LABORATORY PRACTICE I

Lab Journal

NAME :- OJUS P. JAISWAL

YEAR & DIV :- TE A

ROLL NO. :- TACO19108

SEAT NO. :- S191094290

Index

Sr. No.	Title of the Experiment	Page No.
Internet of Things and Embedded Systems		
1	Understanding the connectivity of Raspberry-Pi/Arduino with IR sensor. Write an application to detect obstacle and notify user using LEDs.	2-6
2	Understanding the connectivity of Raspberry-Pi/Arduino circuit with temperature sensor. Write an application to read the environment temperature. If temperature crosses a threshold value, generate alerts using LEDs.	7-11
3	Understanding and connectivity of Raspberry-Pi/Arduino with camera. Write an application to capture and store the image.	12-13
4	Create a small dashboard application to be deployed on cloud. Different publisher devices can publish their information and interested application can subscribe.	14-16

Assignment No. 1

Title: Understanding the connectivity of Raspberry-Pi /Arduino circuit with IR sensor. Write an application to detect obstacle and notify user using LEDs.

Aim: To understand the interface between IR sensor and Arduino.
Demonstrate when IR detects any obstacle it counts the number of times obstacle detected and displays on screen.

Hardware Requirement: Arduino, IR Sensor, LED.

Software Requirement: Tinker CAD.

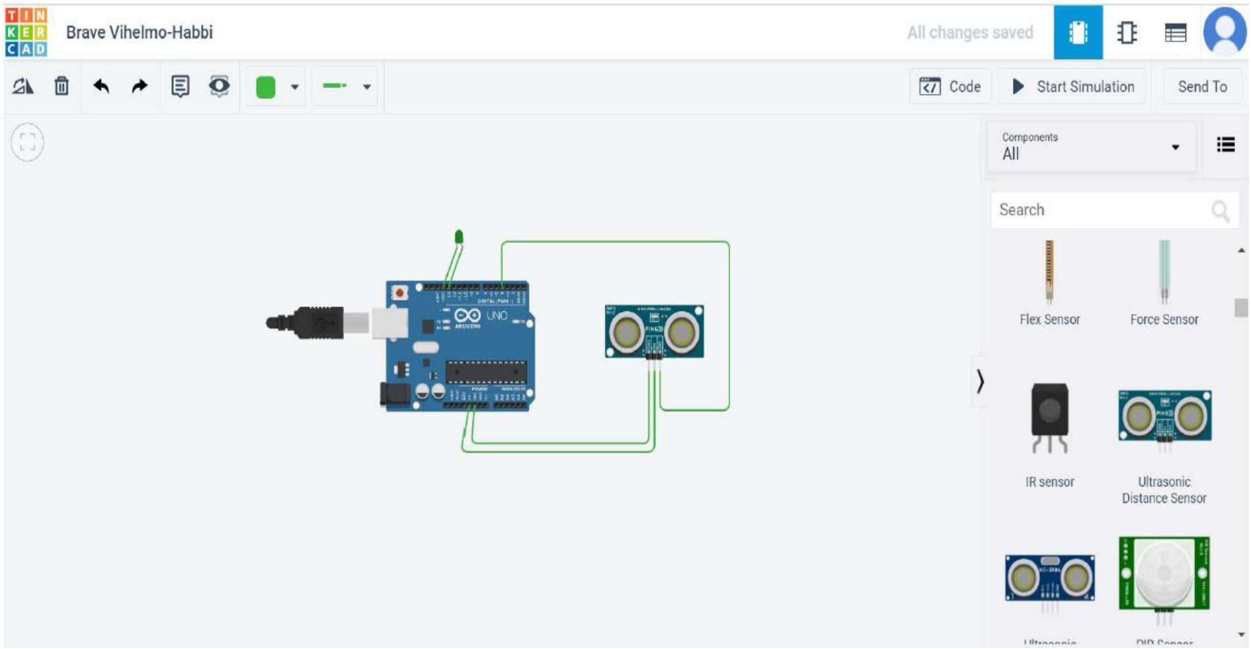
Theory:

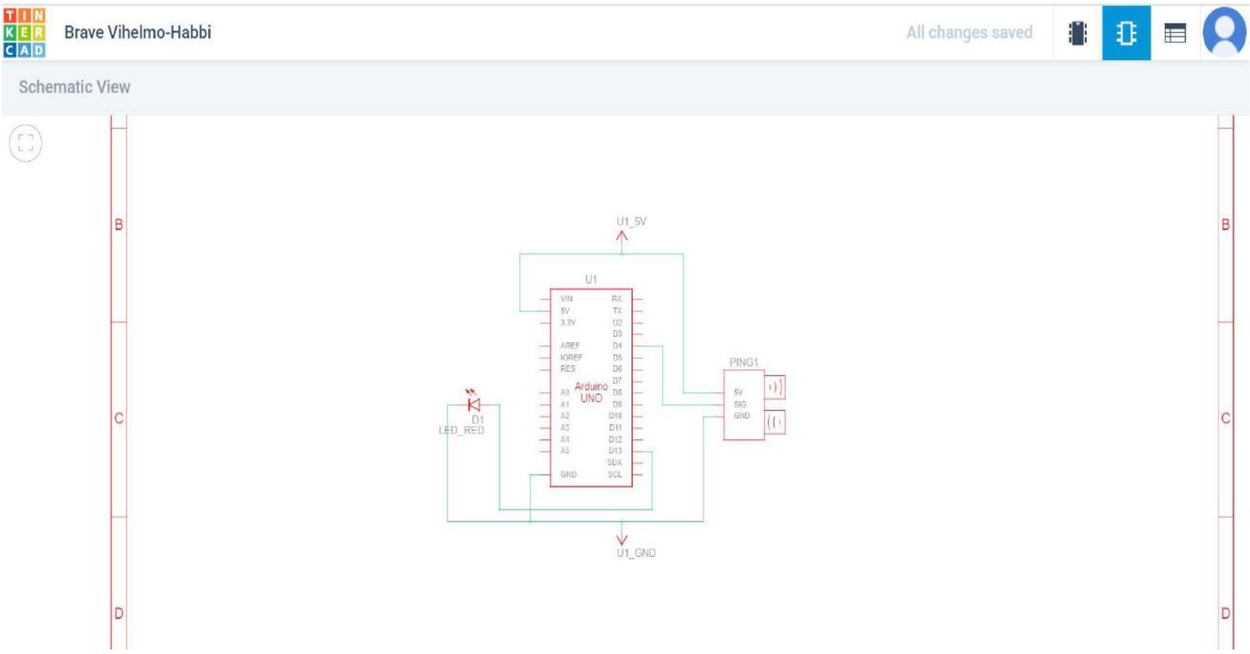
An infrared sensor is an electronic instrument which is used to sense certain characteristics of its surroundings by either emitting and/or detecting infrared radiation. Infrared sensors are also capable of measuring the heat being emitted by an object and detecting motion.
Infrared waves are not visible to the human eye. In the electromagnetic spectrum, infrared radiation can be found between the visible and microwave regions. The infrared waves typically have wavelengths between 0.75 and 1000µm.

