SMART COLLEGE

IoT Project Report

Ahmed Mohamed Hassan (4.1, 4.2)	20201321583
Deeana Fathi Mohamed (4.5, 4.6)	20201447217
Zeyad Mahmoud Bassiouny (4.3, 4.4)	20201374047

Abstract

The concept of a “smart College” is an Internet of Things (IoT) based approach to enhance the efficiency and effectiveness of different functions performed by a College such as resource management, environmental monitoring, security etc. Lecture halls, entrance gate, student affairs office, garden area, garage and library on campus will be equipped with state-of-the-art IoT systems under this initiative. Each space is fitted with dedicated sensors and devices that are aimed at enhancing safety measures while maximizing operational efficiency. The report investigates these IoT solutions’ development and application in order to show how they can transform traditional college environments into smart interconnected spaces.

Keywords: Internet of Things, Smart Campus, Smart College, College Security, Resource Optimization, IoT Applications, Smart Infrastructure, Automation

1. Introduction

Digital technologies development, especially the Internet of Things (IoT), provides an opportunity for College campuses to enhance their operations by automating and connecting systems. By deploying IoT systems across different College ecosystems, our project "Smart College" aims to achieve this goal and thus make a campus that is more adaptive, safer, and more productive for studies.

The lecture hall, student affairs office, garden, garage, and front gate are the six main areas that will have IoT implemented in them under Smart College concept. In each area there are specific sensors and devices installed to perform certain tasks better such as monitoring resources usage or environmental conditions or even security surveillance among others.

Also, this project aims to demonstrate how intelligent technologies can be used in schools to achieve sustainability goals besides operational efficiencies. Operational expenses and the carbon footprint of the College could be significantly reduced by monitoring and controlling environmental parameters as well as energy use.

In other words, the “Smart College” initiative is considered a breakthrough in utilizing IoT technology towards transforming learning spaces. We enhance safety, quicken operations among others through integration of smart devices and sensors in strategic areas across campus thereby making huge contributions towards conservation efforts. This endeavor does not only provide a blueprint for other colleges worldwide on how they can utilize IoT in creating more sustainable, efficient, and engaging learning environments but also demonstrates utilization cutting-edge technologies within an applicable academic setting.

2. Related works

The integration of Internet of Things (IoT) technologies in educational settings has gained significant attention in recent years, with numerous initiatives aimed at creating smart campuses to enhance the overall learning environment. In this section, we review projects relevant to IoT in education.

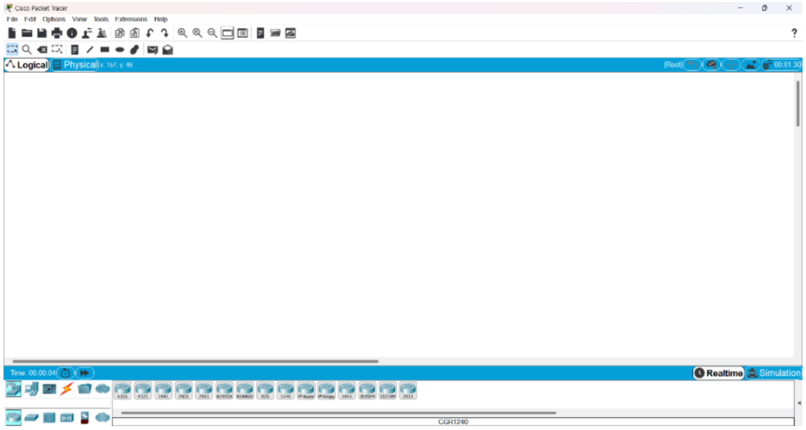
IoT technologies offer promising ideas to change traditional educational practices by providing real-time data collection, analysis, and automation. Numerous studies have explored the potential applications of IoT in educational environments, this includes smart classrooms and campus management systems. These applications aim to improve efficiency, teaching, and overall learning experiences.

One of the main objectives of IoT in smart campuses is to enhance efficiency. By leveraging IoT technologies for energy management and resource optimization, universities can achieve high cost savings and environmental benefits. Examples include the implementation of smart building systems to regulate heating, and ventilation (Which we have done in our project) based on occupancy patterns, as well as the deployment of smart irrigation systems for water conservation in campus gardens.

Despite the great number of benefits of IoT in educational environments, security and privacy concerns remain high. The rapid increase of connected devices and data streams increases the risk of cyber-attacks, data breaches, and privacy violations. This means that universities must implement robust security measures to protect sensitive information and student privacy.

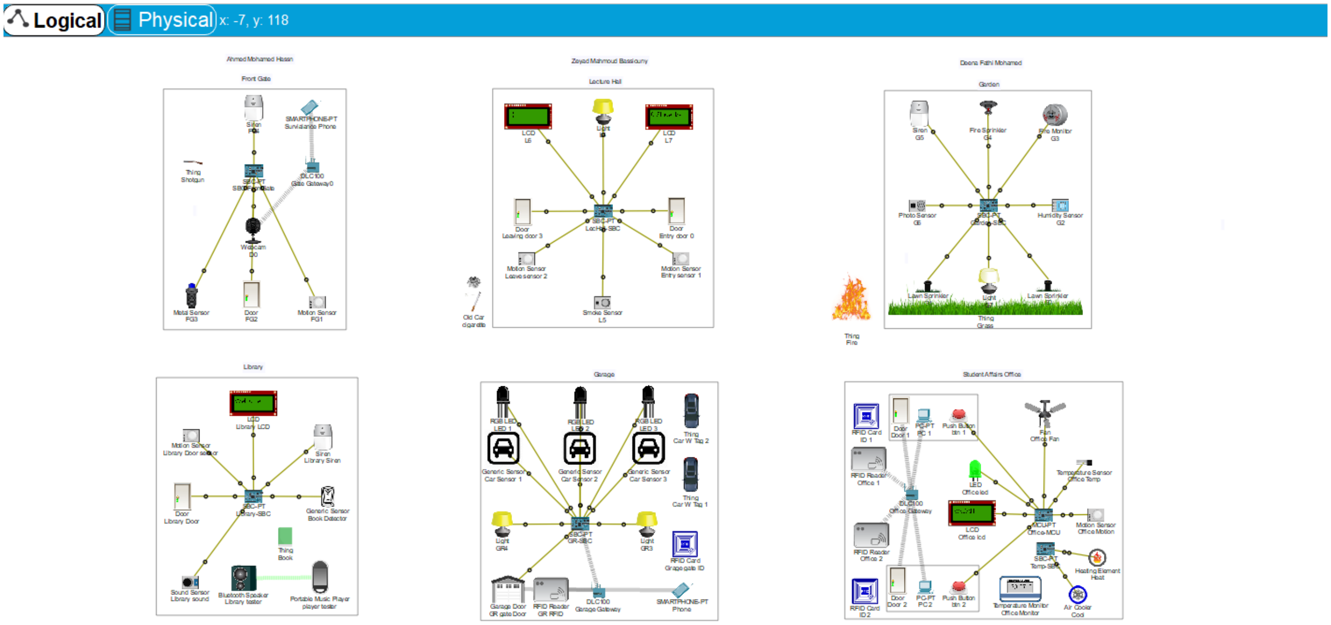
3. Methodology

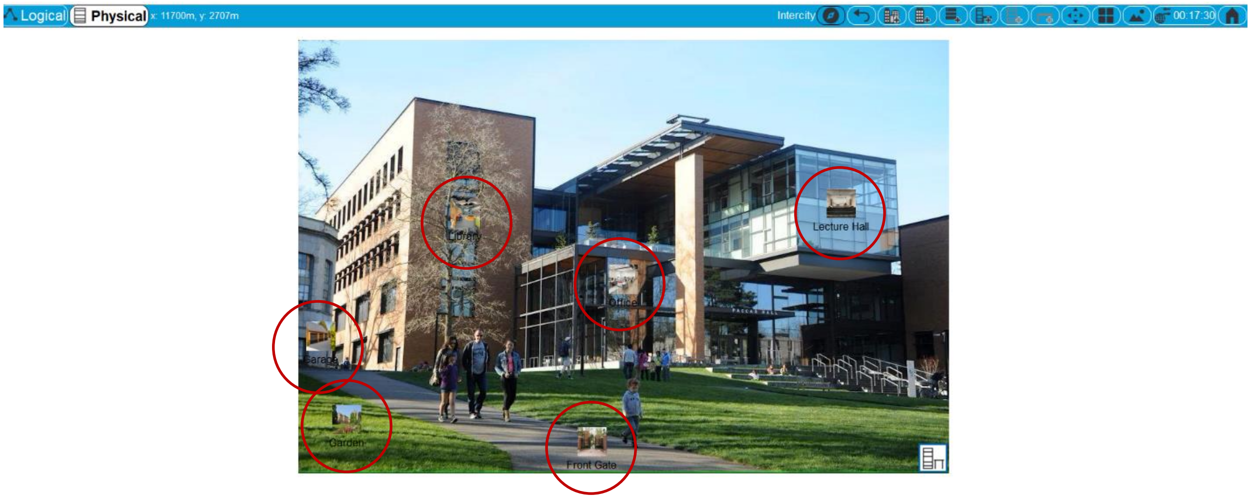
The methodology employed in this project involved the design and implementation of an IoT-based smart college system using Cisco Packet Tracer. The following subsections outline the key steps and components of the methodology:



