VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Electronics Engineering

BECE351E – Internet of Things Lab

Submitted by:

21BEC1851 – Rahul Karthik S

Submitted to:

Dr. Soumya Ranjan Mahapatro

Vellore Institute of Technology
Chennai Campus

Programme: B. Tech Electronics and Communication Engineering
BECE351E: Internet of Things (IoT) Lab

Air Quality Monitoring using TinkerCAD

Name of the student:	Rahul Karthik S
Register Number:	21BEC1851
Date of the Lab. Class:	25.05.2023

1. Introduction:

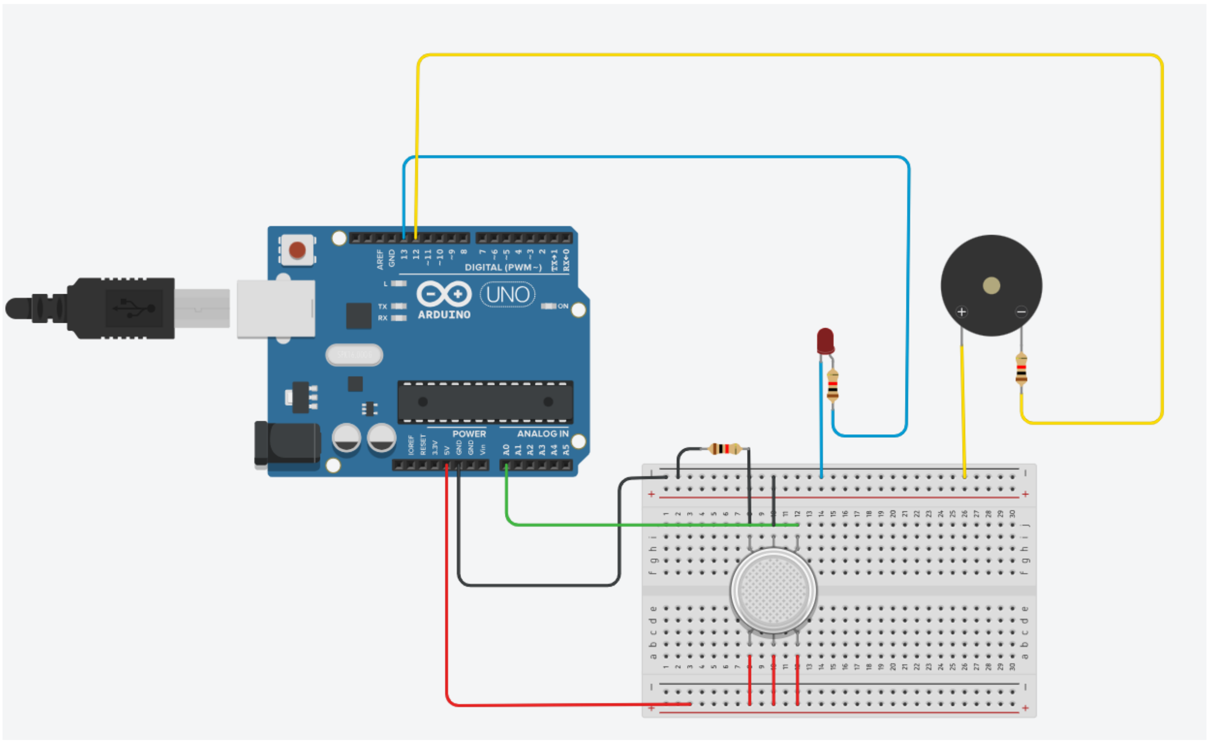
a. What are you targeting to do through this experiment?

To design a circuit to monitor the quality of the air using TinkerCAD.

b. List out the required components to execute this experiment?

Components	Quantity
Arduino UNO R3	1
1 kΩ Resistor	3
Gas Sensor	1
LED	1
Piezo	1

2. Circuit Diagram & Explanation:



a. Write the detailed steps involved to design the circuit diagram in TinkerCAD.

- Begin by setting up a breadboard and connecting a gas sensor to it.
- Attach the B2, H2, and B1 terminals of the gas sensor to the 5V pin on the Arduino board.
- Connect one end of a resistor to the ground pin of the Arduino. Then, connect the other end of the resistor to the A1 end of the gas sensor, and ground the H1 pin of the gas sensor.
- Take the A2 pin of the gas sensor and connect it to the A0 analog pin on the Arduino board.
- Grab an LED and connect its positive end to a resistor. Connect the other end of the resistor to the 12th digital pin on the Arduino board, and ground the remaining end of the LED.
- Ground the positive end of a buzzer. Connect the positive end of the buzzer to the resistor, and attach the other end of the resistor to the 13th digital pin on the Arduino board.

b. Explain how the circuit is functioning to detect pollutant gas.

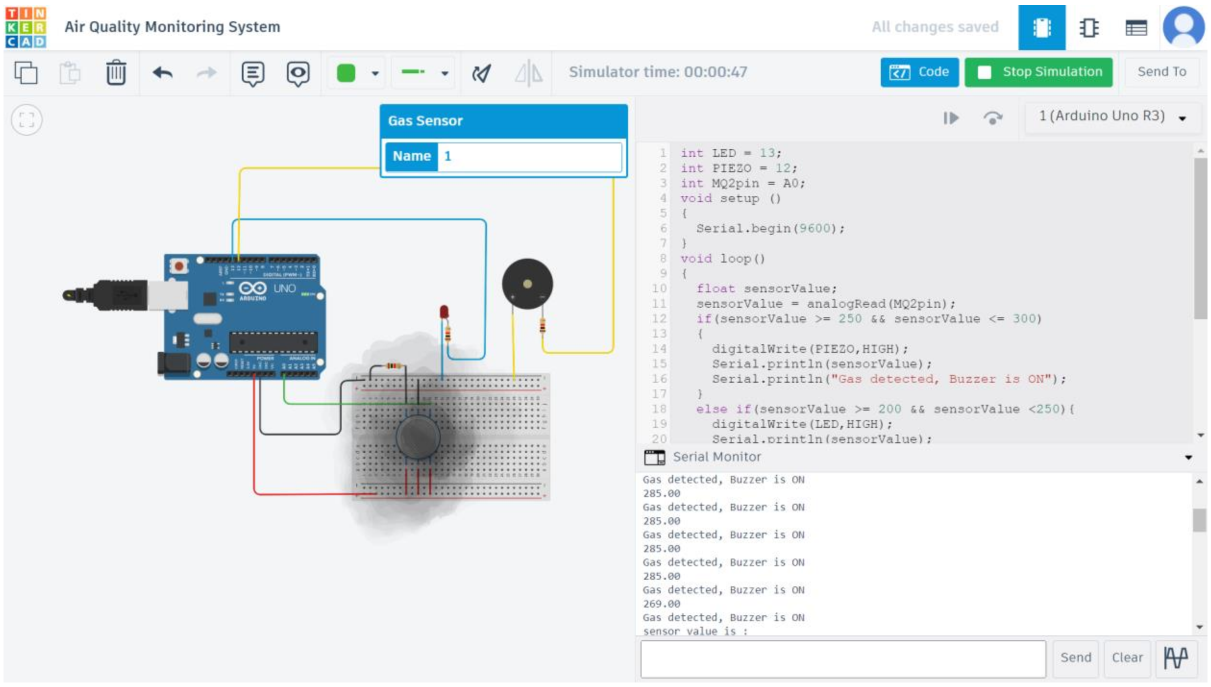
- The gas sensor receives power from a 5V supply and sends its analog data to the Arduino board.
- Based on the board's programming, the LED and buzzer are powered when the sensor value exceeds a specified limit.
- This causes the LED to light up and the buzzer to sound an alarm.
- When the gas leakage is minimal or not dangerous, the power supply to the LED and buzzer is disconnected.
- As a result, the LED turns off, and the buzzer remains silent, indicating the absence of gas or a gas concentration below the threshold.

c. Write and describe the code utilized to run the circuit.

```
int LED = 13;
int PIEZO = 12;
int MQ2pin = A0;
void setup ()
{
  Serial.begin(9600);
}
void loop()
{
  float sensorValue;
  sensorValue = analogRead(MQ2pin);
  if(sensorValue >= 250 && sensorValue <= 300)
  {
    digitalWrite(PIEZO,HIGH);
    Serial.println(sensorValue);
    Serial.println("Gas detected, Buzzer is ON");
  }
  else if(sensorValue >= 200 && sensorValue <250){
          digitalWrite(LED,HIGH);
    Serial.println(sensorValue);
    Serial.println("Gas detected, LED is ON");
  }
  else if(sensorValue <200)
  {
    digitalWrite(LED,LOW);
    digitalWrite(PIEZO,LOW);
    Serial.println("sensor value is :");
    Serial.println(sensorValue);
    Serial.println("No gas Detected");

  }
  delay(1000);
}
```

The provided code is an Arduino sketch that reads values from a gas sensor connected to analog pin A0. Based on the sensor readings, it controls an LED (connected to pin 13) and a buzzer (connected to pin 12). If gas is detected within specific concentration ranges, the buzzer or LED is turned on accordingly. The code also prints messages to the Serial Monitor to indicate the gas levels detected and the status of the LED and buzzer. The loop repeats continuously with a 1-second delay between iterations.

3. Output of the experiment:



```
int LED = 13;
int PIEZO = 12;
int MQ2pin = A0;
void setup ()
{
  Serial.begin(9600);
}
void loop()
{
  float sensorValue;
  sensorValue = analogRead(MQ2pin);
  if(sensorValue >= 250 && sensorValue <= 300)
  {
    digitalWrite(PIEZO,HIGH);
    Serial.println(sensorValue);
    Serial.println("Gas detected, Buzzer is ON");
  }
  else if(sensorValue >= 200 && sensorValue <250){
    digitalWrite(LED,HIGH);
    Serial.println(sensorValue);
```

```
Gas detected, Buzzer is ON
285.00
Gas detected, Buzzer is ON
285.00
Gas detected, Buzzer is ON
285.00
Gas detected, Buzzer is ON
285.00
Gas detected, Buzzer is ON
269.00
Gas detected, Buzzer is ON
sensor value is :
```

a. **Write the detailed steps involved to run the simulation in TinkerCAD.**
   - Click the code button in the top right and type the code in the pane.
   - Then, click the start simulation button and observe the output.

b. **How are you assessing the output displayed in the Serial Monitor?**
   - If the sensor value exceeds 200, a message is displayed indicating "Gas detected." Otherwise, the message displayed is "No gas detected".
   - Additionally, when the sensor value is above 200, the LED lights up, and the buzzer emits a sound, allowing us to determine the presence of gas leakage.

Vellore Institute of Technology
Chennai Campus

Programme: B. Tech Electronics and Communication Engineering
BECE351E: Internet of Things (IoT) Lab

Detection of Soil Moisture using TinkerCAD

Name of the student:	Rahul Karthik S
Register Number:	21BEC1851
Date of the Lab. Class:	01.06.2023

1. Introduction:

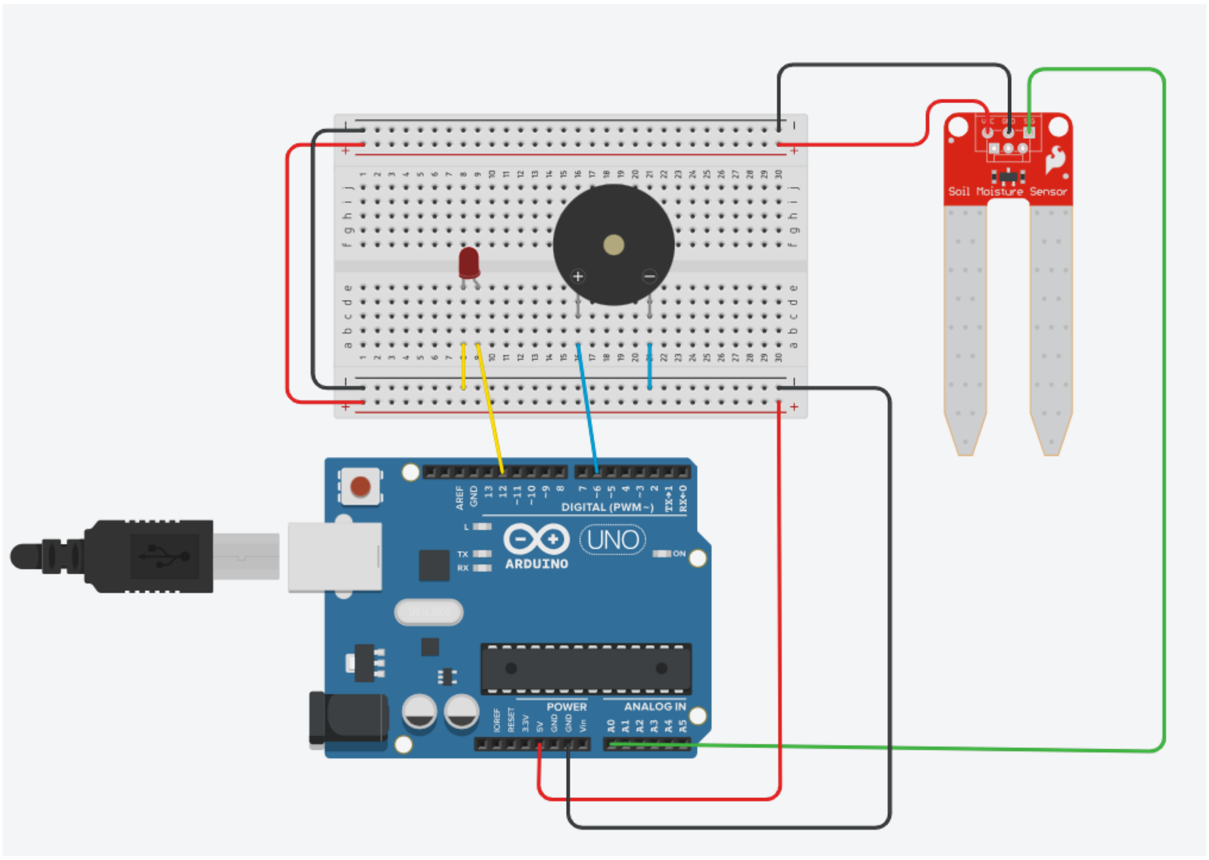
a. What are you targeting to do through this experiment?

