

EEL 7170 : Introduction to IoT

Lab Report



॥ त्वं ज्ञानमयो विज्ञानमयोऽसि ॥

Name: **Anushkaa Ambuj**
Roll Number: **B21ES006**
Program: **B.Tech in ES**

1 Lab-1

1.1 Objective

To interface the NodeMCU with APIs for fetching and sending data

1. Fetch data from OpenWeather API and upload it to ThingSpeak API
2. Generate Random Numbers and send it to ThingSpeak API

1.2 Components Used

1. NodeMCU (ESP8266 or ESP32)
2. Laptop or desktop with Arduino IDE installed
3. USB cable (for connecting NodeMCU)
4. Internet access
5. OpenWeather API account
6. ThingSpeak API account

1.3 Procedure

- Set WIFI Configurations
- Initialize the Openweather and ThingSpeak API Keys
- Modify the API endpoint by adding the code block mentioned below:

```
String serverPath = (serverName + "&field1=" + JSON.stringify(myObject["main"]["temp"])  
+ "&field2=" + JSON.stringify(myObject["main"]["pressure"]) +  
"&field3=" + JSON.stringify(myObject["main"]["humidity"]) +  
"&field4=" + JSON.stringify(myObject["wind"]["speed"]) + "&field5=" + String(random(126)));
```

- Note: The JSON should be converted into 'string' before sending to ThingSpeak API
- Alter the code by the number $(21*6) = 126$ for generating random numbers, in code snippet *random(126)*

1.4 Code

```
1  #include <ESP8266WiFi.h>  
2  #include <ESP8266HTTPClient.h>  
3  #include <WiFiClient.h>  
4  #include <Arduino_JSON.h>  
5  
6  // Wifi Configurations  
7  const char* ssid = "IDEAPAD";  
8  const char* password = "Hello144";  
9  
10 // OpenWeather API Key  
11 String openWeatherMapApiKey = "56d550e863d263ebf0f515693bc3186b";
```

```

12 // Add THINGSPEAK.COM API KEY
13 String serverName = "http://api.thingspeak.com/update?api_key=4
    ↳ E785DAEOKBXTSZC";
14
15 // Set your country code and city
16 String city = "Jodhpur";
17 String countryCode = "IN";
18
19 // THE DEFAULT TIMER IS SET TO 10 SECONDS FOR TESTING PURPOSES
20 // For a final application, check the API call limits per hour/minute to
    ↳ avoid getting blocked/banned
21 unsigned long lastTime = 0;
22 // Timer set to 10 minutes (600000)
23 //unsigned long timerDelay = 600000;
24 // Set timer to 10 seconds (10000)
25 unsigned long timerDelay = 10000;
26
27 String jsonBuffer;
28
29 void setup() {
30     Serial.begin(115200);
31
32     WiFi.begin(ssid, password);
33     Serial.println("Connecting");
34     while(WiFi.status() != WL_CONNECTED) {
35         delay(500);
36         Serial.print(".");
37     }
38     Serial.println("");
39     Serial.print("Connected to WiFi network with IP Address: ");
40     Serial.println(WiFi.localIP());
41
42     Serial.println("Timer set to 10 seconds (timerDelay variable), it will
    ↳ take 10 seconds before publishing the first reading.");
43 }
44
45 void loop() {
46     // Send an HTTP GET request
47     if ((millis() - lastTime) > timerDelay) {
48         // Check WiFi connection status
49         if(WiFi.status() == WL_CONNECTED){
50             String serverPath1 = "http://api.openweathermap.org/data/2.5/weather
    ↳ ?q=" + city + "," + countryCode + "&APPID=" +
    ↳ openWeatherMapApiKey;
51
52             jsonBuffer = httpGETRequest(serverPath1.c_str());
53             Serial.println(jsonBuffer);
54             JSONVar myObject = JSON.parse(jsonBuffer);
55
56             // JSON.typeof(jsonVar) can be used to get the type of the var
57             if (JSON.typeof(myObject) == "undefined") {
58                 Serial.println("Parsing input failed!");
59                 return;
60             }