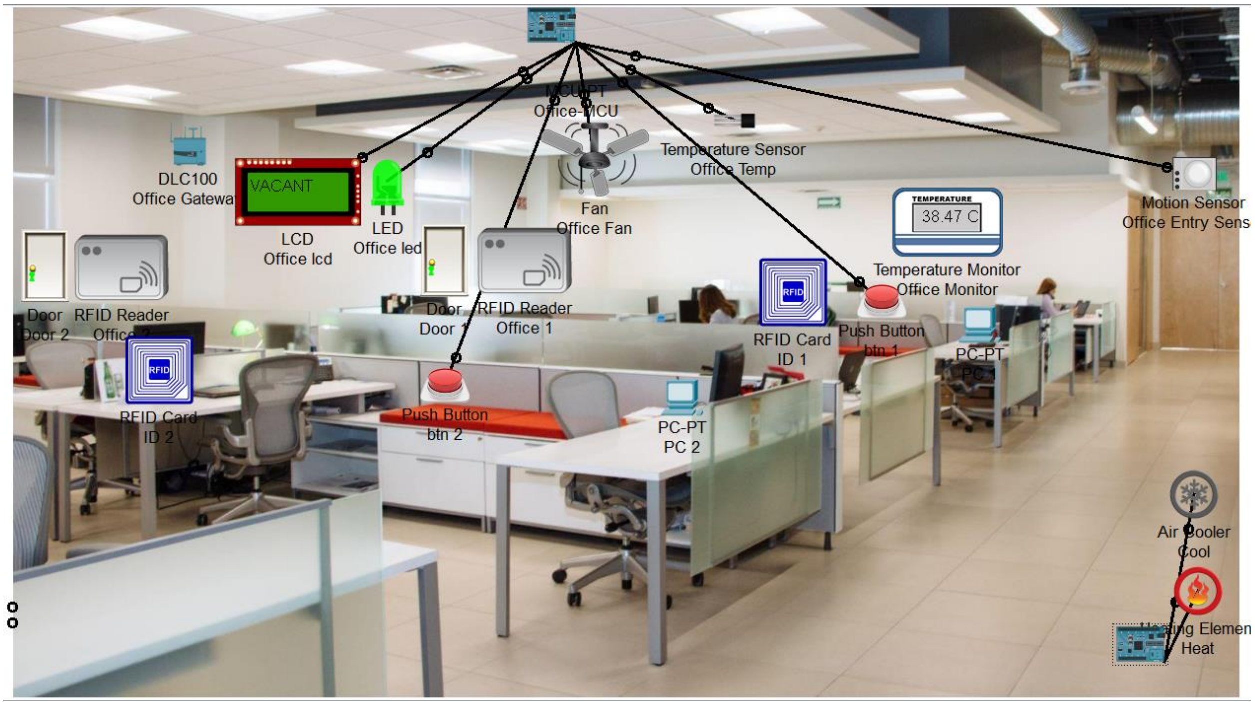
3.1 Design Phase

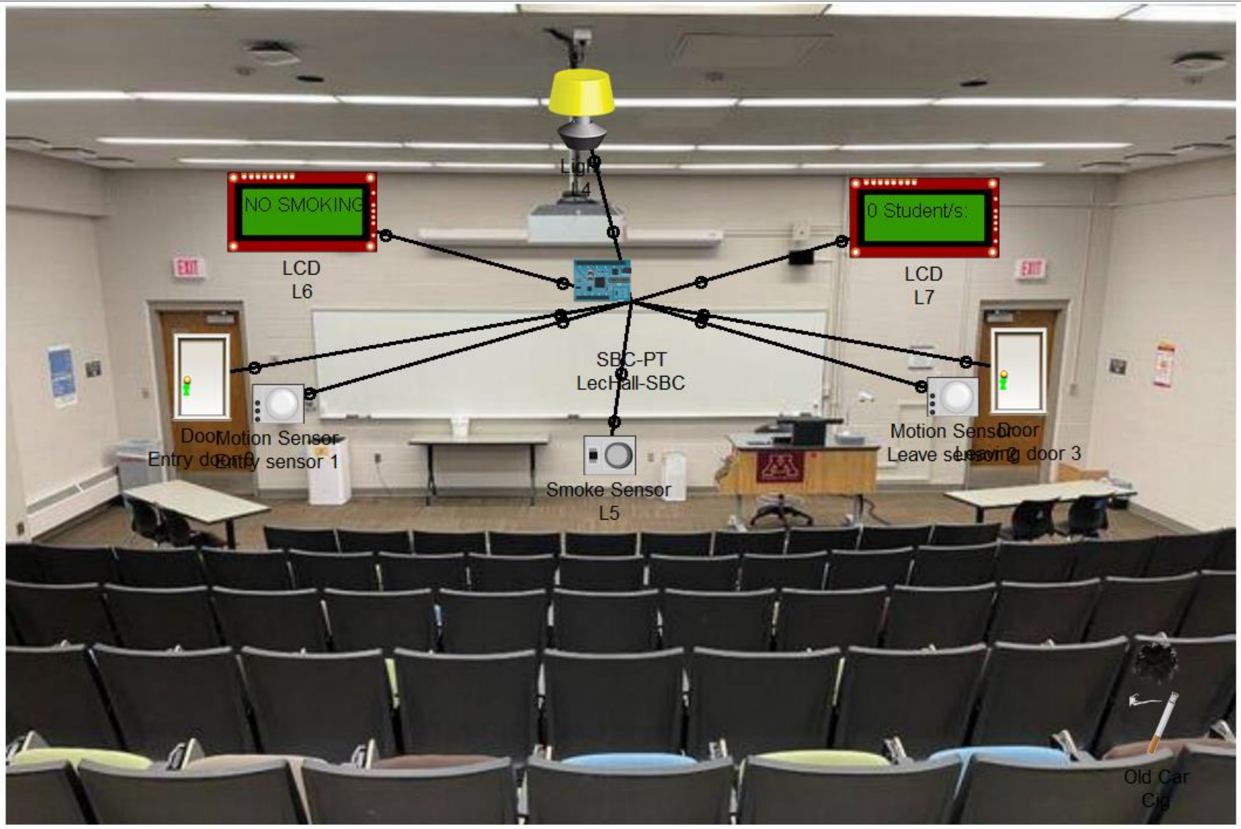
In the initial stage of this project, a plan was developed to conceptualize the architecture and functionalities of the smart college system. This phase involved identifying the requirements and objectives of the project, including the integration of various IoT devices and sensors in common faculty rooms found in any generic college and how they should be integrated, **and here are the Logical & Physical views:**

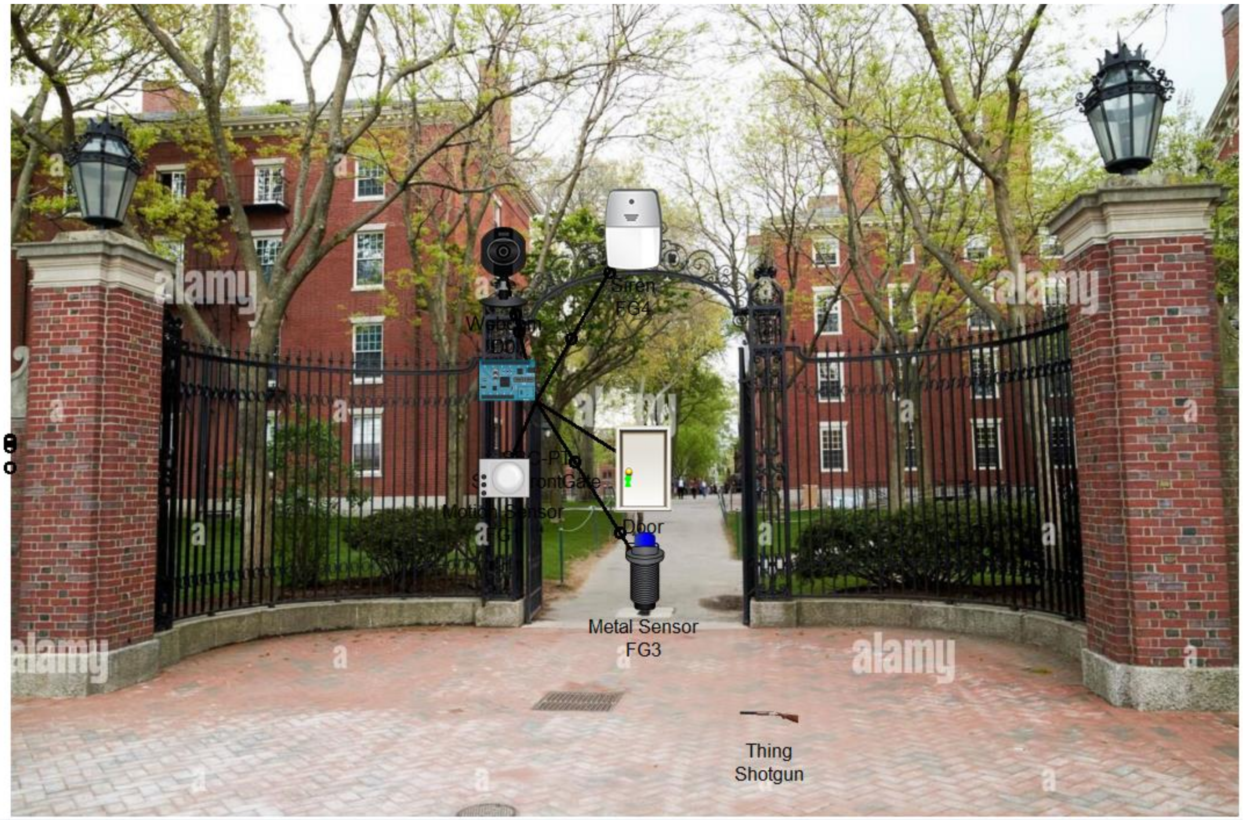




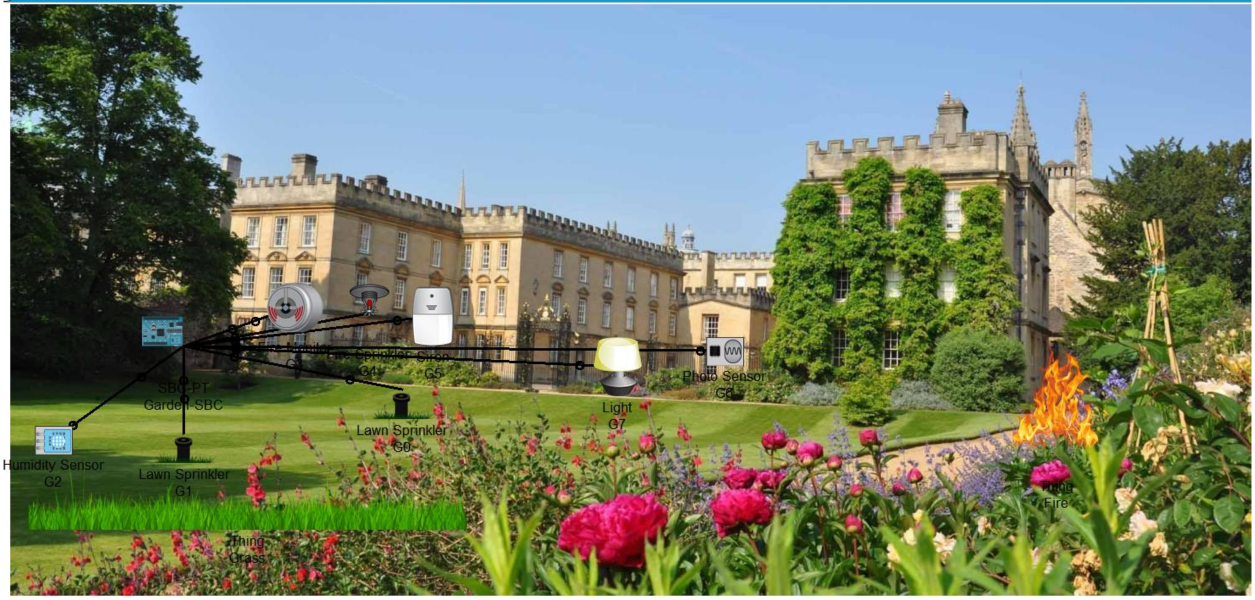
Each Room is marked by red circles and here is a view of each one:


MCU-PT
Office-MCU
Temperature Sensor
Office Temp
DLC100
Office Gatewa
VACANT
LCD
Office lcd
LED
Office led
Fan
Office Fan
Motion Sensor
Office Entry Sens
TEMPERATURE
38.47 C
Temperature Monitor
Office Monitor
Door
Door 2
RFID Reader
Office 2
Door
Door 1
RFID Reader
Office 1
RFID Card
ID 1
Push Button
btn 1
PC-PT
PC
RFID Card
ID 2
Push Button
btn 2
PC-PT
PC 2
Air Cooler
Cool
Heating Elemen
Heat

