ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA Y SISTEMAS DE TELECOMUNICACIÓN

PROYECTO FIN DE GRADO

TÍTULO: Desarrollo de un Sistema de Automatización de Vivienda basado en Arduino.

AUTOR: ……Yiman Chen……..

TITULACIÓN: Grado en Ingeniería de Sistemas de Telecomunicación……..

TUTOR ……Ari Rantala……

UNIVERSIDAD: ……TAMK University of Applied Sciences……

CENTRO: ……Main Campus……

PAÍS: ……Finlandia……

Fecha de lectura:

Calificación:

El Coordinador de Movilidad,

RESUMEN

En este documento se va a realizar el estudio y la implementación de sistemas de automatización de la vivienda basado en una plataforma de hardware libre llamado Arduino, este sistema en el sector eléctrico comúnmente se conoce como domótica o el hogar inteligente, que es uno de los sectores más emblemáticos de hoy el día en el campo de las nuevas tecnologías.

Vivimos en un mundo que cada vez esta más comprometido al medio ambiente, eso hace que la evolución del desarrollo de las tecnologías da mucha importancia al tema del consumo energético, hoy el día, sin duda las viviendas ocupa la mayor parte del consumo, por lo tanto muchas empresas empiezan a investigar y desarrollar nuevos sistemas eléctricos que ayuda a controlar y disminuir el consumo, como por ejemplo los sensores inteligentes o el internet de las cosas. Por el otro lado, cada vez hay más dispositivos conectados a la red, eso implica que la seguridad digital es un tema primordial que hay que mejorar, y gracias a los protocolos establecidos anteriormente, los nuevos softwares y hardwares hacen que eso sea posible, de manera que los usuarios puedan usar la red con más confianza y facilidad.

En este proyecto se va a realizar un sistema de automatización que será capaz de realizar las siguientes tareas: la activación de la luz mediante el teléfono móvil con un botón o la voz, la detección de llamas junto con el alarma de aviso en el caso de incendio, la indicación de la temperatura y la humedad del ambiente de la vivienda mediante una pantalla módulo PLD y también con el teléfono móvil, sistema de seguridad que comunica al inquilino mediante una notificación al teléfono móvil si detecta la entrada de algún desconocido. Para su implementación realizaremos un estudio detallado de los sensores utilizados, después implementaremos el pseudocódigo en un programa software para comprobar el correcto funcionamiento del sistema montado, luego realizaremos un estudio de las aplicaciones o software que nos ayuda a realizar la conexión entre el Arduino, ordenador y el teléfono móvil. Por lo último explicaremos los distintos interfaces de usuario que controla todo el sistema de automatización.

ABSTRACT

This is a document with study and implementation of home automation systems based on a free hardware platform. This system in the electrical sector is commonly known as home automation or smart home, which is one of the most emblematic nowadays in the field of new technologies.

We live in a world that is increasingly committed to the environment, that makes the evolution of the development of technologies gives great importance to the issue of energy consumption, nowadays, without a doubt homes occupy most of the consumption, so many companies are beginning to investigate and develop new electrical systems that help to reduce and control the consumption, such as smart sensors or the internet of things. On the other hand, there are more and more devices connected to the network, which implies that digital security is a fundamental issue that needs to be improved, and thanks to the protocols previously established, the new software and hardware make that possible, so that users can use the network with more confidence and ease.

This is an automation system project that will be able to carry out the following tasks: the activation of the light with the mobile phone through a virtual button or the voice, the detection of flames with the warning alarm in the case of fire, the indication of the temperature and humidity of the home environment through a PLD module screen and also with the mobile phone, security system that communicates to the tenant by a push notification to the mobile phone if it detects the entry of some stranger. For this implementation we will carry out a detailed study of the used sensors, then we will implement the pseudocode in a software program to check the correct functioning of the system. Later, we will study what kinds of applications o software we need to connect the Arduino, laptop and mobile phone. At the end, we will talk about the user interface that we have implemented to control the whole automation project.

KEYWORDS

Arduino, sensors, smart home, domestic, automation system, web server, embedded system.

CONTENTS

FIGURES

TABLE

1. INTRODUCTION

Thanks to the advancement of the development of new technologies, especially telecommunications systems, network systems, new software and hardwares, which have led to the improvement of the housing standard. The concept of the smart home system is gradually being integrated into our daily lives and has become one of the most controversial issues in the field of automatic control. Due to its wide perspective of applications and huge demand in the market, there are more and more technological companies that invest in this sector, generating a market of free competition.

This project is divided into 7 sections, the first section is the specifications and design restrictions, where the specifications of the materials used in the project will be discussed, the second section is the description of the proposed solution with a detailed description of the functions who will carry out this project. then there is the section of the sensors, and within this section there are 6 mini sections that will make an explanation with the most relevant characteristics of each sensor in detail.
then there is the Arduino section, which is the heart of our project, in this section there will be an explanation of what the Arduino is, its development ideology, the Arduino model used in this project and the shields compatible with Arduino. In the next section speaks of the ethernet shield with the explanation of each of its part, then this is the design of the automation system and focuses on the hardware, software and applications used for implementation. The last one is the conclusion of the project.

2. ACRONYMS AND CONCEPT DEFINITION

HTML Hypertext Markup Language. Standard markup language used to create web pages. HTML code is rendered by the web browsers to visible web pages.

IP address Internet Protocol Address. Unique address identifier that is given to device connected to Internet.

LAN Local Area Network. A group of devices on a small geographic area that share a common communications line or wireless link.

HC-SR501 Motion sensor with infrared technology, sensor distance adjustable.

DHT11 Temperature and humidity sensor

C language High level computer programming language with a structures programming, lexical variables scope and recursion, process low-level memory, generate a small amount of machine code, it runs without any support of the operating environment.

C++ High level computer programming language that inherited from C. It can perform object based programming with abstract data types.

USB CABLE Universal Serial Bus cable, it was invented to connect the personal computers and peripheral devices.

HVAC Heating,Ventilation and air Conditioning, it is air conditioning system, including the temperature, air clarity, humidity and air circulation control.

LED Light Emitting Diode, electronic device that emits light when an electrical current is passes through it.

NTC: Negative Temperature Coefficient, resistance decreases as the temperature rises, it is usually used as a temperature sensor, or in series with a circuit as an input current limiter.

UARTs: Universal Asynchronous Receiver-Transmitter, it is a hardware device which the main work of this chip is to convert received parallel data to serial, or convert serial data to parallel, it is programmable, you can set up the speed, bits, length and parity.

RH Relative Humidity, is the ratio of the partial pressure of water vapor to the equilibrium vapor pressure of water at a given temperature. It also depends on the pressure of the system of interest.

SPI Serial Peripheral Interface, synchronous serial communication interface for show distance communication.

SD card Secure Digital card, non volatile memory card format used for portable de vices.

GND Ground, reference point in an electrical circuit.

3. SPECIFICATIONS AND DESIGN RESTRICTIONS

The specifications of the different used technologies are the following:

● Ethernet shield W5100: It is based on the Wiznet W5100 ethernet chip providing a network (IP) stack capable of both TCP and UDP. The on-board micro-SD card slot can be used to store files for serving over the network. The W5100 and SD card share the SPI bus, only one can be active at a time. If you are using both peripherals in your program, this should be taken care of by the corresponding libraries. If you're not using one of the peripherals in your program, however, you'll need to explicitly deselect it. The shield provides a standard RJ45 ethernet jack.

● Arduino MEGA 2560: USB feed with Atmel AVR micro controller, use memory static random access SRAM with Flash storage capacity and EEPROM.

● 830 point solder-less test plate with male connection cables.

● RGB LEDs (diodes that emit light) that work with a voltage between 2V and 3.2V, a maximum current of 20 mA and a maximum brightness (RGB) of 2800 mcd, 6500 mcd and 1200 mcd.

● The HC-SR501 motion sensor is based on infrared technology, module Automatic control, with the design of the LHI778 probe, high sensitivity, high Reliability, ultra low voltage operating mode.

● The Android controller of Arduino via ethernet.

- DHT11 humidity and temperature sensor with power supply between 3V and 5V, Temperature measurement range from 0ºC to 50º C, temperature resolution of 0.1ºC, Humidity measurement range 20% to 90% RH and Humidity Resolution: 1% RH, the time it takes to measure the humidity is 1 second.

- We will use the C ++ programming language in the Arduino software.

- The flame sensor that can detect ordinary wavelength light source in the range of 760 nm to 1100 nm and a maximum detection distance of 100cm.

4. DESCRIPTION OF THE PROPOSED SOLUTION

The main objective of this project is the development and study of a automation system of home installations based on a free hardware platform, Arduino, in which we will need other auxiliary devices for its implementation. The main functionalities that will be analysed and studied in this project are the following:

● Detection of the temperature and humidity of the environment through a temperature and humidity sensor (DHT11) and display the information on an LCD screen and can also check the ambient information with web server. This information will be updated every 10 minutes.

● A security system if it detects some movement with the PIR motion sensor will activate and send a message to the mobile phone of the tenant to notify him of the entrance of the home of some strangers.

● A fire detection system to improve home security through a flame sensor, which will warn the tenants and neighbours through an alarm.

● Connection of the Arduino MEGA 2560 to the mobile phone through a ethernet shield, which allows us to do some operations through the web server or applications. It can receiving connections from a PC, Tablet or mobile, is able to generate connections with others devices equipped with web server.

● Control the lights of the home from a PC or a mobile phone through the connection previously done with the ethernet shield, in which we can perform operations such as turning the light on and off from the web server, with some virtual switch or voice control.

5. SENSORS

The sensor is a device or subsystem that the main propose is to detect physical events or changes in its environment and to convert it into an electrical signal that is compatible with electronic circuits. There are two types of sensors for Arduino, digital and analog. Digital sensors work with output between zero and one, which is translated to sensors voltage range.

5.1 Temperature and humidity sensor

DHT11 digital temperature and humidity sensor (figure 1 left) contains calibrated digital output of the temperature and humidity. It has digital modeles collection technology and the temperature and humidity sensing technology, the sensor has high reliability and excellent long-term stability. It is formatted with a resistive sense of wet components and an NTC temperature measurement devices, and connected with a high-performance 8-bit micro controller. The size of the sensor is in the figure 1 right.

DHT11 digital temperature and humidity sensor has a excellent quality, fast response, a strong anti-interference ability, long distance signal transmission with a precise calibration, it is low cost, relative humidity and temperature measurement, using low voltage. The sensor has a power supply of DC 3.5V to 5.5V, the sampling period is more than 2 seconds and the supply current is 0.3 mA with standby of 60 µA.DHT11 uses a simplified single bus communication. Single bus that only one dateline, the system of data exchange, control by a single bus to complete. This sensor is used in many applications, for example HVAC, dehumidifier, testing and inspection equipment, automotive, automatic control, humidity regulator, home appliances, medical and humidity measurement and control. The main parameters of this sensor are in the table 1 and table 2.

Humidity	
Resolution	16 Bit
Repeatability	±1% RH
Accuracy	At 25ºC± 5%RH
Interchangeability	Fully interchangeable
Response time	1/e 25 ºC 6s 1m /s air 6s
Hysteresis	< ± 0.3% RH
Long-term stability	<±0.5% RH

TABLE 1. Main Parameters of DHT11(Humidity)

Temperare	
Resolution	16 Bit
Repeatability	±1% RH
Accuracy	At 25ºC± 5%RH
Interchangeability	Fully interchangeable
Response time	1/e 25 ºC 6s 1m /s air 6s
Hysteresis	< ± 0.3% RH
Long-term stability	<±0.5% RH

TABLE 2. Main Parameters of DHT11(Temperature)

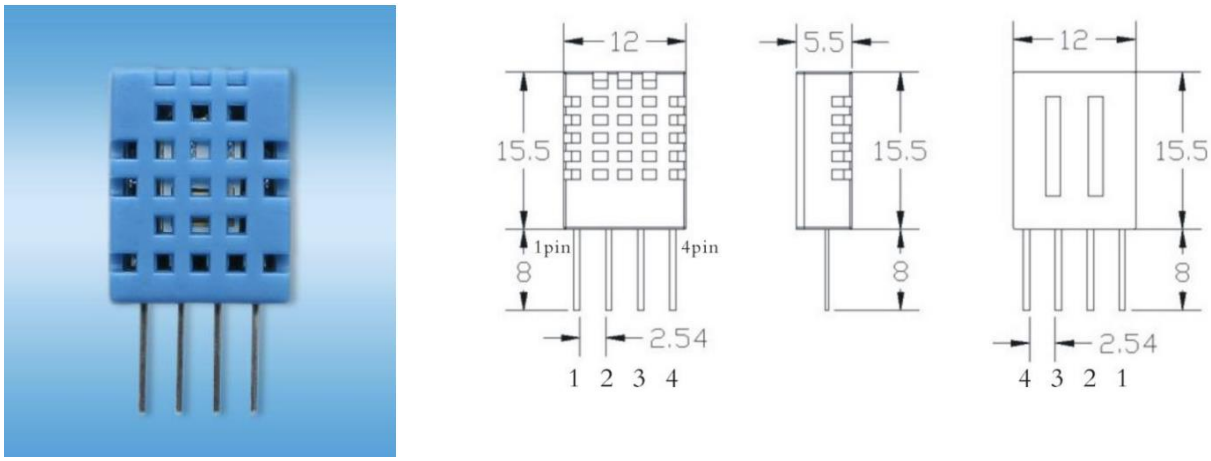

FIGURE 1. DHT11 Humidity and Temperature Sensor(left) and Dimensions (Unit: mm)

5.2 Flame sensor

KY-026 flame sensor (figure 3) is sensitive to the flame and radiation, it can detect light with the range of the wavelength from 760nm to 1100nm, the maximum distance to detect is one meter, more parameters are in the tabla 3. The output of this sensor can be two outputs mode, analog or digital, with analog outputs mode there is a real time output voltage signal on the thermal resistance, and with the digital outputs mode when the temperature rises to certain threshold, the output high and low signal threshold can be adjustable with a potentiometer.

