IoT Thermostat with Passive Notification System

Iacopo Erpichini
Università degli Studi di Firenze
Computer Engineering
iacopo.erpichini@stud.unifi.it

***Abstract*—This paper provides a new interface for notifications using the idea of Ubiquitous computing in a smart home system.**

It is realized with a Micro Controller with a Wi-fi module on board, it is designed for all types of sensor and shows the notification in passive way with a RGB Wi-Fi bulb. In this case the bulb is connected to a temperature sensor and the notification are about the measured temperature by the sensor. Substantially how much cold or hot the house is. The thermostat system is built using low-cost components and Open Source platforms to make it reproducible by everyone.

In this project have been developed a physical device and an interface via a Telegram bot.

***Index Terms*—IoT, NodeMcu ESP-12, Arduino, Ubiquitous computing, domotic system, Telegram**

I. Introduction

The Internet of Things is creating much buzz while it goes about transforming our lives. IoT is everywhere, even though we don't always see it or know that a device is part of the IoT. Furthermore, the Internet of Things is turning physical object into a complete ecosystem of information, shared between devices that are wearable, portable, and even implantable, turning our lives rich of technology and data.

Nowadays, the smartest IoT home automation devices allow users to control them via an app or even via voice commands.

The purpose of this paper is to build a IoT system for passive notification using the idea of Ubiquitous computing [1] (or "ubicomp") that is a concept in software engineering and computer science where computing is made to appear anytime and everywhere.

In this case the system wants to control a simple IoT device that is an RGB bulb with Wi-Fi module build in to make passive notification using the property of the bulb of changing color and brightness in a certain range of time.

The core of the system is a chip named Node Mcu v1.0 ESP-12 Figure [1].

The system is designed to be built using low-cost components and Open Source platforms in such a way as to be simply reproduced by a user with few computer skills.

The micro controller can be programmed via Arduino IDE [2], the code is written in C++ with an addition of special methods, functions and libraries, to integrate every sensor of the project. The 'sketch' (the name given to Arduino code files) created, it is processed and compiled to machine language and flashed on board via the IDE.

Fig. 1. Node Mcu V1.0 ESP-12 with ESP 8266 Wi-Fi chip on board.

The entire code and more instruction to reproduce the system are on GitHub repo [8], the entire code has been made in the most generic way.

It is designed to be able to change the operation of the system, for example replacing the temperature sensor with a gas detection sensor or with an infrared sensor.

II. Functionality

The system has two main functions:

- **Device:** The aim is to provide a device that can be installed in an environment connected to a smart bulb that changes color and light intensity in relation to the temperature measured by a sensor.
  The light bulb color changes slowly over a long period of time otherwise we would not have the effect of passive notification, but more a kind of stroboscopic effect.
  The device has a temperature sensor, a WiFi connection, a buzzer to receive active notification via interaction with an interface and a button to stop the change of color bulb.
  There is also a build-in led that displays in real time the state of the temperature sensor.
- **Interface:** The interface is realized through a Telegram Bot, using Universal Telegram Bot library [3] to interact directly with the micro controller and its sensors from the outside.
  This allows users to get real time information about temperature and humidity and to select a bound of desired temperature to change the color of the bulb.
  By default the temperature bounds are set on [16,24] Celsius degree and the bulb turns to red slowly if the

temperature is higher than 24 degree and turns blue if is lower than 16 degree.

It is also possible to notify people near the sensor with a buzzer.

In conclusion this system allows us to passively notify users who are in the room where this smart bulb is installed simply through the change of color and light intensity of the smart lamp, the color change must not be invasive so it is carried out over a medium-long period of time.

The temperature of the house where it is installed needs to be changed through a classical thermostat.

III. Hardware Component

A. Node Mcu v1.0 ESP-12

This micro controller has on board the ESP8266 Wi-Fi module and it is basically a System on Chip (SoC). It has Lua based firmware which is used to develop IoT based applications. It is easy to work on this inexpensive SoC and make our device smart.

The development board reported is useful for the cable management when the system is build.

