# *Fundamentals of Internet of Things (IoT)*

**Piyali Ganguly**
**piyaliganguly1992@gmail.com**

# *Basics of Internet of Things (IoT)*

# Introduction To Internet of Things (IoT)

IoT is a combination of

**Embedded system + Network model + Networking protocol**

# Conception of Various Nodes in IoT

- IoT node or Sensor node or Data Acquisition node
- Fog node
- Gateway device
- Local Server
- Sensor
- Communication module
- Actuator

# How to Create an IoT node

- IoT node selection
- Sensor connectivity
- Communication module connectivity
- coding

# Multi-layer IoT Architecture

The architecture of our proposed system consists of three additional layers, such as[1]:

**Device layer:**

- The first layer is constructed with several IoT nodes.

**Fog layer:**

- fog layer is composed of several smart gateways

**Cloud layer**:

- This layer performs global data analysis.

[1]P.Ganguly, A. Bose, A. Chakrabarti, A. Dasgupta, "Development of a multi fog based water quality monitoring system in IoT platform", IEEE, iSeS, 2018, Hyderabad, India

# Basic operations in IoT

- Data Acquisition
- Data Communication
- Data processing and decision making

# *Basics of Arduino (IoT node)*

# Basics of Arduino

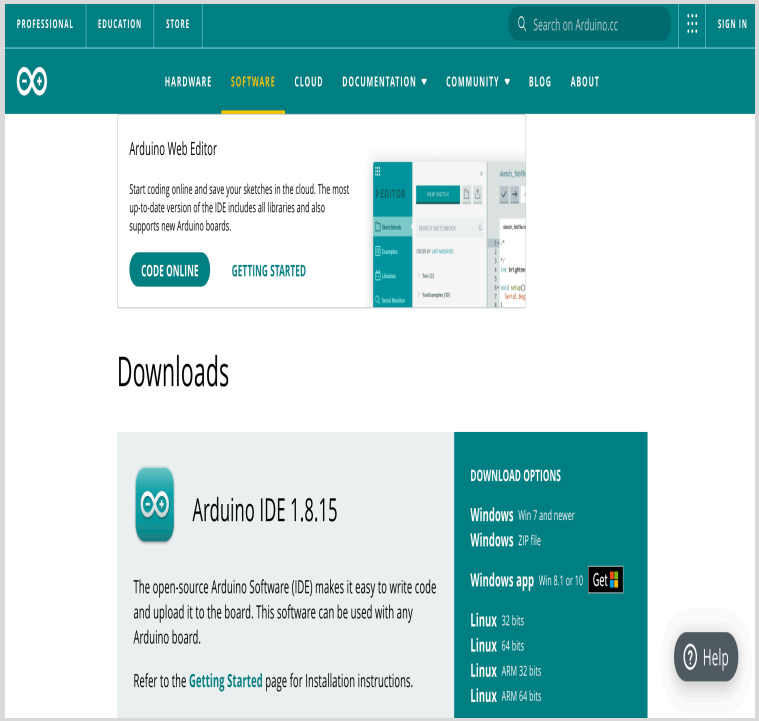
- Arduino is an open-source platform used for building electronics projects.
- Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment).
- IDE runs on computer, used to write and upload computer code to the physical board.

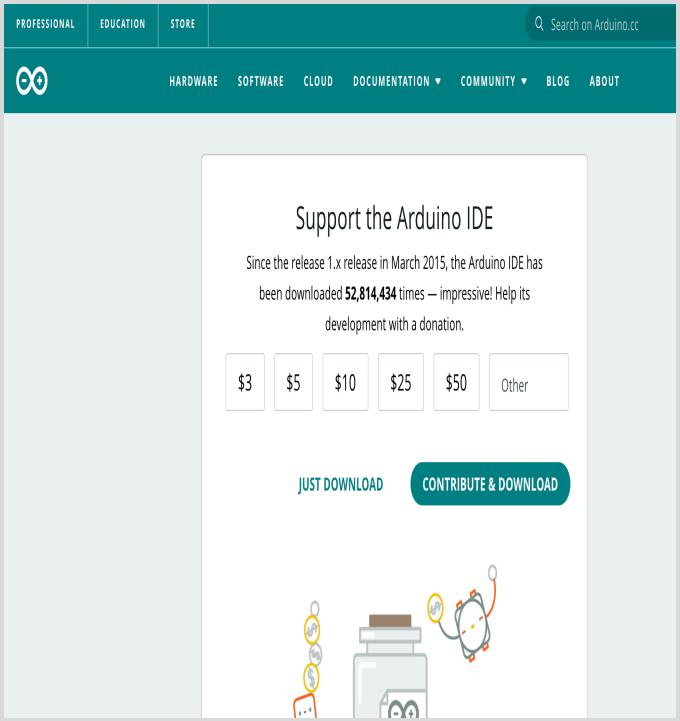
# Basics of Arduino

- ATMEGA 328 P microcontroller
- Digital port
- Analog port
- 5v
- Gnd
- 3v
- Tx, Rx

