Green University of Bangladesh
Department of Computer Science and Engineering (CSE)
Faculty of Sciences and Engineering
Semester: (Fall, Year: 2021), B.Sc. in CSE (Day)

Course Title: Computer Network Lab
Course Code: CSE-312 Section: DB

Lab Project Name: Smart Office System (IOT) by Cisco Packet Tracer

Student Details

	Name	ID
1.	Md. Torikul Islam	192002131

Submission Date : 05 January 2022
Course Teacher's Name : Mr. Palash Roy

[For Teachers use only: Don't Write Anything inside this box]

Lab Project Status

Marks: …… Signature: .....

Comments: .....Date: .....

Table of Contents

Chapter 1 Introduction **3**
1.1 Introduction 3
1.2 DesignGoals/Objective 4

Chapter 2 Design/Development/Implementation of the Project **5**
2 Flow Chart of the Network 5
2.1 ISP Network Configuration (Cluster) 6
2.2 Configure Smart Office Server 6
2.3 Configure Cable Modem 7
2.4 Configure Office Main Wireless Router 8
2.5 Smart Door Configuration 10
2.6 Smart Window Configuration 16
2.7 Smart Fan Configuration 19
2.8 Smart Garage Configuration 22
2.9 Fire Sprinkler Configuration 24
2.10 Trip Senson (Theif Sensor) Configuration 28
2.11 Smart Music Room Configuration 28
2.12 Smart Street Lamp Configuration 28
2.13 Smart Solar Panel Configuration 30

Chapter 3 PerformanceEvaluation **34**
3.1 SimulationEnvironment/SimulationProcedure 34
3.2 ResultsandDiscussions 34

Chapter4 Conclusion **39**
4.1 Introduction 39
4.1 PracticalImplications 39
4.2 Scope of Future Work 39

References **40**

Chapter 1

1.1 Introduction:

Smart Office denotes a style of modern office design that focuses on applying IoT(Internet of Things) technology, to which data is connected via internet automatically. With this technology, you can easily access data in remote areas. The connection of office equipment with IoT facilitates the work of employees in many ways. It saves time traveling to the workplace; leads to more focus on work; allows employees to work independently with less pressure and promotes optimum productivity.

A smart office is a smart workplace that traditionally combines several data-driven and digitally-enabled services and solutions to, among others, enhance worker experience and efficiency, realize cost and energy savings, and add value to the office (building) with apps for various stakeholders.

These services are mainly based on the same technologies we meet in the broader context of smart buildings and spaces in general. They include IoT (Internet of Things), data analytics, cloud computing, and the technologies that existed to manage aspects of office buildings (e.g., HVAC, light control) for a longer time, and now increasingly get 'smart' and integrated into 'smart' vertical platforms. Other technologies that get used, sometimes for particular smart office use cases, are VR/AR and artificial intelligence, to mention a few.

1.2 Goal/ Objectives:

1. Saves energy: IoT technology can be used to control energy usage by setting up times for use or automatically turn off the devices when no use is detected. Energy output is controlled by IoT to reduce business operating costs.

2. Work to full capacity: When IoT technology is applied to help manage and control the work environment, employees focus more on their work. This modern environment helps facilitate work operations, creating a highly-motivated organization where employees are able to work to full capacity and aspire to maintain effective output.

3. A more effective business operation: With IoT technology, it is much easier to manage and control office equipment creating a smooth and effective workflow, leading the company to maintain a high level of success against competitors.

4. High level security system: IoT technology brings innovation and modern functions to promote effective security for the office. This guarantees that all important documents and data are securely stored in the cloud system and offers complete protection against theft.

However, employing IoT for a smart office has certain limitations. To take advantage of IoT with its modern advances, a reliable, fast internet connection is required, run on high-quality devices and software. Such devices may be costly, but the return in efficiency outweighs the investment. Further examples of the additional benefits of a smart office follow below:

Security: CCTV that can be operated and controlled via mobile applications; smart-face scanners used to control the checking in/out of employees and self-registration; a check-in system for visitors via kiosk without the need of a staff member to be present; innovative data storage on the cloud system allowing easy access to data from any location and the protection and security of important data.

Energy control: The use of sensors to detect motion and to automatically control the switching on/off of appliances such as air conditioners, lights, roller blinds and meeting room equipment.

Communication: Smart meeting room technology, modern and effective.

Smart equipment and smart furniture: Tables and office equipment can be adjusted independently according to the preferences of each employee to facilitate their workflow with improved performance and productivity.

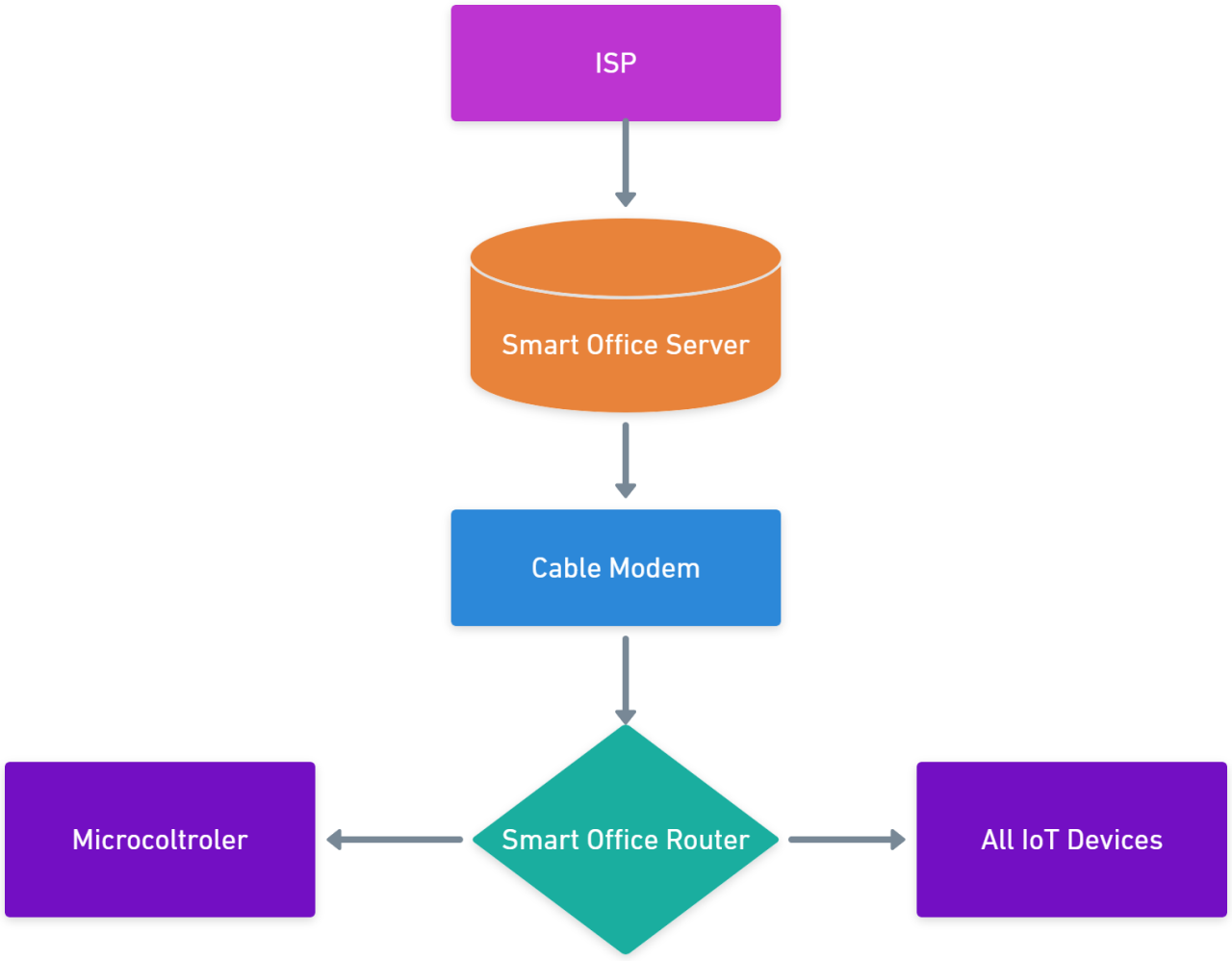
Some Key Points

1. Smart office in the era of the 'new normal' employs IoT technology to help promote the smooth and continuous operation of businesses, the well-being of employees and improved performance and productivity.
2. To take advantage of a smart office, experts are on hand to help design and control the workspace in accordance with the objectives of your organization. At Toppan, we provide this expertise, offering the best solutions for your office.
3. To benefit your organization on the return to the office, to work at full capacity and outgrow your competitors, do not hesitate to contact Toppan. We provide expert consultation on the best way to employ smart office in your workplace to give your business and its employees a modern advantage and a sustainable future.

Chapter 2

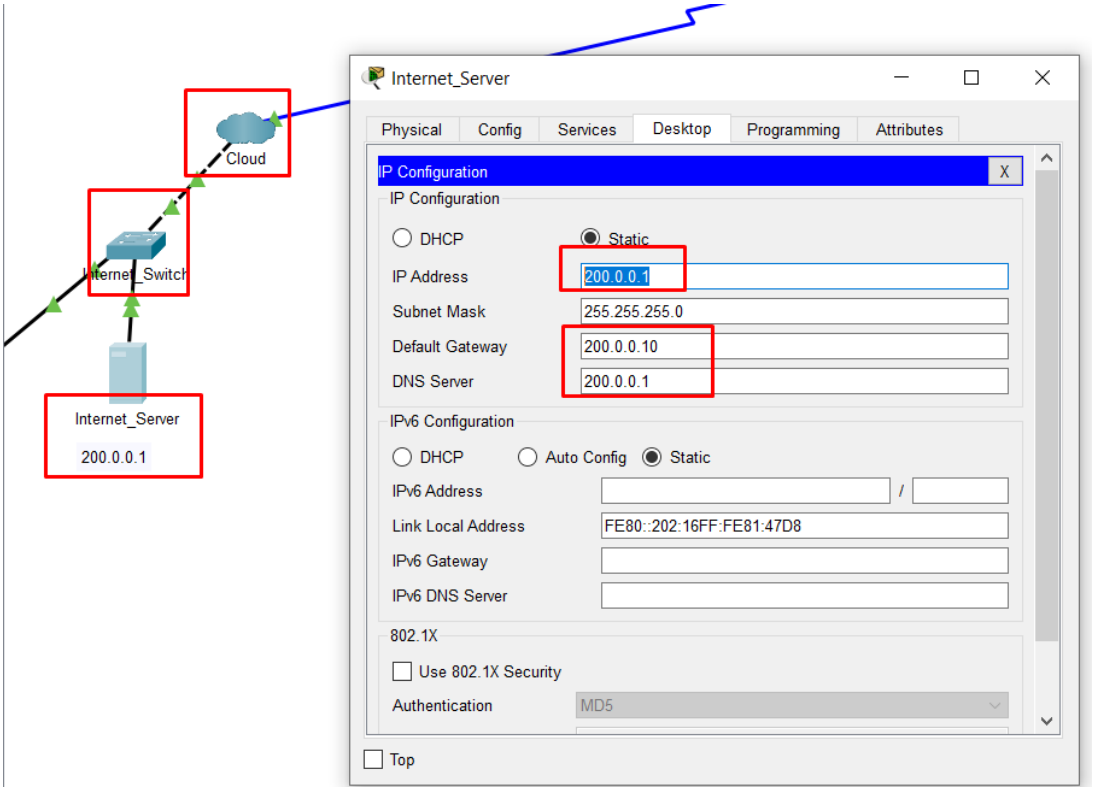
2. Design/ Development:

The flow chart of this project:



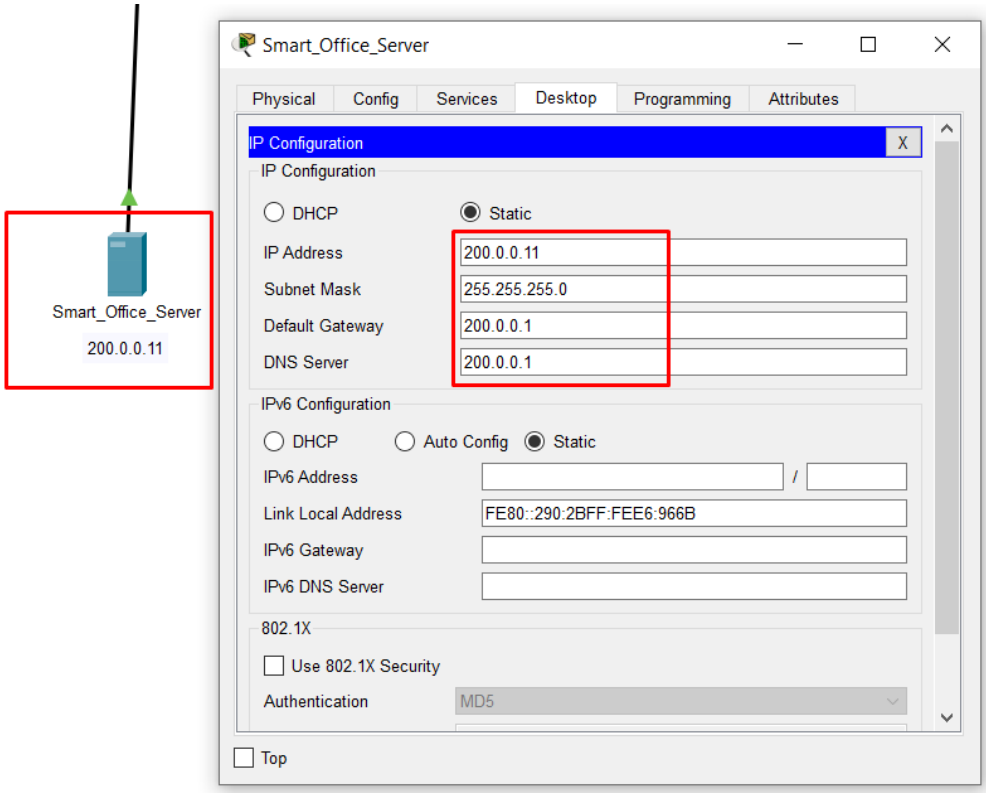
Smart Office System Networking Flow Chart
@Md Torikul Islam

2.1. ISP Network Configuration (Cluster): At first I am talking an internet connection by cloud and switch and create an internet server for all of my network. My internet server IP is 200.0.0.1. DNS server is 200.0.0.1 and default gateway 200.0.0.10. Cloud and switch is connected copper cross over wire and switch and internet server is connected by copper straight.

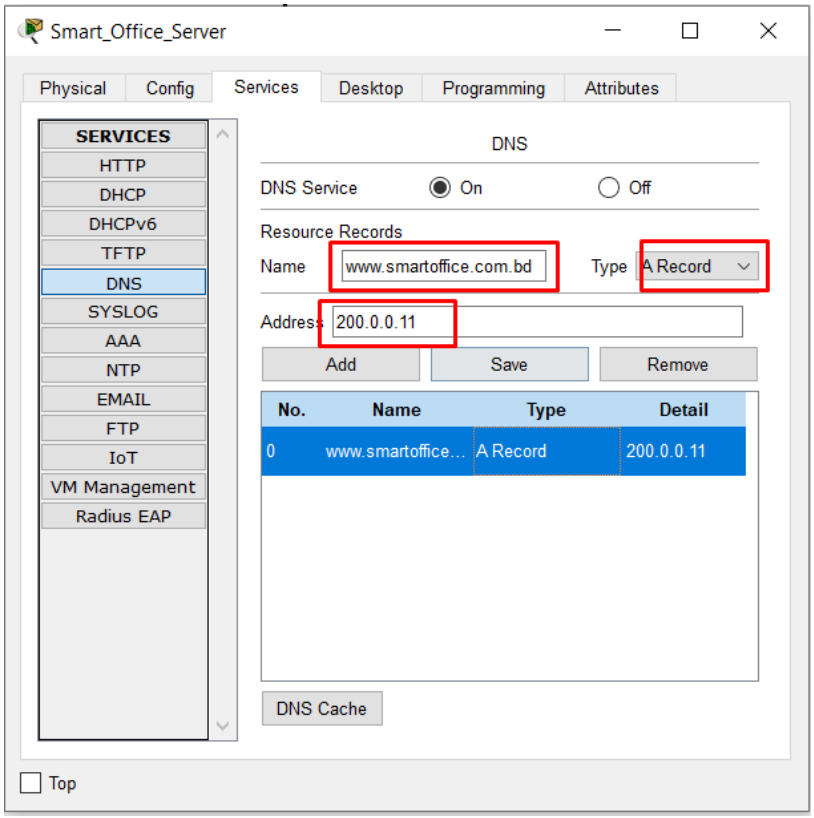


Then I create a cluster network by clicking cluster icon. Now I am getting a cluster network from IPS for my Smart office.

2.2. Smart Office Server Configuration: For controlling all of my devices, data and mail, IoT devices, DNS server, I am taking a server in my smart office which name is smart office server. IP address is 200.0.0.11. The server connecter to the ISP cluster by copper straight wire. The configuration figure below:



Then I add a DNS A record in this server. The domain is www.smartoffice.com and IP is 200.0.0.11. I am creating an user and pass. User is: admin and Pass: admin for all IoT devices.



2.3. Cable Modem Configuration: Cable modem connected to the cluster by coaxial cable.

2.4. Main Office Wi-Fi Router Configuration: The router is connected to the cable modem by copper straight wire. At first I was going to router setting and give a display name is Smart_Office. Then I go to LAN and give the IP of my LAN / Router default which is 192.168.0.1. And finally enable dynamic host configuration protocol (DHCP).