```

```

61     Serial.print("JSON_object=");
62     Serial.println(myObject);
63     Serial.print("Temperature:");
64     Serial.println(myObject["main"]["temp"]);
65     Serial.print("Pressure:");
66     Serial.println(myObject["main"]["pressure"]);
67     Serial.print("Humidity:");
68     Serial.println(myObject["main"]["humidity"]);
69     Serial.print("Wind_Speed:");
70     Serial.println(myObject["wind"]["speed"]);
71
72     WiFiClient client;
73     HTTPClient http;
74
75     String serverPath = (serverName + "&field1=" + JSON.stringify(
76         ↳ myObject["main"]["temp"]) + "&field2=" + JSON.stringify(
77         ↳ myObject["main"]["pressure"]) + "&field3=" + JSON.stringify(
78         ↳ myObject["main"]["humidity"]) + "&field4=" + JSON.stringify(
79         ↳ myObject["wind"]["speed"]) + "&field5=" + String(random(126)))
80         ↳ ;
81
82     // Your Domain name with URL path or IP address with path
83     http.begin(client, serverPath.c_str());
84
85     // Send HTTP GET request
86     int httpResponseCode = http.GET();
87
88     if (httpResponseCode>0) {
89         Serial.print("HTTP_Response_code:");
90         Serial.println(httpResponseCode);
91         String payload = http.getString();
92         Serial.println(payload);
93     }
94     else {
95         Serial.print("Error_code:");
96         Serial.println(httpResponseCode);
97     }
98     // Free resources
99     http.end();
100 }
101 else {
102     Serial.println("WiFi_Disconnected");
103 }
104 lastTime = millis();
105 }
106 }
107
108 String httpGETRequest(const char* serverName) {
109     WiFiClient client;
110     HTTPClient http;
111
112     // Your IP address with path or Domain name with URL path
113     http.begin(client, serverName);

```

```

110
111 // Send HTTP POST request
112 int httpResponseCode = http.GET();
113
114 String payload = "{}";
115
116 if (httpResponseCode>0) {
117     Serial.print("HTTP_Response_code: ");
118     Serial.println(httpResponseCode);
119     payload = http.getString();
120 }
121 else {
122     Serial.print("Error_code: ");
123     Serial.println(httpResponseCode);
124 }
125 // Free resources
126 http.end();
127
128 return payload;
129 }

```

1.5 Results

We successfully obtain the Temperature, Pressure, Humidity, Wind Speed and Random number Plots on the ThinkSpeak website using NodeMCU microcontroller.

