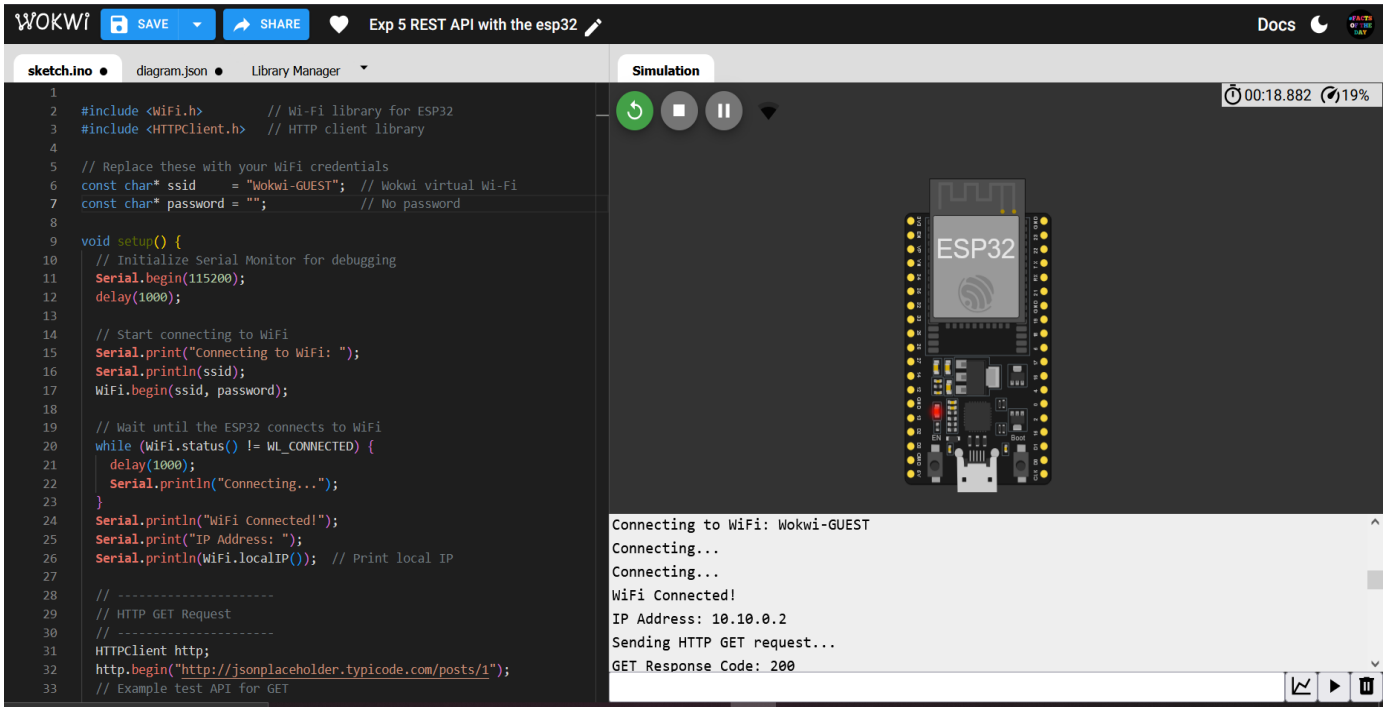
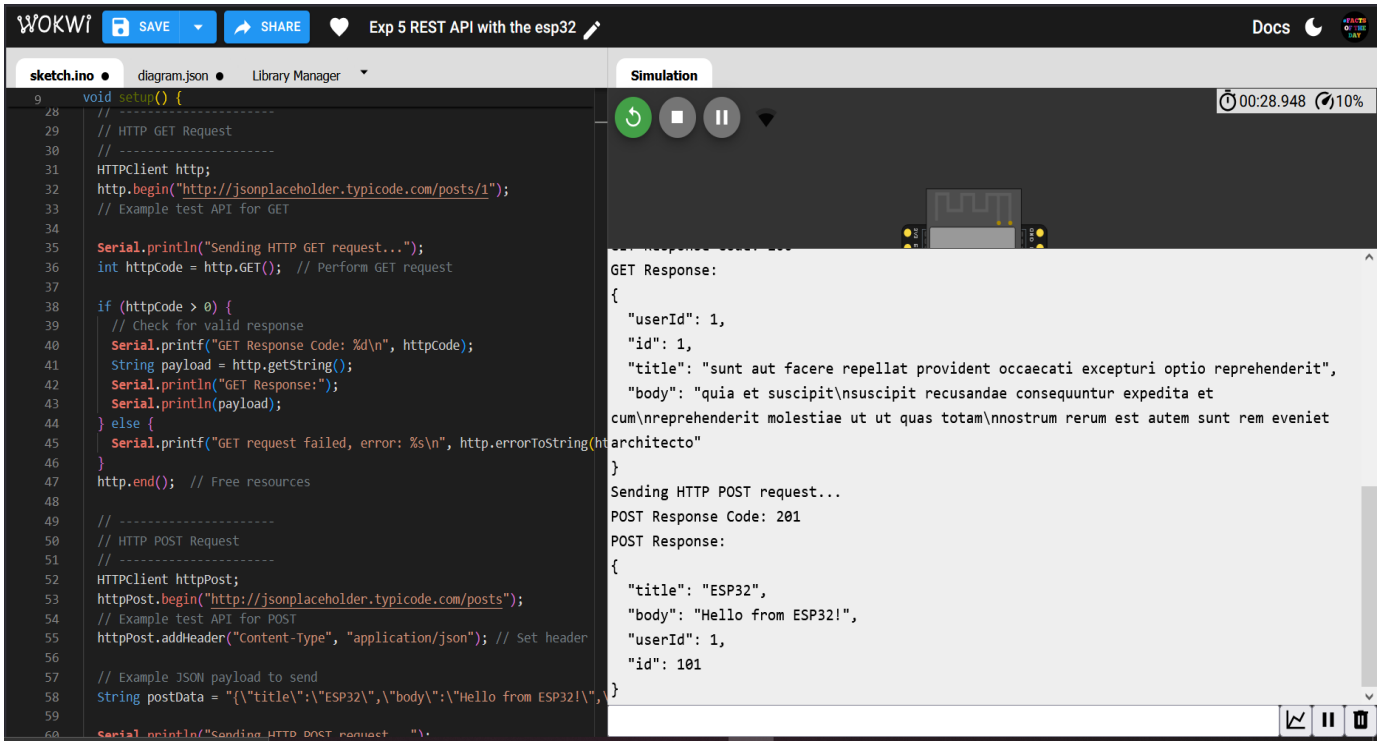
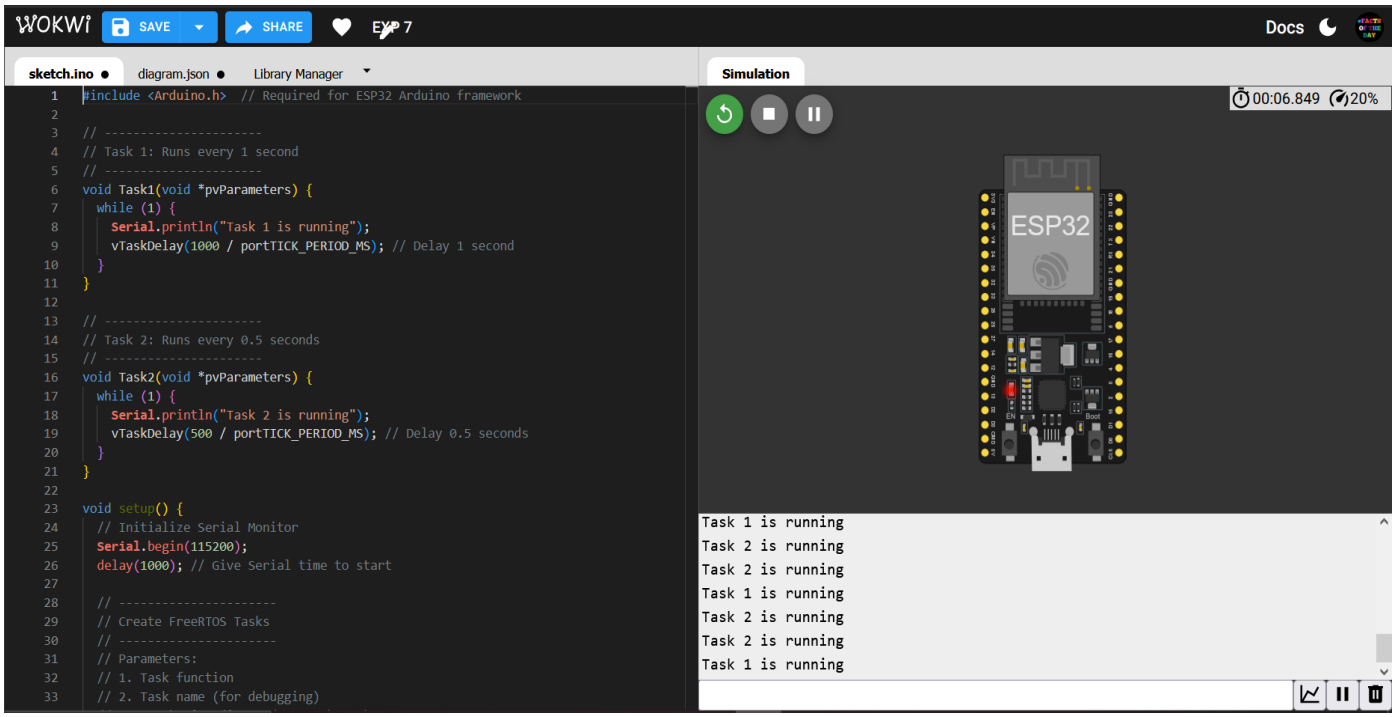
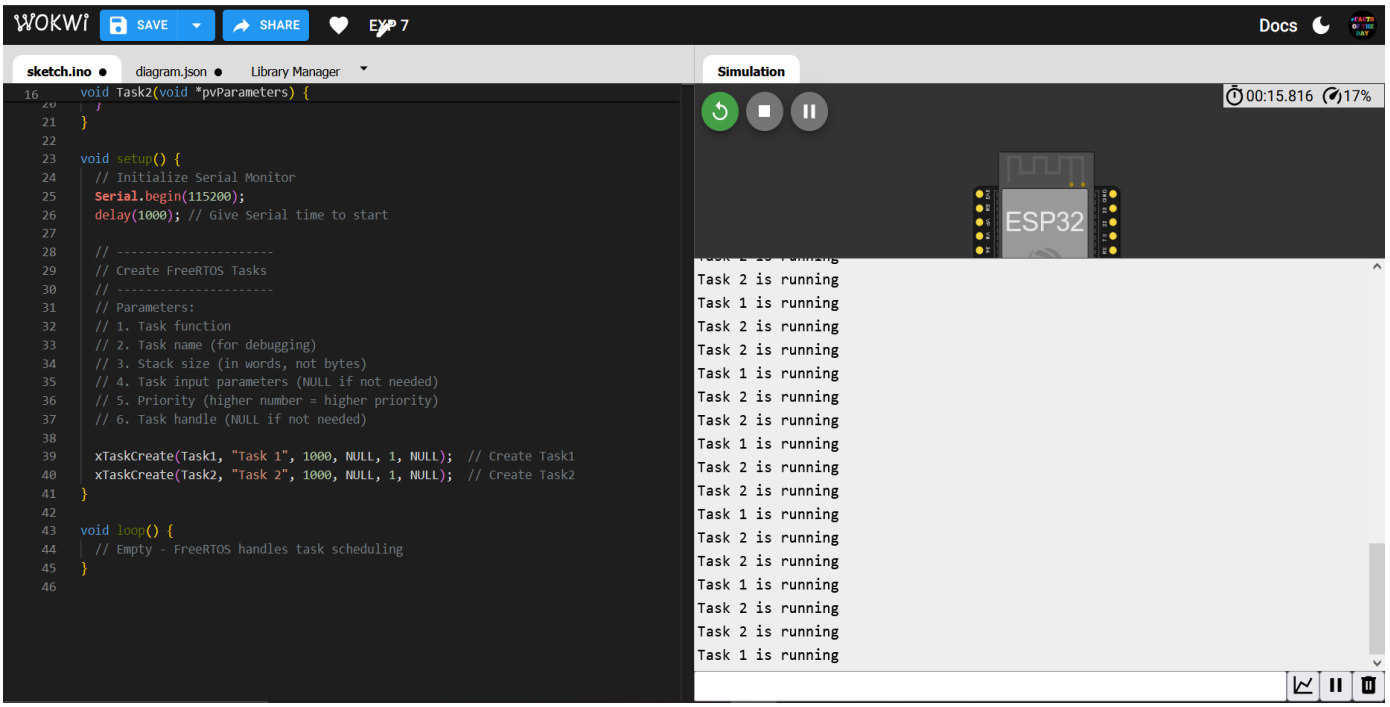
OUTPUT:


WOKWI
SAVE
SHARE
Exp 5 REST API with the esp32
Docs
sketch.ino
diagram.json
Library Manager
Simulation
00:18.882
19%
ESP32
#include <WiFi.h> // Wi-Fi library for ESP32
#include <HTTPClient.h> // HTTP client library
// Replace these with your WiFi credentials
const char* ssid = "Wokwi-GUEST"; // Wokwi virtual Wi-Fi
const char* password = ""; // No password
void setup() {
// Initialize Serial Monitor for debugging
Serial.begin(115200);
delay(1000);
// Start connecting to WiFi
Serial.print("Connecting to WiFi: ");
Serial.println(ssid);
WiFi.begin(ssid, password);
// Wait until the ESP32 connects to WiFi
while (WiFi.status() != WL_CONNECTED) {
delay(1000);
Serial.println("Connecting...");
}
Serial.println("WiFi Connected!");
Serial.print("IP Address: ");
Serial.println(WiFi.localIP()); // Print local IP
// ----------------------
// HTTP GET Request
// ----------------------
HTTPClient http;
http.begin("http://jsonplaceholder.typicode.com/posts/1");
// Example test API for GET
Connecting to WiFi: Wokwi-GUEST
Connecting...