This flame sensor has three components on its board, the sensor unit, amplifier and the comparator. The sensor unit is at in front of the module measures the area physically and send an analog signal to the amplifier. The amplifier amplifies the analog signal according to the resistant value of the potentiometer, then sends the analog signal to the analog output of the module. The third component is a comparator which switches the digital output and the LED if the signal falls under a specific value. Usually it is used in flame alarm and in fire fighting robots.

Flame Sensor	
Detection distance	20 cm with 4.8V and 1m with 1v
Detection angle	60 degrees
Operating voltage	3.3 V to 5 V
Digital output DO	Switch outputs 0 and 1
Analog output AO	Voltage output

TABLE 3. Main Parameters of KY-026 Flame Sensor

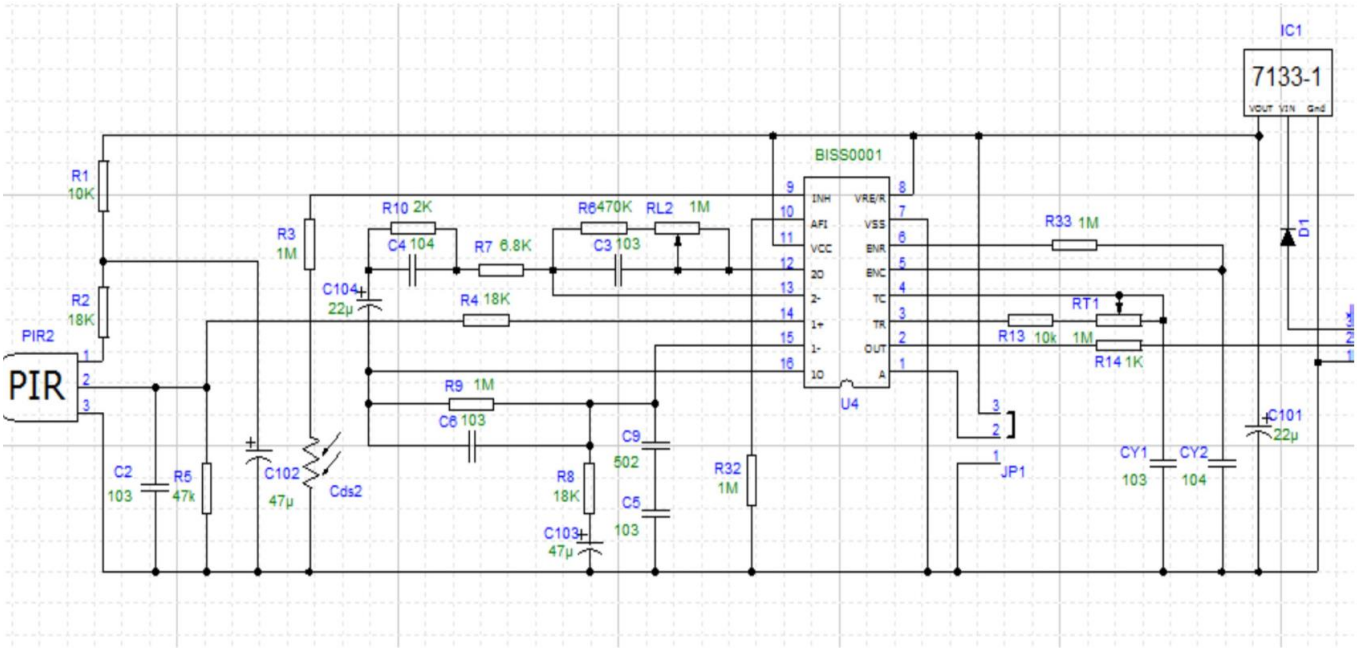

FIGURE 2. Flame Sensor with Interface Description

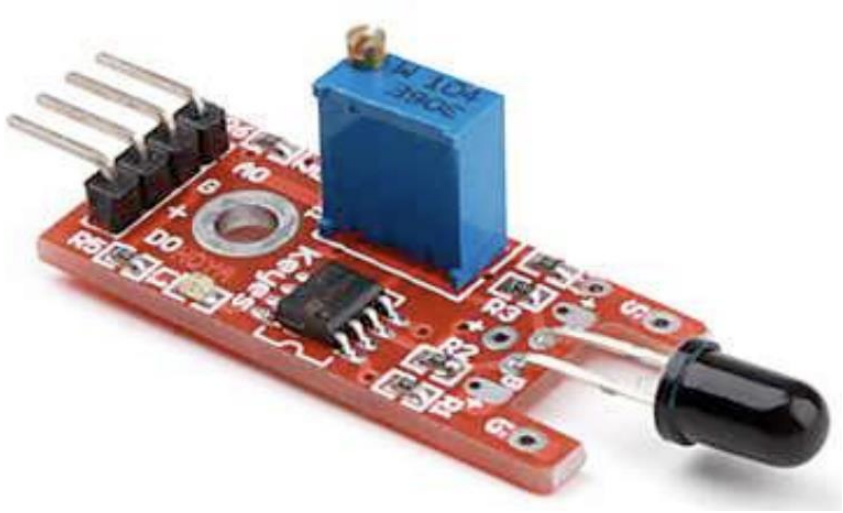

FIGURE 3. KY-026 Flame Sensor

5.3 Motion sensor

HC-SR501 PIR motion detector (figure 4) is based on infrared technology, it has a high sensibility and high reliability with automatic induction, the photosensitive control depends on the day or light intensity without induction. Furthermore, it has the temperature compensation that can be used for performance compensation, if the ambient temperature rises the distance is slightly shorter.
Triggered in two ways, the first one called non-repeatable trigger, the sensor output high, the delay times is over, the outputs automatically changed from high level to low level. The second one called repeatable trigger, the sensor output high, the delay times is periodic, if there are some motion detected in its sensing range, the output will always maintain high until the motion left after the delay will be high level goes low.

With induction blocking time, default setting is 2.5 seconds blocked time, when the sensor changes the output (from high to low), it is followed by blockade set period of time, during this time, the sensor will not listen to any petition of sensor signal. This feature can be achieved sensory put time and blocking time interval between the work can be applied to interval detection products and this function can inactivate a variety of interference in the process of load switching.

The wide operating voltage range DC 4.5V to 20V. The micro-power consumption is less than 50 microamps, specially suits for battery-powered automatic control products. The output high signal is easy to achieve docking with the various types of circuit and the sensor incorporates a distance potentiometer clockwise rotation, that gives you the possibility to increase or decrease the sensing distance, the maximum is about 7 meters and the minimum 3 meters. It also has a delay potentiometer clockwise rotation sensor the delay lengthened 300 seconds, or shorten the induction delay 5 seconds. For more parameters can consult the table 4.

HC-SR501 PIR	MOTION DETECTOR
Voltage	5V - 20V
Power consumption	65mA
TTL output	3.3V, 0V
Delay time	Ajustable 3 sec > 5 min
Lock time	0,2 sec
Trigger methods	L -disable repeat trigger, H -enable repeat trigger
Sensing range	Less than 120 degree, within 7 meters
Temperature	- 15 ºC to +70 ºC
Dimensions	32*24 mm, distance between screw 28mm, Lens dimension in diameter 23 mm

TABLE 4. Main Parameters of HC-SR501 PIR Motion Sensor

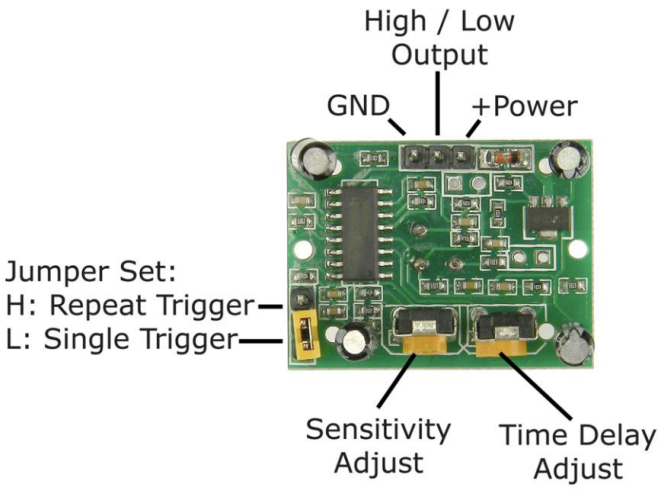

FIGURE 4. HC-SR501 PIR Motion Detector Circuit

5.4 Buzzer

The KY-012 buzzer (figure 5) is a audio signal device with integrated role in structure of electronic transducers, it has DC voltage power supply, this buzzer con be directly connected to a continuous sound and it produces a single tone sound when the signal is high, it has three pins, the alimentation pin, the input pin and the groups pin. There is no oscillation source, need square wave to drive. In the table 5 you can see the main parameters of the buzzer.

KY-012 Buzzer	
Working Voltage	3.5V - 5.5V
Working current	<25mA with a max of 30mA
Resonance frequency	2500Hz ± 300Hz
Minimum sound output	85dB 10 cm
Working temperature	- 20 ºC to +70 ºC
Dimensions	18.5mm x 15mm

TABLE 5. Main Parameters of KY-012 Buzzer

FIGURE 5. KY-012 Buzzer

5.5 LED

Light Emitting Diode is a semiconductor device that can emit visible light when an electronic current passes through it. The light is not very strong and it is monochromatic, occurring at a single wavelength, the range of the colour that the LED can emit is from red with a wavelength of approximately 700 nanometers to blue-violet, it is more o less 400 nanometers. The dimensions in millimetre of the LED is showed is in the figure 6.

One of the features of the LED is that has a low power consumption with long life-solid state reliability, available on tape and reel. The operating parameters are in the table 6:

LED Electrical Characteristics	
Forward voltage	Max 2.5 V
Reverse current	10 µA
Power dissipation	75 mW
DC Forward Current	30 mA
Peak forward current	155 mA
Reverse voltage	5 V
Operating temperature	- 40 ºC to +85 ºC

TABLE 6. Main Parameters of LED

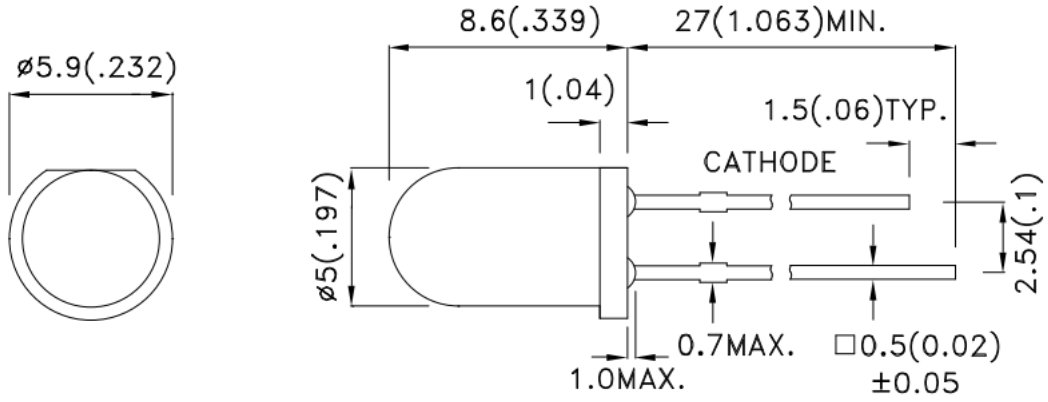

FIGURE 6. LED Dimensions (unit: mm)

5.6 LCD Display

The HD44780U dot-matrix liquid crystal display controller and driver LSI displays alphanumerics, Japanese kana characters, and symbols. All the functions are internally provided on one chip, such as display RAM, character generator, and liquid crystal driver, required for driving a dot-matrix liquid crystal display a minimal system can be interfaced with this controller/driver. A single HD44780U can display up to one 8-character line or two 8-character lines.

The process of controlling the display involves putting the data that form the image of what you want to display into the data registers, then putting instructions in the instruction register. The LiquidCrystal Library simplifies this for you so you don't need to know the low-level instructions. The Hitachi-compatible LCDs can be controlled in two modes: 4-bit or 8-bit. The 4-bit mode requires seven I/O pins from the Arduino, while the 8-bit mode requires 11 pins. It has an automatic reset circuit that initialises the controller/driver after power on, with internal oscillator with external resistors, lower power consumption.

