Smart Room

Iorio Matteo, Vincenzi Fabio

December 2022

1 Introduction

1.1 Smart Room

We want to realise an IoT system implementing a simplified version of a smart room, as a smart system monitoring and controlling the state of a room (e.g. in a Campus). Room Sensor-Board (esp) embedded system to monitor the state of the room by using a set of sensors It interacts with the Room Service (via MQTT1) Room Service (backend - pc) service functioning as the main unit governing the management of the room it interacts through the serial line with the Controller (arduino) it interacts via MQTT with the Room SensorBoard (esp) it interacts via HTTP with the Dashboard (frontend/PC) Room Controller (Arduino) embedded system controlling lighting and roller blinds it interacts via serial line with the Room Service and via BT with the Room App Room App (Android - smartphone) mobile app that makes it possible to manually control lights and roller blinds it interacts with the Room Controller via Bluetooth Room Dashboard (Frontend/web app on the PC) front-end to visualise and track the state of the room it interacts with the Room Service

Hardware components

Room Sensor-board SoC ESP32 board (or ESP8266) including a green led 1 PIR 1 photoresistor analog sensor Room Controller Microcontroller Arduino UNO board including: 1 green led simulating a light subsystem 1 servo motor simulating the roller blind subsystem 1 Bluetooth module HC-06 o HC-05

General Behaviour of the system

The Smart Room system is meant to control the lighting system and roller blinds according to the following policy:

If no one is in the room, the light (of the lighting subsystem) should be off If someone enters in the the room and the room is dark, then the light should be turned on (if it was off) The roller blinds are fully rolled up automatically the first time someone enters in the room, from 8:00 (if someone enters) The roller blinds are fully unrolled at 19:00 (if they are up and no one is in the room), or as soon as someone who is still in the room at 19:00 leaves the room. Through the mobile app, a user can: turn on or off the light roll up / unroll – also partially (from 0 to 100) Through the dashboard a room manager can: track the state of the room in particular in which hours and how long the lights where on fully control the light and roller blinds

Remark: the light controlled by this policy representing the lighting system is represented by the green led in the Room Controller.

It can be assumed that the room is accessed from 8:00 to 19:00.

Further details:

About the Room Sensor-board The led should be on when someone is in the room and off when no one is the room About the Room Controller The servo motor controls/simulates the roller blinds 0° means roller blinds completely rolled-up 180° means roller blinds completely unrolled the green light simulates the lighting system: on/off

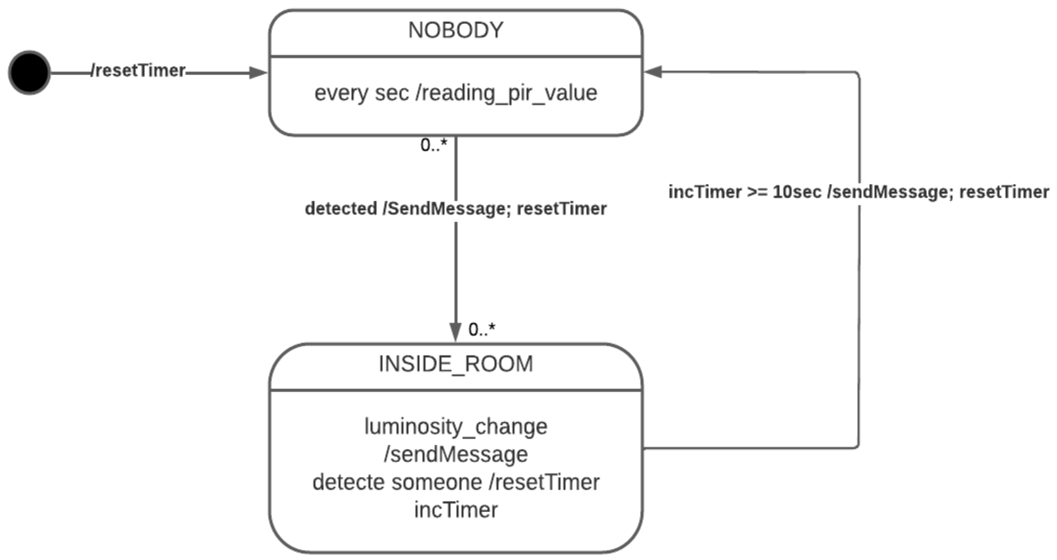
No specific constraints/requirements are given for the Room Mobile App and the Room Dashboard

