Grado Universitario en Ingeniería en Tecnologías Industriales
Curso 2018-2019

Trabajo Fin de Grado

“Control domótico de dispositivos del hogar a través de una aplicación móvil”

Felix Görtz
Tutor
David Griol Barres

Leganés, septiembre de 2019

[Incluir en el caso del interés de su publicación en el archivo abierto]
Esta obra se encuentra sujeta a la licencia Creative Commons Reconocimiento – No Comercial – Sin Obra Derivada
SUMMARY

Domotics is a science dedicated to the control and automation of the household. It implies the installation of an electronic system in the household to improve comfort, life quality, security and efficiency.

The project’s objective is the design, development and implementation of a domotic system. The general idea is to use a phone application to wirelessly control or automate different electronic components or systems belonging to a household. A component is referred to as an electro-domestic device that will be connected via the digital inputs or outputs of the microcontroller. For example, lights, and alarm system, a thermostat or heating system, a watering system and so on.

The phone application will communicate the user’s instructions to the microcontroller, through a graphical interface, which will manage the electronic components or systems that the user wants to automate or control.

The communication consists of an initial data transfer via a Bluetooth connection followed by an internet communication, using the HTTP protocol, between microcontroller and phone. The internet communication is responsible for the wireless component control and the Bluetooth communication for the initial data transference, like the microcontrollers local IP or the type and number of components it is wired to, as the project is built in a way that allows many different microcontrollers of different design to be used by the same user and app. Microcontrollers of different design refers to different types and/or number of components connected to said microcontroller via its digital I/O.

The microcontroller acts as a Web Server which can be accessed through its URL, composed by the microcontrollers local IP, and will return a web page, an HTML file, when a request is made. The request, made by the phone application, holds the instruction passed from the user to control a certain component connected to the microcontroller.

The finality of the project is to establish wireless internet communication between microcontroller and phone so the microcontroller can be personalized depending on the users needs, installed in the household and easily controlled from the developed phone application.

Key words: Domotics, Network, Web Server, Microcontroller, Automation.
ACKNOWLEDGMENTS

Firstly, I would like to thank my family for the amazing support they have shown.

At first mention of the project I was about to take on, I received some reluctant responses as it is indeed a complex project. Nevertheless, when I chose to proceed with it, I received nothing but help and support.

I would also like to thank my flatmates, as they have supported and helped me in all the ways they could.

Lastly, I would like to thank my tutor for giving me the chance to take on such an interesting project and for all the support and help he has offered during its development.
INDEX OF CONTENTS

1. INTRODUCTION .................................................................................................................. 1
   1.1. Context .......................................................................................................................... 1
   1.2. Objectives ..................................................................................................................... 2
   1.3. Motivation .................................................................................................................... 3
   1.4. Temporal planification and development ................................................................. 4
2. STATE OF THE ART .............................................................................................................. 6
   2.1. Problem approach ......................................................................................................... 6
   2.2. Current solutions ......................................................................................................... 6
   2.3. Hardware analysis ....................................................................................................... 8
   2.4. Software analysis ........................................................................................................ 15
3. PROJECT DEVELOPMENT .................................................................................................. 19
   3.1. Description .................................................................................................................. 19
   3.2. Project Design ............................................................................................................ 19
   3.3. Functionalities ............................................................................................................ 19
   3.4. Restrictions ................................................................................................................ 20
   3.5. Uses ............................................................................................................................. 21
   3.6. Selected materials ..................................................................................................... 21
   3.7. Costs ............................................................................................................................ 23
4. PROJECT DESIGN AND ARCHITECTURE ....................................................................... 25
   5.1. Architecture ................................................................................................................ 25
   5.2. Communication .......................................................................................................... 26
   5.4. Database ...................................................................................................................... 39
   5.5. Phone App .................................................................................................................. 41
   5.6. Microcontroller .......................................................................................................... 50
5. CONCLUSION ...................................................................................................................... 52
   6.1. Future developments ................................................................................................. 53
BIBLIOGRAPHY ....................................................................................................................... 54
## INDEX OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fig. 1.1</td>
<td>Gant diagram</td>
<td>5</td>
</tr>
<tr>
<td>Fig. 2.1</td>
<td>DSD Tech HC-05 Module</td>
<td>8</td>
</tr>
<tr>
<td>Fig. 2.2</td>
<td>Arduino Wi-Fi shield</td>
<td>9</td>
</tr>
<tr>
<td>Fig. 2.3</td>
<td>ESP8266 Wi-Fi Module</td>
<td>10</td>
</tr>
<tr>
<td>Fig. 2.4</td>
<td>Arduino Uno</td>
<td>11</td>
</tr>
<tr>
<td>Fig. 2.5</td>
<td>Arduino Nano</td>
<td>12</td>
</tr>
<tr>
<td>Fig. 2.6</td>
<td>Arduino Mega</td>
<td>12</td>
</tr>
<tr>
<td>Fig. 2.7</td>
<td>Raspberry Pi</td>
<td>13</td>
</tr>
<tr>
<td>Fig. 2.8</td>
<td>Node MCU</td>
<td>14</td>
</tr>
<tr>
<td>Fig. 2.9</td>
<td>Android Studio Logo</td>
<td>15</td>
</tr>
<tr>
<td>Fig. 2.10</td>
<td>MIT App Inventor example [63]</td>
<td>16</td>
</tr>
<tr>
<td>Fig. 3.1</td>
<td>Android Platform Version [66]</td>
<td>22</td>
</tr>
<tr>
<td>Fig. 5.1</td>
<td>Projects Architecture [53]</td>
<td>26</td>
</tr>
<tr>
<td>Fig. 5.2</td>
<td>Project communication</td>
<td>27</td>
</tr>
<tr>
<td>Fig. 5.3</td>
<td>General Bluetooth connection diagram</td>
<td>30</td>
</tr>
<tr>
<td>Fig. 5.5</td>
<td>Specific Bluetooth communication diagram</td>
<td>30</td>
</tr>
<tr>
<td>Fig. 5.6</td>
<td>light HTML response</td>
<td>32</td>
</tr>
<tr>
<td>Fig. 5.7</td>
<td>OkHttp Android Studio implementation [59]</td>
<td>33</td>
</tr>
<tr>
<td>Fig. 5.8</td>
<td>OkHttp GetURL method [59]</td>
<td>33</td>
</tr>
<tr>
<td>Fig. 5.9</td>
<td>Used OkHttp method</td>
<td>34</td>
</tr>
<tr>
<td>Fig. 5.10</td>
<td>Component modification</td>
<td>35</td>
</tr>
<tr>
<td>Fig. 5.11</td>
<td>XML Internet Security file</td>
<td>36</td>
</tr>
<tr>
<td>Fig. 5.12</td>
<td>Bluetooth module and microcontroller connection</td>
<td>37</td>
</tr>
<tr>
<td>Fig. 5.13</td>
<td>Light analog circuit</td>
<td>37</td>
</tr>
<tr>
<td>Fig. 5.14</td>
<td>Alarm analog circuit</td>
<td>38</td>
</tr>
<tr>
<td>Fig. 5.14</td>
<td>SQL Database structure</td>
<td>39</td>
</tr>
<tr>
<td>Fig. 5.15</td>
<td>Activity life cycle [62]</td>
<td>41</td>
</tr>
<tr>
<td>Fig. 5.16</td>
<td>Main Activity without information</td>
<td>43</td>
</tr>
<tr>
<td>Fig. 5.17</td>
<td>Main Activity with information</td>
<td>43</td>
</tr>
<tr>
<td>Fig. 5.18</td>
<td>Component Activity Paired Device Fragment</td>
<td>44</td>
</tr>
<tr>
<td>Fig. 5.19</td>
<td>Component Activity Internet Fragment</td>
<td>45</td>
</tr>
<tr>
<td>Fig. 5.20</td>
<td>Component Activity Connecting microcontroller</td>
<td>45</td>
</tr>
</tbody>
</table>
Fig. 5.21 Launched Main Activity................................................................. 46
Fig. 5.22 Lights Activity ........................................................................... 47
Fig. 5.22 Alarm Activity ........................................................................... 47
Fig. 5.23 Edit Light Activity List ............................................................... 48
Fig. 5.24 Edit name ................................................................................... 49
Fig. 5.25 Edit password dialog box ......................................................... 49
Fig. 5.26 Password Activity ..................................................................... 50
Fig. 5.27 Microcontroller Loop ................................................................. 51
INDEX OF TABLES

TABLE 2.1 TECH HC-05 BLUETOOTH MODULE TECHNICAL SPECIFICATIONS ..................... 8
TABLE 2.2 TECH HC-06 BLUETOOTH MODULE TECHNICAL SPECIFICATIONS ..................... 9
TABLE 2.3 ARDUINO WI-FI SHIELD TECHNICAL SPECIFICATIONS .................................. 9
TABLE 2.4 ESP8266 WI-FI MODULE TECHNICAL SPECIFICATIONS ................................. 10
TABLE 2.5 ARDUINO UNO TECHNICAL SPECIFICATIONS ............................................. 11
TABLE 2.6 ARDUINO NAMO TECHNICAL SPECIFICATIONS ........................................... 12
TABLE 2.7 ARDUINO MEGA TECHNICAL SPECIFICATIONS ........................................... 13
TABLE 2.8 RASPBERRY PI TECHNICAL SPECIFICATIONS ............................................. 13
TABLE 2.9 NODE MCU TECHNICAL SPECIFICATIONS ............................................... 14
TABLE 3.1 HARDWARE COSTS ....................................................................................... 24
TABLE 3.2 TIME INVESTMENT COSTS .......................................................................... 24
TABLE 3.3 INVESTED TIME .......................................................................................... 24
TABLE 5.1 DATABASE STRUCTURE .............................................................................. 40
1. INTRODUCTION

This document’s objective is to describe the development, objectives and purpose of the project at hand.