HD44780U	
Lower voltage operation support	2.7V to 5.5V
Wide range of liquid crystal display driver power	3 V to 11V
Liquid crystal drive waveform	One line frequency AC waveform
Correspond to high speed MPU bus interface	2MHz when Vcc = 5V
Capacity	8 character per line

TABLE 7. Electrical Characteristics of LCD Display

The micro-controller has to manipulate several interface pins at once to control the display, LCD display has a parallel interface, The interface consists of the following pins (figure 7):

- A register select (RS) pin that controls where in the LCD's memory you're writing data to. You can select either the data register, which holds what goes on the screen, or an instruction register, which is where the LCD's controller looks for instructions on what to do next.

- Read/Write (R/W) pin that selects reading mode or writing mode.

- An Enable pin that enables writing to the registers.

- 8 data pins (D0 -D7). The states of these pins (high or low) are the bits that you're writing to a register when you write, or the values you're reading when you read.

There's also a display constrast pin (VE), power supply pins (VSS +5V and VDD Ground) and LED Backlight (Bklt+ and BKlt-) pins that you can use to power the LCD, control the display contrast, and turn on and off the LED backlight, respectively.

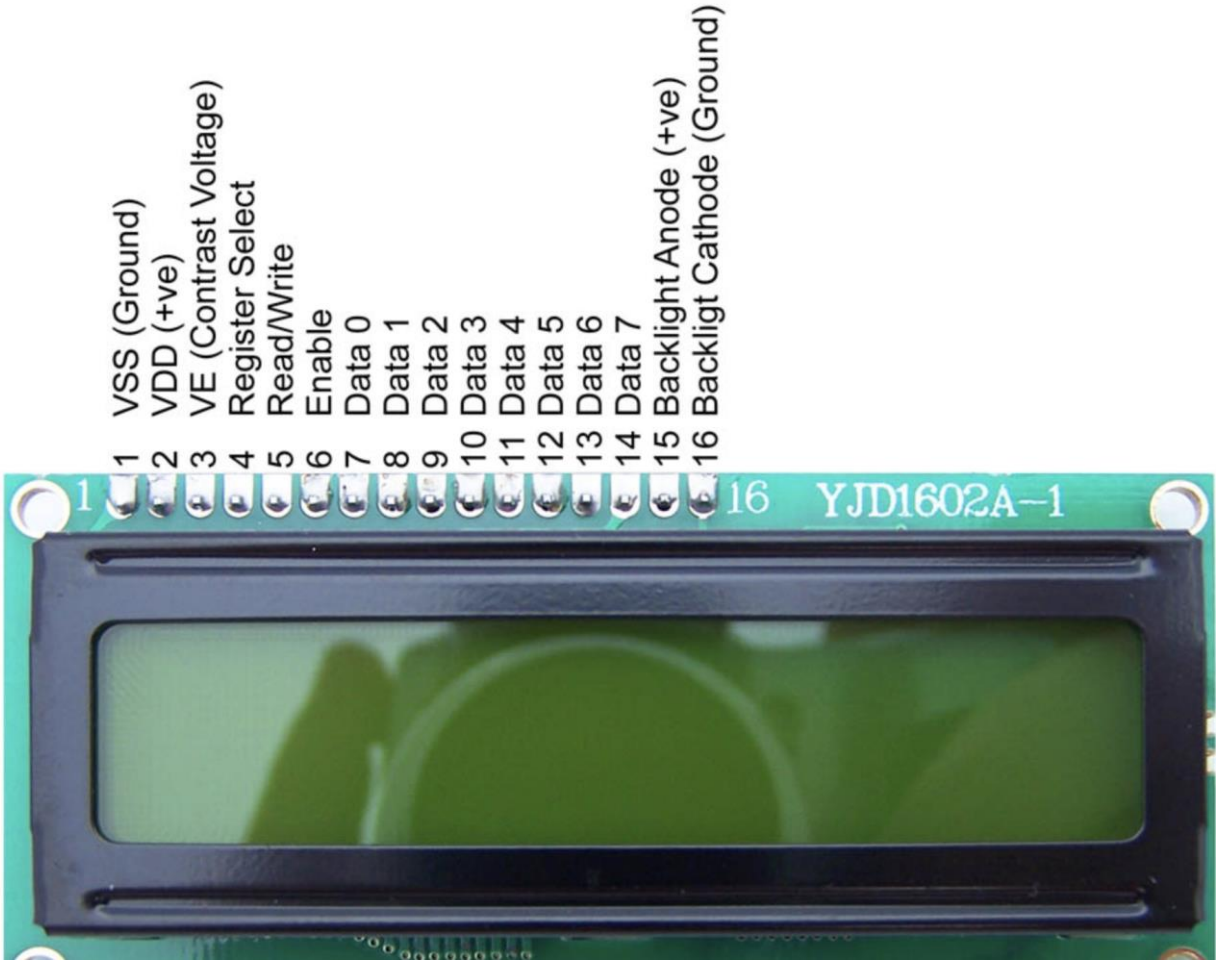

FIGURE 7. Micro-Controller of LCD display

6. ARDUINO

Arduino is an open source tool for developing computers, it usually works with sensors and control more of the physical world than a laptop. The platform is based on a simple micro controller board, and a development environment for writing software for the board.

6.1 Arduino IDE

Arduino integrated development environment contains different sections and different functions , there is a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino and Genuino hardware to upload programs and communicate with them. The file of the programs written are called sketches. These sketches are written in the text editor and saved with the file extension .ino. There is a message area when you compile your code it will give you the feedback and also displays errors. The toolbar buttons allow you to verify and upload programs, create, open , and save sketches(figure 8) and additional commands within five menus.

We can manage more than one sketch and it can be a normal Arduino code files(no visible extension), C files (.c extension), C++ files (.cpp), or header files(.h). Before uploading your sketch, you need to select the correct type of board and the working port of the Arduino.

Also you can import libraries provide extra functionality for use in sketches from Import Library menu. This will insert one or more #include statements at the top of the sketch and compile the library with your sketch. Arduino IDE has a serial monitor is a display where you can check if the sketch is working correctly.

Verify

Checks your code for errors compiling it.

Upload

Compiles your code and uploads it to the configured board. See uploading below for details.

Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"

New

Creates a new sketch.

Open

Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the **File** | **Sketchbook** menu instead.

Save

Saves your sketch.

Serial Monitor

Opens the serial monitor.

FIGURE 8. Arduino Buttons Tools

6.2 Arduino Mega 2560

The Arduino Mega 2560 (figure 9) is a micro controller board based on the ATmega2560. It has 54 digital input/output pins, where are 14 pins are used as PWM outputs, 16 analog inputs, 4 UARTs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. You only have to connect it to the computer with a USB cable or power it with a AC to DC adapter or battery to get started and the power source is selected automatically. This Arduino is compatible with the most shields designed for the Arduino Duemilanove or Diecimila. The operating voltage is 5V, the recommended input voltage is 7V to 12V, if supplied with less than 7V, however, the 5V pin may supply less than 5V and the board may be unstable and if using more than 12V, the voltage regulator may overheat and damage the board. DC current per input/output pin is 40 mA, DC current for 3.3V pin is 40 mA, a 256 KB flash memory, which 8 KB used by bootloader, a 8 KB SRAM, a 4KB EEPROM and 16 MHz clock speed.

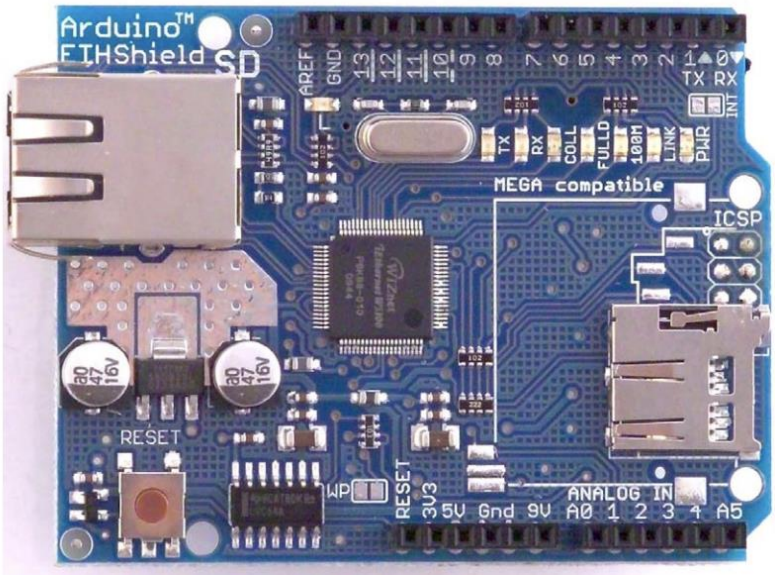

FIGURE 9. Arduino MEGA 2560

6.3 Arduino Shields

Shields are boards that can be plugged on top of the Arduino PCB extending its capabilities. The different shields (figure 10) follow the same philosophy as the toolkit, they are very easy to mount , and cheap to produce. Almost every model of Arduino is compatible with shields designed to it. Shields aren't only improve Arduino by giving it more connected sensors or circuits, they also contain code libraries made for the specific usage of the shield, the reason that we need those shield for the Arduino is because Arduino project requieres more input or output devices. Nowadays, Arduino manufactures have started to embed shields directly into Arduino circuit boards. The easy way of install and removal of those shields gives Arduino more opportunities to do different type of jobs.

FIGURE 10. Different Shields for Arduino.

7. ETHERNET SHIELD

The Arduino Ethernet shield (figure 11) allows an Arduino board to connect to the internet. It is based on the Wiznet W5100 ethernet chip, the Wiznet W5100 provides a network (IP) stack capable of both TCP and UDP. It can be connected to four socket at same time. To connect to internet using the shield has to use the ethernet library to write sketches. This keeps the ion layout intact and allows another shield to stacked on top. In the shield there is a reset controller, to ensure that the W5100 ethernet module is property reset on power up.

The communication of W5100 and SD card with the Arduino is using the SPI bus (through the ICSP header). The SPI bus is on digital pins 50, 51, and 52 on the Arduino Mega. The pin 10 is used to select the W5100 and pin 4 for the SD card. These pins cannot be used for general i/o. On the Mega, the hardware SS pin, 53, is not used to select either the W5100 or the SD card, but it must be kept as an output or the SPI interface won't work.

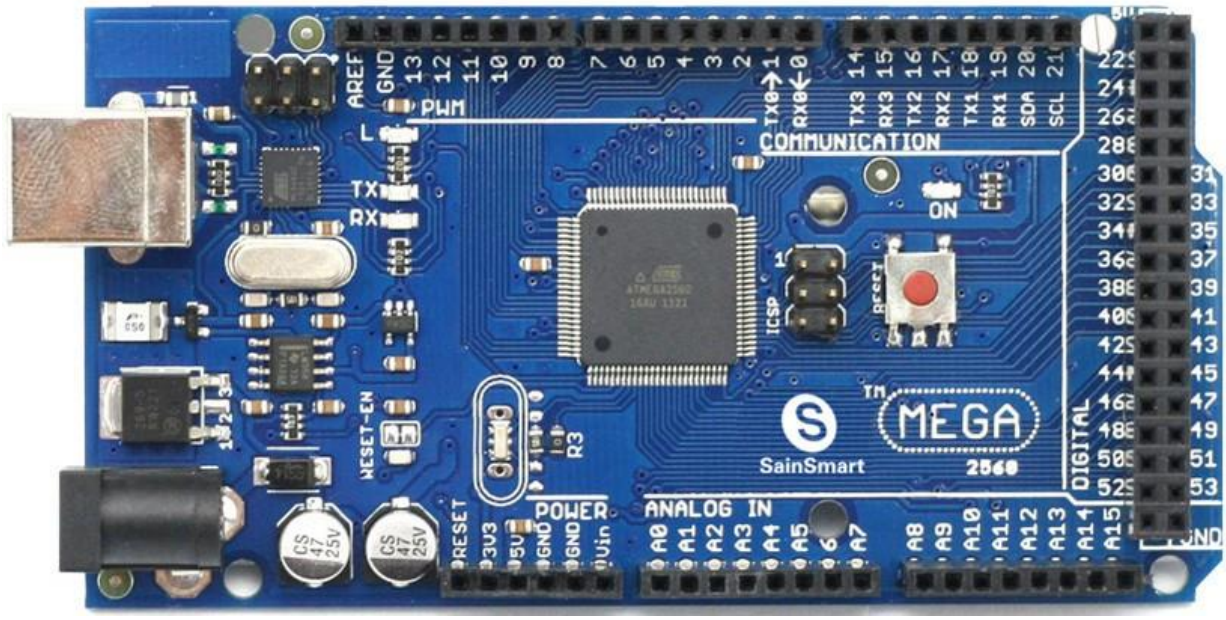

FIGURE 11. Ethernet Shield

The reset button on the shield resets both the W5100 and the Arduino board. The shield contains a number of informational LEDs:

PWR: indicates that the board and shield are powered.