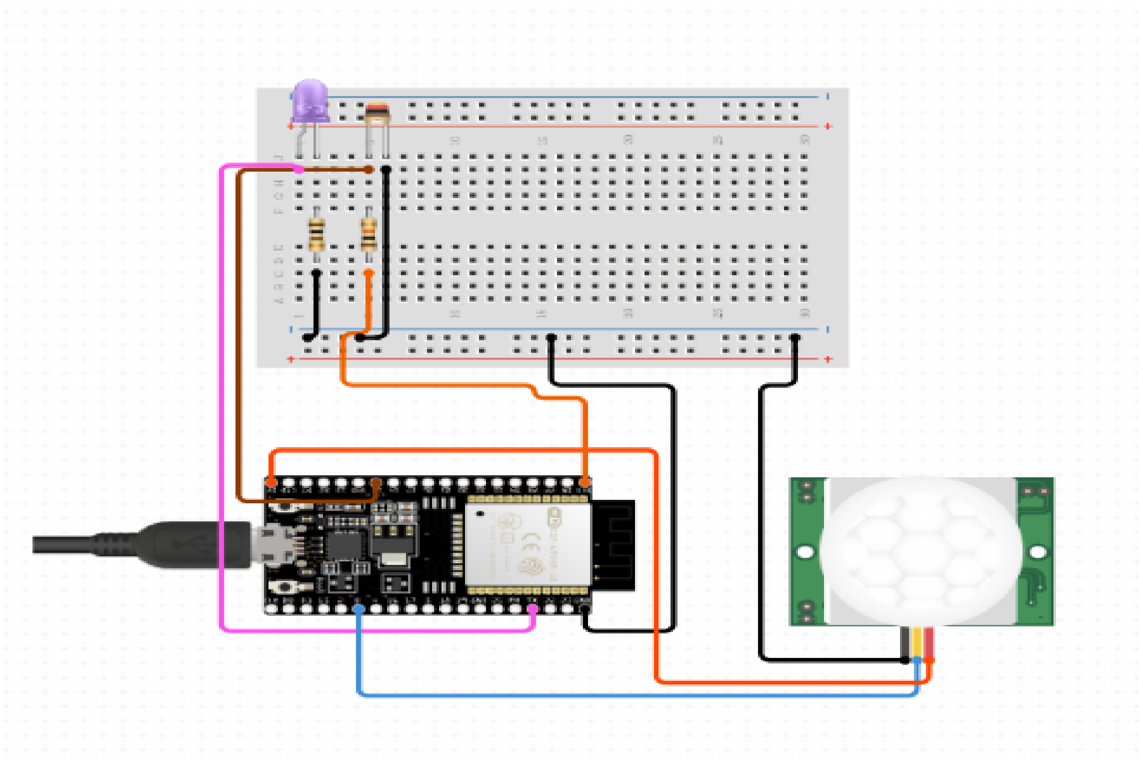
2 Room Sensor Board

2.1 Introduction

Inside the Room Sensor Board we managed the ESP-32. Inside this component we can find a single led, that is turned ON if the PIR detect someone and then the led is turned OFF if inside the room there is no one anymore. We used as protocol for the communication with the Server the MQTT protocol. We send to the Broker a JSON file structured like this :
{
"who": "Room Sensor-Board"
"inside_room": True if someone is inside the room False otherwise
"state": ON if the led should be ON, OFF otherwise
"time": The event's time formatted as HH:MM:SS
}
The key "inside_room" is set by the PIR, if someone is inside the Room this value is True, False otherwise. When the PIR detects someone, the Photoresistor catchs the value of the luminosity inside the Room, if there is too much lite then the led should be OFF "state":"OFF", otherwise the led should be on "state":"ON". A message is sent when the Pir detects someone or when the "state" changes but in this case the value of "inside_room" must be True. And last the message is sent when a Person leaves the Room. The time is recovered by doing a request to a NTP_SERVER

Link to see the entire Room Sensor-Board. https://github.com/MatteoIorio11/iot-assignments/tree/smart-room/room-board