Connecting...
WiFi Connected!
IP Address: 10.10.0.2
Sending HTTP GET request...
GET Response Code: 200

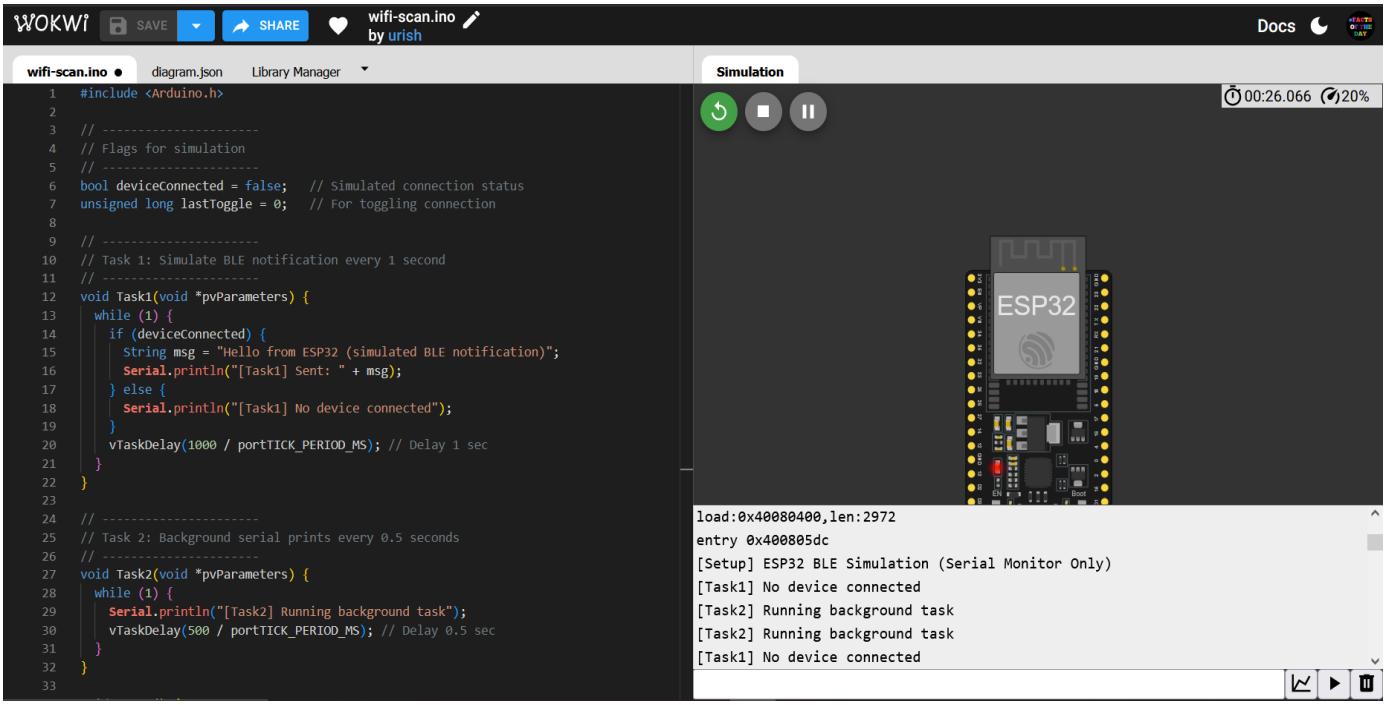

WOKWI
SAVE
SHARE
Exp 5 REST API with the esp32
Docs
sketch.ino
diagram.json
Library Manager
Simulation
00:28.948
10%
void setup() {
// HTTP GET Request
HTTPClient http;
http.begin("http://jsonplaceholder.typicode.com/posts/1");
// Example test API for GET
Serial.println("Sending HTTP GET request...");
int httpCode = http.GET(); // Perform GET request
if (httpCode > 0) {
// Check for valid response
Serial.printf("GET Response Code: %d\n", httpCode);
String payload = http.getString();
Serial.println("GET Response:");
Serial.println(payload);
} else {
Serial.printf("GET request failed, error: %s\n", http.errorToString(ht
}
http.end(); // Free resources
// HTTP POST Request
HTTPClient httpPost;
httpPost.begin("http://jsonplaceholder.typicode.com/posts");
// Example test API for POST
httpPost.addHeader("Content-Type", "application/json"); // Set header
// Example JSON payload to send
String postData = "{\"title\":\"ESP32\",\"body\":\"Hello from ESP32!\",\
GET Response:
{
"userId": 1,
"id": 1,
"title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",
"body": "quia et suscipit\nsuscipit recusandae consequuntur expedita et
cum\nreprehenderit molestiae ut ut quas totam\nnostrum rerum est autem sunt rem eveniet
architecto"
}
Sending HTTP POST request...
POST Response Code: 201
POST Response:
{
"title": "ESP32",
"body": "Hello from ESP32!",
"userId": 1,
"id": 101
}

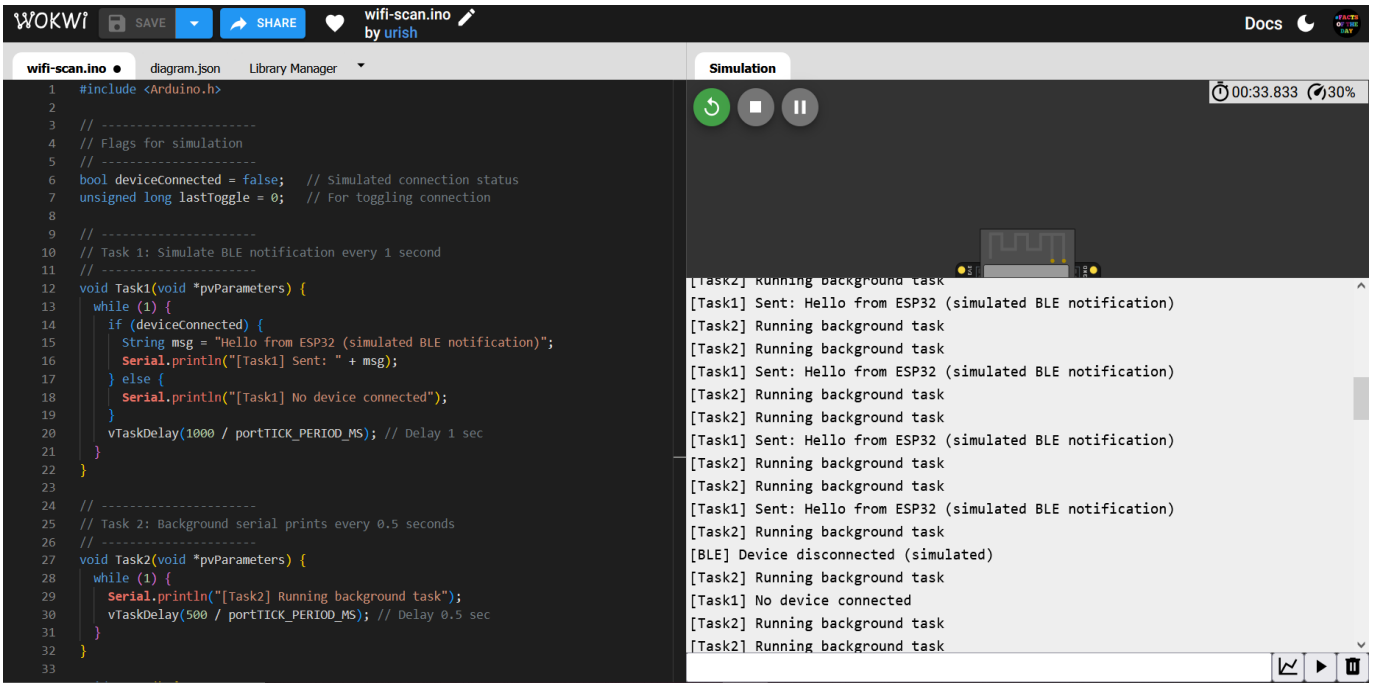
OUTPUT:


WOKWi
SAVE
SHARE
EXP 7
Docs
sketch.ino
diagram.json
Library Manager
Simulation
00:06.849
