DIPLOMA IN COMPUTER ENGINEERING

4052550 – CLOUD COMPUTING AND INTERNET OF THINGS PRACTICAL

N – SCHEME

(V SEM)

2022 – 2023

NAME : ……...

ROLL NO : ……...

CLASS : ……...

INDEX

SL. NO	DATE	NAME OF THE EXPERIMENT	MARKS	SIGN
PART A – CLOUD COMPUTING				
1.		IMPLEMENTATION OF SAAS USING WORD		
2.		IMPLEMENTATION OF SAAS USING EXCEL		
3.		IMPLEMENTING WEB SERVICES		
4.		INSTALLING GOOGLE APP ENGINE		
5.		INSTALLING VIRTUAL BOX		
6.		INSTALLING OPEN STACK		
7.		CASE STUDY Amazon EC2 and Azure		
PART B – INTERNET OF THINGS				
8.		LED BLINK AND LED PATTERN		
9.		LED PATTERN WITH PUSH BUTTON		
10.		DISPLAYING HELLO WORLD		
11.		IMPLEMENTING SERVO MOTOR CONTROL		
12.		IMPLEMENTATION OF LM35 TEMPERATURE SENSOR		
13.		IMPLEMENTATION OF IR SENSOR WITH ANALOG INPUT		
14.		TEMPERATURE SENSOR MONITORING WITH RASPBERRY PI		

EX. NO : 1 **Date:**	**IMPLEMENTATION OF SAAS USING WORD**

AIM:

To implement program on SaaS to create a word document of your class time table and store locally and on cloud with .doc and .pdf format

PROCEDURE:

- Create a new document
- You can create a new document right in Docs or in Google Drive.
- In Docs, click Create new document.
- In Drive, click New → Google Docs → Blank document
- Create Class timetable

To share documents

1. Open the file you want to share.
2. Click Share .
3. Enter the email addresses or Google Groups you want to share with.
4. Click Send.
5. Everyone you shared the document with receives an email with a link to the document.

OUTPUT:

RESULT:

Thus, we have implemented program on SaaS to create an word document and store locally and on cloud .

EX. NO : 2 **Date:**	**IMPLEMENTATION OF SAAS USING EXCEL**

AIM:

To implement program on SaaS to Create a spread sheet to generate a mark sheet for student progress report.

PROCEDURE:

- We can use Google Spreadsheet directly from the web browser .
- Visit **docs.google.com/spreadsheets** and sign in with your Google or Gmail account.
- Create a new spreadsheet.
- Type in the name of the spreadsheet and click the "OK" button. The name changes immediately.
- Create a student marksheet and type the contents.
- There's no need to save Google Sheets as everything is automatically saved at regular intervals.
- Exit the spreadsheet when the work gets finished.

OUTPUT:

RESULT:

Thus, we have implemented program on SaaS to create an excel spreadsheet.

EX. NO : 3 **Date:**	**IMPLEMENTING WEB SERVICES**

AIM:

To implement web services by create your BlogSpot and Collaborating via Wikis

PROCEDURE:

Adding a wiki

- Activate the Wiki plugin and select **Plugins → All**
- We will see a new Wiki menu item added to the main navigation.
- Creating a new wiki is just like creating a new post on the blog.
- Go to **Wikis → Add Wiki** in the dashboard.
- Give a title to the wiki page. Type in the content that we want to start with.
- This can be changed on the front end of the blog at any time later.
- Choose who will be able to edit the content.
- Using the Discussion tab on the wiki either select or deselect **"Allow Comments"** in the **Discussion module**.
- Decide if you want to enable email notifications of updates on the wiki page.
- Email notifications are used to provide updates on when changes are made to the wiki page.
- Click on Publish in the upper right corner.

Editing a page

- For more advanced editing, and to upload images or media to your wiki page, you just need to click the Advanced link .
- Clicking on Advanced takes you to the Advanced editor in the WordPress admin dashboard.
- Only people who can edit posts will see the Advanced link.

OUTPUT:

RESULT:

Thus we have implemented web services to create BlogSpot and Collaborate via Wikis.

EX. NO : 4 **Date:**	**INSTALLING GOOGLE APP ENGINE**

AIM:

To implement on PaaS to Install Google App Engine, create a program to validate User and create a database login in mysql and deploy to cloud.

PROCEDURE:

Before we proceed with the installation process follow the steps.