To design a circuit to monitor the moisture of the soil using TinkerCAD.

b. List out the required components to execute this experiment?

Components	**Quantity**
Arduino UNO R3	1
Soil Moisture Sensor	1
LED	1
Piezo	1

2. Circuit Diagram & Explanation:



a. **Write the detailed steps involved to design the circuit diagram in TinkerCAD.**

- Gather the required components like Arduino UNO, Soil moisture sensor (analog or digital), Jumper wires, piezo and Breadboard.
- Connect the soil moisture sensor to the Arduino board, for soil moisture sensor:
    - Connect the sensor's VCC pin to Arduino's 5V pin.
    - Connect the sensor's GND (ground) pin to Arduino's GND pin.
    - Connect the sensor's OUT pin to any analog input pin on Arduino (e.g., A0).
- Write the Arduino code and click "Start Simulation".

b. **Explain how the circuit is functioning to detect pollutant gas.**

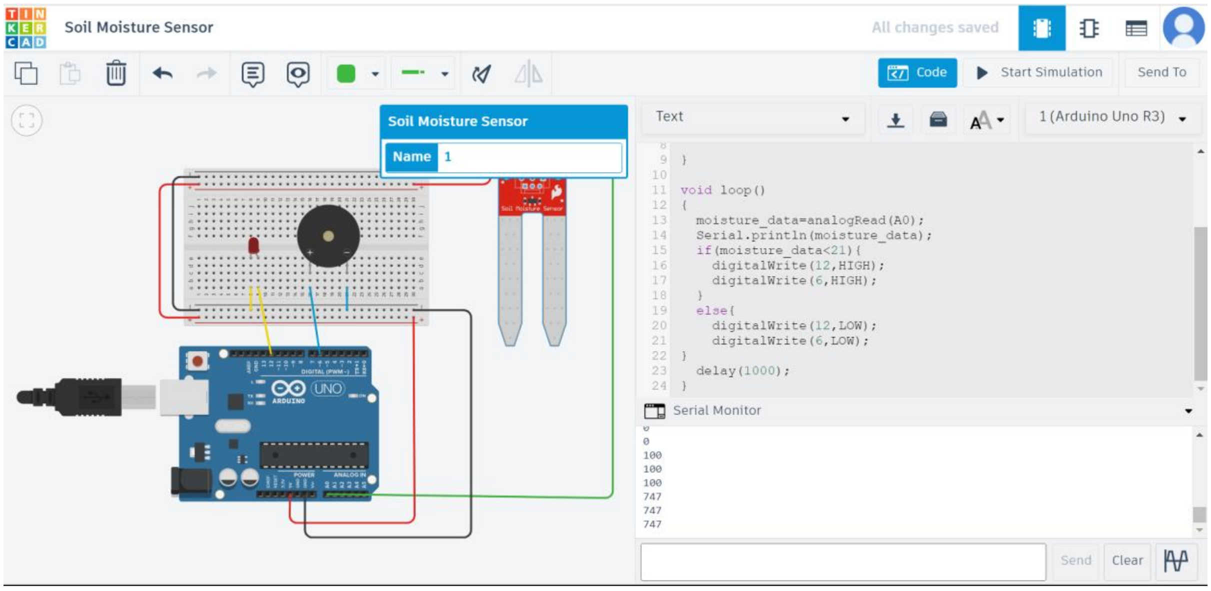
Soil moisture sensors measure water content in soil. Capacitive sensors use changes in capacitance to determine moisture levels. Tensiometers rely on soil water potential and water movement in a tube. TDR sensors measure soil's electrical conductivity by sending pulses. Neutron moisture meters use neutron interaction with hydrogen atoms. Calibration is needed for accurate readings. These sensors help optimize irrigation and assess soil health.

c. **Write and describe the code utilized to run the circuit.**

```
int moisture_data=0;
void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);
  pinMode(12,OUTPUT);
  pinMode(6,OUTPUT);

}

void loop()
{
  moisture_data=analogRead(A0);
  Serial.println(moisture_data);
  if(moisture_data<21){
    digitalWrite(12,HIGH);
    digitalWrite(6,HIGH);
  }
  else{
    digitalWrite(12,LOW);
    digitalWrite(6,LOW);
}
  delay(1000);
}
```

This code reads the moisture level from a sensor connected to pin A0. If the moisture level is less than 21, it turns on two output pins (12 and 6). Otherwise, it turns them off. The moisture level is printed to the serial monitor for monitoring.

3. **Output of the experiment:**



```
}

void loop()
{
  moisture_data=analogRead(A0);
  Serial.println(moisture_data);
  if(moisture_data<21){
    digitalWrite(12,HIGH);
    digitalWrite(6,HIGH);
  }
  else{
    digitalWrite(12,LOW);
    digitalWrite(6,LOW);
}
  delay(1000);
}
```

```
0
0
100
100
100
747
747
747
```

a. **Write the detailed steps involved to run the simulation in TinkerCAD.**

- Click the “Code” button and type the code.
- Then click the “Start Simulation” button.
- Then, adjust the level in the soil moisture sensor to observe the values for different levels of moisture.

b. **How are you assessing the output displayed in the Serial Monitor?**

When the level in the soil moisture sensor is varied, we can see the difference in the values in the serial monitor.

Vellore Institute of Technology
Chennai Campus

Programme: B. Tech Electronics and Communication Engineering
BECE351E: Internet of Things (IoT) Lab

Street Light System using TinkerCAD

Name of the student: Rahul Karthik S
Register Number: 21BEC1851
Date of the Lab. Class: 15.06.2023

1. Introduction:

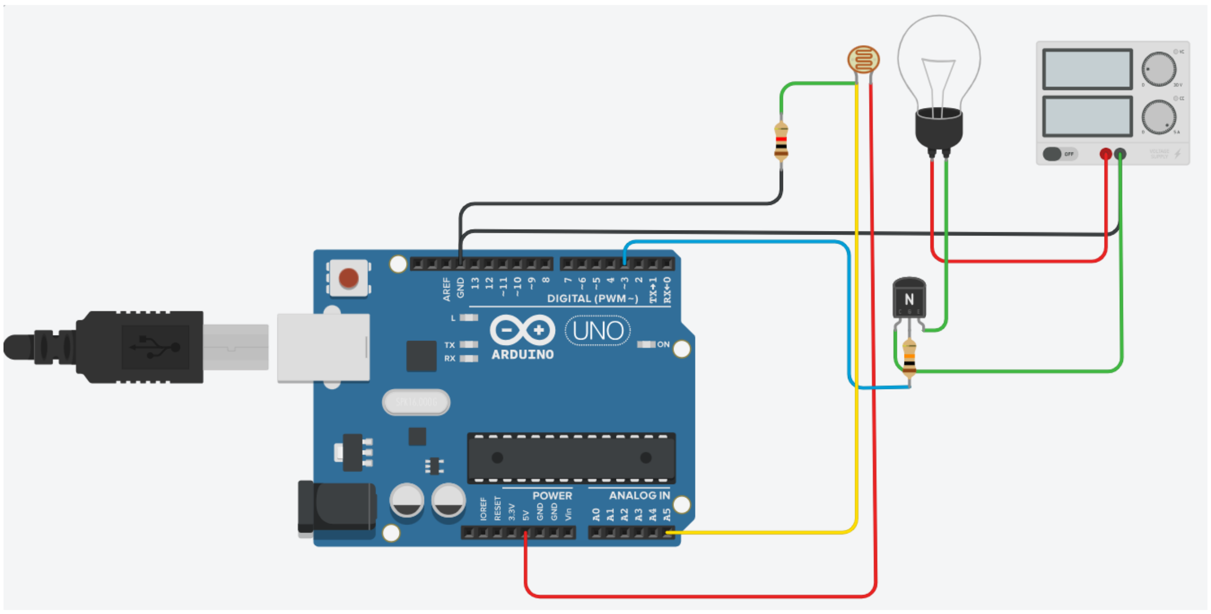
a. What are you targeting to do through this experiment?

To design a circuit to turn on the street light system using TinkerCAD.

b. List out the required components to execute this experiment?

Components	Quantity
Arduino UNO R3	1
1 kΩ Resistor	1
Photo Resistor	1
NPN Transistor	1
5, 5 Power Supply	1
10 kΩ Resistor	1
Light Bulb	1

2. Circuit Diagram & Explanation:



a. Write the detailed steps involved to design the circuit diagram in TinkerCAD.

- Create a new design: Click on the "Create New Design" button to start a new project.
- Select components: In the components panel on the right-hand side, search for and add the following components to your workspace:
    - Arduino UNO R3
    - 1 kΩ Resistor
    - Photo Resistor (Light-Dependent Resistor)
    - NPN Transistor (e.g., BC547)
    - 5V Power Supply (from the Power section)
    - 10 kΩ Resistor
    - Light Bulb (use the lamp component)
- Connect the components:
    - Connect the 5V power supply's positive terminal (+) to the Arduino UNO's 5V pin.
    - Connect the power supply's negative terminal (-) to the Arduino UNO's GND (Ground) pin.
    - Connect one leg of the photo resistor to the Arduino UNO's A5 (Analog Input) pin.
    - Connect the other leg of the photo resistor to the Arduino UNO's GND (Ground) pin.
    - Connect the base (B) pin of the NPN transistor to the junction of the photo resistor and the Arduino UNO's A5 pin.
    - Connect the emitter (E) pin of the NPN transistor to the Arduino UNO's GND (Ground) pin.
    - Connect the collector (C) pin of the NPN transistor to one terminal of the light bulb.
    - Connect the other terminal of the light bulb to the power supply's positive terminal (+).
    - Connect a 1 kΩ resistor between the Arduino UNO's digital pin 13 (D13) and the base (B) pin of the NPN transistor.
    - Connect a 10 kΩ resistor between the collector (C) pin of the NPN transistor and the Arduino UNO's 5V pin.
- Make sure all the connections are properly made and components are placed correctly.
- Save and simulate: Save your circuit design and click the "Start Simulation" button to test your circuit. You can adjust the light intensity on the photo resistor by changing the lighting conditions in the simulation.

b. Explain how the circuit is functioning to turn on street light.

A photoresistor, also known as a light-dependent resistor (LDR), is a passive electronic component that exhibits a change in resistance based on the intensity of light falling on its surface. The theory behind the photoresistor involves its light-sensitive properties and the variation of resistance with light exposure. Photoresistors are typically made of semiconductor materials, such as cadmium sulfide (CdS) or lead sulfide (PbS), that possess light-sensitive characteristics. When photons from incident light strike the surface of the photoresistor, they excite the electrons in the semiconductor material, causing a change in its electrical conductivity.

The resistance of a photoresistor decreases as the intensity of incident light increases. This behavior arises due to the phenomenon of photoconductivity, where the absorption of light increases the number of free charge carriers (electrons and holes) available for conduction in the semiconductor material. As a result, more current can flow through the photoresistor, leading to a lower resistance.

