Arwa salah saad 20201032042 Jailan refaat elsayed 20201310934 Sara Mohamed Yassin 20201383064

<u>Smart hotel project</u>

Abstract:

Our goal is to establish hospitality by offering guests control, Security and convenience by integrating cutting-edge technologies. Hotel has 3 floors ,First floor(Outdoor, Lobby) ,Second floor(bathroom,corridor) and Third floor(Hotel room,Kitchen).Our Smart Hotel allow guests to open door (with RFID card (hotel card), motion sensor and button),open AC, window,coffee machine from smartphone ,when the guest get to the bathroom cabinet the water open and drain opens to get water out ,can open the hood with button, grass man to open water sprinkler and close automatic with specific water level, garage open when sensor sense motion of the car,when there is a thief and get around the locker the siren will make then web camera record to recognize the thief,detection of smoke when it sense smoke it will open siren ,fire detection when there is fire the sprinkler and the emergency door will open,leds of the corridor open when someone walk.

Keywords:

- Automation
- Hotel
- Fire detection
- Motion Detection
- Safety Monitoring
- Smart Control

1. Introduction

A smart hotel integrates advanced technology to enhance guest experiences and streamline operations. From personalized room settings controlled via apps, smart hotels prioritize convenience, efficiency, and customization for guests.

For Guests: Smart hotels offer features like controlling room settings (lighting, temperature) through their phones or with voice commands, allowing for a more comfortable and convenient stay. Guests can also use apps to order room service, access hotel amenities, or get recommendations for local attractions.

For Hotel Management: Smart hotels allow for real-time monitoring of energy use, room occupancy, and other factors that can help optimize operations and reduce costs. They can also use data from guest preferences to personalize their stay.

2. Related works:

To build our project we gained knowledge from the coursework and lab projects and use of those information to start implementing .
Learned how to do wireless connection by connecting Home gateways with smartphones or tablets, how to add and connect sensors, actuators with microcontroller by program it with a specified code.

Around 70% of our work relied on labs and 30% on google search for other related work but we couldn't find anything relevant to Hotels, we found just advanced topics such as designing and configuring dynamic routing protocols like RIP and OSPF for a Smart Hotel environment.

In contrast to previous studies which mainly focused on designing smart hotels ,the focus of our project is to model and simulate an IoT system for a smart hotel . Different IoT devices are connected and addressed in such a way to allow the transmission of valuable information. In addition to that, the IoT devices are connected using home gateway to the Internet in order to be controlled remotely elsewhere by smart phone.

3. Methodology:

Hardware Setup:

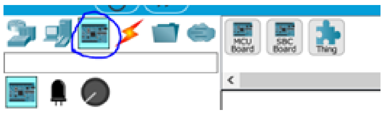
Gather the necessary hardware components Located down in Cisco PT:

- Choose Network Devices > Wireless Devices > Home Gateway
- **Components :**
    - Boards > MCU
    - Sensors > Motion sensor, Push Button, Toggle push button, Rocker switch, Potentiometer
    - Actuators > LED, ceiling sprinkler, floor sprinkler, Heating element
- **End Devices:**
    - End Devices> Smart Device, WIreless Tablet
    - Home > AC, coffee machine (appliance) , Door, garage door, lawn sprinkler, Motion Detector, Smoke Detector, Siren, Thermostat, water drain, water level monitor, Webcam, Window
    - Smart City > Old Car, RFID card, RFID reader
    - Industrial > Fire Monitor, Fire Sprinkler, Trip sensor
- **Connections:** IOT custom cable

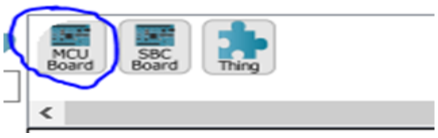
Adding the Necessary Devices:

MCU:

- Click on Components

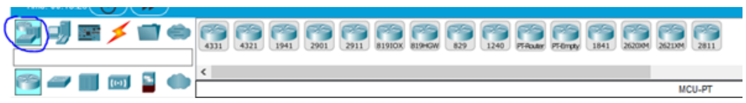


- The choose MCU-PT board

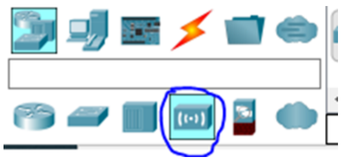


Home gateway:

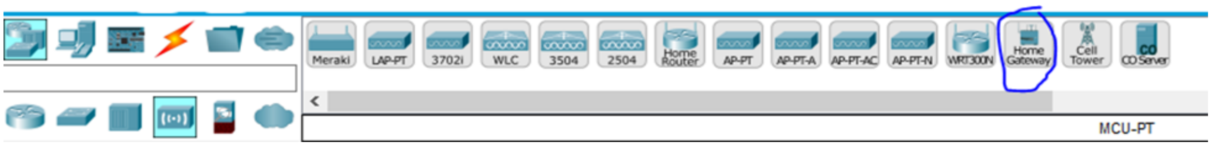
- Click on Network Devices



- Click on Wireless devices

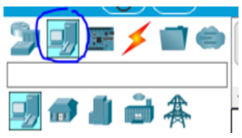


- Choose DLC100

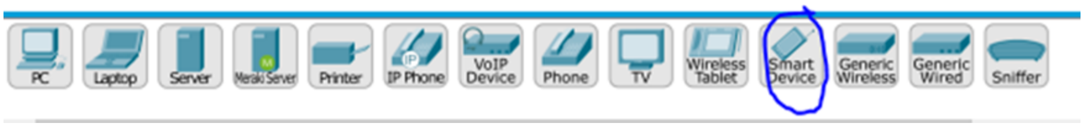


Smart Phone:

- Click on End Devices

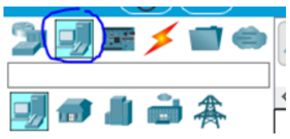


- Click on SMARTPHONE-PC

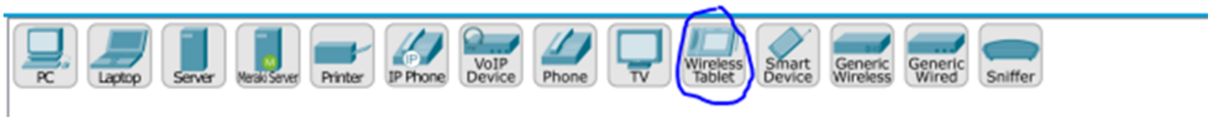


Tablet:

- Click on End Devices



- Click on Tablet Pc-pt



Software Setup:

How to connect MCU with devices?

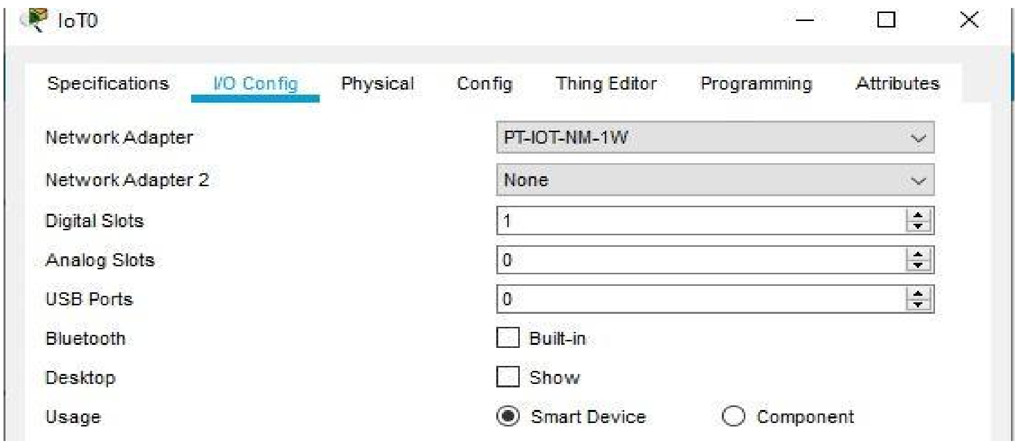
Steps:

first: choose the board of microcontroller (MCU-PT)
second: Place every device we need in the smart building to connect it to the controller
third: connect the components
fourth: run the program

How to connect Wi-Fi to devices?