- Create a Compute Engine instance
- Install MySQL
- Connect to MySQL

To create a Compute Engine instance in the Google Cloud console:

1. Open the Google Cloud console.
2. Select your newly created project and click **Continue**.
3. Click **Create instance**
4. Name the instance **mysql-test**.
5. In the **Public images** tab, select an operating system and then click **Save**.
6. Click **Create**.

Set up a Cloud SQL instance

- Enable the Cloud SQL Admin API in the project .
- Create a Cloud SQL for MySQL instance.
- By default, Cloud SQL assigns a public IP address to a new instance. We also have the option to assign a private IP address.

Create a Cloud SQL Second Generation instance.

We can set the password for the default user on your Cloud SQL instance:

gcloud sql users set-password root --host=% --instance [INSTANCE_NAME] --password[PASSWORD]

Record the connection name for the instance:

gcloud sql instances describe [INSTANCE_NAME]

Create a database on your Cloud SQL instance.

```
gcloud sql databases create [DATABASE_NAME] --instance=[INSTANCE_NAME]
```

Setting up your local environment

Authenticate the gcloud tool to use the proxy to connect from your local machine:

```
gcloud auth application-default login
```

Setting connection strings and adding a library

Set up the local environment to support connections for local testing. For example, for the provided code sample:

```
export MYSQL_DSN="mysql:host=127.0.0.1;port=3306;dbname=DATABASE;"
export MYSQL_USER=USER
export MYSQL_PASSWORD=PASSWORD
```

To allow your app to connect to your Cloud SQL instance when the app is deployed, add the user, password, database, and instance connection name variables from Cloud SQL to the related environment variables in the app.yaml file:

```
env_variables:
  # Replace USER, PASSWORD, DATABASE, and CONNECTION_NAME with the
  # values obtained when configuring your Cloud SQL instance.
  CLOUDSQL_USER:
  CLOUDSQL_PASSWORD:
  CLOUDSQL_DSN: "mysql:dbname=DATABASE;unix_socket=/cloudsql/CONNECTION_NAME"
```

OUTPUT:

RESULT:

Thus, we have installed Google App Engine to implement PaaS.

EX. NO : 5 **Date:**	**INSTALLING VIRTUAL BOX**

AIM:

To install Virtual Box on Windows operating System.

PROCEDURE:

- Open Virtual Box download page, scroll down and find the latest version
- Click on the latest version number.
- Scroll down, find the `.exe` file, and download it
- Once VirtualBox Windows installer is downloaded, run the executable file.
- Follow through onscreen instructions to install VirtualBox on Windows.
- VirtualBox installs virtual network adapters and therefore we may lose network connectivity temporarily during installation.
- After VirtualBox installation finishes we will have to restart your computer.
- After reboot, VirtualBox should be available in your apps as shown below.
- We can now run VirtualBox and create Windows virtual machine with almost any OS.

OUTPUT:

RESULT:

Thus, we have installed Virtual Box on top of Windows operating system.

EX. NO : 6 **Date:**	**INSTALLING OPEN STACK**

AIM:

To install OpenStack and use it as Infrastructure as a Service and use technology own Cloud.

PROCEDURE:

The steps required to install OpenStack are as follows:

- **Install Virtual Box or Create Virtual Machine.**

Download the Oracle virtual box and create the VM machine with a specific configuration of 64 bit OS with 8GB RAM and 300 GB of memory.

After creating the VM for a specific OS required, open the terminal and disable the firewall.

- **Download the OpenStack version.**

Use the below command to download the OpenStack version through the terminal. The command is 'yum install –y centos-release-OpenStack-newton.'

- **Update the packages.**

Use the below command to update the package.

The command is 'yum update –y'.

- **Use the tool to install OpenStack.**

The command is 'yum install –y OpenStack-packstack'.

- **Installing services.**

The below command is used to install all the services or components for OpenStack. The command is 'packstack –allinone'.

It will install everything and installation complete for OpenStack.

OUTPUT:

RESULT:

Thus we have installed OpenStack to use it as an IaaS.

EX. NO : 7 **Date:**	**CASE STUDY**

AIM:

Case study on open source and commercial cloud.

THEORY:

Amazon Web Services (AWS) and Microsoft Azure are the two giants in the world of cloud computing.

While AWS is the largest cloud computing platform, Microsoft Azure is the fastest-growing and second-largest.

Azure is a cloud computing platform and an online portal that allows you to access and manage cloud services and resources provided by Microsoft. These services and resources include storing your data and transforming it, depending on your requirements. To get access to these resources and services, all you need to have is an active internet connection and the ability to connect to the Azure portal.

