ESCUELA TÉCNICA SUPERIOR DE INGENIEROS INDUSTRIALES Y DE TELECOMUNICACIÓN

UNIVERSIDAD DE CANTABRIA

Trabajo Fin de Grado

Diseño e implementación de un sistema domótico de vigilancia controlado por dispositivos embebidos

(Design and implementation of a smart surveillance system controlled with embedded devices)

Para acceder al Título de

Graduado en Ingeniería de Tecnologías de Telecomunicación

Autor: José Ramón Hoz Torre

Julio – 2021

E.T.S. Ing. Industriales
y de Telecomunicación
Universidad de Cantabria

UNIVERSIDAD DE CANTABRIA

CALIFICACIÓN DEL TRABAJO FIN DE GRADO

Realizado por: José Ramón Hoz Torre
Director del TFG: Roberto Sanz Gil
Título: "Diseño e implementación de un sistema domótico de vigilancia controlado por dispositivos embebidos"
Title: "Design and implementation of a smart surveillance system controlled with embedded devices"

Presentado a examen el día:

para acceder al Título de

GRADUADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN

Composición del Tribunal:
Presidente: Marta García Arranz
Secretario: Antonio Quintela Incera
Vocal: Roberto Sanz Gil

Este Tribunal ha resuelto otorgar la calificación de: .....

Fdo.: El Presidente

Fdo.: El Secretario

Fdo.: El Vocal Fdo.:

El Director del TFG
(sólo si es distinto del secretario)

Vº Bº del Subdirector

Trabajo Fin de Grado N.º
(a asignar por Secretaría)

Acknowledgements

First thing I want to say is a special *thank you* to my family. They have always had my back, no matter the circumstances, and they have supported and encouraged me through 24 years. This project is to show them what I have achieved and that their unconditional help and support led me to the man I am today.

Also, to my friends, the oldest and the newest ones, growing up with them forged my personality, for better or worse, but the most important thing is that I am very happy and satisfied for who I am and what I have achieved today.

Last but not least, to the manager and supervisor of this project, Roberto Sanz, I told him that I wanted to work on a project related to something I would feel passionate about, and he gave me the chance to discover the Raspberry Pi world and its potential. Thanks to this project, I am quite eager to keep exploring the IoT industry and I am very confident about what is going to be my next step which I consider it is crucial in this world we live.

Index

1.- Introduction

1.1.- Main goal of the project

The main goal of this project is to show people that it is relatively simple to install smart devices at their homes with an embedded system and a few lines of code and it can improve their living standards. There are tons of projects around the internet regarding Internet of Things (IoT) with all the steps explained, so you only need to invest some time trying to figure out what is needed and start working on it.

The IoT is expected to transform how we live, work and play. From factory automation and automotive connectivity to wearable body sensors and home appliances, the IoT is set to touch every facet of our lives. We will "author" our life with networks around us that constantly change and evolve based on our surroundings and inputs from other systems. It will make our lives safer with cars that sense each other to avoid accidents. It will make our lives greener with lighting systems that adjust based on the amount of daylight from windows. It will make our lives healthier with wearables that can detect heart attacks and strokes before they happen. There is a long road ahead to the IoT of 2030. But one thing is for sure, it is going to be amazing. [1]

I chose this project thinking about people who have secondary houses on the countryside or another city where they live and they want to make sure that their house is protected and safe by seeing in their smartphones the actual state of the alarm and if there has been any movement, recently, near the main door of the house.

1.2.- Personal motivation

First of all, I wanted to be involved in a real project. I did not want to make some research or some kind of simulation work. I wanted to see in real life that my project has been carried out and that it was going to be useful to someone in the future. Therefore, I would like this project to be open source. Also, one of the benefits of being open source is that anyone can make some improvements to the project from time to time. So, I will be able to know what things could have done better to keep learning about this beautiful specialization like smart devices and IoT.

Furthermore, seeing all those developers and engineers contributing voluntarily with non-profit purposes made me want to play a part in that.

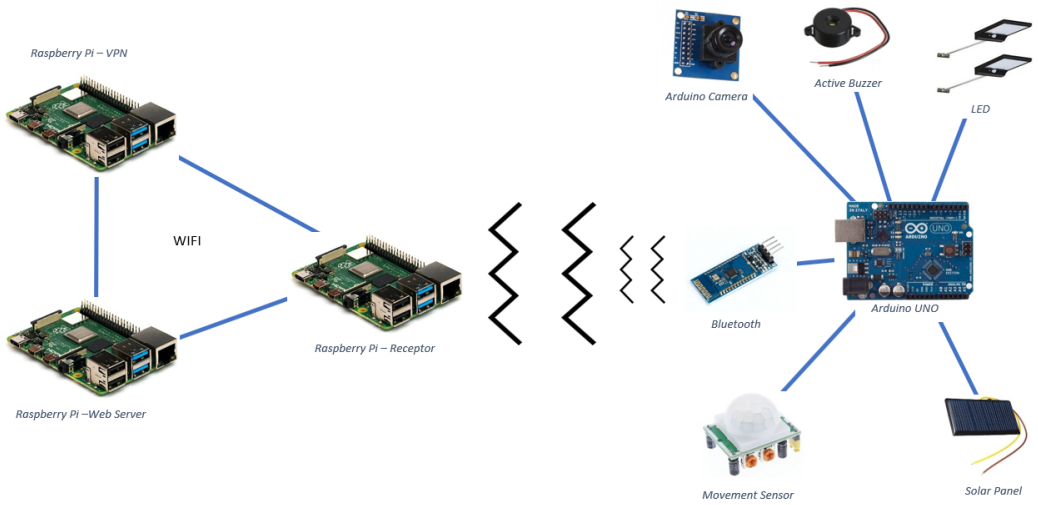
As I was working on it, my curiosity and my interest on IoT has increased that much until the point that I want to continue learning about IoT technologies so, I am quite determined to study a Master´s Degree based on IoT.

1.3.- Evolution since the first thought to the final approach

Project management life cycle
Source: Kissflow

In this chapter I am going to focus on the planning and execution stages. This is where you design and test the project and, depending on the results, you decide to implement some things and discard others. In this project, it went like this:

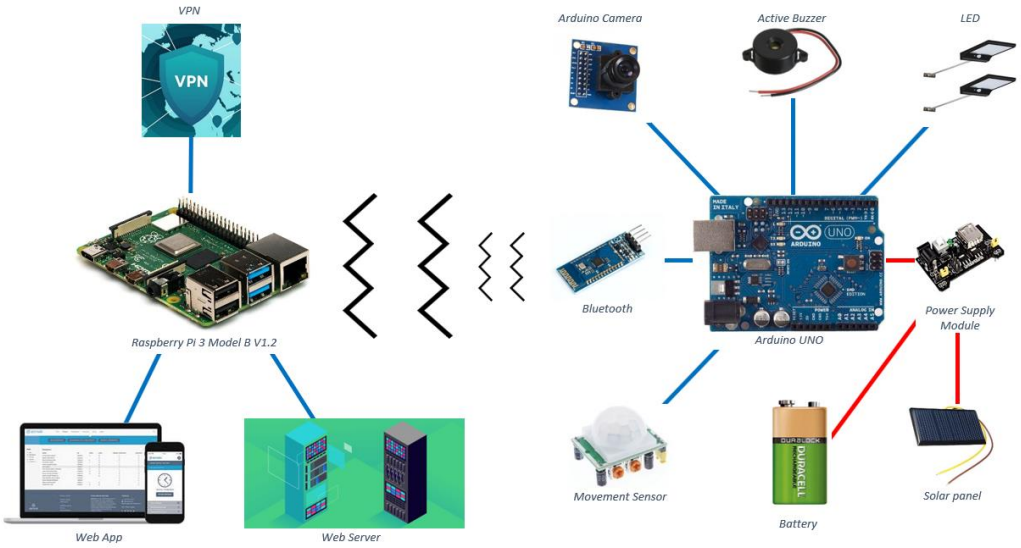
1.0 Version:



1.0 Version

This was the very first approach. I was not very familiar with the Raspberry Pi technology so; this is why I thought 3 applications running at the same time on the Raspberry was not going to work. Then, I decided to divide the tasks and make them communicate wireless via Wi-Fi (IEEE 802.11). Another wireless communication I wanted to implement was the one between the Arduino and the Raspberry Pi, due to several complications on the configuration I had to discard it too.

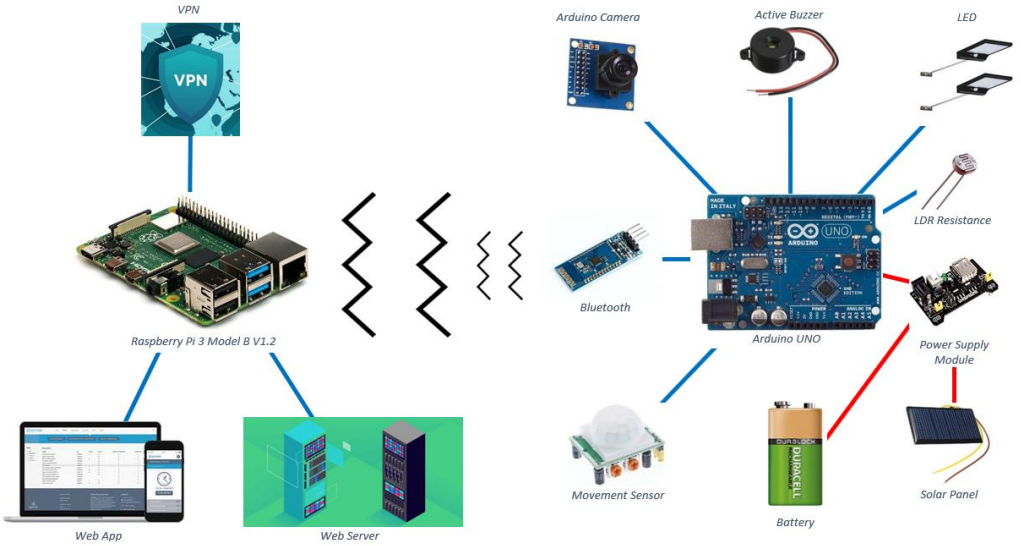
2.0 Version:



2.0 Version

Once I made a little research and talked with my tutor, I realized that I could implement all the applications I wanted to run in one Raspberry Pi. Also, I thought about a back-up power to the Arduino in case the power of the solar panel is not enough.

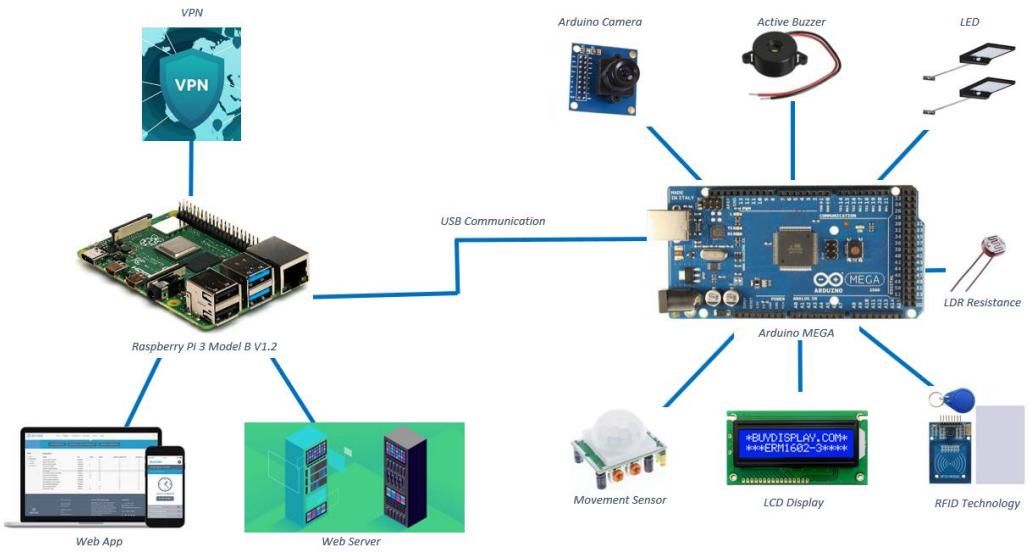
2.1 Version:



2.1 Version

In this version, I thought about including an LDR Resistance to know when there is no light so the LED of illumination must turn on in order to save energy during the day.

3.0 Version:



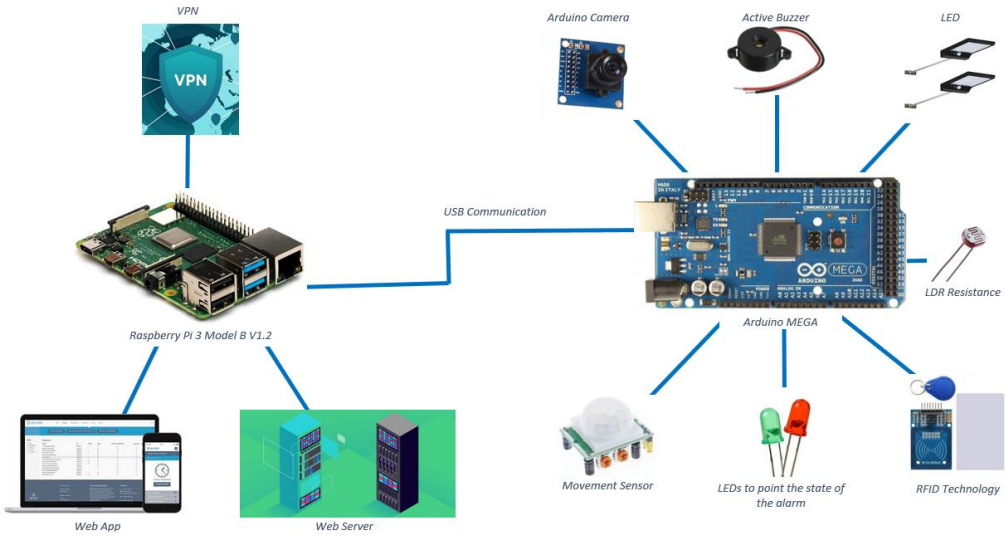
3.0 Version

The camera and the RFID reader use a lot of pins. Arduino has available a bigger board than the Arduino UNO, the Arduino MEGA, it has 54 digital ports (40 more digital ports than Arduino UNO), 16 analog ports (10 more ports than Arduino UNO). I thought about including the LCD Display because I thought I had enough number of ports, and it would display the state of the alarm.