Conversely, in the absence of light or low light levels, fewer photons are available to excite the electrons, reducing the number of free charge carriers and resulting in higher resistance.

The relationship between the resistance of a photoresistor and the incident light intensity is typically nonlinear, and it varies based on the specific characteristics and materials used in the photoresistor construction. Calibration may be necessary to establish a reliable relationship between resistance and light intensity for specific applications.

Photoresistors find wide applications in various fields, including light-sensitive circuits, automatic lighting systems, photographic light meters, and light detection and ranging (LiDAR) systems. By understanding the theory behind photoresistors, engineers and designers can effectively utilize these components for light sensing and control purposes.

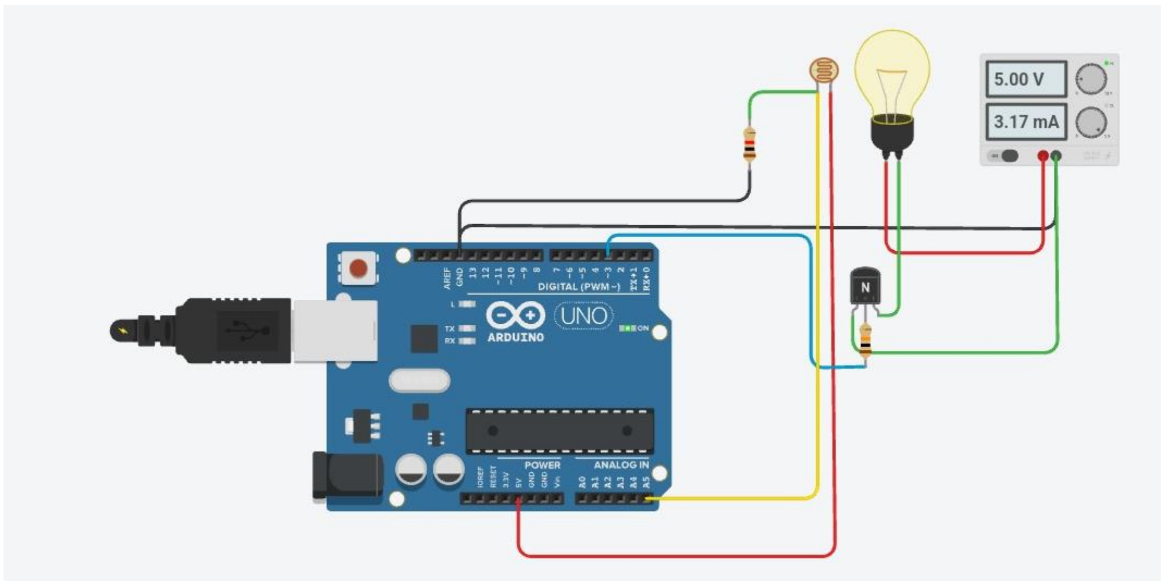
c. Write and describe the code utilized to run the circuit.

```
int ldr_pin=A5;
int ldr_value;
int light=3;

void setup()
{
  pinMode(light,OUTPUT);
  pinMode(ldr_pin,INPUT);
}
void loop()
{
  ldr_value=analogRead(ldr_pin);
  if(ldr_value>512)
    digitalWrite(light,0);
  else
```

```
   digitalWrite(light,1);
}
```

This code reads the analog value from an LDR sensor connected to pin A5 and controls a light connected to pin 3 based on the light intensity. If the light intensity is above a threshold of 512, the light is turned off. If it is below or equal to the threshold, the light is turned on.

3. **Output of the experiment:**



a. **Write the detailed steps involved to run the simulation in TinkerCAD.**

4. Click the "Code" button and type the code.
5. Then click the "Start Simulation" button.
6. Then, adjust the level in the photo Resistor to observe the turning on or turning off of light.

b. How are you assessing the output displayed in the Serial Monitor?

When the level in the photo Resistor is varied, we can see that the light is turning on or off. If the light is there (value in photo Resistor is greater than 512), the bulb won't and if the light is not there (value in photo Resistor is less than 512), and the bulb will glow.

Vellore Institute of Technology
Chennai Campus

Programme: B. Tech Electronics and Communication Engineering
BECE351E: Internet of Things (IoT) Lab

Smart Parking System using TinkerCAD

Name of the student:	Rahul Karthik S
Register Number:	21BEC1851
Date of the Lab. Class:	15.05.2023

1. Introduction:

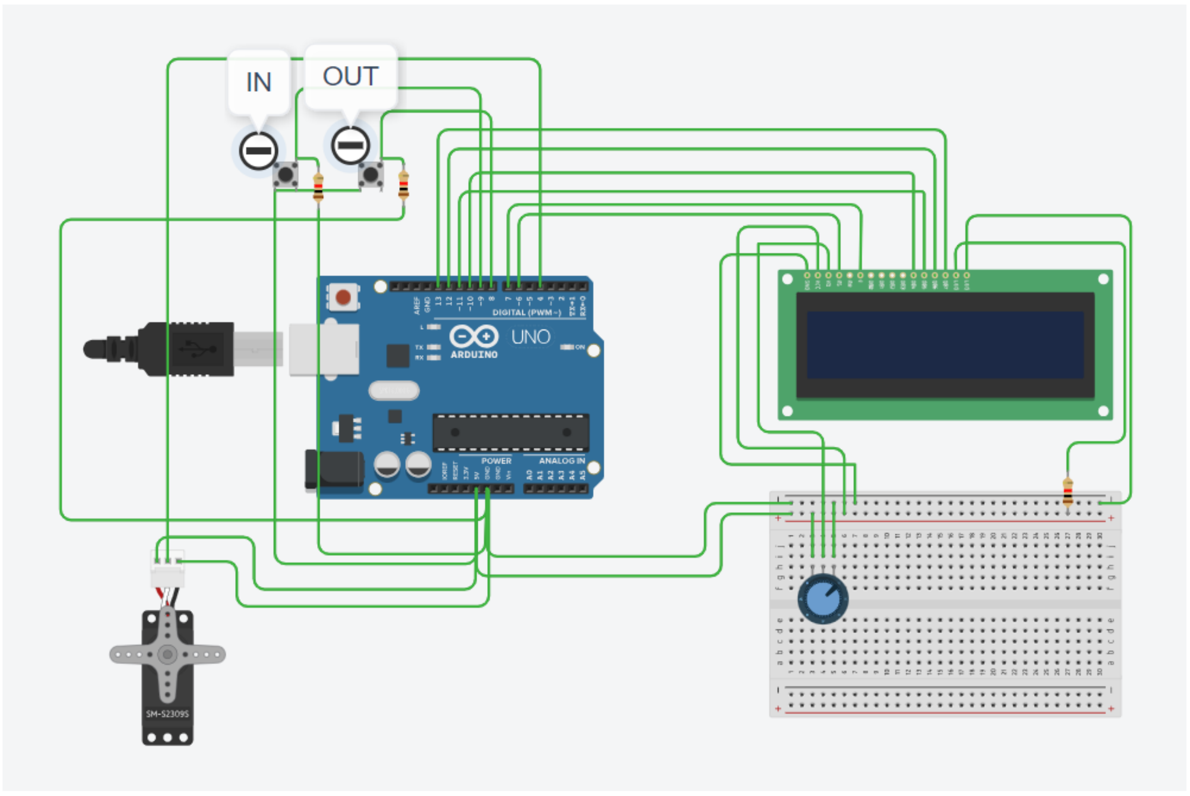
a. What are you targeting to do through this experiment?

To design a circuit to design a smart parking system using TinkerCAD.

b. List out the required components to execute this experiment?

Components	Quantity
Arduino UNO R3	1
250 kΩ Potentiometer	1
1 kΩ Resistor	3
LCD 16x2	1
Pushbutton	2
Micro Servo	1

2. Circuit Diagram & Explanation:



a. Write the detailed steps involved to design the circuit diagram in TinkerCAD.

- Take the components from the components pane.
- Connect the Microservo, Resistor, Arduino and LCD from the breadboard.
- Connect the DB Pins from the LCD to Arduino.
- Connect the push buttons to the Arduino.
- Finally, connect the servo motor to Arduino UNO.

b. Explain how the circuit is functioning to detect car parking.

When the IN Button is pressed, it denotes that the car is going inside and when the OUT Button is pressed, it denotes that the car is going outside.

c. Write and describe the code utilized to run the circuit.

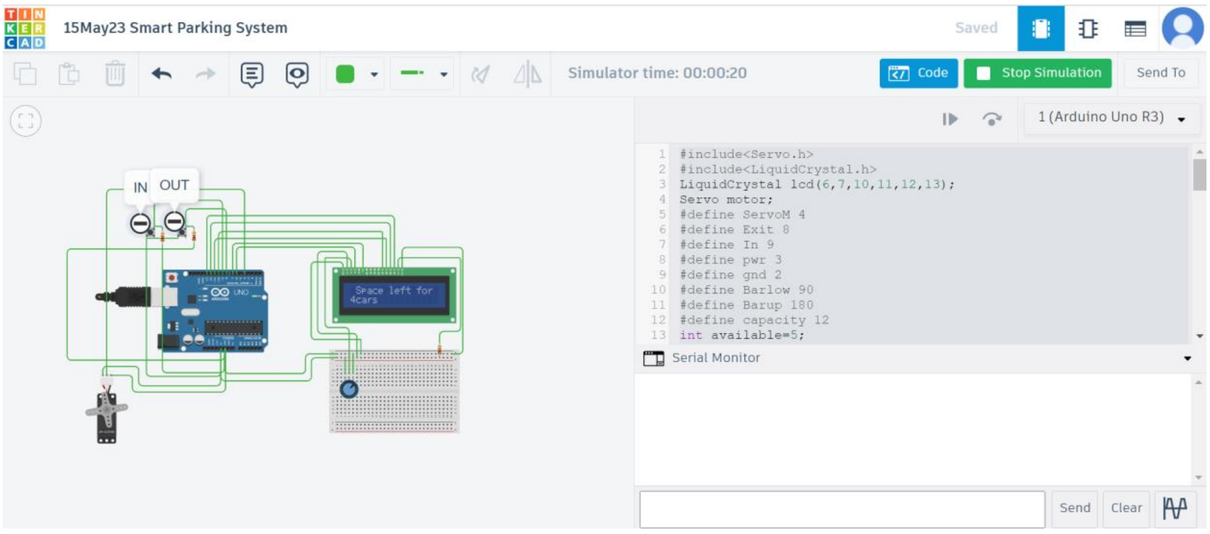
```
#include<Servo.h>
#include<LiquidCrystal.h>
LiquidCrystal lcd(6,7,10,11,12,13);
Servo motor;
#define ServoM 4
#define Exit 8
#define In 9
#define pwr 3
#define gnd 2
#define Barlow 90
#define Barup 180
#define capacity 12
int available=5;
void setup()
{
  motor.attach(ServoM);
  lcd.begin(16,2);
  lcd.print("Space left for");
  pinMode(pwr,OUTPUT);
  pinMode(gnd,OUTPUT);
  pinMode(In,INPUT);
  digitalWrite(pwr,HIGH);
  digitalWrite(gnd,LOW);
  motor.write(Barlow);
}
void loop()
{
  lcd.clear();
  lcd.setCursor(1,0);
  lcd.print("Space left for");
  lcd.setCursor(0,1);
  lcd.print(available);
  lcd.print("cars");
  if(available>=1)
  {
    lcd.clear();
    lcd.setCursor(1,0);
    lcd.print("Space left for");
    lcd.setCursor(0,1);
    lcd.print(available);
    lcd.print("cars");
  }
  else
  {
    lcd.clear();
    lcd.setCursor(1,0);
    lcd.print("Sorry!");
    lcd.setCursor(0,1);
    lcd.print("No vacant places left");
  }
  if(digitalRead(In)==1)
  {
    if(available!=0)
    {
      available--;
     motor.write(Barup);
     delay(3000);
     motor.write(Barlow);
    }
  }
  if(digitalRead(Exit)==1)
  {
    if(available!=capacity)
    {
      available++;
      motor.write(Barup);
      delay(5000);
      motor.write(Barlow);
    }
  }
}
```

The code initializes the necessary pins and libraries. It sets the initial number of available parking spaces and positions the servo motor to lower the barrier.

In the main loop, it continuously updates the LCD display to show the available parking spaces. If the entrance button is pressed and there are available spaces, it decreases the available count, raises the barrier to allow entry, and lowers it after a delay. Similarly, if the exit button is pressed and there are occupied spaces, it increases the available count, raises the barrier to allow exit, and lowers it after a delay.