Azure provides more than 200 services, are divided into 18 categories. These categories include computing, networking, storage, IoT, migration, mobile, analytics, containers, artificial intelligence, and other machine learning, integration, management tools, developer tools, security, databases, DevOps, media identity, and web services.

Amazon Elastic Compute Cloud (Amazon EC2) provides scalable computing capacity in the Amazon Web Services (AWS) Cloud. Using Amazon EC2 eliminates your need to invest in hardware up front, so you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. Amazon EC2 enables you to scale up or down to handle changes in requirements or spikes in popularity, reducing your need to forecast traffic.

EX. NO : 8 **Date:**	**LED BLINK AND LED PATTERN**

AIM:

To implement LED Blink and LED Pattern With Arduino

PROCEDURE:

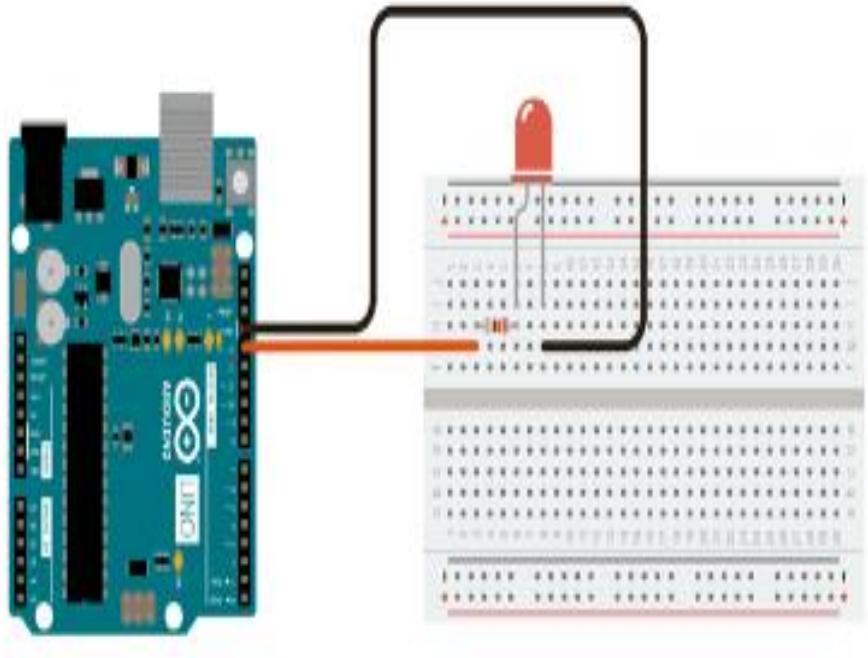
COMPONENTS REQUIRED

- This example uses the built-in LED that most Arduino boards have.
- This LED is connected to a digital pin and its number may vary from board type to board type.
- After we build the circuit plug the Arduino board into the computer, start the Arduino Software (IDE) and enter the code .
- The first thing to do is to initialize LED_BUILTIN pin as an output pin with the line.

CODE:

```
void setup()
{
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop()
{
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

CIRCUIT DIAGRAM



OUTPUT:

RESULT:

Thus, to implement led blink and led pattern with Arduino.

EX. NO : 9 **Date:**	**LED PATTERN WITH PUSH BUTTON**

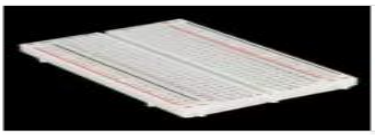
AIM:

To implement LED pattern with push button control with Arduino.

PROCEDURE:

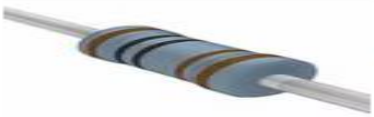
COMPONENTS REQUIRED

Arduino UNO



Breadboard (generic)

LED (generic)



Resistor 1k ohm

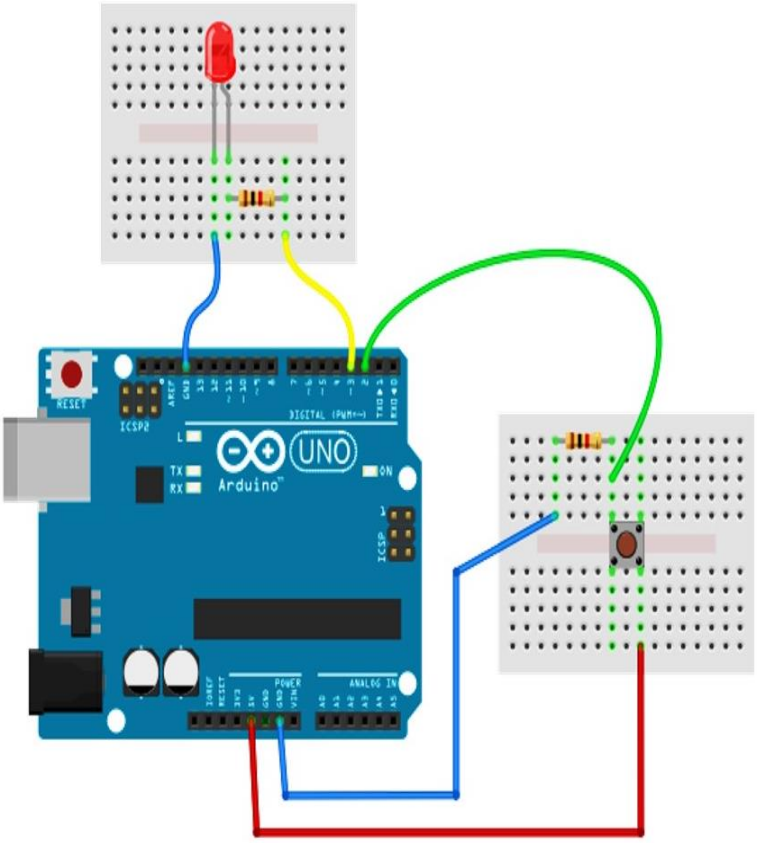
Jumper wires (generic)

CODE