# Arduino IDE installation[1]

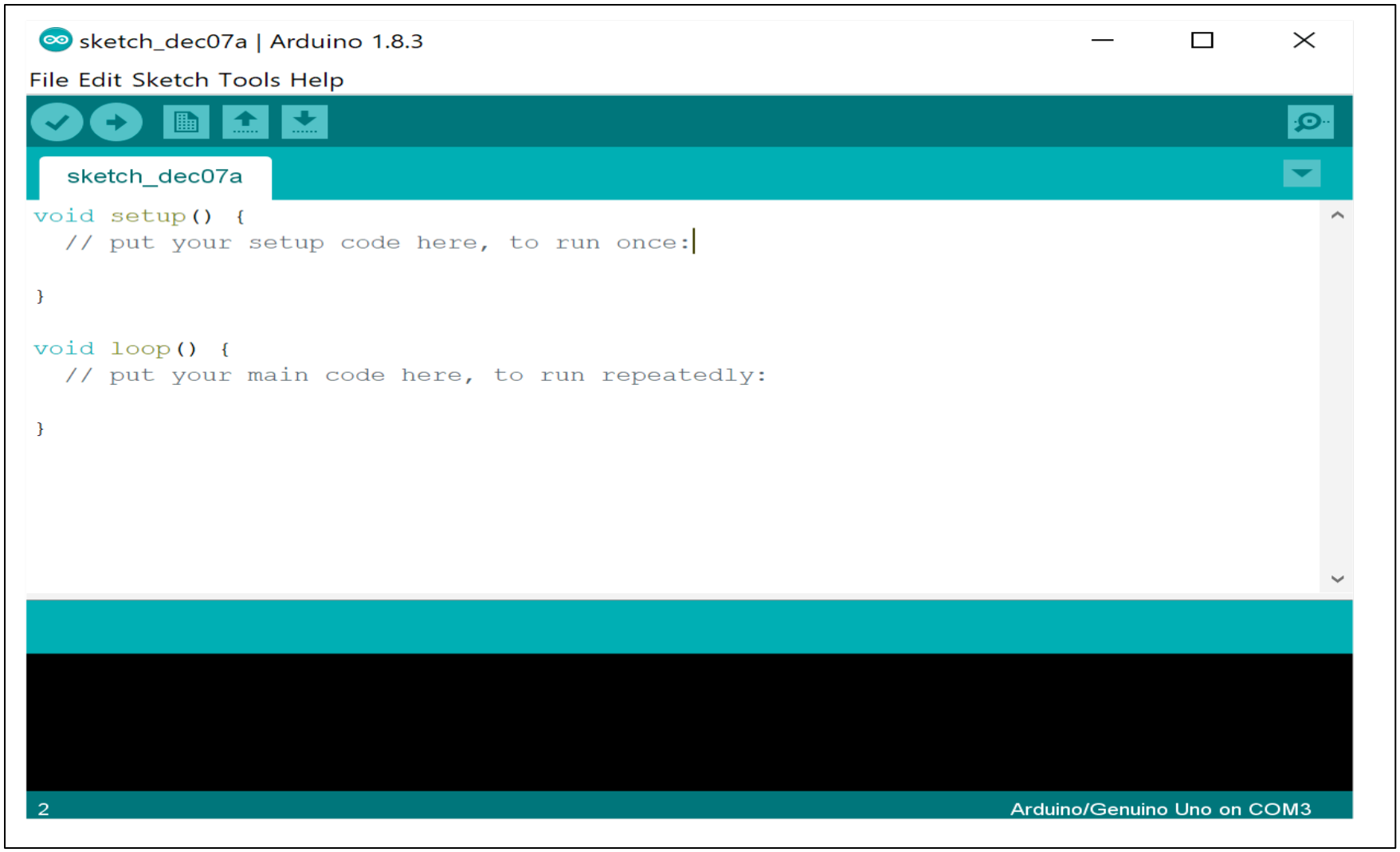
- **Link: https://www.arduino.cc/en/software**