In summary, the code monitors the parking spaces, displays the availability on the LCD, and controls the barrier using the servo motor for efficient management of the parking system.

3. Output of the experiment:



```
#include<Servo.h>
#include<LiquidCrystal.h>
LiquidCrystal lcd(6,7,10,11,12,13);
Servo motor;
#define ServoM 4
#define Exit 8
#define In 9
#define pwr 3
#define gnd 2
#define Barlow 90
#define Barup 180
#define capacity 12
int available=5;
```

a. Write the detailed steps involved to run the simulation in TinkerCAD.

- ➢ Click the "Code" button and type the code.
- ➢ Then click the "Stop Simulation" button.
- ➢ Then, click the IN and OUT button in the simulation.

b. How are you assessing the output displayed in the Serial Monitor?

- ➢ It is defined that the maximum car parking space is 12.
- ➢ If the IN button is clicked it means the car is entered the parking.
- ➢ If the OUT button is clicked it means the car is moving out of the parking.
- ➢ The available car parking space is displayed in the LCD display.

Vellore Institute of Technology
Chennai Campus

Programme: B. Tech Electronics and Communication Engineering
BECE351E: Internet of Things (IoT) Lab

Earthquake Monitoring System using Node-RED

Name of the student:	Rahul Karthik S
Register Number:	21BEC1851
Date of the Lab. Class:	25.05.2023

1. Introduction:

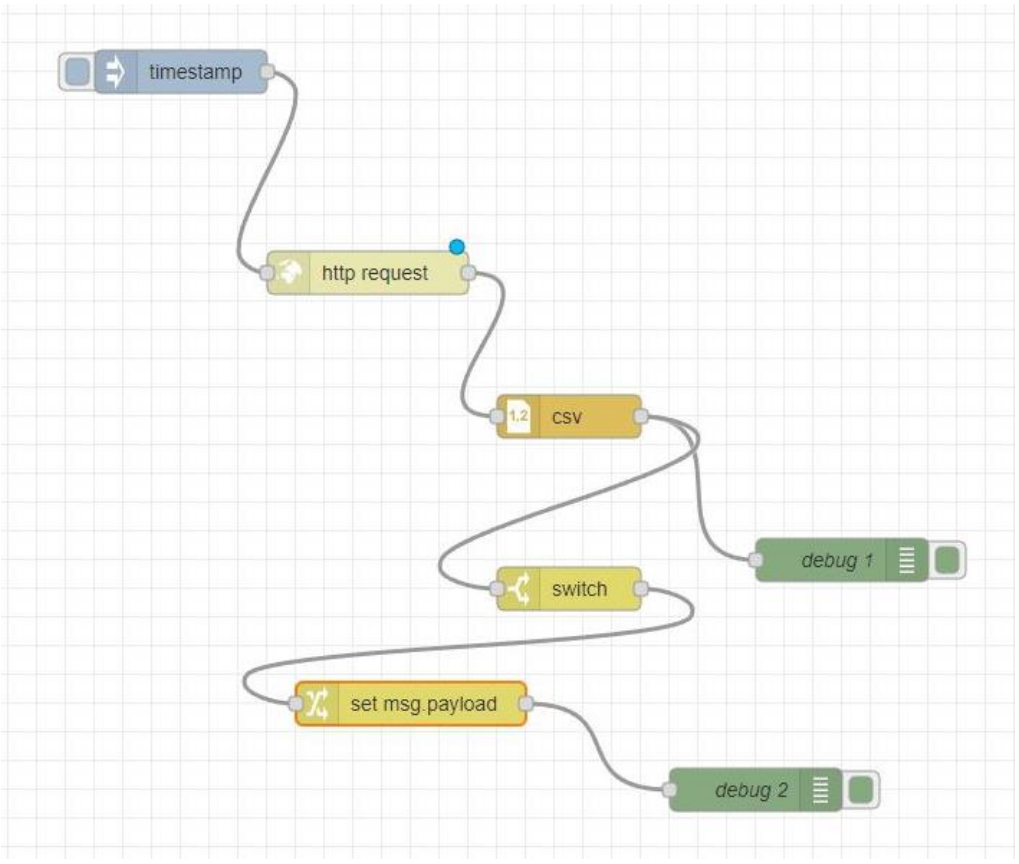
a. What are you targeting to do through this experiment?

To perform Earthquake Data analysis using NODE Red.

b. List out the required components to execute this experiment?

Inject Nodes, Debug Node, HTTP Request node, Switch node, Change node, CSV node and connecting wires.

2. Circuit Diagram & Explanation:



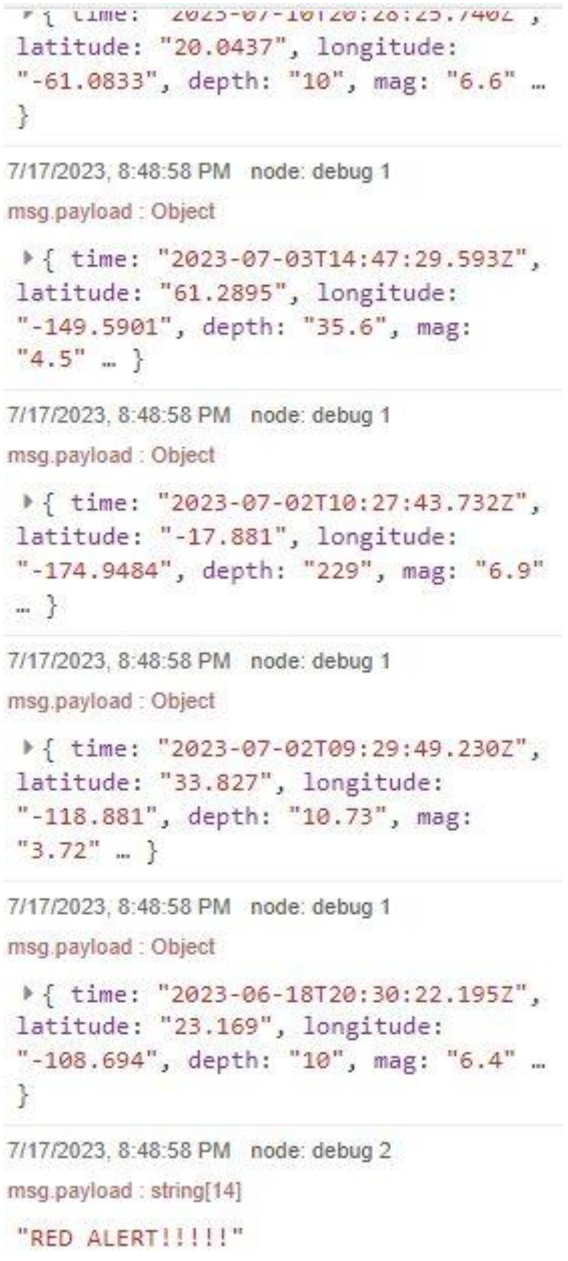
a. Write the detailed steps involved to design the circuit diagram in TinkerCAD.

- Based on the specific requirements, add an "Inject" node to the flow and provide a suitable name for it, along with the associated temperature or humidity value.
- Include 2 "Debug" nodes, 1 http node, a switch node, a csv node and a change node in the flow and connect it according to the circuit diagram

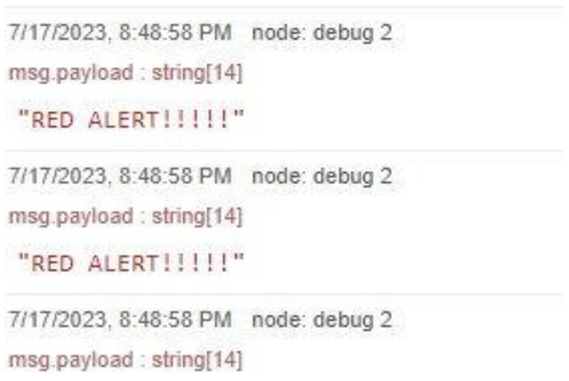
b. Explain how the circuit is functioning to perform Earthquake Analysis.

Once the circuit is deployed, the data from the CSV file is injected to the debug monitor and if the magnitude is equal to or greater than 6.5, it displays "RED ALERT".

3. Output of the experiment:



```
 ▾{ time: "2023-07-10T20:28:25.740Z",
 latitude: "20.0437", longitude:
 "-61.0833", depth: "10", mag: "6.6" …
 }

7/17/2023, 8:48:58 PM   node: debug 1
msg.payload : Object
 ▸{ time: "2023-07-03T14:47:29.593Z",
 latitude: "61.2895", longitude:
 "-149.5901", depth: "35.6", mag:
 "4.5" … }

7/17/2023, 8:48:58 PM   node: debug 1
msg.payload : Object
 ▸{ time: "2023-07-02T10:27:43.732Z",
 latitude: "-17.881", longitude:
 "-174.9484", depth: "229", mag: "6.9"
 … }

7/17/2023, 8:48:58 PM   node: debug 1
msg.payload : Object
 ▸{ time: "2023-07-02T09:29:49.230Z",
 latitude: "33.827", longitude:
 "-118.881", depth: "10.73", mag:
 "3.72" … }

7/17/2023, 8:48:58 PM   node: debug 1
msg.payload : Object
 ▸{ time: "2023-06-18T20:30:22.195Z",
 latitude: "23.169", longitude:
 "-108.694", depth: "10", mag: "6.4" …
 }

7/17/2023, 8:48:58 PM   node: debug 2
msg.payload : string[14]
 "RED ALERT!!!!!"
```


```
7/17/2023, 8:48:58 PM   node: debug 2
msg.payload : string[14]
 "RED ALERT!!!!!"
7/17/2023, 8:48:58 PM   node: debug 2
msg.payload : string[14]
 "RED ALERT!!!!!"
7/17/2023, 8:48:58 PM   node: debug 2
msg.payload : string[14]
```

a. Write the detailed steps involved to run the simulation in Node-RED. Deploy the flow to activate the simulation.

- Monitor and analyze the simulation by observing messages passing through nodes.
- Optionally, interact with the simulation during runtime using input nodes or external triggers.
- Stop the simulation by clicking the "RED" square button in the Node-RED interface.

b. How are you assessing the output displayed in the Serial Monitor?

By injecting the values of the magnitude recorded in the csv file, the corresponding values will be displayed in the debug window. Further, data with magnitude greater than or equal to magnitude 6.5 will be mentioned as red alert magnitudes.

Vellore Institute of Technology
Chennai Campus

Programme: B. Tech Electronics and Communication Engineering
BECE351E: Internet of Things (IoT) Lab

Temperature and Humidity Monitoring

Name of the student:	Rahul Karthik S
Register Number:	21BEC1851
Date of the Lab. Class:	15.06.2023

1. Introduction:

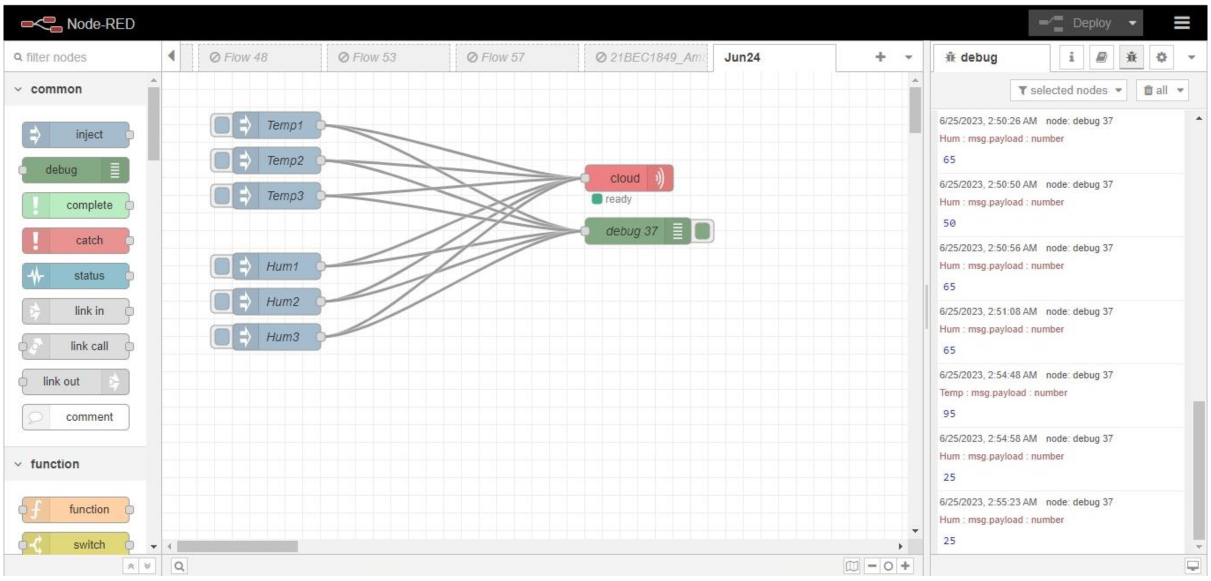
a. What are you targeting to do through this experiment?