Steps:

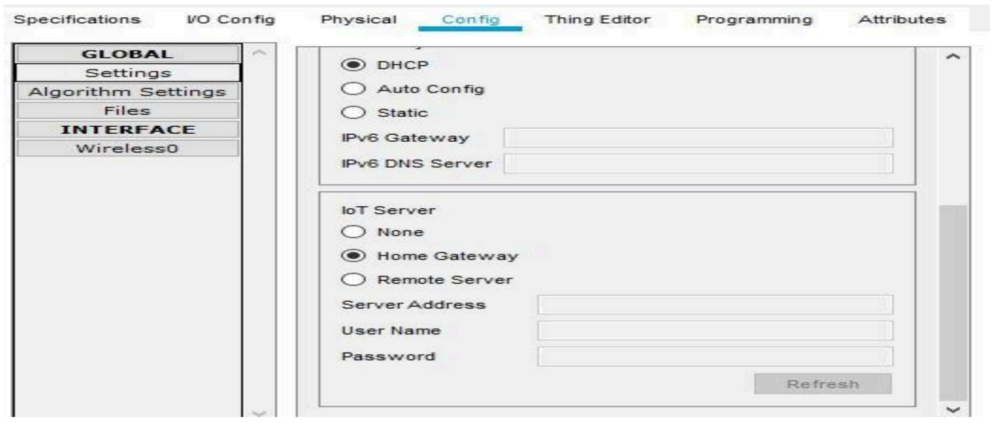
- First: click on the device then click on advanced from I/O config change (network Adapter- choose PT-IOT-NM-1W)


IoT0

Specifications | I/O Config | Physical | Config | Thing Editor | Programming | Attributes

Network Adapter	PT-IOT-NM-1W
Network Adapter 2	None
Digital Slots	1
Analog Slots	0
USB Ports	0
Bluetooth	☐ Built-in
Desktop	☐ Show
Usage	[✓ Smart Device] ○ Component

- Second: Press Config tab and select home Gateway radio button to control it with your smartphone via gateway.



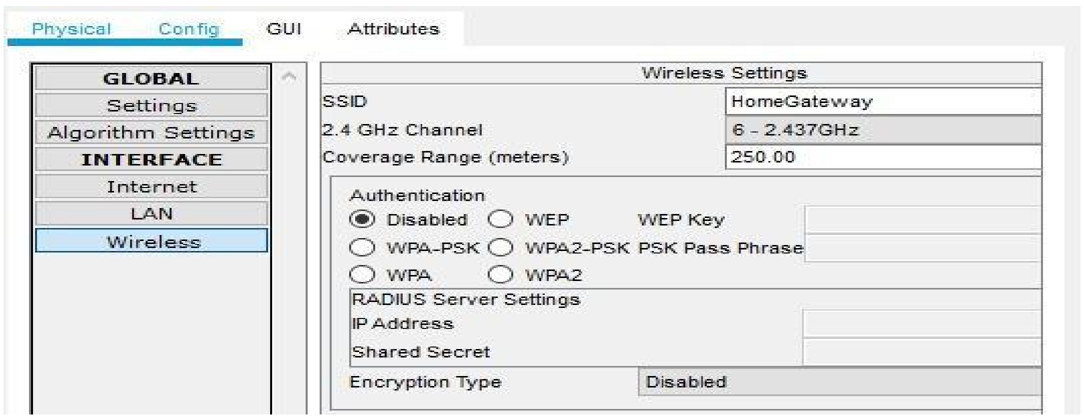
Specifications | I/O Config | Physical | Config | Thing Editor | Programming | Attributes

GLOBAL
Settings
Algorithm Settings
Files
INTERFACE
Wireless0

[✓ DHCP]
Auto Config
Static
IPv6 Gateway
IPv6 DNS Server

IoT Server
None
[✓ Home Gateway]
Remote Server
Server Address
User Name
Password
Refresh

- Third: Click on the home gateway, Copy Service Set Identifier (SSID) of gateway and click on Smartphone to configure mobile to gateway.
  Change SSID in wireless.



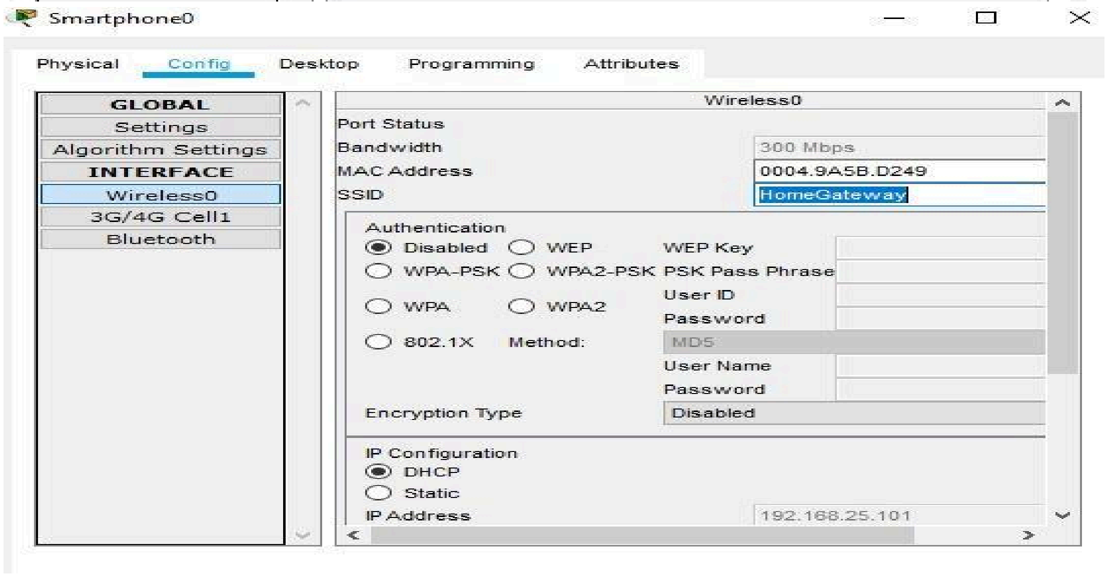
Physical | Config | GUI | Attributes

GLOBAL
Settings
Algorithm Settings
INTERFACE
Internet
LAN
Wireless

Wireless Settings

SSID	HomeGateway
2.4 GHz Channel	6 - 2.437GHz
Coverage Range (meters)	250.00

Authentication
[✓ Disabled] WEP WEP Key
WPA-PSK WPA2-PSK PSK Pass Phrase
WPA WPA2
RADIUS Server Settings
IP Address
Shared Secret
Encryption Type: Disabled



Smartphone0

Physical | Config | Desktop | Programming | Attributes

GLOBAL
Settings
Algorithm Settings
INTERFACE
Wireless0
3G/4G Cell1
Bluetooth

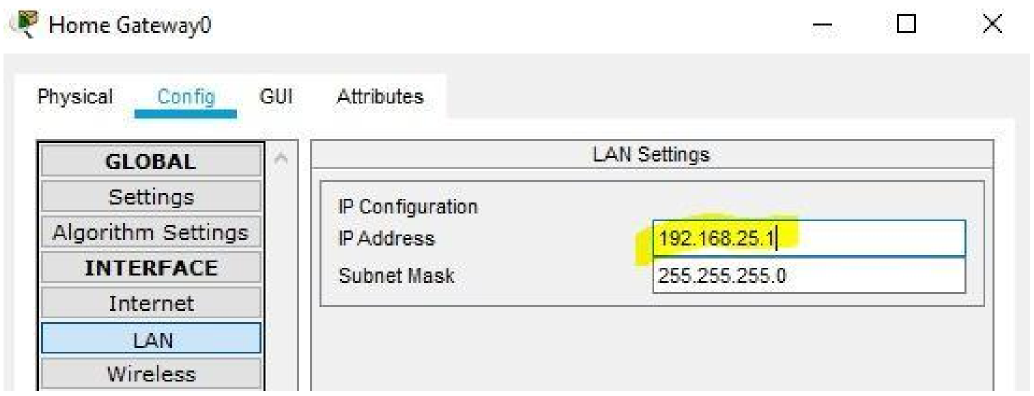
Wireless0

Port Status	
Bandwidth	300 Mbps
MAC Address	0004.9A5B.D249
SSID	HomeGateway

Authentication
[✓ Disabled] WEP WEP Key
WPA-PSK WPA2-PSK PSK Pass Phrase
WPA WPA2 User ID
Password
802.1X Method: MD5
User Name
Password
Encryption Type: Disabled