The first chapter is a general introduction to the project. The context of domotics in modern society, the motivation to pursue this field in particular and the general objectives the project strives to achieve.

Following the overall introduction is the state of the art. The chapter where the available technologies are analyzed as well as the current development of the field by different companies in modern society.

The overall functionalities, design and aspects of the project are displayed in the chapter following the state of the art, project design.

Finally, the detailed description of the developed project is defined in the second to last chapter of the project, design and architecture.

The last chapter is the conclusion, where as the name dictates, conclusion are drawn from the analyzed content and the experience of building the project in general.

1.1. Context

Domotics (from the Latin word “domus”, house) is a science devoted to the control and automation of the household [1].

The first ever smart device was developed during the 1960 by James Sutherland, an electrical engineer. ECHO IV [2] (Electronic Computing Home Operator) was a home computer [3], that is, an inexpensive microcomputer [4], a computer that uses a microcontroller as its CPU (Central Processing Unit), intended for non-technical users that could control home appliances, build shopping lists and monitor the house’s temperature.

Following the ECHO IV in the 1970 Honeywell developed the Honeywell 316 [5] or more commonly known as the “Kitchen Computer” [6]. The 50kg minicomputer with a price of about 10,000$ was capable of managing food recipes. It was immensely complex to use as it used a toggle-switch input and a binary light output. A two-week course was needed to operate the complex user-interface. None of these devices were ever sold.

During the mid-1970 the microprocessor was finally invented. Because of its reduced-price, electronic devices became much more affordable.

Finally, the combination of the internet and the microprocessor during the 1990 and the 2000 increased the popularity of home automation as the era’s technologies made it a possible option although an expensive one, nevertheless [7].

Nowadays it is an ever-growing presence in modern society. It makes the household more comfortable, safe and efficient as it gives the user a better control and automation over certain aspects of the household, like the alarm, electronic systems, lights, water systems, locks, fire alarm…
The automation consists in a trigger that defines when an action should take place, and the actual action, which defines what the smart device should do.

It is widely used by different people and companies as it can extend well over private households and can be used to optimize company buildings or warehouses.

It is also very useful for incapacitated people as they cannot perform certain tasks on their own and they must be automated or made by a machine.

1.2. Objectives

The final objective of this project is the control and automation of the household components through a wireless internet communication between phone and microcontroller.

The project is comprised of two fundamental components. The phone application and the microcontroller, each consisting of their own separate objectives for when combined, the final goal can be achieved.

- **Phone App**

  The phone application is responsible for communicating the user’s intentions to the microcontroller.

  From the phone app, through a graphical interface, the user will be able to visualize the state of the different components and control or automate them as the user sees fit.

  Among the phone app’s functionalities,

  - Communicates the user’s intentions to the microcontroller by means of a wireless communication.
  - Manages the data received from the user as well as the data received from the microcontroller and discerns the relevant information.
  - Builds databases based on the microcontroller’s information, as more than one microcontroller will be able to connect to the same phone application.

- **Microcontroller**

  The microcontroller’s primary objective is to manage the components depending on the phone’s instructions.

  Among the microcontroller’s functionalities,

  - Communicates information received from the different components to the phone application.
  - Works as a Web Server so any phone can connect to it independently of the distance and send or receive information.
  - Sends and receives digital inputs/outputs as to control the different electronic components connected.
The microcontroller will be installed in the household with components depending on the user’s necessities and the app must be installed on the user’s smart phone.

As for how the communication process will work between the phone app and the microcontroller. Firstly, the phone application and the microcontroller will connect via Bluetooth. By means of this connection the phone will build an independent database with the microcontroller’s components information and local IP address. Each microcontroller can have different components depending on the user’s preferences, like for example, light control, alarm, irrigation or other electronic systems and several microcontrollers can be controlled by the same user and app.

The microcontroller will also have to be connected to a private Wi-Fi network, as it will act as a Web Server. The phone app will send the Wi-Fi’s accreditations to the microcontroller via the Bluetooth connection and once the microcontroller is successfully connected to the local Wi-Fi network, it will terminate the Bluetooth connection as sufficient information has been shared for the phone app to generate a web request towards the microcontroller, working as a Web Server, each time the user wants to send a certain instruction to the microcontroller.

Once internet communication has been established, the phone app will have built its database with the microcontroller’s components information and it will be able to control and monitor its digital outputs and inputs to change or receive relevant information from and to the electronic components connected to the respective microcontroller.

1.3. Motivation
This chapter describes the fundamental reason this project in particular was selected among all the offered projects.

The main reason for the election of this project is self-improvement and my personal likeness towards the subject in general as well as the tools needed to complete the project at hand. Learning and mastering new programming languages is extremely useful as well as knowledge on wireless communications, like Bluetooth or the Internet. This project uses five different programming languages (Java, C/C++, XML, HTML, SQL), database management and internet and Bluetooth connections, among other things. All of which is extremely useful and interesting knowledge.

Regarding motivation external to personal improvement and liking, nowadays, technology has advanced on such a level that it allows certain commodities at a decent price.

Home automation, or domotics, doesn’t just make it easier and more comfortable to control and manage a household or building, but also extends to optimization of processes and generally higher efficiency. It helps in cost reduction and negative environmental impacts and it can extend well over households as it can be implemented in company warehouses or company buildings.

A good comparison in my opinion is the automatic car. An automatic car changes gears in an automated way increasing efficiency as it changes gears in an optimal manner to reduce fuel consumption, not to mention the commodity and easiness it provides the user
with. Much in the same way, an automated home allows the electronic devices to synchronize and maximize efficiently. For example, an automated heating system shouldn’t work while the house is empty, reducing overall consumption, or an automated boiler, should stop heating the water if the house is empty.

Because of these great advantages domotics offers, many companies like Google or Apple, have started branching out in this new field of electronics but with extremely expensive and difficult to set up equipment.

Concluding the motivation, excluding personal reasons, the installation of a domotic system is extremely expensive, and as stated previously, evermore necessary. If not for people who need it because of the increasing complexity of the devices in the household, for people with physical or mental disabilities that need assistance with every-day tasks.

1.4. **Temporal planification and development**

I initially ventured in this project at the beginning of October of 2019.

I started off, as with any project of this magnitude, investigating on its current state in the market and studying its functionalities in different relevant companies.

After careful consideration and assessment, I started planning and designing the overall functionalities and distinctions of my project.

Once I decided what my project was going to consist of, the functionalities, the objectives and the overall design, I started researching different methods and tools I could use to reach my ends need.

As many of the tools I was going to need for the development of the project I had no knowledge of, I proceeded by learning one by one the high-level programming languages and the respective environments in question. First of all, the programming language of Java [65], as the Android phone application, using the Android Studio IDE, is programmed in said language. As the IDE also uses XML for the phone app’s graphical interface, once I learned Java and started learning how the actual environment works, as it is an immensely complex computer program, I parallely learnt XML.

I proceeded by choosing a microcontroller fit for the set objectives and learning the necessary language to program it. Fortunately, thanks to some of the subjects I took, I was familiar with microcontrollers with a higher complexity to the one I chose, and I had already used Arduino once before, so it was a matter of remembering how to use it. However, as I had never used the necessary libraries, I had to do a fair bit of research before I deemed it possible with the use of the microcontroller I had chosen.

As I proceeded building the app, I learnt SQL, as SQL is used for the creation of the database and XML.

Regarding the microcontroller, I also had to learn HTML as the Web Server returns a web page.

The following Gant diagram represents and approximate estimate of the development of the project.
In the above image many tasks are superimposed. The reason for this is, although I considered the task finished, I kept on acquiring new knowledge about the supposed closed task even as I started something new. Everything is intimately related, so I never truly stopped learning about all the subjects.

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Q4</th>
<th>Q1</th>
<th>Q2</th>
<th>Q3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Learn Java</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learn Android Studio</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learn XML</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Develop mec applications</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project app development</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learn microcontroller software</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Develop microcontroller software</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learn and apply SQL</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learn and apply HTML</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design and implement analog circuits</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Fig. 1.1 Gant diagram
2. STATE OF THE ART

This section’s objective is to evaluate the different technologies currently available and analyze the advantages and disadvantages of each, regarding the technological specifications of this project in particular.

It is also necessary to evaluate the current state of domotics in the market as many companies have branched out in this particular field and it is important to reach a conclusion based on the current state of domotics in modern society as well as the current available technologies.

However, firstly it is important to analyze why Domotics is important to develop as a science in modern society and the advantages it has over the conventional non-automated household.

2.1. Problem approach

As stated beforehand, domotics is a growing science and used mostly to improve the commodity and efficiency in a household. Nowadays, technology is continually increasing and growing in complexity, which makes it more complicated to follow and manage all the available functionalities of the household individually. Domotics allows control, automation and management of all these complex systems, making it much simpler for the user to keep track and manage everything happening in the household.

In addition, domotics relates all the individual functionalities in a household so they can work in synchronization and maximize their efficiency and it allows a profound personalization and adaptation of the household to suit the user’s needs.