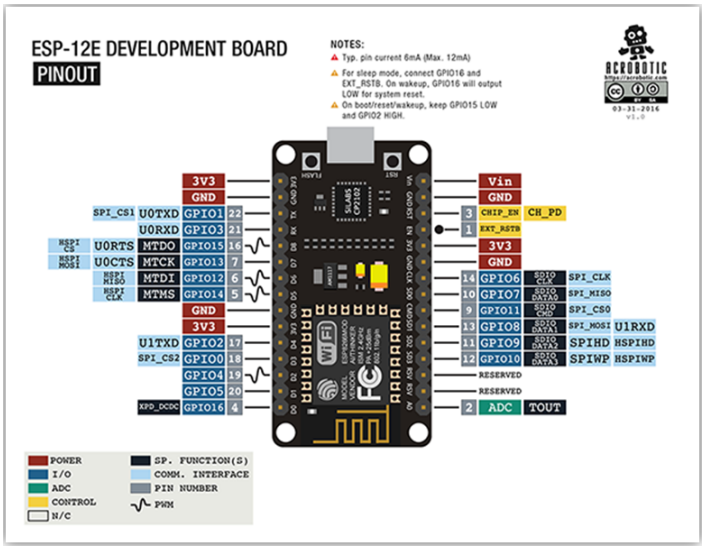

Fig. 2. Node Mcu V1.0 ESP-12 datasheet pin-out.

ESP-12 is the heart of the project, it allows us to connect to the internet and fully manage all the sensors connected to it, furthermore once connected to the Wi-Fi of the environment in which it is located the chip runs the instance of the Telegram Bot used to control the sensor.

B. DHT-22

The DHT-22 is a basic, low-cost, digital, temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin. It is fairly simple to use, but requires careful timing to grab data. The only real downside of this sensor is that you can only get new data from it once every 2 seconds.

It is used in the system to monitor the temperature and control the led on board and a smart bulb; if the measured temperature is higher or lower than the bounds saved in the micro controller the light changes colors.

Fig. 3. DHT 22 Sensor for Temperature and Humidity

C. RGB Led Common Anode

This is an RGB led installed on the device that displays the state of the device real time: the light color is green if the temperature measured in real time is in the bounds; it turns blue or red if the temperature is respectively under or over the bounds.

RGB led are of two types: some have common positive terminal (anode) and some have common negative terminal (cathode). The terminal is the longest leg of the led since they are indistinguishable, it is possible to distinguish them according to their functioning attached at a power supply.

Fig. 4. RGB Led Common Anode

D. Philips Hue

Philips Hue RGB is the bulb controlled by this system for the color changing notification system.

The bulb can be controlled via HTTP request according to the Philips documentation of Hue [4].

Fig. 5. Philips Hue Rgb bulb

E. Piezo Buzzer

The Piezo buzzer produces sound based on reverse of the piezoelectric effect. The generation of pressure variation or strain by the application of electric potential across a piezoelectric material is the underlying principle.

This buzzer is used to actively notify people near the IoT device.

Fig. 6. Piezo Buzzer

F. Push Button

The push button is a component that connects two points in a circuit when you press it.

In the device, when the button is hold it changes the temperature bounds; users can see the success of this action by a red blink given from the led mounted on the device.

This is useful because if a user sees the light turning to red or blue but doesn't want to open the telegram bot and change the temperature bound, user can press the button and the upper bound is incremented by a value chosen in the firmware of the micro controller and lower bound is decremented of this value.

Fig. 7. Mini switch push button

G. Li-Po Battery 3.7 V

A lithium polymer battery (Li-Po), is a rechargeable battery of lithium-ion technology using a polymer electrolyte instead of a liquid electrolyte.

It is connected to the Node Mcu at Vin and Gnd pin of the board to power the system without cable.

Fig. 8. Li-Po Battery 3.7 V

H. Other component

For this circuit is necessary to have other component in particular a transistor NPN 2n2222 to help life during of buzzer, jump wires, resistor, a breadboard and a USB to mini USB cable to flash the firmware on the micro controller.

To make the same circuit choose the resistor used according with a resistor color table.

IV. Circuit

The circuit was initially designed to be built with an Arduino Uno connected to an ESP8266 (Figure [12]), effectively using the Arduino as a bridge to flash the firmware on the Wi-Fi chip.

This initial circuit was subsequently discarded since the properties of the Arduino were not completely used and was more complex and more expensive than using an ESP-12 micro-controller.