```
const int BUTTON = 2;
const int LED = 3;
int BUTTONstate = 0;

void setup()
{
  pinMode(BUTTON, INPUT);
  pinMode(LED, OUTPUT);
}
```

```
void loop()
{
  BUTTONstate = digitalRead(BUTTON);
  if (BUTTONstate == HIGH)
  {
    digitalWrite(LED, HIGH);
  }
  else{
    digitalWrite(LED, LOW);
  }
}
```

CIRCUIT DIAGRAM



OUTPUT:

RESULT:

Thus, we have implemented LED pattern with push button control with Arduino.

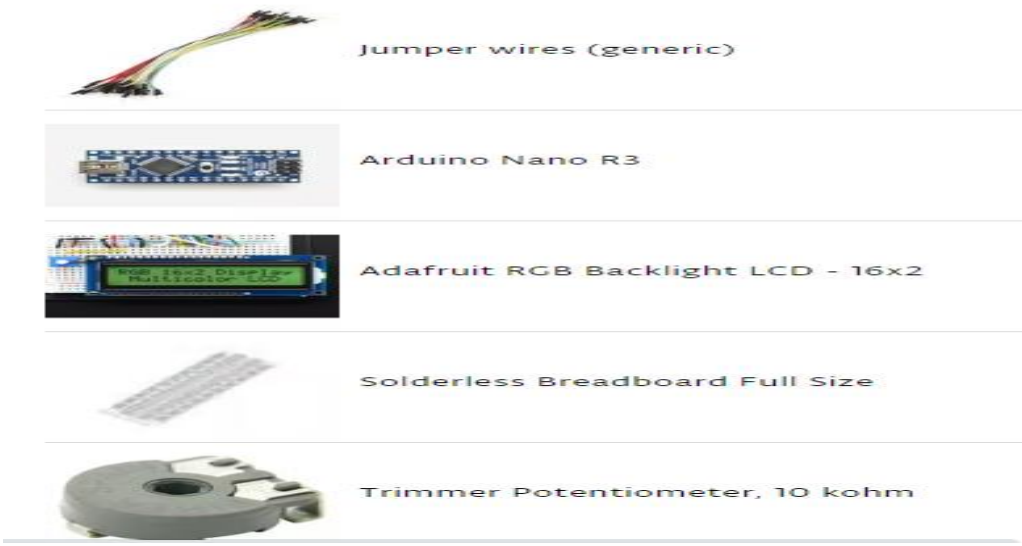
EX. NO : 10 **Date:**	**DISPLAYING HELLO WORLD**

AIM:

To display "Hello World" in LCD 16x2 display with Arduino.

PROCEDURE:

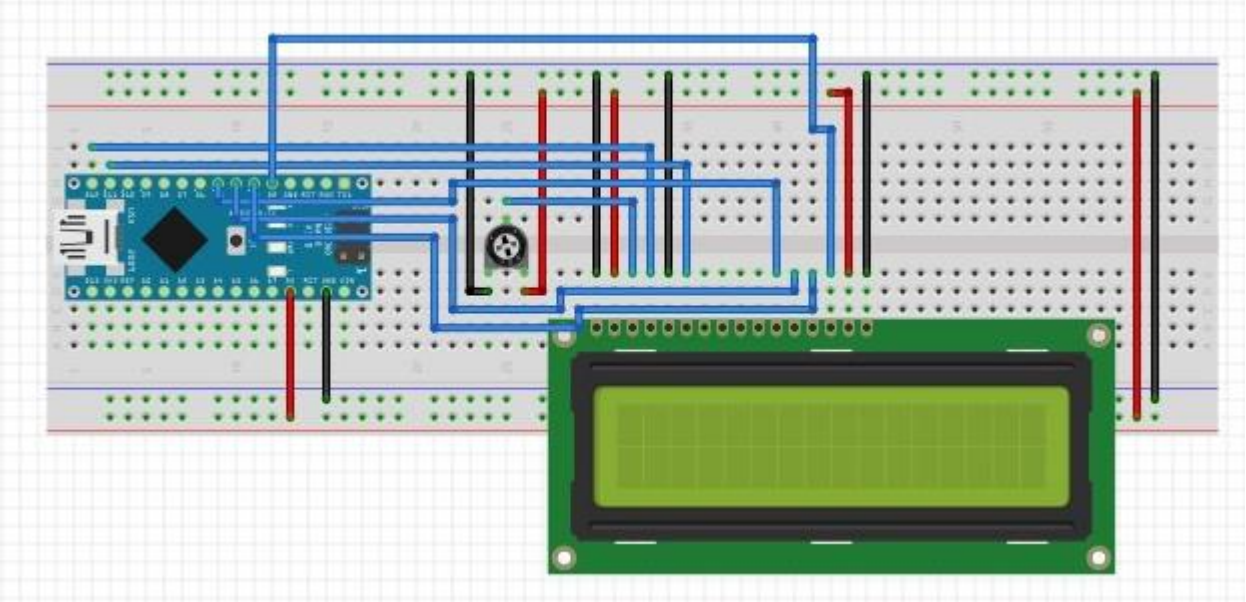
COMPONENTS REQUIRED:



CODE