Finally, another important aspect it provides is security. To be able to set an efficient security system and manage it from a distance is an amazing improvement to the standard household alarm.

2.2. Current solutions

As stated previously, it is important to analyze the market regarding domotics in modern society as it can provide vital information as how the different companies are approaching the subject. Therefore, this chapter is dedicated to the analysis of the most influencing companies that have branched out in this field of science.

As domotics is increasing in popularity, many important and wealthy companies, such as the listed below, have started to branch out in this new science and developed their own system of house control and automation. Among them,

- **Google**
  
  Google has developed “Google Assistant”.

  Google assistant works on the Google Home device, a smart speaker or a smart display [9], and is capable of answering questions, playing music and setting timers, but it is also used for domotics, as it can be configured to control smart devices like lights, locks or security cameras.

  The advantage of using Google Assistant is that it is compatible with a large rage of smart devices. It works with over 1000 smart home brands [8], and the user can
control all these systems with a simple voice command. However, it is a difficult configuration process [10] compared to the proposed project app, and the smart devices, as well as the Google Home device, are very expensive items to acquire.

- **Apple**

  Apple has developed “HomeKit”.

  The Apple Homekit works with a smaller variety of devices. The components are also very expensive to acquire and requires IOS devices [11].

  Unlike Google, it does not require a speaker. It can all be managed from the iPhone, using the Apple Home app or using Siri, the virtual smart Apple assistant, not unlike Google assistant [12].

  In addition, the configuration process is much easier as each individual smart device is identified by a unique code, which can be used to connect the device to the home App.

  However, as with Google, the components are extremely expensive to purchase.

- **Easydom - Microsoft**

  Easydom is a fully customizable domotics system that can communicate with the user’s home and learn the user’s habits. It also integrates an automatic search for the smart devices already installed in the household and manage them from a single piece of software [13]. Access, control and automation can be managed through a graphical interface.

  Easydom can communicate with Google assistant or Amazon Alexa, it integrates remote control from a phone application, helps reduce electric consumption and increases the security level of the building [14].

- **Amazon Alexa**

  Alexa, much like Google assistant, is a virtual smart assistant. It is developed by amazon and is capable of voice interaction, music playback, making to-do lists, setting alarms, playing audiobooks, streaming podcasts, providing weather and so on [15].

  It is also used to control or automate smart devices. Users can extend Alexa’s capabilities by installing different functionalities named “skills”.

  The assistant Alexa, like Google Home, is accessed with a speaker.

The biggest inconvenience of all these companies is the immense cost it implies to install a domotic system and the complexity of the configuration. Most of the proposed solutions consist of expensive individual smart devices controlled or automated by a center software piece like a phone app a speaker or a display.
2.3. Hardware analysis

As the only hardware component that is to be used in this project is the microcontroller, a side from the negligible electronic components like LEDs and resistors, we must perform a comparison between the most suitable microcontrollers for this project.

- **Arduino**
  An Arduino is a microcontroller motherboard. It is a simple computer able to run one program at a time.

  It is an open-source platform that consists of hardware (the microcontroller motherboard) and of software, the Integrated Development Environment (IDE), that runs on a computer used to write and upload the written code to the actual physical Arduino board.

  Unlike most microcontrollers, Arduino does not require a separate piece of hardware in order to upload its code from the computer. It is also very easy to program as it uses a High-level language similar to C++ and C.

  Arduino is also compatible with a large range of different modules that allow the board to perform tasks it would not be able to on its own. For example, Bluetooth and internet connections.

  Regarding the Arduino Bluetooth modules, among the most popular,

  - **DSD Tech HC-05**
    Among other things, this module is cheap, and lots of information about it is available online. It can work as master and slave, meaning it can initiate connection to another device as well as accept it.

    ![](Fig. 2.1 DSD Tech HC-05 Module)

    **TABLE 2.1**
    | TECH HC-05 BLUETOOTH MODULE TECHNICAL SPECIFICATIONS |
    |---------------------------------|-----------------|----------------|-----------------|------------------|
    | Operating Voltage | Operating Current | Range   | Baud Rate | Average Price |
    | 4-6 V             | 30 mA            | <100m   | 9600       | 8€              |
The above table consist in some of the most important of the module’s technical specifications [16] [18].

The disadvantage of this module is that it does not work for IOS. However, as the project is an Android application it is of little consequence.

- **DSD Tech HC-06**
  This module is very similar to the HC-05 aside for it can only accept a connection from another device and not provide one. It also differs in some of the technical specifications as shown in the table below [17] [19].

<table>
<thead>
<tr>
<th>Operating Voltage</th>
<th>Operating Current</th>
<th>Range</th>
<th>Baud Rate</th>
<th>Average Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-4 V</td>
<td>30 mA</td>
<td>&lt;20m</td>
<td>9600</td>
<td>8€</td>
</tr>
</tbody>
</table>

As for the Arduino modules that allow internet connection,

- **Arduino Wi-Fi shield**
  The Arduino Wi-Fi shield is an Arduino module capable of connecting the Arduino to the wireless local Wi-Fi network. Much like Arduino, it is also an open-source software [20] [21].

Fig. 2.2 Arduino Wi-Fi shield

Its relevant technical specifications are shown in the table below.

<table>
<thead>
<tr>
<th>Operating</th>
<th>Operating</th>
<th>Range</th>
<th>Baud Rate</th>
<th>Average Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltage</td>
<td>Current</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3-4 V</td>
<td>30 mA</td>
<td>&lt;20m</td>
<td>9600</td>
<td>8€</td>
</tr>
</tbody>
</table>
The ESP8266 is a low-cost microchip with Wi-Fi connection, compatible with the TCP/IP protocols. It’s primary objective is to provide any microchip with internet access [22] [23].

![ESP8266 Wi-Fi Module](image)

**Fig. 2.3 ESP8266 Wi-Fi Module**

It is very easy to use as it can be programmed with Arduino IDE.

As for its technical specifications, the most relevant ones can be observed in the table below.

**TABLE 2.4**

<table>
<thead>
<tr>
<th></th>
<th>Connection</th>
<th>Security</th>
<th>Average Price</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operating Voltage</strong></td>
<td><strong>Connection</strong></td>
<td><strong>Security</strong></td>
<td><strong>Average Price</strong></td>
</tr>
<tr>
<td>3.3 V</td>
<td>802.11b/g networks</td>
<td>WEP, WPA and WPA2</td>
<td>3€</td>
</tr>
</tbody>
</table>

Regarding the Arduino brand, a few different types of Arduino boards are worth mentioning,

- **Arduino UNO**
It is a very robust board, the most used and the one with the most documentation online. It is a microcontroller board based on the ATmega328P that consists of 14 I/O digital pins, where six of the pins can be used as PWM, 6 analog inputs, a reset button, a power jack and a USB connection [24].

The technical specs are the following,

**TABLE 2.5**

**ARDUINO UNO TECHNICAL SPECIFICATIONS**

<table>
<thead>
<tr>
<th>Operating Voltage</th>
<th>DC current per I/O pin</th>
<th>Flash Memory</th>
<th>SRAM</th>
<th>EEPROM</th>
<th>Clock Speed</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 V</td>
<td>20 mA</td>
<td>32 KB, 0.5 KB used by bootloader</td>
<td>2 KB</td>
<td>1 KB</td>
<td>16 MHz</td>
<td>20 €</td>
</tr>
</tbody>
</table>

- **Arduino NANO**
  It is a small board based on the ATmega328. It lacks a DC power jack and works with a Mini-B USB cable instead of the standard one [25].

  It implements 22 digital I/O pins, six of which can be used as PWM, and 8 analog pins.
The technical specs are the following,

**TABLE 2.6**

**ARDUINO NANO TECHNICAL SPECIFICATIONS**

<table>
<thead>
<tr>
<th>Operating Voltage</th>
<th>DC current per I/O pin</th>
<th>Flash Memory</th>
<th>SRAM</th>
<th>EEPROM</th>
<th>Clock Speed</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 V</td>
<td>40 mA</td>
<td>32 KB, 2 KB used by bootloader</td>
<td>2 KB</td>
<td>1 KB</td>
<td>16 MHz</td>
<td>20 €</td>
</tr>
</tbody>
</table>

- **Arduino MEGA**
  The Arduino MEGA 2560 is a microcontroller board based on the ATmega2560. It has 54 digital input/output pins, of which 15 can be used as PWM outputs, 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button [26].

  It is a microcontroller designed for projects of a higher complexity than the ones developed on the Arduino UNO or on the Arduino NANO.
The technical specs are the following,

<table>
<thead>
<tr>
<th>Operating Voltage</th>
<th>DC current per I/O pin</th>
<th>Flash Memory</th>
<th>SRAM</th>
<th>EEPROM</th>
<th>Clock Speed</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 V</td>
<td>20 mA</td>
<td>256 KB, 8 KB used by bootloader</td>
<td>8 KB</td>
<td>4 KB</td>
<td>16 MHz</td>
<td>35 €</td>
</tr>
</tbody>
</table>

- **Raspberry PI**

Finished with the Arduino family, the Raspberry PI is composed by a series of small single-board computers [28], that is, a series of computers built on single circuit boards that include their own memory and microprocessor [27].

This allows the Raspberry, unlike the Arduino, to run several processes at once. It does, however, require an operating system.