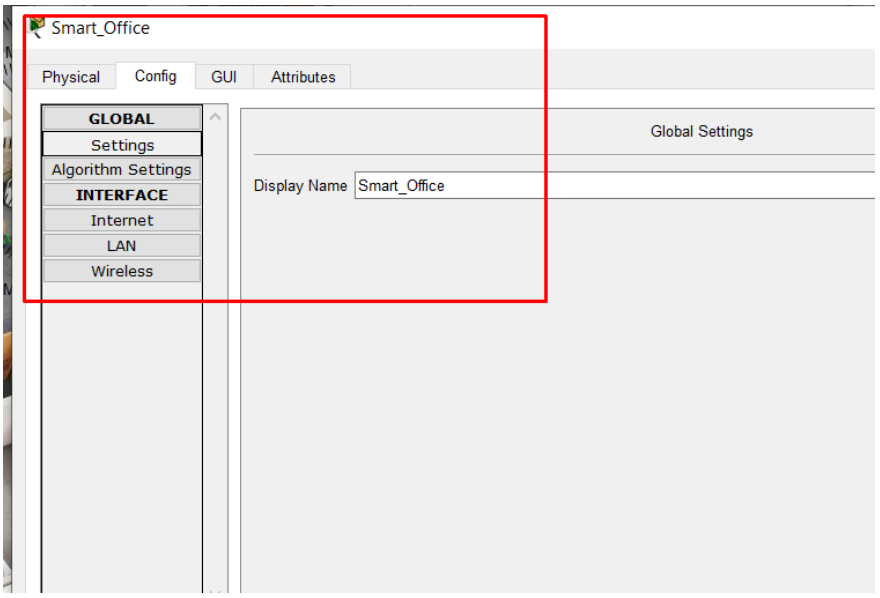


Figure: Router Display name setup

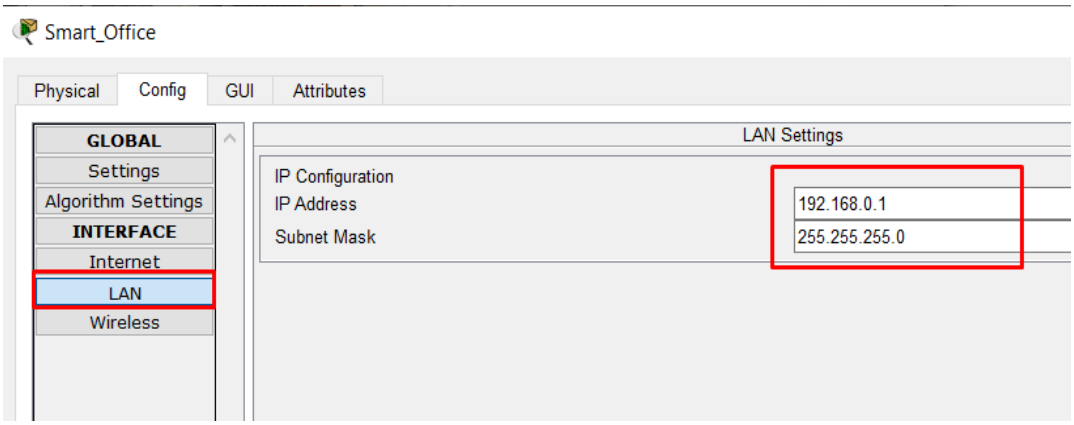


Figure: Router IP Configuration

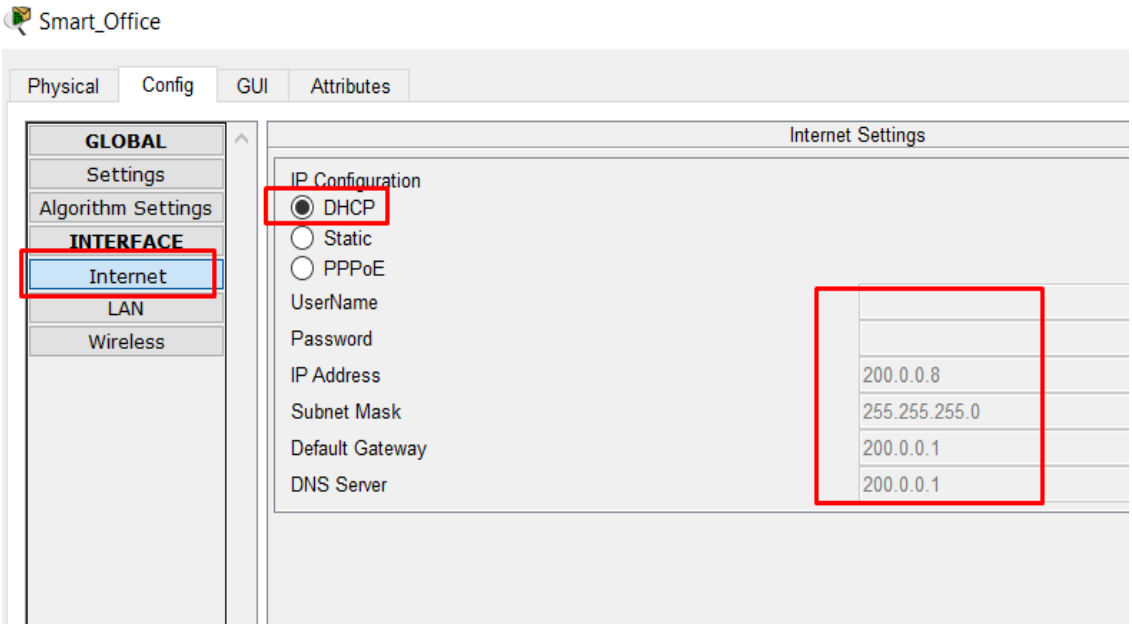
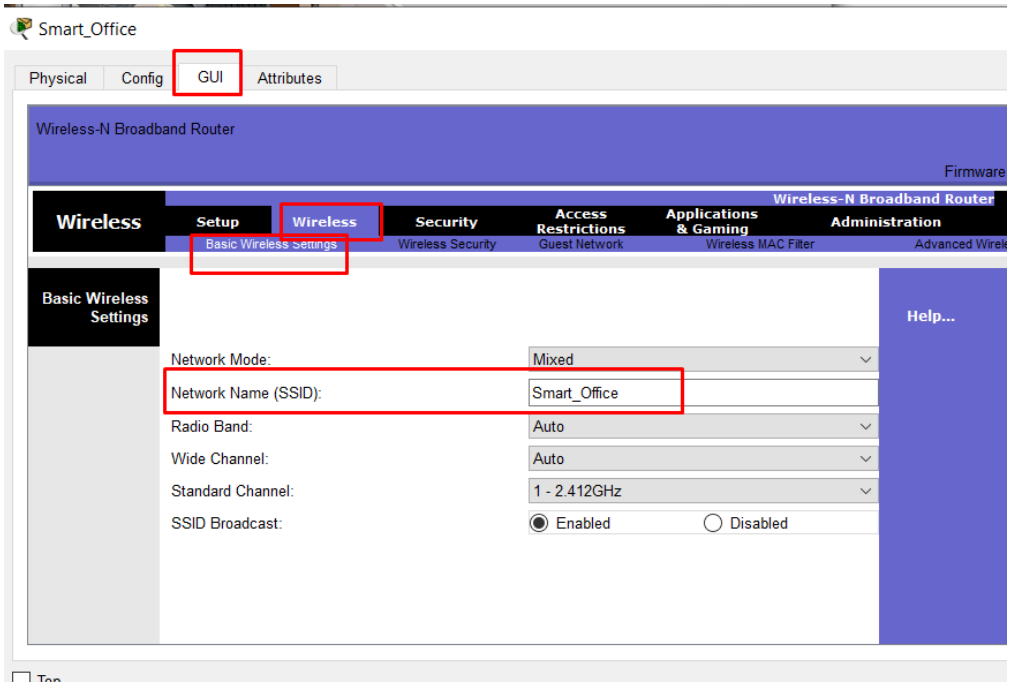
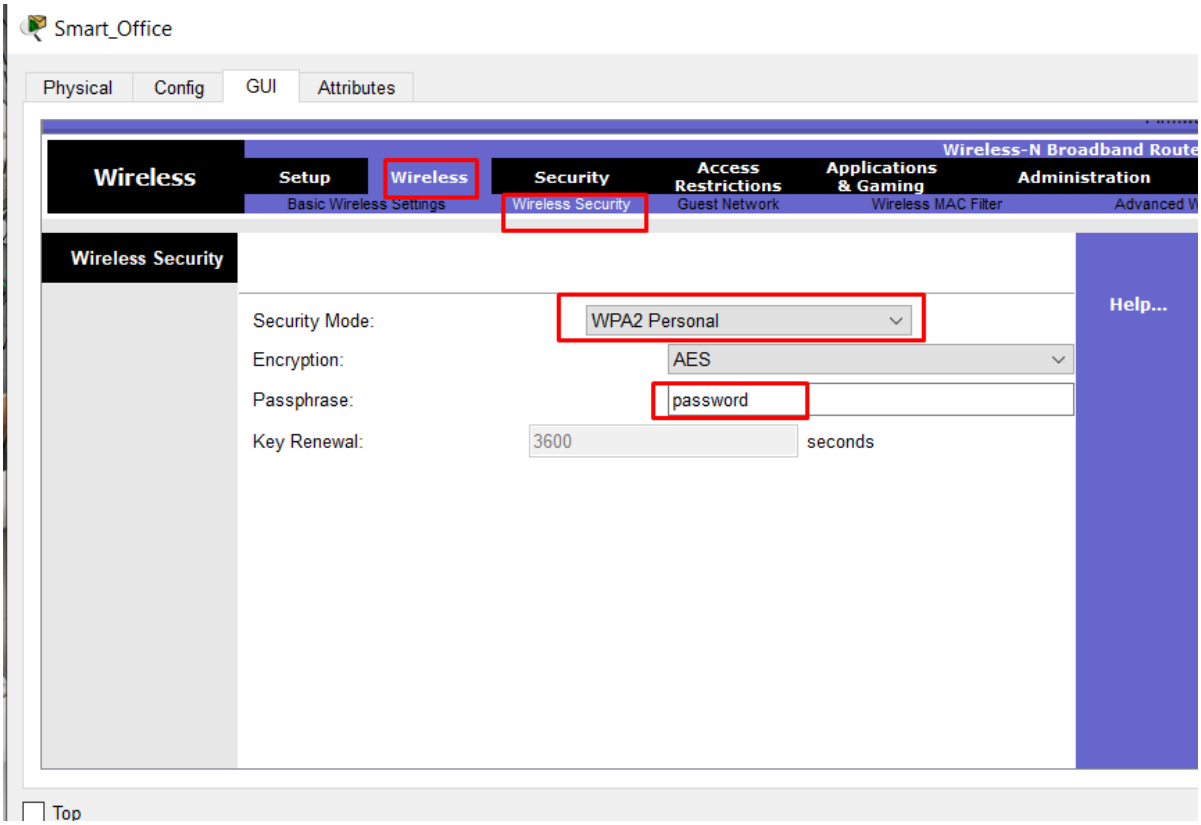


Figure: DHCP Enable of the Router

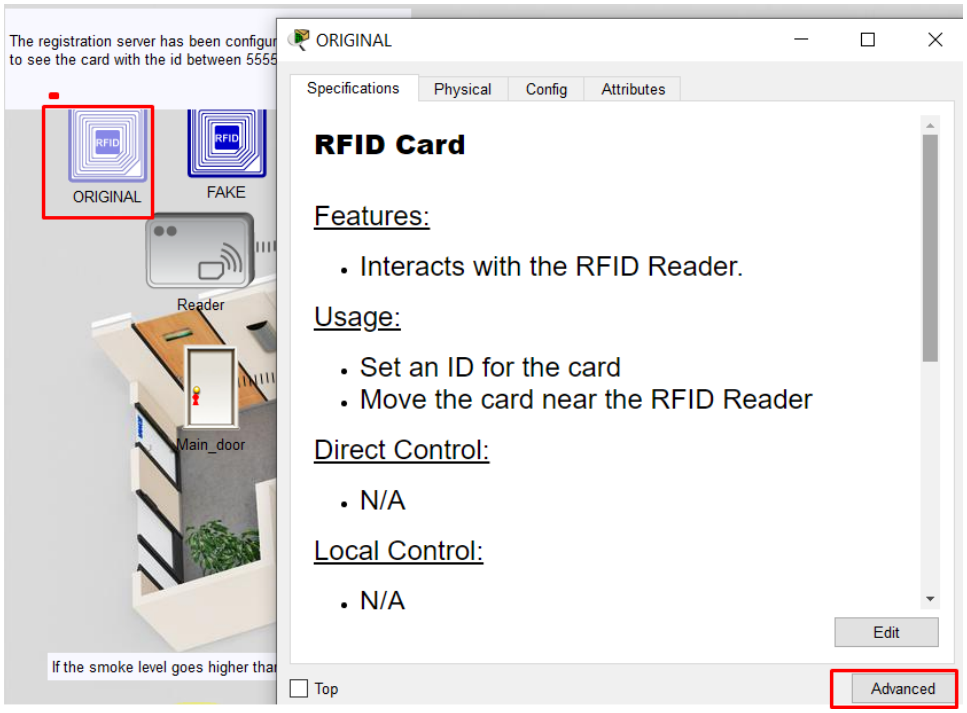
Now I am going to set up router SSID and Password by using GUI. The SSID is Smart_Office and Password is password. Here is the configuration. At first go to GUI and click Wireless tab then Basic wireless setting as follow as the picture below:

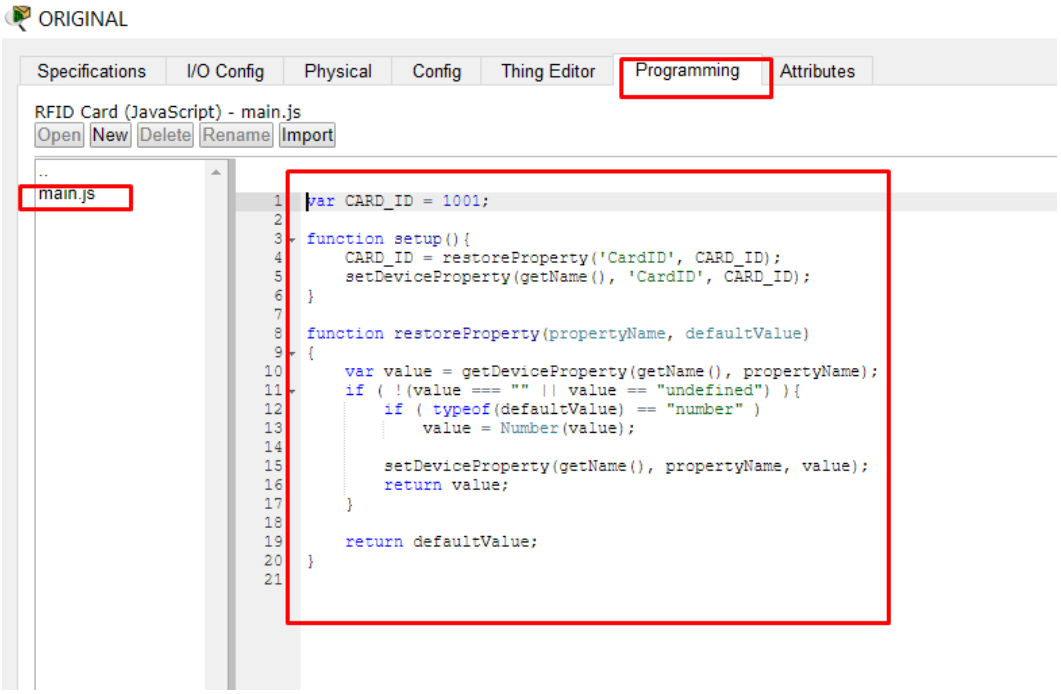


Then again click security tab and wireless security tab and set up the wifi password. Security mode will be WPA2.



2.5. Smart Door and RFID card Configuration: RFID is an IoT device. Just click double click the devices and go the advance and I put down the js code in programming section for validation of the RFID card.



