IoT Lab Box

André António Feliciano Santos

Dissertation to obtain Master's degree in

Telecommunications and Informatics Engineering

Supervisor(s): Prof. Alberto Manuel Ramos da Cunha

November 2021

Abstract

Every day, new technologies emerge. New technologies are created and very often to make our lives simpler and easier. The possibility of having everything connected around us is something that generates interest in many because it allows having a single device where we can have control of all the other devices in a very easy-going way. New terms, related to technology keep on surging every day and the IoT one is currently a term that we are getting used to more and more and as the days go by it its popularity keeps on increasing.

IoT, Internet of Things, is the term used to describe a network composed of physical objects ("things"), that are embedded with sensors, software, and other technologies to connect everything around us, by exchanging data between services and systems over the internet. This project that is going to be developed, has its focus on this new world and focuses on exploring the Internet of Things by creating a system where multiple physical sensors are connected to one or more Arduinos and devices like the ESP32, giving us a more accurate and clearer perception on how we can perceive the environment surrounding us using these devices to our advantage.

To develop these kinds of projects, having a physical place to work is key. Since we are working with software and hardware, having a physical location allows for assembling sensors and actuators in a local and fixed place while at the same time can map the environment where these are located. A project involving hardware benefits from this, because moving around all the physical devices can be very time-consuming and increases the chances of damaging the materials. Nevertheless, also having the possibility of building a mobile project is important, since it allows us to map more environments and interact with different objects, enriching our experience.

The "IoT Lab Box" is a cyber-physical system where the software and hardware world are put together, creating a unified system working as one. By connecting multiple Arduinos devices and assembling sensors and actuators to it, the goal of perceiving the environment around us can be achieved. By providing wireless capabilities to this box, like Bluetooth and WIFI, this project becomes optimal for learning more about the IoT world, wherever we are. This "box" can be either imaginary or an actual physical box, where all the materials are concentrated in one place. By having a system designed like this, we can place a box in a fixed location, like a laboratory, creating a sort of intelligent room, or we can move the box to another place like a studying room, allowing multiple people to experience this project in a way that they desire. With the help of software to analyze all the data put out by the hardware, we will create an intelligent system, a system belonging to the Internet of Things.

Keywords

Technology; IoT; Internet of Things; physical objects; sensors; software; internet; Arduino; ESP32;

Resumo

Todos os dias, novas tecnologias surgem. Novas tecnologias são criadas e muitas vezes com o objetivo de tornar as nossas vidas mais simples e fáceis. A possibilidade de ter tudo ligado à nossa volta é algo que gera interesse em muitos, porque permite ter um único dispositivo onde podemos ter o controlo de todos os outros de uma forma muito fácil. Novos termos, relacionados com a tecnologia continuam a surgir todos os dias e o da IoT, é, atualmente, um termo a que nos estamos a habituar cada vez mais e à medida que os dias passam a sua popularidade continua a aumentar.

IoT, Internet das Coisas, é o termo utilizado para descrever uma rede composta por objetos físicos ("coisas"), que estão incorporados com sensores, software, e outras tecnologias com o objetivo de ligar tudo à nossa volta, através de uma transferência de dados entre serviços e sistemas através da Internet. Este projeto que vai ser desenvolvido, tem o seu foco neste novo mundo e centra-se na exploração da Internet das Coisas, criando um sistema onde múltiplos sensores físicos são ligados a um ou mais Arduinos e dispositivos como o ESP32, dando-nos uma perceção mais precisa e clara sobre como podemos perceber o ambiente que nos rodeia, utilizando estes dispositivos em nosso proveito.

Para desenvolver este tipo de projetos, ter um local físico para trabalhar é fundamental. Uma vez que estamos a trabalhar com software e hardware, ter uma localização física permite a montagem de sensores e atuadores num local fixo e ao mesmo tempo podemos mapear o ambiente onde estes estão localizados. Um projeto que envolve hardware beneficia disto, porque a deslocação de todos os dispositivos físicos pode ser muito demorada e aumenta as hipóteses de danificar os materiais. No entanto, ter também a possibilidade de construir um projeto móvel é importante, uma vez que nos permite mapear mais ambientes e interagir com diferentes objetos, enriquecendo a nossa experiência.

A "IoT Lab Box" é um sistema ciberfísico onde o mundo do software e hardware são reunidos, criando um sistema unificado que funciona como um só. Ao ligar vários dispositivos Arduino entre si e ao montar sensores e atuadores, o objetivo de recolher dados sobre o ambiente que nos rodeia pode ser alcançado. Ao fornecer capacidades sem fios a esta caixa, como Bluetooth e WIFI, este projeto torna-se ideal para aprender mais sobre o mundo IoT, onde quer que estejamos. Esta "caixa" pode ser imaginária ou uma caixa física real, onde todos os materiais estão concentrados num único local. Tendo um sistema concebido desta forma, podemos colocar uma caixa num local fixo, como um laboratório, criando uma espécie de sala inteligente, ou podemos mover a caixa para outro local como uma sala de estudo, dando a oportunidade a múltiplas pessoas de experimentar este projeto da forma que desejarem. Com a ajuda de software para analisar todos os dados disponibilizados pelo hardware, vamos criar um sistema inteligente, um sistema que pertence à Internet das Coisas.

Table of Contents

List of figures

List of Tables

List of Acronyms

ACIC	Applications and Computation for the Internet of Things
AI	Ambient Intelligence
APK	Android Package
BLE	Bluetooth Low Energy
CMU	Mobile and Ubiquitous Computing
DIIC	Internet of Things Interaction Design
EUI	Extended Unique Identifier
GAP	Generic Access Profile
GATT	Generic Attribute Profile
I2C	Inter-Integrated Circuit
IDE	Integrated Development Environments
IP	Internet Protocol
IST	Instituto Superior Técnico
IoT	Internet of Things
LED	Light-Emitting Diode
MAC	Media Access Control
MQTT	Message Queuing Telemetry Transport
OS	Operating System
UART	Universal Asynchronous Receiver Transmitter
UUID	Universally Unique Identifier

1. Introduction

Since the beginning of our time, technology has played a very important role in the development of civilizations and the development of people in general. Nowadays it is something so present in our lives that we don’t even really think about it. Whether is a very simple thing or a very complicated one, technology is everywhere we go and, in many cases, contributes to a better and simpler life. New terms, related to this technology concept appear more and more often and the term IoT is one of the most important ones by being constantly around us. IoT means the Internet of Things. It is the term that describes the network composed of physical objects (“things”), that can be embedded with sensors, software, and other technologies to connect everything around us, by exchanging data between services and systems over the internet. The initial idea of this term, a network of smart devices, was addressed as early as 1982, with an updated Coca-Cola vending machine at Carnegie Mellon University being the first Internet-connected appliance, able to report its inventory and whether recently loaded beverages were cold or not. The most common association of Internet of Things, and the interconnection between all the devices, many times is, for example, a smart home, where everything is connected to a single device, like a smartphone, and with that, we can control everything and see what is happening with all our devices. But it goes much deeper than this. Nowadays, self-driving vehicles are more and more popular and, all the bigger brands in the automotive industry are all competing to build the ultimate autonomous vehicle. This is a very good example of IoT. The possibility to connect all the infrastructure present, like roads, traffic signals, and lights to the vehicle is one of the most important challenges nowadays. But to have all of this, we must have a device or multiple devices capable of this connection and coordination between all of this.

The Arduino is an open-source hardware and software company that designs single-board microcontrollers and microcontroller kits for building digital devices. These boards, use a variety of microprocessors and controllers, and by connecting these devices to the internet, and the devices that we are trying to connect to it, we can use these boards to act as the brain of our entire operation. By acting as the brain of the system and processing data from the sensors and devices connected to it, it can control what we need. Since the Arduino board is an open-source tool, we can code it, using its specific language, C/C++, to create features suiting our needs. It has its IDE, and after the coding process, the upload process to the board is as simple as pressing a button. One of the great advantages of these boards is the fact that they can be programmed “n” number of times, allowing the creation of several IoT projects by simply changing the code and not having the need to have a completely different device to serve as our brain.

Combining these two worlds, the physical sensors, and the software, allows us to gain control of devices in our home. A clear example of an application of the IoT. This concept simplifies our lives in a very good way. Whether is to turn ON/OFF a light present in another division, or to assure us that we turned OFF all of our electronic devices when we go on vacations or the ability to control the power consumption of our gadgets, giving us the possibility to see what are the most demanding electronic devices regarding power consumption and helping us, reduce the power bill at the end of the month, IoT is something that we all need and it’s here to stay.

Because of all this, a project called, IoT lab Box was purposed, aiming at integrating cyber-physical interfaces and by exploring wireless connectivity we can explore even further this world by connecting the box to the Internet or by using Bluetooth interfaces.

This project benefits greatly from a physical location to work with, like a laboratory, but is not restricted to it. This box can be perceived as an actual box or an imaginary one, meaning that we do not need a box to put the devices inside. By having them in a workstation, like a laboratory, it is easier to assemble all the components to map the environment, knowing that they will always be there. By creating a room, where this possibility is explored, we can design a place with multiple sensors and actuators incorporated in this location, giving the possibility to students and others to have a place to develop this work with all the materials necessary always available. On the other side, by having all the components placed in a box, we now have the mobility to transport the components to where we want, making every location an intelligent one, if this project is present. From an educational point of view, this is a great advantage since the people who want to use this box are not limited to a specific location to interact with it. This “box”, as said, by having multiple Arduinos connected will act as the brain of our IoT device, outputting values and information and with the help of software, these values can be seen and analyzed, making us aware of what is going on around us. The brain of the box is made by devices like the Arduino Uno and devices like the ESP32. These devices are very similar regarding the way that they are mounted and configured but have hardware differences that may suit different purposes. In this project, we will explore them and pointing the advantages and disadvantages of both. The project will be executed on both devices, giving more information to the user on what device to use and why.

1.1. Objectives

The main goal of this project is the educational one and to accomplish that, we must integrate this box into courses that can use it to teach something to the students. The educational one is not complete only by making this project available and understandable to students who have a background in these subjects. It also involves those who want to learn more about these systems, even if they do not have a background, this project must be open and understandable to everyone. At IST, Instituto Superior Técnico, we have different courses that focus on the IoT area. The ACIC course, Application and Computation for the Internet of Things[1], is one of the courses that use the Arduino platform in laboratory experiments [1], introducing it to the students. In the following Chapter 2, more about this will be explored and explained on how this course uses this platform to teach students subjects regarding cyber-physical interfaces.

[1] https://fenix.tecnico.ulisboa.pt/disciplinas/ACPIC7/2020-2021/1-semestre (Accessed 10/2020)

The Ambient Intelligence (AI)[2] [2] course presents to the students the concept of AI and the use of Information Technology (IT) systems to monitor, control, and interact with environments frequented by people, including Smart Cities, Smart Homes, IoT, and related technology. In the following Chapter 2, we will explain how does the course work and how can this project help the students who study it.

The Internet of Things Interaction Design[3] (DIIC) [3] course, also deals with the subject of IoT but it focuses more on the design of the systems. However, despite the main subject of study being the design of these systems, students also study and learn how the hardware and software present is used to build these systems. How this project will help and can be used by the students to be applied at this course will be explained in the following Chapter 2.

The Mobile and Ubiquitous Computing[4] (CMU) course [4], makes the students aware and makes them understand the fundamental challenges and problems underlying the design and development of software (middleware and operating system) supporting applications for mobile and ubiquitous scenarios (users, hardware, software). Design, specify, analyze, and implement software systems (mobile/ubiquitous middleware and operating system) that can support mobile/ubiquitous applications. This project can have useful applications to the students attending this course and in the following Chapter 2, we will learn how.

However, as said, this box can be used by other people than just the students enrolled in this type of course. Since this box, is a combination of software interfaces and mobile hardware, this box can be used by students outside the typical courses that use these types of materials to increase their knowledge in these areas. Providing a location for this box in a specific place and at the same time having the possibility for it to be requested by all students that desire and want to develop new skills outside its fixed location or to study this type of subject is a priority giving the ability to the IoT lab box to be carried anywhere they want. For example, if we had one or two examples of this box in a library or a study room, any student could have access to it and practice their skills and develop a new passion for this field, simply by using it. The restrictions of the use of this box by only the students enrolled in the electronic or software courses doesn't make any sense and we must work to get every student interest in this, enriching their knowledge and presenting them a different opportunity and way of learning.

1.2. Organization of the Document

The following document is organized as follows: Chapter 2 refers to the State of the art and the related work where we can learn more about IST courses and other examples where students and universities developed projects in the IoT field. This provides an idea of where this project is to be used and how can we explore similar work to add and complement our own. Chapter 3 refers to the Architecture where we can learn about the requirements for this project, and how does each component work, and how is

[2] https://fenix.tecnico.ulisboa.pt/disciplinas/AI514/2019-2020/2-semestre (Accessed 10/2020)
[3] https://fenix.tecnico.ulisboa.pt/disciplinas/DIIC13/2020-2021/1-semestre (Accessed 10/2020)
[4] https://fenix.tecnico.ulisboa.pt/disciplinas/CMov4/2019-2020/2-semestre (Accessed 10/2020)

going to be used. Chapter 4 refers to the Implementation to show how the project is going to be implemented and where all the work produced is explained. Chapter 5 refers to the Evaluation of the project. Chapter 6 refers to the conclusion and future work.

2. State of the Art

Since this project is going to be developed using the Arduino platform, it is important to explore how the Arduino is used in the educational context with similar or related applications to what we are trying to achieve with the construction of this project.

This project has an educational goal, and the development of this project focuses on the IoT world and most importantly how can a project regarding this subject have a positive impact in the laboratory classes at IST and autonomously by the students.

We start by considering IST courses where this platform can be used and where all the technologies present in the box can fit and be seen as an advantage. The use of the Bluetooth and WIFI technologies is something that it's not very common to use at IST but it is used by other universities and for that reason, it is important to give some context regarding this specific application of these wireless capabilities. Wireless technologies play a major part in the IoT world and, basically in all the devices that all the users use daily. The pandemic context we are now living in, begs even more for the implementation of technologies that require almost no contact for the safety of all the users and the safety of everyone.

2.1. IST Context

In section 1.1 of the Introduction Chapter, we talked about courses at IST that could benefit from using this project. Developing this project and at the same time having the possibility to adapt it to serve the needs of several courses is a determinant factor for the evaluation of this work. Now we must understand why.

In the ACIC course, through laboratory guides, most of the students have first contact with the Arduino world. The students are prompted to assemble sensors and actuators and create features and implement code into the Arduino Uno, being evaluated by demonstrating their project/laboratory behavior. The opportunity to adapt this box, giving the students the possibility of using a project like this, the IoT Lab Box, to take advantage of it and improve their work, developing more complex and interesting laboratories and projects is a keen factor in deciding whether the project has value or not.

In Ambient Intelligence, by the end of the course, students must develop a project where they must monitor an environment and how it reacts to changes applied to him. Some students choose to use the Arduino platform to develop their project and this box might be something that provides them an advantage to the development of their project, whether by helping them or by motivating them to use this box to develop a more complex project.

On the Internet of Things Interaction Design course, in their final project, students must develop interactive prototypes, combining hardware and software, using an IoT solution. There is a set of hardware that they can choose from, including the Arduino Starter Kit. This kit is also used during laboratory classes to do the exercises. Since this project is all about connectivity and IoT, having this project available will be something of their ultimate interest.

In the Mobile and Ubiquitous Computing course, students must develop a project where they create an android app. This type of box could be interesting for them, by having all these cyber-physical sensors available, if their project requires some and since this box will be possible to control via an android app, we put the two worlds together and if they wish, they have a system ready for them to use, to adapt and to interact.

Table 1 - IoT Lab Box applied to IST courses

Course	Objective of course	Advantage of IoT Lab Box
Application and Computation for the Internet of Things	First contact with the Arduino world. Assembling physical hardware and coding it.	Having a base for their projects with hardware and software previously developed to expand their projects and achieve new goals.
Ambient Intelligence	Develop a project where they must monitor an environment and how it reacts to changes applied to him.	The box is assembled with sensors that respond to the changes in the environment where they are inserted. Provides a base for the students for the development of their project with already pre-built features.
Internet of Things Interaction Design	Develop interactive prototypes, combining hardware and software, using an IoT solution.	Build their prototypes to see if the conceptual project that they built works in the real world.
Mobile Ubiquitous Computing	Develop a project where they create an android app.	Develop communication between the android app and the IoT Lab box to explore the limitations of their app and to map the environment within the app.

All the courses previously described, are courses where the students already have a background in software or hardware or both. By having this background, combining the IoT Lab box with their courses and knowledge is easier. But, what about the students who do not have such a specific background? While developing this project, this is something to look out for.

Nowadays, the interest in the IoT world keeps on increasing and many people who do not study these types of subjects have also grown an interest in this area. This belief is reflected in the construction of this IoT Lab box project. Whether is the use of simple the possibilities of the box or to explore, even more, this box can be adapted and used by someone who has a background in software and hardware or simply by someone who wishes to learn more and use this box as an entrance to this new world.

This box is a collection of hardware devices in combination with software to create a seamless system. By having all the code and all the assembling processes documented, those who don't have a background can learn easily and understand what has been made and have the opportunity of working with a system already built, ready for action, and to explore how everything works.

2.2. Related Work

In the field of education, Arduino technology has been used in many contexts, through various institutions and organizations. This segment introduces some of them with a view of getting a deeper understanding of how the platform is used and what functionalities are useful in this sense. Since this project is going to be used in courses and students attending IST, this organization was first approached in the previous section, providing us with some background. Previously, courses, where this project could be interesting at IST, were mentioned. This box can be used by more people than just the students enrolled in these types of courses. Since this box, is a combination of software interfaces and mobile hardware, this box can be in a specific location and can be requested by all the students that desire and want to develop new skills or to study this type of subject and carry this IoT lab box to anywhere that they want. In the next topics, we will describe and provide some context, as said, to where and how these types of projects are applied and how they can contribute to the knowledge and enrichment of the students that use them.

University College of Southeast Norway - Electric Engineers students at this university [5] used to learn the modules by programming in C/C++ in one semester and in the following one used to program in assembly and C PIC18 microcontrollers. Most of the work was accomplished by using simulators since the use of real microcontrollers took too much time from the students and some of them found it difficult, making the motivation of the students poor, leading to some students giving up the course and making the results poor.

In 2013, the program of these courses was revised, and the learning method was replaced by the Arduino platform. The motivational part that regards the students was easy and improved because now the students can visualize what they are programming. The first part of the semester consists of lectures and practical assignments. The second part consists of practical projects that students must complete and present at the end of the semester. These practical projects also replaced the written exams.