To measure the temperature and humidity using Node Red and visualize using thingspeak.

b. List out the required components to execute this experiment?

Inject Nodes, Debug Node, Thingspeak 42 Node and Connecting wires.

2. Circuit Diagram & Explanation:



```
6/25/2023, 2:50:26 AM   node: debug 37
Hum : msg.payload : number
65
6/25/2023, 2:50:50 AM   node: debug 37
Hum : msg.payload : number
50
6/25/2023, 2:50:56 AM   node: debug 37
Hum : msg.payload : number
65
6/25/2023, 2:51:08 AM   node: debug 37
Hum : msg.payload : number
65
6/25/2023, 2:54:48 AM   node: debug 37
Temp : msg.payload : number
95
6/25/2023, 2:54:58 AM   node: debug 37
Hum : msg.payload : number
25
6/25/2023, 2:55:23 AM   node: debug 37
Hum : msg.payload : number
25
```

a. Write the detailed steps involved to design the circuit diagram in Node – RED.

- ➢ Based on the specific requirements, add an "Inject" node to the flow and provide a suitable name for it, along with the associated temperature or humidity value.
- ➢ Include a "Debug" node in the flow and connect it to all the "Inject" nodes.
- ➢ Integrate a "ThingSpeak 42" node into the flow and connect it to all the "Inject" nodes. Provide a name for the node and ensure that the topic

matches the name assigned to the respective "Inject" node. Additionally, set the desired delay value for the transmission.

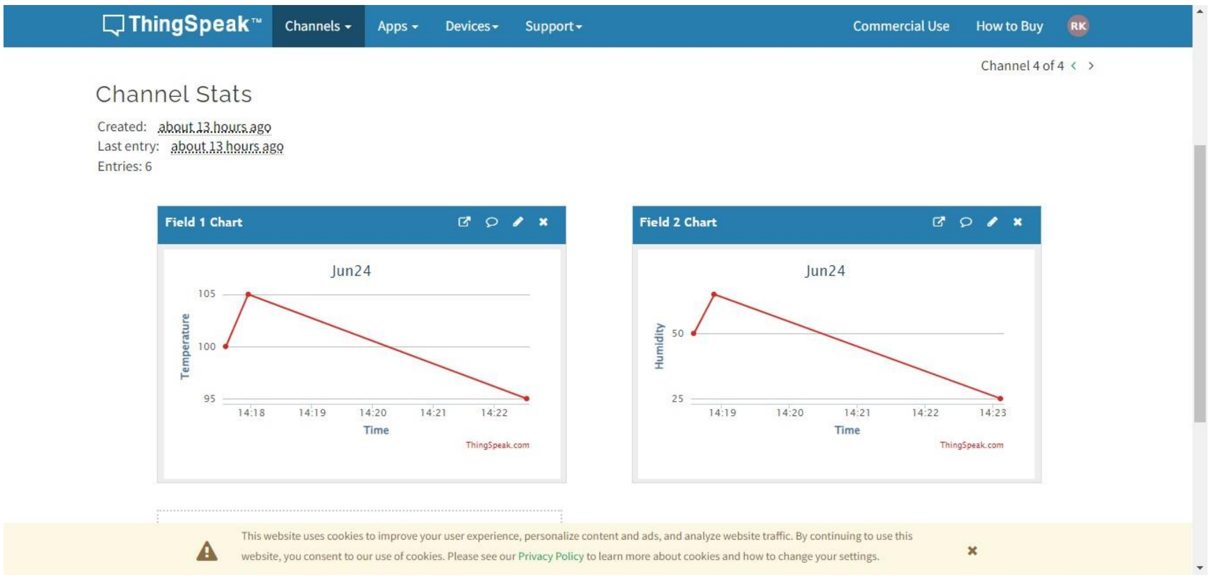
b. Explain how the circuit is functioning to measure temperature and humidity.

Once the circuit is deployed, injecting the temperature/humidity node will result in the corresponding values being displayed in the Serial Monitor. Simultaneously, these values will also be plotted on a graph in the ThingSpeak platform.

c. Write and describe the steps for visualize the temperature and humidity in thingspeak.

First, create a channel on the ThingSpeak platform and copy the API key associated with the channel. Paste this API key into the designated API key column of the ThingSpeak 42 node. At this point, two empty graphs will be shown. To visualize data, inject values accordingly and observe the graphs as they dynamically update based on the injected values.

3. Output of the experiment:



Channel Stats

Created: about 13 hours ago
Last entry: about 13 hours ago
Entries: 6

This website uses cookies to improve your user experience, personalize content and ads, and analyze website traffic. By continuing to use this website, you consent to our use of cookies. Please see our Privacy Policy to learn more about cookies and how to change your settings.

a. Write the detailed steps involved to run the simulation in Thingspeak.

Once the circuit implementation is complete, disable all other flows except the one for the temperature and humidity monitoring system. Deploy the circuit to make it operational. Open the debug window in the Arduino IDE or other relevant software to view the real-time output of the system. Additionally, access the ThingSpeak website to monitor and analyse the temperature and humidity data plotted on the graphs.

b. How are you assessing the output displayed?

By injecting the temperature or humidity node, the corresponding values will be displayed in the debug window. This enables us to verify the correctness of the obtained output by comparing it with the expected values.

Vellore Institute of Technology
Chennai Campus

Programme: B. Tech Electronics and Communication Engineering
BECE351E: Internet of Things (IoT) Lab

Temperature and Humidity Monitoring using DHT11 Sensor

Name of the student:	Rahul Karthik S
Register Number:	21BEC1851
Date of the Lab. Class:	06.07.2023

1. Introduction:

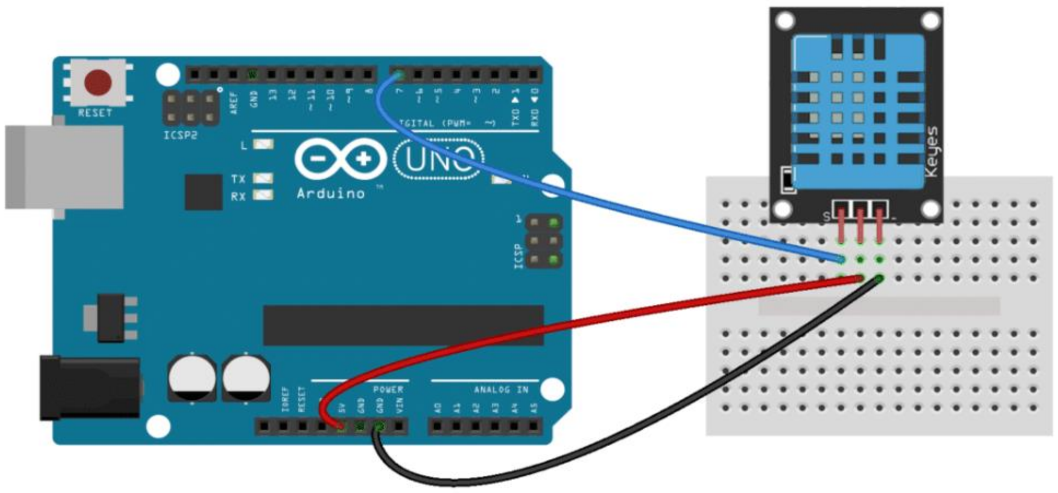
a. What are you targeting to do through this experiment?

To design a circuit to monitor the temperature and humidity continuously using DHT11 sensor.

b. List out the required components to execute this experiment?

Components	**Quantity**
Arduino UNO R3	1
DHT11 Sensor	1
Breadboard	1
Connecting Wires	3

2. Circuit Diagram & Explanation:



a. Write the detailed steps involved to design the circuit diagram in TinkerCAD.

1) Gather the necessary components:
   a. Arduino board (e.g., Arduino Uno)

    b. DHT11 sensor
    c. Breadboard
    d. Jumper wires
    e. USB cable for Arduino board
2) Connect the DHT11 sensor to the Arduino:
    a. Locate the pinout diagram of the DHT11 sensor. It typically consists of four pins: VCC, GND, DATA, and NC (not connected).
    b. Connect the VCC pin of the DHT11 sensor to the 5V pin on the Arduino.
    c. Connect the GND pin of the DHT11 sensor to the GND pin on the Arduino.
    d. Connect the DATA pin of the DHT11 sensor to any available digital pin on the Arduino (e.g., pin 7).
3) Add a pull-up resistor to the DATA pin:
    a. Connect the leg of the resistor to the 5V pin on the Arduino.
4) Connect the Arduino to a computer:
    a. Use a USB cable to connect the Arduino board to your computer.
5) Set up the Arduino IDE and libraries:
    a. Install the Arduino IDE software on your computer if you haven't already.
    b. Open the Arduino IDE and create a new sketch.
    c. Include the DHT library by navigating to Sketch > Include Library > DHT Sensor Library.
6) Write the Arduino code in the IDE.
7) Verify that the code compiles without any errors.
    a. Select the correct Arduino board and port from the Tools menu.
    b. Click on the "Upload" button to upload the code to the Arduino board.
8) Monitor the output:
    a. Open the serial monitor in the Arduino IDE by clicking on the magnifying glass icon or navigating to Tools > Serial Monitor.
    b. Set the baud rate in the serial monitor to match the one specified in the Arduino code (e.g., 9600).
    c. Observe the temperature and humidity readings displayed in the serial monitor.

b. Explain how the circuit is functioning to measure temperature and humidity using DHT11.

The DHT11 sensor is a low-cost digital sensor widely used for measuring temperature and humidity in various applications. It utilises a capacitive humidity sensor and a thermistor to accurately measure these environmental parameters when connected to an Arduino board.

The DHT11 sensor employs a capacitive humidity sensor that consists of a thin polymer film. This film absorbs moisture from the surrounding air, causing a change in the electrical capacitance of the sensor. By measuring this capacitance change, the DHT11 sensor converts it into a digital humidity value, providing an accurate representation of the relative humidity.

In addition to the humidity sensor, the DHT11 also integrates a thermistor to measure temperature. A thermistor is a type of resistor whose resistance varies with temperature. By observing the resistance of the thermistor, the DHT11 sensor can determine the ambient temperature.

Communication between the DHT11 sensor and the Arduino board occurs through a simple single-wire protocol. The Arduino initiates communication by sending a start signal, to which the sensor responds with a handshake signal. Once the handshake is established, the DHT11 sensor transmits a 40-bit data stream to the Arduino, consisting of two bytes for humidity, two bytes for temperature, and one byte for checksum verification.

To integrate the DHT11 sensor with an Arduino board, specific connections must be made. The VCC pin of the DHT11 sensor connects to the 5V pin of the Arduino, while the GND pin connects to the Arduino's ground pin. The OUT pin of the DHT11 sensor is connected to a digital pin of the Arduino to enable data transmission.

To read and process data from the DHT11 sensor, the Arduino code utilizes a library such as the DHT sensor library. This library handles the communication protocol, extracts the temperature and humidity values from the received data stream, and typically returns them as floating-point numbers. These values can then be further processed or displayed according to the project's requirements.

By understanding the principles mentioned above and employing the appropriate code and library, it is possible to successfully interface the DHT11 sensor with an Arduino board, enabling accurate measurement of temperature and humidity.

c. Write and describe the code utilized to run the circuit.

```
#include<dht.h>
dht DHT;
#define DHT11_PIN 7
void setup(){
Serial.begin(9600);
}
void loop(){
        int chk = DHT.read11(DHT11_PIN);
        Serial.print("Temperature =");
        Serial.println(DHT.temperature);
        Serial.print("Humidity");
        Serial.println(DHT.humidity);
        delay(1000);
}
```

This code utilizes the DHT library to interface with a DHT11 sensor, which is capable of measuring temperature and humidity. The setup() function initializes the serial communication at a baud rate of 9600, allowing communication between the Arduino board and a computer through the serial monitor.