![Raspberry PI](image)

Fig. 2.7 Raspberry PI

Some of its technical specs are the following [29],

<table>
<thead>
<tr>
<th>Processor</th>
<th>Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit SoC. 1.5 GHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory</td>
<td>1GB, 2GB or 4GB LPDDR4 (depending on model)</td>
</tr>
<tr>
<td>Price</td>
<td>35 €</td>
</tr>
</tbody>
</table>
- **Node MCU**

Node MCU is an open source IoT (internet of things) platform. That is, it is a microcontroller not unlike Arduino but, is provided with a unique identifier and has the ability to transfer data over a network. In other words, a microcontroller that includes internet connectivity.

It includes a firmware, which runs on the ESP8266 Wi-Fi SoC, compatible with the Arduino platform, and hardware, which is based on the ESP-12 module [30].

![Fig. 2.8 Node MCU](image)

Among its technical specifications [31],

<table>
<thead>
<tr>
<th>Microprocessor</th>
<th>Tensilica Xtensa 32-bit LX106 RISC microprocessor which operates at 80 to 160 MHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory</td>
<td>128 KB RAM, 4MB Flash memory (for program data and storage)</td>
</tr>
<tr>
<td>Wi-Fi</td>
<td>802.11b/g/n HT40 Wi-Fi transceiver</td>
</tr>
<tr>
<td>Cost</td>
<td>8 €</td>
</tr>
</tbody>
</table>

- **PIC18F2525**

This is a very simple and cheap microcontroller. However, it is very complex to use as the hardware has to be configured, unlike the Arduino or Raspberry PI, and is not compatible with the Arduino modules. Meaning the intended Bluetooth and Wi-Fi connection is very difficult to accomplish and poorly documented.

Its technical specifications and the necessary memory configuration is available in the microcontrollers datasheet [32].
2.4. Software analysis
Software is used to develop the phone app as well as to program the microcontroller. To accomplish these objectives the necessary IDE’s (Integrated Development Environment) must be used. Starting with the most popular phone application development tools,

2.4.1. Phone development software
Among the environments and programming languages that have been evaluated for the phone application of the project,

- **Android Studio**
    Android studio is the official integrated development environment (IDE) for Google’s Android operating system, built on JetBrains’ IntelliJ IDEA software and designed specifically for Android development. It is a replacement for the Eclipse Android Development Tools (ADT) [33].

![Android Studio Logo](image)

Fig. 2.9 Android Studio Logo

Android studio splits the code in two critical sections. The graphical part, which is programmed in XML, and the logical part, which is programmed with Java.

Among the features Android studio implements, it consists of a visual layout editor that makes creating graphical interfaces much easier and intuitive, an APK analyzer which helps to reduce the app size, a phone emulator and an intelligent code editor so it makes it easier and quicker to actually write the code [34].

It is also very well documented online and has a site with all the information of the methods, classes and packages that each API contains [35].

- **MIT App inventor**
    MIT App Inventor is a visual programming environment for developing phone applications. It is a web application integrated development environment (IDE) maintained by the Massachusetts institute of technology (MIT) but was originally provided by Google [36].
It allows the user to use a graphical user interface (GUI) which allows them to drag and drop visual objects to develop the application.

The blocked-based tool makes it extremely easy and fast to develop phone apps and very intuitive, allowing even people without programming knowledge to develop complex apps [37].

It is also well documented online, and it allows the user to create application software for Android as well as for IOS.

- **Eclipse**
  Eclipse is also an integrated development environment (IDE). It is very versatile application as it supports various programming languages and extensive plug-in systems allowing the environment customization. Among them, the Android Development Tool (ADT) Plug-in for Android phone application development [38]. This plug-in provides Eclipse with a powerful IDE in which to develop Android phone applications [39]. In addition, not unlike Arduino studio, the logic part of the app is programmed using Java, as for the graphical interface, it is programmed with XML.

**2.4.2. Microcontroller development software**
Regarding the microcontroller development software, among the most popular environments,

- **Arduino IDE**
  It is a cross-platform (implemented on multiple computing platforms) open source software integrated development environment (IDE) compatible with any Arduino board, and other development boards [40].
The environment is written in Java and it is based on Processing and other open source software [41].

It is a very user-friendly environment to use and makes programming the microcontroller very easy, as it uses the programming languages C or C++ using special rules of code constructing. It also provides a software library, which provide many additional functions that makes it easier to program the board.

The user-written code only requires two functions. A setup function, which is only executed once when the board is loaded or reset, and a loop function that is constantly running. The loop function is equivalent in C to a “while(true)” statement.

- **ASSEMBLER**

Assembler is a computer program that translates assembly language [43], that is, basic computer instructions, into machine code. A pattern of bits that the computers processor can use to perform basic operations [42].

Assembly language is a low-level programming language and is very tedious and difficult to use. The high-level programming languages, such as C, C++ or Java, are much easier to use and thanks to the compilers, it can be translated to lower-level programming languages such as assembly [64].

Assembly is used to program microcontrollers because it allows the development of a much more efficient code, thus optimizing the use of the microcontroller’s memory and processor.

- **Scratch**

Scratch is a visual programming language based on blocks, much like the language used in App Inventor. It is developed by the MIT Media Lab and is targeted primarily at children [44].

This is one of the languages that can be used to program the Raspberry PI.

- **Python**

Python is an interpreted (it directly executes the code instructions, translating each statement into a sequence of one or more subroutines, that is, unlike a complier that produces a low-level code like assembler, the interpreter directly executes the code [46]) high-level, general-purpose [47] programming language.

The language is object oriented and helps the programmers develop clear code. It supports multiple programming paradigms (language classification based of its features) like object-oriented, functional programming and procedural [45].

Among its many uses, it is used to program the Raspberry PI.
2.4.3. Phone operating systems

Finally, the most popular phone operating systems must be evaluated.

An operating system (OS) is the software responsible for managing the device’s software.

Among the most popular phone operating systems,

- **Android**
  Android is the currently most used operating system available. It has been the best-selling OS regarding smartphones since 2011 and regarding tables since 2013. As of May of 2017, it has over two billion monthly active users. Currently is has a 76% market share [49].
  It is a Linux-Based OS (Linux Kernel) developed by Android Inc., which Google bought in 2005. It was initially developed for touchscreen mobile devices but is not extending tablets, PC’s, televisions (Android TV), cars (Android auto), and watches (Wear OS) [48].

- **IOS**
  IOS is the mobile operating system (originally iPhone OS) developed by Apple Inc. It is exclusive to Apple hardware so it can only be used by devices like the iPhone, iPad and so on. It currently has a 22% market share [49] making it the second most popular operating system after Android [50].
  One of its biggest advantages over other OS is its simplicity and its great efficiently. However, this reduces greatly the phones customization.

- **KaiOS**
  KaiOS, like Android, is a Linux-Based operating system. It is owned by KaiOS Technologies and it currently has a 0.80% market share [49], making it the third most popular operating system [51].
  The operating system brings smartphone-like functionalities to affordable phones [52].
3. PROJECT DEVELOPMENT

Having evaluated all the existing services the different companies offer regarding domotics, it is possible to analyze them and draw conclusions to try and better the design and functionalities of the developed project.

3.1. Description

The project’s objective is to wirelessly control and automate different household components such as lights and alarms with the use of a phone application.

There are many different ways the phone application can communicate to the microcontroller, but the chosen communication method is through an internet connection, as nowadays Wi-Fi is available in practically every building and it allows communication to take place independently of the distance between devices.

Having analyzed the market, it can be observed that most of the analyzed companies use what they call “Smart devices”. These consist of electro-domestic components that can receive instructions from a source as a phone app or a speaker through wireless communication.

The smart devices are extremely useful as one only needs to buy it, download the corresponding app, or buy the corresponding electronic device, configure it and it should be ready to work.

The downfall of all smart devices is their extremely elevated price. These are extremely expensive and most of the time quite hard to configure.

3.2. Project Design

Having concluded the faults in the current companies supporting Domotics and having evaluated the current available technologies, it is possible to proceed with the design of the project.

Instead of using various smart devices, like most of the analyzed companies, using a single microcontroller for many different components allows the automation and control of said components through the microcontroller without the need of them being “Smart”. This immensely reduces the cost, as the components no longer have to be “Smart” but normal working electro domestics. It also increases simplicity, as they will all be controlled from the same source.

3.3. Functionalities

Regarding the different functionalities of the project, it is important to differentiate between the graphical interface functionalities and the functionalities of the actual process of communication between microcontroller and phone application.

- Communication

Communication is a vital part of this project. It allows the microcontroller to send and receive vital information that configures the Internet connection of the microcontroller. It allows the user to control the microcontroller, using the phone applications graphical interface, and it allows the actual microcontroller to manage the components it’s attached to.
The components receive digital signals from the microcontroller they are connected to. These signals are a direct response to wireless instructions received from the phone, through an internet communication, generated by the user’s commands through the applications graphical interface.

The graphical interface allows the user to manage the connected household components, but it also allows the user to change the workings of the actual app.

- **Graphical Interface**
  The graphical interface consists of various lists that hold information of the components that are connected to the microcontroller in question, and various buttons which the user can use to interact and send instructions to the microcontroller or manage the actual app. Each button represents a different functionality. Among them,

  A. **On/Off or Enable/Disable**
     This allows the user to Enable or Disable the component in question. For example, switching a light on or off.

  B. **Change Name**
     This functionality allows the user to rename the component to her or his liking without affecting its functionality. This functionality is very practical as the components ID’s aren’t very user-friendly names to remember the components by.