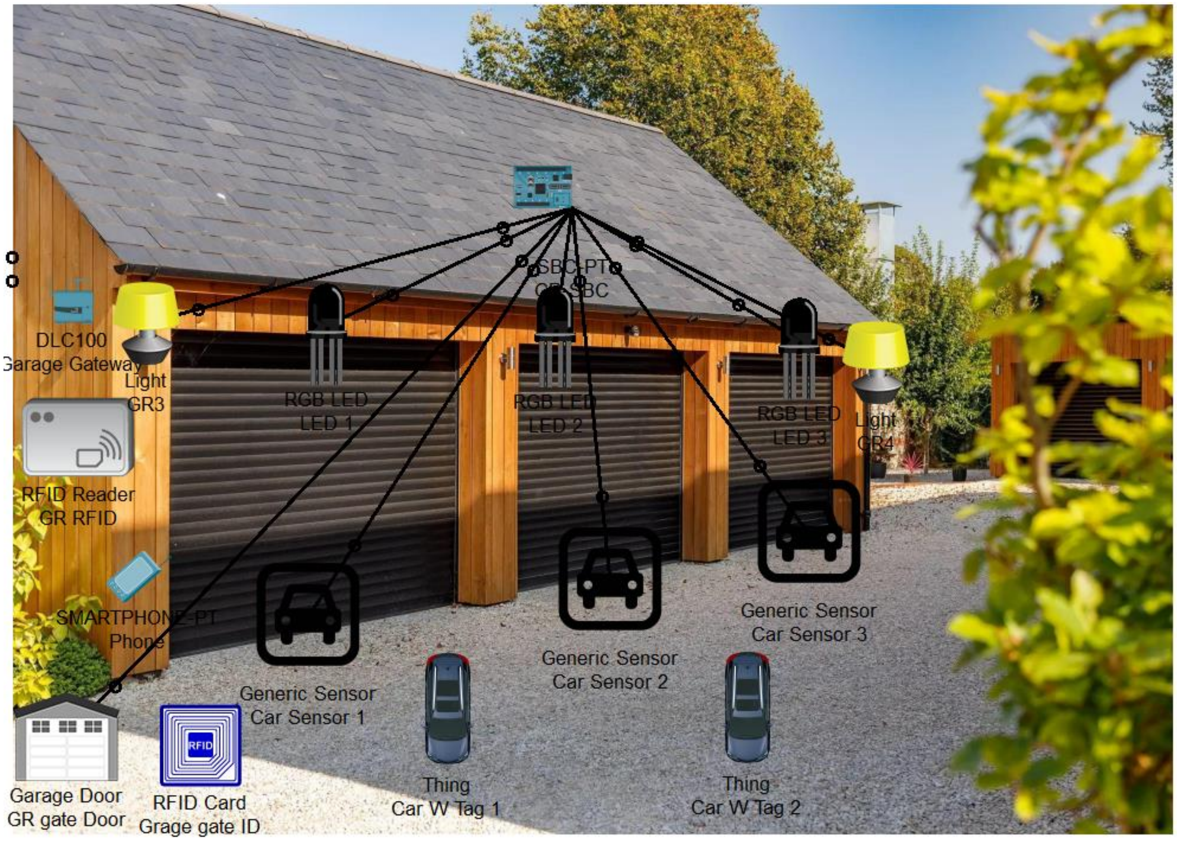

Light
L4
NO SMOKING
LCD
L6
0 Student/s:
LCD
L7
EXIT
EXIT
SBC-PT
LecHall-SBC
Door
Entry door 0
Motion Sensor
Entry sensor 1
Smoke Sensor
L5
Motion Sensor
Leave sensor 2
Door
Leaving door 3
Old Car
Cig

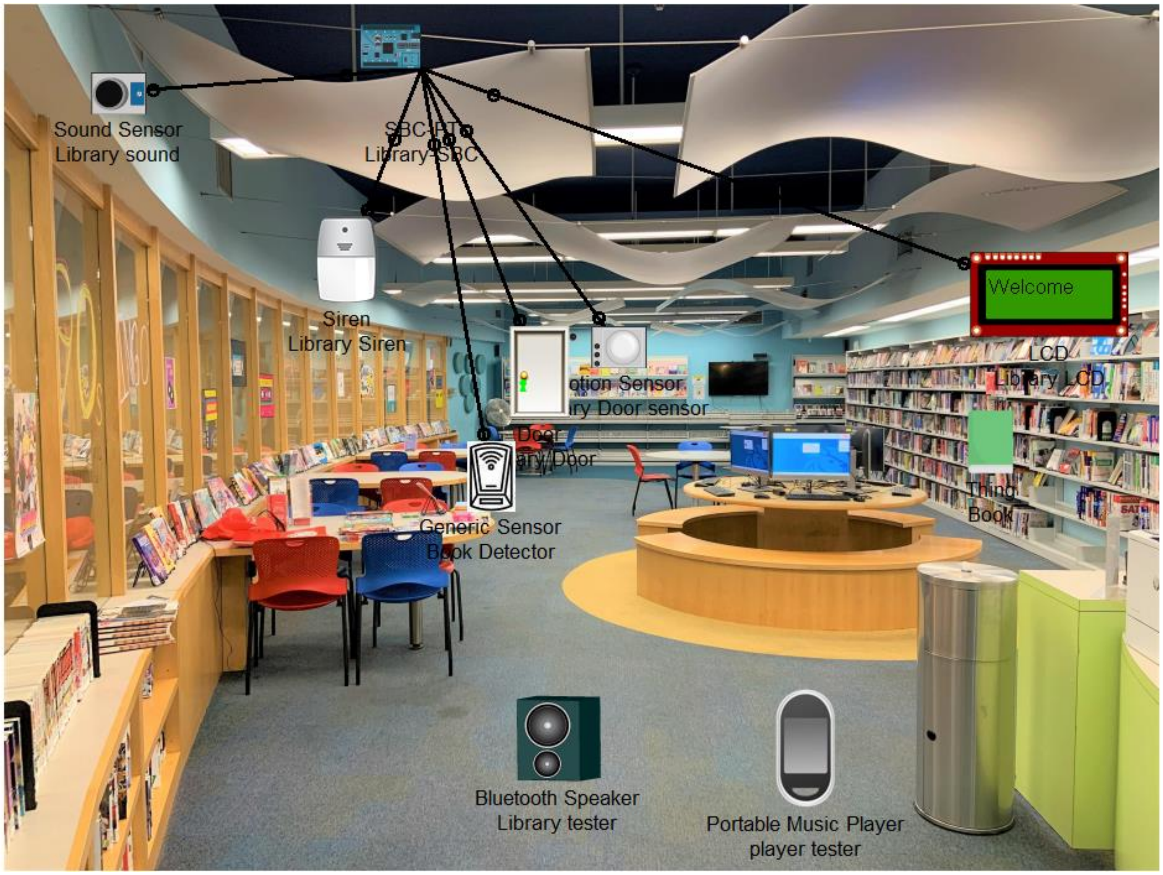


Siren
FG4
Webcam
D0
SBC-PT
SBC-FrontGate
Motion Sensor
FG1
Door
Metal Sensor
FG3
Thing
Shotgun



Fire Monitor
G3
Fire Sprinkler
G4
Siren
G5
Photo Sensor
G6
Light
G7
SBC-PT
Garden-SBC
Lawn Sprinkler
G0
Humidity Sensor
G2
Lawn Sprinkler
G1
Thing
Grass
Fire


SBC-PT
GR SBC
DLC100
Garage Gateway
Light
GR3
RGB LED
LED 1
RGB LED
LED 2
RGB LED
LED 3
Light
GR4
RFID Reader
GR RFID
SMARTPHONE-PT
Phone
Generic Sensor
Car Sensor 1
Generic Sensor
Car Sensor 2
Generic Sensor
Car Sensor 3
Thing
Car W Tag 1
Thing
Car W Tag 2
Garage Door
GR gate Door
RFID
RFID Card
Grage gate ID


Sound Sensor
Library sound
SBC-PT
Library SBC
Siren
Library Siren
otion Sensor
ry Door sensor
Door
ary Door
Generic Sensor
Book Detector
Welcome
LCD
Library LCD
Thing
Book
Bluetooth Speaker
Library tester
Portable Music Player
player tester

3.2 Selection of Sensors and Devices

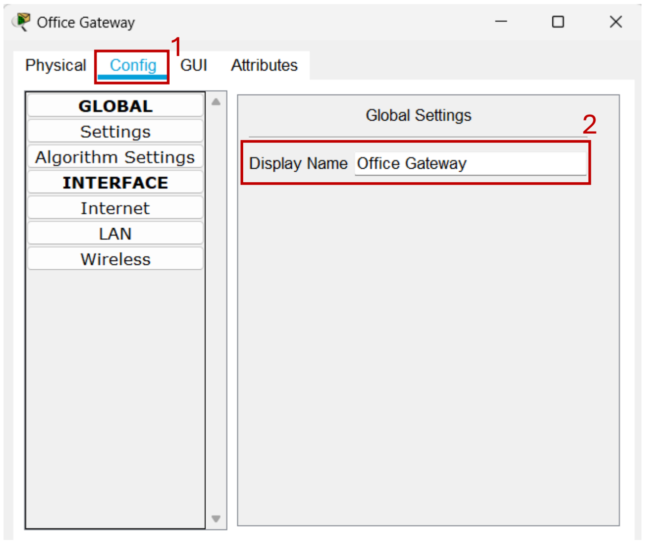
A critical aspect of the design process was the selection of appropriate sensors and devices to be added into each room. These sensors were chosen based on their availability within Cisco Packet Tracer and their relevance to the desired smart campus applications and rooms. Common types of sensors utilized in the project included:

Sensor	Usage
Temperature sensor	Control air conditioning
Motion sensors	Occupancy detection in classrooms, study areas, and trespassers.
Proximity sensor	Car parking system
Metal detector sensor	Weapon detection
Smoke sensor	Cigarettes detection in classrooms
Humidity sensor	Garden watering
Book sensor	Protect the college library from theft
Sound sensor	Keep the calm quite atmosphere
Fire Monitor	Check if there are any fires around
Photo sensor	Detects light around area
Generic Sensor	Custom sensor to detect the tag attribute in cars

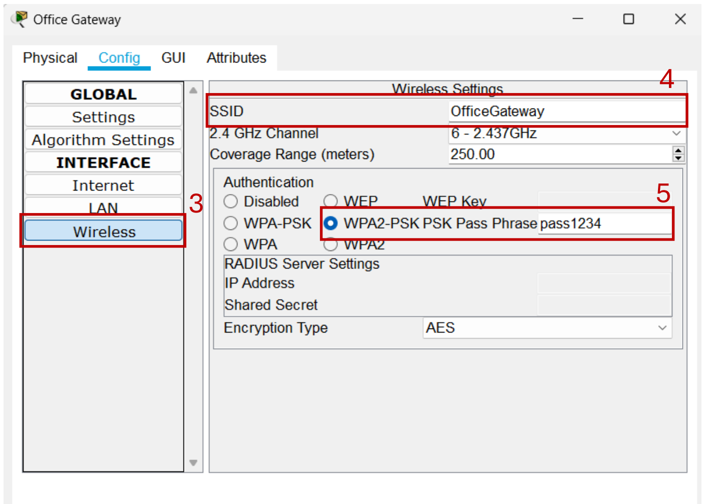
3.3 Network Configuration

Once the sensors and devices were selected, the next step involved configuring the network infrastructure within Cisco Packet Tracer to facilitate communication and data exchange between the IoT components. This included setting up virtual networks, configuring routers and switches, and connecting wires between different devices. Ultimately using a home gateway for any wireless connection.