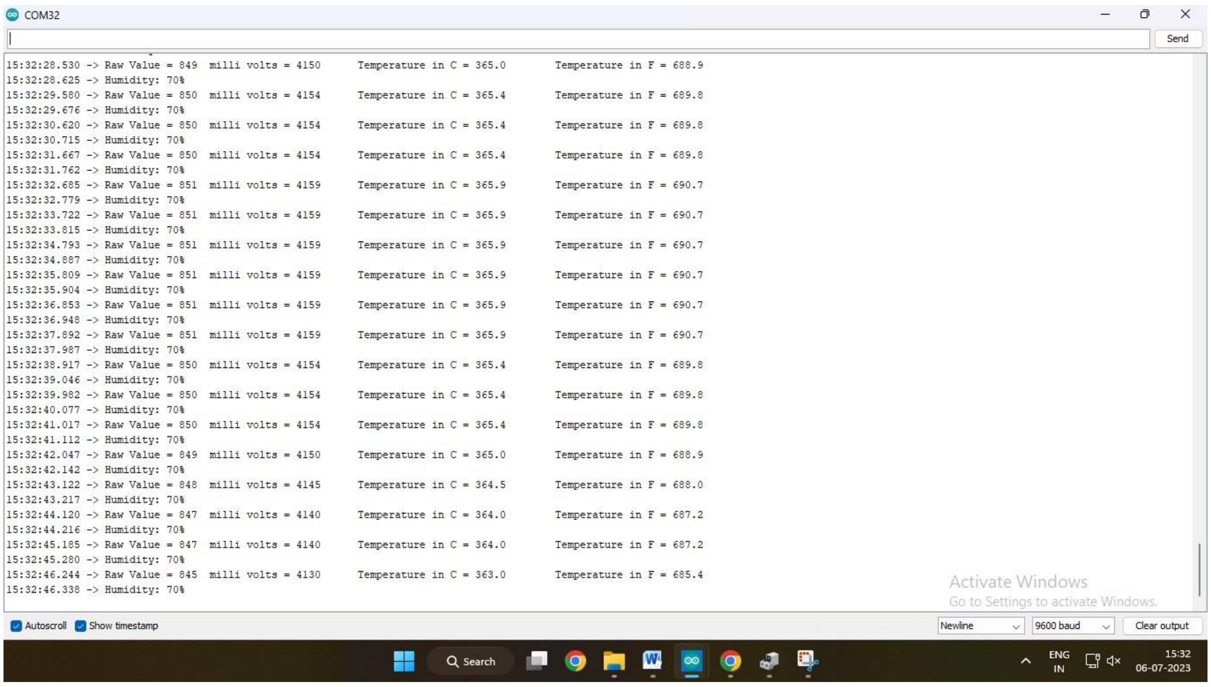
The main execution occurs in the loop() function, which is a continuous loop that repeats indefinitely. Within the loop, the DHT.read11() function is called to read data from the DHT11 sensor connected to pin 7 (as specified by DHT11_PIN).

The obtained temperature and humidity values are then printed to the serial monitor using the Serial.print() and Serial.println() functions. The Serial.print() function displays the label or text, followed by the corresponding value retrieved from DHT.temperature and DHT.humidity. The Serial.println() function is used to print a new line after each output.

To introduce a delay of one second between readings, the delay() function is employed, causing the program to pause for 1000 milliseconds (1 second) before starting the next iteration of the loop.

Essentially, this code continuously reads temperature and humidity data from the DHT11 sensor and displays the values on the serial monitor with a one-second interval.

3. Output of the experiment:



```
15:32:28.530 -> Raw Value = 849  milli volts = 4150     Temperature in C = 365.0       Temperature in F = 688.9
15:32:28.625 -> Humidity: 70%
15:32:29.580 -> Raw Value = 850  milli volts = 4154     Temperature in C = 365.4       Temperature in F = 689.8
15:32:29.676 -> Humidity: 70%
15:32:30.620 -> Raw Value = 850  milli volts = 4154     Temperature in C = 365.4       Temperature in F = 689.8
15:32:30.715 -> Humidity: 70%
15:32:31.667 -> Raw Value = 850  milli volts = 4154     Temperature in C = 365.4       Temperature in F = 689.8
15:32:31.762 -> Humidity: 70%
15:32:32.685 -> Raw Value = 851  milli volts = 4159     Temperature in C = 365.9       Temperature in F = 690.7
15:32:32.779 -> Humidity: 70%
15:32:33.722 -> Raw Value = 851  milli volts = 4159     Temperature in C = 365.9       Temperature in F = 690.7
15:32:33.815 -> Humidity: 70%
15:32:34.793 -> Raw Value = 851  milli volts = 4159     Temperature in C = 365.9       Temperature in F = 690.7
15:32:34.887 -> Humidity: 70%
15:32:35.809 -> Raw Value = 851  milli volts = 4159     Temperature in C = 365.9       Temperature in F = 690.7
15:32:35.904 -> Humidity: 70%
15:32:36.853 -> Raw Value = 851  milli volts = 4159     Temperature in C = 365.9       Temperature in F = 690.7
15:32:36.948 -> Humidity: 70%
15:32:37.892 -> Raw Value = 851  milli volts = 4159     Temperature in C = 365.9       Temperature in F = 690.7
15:32:37.987 -> Humidity: 70%
15:32:38.917 -> Raw Value = 850  milli volts = 4154     Temperature in C = 365.4       Temperature in F = 689.8
15:32:39.046 -> Humidity: 70%
15:32:39.982 -> Raw Value = 850  milli volts = 4154     Temperature in C = 365.4       Temperature in F = 689.8
15:32:40.077 -> Humidity: 70%
15:32:41.017 -> Raw Value = 850  milli volts = 4154     Temperature in C = 365.4       Temperature in F = 689.8
15:32:41.112 -> Humidity: 70%
15:32:42.047 -> Raw Value = 849  milli volts = 4150     Temperature in C = 365.0       Temperature in F = 688.9
15:32:42.142 -> Humidity: 70%
15:32:43.122 -> Raw Value = 848  milli volts = 4145     Temperature in C = 364.5       Temperature in F = 688.0
15:32:43.217 -> Humidity: 70%
15:32:44.120 -> Raw Value = 847  milli volts = 4140     Temperature in C = 364.0       Temperature in F = 687.2
15:32:44.216 -> Humidity: 70%
15:32:45.185 -> Raw Value = 847  milli volts = 4140     Temperature in C = 364.0       Temperature in F = 687.2
15:32:45.280 -> Humidity: 70%
15:32:46.244 -> Raw Value = 845  milli volts = 4130     Temperature in C = 363.0       Temperature in F = 685.4
15:32:46.338 -> Humidity: 70%
```

a. Write the detailed steps involved to run the simulation in TinkerCAD.

- Write the Arduino code in the IDE.
- Verify that the code compiles without any errors.
  - Select the correct Arduino board and port from the Tools menu.
  - Click on the "Upload" button to upload the code to the Arduino board.
- Monitor the output:
  - Open the serial monitor in the Arduino IDE by clicking on the magnifying glass icon or navigating to Tools > Serial Monitor.
  - Set the baud rate in the serial monitor to match the one specified in the Arduino code (e.g., 9600).
  - Observe the temperature and humidity readings displayed in the serial monitor.

b. How are you assessing the output displayed in the Serial Monitor?

In this code, the output displayed in the Serial Monitor for the DHT11 sensor is assessed using the ‘Serial.print()’ and ‘Serial.println()’ functions.

The ‘Serial.print()’ function is used to display the label or text, followed by the corresponding value retrieved from the DHT11 sensor. For example, ‘Serial.print("Temperature =")’ is used to display the text "Temperature =" on the Serial Monitor.

After that, the ‘Serial.println()’ function is used to print the actual temperature or humidity value obtained from the DHT11 sensor. For example, ‘Serial.println(DHT.temperature)’ is used to print the temperature value obtained from the DHT11 sensor on a new line in the Serial Monitor.

By using these print functions, the code displays the temperature and humidity readings obtained from the DHT11 sensor on the Serial Monitor, allowing you to visually observe the values as they change over time.

Vellore Institute of Technology
Chennai Campus

Programme: B. Tech Electronics and Communication Engineering
BECE351E: Internet of Things (IoT) Lab

Ultrasonic Distance Measurement using Arduino

Name of the student:	Rahul Karthik S
Register Number:	21BEC1851
Date of the Lab. Class:	06.07.2023

1. Introduction:

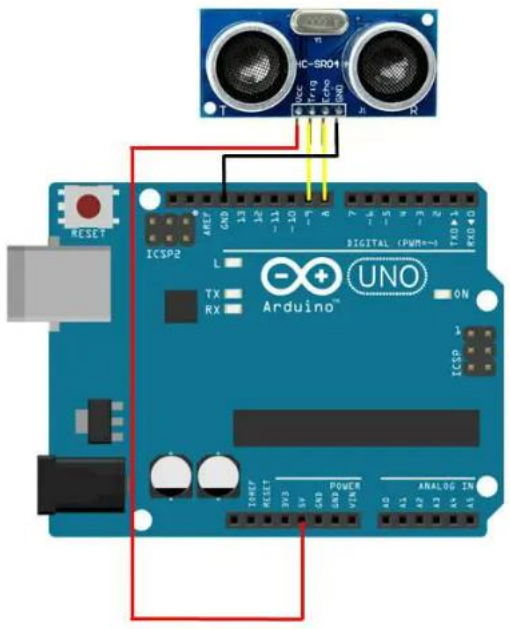
a. What are you targeting to do through this experiment?

To measure the distance between the object and the ultraviolet sensor.

b. List out the required components to execute this experiment?

Components	Quantity
Arduino UNO R3	1
Ultrasonic Sensor	1
Connecting Wires	4

2. Circuit Diagram & Explanation:



a. Write the detailed steps involved to design the circuit diagram in TinkerCAD.

To design a circuit for measuring distance using an ultrasonic sensor and Arduino,

1. The following components are required:
   a. Arduino board (e.g., Arduino Uno)

   b. Ultrasonic sensor module (e.g., HC-SR04)
   c. Breadboard
   d. Jumper wires
   e. Resistors (optional, depending on the sensor module used)
2. The components mentioned above are gathered.
3. The ultrasonic sensor is connected to the Arduino board using jumper wires:
- The Vcc pin of the sensor is connected to the 5V pin on the Arduino.
- The GND pin of the sensor is connected to the GND pin on the Arduino. The Trig pin of the sensor is connected to any digital pin (e.g., Pin 2) on the Arduino.
- The Echo pin of the sensor is connected to any digital pin (e.g., Pin 3) on the Arduino.
- If the ultrasonic sensor module has a separate Vcc and GND pin for the Echo signal, they are connected accordingly. Otherwise, the Echo pin shares the Vcc and GND connections with the sensor.
4. If the ultrasonic sensor module requires pull-up resistors for the Trig and Echo pins, a 10K resistor is connected between the Trig pin and the 5V pin on the Arduino, and another 10K resistor is connected between the Echo pin and the 5V pin on the Arduino.
5. The Arduino is connected to the computer using a USB cable.
6. The Arduino IDE is launched on the computer, and the code for measuring the distance using the ultrasonic sensor is written. The code is uploaded to the Arduino board.
- The Serial Monitor in the Arduino IDE is opened to view the distance measurements, with the baud rate set to 9600.
- The circuit is powered on, and an object is placed in front of the ultrasonic sensor. The measured distance is displayed on the Serial Monitor.

b. Explain how the circuit is functioning to detect distance.

The ultrasonic sensor is a popular device used with Arduino to measure distance. It works based on the principle of sound waves. Here's a descriptive theory explaining how an ultrasonic sensor measures distance using an Arduino:

<u>Working Principle</u>: The ultrasonic sensor emits high-frequency sound waves (typically in the ultrasonic range, above the human hearing range) and measures the time it takes for the sound waves to bounce back after hitting an object. By knowing the speed of sound in the medium (usually air), the sensor can calculate the distance to the object.

<u>Measurement Process</u>:
a. <u>Trigger Pulse</u>: The Arduino sends a short electrical pulse (usually around 10 microseconds) to the ultrasonic sensor's trigger pin.
b. <u>Sound Wave Emission</u>: Upon receiving the trigger pulse, the sensor emits a burst of ultrasonic waves.
c. <u>Wave Reflection</u>: The waves propagate through the air and hit an object in their path. A portion of the sound waves reflects back towards the sensor.

d. Echo Detection: The receiver in the ultrasonic sensor picks up the reflected sound waves.
e. Signal Conversion: The receiver converts the received sound waves into an electrical signal and sends it to the Arduino.
f. Time Calculation: The Arduino measures the time it took for the ultrasonic waves to travel to the object and back. This is typically done by measuring the duration between the trigger pulse and the moment the echo signal is received.
g. Distance Calculation: Knowing the speed of sound in the medium, the Arduino uses the time measurement to calculate the distance to the object using the formula: Distance = (Speed of Sound * Time) / 2.

