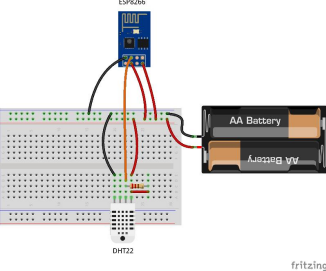
**INTERNET OF THINGS (IOT)**

NAME: PIKU MAITY

USN: 1RV21MC071

1. **Integrating adafruit or thingsboard with any programming board and a sensor or an actuator.**



**Thingsboard with dht11 sensor**.

#include "DHT.h"

#include <ESP8266WiFi.h>

#include <ThingsBoard.h>

#define WIFI\_AP "YOUR\_WIFI\_AP"

#define WIFI\_PASSWORD "YOUR\_WIFI\_PASSWORD" #define TOKEN "ESP8266\_DEMO\_TOKEN"

// DHT

#define DHTPIN 2

#define DHTTYPE DHT22

char thingsboardServer[] = "YOUR\_THINGSBOARD\_HOST\_OR\_IP"; WiFiClient wifiClient;

// Initialize DHT sensor.

DHT dht(DHTPIN, DHTTYPE);

ThingsBoard tb(wifiClient);

int status = WL\_IDLE\_STATUS;

unsigned long lastSend;

void setup()

{

Serial.begin(115200);

dht.begin();

delay(10);

InitWiFi();

lastSend = 0;

}

void loop()

{

if ( !tb.connected() ) {

reconnect();

}

if ( millis() - lastSend > 1000 ) { // Update and send only after 1 seconds getAndSendTemperatureAndHumidityData();

lastSend = millis();

}

tb.loop();

}

void getAndSendTemperatureAndHumidityData()

{

Serial.println("Collecting temperature data.");

// Reading temperature or humidity takes about 250 milliseconds! float humidity = dht.readHumidity();

// Read temperature as Celsius (the default)

float temperature = dht.readTemperature();

// Check if any reads failed and exit early (to try again). if (isnan(humidity) || isnan(temperature)) {

Serial.println("Failed to read from DHT sensor!");

return;

}

Serial.println("Sending data to ThingsBoard:"); Serial.print("Humidity: ");

Serial.print(humidity);

Serial.print(" %\t");

Serial.print("Temperature: ");

Serial.print(temperature);

Serial.println(" \*C ");

tb.sendTelemetryFloat("temperature", temperature); tb.sendTelemetryFloat("humidity", humidity); }

void InitWiFi()

{

Serial.println("Connecting to AP ...");

// attempt to connect to WiFi network

WiFi.begin(WIFI\_AP, WIFI\_PASSWORD); while (WiFi.status() != WL\_CONNECTED) { delay(500);

Serial.print(".");

}

Serial.println("Connected to AP");

}

void reconnect() {

// Loop until we're reconnected

while (!tb.connected()) {

status = WiFi.status();

if ( status != WL\_CONNECTED) {

WiFi.begin(WIFI\_AP, WIFI\_PASSWORD); while (WiFi.status() != WL\_CONNECTED) { delay(500);

Serial.print(".");

}

Serial.println("Connected to AP");

}

Serial.print("Connecting to ThingsBoard node ...");

if ( tb.connect(thingsboardServer, TOKEN) ) {

Serial.println( "[DONE]" );

} else {

Serial.print( "[FAILED]" );

Serial.println( " : retrying in 5 seconds]" );

// Wait 5 seconds before retrying

delay( 5000 );

}

}

}

2. **Configure esp8266 to connect over a wireless network to communicate sensor values to a serial monitor.**

Sensor chosen: DHT11

DHT11 data is read using esp8266

Pinout:

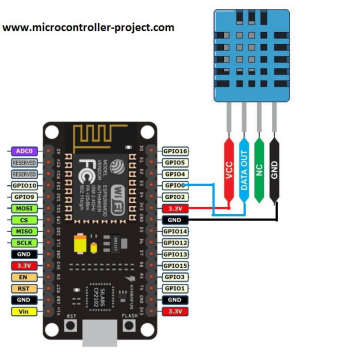
Dht - Nodemcu

Vcc - 3v3

GND - GND

Data - D3 pin

**Fritzing diagram:**



**Code:**

#include <ESP8266WiFi.h>

#include "DHT.h" // DHT11 temperature and humidity sensor

Predefined library

#define DHTTYPE DHT11 // DHT 11

#define dht\_dpin 0 //GPIO-0 D3 pin of nodemcu

const char\* ssid = "Your SSID";

const char\* password = "Your Wifi Password";

DHT dht(dht\_dpin, DHTTYPE);

WiFiServer server(80);

void setup(void)

{

dht.begin();

Serial.begin(9600);

delay(10);

// Connect to WiFi network

Serial.println();

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password); //Begin WiFi

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.print(".");

}

Serial.println("");

Serial.println("WiFi connected");

// Start the server

server.begin();

Serial.println("Server started");

// Print the IP address on serial monitor

Serial.print("Use this URL to connect: ");

Serial.print("http://"); //URL IP to be typed in mobile/desktop browser

Serial.print(WiFi.localIP());

Serial.println("/");

}

void loop() {

float h = 0;

float t=0;

float f=0;

h = dht.readHumidity();

t = dht.readTemperature();

f = (h \* 1.8) + 32;

delay(2000);

Serial.print(“Temperature:”);

Serial.println(t);

Serial.print(“Humidity:”);

Serial.println(h);