LINK: indicates the presence of a network link and flashes when the shield transmits or receives data.

FULLD: indicates that the network connection is full duplex 100M: indicates the presence of a 100 Mb/s network connection (as opposed to 10 Mb/s).

RX: flashes when the shield receives data.

TX: flashes when the shield sends data.

COLL: flashes when network collisions are detected.

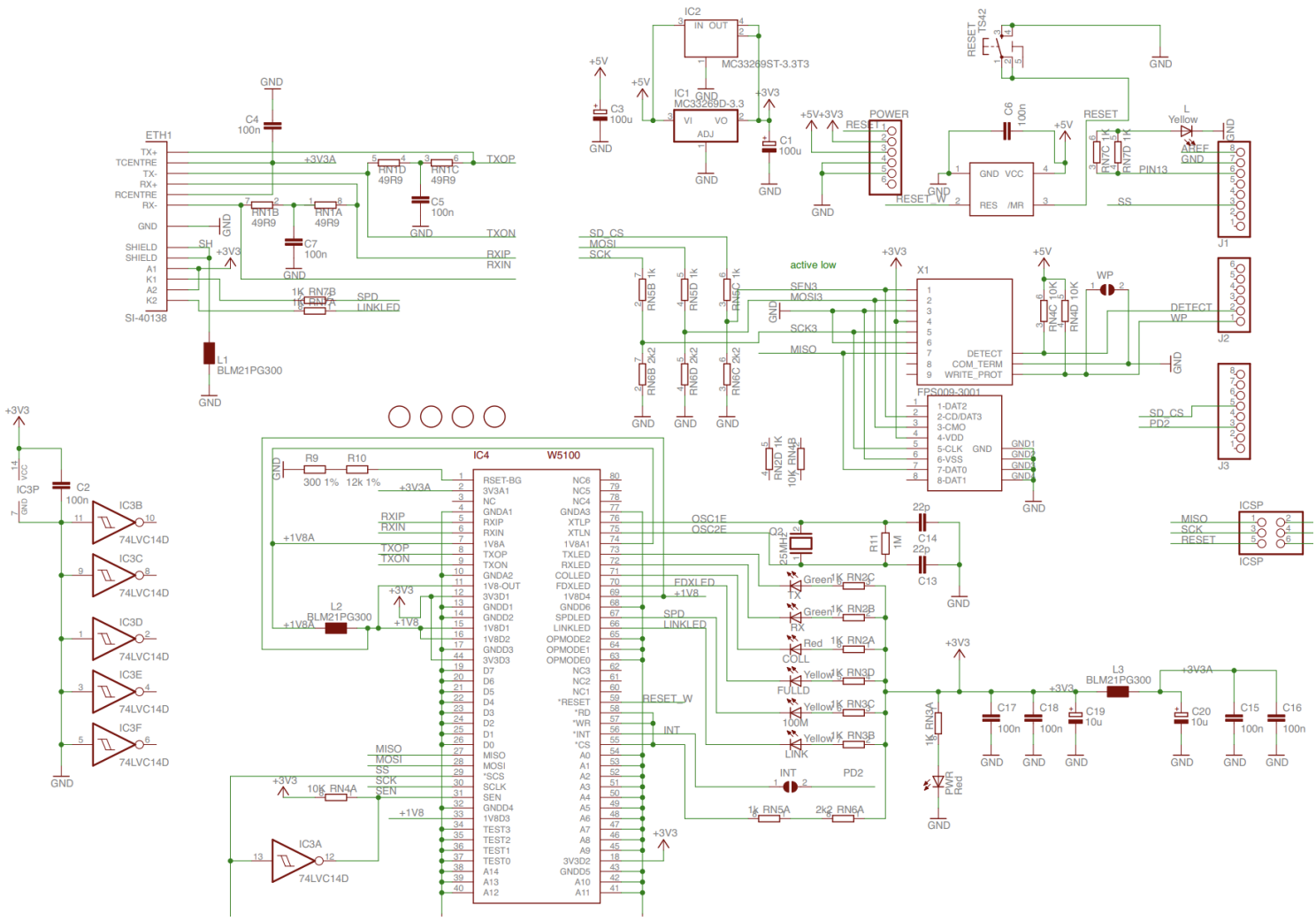

FIGURE 12. Ethernet Shield Intern Circuit

8. AUTOMATION SYSTEM

In this part will show how is the connection of all used material, the specification of the used software, how is the Arduino code work with a flowchart and some demonstration of final circuit with all implementation.

8.1 Design

In this project the system design is devised in three layers: Hardware, Software and Application layer. The hardware layer consists of electrical specifications of the design. In this layer describes how to connect the ethernet shield, the different sensors to Arduino Mega. The second layer is software, where include the programming design of the automation system and the communication, it also describes function to control the ethernet shield and technique used to read sensor data. In the application layer introduces the design of a web server, control of the mobile phone, the application we have used and the way data is transferred from software and application layer. The block diagram of the automation is in the figure 13.

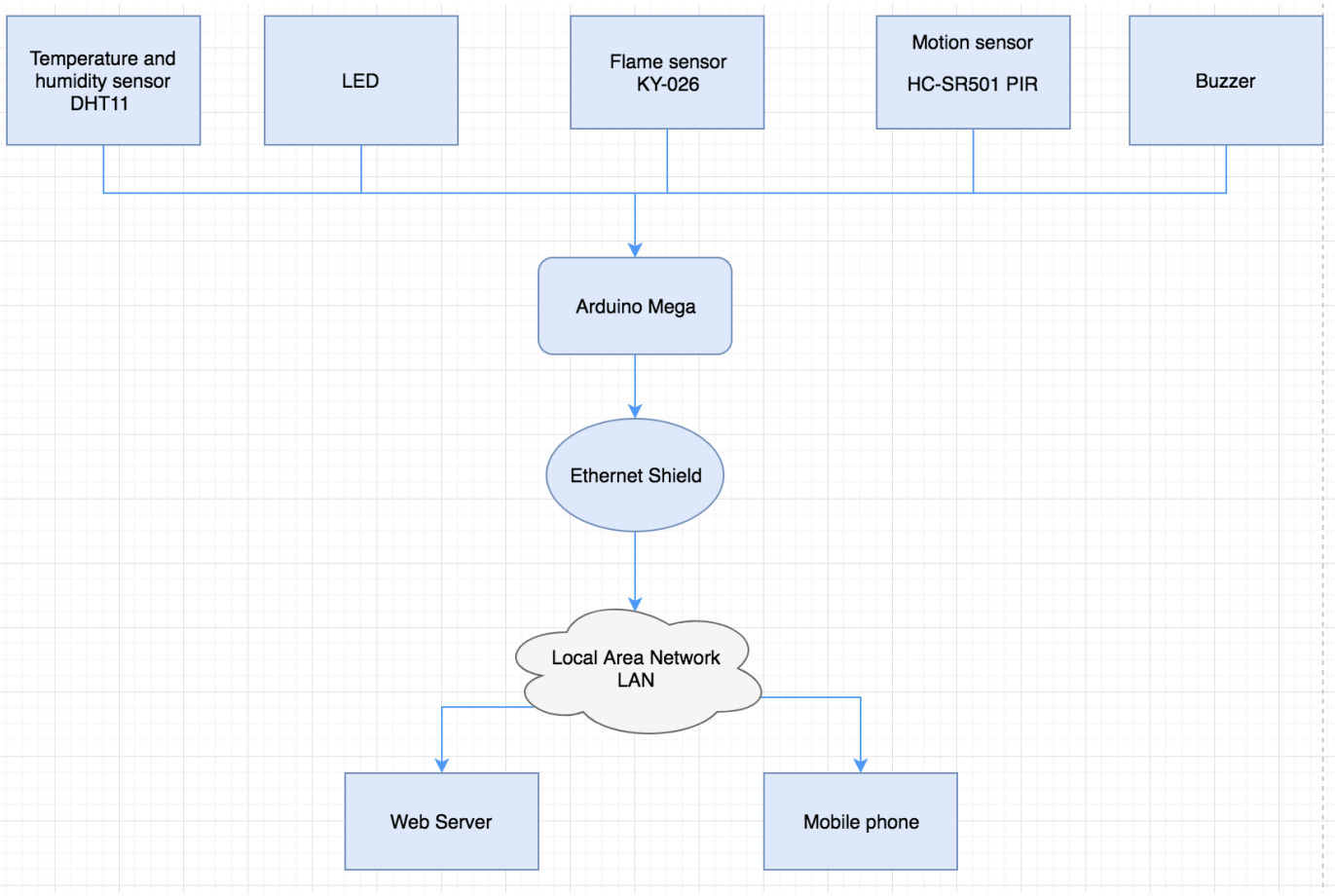

FIGURE 13. Block Diagram of Automation System

8.2 Hardware

Arduino Mega 2560 is the core of the automation system of the smart home. Mega can be connected and read multiple sensors simultaneously, it also provides enough processing power and memory to run the web server. The network shield W5100 is used for the network connectivity in this project, it is added on the top of the main board. The network shield enables web server hosting to local area network (LAN) and it has included SD card slot to store big term data. You can see in the figure 14 left network shield is visible on top of Arduino. The temperature and humidity sensor DHT11 is connected to digital pin 2 of the micro-controller.

The LCD display has 16 micro-controller, in the figure 7 you can see the name of each pin. The first pin is VSS and the last pin is backlight cathode, those have to be connected to GND. The second pin called VDD and the pin number 15 called backlight anode have to be connected to the alimentation with 5 V. The third pin is VE it is a contrast voltage so it has to be connected to a potentiometer to able to modify the contrast of the display, but there is another possibility to modify, is connecting the VE pin to Arduino PWM pin. The forth pin is a register select and it is connected to the digital pin number 28 of the Arduino, the pin 5 is connected to the digital pin number GND, the pin 6 is a enable pin connected to pin number 29 of Arduino, the pin 7 to 14 are data pin and they are connected to the digital pin 24 to 27.

The motion sensor has 3 micro-controller the alimentation pin to 5V, the ground pin to and the data pin, it is connected to pin 7 of the Arduino. The buzzer have the same number if pins like motion sensor and its data pin is connected to pin number 9 of Arduino. The flame sensor has four pins, the data pin can be analog or digital, in this project is using digital pin and its connected to the pin number 6 of Arduino.

For the led its needed a resistance of 220 Ohm to regular the alimentation tension and another pin is connected to GND. The physical circuit with all components is in the figure 14 right.

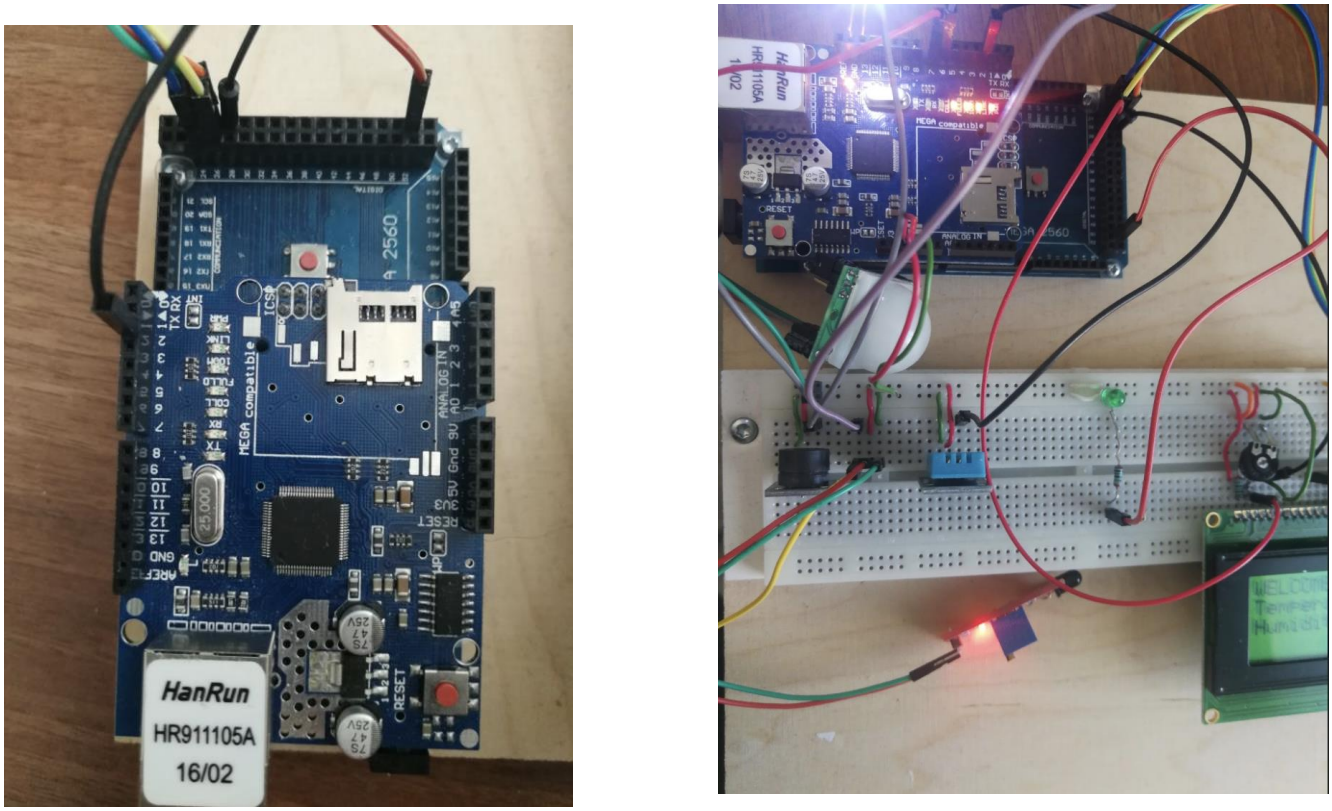

FIGURE 14. Physical Circuit with Ethernet(left) and Physical Circuit with all the Component(right).