/resetTimer
NOBODY
every sec /reading_pir_value
0..*
detected /SendMessage; resetTimer
0..*
incTimer >= 10sec /sendMessage; resetTimer
INSIDE_ROOM
luminosity_change
/sendMessage
detecte someone /resetTimer
incTimer



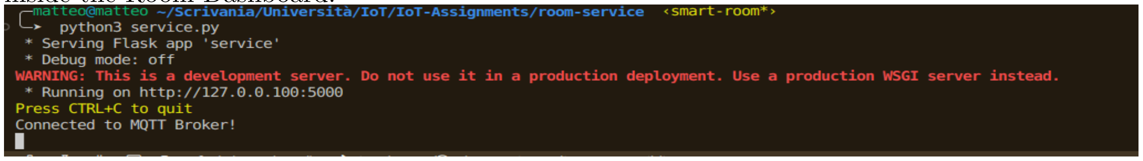
3 Room Service

3.1 Introduction

The Room Service is the key component inside the structure. Our Room Service is done in Python using three technologies :
1) Flask : for the HTTP server
2) Paho.MQTT : for the MQTT's Agent
3) Serial : For the Serial communication between Python and Arduino.

Inside the Room Service we can find three thread, one each for every technology. All the data are "protected" using locks in order to not have problems with the data. The Flask Thread communicate with the Room Dashboard using HTTP. The Client send JSON files to the Python Server, inside the Server the data are parsed and then sent to the Arduino using the Serial, always using the JSON format for example :
LED Format :
{
"hardware":"LED"
"state":True if the led should be ON otherwise False
}
SERVOMOTOR Format :
{
"hardware":"SERVOMOTOR"
"angle":X where X can be a value between 0 and 180.
}
For manage the Time, inside the Room Service when is the 8 o'clock we send a message to Arduino, the same thing is done where whe reach the 19 o'clock. With this strategy we inform the Arduino about the hour.

The MQTT Agent is subscribed to the Same Broker of the Esp, so whenever a message is sent to the Broker, we receive the message and we send a message to Arduino with the informations sent by the PIR. We also use the key "time" inside the packet for display the total seconds of light ON, this will be used inside the Room Dashboard.



```
╭─matteo@matteo ~/Scrivania/Università/IoT/IoT-Assignments/room-service  ‹smart-room*›
╰─➤  python3 service.py
 * Serving Flask app 'service'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
Connected to MQTT Broker!
```

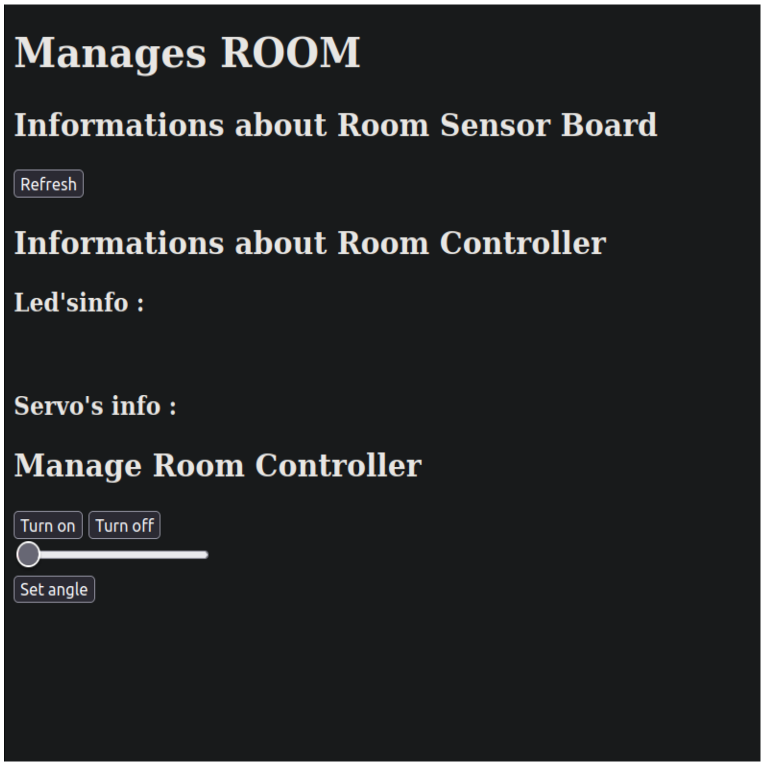
See the full implementation at https://github.com/MatteoIorio11/iot-assignments/tree/smart-room/room-service