The final result can be showed in Figure [13], this chip has directly on board the Wi-Fi module and it can be programmed in the same way as an Arduino.

A. 3D Model

Once the circuit had been designed, it was decided to also create a physical container to contain the sensors and the micro controller itself.

This model was designed using a 3d modeling software and was then printed on a Stereolithographic printer.

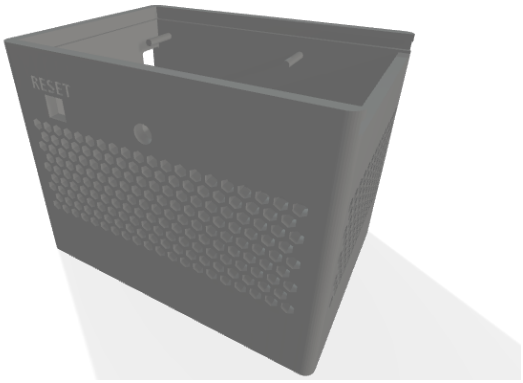

Fig. 9. Render of the box that contains the circuit without top cover.

Instead of using powder or filament, Stereolithography technology uses a liquid resin to produce 3D prints. The 3D files are provided in the GitHub Repository. In Figure [17] there is the result of 3D print.

B. Other ideas

The code for this project was written to be modeled by users with computer skills to reproduce the idea of Ubiquitous Computing in other types of sensors.

Still using a light bulb connected to the micro controller, two other circuit diagrams are provided:

- The first circuit that shows how to build a gas detection sensor that monitors air quality Figure [14].
- The second is an infrared sensor that could be used as an anti-theft / alarm if connected near a door Figure [15].

V. Software

The software side is composed by two parts: the sketch that controls the physical device and the interface provided via the Telegram bot.

As mentioned before, the code has been written to be as simple and generic as possible and has been heavily commented. This is because the device was designed to be replicated or extended in functionality.

A. Telegram Bot

Telegram is a freeware, cross-platform, cloud-based instant messaging (IM) software and application service. The service also provides end-to-end encrypted video calling, VoIP, file sharing and several other features.

An interesting feature are chat bots. There are third-party applications that run inside Telegram. Users can interact with bots by sending them messages, commands and inline requests. Bots are controlled using HTTPS requests to Telegram Bot API [3].

Telegram allows the user to create a chat bot using one of it called BotFather; simply by sending the command `/newbot` and following a few simple instructions, an instance of a bot will be generated that is accessible through a **unique bot token**. Subsequently, the code capable of interpreting the desired commands will be loaded on the micro controller.

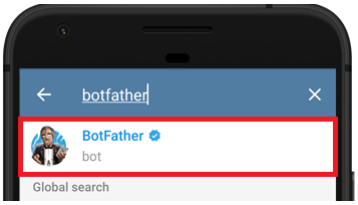

Fig. 10. Bot Father used to create bot in Telegram.

Anyone that knows bot username can interact with it. To make sure that the generated bot ignore messages that are not from any authorized users, it is possible to get **Telegram User ID** with another bot with the command `/getid`.

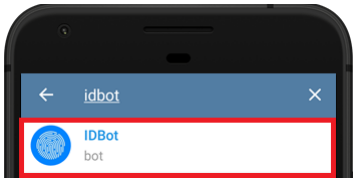

Fig. 11. IdBot a bot that allows user to get personal Telegram ID.

At this point with the two unique credentials it is possible to initialize the connection between the micro controller and Telegram in a safe way.

The features of the bot are simple and are shown in Figure [16], it is possible to control the temperature and the humidity detected by the device. Also, it is possible to set and check the desired temperature bound. At least, in case of emergency or to actively notify users near the device it is possible to reproduce an alarm sound.

B. Flash device

The code named firmware.ino is flashed in a ESP-12 micro controller. To make this possible is necessary to install an add-on for ESP8266 [5] for the Arduino IDE that allows users to program the ESP8266 using the Arduino IDE and its programming language.

The only thing to be done is to plug a board to a computer. Make sure to have the right port selected then click the "Upload" button in the Arduino IDE.

C. Sketch Arduino

The first part of the Sketch is a highly commented setup code, in fact there are some global variables that need to be changed.