The teaching in the modules is based around weekly lectures following a standard programming teaching course where materials are covered, and example of code is shown. This new setup for the course has been evaluated by the students several times. The students are much more positive about the course, and they give positive feedback to the staff continually. It is shown that students put more effort into the course and the grades reflect exactly that. A survey took place and showed that students are happier and as said they felt more motivated and interested in learning these subjects.

This previous example is one of the most important ones since it shows that this platform helps the students and motivates them, going according to the most important objectives of this project. The educational objective is achieved and the satisfaction by the students proves that.

University de Trás os Montes e Alto Douro - In this university, specifically at the masters in electronic engineering [6], because of everything that it’s happening in the world, the COVID situation, they developed a mobile laboratory to ensure that the students can keep working and learning at home with the Arduino platform, by using Arduino kits. The students conduct experiments, followed by the staff via the internet, and at the end of the semester, students will make a public presentation about the results and discoveries found throughout the semester remotely. The course responsible states that even doe these mobile experimental kits are mainly used by students of electronic engineering, this low-cost kit that allows students to make experiments at home and have a more hands-on approach, is an improvement and an add-on to their academic education.

University “Politehnica” Timisoara, Romania - At this university [7], during two different programming courses, they designed and implemented a system to monitor a group of selected environment parameters: humidity, temperature, light intensity, and methane. This brings benefits to the students by making them more connected to the subjects and allowing them to see what they are programming and designing. Moisture and temperature sensors were placed near the roots of the plants, working together with an automation system to create a wireless network capable of controlling the irrigation system. A feature was also developed allowing the system to warn via SMS, the farmers if anything changes. The real purpose of this system proposed to the students is to engage the students in taking remote measurements via mobile devices. This, of course, brings educational benefits. Designing the monitoring system, programming for LabVIEW and Arduino, understanding these concepts, and using mobile devices for this purpose. This type of approach makes the students more involved in the project, challenging them to achieve more and more and always keeping them motivated to see the results of all their hard work.

Miami university - At this university [8], students who attend courses in Electric and computer engineering have the choice and opportunity to use the Arduinos the create their projects as the controlling device. The students also have the freedom to decide what to build with this platform. Over 90% of the students choose to use the Arduino for their midterm and final exams. The main teaching topics of these courses are understanding pin limits, cost, and time for an alarm clock project. Many students design an operational alarm clock and prototype it even doe it was not required and the justification they gave it was that they just wanted to see the system work, proving once again the interest and captivation that this platform offers to the students. It is very interesting to see how a low-cost and a rather simple device, like the Arduino can be so important for the students and how they choose willingly to build the full project, just to see it work.

Lodz University of Technology, Lodz, Poland - At this university [9], the challenge of teaching students programming courses more engagingly and less theoretical was purposed. Programming classes are a very good example in which we can all try to improve this since usually, classes take the shape of written console programs, which are not very enticing because of the lack of graphics and lack of interest. The problem with the participation of students in such courses is that writing console programs provides no opportunities to develop new programs in person. Given the ubiquity of technology, students are urged to select IT majors. For that reason, two courses were chosen to improve this situation, Introduction to Programming and Intelligent Autonomous Systems. These courses were chosen to use the Arduino platform, so that the students could develop their projects using a different approach and for the students to observe physical effects resulting from the code that they were asked to produce. After all of this, a questionary was presented to the students allowing the faculty teachers to understand if this approach was indeed viable or not. The results obtained, indicate that the assumed approach brought a quite positive outcome. The students liked the classes and would like to continue studying similar courses. The use of Arduinos raised their interest in the subjects taught and it was easier for the students to understand the material taught and used. Since the students could see the tangible results of their work, their motivation to learn was also improved.

Johns Hopkins University, Baltimore, Maryland - In this university [10], in the Laboratory of Computational Sensing and Robotics it was purposed to the students attending the courses to develop their laboratory experiments using a new approach, a low-cost one. The proposed experiment kit uses simple Arduino controllers with MATLAB/Simulink interface. The setup can be programmed via Simulink easily and doesn’t take the focus of the students with the micro-controller programming. An inquiry was carried throughout the students, and it showed that the purposed setup was easier with plug-and-play features due to Simulink. Students found that relating theory to the experiments was easier.

Another example, that the low-cost approach of using Arduino kits is well received by the students and the feedback shows that exactly that. And not forgetting, universities have a lot of students attending the courses, so this approach is economically friendly for the universities and goes accordingly with the students’ opinions and thoughts.

Columbia University - At this university [11], a lab kit was introduced, based on an Arduino microcontroller board and open hardware, to allow students to use low-cost, training-specific hardware to complete lab exercises at home. The selection of hardware was deliberately made to allow the adjustment and adaptation of current laboratory exercises to be used for this new platform and, by doing so, this new platform enables the development of new exercises that would not be feasible by using a conventional means of solving laboratory exercises. Since colleges are turning more and more to online courses, laboratory classes and exercises have become a challenge, and the adoption of distributed lab kits can be promoted using free hardware and software. These lab kits had to be available to students with little knowledge, but they had to be strong enough to solve laboratory exercises. As has been said, the original laboratory guides have been updated to suit the current hardware constraints and to encourage students who work from home to solve these guides. New exercises were also introduced to take advantage of the lack of time constraints. These activities may be constrained by the availability and expense of hardware, but cost-effective and instructive exercises can be planned. Since the interface is accessible, students and teachers can continually change and develop the lab kits. This platform is highly promising for distance learning courses and can quickly be extended to other courses.

The following project is called the “Lab-in-a-box”, a name very similar to the project to be developed. This project aims to an improvement regarding teaching students signal processing and analog electronics whether inside or outside of the classroom. While this project is not entirely like the project that is going to be developed, it has similarities and it’s interesting to understand how multiple Arduinos can work together to create a new way of learning for the students and at the same time be a low-cost approach to them. It has the same core principles as this project, the use of multiple Arduinos to improve the educational variant present in the daily lives of the students using it or for those who want to improve their knowledge and understand better some of these subjects.

Stanford University, Stanford - In this university, a project called “Lab-in-a-box” [12] was developed to act as a low-cost approach that combines hardware and software for teaching signal processing and analog electronics. It has the goal to improve the teaching of these concepts by providing a different type of platform that is more understandable to the students and by being a low-cost approach and breaking economic barriers to the students that study these types of courses. Currently, the use of signal processing principles is also taught by programming languages such as MATLAB and Python. This method might be less appealing to students and for that reason, a different approach was deemed appropriate. This project is composed of three boards, each one with a different teaching purpose. The Lab-In-A-Box makes the transition from a leisure student to a student easier with a cheap, robust, and accessible collection of tools that connect the system's students are using today with high-fidelity hardware and software interfaces.

Being the IoT world, a world with so many opportunities and endless new projects more complicated ones appear with the possibility to be applied on a larger scale. The next university is an example of exactly this. The next university proposed a future project that could be applied to an entire university or just a small part of it, called Smart University.

University Politehnica of Bucharest, Bucharest, Romania - In this university a concept was purposed called Smart University [13]. Since nowadays IoT has become a reality in our lives and since more and more objects seem to be interconnected, the possibility to connect everything is real. This concept is like a small world, where with the presence of sensors and the help of networked devices all connected, we can create a new environment. This is a real possibility and has been schematized to be built. All universities are connected to the internet and in these universities, numerous objects exist that can be converted to smart things, integrating them into the IoT category. Objects like doors, windows, printers, projectors, books and more, can all be connected to the internet creating a networked environment where everything is connected and can be accessible. By having all these connections, we create a Smart University and advantages can be taken from this like the monitoring flows of people within the campus; the control of traffic; tracking parking spots; increase prevention of accidents and disasters by controlling aspects of noise, weather, humidity, smoke, and light; reducing the electric consumption and much more. By placing several types of sensors within the campus and spreading them out, we can build a network that interconnects all this and analyses all the information that these sensors are outputting. This creates a Smart University. This model can be used as a whole, or also in small parts, based on the need that the university has. This is just another example of how the IoT area can be extremely helpful in our daily lives and can improve our living quality and make life easier for everyone.

Since this project aims at using different types of interfaces for the communication between the Arduinos and between the Arduino and the user, it is important to understand more about the interfaces that are going to be used. Bluetooth is one of the most important interfaces feature available in this project and for that reason, the following projects are presented to give some context and some examples on how we can use Bluetooth and what kind of projects and functionalities we can have by using it. The use of this interface can have multiple applications, from controlling some IoT objects in a room to creating a life-saving device.

Wayne State University, Detroit - In this university [14], a project was developed with the Arduino and a Bluetooth module to implement remote functionalities of various outputs. The goal of this project is that by using an Arduino and a Bluetooth module and using an android app as its interface, it is possible to control a fan, turning it ON or OFF and a LED strip. By doing this, they expect that the students gain a better understanding of how Bluetooth communications work. They developed a guide for the project, with several steps, where all the functionalities were divided and organized step by step. They evaluated this project as successful, and they would like also to be able in the future to improve the system and try to achieve more using this platform.

University of Engineering & Management, Kolkata, India - In this university at the Department of Electric Engineering [15], with the use of an Arduino Uno and a Bluetooth module, a project where they could control in real-time the power consumption of household electric appliances and measure it, was purposed. To see all of this, an android app is used to display the output values of the Arduino This app by having all the power consumption would also make the user aware of their electrical bill. Their goal is to reduce the maximum demand power consumed by the user, making the bill cheaper for him. By using this platform, they were able to use the Arduino to optimize the schedule where the power consumption is higher so that the differences between average load and maximum demand, would not be so large. This project gives important information to the users by informing them regarding the status of their power consumption, giving the user the ability to better manage their activities that involve the use of power energy and at the same time calculating the bill and giving the information necessary to reduce it.

University of Chittagong, Bangladesh [16] A prototype was developed to monitor body temperature and heart rate. This prototype uses Bluetooth to connect to an android smartphone and its app, making it possible to visualize the results gathered by the sensors, serving as a display. The sensors are connected to a microcontroller, and the microcontroller is connected to the android app via Bluetooth. The values outputted by the microcontroller are then sent via WIFI to the cloud to be stored and analyzed in a database and can be accessed by authorized viewers. In this project, a proper health monitoring system was tried to achieve via an IoT system. The use of this project is directed at rural areas or countries, where real medical care is difficult to obtain and is expensive. By having this low-cost approach and prototype, they tried to, somehow, help in this field, trying to cover up this deficiency present in our days. This system can be expanded to support features like blood pressure and real-time ECG or EEG monitoring. Making this platform user-friendly and cost-effective is always a priority as is the goal to save a life.

University of Hradec Kralove, Hradec Kralove, Czech Republic - In this university, a project called “ChemDuino” [17] was created to improve chemistry teaching and learning. The name of the project is the combination of “chemistry” and “Arduino”. This device can be used as a demonstration measuring device, as a computer, and as a basis for students to make measurements. Since the hardware used for the construction of this kind of project is inexpensive, students can easily create the demonstration device that their teacher wants fast and at a reasonable price, being substantially lower than commercial suppliers' price of similar equipment.

University of Elche, Elche, Spain [18] **-** Microcontrollers have gained a great deal of popularity over the past few years as have visual programming and platforms. The idea of Bring Your Own Device (BOYD) policies implemented at various phases of learning enables the use of laptops or tablets along with home or classroom facilities that give new educational opportunities. This led to the development of a project in which electronic hardware could be developed for the Arduino panels and visual software for education purposes at two separate educational levels. It was created an Advanced Multimeter Shield to cover a necessity in the power electronics lab at this university. By using an Arduino shield and a computer together, it is possible to see the outputted values by the Arduino.

It was also developed a second project, where they designed and implemented complete sensor shields with integrated measurement electronics to provide low-cost hardware for different universities or secondary schools level courses to use. The overarching purpose of this project is to encourage the development and usage of low-cost Arduino compatible shields at various levels of education, whether in early secondary schools or universities.

Table 2 - Key features of these Arduino projects

Project	Key Feature
University College of Southeast Norway	The use of the Arduino motivated the students.
University de Trás-os-Montes e Alto Douro	Created a mobile laboratory allowing the students to work from home.
University "Politehnica" Timisoara, Romania	Take remote measurements via mobile devices.
Miami University	The ability to use Arduino to control their projects and have the freedom to choose what to build.
Lodz University of Technology, Lodz, Poland	The use of Arduinos raised their interest in the subject taught and it was easier for the students to understand the material taught and used.
Johns Hopkins University, Baltimore, Maryland	Due to the use of Arduinos, students found that relating theory to the experiments was easier.
Columbia University	This platform, the Arduino one, is highly promising for distance learning courses and can quickly be extended to other courses.
Stanford University, Stanford	Providing a different type of platform that is more understandable to the students and by being a low-cost approach, this platform breaks economic barriers to the students that study these types of courses.
Wayne State University, Detroit	A better understanding of how Bluetooth communications work.
University of Engineering & Management, Kolkata, India	This project gives important information to the users by informing them regarding the status of their power consumption, giving the ability to the user to manage better their activities.
University of Chittagong, Bangladesh	The use of this project is directed at rural areas or countries, where real medical care is difficult to obtain and is expensive.

2.3. Conclusion

Based on all this, we can determine that the use of the Arduino and similar devices, can contribute in many ways to improve the students’ lives. The use of these types of technologies is a proven method to increase interest and awaken the curiosity within the students and by learning these types of subjects in this way and using this method, the interest among them has grown. It is a viable way of teaching and learning. Also, the combination between the cyber-physical devices, like the Arduino, and Bluetooth shields, leads to new projects and interesting ones to be created and to be developed by the students, and at the same time, since it is harder, it brings up an important challenge in the education that the students must embrace and develop a better project. Due to all of this, the conclusion that I have reached is that the use of these devices and connectivity shields and devices must be used and should be used to create a better project, an interesting one and a project where not just myself, but the students who might use it can be excited and proud to be using something to help them achieve better results and motivate them even more. Also, these conclusions are applied to students or to people who do not have a background in these subjects but have the will to learn more. This project will contribute to their learning in a more educational and fun manner.

3. Architecture

The IoT lab box is a cyber-physical system where multiple interfaces, hardware, and software, work together creating a system that can be used to develop features and give information to the people using it about the environment surrounding them. This box has the objective of demonstrating to everyone using it, how can we perceive the environment surrounding us by using sensors and cyber-physical interfaces to show what is going on and at the same time, giving them a different approach on how to work with sensors and how can we develop code and create features.

This box will present itself as a new way of gathering information and displaying it to everyone, showing different types of applications and ways of making the best of this new thing called the Internet of Things (IoT). A simple box, where multiple physical interfaces work together with the help of software, will allow this idea to be implemented. In the next section, we will explain what type of requirements are necessary to build this box.

Given the scope of the project, after developing the features and assembling all the components, tests must be run to ensure the correct behavior of the system. This system after being completed will be applied to several existent laboratory guides to demonstrate the versatility of the project and that can be applied to several situations and executed easily.

The ACIC course previously referenced, is a course where the first contact with the Arduino platform is presented to the students. This course has already laboratory guides built that do not use the system how it's built, but that limitation does not present itself since the IoT Lab box can be adapted for the purpose desired. By adapting the box, a demonstration of the system and the fact that it is versatile will be made clear.

3.1. Requirements

To develop this box and to use it, different materials are required. This box can be assembled using two main hardware devices. The first one is the Arduino Uno and the second one is the ESP32.

The Arduinos Uno are provided by the teacher. These Arduinos are present in the Arduino Starter Kit, and they are the base of the entire system. These are the brain of the box. It is necessary a computer, mainly the personal computer of the person using the system for them to be able to connect to the Arduinos present in the box and to design and interact with the features. To enable the possibility of having a wireless interaction, a Bluetooth and WIFI shield will be provided. The Bluetooth interface is provided by the teacher, and it will be a key component to develop a new type of connection, a wireless one. By having this interface, more laboratory guides and more interesting approaches can be developed giving the ability to explore the full potential of this box simply because the limitations of our system are now expanded. Although this Bluetooth shield provided by the teacher is the one present at the time, this author acquired a new Bluetooth device for reasons that will be explained later. For this type of connection to work, it is recommended and needed to use the same Bluetooth device as the author. As said, all these reasons will be later explained.

To interact with the box via Bluetooth, we will use a smartphone. This communication will have as its “middleman”, an android app, to be developed, where all the features will be present and will allow the user to interact with the box and visualize all the outputted values from the sensors.

The WIFI interface is also an interface that is needed to provide a direct connection between the Arduino and the internet allowing to obtain values and information directly from the cloud. This WIFI connection will be made using a WIFI shield provided to the students.

By saying all of this, we still need to tell the students what they must have and bring to be able to work with this box.

Since the IoT lab box is provided as a system altogether, the Arduinos are not needed by the ones using it, unless they want to connect more Arduinos to the box to try to develop more features or to explore how the different types of communication between them work. For the most basic, a computer is everything that all anyone may need, mainly their personal computer so that they can have all the software installed in their computers and save all the projects on their disk. If not, they can always use the computers present in the school and the educational goal will be the same. Taking a further step, if the students possess an android smartphone, they can install the app and work with the Arduino in that way, making this the easiest solution since it does not involve the transport of the computer, and all the computers they need, can be carried in their pocket. This android app is built using an online platform that allows to compile the project into an app and install It on the smartphone. This platform also allows the use of an emulator, where the students can install this emulator on the smartphone and load the project that they built onto it, enabling the possibility of visualizing the built app in the smartphone while at the same time if changes are made to the app via the online website, these changes are immediately updated in the smartphone. This functionality will be later explained in a further section.

Nevertheless, this project and all the software and hardware behind it must be compatible with the Arduino IDE to enable the possibility of developing more code and new features besides the ones that are already planned to be developed. As said, while developing this project, compatibility with an android app is a priority and all the future projects or features developed must always keep this in mind. All the sensors present mapping the environment must always be preserved, as this is one of the main features in this box.

As said, the ESP32 is also a new approach to use this box. This device will operate in the same way as the Arduino UNO, and the knowledge necessary to code it and to assemble sensors and actuators on it is the same as to assemble and code using the Arduino. Although something may change, all these differences will be pointed out later, making very clear to the students and everyone using this box and wanting to develop even more features what is necessary. If the ESP32 approach is the one chosen, the WIFI shield and Bluetooth shield will no longer be necessary, since this device already has these two pieces of hardware incorporated.

To develop concrete projects using all this material, the Arduino Starter Kit has a book with several projects highly recommended as a starting point, especially for the ones whose background is null. The materials necessary for the assembling and development of those projects will already be available within the IoT Lab box and upon the completion of those projects, new ways to communicate are present once more within the box. All the extra features developed will be documented to make easier the understanding and the assembling process of the IoT Lab box.

3.2. Functional Architecture

This section is an overview of our system regarding the implementation details of the IoT Lab Box using the materials described in the previous topic. To understand why we use the hardware devices that we used, we must have an overview of the Functional Architecture of our system. This overview is the fundamental plan to move on to the implementation of the IoT Lab Box (Figures 1 and 2). The assembling procedure of these devices and what are the advantages and disadvantages of these when compared to similar devices that we could have used and how do they work and perform will be explained in the following Implementation section.