  C. **Password**
     Much as the name change, initially no password will be associated to the components, but it can be set to restricts component control access.

  D. **Delete**
     If a component is deemed redundant for whatever reason, the user can choose to delete it from the Database. This, however, will not delete it from the microcontroller, but it will cease to exist on the phone application so the user will not be able to control or automate the component in question.

3.4.**Restrictions**

Regarding the project’s restrictions, the use of a single microcontroller to automate and control several different components reduces the cost immensely. However, it implies that the microcontroller must be installed wherever it wants to be used, unlike the smart components, which can simply be bought and configured. This implicates an initial investment, as the existing household electronic devices must be connected to the microcontroller. It also means that the users cannot do it themselves. A team of
technicians must be assembled to install the microcontroller and connect each available component to the corresponding microcontrollers digital I/O.

Another important restriction the developed project has is the number of ports. The microcontrollers are physically connected to the components. As there is a finite number of digital ports, only so many components can be connected to one microcontroller. However, the microcontrollers are not expensive devices. More than one can be installed without having great effects in the final cost of the installation.

Finally, it is important to note that unlike the microcontroller, the smart devices can be added dynamically to the household. That is, the user can buy new “smart” components and add an infinite number to their collection independently of the time the components are bought. Because the microcontroller is connected to all the available components, once it is installed, if the user wants to add a new component, the microcontroller must be changed as it was previously designed for the existing components at the time of the installation.

3.5. Uses

Once stated the restrictions of the project, the uses and applications can be analyzed.

As wireless communication is established between phone and microcontroller, the digital output of the microcontroller’s ports can set to high-level or low-level. This translates in to complete control of the digital input of any component connected to the microcontroller so any user’s desire or need can be satisfied as it is a simple matter of knowing what input a certain electronic devices needs to function and adjusting the microcontrollers digital output to coordinate an appropriate response.

For example, LED lights need about 2 o 3 volts to illuminate. A simple analog circuit design is sufficient to allow the microcontroller to turn on the LED lights with a digital high-level output of 5 volts.

Ultimately, with the correct design, many electronic devices can be wired to the microcontroller to be controlled wirelessly by the user.

3.6. Selected materials

Finally, after careful evaluation of all the possible materials, the following have been chosen to complete the project at hand, taking in to account the chosen design for the domotics system.

Before the selected materials are mentioned, it is important to note that, as explained in the previous chapter, the microcontroller will connect via its digital ports to the already existing electronic components in the corresponding household. As the components are not part of the project they will not be mentioned as selected materials.

Following with the selected materials to use to develop the project in question,

- **Phone OS**

Regarding the phone’s operating system, the choice was based on one aspect only. The number of current active smartphones operating with said operating system.
As android is the dominant OS regarding quantity, this operating system in particular was chosen above the rest.

It is also possible to adapt an Android phone application for an iOS operating system using tools like Mechdome [67].

![Fig. 3.1 Android Platform Version [66]](image)

The image above consists of the different Android platform versions with its respective API and its distribution.

The API [68], or application program interface, is a set of software building tools, protocols for communication and definitions. The higher API’s consist of more modern packets that contain classes and methods. They are more optimized and easier to use but they are not compatible with lower API using phones. The higher the API the developer uses, the more likely it is that the phone won’t be compatible as it might be using a lower API.

Because of this last statement, the most compatible version was chosen, which is the lowest API, the “Ice cream Sandwich” with an API level of 15. This version, as it has the lowest API, is compatible with a 100% of the Android devices.

However, during the programming of the application, a certain class was needed for the internet connection of the application that belongs to API 19. So, it forced the reduction of the phone compatibility from 100% to a 95.3%, using the “KitKat” Android version.
• **Hardware**

Regarding the hardware, that is, the microcontrollers, the best-suited board is the Node MCU.

The optimal hardware board should be cheap and easy to program.

The Node MCU is a cheap board and it includes Wi-Fi connectivity, so no extra Wi-Fi module is necessary, reducing the costs.

It is also easy to program, as it is compatible with Arduino IDE, so it can be programmed in C/C++.

Finally, it is also compatible with both evaluated Bluetooth modules.

Summarizing, all these combined traits make the Node MCU board perfect to perform the task at hand.

Regarding the Bluetooth module, as both evaluated modules have the same price, it is only logical to choose the module with the best technical specifications. In this case, it is the HC-05 module.

• **Software**

Regarding the development of the phone app, after evaluating the pros and cons of every software, the final conclusion is that Android Studio is the best suited for the development of the phone application.

Android Studio is the most popular environment regarding phone applications, therefore, the one with the most available documentation. Besides the information programmers offer online, Android Studio offers information on how to manage different classes and its methods in the API reference [69].

This integrated development environment is not the easiest to work, as complex methods and classes have to be used, and a good understanding of the Java programming language is needed. Not to mention, the complexity of the application itself, as it implements many complex functionalities. However, it offers an immense capability of customization as it provides multiple large packets that contain many classes that allow the developer to develop the app exactly to his or her liking. Unlike other IDE’s that are more restricted due to the many simplifications made to provide an easier IDE.

3.7. **Costs**

After analysing the materials needed to complete the project, an estimate cost can be calculated.

Because the electronic components connected to the digital ports of the microcontroller will not take part in the calculation of the final cost, the reason explained in previous chapters, the only hardware components needed are the microcontroller and the Bluetooth module. Each hardware’s price is known, both the microcontroller and the Bluetooth module has an average price of 8 euros.
Also, the environments used to develop the software are free, so no additional charge must be added regarding software. Not including the app distribution cost.

Taking these costs into account, the total hardware cost is the following.

<table>
<thead>
<tr>
<th>Microcontroller Average Cost (€)</th>
<th>Bluetooth module Average Cost (€)</th>
<th>Total Average Cost (€)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>8</td>
<td>16</td>
</tr>
</tbody>
</table>

As for the time invested in the project, considering an approximate 15 euros per hour from the student developing the software and a 25 euros per hour of the tutor. Taking into account the total hours spent developing the project the labour cost is the following.

<table>
<thead>
<tr>
<th>Student Average Cost (€/h)</th>
<th>Tutor Average Cost (€/h)</th>
<th>Total Average Cost (€/h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>25</td>
<td>55</td>
</tr>
</tbody>
</table>

Finally, making an estimate of the time spent of each member, student and tutor, on the project a final cost can be calculated.

<table>
<thead>
<tr>
<th>Student Average Time (h)</th>
<th>Tutor Average Time (h)</th>
<th>Total Average Time (h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>400</td>
<td>50</td>
<td>500</td>
</tr>
</tbody>
</table>

Total cost:

$$Cost = \frac{€}{h_{Student}} \cdot h_{Student} + \frac{€}{h_{Tutor}} \cdot h_{Tutor} + Hardware (€)$$

$$Cost = 15 \cdot 400 + 25 \cdot 50 + 16 (€)$$

$$Cost = 7,266 €$$
5. PROJECT DESIGN AND ARCHITECTURE

This section is dedicated to the description of the design and architecture of the inner workings of the project.

Firstly, the architecture or structure of the developed project is explained, followed by a detailed explanation of the workings of the communication between microcontroller, phone application and electronic components.

It also details the phone application software as well as the microcontrollers developed software.

5.1. Architecture

The project is based on the MCV architectural panel. That is, the application is separated into three fundamental parts. Model, control and View (MCV).

This is a very commonly used structure as most software engineering projects undertake this architecture or its variants. It allows the reuse of code, and the separation of these vital parts help ease the development and maintenance of the application.

Each of the parts involved in this particular structure performs a crucial part in the inner workings of the application [53].

- **Model**
  Model is responsible for the resources the application manages, the shape the database is built in and the application’s logic.

  It is the responsible of maintaining the data of the application.

- **Control**
  Control is responsible for handling the request, that is, managing the actual data to generate a coherent response.

- **View**
  View represents the user interface. In our case, the graphical user interface. It displays the data and enables the user to interact and change the data.

All three modules interact with each other enabling the app to work efficiently.
As displayed in the image above, the controller manipulates the data. As does the view. The controller also renders the view and receives requests from it.

The developed app functions much in the same manner, as the user inputs an instruction through the view that sends a request to the controller. The logic of the developed application. The controller manipulates the data and renders a new view from the modified data.

5.2. Communication
The project is based on communication, as its primary objective is to communicate the user’s instructions to the microcontroller.

Communication between the microcontroller and the phone application, as stated beforehand, will mainly consist of a wireless internet connection.

The microcontroller functions as a Web Server, so the phone can access it through the corresponding URL (Uniform Resource Locator) affecting the behavior of the microcontroller accordingly. However, the microcontrollers URL depends on its Local IP address, which depends on the microcontroller in use and the local network it is connected to. It will also have to connect to the local Wi-Fi.

Because of these requirements, instead of manually connecting the microcontroller to the local Wi-Fi network from the source code, and manually imputing its local IP address in the phone application, it will firstly generate a Bluetooth communication to send and receive this basic information automatically to commence the Internet communication between Web Server and phone application.

Communication also takes place between the hardware components and the actual microcontroller. Though it is a physical communication, as it works via the microcontroller’s digital inputs and outputs (I/O), it is a form of communication, nevertheless.

The following image represents the overall communications.
As the image details, the phone app communicates with the Bluetooth module which relays the information to the microcontroller and vice versa.

The phone also sends a web request to the microcontroller and receives the HTML file body in response.