The RFID reader would be used to turn on and off the alarm. It is a good idea to have multiple ways to change the state of the alarm, apart from the web app, just in case you run out of battery in your phone.

One of the big parts of the project was the solar panel. I am a big fan of renewable energies and I think they will play a huge role in the near future, but now the communication between both systems is using USB connection so the Arduino is going to be powered by it, then, the solar panel is useless.

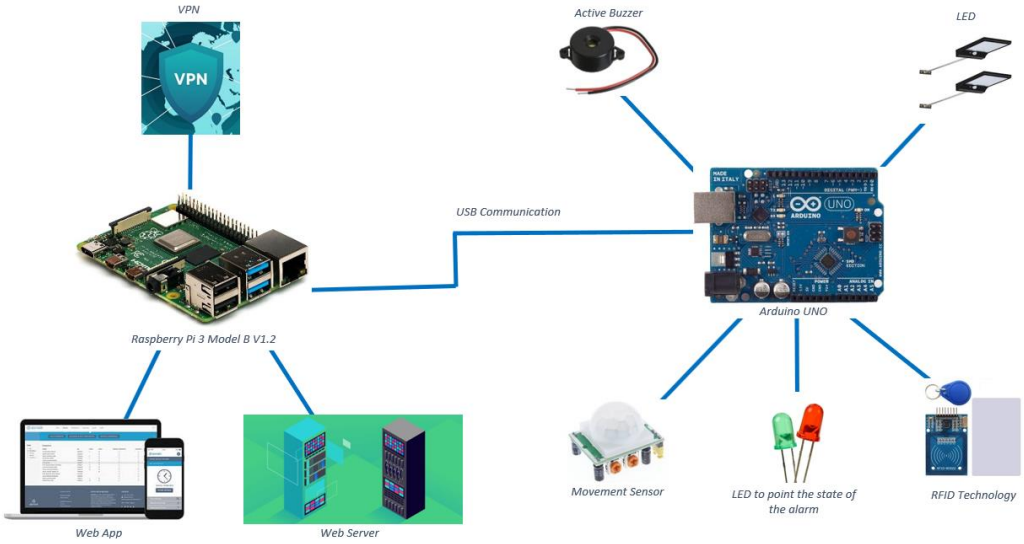
3.1 Version



3.1 Version

Simplifying the project, I replaced the LCD Display with two LEDs to show the state of the alarm.

4.0 Version:



4.0 Version

One of the features that I wanted to implement at all cost was the Arduino Camera but due to the lack of pins available at the Arduino UNO I had to discard it. I thought about buying a brand-new Arduino MEGA board, but I finally decided to use the Arduino UNO because it was the one I had available.

2.- Evolution of IoT devices

The main concept of a network of smart devices was discussed as **early as 1982**, with a modified Coca-Cola vending machine at Carnegie Mellon University becoming the first Internet-connected appliance, able to report its inventory and whether newly loaded drinks were cold or not. [2] [3]

In **1994**, Reza Raji described the concept in IEEE Spectrum as "[moving] small packets of data to a large set of nodes, so as to integrate and automate everything from home appliances to entire factories". [4]

Between 1993 and 1997, several companies proposed solutions like Microsoft's at Work or Novell's NEST. The field gained momentum when Bill Joy envisioned device-to-device communication as a part of his "Six Webs" framework, presented at the World Economic Forum at Davos in **1999**. [5]

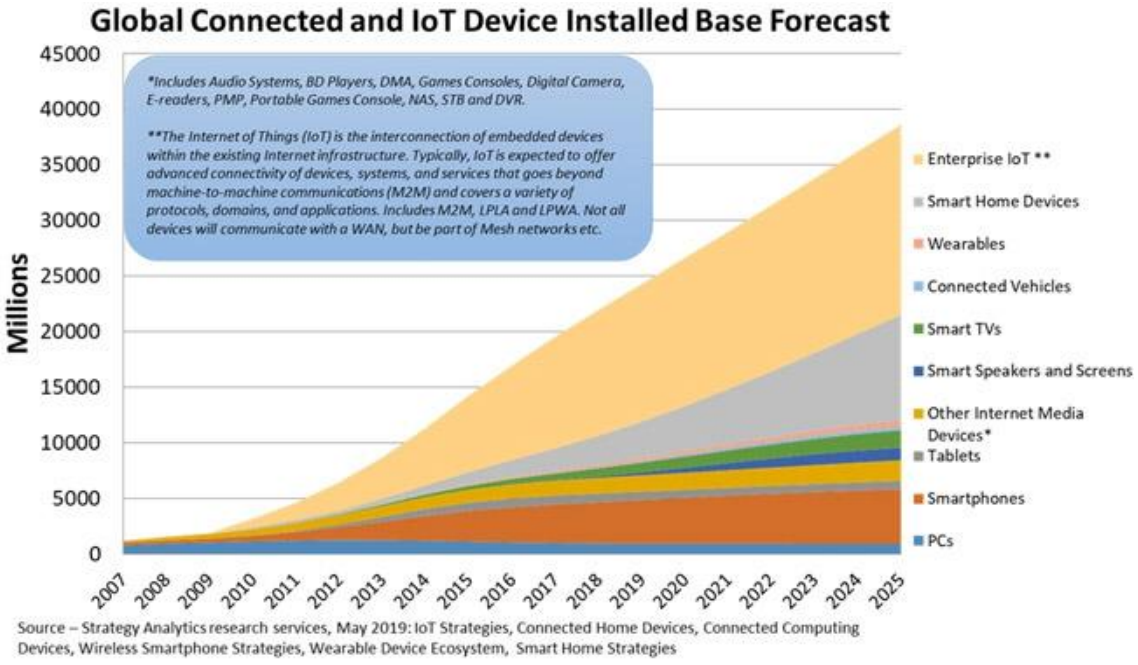
The term "*Internet of Things*" was first used in **1999** by Kevin Ashton, co-founder of the Auto-ID (for Automatic Identification) Center at MIT. His definition of IoT was based on reinventing RFID as a networking technology by linking objects to the internet using the RFID tag.

In 2003, the "Project JXTA-C: Enabling a Web of Things" is published by Bernard Traversat and team at the 36th Annual Hawaii International Conference. According to them, the Project JXTA's aim is to specify a standard set of protocols for ad hoc, pervasive, peer-to-peer computing as a foundation of the upcoming Web of Things. [6]

In 2005, the faculty at the Interaction Design Institute Ivrea (IDII) invents a single-board microcontroller to be used in interactive projects being developed their students. Here, we are talking about the birth of Arduino. The project goal was to create simple, low-cost tools for creating digital projects by non-engineers. [7]

In 2008, different industry stakeholders come together to form the IPSO Alliance to promote connected devices. This was a big leap towards having the IoT implemented for large scale business in real production setups.

From 2016 and beyond, Internet of Things has spread its wings across the consumer, commercial, industrial and infrastructure spaces. As per the market experts, there was supposed to be about 50 billion connected devices by 2020. [8] The goal was not reach though, in 2019, the number of active IoT devices was 26'66 billion. But this number is getting bigger and bigger because every second 127 new devices are connected to the web.



Source: HelpNetSecurity

Although, the definition of IoT has changed from what Kevin Ashton had envisioned it to be with numerous technology inventions, the founding principle of having a network of interconnected devices that are interacting with each other and the surroundings to collect and analyze information using internet has remained the same.

Over the time the RFID based IoT model failed to gain enough attention due to limited connectivity option, high cost of devices and infrastructure. But IoT continued to evolve due to advancements in IP based network connectivity and various other technical innovations that made M2M connection possible over broader range. [6]

Source: Konstant Infosolutions

Nowadays, many companies from different sectors are adopting this technology to simplify, improve, automate, and control different processes. Below, you can see the 5 most important applications of the IoT:

- Wearables: Virtual glasses, fitness bands to monitor for example calorie expenditure and heart beats, or GPS tracking belts.

- Health: The integration of IoT technology into hospital beds, giving way to smart beds, equipped with special sensors to observe vital signs, blood pressure, oximeter and body temperature, among others.

- Traffic monitoring: Applications like Google Maps use our mobile phones as sensors, which collect data from our vehicles, showing the conditions of the different routes, and feeding and improving the information on the different routes to the same destination, distance, estimated time of arrival.

- Fleet management: The installation of sensors in fleet vehicles helps to establish an effective interconnectivity between the vehicles and their managers. It assists with geolocation, performance analysis, telemetry control and fuel savings.

- Agriculture: The quality of soil is crucial to produce good crops, and the Internet of Things offers farmers the possibility to access detailed knowledge and valuable information of their soil condition. Information such as soil moisture, level of acidity, the presence of certain nutrients, temperature and many other chemical characteristics, helps farmers control irrigation, make water use more efficient, specify the best times to start sowing, and even discover the presence of diseases in plants and soil. [9]

3.- Hardware of the system

3.1.- Arduino UNO board

Source: Amazon

The main board of this project is the Arduino UNO. The one I used is not from the original brand *Arduino* but due to the fact that it is open source, there a lot of manufacturers that make their own. The one I have used is the **Elegoo UNO R3 Controller Board**.

Arduino UNO is a microcontroller board based on the ATmega328P (datasheet). It has 14 digital input/output pins (of which 6 can be used as Pulse-width modulation (PWM) outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again. [10]

Characteristics:

Microcontroller:	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P)
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

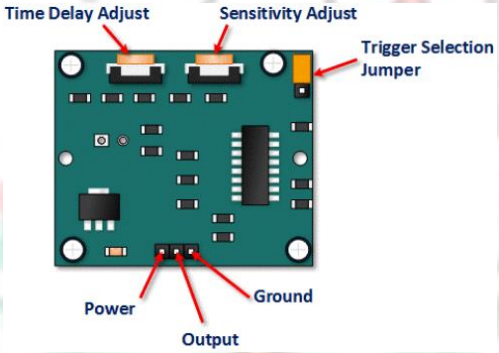
Source: Arduino store

3.1.1.- HC-SR501 PIR sensor

Source: Farnell

The PIR Motion Sensor Detector Module HC SR501 allows you to sense motion. It is almost always used to detect the motion of a human body within the sensor's range. It is often referred to using "PIR", "Pyroelectric", "Passive Infrared" and "IR Motion" sensor. [11]

The PIR sensor itself has two slots. Each slot is made of a special material that is sensitive to IR. When the sensor is idle, both slots detect the same amount of IR, the ambient amount radiated from the room or walls or outdoors. When a warm body like a human or an animal passes by, it first intercepts one half of the PIR sensor, which causes a positive differential change between the two halves. When the warm body leaves the sensing area, the reverse happens, whereby the sensor generates a negative differential change. These change pulses are what is detected. The sensor requires almost a minute to initialize. During this period, sometimes it will output false detection signals.



Source: ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 122)

Pin	Function
Time Delay Adjust	Sets how long the output remains high after detecting motion.... Anywhere from 3 seconds to 5 minutes.
Sensitivity Adjust	Sets the detection range.... from 3 meters to 7 meters
Trigger Selection Jumper	Set for single or repeatable triggers.
Ground pin	Ground input
Output Pin	Low when no motion is detected. High when motion is detected. High is 3.3V
Power Pin	5 to 20 VDC Supply input

Source: ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 122)

Sensitivity Adjustment

The range of detection is approximately from 3 to 7 meters and the view area is 110 degrees. You can adjust the distance you want at the sensitivity adjust potentiometer. Setting the potentiometer fully clockwise decreases the sensitivity and the range will be around 3 meters. Then, setting the potentiometer fully counterclockwise, the range will be around 7 meters. In our project, we are setting the range to the minimum (3 meters). I have to say that after the testing, I realized that I could use the detector in a short space too, thanks to that, I will be able to make a demonstration using a 3D printed house.

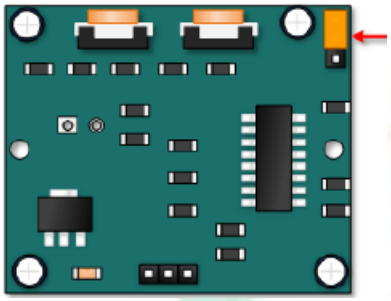
Time Delay Adjustment

The time delay adjustment determines how long the output of the PIR sensor module will remain high after a detection. The range is from about 3 seconds to five minutes. Setting the potentiometer fully clockwise increases the delay and the delay will be around 5 minutes. Then, setting the potentiometer fully counterclockwise, the range will be around 3 seconds. In our case, we are setting the delay to the minimum (3 seconds) to make the tests faster. But it is a matter of testing what are the perfect values for your project. Notice that after the time delay, the detection is blocked for 3 seconds, as a default value. [14]

Trigger Selection Jumper

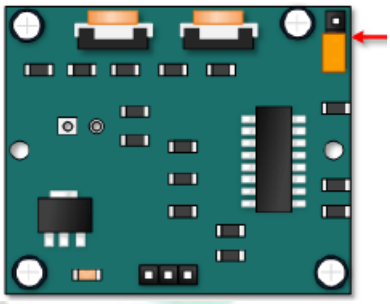
The trigger selection jumper allows you to select between single and repeatable triggers. The effect of this jumper setting is to determine when the time delay begins.

- SINGLE TRIGGER: The time delay begins immediately after the first movement detected.
- REPEATABLE TRIGGER: Each detected motion resets the time delay. Thus, the time delay begins after the last movement detected.