4 Room Dashboard

4.1 Introduction

The Room Dashboard is a simple Web App done in HTML and JavaScript. The communication between the Room Service and the Room Dashboard is done by using the Axios library. At every click of each Button a request is done to the Server, in particular by using GETs.
GET:room_service_ip:PORT/api/data we receive the informations about the Room Controller's led, in particular for how much seconds the Led was ON "Informations about Room Senso Board".
GET:room_service_ip:PORT/api/servo?angle=X we can set the Servo's angle to X
GET:room_service_ip:PORT/api/led?state=ON/OFF we can set the Room Controller's led to ON or OFF.

The message between the Client and the Server are JSON formatted
{
"hardware":"LED"
"state":True if the led should be ON otherwise False
}
SERVOMOTOR Format :
{
"hardware":"SERVOMOTOR"
"angle":X where X can be a value between 0 and 180.
}



Manages ROOM

Informations about Room Sensor Board

Informations about Room Controller

Led'sinfo :

Servo's info :

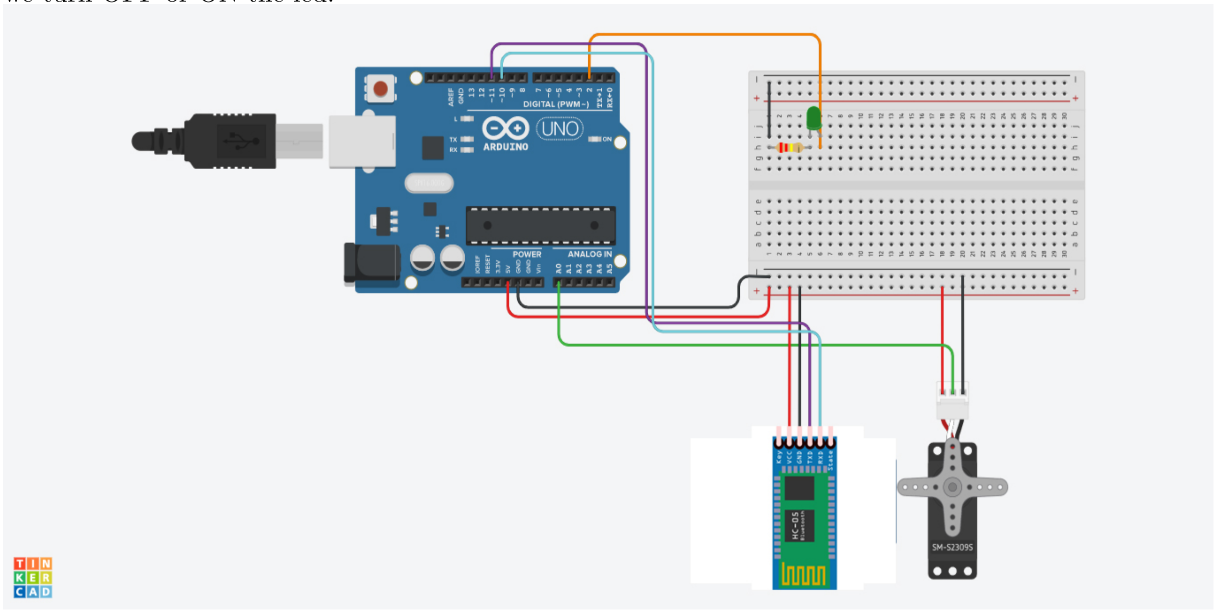
Manage Room Controller

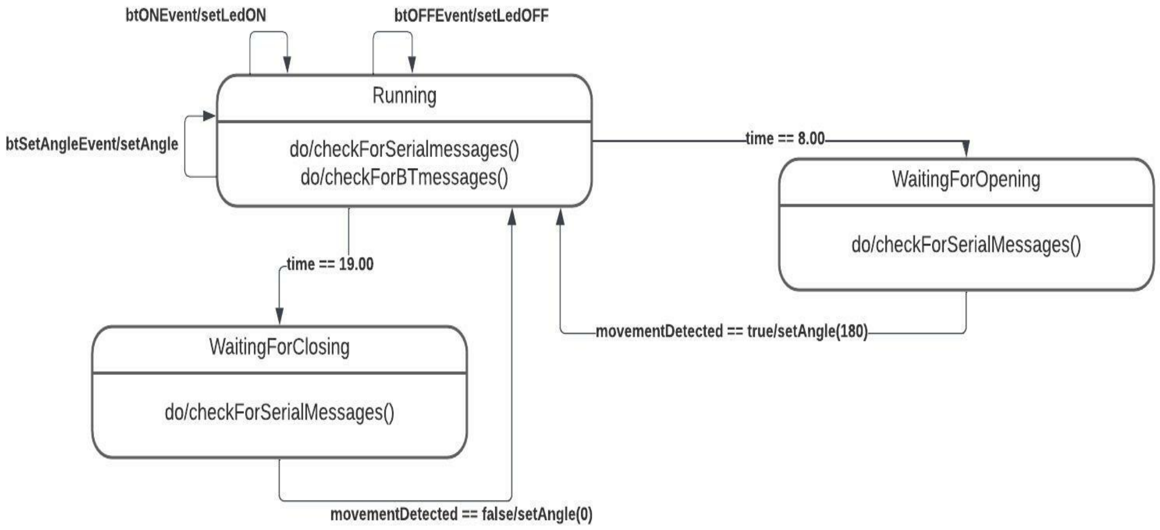
See the full code here
https://github.com/MatteoIorio11/iot-assignments/tree/smart-room/room-dashboard

5 Room Controller

5.1 Introduction

