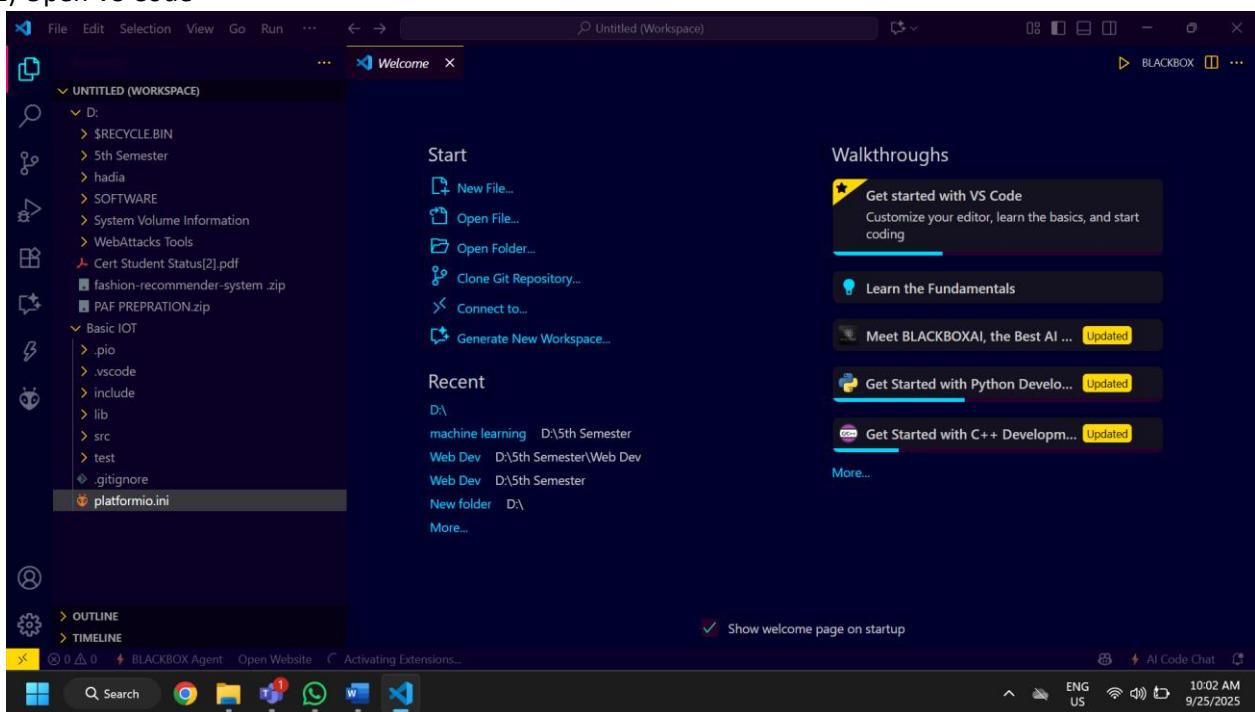




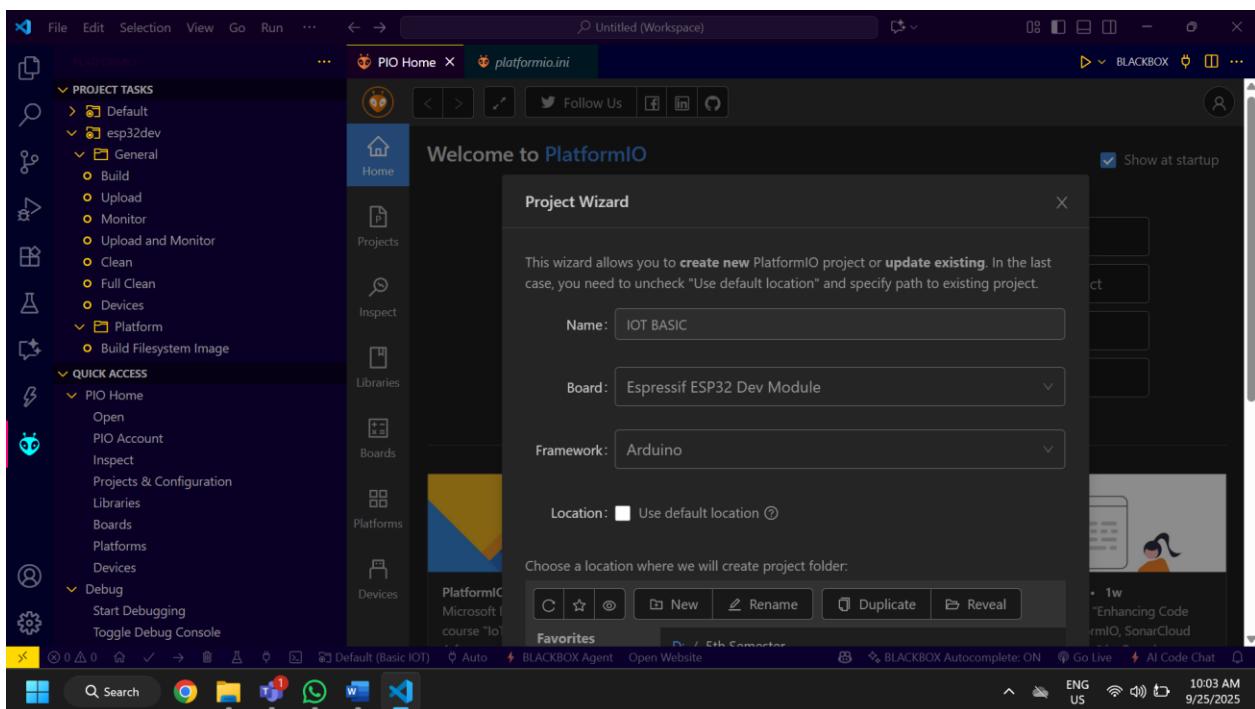
Group Member:	Ibtisam Butt
Registration No:	23-NTU-CS-1269
Class:	BSAI-5 th
Course Name:	Embedded IOT System
Submitted To:	<i>Nasir Mehmood</i>
Submission Date:	23-09-2025