Single Trigger Mode

Source: ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 126)

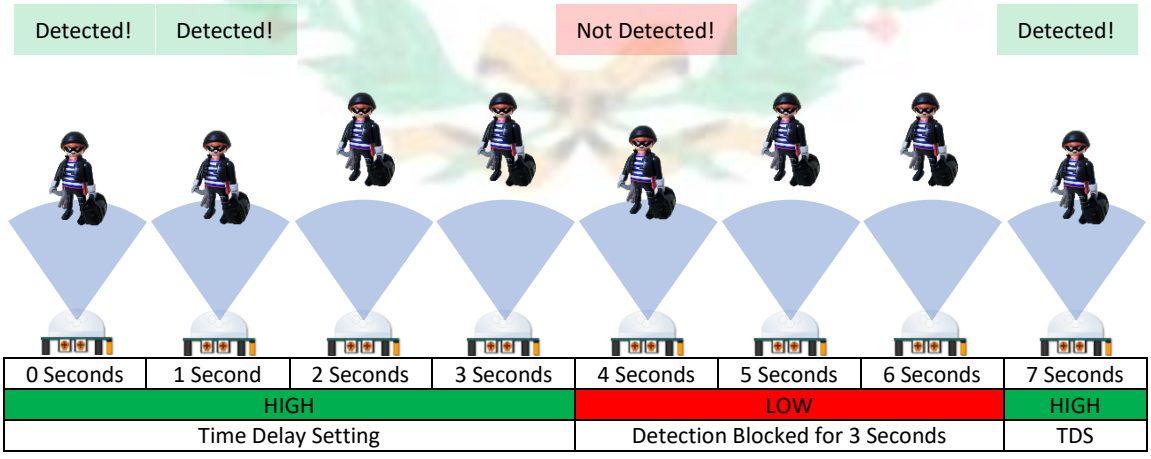


Repeatable Trigger Mode

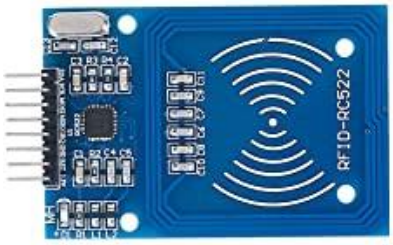
Source: ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 126)

In this project, I have chosen the repeatable trigger mode because I want the buzzer ON while the PIR sensor is detecting movement. I mean when the alarm is ON, every time the PIR sensor detects movement, the buzzer rings in order to warn the people around about the trespassing.

Here, I am going to show an example with the time delay set to 3 seconds and the trigger repeatable:



3.1.2.- RC522 RFID module



Source: Amazon

The RFID (Radio Frequency Identification) is a subset of technologies designed for reading tags at a certain distance wirelessly. RFID consists of two main components, a transponder/tag and a transceiver also known as the reader.

The reader consists of a Radio Frequency module and an antenna which generates high frequency electromagnetic field. On the other hand, the tag is usually a passive device, meaning it does not contain a battery. Instead, it contains a microchip that stores and processes information, and an antenna to receive and transmit a signal.

To read the information encoded on a tag, it is placed near to the reader. A reader generates an electromagnetic field which causes electrons to move through the tag's antenna and subsequently power the chip.

The powered chip inside the tag then responds by sending its stored information back to the reader in the form of another radio signal. This is called backscatter. The backscatter, or change in the electromagnetic/RF wave, is detected and interpreted by the reader which then sends the data out to a computer or microcontroller.

RFID technology is widely used, for example, in alarm systems, like this project, commercial applications using barcodes, electronic locks, payment systems, access controls, among others. [12]

This RFID reader includes a SPI, I2C and UART communication, therefore it is so simple to connect it to Arduino. The MFRC522 works with tags like Mifare S50, Mifare S70, Mifare UltraLight, Mifare Pro and Mifare Desfire. [13]

Source: RS Components

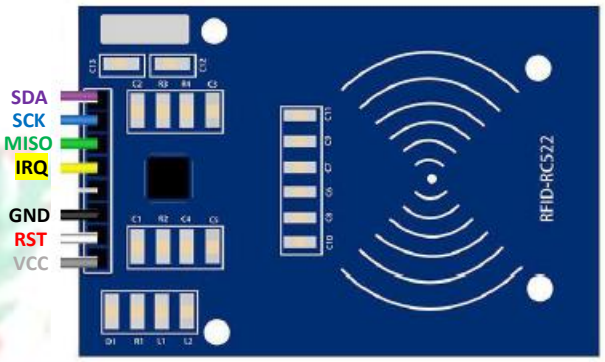
Source: RS Components

Characteristics:

Frequency Range	13’56 MHz ISM Band
Host interface	SPI / I2C / UART
Operating Supply Voltage	2’5 V to 3’3 V
Max. Operating Current	13-26 mA
Min. Current (Power Down)	10 mA
Logic Inputs	5 V Tolerant
Read Range	5cm

Source: Last Minute Engineers

Pinout:



Source: ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 149)

Pin	Function
SDA	This pin acts as Signal input when SPI interface is enabled, acts as serial data when I2C interface is enabled and acts as serial data input when UART interface is enabled.
SCK	Accepts clock pulses provided by the SPI bus Master i.e., Arduino.
MISO	This pin acts as Master-In-Slave-Out when SPI interface is enabled, acts as serial clock when I2C interface is enabled and acts as serial data output when UART interface is enabled.
IRQ	It is an interrupt pin that can alert the microcontroller when RFID tag comes into its vicinity.
GND	It is the Ground Pin.
RST	It is an input for Reset and power-down. When this pin goes low, hard power-down is enabled. This turns off all internal current sinks including the oscillator and the input pins are disconnected from the outside world. On the rising edge, the module is reset.
VCC	It supplies power for the module. You can connect it to 3.3V output from your Arduino. Remember, connecting it to 5V pin will likely destroy your module.

Source: Last Minute Engineers

3.1.3.- Active buzzer

Source: Roboelements

Electronic buzzers are DC-powered and equipped with an integrated circuit. They are widely used in computers, printers, photocopiers, alarms, electronic toys, automotive electronic devices, telephones, timers and other electronic products for voice devices. Buzzers can be categorized as active and passive ones. The difference between the two is that an active buzzer has a built-in oscillating source, so it will generate a sound when electrified. A passive buzzer does not have such a source so it will not tweet if DC signals are used; instead, you need to use square waves whose frequency is between 2K and 5K to drive it. The active buzzer is often more expensive than the passive one because of multiple built-in oscillating circuits. [14]

3.1.4.- LEDs

Source: Microlog

A light-emitting diode (LED) is a semiconductor light source that emits light when current flows through it. Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons. The color of the light (corresponding to the energy of the photons) is determined by the energy required for electrons to cross the band gap of the semiconductor. [15]

LEDs have many advantages over incandescent light sources, including lower energy consumption, longer lifetime, improved physical robustness, smaller size, and faster switching. LEDs are used in applications as diverse as aviation lighting, fairy lights, automotive headlamps, advertising, general lighting, traffic signals, camera flashes, lighted wallpaper, horticultural grow lights, and medical devices. [16]

Unlike a laser, the light emitted from an LED is neither spectrally coherent nor even highly monochromatic. However, its spectrum is sufficiently narrow that it appears to the human eye as a pure (saturated) color. [17] [18]

You cannot directly connect an LED to a battery or voltage source because 1) the LED has a positive and a negative lead and will not light if placed the wrong way and 2) an LED must be used with a resistor to limit or 'choke' the amount of current flowing through it; otherwise, it will burn out.

If you do not use a resistor with an LED, then it may well be destroyed almost immediately, as too much current will flow through, heating it and destroying the 'junction' where the light is produced. The way to tell which is the positive lead of the LED and which the negative is looking at the longest lead which is the positive one. [19]

Source: ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 44)

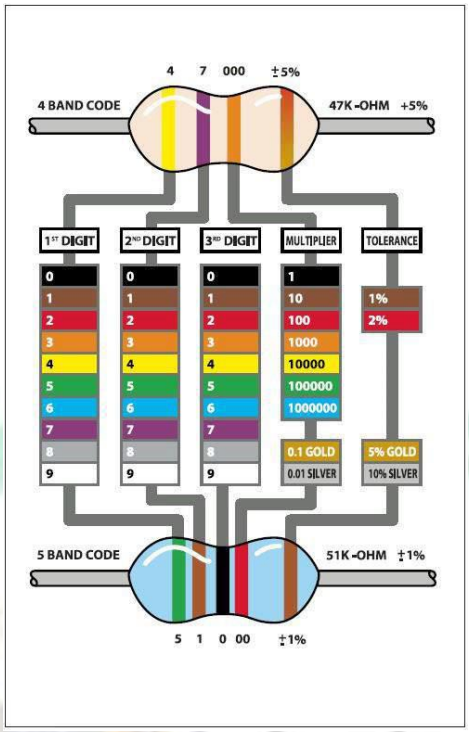
In this project, it is going to be used one green, one red and one yellow LEDs. The red and green LEDs are intended to show the status of the alarm, whether it is enabled or disabled. The yellow LED is used as an illumination light.

3.1.5.- Resistor

Source: Addicore

As the name suggests, resistors resist the flow of electricity. The higher the value of the resistor, the more it resists, and the less electrical current will flow through it. We are going to use this to control how much electricity flows through the LED and therefore, how brightly it shines.

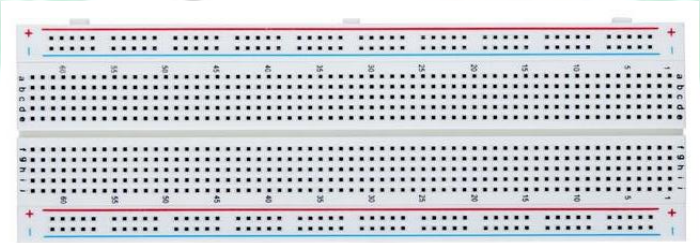
The unit of resistance is called the Ohm, which is usually shortened to Ω, the Greek letter Omega. These resistors all look the same, except that they have different colored stripes on them. These stripes tell you the value of the resistor. [20]



Source: ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 46)

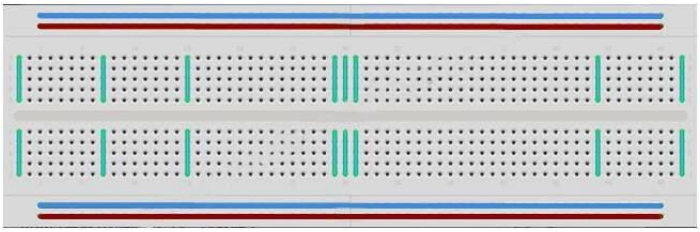
In this project, I am using 3 resistors of 220k Ohms. Each one connected to the LEDs.

3.1.6.- Breadboard



Source: ELEGOO

A breadboard, or protoboard, is a construction base for prototyping electronics. Because the solderless breadboard does not require soldering, it is reusable. This makes it easy to use for creating temporary prototypes and experimenting with circuit design. For this reason, solderless breadboards are also popular with students and in technological education. [21]



Source: The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 43)

While breadboards are great for prototyping, they have some limitations. Because the connections are push-fit and temporary, they are not as reliable as soldered connections. If you are having intermittent problems with a circuit, it could be due to a poor connection on a breadboard. [22]

3.2.- Raspberry Pi 3 Model B V1.2

Source: Amazon

Raspberry Pi is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation in association with Broadcom. [23] Later, the original model became far more popular than anticipated, [24] selling outside its target market for uses such as robotics. It is now widely used in many areas, such as for weather monitoring, [25] because of its low cost, modularity, and open design.

After the release of the second board type, the Raspberry Pi Foundation set up a new entity, named Raspberry Pi Trading, and installed Eben Upton as CEO, with the responsibility of developing technology. [26] The Foundation was rededicated as an educational charity for promoting the teaching of basic computer science in schools and developing countries.

The Raspberry Pi is one of the best-selling British computers. [27] As of December 2019, more than thirty million boards have been sold. [28] Most Pis are made in a Sony factory in Pencoed, Wales, [29] while others are made in China and Japan. [30]

It is one of the most versatile application boards, it is commonly used for some programming purposes especially the ones related to IoT or AI. [31]

Characteristics:

Processor	Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
RAM Memory	1GB LPDDR2 SDRAM
Wi-Fi	2'4 GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN
Bluetooth	4.2
Power Input	5V / 2'5V DC
Ports	40-pin GPIO header
	Full-size HDMI
	4 USB 2.0 ports
	4 pole stereo output and composite video port
	MicroSD port for loading your operating system and storing data

Source: Raspberry Pi Shop

The Raspberry Pi lets students, experts, and hobbyists build innovative computing projects at a very affordable cost. Since its inception 6 years ago, it has enjoyed widespread popularity, thanks to the infinite range of possibilities this system offers. The single-board computer is now in its third major version and is being widely used for numerous tech projects worldwide. The reason behind Raspberry Pi's immense success is that you can use it in unimaginable ways. From computing platforms to full-fledged arcade machines, Raspberry Pi can sit behind every electronic project that requires a computing brain. Here, I am going to stand out two of the most famous Raspberry Pi projects:

- Weather Station: This is one of the best projects for beginners. By building your weather station that collects and analyzes atmospheric data, you will learn the fundamentals of Raspberry Pi and increase your experience for later Raspberry Pi projects. You can choose from a vast number of physical sensors and complimentary Python libraries to control their functioning.

- Wireless Print Server: With an old unused printer you can make a wireless print server out of it. All you will need is a Raspberry Pi board, a storage space, and some will. It is one of the best Raspberry Pi projects for students trying to build some quick raspberry projects. The print server can be accessed from any device you interface for and will provide a useful means to do that quick office work. [32]

4.- Software of the system

4.1.- Arduino UNO board

Within the project, it is easy to differentiate two main parts, the body that collects all the signals from the environment (Arduino), and the body that controls the receptor (Raspberry Pi). So, the Arduino's code will drive the alarm system whether is enabled or disabled and it will receive if the user wants to turn on or turn off the alarm.