1. Home Gateway Setup:



Display Name Office Gateway



SSID OfficeGateway

2.4 GHz Channel 6 - 2.437GHz

Coverage Range (meters) 250.00

Authentication

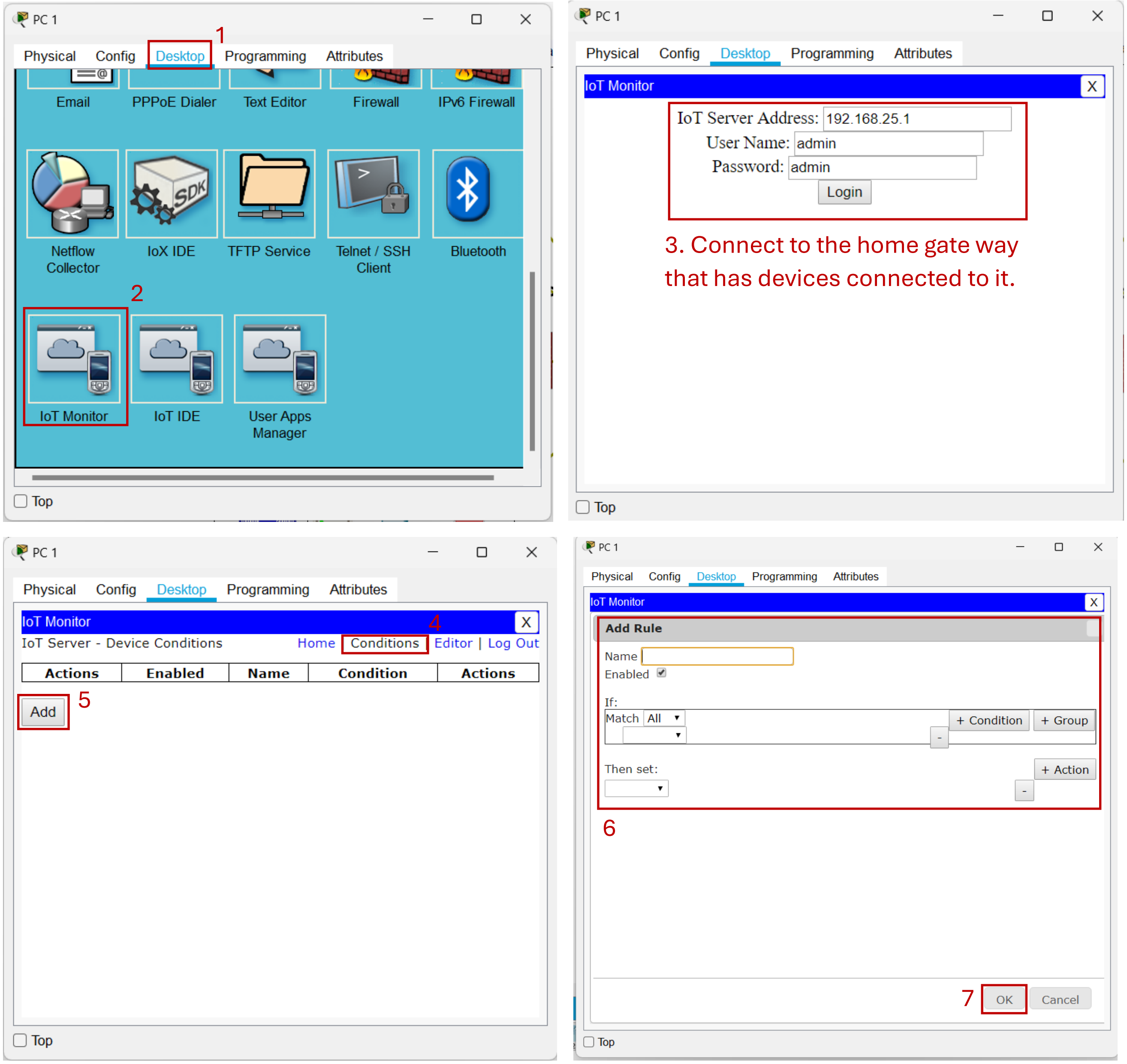
○ Disabled ○ WEP WEP Key

○ WPA-PSK ● WPA2-PSK PSK Pass Phrase pass1234

○ WPA ○ WPA2

Encryption Type AES

Adding Conditions form any device:



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

3. Connect to the home gate way that has devices connected to it.

IoT Server - Device Conditions

Actions	Enabled	Name	Condition	Actions

Name
Enabled ☑
If:
Match All
Then set:

3.5 Testing and Validation

Following the integration of IoT components in Cisco Packet Tracer, various tests were done to validate the functionality and performance of each smart college system. This includes simulated scenarios to test sensor responsiveness, and reliability under various conditions. Feedback from these tests was used to enhance and optimize the systems as needed.

4. Implementation and Results

4.1 Front Gate:

IoT devices/sensors	Connection Port	home gateway
Webcam	D0 to SBC	192.168.27.1
Motion sensor	D1 to SBC	-
Door	D2 to SBC	-
Metal sensor	D3 to SBC	-
Siren	D4 to SBC	-
Phone		192.168.27.1

Overall Front Gate Security System Behavior

- Metal Detection and Safety Control: Utilizes a metal sensor to detect potential weaponry and prevent entry while activating safety protocols.
- Motion Detection and Access Management: Manages door access based on motion detection, allowing entry only when safe.
- Surveillance Activation: Employs a webcam to visually record all entry attempts, activated by security sensors.
- Alarm System: Siren is triggered in response to metal detection, signaling a security alert.

Functional Overview

1. Metal and Weaponry Detection:

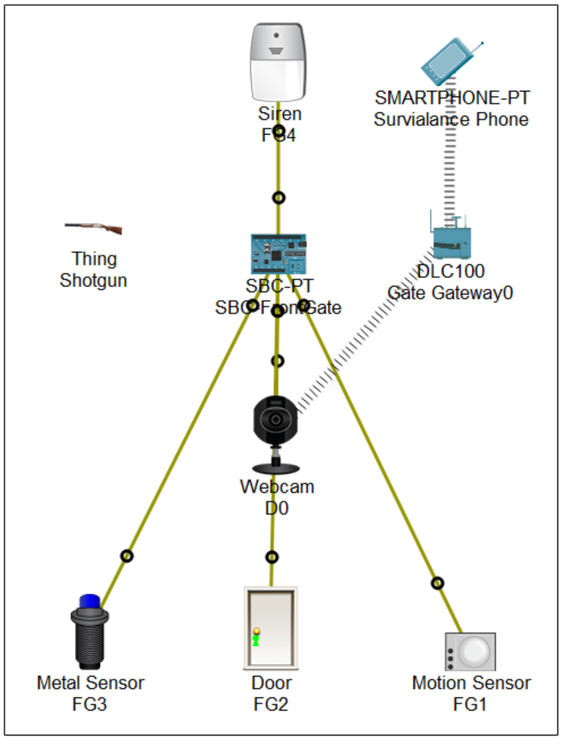
- A metal sensor detects metallic objects and potential weapons.
- The siren and webcam are activated if metal is detected, with the door remaining locked.

2. Motion-Based Access Control:

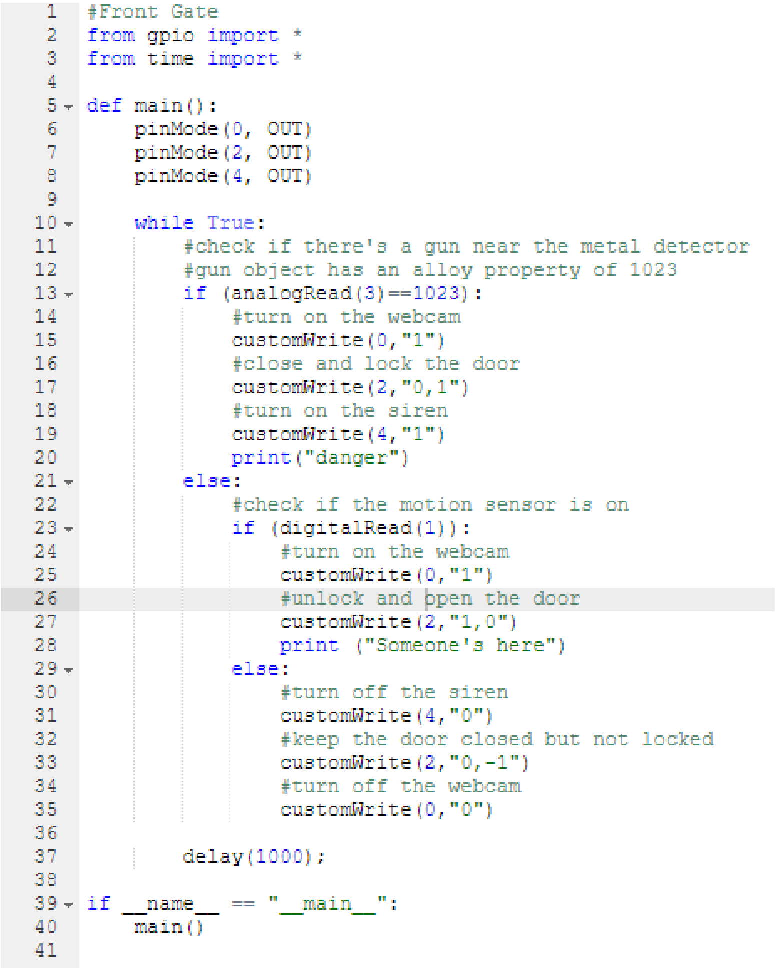
- A motion sensor triggers the door to unlock and open if no metal is detected, ensuring controlled access.
- The webcam captures footage of individuals entering when motion is detected.

3. Emergency Response and Monitoring:

- Immediate alerts and visual evidence are generated in response to potential threats, enhancing security measures.



Front Gate SBC Code:



```
#Front Gate
from gpio import *
from time import *

def main():
    pinMode(0, OUT)
    pinMode(2, OUT)
    pinMode(4, OUT)

    while True:
        #check if there's a gun near the metal detector
        #gun object has an alloy property of 1023
        if (analogRead(3)==1023):
            #turn on the webcam
            customWrite(0,"1")
            #close and lock the door
            customWrite(2,"0,1")
            #turn on the siren
            customWrite(4,"1")
            print("danger")
        else:
            #check if the motion sensor is on
            if (digitalRead(1)):
                #turn on the webcam
                customWrite(0,"1")
                #unlock and open the door
                customWrite(2,"1,0")
                print ("Someone's here")
            else:
                #turn off the siren
                customWrite(4,"0")
                #keep the door closed but not locked
                customWrite(2,"0,-1")
                #turn off the webcam
                customWrite(0,"0")

        delay(1000);

if __name__ == "__main__":
    main()

```

4.2 The Library:

IoT devices/sensors	Connection Port
sound sensor	D0 to SBC
motion sensor	D1 to SBC
book detection sensor	D2 to SBC
LCD	D3 to SBC
Door	D4 to SBC
Siren	D5 to SBC

Overall Library System Behavior

- Automated Entry and Exit: Motion sensors activate doors, providing a seamless user experience.
- Noise Level Monitoring: Noise sensors maintain a quiet environment, displaying messages when sound thresholds are exceeded.
- Noise simulation: Connecting a Bluetooth speaker to a music player to simulate noise.
- Theft Prevention: Book detection sensors at exit points alert staff to unauthorized material removal through an integrated alarm system.

Functional Overview

1. Motion-Activated Access:

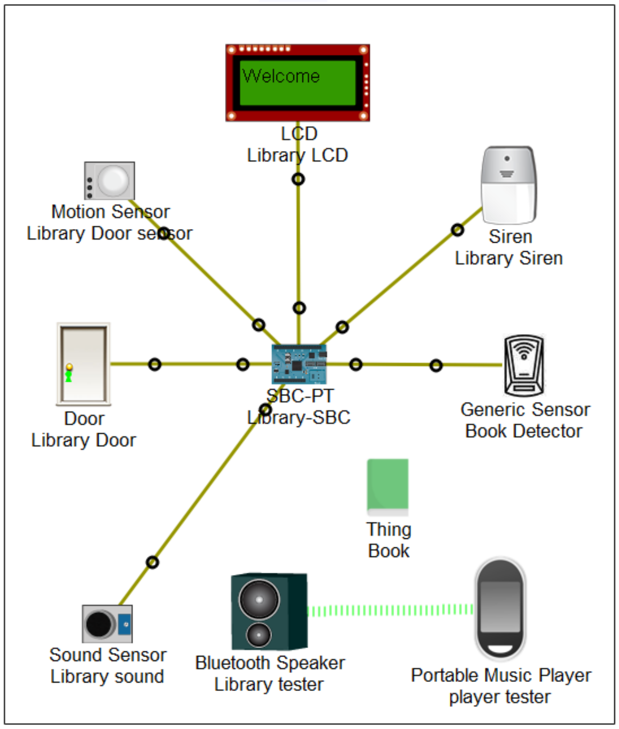
- The door automatically opens or closes in response to motion sensor detection, facilitating easy entry and exit.