}

3. **Send email alert or sms notification from esp8266 using wifi communication.**

**Gmail SMTP Server Settings**

If you’re using a Gmail account, these are the SMTP Server details:

∙ SMTP Server: **smtp.gmail.com**

∙ SMTP username: Complete Gmail address

∙ SMTP password: Your Gmail password

∙ SMTP port (TLS): **587**

∙ SMTP port (SSL): **465**

∙ SMTP TLS/SSL required: **yes**

#include <Arduino.h>

#if defined(ESP32)

#include <WiFi.h>

#elif defined(ESP8266)

#include <ESP8266WiFi.h>

#endif

#include <ESP\_Mail\_Client.h>

#define WIFI\_SSID "REPLACE\_WITH\_YOUR\_SSID"

#define WIFI\_PASSWORD "REPLACE\_WITH\_YOUR\_PASSWORD"

#define SMTP\_HOST "smtp.gmail.com"

#define SMTP\_PORT 465

/\* The sign in credentials \*/

#define AUTHOR\_EMAIL "YOUR\_EMAIL@XXXX.com" #define AUTHOR\_PASSWORD "YOUR\_EMAIL\_PASS" /\* Recipient's email\*/

#define RECIPIENT\_EMAIL "RECIPIENTE\_EMAIL@XXXX.com"

/\* The SMTP Session object used for Email sending \*/ SMTPSession smtp;

/\* Callback function to get the Email sending status \*/

void smtpCallback(SMTP\_Status status);

void setup(){

Serial.begin(115200);

Serial.println();

Serial.print("Connecting to AP");

WiFi.begin(WIFI\_SSID, WIFI\_PASSWORD);

while (WiFi.status() != WL\_CONNECTED){

Serial.print(".");

delay(200);

}

Serial.println("");

Serial.println("WiFi connected.");

Serial.println("IP address: ");

Serial.println(WiFi.localIP());

Serial.println();

/\*\* Enable the debug via Serial port

\* none debug or 0

\* basic debug or 1

\*/

smtp.debug(1);

/\* Set the callback function to get the sending results \*/ smtp.callback(smtpCallback);

/\* Declare the session config data \*/

ESP\_Mail\_Session session;

/\* Set the session config \*/

session.server.host\_name = SMTP\_HOST;

session.server.port = SMTP\_PORT;

session.login.email = AUTHOR\_EMAIL;

session.login.password = AUTHOR\_PASSWORD;

session.login.user\_domain = "";

/\* Declare the message class \*/

SMTP\_Message message;

/\* Set the message headers \*/

message.sender.name = "ESP";

message.sender.email = AUTHOR\_EMAIL;

message.subject = "ESP Test Email";

message.addRecipient("Sara", RECIPIENT\_EMAIL);

/\*Send HTML message\*/

String htmlMsg = "<div style=\"color:#2f4468;\"><h1>Hello World!</h1><p>- Sent from ESP board</p></div>";

message.html.content = htmlMsg.c\_str();

message.html.content = htmlMsg.c\_str();

message.text.charSet = "us-ascii";

message.html.transfer\_encoding = Content\_Transfer\_Encoding::enc\_7bit;

/\*

//Send raw text message

String textMsg = "Hello World! - Sent from ESP board";

message.text.content = textMsg.c\_str();

message.text.charSet = "us-ascii";

message.text.transfer\_encoding = Content\_Transfer\_Encoding::enc\_7bit; message.priority = esp\_mail\_smtp\_priority::esp\_mail\_smtp\_priority\_low; message.response.notify = esp\_mail\_smtp\_notify\_success | esp\_mail\_smtp\_notify\_failure | esp\_mail\_smtp\_notify\_delay;\*/

/\* Set the custom message header \*/

//message.addHeader("Message-ID: <abcde.fghij@gmail.com>");

/\* Connect to server with the session config \*/

if (!smtp.connect(&session))

return;

/\* Start sending Email and close the session \*/

if (!MailClient.sendMail(&smtp, &message))

Serial.println("Error sending Email, " + smtp.errorReason());

}

void loop(){

}

/\* Callback function to get the Email sending status \*/

void smtpCallback(SMTP\_Status status){

/\* Print the current status \*/

Serial.println(status.info());

/\* Print the sending result \*/

if (status.success()){

Serial.println("----------------");

ESP\_MAIL\_PRINTF("Message sent success: %d\n", status.completedCount()); ESP\_MAIL\_PRINTF("Message sent failled: %d\n", status.failedCount()); Serial.println("----------------\n");

struct tm dt;

for (size\_t i = 0; i < smtp.sendingResult.size(); i++){

/\* Get the result item \*/

SMTP\_Result result = smtp.sendingResult.getItem(i);

time\_t ts = (time\_t)result.timestamp;

localtime\_r(&ts, &dt);

ESP\_MAIL\_PRINTF("Message No: %d\n", i + 1);

ESP\_MAIL\_PRINTF("Status: %s\n", result.completed ? "success" : "failed"); ESP\_MAIL\_PRINTF("Date/Time: %d/%d/%d %d:%d:%d\n", dt.tm\_year + 1900, dt.tm\_mon + 1, dt.tm\_mday, dt.tm\_hour, dt.tm\_min, dt.tm\_sec);

ESP\_MAIL\_PRINTF("Recipient: %s\n", result.recipients.c\_str()); ESP\_MAIL\_PRINTF("Subject: %s\n", result.subject.c\_str()); }

Serial.println("----------------\n");

}

}