These parameters are the Network SSID and password, the IP address of light bulb, the Telegram credential mentioned before and an extra public API token to make an integration with ThingSpeak which is an optional tool.

The main parts of the Sketch are:

- setup() function: is executed one time and it sets the initial parameters and establish the connection with local network.
- loop() function: the listener of all events. When a message is written to the bot the loop passes it to handleNewMessages() function, also the loop check the temperature bounds that are local variables of micro controller and set the Hue bulb color and brightness.
- handleNewMessages(): it defines all the actions of the telegram bot; the layout of keybord button sent to a user that messages with the bot; finally, responds to all the messages that are arrived at the micro controller via telegram
- setHue(String Command): set the color of the hue bulb. The Command is a String that represent a JSON with the hue parameters according to hue documentations. The command is sent to the light bulb with an HTTP GET request.
- uploadTemperatureHumidity(): this function sends an HTTP request like for the Hue to a site named ThingSpeak.
  It is used to save IoT data in cloud and make simple statistics based on the time course of the measurements.

D. ThingSpeak for IoT Projects

ThingSpeak is a free Data collection in the cloud with advanced data analysis using MATLAB software. Here is possible to see the application of use at this problem [6].

ThingSpeak enables sensors, instruments, and websites to send data to the cloud where they are stored in either a private

or a public channel. Once data are in a ThingSpeak channel, users can analyze and visualize it, calculate new data, or interact with social media, web services, and other devices.

Make a public channel with this tools is very simple (Guide [7]). Once created, the site provided to the user a Write API Key and a Read API Keys, for this project the site was used to create a history of the temperature and humidity data detected by the device.

VI. Usability Tests

The last phase of development focused on an evaluation of the prototype through a usability analysis. Two types of tests were implemented.

The first concerns a user without computer skills who finds the system completely assembled in his home to test chat bot and check if the prototype made is sufficiently user-friendly.

While the second test is aimed at users with IT skills and consists in replicating the whole system or circuit assembly, flashing the firmware once modified the code with local parameters and creating the Telegram bot as are explained in this article.

a) Test 1: The questions presented are of the SEQ type, with a scale ranging from 1 to 7, where 1 represents Strongly disagree and 7 Strongly agree. Furthermore, all the tasks are of the "closed" type, that is, there is only one procedure to carry them out.

The test was submitted at 8 people that have an age between 24 and 58 years old.

b) Test 2: This test is provided with this paper used like a sort of guide to assembly the circuit and retrieve information for all the questions.

The questions are about the assembly of the circuit, the creation of the view in ThingSpeak, the creation of Telegram Bot and using of Arduino IDE.

The test was submitted at two people with IT skills in particular one is passionate about the Arduino world, one is a computer engineer.

VII. Test Result

Here are reported the results of the tests carried out; it is possible to find the detailed results with all the answers in the GitHub repository.

TABLE I
Test 1 Result

Question	**Mean Rating**	σ
Find Bot on Telegram app and start messaging	6.75	0.187
Retrieve information about Temperature	6.875	0.109
Retrieve information about Humidity	6.875	0.109
Change Thermostat Bound	5.125	1.109
Check Thermostat Bound	6.375	0.734
Launch the alarm through bot	6.75	0.187
Reset physical device	6.375	0.484
The system is easy to learn	6	0.5
Overall, I am satisfied the system	6.125	0.359

TABLE II
Test 2 Result

Question	**Mean Rating**	σ
Build the circuit	6	1
Analyze firmware.ino code and change parameters	7	0
Install add-on in Arduino IDE	6.5	0.25
Flash the ESP 12	7	0
Create Telegram Bot	7	0
Create ThingSpeak Channel	6	1

VIII. Conclusion

The proposed system allows to have a physical device for passive notification of the temperature in a room. In addition, it allows users to have an IoT interface that can be remotely controlled via a highly customizable Telegram Bot.

A negative aspect that emerges from the first test is related to the setting of the bounds.

Unfortunately, the Telegram API only provides an interaction via text commands and therefore to set a bound, for example the lower one, it was decided to do it via a text string with the command `/ub=<intger number>`.

This behavior could be simplified if, for example, Telegram provides new API with a number picker linked to a text command or to a button.