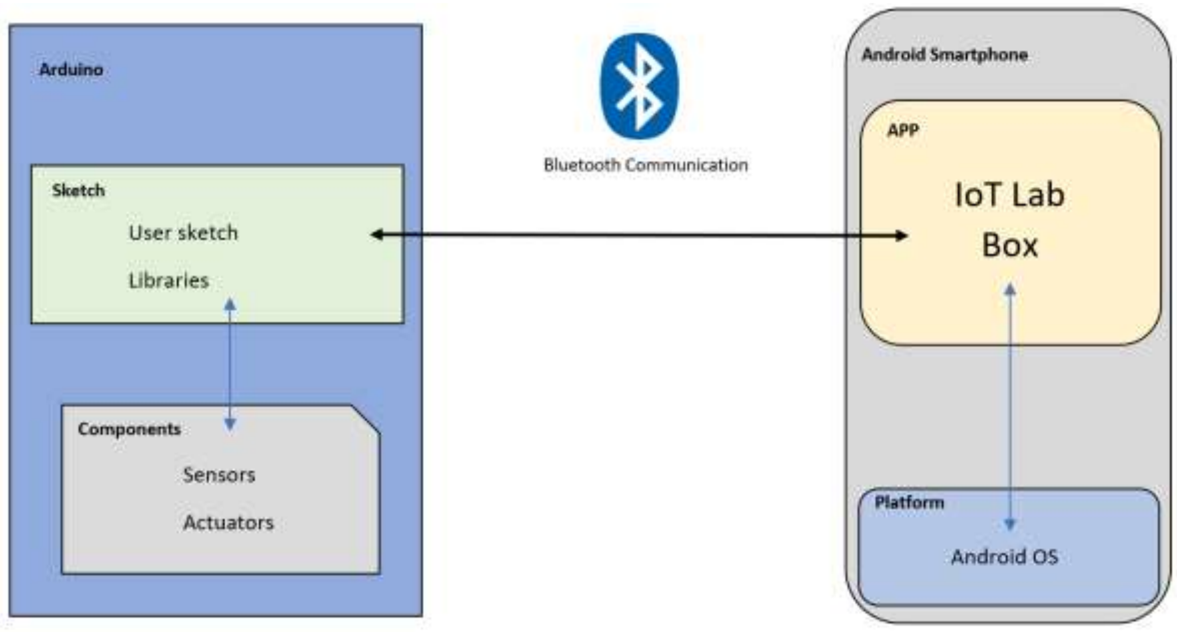


Figure 1 - Functional Architecture Bluetooth

Looking at Figure 1, we have an overview of the system regarding Bluetooth communications. To implement this communication, we need an Arduino and a Bluetooth module. As we can see in Figure 1, the Arduino has two main parts, the **Sketch,** and the **components**. The **Components** part of our system includes the sensors and actuators needed for the project. The sensors map the environment where the system is placed and send that information to Sketch. The Sketch is composed of two main features: the User sketch and the libraries. The User sketch is what we code. The user creates a program to be able to read the information sent by the sensors and can send information back to the actuators, that as the name suggest, **act.** When creating a sketch, several libraries can be used, depending on what we need for our system to work properly. For example, to enable the communication between the Arduino and the Bluetooth, a specific library, responsible for this communication must be added to our user sketch so that the system performs as expected.

Having created the code needed for our program and imported and used the libraries to enable the Bluetooth communication, we send the information from the Arduino to the Android Smartphone. The smartphone, running the Android OS platform has the App for our project, the IoT Lab Box App. The values sent by the Arduino are received in this App and displayed there. As seen in Figure 1, the arrow responsible for demonstrating this communication is bidirectional. This means that the communication between the Arduino and the smartphone can be made in both ways. Both can send and receive information. When sending information from the App in the smartphone to the Arduino, the Sketch part of the system, is the one responsible for filtering the data, showing the user the correct information.

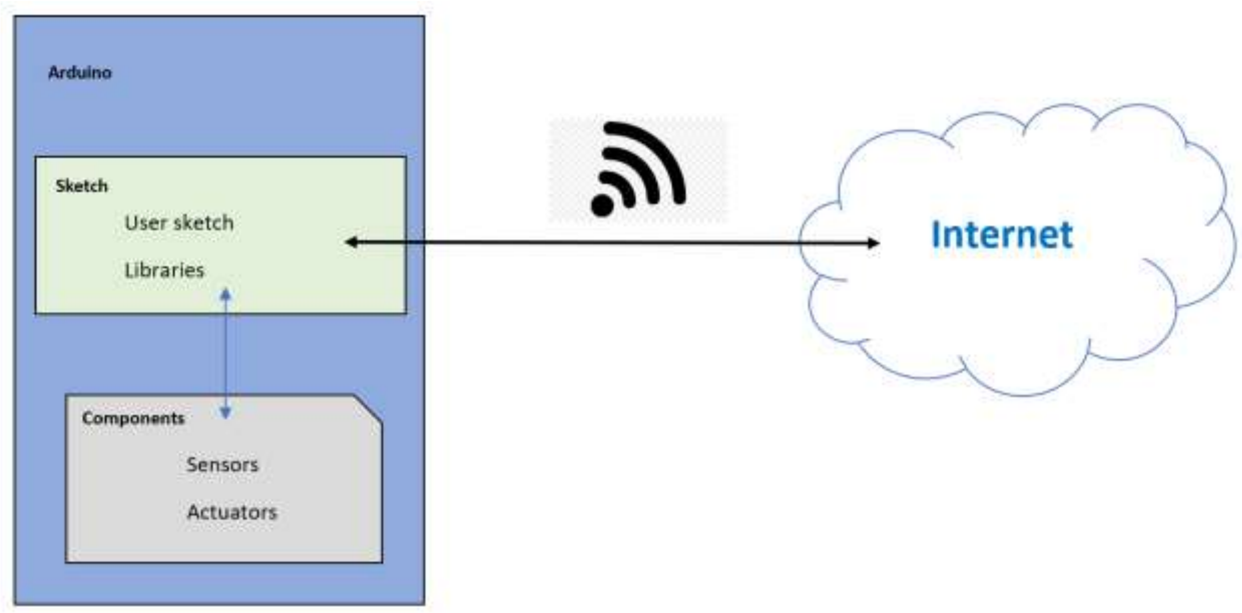


Figure 2 - Functional Architecture WIFI

Looking at Figure 2, the logic behind it is the same as in Figure 1. Our Arduino part of the system is divided into two main parts, the **Sketch,** and the **Components.** The way these communication works is the same as it was described in the previous paragraph. Here, the only changes that we encounter are the libraries used, since now we are using the WIFI as the communication method and not the Bluetooth. Nevertheless, the overview of the system whether we are using Bluetooth or WIFI is the same. This WIFI communication is also a bidirectional one since data can be sent and received from both devices.

As said in the previous section, the implementation of this project has two main approaches. Firstly, the author approached the project using the Arduino platform, using the Arduino Uno board. In the initial stage of the project, only one Arduino Uno is necessary but later, one more will be implemented to develop new features in our box.

We can define a structure for this project in a simple way to understand better everything used.

In the first stage, the author assembled one/two Arduino UNOs boards to develop the base for the project. On a second stage, the Bluetooth module was assembled, and the android app was developed to work with the previous stage of the project, exploring this wireless form of connectivity. The third stage was the assembling of the WIFI shield and the development of this communication using a WIFI network and the MQTT broker to enable communication between the two Arduino boards.

In the final stage of the project, the IoT Lab box implementation was changed to be developed with ESP32 devices and the re-assembling of all the sensors and actuators was done, now using these new devices. While using the ESP32, the need to assemble a Bluetooth module and a Wi-Fi module was no longer necessary, since the ESP32 as already incorporated these two modules. There was however the need to adapt the android app previously developed to work with the ESP32. The overview of the architecture of our system, now using the ESP32, is the same as using the Arduino Uno board.

By using these two similar but at the same time different devices from the Arduino platform, the differences between the two were explored and allowed for a bigger development and an educational enrichment since now two devices are used. This allows for a better learning opportunity and experience and the users of this box have now two devices to choose from. These differences and the conclusion taken from the use of these two devices will be discussed in a further section.

4. Implementation

Based on the previous section, an overview regarding the Functional Architecture of this project was exposed. All the details regarding the assembling, the coding, and the implementation must be explained and a guide must be provided so that anyone who wishes to use this box knows how it works. The next section includes an explanation about the different materials used and why they are important to develop our work. After this explanation, we will enter the development of the IoT Lab Box project. All the code produced for this project will be available on the André Santos' GitHub[5].

4.1. Background

Considering the hardware requirements necessary for our project, this section gives us a more elaborate explanation of the materials necessary for the development of the IoT Lab Box project. Here, we cover all the materials necessary for all communications in our project and the importance of these in the process of developing the box. In Chapter 2, the IST Context section, the use of the Arduino in the IST courses was approached, and the selection of the materials was made keeping in mind the opportunity that this box gives to those same courses. The selection process of the materials took in also in consideration the key components necessary for an enriched learning process for everyone, providing a versatile project to those who use it.

4.1.1. Arduino Uno

Let's start with the Arduino UNO (Figure 3). As previously described, the Arduino Uno is the basis for our project. It is the core of our IoT Lab Box. This hardware device is the start point of our project mainly because it is the device available at the IST and it was the device made available for the author.

This board has a lot of computational power and a very accessible price. The box, containing all the materials necessary for the project can be placed in a fixed location, to be determined. But we can theoretically say where it can be placed. For example, if the box is placed in a laboratory room when a student comes to the laboratory and because this box is mobile, the student can put the box next to his or her computer and connect their PC to it. The students can also place the computer next to it and work from there. Another possible location for the box is a study room. Here is where we can find students from multiple courses, courses that may include students who do not have a background using these systems but have the will to learn. For that reason, the location of the box must provide an equal chance to be used by everyone.

By achieving wireless communication with this box and by installing the android app on a phone, the box can be placed in a fixed location in the room and the people can have an interaction with it anywhere they seat while being in the range of the device.

[5] https://github.com/santosandre123/IoTLabBox (Accessed 10/2021)

To develop the code and to upload it to the Arduino, the Arduino IDE software was used. This software was also used to verify all the values and to perform debug when necessary, using the Serial communication feature present in it. After verifying that everything was performing correctly, only then we can move on to the next stage. Here is where the new types of communication are introduced, the Bluetooth communication and the WIFI one.

Figure 3 - Arduino Uno board

4.1.2. PC

To set up the box and all the features, we must first set up the Arduinos and program them through an appropriate IDE. The IDE chosen for this was the Arduino IDE. It is already the most popular IDE, and it is the one developed by the Arduino platform and the recommended one to work with these devices. To set up all the software needed, we will connect the Arduino through the USB interface present in the Arduino UNO and present in the computer. Without this interface, the setup will not be possible because we cannot make a connection to the computer. Through this connection, we can develop code using the IDE, including features related to the physical sensors. The development of all laboratory activities also starts here since, using this IDE we can adapt the code needed to develop the laboratory guides and develop new code to act as a library so that the students can have a starting point for their activities. The communication between the Arduino and the computer is made effortlessly since we only need to connect the two devices and after that simply press a button present in the Arduino IDE and upload the code to the Arduino. This communication between them is called the Serial Communication since the Arduino UNO has a Serial Port dedicated to this. This communication is what allows the visualization of the data outputted by the Arduino through a Serial Monitor present in the Arduino IDE. It is also, through this Serial Monitor, that we can perform the debugging process and more and understand better and understand if what we code is working or not.

The computer offers us the possibility to unlock all the potential present in this box and create more, using the libraries already present and at the same time, challenge us to create new features and to adapt the existing ones to improve or even to redesign the ones that already exist to build something better.

4.1.3. I2C

As opposed to a wireless experience, we cannot neglect one of the most important types of communication, I2C. This wired type of connection provides us with a different approach to how everything works. By connecting Arduinos directly, we obtain a more reliable form of communication and a very fast one. Laboratory guides that use this type of connection already exist, a many more can be created. By having more than one type of connection, the development of exercises will be even more necessary, and we can, of course, modify existent by adding more features

This traditional type of communication is not our goal here. The I2C communication is already used in the laboratory guides, and it is only used to make sure that the assembling of the laboratory was made in the right way and that all the sensors and actuators are doing what was intended. Going in-depth on this is not something that we are going to do or need to do, since it is not "something new".

4.1.4. BLE Shield

Let's focus on one of the main new features present in the IoT Lab Box. As previously said, the wireless form of connectivity is something that going to be explored in this project. It is something that requires a more depth understanding of how things work and provides a better understanding of the IoT world to those who explore this. As described in the State-of-the-Art section, this form of communication is already used by other universities, and it has a lot of applications for a variety of projects and more. Because of this, it is without a doubt something to explore and to develop.

The Arduino UNO itself has no Bluetooth communication capability and to have this type of communication, it is necessary to add a piece of hardware to provide Bluetooth capabilities. Without the addition of this hardware device, it's not going to be possible to achieve our goal.

A Bluetooth shield was provided to the author to connect it to the Arduino (Figure 4 and 5). This specific shield, 1Sheeld +, was developed by the 1Sheeld Company[6] and it was the hardware available at the and the hardware received for this part of the IoT Lab Box project.

This shield uses a standard HM-10 BLE module (Bluetooth Low Energy). It has a range of up to 30 feet and communicates with the Arduino board using UART (Serial communication).

One of the main features of this shield, it's the fact that uses BLE and not standard Bluetooth. Although they work in a very similar way, the BLE is most used for applications that don't need to exchange much

[6] https://1sheeld.com Accessed (05/2021)

data and use less energy than the standard one, being more economical. The power requirements are also lower.

This shield has already pre-uploaded firmware onto its module and on its website, it is possible to find projects and tutorials to develop projects. This company also provides a smartphone app to be installed in a smartphone allowing the user to control the shield via the app.

The assembling of this shield is extremely easy if nothing is connected to the Arduino. This shield is connected on top of the Arduino UNO board and has the same interface's inputs as the Arduino, meaning that everything that we want to connect to Arduino, we still can but now we connect it to the shield.

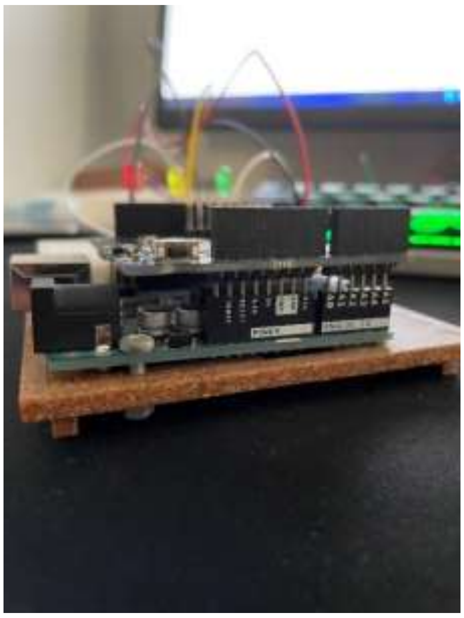


Figure 4 - BLE shield

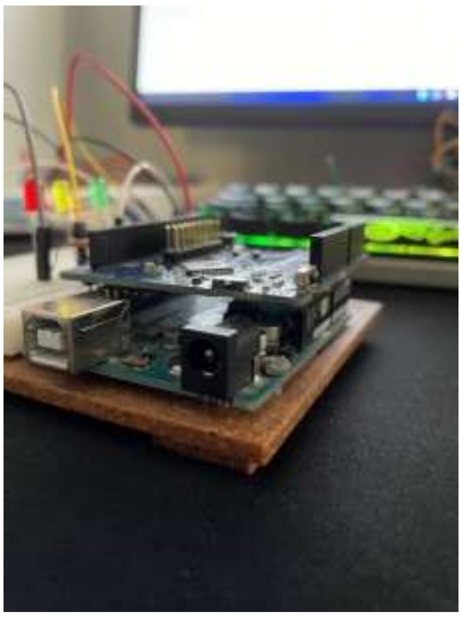


Figure 5 - BLE shield 2

4.1.5. HC-06 module

The author also acquired a Bluetooth module, called HC-06 (Figures 6 and 7). This is a very cheap device that when connected to the Arduino board provides Bluetooth connectivity to it. The HC-06 Bluetooth module is a slave Bluetooth module designed for wireless serial communication. It is a slave module, meaning that it can receive serial data when serial data is sent out from a master Bluetooth device (the device can send serial data through the air).

This device is a default Bluetooth device and not a BLE – Bluetooth Low Energy one. At first, this device was acquired to explore the differences between the classic Bluetooth and the BLE. It is simpler to work with and cheaper than a BLE one but most. The implementation of both these devices was always the goal but as the implementation began, limitations and problems while implementing the BLE were discovered and the use of the BLE was put to the side because of them. These problems and limitations will be explained in the following section, the Implementation. With the purchase of the ESP32 that already has a BLE incorporated the goal was achieved and the work was also done using BLE. By having these two devices, the author was able to develop and demonstrate what is like to work with the two devices and can show the differences between them.

By choosing this use this HC-module, the author did not have an app to work with and so the development of one was a requirement.

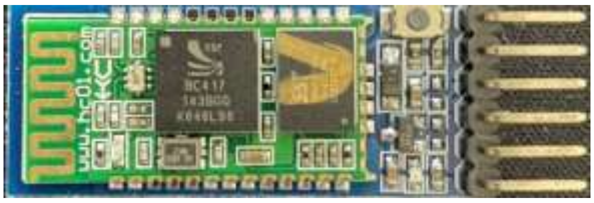


Figure 6 - HC-06 module (front)

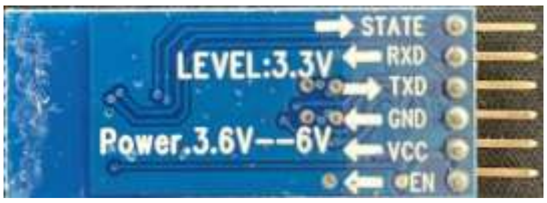


Figure 7 - HC-06 module (back)

4.1.6. MIT App Inventor

As previously said, to use the HC-06 module with the smartphone and with the Arduino board, an app is necessary. The operating system chosen was the android one, based on its simplicity and because it is open source.

There are several platforms and software available to create an android app, more sophisticated or simpler and after searching and reading about all the different platforms, the MIT App Inventor[7] was the chosen one (Figure 8).

As described on their website, "The MIT App Inventor is an intuitive, visual programming environment that allows everyone – even children – to build fully functional apps for Android and iOS smartphones and tablets". This online platform allows the creation in a simple way of Android apps and has a very close connection with the IoT world, including, of course, the Arduino platform.

