

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 as¹:

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.

¹P.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 installation¹

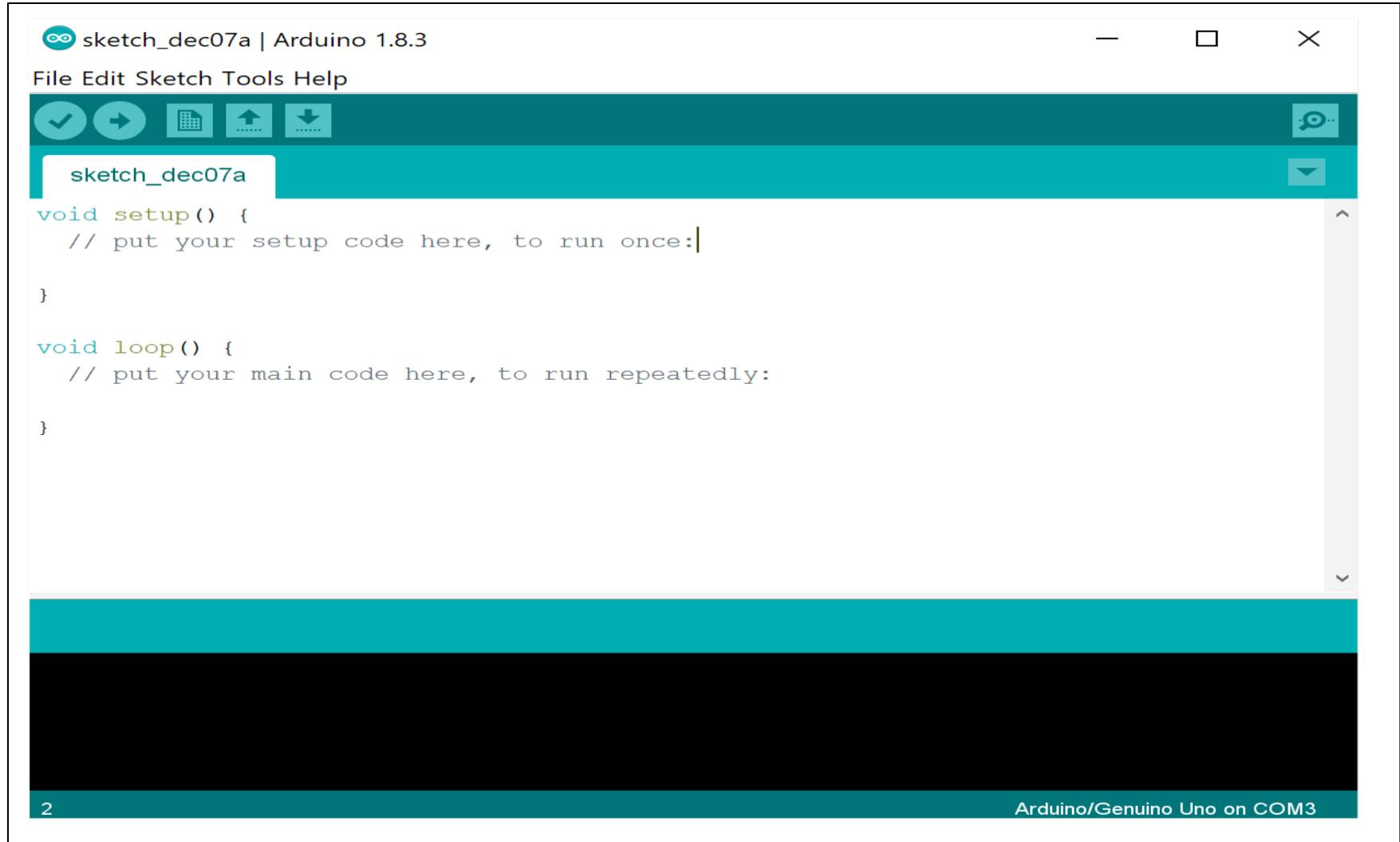
- Link: <https://www.arduino.cc/en/software>

The screenshot shows the Arduino website's software page. At the top, there are navigation links for PROFESSIONAL, EDUCATION, and STORE, along with a search bar and a sign-in button. Below the header, there are links for HARDWARE, SOFTWARE, CLOUD, DOCUMENTATION, COMMUNITY, BLOG, and ABOUT. A prominent feature is the "Arduino Web Editor" section, which allows users to code online and save sketches in the cloud. It includes a preview window showing a sketch with pins and variables like "led" and "brightness". Below this, there are two buttons: "CODE ONLINE" and "GETTING STARTED". The main content area is titled "Downloads" and features a large image of the Arduino IDE 1.8.15 icon. To the right of the icon, there is a "DOWNLOAD OPTIONS" section with links for Windows (Win 7 and newer), Windows ZIP file, Windows app (Get Windows app), Linux (32 bits, 64 bits), Linux ARM (32 bits, 64 bits), and a "Help" button.