The system is designed to be replicated by a user who has IT skills to make a standard environment more smart in every home and the test submitted to the two people that have IT skills confirms this result.

Furthermore, home automation is becoming increasingly popular and finding low-budget solutions will become more and more frequent.

IX. Future work

This project has been entirely designed to be modular and extensible. The idea of passive notification using a light bulb can be used in many fields as reported in the circuits shown earlier in this report.

References

[1] Ubiquitous Computing, Wikipedia
[2] Arduino IDE download, Arduino site
[3] Universal Telegram Bot library, GitHub, Brian Lough
[4] Philips Hue specification, Philips Hue Site
[5] ESP8266 Library, ESP8266 Community, Official Arduino
[6] ThingSpeak for IoT Projects, PassiveIotThermostat Public Channel,Channel
[7] ThingSpeak Official Documentation, Make a Channel, hrefhttps://www.mathworks.com/help/thingspeak/collect-data-in-a-new-channel.htmllink
[8] GitHub Repository of Passive Thermostat Notifier,link, Iacopo Erpichini

APPENDIX

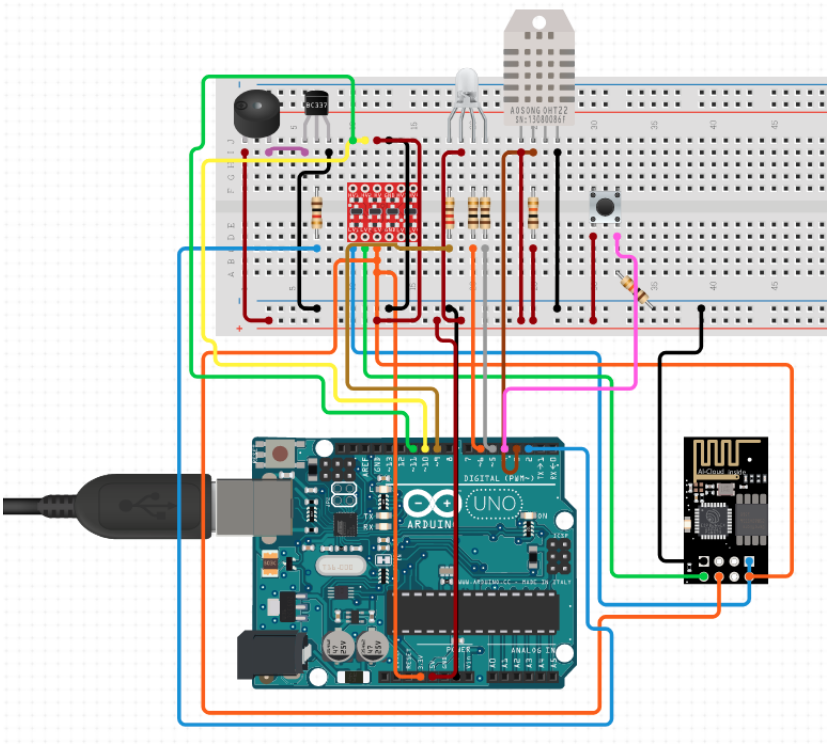

Fig. 12. The first circuit idea was with an Arduino Uno used like a bridge to flash the ESP8266 micro controller, there is also a power reducer from 5V to 3.3V to use correctly the ESP 8266 Chip.