It's a block-based coding platform, making it easier to create and modify the code and see in real-time what we are developing. One of the great features of this platform is the fact that it has an emulator that allows experimenting and debugging in real-time with our smartphones.

It is a very interactive platform. A lot of information about the use of this platform with Arduino boards is present on the internet. Since it is a block-based coding platform, the user does not need to know how to code an android app specifically. The knowledge behind specific programming languages, is no longer a limitation. Although the coding procedure for this android app using this platform is something relatively easy, it's something that requires a lot of thought and like everything has a lot of trial and error. Because it's an online platform, it provides the ability to work anywhere desired, having the advantage of having all the projects saved on the cloud.

One of the main features of this platform is the fact that it has incorporated an emulator, allowing the user to experiment and debug on the spot. By simply downloading the emulator app to the smartphone device, we can change and experiment with the code developed on the spot while connected with the

[7] https://appinventor.mit.edu/ (Accessed 05/2021)

Arduino board, making the interaction more interesting because we can see in real-time what we are making. This platform allows creating code via blocks (back-end) and at the same time change and creating new front-end solutions to serve our purpose in the best way possible.

The goal of keeping everything open to everyone that wants to explore this theme and the IoT world, using this project as its base it's achieved by doing the project in this way. All the core code needed for the app will be provided and can be modified without much trouble.

4.1.7. WIFI

Having a WIFI interface will provide the Arduino access to the internet and unlock the possibility to connect the Arduino to a Wi-Fi network allowing the user to obtain information relevant from it to the project.

To connect the Arduino UNO board to the internet, we don't necessarily need specific WIFI hardware, like a shield, but having this device will help, and it will make our work simpler. For example, using specific software, like the Ardulink software is possible to build a bridge between the Arduino and the internet using our computer as the "middleman", but the process to configure it, is rather complicated and frustrating. If one chooses this approach, he/she will soon realize that information regarding its use, like the commands necessary by the user to make this work is almost null. Although the information regarding these commands is little, this limitation could be overcome by trying to find the documentation about this software and its configuration. Here, another limitation presents itself since the documentation needed is almost null. Taking into consideration that the goal and focus of the project is the educational one by getting the students and everyone who wishes to use this project interested in a different approach to cyber-physical interfaces, this approach will not comply with our goal. Several big limitations can withdraw the interest from the people that want to use this project and presenting them with limitations as big as this one, is definitively not ideal.

By having explained why trying to connect the Arduino Uno board to the internet without a shield, it's not a good idea, the decision to use a shield was made, and one was provided to the author to enable this new type of communication. This shield was the one available at the time and for that reason, it was the one to be used.

The WIFI shield to use in this project is the ESP8266 ESP-12E UART Wireless WIFI Shield TTL Converter, and it will enable us the cloud connection to our project (Figures 9 and 10).

It is called "ESP8266 Wifi shield Version 1.0 by Wang Tongze". This shield when connected to the Arduino board, gives the possibility of connecting the Arduino to a WIFI network. This network can be a network in our school, or simply the network present in our homes. There isn't a limitation. This shield was developed to provide WIFI capabilities to the Arduino board when connected to it. It uses the ESP8266 module, a very well-known module, with powerful computational capabilities, more than the

Arduino has and can be used as simply has a device that adds WIFI to the Arduino board or it can be used has an Arduino itself.

This shield can be mounted on top of the Arduino board. There is no longer the need to mount a circuit with several components and wires to interconnect an ESP8266 to the Arduino board. We can simply attach the board and the shield, position the DIP switch path according to the shield operating mode as we desire, and code it so that the Arduino can connect to WiFi networks. This shield can still be mounted in a different way, where we interconnect the Arduino board to it. It can be coded as easily and using the same programming language as the Arduino. This means that, if the user no longer wishes to use the Arduino board, the user can program the module and work exclusively on it.

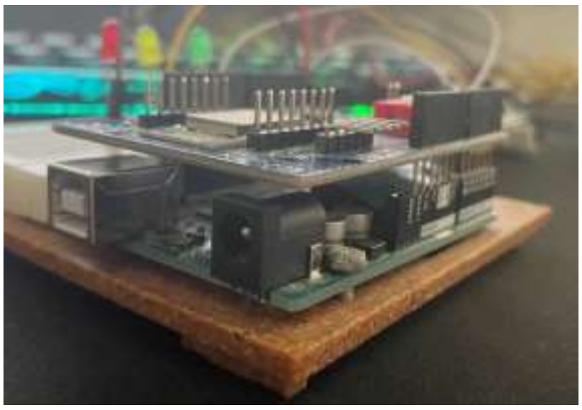


Figure 8 - WIFI shield

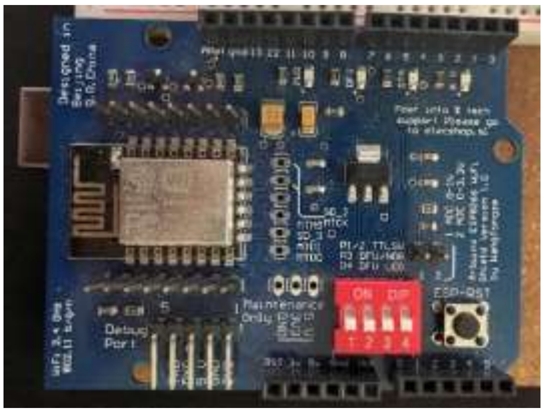


Figure 9 - WIFI shield

4.1.8. ESP32

The final stage of this project is done using the ESP32 device (Figure 10). When the IoT Lab Box was purposed, the ESP32 was not part of the plan. As the development of the project keeps on advancing, new solutions and new ways to do this project keep on appearing.

The ESP32 is a very well-known device. This device is seen as an upgrade to the ESP83266 module, present in the WIFI shield used for this project. Being a more advanced module, with better specs and more capabilities, it was for sure, a device to be considered. The author began the search to find answers that could help him decide on using this module or not. The search led to the conclusion that the use of the ESP32 module has something to do.

The ESP32 module is a module with a lot of computational power. It is more powerful than an Arduino Uno board or the ESP8266 module. But the deciding factor for this purchase was not this. The ESP32 device acquired by the author had already WIFI and Bluetooth features incorporated. This means that all the additional WIFI and Bluetooth shields needed for the Arduino Uno are no longer necessary and all the work can be done with only one device. Furthermore, this device has a reduced dimension, it is easily transported and stored. Additionally, the Bluetooth module present in this device is the BLE (Bluetooth Low Energy) one. As explained previously, the BLE module was always the one to use, due to its advantages. By having the ESP32 module, it was not necessary to acquire a new Bluetooth module, since the ESP32 already had the one desired from the start. This allowed comparing the two modules. The coding process was relatively the same as the Arduino Uno, so the need to learn completely new libraries was not necessary. The new libraries needed to work with this module were the same learned by the author when using the ESP8266 module as an Arduino itself. The assembling of the sensors and actuators was the same, although the pins used for these connections are not the same as in the Arduino, the logic behind it it's the same and the existing documentation can be consulted for a deeper understanding.

This new approach allowed also new features to be developed. For example, it was possible to set up a local webserver running on the ESP32 and display all the data outputted by the sensors on a local website. This was not yet done and enables new possibilities for the future. The author did not find limitations to the use of the ESP32, or at least significant limitations that would compromise the goal of the project, and of course, used this module to perform the project. This provides a new platform to be compared with the Arduino.

The details of the assembling process and the implementation of the project using this device will be explained in-depth in a further section, alongside will all the conclusions reach by the author.

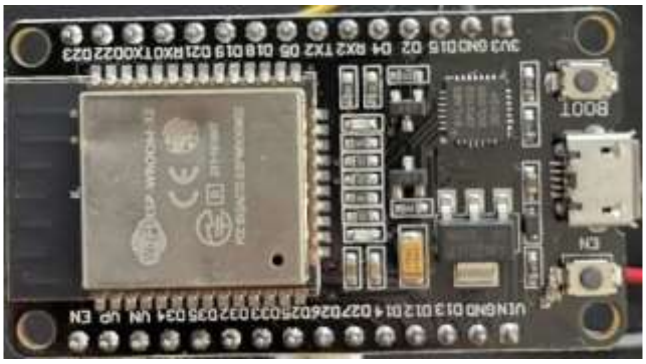


Figure 10 - ESP32 module

4.2. Base Development

The assembling of our IoT Lab box began by choosing sensors and actuators to work with. Having chosen the materials, the assembling process started.

Initially, the author began by assembling the TMP36 sensor (analogic temperature sensor). Having assembled this sensor, the author also chose to assemble a LED to light up when the temperature reached a determined threshold. Having both pieces of hardware assembled, the coding procedure took place. The Arduino IDE program was the platform chosen to perform the coding procedure and to verify if the values outputted by the sensor, the Serial Monitor present in the Arduino IDE was used. Having the temperature sensor coding part completed, the LED was next. The behaviour expected was achieved.

4.2.1. BLE Shield

The first step in using the BLE shield is to assemble it on top of the Arduino UNO, as described in Figures 4 and 5. Having the shield assembled, the next step was to assemble the temperature sensor previously used by connecting them to the BLE shield. As described, when the BLE shield is assembled on top of the Arduino Uno board, all the sensors and actuators that we now want to connect to the Arduino cannot be connected directly to it but instead, we must connect all these devices to the BLE shield.

With the sensor connected, the author then tried to see, using the Serial Communication, if any values outputted by the sensor were being shown. This was not true anymore. All the communication previously established was not there anymore. Why?

The answer to this was very important since, without this communication, it would not be possible to use Bluetooth communication and any other shields. Since the shield was mounted on top of the Arduino board with all the pins connected, two very important pins were now obsolete, the D1 and D0 pins.

These two pins are responsible for the Serial communication between the Arduino and the computer. Since the Arduino is powered via USB and it constantly communicates via serial with the computer for debugging purposes and more, when connecting a shield to the board that uses these two pins, this communication is no longer possible. For example, when uploading a sketch to the Arduino UNO, the communication between the computer and the board is made through USB and these pins. This means that now, the communication between the Arduino and the computer was lost.

To fix this, there are two possible ways. The first being, every time we want to upload a sketch to the Arduino, we must disconnect the shield from the Arduino board. This is not viable since the shield is mounted on top of the board and by disassembling and assembling the shield every time, we need to upload a sketch is not practical and we can damage the pins doing this. The second way is bending the two pins in the shield, corresponding to the D1 and D0 connection, and using an auxiliary library called “SoftwareSerial”.

a) SoftwareSerial Library[8]

As described in the previous topic, the pins D0 and D1 from the Arduino UNO board are responsible for the Serial Communication between the board and the computer. Although there is no actual problem when connecting something to these pins, if there is a need to communicate with the computer, these pins must remain clear. When a shield is assembled on top of the Arduino board, occupying these two pins, this communication is no longer possible. An alternative must be found.

To communicate between the Arduino board and the Bluetooth shield, a library called SoftwareSerial was used. The SoftwareSerial library has been developed to allow serial communication on other digital pins of the Arduino, using software to replicate the functionality (hence the name "SoftwareSerial"). This means that by using this library we can define other

[8] https://www.arduino.cc/en/Reference/softwareSerial (Accessed 05/2021)

digital pins to replicate what the pins D0 and D1 do. By defining two new digital pins, for example, D4 and D5, these pins are now responsible for the Serial communication between the Arduino board and Bluetooth shield. This means that the hardware limitation regarding Serial Communication is now overcome.

Having implemented the SoftwareSerial library, our problem has now been overcome. The communication between the temperature sensor and the Arduino Uno was once more established and we can move on to the implementation of the BLE communication.

The company that made the Bluetooth shield, the 1Sheeld Company, has already developed an app, facilitating communication with the shield. This app was open-source and because of this, it was possible to modify the code to serve the purpose needed. This was a huge feature since the development of an app from scratch was not necessary anymore. The code was obtained in GitHub and the author began to experiment and began adapting the code so that it would work with this project.

After experimenting, problems keep on appearing. As previously explained, this Bluetooth shield had previously installed firmware to only work with this app and this app and the back end of this app was very complicated to modify and adapt to the project. This complication refers to the firmware installed, and the libraries that needed to be learned. Furthermore, the shield did not work always as intended. The Bluetooth communication was ON and OFF, and the sketch uploaded to the Arduino, did not allow every time to do what we needed. Not using the libraries and using the specific coding procedures needed for this shield to work, caused limitations where we didn't want them.

This was not the goal. The one thing to always keep in mind is that this will be used by other students and to be used by people with no background in these kinds of subjects and the simplicity and the ability to change and modify is very important. By using this shield, this was not going to be achieved.

A new alternative was meant to be found. The author decided to put aside the BLE shield and use the classic Bluetooth module, the HC-06, to develop this communication.

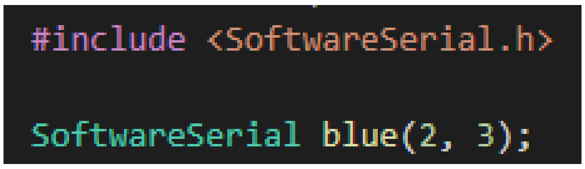
This is not the only problem found. Keeping in mind the goal desired for this IoT Lab box, being the simplicity and providing the opportunity for anyone with or without a background on embedded or IoT systems, this shield was not fulfilling these requirements. As mentioned, the shield already has pre-installed firmware on it. To be able to work with it, it is necessary to learn all the libraries previously built and how they operate. This is a very time-consuming task and may not be easy for everyone.

The author began by doing this but realized that all the libraries built were not easy to understand. Although on their website, there are previously built projects using this shield, these projects are very specific and only work with these shields and not others. This is not ideal. The HC-06 module was the path to take.

4.2.2. HC-06

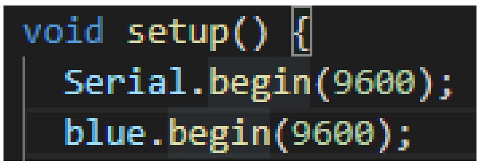
Putting aside the BLE shield, Bluetooth communication is now being developed with the HC-06 module.

The assembling of this device is done in a very simple way. The module has 6 pins (Figure 6) but to connect it to the Arduino board, but only 4 of them are used (Figure 7). The pins used are the RX, TX, GND, and VCC pins. The VCC pin is connected to the Arduino 5V one since it is powered by the board. The same happens with the GND (ground) one. The two other pins, TX, and RX must be connected to the TX and RX pins of the Arduino board, D1, and D0 pins. The TX and RX pins in both devices are responsible for the serial communication. As explained before, we cannot connect anything to these pins. The solution is to use the SoftwareSerial library and define new virtual TX and RX pins (Figure 11) in the Arduino board and connect the two pins from to module to these newly defined pins. When all the pins are connected, we must test them.



```
#include <SoftwareSerial.h>

SoftwareSerial blue(2, 3);
```

Figure 11 – Define Digital Pins



```
void setup() {
  Serial.begin(9600);
  blue.begin(9600);
```

Figure 12 - Initialize SoftwareSerial communication

Initially, we begin by testing if the values of the temperature sensor are being shown in the Serial Monitor. If so, the communication is established correctly. Secondly, we must see if it is possible to connect our smartphone to the HC-06 module.

To test the module, we need a smartphone, an android one, since it is the platform for our App. We begin by going to the Bluetooth menu present in our smartphone and begin the scanning process for Bluetooth devices. The HC-06 module has a rapidly flashing red led, informing us that it is ON, but nothing is connected to it. Once our smartphone detects the module, we connect to it and if the red led remains still, we know that the connection has been established.

This module has a very simple configuration and assembling process. After the completion of all these steps, no problems have emerged. The module was fully prepared for the user to proceed with the development of the code needed for the communication.

By using the SoftwareSerial library, the Bluetooth module communicates with the Arduino through Serial communication and the initialization of this communication must be done (Figure 12). Having all these steps completed, how can we see visualize now the communication between the Arduino Uno and the Bluetooth module? To see this communication, an app must be built.

In the previous section, the platform chosen to develop the app was described. The MIT App Inventor is a code-based platform, making the development of apps easier since it does not require an understanding of coding languages and mobile app development. This was the deciding factor in choosing the platform to use to develop our app. Since this project aims at students and people who may not have a background in developing IoT solutions and app development, by using this platform no one is left out and everyone can develop the app they want.

By having a Bluetooth module and a temperature sensor connected to our Arduino, the first objective for our app was to be able to see the values outputted by the sensor. We are building a mobile Serial Monitor. Although there are already Serial Monitor apps, the decision to make this app made sense since it allows for a better understanding of how we can develop an app and allows for the possibility of developing the features needed for our project. The platform has already pre-built functions to allow sending/receiving messages to the HC-06 module using also Serial communication. It is only necessary to create a function to act as a listener for communication received via serial communication and store the messages received the way desired.

To interact with the app, we can use two different methods. The first one is the one where we compile the app and generate a **.apk** file and install it on our android smartphone. The second one is to download an emulator app present in the PlayStore specific for this platform. This is the recommended method and the chosen by the author to work with the app. Using these methods gave us several advantages. Using this method saves us time because we do not have to compile the app and copy it to our smartphone to work with. If there is an error with our app and we need to address it, it implies that after fixing the bug, we must then recompile the app and repeat the same process repeatedly. By using the emulator, the app is virtually compiled and all we can interact with it the same way as if the app was installed in our smartphone and at the same time allows the user to modify the app, whether it is the design or the code behind it on the spot and visualize these changes in real-time. For example, if we detect a bug in our app, we can simply fix it while the app is running in the emulator and test it right away without the need for the installation of a new version. Since this app works using an online platform, all our projects are saved in our account, and we can access them anywhere and anytime we want to make all the changes necessary. The app with a Serial Monitor included was built (Figures 13 and 14).