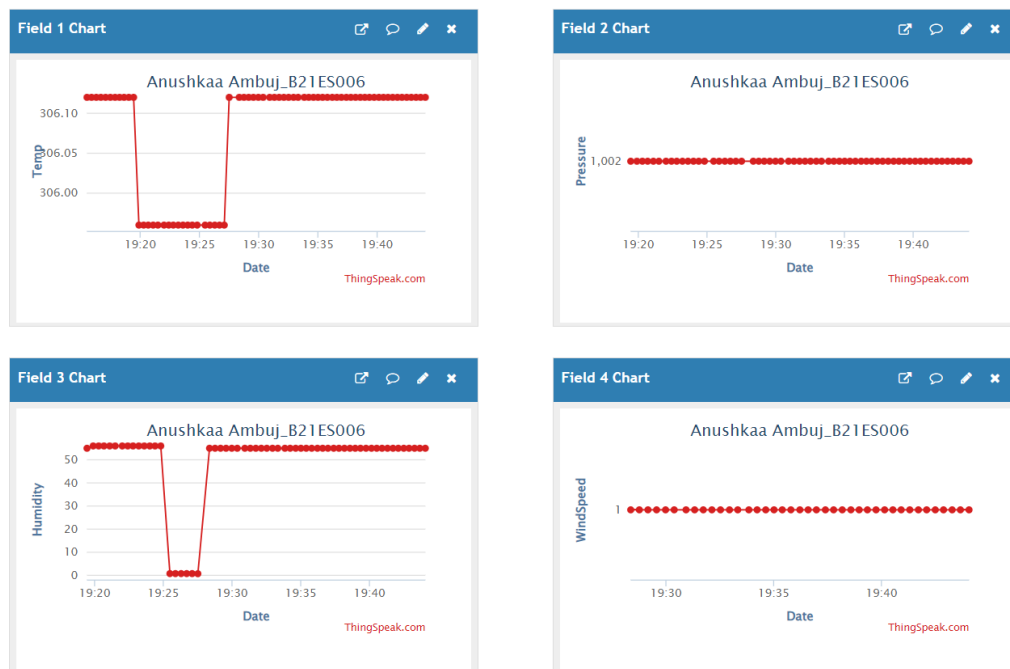


Figure 1: Plots of data received from Openweather API

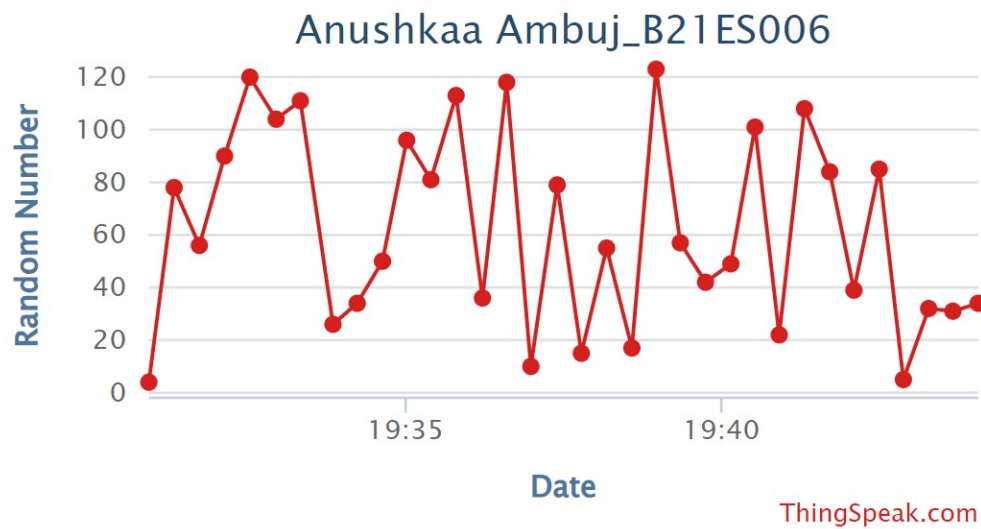


Figure 2: Plot of Random Number sent to ThingSpeak

```

Lab-1.ino
10 // OpenWeatherMap API Key
11 String openWeatherMapApiKey = "56d550e863d263ebf0f515693bc3186b";
12 // Add THINGSPEAK.COM API KEY
13 String serverName = "http://api.thingspeak.com/update?api_key=4E785DAE0K8XT52C";

Output Serial Monitor x
Message (Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)' on 'COM5')
New Line 115200 baud

0
HTTP Response code: 200
{"coord":{"lon":73.03,"lat":26.2867},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"},"base":"stations","main":{"temp":306.12,"feels_like":310.9,"temp_mi
JSON object = {"coord":{"lon":73.03,"lat":26.2867},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"},"base":"stations","main":{"temp":306.12,"feels_like":
Temperature: 306.12
Pressure: 1002
Humidity: 55
Wind Speed: 0.71
HTTP Response code: 200
65
HTTP Response code: 200
{"coord":{"lon":73.03,"lat":26.2867},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"},"base":"stations","main":{"temp":306.12,"feels_like":310.9,"temp_mi
JSON object = {"coord":{"lon":73.03,"lat":26.2867},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"},"base":"stations","main":{"temp":306.12,"feels_like":
Temperature: 306.12
Pressure: 1002
Humidity: 55
Wind Speed: 0.71
HTTP Response code: 200
0
HTTP Response code: 200
{"coord":{"lon":73.03,"lat":26.2867},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"},"base":"stations","main":{"temp":306.12,"feels_like":310.9,"temp_mi
JSON object = {"coord":{"lon":73.03,"lat":26.2867},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"},"base":"stations","main":{"temp":306.12,"feels_like":
Temperature: 306.12
Pressure: 1002
Humidity: 55
Wind Speed: 0.71
HTTP Response code: 200
69
  