The IR Transmitter transmits a signal at 38 KHz. A IC555 generates the 38KHz frequency which is then transmitted through the IR transmitter.
The IR receiver is a TSOP1738 sensor module. This IR sensor module consists of a PIN diode and a preamplifier which are embedded into a single package. The output of TSOP is active low and it gives +5V in off state. When IR waves, from a source, with a center frequency of 38 kHz incident on it, its output goes low. TSOP module has an inbuilt control circuit for amplifying the coded pulses from the IR transmitter. A signal is generated when PIN photodiode receives the signals. This input signal is received by an automatic gain control (AGC). For a range of inputs, the output is fed back to AGC in order to adjust the gain to a suitable level. The signal from AGC is passed to a band pass filter to filter undesired frequencies. After this, the signal goes to a demodulator and this demodulated output drives a NPN transistor.

The collector output of the transistor is obtained at pin 3 of TSOP module.
As soon as we execute the program the sensor starts sensing if there is any obstacle around it. If it found any kind of obstacle it sends input to program and according to the logic if any input is detected the no of persons count gets increased and the count is displayed in screen.

Connection Diagram:



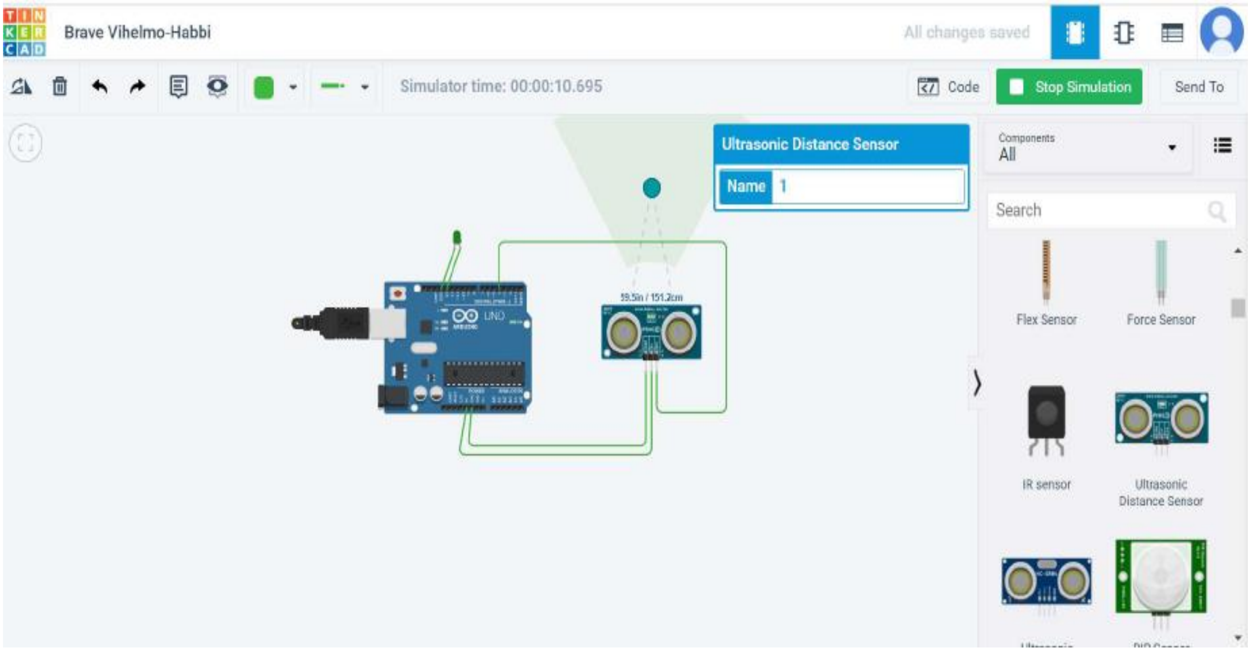


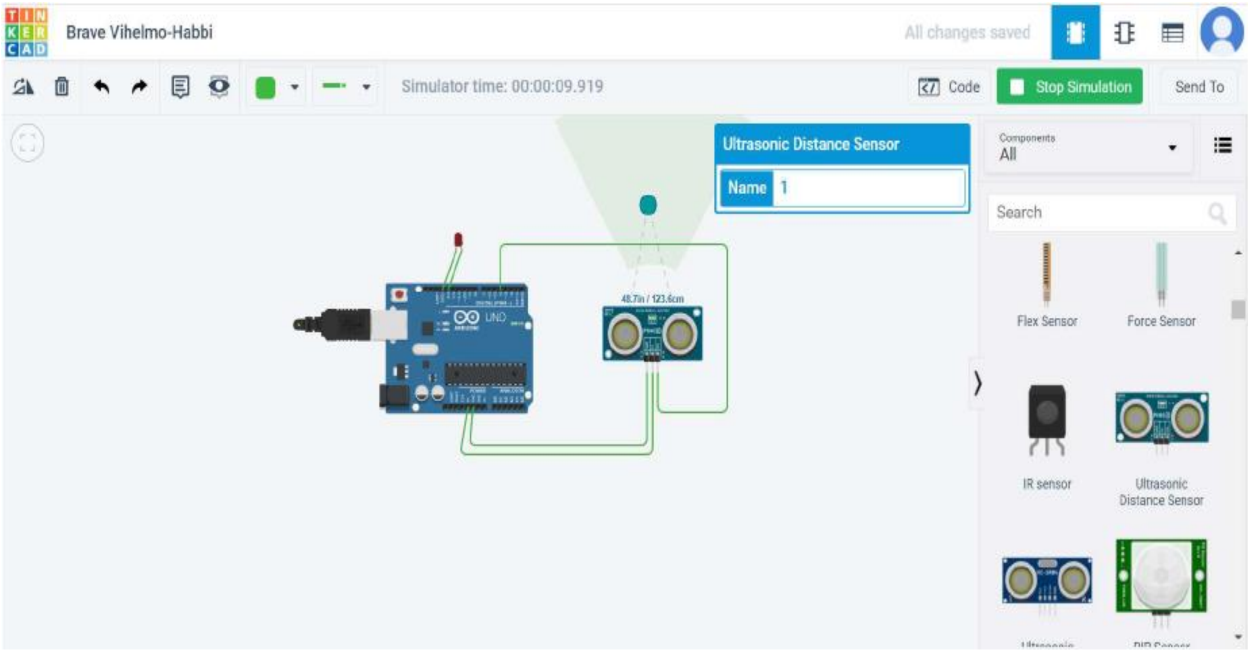
Code:

```
const int pingPin = 4; // Trigger Pin of Ultrasonic Sensor
const int ledPin = 13; // led Pin of Ultrasonic Sensor

void setup() {
   Serial.begin(9600);
  pinMode(ledPin,OUTPUT);// Starting Serial Terminal
}

void loop() {
   long duration,cm;
   pinMode(pingPin, OUTPUT);
   digitalWrite(pingPin, LOW);
   delayMicroseconds(2);
   digitalWrite(pingPin, HIGH);
   delayMicroseconds(5);
   digitalWrite(pingPin, LOW);

   pinMode(pingPin, INPUT);
   duration = pulseIn(pingPin, HIGH);

   cm = microsecondsToCentimeters(duration);
   if(cm<100){
    digitalWrite(ledPin,HIGH);
   }
   else{
    digitalWrite(ledPin,LOW);
   }
   delay(100);
}

long microsecondsToCentimeters(long microseconds) {
   return microseconds / 29 / 2;
}
```

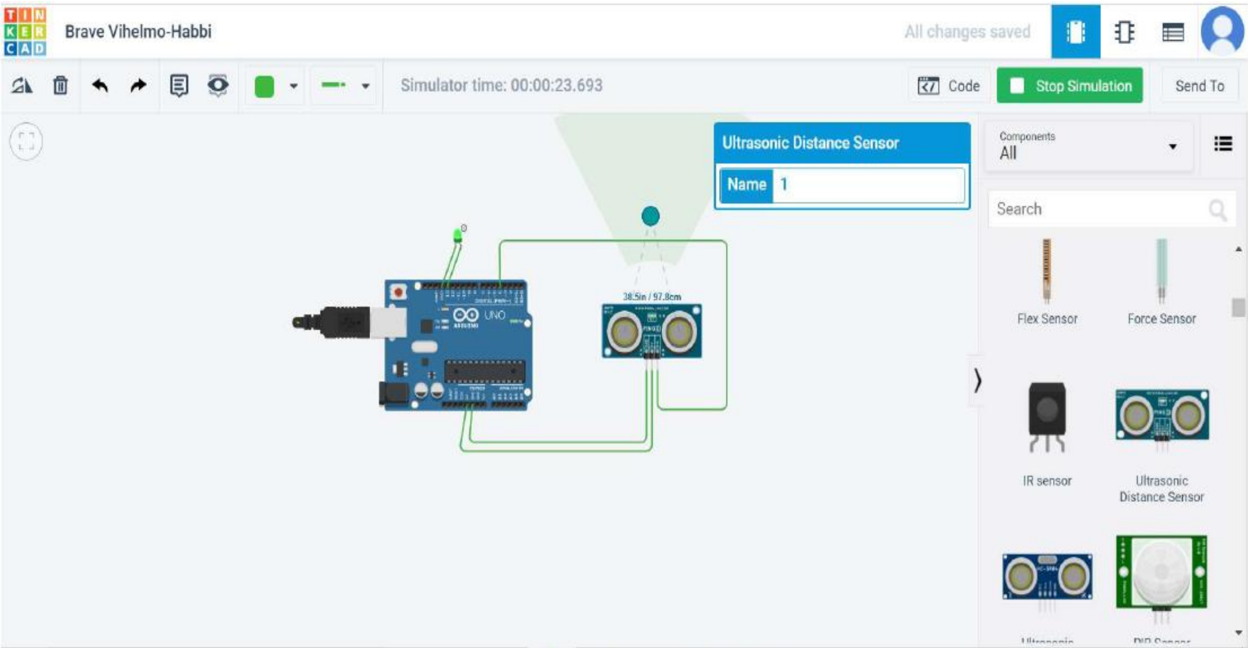
Simulation Screenshots:

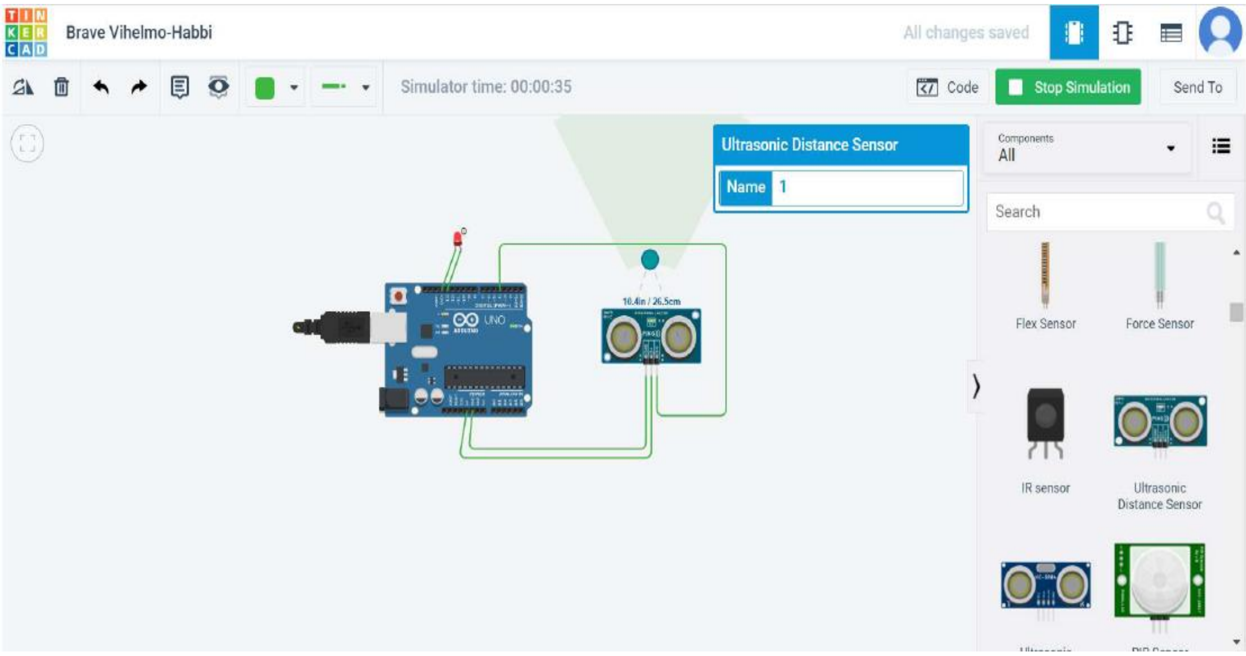
1. When distance between IR sensor and is greater than 100 cm, LED is off =>





2. When distance between IR sensor and obstacle is less than 100 cm, LED turns on =>





Conclusion: We have successfully executed the program of displaying no. of obstacle found using IR sensor. And displayed on screen.

Assignment No. 2

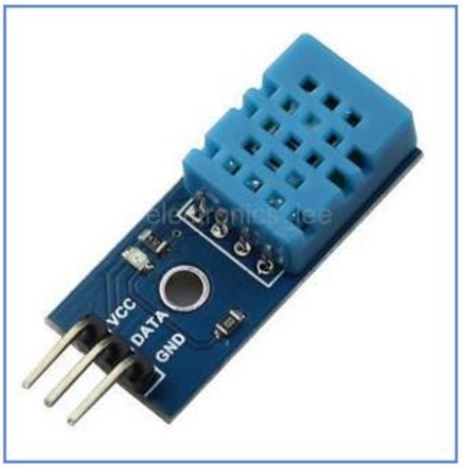
Aim: Understanding the connectivity of Raspberry-Pi/Arduino circuit with temperature sensor. Write an application to read the environment temperature. If temperature crosses a threshold value, generate alerts using LEDs.

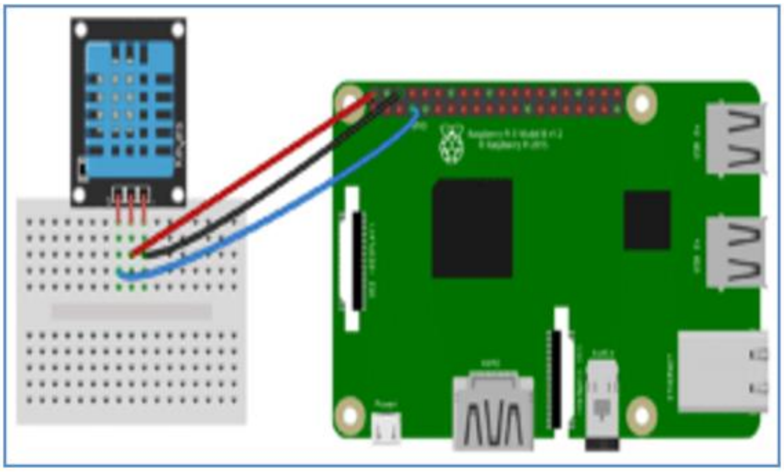
Requirement: Temp36 sensor, Arduino, breadboard.

Theory:

Three Pin DHT11 with SSH Output =>

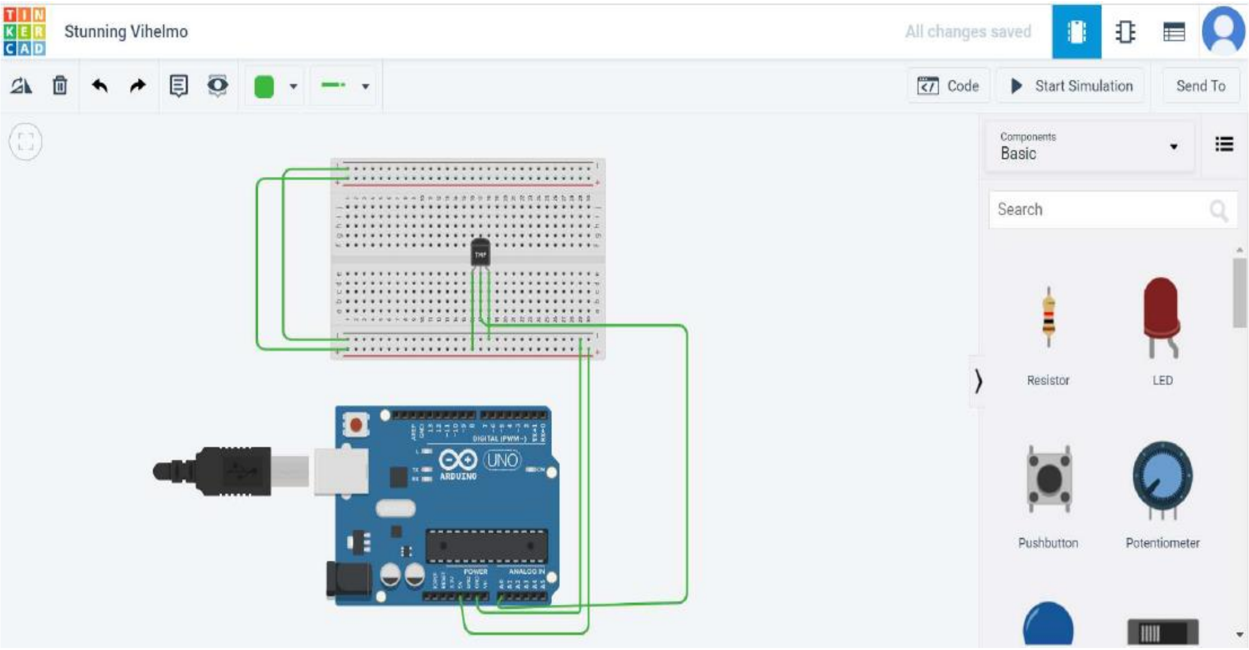
DHT11 is a Humidity and Temperature Sensor, which generates calibrated digital output. DHT11 can be interface with any microcontroller like Arduino, Raspberry Pi, etc. and get instantaneous results. DHT11 is a low-cost humidity and temperature sensor which provides high reliability and long-term stability.

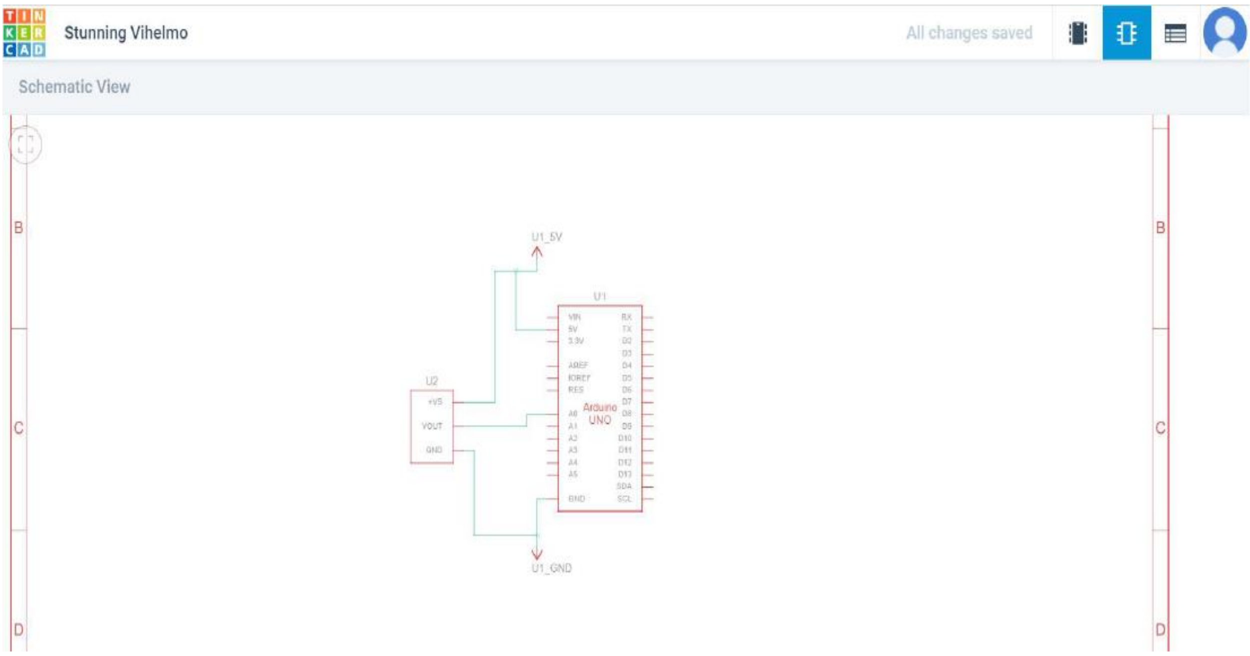




DHT11 Temperature sensor   Connection of DHT11 Temperature sensor to Arduino

Connection Diagram:



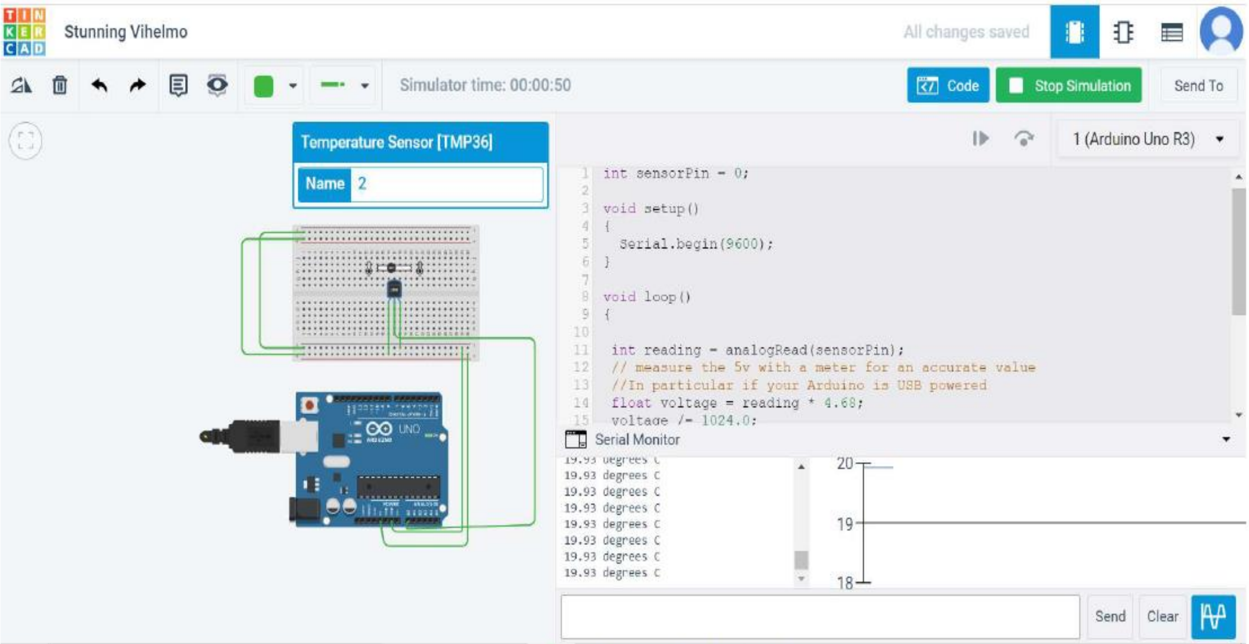


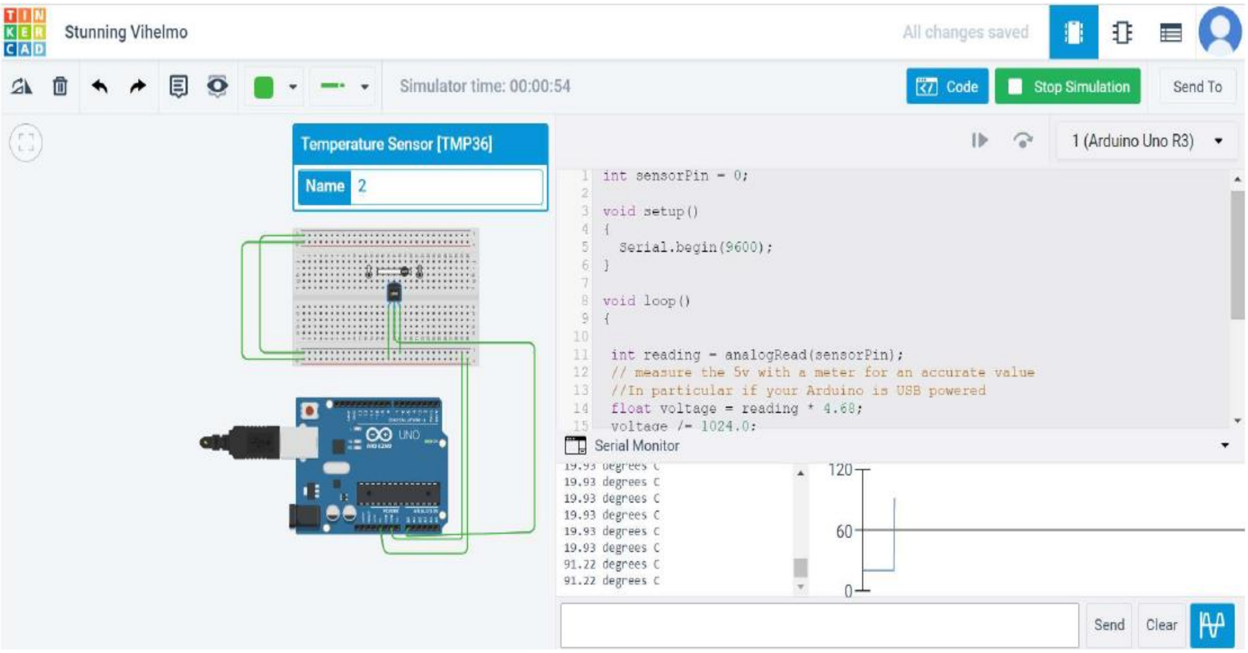
Code:

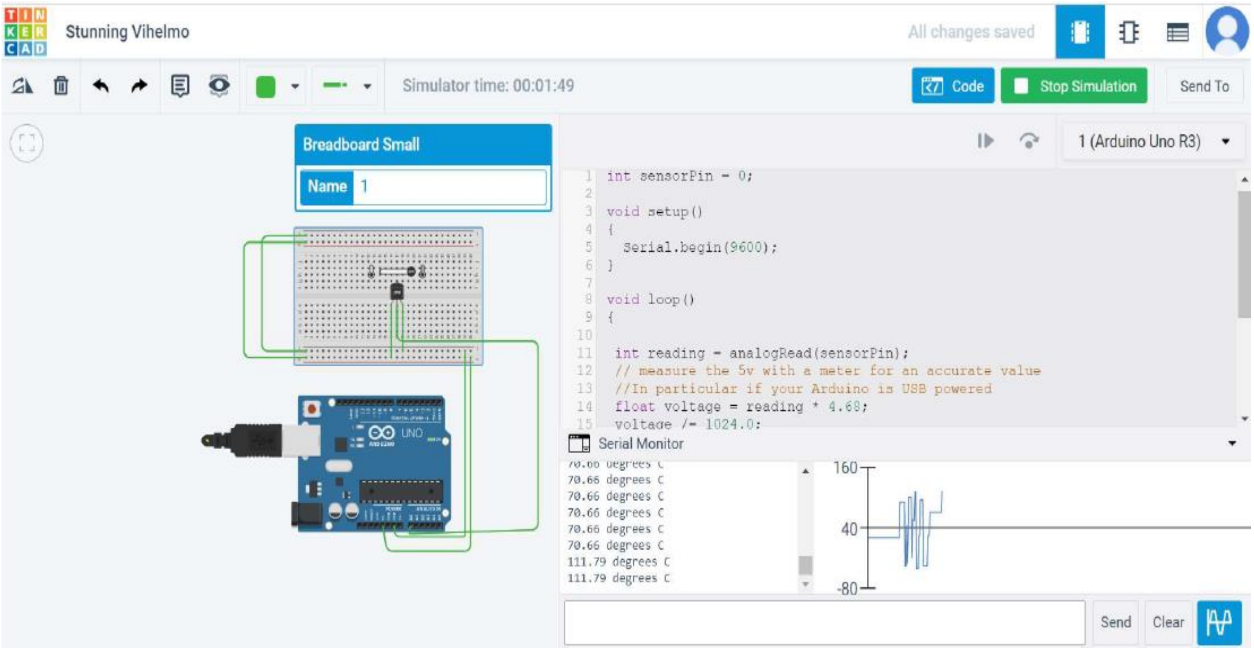
```
int sensorPin = 0;

void setup ()
{
        Serial.begin(9600);
}

void loop ()
{

        int reading = analogRead(sensorPin);
        // measure the 5v with a meter for an accurate value
        //In particular if your Arduino is USB
        powered float voltage = reading * 4.68;
        voltage /= 1024.0;

        // now print out the temperature
        float temperatureC = (voltage - 0.5) *
        100; Serial.print(temperatureC);
        Serial.println(" degrees C");

        delay (1000);
}
```

Simulation Screenshot:

Temperature reading with graph =>







Conclusion: Thus, we have studied connectivity of Arduino with temperature sensor.

Assignment No. 3

Title: Understanding and Connectivity of Raspberry Pi/Arduino with Camera.
Write an application to capture and store the image.

Aim: To understand the connectivity of Camera and store the image with Raspberry Pi Board. Demonstrate a smart traffic system using Raspberry Pi and Python.

Hardware Requirement:
1. Raspberry Pi Kit
2. The Raspberry Pi Camera Module v2 has a resolution of 5 megapixel and has a fixed focus lens onboard.
3. The camera is capable of 2592x1944-pixel static images

Software Requirement: Raspbian O.S, Python.

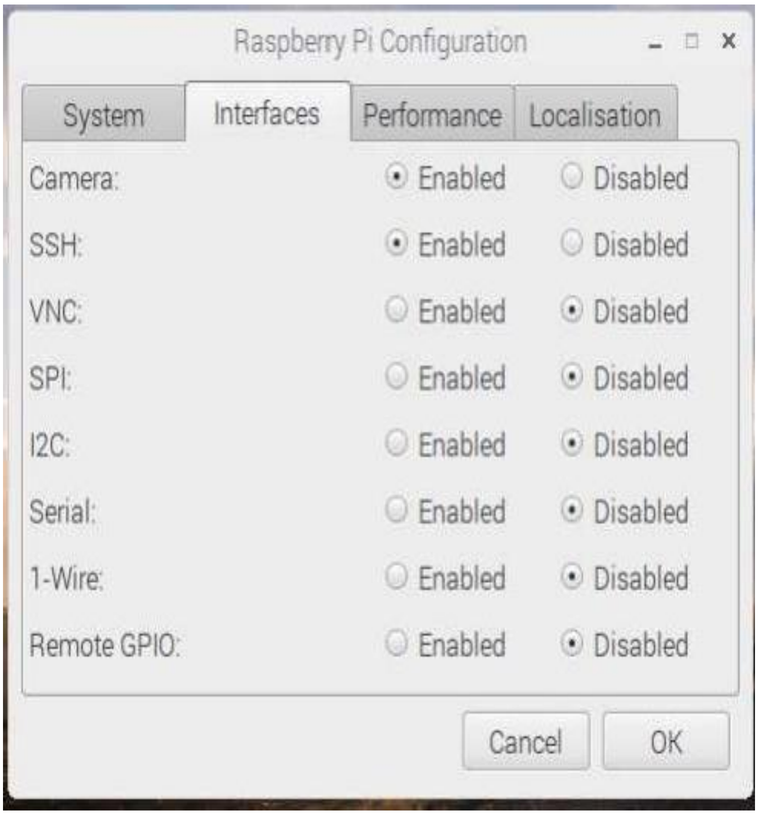
Theory:

Raspberry Pi Camera Module =>

1. The Raspberry Pi Camera Module v2 replaced the original Camera Module in April 2016.
2. The v2 Camera Module has a Sony IMX219 8-megapixel sensor (compared to the 5- megapixel Omni Vision OV5647 sensor of the original camera).
3. The Camera Module can be used to take high-definition video, as well as stills photographs. It's easy to use for beginners, but has plenty to offer advanced users if you're looking to expand your knowledge.
4. You can also use the libraries we bundle with the camera to create effects.
5. You can read all the gory details about IMX219 and the Exmore R back-illuminated sensor architecture on Sony's website, but suffice to say this is more than just a resolution upgrade: it's a leap forward in image quality, color fidelity, and low-light performance.
6. It supports 1080p30, 720p60 and VGA90 video modes, as well as still capture. It attaches via a 15cm ribbon cable to the CSI port on the Raspberry Pi
7. The camera works with all models of Raspberry Pi 1, 2, and 3.
8. It can be accessed through the MMAL and V4L APIs, and there are numerous third-party libraries built for it, including the Pi camera Python library.
9. The camera module is very popular in-home security applications, and in wildlife camera traps.

Steps to install the Pi Camera =>

1. Go to Preferences
2. Open Raspberry Pi configuration.
3. Configure Raspberry Pi system.
4. Enable the camera Interface as shown in the diagram.