[1]**https://www.arduino.cc/en/software**

# Arduino IDE[1]



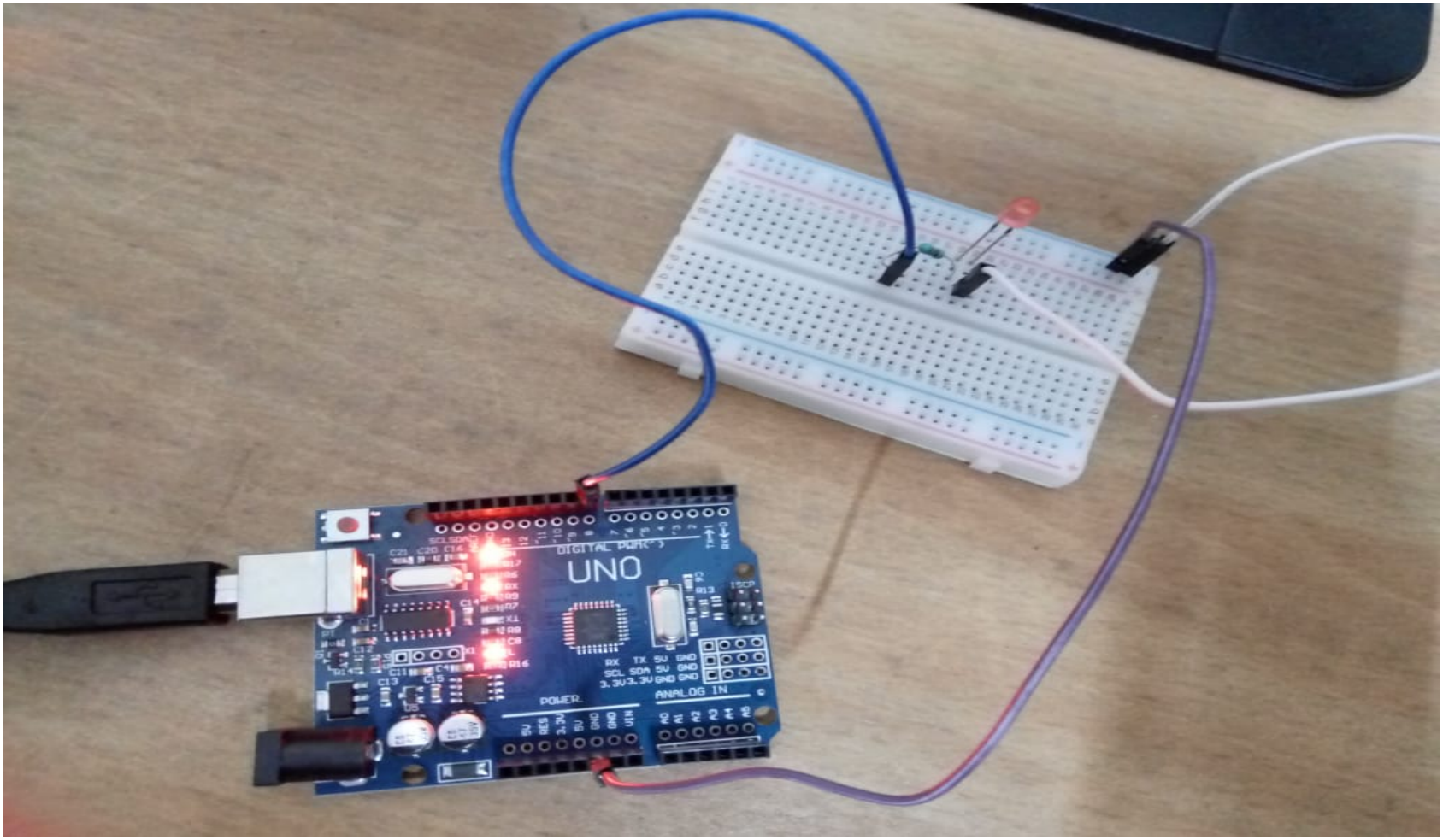

[1]www.arduino.cc

# Arduino IDE[1]

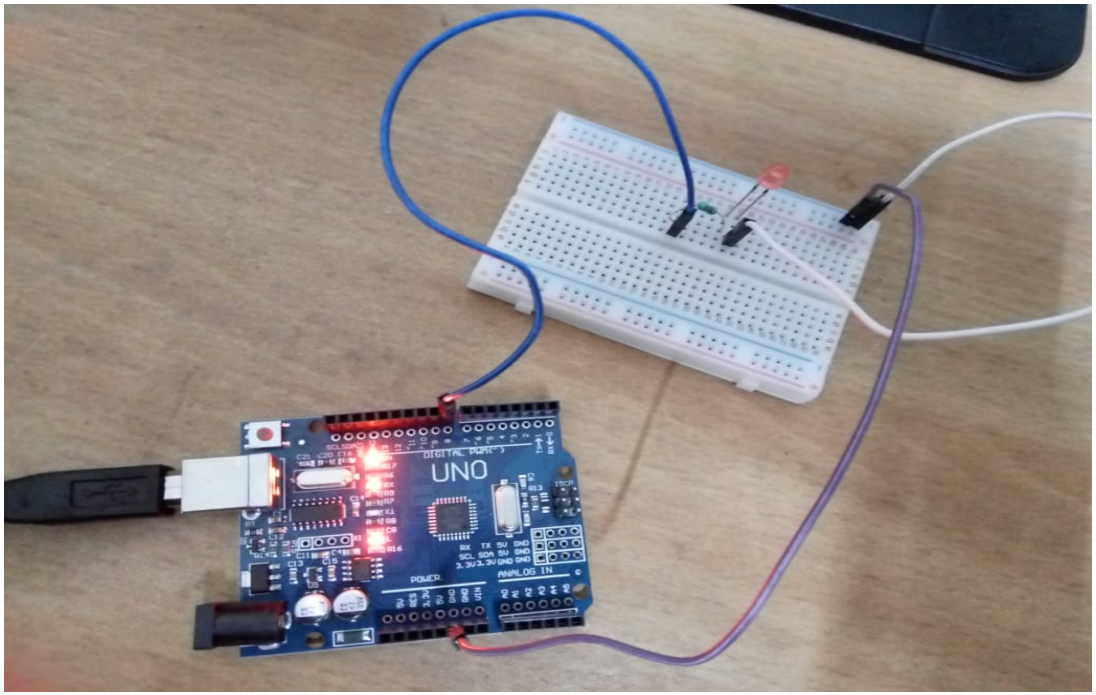
```
Void setup()
{
        Serial.begin(9600);
}
Void loop ()
{
        …….Code …….
}
```