```

Figure 3: Serial Monitor Output

2 InLab-1

2.1 Problem 1

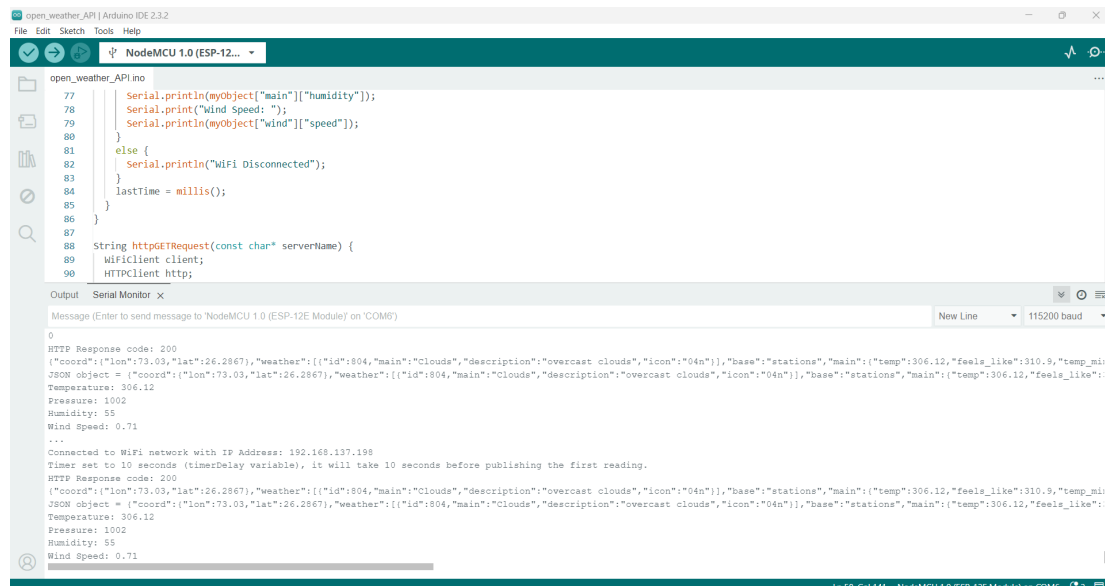
Objective

- Fetching and Sending Data from OpenWeather API: To fetch weather data and display it on the serial console.

Procedure

- To display data from OpenWeather on the serial monitor, we utilized the API.ino file provided through Google Classroom. First, we updated the Wi-Fi credentials by setting the ssid and password variables to match our network details. After creating an OpenWeather account to obtain the API key, we inserted the key into the openWeatherMapApiKey variable in the Arduino IDE. Finally, we uploaded the code to the NodeMCU and observed the output on the Serial Monitor.
- To begin, open the thingspeak.ino file provided on Google Classroom and update the Wi-Fi configuration by entering your network credentials. Next, create a ThingSpeak account and set up a new channel. Copy the write API key from your ThingSpeak channel and paste it into the appropriate section in the Arduino IDE. After updating the code, upload it to the NodeMCU and verify that the serial console displays an "HTTP Response code: 200". Once successful, you can visualize the uploaded data on the ThingSpeak platform.

Results



The screenshot displays the Arduino IDE interface. The top pane shows the code for `open_weather_API.ino`, which includes headers for `Arduino.h`, `WiFi.h`, and `HTTP.h`. It defines a `myObject` for weather data and a `httpGETRequest` function. The `setup` function initializes the serial port and connects to the WiFi network. The `loop` function prints the weather data from `myObject` and checks for a WiFi connection. The bottom pane shows the Serial Monitor output, which displays the HTTP response code (200) and the JSON object containing weather data for London, including temperature, humidity, wind speed, and cloud cover.

```
open_weather_API.ino
77 Serial.println(myObject["main"]["humidity"]);
78 Serial.print("Wind Speed: ");
79 Serial.println(myObject["wind"]["speed"]);
80 }
81 else {
82   Serial.println("WiFi Disconnected");
83 }
84   lastTime = millis();
85 }
86 }
87
88 String httpGETRequest(const char* serverName) {
89   WiFiClient client;
90   HTTPClient http;
```