2. Sound Monitoring and Communication:

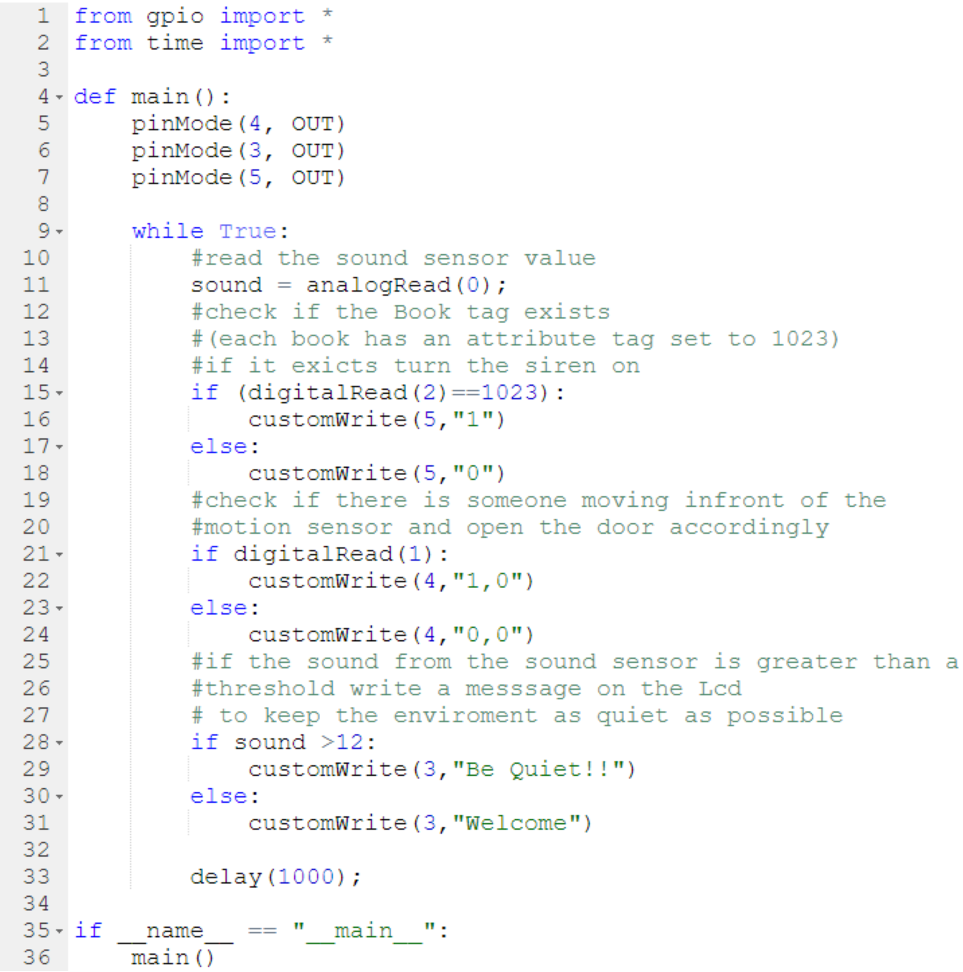
- A sound sensor assesses noise levels, with an LCD displaying messages to encourage quiet or welcome patrons based on ambient noise.

3. Security Against Unauthorized Removal:

- Custom sensor tags in library materials trigger an alarm if items are removed without proper checkout, detected by the book detection sensor.



Library SBC Code:



```
from gpio import *
from time import *

def main():
    pinMode(4, OUT)
    pinMode(3, OUT)
    pinMode(5, OUT)

    while True:
        #read the sound sensor value
        sound = analogRead(0);
        #check if the Book tag exists
        #(each book has an attribute tag set to 1023)
        #if it exicts turn the siren on
        if (digitalRead(2)==1023):
            customWrite(5,"1")
        else:
            customWrite(5,"0")
        #check if there is someone moving infront of the
        #motion sensor and open the door accordingly
        if digitalRead(1):
            customWrite(4,"1,0")
        else:
            customWrite(4,"0,0")
        #if the sound from the sound sensor is greater than a
        #threshold write a messsage on the Lcd
        # to keep the enviroment as quiet as possible
        if sound >12:
            customWrite(3,"Be Quiet!!")
        else:
            customWrite(3,"Welcome")

        delay(1000);

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

4.3 Lecture Hall:

IoT devices/sensors	Connection Port
Doors	D0, D3 to SBC
Motion sensor	D1, D2 to SBC
Light	D4 to SBC
Smoke sensor	D5 to SBC
LCDs	D6, D7 to SBC

Overall Classroom Management System Behavior

- <u>Student Counting:</u> Monitors and updates the number of students in the classroom using entry and exit sensors.
- <u>Smoke Detection:</u> Employs a smoke sensor to detect cigarette smoke and displays a warning on the left-side LCD.
- <u>Light Automation:</u> Activates room lighting based on the presence of students.

Functional Overview

1. Student Tracking and Display:

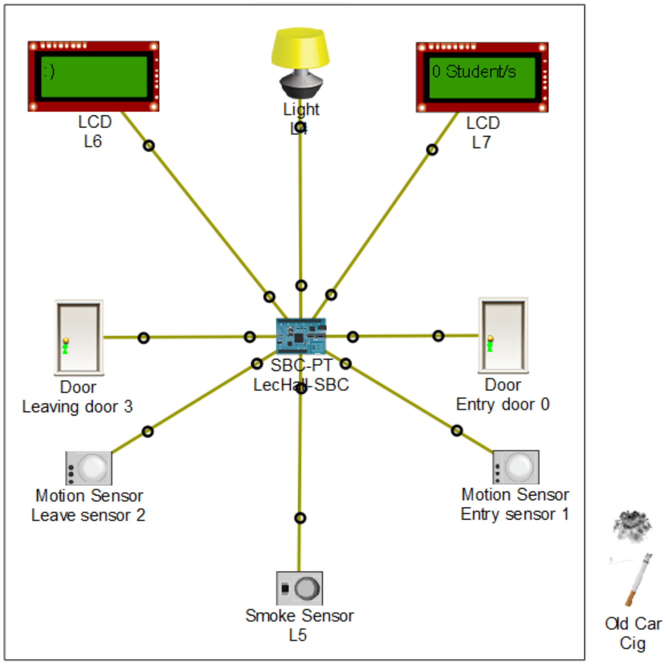
- Utilizes motion sensors at the entry and exit to count students entering and exiting.
- Displays the current student count on the right-side LCD.

2. Smoke Monitoring and Alerting:

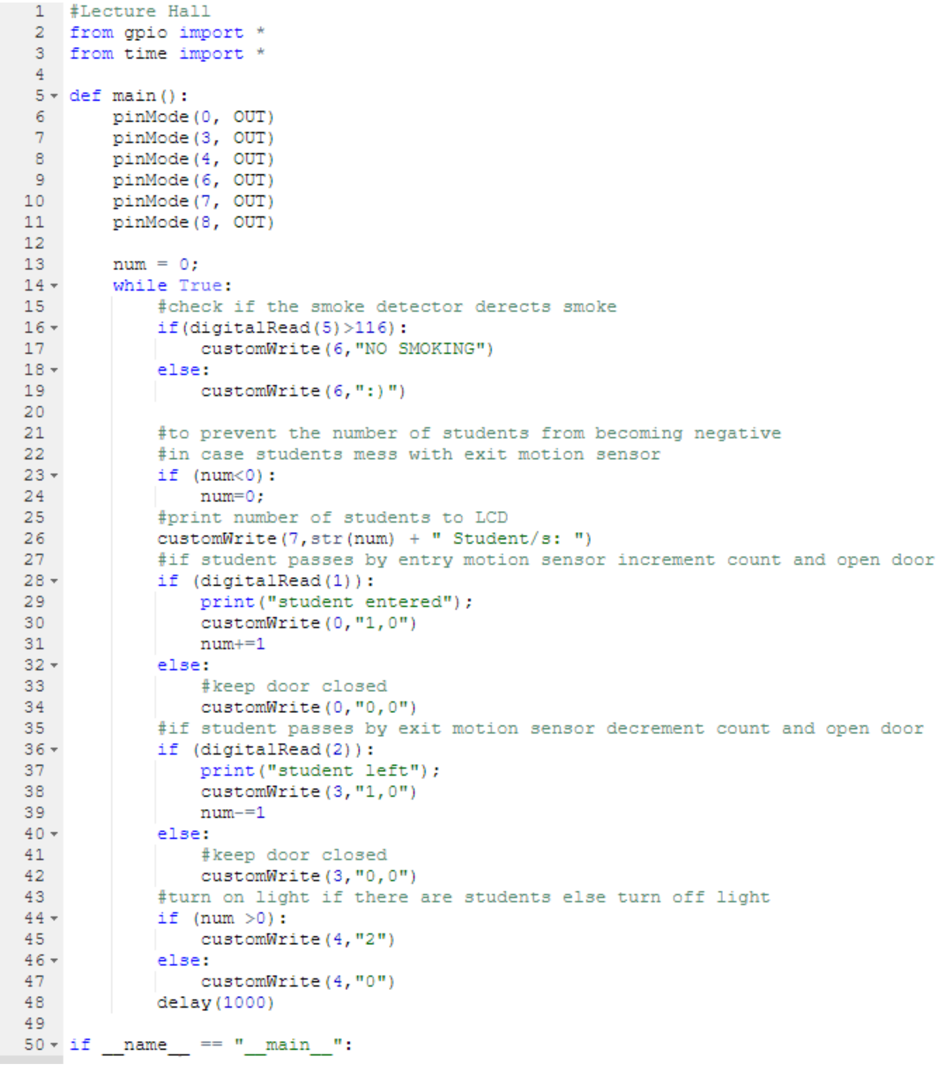
- A smoke sensor checks for cigarette smoke in the classroom.
- Triggered smoke detection results in a "NO SMOKING" warning on the left-side LCD.

3. Room Lighting Control:

- The room light is automatically managed to ensure it is on when students are present and off when the room is empty.



Lecture Hall SBC Code:



```
#Lecture Hall
from gpio import *
from time import *

def main():
    pinMode(0, OUT)
    pinMode(3, OUT)
    pinMode(4, OUT)
    pinMode(6, OUT)
    pinMode(7, OUT)
    pinMode(8, OUT)

    num = 0;
    while True:
        #check if the smoke detector derects smoke
        if(digitalRead(5)>116):
            customWrite(6,"NO SMOKING")
        else:
            customWrite(6,":)")

