**INTERNET OF THINGS MAJOR PROJECT**

Name : Jaidev Anandhev

Batch : April 5th

Course Opted : Internet of things.

Project Abstract :

This project is about creating a smart weather system and checking the weather system using Arduino UNO and simulating the circuit using Tinker CAD.

**TABLE OF CONTENTS :**

* Introduction of Arduino
* Features of Smart weather monitoring system
* Hardware required.
* Software Required.
* Brief explanation of ESP8299 Wi-Fi Module.
* Brief explanation of Temperature sensor (LM35).
* Brief explanation of Photoresistor (LDR).
* Steps to be followed.
* Results
* Code used.
* Conclusion

**INTRODUCTION OF ARDUINO :**

Arduino is a microcontroller platform that allows you to create interactive projects using electronics, programming, and creativity. It is an open-source platform, which means that the hardware and software are freely available to everyone. This makes it a great option for beginners and experienced users alike. Arduino boards are made up of a microcontroller, which is a small computer that can be programmed to perform specific tasks. The microcontroller is connected to a variety of input and output ports, which allow you to interact with the physical world. For example, you can use input ports to read data from sensors, such as light sensors or buttons. You can use output ports to control devices, such as motors or LEDs. Arduino is programmed using a simple programming language called C++. C++ is a powerful language that allows you to create complex projects. However, Arduino also includes a simplified version of C++ called Arduino Sketches, which is designed for beginners.

**FEATURES OF SMART WEATHER MONITORING SYSTEM :**

1. Ease of monitoring your local weather conditions in real time from anywhere in the world.
2. For storing weather and environment data for short and long term for studying weather pattern changes and to understand how human induced climate change affected your local weather.
3. Easy deployment of the setup for monitoring local atmospheric conditions and microclimates for weather forecasting and prediction.

**HARDWARE REQUIRED:**

1. Arduino UNO Board
2. 2 Small sized breadboards.
3. Potentiometer
4. LCD 16 \* 2 Display.
5. Wi-Fi ESP8266 Module.
6. 4 (1 KΩ) Resistors.
7. Photoresistor
8. Temperature Sensor (TMP36)
9. Jumper Wires
10. Microbot USB Cable A- B for Arduino UNO

**SOFTWARE REQUIRED :**

1. Thingspeak API ( Used for collecting the record of values).
2. Tinker CAD (For simulating the circuit).

**BRIEF EXPLANATION OF ESP8299 WI-FI MODULE:**



The Arduino Uno Wi-Fi is an Arduino Uno with an integrated Wi-Fi module. The board is based on the ATmega328P with an ESP8266WiFi Module integrated. The ESP8266WiFi Module is a self-contained SoC with integrated TCP/IP protocol stack that can give access to your Wi-Fi network (or the device can act as an access point). One useful feature of Uno Wi-Fi is support for OTA (over-the-air) programming, either for transfer of Arduino sketches or Wi-Fi firmware.

The Arduino Uno Wi-Fi is programmed using the Arduino Software (IDE), our Integrated Development Environment common to all our boards and running both online and offline.

**USES:**

It is used to send the records collected by the sensors through Arduino Board and sends the data via internet the IoT Cloud platform like Thingspeak API.

**BRIEF EXPLANATION OF TEMPERATURE SENSOR (LM35):**

A close-up of a transistor

Description automatically generated

LM35 is a temperature sensor which can measure temperature in the range of -55°C to 150°C.

It is a 3-terminal device that provides analog voltage proportional to the temperature. Higher the temperature, higher is the output voltage.

The output analog voltage can be converted to digital form using ADC so that a microcontroller can process it.

Connect VCC or Supply pin of LM35 to 5v of Arduino UNO (3.3v is also allowed), the Ground pin of LM35 to Ground Pin of Arduino UNO, Input Pin of LM35 to A2 of Arduino UNO

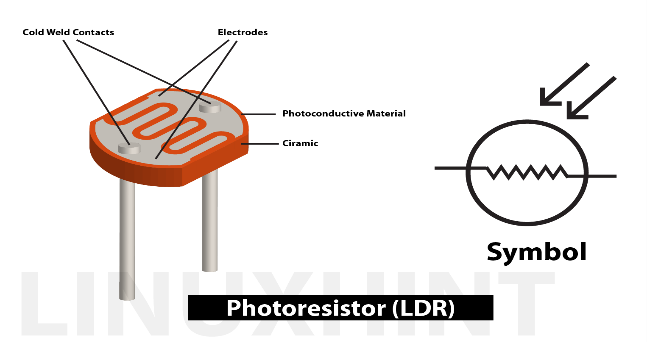
**USES:**

It is used to collect the temperature of the surroundings and the data is collected by Arduino board.

**BRIEF EXPLANATION OF PHOTORESISTOR (LDR):**

A picture containing circuit component, passive circuit component, electronic component, hardware programmer

Description automatically generated



Photoresistors are analog sensors that can measure light. They are light dependent resistors that change resistance with the amount of light that hits the sensor. Photoresistors are perfect for making light controlled switches. One common application of photoresistors is to control 5V relays to switch on a light bulb when it gets dark. They are also used to track daylight in weather stations and environmental monitoring system.