Serial Monitor Output:

```
0
HTTP Response code: 200
{"coord":{"lon":73.03,"lat":26.2867},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"},"base":"stations","main":{"temp":306.12,"feels_like":310.9,"temp_mi
JSON object = {"coord":{"lon":73.03,"lat":26.2867},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"},"base":"stations","main":{"temp":306.12,"feels_like":
Temperature: 306.12
Pressure: 1002
Humidity: 55
Wind Speed: 0.71
...
Connected to WiFi network with IP Address: 192.168.137.198
Timer set to 10 seconds (timerDelay variable), it will take 10 seconds before publishing the first reading.
HTTP Response code: 200
{"coord":{"lon":73.03,"lat":26.2867},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"},"base":"stations","main":{"temp":306.12,"feels_like":310.9,"temp_mi
JSON object = {"coord":{"lon":73.03,"lat":26.2867},"weather":[{"id":804,"main":"Clouds","description":"overcast clouds","icon":"04n"},"base":"stations","main":{"temp":306.12,"feels_like":
Temperature: 306.12
Pressure: 1002
Humidity: 55
Wind Speed: 0.71
```

Figure 4: OpenWeather Data on the Serial Monitor

2.2 Problem-2

Objective

- Sending Data to ThingSpeak API: To upload random data to ThingSpeak and visualize it.

Results

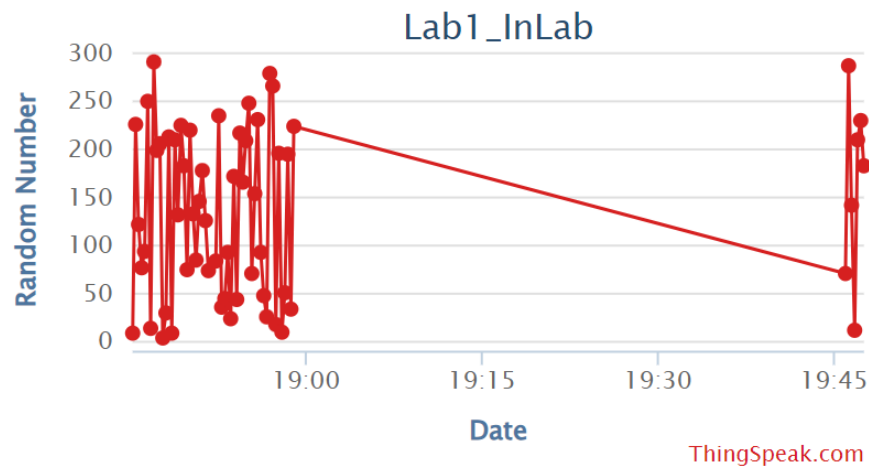


Figure 5: Plot of Random Number on Thingspeak

```
thing_speak.ino
38 String serverPath = serverName + "&field=" + String(random(300));
39
40 // Your Domain name with URL path or IP address with path
41 http.begin(client, serverPath.c_str());
42
43 // Send HTTP GET request
44 int httpResponseCode = http.GET();
45
46 if (httpResponseCode>0) {
47   Serial.print("HTTP Response code: ");
48   Serial.println(httpResponseCode);
49   String payload = http.getString();
50   Serial.println(payload);
51 }
```

Output Serial Monitor x

Message (Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)' on 'COM6')

New Line 115200 baud

Pressure: 1002
Humidity: 55
Wind Speed: 0.71
HTTP Response code: 200
0
HTTP Response code: 200
0
HTTP Response code: 200
0
HTTP Response code: 200
0
HTTP Response code: 200
0
HTTP Response code: 200
0
HTTP Response code: 200
0

In 49, Col 23 NodeMCU 1.0 (ESP-12E Module) on COM6

Figure 6: Random Number on the Serial Monitor