        #to prevent the number of students from becoming negative
        #in case students mess with exit motion sensor
        if (num<0):
            num=0;
        #print number of students to LCD
        customWrite(7,str(num) + " Student/s: ")
        #if student passes by entry motion sensor increment count and open door
        if (digitalRead(1)):
            print("student entered");
            customWrite(0,"1,0")
            num+=1
        else:
            #keep door closed
            customWrite(0,"0,0")
        #if student passes by exit motion sensor decrement count and open door
        if (digitalRead(2)):
            print("student left");
            customWrite(3,"1,0")
            num-=1
        else:
            #keep door closed
            customWrite(3,"0,0")
        #turn on light if there are students else turn off light
        if (num >0):
            customWrite(4,"2")
        else:
            customWrite(4,"0")
        delay(1000)

if __name__ == "__main__":
```

4.4 Garage:

IoT devices/sensors	Connection Port	home gateway
Car sensor	D0, D1, D2 to SBC	-
Light	D3, D4 to SBC	-
LEDs	D5, D6, D7 to SBC	-
Garage door	D8 to SBC	192.168.26.1
RFID reader	-	192.168.26.1
Phone	-	192.168.26.1

Overall Garage System Behavior

- Entry Authentication: Utilizes an RFID system that grants access through the use of designated cards.
- Parking Spot Management: Equipped with custom sensors that detect a custom attribute in cars and RGB LEDs to indicate the availability of parking spots.
- Lighting Automation: Lights are activated upon garage entry and managed based on garage door status.
- RFID and Garage Door Integration: Both connected to a home gateway and controlled via programmed conditions.

Functional Overview

1. RFID Access Control:

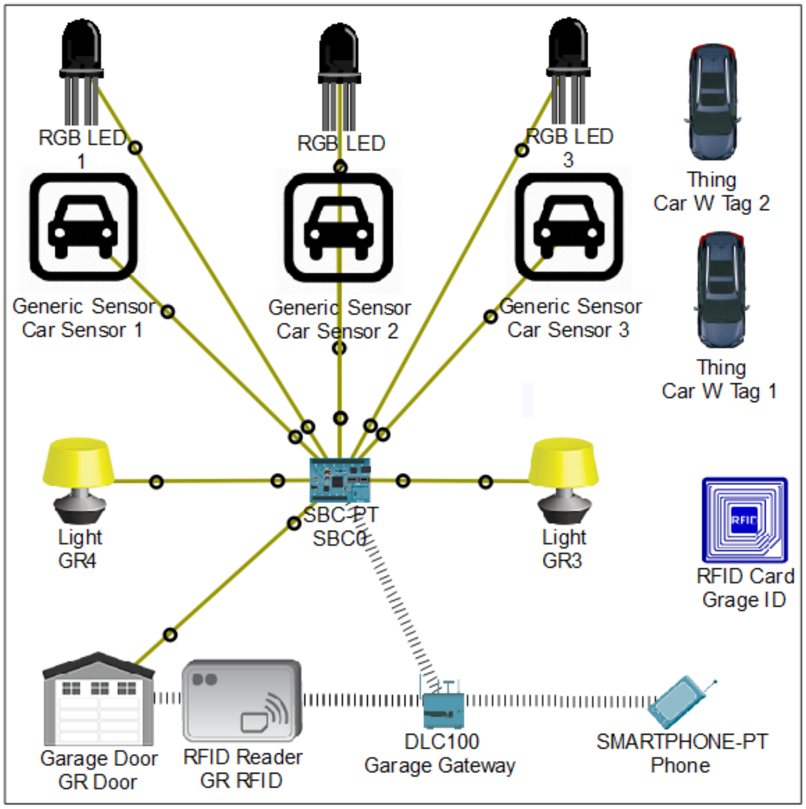
- An RFID reader enables authorized entry to the garage.
- It's linked to a home gateway that handles access logic.

2. Parking Management:

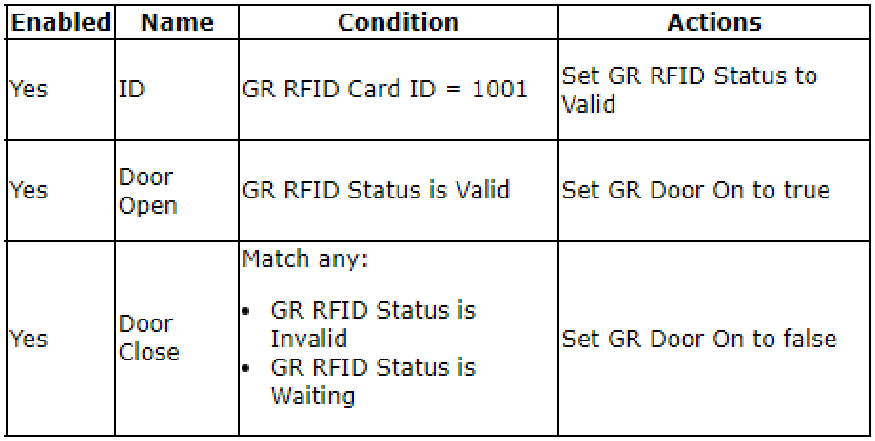
- Three parking sensors detect cars with specific tags, triggering corresponding RGB LEDs to indicate spot occupancy.
- Car sensors for each parking spot; LED for corresponding RGB LEDs.

3. Lighting and Door Control:

- Lights are controlled by the garage door's status; turning on when the door is open.
- The garage door's operational status is tracked and controlled.

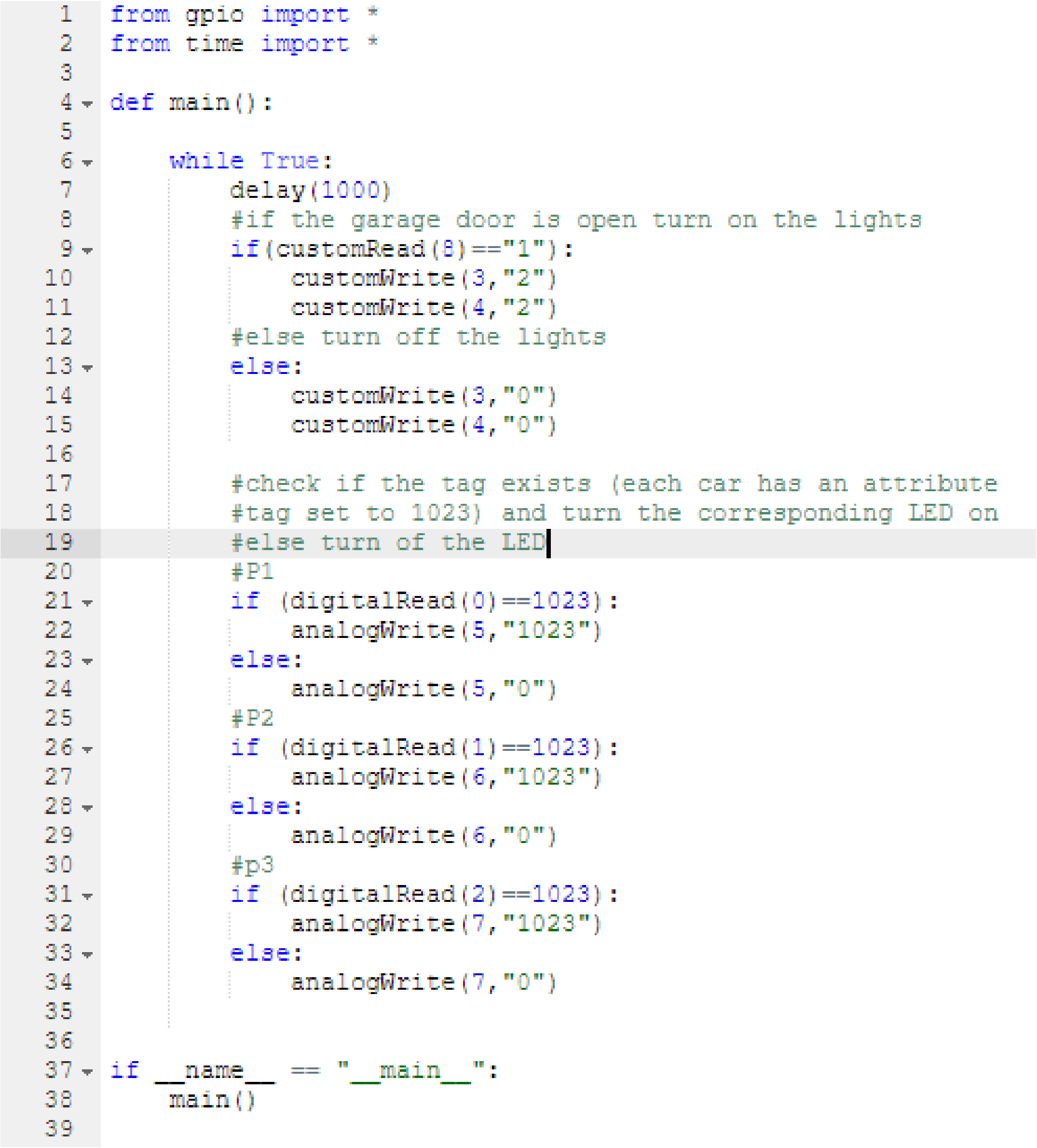


Grage RFID Conditions:



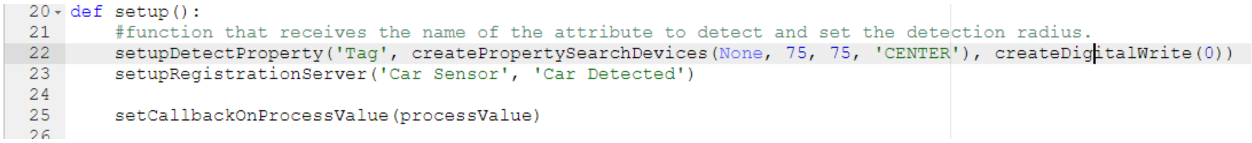
Enabled	Name	Condition	Actions
Yes	ID	GR RFID Card ID = 1001	Set GR RFID Status to Valid
Yes	Door Open	GR RFID Status is Valid	Set GR Door On to true
Yes	Door Close	Match any: • GR RFID Status is Invalid • GR RFID Status is Waiting	Set GR Door On to false

Garage SBC Code:



```
from gpio import *
from time import *

def main():

    while True:
        delay(1000)
        #if the garage door is open turn on the lights
        if(customRead(8)=="1"):
            customWrite(3,"2")
            customWrite(4,"2")
        #else turn off the lights
        else:
            customWrite(3,"0")
            customWrite(4,"0")

        #check if the tag exists (each car has an attribute
        #tag set to 1023) and turn the corresponding LED on
        #else turn of the LED
        #P1
        if (digitalRead(0)==1023):
            analogWrite(5,"1023")
        else:
            analogWrite(5,"0")
        #P2
        if (digitalRead(1)==1023):
            analogWrite(6,"1023")
        else:
            analogWrite(6,"0")
        #p3
        if (digitalRead(2)==1023):
            analogWrite(7,"1023")
        else:
            analogWrite(7,"0")


if __name__ == "__main__":
    main()

```

Car Sensor edited Code:



```
def setup():
    #function that receives the name of the attribute to detect and set the detection radius.
    setupDetectProperty('Tag', createPropertySearchDevices(None, 75, 75, 'CENTER'), createDigitalWrite(0))
    setupRegistrationServer('Car Sensor', 'Car Detected')

    setCallbackOnProcessValue(processValue)
```

4.5 Garden:

IoT devices/sensors	Connection Port
Lawn sprinklers	D0, D1 to SBC
Humidity sensor	D2 to SBC
Fire monitor	D3 to SBC
Fire sprinklers	D4 to SBC
Siren	D5 to SBC
Photo sensor	D6 to SBC
Light	D7 to SBC

Overall Garden System Behavior

- Fire Detection: Triggered by the fire monitor, activating the siren and fire sprinklers.
- Humidity Monitoring: Humidity levels are checked using humidity sensor, triggering lawn sprinklers below a 50% threshold to conserve water.
- Light Control: Managed by a photo sensor to control garden lights based on day/night changes.
- Siren and Sprinkler Activation: Managed by the SBC according to sensor inputs.

Functional Overview

1. Fire Response Handling:

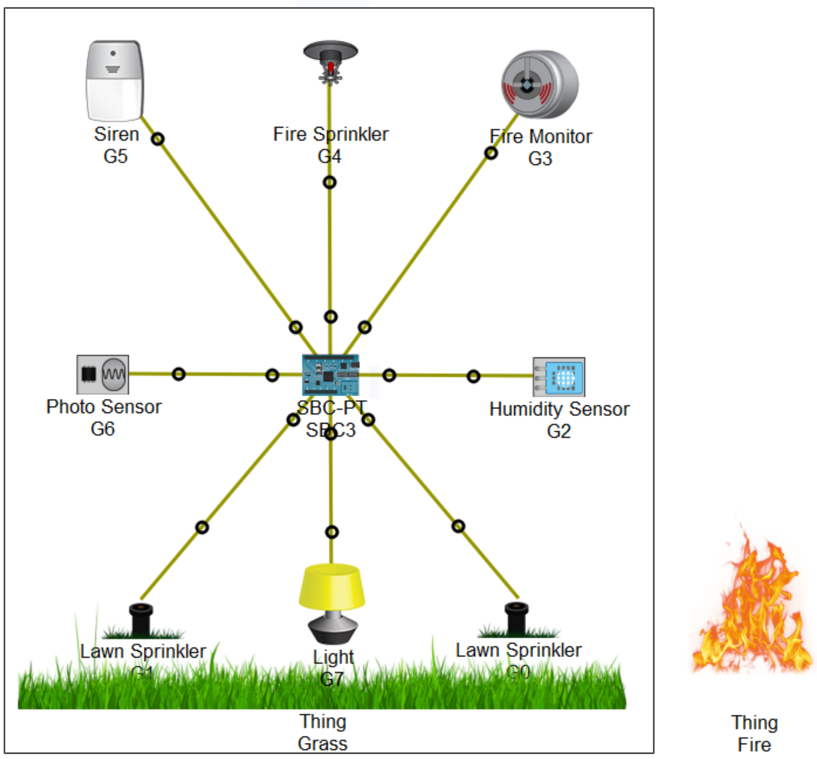
- The siren and fire sprinklers are activated whenever the fire monitor detects flames.
- The components are connected to an SBC microcontroller, managed by specific pin assignments.

2. Humidity and Lawn Irrigation Control:

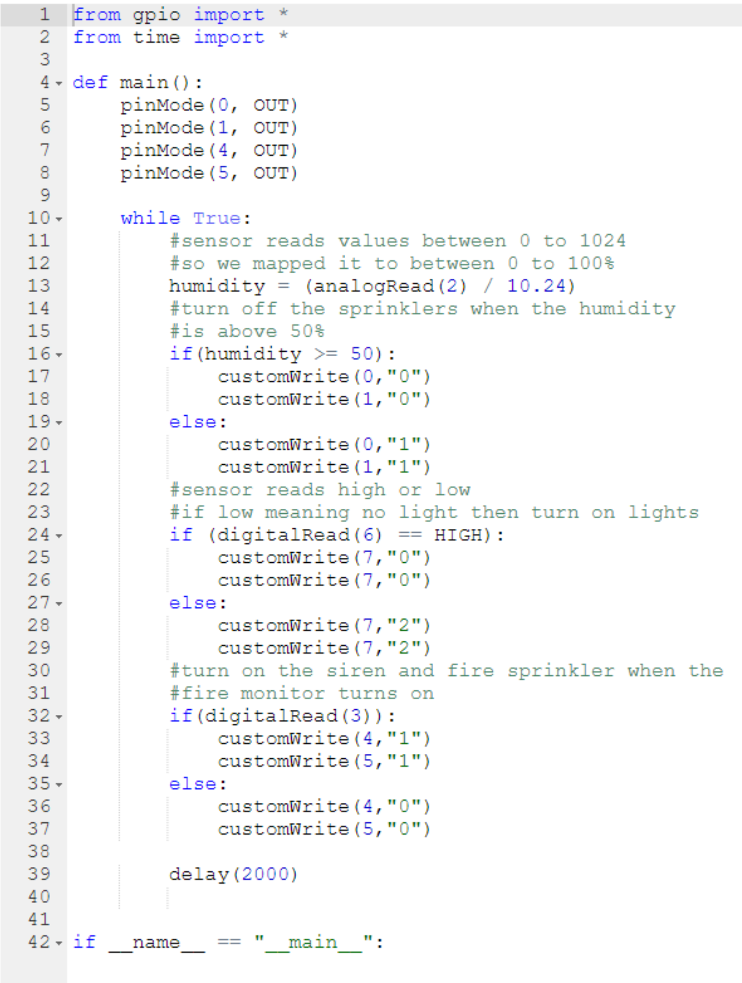
- A humidity sensor monitors the garden's moisture levels.
- Lawn sprinklers activated if humidity falls below 50%, aimed at water preservation.

3. Day/Night Light Control:

- A photo sensor detects ambient light levels, controlling the garden lights based on day or night conditions.



Garden SBC Code:



```
from gpio import *
from time import *

def main():
    pinMode(0, OUT)
    pinMode(1, OUT)
    pinMode(4, OUT)
    pinMode(5, OUT)

    while True:
        #sensor reads values between 0 to 1024
        #so we mapped it to between 0 to 100%
        humidity = (analogRead(2) / 10.24)
        #turn off the sprinklers when the humidity
        #is above 50%
        if(humidity >= 50):
            customWrite(0,"0")
            customWrite(1,"0")
        else:
            customWrite(0,"1")
            customWrite(1,"1")
        #sensor reads high or low
        #if low meaning no light then turn on lights
        if (digitalRead(6) == HIGH):
            customWrite(7,"0")
            customWrite(7,"0")
        else:
            customWrite(7,"2")
            customWrite(7,"2")
        #turn on the siren and fire sprinkler when the
        #fire monitor turns on
        if(digitalRead(3)):
            customWrite(4,"1")
            customWrite(5,"1")
        else:
            customWrite(4,"0")
            customWrite(5,"0")

        delay(2000)