Before the individual connections are explained in detail, it is important to note that the chapter is divided in Bluetooth connections and Internet connections. Both divisions are subdivided, offering detailed explanations of the connection process as well as a detailed explanation of the actual communication process between microcontroller and phone application.

Starting with the Bluetooth connection,

- **Bluetooth connection**
  Bluetooth is a form of communication between electronic devices, within a short distance, via short-wavelength UHF (Ultra High Frequency) radio waves (2.4 GHz radio band). It generates a small (10m) personal area network (PAN) which can network between two and eight nearby devices. Its primary objective is to substitute cables. It requires less power and cost than Wi-Fi although, it generates a slower connection [54].

  The Bluetooth connection is essential in the project as not only does it allow the user to connect the microcontroller to the local network and send the local IP address to the phone application so it can access the Web Server. It also allows the microcontroller to send the relative information of the components it implements so the phone application can complete its database with the corresponding information.
The connection is made by the Arduino compatible Bluetooth module. The microcontroller sends the information to the module via its digital inputs and outputs and the module forwards if wirelessly through the Bluetooth connection. The process also takes place in reverse as the information is received from the phone.

Once the necessary parameters have been shared, the Bluetooth connection can be disabled and the Internet connection can start, that is, the microcontroller can start working as a Web Server.

Regarding the Bluetooth connection from the phone app and the microcontroller’s point of view,

- **Microcontroller**
  The microcontroller is programmed using the application Arduino IDE. The environment implements a library (SoftwareSerial.h) with specific classes and functions regarding Bluetooth connections and communications.
  These methods allow the microcontroller to connect via Bluetooth to the phone and transmit or receive information
  The source code begins an infinite loop that reads any information transmitted within the established connection. In this infinite loop, the Bluetooth communication takes place and the microcontroller sends information and manages the received data.

- **Phone App**
  As with the microcontroller, in the phone applications source code, the necessary classes and its methods must be implemented to start and manage the Bluetooth connection.
  Firstly, the devices that the phone is paired with will appear to the user, so the microcontroller must initially be paired with the corresponding phone. Once the paring with the microcontroller’s Android compatible Bluetooth module is complete, it will appear in the developed apps paired devices list and the connection between the paired device and the phone can proceed.

- **Bluetooth communication**
  Once the Bluetooth connection has been established, the actual Bluetooth communication between devices can take place.
  The communication consists in an exchange of characters that each source code (phone app software or microcontroller software) interprets as different instructions.
  Firstly, after the connection is successfully established, the phone app requests an input with the local Wi-Fi network accreditations. After the user inputs the corresponding information the transmission of information commences.
The phone sends the accreditations as an array of characters in the form of “ssid/password/0/”.

As the microcontroller is constantly searching for information from the connection, when it receives a character array that ends in “/0/” it executes a certain function that attempts to connect the microcontroller to the local Wi-Fi network with the accreditations sent from the phone app.

Once the attempt is started, the microcontroller sends a character array to the phone (“x101x”) to inform the phone that the internet connection attempt has started.

If the attempt is successful, the microcontroller sends the array “x105x”. If it fails however, it sends “x104x”.

With said received information, the phone either informs the user the connection was unsuccessful and will request the accreditations again or sends an instruction to the microcontroller that the internet connection was successful and that the phone app is ready to receive the connected components information.

As there may exist various components connected to the microcontroller and the connection only allows short character array transmissions, the components ID’s are sent individually following “107x” as one character array. The phone app receives the ID and fills a dynamic string array. When the last component ID is sent it follows “106x” indicating the end of the component array.

The phone proceeds in sending an instruction indicating it is ready to receive the microcontrollers local IP address. As it is a long array of characters it is split in to two transmissions much like the components ID’s.

Once the IP address has been received, the phone applications database is constructed with the corresponding information and the Bluetooth connection is severed as all the information needed to complete the internet communication is in the database.

The following block diagram displays the order of the steps taken in the exchange of information,
And a more detailed block diagram of the exchanged instructions between the phone app and the microcontroller is represented in the following image,
• **Internet connection**

The internet is an immense system of interconnected networks. It works in accordance with certain protocols to transmit data through various types of media [55].

A protocol is nothing more than a set of rules under which computers must communicate with each other. The primary internet protocols are TCP (Transport Control Protocol) and IP (Internet Protocol).

The microcontroller works as a Web Server [70]. That is, software or hardware that runs a software, that stores a web page. The Web Server processes a web request and returns a response.

In this case, the response is an HTML file processed over HTTP (Hypertext Transfer Protocol). The body of the HTML file is formed by the data the microcontroller recovers from the components, thus returning this information every time a request is made.

  o **Microcontroller**

Regarding the Microcontroller, as stated before, it works as a Web Server.

As the Wi-Fi module is integrated in the microcontroller no external connections are necessary for the connection to take place.

A library must be included (“ESP8266WiFi.h”) in the source code for special methods and classes must be used to manage the internet connection.

Firstly, the microcontroller must be connected to the local Wi-Fi network and once it is successfully connected, the code initializes an infinite loop that searches for a web request. Once the client is available it sets the components depending on the information passed in the URL, that is, Universal Resource Locator, a reference to web resources located in a specific network [56], and returns the HTML response.

To access the Web Server, its address or URL must be used. The URL consists of its local IP, so to access the HTML document the Web Server generated, the following URL must be used “http://local IP/additional information”.

The HTML file is created using the method “Client.print()” and has the following structure.
HTML [57], or Hypertext Markup Language is a Markup Language [58], that is, a language that defines elements of a document using tags, used to design web pages. The response the microcontroller returns, the HTML file, its content depends on the components state. For example, if a light is on, it will reflect in the HTML response so the phone app can update with up to date information. This allows the phone app to update with real time information regarding the different component’s state.

○ **Phone App**

As for the Phone App, it sends the corresponding URL and receives a response, which is the HTML body of the Web Server. The body will change depending on the state of the different components, so through the internet connection the current state of each component can be known by the phone each time it makes a web request. The internet connection through the app is achieved using OkHttp.

OkHttp is an open software HTTP client. The software contains certain classes with their corresponding methods that allow to send a web request. That is, an URL [59].
The following instruction is included in the Android Studio project gradle allowing the use of the OkHttp methods,

```
implementation("com.squareup.okhttp3:okhttp:4.1.1")
```

Fig. 5.7 OkHttp Android Studio implementation [59]

Using the OkHttp software, it integrates a class named “OkHttpClient” that when instantiated allows the phone to send a Web Request and receive the appropriate response.

The official documentation suggests the following use of the method,

```
OkHttpClient client = new OkHttpClient();

String run(String url) throws IOException {
    Request request = new Request.Builder()
        .url(url)
        .build();
    try (Response response = client.newCall(request).execute()) {
        return response.body().string();
    }
}
```

Fig. 5.8 OkHttp GetURL method [59]

However, to suit the project’s unique needs the code that is used is the following,
As can be observed in both images, the only modification is the elimination, in the original suggested code, of the method encapsulating the necessary method to retrieve the body of the HTML file as a string.

- **Internet communication**
  When the phone sends a web request via the URL, that consists of the microcontrollers local IP address, it also contains additional information that modifies the state of the components.
  
Through this method of communication, the phone can command instructions to the Web Server.

The microcontroller is accessed via de URL address and the additional information attached to the address is accessed through a method that returns the information as a string. Using this string the components can be updated depending on its content.

The following image of code represents the previously stated,
As for receiving information, as stated beforehand, the microcontroller, acting as a Web Server, creates an HTML response, that the phone receives every time it sends the URL through the network. This permits the phone application to update its information with the current state of each of the components.

The communication, however, cannot take place unless several security measures are taken.

A new XML resource file (“network_security_config.xml”) had to be created to configure the internet security parameters, as by default clear text traffic is not permitted.

The code of internet security configuration can be observed in the following image,
Component communication

The individual components are controlled by the microcontroller, which is controlled by the phone application which is commanded by the user’s instructions by means of the graphical interface.

It consists of a physical communication as the components are physically connected to the microcontroller's inputs and outputs that generate and receive digital signals.

The number and type of components is virtually limitless, conditioned by the imagination. However, in the time I have had to develop this project I have had time to only add a few components to the microcontroller.

Also, connected physically, via cables, to the microcontroller is the Bluetooth module.

The respective connection between the microcontroller and the Android compatible Bluetooth module is the following.
As for the analog circuit of the components,

- **Lights**
  The lights are connected to the microcontrollers digital output port in the following manner,
The diodes normal functioning voltage ranges between 2 and 4 volts. As for the current, it usually works under, 20mA.

As the microcontroller’s digital output is 5V, a resistor is needed to consume some of the voltage and produce a 20mA current.

Using Ohm's law, the value of the resistor needed to meet said requirements,

\[ V = R \cdot I \]
\[ V_{\text{micro}} - V_{\text{diode}} = R \cdot I_{\text{diode}} \]
\[ R = \frac{V_{\text{micro}} - V_{\text{diode}}}{I_{\text{diode}}} \approx \frac{5V - \frac{2}{2} + 4V}{20 \cdot 10^{-3}A} \]
\[ R \approx 100 \, \Omega \]

**Alarm**

The alarm is built using a proximity sensor. The sensor is composed by an infrared diode and a phototransistor that interrupts the current when the infrared light is disrupted.