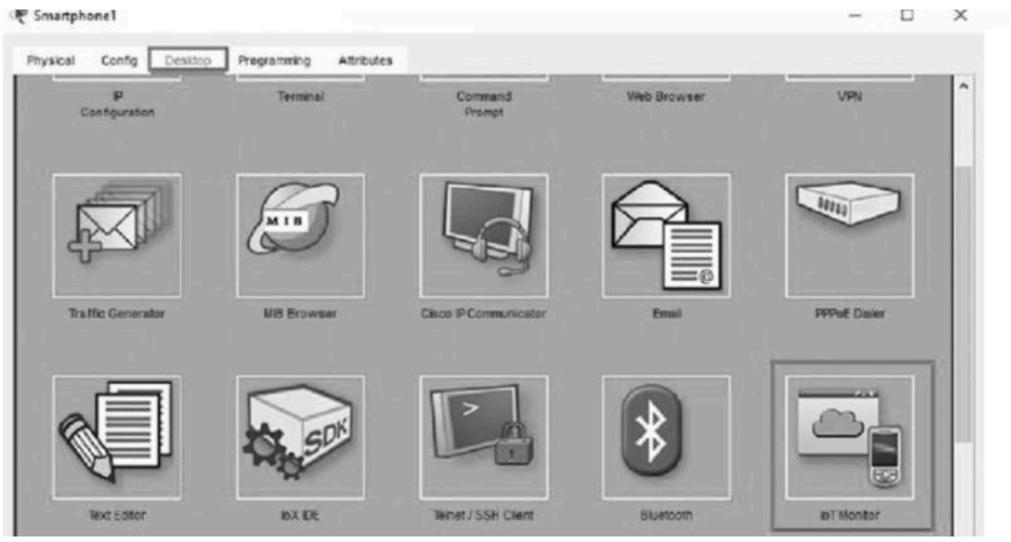
IP Configuration
[✓ DHCP]
Static
IP Address: 192.168.25.101

- Fourth: Configure IP Address of Smart Phone with Gateway.

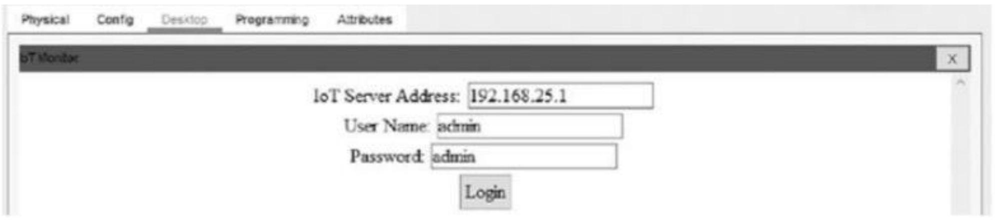


IP Configuration
IP Address 192.168.25.1
Subnet Mask 255.255.255.0

- Fifth click on Smartphone >Desktop and choose iot monitor



- Sixth: Login screen will appear. Provide IoT server address, which should be the IP address of gateway.



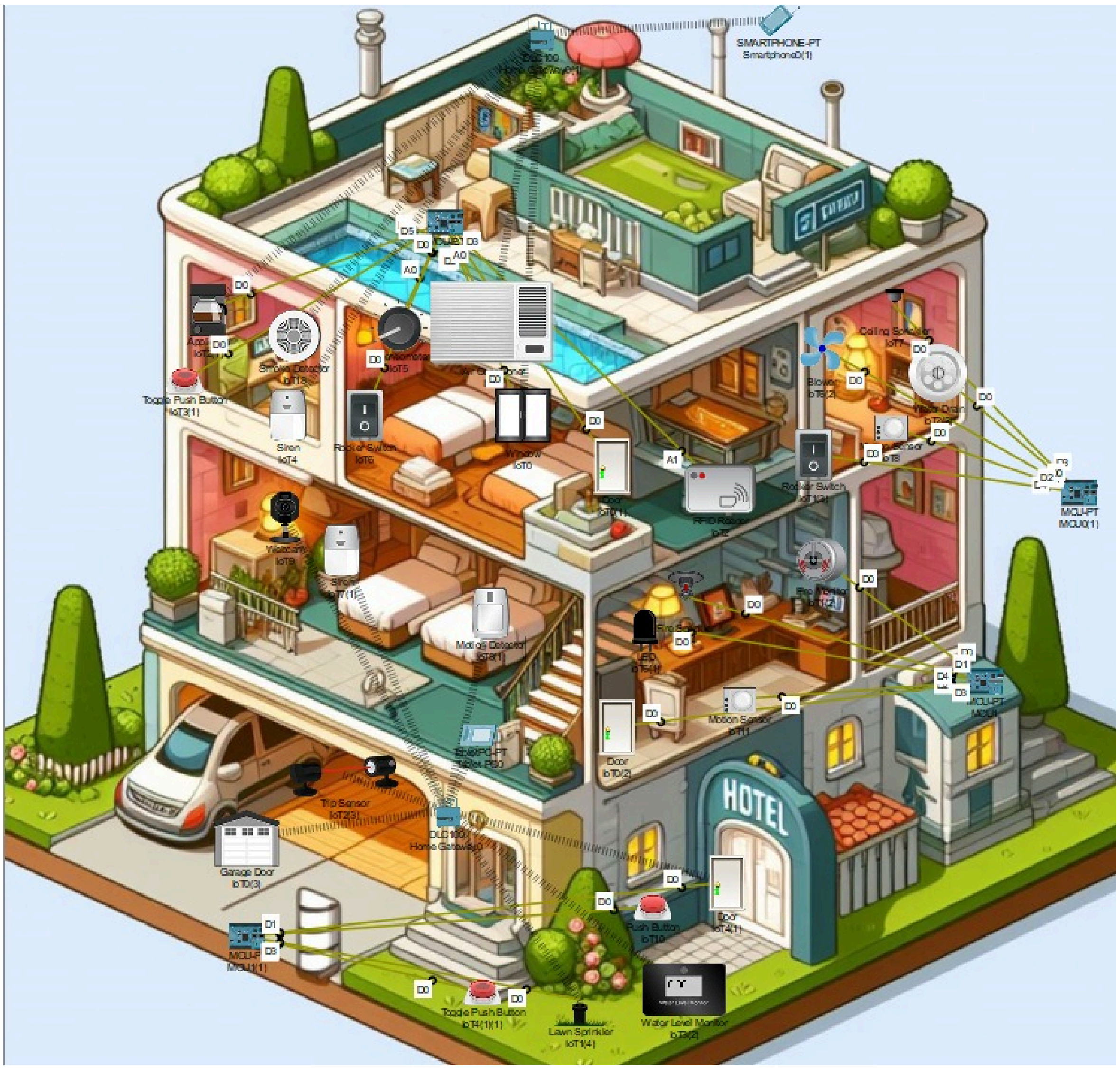
IoT Server Address: 192.168.25.1
User Name: admin
Password: admin

- Seventh: Then click on the button to open and close the device.

4. Implementation and Results:

Project:

This picture shows the scenario of a designed and simulated smart hotel.The smart hotel includes Air conditioner, smart led ,smart door, smart fan, smoke detector, smart coffee maker, garage door, water level monitor,Siren,water sprinkler,drain,window. As shown in the Figure, all IoT devices are connected to the home gateway.



<u>1- Main room:</u>

- ➔ actuator: Air Conditioner
- ➔ actuator: Window
- ➔ sensor: Potentiometer : is an analogue **sensor**
- ➔ sensor: Rocker Switch

Steps:

1- Air conditioner (pin 2) and Potentiometer

has range (0-225) (A1) connected to MCU with IOT cable.