The screenshot shows a "Support the Arduino IDE" section on the Arduino website. It features a message about the IDE's success since its March 2015 release, stating it has been downloaded 52,814,434 times. It encourages users to support development with a donation through various amounts: \$3, \$5, \$10, \$25, \$50, and "Other". Below this, there are "JUST DOWNLOAD" and "CONTRIBUTE & DOWNLOAD" buttons. To the right, there is a small illustration of a circuit board and some components.

¹<https://www.arduino.cc/en/software>

Arduino IDE¹



The screenshot shows the Arduino IDE interface. The title bar reads "sketch_dec07a | Arduino 1.8.3". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has icons for save, upload, and download. The code editor contains the following code:

```
sketch_dec07a

void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

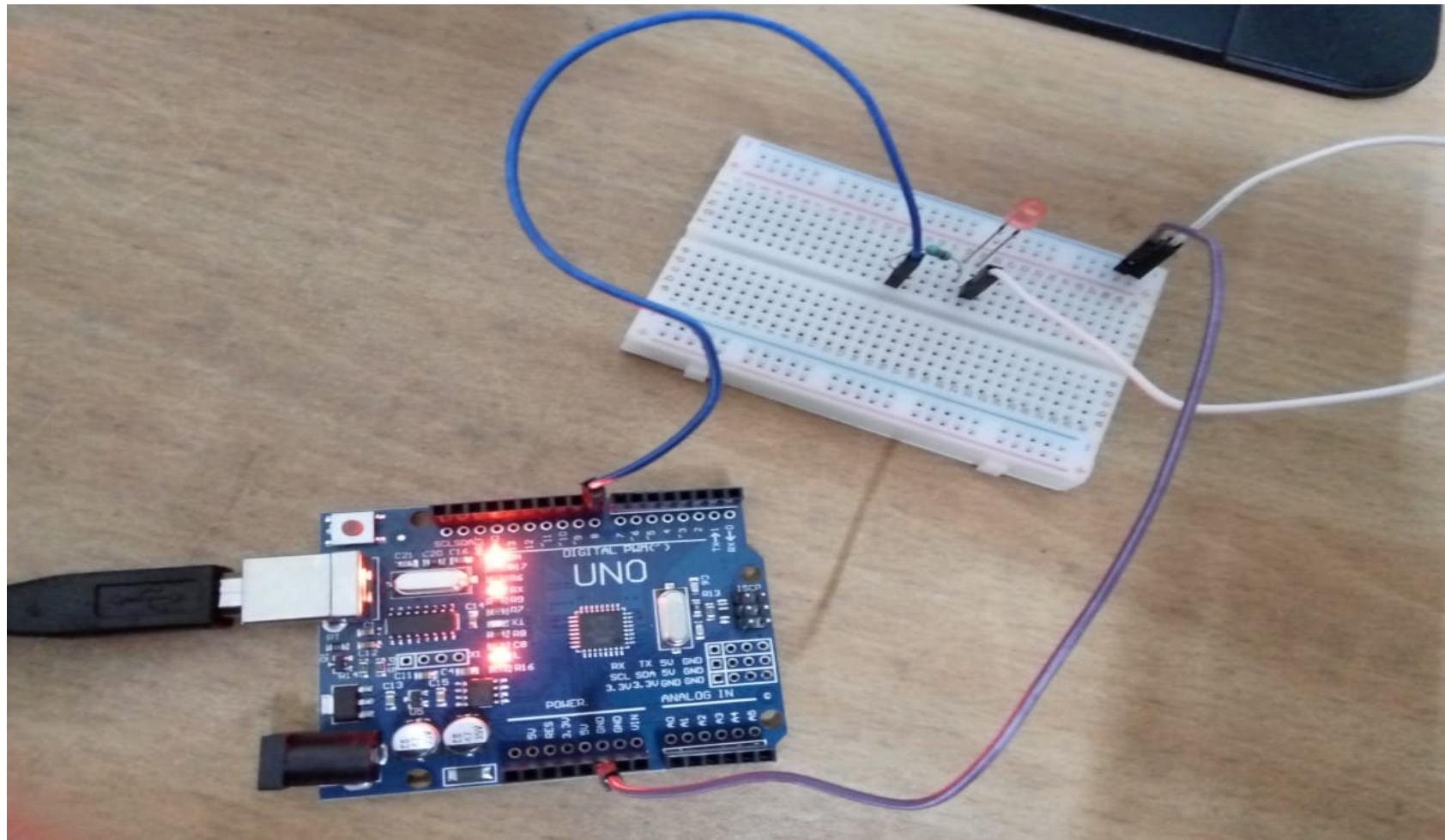
The status bar at the bottom indicates "Arduino/Genuino Uno on COM3" and the number "2".

Arduino IDE¹

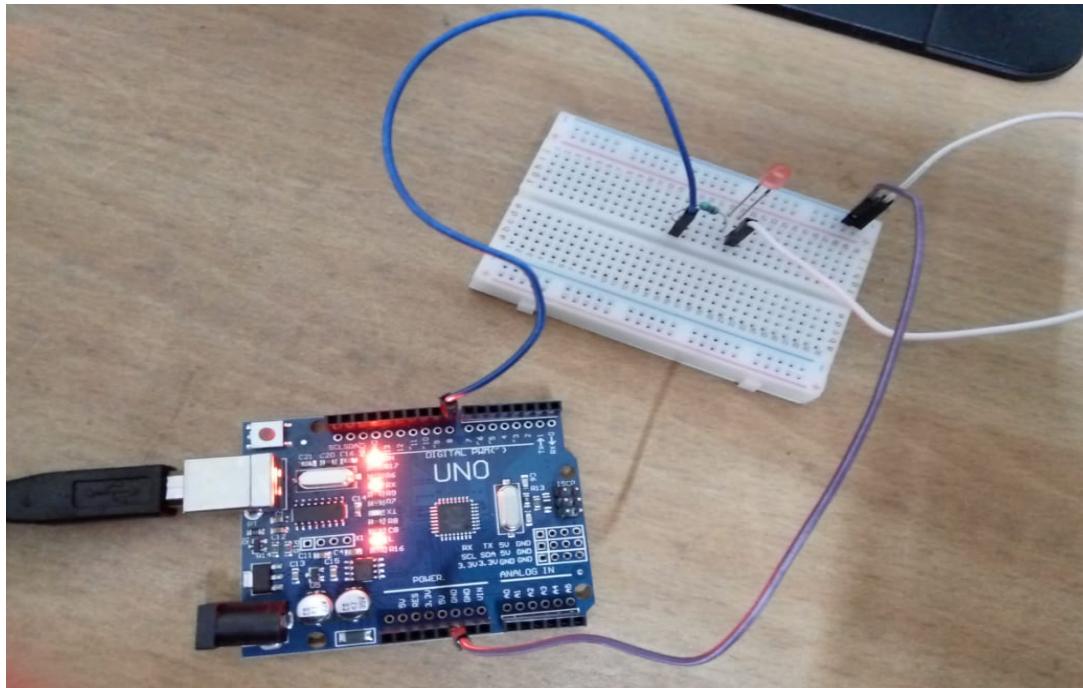
```
Void setup()  
{  
    Serial.begin(9600);  
}  
  