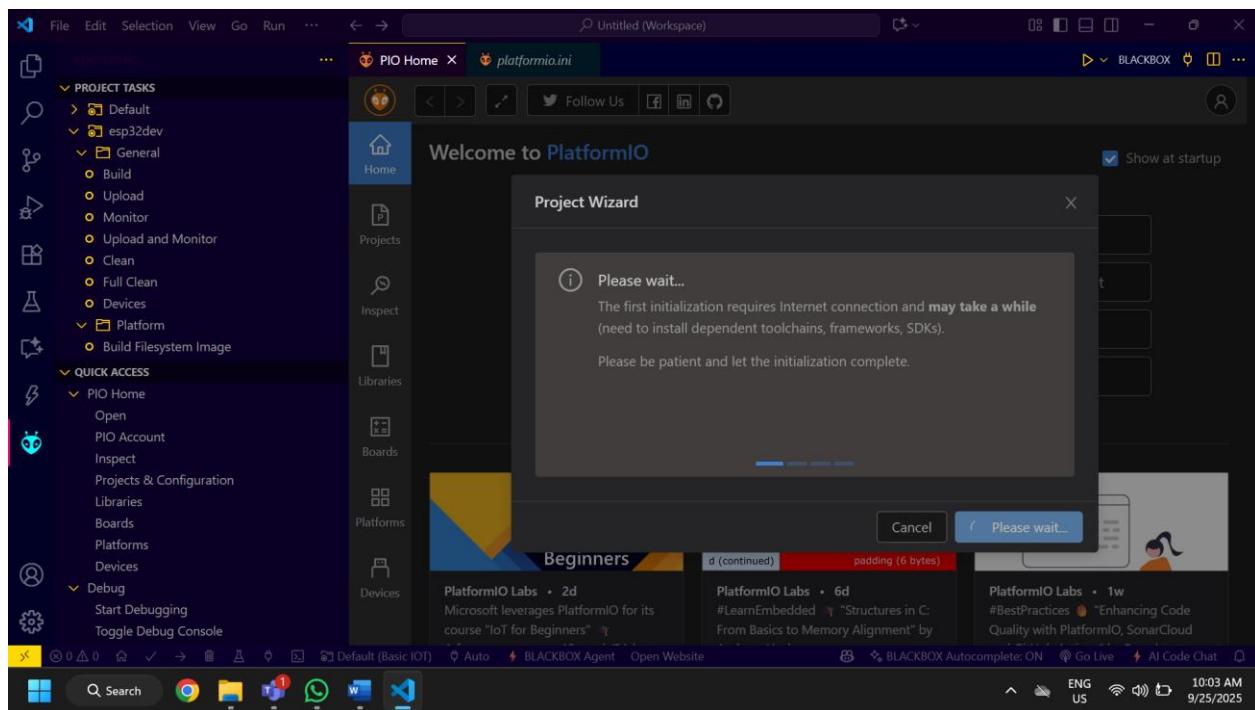
Embedded IOT System

1) Open VS Code



2) Create the project





3) platformio.ini (verify this)

A screenshot of the VS Code code editor. The left sidebar shows a file tree with a folder named "IOT BASICCC" containing a "platformio.ini" file. The main editor area displays the contents of the "platformio.ini" file. The file is a PlatformIO Project Configuration File, starting with a header and then defining an environment for esp32dev. The code is as follows:

```
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
```

The status bar at the bottom right shows the date and time as 9/25/2025, 10:05 AM.

The screenshot shows the PlatformIO IDE interface with the title bar "Untitled (Workspace)". The main area displays the contents of the "platformio.ini" configuration file. The file is a PlatformIO Project Configuration File, version 5.0.0, and includes settings for the [env:esp32dev] environment, specifying the platform as espressif32, board as esp32dev, and framework as arduino.

```
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)

The screenshot shows the PlatformIO IDE interface with the title bar "Untitled (Workspace)". The main area displays the contents of the "main.cpp" file, which is a basic Arduino sketch for an ESP32. The code includes an include statement for "Arduino.h", defines the LED_PIN, and contains setup and loop functions to toggle an LED.

```
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-I: 8 (often active-low)
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 }
```

5) Build, upload, and monitor

The screenshot shows the PlatformIO IDE interface. The left sidebar displays the project structure under 'UNTITLED (WORKSPACE)' with two main branches: 'IOT' and 'Basic IOT'. The 'Basic IOT' branch contains sub-folders '.pio', '.vscode', 'include', 'lib', 'src', 'test', '.gitignore', and 'platformio.ini'. The right side of the screen shows the code editor with 'main.cpp' open, containing C++ code for an Arduino project. Below the code editor is the 'TERMINAL' tab, which displays the build logs. The logs show the compilation process, including linking, size checking, and successful creation of the esp32 image. The terminal output ends with a message: '[SUCCESS] Took 18.16 seconds' and a note that the terminal will be reused by tasks.

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

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.
```