Practical No 22

AIM: To interface 16×2 LCD, Push Button, Potentiometer with Arduino and WAP to

display message on LCD when push button is pressed.

Objectives:

To learn Arduino UNO basics

To Learn Breadboard basics

To Learn about 16x2 LCD.

To Learn about Potentiometer

WAP to display message on LCD when push button is pressed.

Theory:

Concept of Arduino UNO:
Arduino/Genuino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

Breadboard:
A breadboard allows for easy and quick creation of temporary electronic circuits or to carry out experiments with circuit design. Breadboards enable developers to easily connect components or wires thanks to the rows and columns of internally connected spring clips underneath the perforated plastic enclosure.

Resistor:
A resistor is an electrical component that limits or regulates the flow of electrical current in an electronic circuit. Resistors can also be used to provide a specific voltage for an active device such as a transistor. All other factors being equal, in a direct-current (DC) circuit, the current through a resistor is inversely proportional to its resistance, and directly proportional to the voltage across it. This is the wellknown Ohm's Law. In alternating-current (AC) circuits, this rule also applies as long as the resistor does not contain inductance or capacitance.

Jumper Wires: A jumper wire is an electric wire that connects remote electric circuits used for printed circuit boards. B y attaching a jumper wire on the circuit, it can be short-circuited and short-cut (jump) to the electric circuit.

16x2 LCD:

A 16x2 LCD means **it can display 16 characters per line and there are 2 such lines**. In this LCD each character is displayed in 5x7 pixel matrix. The 16 x 2 intelligent alphanumeric dot matrix display is capable of displaying 224 different characters and symbols. This LCD has two registers, namely, Command and Data.

Potentiometer:

A potentiometer is **a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider**. If only two terminals are used, one end and the wiper, it acts as a variable resistor or rheostat. Potentiometer. A typical single-turn potentiometer.

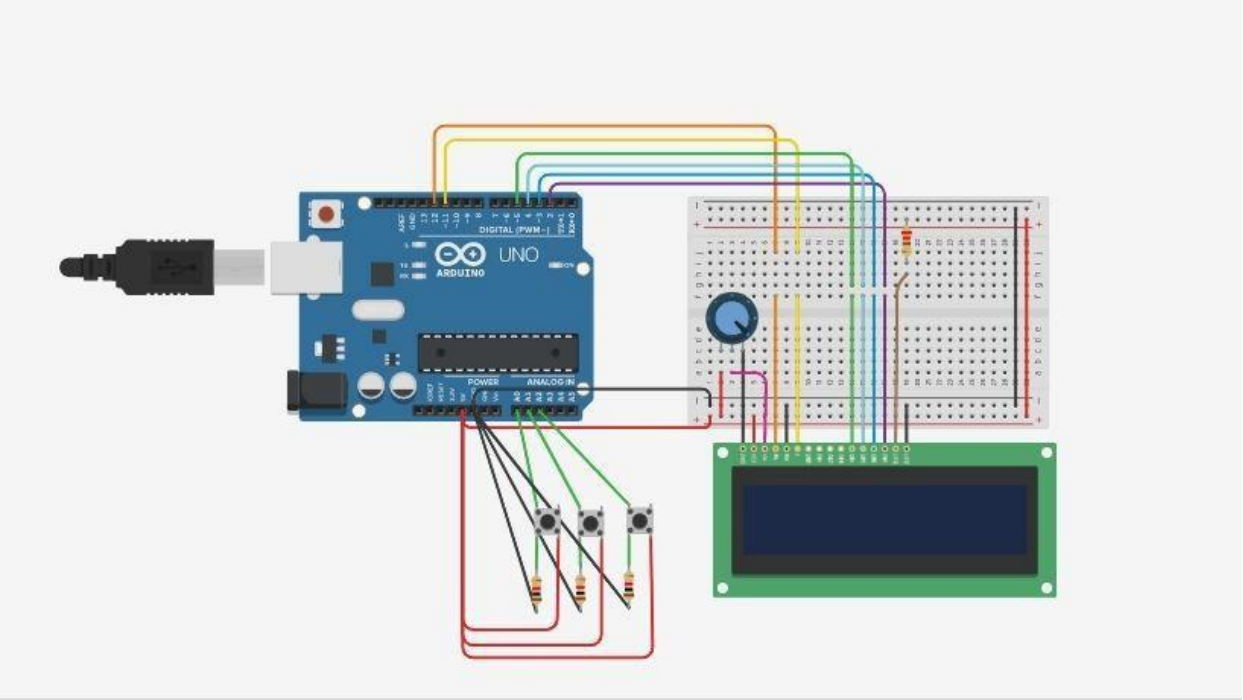
Functions:

setup(): The setup() function is called when a sketch starts. Use it to initialize variables, pin modes, start using libraries, etc. The setup function will only run once, after each powerup or reset of the Arduino board. loop():After creating a setup() function, which initializes and sets the initial values, the loop() function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board. **pinmode():**The pinMode() function is used to configure a specific pin to behave either as an input or an output. It is possible to enable the internal pull-up resistors with the mode INPUT_PULLUP. Additionally, the INPUT mode explicitly disables the internal pull-ups.