8.3 Software

In Arduino programming there are two main functions, setup() and loop(). The setup function is only operated one time when device is booted up, usually is used to initialise the variables. Loop() is ran after setup() function, this function will be running constantly until power off or reset button is pushed. In Arduino programming, there is open-source libraries are available from Arduino community and also you can import another libraries to the Arduino environment. The flowchart of this Arduino program is in the figure 15.

The sketch of Arduino was started from the sensors, use Arduino IDE provides libraries to simplify the code of DHT11 and LCD display. All the information about sensors are read directly from digital pins. Data from temperature, humidity sensor and flame sensor are printed out to Arduino serial monitor (figure 19). To provide user remotely controls the house needs to establish a web-server. Web-server libraries are created for Arduino using HTML language to design a precisely user interface. The web-server receive Arduino´s order to modify the temperature and humidity, on the other hand, the web-server can send dato to Arduino to change the state of the light.

There are 5 different functions in the sketch, once the code runs in the Arduino IDE, after enter the loop, the first function is the flame sensor, initially is declared low level, if there are some flame detected, the flame sensor will turn to high level, then it will turn the buzzer on. The second function is the serial monitor, firstly the program will ask if it is available, if it is, so the serial monitor can be a input to change the state of the led, if it is not available then it will return to the loop. The third one is the connection of the web-server, once it is connected, the program will ask if there is client available, if it is, so the program will send information of the temperature and humidity sensor to the web-server, furthermore, we can see the state of the light and there are two switches to change the state(figure 24). The fourth function is the detection of the temperature and humidity, once it is read, it will be written on the LCD display, we can control the time of update by wait (number_milisecond), this function is a infinite loop. The last one is the motion sensor, at the beginning it is declared with low level, if there is motion detected, then it will change to high level, after that Arduino will send this information to google´s android cloud and we will receive the notification with the mobile phone with Newtifry that is connected to a google account.

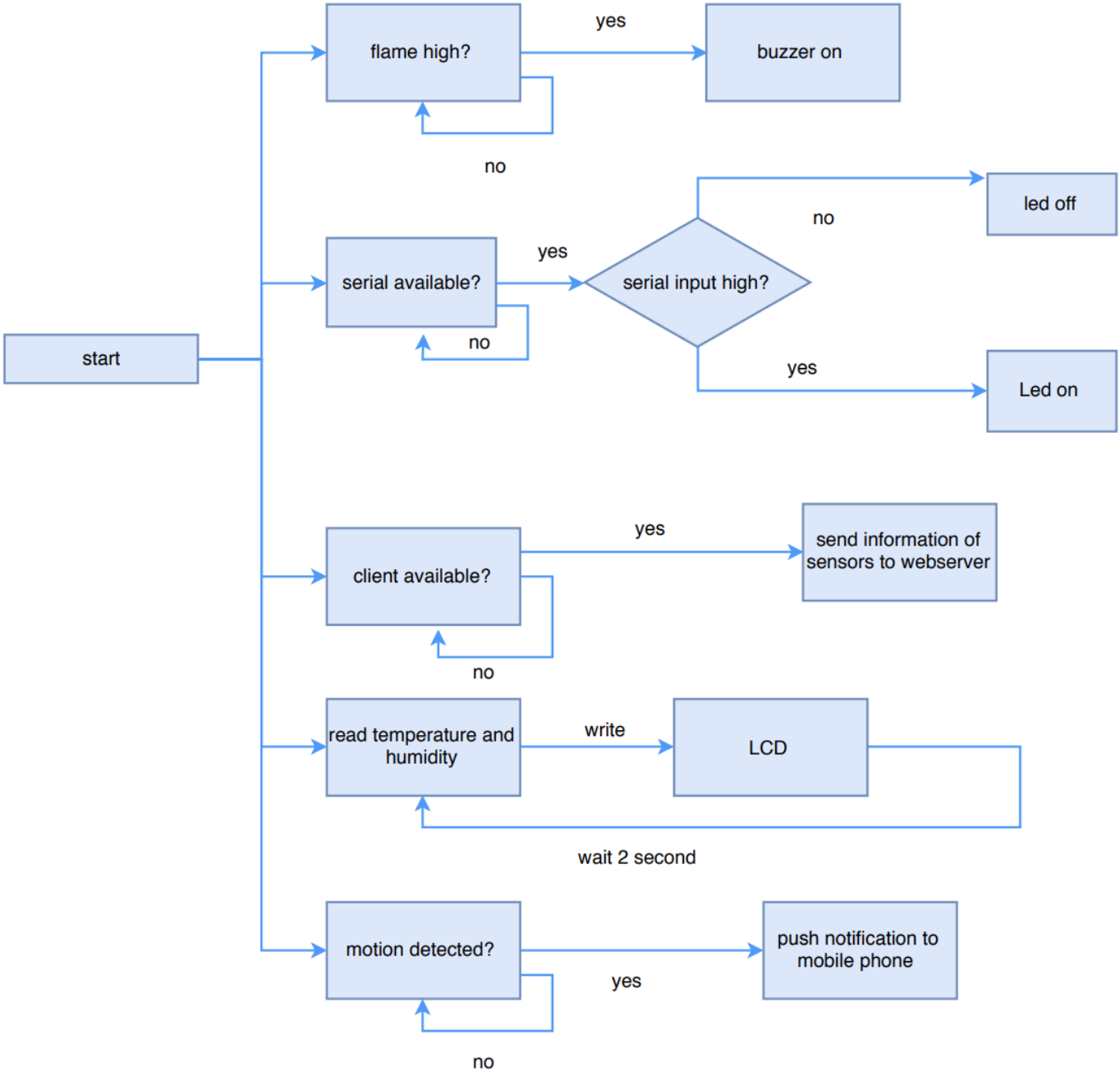

FIGURE 15. Flowchart of the Arduino Program.

In the section below there are all the applications that are used in this project:

The first application called **EvenGhost** (figure 16), it is a home automation tool that allows some technologies to work together by providing them a platform to facilitate them to communicate with each other. With EventGhost you are able to create sets (macros) of tasks (actions) to perform. In that macro you would place the notification(s) (event) that you want to run that macro that contains the actions you wish to perform. The events that are displayed (triggered) can be generated by anything that has the ability to communicate. One of the features about EventGhost is that it is extensible, and has over 300 unique extensions (plugins) available. Some examples of devices that EventGhost is able to talk to are Yamaha, Denon/Marantz, Pioneer, Sony, Sonos, Samsung, RTI, Amazon Echo, MicasaVerde Vera, Phillips Hue, and many others. Supported automation protocols include but not limited to MQTT, IFTTT, TCP/IP, Serial (RS-232), CIR (IR remotes), X10, xAP, xPL. EventGhost can even automate tasks on the computer it is installed on. EventGhost is simple tool program for nowadays its only available for Microsoft Windows to personalise or automate. It normally used with different input devices like infrared, USB or wireless remote controls to run macros, that on their part control a computer and its attached hardware. For that reason, it can be used to control a computer with a normal consumer remote.

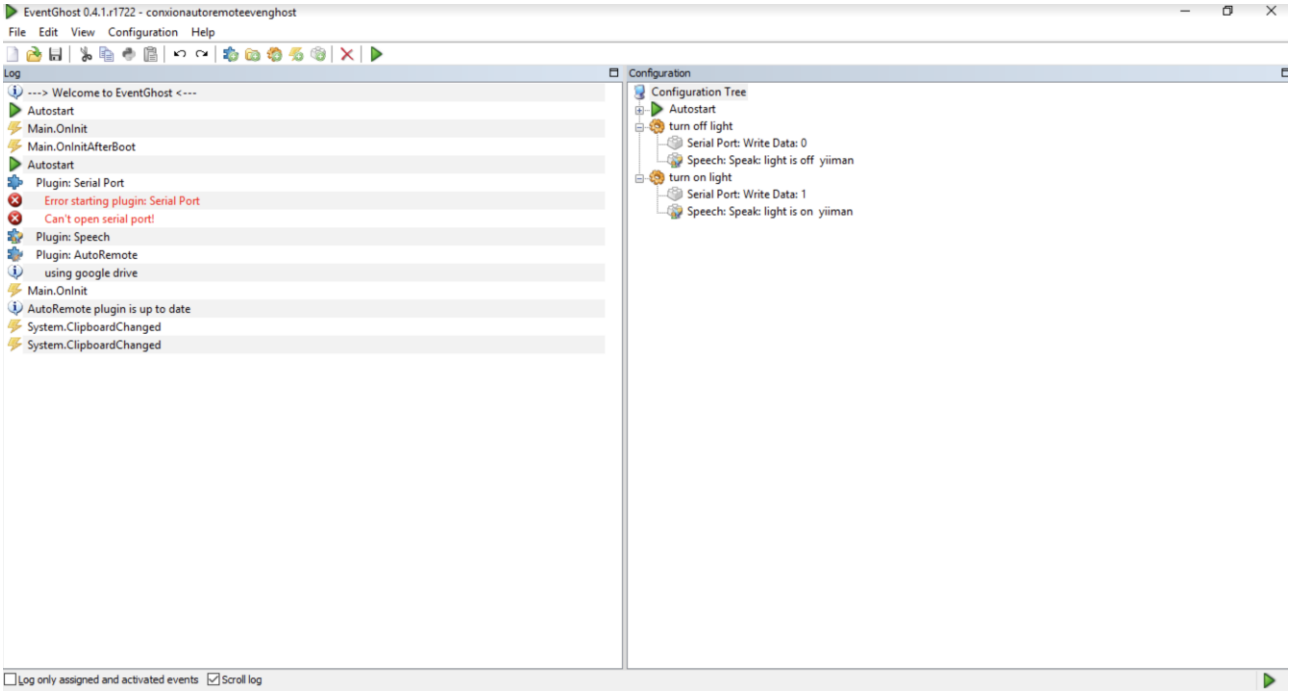

FIGURE 16. Evenghost Program.

The second application is called **Newtifry** (figure 17 right), is an Android program and a server backend to deliver notifications to Android phones using the Google Cloud Messenger (GCM) APIs. This is designed to replace services like SMS messages, which would normally be used to send server notifications. The application delates the received messages periodically to free the space of the mobile phone. When you receives the notification the mobile phone will read for you the message that you have received.

The next one is **Autoremote** (figure 17 left), is a powerful app allows a communication system designed for the devices to communicate, you can control one of your devices from another device, And with a Tasker plugin to send messages via AutoRemote, for the connection is necessary have a google account, it will provide you a personal URL. This URL is both used for registering your device with other devices, and for accessing AutoRemote's web access. Opening the URL in a browser will present you with a page where you can send messages to your device, as well as instructions for accessing AutoRemote's second personal code, the key, which is used for some parts of the AutoRemote system. After you access to the application, there is a menu to get into the list of registered devices. Here you can register a new device by using the personal URL of that device, and this way connect devices together. You'll have to do this on both devices in order for both of them to be able to send messages to the other. Any devices registered in this list will be available as an option when you go to send a message.

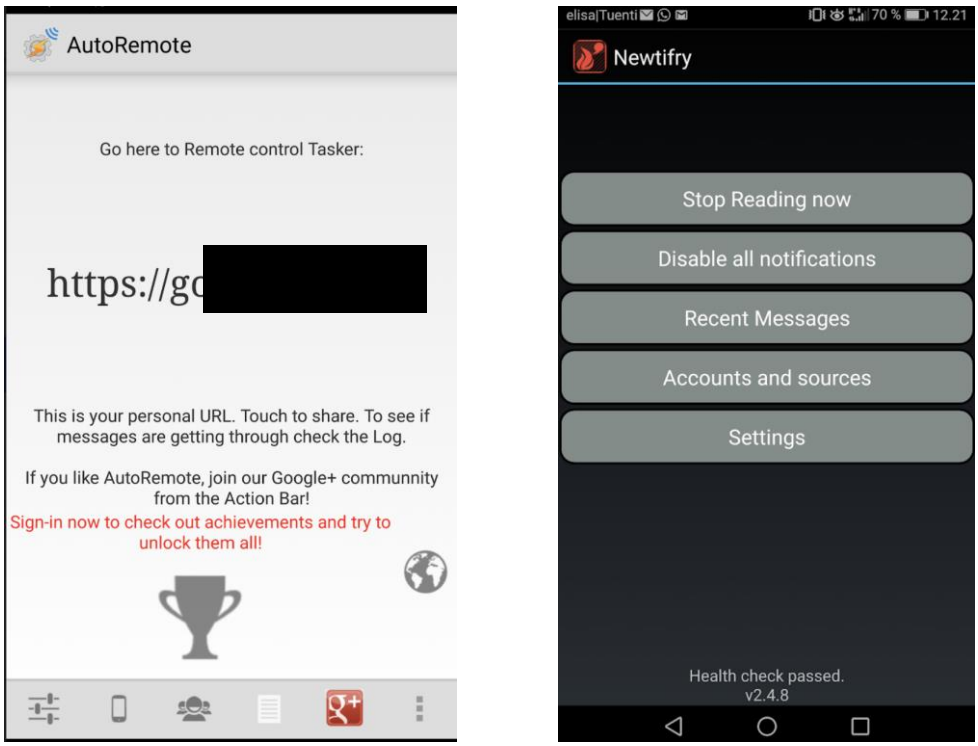

FIGURE 17. Autoremote Application in Android Mobile Phone(left) and Newtifry Application (right).