The code of every Arduino is structured in two blocks or functions, the *void setup* and *void loop*. The *void setup* executed only one time and the *void loop*, as the proper name says, executes like a loop. The *void setup* in my code is used to initialize some variables.

```
28 void setup()
29 {
30   pinMode(pirPin, INPUT); //Movement sensor INPUT
31
32   pinMode(buzzer, OUTPUT);//initialize the buzzer pin as an output
33
34   pinMode(lGreen, OUTPUT);
35   pinMode(lRed, OUTPUT);
36   digitalWrite(lRed, HIGH);
37   pinMode(lBlue, OUTPUT);
38
39   //Initialize the serial port
40   Serial.begin(9600); //Initialise the communication with the Rasp
41   while (!Serial);    // Wait untilSerial is ready - Leonardo
42
43   //Initialize the RFC reader
44   SPI.begin();         // Init SPI bus
45   mfrc522.PCD_Init();  // Init MFRC522 card
46
47   // Prepare key - all keys are set to FFFFFFFFFFFFh at chip delivery from the factory.
48   for (byte i = 0; i < 6; i++)
49   {
50     key.keyByte[i] = 0xFF;
51   }
52
53   //Key Chain -> Joserra
54   joserraUID[0] = 0x19;
55   joserraUID[1] = 0xEF;
56   joserraUID[2] = 0x2A;
57   joserraUID[3] = 0xB9;
58
59   //RFC card -> Guests
60   guest[0] = 0x69;
61   guest[1] = 0xD1;
62   guest[2] = 0xF6;
63   guest[3] = 0x97;
64 }
```

The user will be able to turn on or turn off the alarm by two ways, using the web app or using the RFID card. At the beginning of every loop, Arduino is going to check if the user wants the alarm to be changed.

```
125 void loop()
126 {
127   rfidRead();
128   raspCommands();
```

The rfidRead() function look like this:

```
66 void rfidRead()
67 {
68   int matches = 0;
69
70   // Look for new cards, and select one if present
71   if (!mfrc522.PICC_IsNewCardPresent() || !mfrc522.PICC_ReadCardSerial())
72   {
73     delay(50);
74     return;
75   }
76
77   // Now a card is selected. The UID and SAK is in mfrc522.uid.
78
79   // Now, print the UID of the Card
80   //Serial.print(F("Card UID:"));
81   for (byte i = 0; i < mfrc522.uid.size; i++)
82   {
83     //Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
84     //Serial.print(mfrc522.uid.uidByte[i], HEX);
85
86     if(mfrc522.uid.uidByte[i] == joserraUID[i] || mfrc522.uid.uidByte[i] == guest[i])
87     {
88       matches++;
89     }
90   }
91   //Serial.println();
92
93   if(matches == 4)
94   {
95     //Serial.println("Bienvenido");
96     //Serial.println();
97
98     stateAlarm = !stateAlarm;
99     ndetections = 0;
100    //Serial.println(stateAlarm);
101  }
102
103  delay(1500);
104 }
```

The raspCommands() function checks if there has been sent any command to turn on or turn off the alarm from the Raspberry Pi:

```
106 void raspCommands()
107 {
108   if (Serial.available())
109   {
110     char c = Serial.read(); //Store the command in a variable
111     if (c == 'H')
112     {
113       //Turn ON the alarm
114       stateAlarm = true;
115       ndetections = 0;
116     } else if (c == 'L')
117     {
118       //Turn OFF the alarm
119       stateAlarm = false;
120       ndetections = 0;
121     }
122   }
123 }
```

At the beginning of every loop, Arduino sends to the Raspberry Pi two values. The first one is the state of the alarm, and it will be a '0' which means that the alarm is off or a '1' which means that the alarm is on. The second value is the number of detections. This value stores how many times the PIR sensor has detected some movement if the alarm is on, if the alarm is off, it does not count the times the PIR sensor detected someone.

```
130   Serial.println(stateAlarm);
131   Serial.println(ndetections);
```

Finally, the main function of the alarm system. The only difference on the functionality if the alarm is on or off is that the buzzer will ring only if the alarm is on.

```
125 void loop()
126 {
127   rfidRead();
128   raspCommands();
129
130   Serial.println(stateAlarm);
131   Serial.println(ndetections);
132
133   //Checks if the alarm is on or off
134   if(stateAlarm)
135   {
136     digitalWrite(lGreen, HIGH);
137     digitalWrite(lRed, LOW);
138
139     pirValue = digitalRead(pirPin);
140 //    Serial.println("pirValue: ");
141 //    Serial.print(pirValue);
142 //    Serial.println();
143
144     //if pirValue = 1 --> Movement
145     //if pirValue = 0 --> No movement
146
147     while(pirValue == 1)
148     {
149       if(oneTime == 0)
150       {
151         ndetections++;
152         oneTime++;
153       }
154         digitalWrite(lBlue, HIGH);
155         digitalWrite(buzzer, HIGH);
156         delay(2);//wait for 2ms
157         digitalWrite(buzzer, LOW);
158         delay(2);//wait for 2ms
159         pirValue = digitalRead(pirPin);
160     }
161     oneTime = 0;
162     digitalWrite(lBlue, LOW);
163   }
```

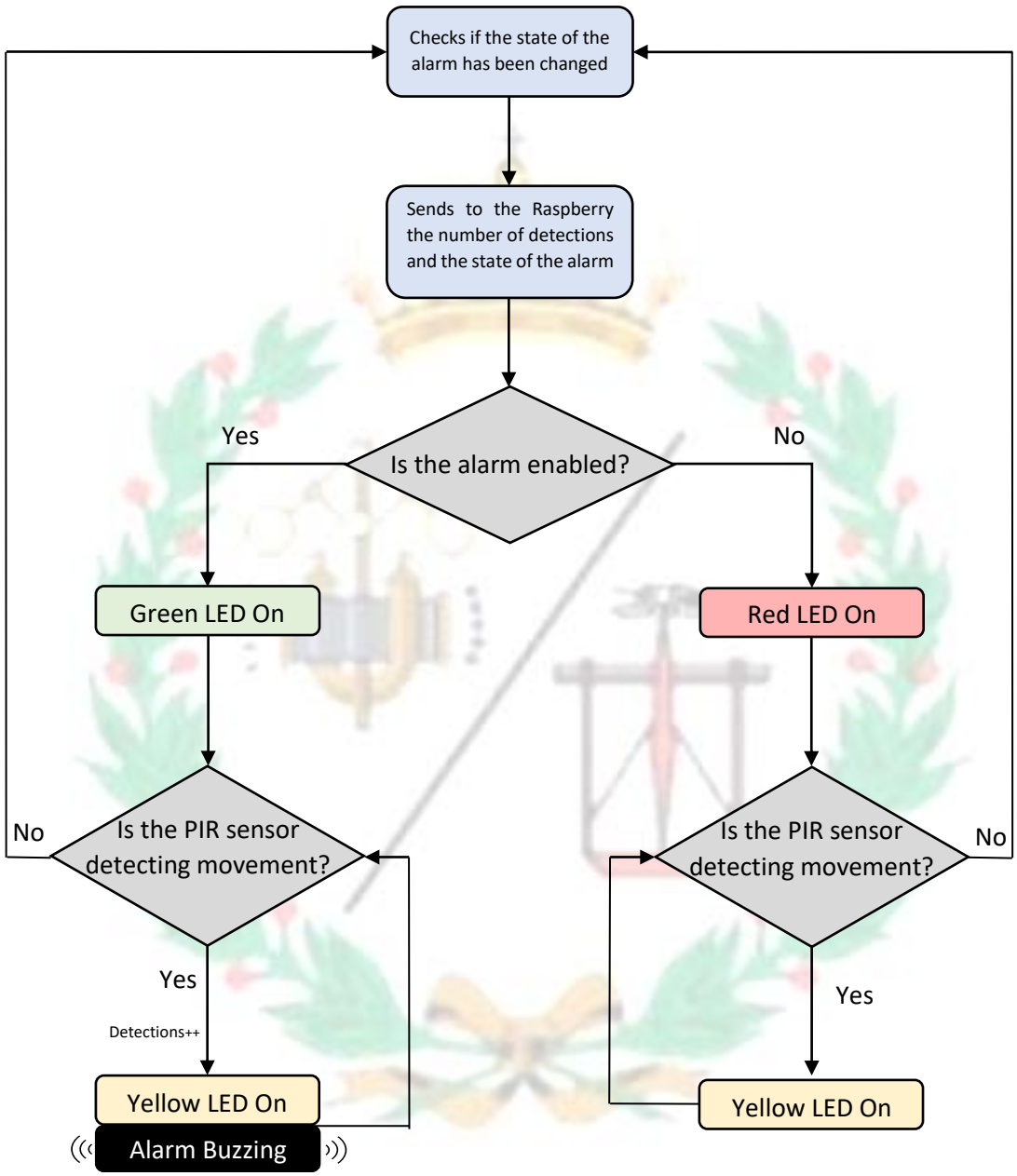
The functionality when the alarm is on

```
164   else
165   {
166     digitalWrite(lGreen, LOW);
167     digitalWrite(lRed, HIGH);
168
169     pirValue = digitalRead(pirPin);
170 //    Serial.println("pirValue: ");
171 //    Serial.print(pirValue);
172 //    Serial.println();
173
174     //if pirValue = 1 --> Movement
175     //if pirValue = 0 --> No movement
176
177     while(pirValue == 1)
178     {
179       digitalWrite(lBlue, HIGH);
180       pirValue = digitalRead(pirPin);
181 //      Serial.println("pirValue: ");
182 //      Serial.print(pirValue);
183 //      Serial.println();
184     }
185     digitalWrite(lBlue, LOW);
186   }
187 }
```

The functionality when the alarm is off

The code is very similar, it may seem kind of redundant, but due to the fact that I want it to make it open source, I think it can help the beginners to understand the code better.

The idea of this piece of code is to show, depending on the state of the alarm, the red LED if the alarm is disabled or the green LED if the alarm is enabled. Also, if the alarm is enabled and the PIR sensor is activated, the buzzer will ring during the time that something in front of the PIR sensor is moving. A summary of the Arduino's code can be seen on the next flowchart.



Flowchart of the Arduino's code

4.2.- Raspberry Pi 3 Model B V1.2

4.2.1.- Web server

The web server application used in this project is Apache. Apache is a free and open source cross-platform web server software, released under the terms of Apache License 2.0. Apache is developed and maintained by an open community of developers under the auspices of the Apache Software Foundation. The vast majority of Apache HTTP Server instances run on a Linux distribution, [33] and a wide variety of Unix-like systems.

Source: The Apache HTTP Server Project

Originally based on the NCSA HTTPd server, development of Apache began in early 1995 after work on the NCSA code stalled. Apache played a key role in the initial growth of the World Wide Web, [34] quickly overtaking NCSA HTTPd as the dominant HTTP server. Until 2005, it was the most popular method to host web pages, in 2005 Apache was hosting the 70% of the web pages worldwide. Since then, its popularity decreased with the rise of other technologies like IIS, NGIX, or NodeJS. Although, Apache is still one of the main alternatives to build a web server. In 2009, it became the first web server software to serve more than 100 million websites. [35] As of January 2021, Netcraft estimated that Apache served 24.63% of the million busiest websites.

I have used this tool to host the web application which will be used to control the system.

4.2.2.- Virtual Private Network (VPN)

A virtual private network, or VPN, is an encrypted connection over the Internet from a device to a network. The encrypted connection helps ensure that sensitive data is safely transmitted. It prevents unauthorized people from eavesdropping on the traffic and allows the user to conduct work remotely. VPN technology is widely used in corporate environments. [36]

Source: El Español

True VPNs can be traced back to 1996 when Microsoft created a point-to-point tunneling protocol, also known as peer-to-peer tunneling protocol or PPTN. This was a method of creating a secure network between users by encrypting data and forming a tunnel over a LAN or WAN connection. PPTN made the transfer of data safer, and it remains a popular VPN. [37] Soon other protocols were developed to provide VPN services. Initially they were used by businesses and public bodies, but as wider internet use took off, inexpensive or free VPNs were developed with the aim of helping the general public use the internet safely and securely. [38] The aim of these B2C VPNs was to protect users against censorship, hackers, data mining, and spam. Most VPN services were founded after 2005. [39]

The use of this feature is to be able to control the system when the user is connected to a different network than the alarm system.

4.2.3.- Web application

To make the control of the system easier, I have developed a web application to access the system to be able to enable or disable the alarm, see how many times a movement was detected and see when the system was turned on and off. This application is based on HTML, CSS, PHP, MySQL and Python.

HTML and CSS helps the design of the UI (User Interface) which means how the app will look like. I have decided to make it simple and smooth to give it a modern look, like a proper IoT application.

PHP and MySQL are used to create the functionality of the app, the background itself. I have decided to implement a database and access it through MySQL because I thought it would be useful to store some temporary values like the state of the alarm or the number of detections, and some lasting values like the exact date when the alarm changed its state.

The tool I have used to create the database is phpMyAdmin. It is a tool I am familiar with, and it is very simple to use. phpMyAdmin is a free and open source administration tool for MySQL and MariaDB. As a portable web application written primarily in PHP, it has become one of the most popular MySQL administration tools, especially for web hosting services. [40] phpMyAdmin became one of the most popular PHP applications and MySQL administration tools, with a large community of users and contributors. [41]

In this tool there are 3 databases. One which stores the state of the alarm, other one that stores the number of detections during the ON state of the alarm and the last one stores the record of movements involving the changes on the state of the alarm.