[1]www.arduino.cc

# LED blinking Arduino

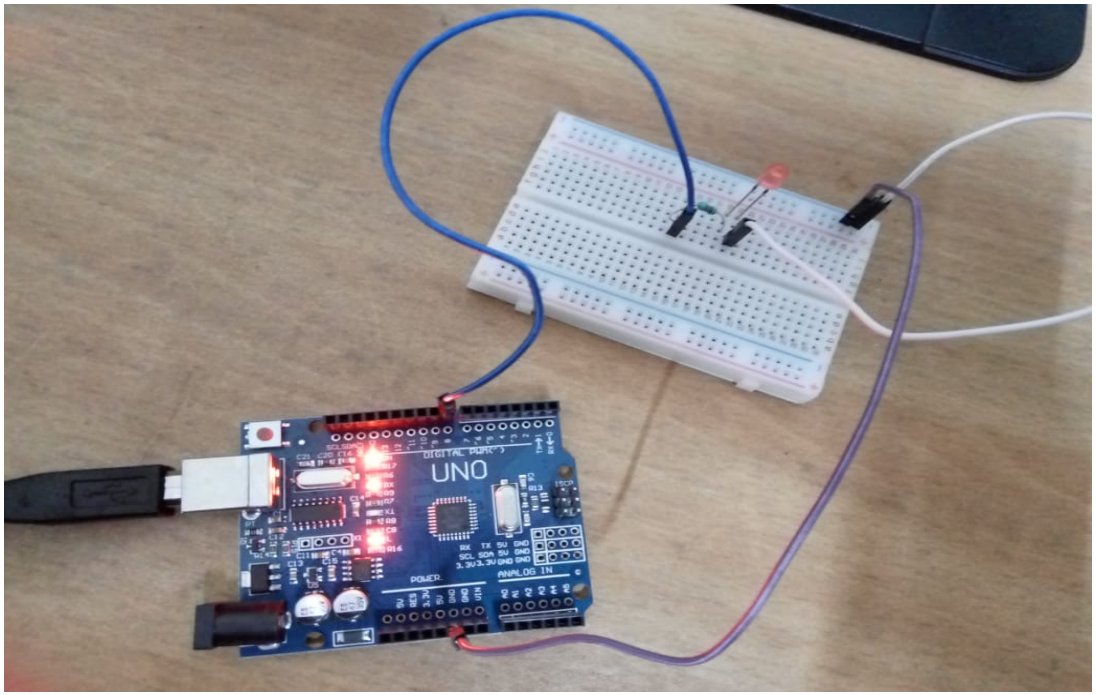


# LED blinking Arduino




| Requirements |
| --- |
| Arduino |
| Cable |
| LED |
| Bread board |
| Resistor 1 K ohm |
| Jumper wire (Male female) |

# LED blinking Arduino




| Requirements |
| --- |
| Arduino |
| Cable |
| LED |
| Bread board |
| Resistor 1 K ohm |
| Jumper wire (Male female) |

LED’s shorter leg ----→ Ground
LED’s positive leg ----→ Digital port

# LED blinking Arduino Code

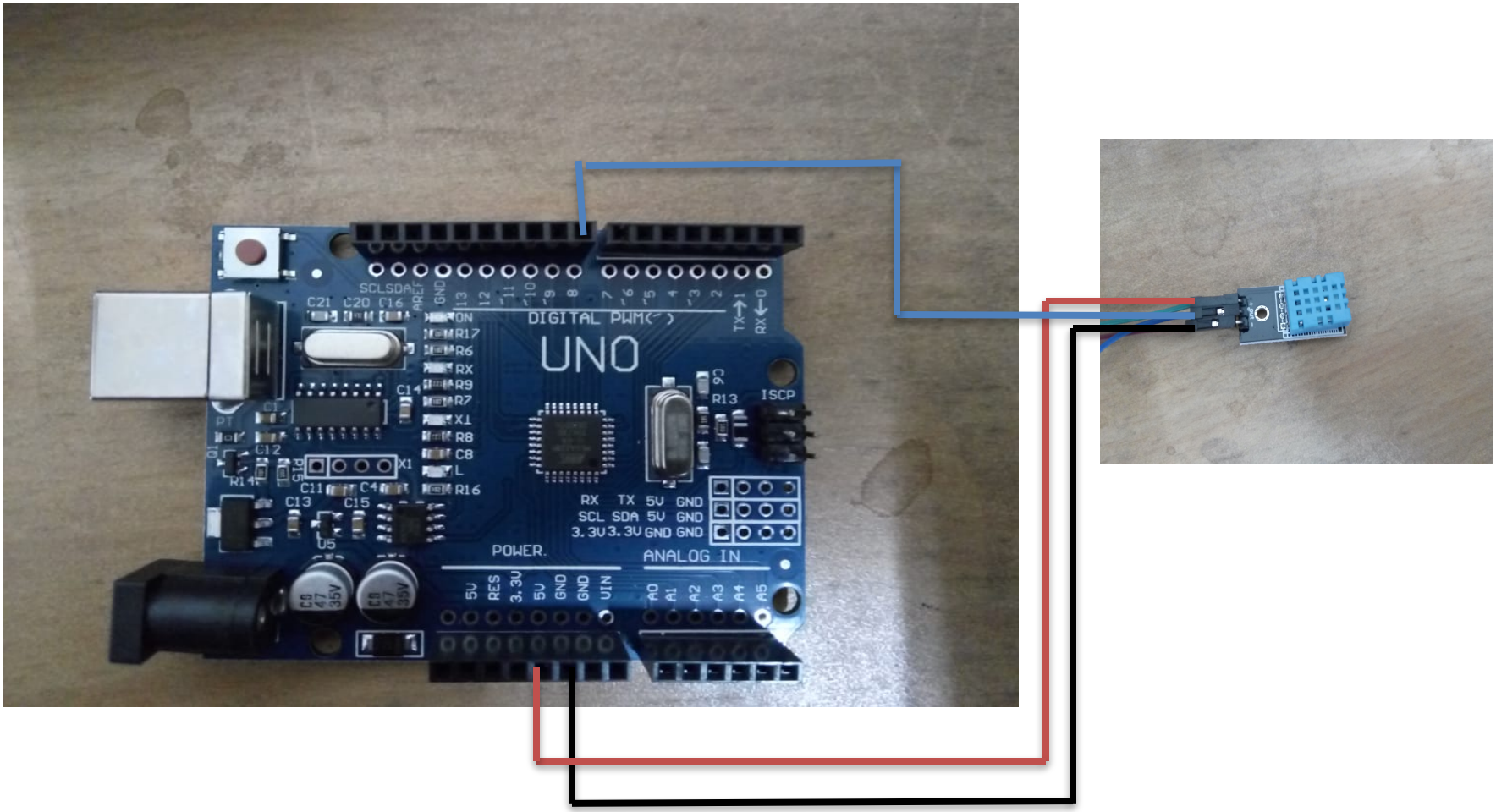
```
void setup()
{
   pinMode(8, OUTPUT);

  pinMode(7, OUTPUT);

  Serial.begin(9600);
}
```