The last application used in this project is **Tasker** (figure 18), is a paid automation application for Android that lets you trigger certain actions to be run if and only if certain conditions are met, performs tasks based on contexts like gestures, event, time.The idea is simple, assign an event to an action. It can be manual or automatic, Tasker monitors the phone for the contexts and performs tasks based on the contexts. The conditions are stored in profiles, profiles are connected to tasks that you want to run in response to any conditions you have chosen. One task can be formed with a group of actions, all the actions will run one after the others when the task is triggered. Furthermore, once a profile has been made, you can active or desactive it at any time without affecting any other profiles.

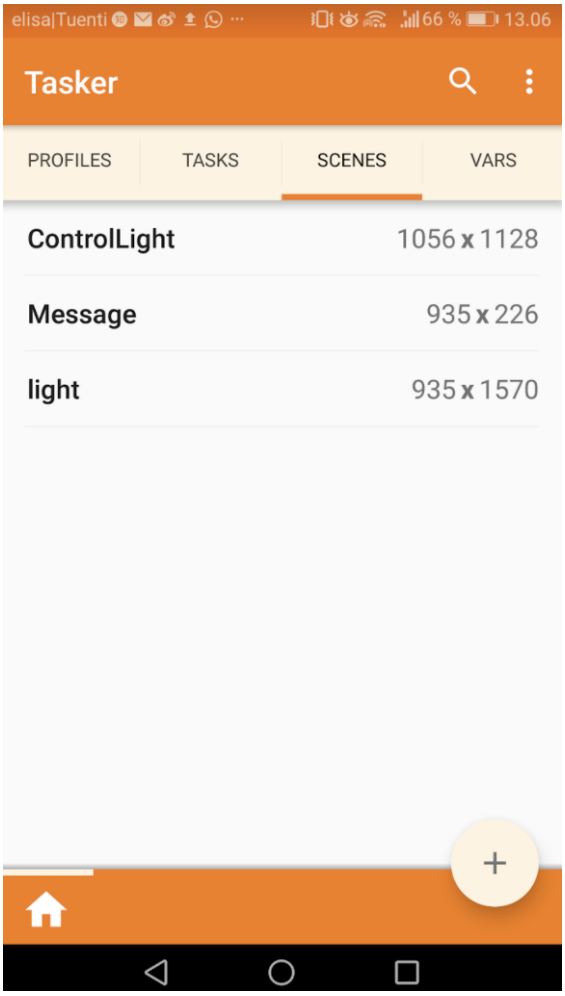

FIGURE 18. Tasker Application.

8.4 User Interface

Arduino provides tools to assist the implementation of the automation system. Arduino IDE run the code and enables serial communication with Arduino micro controller through USB cable. The serial monitor (figure 19) is used to print some function of the code, it helps to see clearly the events between the hardware and software layer. On the other hand, Arduino open source libraries, which are used to simplify sensor data fetching and network connectivity, provide the implementation of the sensors and network shield on the Arduino Mega.

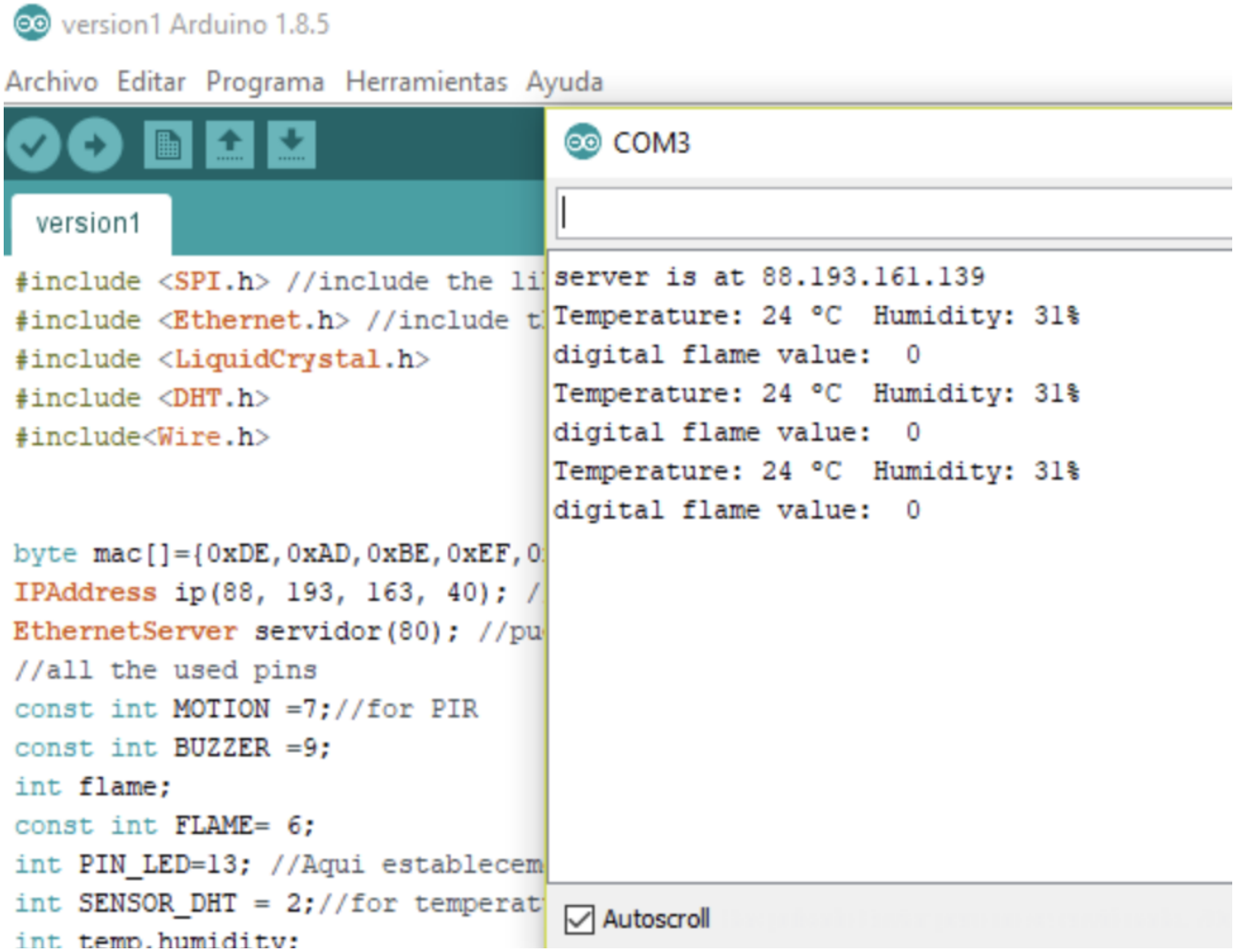

FIGURE 19. Arduino Serial Monitor.

For our light system, it can be controlled by two different ways, with the web server and mobile phone, with the mobile phone can be controlled manually or with the voice.
The control of mobile phone needs three tools, the first one is Evenghost, add the serial port that has rung with Arduino IDE as a plugin, then add two actions for serial port as write data (figure 20), one for turn light on and another one to turn light off.

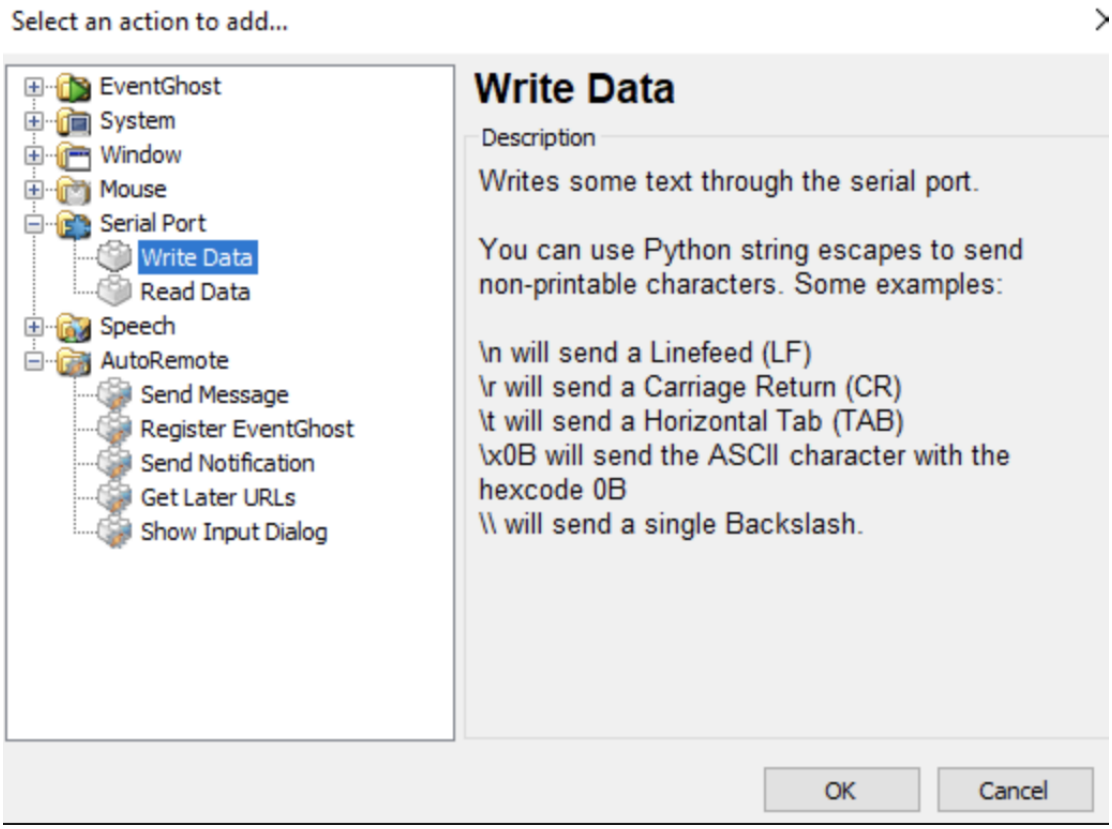

FIGURE 20. Write Data Serial Port Window in Evenghost.

For the control of the light with the mobile phone and voice, we need another two applications called Autoremote and Tasker, Autoremote as a middleware to connect Evenghost with Tasker, Autoremote will give us a personal URL (figure 21), this URL is used for the connection of Evenghost and Autoremote.

