Name: _____ | Class: D16 | Roll No: __

EXPERIMENT NO: 03

Aim: To interface temperature and humidity sensor with Node MCU (ESP8266) and receive the data via SMS through Twilio Console.

Apparatus: NODE MCU, Twilio Console, API, Temperature and Humidity Sensor (DHT11), Micro USB Cable, Breadboard, Female to female connectors, Twilio website(https://www.twilio.com/en-us) for sending data SMS to phones.

INTRODUCTION:

In this experiment, we explore the integration of cloud-based communication services with IoT devices to enable real-time data transmission. The Twilio Console is a key component of this setup, providing an easy-to-use platform for managing and deploying communication services such as SMS.

Twilio's robust API allows developers to send and receive messages globally, making it an ideal choice for IoT applications where remote monitoring and instant alerts are crucial. By leveraging Twilio's capabilities, the Node MCU (ESP8266) in this experiment can efficiently send temperature and humidity readings directly to a user's mobile device via SMS. This integration not only simplifies the communication process but also enhances the practicality and reach of IoT projects, enabling real-time data delivery and user notifications.

Theory:

Twilio is a cloud communications platform that provides APIs for sending and receiving SMS, voice messages, and other forms of communication. Twilio's services are widely used in IoT (Internet of Things) applications to enable devices to communicate with users via SMS or other messaging protocols.

In IoT applications, Twilio is often used to enable devices to send alerts or data to users via SMS. For example, a sensor monitoring environmental conditions (like temperature or humidity) can automatically send a text message to the user when certain thresholds are exceeded. This is particularly useful in scenarios where real-time monitoring is critical, such as in smart homes, industrial automation, or environmental monitoring systems.

GETTING API KEY:

1. Go to **Twilio Console** and create an account and also get your account verified using OTP.
2. Next, Enter your mobile number and get verified using OTP, recovery code will be generated save it
3. Now, Welcome window will appear select appropriate option and get started with Twilio
4. A new window will appear Now click on the **Get phone numbe**r option.
5. Enter **Account SID, Authentication Token, My Twilio Phone number** generated on Twilio Dashboard also add your registered mobile number in code.
6. Make sure you add network credentials in the code.

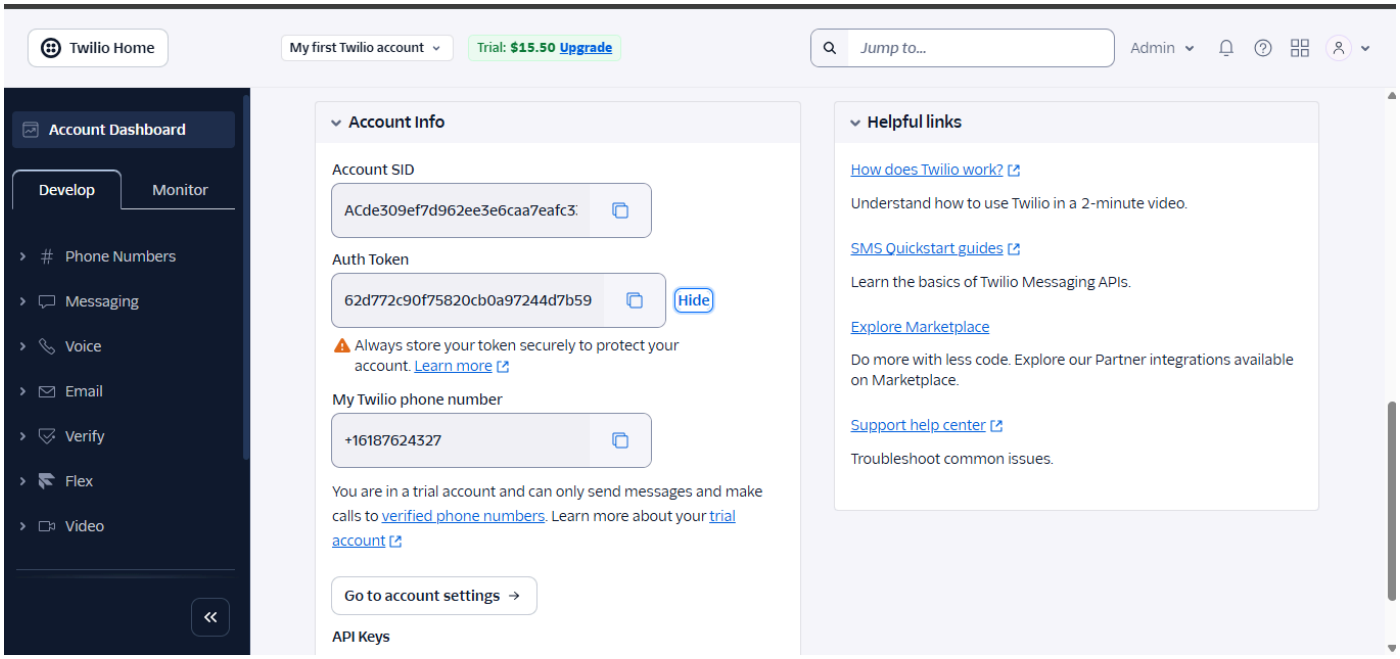

Fig. (1) Account Dashboard of Twilio

PROCEDURE:

1. Interface the sensor (DHT11) with ESP8266
2. Upload the code directly to the ESP8266 Module. This means that the existing code or firmware will be erased.
3. Before uploading the code, connect the GPIO to GND and RESET the ESP Module to enable Programming Mode.
4. Also, select "Generic ESP8266 Module" in the Boards section of the Arduino IDE. Make sure that the correct COM PORT is selected.
5. Upload the code to the NODE MCU by selecting the right board and port.
6. If there are no errors, the code will be compiled and uploaded to the Node MCU successfully else check for the errors mentioned.
7. Now, you will receive SMS from Twilio trial account which contains information about Temperature and Humidity.

CODE:

```
#include <ESP8266WiFi.h>
#include <ESP8266HTTPClient.h>
#include "DHT.h"

#define DHTPIN D7
#define DHTTYPE DHT11  // or DHT22
DHT dht(DHTPIN, DHTTYPE);

// WiFi credentials
const char* ssid = "shreyash";
const char* password = "shreyash";

// Twilio API credentials
const char* accountSid = "ACde309ef7d962ee3e6caa7eafc338e8a1";
const char* authToken = "62d772c90f75820cb0a97244d7b59001";
const char* fromNumber = "+16187624327"; // Twilio number
const char* toNumber = "+917977529772"; // Your phone number

void setup() {
  Serial.begin(115200);
  WiFi.begin(ssid, password);

  Serial.print("Connecting to WiFi");
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("\nConnected to WiFi");
  dht.begin();
}

void loop() {
  float humidity = dht.readHumidity();
  float temperature = dht.readTemperature(); // Celsius

  if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Failed to read from DHT sensor!");
    delay(2000);
    return;
  }
```

```
  String message = "Temp: " + String(temperature) + "C, Humidity: " +
String(humidity) + "%";
  sendSMS(message);

  delay(60000); // send every 1 minute
}

void sendSMS(String body) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;
    WiFiClientSecure client;
    client.setInsecure(); // skip SSL verification for simplicity

    String url = "https://api.twilio.com/2010-04-01/Accounts/" +
String(accountSid) + "/Messages.json";

    http.begin(client, url);
    http.addHeader("Content-Type", "application/x-www-form-urlencoded");
    http.setAuthorization(accountSid, authToken);

    String postData = "From=" + String(fromNumber) + "&To=" + String(toNumber) +
"&Body=" + body;

    int httpCode = http.POST(postData);
    String payload = http.getString();

    Serial.println("HTTP Code: " + String(httpCode));
    Serial.println("Response: " + payload);

    http.end();
  } else {
    Serial.println("WiFi Disconnected");
  }
}
```

OUTPUT:

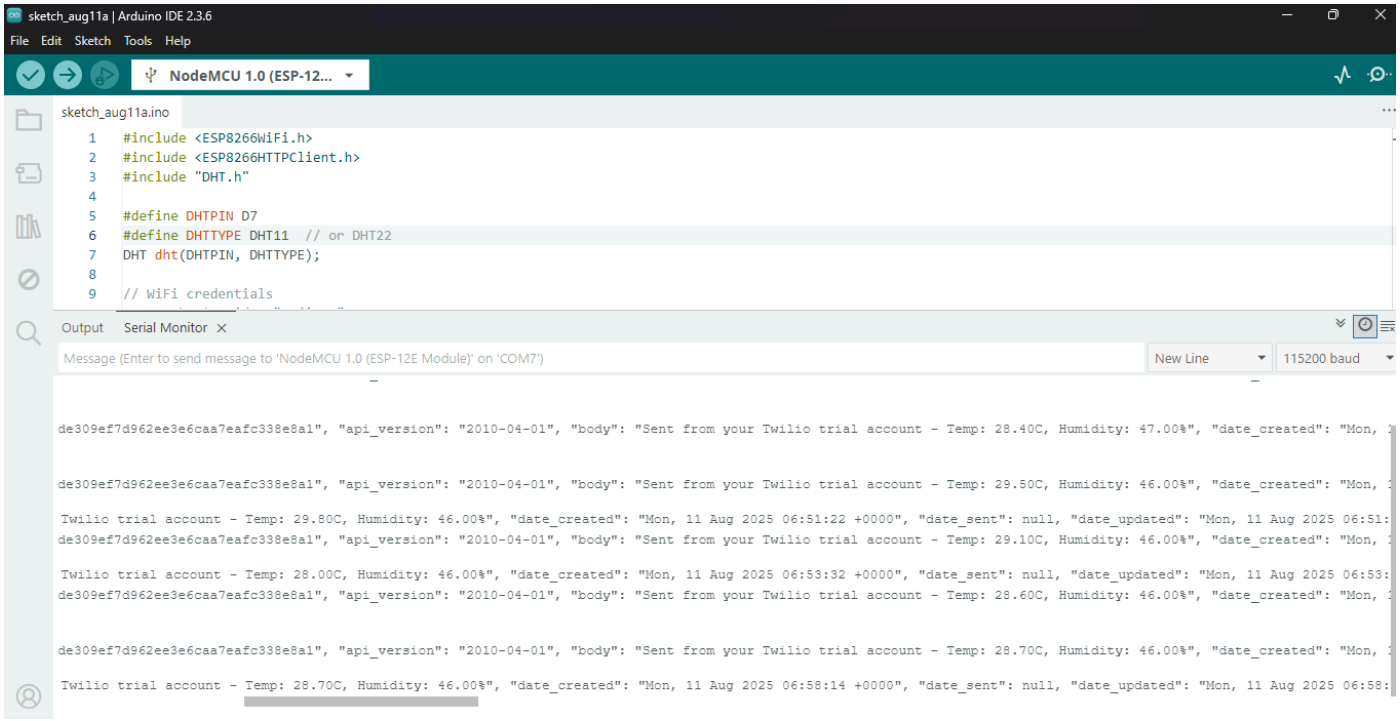

Fig. (2) Serial Monitor showing Temperature and humidity and sends data to Twilio

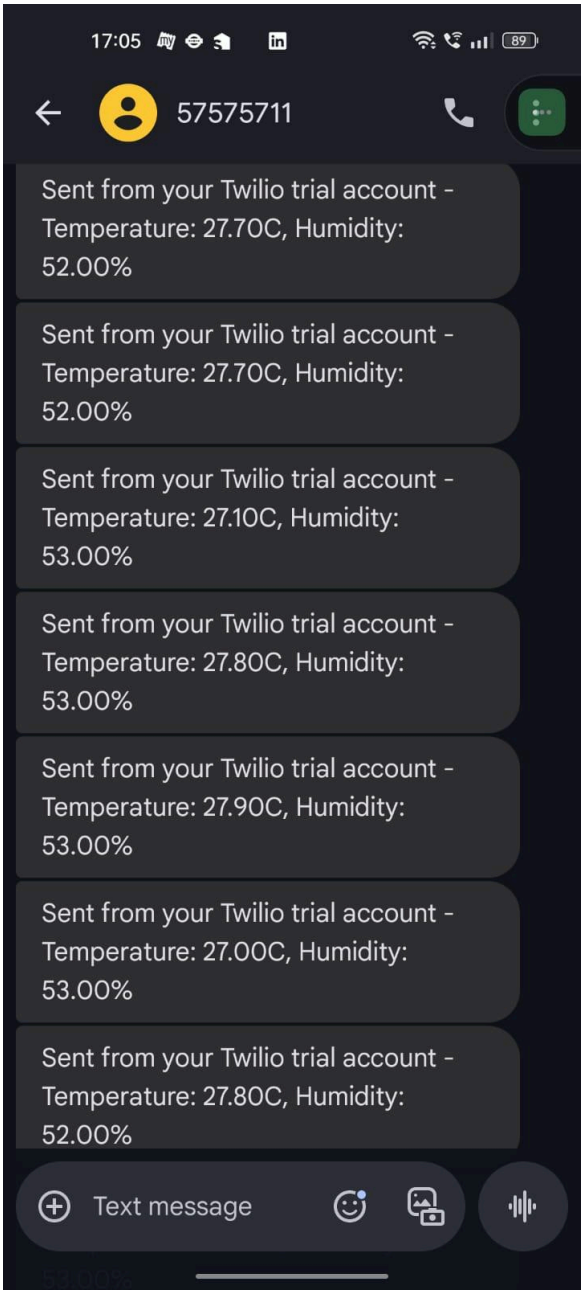

Fig. (3) Receiving SMS to Registered Number from Twilio Website

CONCLUSION:

1. Illustrated how IoT devices can effectively communicate with users via SMS.
2. Demonstrated real-time monitoring and remote alerts using Twilio's API.
3. Successfully connected the DHT11 sensor to the NodeMCU and transmitted data via SMS using Twilio.
4. Validated the effectiveness of integrating IoT with cloud communication for practical applications.