Calibration: Before accurate distance measurements can be obtained, it is necessary to calibrate the sensor. This involves compensating for any timing delays caused by factors such as signal propagation, signal processing, and sensor response time. Calibration ensures accurate distance measurements.

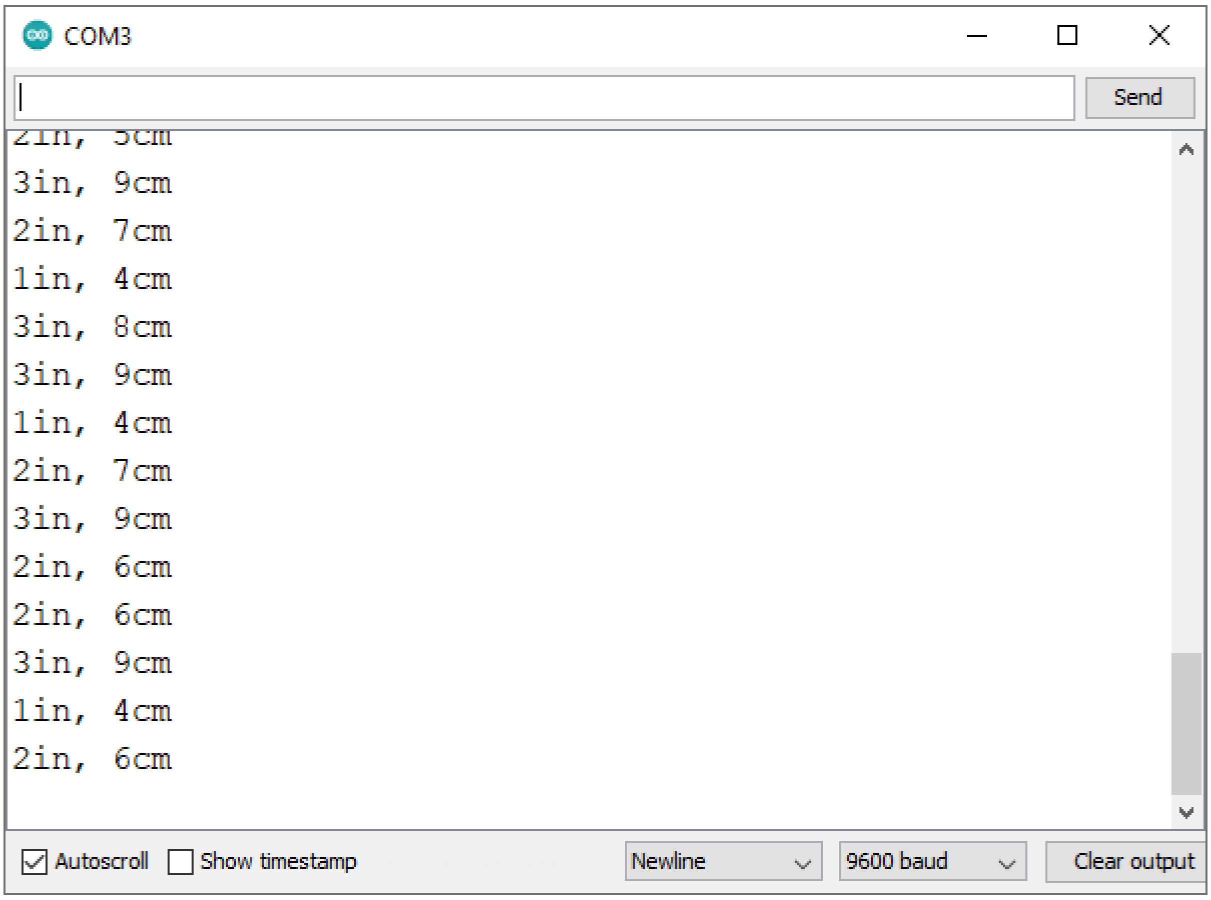
Application: Once the distance is calculated, the Arduino can use the information for various applications, such as obstacle avoidance in robotics, measuring fluid levels, or creating interactive distance-based projects.

c. Write and describe the code utilized to run the circuit.

```
const int pingPin = 7; // Trigger Pin of Ultrasonic Sensor

const int echoPin = 6; // Echo Pin of Ultrasonic Sensor

void setup() {

  Serial.begin(9600); // Starting Serial Terminal

}

void loop() {

  long duration, inches, cm;

  pinMode(pingPin, OUTPUT);

  digitalWrite(pingPin, LOW);

  delayMicroseconds(2);

  digitalWrite(pingPin, HIGH);

  delayMicroseconds(10);

  digitalWrite(pingPin, LOW);

  pinMode(echoPin, INPUT);

  duration = pulseIn(echoPin, HIGH);

  inches = microsecondsToInches(duration);
  cm = microsecondsToCentimeters(duration);
  Serial.print(inches);
  Serial.print("in, ");
  Serial.print(cm);
  Serial.print("cm");
  Serial.println();
  delay(100);
}
long microsecondsToInches(long microseconds) {
  return microseconds / 74 / 2;
}
long microsecondsToCentimeters(long microseconds) {
  return microseconds / 29 / 2;
}
```

This Arduino code uses an ultrasonic distance sensor to measure the distance between the sensor and an object. It calculates the distance in inches and centimeters and displays the results in the Serial Monitor. The code sends a trigger signal to the sensor, measures the time it takes for the echo signal to return, and converts that time into distances. The process repeats in a loop with a short delay between measurements.

3. Output of the experiment:



```
3in, 9cm
2in, 7cm
1in, 4cm
3in, 8cm
3in, 9cm
1in, 4cm
2in, 7cm
3in, 9cm
2in, 6cm
2in, 6cm
3in, 9cm
1in, 4cm
2in, 6cm
```

a. Write the detailed steps involved to run the simulation in TinkerCAD.

- Write the Arduino code in the IDE.
- Verify that the code compiles without any errors.
    - Select the correct Arduino board and port from the Tools menu.
    - Click on the "Upload" button to upload the code to the Arduino board.
- Monitor the output:
    - Open the serial monitor in the Arduino IDE by clicking on the magnifying glass icon or navigating to Tools > Serial Monitor.
    - Set the baud rate in the serial monitor to match the one specified in the Arduino code (e.g., 9600).
    - Observe the temperature and humidity readings displayed in the serial monitor.

b. How are you assessing the output displayed in the Serial Monitor?

When any object is placed at some distance, the ultrasonic sensor will give the distance as output. We can verify the distance with the measurement ruler.

Vellore Institute of Technology
Chennai Campus

Programme: B. Tech Electronics and Communication Engineering
BECE351E: Internet of Things (IoT) Lab

Detection of Soil Moisture with Arduino (Hardware)

Name of the student:	Rahul Karthik S
Register Number:	21BEC1851
Date of the Lab. Class:	22.06.2023

1. Introduction:

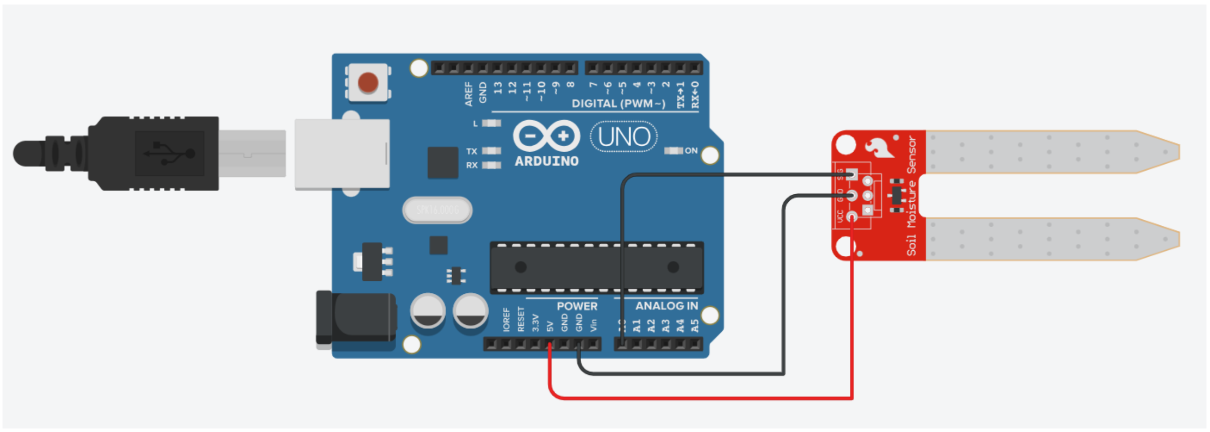
a. What are you targeting to do through this experiment?

To design a circuit to monitor the moisture of the soil using Arduino in Hardware.

b. List out the required components to execute this experiment?

Components	**Quantity**
Arduino UNO R3	1
Soil Moisture Sensor	1
Connecting Wires	3

2. Circuit Diagram & Explanation:



a. Write the detailed steps involved to design the circuit diagram in TinkerCAD.

- Gather the required components:
  - Arduino board (e.g., Arduino Uno)
  - Soil moisture sensor (analog or digital)
  - Jumper wires
  - Breadboard (optional, for prototyping)
- Connect the soil moisture sensor to the Arduino board, for analog soil moisture sensor:
  - Connect the sensor's VCC pin to Arduino's 5V pin.

    - Connect the sensor's GND (ground) pin to Arduino's GND pin.
    - Connect the sensor's OUT pin to any analog input pin on Arduino (e.g., A0).
- Connect the Arduino board to the computer using a USB cable.
- Write the Arduino code to read and display the moisture level:
- Open the Arduino Integrated Development Environment (IDE).
- Create a new sketch.
- Write the code to read the moisture level from the sensor and display it on the serial monitor.
- Upload the code to the Arduino board.
- Test the circuit:
    - Place the soil moisture sensor in the soil you want to measure.
    - Open the serial monitor in the Arduino IDE to observe the moisture level readings.
- Calibrate the sensor (if required):
    - Depending on the sensor and soil conditions, you may need to calibrate the readings to match the desired moisture levels. Refer to the sensor's datasheet or documentation for calibration instructions.

b. Explain how the circuit is functioning to detect moisture of the soil.

Soil moisture sensors measure water content in soil. Capacitive sensors use changes in capacitance to determine moisture levels. Tensiometers rely on soil water potential and water movement in a tube. TDR sensors measure soil's electrical conductivity by sending pulses. Neutron moisture meters use neutron interaction with hydrogen atoms. Calibration is needed for accurate readings. These sensors help optimize irrigation and assess soil health.

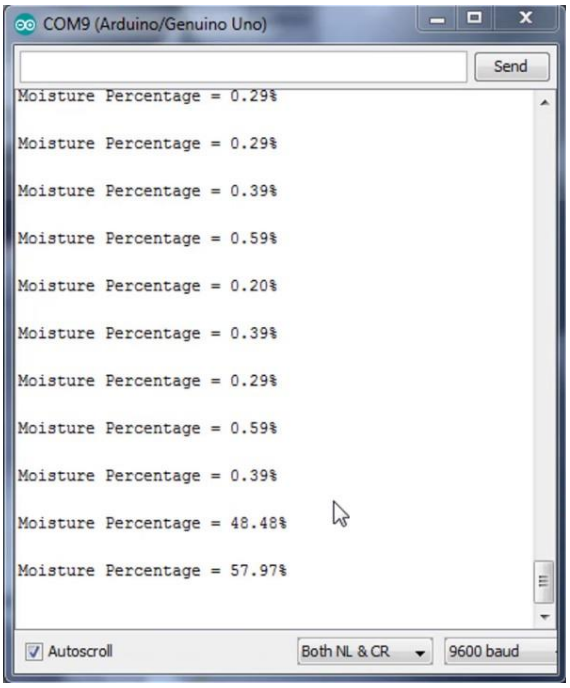
c. Write and describe the code utilized to run the circuit.

```
int moisture_data=0;

void setup()

{

 pinMode(A0, INPUT);

 Serial.begin(9600);

}

void loop()

{

 moisture_data=analogRead(A0);

 int x = 100 – (moisture_data/1023.00)*100;

 Serial.print("Moisture Percentage = ");

 Serial.println(x);
```

```
  delay(1000);
}
```

In the setup() function, the program initializes the pin A0 as an input and starts the serial communication at a baud rate of 9600. In the loop() function, the program reads the analog value from pin A0 using analogRead(A0). It then calculates the moisture level percentage by converting the analog reading to a range of 0-100 using the formula 100 - (moisture_data / 1023.00) * 100, where moisture_data is the raw analog reading. The calculated percentage value is printed to the serial monitor using Serial.println(x), where x represents the moisture level percentage. The program then introduces a 1-second delay using delay(1000) before repeating the loop, ensuring a regular interval between readings.