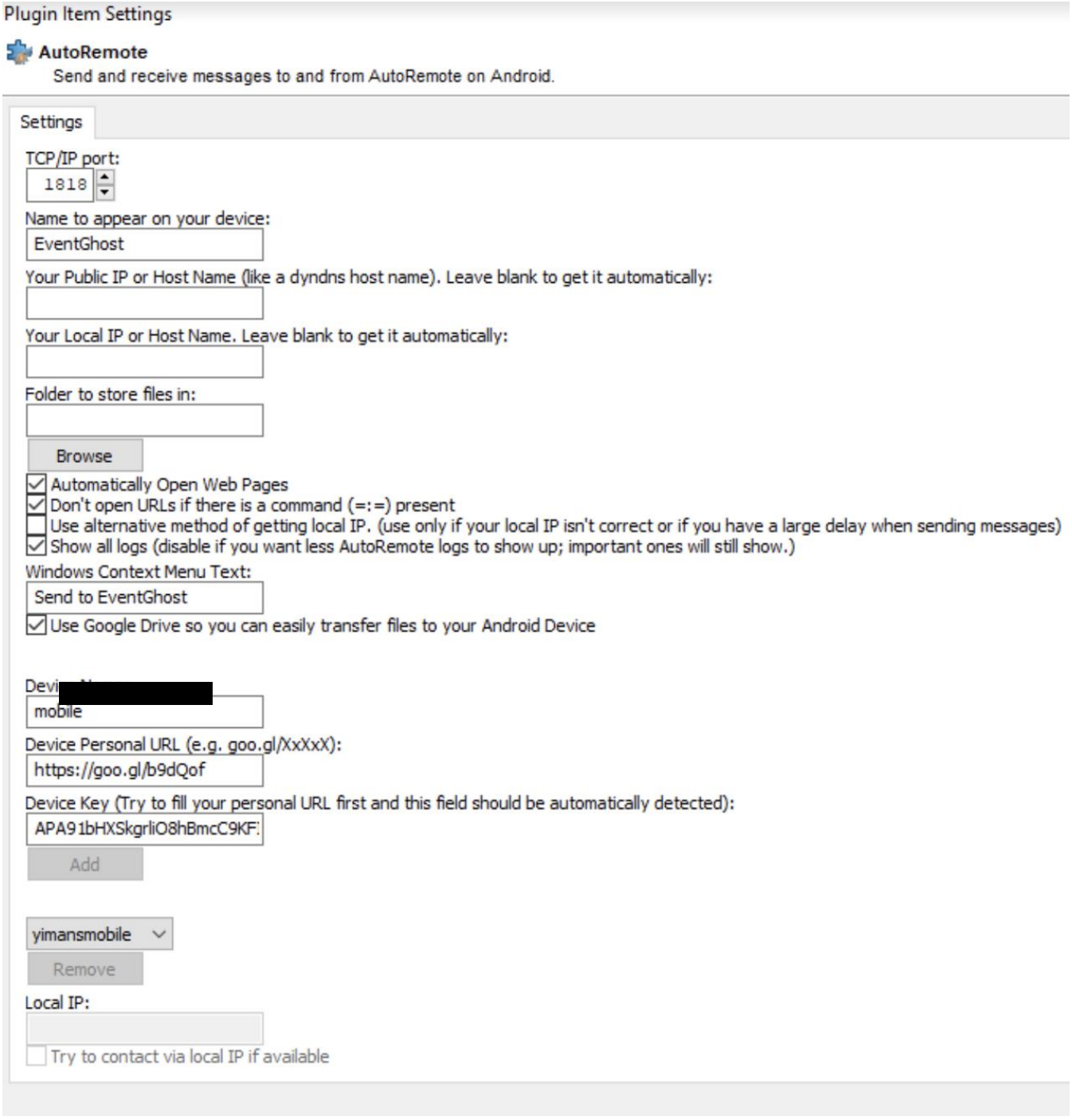

FIGURE 21. Autoremote Plugin Window with Personal URL.

If the connection is successful , in Autoremote should apear the figure 22 and able to send messages to Evenghost.

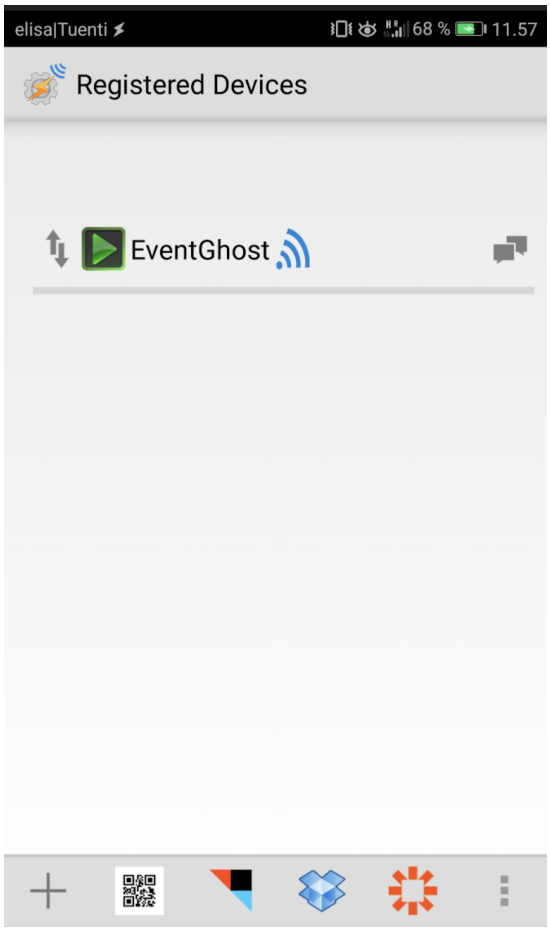

FIGURE 22. Connection of Evenghost and Autoremote.

Use the application Tasker to create three switches, used for the voice control, manually control and another to close the application(figure 23) also setup an AutoRemote profile in Tasker to react to the message after that its necessary to establish plugin with Autoremote to able to control the light with such actions.

FIGURE 23. The Application Tasker.

The web server (figure 24) control is using LAN, and it can turn on the light or turn it off, it also shows the state of the light, the temperature and humidity of the home.

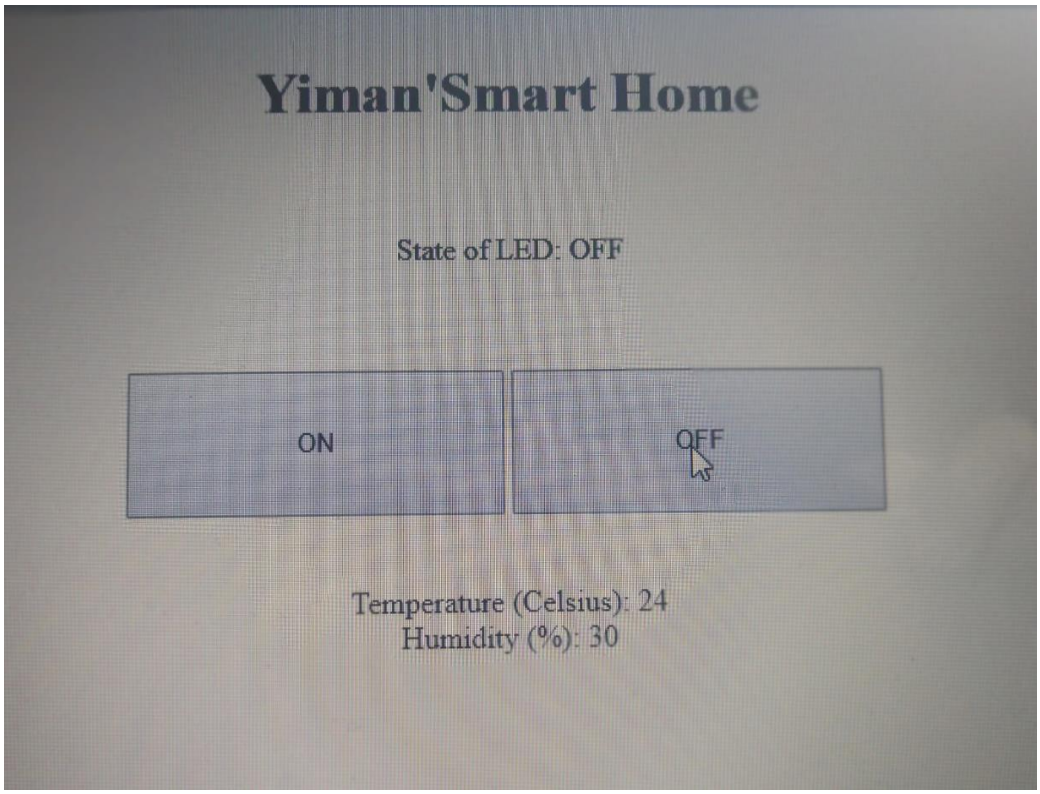

FIGURE 24. Webserver

The temperature and humidity of the house are also showing with LCD display, it is used for short distance and it shows the messages below in the figure 25.

FIGURE 25. LCD Display with Received Messages.

The push notifications system is implemented with the motion sensor, Autoremote and Newtifry, we have to download Newtifry in the mobile phone and log in with a google account, Autoremote is a middleware to connect the Arduino and Newtifry, if the motion sensor detect some motion will notify to Arduino, and after that, Arduino will send a notification to Newtifry, the notification will be read and showed in the mobile phone(figure 26).

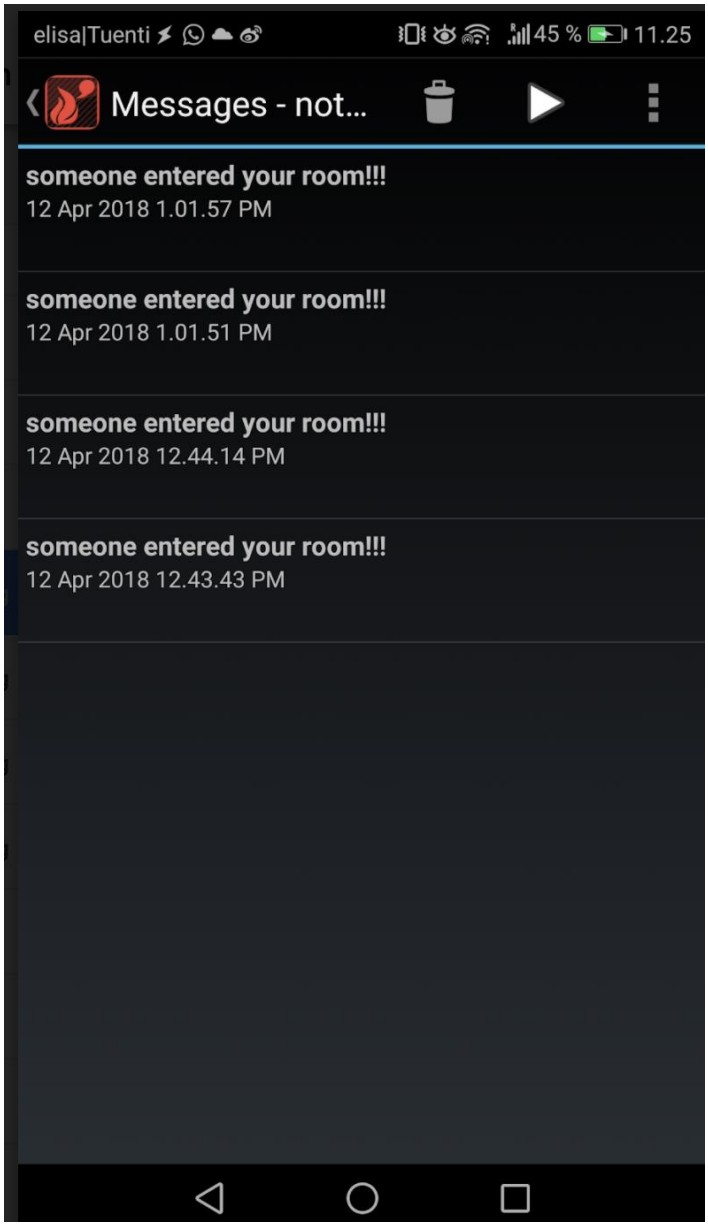

FIGURE 26. Newtifry with Received Messages.

9. BUDGET

The budget of the project of all used materials are in the table 8, the prices can be variable with different pages.:

Electrical material	price (€)
Arduino Mega 2560	11.99
Motion sensor	2.43
Flame sensor	6.73
Temperature and humidity sensor	1.46
LED	0.42
Potentiometer	0.22
Ethernet Shield	11.39
Buzzer	1.12
Cables	0.50
Total:	36.26

TABLA 8. Budget of all Needed Materials

10. CONCLUSION

The home automation system has implemented successfully, this project discussed the designed modules like sensors circuits, mobile phone push notifications and web server. A several experiments have been carried out on the proposed automation system. These experiments show how to detect any intruder to the home sending a push notification to the tenant, detect the fire and the ambient weather of the room. In addition, helps people with reduced movement to be more independent and able to control their environment themselves. It also helps to reducing the power consumption because in the web server shows all the information of room's light is turned on with mobile phone and turn it off easily.

To improve this implemented system, it could be added a watering system for the garden, first the humidity sensor detect the Soil moisture, there is a established percentage number, if it is lower than that, so the system will watering the garden. A voice control for all system, it could simplify the way of control system. Furthermore, for a better security of the house, there are another sensor and devices that can be used, for example, a monitoring camera. Moreover, you can replace the ethernet network connection with WIFI technology, Bluetooth or GSM module to make this system more professional, the connection won't need an ethernet cable to connect to the wire. On the other hand, there is the choice to do a smart garage that measure the length of the car and put it in the most suitable block to make the parking easy for the tenant in his garage.

There is no doubt that the automation system will make our life easier, this field is continuing to evolve, the number of customers are increasing and the manufacturers are creating and innovating products to better suit customers needs. However, there still has some serious hurdles to overcome before it reaches ubiquity.

LIST OF REFERENCES

[1] S. A. Ram, N. Siddarth, N. Manjula, K. Rogan and K. Srinivasan, “Real-time automation system using Arduino,”2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS), Coimbatore, India, 2017, pp. 1-5

[2] S. Gunputh, A. P. Murdan and V. Oree, “Design and implementation of a low-cost Arduino-based smart home system,”2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN), Guangzhou, 2017, pp. 1491-1495.

[3] Official page of Arduino, https://www.arduino.cc/en/guide/introduction?setlang=en, Consultado el 2018-02- 05.