Void loop ()  
{  
    .....Code .....  
}
```

¹www.arduino.cc

LED blinking Arduino



LED blinking Arduino



Requirements

Arduino

Cable

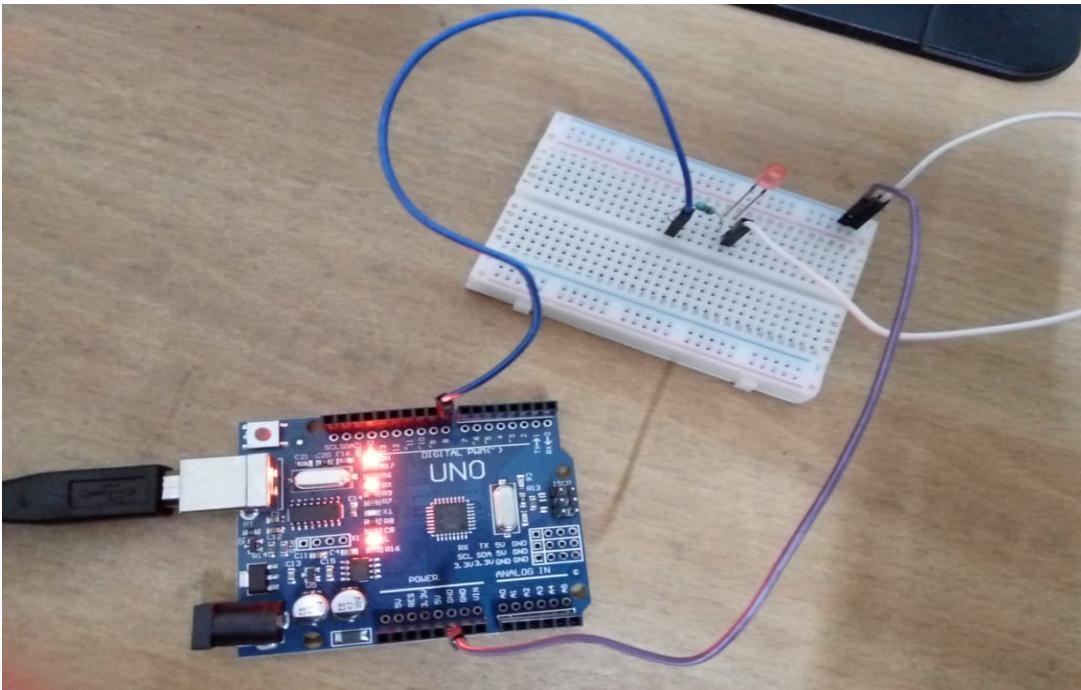
LED

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);

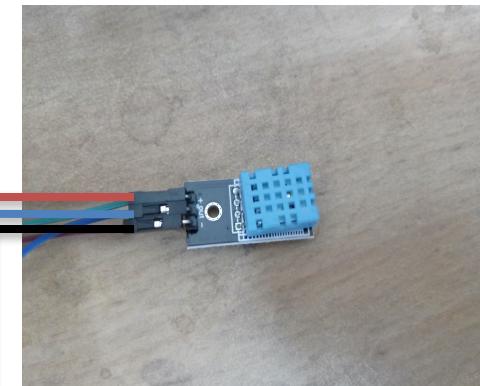
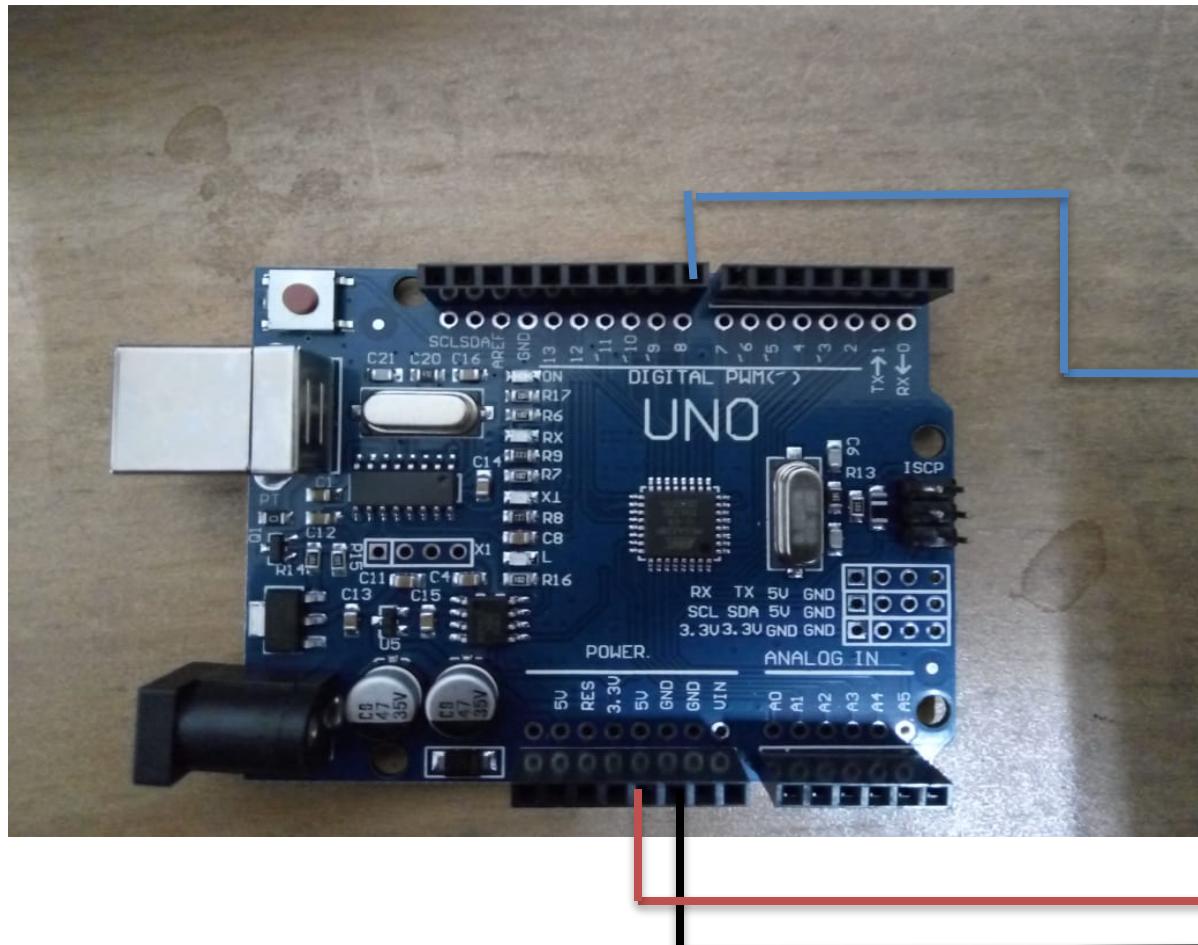
    Serial.begin(9600);
}
```