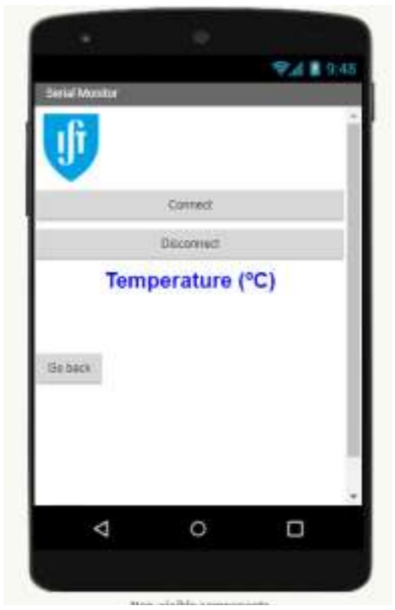
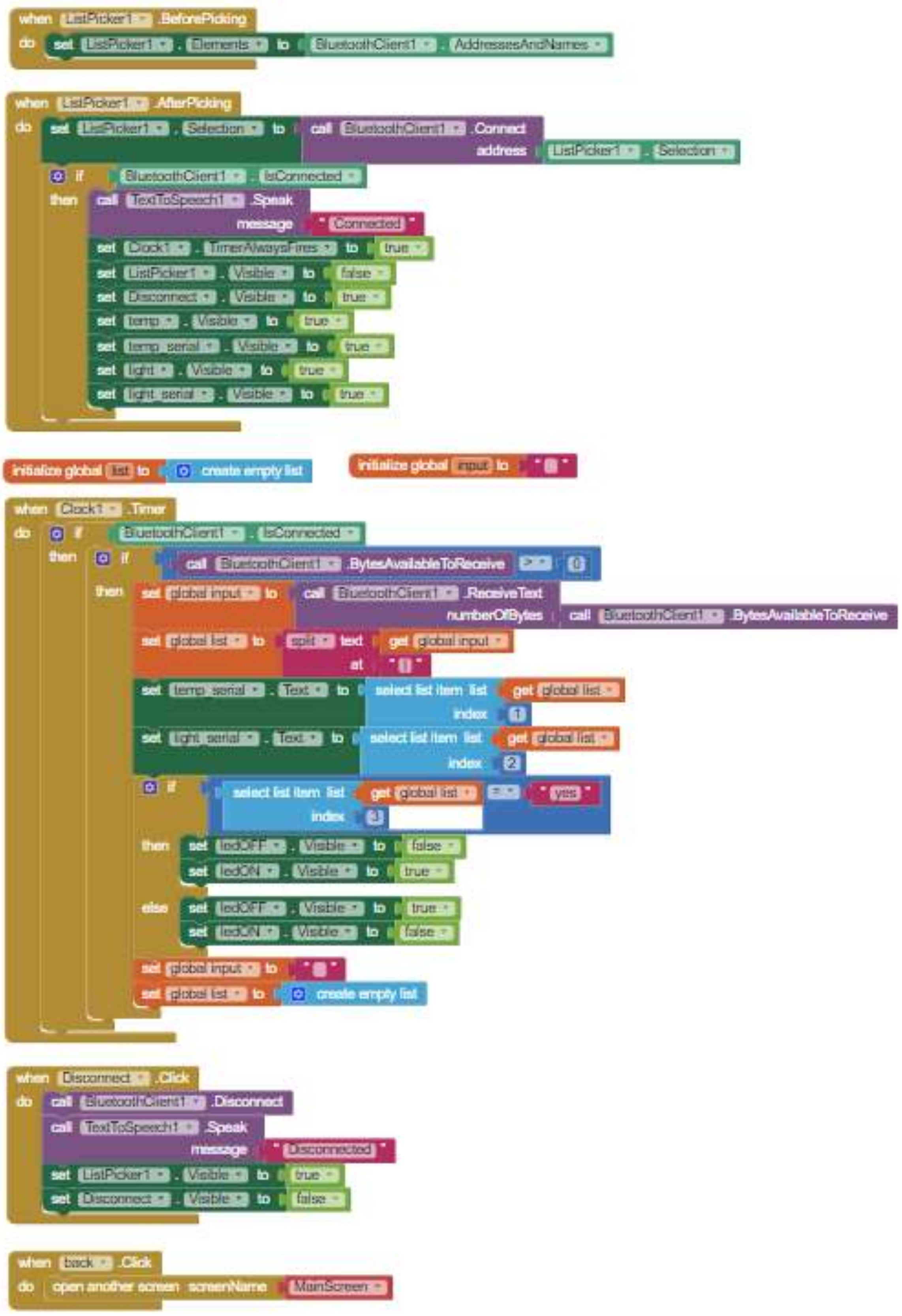


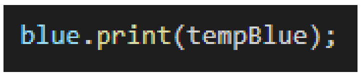
Figure 13 – Mobile Serial Monitor



```
when ListPicker1 .BeforePicking
do  set ListPicker1 . Elements to  BluetoothClient1 . AddressesAndNames

when ListPicker1 .AfterPicking
do  set ListPicker1 . Selection to  call BluetoothClient1 .Connect
                                                  address  ListPicker1 . Selection
    if    BluetoothClient1 . IsConnected
    then  call TextToSpeech1 .Speak
                      message  " Connected "
          set Clock1 . TimerAlwaysFires to  true
          set ListPicker1 . Visible to  false
          set Disconnect . Visible to  true
          set temp . Visible to  true
          set temp_serial . Visible to  true
          set light . Visible to  true
          set light_serial . Visible to  true

initialize global list to  create empty list        initialize global input to  " "

when Clock1 .Timer
do  if    BluetoothClient1 . IsConnected
    then  if    call BluetoothClient1 .BytesAvailableToReceive  >  0
          then  set global input to  call BluetoothClient1 .ReceiveText
                                         numberOfBytes  call BluetoothClient1 .BytesAvailableToReceive
                set global list to  split  text  get global input
                                           at    " | "
                set temp_serial . Text to  select list item  list  get global list
                                                              index  1
                set light_serial . Text to  select list item  list  get global list
                                                               index  2
                if    select list item  list  get global list   =   " yes "
                                        index  3
                then  set ledOFF . Visible to  false
                      set ledON . Visible to  true
                else  set ledOFF . Visible to  true
                      set ledON . Visible to  false
                set global input to  " "
                set global list to  create empty list

when Disconnect .Click
do  call BluetoothClient1 .Disconnect
    call TextToSpeech1 .Speak
                message  " Disconnected "
    set ListPicker1 . Visible to  true
    set Disconnect . Visible to  false

when back .Click
do  open another screen  screenName  MainScreen
```

Figure 14 – Block-based coding

Having built an app with an incorporated Serial Monitor it was time to test it. To see if all the components of the system are working properly, we need to visualize the temperature values in the smartphone. To double-check the results, the results are printed in the Serial Monitor present in the Arduino IDE and sent via Bluetooth to our app. The HC-06 module communicates with the Arduino board using Serial communication. While using the Arduino Uno board, to communicate with the Serial Monitor the command "Serial.print()" is used. Now with a new digital Serial communication with the name **blue** (Figure 12) the communication is made using the command "**blue.print()**" (Figure 15).



```
blue.print(tempBlue);
```

Figure 15 - Command to send values through the Bluetooth

If both values correspond every time they are sent, the system is working properly. This was exactly what the project did.

Having the Bluetooth communication base complete, it was time to move on to the WIFI communication. We start by only working with one sensor to establish a core base for our project.

4.2.3. WIFI Shield

The shield is powered using Arduino voltage (5V) and has a voltage regulator reducing the power supplied by the Arduino to 3.3V. It has a built-in logic level converter, so the Arduino TTL level (5V) does not damage the ESP8266 that operates with TTL 3.3V level. This means that the Serial communication between the board and the shield is regulated. But what the author verified is that this assumption is not true.

To use this shield together with the Arduino board there are a few steps that need to complete. First, we need to make sure that the shield and the board can communicate with each other. The ESP8266 module has previously installed firmware that may be outdated or may not be compatible with our Arduino board and we need to make sure that they are compatible. To do this, we must connect the D1 and D0 pins of the Arduino board to the shield. After connecting them, we use several commands called "AT commands" that allow us to change and verify the firmware. Using these commands, we can send messages between the board and the shield. We start by sending basic commands from the Arduino to the module to verify if we obtain a response. After obtaining this response, we began by testing if the module can connect to Wi-Fi. We choose a network, insert the password of this network, and verify if the connection has been established. If all this works, there is only one thing left to do. Change the baud rate. The ESP8266 has a baud rate of 115200 and while the Arduino serial communication can operate at this speed, it is not recommended. We can simply change the speed to 9600 and the process of verifying the shield is completed.

After this, we began by assembling the temperature sensor in the Arduino board and connecting the shield to the Arduino board. The communication between the board and the shield is made by using the SoftwareSerial library.

This shield has the same problem as the Bluetooth one. When mounting the shield on top of the Arduino board, the Arduino's D0 (RX) and D1 (TX) pins that correspond to native serial/USB communication did not work. These pins stay busy whenever we send code to the board or use the serial monitor. Meaning that the serial communication began to fail, and we no longer could send code to the board. The solution found to fix this was bending the pins out of the way and leaving the D0 and D1 pins without anything connected to them.

As previously said, the Wi-Fi shield is powered by the Arduino voltage (5V), but it operates only on 3.3V. The same applies to Serial communication. This means that if the shield does not have a voltage regulator, the shield will receive more than can handle and will malfunction or die. As previously said, there is a problem. The voltage regulator in the Wi-Fi shield, responsible for regulating the voltage regarding the Serial Communication, is the wrong one and although there is a voltage regulator, this regulator is not doing what is intended. This is a problem that comes from the factory and must be corrected if we want the correct function of the shield and the communication. There are two ways of fixing this problem. The first one is to remove the voltage regulator present in the shield and replace it with the correct one. This involves unsoldering and soldering a tiny piece of hardware and not everyone can do it. The second alternative is to add a logical converter and put it between the pins that are used to allow communication between these two devices. This is what the author did. These logic converters are very cheap and can be found easily. Only after adding this piece of hardware, we can find stable communication and a safe one too (Figures 16 and 17).

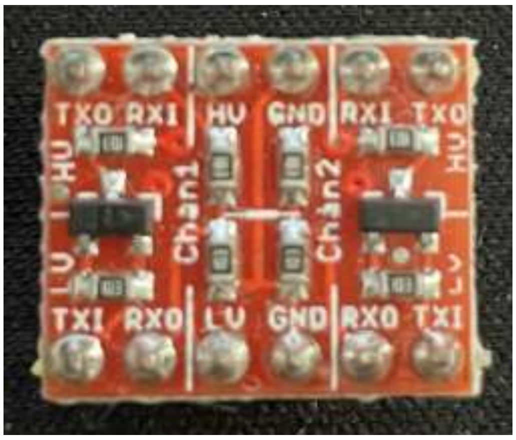


Figure 16 - Logic Converter 3.3 - 5V

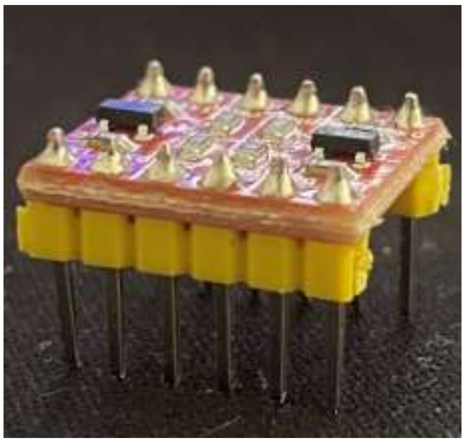


Figure 17 - Logic Converter 3.3 - 5V

As said, we can use it as an add-on device to the Arduino, or we can use it as an Arduino itself. Assuming that we are going to use it as an Arduino itself, the temperature sensor assembling, and the programming took place. Having this stage completed, it was time to test it. While testing the system several problems appeared.

- **Using the ESP8266 as an Arduino**

The first thing that we need to keep in mind is that, however, the coding part and programming language used when coding the ESP module as an Arduino, it's the same as when we are working with the Arduino, there are a few differences. New libraries must be downloaded and used to make the module work. The documentation regarding these libraries must be read, to fully understand how they work. This is not difficult but still, it is time-consuming. Nevertheless, this was done by the author.

More obstacles appeared, however. After the coding of the module, when trying to connect the module to a WIFI network, this connection wasn't always possible. Sometimes the network that we wanted to connect would appear and the module would connect to it. Sometimes this network appeared, but the module couldn't connect to it or if it did connect, the connection sometimes would not be up for long. And of course, there were times when the module, after performing a scan of all available networks, would not detect the one that we wanted.

Another interesting fact observed by the author was the fact that the ESP module was very keen on failing when it had other electronic devices around. Not all electronic devices would cause the malfunction of the module, but giving a practical example, if the ESP module was trying to connect to a WIFI network with a PS4 turned ON next to it, this connection was not established.

Even by having all these problems, the author was able to make the sensor work in the desired way.

When the ESP8266 module is connected to a WIFI network it acquires a local IP address. In this local IP address, is possible to set up a local webpage to display the information received by the sensor but the local webpage did not perform always as expected. Sometimes the webpage would not load, the webpage did not reload automatically, and since the connection between the module and the WIFI network was not always stable and kept on failing. The author then approached this implementation differently.

- **Using the ESP8266 alongside the Arduino**

As said, it is possible to have the Arduino board and WIFI shield interconnected and have the WIFI shield work as an extender for the Arduino board, giving it WIFI possibilities. The process behind it, it's a very simple one. We connect both devices, use the SoftwareSerial library, previously explained to establish a connection between them, and we code and the Arduino board in the traditional way. Using this approach simplifies the implementation. Firstly, we do not need to code the ESP8266 module itself and for that reason, the use of specific libraries is not required. We connect the temperature sensor to the Arduino and through the Serial ports previously defined, send the sensor values to the ESP8266 module which displays the sensor values in a local webpage. Using this implementation allowed for a much stable connection.

Having a local webpage implementation complete, the author began to think of another implementation possibility for the WIFI. Here is where we introduce the MQTT protocol.

b) MQTT[9]

Another way to explore and incorporate this new form of communication in our project is exploring communication between shields and Arduinos. By having two WIFI shields connected to two Arduinos we can now explore the communication between them wirelessly and not via I2C for example.

By having two WIFI shields, we can connect them to each Arduino. By doing this, our goal is to eliminate the I2C communication and communicate strictly via WIFI by using the MQTT protocol (Message Queuing Telemetry Transport).

The MQTT protocol is described as a lightweight publish and subscribe system. It is useful for devices with low bandwidth, where we can send commands, sensor values, or messages over the Internet with little effort. The mechanism behind this protocol is very simple to understand. We begin by having a node, for example, an Arduino board with a WIFI module that sends a payload (message) to a broker. A broker is a kind of "middle-point" server, that stores payloads sent to it in something called topics. A topic is a definition of what type of data it contains. It could be a temperature reading or anything else. We then have another node, for example, another Arduino with another WIFI shield that subscribes to this information, subscribes to this topic, from the broker and the data is moved from node A to node B over the Internet (Figure 18)[10].

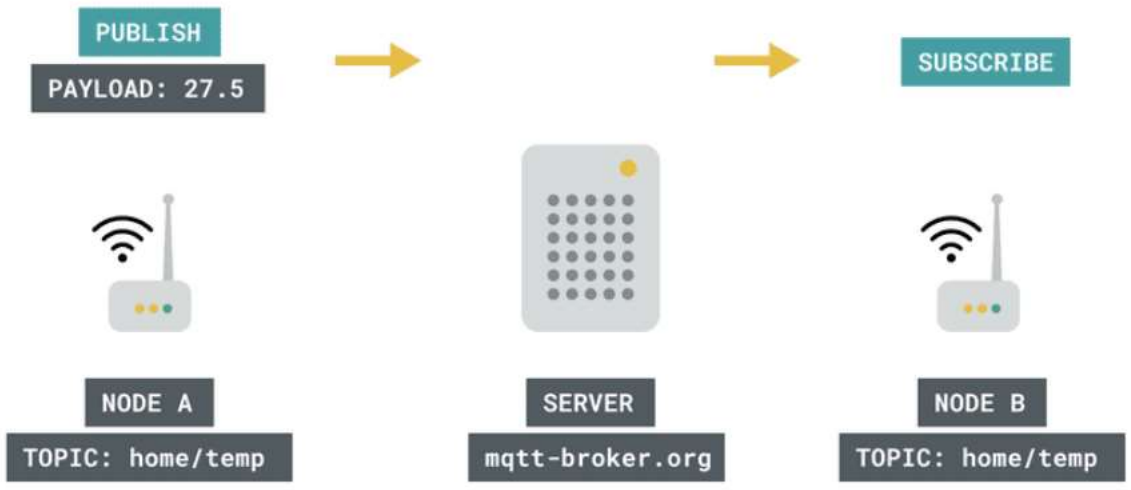


Figure 18 - MQTT Protocol

[9] https://www.arduino.cc/reference/en/libraries/mqtt-client/ (Accessed 06/2021)
[10] https://docs.arduino.cc/tutorials/uno-wifi-rev2/uno-wifi-r2-mqtt-device-to-device (Accessed 07/2021)

Having understood the basics of how the MQTT protocol works, we are now ready to move on to our project.

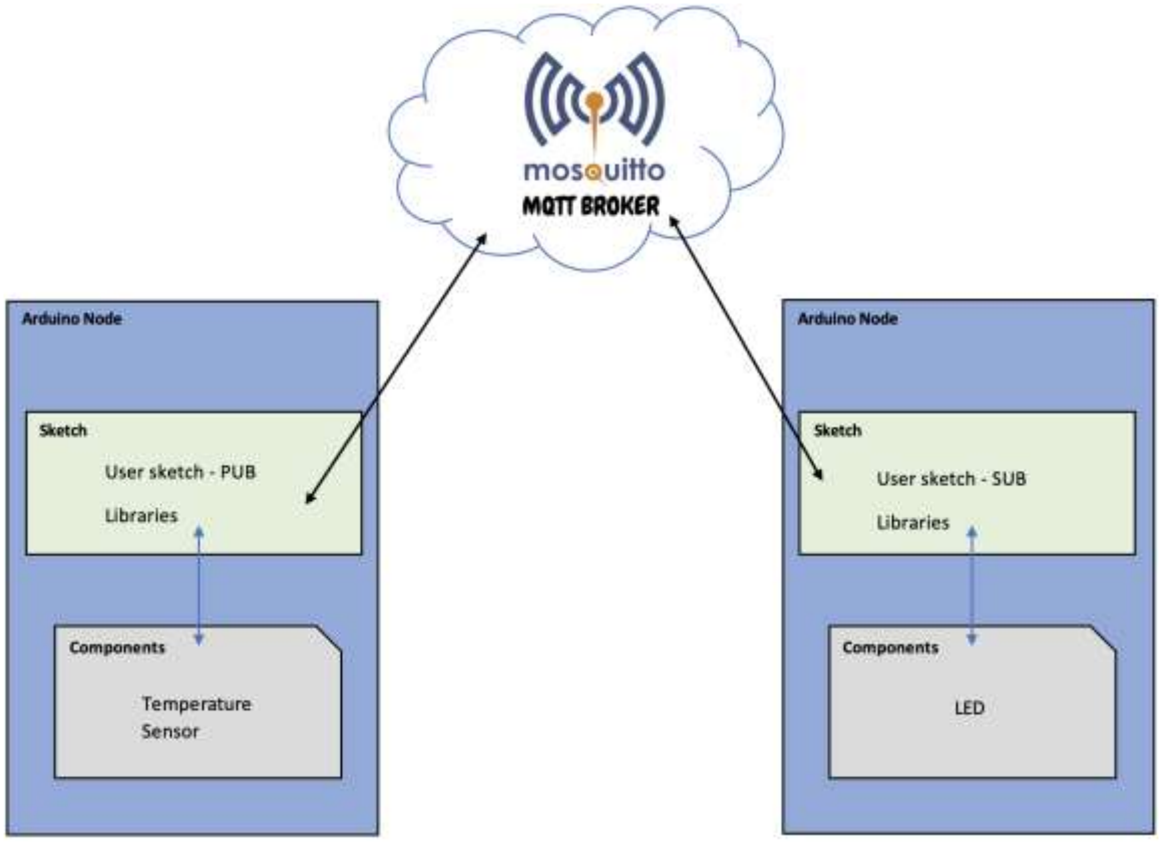


Figure 19 - MQTT protocol in IoT Lab box

To develop the base code for the MQTT protocol we are going to use the temperature sensor and a LED. The objective is the same as in the beginning but now over the internet. In the beginning, every time the temperature sensor reached a threshold value, the LED would light up. Now, we are going to place the temperature sensor in one Arduino Uno and the LED in the other one. The temperature values are going to be sent via the Internet to the other Arduino Uno and if the value is equal or greater than the threshold, the LED will turn ON. We have coded already the sensor and the actuator to perform this task, but now we must add to this code the MQTT protocol library and the WIFI one enabling the communication between the Arduino and the Internet and the MQTT server.