delay(): The way the delay() function works is pretty simple. It accepts a single integer (or number) argument. This number represents the time (measured in milliseconds). The program should wait until moving on to the next line of code when it encounters this function. digitalwrite() : Write a HIGH or a LOW value to a digital pin. If the pin has been configured as an OUTPUT with pinMode() , its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH , 0V (ground) for LOW.

digitalRead(): The digitalRead() function is used to read the logic state at a pin. It is capable to tell wether the voltage at this pin is high (~ 5V) or low (~ 0V) or, in other words, if the pin is at logic state 1 or 0 (or HIGH/LOW).

Circuit Diagram:

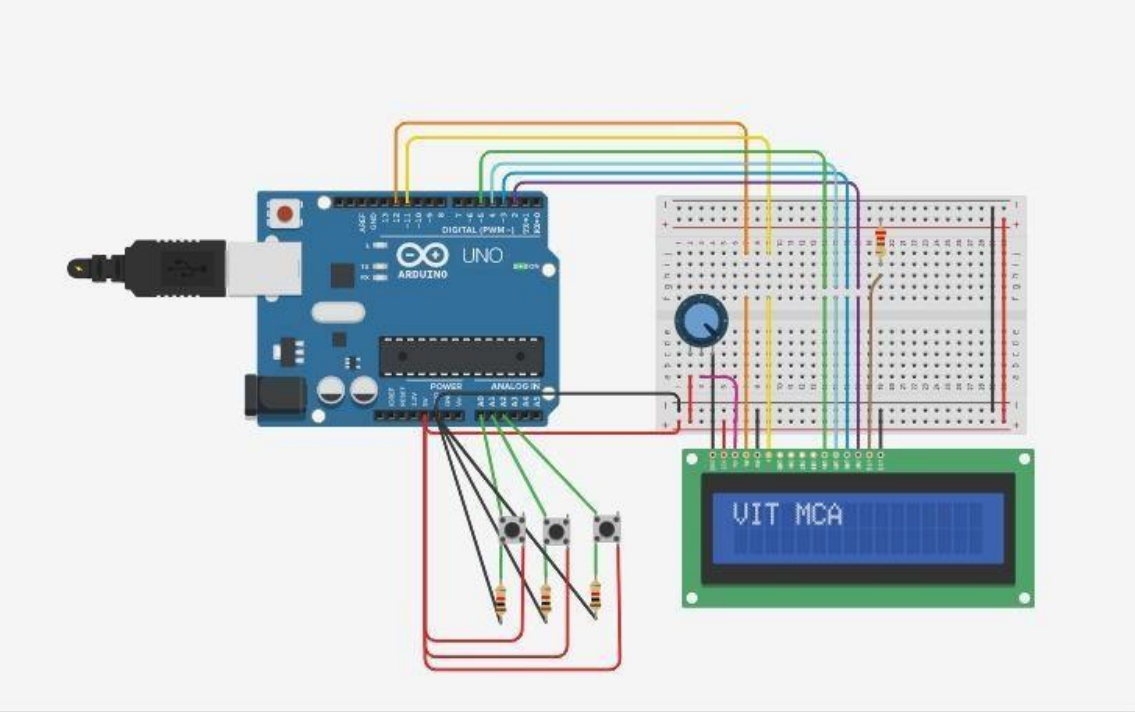


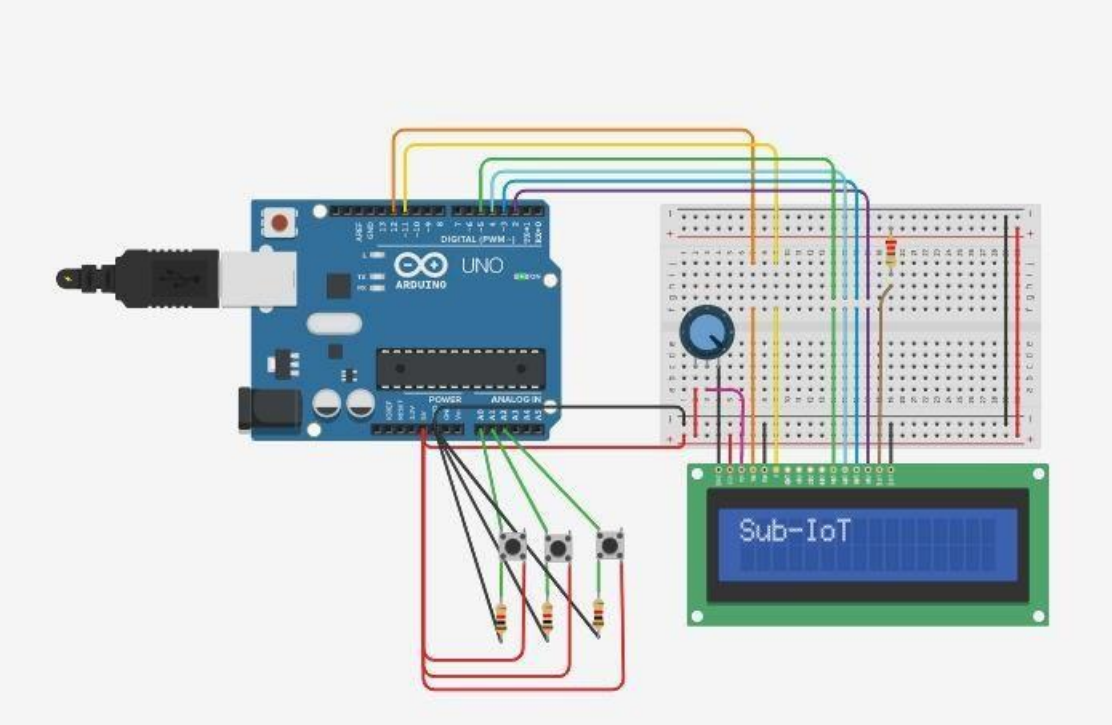
Code:

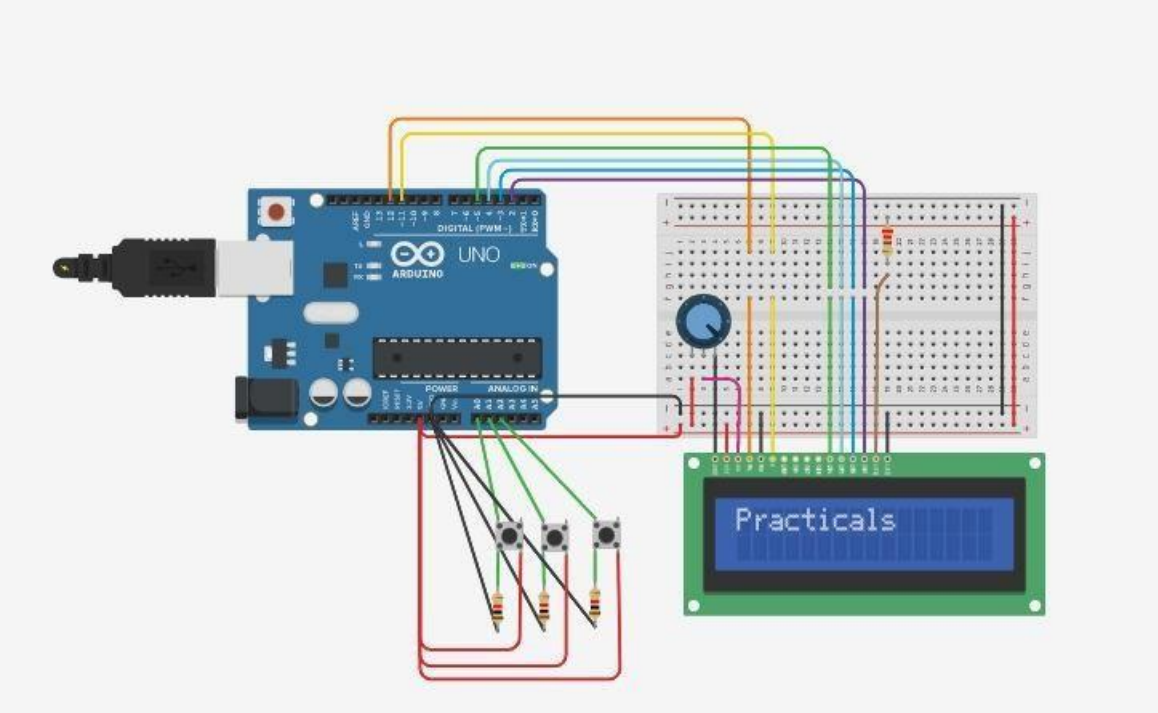
```
// include the library code:
#include <LiquidCrystal.h>
// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
void setup()
{
 pinMode(A0, INPUT);
 pinMode(A1, INPUT);
 pinMode(A2, INPUT);
}
void loop()
{
// set the cursor to column 0, line 1
//(note: line 1 is the second row, since counting begins with 0):
lcd.setCursor(0, 1);

 if (digitalRead(A0) == HIGH)
 {
 lcd.print( "VIT MCA");
 }
if (digitalRead(A1) == HIGH)
```

```
 {
 lcd.print("Sub-IoT");
 }

if (digitalRead(A2) == HIGH)
 {
 lcd.print("Practicals");
 }
}
```

OUTPUT:




UNO
ARDUINO
Sub-IoT


UNO
ARDUINO
Practicals

Conclusion:

Thus, learnt about basic components of IoT like Arduino UNO (blink Arduino onboard LED), Breadboard , 7 segment display and interfacing 16×2 LCD, Push Button, Potentiometer with Arduino and displaying message on LCD when push button pressed with connections of resistors, breadboard, Jumper wires & Arduino.