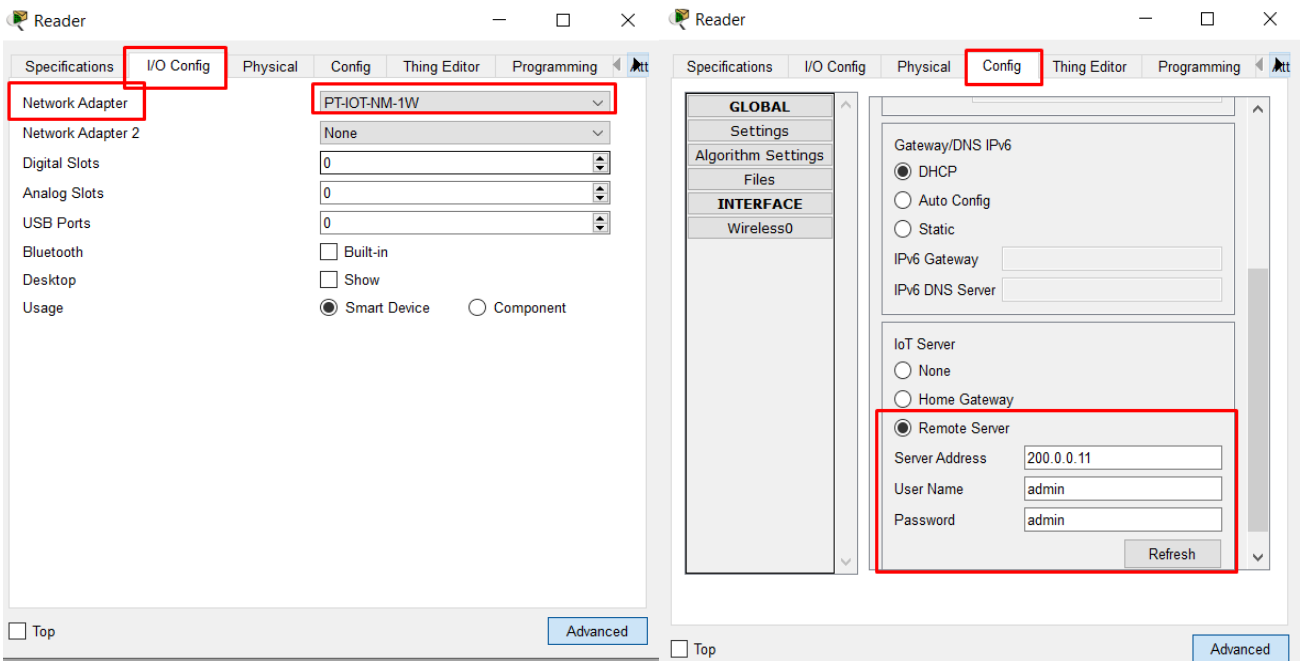


JS Code:

```
var CARD_ID = 1001;

function setup(){
    CARD_ID = restoreProperty('CardID', CARD_ID);
    setDeviceProperty(getName(), 'CardID', CARD_ID);
}

function restoreProperty(propertyName, defaultValue)
{
    var value = getDeviceProperty(getName(), propertyName);
    if ( !(value === "" || value == "undefined") ){
        if ( typeof(defaultValue) == "number" )
            value = Number(value);

        setDeviceProperty(getName(), propertyName, value);
        return value;
    }

    return defaultValue;
}
```

RFID Card Reader: At first double click the card reader and then go the advance tab. Then go I/O config and ser network adapter. Finally go to config tab and enable DHCP mode and Remote server. Give the Setver address, user name and pass as like 200.0.011 user: admin, pass: admin to connect with the Smart_Office server by Smart_office

router.



Smart Door JS Code:

```
var ENVIRONMENTS = ["Argon", "CO", "CO2", "Hydrogen", "Helium", "Methane",
"Nitrogen", "O2", "Ozone", "Propane", "Smoke"];
var ENVIRONMENT_MAX_IMPACT = -0.02; // 2% max when door opens
var TEMPERATURE_TRANSFERENCE_MULTIPLIER = 1.25; // increase speed 25% when door open
var HUMIDITY_TRANSFERENCE_MULTIPLIER = 1.25;
var GASES_TRANSFERENCE_MULTIPLIER = 2;

var doorState = 0;  // 0 is closed, 1 is opened
var lockState = 0;  // 0 is unlocked, 1 is locked

function setup () {
    IoEClient.setup({
        type: "Door",
        states: [{
            name: "Open",
            type: "bool"
        }, {
            name: "Lock",
            type: "options",
            options: {
                "0": "Unlock",
                "1": "Lock"
            },
```

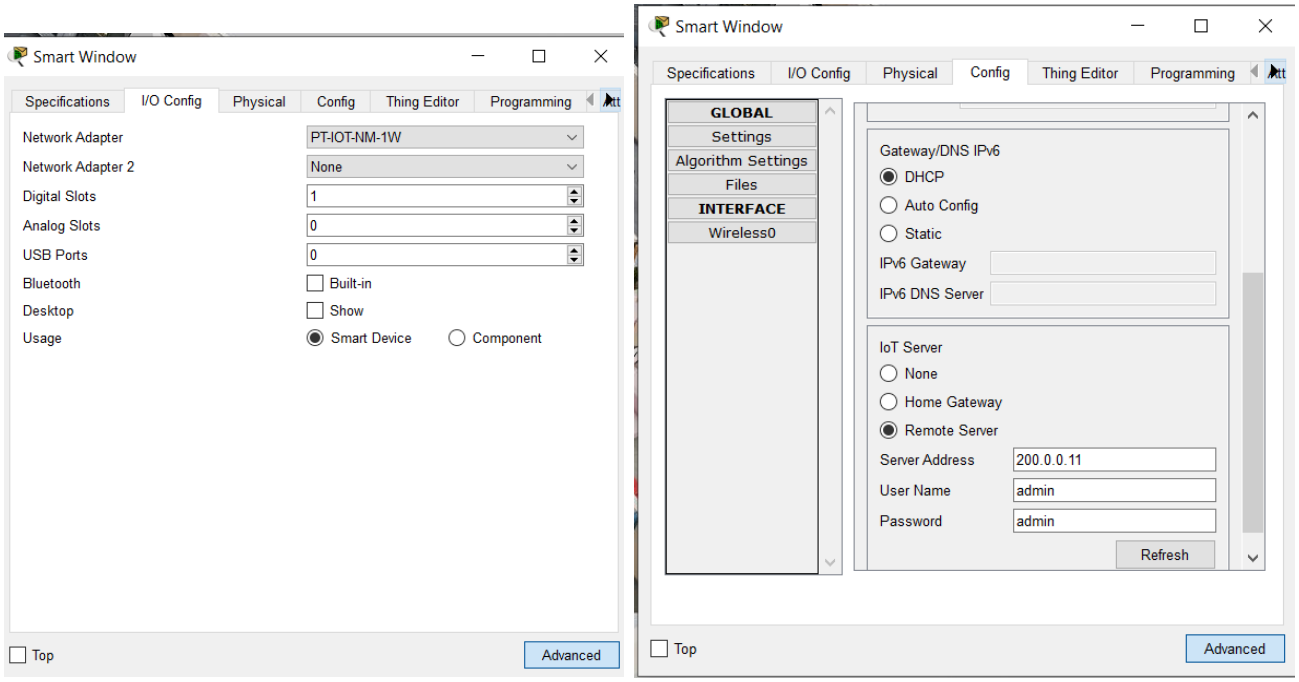
```
            controllable: true
        }]
    });

    IoEClient.onInputReceive = function (input) {
        processData(input, true);
    };

    attachInterrupt(0, function () {
        processData(customRead(0), false);
    });

    setDoorState(doorState);
    setLockState(lockState);
}

function mouseEvent (pressed, x, y, firstPress) {
    if (firstPress) {
        if ( isPointInRectangle(x, y, 10,40,5,10) )
        {
            if ( lockState === 0 ) {
                setLockState(1);
            } else {
                setLockState(0);
            }
        } else {
            if ( doorState === 0 ) {
                openDoor();
            } else {
                closeDoor();
            }
        }
    }
}

function loop () {

}

function processData (data, bIsRemote) {
    if ( data.length <= 0  ) {
        return;
    }
    Serial.println(data);

    data = data.split(",");
    var doorStateData = parseInt(data[0]);
    var lockStateData = parseInt(data[1]);
    if ( lockStateData > -1 ) {
```

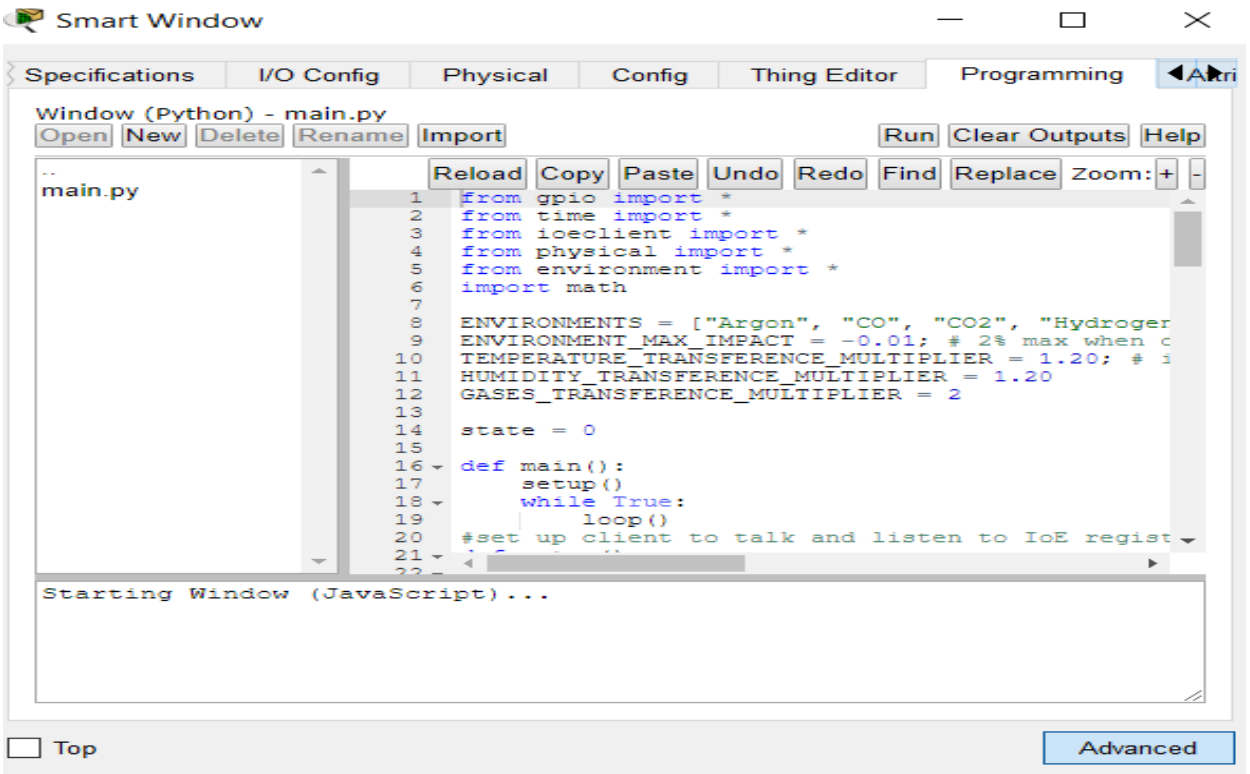
```
        setLockState(lockStateData);
    }

    if ( doorStateData > -1 && !bIsRemote ) {
        if ( doorStateData === 0 ) {
            closeDoor();
        } else {
            openDoor();
        }
    }
}
function sendReport () {
    var report = doorState+","+lockState;   // comma seperated states
    customWrite(0, report);

    IoEClient.reportStates(report);
    setDeviceProperty(getName(), "door state", doorState);
    setDeviceProperty(getName(), "lock state", lockState);
}

function closeDoor () {
    setDoorState(0);
    updateEnvironment();
}

function openDoor () {
    if ( lockState===0 ) {
        setDoorState(1);
        updateEnvironment();
    } else {
        Serial.println("can't open locked door");
    }


}

function setDoorState (state) {
    if ( state === 0) {
        digitalWrite(1, LOW);
        setComponentOpacity("led", 1);  // show the led
    } else {
        digitalWrite(1, HIGH);
        setComponentOpacity("led", 0);  // hide the led
    }
    doorState = state;
    sendReport();
}

function setLockState (state) {
```

```
    if ( state === 0 ) {
        digitalWrite(2, LOW);
    } else {
        digitalWrite(2, HIGH);
    }

    lockState = state;
    sendReport();
}


function updateEnvironment () {
    var rate,max;
    if ( doorState == 1) {
        for (var i=0; i<ENVIRONMENTS.length; i++) {
            max = Environment.get(ENVIRONMENTS[i]) * ENVIRONMENT_MAX_IMPACT;
            // the max is reached in an hour, so we divide by 3600 to get seconds
            // then this rate is also based on 100,000 cubic meters (approx.
coporate office size)
            rate = max / 3600 * 100000 / Environment.getVolume();
            Environment.setContribution(ENVIRONMENTS[i], rate, max);
            Environment.setTransferenceMultiplier(ENVIRONMENTS[i],
GASES_TRANSFERENCE_MULTIPLIER);
        }

        Environment.setTransferenceMultiplier("Ambient Temperature",
TEMPERATURE_TRANSFERENCE_MULTIPLIER);
        Environment.setTransferenceMultiplier("Humidity",
HUMIDITY_TRANSFERENCE_MULTIPLIER);
    } else {
        for (var i=0; i<ENVIRONMENTS.length; i++) {
            Environment.setContribution(ENVIRONMENTS[i], 0, 0);
            Environment.removeCumulativeContribution(ENVIRONMENTS[i]);
            Environment.setTransferenceMultiplier(ENVIRONMENTS[i], 1);
        }
        Environment.setTransferenceMultiplier("Ambient Temperature", 1);
        Environment.setTransferenceMultiplier("Humidity", 1);
    }
}

function isPointInRectangle (x,y, rx, ry, width, height) {
    if (width <= 0 || height <= 0) {
        return false;
    }

    return (x >= rx && x <= rx + width && y >= ry && y <= ry + height);
}
```