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: ");
```

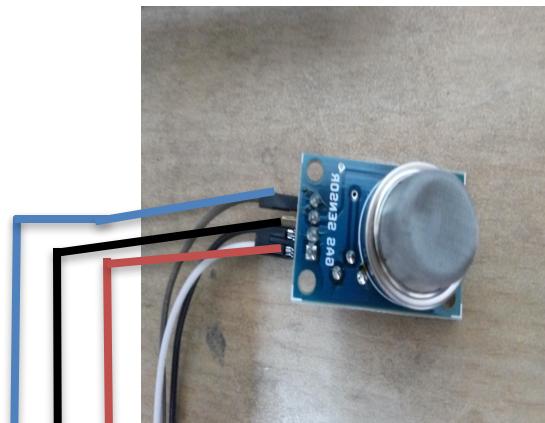
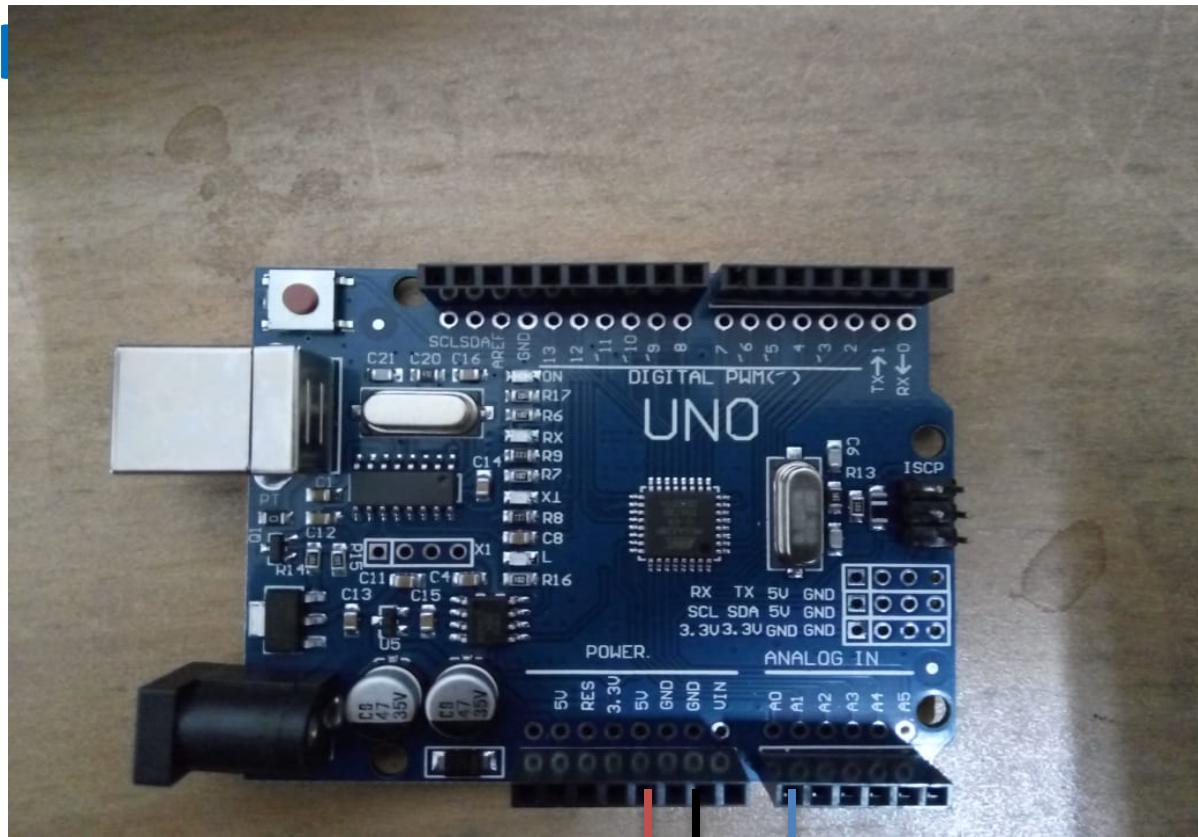
```
Serial.print(temp);
```