4.3.- Communication between Arduino and Raspberry Pi

One of the features I wanted to implement, as I have said before, was the Bluetooth communication between both devices. After a deep research, I was not able to manage the Bluetooth communication to work. Then I thought about a simpler way, this is when the serial communication came into my mind.

The communication between both devices is bidirectional, from Raspberry Pi to Arduino and from Arduino to Raspberry Pi.

The communication from the Raspberry Pi to the Arduino is very helpful because this method is used to know the state of the alarm every time the app is loaded. When the app is loading, it executes two python scripts. The first one was developed to wait for the Arduino to send the state of the alarm and read it.

state.py

```
1  import serial
2
3  arduino = serial.Serial('/dev/ttyACM0', 9600)
4
5  while True:
6      state = arduino.readline()
7      #print(state[0])
8      if (state[0] == '1'):
9          print("1")
10         break
11     elif (state[0] == '0'):
12         print("0")
13         break
14 arduino.close() #End of the communication
15
```

Script that reads from the Arduino the state of the alarm

The second one, in case the alarm is ON, will be used to know how many movements were detected by the PIR sensor.

stated.py

```
1  import serial
2
3
4  arduino = serial.Serial('/dev/ttyACM0', 9600)
5
6  i = 0
7
8  while True:
9      state = arduino.readline()
10
11     if(state[0] != None and i == 0):
12         stated = state[0]
13         #print("Estado: ")
14         #print(state[0])
15         i=1
16     elif(state[0] != None and i == 1):
17         times = state
18         #print("Times: ")
19         #print(state[0])
20         print(times)
21         break
22 arduino.close() #End of the communication
23
```

Python script that reads the number of detections

The variable *times* stores the number of movements detected and prints the value, thanks to that, PHP can store that value as the output.

Once the app is loaded, we can see in the app the current state of the alarm. Then, we can modify the state by clicking in the state button. When the button is clicked, it is redirected to the backend page where the python script is executed and changes the state of the alarm.

```python
import serial

arduino = serial.Serial('/dev/ttyACM0', 9600)

comando = 'L' #Input
arduino.write(comando) #Send the command to Arduino

print('LED OFF')

arduino.close() #End the communication

```

Python script that sets the alarm OFF

```python
import serial

arduino = serial.Serial('/dev/ttyACM0', 9600)

comando = 'H' #Input
arduino.write(comando) #Send the command to Arduino

print("LED ON")

arduino.close() #End the communication

```

Python script that sets the alarm ON

The communication from the Arduino to the Raspberry is way easier. Arduino is constantly sending the value of the state of the alarm to the Raspberry at the beginning of every loop.

```
130   Serial.println(stateAlarm);
131   Serial.println(ndetections);
```

And when the Raspberry Pi sends the command to activate or deactivate, the Arduino reads it here:

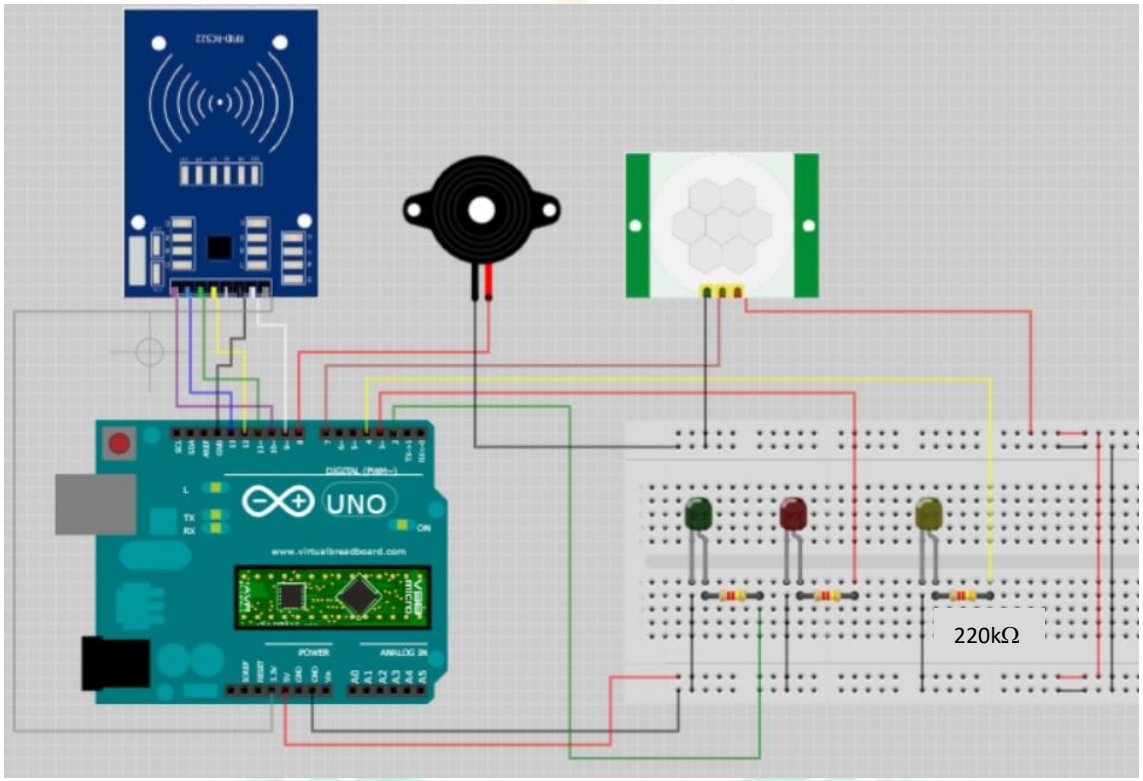
```
106 void raspCommands()
107 {
108   if (Serial.available())
109   {
110     char c = Serial.read(); //Store the command in a variable
111     if (c == 'H')
112     {
113       //Turn ON the alarm
114       stateAlarm = true;
115       ndetections = 0;
116     } else if (c == 'L')
117     {
118       //Turn OFF the alarm
119       stateAlarm = false;
120       ndetections = 0;
121     }
122   }
123 }
```

I want to give special credit to Mario Pérez Esteso for the article *"Arduino + Raspberry Pi - RaspDuino"* where I learned to use this piece of code that helped me in the communication between devices.

5.- Test of the project step by step

In this chapter it is going to be explained the process of building and connecting everything within this project and, in the end, it is going to be showed if the project is working properly or not. Every script I have used in this project is available at my GitHub: https://github.com/Joserra13/TFG

First of all, on the Arduino side, which is the main side and most important, the connection of all the sensors with the Arduino must the like the picture below this paragraph. Then, connect the Arduino to the computer and upload the code through the Arduino IDE.



Once the code is running correctly, lets change to the Raspberry Pi side. The first thing to check is if the web server with PHP and MariaDB (or similar), the VPN, the phpMyAdmin and the pertinent databases are installed, configured, and working smoothly. It there is any doubt how to install any tool, it is explained in the appendix of this project (Page 49).

The next thing to do is include the python scripts that will make the Raspberry Pi interact with the Arduino. This will be located at the path */var/www/html*, the same path as the web app. And connect the Arduino and the Raspberry Pi using an USB cable.

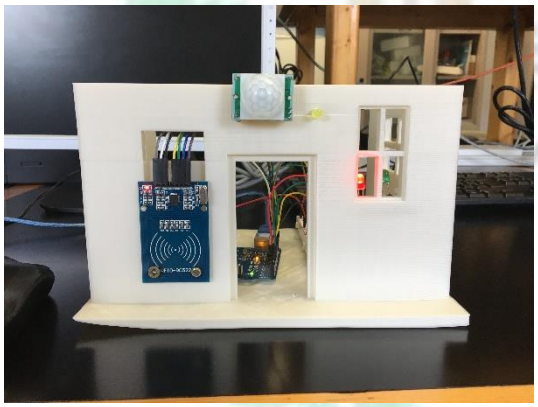
It is a good idea to check if the scripts work according to the expectations. For example, I have a copy of the scripts on the desktop, an easy to access path from the console, and you can execute any python script using:

```
python <name_of_the_script>.py.
```

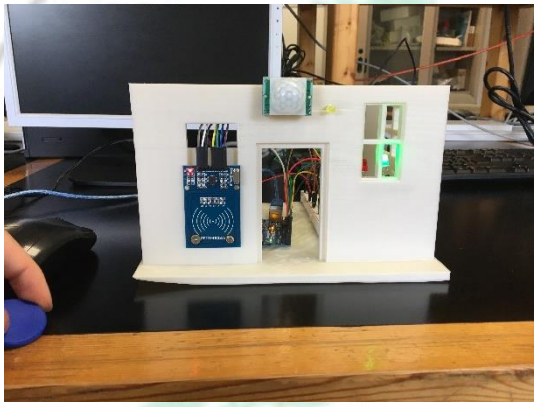
I personally recommend running a few times the script *state1.py* which the same script as state.py but with an infinite loop, so you can change the state of the alarm with the RFID tag and see the change in real time in the console. The script, each loop, shows two values, the first is the state of the alarm and the second the number of detections made by the PIR sensor (only when the alarm is on).

The script tested is state.py and, as explained on the previous chapter, it reads the current state of the alarm, and we can verify it works.

```
pi@raspberrypi:~/Desktop $ python state.py
0
```



```
pi@raspberrypi:~/Desktop $ python state.py
1
```



The next thing to do is giving permissions to the web server to execute the scripts on the port of the Arduino. **It is quite important to know that in this project, as a demonstration, I will give full privileges to the web server on that port but if anyone thinks about deploying this project in their own house be careful because it may imply some security breaches**.

Back to the project, there is needed to type on the console the command:

```
ls -l /dev/tty*
```

This command displays all the ports available on the Raspberry Pi, but the one that requires the attention is **/dev/ttyACM0**, which is the port of the Arduino. So, to give full privileges, type the command:

```
sudo chmod 777 /dev/ttyACM0
```

After typing that command, you will see that the Arduino port is available to anyone.

```
crw-rw---- 1 root dialout 166,  0 May 30 18:01 /dev/ttyACM0
```

Before sudo chmod 777 /dev/ttyACM0

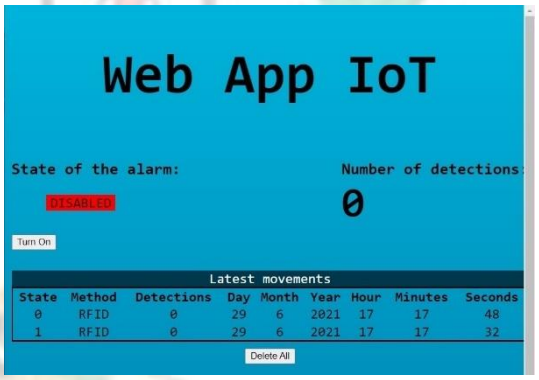
```
crwxrwxrwx 1 root dialout 166,  0 Jun  1 12:19 /dev/ttyACM0
```

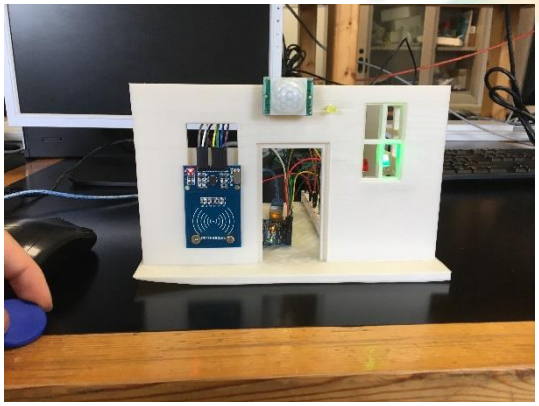
After sudo chmod 777 /dev/ttyACM0

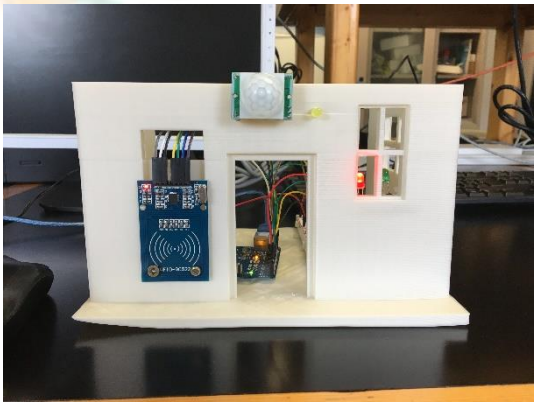
The letter 'c' stands for special character file (tty device). The next 9 values are the privileges of each user in the system. Each 3 characters refer to owner, group, and rest of user privileges. 'r' means *read*, 'w' means *write* and 'x' means *execution*. Now the project is ready and there is only the testing part left that will verify if the project works properly or not.

Now the project is fully functional and ready to be tested. To do so, it is going to be presented few use cases to show that the project works perfectly in different scenarios.