```
#include <LiquidCrystal.h>

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

void setup()
{
        lcd.begin(16, 2);
        lcd.clear();
}

void loop()
{
        lcd.print(" Hello world!");
        lcd.setCursor(0, 1);
}
```

CIRCUIT DIAGRAM:



OUTPUT:

RESULT:

Thus, we have used 16x2 LCD display to display Hello World.

EX. NO : 11 **Date:**	**TO IMPLEMENT SERVO MOTOR CONTROL**

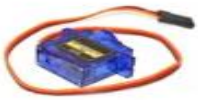
AIM:

To implement Servo motor control with Arduino.

PROCEDURE:

COMPONENTS REQUIRED

Arduino UNO



SG90 Micro-servo motor

UTSOURCE Electronic Parts

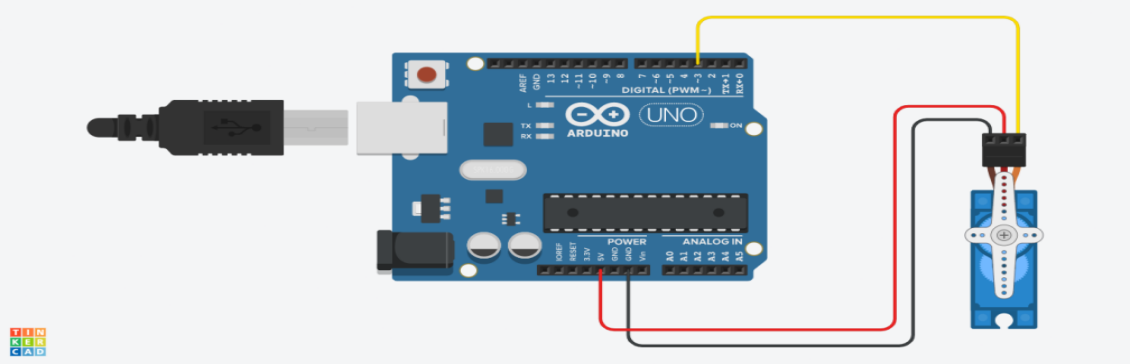
Jumper wires (generic)

CODE

```
#include<Servo.h>
Servo Myservo;
int pos;
void setup()
{
Myservo.attach(3);
}
void loop()
{
     for(pos=0;pos<=180;pos++){
     Myservo.write(pos);
     delay(15);
}
  delay(1000);
```