```
Serial.print("humidity: ");
```

```
Serial.print(humid);
```

```
Serial.println(" ");
```

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



Data —————
VCC —————
GND —————



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)

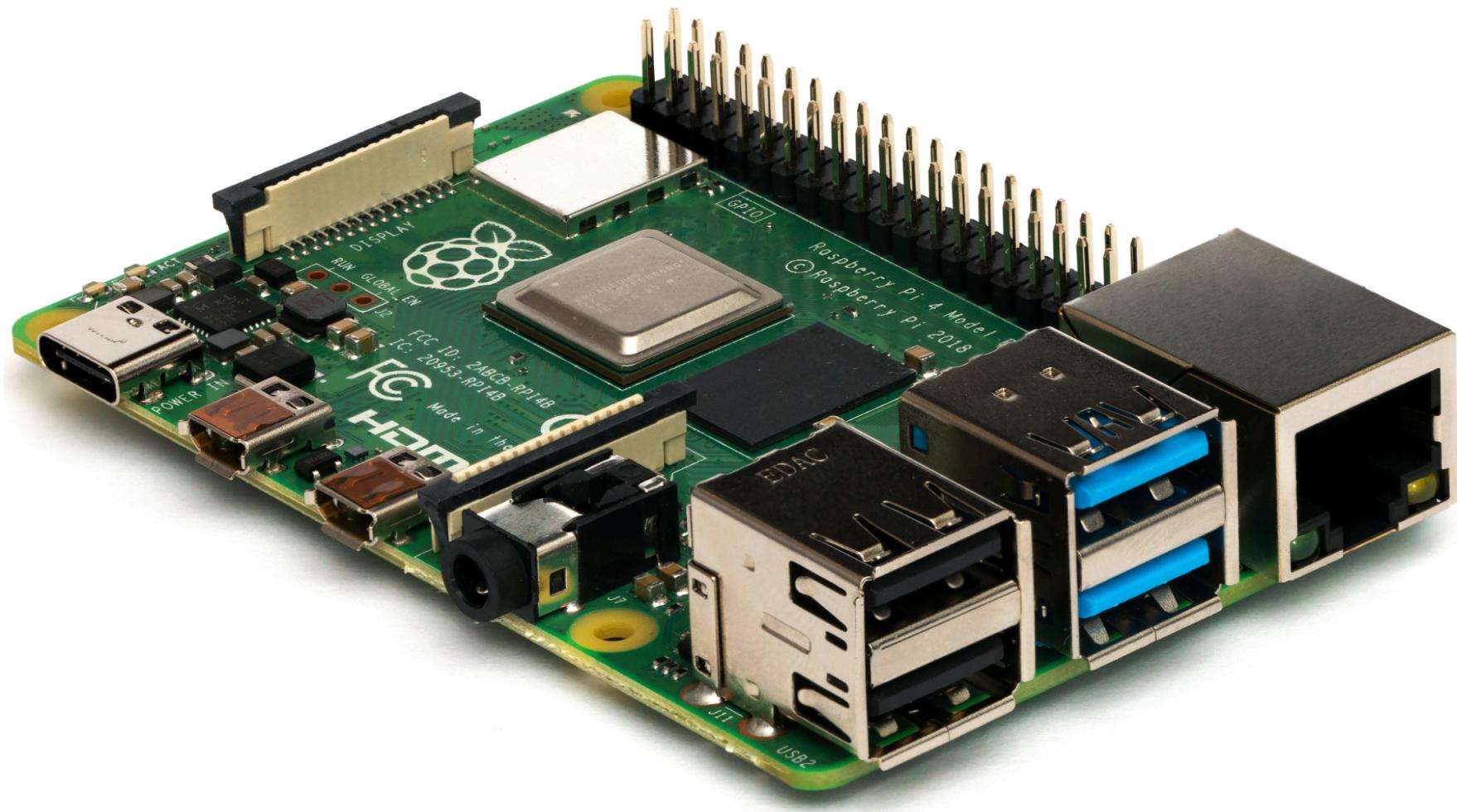
```
#include "MQ135.h"
Float gas_data;

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

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

Fog Node (Raspberry pi)