2.6. Smart Window Configuration: At first double click the window and then go the advance tab. Then go I/O config and ser network adapter. Finally go to config tab and enable DHCP mode and Remote server. Give the Setver address, user name and pass as like 200.0.011 user: admin, pass: admin to connect with the Smart_Office server by Smart_office router.

Then go to programming tab and write some python code for doing smart of this window.





Python Code:

```
from gpio import *
from time import *
from ioeclient import *
from physical import *
from environment import *
import math

ENVIRONMENTS = ["Argon", "CO", "CO2", "Hydrogen", "Helium", "Methane", "Nitrogen",
"O2", "Ozone", "Propane", "Smoke"]
ENVIRONMENT_MAX_IMPACT = -0.01; # 2% max when door opens
TEMPERATURE_TRANSFERENCE_MULTIPLIER = 1.20; # increase speed 25% when door open
HUMIDITY_TRANSFERENCE_MULTIPLIER = 1.20
GASES_TRANSFERENCE_MULTIPLIER = 2

state = 0

def main():
    setup()
    while True:
        loop()
#set up client to talk and listen to IoE registration server
def setup():
    IoEClient.setup({
        "type": "Window",
        "states": [{
            "name": "On",
            "type": "bool",
            "controllable": True
        }]
    })

    IoEClient.onInputReceive(onInputReceiveDone)
    add_event_detect(0, detect)

    state = restoreProperty("state", 0)
    setState(state)

def onInputReceiveDone(data):
    processData(data, True)

def detect():
    processData(customRead(0), false)

def restoreProperty(propertyName, defaultValue):
    value = getDeviceProperty(getName(), propertyName)
    if  not (value is "" or value is None):
```

```
        if  type(defaultValue) is int :
            value = int(value)

        setDeviceProperty(getName(), propertyName, value)
        return value
    return defaultValue


def mouseEvent(pressed, x, y, firstPress):
    global state
    if firstPress:
        if state == True:
            setState(0)
        else:
            setState(1)


#update carbon dioxide and carbon monoxide and send new data to registration server
def loop():
    updateEnvironment()
    delay(1000)


#process data received from server
def processData(data, bIsRemote):
    if  len(data) <= 0 :
        return
    data = data.split(",")
    setState(int(data[0]))


#set state and update component image to reflect the current state
def setState(newState):
    global state
    if  newState is 0 :
        digitalWrite(1, LOW)
    else:
        digitalWrite(1, HIGH)

    state = newState
    customWrite(0, state)
    IoEClient.reportStates(state)
    setDeviceProperty(getName(), "state", state)

def updateEnvironment():
    global ENVIRONMENTS
    global ENVIRONMENT_MAX_IMPACT
    global GASES_TRANSFERENCE_MULTIPLIER
    global TEMPERATURE_TRANSFERENCE_MULTIPLIER
```

```
    global HUMIDITY_TRANSFERENCE_MULTIPLIER
    global state

    if  state == 1:
        for i in range (0,len(ENVIRONMENTS)):
            max = Environment.get(ENVIRONMENTS[i]) * ENVIRONMENT_MAX_IMPACT
            # the max is reached in an hour, so we divide by 3600 to get seconds
            # then this rate is also based on 100,000 cubic meters (approx. coporate
office size)
            rate = float(max) / 3600 * 100000 / Environment.getVolume()
            Environment.setContribution(ENVIRONMENTS[i], rate, max, True)
            Environment.setTransferenceMultiplier(ENVIRONMENTS[i],
GASES_TRANSFERENCE_MULTIPLIER)

        Environment.setTransferenceMultiplier("Ambient Temperature",
TEMPERATURE_TRANSFERENCE_MULTIPLIER)
        Environment.setTransferenceMultiplier("Humidity",
HUMIDITY_TRANSFERENCE_MULTIPLIER)
    else:
        for j in range (0, len(ENVIRONMENTS)):
            Environment.setContribution(ENVIRONMENTS[j], 0, 0, True)
            Environment.removeCumulativeContribution(ENVIRONMENTS[j])
            Environment.setTransferenceMultiplier(ENVIRONMENTS[j], 1)

        Environment.setTransferenceMultiplier("Ambient Temperature", 1)
        Environment.setTransferenceMultiplier("Humidity", 1)

if __name__ == "__main__":
    main()
```

2.7. Smart Fan Configuration: At first double click the fan and then go the advance tab. Then go I/O config and ser network adapter. Finally go to config tab and enable DHCP mode and Remote server. Give the Setver address, user name and pass as like 200.0.011 user: admin, pass: admin to connect with the Smart_Office server by Smart_office router.

Then go under the advance tab → Programming Tab

Smart Fan Python Code:

```
from gpio import *
from time import *
from ioeclient import *
from physical import *
from environment import *
```

```
import math
FAN_SPEED_LOW = 0.4; # kph
FAN_SPEED_HIGH = 0.8; # kph
COOLING_RATE = float(-1)/3600; # -1C/hour
HUMDITY_REDUCTION_RATE = float(-1)/3600; # -1%/hour

VOLUME_AT_RATE = 100000;    # the given rates are based on this volume

state = 0;  # 0 off, 1 low, 2 high
level = 0

def main():
    global state
    IoEClient.setup({
        "type": "Ceiling Fan",
        "states": [
        {
            "name": "Status",
            "type": "options",
            "options": {
                "0": "Off",
                "1": "Low",
                "2": "High"
            },
            "controllable": True
        }
        ]
    })

    IoEClient.onInputReceive(onInputReceiveDone)
    add_event_detect(0, detect)
    state = restoreProperty("state", 0)
    setState(state)
    while True:
        delay(1000)

def onInputReceiveDone(data):
    processData(data, True)

def detect():
    processData(customRead(0), False)

def restoreProperty(propertyName, defaultValue):
    value = getDeviceProperty(getName(), propertyName)
    if  not (value is "" or value is None):
        if  type(defaultValue) is int :
            value = int(value)

        setDeviceProperty(getName(), propertyName, value)
```

```
        return value
    return defaultValue

def mouseEvent(pressed, x, y, firstPress):
    if firstPress:
        toggleState()


def processData(data, bIsRemote):
    if  len(data) <= 0 :
        return
    data = data.split(",")
    setState(int(data[0]))

def sendReport():
    global state
    global report
    report = state # comma seperated states
    customWrite(0, report)
    IoEClient.reportStates(report)
    setDeviceProperty(getName(), "state", state)


def setState(newState):
    global state
    analogWrite(A1, newState)
    state = newState

    sendReport()
    updateEnvironment()

def toggleState():
    global state
    state += 1
    if  int(state) >= 3:
        state = 0

    setState(state)

def updateEnvironment():
    global VOLUME_AT_RATE
    global FAN_SPEED_LOW
    global COOLING_RATE
    global HUMDITY_REDUCTION_RATE
    global FAN_SPEED_HIGH
    global state
    volumeRatio = float(VOLUME_AT_RATE) / Environment.getVolume()

    if  int(state) == 0:
```

```
        Environment.setContribution("Wind Speed", 0, 0, True)
        Environment.setContribution("Ambient Temperature", 0, 0, True)
        Environment.setContribution("Humidity", 0,0, True)

    elif  int(state) == 1:
        Environment.setContribution("Wind Speed", FAN_SPEED_LOW, FAN_SPEED_LOW, False)

        # everytime the fan restarts, it can do another -100C
        Environment.setContribution("Ambient Temperature", float(COOLING_RATE)/2*volumeRatio,
Environment.getCumulativeContribution("Ambient Temperature")-100, True)

        Environment.setContribution("Humidity", float(HUMDITY_REDUCTION_RATE)/2*volumeRatio,
Environment.getCumulativeContribution("Humidity")-100, True)
    elif  int(state) == 2:
        Environment.setContribution("Wind Speed", FAN_SPEED_HIGH, FAN_SPEED_HIGH, False)

        Environment.setContribution("Ambient Temperature", float(COOLING_RATE)/2*volumeRatio,
Environment.getCumulativeContribution("Ambient Temperature")-100, True)

        Environment.setContribution("Humidity", HUMDITY_REDUCTION_RATE*volumeRatio,
Environment.getCumulativeContribution("Humidity")-100, True)

if __name__ == "__main__":
    main()
```

2.8. Smart Garage Configuration: At first double click the smoke detector and then go the advance tab. Then go I/O config and ser network adapter. Finally go to config tab and enable DHCP mode and Remote server. Give the Setver address, user name and pass as like 200.0.011 user: admin, pass: admin to connect with the Smart_Office server by Smart_office router.