Action

when hotel guest scrolls it to more than 19 the AC Will turn on otherwise off

2-Window (pin 1) and Rocker switch (pin0) connected to MCU with IOT cable

Action:

when hotel guest clicks on Rocker switch the window will open then click off window will close

Components:

- ➔ actuator: Door
- ➔ sensor: RFID READER:Read the ID of an RFID Card
- ➔ RFID Card:has ID on the card

Steps:

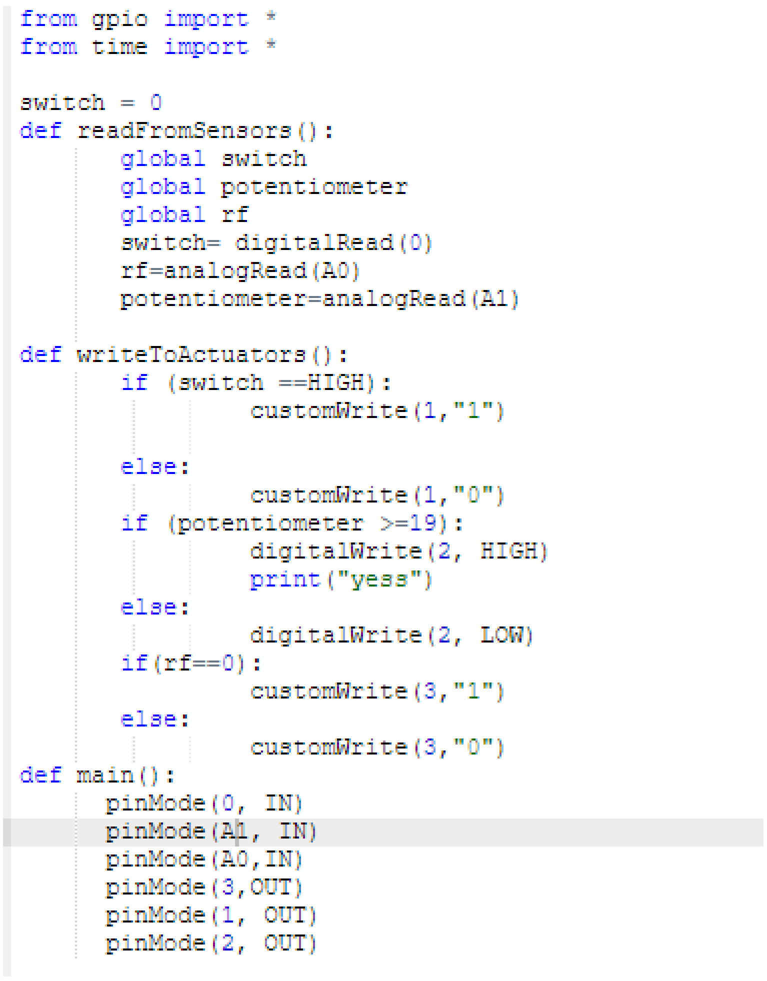
3-Door (pin3) and RFID READER (A0)

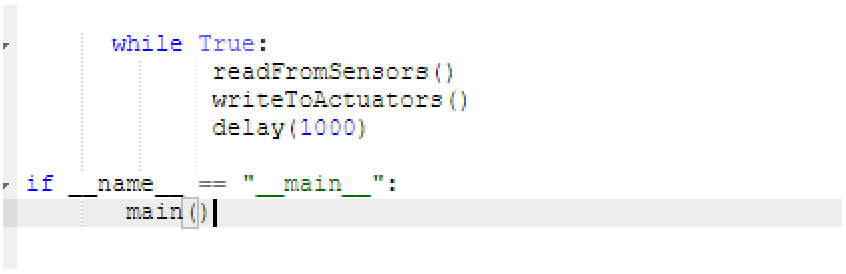
Connected to MCU with IOT cable.

Action:

When a hotel guest moves the RFID card of the room on RFID READER It will be Green then the door of the room will open.

code:


```
from gpio import *
from time import *

switch = 0
def readFromSensors():
        global switch
        global potentiometer
        global rf
        switch= digitalRead(0)
        rf=analogRead(A0)
        potentiometer=analogRead(A1)

def writeToActuators():
        if (switch ==HIGH):
                customWrite(1,"1")

        else:
                customWrite(1,"0")
        if (potentiometer >=19):
                digitalWrite(2, HIGH)
                print("yess")
        else:
                digitalWrite(2, LOW)
        if(rf==0):
                customWrite(3,"1")
        else:
                customWrite(3,"0")
def main():
        pinMode(0, IN)
        pinMode(A1, IN)
        pinMode(A0,IN)
        pinMode(3,OUT)
        pinMode(1, OUT)
        pinMode(2, OUT)
```


```
        while True:
                readFromSensors()
                writeToActuators()
                delay(1000)

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

Explanation:

Imports:

- import necessary modules GPIO for working with pins (input/output) and time-related functions (delay, sleep).

Variables:

- switch = 0: Declares a global variable switch initialized to 0

Functions:

readFromSensors():

- Reads the switch state connected to pin 0 using digitalRead(0).
- Reads the RF value from analog pin A0 using analogRead(A0).
- Reads the potentiometer value from analog pin A1 using analogRead(A1). Updates the (implicit) global potentiometer variable with the reading (0-1023, depending on the ADC resolution).

writeToActuators():

- Checks the switch value:
  - If the switch is high (1) (assuming switch on activates), it calls customWrite(1, "1") to send a signal ("1"?) to an actuator on pin 1 using a custom function customWrite.If switch is low (0) (assuming switch off deactivates), it calls customWrite(1, "0") to send a signal ("0"?) to the actuator.
- Checks the potentiometer value:
  - If the potentiometer is greater than or equal to 19 (a threshold value), it turns on an LED (or actuator) connected to pin 2 using digitalWrite(2, HIGH). It also prints "yes" to the console.
  - If the potentiometer is less than 19, it turns off the LED (or actuator) connected to pin 2 using digitalWrite(2, LOW).
- Checks the RFID value:
  - If the RF equal zero then door will open(pin 3) .
  - otherwise,door will be closed .

Main Loop :

- Sets pin 0 as input (pinMode(0, IN)) to receive switch data.
- Sets pin A1 as input (pinMode(A1, IN)) to receive switch data.
- Sets pin A0 as input (pinMode(A0, IN)) .
- output (pinMode(1, OUT))to make an act.
- set pin 3 as output ,pinMode(3, OUT).
- Sets pin 2 as output (pinMode(2, OUT)) to control the LED (or actuator).
- The while True loop continuously executes:
  - Calls readFromSensors() to update switch and potentiometer with sensor readings.
  - Calls writeToActuators() to check sensor values and control the LED/actuator based on switch and potentiometer readings.
  - Calls delay(1000) (implementation might vary) to pause for 1 second.

2-Outdoor room(garage and garden):

components:

- ➔ Sensor: Toggle Push Button
- ➔ Actuator: Lawn Sprinkler
- ➔ Sensor: Water level Monitor

Steps:

Connect Toggle button (pin0) to sense and Lawn sprinkler (pin1) to make the action(spread water) to the MCU

Action:

When the grass man want to water the grass he click on the toggle push button to open lawn sprinkler there is three options to close lawn sprinkler

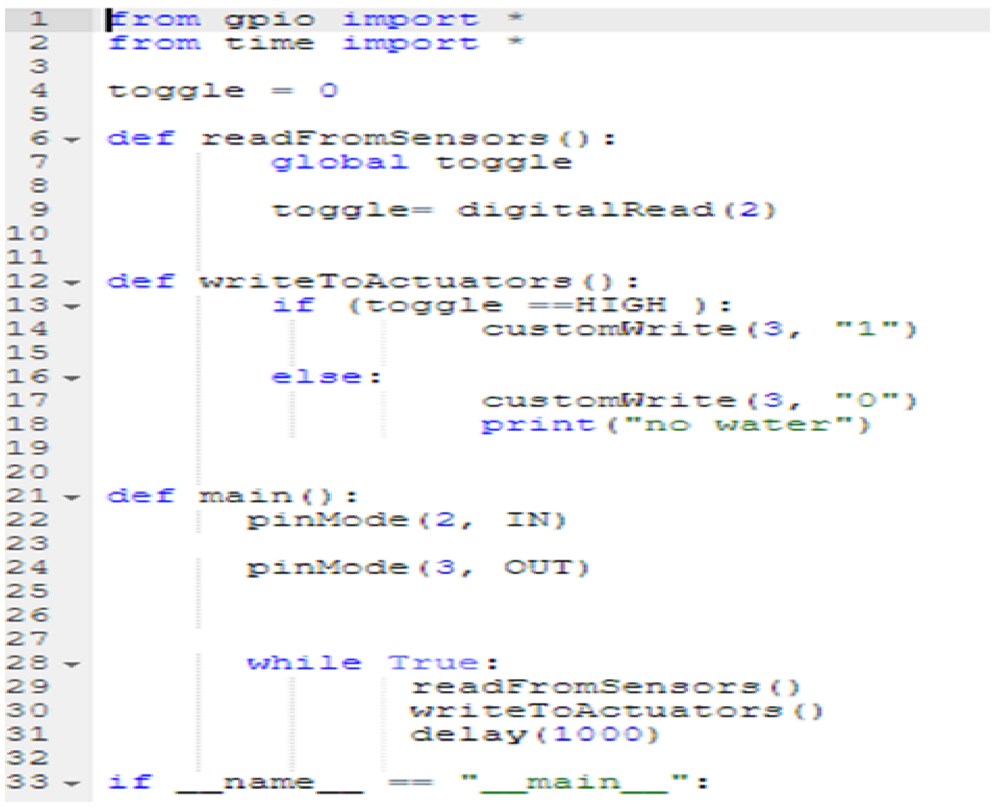
1- from toggle push button

2-From water level monitor that is used to measure the water level (connected to Home gateway)

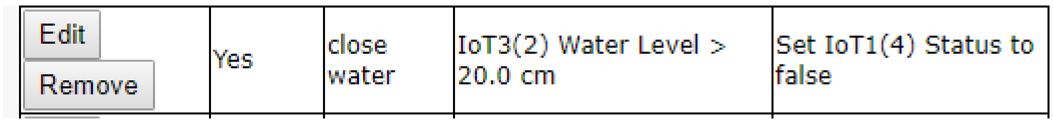
I put condition when the sprinkler is open and water level exceeds 10 cm (known from water monitor it will close the lawn Sprinkler)

3-Open and Close from TabletPC-PT connected to Home gateway

code:



```
from gpio import *
from time import *

toggle = 0

def readFromSensors():
        global toggle

        toggle= digitalRead(2)


def writeToActuators():
        if (toggle ==HIGH ):
                customWrite(3, "1")

        else:
                customWrite(3, "0")
                print("no water")


def main():
      pinMode(2, IN)

      pinMode(3, OUT)



      while True:
              readFromSensors()
              writeToActuators()
              delay(1000)

if __name__ == "__main__":
```



Edit Remove	Yes	close water	IoT3(2) Water Level > 20.0 cm	Set IoT1(4) Status to false

Imports:

- from gpio import *: This line tries to import everything from a gpio module
- from time import *: This imports all functions from the time module, potentially including sleep() for pausing, time() for current time, and delay() for delays

Variables :

- toggle = 0: Declares a global variable toggle initialized to 0.

Functions:

- readFromSensors():
  - reads the sensor state connected to pin 2 using digitalRead(2).
  - Updates the global toggle with the sensor reading (0 or 1).
- writeToActuators():

  hecks the value of toggle:

  - If the toggle is high (1) (possibly indicating water), it calls customWrite(3, "1") to send a signal ("1"?) to an actuator on pin 3 using a custom function customWrite. The details of customWrite depend on the specific hardware being controlled.
  - If the toggle is low (0), it prints "no water" to the console.

Main Loop :

- Sets pin 2 as input (pinMode(2, IN)) to receive sensor data.
- Sets pin 3 as output (pinMode(3, OUT)) to send signals to an actuator.
- The while True loop continuously executes:
  - Calls readFromSensors() to update toggle with the sensor state.
  - Calls writeToActuators() to check toggle and send signals or print messages.
  - Calls delay(1000) (implementation might vary) to pause for 1 second.

Conditional Execution (if __name__ == "__main__":):

- Ensures main() runs only when the script is executed directly.

Components:

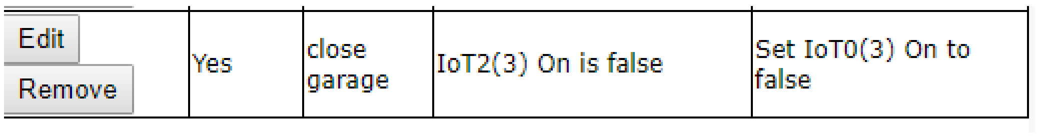
- ➔ Actuator :Garage Door
- ➔ Sensor:Trip Sensor

Steps:

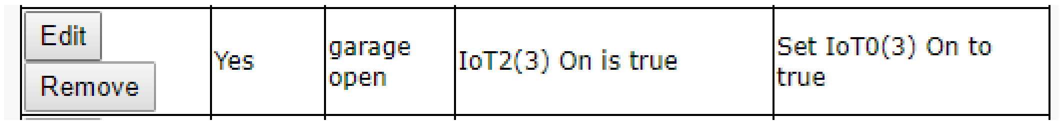
Garage and tripWire with Homegateway (wireless connection that can be managed from tablet)

Action :

When a car moves into trip wire sensor (motion sensor) the garage door will open



Edit Remove	Yes	close garage	IoT2(3) On is false	Set IoT0(3) On to false



Edit Remove	Yes	garage open	IoT2(3) On is true	Set IoT0(3) On to true

3- bathroom:

Components:

- ➔ sensor: motion sensor: digital sensor to detect nearby people(pin0)
- ➔ Actuator: Water drain: digital actuator to suck up and drain the water (pin3)
- ➔ Actuator: ceiling sprinkler: placed it to represent a shower (pin2)

Steps:

Connect sensor pin0 to MCU to be ready for sense the motion, connect the actuator pin2 and 3 to the MCU to interact with their sensor

Action:

When someone comes under the shower (ceiling sprinkler), it opens the water and water drains to suck out the water.

Components:

- ➔ board: MCU
- ➔ sensor: rocker switch: digital button for operation that can be pressed on either end to connect or disconnect an electrical circuit (pin1)
- ➔ Actuator: blower: increase airflow like a bathroom hood (pin4)

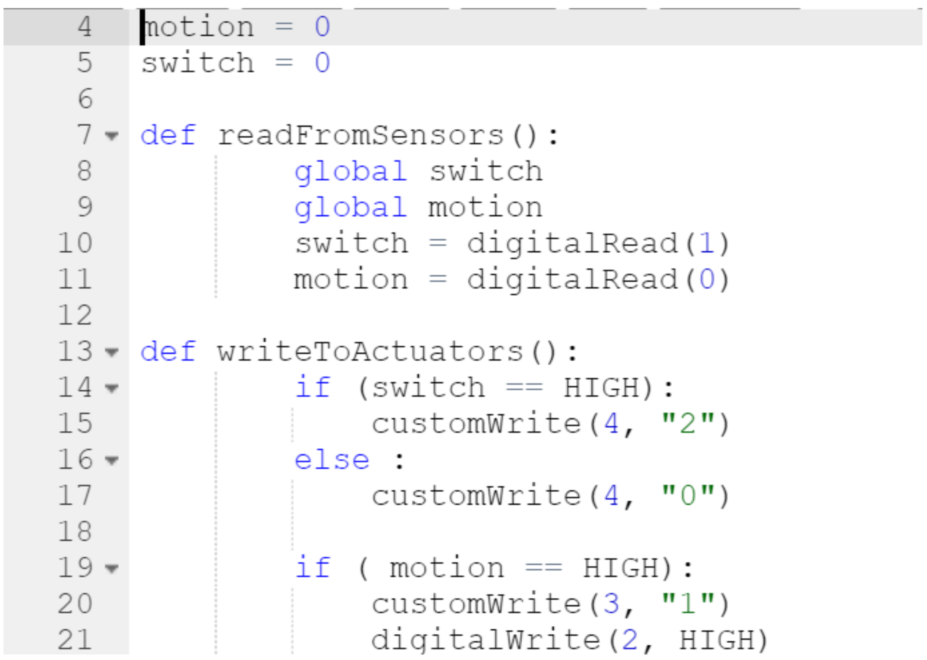
Steps:

Connect the switch pin1 to be ready for pressed to turn on and off the hood, connect the hood to the pin4 to do its job

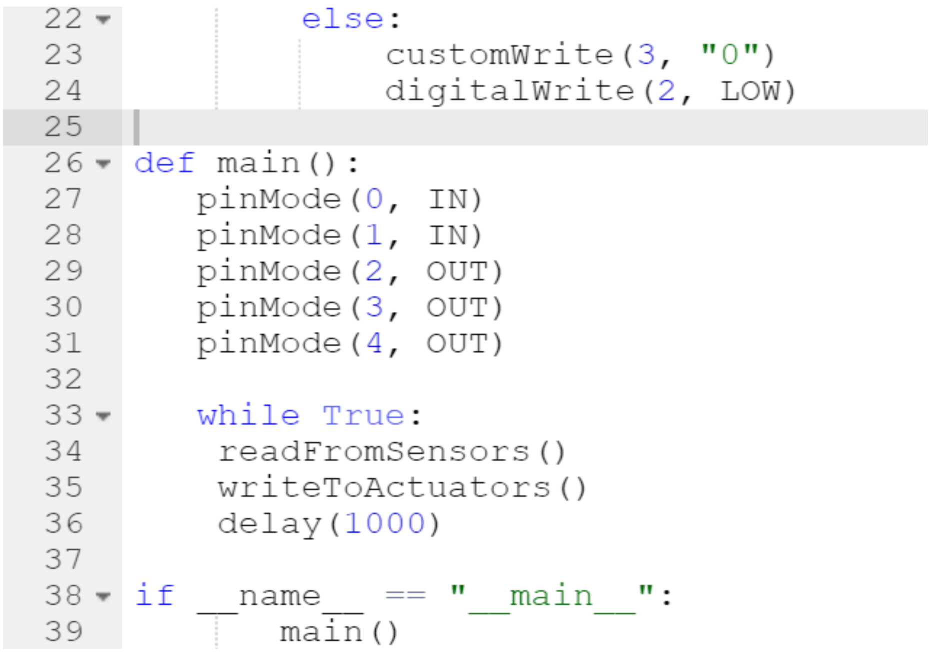
Action:

person can open and close the hood through the switch

Code:



```
motion = 0
switch = 0

def readFromSensors():
        global switch
        global motion
        switch = digitalRead(1)
        motion = digitalRead(0)

def writeToActuators():
        if (switch == HIGH):
            customWrite(4, "2")
        else :
            customWrite(4, "0")

        if ( motion == HIGH):
            customWrite(3, "1")
            digitalWrite(2, HIGH)
```



```
        else:
            customWrite(3, "0")
            digitalWrite(2, LOW)

def main():
    pinMode(0, IN)
    pinMode(1, IN)
    pinMode(2, OUT)
    pinMode(3, OUT)
    pinMode(4, OUT)

    while True:
     readFromSensors()
     writeToActuators()
     delay(1000)

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

Imports:

- imports necessary functions from the `gpio` module and the `time` module.

Variables:

- motion: Stores the state of the motion sensor connected to pin 0. Initialized to 0.
- switch: Stores the state of the switch connected to pin 1. Initialized to 0.

Functions:

- readFromSensors(): Reads the state of the switch (pin 1) and the motion sensor (pin 0) and updates the corresponding variables.
- writeToActuators(): Controls the actuators based on the states of the switch and motion sensor.
  - If the switch is ON (HIGH state), it activates the blower (pin 4) by setting it to HIGH.
  - If the motion sensor detects motion (HIGH state), it activates the ceiling sprinkler (pin 2) and water drain (pin 3) by setting them to HIGH. Otherwise, it deactivates them by setting them to LOW.

main loop :

- The `main()` function sets the pin modes (pin 0 and pin 1 as input, and pin 2, pin 3, and pin 4 as output) and enters an infinite loop where it repeatedly reads from sensors and writes to actuators with a delay of 1000 milliseconds (1 second).

4-lobby:

Components:

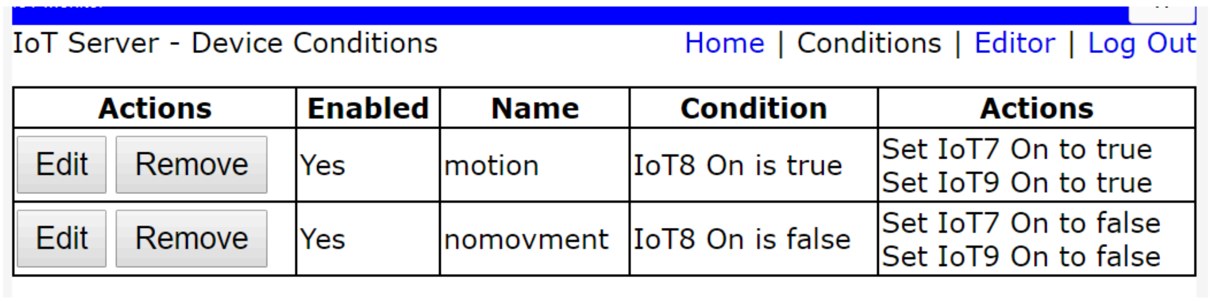
- ➔ end device: tablet: To connect devices to the tablet so that they can access Wi-Fi and control them through it
- ➔ Sensor: motion detector
- ➔ Actuator: Siren: to alert those who are outside that something dangerous is approaching (thief approaches the safe to steal important items)
- ➔ Actuator: Web cam: record to a computer (When there is a theft to identify the thief)

Steps:

connected the devices to a tablet here, as we mentioned the method above, set conditions for the devices to interact and function properly

We have 2 conditions:

- The first condition is true: it reacts when it senses the presence of a thief's movement near the safe. It raises the siren and activates the camera to record.
- As for the second, it does the opposite. When there is no movement near the safe, it does not activate anything.


IoT Server - Device Conditions — Home | Conditions | Editor | Log Out

Actions	Enabled	Name	Condition	Actions
Edit Remove	Yes	motion	IoT8 On is true	Set IoT7 On to true Set IoT9 On to true
Edit Remove	Yes	nomovment	IoT8 On is false	Set IoT7 On to false Set IoT9 On to false

Action:

When the thief approaches the safe, there is a detector detect his movement, the siren lights up and makes a sound, and webcam records a video, There is a tablet that controls these actuators as well

Components:

Sensor: push button: digital simple switch mechanism to control some aspect of a machine or a process (pin0)

Actuator: door (pin1)

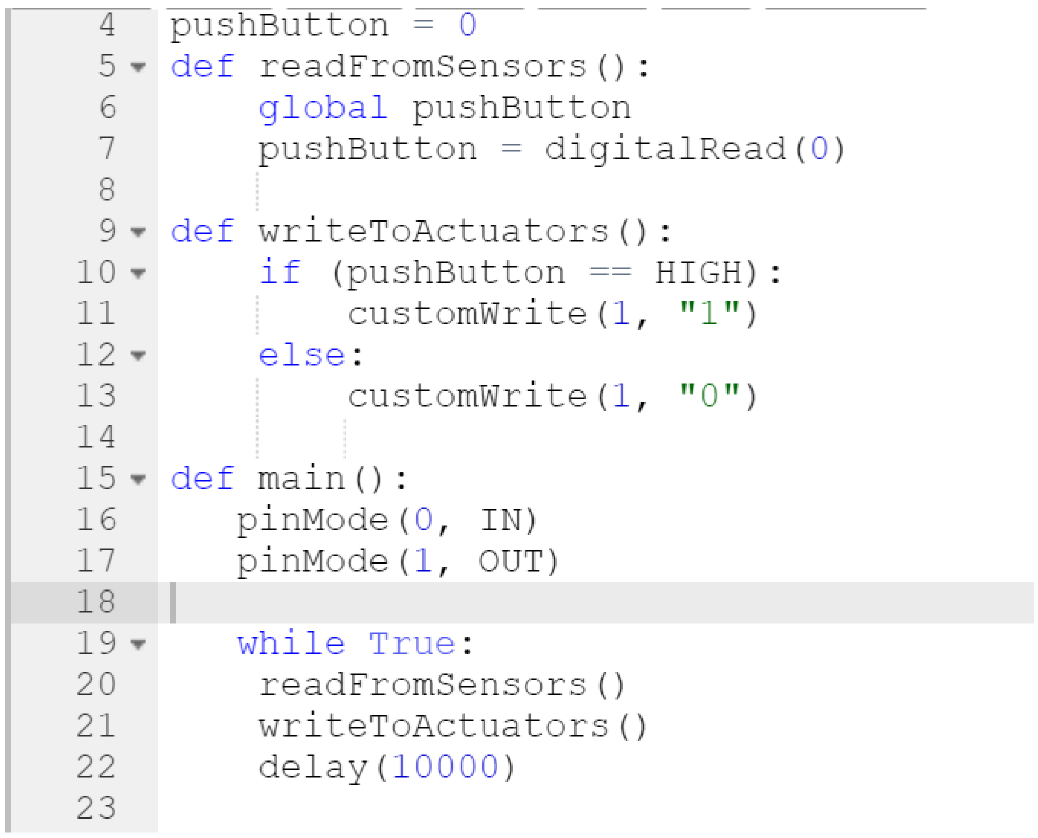
Steps:

connect the tablet to the door for workers to control it easily , Connect the push button pin0 to be ready for pressed , connect the door pin1 to open and close it through the push button

Action:

the workers can open the workers' door using a tablet in front of the door, or we can open it by pressing the button when the tablet malfunctions

Code:



```
pushButton = 0
def readFromSensors():
    global pushButton
    pushButton = digitalRead(0)

def writeToActuators():
    if (pushButton == HIGH):
        customWrite(1, "1")
    else:
        customWrite(1, "0")

def main():
   pinMode(0, IN)
   pinMode(1, OUT)

   while True:
    readFromSensors()
    writeToActuators()
    delay(10000)
```

Imports:

- imports necessary functions from the `gpio` module and the `time` module.