# LED blinking Arduino Code

```
void loop()
{
  digitalWrite(8, HIGH);
  delay(1000); // wait for a second
  digitalWrite(8, LOW);
  delay(1000);
  digitalWrite(7, HIGH);
   delay(1000);  // wait for a second
  digitalWrite(7, LOW);
   delay(1000);
  }
```

# *Sensor interfacing with Arduino*

# Temperature and Humidity Data Acquisition using Arduino (Circuit Diagram)



# Temperature and Humidity Data Acquisition using Arduino

| Requirements |
| --- |
| Arduino |
| Cable |
| DHT11 or DHT22 |
| Jumper wire (Male female) |
| Installation of DHT library in Arduino (https://github.com/adafruit/DHT-sensor-library) |

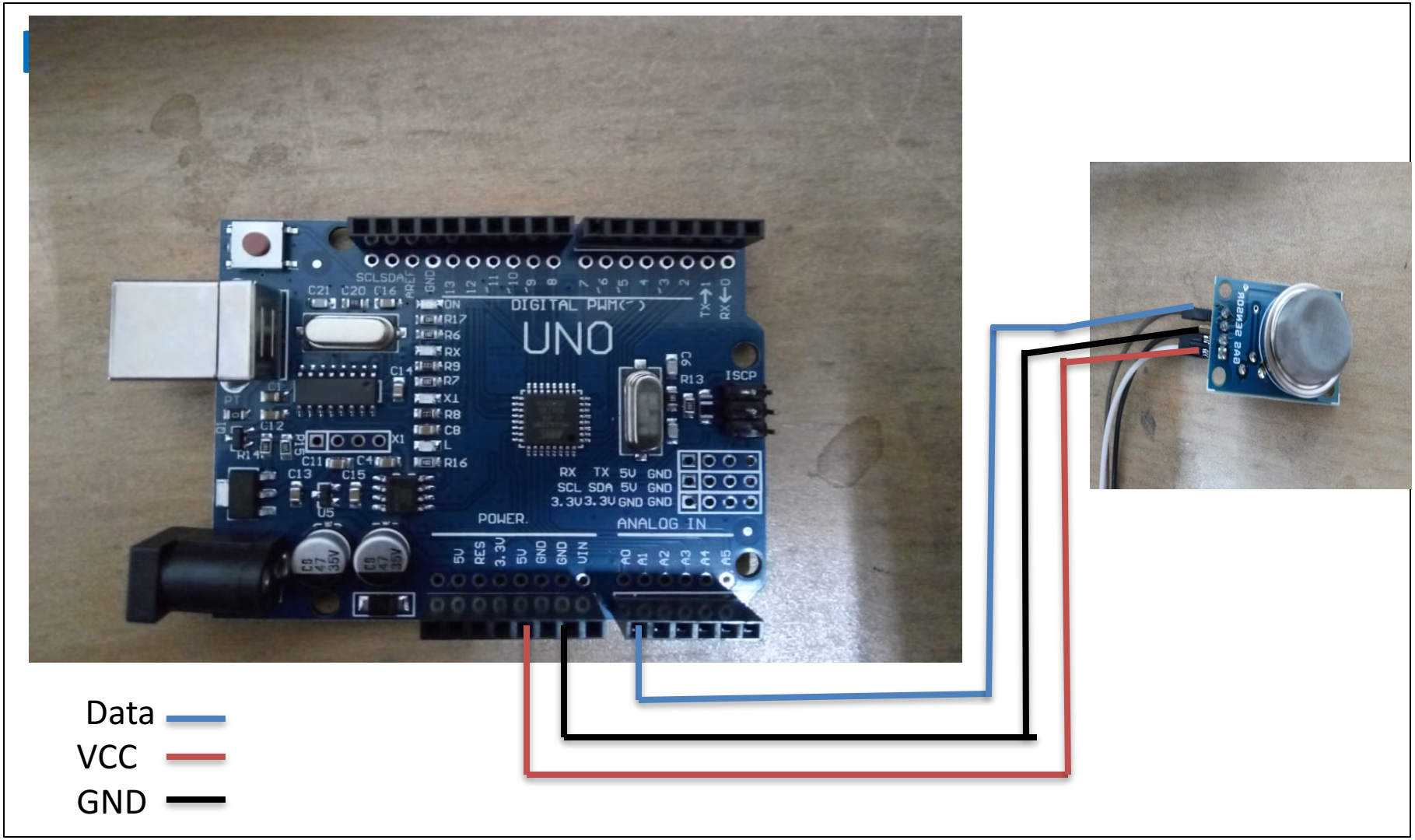
# Temperature and Humidity Data Acquisition using Arduino

```
##Import Header files
#include <dht.h>

# DHT's Data Pin is connected with Digital Pin 8
#define dht_apin 8
```