[4] Ethernet W5100, http://www.hobbytronics.co.uk/datasheets/W5100_Datasheet_v1_1_6.pdf

[5] Exercises of different level to start in electronics with Arduino, https://www.drouiz.com/tutoriales-arduino- 2/

[6] A. Adriansyah and A. W. Dani, “Design of Small Smart Home system based on Arduino,” 2014 Electrical Power, Electronics, Communications, Control and Informatics Seminar (EECCIS), Malang, 2014, pp. 121-125.

[7] Noble, Joshua, Programming Interactivity: A Designer’s Guide to Processing, Arduino, and openFramework (1ª edición). O’Reilly Media. p. 768.

[8] Description of product HC-SR501 PIR, https://www.mpja.com/download/31227sc.pdf.

[9] Manual de DHT, Temperature and Humidity modele, https://akizukidenshi.com/download/ds/aosong/DHT11.pdf.

[10] Manual of flame sensor module, http://www.fut-electronics.com/wp- content /plugins/fe_downloads/Uploads/Flame-sensor- arduino.pdf.

[11] John Nussey, 2013 “Arduino for Dummies”, Chichester,England, John Wiley & Sons.

[12] Price of KY-012 buzzer in amazon, https://www.amazon.com/s/ref=nb_sb_ss_fb_1_10?url=search-alias%3Daps&field-keywords=KY-012+buzzer+&rh=i%3Aaps%2Ck%3AKY-012+buzzer+ ,[consulted 3 march 2018]

[13] Price of potentiometer in amazon, https://www.amazon.com/DealMux-a13120900ux0101-individual-potenci%C3%B4metro-resistor/dp/B078M2R4B4/ref=sr_1_69?s=industrial&ie=UTF8&qid=1525167117&sr=1-69&keywords=potenciometro [consulted 3 March 2018]

[14] Price of Arduino Mega 2560 in Amazon, https://www.amazon.es/Tarjeta-Microcontrolador-ATmega2560-ATmega16U2-Compatible/dp/B06Y3ZHPWC/ref=sr_1_1_sspa?ie=UTF8&qid=1525167218&sr=8-1-spons&keywords=arduino+mega+2560&psc=1 [consulted 9 March 2018]

[15] Price of HC-SR501 PIR motion sensor in amazon, https://www.amazon.es/Neuftech-Sensor-Motion-Infrarrojos-Piroelectricidad/dp/B01C6O9C38/ref=sr_1_3?ie=UTF8&qid=1525167385&sr=8-3&keywords=motion+sensor [consulted 9 March 2018]

[16] Price of ethernet shield W5100 in amazon, https://www.amazon.es/Demarkt-Arduino-Ethernet-Shield-W5100/dp/B01HMIU9VM/ref=sr_1_1?ie=UTF8&qid=1525167485&sr=8-1&keywords=ethernet+shield+arduino [consulted 25 March 2018]

[17] Price of LED sodial(R) 75 x 3mm in amazon, https://www.amazon.es/SODIAL-Emisores-Amarillo-Colores-Surtidos/dp/B00E34MNYU/ref=sr_1_1?ie=UTF8&qid=1525167553&sr=8-1&keywords=led+arduino [consulted 27 March 2018]

[18] Price of KY-026 flame sensor module in amazon, https://www.amazon.com/flame-sensor-module-ARDUINO-KY-026/dp/B0100AAKT0 [consulted 27 March 2018]

[19] Arduino Mega 2560 datasheet, https://www.robotshop.com/media/files/pdf/arduinomega2560datasheet.pdf [consulted 2018]

[20] Newtifry official page, https://newtifry.appspot.com/ [consulted 1 May 2018]

[21] Arduino integrated development environment guide, https://www.arduino.cc/en/Guide/Environment [consulted 20 April 2018]

[22] LED parameters, https://www.sparkfun.com/datasheets/Components/LED/COM-09590-YSL-R531R3D-D2.pdf [consulted 1 April 2018]

[23] Details information related to Led, https://cdn-shop.adafruit.com/datasheets/WP7113SRD-D.pdf [consulted 22 April 2018]

[24] Code example of KY-026 in Arduino IDE, http://sensorkit.en.joy-it.net/index.php?title=KY-026_Flame-sensor_module [consulted 13 March 2018]

[25] RJ45 connector with integrated magnetics datasheet, http://pdf-file.ic37.com/uploadpdf_old/icpdf_datasheet_9/HR911105A_datasheet_900074/146705/HR911105A_datasheet.pdf [consulted 1 May 2018]

[26] HD44780U LCD display datasheet, http://fab.cba.mit.edu/classes/863.06/11.13/44780.pdf [consulted 25 April 2018]

[27] Evenghost home page, http://www.eventghost.net/ [consulted 25 April 2018]

APPENDICES

Appendix 1. Arduino Automation System Code

```
#include <SPI.h> //include the library SPI
#include <Ethernet.h> //include the library Ethernet
#include <LiquidCrystal.h>
#include <DHT.h>
#include<Wire.h>


byte mac[]={0xDE,0xAD,0xBE,0xEF,0xFE,0xED}; // MAC
IPAddress ip(88, 193, 163, 40); //my IP address
EthernetServer servidor(80); //puert 80
//all the used pins
const int MOTION =7;//for PIR
const int BUZZER =9;
int flame;
const int FLAME= 6;
int PIN_LED=13; //Aqui establecemos la variable PIN_LED como un valor entero
int SENSOR_DHT = 2;//for temperature and humiduty sensor
int temp,humidity;
int LED=53;


//pin digital
/*LCD pins information:
*/
int RS = 28;//WRITE TO SCATE
//pin enable
int E = 29;
//ports show the information
int D4 = 24;
int D5 = 25;
int D6 = 26;
int D7 = 27;

String readString=String(20); // read characters of a secuence of a string
//string is represented like a array of characters
String state=String(3);

//the temperature sensor and LCD
DHT dht(SENSOR_DHT,DHT11);
LiquidCrystal lcd (RS, E, D4, D5, D6, D7);
```

```
void setup() {
  //connection
  Serial.begin(9600);
  Ethernet.begin(mac); //start the directions.
  servidor.begin(); //start the server
  Serial.print("server is at ");
  Serial.println(Ethernet.localIP());

  //1.my dht11 sensor and lcd begin to work,size of my lcd and lcd begin to work
  dht.begin();
  lcd.begin(16,4);
  //2.flame detector and buzzer
  pinMode(BUZZER,OUTPUT);
  pinMode(FLAME,INPUT);
  //3.Motion sensor as a inputmode
  pinMode(MOTION,INPUT);
  pinMode(LED,OUTPUT);
  //4.light
  pinMode(PIN_LED,OUTPUT);//declare PIN_LED as a output mode
  digitalWrite(PIN_LED,LOW);//start with LED in state off
  state="OFF";
}
```

```
void loop() {
      //EthernetClient create a client who can connect to the specific IP.
      EthernetClient client= servidor.available();

      //read humidity and temperture using library of DHT.................
       humidity = dht.readHumidity();
       temp = dht.readTemperature();
      //start working sensors.............................................
      digitalWrite(LED, digitalRead(MOTION));
      writeLed();
      writeLCD();
      readFlame();
```

```
      if(client) {
        boolean emptyline=true;
        while(client.connected()) {
          if(client.available()) {
            char c=client.read(); //read word by word what client(petition of HTML) say
            if(readString.length()<20) {
              readString.concat(c);//write word by word to a string named readString

            }
            if(c=='\n' && emptyline) //if the http petition is finish
              {
              int LED = readString.indexOf("LED=");
              if(readString.substring(LED,LED+5)=="LED=T") {
                  digitalWrite(PIN_LED,HIGH);
                  state="ON"; }
              else if (readString.substring(LED,LED+5)=="LED=F") {
                  digitalWrite(PIN_LED,LOW);
                  state="OFF";
              }
```

```
 //send to the client a HTTP answer
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println("Refresh: 120");
client.println(); //Página Web en HTML

//Standar head in HTTP format
client.println("<html>");
client.println("<head>");
client.println("<title>yiman_smart_home</title>");
client.println("</head>");
client.println("<body width=100% height=100%>");
client.println("<center>");
client.println("<img scr='imagenes/smarthome.png'>");
client.println("<h1>Yiman'Smart Home</h1>");
client.print("<br><br>");
client.print("State of LED: ");
client.print(state);
client.print("<br><br><br><br>");
client.println("<input type=submit value=ON style=width:200px;height:75px onClick=location.href='./?LED=T\'>");
client.println("<input type=submit value=OFF style=width:200px;height:75px onClick=location.href='./?LED=F\'>");
```

```
    //display the temperature and humidity of the home
    client.println("<br /><br /><br />");
    client.print("Temperature (Celsius): ");
    client.println(temp);
    client.println("<br />");
    client.print("Humidity (%): ");
    client.println(humidity);
    client.println("<br />");

    client.println("</center>");
    client.println("</body>");

    //Cierro conexión con el cliente
    client.stop();
    readString="";
    }
   }
  }
}
void writeLed(){
  if (Serial.available()>0)
  {
    if (Serial.read()=='1')
    {digitalWrite(PIN_LED,HIGH);
    }
    else{
      digitalWrite(PIN_LED,LOW);
      }
    }
}

void readFlame(){
  flame=digitalRead(FLAME);
     Serial.print("digital flame value:  ");
    Serial.println(flame);

   while(flame>0){
      digitalWrite(BUZZER,200);
      delay(100);
      digitalWrite(BUZZER,0);
      delay(100);
      }
  }
```

```
void writeLCD(){
     //(colum = 0,fila = 0)
      lcd.clear();
      lcd.setCursor(0,0);
      lcd.print("WELCOME YIMAN! ");
      lcd.setCursor(0,1);
      lcd.print("Temperature: ");
      lcd.print(temp);
      lcd.print(" °C");

      lcd.setCursor(4,2);
      lcd.print("Humidity: ");
      lcd.print(humidity);
      lcd.print("%");

      Serial.print("Temperature: ");
      Serial.print(temp);
      Serial.print(" °C  Humidity: ");
      Serial.print(humidity);
      Serial.println("%");
      delay(2000);
 }
```

Appendix 2. Arduino Push Notification Code

```
#include <SPI.h>
#include <Ethernet.h>

  //////////////////////////
 // Yiman´s Notification //
//////////////////////////
byte mac[] = { 0x00, 0xAA, 0xBB, 0xCC, 0xDE, 0x19 };   // my Mac


char DEVID1[] = "v56E9FEB73934498";        //Unique number given by pushing box


//Numeric Pin where you connect your switch
uint8_t motionSensor = 7; // switch is connect to the Pin 7

// Debug mode
boolean DEBUG = true;


char serverName[] = "api.pushingbox.com";
boolean motionSensorState = false;               // Save the last state of the Pin for DEVID1
boolean lastConnected = false;                  // State of the connection last time through the main loop
```

```
  // Initialize the Ethernet client library
  // with the IP address and port of the server
  // that you want to connect to (port 80 is default for HTTP):
  EthernetClient client;

  void setup() {
    Serial.begin(9600);
    pinMode(motionSensor, INPUT);

    // start the Ethernet connection:
    if (Ethernet.begin(mac) == 0) {
      Serial.println("Failed to configure Ethernet using DHCP");
      // no point in carrying on, so do nothing forevermore:
      while(true);
    }
    else{
      Serial.println("Ethernet ready");
      // print the Ethernet board/shield's IP address:
      Serial.print("My IP address: ");
      Serial.println(Ethernet.localIP());
    }
    // give the Ethernet shield a second to initialize:
    delay(1000);
  }
```

```
void loop()
{
      ////
      // Listening for the motionSensor state
      ////
      if (digitalRead(motionSensor) == HIGH && motionSensorState == false) // motion detected
      {
        if(DEBUG){Serial.println("motionSensor is HIGH");}
        motionSensorState = true;
        //Sending request to PushingBox when the pin is HIGH
        sendToPushingBox(DEVID1);
      }
       if (digitalRead(motionSensor) == LOW && motionSensorState == true) // no motion detected
      {
        if(DEBUG){Serial.println("motionSensor is LOW");}
        motionSensorState = false;
        //Sending request to PushingBox when the pin is LOW

      }




        //DEBUG part
        // this write the respons from PushingBox Server.

        if (client.available()) {
          char c = client.read();
          if(DEBUG){Serial.print(c);}
        }

        // if there's no net connection, but there was one last time
        // through the loop, then stop the client:
        if (!client.connected() && lastConnected) {
          if(DEBUG){Serial.println();}
          if(DEBUG){Serial.println("disconnecting.");}
          client.stop();
        }
        lastConnected = client.connected();
}
```

```
//Function for sending the request to PushingBox
void sendToPushingBox(char devid[]){
  client.stop();
  if(DEBUG){Serial.println("connecting...");}

  if (client.connect(serverName, 80)) {
    if(DEBUG){Serial.println("connected");}

    if(DEBUG){Serial.println("sendind request");}
    client.print("GET /pushingbox?devid=");
    client.print(devid);
    client.println(" HTTP/1.1");
    client.print("Host: ");
    client.println(serverName);
    client.println("User-Agent: Arduino");
    client.println();
  }
  else {
    if(DEBUG){Serial.println("connection failed");}
  }
}
```