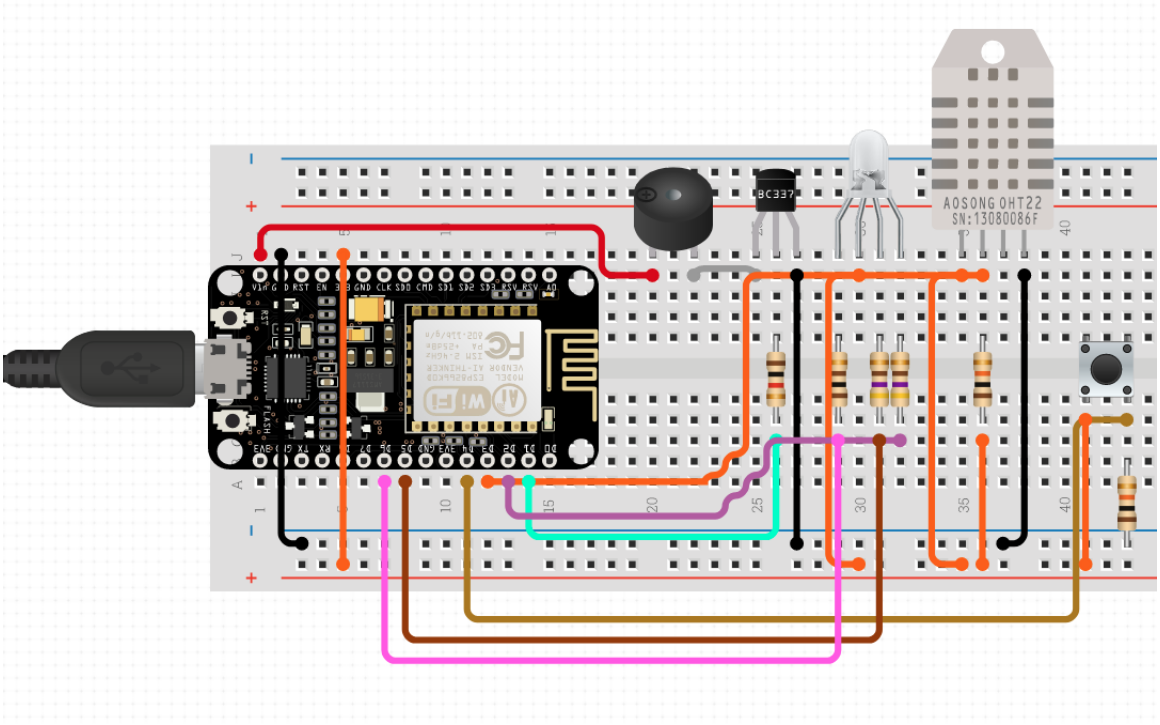

Fig. 13. Final Circuit realized with the ESP-12 Node Mcu v1.0 that have an ESP8266 Wi-Fi chip on board.