# Temperature and Humidity Data Acquisition using Arduino

```
# Data Pin Checking
float chk = DHT.read11(DHT11_PIN ) ;
#Temperature Data Acquisition
temp = DHT.temperature;

#Humidity Data Acquisition
humid = DHT.humidity;

# Data Printing in Serial Monitor
Serial.print("temperature: ");
Serial.print(temp);
Serial.print("humidity: ");
Serial.print(humid);
Serial.println(" ");
```

# Gas Sensor (MQ135) Data Acquisition using Arduino (Circuit Diagram)



# Gas Sensor (MQ135) Data Acquisition using Arduino

| Requirements |
| --- |
| Arduino |
| Cable |
| MQ135 |
| Jumper wire (Male female) |
| Installation of MQ135 library in Arduino (https://www.arduino.cc/reference/en/libraries/mqunifiedsensor/) |

# Gas Sensor (MQ135) Data Acquisition using Arduino (Code)

```
#include "MQ135.h"
Float gas_data;

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

void loop ( )
{
MQ135 gasSensor = MQ135(A0);
gas_data = gasSensor.getRZero();
 delay ( 1000 ) ;
}
```

*Fog Node (Raspberry pi)*

# Raspberry pi 4[1]


[1] Picture taken from https://en.wikipedia.org/wiki/Raspberry_Pi

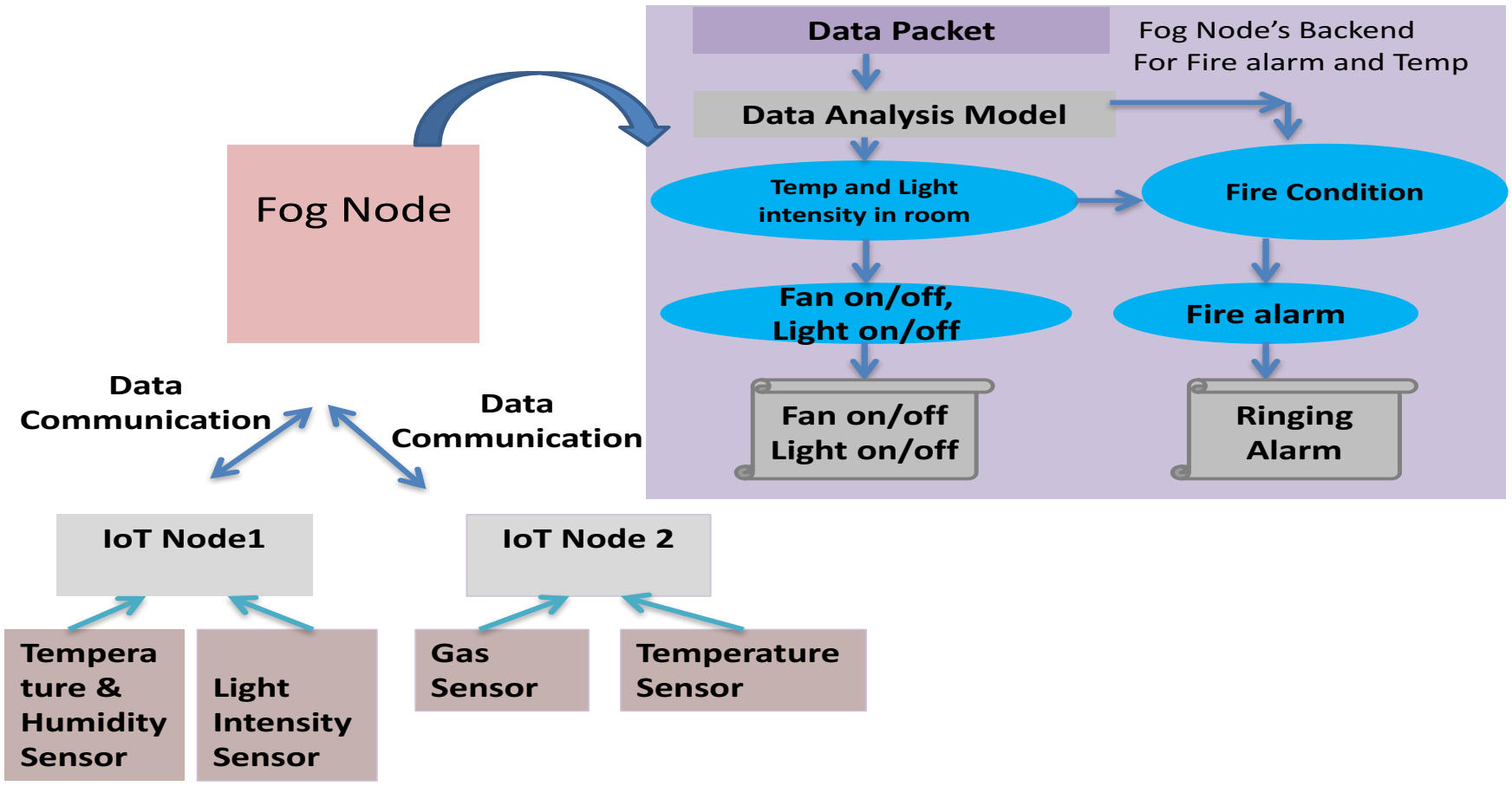
# *IoT Simulation Platform using IfogSim Simulator*