```
  for(pos=180;pos>=0;pos--)
{
      Myservo.write(pos);
      delay(15);
}
  delay(1000);

}
```

CIRCUIT DIAGRAM



OUTPUT:

RESULT:

Thus, we have implemented Servo motor control with Arduino.

EX. NO : 12 Date:	IMPLEMENTATION OF LM35 TEMPERATURE SENSOR

AIM:

To implement and monitor LM35 temperature sensor and ultrasonic distance measurement with Arduino.

PROCEDURE:

LM35 TEMPERATURE SENSOR

COMPONENTS REQUIRED

Arduino UNO

You could use any other Arduino board as well. The Keyestudio Uno is used in this example.

USB-A to B Cable

Depends on the Arduino.

Jumper wires (generic)

1 Male to Male Jumper Wire.

Texas Instruments LM35 Temperature Sensor

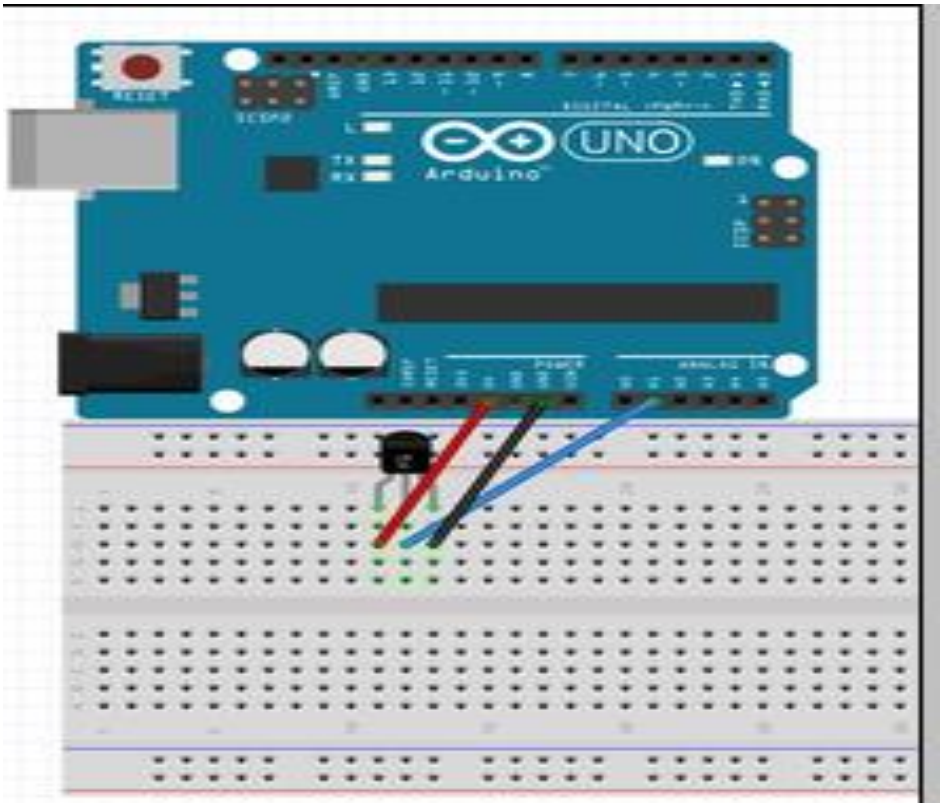
Breadboard (generic)

CODE