20%
ESP32
Task 1 is running
Task 2 is running
Task 2 is running
Task 1 is running
Task 2 is running
Task 2 is running
Task 1 is running


WOKWi
SAVE
SHARE
EXP 7
Docs
sketch.ino
diagram.json
Library Manager
Simulation
00:15.816
17%
ESP32
Task 2 is running
Task 1 is running
Task 2 is running
Task 2 is running
Task 1 is running
Task 2 is running
Task 2 is running
Task 1 is running
Task 2 is running
Task 2 is running
Task 1 is running
Task 2 is running
Task 2 is running
Task 1 is running
Task 2 is running
Task 1 is running

OUTPUT:


WOKWI
SAVE
SHARE
wifi-scan.ino
by urish
Docs
wifi-scan.ino
diagram.json
Library Manager
Simulation
00:26.066
20%
ESP32
#include <Arduino.h>
// ----------------------
// Flags for simulation
// ----------------------
bool deviceConnected = false; // Simulated connection status
unsigned long lastToggle = 0; // For toggling connection
// ----------------------
// Task 1: Simulate BLE notification every 1 second
// ----------------------
void Task1(void *pvParameters) {
while (1) {
if (deviceConnected) {
String msg = "Hello from ESP32 (simulated BLE notification)";
Serial.println("[Task1] Sent: " + msg);
} else {
Serial.println("[Task1] No device connected");
}
vTaskDelay(1000 / portTICK_PERIOD_MS); // Delay 1 sec
}
}
// ----------------------
// Task 2: Background serial prints every 0.5 seconds
// ----------------------
void Task2(void *pvParameters) {
while (1) {
Serial.println("[Task2] Running background task");
vTaskDelay(500 / portTICK_PERIOD_MS); // Delay 0.5 sec
}
}
load:0x40080400,len:2972
entry 0x400805dc
[Setup] ESP32 BLE Simulation (Serial Monitor Only)
[Task1] No device connected
[Task2] Running background task