Raspberry pi 4¹



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

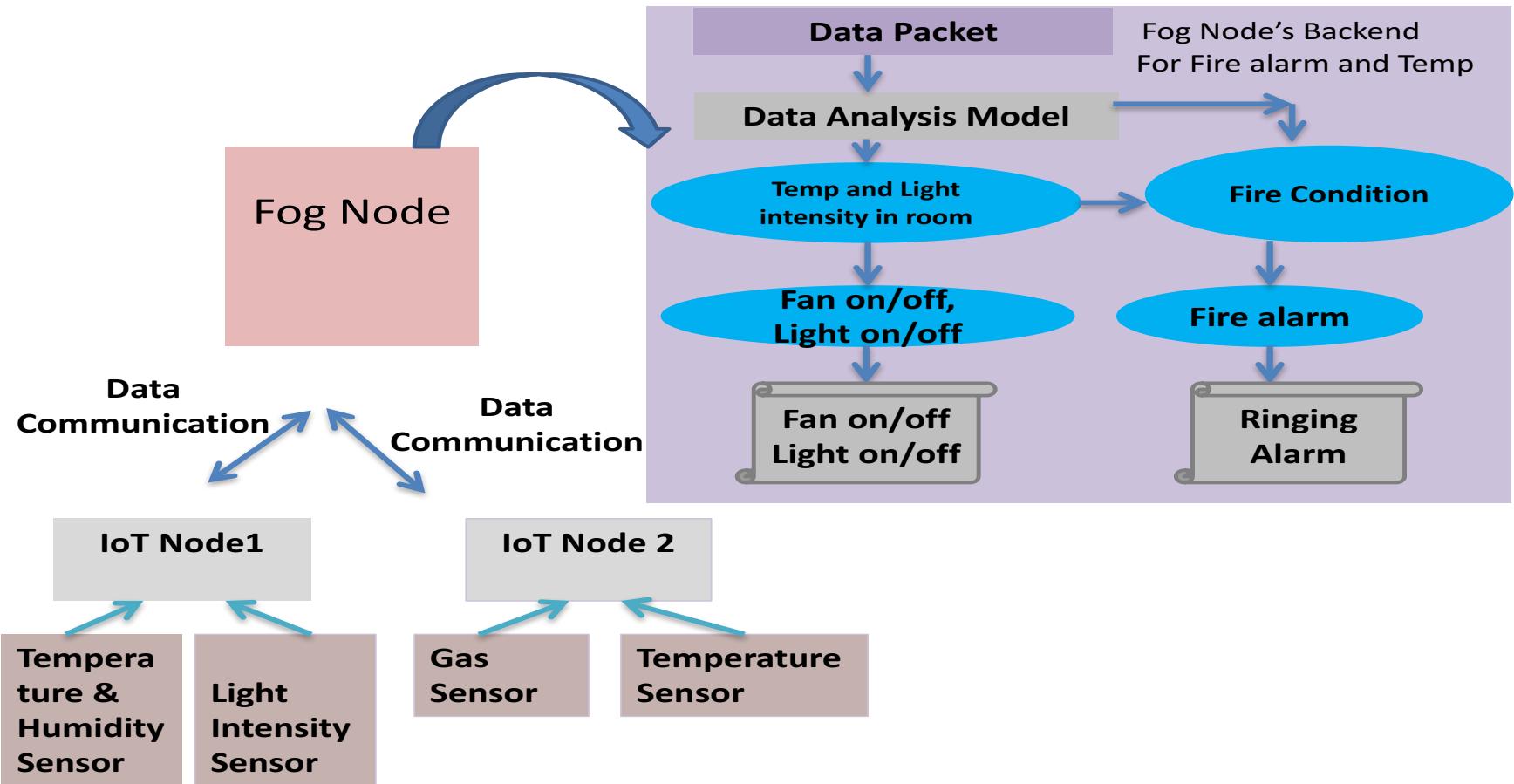
IoT Simulation Platform using IfogSim Simulator

Introduction to IFogSim¹

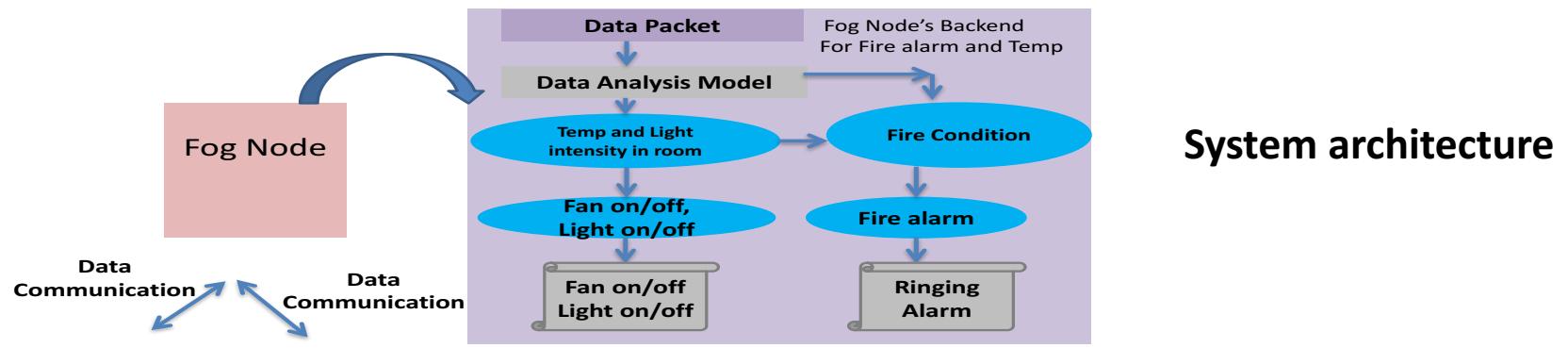
- Java based
- Runs under Eclipse
- Extension of Cloudsim
- Resource Estimation