**USES :**

It uses light as the resistor to control the lighting of the system and also in this project we use to find the light index of sunlight.

**STEPS TO BE FOLLOWED:**

**Step 1 :**

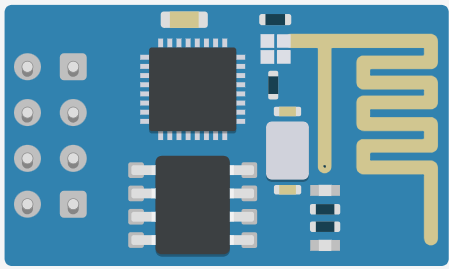
Connect the required components like Potentiometer, Resistors, Photoresistor(LDR) and Temperature Sensor to the breadboard.

**Step 2 :**

Connect the LCD 16 \* 2 display with the breadboard.

**Step 3 :**

Take the ESP8266 Wi-Fi Module and connect using the below pin connections:



Tx



Ground



Enable



GPIO 2 and GPIO 0



Power



Rx



Reset



* Connect Tx to the Digital Input (D0) of Arduino board/
* Connect the ground below the Second resistor.(Refer the Diagram below).

A picture containing diagram, rectangle, line, screenshot

Description automatically generated

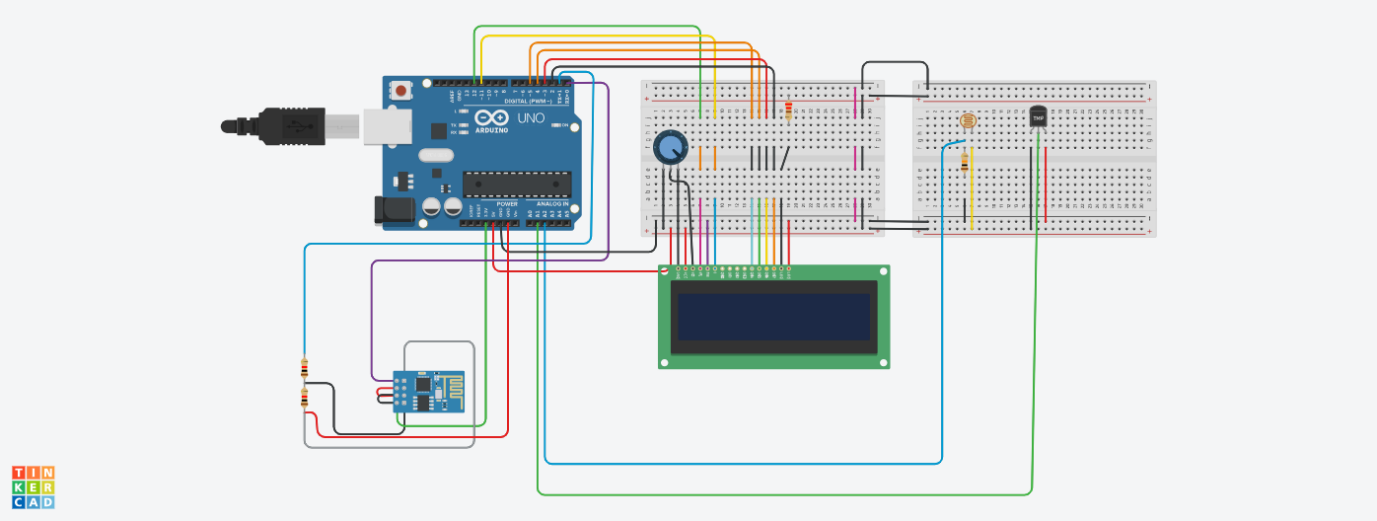


* Connect Enable with Reset of ESP8266 Wi-Fi Module.
* Connect Reset pin with Power pin of ESP8266 Wi-Fi Module.
* Connect the Power pin to the 3.3V power slot of Arduino UNO Board.
* Connect Rx between the connections of two resistors. (Refer the Diagram below).

A picture containing diagram, rectangle, line, screenshot

Description automatically generated

**Step 4 :**



Connections are made as per the circuit diagram in the breadboard and run the code to collect the data from Arduino board and send the data to IoT Cloud platform (Thingspeak API).