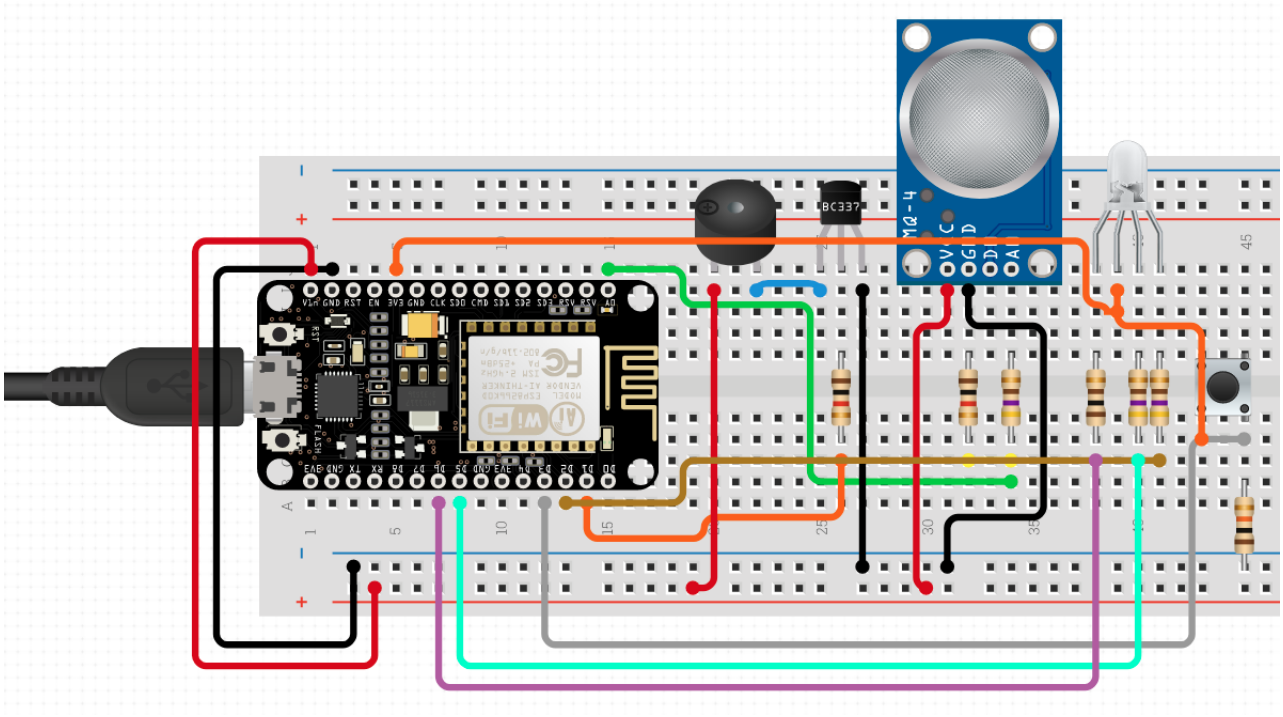

Fig. 14. Example 2 of Ubicomp: A circuit that connects a smart bulb with a Gas Sensor used to monitor the quality of the air.

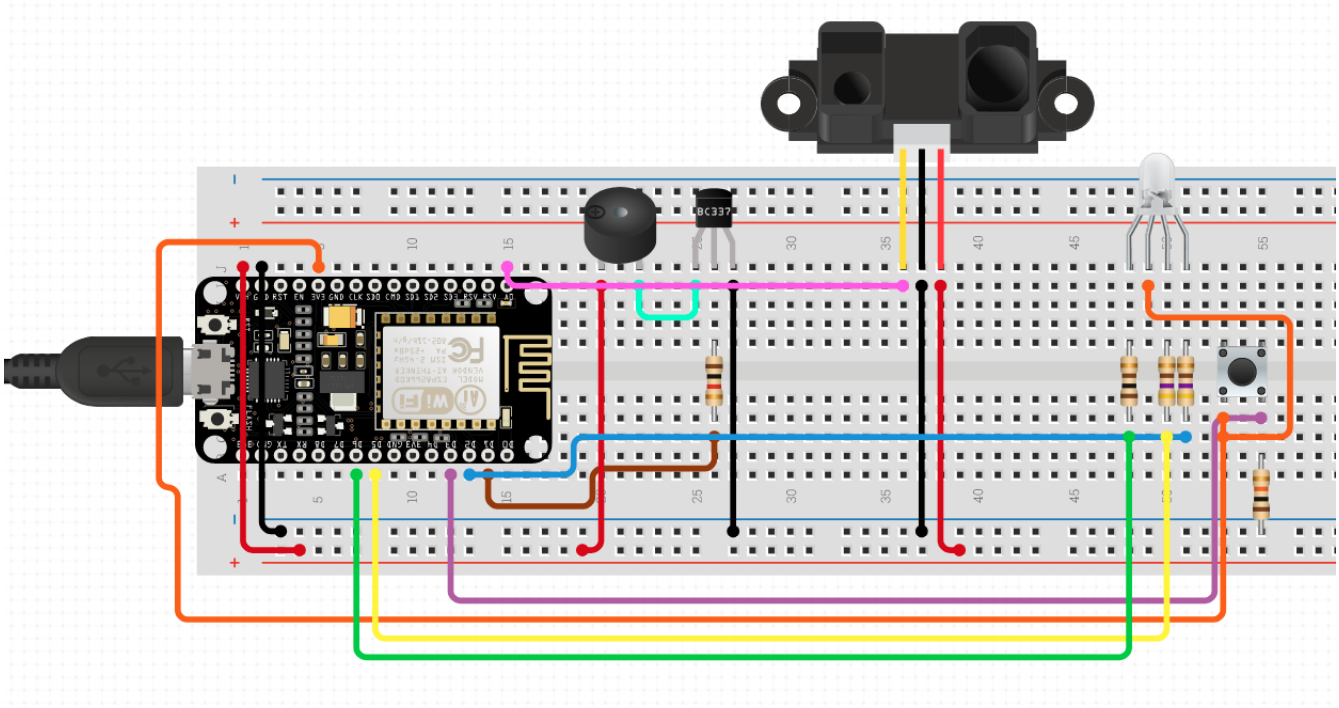

Fig. 15. Example 3 of Ubicomp: A circuit that connects a smart bulb with a Proximity Sensor maybe can be installed on a door jamb and used like an intrusion detector.