Then go under the advance tab --> Programming Tab

Smoke Detector JavaScript Code:

```
var ENVIRONMENT_NAME = "Smoke";

var state = 0;
var level = 0;
var ALARM_LEVEL = 10.5;

function setup() {

    IoEClient.setup({
        type: "Smoke Detector",
```

```
        states: [{
            name: "Alarm",
            type: "bool",
            controllable: false
        },
        {
            name: "Level",
            type: "number",
            controllable: false
        }]
    });

    IoEClient.onInputReceive = function(input) {
        processData(input, true);
    };

    attachInterrupt(0, function() {
        processData(customRead(0), false);
    });

    state = restoreProperty("state", 0);
    restoreProperty("Alarm Level", 40);

    setState(state);


}

function restoreProperty(propertyName, defaultValue)
{
    var value = getDeviceProperty(getName(), propertyName);
    if ( !(value === "" || value == "undefined") ){
        if ( typeof(defaultValue) == "number" )
            value = Number(value);

        setDeviceProperty(getName(), propertyName, value);
        return value;
    }

    return defaultValue;
}

function loop() {
    var value = Environment.get(ENVIRONMENT_NAME);
    //Serial.println(value);
    if (value >= 0)
        setLevel(Environment.get(ENVIRONMENT_NAME));

    delay(1000);
```

```
}

function processData(data, bIsRemote) {
    if (data.length <= 0 )
        return;
    data = data.split(",");
    setState(parseInt(data[0]));
}

function sendReport()
{
    var report = state + "," + level;   // comma seperated states
    IoEClient.reportStates(report);
    setDeviceProperty(getName(), "state", state);
    setDeviceProperty(getName(), "level", level);
}

function setState(newState) {
    state = newState;

    if (newState === 0)
        digitalWrite(1, LOW);
    else
        digitalWrite(1, HIGH);

    sendReport();
}

function setLevel(newLevel) {
    if (level == newLevel)
        return;

    level = newLevel;
    if (level > ALARM_LEVEL)
        setState(1);
    else
        setState(0);

    sendReport();
}
```

2.9. Fire Monitor and Sprinkler Configuration: At first double click the Fire Monitor and Sprinkler and then go the advance tab. Then go I/O config and ser network adapter. Finally go to config tab and enable DHCP mode and Remote server. Give the Setver address, user name and pass as like 200.0.011 user: admin, pass: admin to connect with

the Smart_Office server by Smart_office router. Then Go to advance tab and programming:

Fire Monitor JavaScript Code:

```
// Purpose:
// This is a generic single slot output sensor.
// It can check environment variables, objects properties in range, and send a
signal.
// It is designed to have setup functions called that determine how it behaves.
// The setup functions are defined in the library.

// A delay will run each update.  It will be for this amount of time.
// Setting to 0 means there is no delay.
// If making a custom write, you can't have a delay.  customWrite requires the value
to be set each loop.
// The custom write thing seems to be a bug in the engine, it should get fixed
sometime.

var g_delayTime = 1000;

// IR wavelength that counts as fire.  760 nm ~ 1100 nm light, etc.
var g_sensitivity = {low: 760, high: 1100};
// Purpose:
// Sets the sensor up to look for a device IR and determine if it is a fire.
function setup()
{
    setupDetectProperty('IR', createPropertySearchDevices(null, 500, 600, 'CENTER'),
createDigitalWrite(0));

    setupRegistrationServer('Fire Sensor', 'Fire Detected');
}

function loop()
{
    // Update the sensor.
    updateSensor();
}

function processValue(value, deviceIDs)
{
    // From the readings, determine what is the highest in detection reading in the
correct range range and convert to be a strength [0, 255]
    var maxInRange = 0;
    var outOfRange = false;

    for(var ind = 0; ind < value.length; ++ind)
```

```
    {
        if(value[ind] > maxInRange)
            maxInRange = value[ind];
    }

    // Convert the in range value to a digital signal.

    if((g_sensitivity.low <= maxInRange) &&
       (g_sensitivity.high >= maxInRange))
        value = HIGH;
    else
        value = LOW;

    return value;
}
```

Sprinkler Python Code:

```
from options import Options
from time import *
from physical import *
from gpio import *
from environment import Environment
from ioeclient import IoEClient


WATERLEVEL_RATE = 0.1 # 0.1 cm per second        # var WATERLEVEL_RATE
HUMIDITY_RATE = 5.0 / 3600 # 5% per hour        # var HUMIDITY_RATE
VOLUME_AT_RATE = 100000        # var VOLUME_AT_RATE
MAX_RATE = 1.e6

state = 0 # 0 off, 1 on        # var state



def on_input_received(rinput):
    processData(rinput, True)

def on_event_detect_0():
    processData(customRead(0), False)

def setup ():


    IoEClient.setup({
```

```
        "type": "Fire Sprinkler",
        "states": [{
            "name": "Status",
            "type": "bool",
            "controllable": True
        }]
    })


    IoEClient.onInputReceive( on_input_received )

    add_event_detect(0, on_event_detect_0)

    setState(state)



def mouseEvent (pressed, x, y, firstPress):
    if firstPress:
        setState(0 if state else 1)




def processData (data, bIsRemote):
    if len(data) <= 0:
        return
    setState(int(data))



def setState (newState):
    global state
    state = newState

    analogWrite(A1, state)
    customWrite(0, state)
    IoEClient.reportStates(state)
    setDeviceProperty(getName(), "state", state)

    updateEnvironment()



def updateEnvironment ():

    if state == 1:
        volumeRatio = VOLUME_AT_RATE / Environment.getVolume()        # var volumeRatio
        Environment.setContribution("Water Level", WATERLEVEL_RATE * volumeRatio, MAX_RATE, True)
```

```
        Environment.setContribution("Humidity", HUMIDITY_RATE * volumeRatio, MAX_RATE, True)
    else:
        Environment.setContribution("Water Level", 0, MAX_RATE, True)
        Environment.setContribution("Humidity", 0, MAX_RATE, True)


if __name__ == "__main__":
    setup()
    while True:
        #loop()
        sleep(0)
```

2.10. Trip Sensor Configuration: At first double click the trip sensor and then go the advance tab. Then go I/O config and ser network adapter. Finally go to config tab and enable DHCP mode and Remote server. Give the Setver address, user name and pass as like 200.0.011 user: admin, pass: admin to connect with the Smart_Office server by Smart_office router.

2.11. Smart Music Room Configuration: At first double click the music player and speaker and then go the advance tab. Then go I/O config and ser network adapter. Finally go to config tab and enable DHCP mode and Remote server. Give the Setver address, user name and pass as like 200.0.011 user: admin, pass: admin to connect with the Smart_Office server by Smart_office router.

2.12. Smart Street Lamp Configuration: At first double click the street lamp and then go the advance tab. Then go I/O config and ser network adapter. Finally go to config tab and enable DHCP mode and Remote server. Give the Setver address, user name and pass as like 200.0.011 user: admin, pass: admin to connect with the Smart_Office server by Smart_office router.

Then go under the advance tab --> Programming Tab

Sensor Light JavaScript Code:

```
// cl-sensor-light.js
// Env.Sunlight is assumed to be from 0 to 100%

var SensorLight = function () {
    this.eLightMin = 0;
    this.eLightMax = 100; // this is %, per Environment semantics for sunlight
    this.eLightMinValueOn = this.eLightMin+(this.eLightMax-this.eLightMin) / 3;
```

```
    // init
    setComponentOpacity("SensorLightOff", 0);
    setComponentOpacity("SensorLightOn", 0);
    setComponentOpacity("SensorLight", 0);

    this.elight = Environment.get("Sunlight");
    this.sync_to_env();
    this.update_visuals();
};

SensorLight.prototype.update = function (sensors, devices) {
    this.sync_to_env();
    this.update_visuals();
};

SensorLight.prototype.valueDirection = function () {
    if(this.elightPrev === this.elight)
        return 0;
    else if(this.elightPrev < this.elight)
        return 1;
    return -1;
};

SensorLight.prototype.value = function () {
    return this.elight;
};

// private
SensorLight.prototype.update_visuals = function () {
    // value
    var opacity = 0,
        value = this.value();
    if(value < this.eLightMinValueOn) {
        opacity = 1 - (value - this.eLightMin)/(this.eLightMinValueOn -
this.eLightMin);
        setComponentOpacity("SensorLightOn", 1);
        setComponentOpacity("SensorLightOff", 0);
    }
    else{
        setComponentOpacity("SensorLightOn", 0);
        setComponentOpacity("SensorLightOff", 1);
    }
    setComponentOpacity("SensorLightLevel", opacity);
};

// private
SensorLight.prototype.sync_to_env = function (){
    this.elightPrev = this.elight;
    this.elight = Environment.get("Sunlight");
```

```
    if(this.elight < this.eLightMin)
        this.elight = this.eLightMin;
    if(this.elight > this.eLightMax)
        this.elight = this.eLightMax;
};
```

2.13. Smart Solar Battery Configuration: At first double click the solar battery and then go the advance tab. Then go I/O config and ser network adapter. Finally go to config tab and enable DHCP mode and Remote server. Give the Setver address, user name and pass as like 200.0.011 user: admin, pass: admin to connect with the Smart_Office server by Smart_office router.

Then go under the advance tab --> Programming Tab

Solar Battery Python Code:

```
from gpio import *
from time import *
from ioeclient import *
from environment import *
from physical import *
import math

INPUT_PIN = 0
OUTPUT_PIN = 1
MAX_STORED_POWER = 1500.0 #2000 watts
MULTIPLIER = float(255)/1023
LOG_BASE = 1.074911103457137335981548986755 8
availablePowerPercent = 0.0
availablePower = float(MAX_STORED_POWER)/2.0
receivedPower = 0
AVAIL_POWER_PROP="Available power (%)"
CHARGE_RATE = 19
LOOP_DELAY = 500

def setup():
    setImage('100')
    IoEClient.setup({
        "type": "Battery",
        "states": [
            {
            "name": "Available power",
            "type": "number",
            "unit": '%',
```

```
            "controllable": False
            }
     ]
     })

    IoEClient.onInputReceive(onInputReceiveDone)

    global availablePowerPercent
    global availablePower

    init_percent = availablePower/MAX_STORED_POWER*100
    availablePowerPercent = restoreProperty(AVAIL_POWER_PROP, init_percent)
    if availablePowerPercent < 0 or availablePowerPercent > 100:
        availablePowerPercent = init_percent
    availablePower = availablePowerPercent*MAX_STORED_POWER/100

    sendReport()


def onInputReceiveDone(input):
    processData(input, True)

def restoreProperty(propertyName, defaultValue):
    value = getDeviceProperty(getName(), propertyName)
    if  not (value is "" or value is None):
        if  type(defaultValue) is float :
            value = float(value)

        setDeviceProperty(getName(), propertyName, value)
        return value
    return defaultValue


def main():
    global availablePowerPercent
    global availablePower
    global MAX_STORED_POWER
    setup()
    while True:
        availablePowerPercent = (float(availablePower)/MAX_STORED_POWER) * 100
        if availablePowerPercent < 10:
            setImage('Battery_0')
        elif availablePowerPercent < 30:
            setImage('Battery_20')
        elif availablePowerPercent < 50:
            setImage('Battery_40')
        elif availablePowerPercent < 70:
            setImage('Battery_60')
        elif availablePowerPercent < 90:
```

```
            setImage('Battery_80')
        elif availablePowerPercent <= 100:
            setImage('Battery_100')

        sendReport()
        delay(LOOP_DELAY)
        receivePower(LOOP_DELAY*3)
        sendPower()
        delay(LOOP_DELAY*2)

def receivePower(timeDeltaMsec):
    global availablePower
    global receivedPower
    receivedPowerRaw = analogRead(0)
    if receivedPowerRaw > 0:
        receivedPower = math.pow(LOG_BASE, receivedPowerRaw * MULTIPLIER)
        receivedPower = min(receivedPower, CHARGE_RATE * timeDeltaMsec * 0.001)
    else:
        receivedPower = 0
    availablePower += receivedPower
    if availablePower > MAX_STORED_POWER :
        availablePower = MAX_STORED_POWER



def sendPower():
    global availablePower
    totalPowerUsed = 0
    currentWattage = 0
    numberOfSlots = int(getSlotsCount())
    deviceAtPort = False
    for i in range (1, numberOfSlots):
        #Start at 1 because 0 is the input port
        if getAttributeOfDeviceAtSlot('wattage', i):
            currentWattage = float(getAttributeOfDeviceAtSlot('wattage', i))
            deviceAtPort = True
        else:
            currentWattage = 0


        totalPowerUsed += currentWattage
        if deviceAtPort == True:
            if availablePower > currentWattage:
                outputElectricity(i, currentWattage)
                availablePower -= currentWattage
            else:
                outputElectricity(i, 1)

        deviceAtPort = False
```

```
    if availablePower < 0:
        availablePower = 0




def sendReport():
    global availablePowerPercent
    IoEClient.reportStates(availablePowerPercent)
    setDeviceProperty(getName(), AVAIL_POWER_PROP, availablePowerPercent)


def setImage(component):
    setComponentOpacity(component, 1)

    if not 'Battery_0' == component :
        setComponentOpacity('Battery_0', 0)
    if not 'Battery_20' == component :
        setComponentOpacity('Battery_20', 0)
    if not 'Battery_40' ==  component :
        setComponentOpacity('Battery_40', 0)
    if not 'Battery_60' ==  component :
        setComponentOpacity('Battery_60', 0)
    if not 'Battery_80' ==  component :
        setComponentOpacity('Battery_80', 0)
    if not 'Battery_100' ==  component :
        setComponentOpacity('Battery_100', 0)


def outputElectricity(port, value):
    global LOG_BASE
    el_log = math.floor(float(math.log(value))/math.log(LOG_BASE))
    if el_log < 0:
        el_log = 0
    elif el_log > 255:
        el_log = 255
    analogWrite(port, el_log)

if __name__ == "__main__":
    main()
```

Chapter 3

3.1 Simulation Environment/Simulation Procedure: At first go to cisco packet tracer and part by part check our whole project simulation.

