the bottom shows the output of a PlatformIO build, including archiving, linking, and memory usage statistics, ending with a successful build message.

```

1  #include <Arduino.h>
2  // Prefer LED_BUILTIN if your board defines it.
3  // If your board doesn't, uncomment the correct pin below.
4  // Common pins:
5  //
6  //
7  //
8  // - ESP32-DevKitC/V1: 2 - Some ESP32-S2/S3: 13 or 48 (varies) - ESP32-C3 DevKitM-1: 8 (often active-
9  // #define LED_PIN 2
10 #ifndef LED_BUILTIN
11 #define LED_BUILTIN 2 // fallback; adjust if your board uses another pin
12 #endif
13 // Set this to true if your LED is active-low (many ESP32-C3 boards)
14 const bool ACTIVE_LOW = false;

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS PlatformIO: Build (Basic IOT) - Task ✓ + ▢ ✕ ... | 🔍

```

Archiving .pio/build/esp32dev/libFrameworkArduino.a
Linking .pio/build/esp32dev/firmware.elf
Retrieving maximum program size .pio/build/esp32dev/firmware.elf
Checking size .pio/build/esp32dev/firmware.elf
Advanced Memory Usage is available via "PlatformIO Home > Project Inspect"
RAM: [== ] 6.4% (used 21112 bytes from 327680 bytes)
Flash: [== ] 18.3% (used 239893 bytes from 1310720 bytes)
Building .pio/build/esp32dev/firmware.bin
esptool.py v4.9.0
Creating esp32 image...
Merged 2 ELF sections
Successfully created esp32 image.
===== [SUCCESS] Took 18.16 seconds =====
Terminal will be reused by tasks, press any key to close it.

```