The libraries used to perform these connections are the “WiFiEsp” and the “PubSubClient”. The first one refers to the connection between the Arduino and the Internet. The second refers to the connection between the Arduino and the MQTT server.

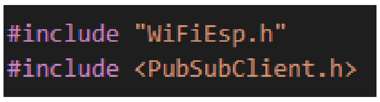
The WiFiEsp[11] library, allows the Arduino board to connect to the internet as said. It can serve as either a server, accepting incoming connections or a client, making outgoing ones. After installing the library,

[11] https://www.arduino.cc/reference/en/libraries/wifiesp/ (Accessed 07/2021)

it has already pre-built examples that we can choose to start our project. We are going to use our Arduino as a client, so we begin by choosing the file that allows us to create a Web client.

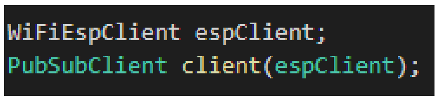
The PubSubClient[12] library provides a client the ability for doing simple publish/subscribe messaging with a server that supports MQTT. To create an MQTT client, we can start by using the example provided by the library. In this example, the WiFiEsp library is already included, so we use this file to begin the construction of our project (Figure 20).

We must first define two very important things. The first one is the WIFI client and the second one the MQTT client, which uses our previously defined WIFI client to set up the connection to the MQTT server (Figure 21).



```
#include "WiFiEsp.h"
#include <PubSubClient.h>
```

Figure 20 – Declare Libraries



```
WiFiEspClient espClient;
PubSubClient client(espClient);
```

Figure 21 – Create WIFI and MQTT client

Based on Figure 22, we can see that we defined a WIFI client named **espClient** and an MQTT client named **client** that uses that same WIFI client to connect to the MQTT server.

Having these parameters set, we are now ready to move on. We are going to have two separate files. One file concerns the Arduino responsible for the publishing part and the second file refers to the Arduino responsible for the subscription. The payload refers to the temperature value read by the sensor. The full code detailing the implementation steps will be available alongside all the code used for this project in the link regarding the author's GITHUB.

The system performed as expected. Initially, both Arduinos Uno connects to a WIFI network. Then they both connect to the MQTT server. Two things to keep in mind. One is the fact that when connecting to the MQTT server, the client must have a username. This username cannot be the same for both Arduinos Uno's because if that is the case, anytime the connection is being established, if it detects two clients with the same username, it won't be possible to connect. It's like having two clients trying to push the other one out since the username is the same. The second one is the fact that the payload we want to send must be a variable of the char type. This means that, if have a variable that is an Integer, we must convert it to Char, and only after this conversion we can send the payload to the server. The server used for this project was the "**test.mosquitto.org**". This server was chosen because it is free. After that, a topic must be created to decide where to publish the payload and the Arduino Uno with the LED must subscribe to this topic to view the values being sent there. The system performed as expected and the LED was able to respond to changes captured by the temperature sensor successfully.

[12] https://www.arduino.cc/reference/en/libraries/pubsubclient/ (Accessed 08/2021)

To this point, the system can communicate via Bluetooth and WIFI. With the Bluetooth module, it is possible to send data and visualize it in an android app. Via WIFI, it is possible to create a local webpage to visualize all the data produced by the sensors and to send sensor values and information via the Internet using the MQTT protocol.

The author began to think and realized that this box could have applications in certain courses directly. The ACIC course previously approached, is a course where the students use the Arduino Uno to perform laboratory guides and have a first contact with the Arduino world. An interesting idea is to see and prove if this IoT Lab box can be adapted to solve the laboratory guides and to improve some of them by adding to this course wireless communications features.

4.3. Adaptation of the IoT Lab Box

As said, this IoT Lab Box is a versatile project with the capabilities of being adapted to the needs of the user. One of the most important tests to enforce in this box is to prove that it can be adapted to solve the ACIC laboratory guides and the implementation of these. These laboratory guides are well defined and are solved to this day using the Arduino Uno board. Now is going to be proved that this box can be adapted to solve these laboratory guides and explore them using wireless communication. This implementation will be divided into several stages, begging with the assembling of all the components used to solve the laboratory guides traditionally.

This was the first stage of our project. All the sensors and actuators were assembled, as explained in the laboratory guides, and the Arduino Uno boards were coded to make sure that all the laboratory guides perform as expected. After this initial phase was complete, it was time to move on to our real project.

We can divide the implementation of the IoT Lab Box into three main phases. The first one is the implementation of the Arduino Uno board with the HC-06 Bluetooth module and the android app. Here the sensors and actuators will be assembled using the Arduino Uno board and we will establish a new form of connection, a wireless one, using the Bluetooth module. To communicate with the board, we will use the developed android app and will make sure that all the results obtained are the same as the laboratory guides, we used as our base. All the libraries used and all the information necessary regarding Bluetooth communication and the development of the android app are the same used as before and will be available to everyone to ensure that all is understood by anyone who wants to use this approach.

The second phase is the development of the communication between the Arduino Uno with the WIFI shield. Again, by assembling all the sensors and actuators to the Arduino Uno board, we will compare the results obtained using this new form of connection, the WIFI one, to the results obtained using the laboratory guides. Again, the libraries used for the development of the WIFI communication, and the code used for this implementation are the same as used while developing the base for this communication, and the code and the implementation details will be available so that everyone who wishes to use this approach has no difficulty while using it.

The final phase is the introduction of the ESP32. The ESP32 is a new device, introduced in this project and was not used in the ACIC course. The decision of using this device was already explained in the previous section and will provide us with a new device to compare to the Arduino and demonstrate a new way of performing the laboratory guides. Since this device is new and none of the previous laboratory guides use this approach, these laboratory guides will be modified to accommodate this new approach and will be available with all the documentation needed.

The IoT Lab box is divided into these three main phases and all the implementation procedures will be explained in detail making sure that everything is explained in the best way possible and a succinct one.

So, to implement these new approaches we must make sure that we have a previous base for comparison. By having all the laboratory guides assembled and, after gathering all the results we can begin the development of our project. All the development will be done to obtain the same results or better ones, giving us a guaranty that the project is performing the way expected.

As previously said, the project is divided into three main phases. The assembling and implementation details will be explained in this next section regarding all the phases.

4.3.1. Phase 1 - (ARDUINO + BLUETOOTH + APP)

This first phase is where we begin to explore a new form of connection. In this phase, a wireless communication method is introduced, the Bluetooth one. To explore this new type of communication several steps were taken to have the system perform as expected.

The first step in this phase is the assembling of all sensors and actuators onto the Arduino Uno board as described in the first laboratory guide. The assembling process is the same, so there aren’t any surprises and new information needed for this part. After the assembling process is complete, we began by using the code used in the first stage of the process and verifying once more if the behavior of the system is the same as before. Having this part complete, we can now begin to assemble the HC-06 module. As described in the previous section, this module has 6 pins, but we are only going to use 4 pins. The pins necessary are the VCC, GND, TX, and RX. By identifying these pins, we then must connect the VCC and GND to the corresponding VCC and GND pins of the Arduino Uno board and interconnect them. The Arduino Uno board has already TX and RX hardware pins, D1 and D0, but these pins cannot be used, or else the Serial Communication between the Arduino board and the computer is lost. This means that the uploading process and the visualization of the values outputted by the board and the Serial Monitor are not possible. To overcome this limitation the “SoftwareSerial” library is used, as it was before.

After declaring these new software pins, the communication between the Arduino board and the HC-06 module is now established. Simple as that. Completing this stage is key because now we can move on to the final stage of this phase, the android app. This core development of this app was made previously and now it requires adaptation to accommodate the new user needs.

Having the communication between the Arduino Uno board and HC-06 module all set up, we now must establish a communication between the HC-06 module and the android app to enable the possibility of visualizing the data outputted by the sensors and to allow a 2-way communication (send/receive messages) from the board to the app and vice-versa.

Previously, the android app had already Serial Monitor capabilities and these are the ones needed.

- **LAB GUIDE 1**

Let’s start with the laboratory guide one. In this guide, the system has one push-button and 4 LEDs. When the push button is pressed the LEDs will turn ON and start flashing in a sequence. If the push button is pressed again the system pauses and when is pressed again, the LED’s flashing sequence is resumed. The author modified the system to send a message to the app informing which LED is ON. By having this message, imagining that we do not have enough physical LEDs, the laboratory guide could be performed the same since now, it is possible to know which LED is always ON, even if we do not have a physical one.

- **LAB GUIDE 2**

Lab guide two is assembled using three sensors and three LEDs changing their behavior based on the sensor's one. The yellow LED turns ON anytime the temperature sensor obtains a value equal to or higher than a certain temperature. The LDR (photoresistor) dims the red LED based on the light intensity. If the intensity is high, the red LED dims its light and vice-versa. The green LED flashes with a certain time interval depending on the rotation applied to the potentiometer.

There are several changes and possibilities to be made to the IoT Lab box to implement this lab guide. The author chose the following one. The Arduino is going to send to the app the values read by the sensors: the temperature of the room where the sensor is, the light intensity read by the LDR, and the potentiometer reading. This will allow for the user to explore and add more information to the Serial Monitor built and remove the necessity to use the Serial Monitor of the Arduino IDE. Since the temperature threshold is decided by the user before the upload of the code to the Arduino and to explore the 2-Way communication between the Bluetooth module and the Arduino Uno, the app now allows changing this threshold anytime the user wants. By having this possibility, when we want to change the threshold value which leads to the yellow LED to light turn ON, instead of uploading a new code to the Arduino, we do not have to do it. We simply change the value in the app and send it to the Arduino Uno (Figures 22 and 23).

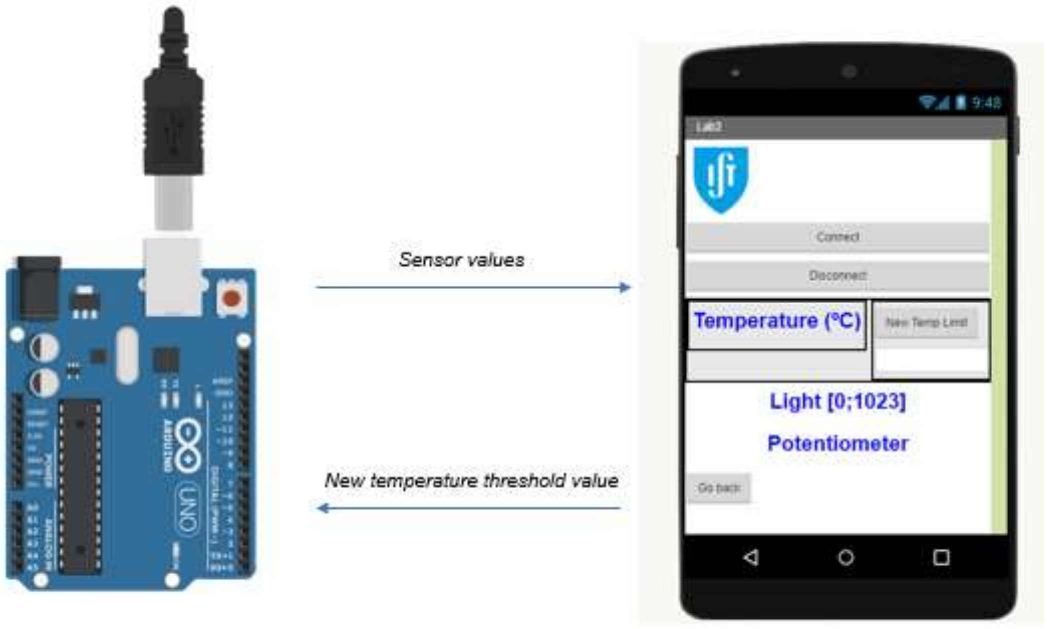


Figure 22 - Bluetooth Communication between Arduino and App

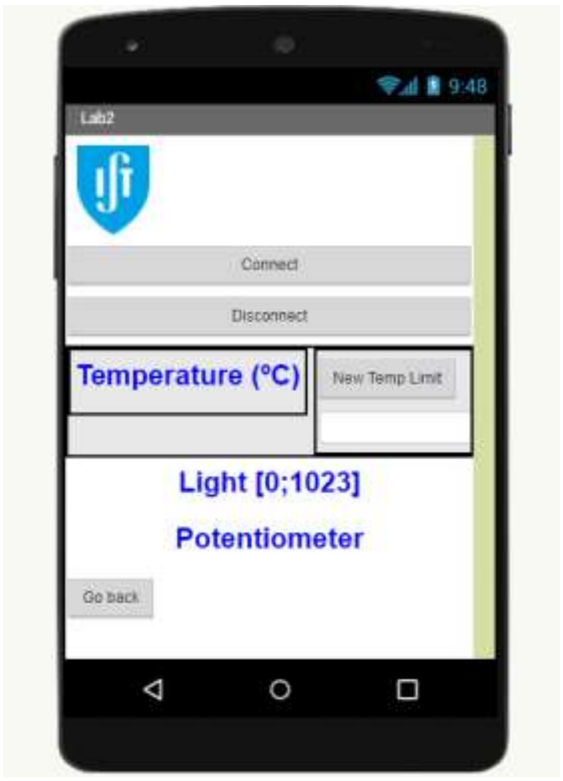


Figure 23 - Mobile App interface

4.3.2. Phase 2 - (ARDUINO + WIFI + MQTT)

After completing phase one, we are ready to move on to the next stage of our IoT Lab Box project.

In this phase two, we are going to explore another form of wireless communication, WIFI. In the previous section, the WIFI shield and its module were described and how do they work. A few problems were found while developing this new method of communication, also described in the previous section alongside how these limitations were overcome.

The initial procedure to assemble the components, sensors, and actuators, is the same as always. First, we must assemble all the sensors onto the Arduino Uno board and after being completed with this assemble procedure, we must test it. By uploading the code, previously developed, we upload it to the Arduino Uno board and see the results obtained. If the results obtained are the results expected, we can move on to the next stage of this phase.

By introducing a new method of communication, new libraries are also introduced and needed for the correct behavior of our system. Th WIFI shield is connected to the Arduino Uno board and acts as an extender, simple to provide WIFI capabilities to this board. As explained in the previous section, the possibility of using the WIFI shield as an Arduino itself it's possible but the procedures needed to make it work properly are rather complicated and for someone that is not as experienced in these kinds of systems, for example, a student who studies these subjects, it's very complicated. The goal to achieve is not to complicate the learning process of someone, it's the opposite. For that and other reasons, the use of a WIFI shield as an Arduino itself was not a viable way.

After having all the components assembled and after verifying that everything is working as expected, the WIFI shield is ready to be assembled interconnected with the Arduino board. The IoT Lab box project adaptation to the ACIC laboratory guides regarding the use of the WIFI is going to be implemented using the third laboratory guide.

The WIFI shield is assembled and once again the SoftwareLibrary was used to enable the communication between the shield and the Arduino board, but this library alone is not enough for us to be able to achieve a connection between the shield a WIFI network as explained. Here, the use of an additional library is required by the user. The library used is the "PubSubClient" and the "WiFiEsp" one. This is the same method previously used.

Having established a connection between the shield and a WIFI network we were ready to move on.

By having the Arduino Uno board connected to a WIFI network, we are now able to communicate with the cloud and adapt our lab box to be able to implement the laboratory guide number three, performing it in a new way. The first one is to use the local IP address obtained by the shield when connected to the WIFI network and set up a local webserver with that IP and send all the values and results obtained from the Arduino and displayed on a webpage. The second one is to implement the MQTT protocol in the laboratory guide.

- **LOCAL WEBPAGE**

In the third lab guide, we have actuators and sensors. The actuators respond to changes occurring on the side of the sensors. These are the same sensors and actuators used in lab guide two. Two view the results we must observe the behavior of the actuators and the outputted values given by the sensors in the Serial Monitor to see if they are responding to the correct changes.

By having the WIFI shield connected we can now make some changes to use this new form of connectivity. After connecting the WIFI module and connecting it to a WIFI network, the module acquires a local IP address. By having this IP address, we can set up a local webpage to visualize all the data outputted by the sensors making the use of the Serial Monitor not necessary anymore. Using this new method, we send all the values from the sensors to the cloud and after that display this information in the local webpage previously created.

The previous limitations encountered, were observed once more. The connection was not always stable and now since we are displaying more data the webpage sometimes did not load. Again, another limitation is the fact that the webpage does not refresh automatically and so to visualize the results in real-time we must press the refresh button repeatedly. This task may cause the crash of the system since the refreshing procedure may coincide with the upload of the data to the webpage.

Although it is possible to build a Serial Monitor webpage, the limitations encountered when working with the Arduino Uno, make this solution not viable when trying to work with more than one sensor.