3.2 Results and Discussions: Here is my full project at a glance:

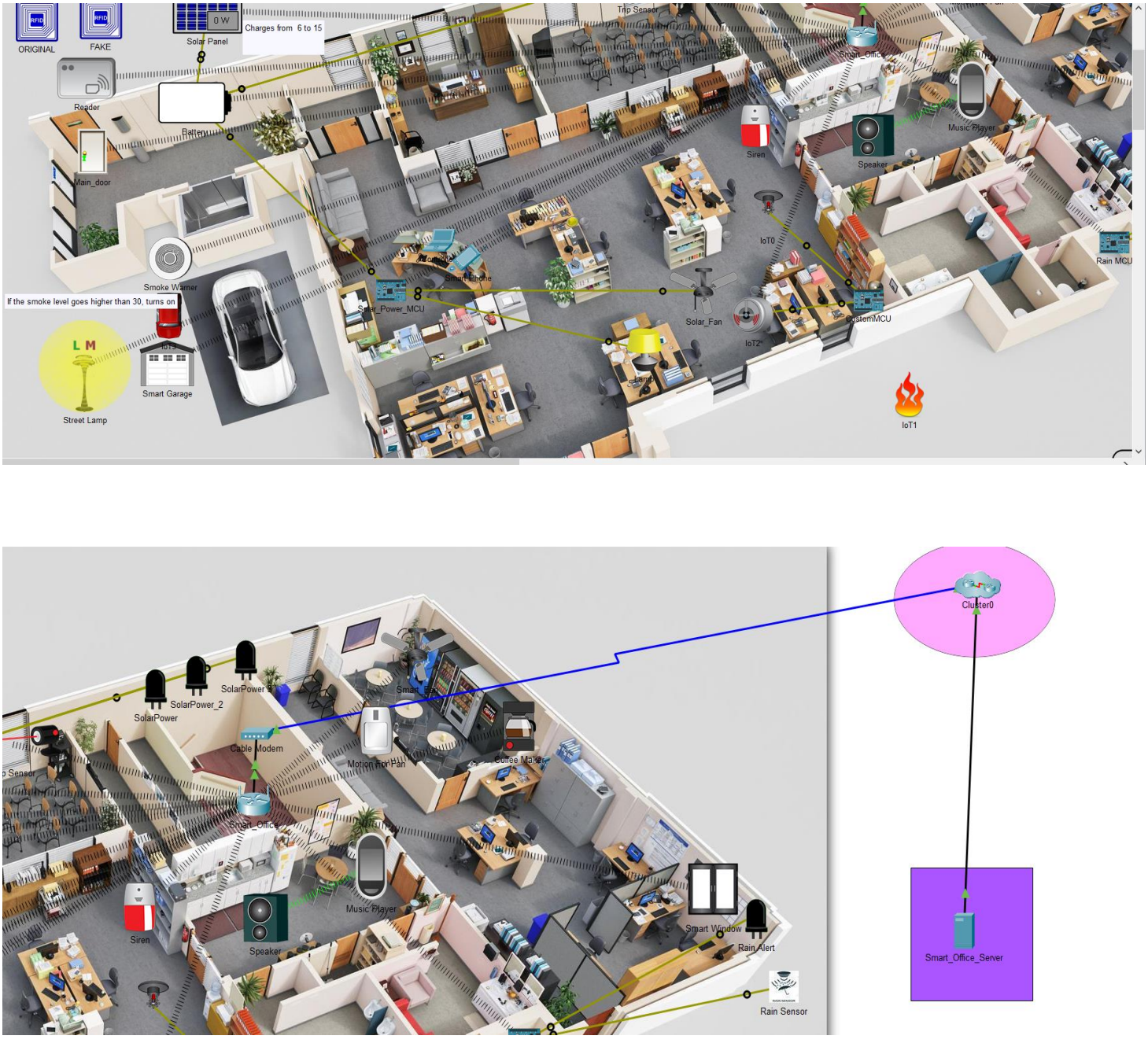
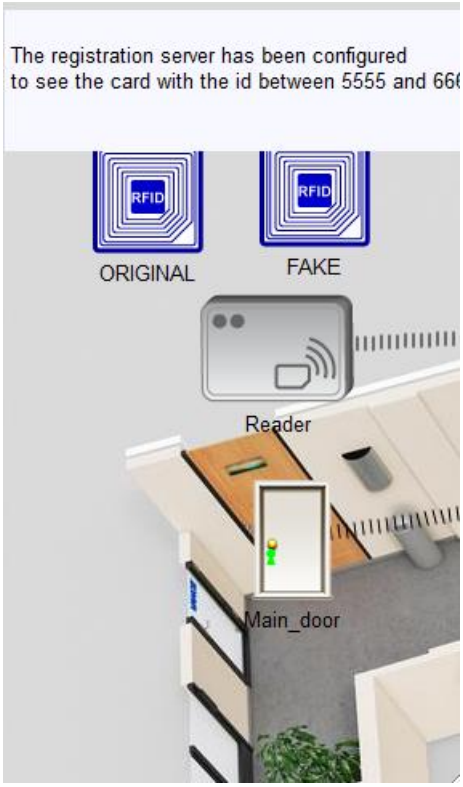


Figure: Smart Office System by Using IoT Devices

If we give correct RFID card then Card Reader Read the RFID card and our Smart door will be automatically open. But If we give wrong RFID card the door will not open. Here is the picture



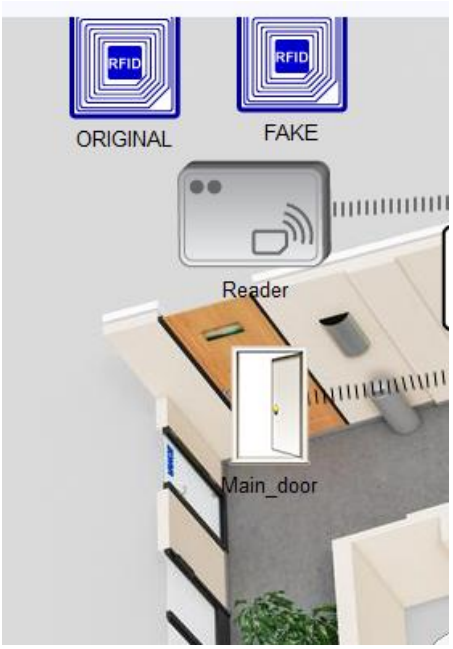
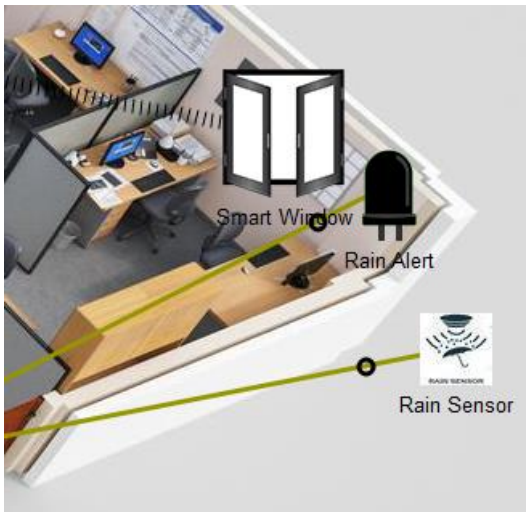


Figure: Right RFID Card

Figure: Door Open

Smart Window is open when there is no rain outside and in day. Not only that if there are harmful gas comes in then the window will be closed. It will be closed at Night automatically. If rain sensor detect rain then the Led will be glow and then window will be closed automatically.



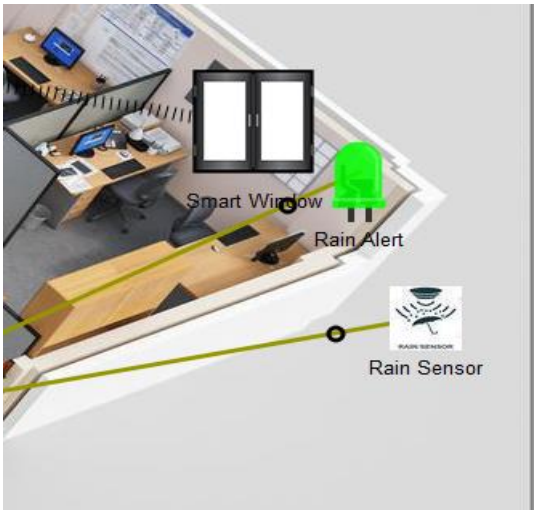


Figure: Open Window

Figure: Closed window

Now for Smart Garage, When smoke detector detect smoke for a certain level then the garage door will be open automatically. If the smoke will free then the garage door will be closed automatically.

Figure: Garage Door Closed

Figure: Garage door open

Now Fire Monitor and Fire Sprinkler and Siren. When any fire detect in fire monitor then the siren will be turn on and the sprinkler will be automatically open to save office and employees.

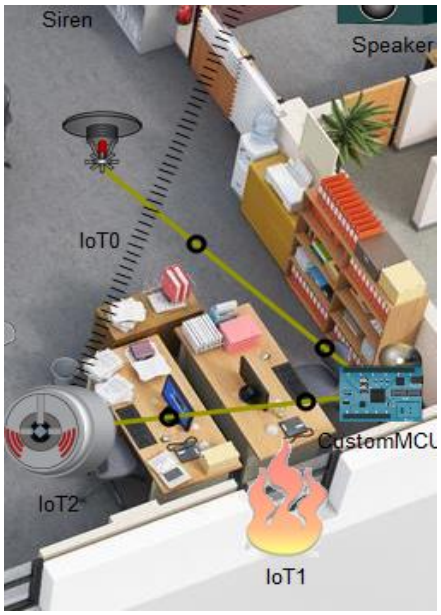


Figure: No Fire Detect

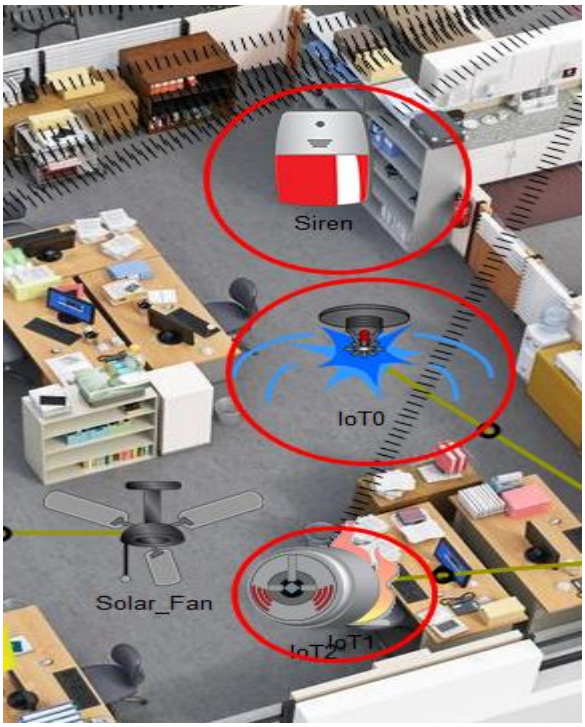


Figure: Fire Detect

If anyone comes in coffee room, the motion sensor will be detect and then the coffee room fan will be automatically open and the coffee maker will be start.