Code:

```
import picamera

from time import sleep

camera = picamera.PiCamera()

camera.capture('image1.jpg')

sleep(5)
```

Conclusion: We have successfully captured and stored the image using Raspberry Pi.

Assignment No. 4

Title: Create a small dashboard application to be deployed on cloud. Different publisher devices can publish their information and interested application can subscribe.

Aim: Create a small dashboard application to be deployed on cloud.

Hardware Requirement: PIV, 2GB RAM, 500 GB HDD, Lenovo A13-4089Model.

Software Requirement: Raspbian O.S, Python.

Theory:

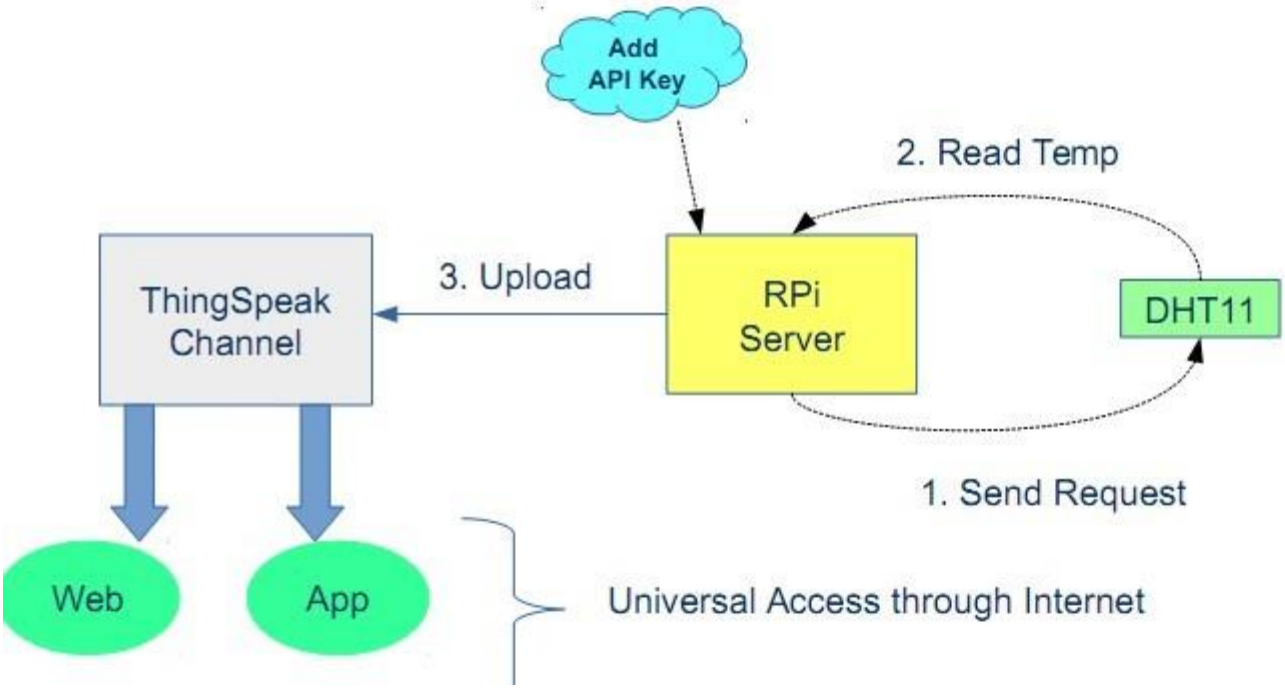
The IoT platforms are suites of components those help to setup and manage the internet connected devices.
A person can remotely collect data, monitor and manage all internet connected devices from a single system.
There are a bunch of IoT platforms available online but building an IoT solution for a company is all depend on IoT platform host and support quality.

1. Kaa => IoT Platform
2. SiteWhere => Open Platform for the Internet of Things
3. ThingSpeak => An open IoT platform with MATLAB analytics
4. DeviceHive => IoT Made Easy
5. Zetta => API-First Internet of Things Platform
6. DSA => Open-Source Platform & “Toolkit” for Internet Of Things Devices
7. Thingsboard.io => Open-source IoT Platform
8. Thinger.io => The Opensource Platform for Internet of things
9. WSo2 => Open-source platform for Internet of Things and mobile projects

ThingsSpeak =>

1. Collect data in private channels
2. Share data with public channels
3. RESTful and MQTT APIs
4. MATLAB analytics and visualizations
5. Alerts
6. Event scheduling
7. App integrations
8. Worldwide community


ThingSpeak.com
imp cloud
Internet
ThingSpeak
user
setup
BlinkUp
imp
WiFi


Add
API Key
2. Read Temp
3. Upload
ThingSpeak
Channel
RPi
Server
DHT11
1. Send Request
Web
App
Universal Access through Internet

Steps =>

1. Create account on ThingSpeak site.
2. Login to ThingsSpeak account
3. Click on My Channel = Create New Channel
4. Click on Sharing = Select Share Channel view with everyone.
5. Click on API Key = Copy API key Paste in Key field of Code.
6. Now Run the code.
7. See the Result of Temp and Humidity Graph on ThingsSpeak Account.

Code:

```
import httplib,urllib
import time,Adafruit_DHT
key='8QPPQALZTUZUZ7IS'
while True:
	h,t=Adafruit_DHT.read_retry(11,4)
	print "temp:",t
	param=urllib.urlencode({'field1':t,'key':key})
	headers={"content-typZZe":"application/x-www-form-urlencoded","Accept":"text/plain"}
	conn=httplib.HTTPConnection("api.thingspeak.com:80")
	try:
		conn.request("POST","/update",param,headers)
		response=conn.getresponse()
		printresponse.status,response.reason
		data=response.read()
		conn.close()
	except:
		print "connection Failed"
```

Conclusion: We have successfully executed the program and able to import the output of sensor on cloud account using Raspberry-Pi.