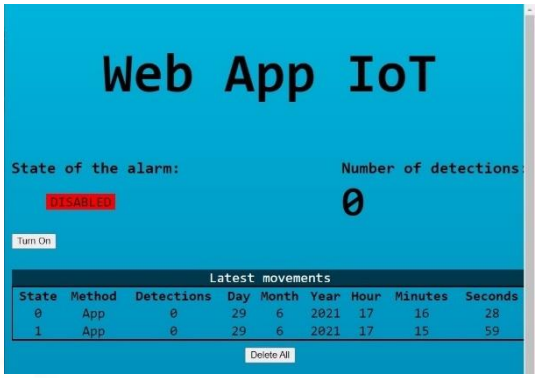
Turning on and off the alarm through the RFID tag:

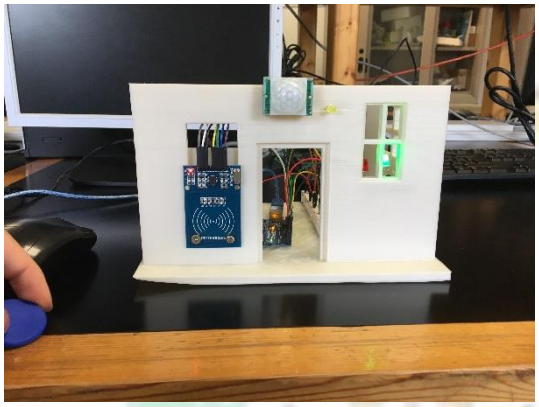


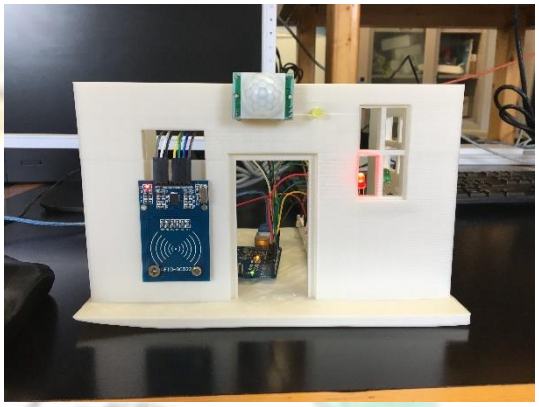




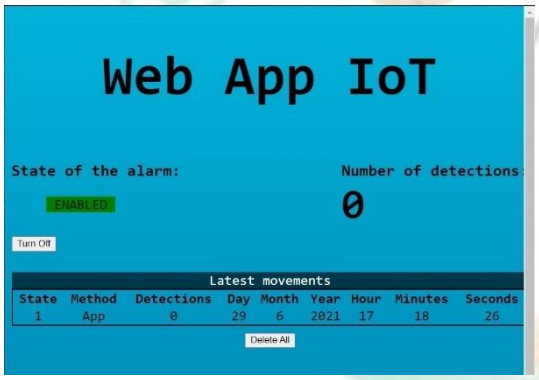
Turning on and off the alarm through the Web App:

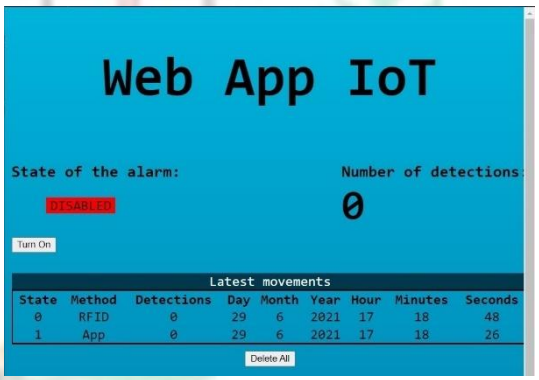


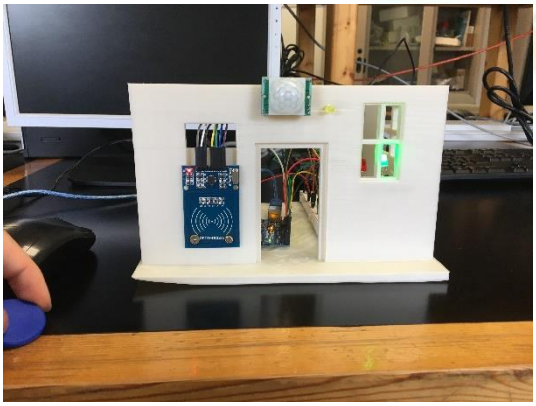


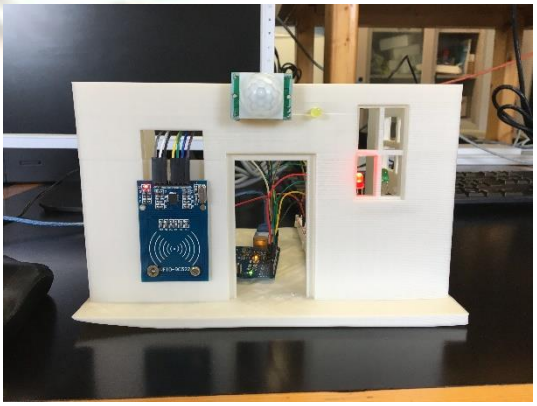


Turning on the alarm through the Web App and turning it off using the RFID tag:

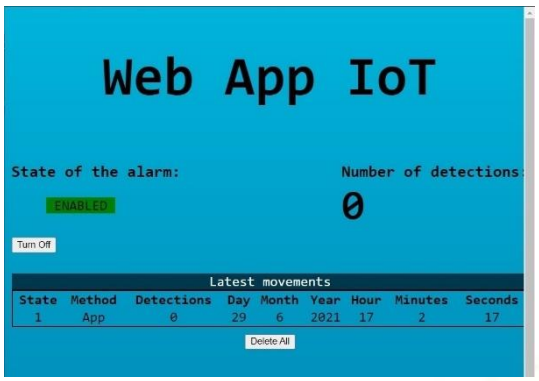


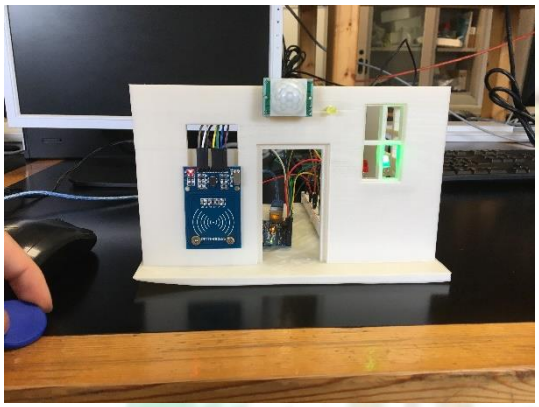


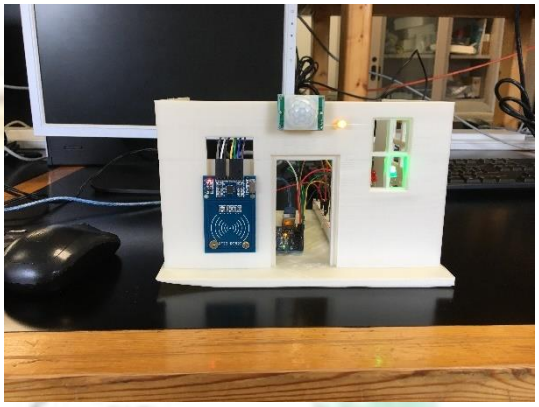




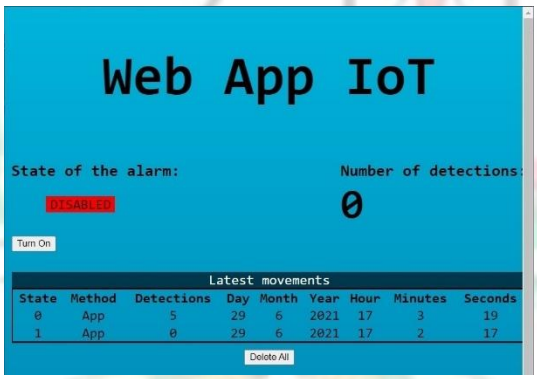
Turning on the alarm and detecting several movements:

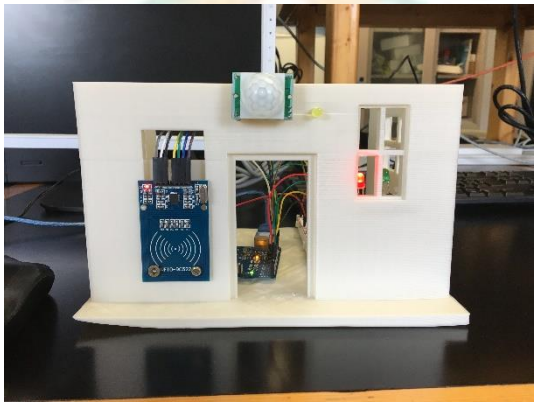






The app displays the number of movements detected in the period when the alarm is enabled. Then, when the alarm is turned off, it displays in the table how many movements were detected in that period in total.





6.- Summary and evaluation

6.1.- Summary

To conclude this wide explanation of what this project is about and which are the goals to be reached. Since I want it to be open source it is important to write a very easy and accessible explanation so everyone can understand, regardless the education.

This project is about helping people to keep their homes a little bit safer with a very simple gadget that is far cheaper than most alarms on the market and the most important thing these days, you own the data generated from this tool.

I tried my best to make the clearest explanation possible for this project, but in case there are some loose ends, there is a huge open source community always willing to help. But I tried to explain the process since the beginning to the very end, from the installation of the necessary elements to the test that certifies that everything is working correctly.

6.2.- Evaluation

This project has had a huge impact on me. It has helped me discover the IoT world and the potential that this industry has. Also, I have enjoyed every moment working on this project, which was the first goal I set before starting the final project.

In addition, I have enjoyed that much working on this project until the point I am going to focus the next step of my professional career in learning more things about IoT technologies.

Also, the results of the project are quite positive. As shown in chapter 5, the project works according to the expectations in every possible scenario. Apparently, there are no bugs detected which is one of the most important things in every software project.

Summing up everything, I am very happy with the development and the final result of this project, and I hope it will help someone who want to make it at home. I have tried my best to make the explanation as easy as possible due to the fact it is going to be an open source project. If it helps at least one person, I will be satisfied.

7.- Feasible improvements

In the hardware part, I wanted to include the solar panel besides the external battery because I am very concerned about the global warming and climate change, and I think these modern devices that are currently in development are thought to be more environmentally friendly as time passes by. Also, it could be a good idea to go one step further and do a little more research on the Bluetooth connection between Arduino and Raspberry Pi to make the project more flexible in terms of not feeling the need of having both devices close and connected though an USB wire. Furthermore, and I think the most important improvement or the first thing I would include once the project is presented is the camera. After some research, I realized that there are some IP cameras that can be connected to the Raspberry Pi using the GPIO pins, they are quite expensive but, I would love to try this out definitely. Apart from other sensors that could have been included like temperature and humidity meter, a LDR resistor to save energy and only turn on the light when it is dark, a sound sensor and so on… There are plenty of sensors.

In the software part, from my point of view there is room for improvement, but I kept this style to be easier for the reader to understand. There are some pieces of code, in the Arduino and in the Raspberry side, that are repeated but the only point to keep it that way is the one I have mentioned above. One of the disadvantages on the Arduino side is that the PIR sensor, due to its way of construction, when it does not detect movement no more, it keeps blocked for a short period of time but there is nothing I can do about it, I just keep this delay for 3 seconds, its minimum. In addition, a good improvement could be to design a method of notifications like some email or a pop-up alert, every time movement is detected when the alarm is enabled and a log in system where the app is able to identify who turns the alarm on/off.

Apart from that if someone wants to install it outdoors, they can design their own case and print it with a 3D printer, like I did to help the demonstration of the project to make it clearer and easier to understand the main idea, to protect the boards.

8.- Bibliography

[1] J. Chase, "The Evolution of Internet of Things," in White Paper, Texas Instruments, p. 6, September 2013.

https://www.ti.com/lit/ml/swrb028/swrb028.pdf?ts=1613724554660&ref_url=https%253A%252F%252Fwww.google.com%252F

[2] “The "Only" Coke Machine on the Internet". Carnegie Mellon University. November 2014.

https://www.cs.cmu.edu/~coke/history_long.txt

[3] "Internet of Things Done Wrong Stifles Innovation". InformationWeek. 7 July 2014. Retrieved November 2014.

https://www.informationweek.com/strategic-cio/executive-insights-and-innovation/internet-of-things-done-wrong-stifles-innovation/a/d-id/1279157

[4] Raji, R.S. (1994). "Smart networks for control". IEEE Spectrum. 31 (6): 49–55.

https://ieeexplore.ieee.org/document/284793

[5] Pontin, Jason (29 September 2005). "ETC: Bill Joy's Six Webs". MIT Technology Review. Retrieved November 2013.

https://www.technologyreview.com/view/404694/etc-bill-joys-six-webs/

[6] Unraveling the Story of Evolution of IoT and Its Rapid Adoption. Embitel. May 2018.

https://www.embitel.com/blog/embedded-blog/unraveling-the-story-of-evolution-of-iot-and-its-rapid-adoption

[7] Getting started with Arduino. Arduino Projecthub.

https://create.arduino.cc/projecthub/yeshvanth_muniraj/getting-started-with-arduino-bcb879

[8] 2020: Life with 50 billion connected devices. March 2018.

https://ieeexplore.ieee.org/document/8326056

[9] The 9 most important applications of the Internet of Things (IoT). Fracttal USA. August 2019.

https://www.fracttal.com/en/blog/the-9-most-important-applications-of-the-internet-of-things

[10] Arduino UNO Rev3. Arduino Store.

https://store.arduino.cc/arduino-uno-rev3

[11] PIR Motion Sensor Detector Module HC-SR501. Robu.In Store.

https://robu.in/product/pir-motion-sensor-detector-module-hc-sr501/

[12] What is RFID? How It Works? Interface RC522 RFID Module with Arduino.

https://lastminuteengineers.com/how-rfid-works-rc522-arduino-tutorial/

[13] Lectura de tarjetas RFID con Arduino y lector MIFARE RC522. October 2016.

https://www.luisllamas.es/arduino-rfid-mifare-rc522/

[14] ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 64).

[15] Edwards, Kimberly D. "Light Emitting Diodes" (PDF). University of California at Irvine. p. 2. Retrieved January 12, 2019.

https://cpb-us-e2.wpmucdn.com/faculty.sites.uci.edu/dist/a/326/files/2013/11/RDGLED.pdf

[16] Peláez, E. A; Villegas, E. R (2007). LED power reduction trade-offs for ambulatory pulse oximetry. 2007 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. 2007. pp. 2296–9.

https://ieeexplore.ieee.org/document/4352784

[17] "LED Basics | Department of Energy". www.energy.gov. Retrieved October 22, 2018.

https://www.energy.gov/eere/ssl/led-basics

[18] "LED Spectral Distribution". optiwave.com. July 25, 2013. Retrieved June 20, 2017.

https://optiwave.com/resources/applications-resources/optical-system-led-spectral-distribution/

[19] ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 44-45).