# Introduction to IFogSim[1]

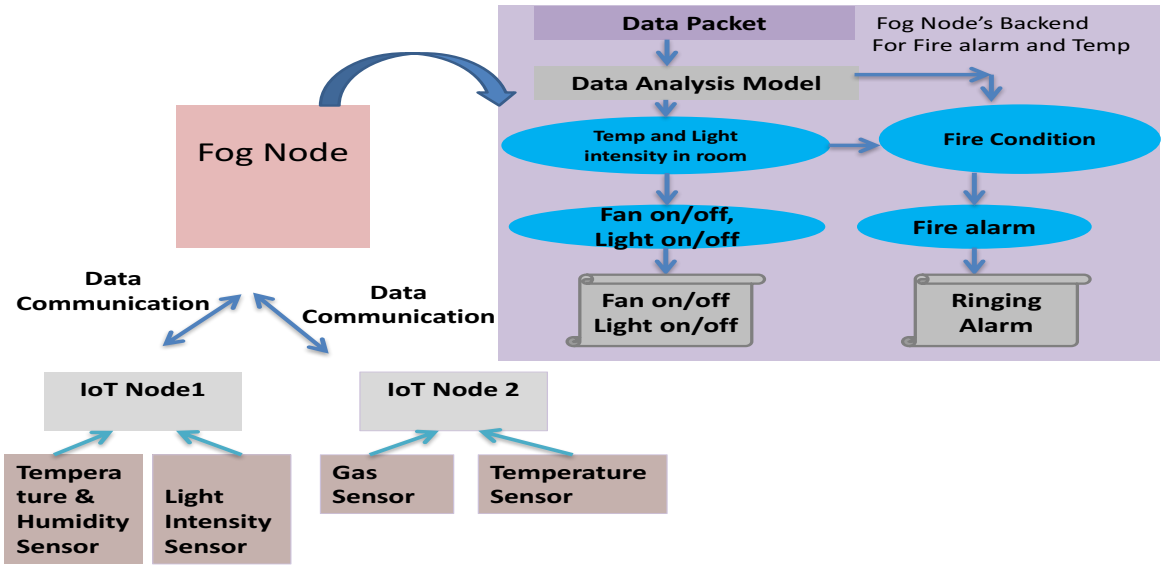
- Java based
- Runs under Eclipse
- Extension of Cloudsim
- Resource Estimation

[1]Redowan Mahmud and Rajkumar Buyya. Modelling and simulation of fog and edge computing environments using ifogsim toolkit. *Fog and edge computing: Principles and paradigms*, pages 1–35, 2019.

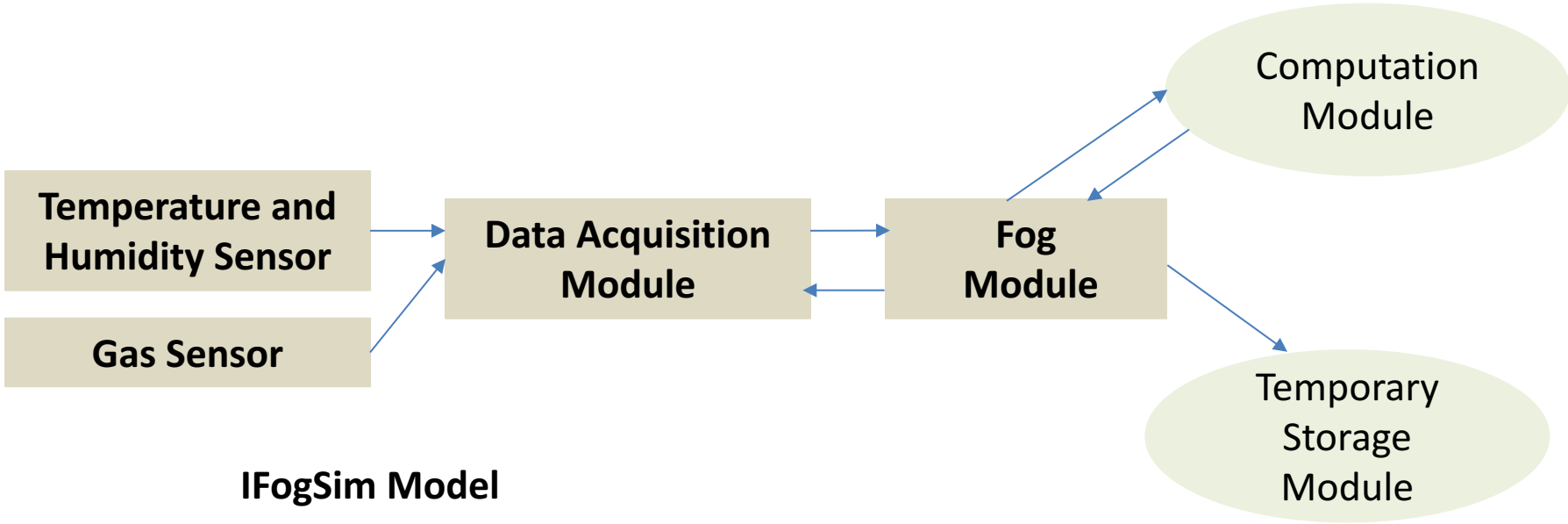
# Introduction to IFogSim (Smart room Feature)