¹Redowan 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.

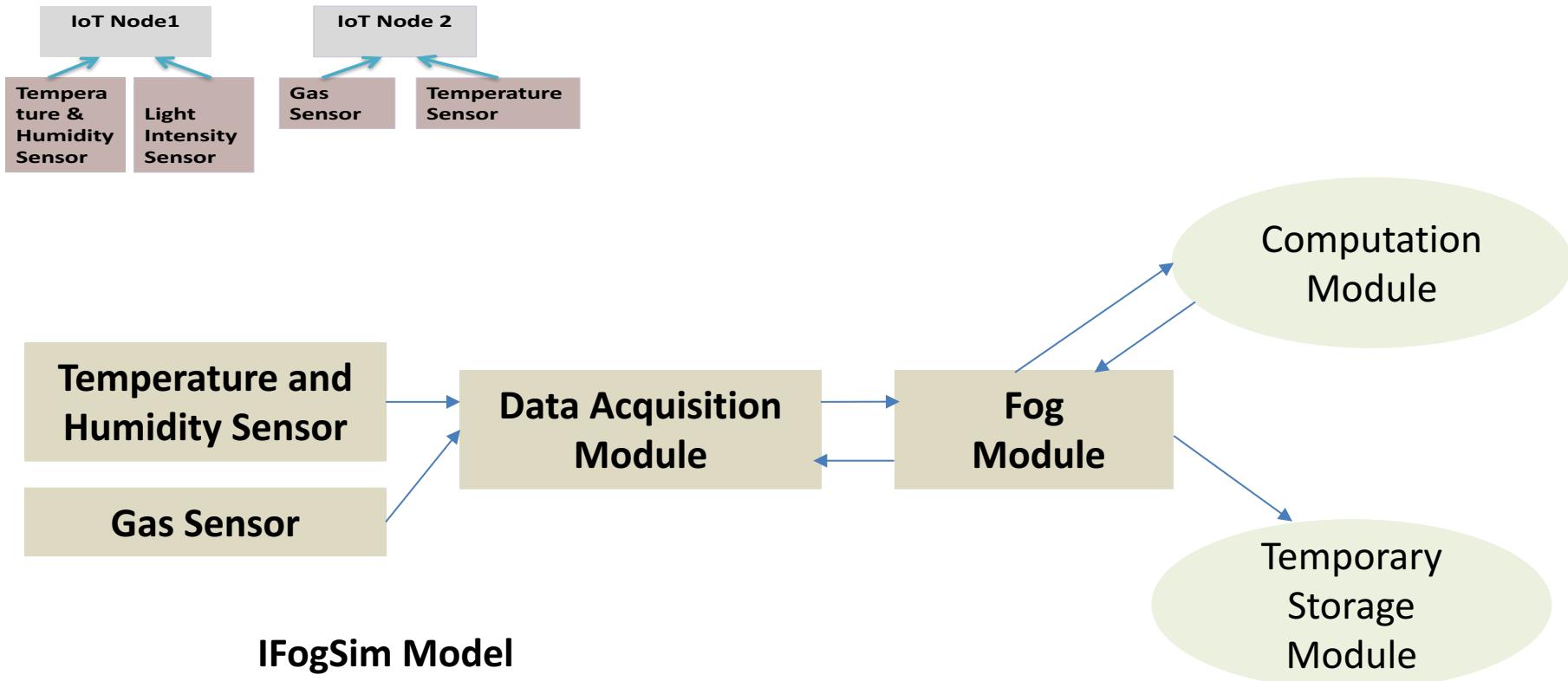
Introduction to IFogSim (Smart room Feature)



Introduction to IFogSim Model Development



System architecture



Introduction to IFogSim¹ (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 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 management techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.

Introduction to IFogSim¹

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 management techniques in the internet of things, edge and fog computing environments. *Software: Practice and Experience*, 47(9):1275–1296, 2017.

Introduction to IFogSim¹

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 management 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 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).

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

²Redowan 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