Practical No. 23

Part 1

Aim: To upload data on Thingspeak cloud

Manually.Steps:

1. Go to Google and search for Thingspeak.

2. If you are new to Thingspeak, Do sign up and make sure you are on your Channel page.

3. Click on the NEW CHANNEL button (Green color) and create a new channel.

4. Enter a channel name, any description of your choice, and make sure one field is selected orticked and give that field a name of your choice. Click on save.

5. Now in the private view, make sure you see a graph (empty).

6. Now click on API KEYS tab, scroll down to find API requests section and in that copy the link of Write a Channel Feed and paste it in the Address bar of your browser. And press enterto get a blank screen with a number which indicates the number of data uploaded manually. Following is the example link:

https://api.thingspeak.com/update?api_key=6WEDQNFN3GBKNCQ3&field1=0

7. Suppose you want to change the data to be entered in the graph, just change the

8. =0 to any value of your choice in the link. Above is the link, where we had changed 0 to 40.

https://api.thingspeak.com/update?api_key=6WEDQNFN3GBKNCQ3&field1=40

9. As a result, to see the visualization (graph), go to the private view and see the graph.

Practical No. 23
(Part 2)

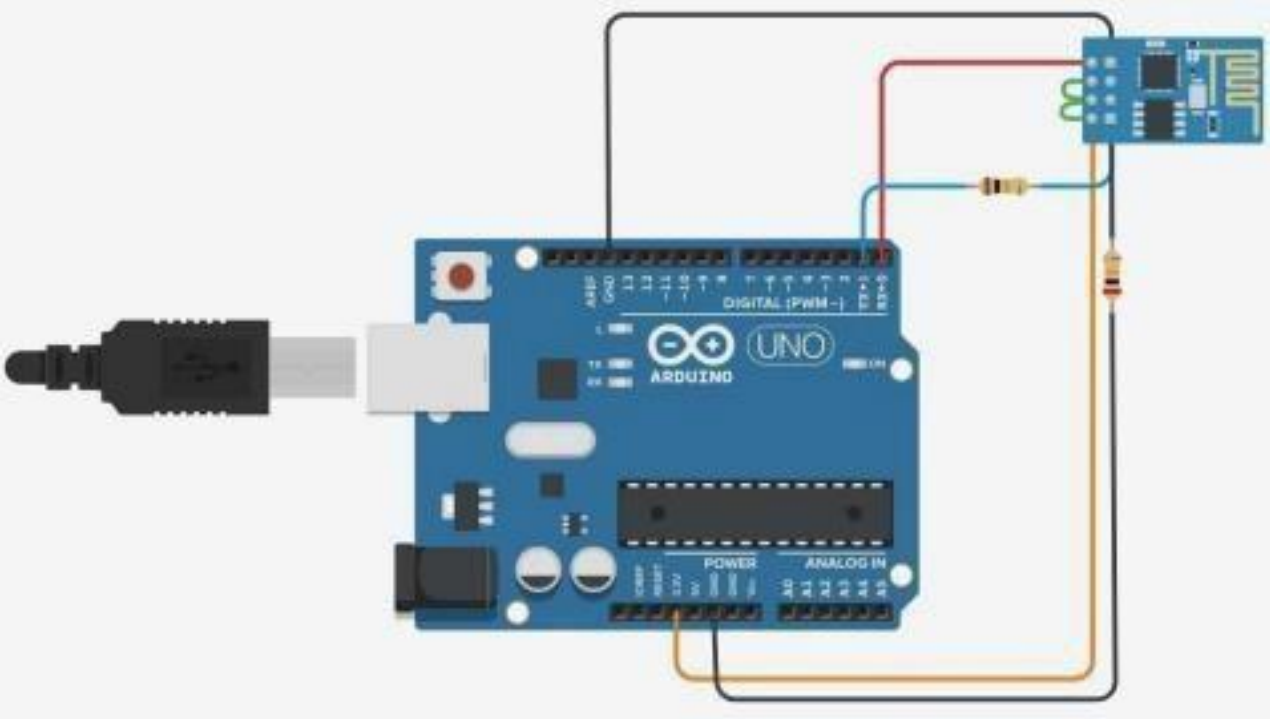
Aim: To update readings to Thingspeak from Arduino

Using Tinkercad.Steps:

1. Click on the NEW CHANNEL button (Green colour) and create a new channel.

2. Enter a channel name, any description of your choice, and make sure one field is selected or tickedand give that field a name of your choice. Click on save.