Figure: Smart Coffee Room

When anyone comes in trip sensor then the siren will be turn on. Example any thief will forcefully come in office main server then the trip sensor send data to siren and the server room door will be close automatically.

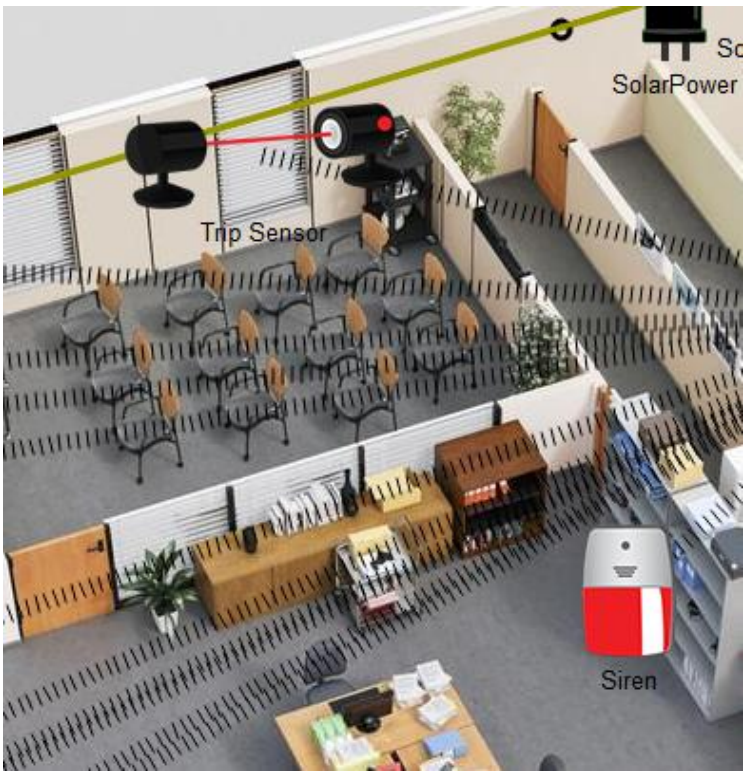


Figure: Trip Sensor

Now for Music Room anyone can play music.

The very important feature is that we can control our all IoT devices by using any Mobile or Web application. Just for now I am going a mobile web browser and go the IP is 2000.0.0.11 hit enter. Then give user: admin, pass: admin. And login in let's see

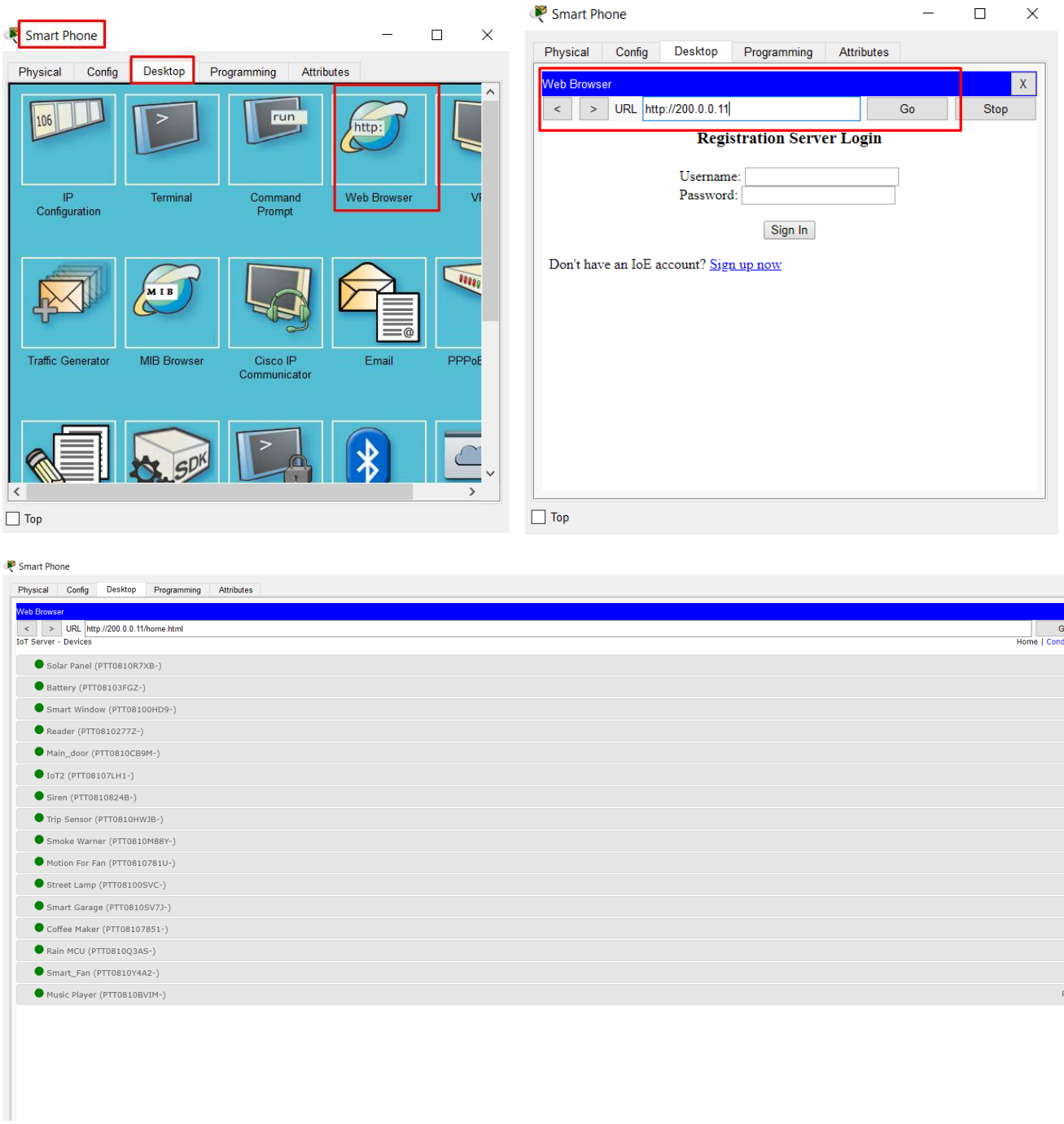
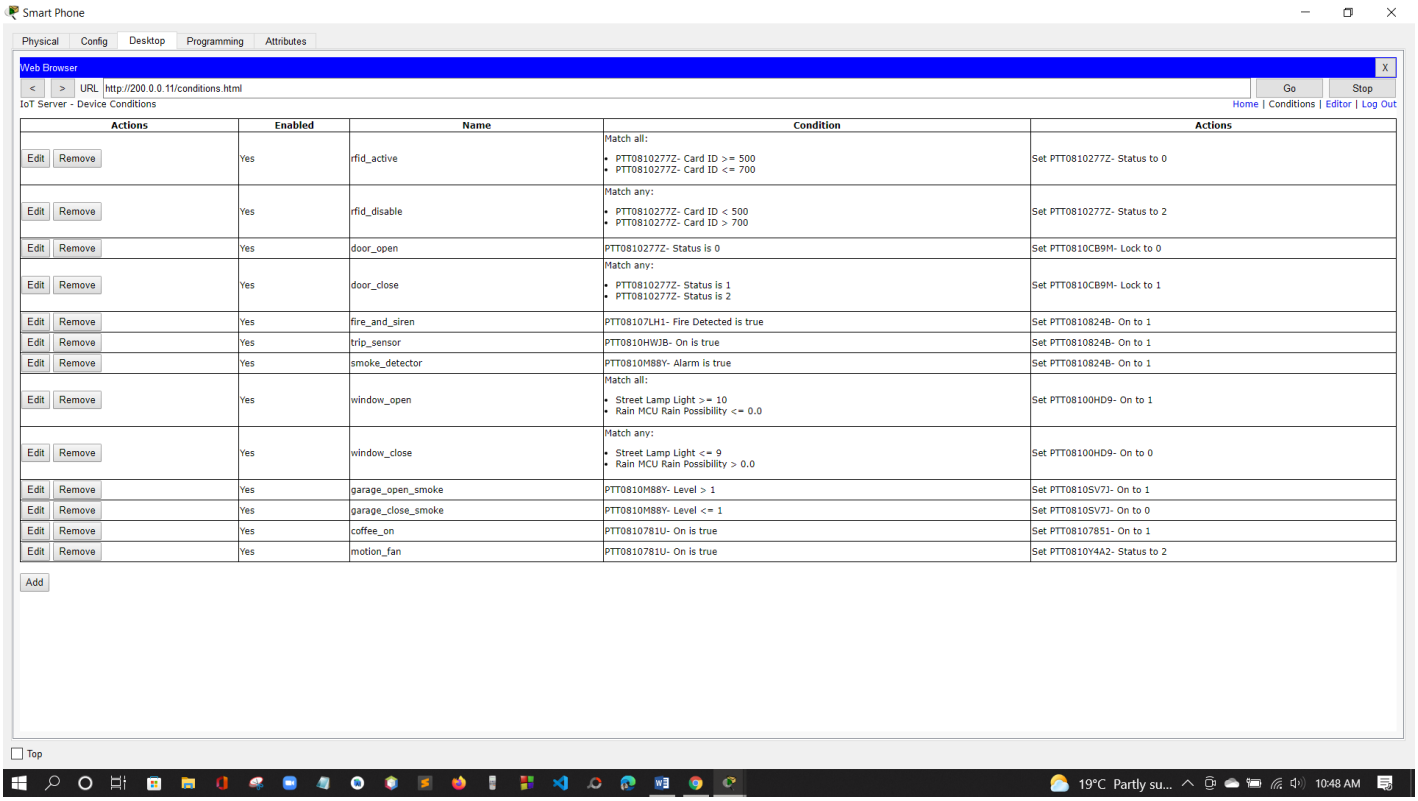


Figure: All IoT Devices

The condition of all IoT devices are



Chapter-4

Conclusion

4.1 Introduction: This project is so helpful for any kind of office to manage smartly from office and outside office by Mobile application. As I am focusing on only IoT devices I can't work for any other work in here like More PC, Server, and Phone Connection etc. This project is all about IoT devices use and control automation.

4.2 Practical Implications: We can easily implement any kind of office but it will be design for a Software Base company Office.

4.3 Scope of Future Work: Now a days there are many scope to implementation or use this technology for any kind of office as well as house also. In future we can upgrade our logical condition in software and hardware part and do more flexible and smooth life for all of us.

References

1. https://github.com/engineertorikulislam/Computer_Network_Project
2. https://unsplash.com/ (Background Image)
3. https://cse.final-year-projects.in/ (Project Idea)