![Fig. 5.14 Alarm analog circuit](image)

As shown in the image, the microcontroller’s digital input is connected between the phototransistor and the resistor. When the phototransistor is in the presence of the infrared light, it allows current to pass. When the infrared light is absent, however, the current ceases to pass as the transistor opens the circuit.
The general idea is to put both components, phototransistor and led, on the door or window and on the frame so when it is active, if the door or window is opened, the phototransistor and infrared led separate provoking the microcontroller’s digital input to change from high-level to low-level and the component status in the body of the HTML file will be updated with said information alerting the phone app that the alarm has been triggered.

5.4. Database
A database is a set of data with a certain structure stored on an electronic device.

The phone application uses a dynamic database that is formed by the information the microcontroller transmits to the phone app, via Bluetooth and the internet, that consists of the available components and their respective information.

The microcontroller data, however, is stored as constant variables in its source code or is retrieved dynamically from the components depending on their state.

The database of the app is managed and structured by a mixture of the languages of Java and SQL.

The structure of the app’s database is managed in SQL (Structured Query Language [60]). It is a domain-specific-language (DSL) [61], that is, a programming language specific to a certain domain, with the sole purpose of managing data belonging to a relational database management system (RDBMS). A software system that allows creation and management of databases.

The app’s database is created by the following variables in the source code,

```
//DEVICE IASLA
public static final String DEVICE = "Device";
public static final String AD_NUM_COLUMN = "Device_id";
public static final String AD_COMP_COLUMN = "Component_id";
public static final String AD_COMP_NAME_COLUMN = "Component_name";
public static final String AD_DEVICE_IP_COLUMN = "ip";
public static final String AD_DEVICE_PASS_COLUMN = "password";

private static final String SQL_CREATE_DEVICE_ENTRIES =
"CREATE TABLE " + Component_DDBB.DEVICE + " (" +
Component_DDBB.AD_NUM_COLUMN + " INTEGER PRIMARY KEY AUTOINCREMENT," +
Component_DDBB.AD_COMP_COLUMN + " TEXT," +
Component_DDBB.AD_COMP_NAME_COLUMN + " TEXT," +
Component_DDBB.AD_DEVICE_IP_COLUMN + " TEXT," +
Component_DDBB.AD_DEVICE_PASS_COLUMN + " TEXT")";

private static final String SQL_DELETE_DEVICE_ENTRIES =
"DROP TABLE IF EXISTS " + Component_DDBB.DEVICE;
```

Fig. 5.14 SQL Database structure
As can be observed in the image, firstly the name of the database, the table and the columns are defined as string variables which are then used to complete the string “SQL_CREATE_DEVICE_ENTRIES”.

This string is then passed as an argument in several overridden functions belonging to the “SQLiteOpenHelper” class creating the actual database.

Its structure is defined by the “SQL_CREATE_DEVICE_ENTRIES” string. The “DEVICE” string static variable is set as the table name. Following the table name is the table primary key. A primary key is a character unique to each row. It serves as a reference to access a certain row as rows may contain the same information, but they will never share the same primary key.

In the string, the SQL code sets the primary key as an integer and to autoincrement, so it will increment in number automatically as new rows are added to the database.

The database has the following structure,

<table>
<thead>
<tr>
<th>Device_id (primary key)</th>
<th>Component_id</th>
<th>Component_name</th>
<th>Ip</th>
<th>Password</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>String</td>
<td>String</td>
<td>String</td>
<td>String</td>
</tr>
</tbody>
</table>

The “Component_id” column is set from the information received from the microcontroller through the Bluetooth connection as mentioned earlier in the document. “Component_name” is a column that initially holds the components ID but can be changed by the user. This is the string that is displayed when the list is formed from the database information. It can be changed with the “change name” functionality.

The “IP” column is the components microcontroller local IP address. This is the string that is used for the URL to request a response from the Web Server. As the IP address is stored individually for each component, multiple microcontrollers can be set up and controlled by the app simultaneously.

Finally, the “Password” column is initialized as a string with an initial value of “NoPass”. When the components are controlled by the user, and this field holds this value in particular, it can be controlled without any password input. If the user, however, sets a password for a certain component, the password column will be updated with the value the user inputs and every time the component is accessed the same value must be introduced in the app to access control.

Once the structure of the database is defined, the database class must be instantiated. By means of Java SQL methods the database can be created, updated and deleted.
5.5. Phone App
The phone application is programmed in Android Studio.

As stated previously, it is an integrated development environment (IDE) and separates the app in two fundamental sections. The graphical interface, programed in XML (eXtensible Markup Language), and the logic, programed in Java.

A phone application is composed by a set of activities.

An activity is a screen the user can interact with and affect the apps behavior. Its life cycle is represented by the following diagram,

![Activity life cycle diagram](image)

Fig. 5.15 Activity life cycle [62]

When the activity is launched the “onCreate()”, “onStart()”, and “onResume()” methods are executed in sequence until it is successfully running. The methods belong to the “AppCompatActivity” class.

If the app is paused, that is, if it is sent to run in the background, the “onPause()” method will be executed. When it is reestablished as the primary running app it will execute the “onResume()” method.
If the Activity is stopped, the “onStop()” method will be called and depending on the situation of the app, it will either shut down, return to “onCreate()” or “onStart()”.

These methods allow the developer to execute instructions in certain stages of the activity. For example, when the “onCreate()” method is called, it is a good idea to initialize the variables the Activity will use as it has just been launched.

When the Activity is destroyed all the variables belonging to said activity are destroyed with it. This makes information communication between Activities necessary as they are not globally stored.

The Android class “Intent” is used to manage Activity operations. With it, other Activities can be initialized, and information can be transmitted from one Activity to the other.

The developed app consists of five activities, two fragments, three classes, and one interface.

Fragments are application components that allow the developer to instantiate Activity like elements within an Activity. This allows code reuse, as the fragment can be instantiated in any activity, and to change the view information within a same Activity as to reduce the number of Activities that form the app.

An interface is a Java class with the unique distinction that it is completely abstract. That is, it can’t be instantiated. When an interface is applied to a certain class, it obliges the developer to override the interface methods in said class.

It allows an interface method to be executed in a certain class from a Fragment. Without the interface applied to the class, no method would be executable from a Fragment instantiated in said class.

The generated activities are the following,

- **MainActivity**
  This is the Activity executed by default when the app is launched. It consists of a list of available component types, and access to create new components. That is, connect a microcontroller.

  When the app is first opened, the list is empty as it retrieves information from the database which is built from the microcontrollers information. As no microcontroller has yet been connected the database is empty.
Once the microcontroller has been connected and the information has been successfully exchanged the app will automatically return to the "MainActivity" and the new component type list will show.
Components

After clicking the “ADD COMPONENTS” button in the “MainActivity”, the Activity is changed via the “Intent” class to the “Components” Activity.

Firstly, a list with all the paired devices on the phone will show and will be clickable.

The paired devices list is a fragment named “Fragment_Paired_Devices_List”. It automatically instantiates when the activity is launched.

The Bluetooth module has already been paired and shows in the list as “DSD TECH HC-05”.

Once the modules device name is clicked the app will attempt to connect with the module via a Bluetooth connection.

When the Bluetooth connection is established the “Fragment_Set_Internet_Data” fragment is instantiated. This fragment requests the input of the local Wi-Fi network credentials to connect the microcontroller to the network. The Activities title remains untouched. The fragment replaces the previous paired device list fragment.
Once the credential fields are filled with the corresponding information and the “CONNECT!” button is clicked the Activity updates as showed in the following image,
The user is informed the internet connection is being attempted. There is also a text that displays all the incoming Bluetooth transmissions.

If the connection fails, the above text will update with a failure warning. If it is successful, the “MainActivity” Activity will be launched as can be observed in the following image.

![Fig. 5.21 Launched Main Activity](image)

As the database has been updated with the microcontrollers information, the component type list is shown and can be clicked.

If any of the component types are selected, the corresponding component activity will be executed showing a list with the respective component names.

As an example, the lights list.
• **Lights**

![Lights Activity](image)

Fig. 5.22 Lights Activity

Or the alarm list.

• **Alarm**

![Alarm Activity](image)

Fig. 5.22 Alarm Activity
The list consists of various elements. Among them a text, which is the name of the component initialized as its ID, a switch type button, allowing the user to enable or disable the component, and a configuration button.

If this last element is selected, the list changes and allows the user to change the components name, password or eliminate the component from the database all together.

![Fig. 5.23 Edit Light Activity List](image)

It is important to note that the Activity is the same. The modified element is the list as the class “BaseAdapter” allows the developer to pass a layout as a list element and various lists can be created and instantiated in the same Activity.

The different functionalities are defined by the “component_config” class and instantiated in the components activities as to make use of its methods.

Regarding these functionalities, when the user selects the edit button a dialog will allow the user to modify the components name.
The delete functionality also makes use of a dialog as to ensure the user did not select the option by mistake.

Much like the delete functionality, if the user selects the password button, a dialog box activates as shown in the image below.
• **Password**
If the positive button is selected, that is, “YES!”, unlike the component name change, the “change_password” activity is launched.
If the component has no previous password, the needed input values are simply the new password. It also requires the repetition of said password in case of user error.

If a password was previously added however, it will request said password as well as a new password twice, ensuring no user error.

![Password Activity](image)

Fig. 5.26 Password Activity

5.6. **Microcontroller**
The microcontroller is programmed in the application Arduino IDE.
The default source code consists in two main functions. A set up function that is executed only once when the microcontroller is run for the first time and an infinite loop function that runs in a cycle until the microcontroller is turned off, reset or reprogramed.
In the developed source code, each function has its unique purpose,

• **Setup**
The set up is used to initialize all the used variables, configure the Bluetooth module and configure the digital ports.
Loop
The loop function is ideal for the program’s needs as both the Bluetooth connection and internet connection must constantly search for incoming wireless transmissions.