- **MQTT**

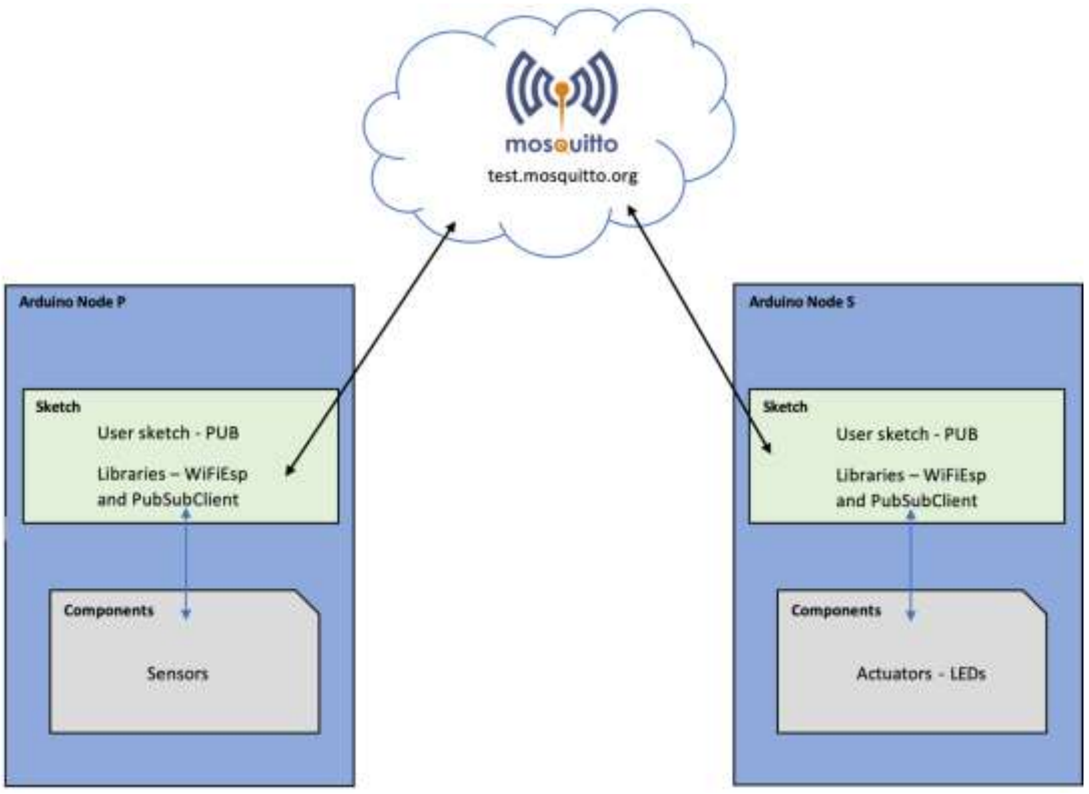
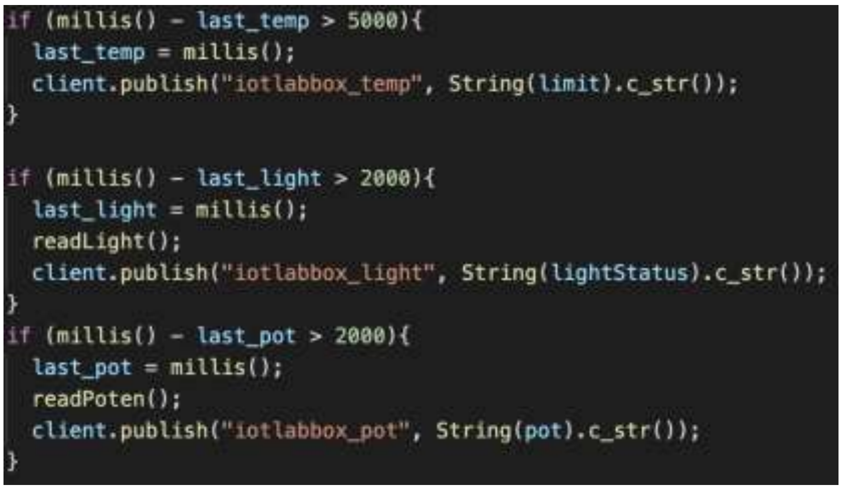


Figure 24 - Functional Architecture between Arduinos with MQTT protocol

To implement laboratory guide number three using the IoT Lab box, some adaptations must be made. We can start by combining the code used to develop laboratory guide number three traditionally with the one produced while developing the code base for our IoT Lab box. We are going to define two nodes. The first node, node P, is where we find the Arduino Uno Board with the WIFI shield and the three sensors. This node, P, is responsible for Publishing the values obtained by the sensors to a topic. The second node, node S (Subscription node), is where we find the other Arduino Uno with the WIFI shield and the actuators, the LEDs. As said, the behavior of the actuators is directly associated with changes registered by the sensors of the surrounding environment.

Using the code previously built to solve the third lab guide as our base, we now must make changes and use new libraries so that we can create a WIFI client, connect the WIFI client to the MQTT client, and after that use this WIFI client to publish the payload to the broker (Figure 24). To simplify the sending of the payload from the Arduino to the MQTT broker and since we have three sensors, the author created three topics regarding each sensor. This MQTT client is publishing the readings obtained from the sensors in a pre-established time interval (Figure 25).

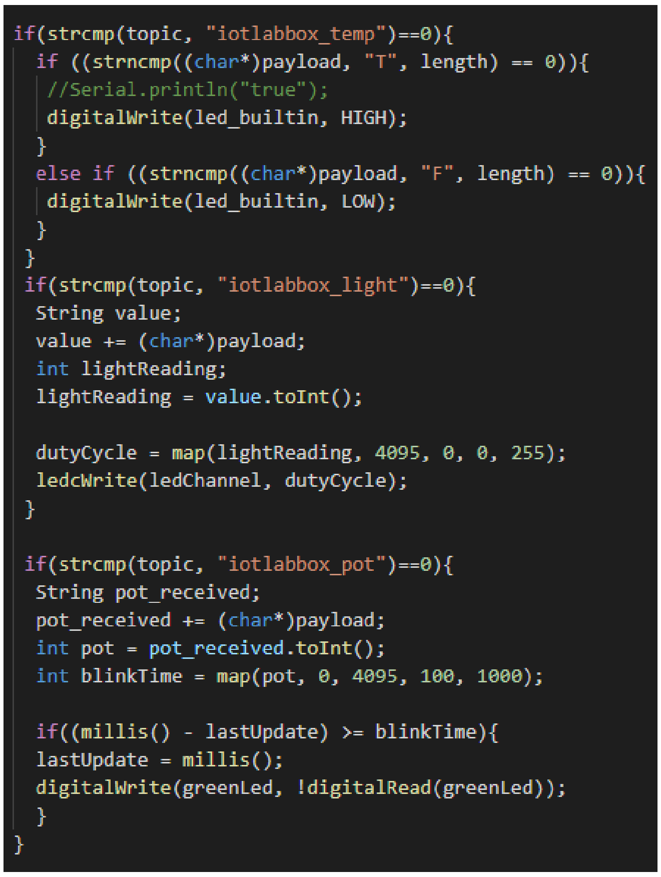


```
if (millis() - last_temp > 5000){
  last_temp = millis();
  client.publish("iotlabbox_temp", String(limit).c_str());
}

if (millis() - last_light > 2000){
  last_light = millis();
  readLight();
  client.publish("iotlabbox_light", String(lightStatus).c_str());
}
if (millis() - last_pot > 2000){
  last_pot = millis();
  readPoten();
  client.publish("iotlabbox_pot", String(pot).c_str());
}
```

Figure 25 - Sending values (PUB node)

As shown in Figure 27, each sensor value is being sent to the MQTT server into the corresponding topic. This allows for a much clearer reading of the information from the Arduino Uno subscribing to the topics since the information subscribed to is organized. The time interval in which the payload is sent to the server is also important, giving time to the system to process the information sent and received.

On the Arduino with the actuators, we must create a WIFI client and connect it to the MQTT client, and now subscribe to these three topics to receive the messages sent by the sensors. After receiving the messages, the actuator's behavior is now altered accordingly to the value received (Figure 26).



```
if(strcmp(topic, "iotlabbox_temp")==0){
  if ((strncmp((char*)payload, "T", length) == 0)){
   //Serial.println("true");
   digitalWrite(led_builtin, HIGH);
  }
  else if ((strncmp((char*)payload, "F", length) == 0)){
   digitalWrite(led_builtin, LOW);
  }
 }
 if(strcmp(topic, "iotlabbox_light")==0){
  String value;
  value += (char*)payload;
  int lightReading;
  lightReading = value.toInt();

  dutyCycle = map(lightReading, 4095, 0, 0, 255);
  ledcWrite(ledChannel, dutyCycle);
 }

 if(strcmp(topic, "iotlabbox_pot")==0){
  String pot_received;
  pot_received += (char*)payload;
  int pot = pot_received.toInt();
  int blinkTime = map(pot, 0, 4095, 100, 1000);

  if((millis() - lastUpdate) >= blinkTime){
  lastUpdate = millis();
  digitalWrite(greenLed, !digitalRead(greenLed));
  }
}
```

Figure 26 - Receiving values(subscribing)

Having all the coding done in both Arduinos, we are now ready to test our project. We connect the two Arduinos and the corresponding WIFI shields to a WIFI network and verify if we obtain a successful connection. Having this done, we then must verify if the MQTT client connection to the broker is established. Having these two initial steps verified, we can begin to send messages.

Let's begin with the Arduino with the sensors. To verify if this Arduino is sending the messages to the broker and the corresponding topic correctly, we can print the messages sent and can also subscribe to the topic desired using the terminal in our computer. By having the MQTT installed in our computer, we must open a terminal window and write the following command[13]: "**mosquitto_sub -h test.mosquitto.org -t "topic/subtopic" -v**". If we subscribe to the topic and we verify that the messages are being sent and are appearing in the terminal, we are successfully sending the payload to the broker and the correct topic. After this, we move on to the Arduino with the actuators and print all the messages received to see if they are the same as the ones we are viewing in our terminal. This is a very good practice since our computer has a lot more computational power when comparing it to the Arduino and because of that, it can receive the messages more rapidly. If the messages match, we can assume that the Arduino is working correctly. In our specific case, the command to subscribe

[13] https://mosquitto.org/documentation/ (Accessed 07/2021)

to one of the topics would be the following: “**mosquitto_sub -h test.mosquitto.org -t “iotlabbox_temp” -v”.**

After verifying if the messages are being sent and received, it’s time to observe the behavior of our system. Initially, the system, the system performs the way expected. The messages are sent and received and when the Arduino that holds the actuators receives the messages, the actuators behave correctly. When looping the program for a few minutes, limitations and problems began to appear.

The Arduino begins to disconnect from the internet and because of that loses connection to the MQTT broker. By losing this connection, the messages are not delivered, and the actuators do not perform as expected.

This could be explained due to fact that the Arduino does not have a computational power strong enough to process all the incoming messages and when a timeout has been reached the connection is lost. In addition to this problem the fact that the WIFI network we are connected to and the fact that the broker used is a free broker with heavy traffic, all add up to prove that using Arduino boards with this goal may not be viable. Both the library used for the WIFI and the MQTT connection, have a timeout parameter to allow the device using them to process the incoming messages. When this timeout is achieved, the connection is lost and the process to restore the connection begins automatically. This means that all the incoming messages are lost, and the system stops. The author began to find a solution to fix this limitation.

○ FIXING THE TIMEOUT PROBLEM

To fix this problem, a solution was found. The author began by upgrading the firmware present into the WIFI shield, more specifically, upgrading the firmware present in the ESP8266 module. This upgrade was necessary because now, we are going to use a new library. This new library is a newer version of the library used to enable the possibility of connecting the Arduino to a WIFI network. The process to upgrade the firmware is a process rather complicated and may sometimes fail. The existent documentation for this process is not very clear and it requires some trial and error when performing this process. After upgrading the firmware and changing the Arduino code to use this new library, we tested the system once again. The code changes are almost null. We must change two lines in our code. The name of the library that we are importing and when creating a WIFI client, we must now use a method from this new library.

The system behavior was once more the expected one and for quite some time the system did not crash. After some time, the same problem appeared again, and the timeouts were once again reached, and the actuators began to fail. Although the messages from the Arduino with the sensors were always sent without a problem, the Arduino Uno simply does not have the computational power necessary to keep the connection up while receiving the messages, making the use of an Arduino to subscribe to topics to receive messages, not a viable option.

Although it is possible to publish messages to a topic without a single problem, having another Arduino subscribing to these messages becomes impossible after some time and the system stops completely.

o FIXING THE HIGH LOAD PROBLEM

In another attempt to solve this problem, the author after upgrading the firmware and updating the code with a newer library thought of a way to fix the high load problem. To fix this, we used another WIFI network, one with much less load to see if the problem was fixed. Using a smartphone to act as a hotspot and disabling the internet for all the apps on the phone, the author tested the system. As in the previous case, initially, the system works without a problem but again after some time the limitations start to appear and the subscription part of the system begins to crash, proving once more the limitations of hardware itself.

o FIXING THE PUBLIC BROKER PROBLEM

Another cause for our problem is the fact that to use the MQTT, we use public and a free broker to act as our server. Since it is a public server and a free one, it is used by a variety of devices and people and having very high traffic can cause our system to crash. To fix this problem, the author set up a local broker, using his computer as the host. After setting up this local broker, the system was tested. Using this solution is where is see the biggest improvement. The system performed as expected for the longest time since now the broker does not have high traffic passing through the network. Although this solution was the one that provided the longest operating time by the system and the most stable, the limitations of the hardware were encountered after some time, leading us again to the fact that the Arduino has computational limitations in subscribing to all these messages and is not the viable solution to perform a project of publishing and subscribing to messages. It is possible to do it, but not recommended.

This approach, although proven to be the most viable one by keeping the connection alive for the longest as a negative side. The setting up process of a new broker, in this case, the author's personal computer is not hard but can cause some problems since it involves changing security parameters and enabling Ports in our computer. This process for someone who understands software may not be complicated but for someone who does not, the process is not recommended. Adding this fault to all the other limitations and problems encountered, prove that using the Arduino Uno for this purpose should not be considered.

The user, who wishes to use this project may choose to use this platform to do it, but the results will not be the greatest. If all the user desires are to, publish messages into a topic and visualize them, then it is a viable option. If the user wants to publish many messages and has another Arduino device to subscribe to these topics to visualize the messages, then an alternative must be used. A good implementation of the IoT Lab box alongside the MQTT protocol is for example to have all the groups of this course publish the values obtained by their sensors into a specific topic. By doing this, the MQTT communication and the WIFI one are explored, and the system will perform without fault.

In the previous topics, applying the IoT Lab box its brain, the Arduino, to a specific course was described and the behavior of the system was explained. When using the WIFI shield to connect it to an MQTT broker, the limitations of this device were explained as were the possible solutions to overcome these problems. If we want to send a small number of messages and subscribe to these, the Arduino will work just fine but keep in mind that we cannot do much more than that. If the user desires to increase the number of messages and increase the load of work for the Arduino, the limitations will be soon reached and the goal to achieve will not happen.

As said, the IoT Lab Box originally was to be implemented with Arduino Uno boards only but when developing the project, new alternatives to implement the project were found, mainly the use of the ESP32. Here is where we enter phase three of our project.

4.3.3. Phase 3 - (ESP32)

The ESP32 was an alternative found to the Arduino Uno to develop this IoT Lab Box. It exceeds the Arduino Uno in computational power, and it is easier to work with. This device has already built into it, a Bluetooth module, a BLE module, and WIFI. By having these features, the need for external hardware is no longer necessary, making it easier to assemble and transport.

Phase three is divided into several stages. The first one is the development of the project using the BLE module. The second one is where we develop the project using the WIFI feature.

The implementation of the ESP32 in our IoT Lab box was directly at the previously developed laboratory guides of the ACIC course with the corresponding changes since coding this device is very similar to an Arduino Uno.

The IoT Lab box has already a classic Bluetooth module connected to it, allowing communication via smartphone with the Arduinos Uno present in the box. Because of it, we are going to start using the ESP32 by exploring the Bluetooth Low Energy feature. As said, the ESP32 has also incorporated a classic Bluetooth module like the one used and for that reason will not be used.

4.3.4. Phase 3.1 - (ESP32 + BLE)

Using the Arduino Uno to develop our project, the author used a classic Bluetooth module with an Android app.

As said, the ESP32 has already built into it a Bluetooth module and a BLE module[14] as well. The main difference between the BLE module and the traditional Bluetooth is the fact that the BLE has a lower power consumption, making it the better choice for a long connection.

[14] https://www.arduino.cc/reference/en/libraries/esp32-ble-arduino/ (Accessed 09/2021)

The idea behind the implementation of the BLE is the same as it was while using the classic Bluetooth. We are now going to develop a connection between the ESP32 module and the android app using the BLE communication allowing the user to have a Serial Monitor present in the app, making the Serial Monitor present in the Arduino IDE not necessary anymore. By having already an android app with the sketch needed to receive the communication from the three sensors used previously, the author assembled the same three sensors in the ESP32. The assembling procedure of these three sensors is the same as in the Arduino, we simply must read the datasheet from the ESP32 module to know where to connect the sensors and the actuators.

Having the sensors and actuators connected, we initially begin by testing these. Using the previously developed code, the author made the necessary changes to it to make everything work. After the assembling procedure, we use the Serial Monitor present in the Arduino IDE to see if the sensors are outputting values and if these values make sense given the context. Then we must verify if the actuators are acting accordingly with the values read by the sensors. All these steps were concluded with success and the author was now ready to move on to the BLE communication development.

When using the Bluetooth classic module, for the correct behavior of this module, the SoftwareSerial library was necessary to enable the communication between the module and the Arduino Uno. Now, since the BLE module is already incorporated into the ESP32, this library is not necessary. Nevertheless, new libraries must be used to enable this communication. These new libraries must be used for the BLE capabilities.

To allow the communication between the ESP32 BLE module and the android app, several steps must be completed:

1. **Create BLE server**
2. **Create a BLE Service**
3. **Create a BLE Characteristic on the Service**
4. **Create a BLE Descriptor on the characteristic**
5. **Start the service and Start advertising.**

In BLE communications, devices address each other using a EUI-48 Media Access Control (MAC) address. The Extended Unique Identifiers (EUIs) are assigned to the BLE and other network-enabled devices during the manufacturing.

There are essentially two protocols that are important in communication between two BLE devices, the **GAP,** and **GATT.** Understanding how these two works, is extremely important to programming devices to communicate via the BLE protocol.

- **GAP Protocol**

  GAP stands for **Generic Access Profile** and its responsible for controlling the connections and advertising – Making a device visible and open for connection – in Bluetooth. It defines the roles which devices play in communication and determines how advertising payload is broadcasted.

There are essentially two roles that BLE devices can play based on GAP, **Central Device,** and **Peripheral Device.** These two devices are the BLE's representation for the most common designations, **Client** and **Server** respectively. In IoT solutions, the peripheral devices are usually sensors while the Central devices are usually gateways like smartphone devices. Once a connection between a central and a peripheral device is established, the advertising process will stop, and the **GATT** services and characteristics kick in to facilitate the communication in both directions between these two devices.

- **GATT Protocol**

GATT means **Generic Attribute Profile**, and it defines how two BLE devices, transfer data back and forth between each other, using concepts called Services and Characteristics. It makes use of a generic data protocol called the **Attribute Protocol (ATT)**, to store Services, Characteristics, and related data in a simple lookup table using 16-bit IDs for each entry in the table.

Services are used to group the data into logic entities and contain specific parts of data called characteristics. A service can have one or more characteristics and each service distinguishes itself from others through a unique numeric ID called **UUID** that can be either 16-bit or 128-bit.

Characteristics represent the lowest level concept in the GATT structure. It encapsulates a single data point and just like services, it is distinguished from other characteristics using a Unique numeric ID, the UUID. Characteristics are the major containers that carry data between two devices.