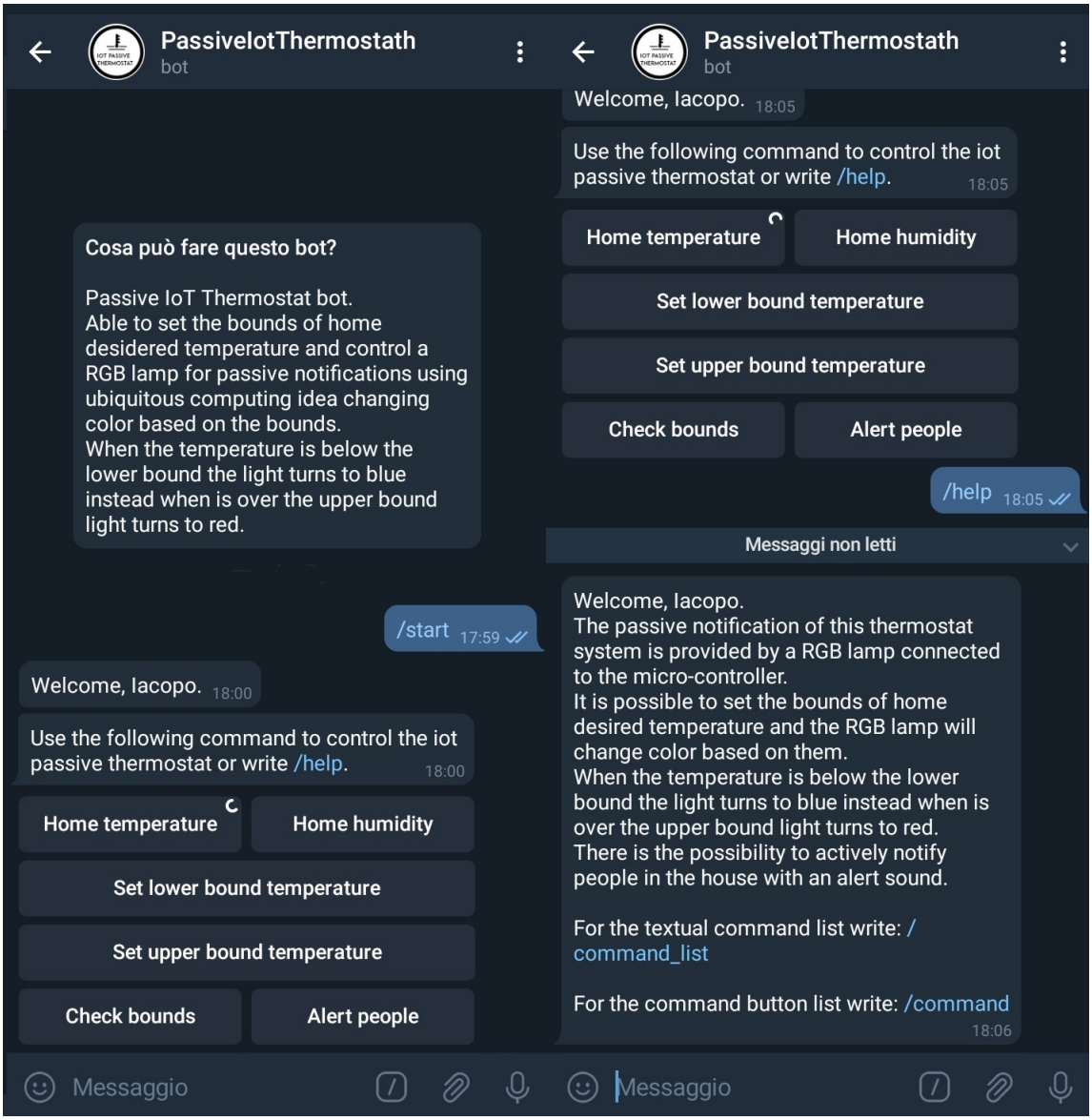

Fig. 16. PassiveIotThermostat bot. The bot send to the user a button keyboard to interact with the device. It is also possible to message him with the keywords suggested by typing "/".

Fig. 17. Passive IoT Thermostat device assembled.