3. Now in the API Keys tab copy the Write API Key and Paste it in your program



PROGRAM:

```
void setup() {

  Serial.begin(1152

  00);delay(1000);

  //if you want thingsspeak through tinkercad use simulator
  wifi as your ssidSerial.println("AT+CWJAP=\"Simulator
  Wifi\",\"\"\r\n");
  delay(3000);
  }
```

```
void loop() {
{

  Serial.println("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\

  n"); delay(5000);int len = 57;//length of line 15

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

  delay(10);
  Serial.print("GET
/update?api_key=ZRGGNNASXTIB4M3B&field1=120
HTTP/1.1\r\n"); // Change the field value to see the
variations in the data
  delay(100);

  Serial.println("AT+CIPCLOSE=0

  \r\n");delay(6000);
}
}
```

NOTE:

1. TEXT IN RED COLOUR IS YOUR WRITE API KEY
2. Make sure you see your Serial Monitor and check if Data is sent to Thingspeak. 3. To check result about data upload, go to thingspeak, click on private view and see the graph.

Practical No. 23
(Part 3)

Aim: To interface Temperature sensor and ESP8266 with Arduino and

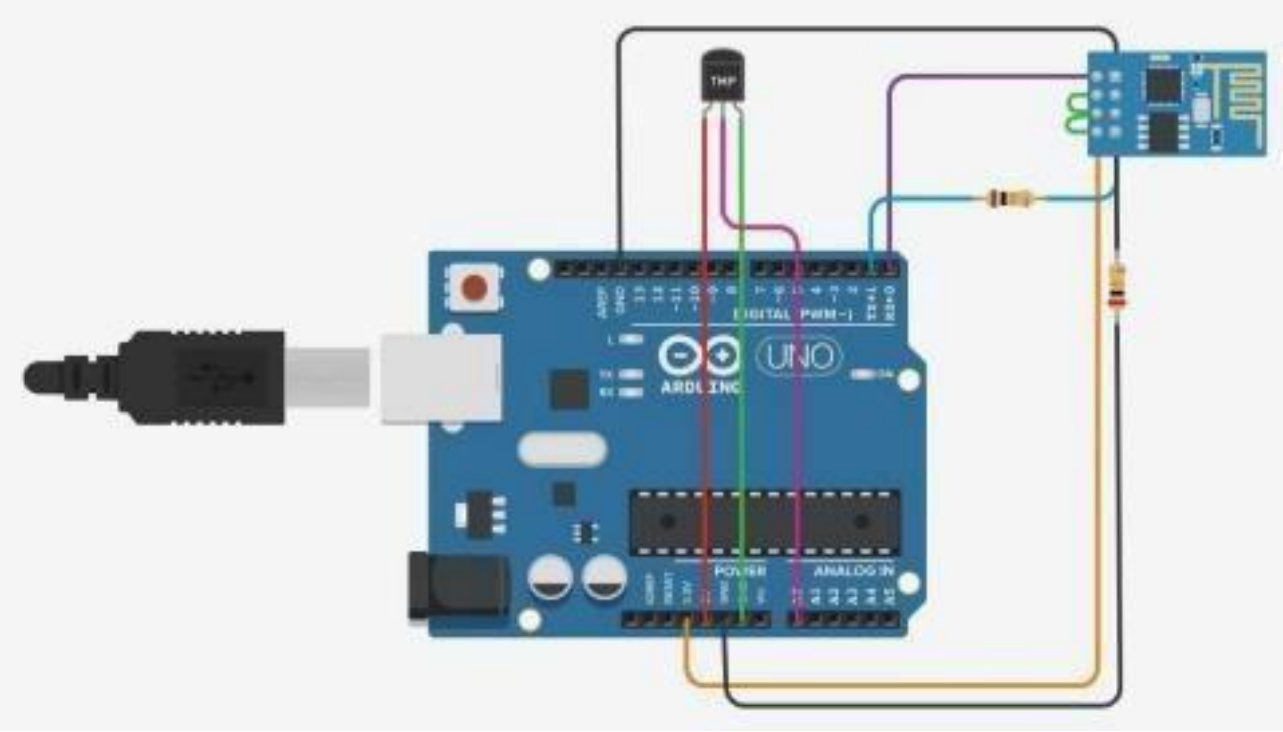
update temperature reading toThingspeak.

Steps:

1. Click on the NEW CHANNEL button (Green colour) and create a new channel.

2. Enter a channel name, any description of your choice, and make sure one field is selected or tickedand give that field a name of your choice. Click on save.

3. Now in the API Keys tab copy the Write API Key and Paste it in your program



PROGRAM:

```
void setup() {
Serial.begin(1152
00);
  Delay(1000);
  Serial.println("AT+CWJAP=\"Simulator Wifi\",\"\"\r\n"); delay(3000);
}
```

```
void loop() {
{
int sensorValue = analogRead(A0);
  float volt = (sensorValue/1020.0) * 4.9;
  //Volts floattempC = (volt -0.5) * 100;
  //Celcius Serial.println(tempC);


  Serial.println("AT+CIPSTART=\"TCP\",\"api.thingspeak.com\",80\r\
  n"); delay(5000);int len = 65; Serial.print("AT+CIPSEND=");
  Serial.println(l
  en);delay(10);
  Serial.print("GET /update?api_key=EDLBQ1UJ9ZLNXD57&field1=" +
String(tempC) +" HTTP/1.1\r\n");
  delay(100);
  Serial.println("AT+CIPCLOSE=0
  \r\n");delay(6000);
}
}
```