# Introduction to IFogSim Model Develoment




System architecture




IFogSim Model

# Introduction to IFogSim[1] (IoT node Creation)

IoT Node creation:

FogDevice **IoTnode** = *createFogDevice*("I-"+id, 5000, 4000, 10000, 10000, 3, 0.0, 107.339, 83.4333);

**FogDevice IoTnode = *createFogDevice*("I-"+id, MIPS, RAM, Uplink bandwidth, downlink, hierarchy, rate MIPS, Busy power, Idle power);**

- MIPS (Million Instructions Per Sec- ond),
- RAM (main memory of the fog node),
- uplink bandwidth, downlink bandwidth,
- level (hierarchy level of the device),
- ratePerMips (cost rate per MIPS used),
- busyPower (the amount of power consumed when the fog node is in busy state)
- idlePower (the amount of power used when the fog node is in the idle state).

[1] Soumya K Ghosh Harshit Gupta, Amir Vahid Dastjerdi and Rajkumar Buyya. ifogsim: A toolkit for modeling and simulation of resource man- agement techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.

# Introduction to IFogSim[1]

Sensor creation:

FogDevice **tempsensor** = *createFogDevice*("a-"+id, 5000, 4000, 10000, 10000, 3, 0.0, 107.339, 83.4333);

**FogDevice tempsensor = *createFogDevice*("a-"+id, MIPS, RAM, Uplink bandwidth, downlink, hierarchy, rate MIPS, Busy power, Idle power);**

- MIPS (Million Instructions Per Second),
- RAM (main memory of the fog node),
- uplink bandwidth, downlink bandwidth,
- level (hierarchy level of the device),
- ratePerMips (cost rate per MIPS used),
- busyPower (the amount of power consumed when the fog node is in busy state)
- idlePower (the amount of power used when the fog node is in the idle state).

[1] Soumya K Ghosh Harshit Gupta, Amir Vahid Dastjerdi and Rajkumar Buyya. ifogsim: A toolkit for modeling and simulation of resource man- agement techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.

# Introduction to IFogSim[1]

Sensor Addition in IoT node:

```
for(int i=0;i<numOfGasSensorsPerArea;i++)
{
addGasSensors(i+"", userId, appId, IoTnode.getId());
}

for(int i=0;i<numOfTempsensorPerArea;i++)
{
addtempSensors(i+"", userId, appId, IoTnode.getId());
}
for(int i=0;i<numOflightSensorsPerArea;i++)
{
addlightSensors(i+"", userId, appId, IoTnode.getId());
}
```

[1] Soumya K Ghosh Harshit Gupta, Amir Vahid Dastjerdi and Rajkumar Buyya. ifogsim: A toolkit for modeling and simulation of resource man- agement techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.

# Introduction to IFogSim[1,2]

## Fog Node creation:

FogDevice **fognode** = *createFogDevice*("a-"+id, 5000, 4000, 10000, 10000, 3, 0.0, 107.339, 83.4333);

**FogDevice fognode = *createFogDevice*("a-"+id, MIPS, RAM, Uplink bandwidth, downlink, hierarchy, rate MIPS, Busy power, Idle power);**

- MIPS (Million Instructions Per Sec- ond),
- RAM (main memory of the fog node),
- uplink bandwidth, downlink bandwidth,
- level (hierarchy level of the device),
- ratePerMips (cost rate per MIPS used),
- busyPower (the amount of power consumed when the fog node is in busy state)
- idlePower (the amount of power used when the fog node is in the idle state).

[1] Soumya K Ghosh Harshit Gupta, Amir Vahid Dastjerdi and Rajkumar Buyya. ifogsim: A toolkit for modeling and simulation of resource man- agement techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.

[2] Redowan Mahmud and Rajkumar Buyya. Modelling and simulation of fog and edge computing environments using ifogsim toolkit. *Fog and edge computing: Principles and paradigms*, pages 1–35, 2019.

***Thank You***