3. Output of the experiment:



```
Moisture Percentage = 0.29%

Moisture Percentage = 0.29%

Moisture Percentage = 0.39%

Moisture Percentage = 0.59%

Moisture Percentage = 0.20%

Moisture Percentage = 0.39%

Moisture Percentage = 0.29%

Moisture Percentage = 0.59%

Moisture Percentage = 0.39%

Moisture Percentage = 48.48%

Moisture Percentage = 57.97%
```

a. Write the detailed steps involved to run the simulation in Arduino IDE.

- ➢ Connect your soil moisture sensor to the Arduino board. Ensure that the sensor's power and ground pins are connected to the appropriate pins on the Arduino, and the signal pin is connected to pin A0.
- ➢ Upload the provided code to your Arduino board.
- ➢ Open the Arduino IDE and go to "Tools" > "Serial Monitor" (or press Ctrl+Shift+M) to open the Serial Monitor window.
- ➢ Ensure that the baud rate in the Serial Monitor matches the baud rate specified in your code (in this case, 9600).

- The Serial Monitor will display the moisture level percentage calculated by the Arduino. The values will update every second due to the delay specified in the code.
- Observe the values displayed in the Serial Monitor. The moisture level percentage should reflect the readings from your soil moisture sensor. Higher values indicate a higher moisture level, while lower values indicate a lower moisture level.

b. How are you assessing the output displayed in the Serial Monitor?

- If we are keeping the sensor in the less moist area, the value which is displayed in the serial monitor will be low.
- If we are keeping the sensor in the more moist area, the value which is displayed in the serial monitor will be high.

Vellore Institute of Technology
Chennai Campus

Programme: B. Tech Electronics and Communication Engineering
BECE351E: Internet of Things (IoT) Lab

Obstacle Detection using IR Sensor

Name of the student:	Rahul Karthik S
Register Number:	21BEC1851
Date of the Lab. Class:	13.07.2023

1. Introduction:

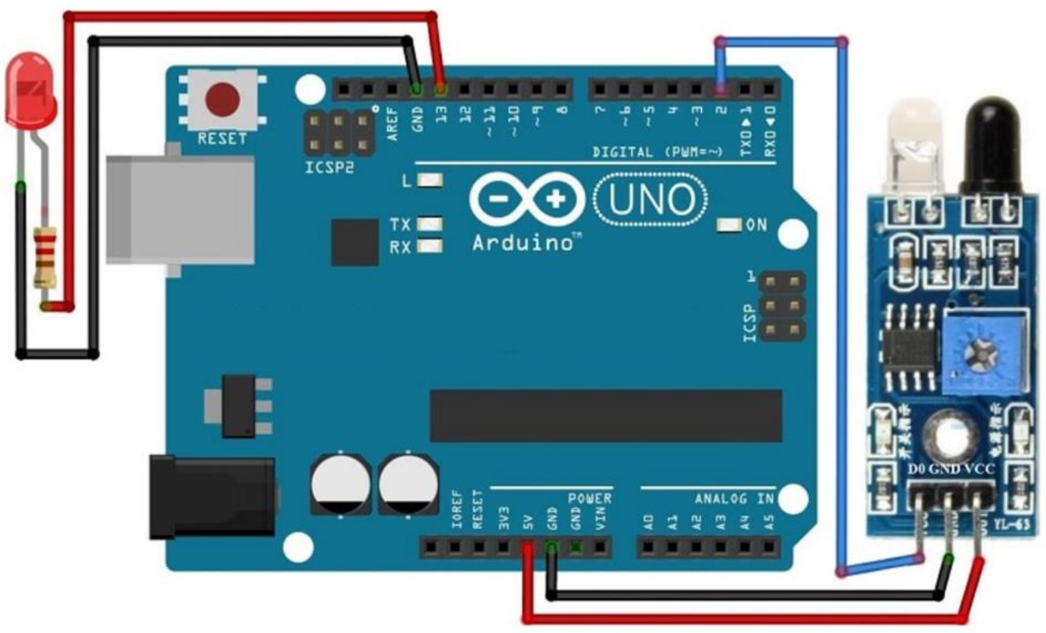
a. What are you targeting to do through this experiment?

To detect obstacle using IR Sensor and Arduino Continuously.

b. List out the required components to execute this experiment?

Components	**Quantity**
Arduino UNO R3	1
IR Sensor	1
LED	1
1 kΩ Resistor	1
Connecting Wires	5
Bread Board	1

2. Circuit Diagram & Explanation:



a. Write the detailed steps involved to design the circuit diagram in TinkerCAD.

- To design a circuit for measuring distance using an ultrasonic sensor and Arduino,
- The following components are required:
    - Arduino board (e.g., Arduino Uno)
    - Ultrasonic sensor module (e.g., HC-SR04)
    - Breadboard
    - Jumper wires
    - Resistors.
- The components mentioned above are gathered.
- The ultrasonic sensor is connected to the Arduino board using jumper wires:
    - The Vcc pin of the sensor is connected to the 5V pin on the Arduino.
    - The GND pin of the sensor is connected to the GND pin on the Arduino.
    - The Trig pin of the sensor is connected to any digital pin (e.g., Pin 2) on the Arduino.
    - The Echo pin of the sensor is connected to any digital pin (e.g., Pin 3) on the Arduino.
    - If the IR sensor module has a separate Vcc and GND pin for the Echo signal, they are connected accordingly. Otherwise, the Echo pin shares the Vcc and GND connections with the sensor.
- If the IR sensor module requires pull-up resistors for the Trig and Echo pins, a 10K resistor is connected between the Trig pin and the 5V pin on the Arduino, and another 10K resistor is connected between the Echo pin and the 5V pin on the Arduino.
- The Arduino is connected to the computer using a USB cable.
- The Arduino IDE is launched on the computer, and the code for measuring the distance using the IR sensor is written.
- The code is uploaded to the Arduino board.
- The Serial Monitor in the Arduino IDE is opened to view the distance measurements, with the baud rate set to 9600.
- The circuit is powered on, and an object is placed in front of the IR sensor. The measured distance is displayed on the Serial Monitor.

b. Explain how the circuit is functioning to detect obstacles.

The theory behind an IR (Infrared) sensor for obstacle detection using Arduino involves the use of infrared light to detect the presence of obstacles or objects in its range. IR sensors are widely used in various applications, including robotics, automation, and proximity sensing. Here's a brief explanation of the theory behind IR sensors for obstacle detection:

Working Principle:

- IR sensors work based on the principle of reflecting or emitting infrared light and detecting its reflection or absence.
- They consist of an infrared transmitter (emitter) and an infrared receiver (detector).
- The transmitter emits infrared light, which is invisible to the human eye.
- The receiver detects the reflected infrared light or absence of it.
- When an object comes into the range of the sensor, it reflects the emitted infrared light back to the receiver.

Reflection and Absorption:

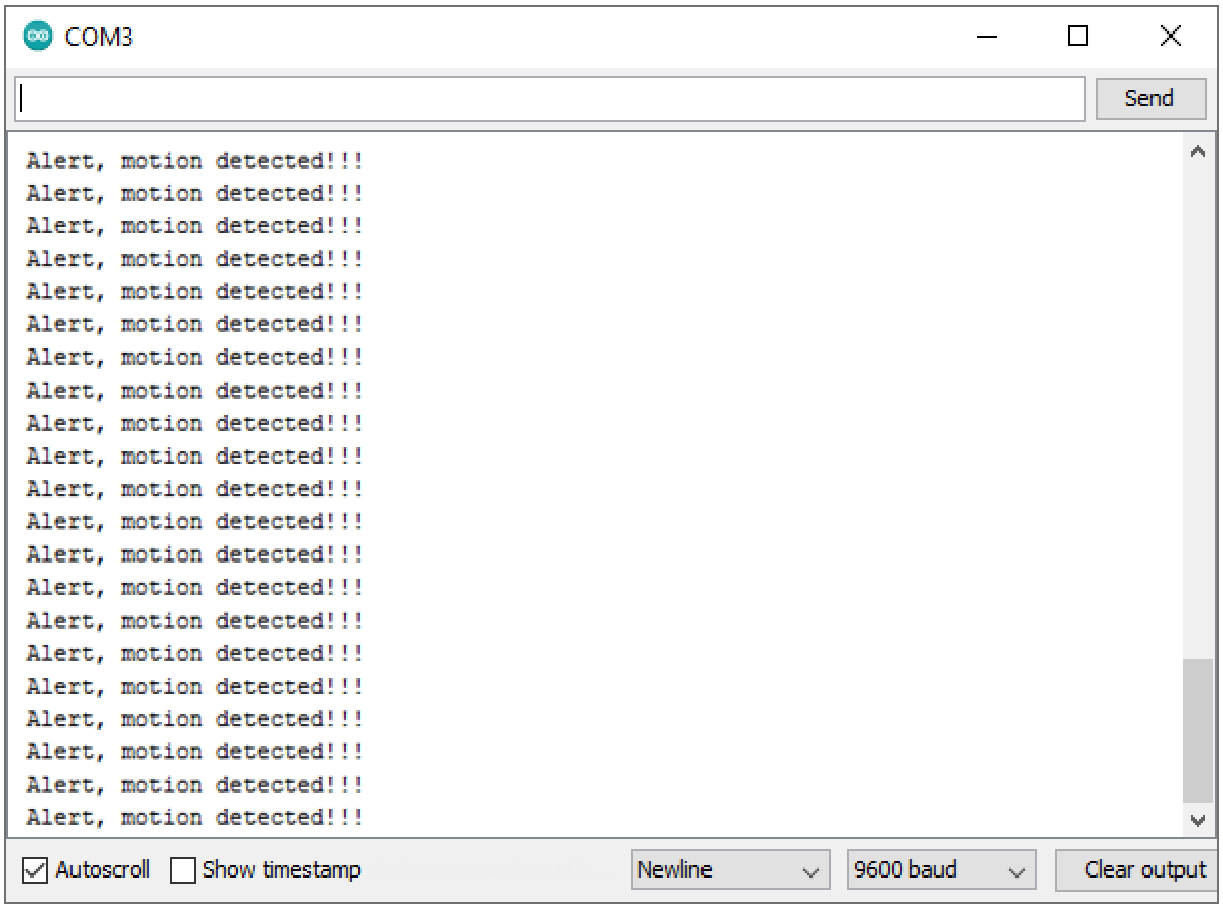
- Different surfaces reflect or absorb infrared light differently, depending on their material properties.
- Light-colored or smooth surfaces tend to reflect more light, while dark-colored or rough surfaces tend to absorb more light.
- When an object is placed in front of the sensor, the amount of reflected infrared light varies based on the surface characteristics of the object.

c. Write and describe the code utilized to run the circuit.

```
int OutputPin = 13;
int SensorPin = 2;
void setup() {
        pinMode(OutputPin, OUTPUT);
        pinMode(SensorPin, INPUT);
        Serial.begin(9600);
}
void loop(){
        int SensorValue = digitalRead(SensorPin);
        delay(1000);
        if (SensorValue == LOW){
                digitalWrite(OutputPin, HIGH);
                Serial.println("Alert, Motion Detected!!!");
        }
        else{
                digitalWrite(OutputPin, LOW);
                Serial.println("Motion is not Detected.");
        }
}
```

This code reads the digital value from a sensor connected to pin 2 and controls an output pin (pin 13) based on the sensor value. The sensor value is printed to the serial monitor, and if it is LOW, the output pin is set to HIGH; otherwise, it is set to LOW.

3. **Output of the experiment:**



```
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
Alert, motion detected!!!
```

a. **Write the detailed steps involved to run the simulation in TinkerCAD.**

- Write the Arduino code in the IDE.
- Verify that the code compiles without any errors.
    - Select the correct Arduino board and port from the Tools menu.
    - Click on the "Upload" button to upload the code to the Arduino board.
- Monitor the output:
    - Open the serial monitor in the Arduino IDE by clicking on the magnifying glass icon or navigating to Tools > Serial Monitor.
    - Set the baud rate in the serial monitor to match the one specified in the Arduino code (e.g., 9600).
    - Observe the temperature and humidity readings displayed in the serial monitor.

b. **How are you assessing the output displayed in the Serial Monitor?**

If the obstacle is detected, the LED connected to the Arduino will glow and the message, “Alert, motion detected!!!” is printed in the serial monitor. Otherwise the LED won’t glow and it will print the message, “Motion is not detected.”