NOTE:

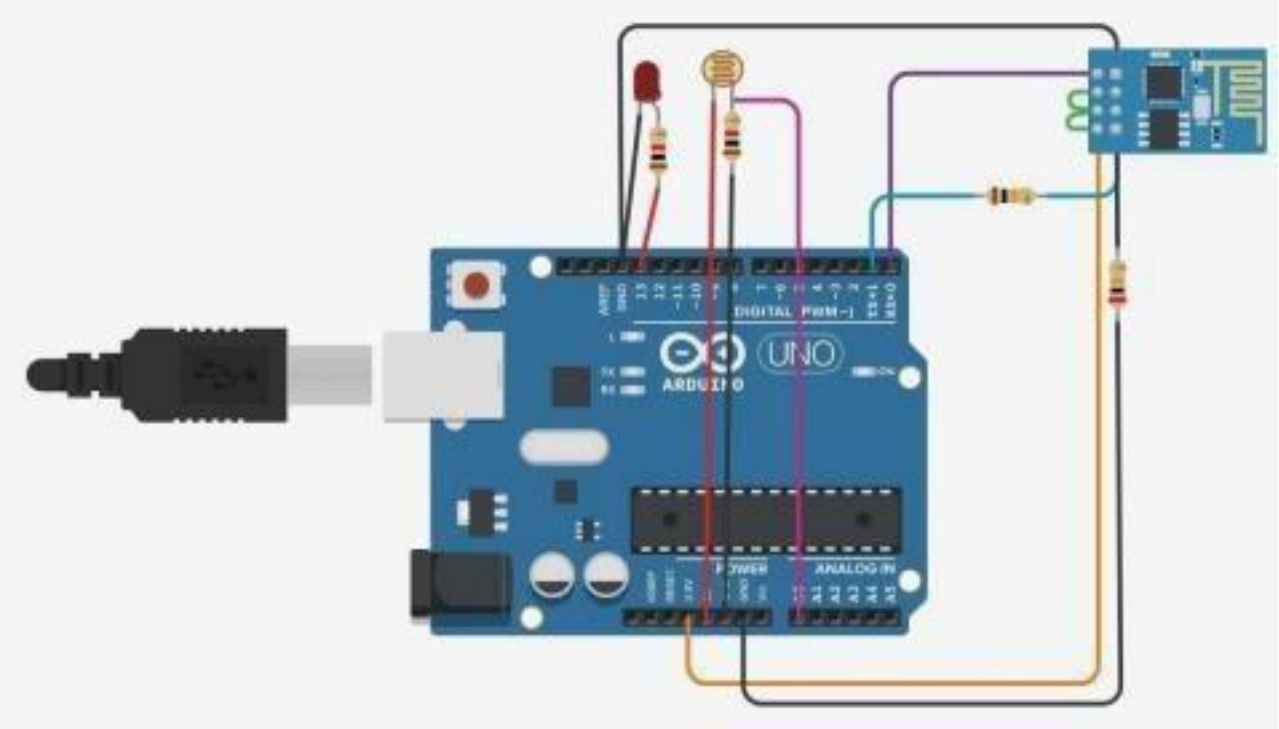
1. TEXT IN RED COLOUR IS YOUR WRITE API KEY
2. Make sure you see your Serial Monitor and check if Data is sent to Thingspeak.
3. To check result about data upload, go to thingspeak, click on private view and see the graph.

Practical No. 24

Aim: To interface LDR sensor, LED and ESP8266 with Arduino and update light intensity values toThingspeak and tweet “LIGHT ON” message on tweeter when light intensity value is less than 300.

Steps:

1. Click on the NEW CHANNEL button (Green colour) and create a new channel.
2. Enter a channel name, any description of your choice, and make sure one field is selected or tickedand give that field a name of your choice. Click on save.
3. Now in the API Keys tab copy the Write API Key and Paste it in your program



PROGRAM:

```
int ldr=A0;//Set A0(Analog Input)
for LDR.int value=0;
void setup()
{
  Serial.begin(115200
  );
```

```
  pinMode(13,OUTP
  UT);delay(1000);
  Serial.println("AT+CWJAP=\"Simulator
  Wifi\",\"\"\r\n");delay(3000);
}


void loop()
{
  {
    value=analogRead(ldr);
          Serial.println("LDR  value
          is  :");Serial.println(value);
          if(value<300)
          {
    digitalWrite(13,HIGH);
          }
          else
          {
    digitalWrite(13,LOW);//Turns the LED OFF in Light.
          }

    Serial.println("AT+CIPSTART=\"TCP\",\"thingspeak
    .com\",80");delay(5000);

    int len = 65;
    Serial.print("AT+CIPSEN
    D=");Serial.println(len);
    delay(10);
```

```
    Serial.print("GET
 /update?api_key=6WEDQNFN3GBKNCQ3&field1="+ String(value)
 +"HTTP/1.1\r\n");
    delay(100);


    Serial.println("AT+CIPCLOSE=0

    \r\n");delay(6000);

   }

 }
```