[20] ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 45-46).

[21] Lyle Russell Williams; see The New Radio Receiver Building Handbook Lulu.com, 2006 ISBN 1847285260 page 83.

https://books.google.es/books?id=XiKgKdeBi6cC&printsec=frontcover&dq=isbn:1847285260&hl=es&sa=X&ved=2ahUKEwjbnPuy3YrvAhU-AWMBHdXuBnQQ6AEwAHoECAAQAg#v=onepage&q&f=false

[22] ELEGOO - The Most Complete Starter Kit for UNO V1.0.19.09.18 (Page 43-44).

[23] "Raspberry Pi Foundation - About Us". Raspberry Pi. Retrieved 23 August 2020.

https://www.raspberrypi.org/about/

[24] "Ten millionth Raspberry Pi, and a new kit – Raspberry Pi". 8 September 2016. Retrieved 9 September 2016.

https://www.raspberrypi.org/blog/ten-millionth-raspberry-pi-new-kit/

[25] Liz Upton (25 April 2013). "The Raspberry Pi in scientific research". Raspberry Pi. Retrieved 3 April 2020.

https://www.raspberrypi.org/blog/the-raspberry-pi-in-scientific-research/

[26] "Eben Upton CBE". Archives of IT. Retrieved 17 November 2020.

https://archivesit.org.uk/interviews/eben-upton-cbe/

[27] Gibbs, Samuel (18 February 2015). "Raspberry Pi becomes bestselling British computer". The Guardian. Retrieved 28 December 2016.

https://www.theguardian.com/technology/2015/feb/18/raspberry-pi-becomes-best-selling-british-computer

[28] Upton, Ebon (14 December 2019). "Ebon Upton tweet - sales up to 30 million". the twitter. Retrieved 26 February 2020.

https://twitter.com/EbenUpton/status/1205646606504275968?s=19

[29] "About Us". sonypencoed.co.uk. Retrieved 27 September 2017.

http://www.sonypencoed.co.uk/about/

[30] Tung, Liam (27 July 2017). "Raspberry Pi: 14 million sold, 10 million made in the UK | ZDNet". ZDNet.

https://www.zdnet.com/article/14-million-raspberry-pis-sold-10-million-made-in-the-uk/

[31] Mod, Shreepanjali. June 2020. Top 11 Applications of Raspberry Pi.

https://www.engineersgarage.com/raspberrypi/top-11-applications-of-raspberry-pi/

[32] Hasan, Mehedi. Top 20 best Raspberry Pi projects that you can start right now. https://www.ubuntupit.com/20-best-raspberry-pi-projects-that-you-can-start-right-now/

[33] "OS/Linux Distributions using Apache". secure1.securityspace.com. Retrieved 2018-09-17.

https://secure1.securityspace.com/s_survey/data/man.201808/apacheos.html

[34] Netcraft Market Share for Top Servers Across All Domains August 1995 - today (monthly updated).

https://news.netcraft.com/archives/category/web-server-survey/

[35] "February 2009 Web Server Survey". Netcraft. Archived from the original on 26 February 2009. Retrieved 2009-03-29.

http://news.netcraft.com/archives/2009/02/18/february_2009_web_server_survey.html

[36] What Is a VPN? - Virtual Private Network.

https://www.cisco.com/c/en/us/products/security/vpn-endpoint-security-clients/what-is-vpn.html

[37] Saoirse Kerrigan (27 April 2018). "Virtual Private Networks - How they Work and Why You Might Need One". Interesting Engineering.

https://interestingengineering.com/virtual-private-networks-how-they-work-and-why-you-might-need-one

[38] Vuk Mujović (17 August 2018). "The history of VPN". Le VPN.

https://www.le-vpn.com/history-of-vpn/

[39] Luka Arežina (14 November 2019). "VPN statistics for 2020 – Keeping internet privacy alive".

https://dataprot.net/statistics/vpn-statistics/

[40] "phpMyAdmin Review". PCWorld. 2011-04-20. Retrieved 2017-10-27.

https://www.pcworld.com/article/233948/phpmyadmin.html

[41] Delisle, Marc (2010). Mastering phpMyAdmin 3.3.x for Effective MySQL Management. Packt Publishing. p. 359. ISBN 978-1-84951-354-8.

APPENDIX.- Configuration of the necessary tools on Raspberry Pi

The main tools we need to make this project work are the web server, the virtual private network, the databases, and the web application. I recommend installing them first, because then the beginners will be able to play with little web pages to learn how does it work and understand what is being done.

Installation of the web server and other plug-ins

To begin with, it is always recommended to make sure that the Raspberry has the latest version of the operating system that is running on the board, in this case, Raspbian. So, using the command line interface (CLI), write:

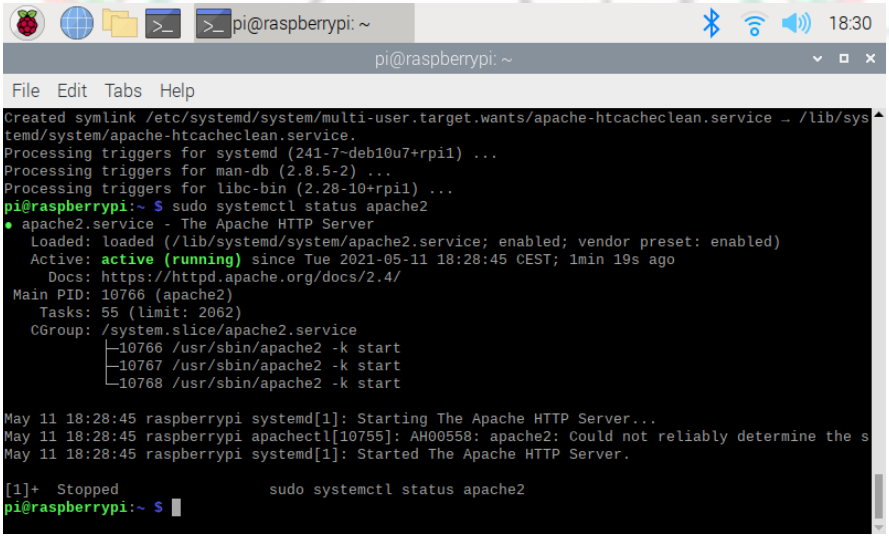
```
sudo apt-get update
sudo apt-get upgrade
```

Followed by the installation of the web server itself. I have chosen Apache [1] because it was the web server which I am more familiarized with, but there are other brands that provide web servers like XAMPP or Nginx.

```
sudo apt install apache2
```

After the installation, it is possible to check if the server works properly with the command:

```
sudo systemctl status apache2
```



*The **Active** line marks the current status of the web sever*

[1] https://alexpro.sytes.net/como-instalar-el-servidor-web-apache-en-raspbian-stretch-9/

To be able to store data from the web application, we are working with MariaDB which is an open source MySQL database manager [2].

```
sudo apt install mariadb-server
```

With the goal of making the database a little more secure, I am executing this command that will require some attention later:

```
sudo mysql_secure_installation
```

There is required a password for the root user, then it is going to be asked some configuration questions on the construction of the database that I kept by default.

To install PHP [3], the tool that will make the web application interactive, run the command:

```
sudo apt install php libapache2-mod-php php-mysql
```

Then, restart the apache server with:

```
sudo systemctl restart apache2
```

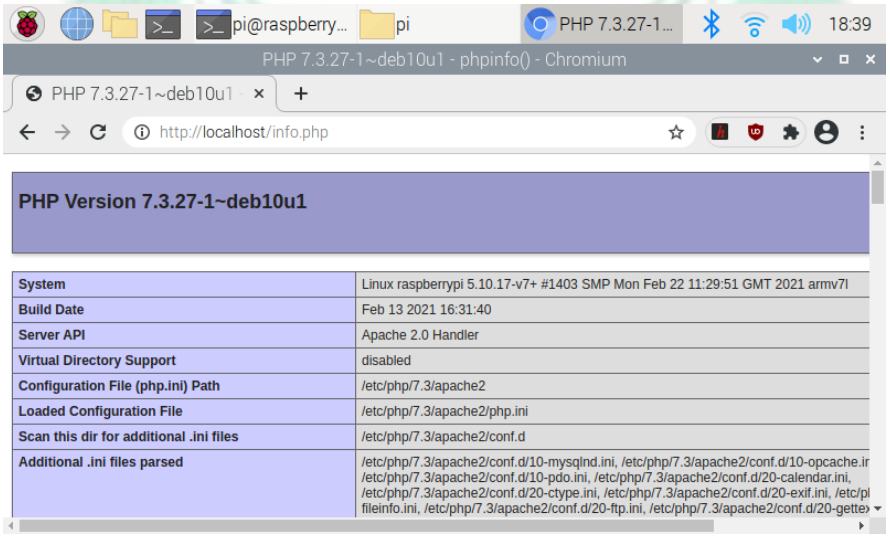
To check if PHP was installed correctly, we are going to run a test web application that will display some information of the PHP version installed.

```
sudo nano /var/www/html/info.php
```

It creates a PHP file and we are going to write:

```
<?php phpinfo(); ?>
```

Then, open the browser and search for the URL: http://localhost/info.php

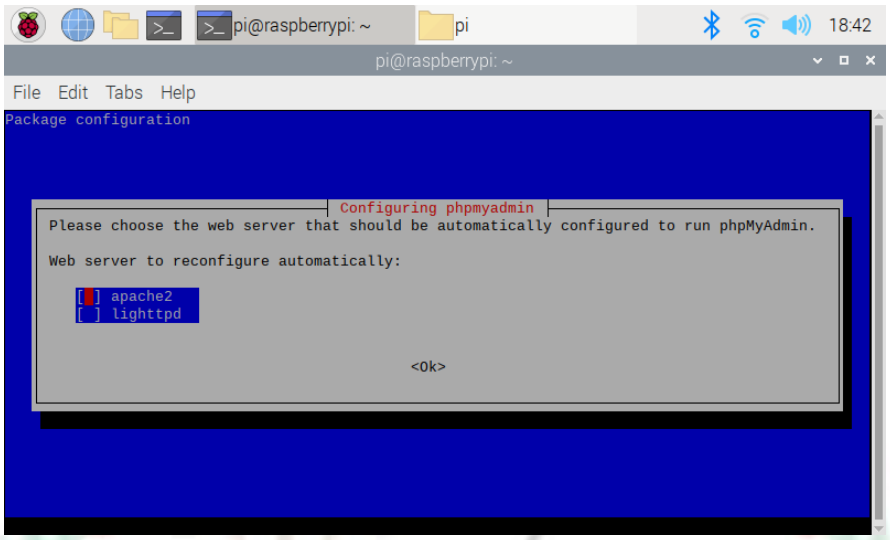


[2] https://alexpro.sytes.net/como-instalar-el-gestor-de-bases-de-datos-mysql-mariadb-raspbian/
[3] https://alexpro.sytes.net/como-instalar-php-en-raspbian/

In addition, there is a tool that makes way easier to work with SQL databases, it is phpMyAdmin [4]. It is a platform I know, and it is going to be used to manage the databases implemented in this project. To install it:

```
sudo apt install phpmyadmin
```

Mark the *apache2* option:



During the installation, it needs a database installed and configured before it can be used. You can provide the initial database but, if not, the platform can configure an initial database itself. Then, it will ask to give a password that the user does not need to know, leave it in blank to generate a random one.

After the installation, there are two commands left needed to make the phpMyAdmin work:

```
sudo ln -s /etc/phpmyadmin/apache.conf /etc/apache2/conf-available/phpmyadmin.conf
```

```
sudo ln -s /etc/apache2/conf-available/phpmyadmin.conf /etc/apache2/conf-enabled/phpmyadmin.conf
```

I recommend after all this heavy installation to reboot the system.

```
sudo reboot
```

Then, the platform will be accessible at http://<ip_address>/phpmyadmin

Where the ip_address is the IP address of the Raspberry Pi.