The Room controller is composed by a Led, representing our Light and a Servomotor. Arduino communicates with the Room Service using the Serial and the messages are JSON formatted, the same is done with the Room App. The Room Controller always check for new messages from the serial (Room Service) or the Bluetooth (Room App), when a new message is detected we check for the type of operation required and then we open the servo at a certain angle or we turn OFF or ON the led.



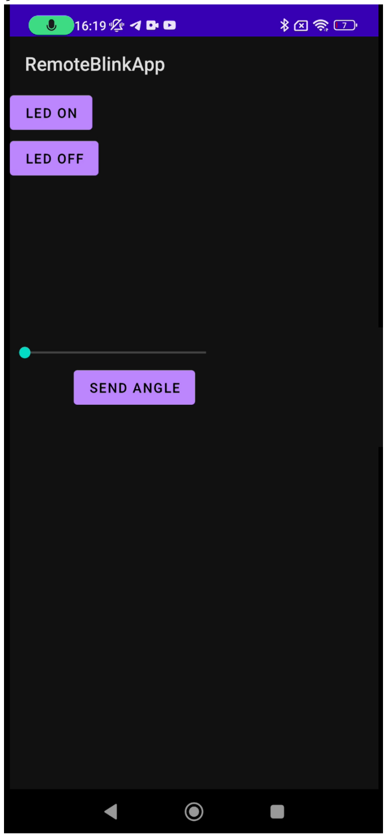

btONEvent/setLedON
btOFFEvent/setLedOFF
btSetAngleEvent/setAngle
Running
do/checkForSerialmessages()
do/checkForBTmessages()
time == 8.00
WaitingForOpening
do/checkForSerialMessages()
time == 19.00
movementDetected == true/setAngle(180)
WaitingForClosing
do/checkForSerialMessages()
movementDetected == false/setAngle(0)

See full code
https://github.com/MatteoIorio11/iot-assignments/tree/smart-room/room-controller

6 Room App

6.1 Introduction

The Room App is a simple Android App with three main options :
1) Turn ON the led
2) Turn OFF the led
3) Set the Servo's angle
The communication between the App and Arduino is by using Bluetooth, all the messages are sent as a JSON file : {
"hardware":"LED"
"state":True if the led should be ON otherwise False
}
SERVOMOTOR Format :
{
"hardware":"SERVOMOTOR"
"angle":X where X can be a value between 0 and 180.
}


16:19
RemoteBlinkApp
LED ON
LED OFF
SEND ANGLE