**RESULTS:**

A computer screen shot of a circuit board

Description automatically generated with low confidence

\*(Circuit Connections in Tinker CAD)

A picture containing text, screenshot, display, diagram

Description automatically generated

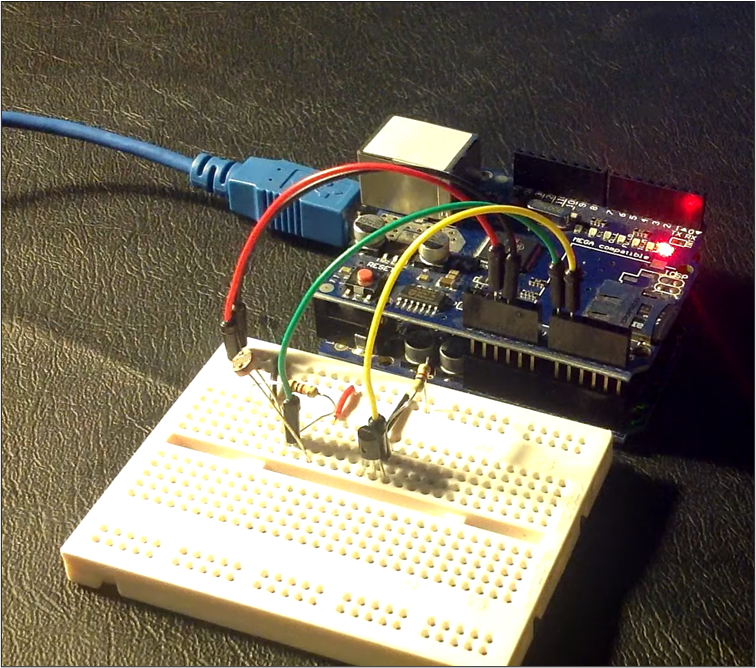
\*(Output in LCD Panel)

\*(It shows the temperature and sunlight level)

A screenshot of a computer

Description automatically generated

\*(Real data collected from Tinker CAD simulated circuit)



\*(LCD Display was not Available) and project done by hardware did not show the results in Thingspeak API because ESP8266 was not working properly so it is done using Tinker CAD to simulate the circuit.

**CODE USED :**

#include <LiquidCrystal.h>

String ssid = "Simulator Wifi"; // SSID to connect to

String password = ""; // Our virtual wifi has no password

String host = "api.thingspeak.com"; // Open Weather Map API

const int httpPort = 80;

String uri1 = "/update?api\_key=C9HH6ZXB6RQB89MH&field1=";

String uri2 = "/update?api\_key=C9HH6ZXB6RQB89MH&field2=";

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

int val = 0;

int tempPin = A1;

int light\_sensor=A2;

int setupESP8266(void) {

// Start our ESP8266 Serial Communication

Serial.begin(115200); // Serial connection over USB to computer

Serial.println("AT"); // Serial connection on Tx / Rx port to ESP8266

delay(10); // Wait a little for the ESP to respond

if (!Serial.find("OK")) return 1;

// Connect to 123D Circuits Simulator Wifi

Serial.println("AT+CWJAP=\"" + ssid + "\",\"" + password + "\"");

delay(10); // Wait a little for the ESP to respond

if (!Serial.find("OK")) return 2;

// Open TCP connection to the host:

Serial.println("AT+CIPSTART=\"TCP\",\"" + host + "\"," + httpPort);

delay(50); // Wait a little for the ESP to respond.

if (!Serial.find("OK")) return 3;

return 0;

}

void tempdata(int tempData) {

int temp = tempData;

// Construct our HTTP call

String httpPacket = "GET " + uri1 + String(temp) + " HTTP/1.1\r\nHost: " + host + "\r\n\r\n";

int length = httpPacket.length();

// Send our message length

Serial.print("AT+CIPSEND=");

Serial.println(length);

delay(10); // Wait a little for the ESP to respond if (!Serial.find(">")) return -1;

// Send our http request

Serial.print(httpPacket);

delay(10); // Wait a little for the ESP to respond

if (!Serial.find("SEND OK\r\n")) return;

}

void lightdata(int light\_reading) {

int lightvalue = light\_reading;

// Construct our HTTP call

String httpPacket = "GET " + uri2 + String(lightvalue) + " HTTP/1.1\r\nHost: " + host + "\r\n\r\n";

int length = httpPacket.length();

// Send our message length

Serial.print("AT+CIPSEND=");

Serial.println(length);

delay(10); // Wait a little for the ESP to respond if (!Serial.find(">")) return -1;

// Send our http request

Serial.print(httpPacket);

delay(10); // Wait a little for the ESP to respond

if (!Serial.find("SEND OK\r\n")) return;

}

void showlight(int light\_reading){

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Sunlight level");

lcd.setCursor(0, 1);

lcd.print(String(light\_reading)+ " mW/cm2");

delay(1000);

}

void showtemp(int cel){

lcd.clear();

lcd.setCursor(0, 0);

lcd.print("Temperature");

lcd.setCursor(0, 1);

lcd.print(String(cel)+ "° Celcius");

delay(1000);

}

void setup() {

Serial.begin(9600);

lcd.begin(16, 2);

setupESP8266();

}

void loop() {

val = analogRead(tempPin);

float mv = val\*(5000/1024);

float cel = mv/10;

float farh = (cel\*9)/5 + 32;

int light\_reading;

light\_reading = analogRead(light\_sensor);

showlight(light\_reading);

showtemp(cel);

tempdata(cel);

lightdata(light\_reading);

}

**CONCLUSION:**

Thus the weather monitoring system using Arduino UNO board and simulating the circuit in Tinker CAD and finally sending the statistical record to IoT cloud platform named Thingspeak API using API Key has been completed successfully.