[4] https://alexpro.sytes.net/como-instalar-phpmyadmin-en-raspbian-stretch-9/

But, before start using the phpMyAdmin, there is one more request, to create a user. So, going back to the command line interface, write:

```
sudo mysql -u root
```

```
USE mysql;
CREATE USER admin@localhost IDENTIFIED BY 'admin';
GRANT ALL PRIVILEGES ON * . * TO admin@localhost;
UPDATE user SET plugin='mysql_native_password' WHERE User='admin';
SET PASSWORD FOR admin@localhost = password('admin');
GRANT ALL ON *.* TO admin@localhost IDENTIFIED BY "admin" WITH GRANT OPTION;
FLUSH PRIVILEGES;
exit;
```

Now, the account to log in to phpMyAdmin is the username *admin* and password *admin*. **It is highly recommended to change this username and password to one more difficult to guess with the purpose of preventing security breaches. This is only a demonstration.**

There is needed one more thing to change. The web app is going to be prioritized, this means that is going to be the main page that will show up when you search for the URL: http://<ip_address>. To achieve that, write in the CLI:

```
sudo nano /etc/apache2/mods-enabled/dir.conf
```

The answer of that command will be something like this:

```
<IfModule mod_dir.c>
    DirectoryIndex index.html index.cgi index.pl index.php index.xhtml
index.htm
</IfModule>
```

So, the write before index.html our app which is index.php:

```
<IfModule mod_dir.c>
    DirectoryIndex index.php index.html index.cgi index.pl index.xhtml
index.htm
</IfModule>
```

And finally, we restart the apache server:

```
sudo systemctl restart apache2
```

Installation of the Virtual Private Network

The installation of the VPN [5] is quite easier than the web server. Like the web servers, there are plenty of open source VPNs and I have decided to use pivpn [6]. There is only needed to write on the CLI:

```
curl -L https://install.pivpn.io | bash
```

Just follow the instructions and keep the values by default. In my case I used the Google DNS provider. It is very important to remember the **port number** and the **public IP**. You can have a look at the commands here: https://www.joshualowcock.com/pivpn/

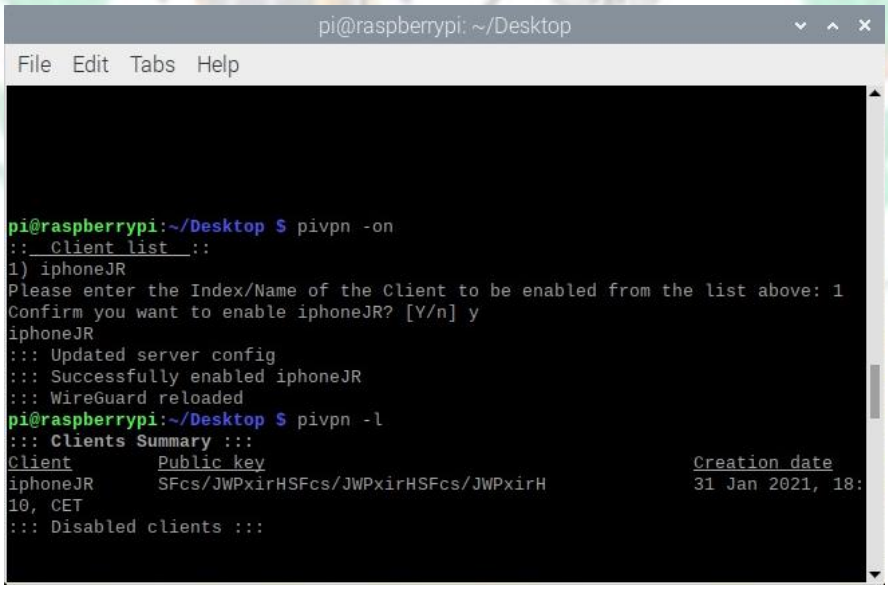
To add a client:

```
pivpn add
```

Then, enable the user by writing:

```
pivpn on
```

And select the user created.

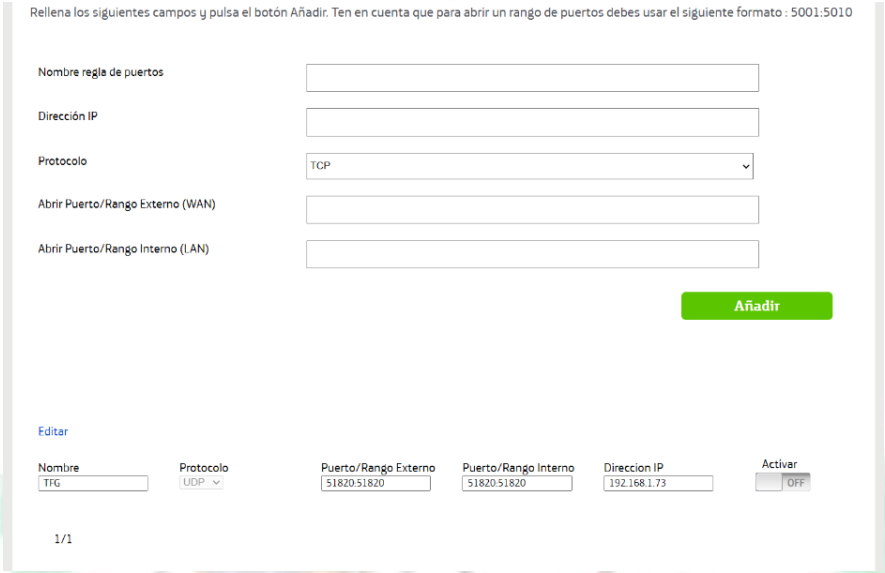


[5] https://alexpro.sytes.net/instalar-un-servidor-vpn-en-raspbian/
[6] https://www.pivpn.io/
[7] https://www.wireguard.com/

Finally, to configure the VPN on the device you want to use the app, download the Wireguard [7] app and scan the QR code using:

```
pivpn -qr
```

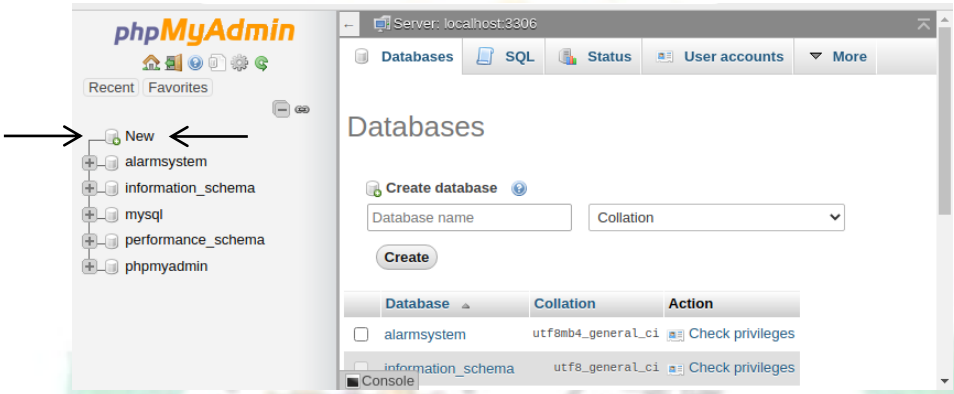
To make the VPN work with your network, it is needed to enable the port of the router (192.168.1.1), this is where you need the **port number** of the VPN and the **IP address** of the Raspberry that were mentioned before.



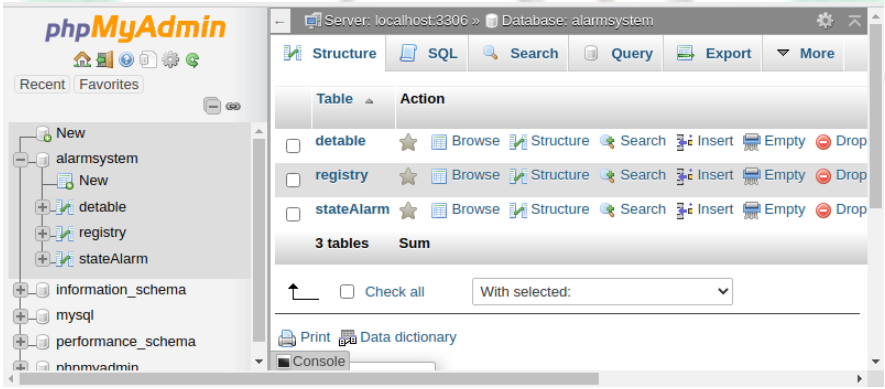
Creation of the databases

It is needed to create 3 tables on the database for this project to work. The purpose of this tables is to store some important values like the history of the movements within the state of the alarm and some volatile values like the current state of the alarm, that is changing constantly, and the detections made by the PIR sensor when the alarm is on.

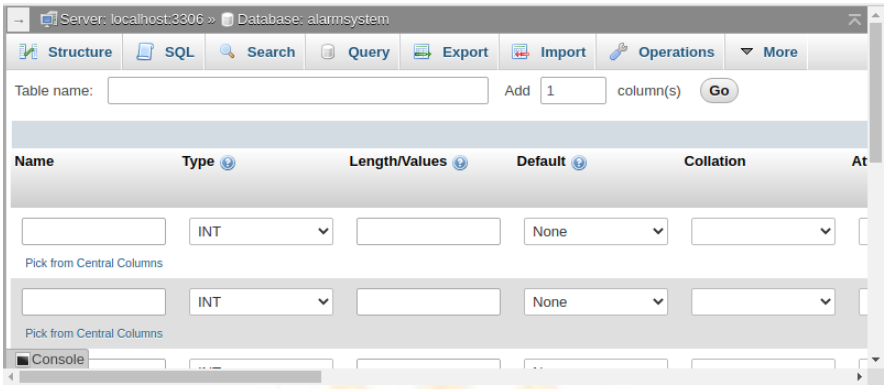
First of all, we need to create a database, at the address http://<ip_address>/phpmyadmin, where the tables are going to be stored. In this case, I named the database "*alarmsystem*".



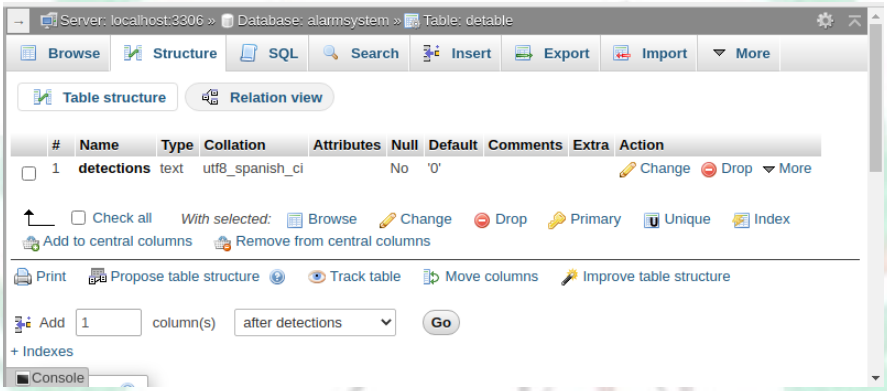
Now you can create every table you want inside the database *alarmsystem*, but this project only needs 3 of them. To create a table, just click on the "new" button below *alarmsystem*.



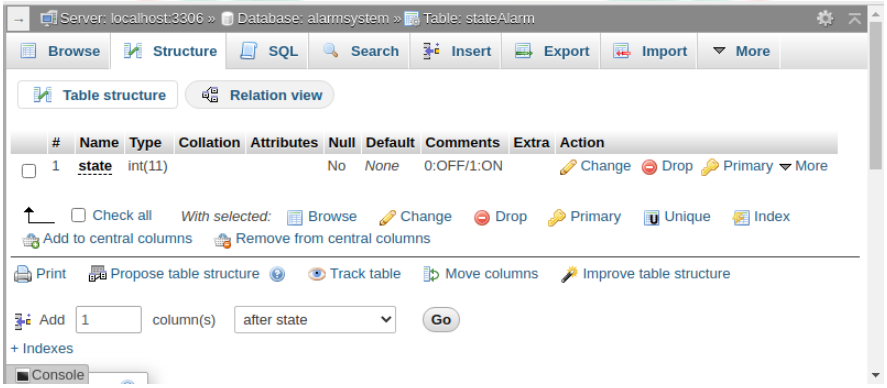
Then, you will configure the structure of each table on a page that will look like this. The structure and explanation of each table is explained just below.



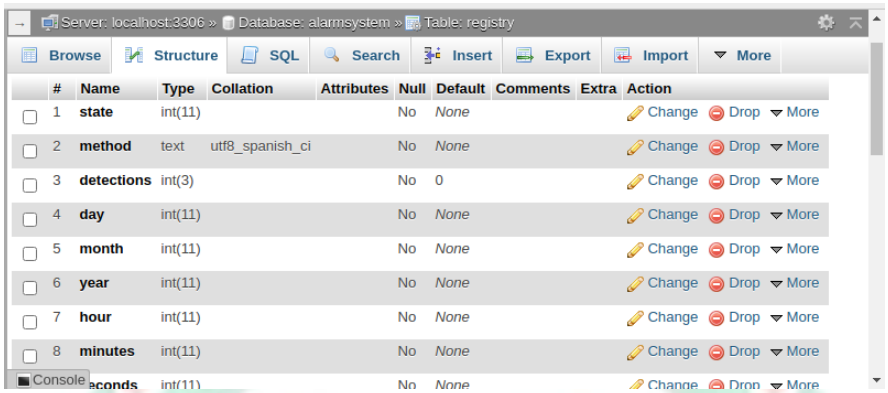
The table named *detable* is going to store the number of detections that were made when the alarm is on, to show it on the app. And the structure looks like this.



The table named *stateAlarm* stores the current state of the alarm every five seconds. The structure looks like this.

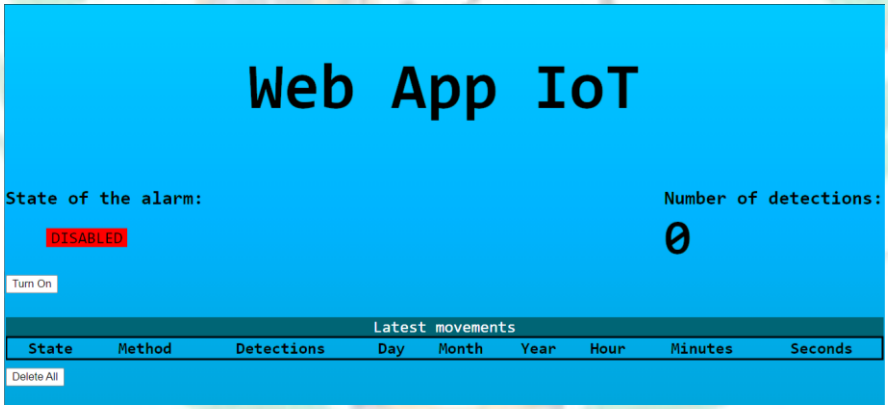


The table *registry* is the most complex one within this project because it stores the history of the movements that were made when someone turned on or off the alarm. It stores values like how the state of the alarm is changed (App/RFID) and the exact date in which the state of the alarm changed. Also, the state field will display a '0' if the alarm is disabled and will display a '1' if the alarm is enabled.



Just in case it does not work, it is available on my GitHub (https://github.com/Joserra13/TFG) a file (*alarmsystem.sql*) that is prepared and ready to be imported to another phpMyAdmin database.

Web application



The app was built taking into account a modern design resulting from the proper nature of the IoT industry. The app needs to be located in one specific path which is inside the web server's directory at */var/www/html*. The files *index.php* and *alarmmanagement.php* must be located in that exact address and the css file, which is *style.css*, must be at */var/www/html/css*.