```
int val;
int tempPin = 1;
void setup()
{
  Serial.begin(9600);
```

```
}
void loop()
{
  val = analogRead(tempPin);
  float mv = ( val/1024.0)*5000;
  float cel = mv/10;
  float farh = (cel*9)/5 + 32;
  Serial.print("TEMPERATURE = ");
  Serial.print(cel);
  Serial.print("*C");
  Serial.println();
  delay(1000);
Serial.print("TEMPERATURE = ");
Serial.print(farh);
Serial.print("*F");
Serial.println();
}
```

CIRCUIT DIAGRAM



ULTRASONIC DISTANCE MEASUREMENT

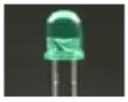
COMPONENTS REQUIRED

Arduino UNO

HC-SR04 Ultrasonic Sensor
Any ultrasonic sensor is fine

Jumper wires (generic)

5 mm LED: Red



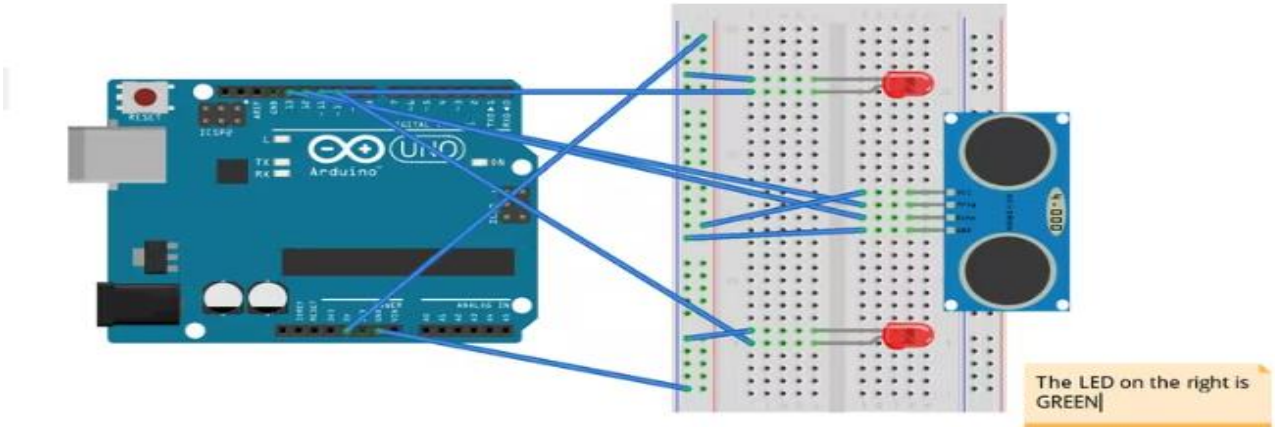
5 mm LED: Green

CODE

```
#define trigPin 13
#define echoPin 12
#define led 11 //red LED
#define led2 10 //green LED

void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(led, OUTPUT);
  pinMode(led2, OUTPUT);
}

void loop() {
  long duration, distance;
  digitalWrite(trigPin, LOW);
```

```
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;
  if (distance < 100) {  //you are free to change this
  digitalWrite(led,HIGH); //distance is less than 100 so red LED turns on
  digitalWrite(led2,LOW);
}
  else {
  digitalWrite(led,LOW); //distance is more than 100 so green LED turns on
  digitalWrite(led2,HIGH);

  }
  if (distance >= 400 || distance <= 0){    //range is 400 cm for HC-SR04 sensor
    Serial.println("Out of range");
  }
  else {
    Serial.print(distance);
    Serial.println(" cm"); //in centimeters
  }
  delay(1000);
}
```

CIRCUIT DIAGRAM



OUTPUT:

RESULT:

Thus we have implemented the code to monitor LM35 temperature sensor and ultrasonic distance measurement with Arduino.

EX. NO : 13 Date:	IMPLEMENTATION OF IR SENSOR WITH ANALOG INPUT

AIM:

To implement IR Sensor with Analog input in Arduino.

PROCEDURE:

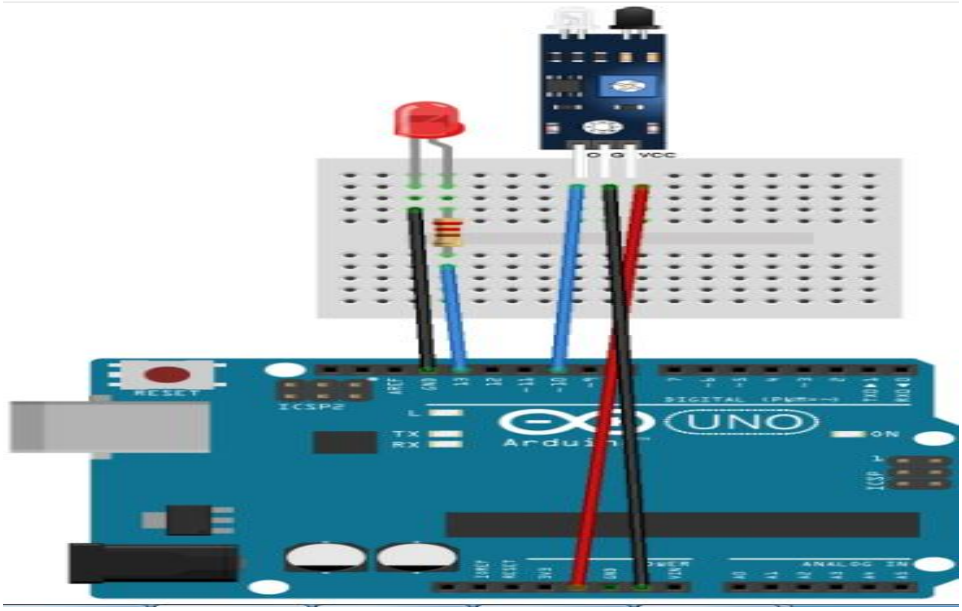
COMPONENTS REQUIRED

CODE:

```
int LEDpin = 13;
int obstaclePin = 10;
int hasObstacle = LOW;  // LOW MEANS NO OBSTACLE

void setup() {
  pinMode(LEDpin, OUTPUT);
  pinMode(obstaclePin, INPUT);
  Serial.begin(9600);
}

void loop() {
  hasObstacle = digitalRead(obstaclePin);

  if (hasObstacle == HIGH) {
    Serial.println("Stop something is ahead!!");
    digitalWrite(LEDpin, HIGH);
  }
```

```
  else
{
    Serial.println("Path is clear");
    digitalWrite(LEDpin, LOW);
  }
  delay(200);


}
```

CIRCUIT DIAGRAM



OUTPUT:

RESULT:

Thus, we have implemented IR Sensor with Analog input in Arduino.

EX. NO : 14 **Date:**	**TEMPERATURE SENSOR MONITORING WITH RASPBERRY PI**

AIM:

To implement cloud reading temperature sensor using Raspberry Pi.

COMPONENTS REQUIRED

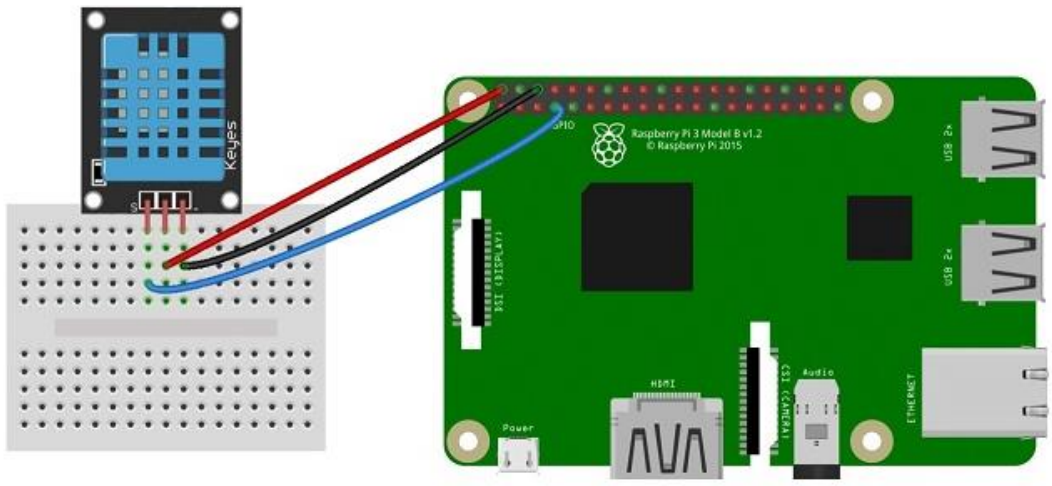
- DHT11 - Temperature and Humidity Sensor Module
- ThingSpeak
- Raspberry Pi
- Putty

CODE

```
#include <ESP8266WiFi.h>
#include "DHT.h"

String apiKey = "";
const char *ssid =  "";
const char *pass =  "";
const char* server = "api.thingspeak.com";

DHT dht(D2, DHT11);
WiFiClient client;

void setup() {
  Serial.begin(115200);
  delay(10);
  dht.begin();
  WiFi.begin(ssid, pass);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
}

void loop() {
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  if (isnan(h) || isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");
```

```
    return;
  }

  if (client.connect(server, 80)) {
    String postStr = apiKey;
    postStr += "&field1=";
    postStr += String(t);
    postStr += "&field2=";
    postStr += String(h);
    postStr += "\r\n\r\n";

    client.print("POST /update HTTP/1.1\n");
    client.print("Host: api.thingspeak.com\n");
    client.print("Connection: close\n");
    client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");
    client.print("Content-Type: application/x-www-form-urlencoded\n");
    client.print("Content-Length: ");
    client.print(postStr.length());
    client.print("\n\n");
    client.print(postStr);

    Serial.print("Temperature: ");
    Serial.print(t);
    Serial.print("\t");
    Serial.print("Humidity: ");
    Serial.println(h);

  }
  client.stop();
  delay(1000);
}
```

CIRCUIT DIAGRAM



OUTPUT:

RESULT:

Thus, we have implemented cloud reading temperature sensor using Raspberry Pi.