[Task2] Running background task
[Task1] No device connected


WOKWI
SAVE
SHARE
wifi-scan.ino
by urish
Docs
wifi-scan.ino
diagram.json
Library Manager
Simulation
00:33.833
30%
#include <Arduino.h>
// ----------------------
// Flags for simulation
// ----------------------
bool deviceConnected = false; // Simulated connection status
unsigned long lastToggle = 0; // For toggling connection
// ----------------------
// Task 1: Simulate BLE notification every 1 second
// ----------------------
void Task1(void *pvParameters) {
while (1) {
if (deviceConnected) {
String msg = "Hello from ESP32 (simulated BLE notification)";
Serial.println("[Task1] Sent: " + msg);
} else {
Serial.println("[Task1] No device connected");
}
vTaskDelay(1000 / portTICK_PERIOD_MS); // Delay 1 sec
}
}
// ----------------------
// Task 2: Background serial prints every 0.5 seconds
// ----------------------
void Task2(void *pvParameters) {
while (1) {
Serial.println("[Task2] Running background task");
vTaskDelay(500 / portTICK_PERIOD_MS); // Delay 0.5 sec
}
}
[Task1] Sent: Hello from ESP32 (simulated BLE notification)
[Task2] Running background task
[Task2] Running background task
[Task1] Sent: Hello from ESP32 (simulated BLE notification)
[Task2] Running background task
[Task2] Running background task
[Task1] Sent: Hello from ESP32 (simulated BLE notification)
[Task2] Running background task
[Task2] Running background task
[Task1] Sent: Hello from ESP32 (simulated BLE notification)
[Task2] Running background task
[BLE] Device disconnected (simulated)
[Task2] Running background task
[Task1] No device connected
[Task2] Running background task
[Task2] Running background task