Variables:

- pushButton: This variable is used to store the state of the push button. It is initialized to 0.

Functions:

- readFromSensors(): This function reads the state of the push button connected to pin 0 and updates the `pushButton` variable accordingly.
- writeToActuators(): This function controls the door actuator connected to pin 1 based on the state of the push button. If the push button is pressed (HIGH state), it sends a signal to pin 1 to activate the door (set it to HIGH). Otherwise, it deactivates the door (sets it to LOW).

Main loop:

- main(): This is the main function of the script. It sets the pin modes (pin 0 as input and pin 1 as output) and enters an infinite loop where it repeatedly reads from sensors and writes to actuators with a delay of 10000 milliseconds (10 second).

Execution:

- The script checks if it's being run directly (`if __name__ == "__main__":`) and if so, it calls the `main()` function to start the program execution.

5-kitchen:

components:

- ➔ board: MCU
- ➔ sensors: Toggle push button
- ➔ Actuators: Coffee machine
- ➔ End devices: smart phone

steps:

Connect the toggle push button and coffee machine with the microcontroller by IOT cable, coffee machine on pin D4 and the toggle button on pin D5 all connected with mcu pin D0. also connect the coffee machine with gateway using smartphone by wireless connection to control it with another way.

Action:

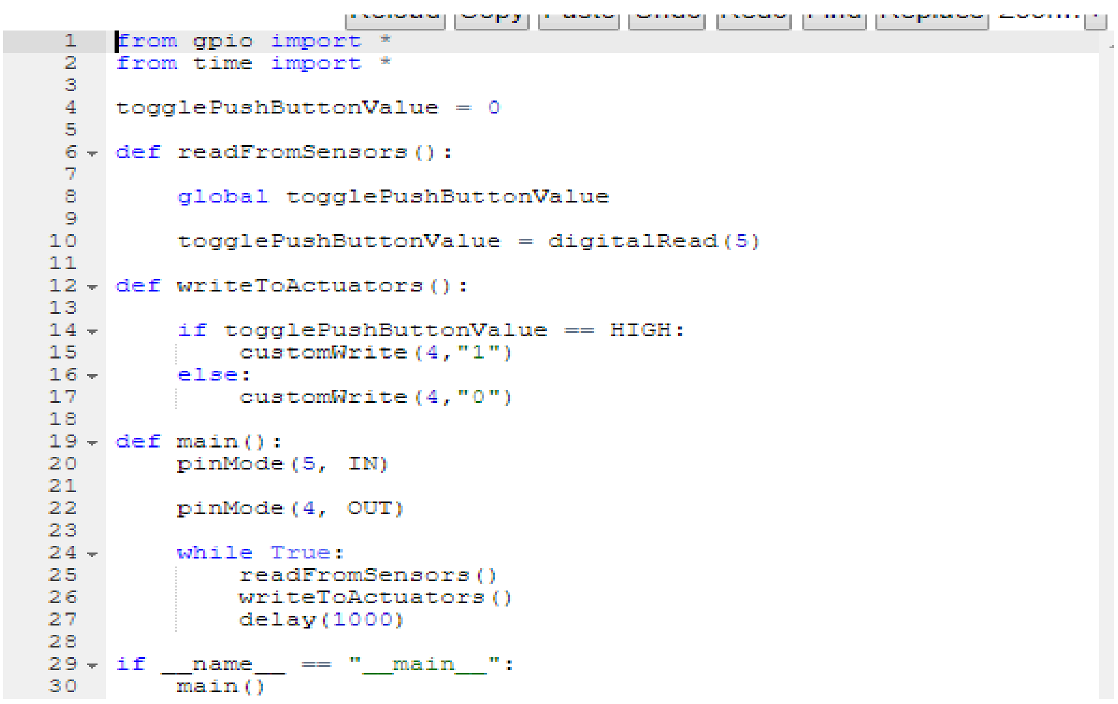
to control turning the coffee machine on/off by pushing and releasing the button

if the button (input pin) is HIGH then set pin4 to "1/on" and else (LOW), set the actuator pin4 to "0/off"

Also, if the guest wants to make coffee remotely, he can easily do that using the smartphone of the hotel room.

program the MCU by clicking on it > choose programming > add new (empty python) > then add the code

code:



```
from gpio import *
from time import *

togglePushButtonValue = 0

def readFromSensors():

    global togglePushButtonValue

    togglePushButtonValue = digitalRead(5)

def writeToActuators():

    if togglePushButtonValue == HIGH:
        customWrite(4,"1")
    else:
        customWrite(4,"0")

def main():
    pinMode(5, IN)

    pinMode(4, OUT)

    while True:
        readFromSensors()
        writeToActuators()
        delay(1000)

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

Imports:

- import necessary modules GPIO for working with pins (input/output) and time-related functions (delay, sleep).

Variables :

- Initialize the Toggle pushbutton with 0 to store the state of the push button.

Functions:

- define readFromSensor() function reads the state of a sensor connected to pin 5 and updates the togglePushButtonValue.
- Function WriteToActuator() : Control the actuator when the button is pressed (the value is HIGH), it uses the customWrite() function from the to control an actuator connected to pin 4, setting it to "1" (ON).
- If the button is not pressed (the value is LOW), it sets the actuator to "0" (OFF).

Main Loop:

- Pin 5 is set as an input pin (IN), which is connected to the toggle push button.
- Pin 4 is set as an output pin (OUT), which controls the actuator.
- Call the functions in the loop to continuously read the state of the sensor, then delay the program for 1 second.

Components:

➔ End devices: Smart phone
➔ Sensors: Smoke detector

- ➔ Actuators: Siren

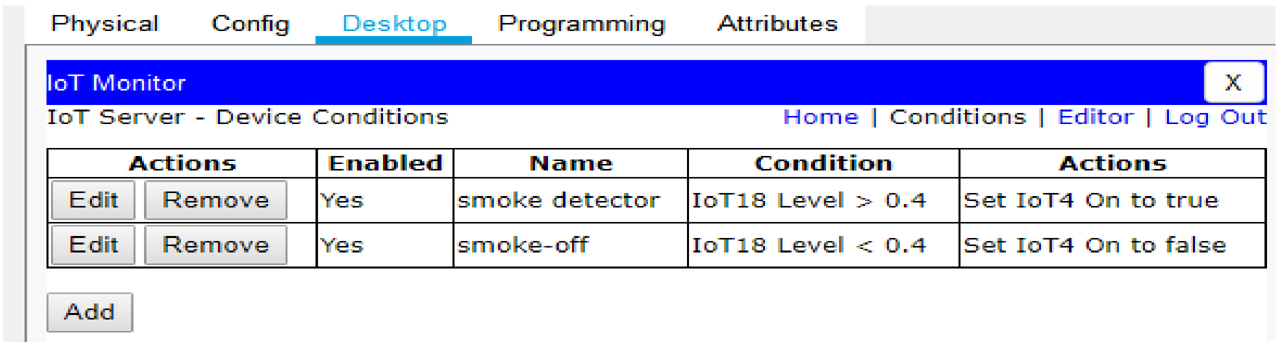
Steps:

Set the configurations (Edit IOT server) of siren, smoke detector and smartphone to connect them wireless with home gateway. After clicking on smartphone and choosing IOT monitor to display the element's buttons, choose add conditions in smartphone to allow the sensor start under those conditions:

1. if the smoke level is greater than > 0.4, set the siren on to true (HIGH).
2. and if level is less than < 0.4, then set the siren on to false (LOW).

IOT18 : smoke detector

IOT4 : siren



IoT Server - Device Conditions

Actions	Enabled	Name	Condition	Actions
Edit Remove	Yes	smoke detector	IoT18 Level > 0.4	Set IoT4 On to true
Edit Remove	Yes	smoke-off	IoT18 Level < 0.4	Set IoT4 On to false

Action:

Enhance safety measures in the hotel kitchen to detect any signs of smoke, when someone is cooking and forgets the food in oven or on gas or anything leads to carbon dioxide.

The system triggers a siren alert to be high and notifies hotel guests and staff of the danger to quickly fix the problem and avoid leading to fire.

using the old car to spread carbon "smoke" in the air to test the system, when the detector catches a smoke the smoking level starts increasing until it reaches the threshold 40%

<u>6-Corridor:</u>

Components:

- ➔ Boards: MCU
- ➔ Sensors: Fire Monitor
- ➔ Actuators: Fire sprinkler, Door, Heating elements

Communication Steps:

Connect All components with microcontroller (MCU on digital pin0) with IOT cable.

Fire monitor on digital pin0, sprinkler on pin D1 and Door on pin D4

Set up the heating element to allow fire monitor to sense it:

click on heating element > advanced > Programming > new js > add the set up line

Action :

To ensure the safety of guests and staff within hotel corridors, we should add fire monitor

that keeps an eye out for fires or if things get too hot. If it senses any danger, it quickly takes action to keep everyone safe.

first adds fire Sprinkler that start spraying water to put out the fire, helps stop the fire from spreading,

At the same time, the Emergency doors should be open automatically so people can go out and stay safe.

Fire monitor used to detect flames, involves checking a specific "IR" (infrared radiation).

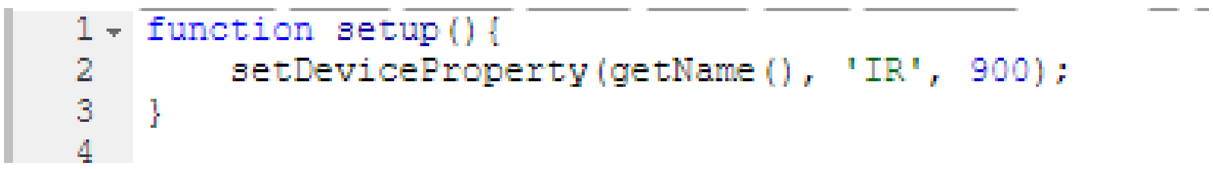
This property is associated with heat, and when it falls within a certain range that our detector recognizes as signal of a fire, it triggers a digital signal

so that we set up a heating element that emits infrared radiation to test the monitor.

heating element code:

We're setting the "IR" property to a value of 900. This value represents the intensity of infrared that our sensor will be looking for.

So, when we put the heating element close to the monitor, it quickly acts (open doors, sprinklers).