if __name__ == "__main__":
```

4.6 Student Affairs Room:

IoT devices/sensors	Connection Port	home gateway
Push Button	D0, D1 to MCU	-
LCD	D2 to MCU	-
LED	D3 to MCU	-
Motion sensor	D4 to MCU	-
Fan	D5 to MCU	-
Temperature sensor	A0 to MCU	-
Heating & Cooling	D0, D1 to SBC	-
RFID readers	-	192.168.25.1
Doors	-	192.168.25.1
PCs	-	192.168.25.1

Overall Student Affairs Office Behavior

- Student Arrival: Detected by the motion sensor, adding the student to the queue.
- Student Appointment Handling: Students are assigned to offices as they become available.
- Departure Detection: Button presses indicate a student has left, making the office available.
- Environmental Comfort: Temperature is monitored using temp sensor and managed via fan control.
- Environment Control: Temp is changed using heating & cooling elements for testing.
- Backdoor Access: Admins access their own room using their respective office tag.

Functional Overview

1. LED and LCD Management:

- The LED is turned on if at least one office is vacant and off when both are occupied.
- The LCD displays whether the office is "VACANT" or "BUSY" and shows which student is assigned to which office once they are assigned.

2. Queue Management:

- Motion sensor detects student entries, assigning them a number and adding them to a queue.
- Button presses that signal an office has become vacant allow for the next student in the queue to be assigned to that office.

3. Environment Control:

- programed an SBC with heating & cooling elements connected to test the room temperature sensor.

4. Temperature Control:

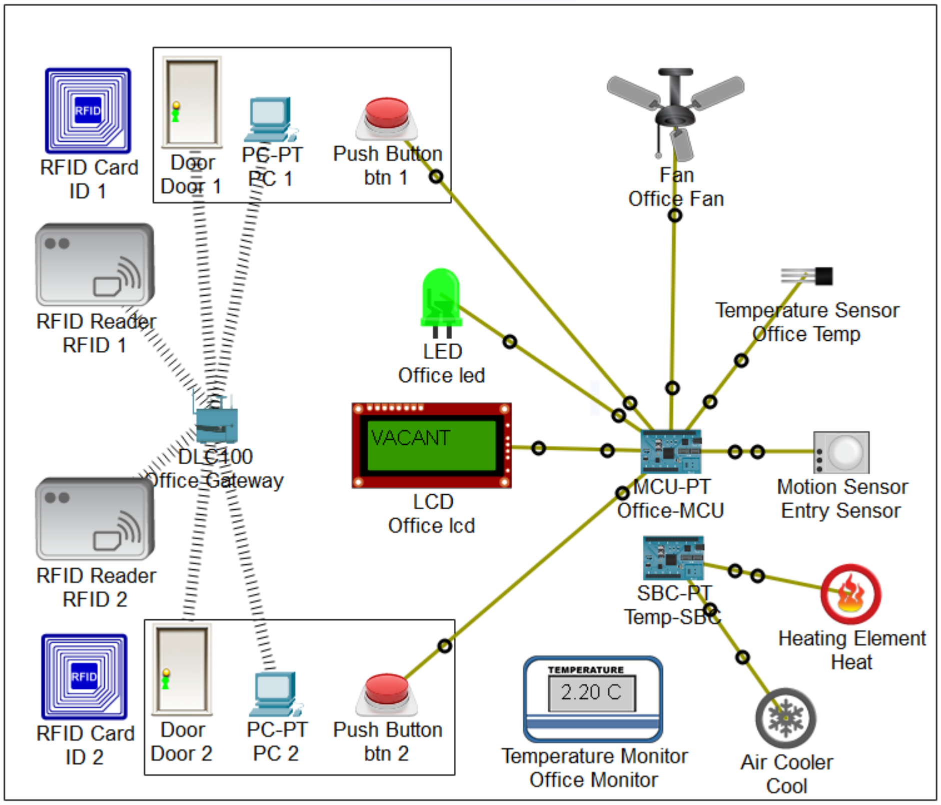
- Room temperature is monitored using a sensor integrated into the system control.
- The fan's operation is controlled based on temperature readings: medium speed between 25°C and 35°C, high speed above 35°C, and off below 25°C.

5. Office Assignment:

- Continuously checks if an office is available and assigns students from the queue to vacant offices.
- Updates the display and LED status as students are assigned or complete their visits.

6. Backdoors Lock Control:

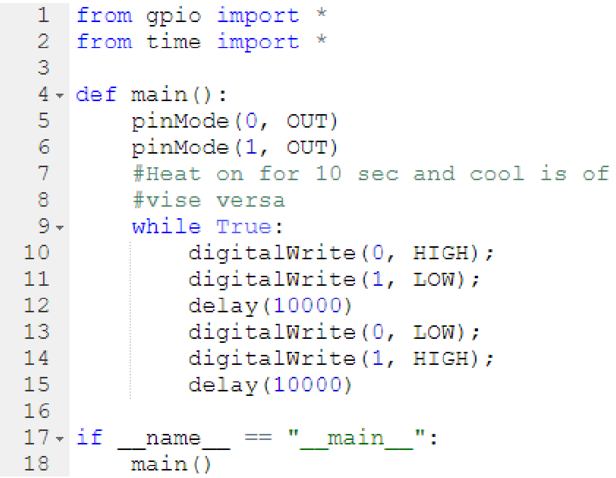
- RFID readers and backdoors are connected to a home gateway and programmed using conditions.



Office RFIDs Conditions:

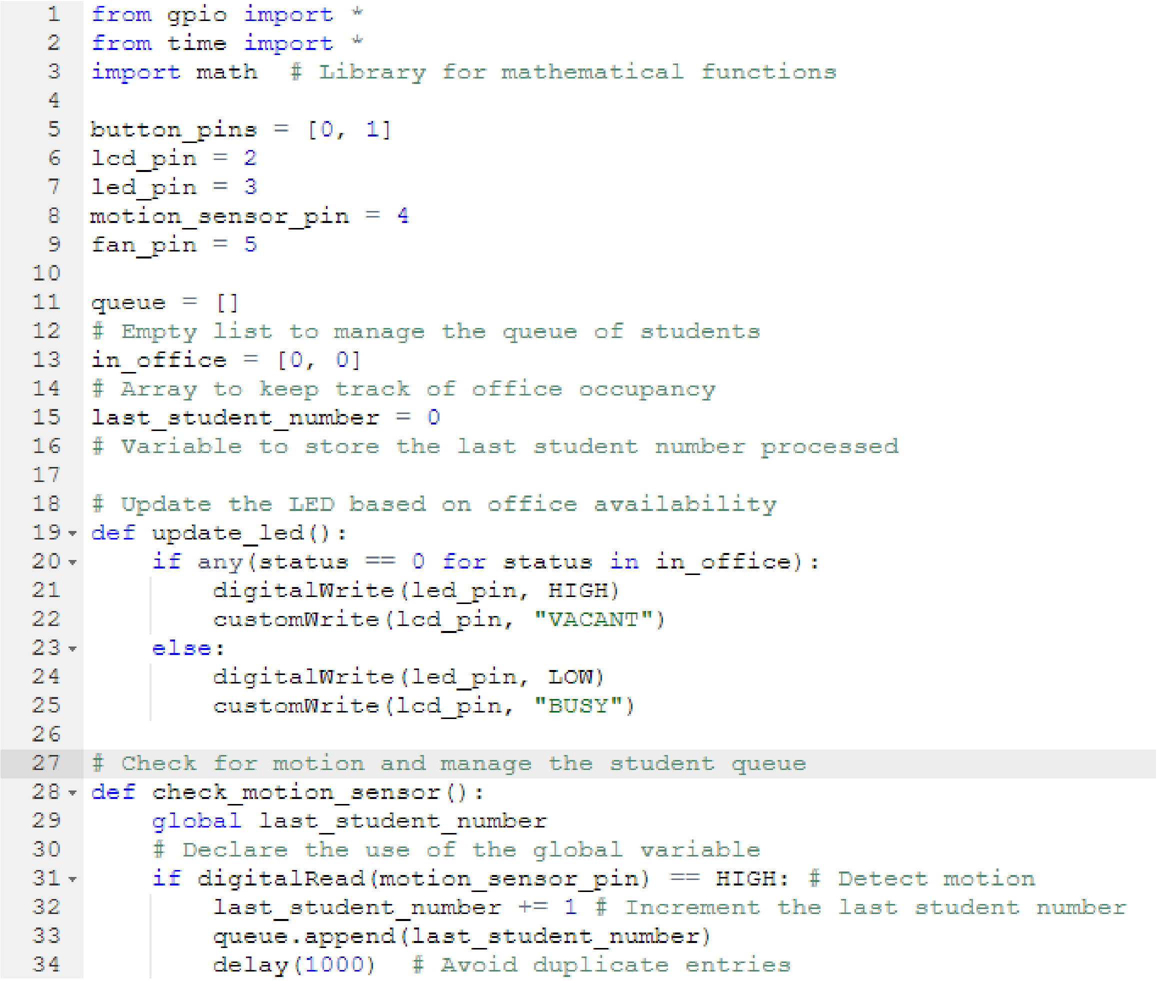
Name	Condition	Actions
ID 1	Office 1 Card ID = 1001	Set Office 1 Status to Valid
Door 1 Open	Office 1 Status is Valid	Set Door 1 Lock to Unlock
Door 1 Close	Match any: • Office 1 Status is Invalid • Office 1 Status is Waiting	Set Door 1 Lock to Lock
ID 2	Office 2 Card ID = 1002	Set Office 2 Status to Valid
Door 2 Open	Office 2 Status is Valid	Set Door 2 Lock to Unlock
Door 2 Close	Match any: • Office 2 Status is Invalid • Office 2 Status is Waiting	Set Door 2 Lock to Lock

Environment Control SBC Code:

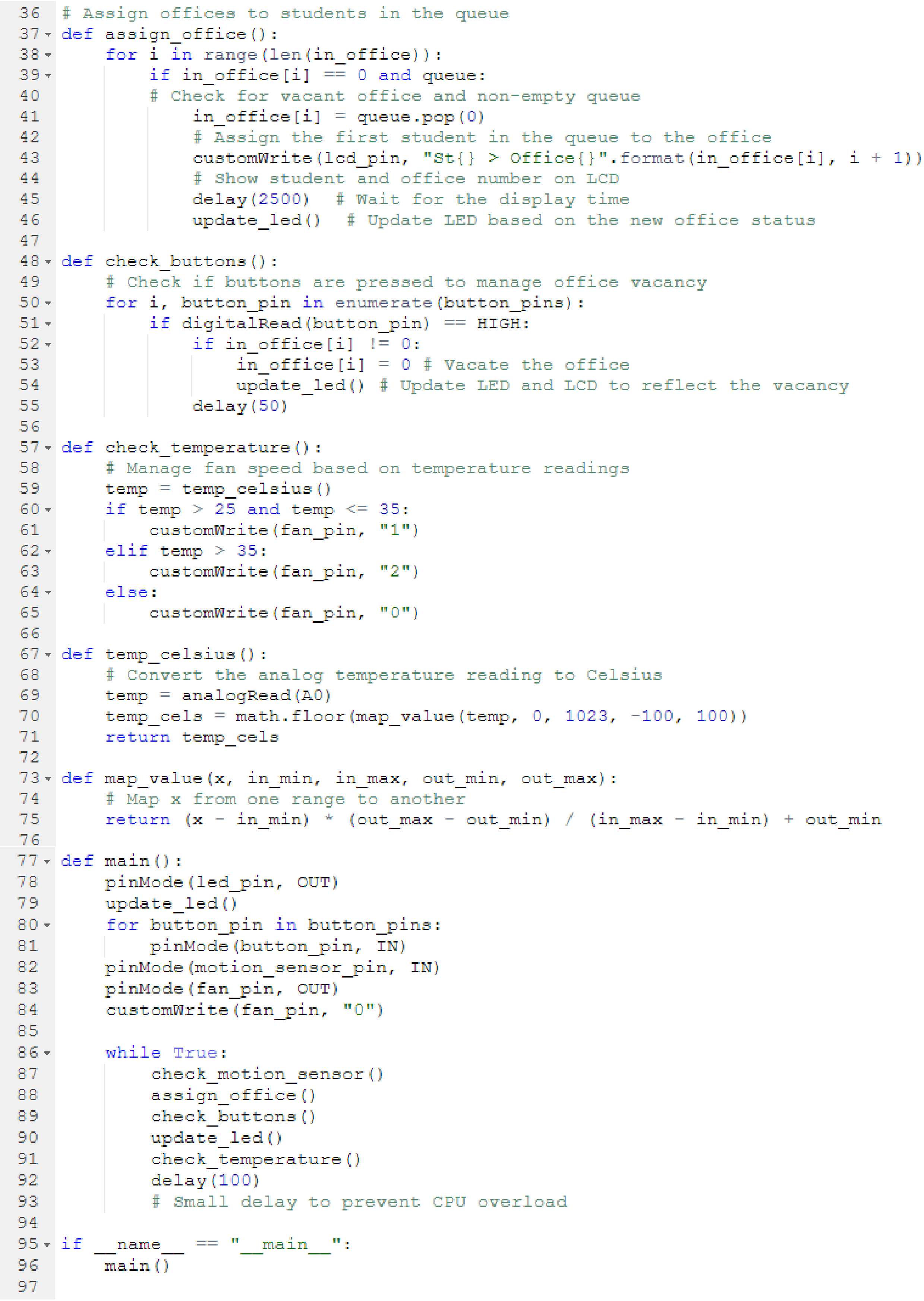


```
from gpio import *
from time import *

def main():
    pinMode(0, OUT)
    pinMode(1, OUT)
    #Heat on for 10 sec and cool is of
    #vise versa
    while True:
        digitalWrite(0, HIGH);
        digitalWrite(1, LOW);
        delay(10000)
        digitalWrite(0, LOW);
        digitalWrite(1, HIGH);
        delay(10000)

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

Office MCU Code:



```
from gpio import *
from time import *
import math  # Library for mathematical functions

button_pins = [0, 1]
lcd_pin = 2
led_pin = 3
motion_sensor_pin = 4
fan_pin = 5

queue = []
# Empty list to manage the queue of students
in_office = [0, 0]
# Array to keep track of office occupancy
last_student_number = 0
# Variable to store the last student number processed

# Update the LED based on office availability
def update_led():
    if any(status == 0 for status in in_office):
        digitalWrite(led_pin, HIGH)
        customWrite(lcd_pin, "VACANT")
    else:
        digitalWrite(led_pin, LOW)
        customWrite(lcd_pin, "BUSY")

# Check for motion and manage the student queue
def check_motion_sensor():
    global last_student_number
    # Declare the use of the global variable
    if digitalRead(motion_sensor_pin) == HIGH: # Detect motion
        last_student_number += 1 # Increment the last student number
        queue.append(last_student_number)
        delay(1000)  # Avoid duplicate entries
```



```python
# Assign offices to students in the queue
def assign_office():
    for i in range(len(in_office)):
        if in_office[i] == 0 and queue:
        # Check for vacant office and non-empty queue
            in_office[i] = queue.pop(0)
            # Assign the first student in the queue to the office
            customWrite(lcd_pin, "St{} > Office{}".format(in_office[i], i + 1))
            # Show student and office number on LCD
            delay(2500)  # Wait for the display time
            update_led()  # Update LED based on the new office status

def check_buttons():
    # Check if buttons are pressed to manage office vacancy
    for i, button_pin in enumerate(button_pins):
        if digitalRead(button_pin) == HIGH:
            if in_office[i] != 0:
                in_office[i] = 0 # Vacate the office
                update_led() # Update LED and LCD to reflect the vacancy
            delay(50)

def check_temperature():
    # Manage fan speed based on temperature readings
    temp = temp_celsius()
    if temp > 25 and temp <= 35:
        customWrite(fan_pin, "1")
    elif temp > 35:
        customWrite(fan_pin, "2")
    else:
        customWrite(fan_pin, "0")

def temp_celsius():
    # Convert the analog temperature reading to Celsius
    temp = analogRead(A0)
    temp_cels = math.floor(map_value(temp, 0, 1023, -100, 100))
    return temp_cels

def map_value(x, in_min, in_max, out_min, out_max):
    # Map x from one range to another
    return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min

def main():
    pinMode(led_pin, OUT)
    update_led()
    for button_pin in button_pins:
        pinMode(button_pin, IN)
    pinMode(motion_sensor_pin, IN)
    pinMode(fan_pin, OUT)
    customWrite(fan_pin, "0")

    while True:
        check_motion_sensor()
        assign_office()
        check_buttons()
        update_led()
        check_temperature()
        delay(100)
        # Small delay to prevent CPU overload

if __name__ == "__main__":
    main()

```

5. Conclusion & Future Work:

Our "Smart College" project exemplifies the transformational impact of integrating IoT technologies into educational environments. By deploying advanced IoT devices and sensors across various campus facilities, including lecture halls, gardens, libraries, and access points, we have significantly enhanced operational efficiency, security, and user experience.

This comprehensive system showcases practical applications in several key areas: automated entry and exit mechanisms at the front gate, environmental monitoring in gardens, and enhanced security in libraries. These implementations not only improve resource management and facility automation but also bolster campus security.

As we continue to innovate, it is crucial to prioritize robust security measures to mitigate risks associated with the increased connectivity and data exchange that IoT systems entail. By responsibly embracing these technologies, our smart campuses can evolve into dynamic, efficient, and secure learning environments that substantially enrich both the teaching and learning experiences.

6. Reference:

Course Labs.

Trial and error in Packet Tracer.

*** Note :** the following questions were answered indirectly throughout the report:

What will the program do?

What the input to the program will be?

What the output from the program will be?