The loop holds two functions. A function that manages the Bluetooth connection and one that manages the internet connection.

```c
void loop(){
  if(loops == 0){
    Bluetooth_loop();
  }else{
    Internet_loop();
  }
}
```

Fig. 5.27 Microcontroller Loop

As the variable “loops” is initialized with the value “0”, once the loop function is called, the Bluetooth method is executed.

This function carries out the Bluetooth connection and its communication. It also attempts to connect the microcontroller to the local Wi-Fi network.

Once the microcontroller is successfully connected to the local Wi-Fi network, the variable “loops” is set to “1”, and the internet loop is called.

The internet loop manages the internet communication, the state of the components and creates the HTML file.
6. CONCLUSION

Finally, to conclude the project, the final thoughts and conclusions will be discussed.

The designed domotics project, is cheap, simple to configure and easy to use. It obviously cannot compete against international companies like Apple or Google, as I am neither an expert programmer nor do I have a team of specialists working with me. Nevertheless, I am very proud of the obtained result.

Despite its restrictions, it is a complete app. It implements functionalities beyond the control of components making the app more user-friendly and covering basic needs like security or appliance redundancy.

Regarding the basic function of the app, the wireless internet communication with the microcontroller, it has limitless applications, as after a quick study on how a certain electro-domestic works the microcontroller can be designed to adjust its digital I/O according to the electro-domestic input demands. Summarizing, the fundamental functionality is the wireless internet communication. Once communication is achieved, it is a question of deciding what to send and how the microcontroller should react to the sent instructions to control the attached components.

During the development of the project I encountered many, many issues. Problems with the phone app, problems with the Bluetooth connection, the internet connection, the database and so on. It is only normal to encounter so many problems in a programing project, especially if the developer is new to the languages used in the project.

One of the biggest inconveniences I came across was security. As the project consists of data transference across the internet, securing that data transfer is an ever-present problem when dealing with this type of communication. Fortunately, as I intend to apply my project in my own home, the microcontroller will only communicate with devices in the same network. However, this limits the domotics system as it cannot be managed from other networks. In order to expand the communication across other networks it is necessary to route the microcontrollers IP address. As the request has to be port-forwarded, the port the microcontroller uses must be changed and the router configured to forward the web request to said port. Depending on what the router supports third parties may be needed or not.

Once the microcontroller can be accessed from anywhere, the safest, easiest security measure is to encrypt the URL, making it harder for people to read the data transference.

Regarding the personal experience of developing this project, it is by far the most complex, complete, entertaining and challenging project I have ever taken on. It requires five programming languages of which I had little previous knowledge, basic knowledge of wireless communications, basic database knowledge and basic analog circuit knowledge. Summarizing, it drove me to learn new concepts and new programming languages which I have always had an innate curiosity for.

Also, as I encountered so many problems and difficulties, as one usually does whilst programming, it forced me to explore many different ways of achieving the same goals, broadening my knowledge of the subjects at hand.
As for future developments, I intend to use and extend the developed project in the near future.

6.1. Future developments

As stated beforehand, the fundamental functionality is the wireless communication between phone application and microcontroller. Once this is established, any number and type of components can be connected to the digital I/O ports of the microcontroller and managed from the phone app through the internet.

As I intend to personally install this project in my home, in time, I will add new components to the source code as to better manage my electro-domestics.

Aside from adding new components, I also intend to add new functionalities. The functionalities help the user better manage the components and the information in the app itself.

Finally, another interesting ampliation is security measures. As cybersecurity is a vast and complicated field, I was reluctant to study it for this project as I already had so much new material to learn. In time, however, I intend to learn about the subject and implement it in the communications between the phone app and the microcontroller.
BIBLIOGRAPHY


[5] https://ipfs.io/ipfs/QmXoypizjW3WknFiJnKLwHCnL72vedxjQkDDP1mXWo6uco/wiki/Honeywell_316.html Consulted the 11th of September 2019


[9] https://store.google.com/category/connected_home Consulted the 21th of August 2019


[17] https://www.itead.cc/wiki/HC06_Serial_Blueooth_Brick Consulted the 15th of April 2019
[18] https://www.amazon.de/s?k=hc05&__mk_de_DE=%C3%85M%C3%95%C3%91&ref=nb_sb_noss_2 Consulted the 15th of April 2019

[19] https://www.amazon.de/s?k=hc06&__mk_de_DE=%C3%85M%C3%95%C3%91&ref=nb_sb_noss Consulted the 15th of April 2019

[20] https://store.arduino.cc/arduino-wifi-shield Consulted the 17th of September 2019


[22] https://programarfacil.com/podcast/esp8266-wifi-coste-arduino/ Consulted the 17th of September 2019

[23] https://www.amazon.es/ESP8266-ESP-01-M%C3%B3dulo-transceptor-inal%C3%A1mbico/dp/B075H77Z3J/ref=sr_1_12_sspa?keywords=esp8266&qid=1568728784&s=gateway&sr=8-12-spons&psc=1&spLa=ZW5jcnlwdGVkUXVhbGlmaWVyPUNFLN1FEWIQ4OFBJMFYmZW5jcnlwGVkSWQ9QTA3NTAyOTkxQ080Mj0yWTQzYQzlJmVuY3J5cHRIZFkSWQ9QTA4MTMxMTA5RFPgSVQ4NFZNE85JndpZGdldE5hbWU9c3BfXRMJmFdGlvbj1jbGljaWZpZW50JmRvTm90TG9nQ2xpY2s9dHJ1ZQ== Consulted the 17th of September 2019

[24] https://store.arduino.cc/arduino-unorev3 Consulted the 15th of April 2019

[25] https://store.arduino.cc/arduino-nano Consulted the 15th of April 2019

[26] https://store.arduino.cc/mega-2560-r3 Consulted the 15th of April 2019

[27] https://www.raspberrypi.org/blog/raspberry-pi-4-on-sale-now-from-35/ Consulted the 15th of April 2019


[31] https://lastminuteengineers.com/esp8266-nodemcu-arduino-tutorial/ Consulted the 19th of April 2019

33] https://developer.android.com/studio/intro/?gclid=EAIaIQobChMIwoLc3PHk5AIYWVCh0iXQPMEAAAYASABEgJofvD_BwE Consulted the 6th of May 2019
34] https://developer.android.com/studio/?&gclid=CjwKCAjw-vjqBRA6EiwAe8TCk68cf93EM85mav7PypPyEzQXfogHjtUYYou6hpZz1V7F0hEfx1ezshoCF4UQAvD_BwE Consulted the 2nd of July 2019
37] https://appinventor.mit.edu/about-us Consulted the 15th of April 2019
38] https://www.eclipse.org/ Consulted the 15th of April 2019
39] https://www.tutorialspoint.com/android/android_eclipse.htm Consulted the 15th of April 2019
42] https://techterms.com/definition/assembler Consulted the 24th of April 2019
43] https://techterms.com/definition/assembly_language Consulted the 2nd of March 2019
44] https://getcodecoach.com/what-is-scratch-programming-language/ Consulted the 15th of April 2019
45] https://www.python.org/doc/essays/blurb/ Consulted the 15th of April 2019
46] https://kb.iu.edu/d/agsz Consulted the 15th of April 2019
47] https://www.javatpoint.com/general-purpose-programming-language Consulted the 19th of April 2019
48] https://www.techopedia.com/definition/25106/android-operating-system Consulted the 7th of July 2019
49] https://gs.statcounter.com/os-market-share/mobile/worldwide Consulted the 10th of July 2019
50] https://www.lifewire.com/what-is-ios-1994355 Consulted the 10th of July 2019
51] https://www.makeuseof.com/tag/what-is-kaios-mobile-os/ Consulted the 17th of August 2019
52] https://www.kaiostech.com/ Consulted the 17th of August 2019
53] https://www.tutorialsteacher.com/mvc/mvc-architecture Consulted the 15th of April 2019
54] https://www.lifewire.com/what-is-bluetooth-2377412 Consulted the 18th of April 2019

[56] https://www.techopedia.com/definition/1352/uniform-resource-locator-url Consulted the 21st of June 2019

[57] https://www.yourhtmlsource.com/starthere/whatishtml.html Consulted the 22nd of June 2019

[58] https://techerms.com/definition/markup_language Consulted 22th of June 2019

[59] https://square.github.io/okhttp/ Consulted the 4th of September 2019

[60] https://www.w3schools.com/sql/sql_intro.asp Consulted the 10th of July 2019

[61] https://www.martinfowler.com/books/dsl.html Consulted the 10th of July 2019


[63] https://appinventor.mit.edu/explore/ai2/paintpot-part2 Consulted the 15th of September 2019


[66] https://source.android.com/setup/start/build-numbers Consulted the 38th of August 2019

[67] https://www.devteam.space/blog/how-to-convert-an-android-app-to-ios/ Consulted the 15th of September 2019

[68] https://www.abc.es/tecnologia/consultorio/20150216/abci--201502132105.html Consulted the 15th of September 2019

[69] https://developer.android.com/docs Consulted the 2nd of May 2019

[70] https://whatis.techtarget.com/definition/Web-server Consulted the 20th of August 2019