NOTE:

1. TEXT IN RED COLOUR IS YOUR WRITE API KEY
2. Make sure you see your Serial Monitor and check if Data is sent to Thingspeak.
3. To check result about data upload, go to thingspeak, click on private view and see the graph.
4. Once you finish doing the above steps go back to Thingspeak and next to the CHANNELS tab ,click on the APPS tab and select React option.
5. Click on the NEW REACT button (Green colour) and give a React name. Here it is LDR Reactand do following settings:

6. In the Action tab in the above figure, select ThingTweet option and then it will ask to link withyour twitter account
7. Link and Choose your Twitter Account.
8. Save the React.
9. Make sure your intensity of LDR is less than 300 in the TinkerCad LDR slider and then checkyour Twitter account for the Tweet of “LIGHT ON” message.

Practical No. 25

Aim: To interface servo motor / DC motor with Arduino and WAP to sweep a servo back and forth through its full range of motion/to control a DC motor.

Objectives:

To learn Arduino UNO basics

To Learn about Potentiometer

To learn DC Motor

To learn about 9V Battery

Theory:

Concept of Arduino UNO:
Arduino/Genuino Uno is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.

Resistor:
A resistor is an electrical component that limits or regulates the flow of electrical current in an electronic circuit. Resistors can also be used to provide a specific voltage for an active device such as a transistor. All other factors being equal, in a direct-current (DC) circuit, the current through a resistor is inversely proportional to its resistance, and directly proportional to the voltage across it. This is the wellknown Ohm's Law. In alternating-current (AC) circuits, this rule also applies as long as the resistor does not contain inductance or capacitance.

Jumper Wires: A jumper wire is an electric wire that connects remote electric circuits used for printed circuit boards. B y attaching a jumper wire on the circuit, it can be short-circuited and short-cut (jump) to the electric circuit.

Potentiometer:

A potentiometer is **a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider**. If only two terminals are used, one end and the wiper, it acts as a variable resistor or rheostat. Potentiometer. A typical single-turn potentiometer.

9V Battery:

A nine-volt battery, either disposable or rechargeable, is usually used in **smoke alarms, smoke detectors, walkie-talkies, transistor radios, test and instrumentation devices, medical batteries, LCD displays, and other small portable appliances**.

DC Motor:

A DC motor is **an electrical machine that converts electrical energy into mechanical energy**. In a DC motor, the input electrical energy is the direct current which is transformed into the mechanical rotation.

NPN transistor:

An NPN transistor is the most commonly used bipolar junction transistor, and is constructed by sandwiching a P-type semiconductor between two N-type semiconductors. An NPN transistor has three terminals– a collector, emitter and base. The NPN transistor behaves like two PN junctions diodes connected back to back

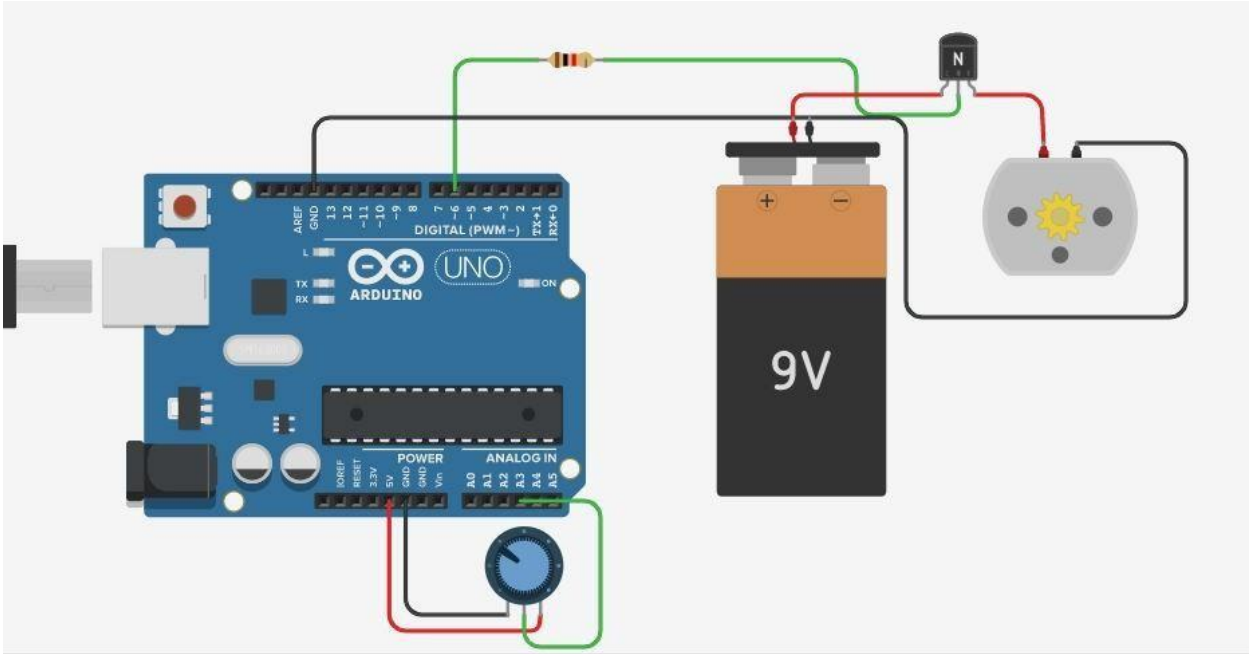
Functions:

setup(): The setup() function is called when a sketch starts. Use it to initialize variables, pin modes, start using libraries, etc. The setup function will only run once, after each powerup or reset of the Arduino board. loop():After creating a setup() function, which initializes and sets the initial values, the loop() function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board. **pinmode():**The pinMode() function is used to configure a specific pin to behave either as an input or an output. It is possible to enable the internal pull-up resistors with the mode INPUT_PULLUP. Additionally, the INPUT mode explicitly disables the internal pull-ups.

delay(): The way the delay() function works is pretty simple. It accepts a single integer (or number) argument. This number represents the time (measured in milliseconds). The program should wait until moving on to the next line of code when it encounters this function. digitalwrite() : Write a HIGH or a LOW value to a digital pin. If the pin has been configured as an OUTPUT with pinMode() , its voltage will be set to the corresponding value: 5V (or 3.3V on 3.3V boards) for HIGH , 0V (ground) for LOW.

digitalRead(): The digitalRead() function is used to read the logic state at a pin. It is capable to tell wether the voltage at this pin is high (~ 5V) or low (~ 0V) or, in other words, if the pin is at logic state 1 or 0 (or HIGH/LOW).

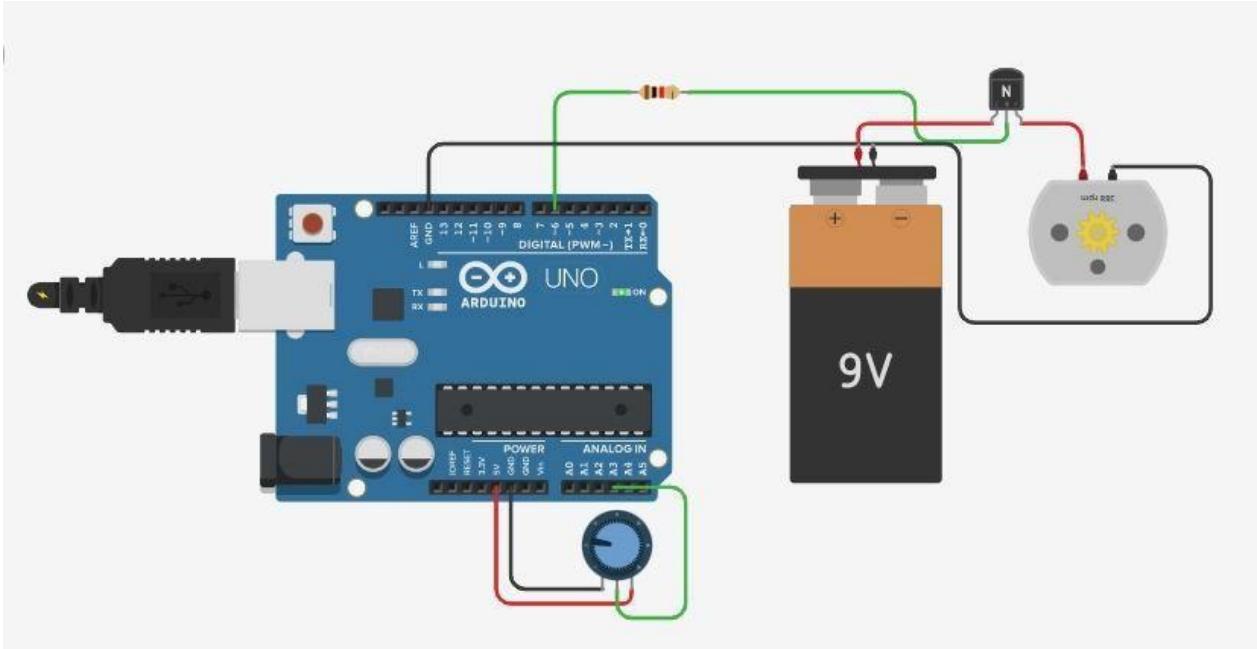
Circuit Diagram:



Code:

```
const int poten = A3;
int var;

void setup()
{
Serial.begin(9600);
pinMode(6, OUTPUT);
}

void loop()
{
var = analogRead(poten);
analogWrite(6, var);
        Serial.println(var);
}
```

OUTPUT:



Conclusion:

Thus, learnt about basic components of IoT like Arduino UNO (blink Arduino onboard LED), Potentiometer with Arduino, resistors, Jumper wires,DC motor,9V battery and NPN resistors.