```
function setup(){
    setDeviceProperty(getName(), 'IR', 900);
}

```

Components:

- ➔ Boards: same MCU
- ➔ Sensors: Motion sensor
- ➔ Actuators: LED

Steps:

Connect All components with microcontroller (MCU on digital pin0) with IOT cable.

LED on pin D2 and motion sensor on pin D3

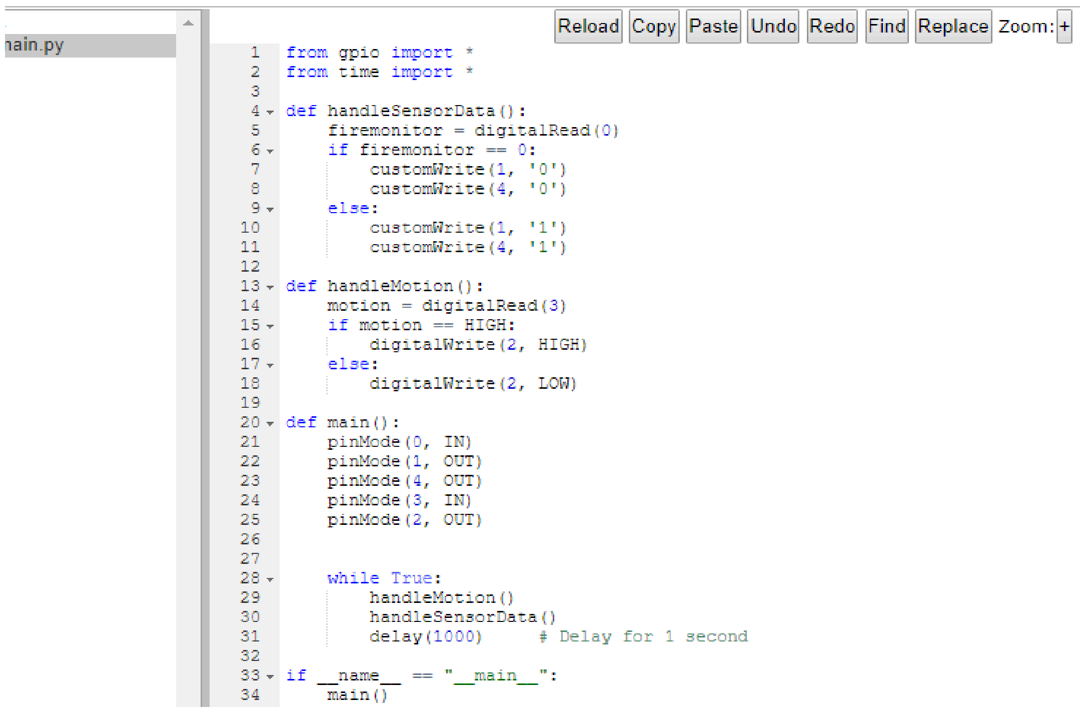
Action:

Motion Sensor and LED:

When someone walks into the corridor, a motion sensor notices and switches on LED light, keeping it while they’re there. Once they leave, the LED stays for 5 seconds and the sensor sees they’re gone and turns off the LED light

When someone walks into the corridor, a motion sensor notices and switches on LED light, keeping it while they're there. Once they leave, the LED stays for 5 seconds and the sensor sees they're gone and turns off the LED light

Code for the whole corridor:



```python
from gpio import *
from time import *

def handleSensorData():
    firemonitor = digitalRead(0)
    if firemonitor == 0:
        customWrite(1, '0')
        customWrite(4, '0')
    else:
        customWrite(1, '1')
        customWrite(4, '1')

def handleMotion():
    motion = digitalRead(3)
    if motion == HIGH:
        digitalWrite(2, HIGH)
    else:
        digitalWrite(2, LOW)

def main():
    pinMode(0, IN)
    pinMode(1, OUT)
    pinMode(4, OUT)
    pinMode(3, IN)
    pinMode(2, OUT)


    while True:
        handleMotion()
        handleSensorData()
        delay(1000)     # Delay for 1 second

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

Imports:

import necessary modules GPIO for working with pins (input/output) and time-related functions (delay, sleep).

Functions:

- Define a function handleSensorData():

  for the first sensor (Fire monitor) ,The code reads data from a fire monitor sensor connected to pin 0.
  If the sensor detects no fire (value is 0) LOW. It turns off two actuators Fire sprinkler connected to pin1 and (Door) connected to pin4, If fire is detected (value is HIGH), it turns on both actuators.

- Function handleMotion():

  Function for the Motion sensor reads data from it connected to pin3, if motion is detected (value is HIGH), it turns on an LED connected to pin 2, If no motion is detected (value is LOW), it turns off the LED.

Main Loop:

- set up the sensor pins as input pins (Receive a signal), and actuators pins as output pins (doing action).

- Inside the while True loop, it continuously monitors the motion sensor and fire sensor by calling handleMotion() and handleSensorData() functions respectively. delays the execution for 1 second using delay(1000).

5. Conclusion and future work

In establishing our Smart Hotel, our paramount goal was to redefine hospitality by seamlessly integrating cutting-edge technologies to empower guests with control, security, and unparalleled convenience throughout their stay. Spanning across three floors, each meticulously designed to cater to various aspects of guest experience, our Smart Hotel represents a paradigm shift in the hospitality industry.

Through the utilization of RFID card access, motion sensors, and smartphone integration, guests are granted effortless control over key amenities such as room access, climate control, and coffee machine activation. This level of convenience not only enhances the guest experience but also sets a new standard for personalized service and comfort.

Moreover, our commitment to guest safety is demonstrated through a comprehensive security infrastructure. From automated responses to potential threats such as unauthorized access or intrusion, to proactive measures against fire hazards and smoke detection, our Smart Hotel prioritizes the well-being and peace of mind of every guest.

Our future works to add Heating system (furnace) ,street lamp ,solar panel to get the electricity , To provide a more comfortable hotel for guests and more energy saving for hotel partners

6. References

https://www.scribd.com/document/558772950/Hotel-Topology-Project-Report

https://dergipark.org.tr/tr/download/article-file/510059

https://www.researchgate.net/publication/368530422_Simulation_of_an_IoT-based_Smart_Home_using_Cisco_Packet_Tracer_730

https://www.slideshare.net/RaoFazal/hotel-network-scenario-implementation-by-using-cisco-packet-tracer