Based on this, to implement the BLE into our project we must define which is the Central device and which is the Peripheral one. The Central device is our smartphone device, containing our app. The Peripheral device is the ESP32, containing the BLE module. Our ESP32 will act as a BLE server to send the sensor data and the android will act as a BLE scanner to find the server and receive the data. Both the server and client will have a **Service UUID** to make a connection between them. We will use two characteristics TX and RX to send data and receive data from the client. In our specific case, the android app will not send data to the ESP, so we only need the TX characteristic from the ESP32. It is important to keep in mind that, the TX characteristic in the ESP32 is the RX characteristic in the android app (Figures 27 and 28).

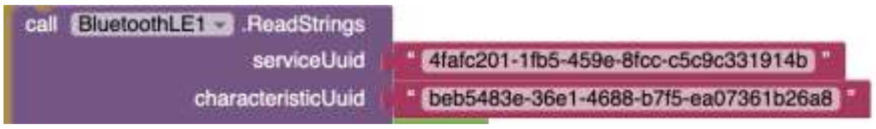
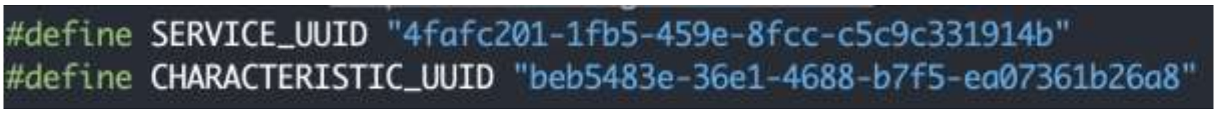

```
call BluetoothLE1 .ReadStrings
            serviceUuid  " 4fafc201-1fb5-459e-8fcc-c5c9c331914b "
     characteristicUuid  " beb5483e-36e1-4688-b7f5-ea07361b26a8 "
```

Figure 27 - Declaring UUID in App

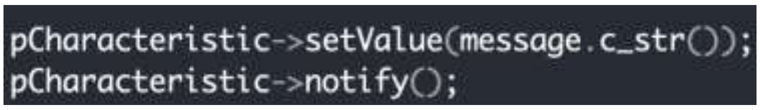


```
#define SERVICE_UUID "4fafc201-1fb5-459e-8fcc-c5c9c331914b"
#define CHARACTERISTIC_UUID "beb5483e-36e1-4688-b7f5-ea07361b26a8"
```

Figure 28 - Declaring UUID in ESP32

To generate a random UUID for any device, we can use the website https://www.uuidgenerator.net/.

We then must initialize the ESP32 as a BLE device and set its name. We must create a BLE server and a BLE service using the Service UUID previously generated and add the characteristics. After that, we start the BLE service and start the advertisement. Having started the BLE server, we use the smartphone to find the created server and connect to it, to allow data to be transferred. When the smartphone is connected to the ESP32 BLE server, the module begins to send the sensor data to the app and the data can now be visualized in the Serial Monitor developed. There are two important methods responsible for the sending of the data. The **setValue()** indicates the message that the characteristic will send to the smartphone. The **notify()** method is responsible for the actual sending procedure (Figure 29).



```
pCharacteristic->setValue(message.c_str());
pCharacteristic->notify();
```

Figure 29 - Sending value through BLE

All the implementation code regarding the ESP32 and the android app will be fully available in the author's GITHUB.

4.3.5. Phase 3.2 - (ESP32 + WIFI + LOCALWEBPAGE)

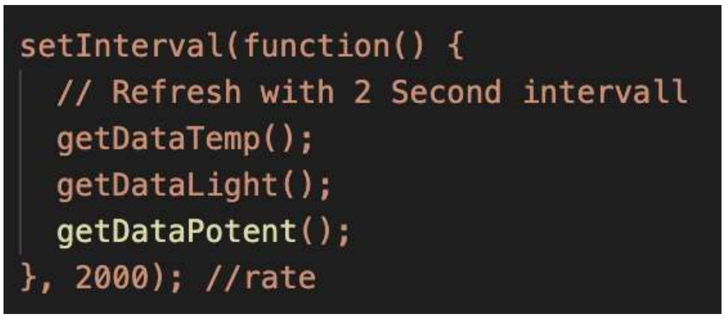
Having a module with WIFI incorporated is something very interesting. Compared with before, now we do not need to have external WIFI shields connected to our main device to provide us with a possibility of being connected to the Internet. Based on this, our box can implement the laboratory guides from the ACIC course also using the ESP32.

Using the Arduino Uno approach, we were able to create a local webpage to act as a Serial Monitor. The goal while using the ESP32 is the same but an improved one.

As said, the coding procedure of the ESP32 is almost the same as in the Arduino Uno. We must import libraries to enable the possibility of connecting our ESP32 to the Internet. Although it is already a WIFI module, we still need to tell the ESP how to connect and to where.

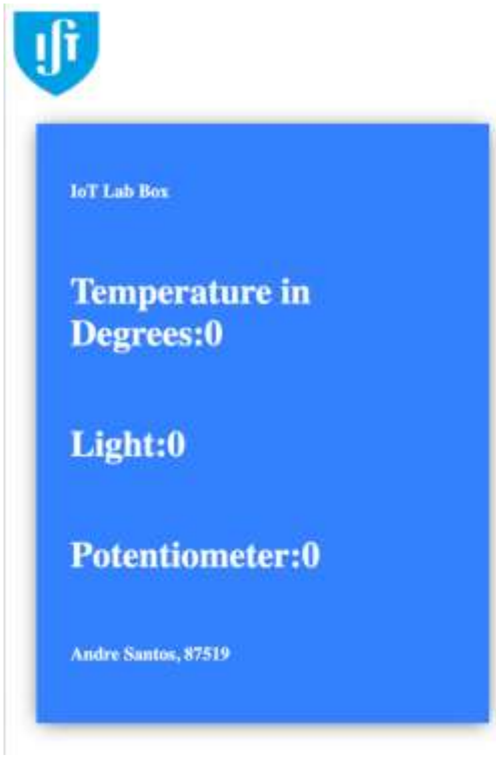
The first library to include is the "WiFi" one. This allows for the ESP32 to connect to the Internet. It can serve as either a server or a client. After including this library and specifying the network name and password, the ESP32 can now connect be connected to the Internet.

After having a successful connection to the Internet, we set our web server to a specific port, port 80. Using the previously created methods regarding the reading of the sensors, we can obtain the values obtained by them. Our webpage will be located, like before, in the local IP address obtained by the ESP32 when connected to the Internet, just like the Arduino Uno. When we open the browser in that specific IP address the webpage will appear as will the sensor values but know with a great advantage. While using the ESP32, we can have an external **header** file creating our webpage. This file is then called by the Arduino code to start this webpage when we start the project. This allows for a cleaner and a more complex webpage will design possibilities. But the best part of having this header file is the fact that we can add a method allowing the webpage to update automatically in a defined time interval (Figure 30). By doing this, we now do not need to refresh the page manually and can see in real-time the values read by the sensors and all the changes registered by them. Now we have a fully functional Serial Monitor set up in a local webpage, taking advantage of the WIFI feature of the ESP32 and its bigger computational power (Figure 31).



```
setInterval(function() {
  // Refresh with 2 Second intervall
  getDataTemp();
  getDataLight();
  getDataPotent();
}, 2000); //rate
```

Figure 30 - Function to perform a refresh action



IoT Lab Box

Temperature in Degrees:0

Light:0

Potentiometer:0

Andre Santos, 87519

Figure 31 – Webpage Interface

Having this header file to create this webpage, has another big advantage. Since this header file gets the data directly from the methods present in the Arduino Uno code, when we add more sensors to expand the different values received from the sensors, we simply add the code to the header file, and it automatically creates a new field in the webpage with that sensor. It has no limitation.

Having adapted our IoT Lab box with the ESP32 as the main device to the laboratory guide is now time to explore one more application possible to the WIFI module. Like while using the Arduino Uno, the MQTT protocol was explored, and limitations were found. Since the ESP32 is a device with more computational power than the Arduino Uno, it will be interesting to see if these limitations also occur when using this device.

4.3.6. Phase 3.3 - (ESP32 + WIFI + MQTT)

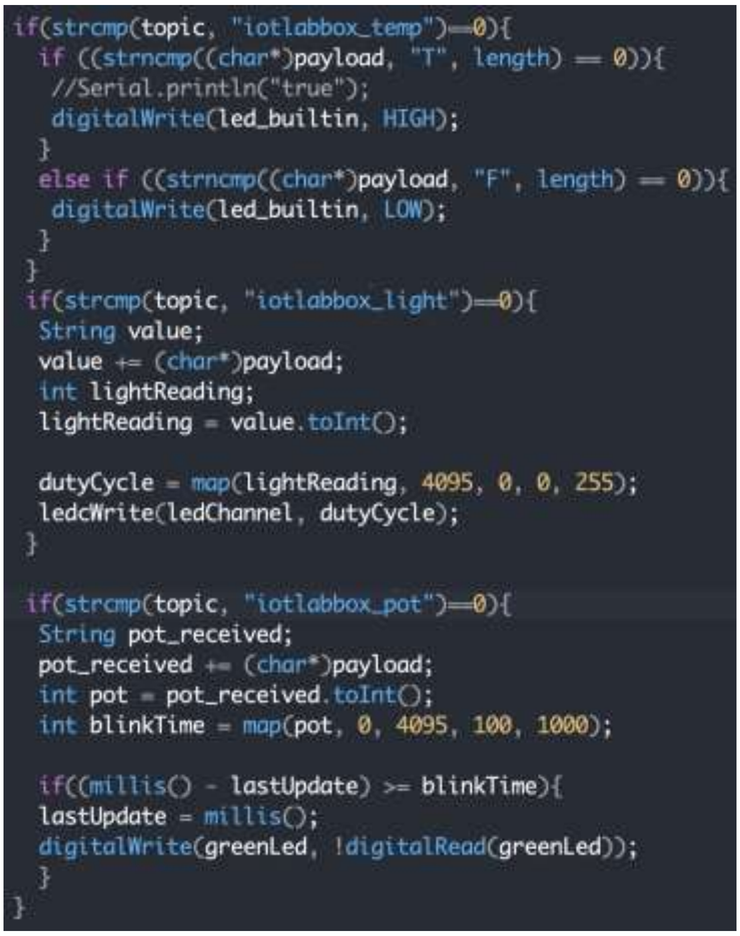
While adapting our IoT Lab box to implement the ACIC laboratory guides, the author implemented the MQTT protocol in the project. Now, we are trying the same approach but using the ESP32.

In the previous phase it was described how can the ESP32 connect to the Internet using the “WiFi” library. To implement the MQTT protocol, this library must use alongside the known “PubSubClient” library.

To implement this protocol, we need two ESP32 devices. One device will act as the client publishing the payload to the broker and the second will be the client subscribing to the topics where the payload is located. The implementation details are the same as in the stage where we implemented the MQTT protocol using the Arduino Uno but now the results obtained are different.

While using the Arduino we observed the failure to maintain a connection between both Arduinos to the internet and the MQTT server when several sensor values were being sent. While using the ESP32, this limitation was not observed. Because the computational power of the ESP32 is superior to the one found in the Arduino, the connection stayed alive for as long as it was kept by the author. The payload containing the sensor values was published to the same three topics and the ESP32 responsible for subscribing to these topics and storing the sensor values causing changes to actuators was able to do so.

To implement these behavior changes into the actuators, the ESP32 code has a **callback function** responsible for verifying if new payloads are arriving at the subscribed topics. If that verification returns true, we know that a new value has arrived at the topic, and the actuators act accordingly (Figure 32). One very important thing to look out for is the fact that the payload publishes to each topic has the char type and we must convert it to an Integer for the actuators to work.



```
if(strcmp(topic, "iotlabbox_temp")==0){
  if ((strncmp((char*)payload, "T", length) == 0)){
   //Serial.println("true");
   digitalWrite(led_builtin, HIGH);
  }
  else if ((strncmp((char*)payload, "F", length) == 0)){
   digitalWrite(led_builtin, LOW);
  }
 }
 if(strcmp(topic, "iotlabbox_light")==0){
  String value;
  value += (char*)payload;
  int lightReading;
  lightReading = value.toInt();

  dutyCycle = map(lightReading, 4095, 0, 0, 255);
  ledcWrite(ledChannel, dutyCycle);
 }

 if(strcmp(topic, "iotlabbox_pot")==0){
  String pot_received;
  pot_received += (char*)payload;
  int pot = pot_received.toInt();
  int blinkTime = map(pot, 0, 4095, 100, 1000);

  if((millis() - lastUpdate) >= blinkTime){
  lastUpdate = millis();
  digitalWrite(greenLed, !digitalRead(greenLed));
  }
}
```

Figure 32 - Receiving messages (subscribing)

When comparing the two approaches we realize that while we cannot use the Arduino Uno to perform a publish and subscribe routine without occurring problems, the opposite can be said when using the ESP32. The extra computational power that this device and the fact that the implementation of the ACIC laboratory while using the ESP32, is proof that using the ESP32 is the most viable solution to implement the MQTT protocol.

4.4. Implementation Conclusions

Based on the previous implementation details of our IoT Lab box adaptation to the laboratory guides from the ACIC course, is time now to make a resume to all the results obtained and formulate conclusions to help us decide if the box is a viable solution as a learning platform and to what purposes is it viable to.

As explained the IoT Lab Box was structured in stages. We started by considering only one sensor to develop the basis for our work. Assembling the sensor, code it was verifying the behavior of the system was fundamental before advancing any further.

Having completed this stage, the Bluetooth implementation took place. As described in the previous chapter, the author had a BLE shield provided to him by the teacher to develop this communication.

After realizing the limitations of this device, the author decided to acquire a Bluetooth classic module, the HC-06. This device was implemented, and an android app was developed, enabling the communication between the Arduino Uno and the smartphone to visualize the data read by the sensor in the app. The author built a mobile Serial Monitor. The development of this Bluetooth communication was a success, and the Arduino Uno was able to send the data from the sensor to the android app. This implementation leads us to conclusions:

- By having a mobile Serial Monitor, using the Serial Monitor from the Arduino IDE is not necessary anymore.
- By having this possibility and having a wireless form of connection to the box, the user does not need to perform a physical connection to the box and can place the box anywhere desired and the interaction to it can be made.

The next step in the development of the IoT Lab box was the WIFI communication.

It was provided to the author by the teacher, a WIFI shield using the ESP8266 module. The author defined two phases for the implementation of this device. The first one was the development of a webpage to visualize the data from the sensor in it. The second one was the implementation of the MQTT protocol and by using a second Arduino Uno device, establishing communication between both devices.

By connecting the WIFI shield to the Arduino and the WIFI shield to a WIFI network, the ESP8266 module, acquired a local IP address. Using this IP, the author was able to set up a local webpage, allowing the sensor to send the values read to it. The objective was to replicate the mobile Serial Monitor, but now using the Internet. This was concluded with success and the system performed as expected but limitations were found. The conclusions reached were as follows:

- A new form of wireless communication was developed.
- Although the local webpage allows us the visualization of the data read by the sensor, the webpage is not able to refresh every time it receives a new value. The process of refreshing the webpage must be made manually.
- The connection from the WIFI shield to the Internet is not always stable and can lead to a system crash.

The second phase was the implementation of the MQTT protocol. The MQTT protocol allows communication between devices using a broker over the internet. The author used a second Arduino Uno device, to receive the data from the first one containing the sensor and by placing a LED in the second Arduino, the objective was to change the behavior of the LED, based on the values received from the Arduino containing the sensor.

The implementation of this communication was developed, and conclusions were made regarding it:

- The MQTT protocol worked successfully in our system.

- The Arduino Uno with the LED was able to change behavior based on the changes received
- The connection between the WIFI shield and the network was stable and allowed for the exchanging of messages between both devices.

Having concluded with success the implementation of these communications protocols in our IoT Lab Box, it was time to think in concrete case studies to apply our box and see if the box can be a viable learning solution. This box was built to be a versatile system, being able to adapt to different projects and purposes serving the need of the user.

The ACIC course has already laboratory guides using the Arduino Uno device as the development platform. Since our system uses the same devices, it is a great case study to see if the box could be adapted to solve these laboratory guides and if the technologies implemented in the system could also be used in that course. The implementation details of the IoT Lab box to the ACIC laboratory guides are explained in the previous chapter.

Having built a project with adaptive capacities to different projects and different types of communications, we can conclude:

- Having a mobile app with a mobile Serial Monitor, allows the user to see multiple sensor data without the need to connect the Arduino Uno to the PC.
- By having this Bluetooth communication, as long we are within range, using an Arduino Uno with multiple sensors mapping the environment sending the values via Bluetooth is viable.
- The Arduino Uno is not a viable solution to develop a webpage to visualize sensor data via the Internet since the webpage does not have auto-refresh capabilities and the Internet connection is not the most stable.
- When using the MQTT Protocol with the Arduino Uno with multiple sensors, the connection is not stable and the system can crash multiple times, making this solution not viable.
- When using the ESP32, the BLE connection is faster than the Bluetooth classic module HC-06, although the implementation is not as simple.
- The WIFI communications when using the ESP32 are faster and more stable.
- Using the ESP32 to implement a webpage allows us for a more stable connection and an auto-refresh webpage.
- The MQTT protocol while using the ESP32 works flawlessly. It allows for a publish and subscribe action without a system crash.
- The ESP32 has more computational power than the Arduino Uno and all the features implemented work better while using the ESP32.
- The ESP32 is the most viable solution to implement a system like the IoT Lab Box and is a viable future solution.

5. Evaluation

Considering all the implementation described in the previous chapter, we must now perform an analysis of the obtained results. The main features are Bluetooth communications, WIFI communications, and the Implementation of the IoT Lab Box to the ACIC laboratories guides.

5.1. Bluetooth Communications

In this project, two different devices were used to implement Bluetooth Communication. The project used an HC-06 Bluetooth classic module and the BLE module present in the ESP32. Both devices performed as expected and the implementation of both, while being different was made achieving the same objective. By using the Bluetooth devices to send data to our smartphone app to build a mobile Serial Monitor, we must considerer an important factor, time. The sensors are constantly reading new values and sending these new values and having a communication with a bigger delay will not allow us to have reliable data.

Table 3 - Bluetooth communication latency

Hardware device	Message latency (average)
Arduino Uno + HC-06 + Android App	0.978 seconds
ESP32 + BLE + Android App	<0.5 seconds

Looking at Table 3, we have a measurement performed by the user indicating the latency between a message being processed in the Arduino Uno/ESP32, arriving at the smartphone, and receiving an OK in the Arduino Uno/ESP32 acknowledging that the smartphone received the message. This procedure was then repeated several hundred times and the average time was calculated, being the one present in the Table above. To measure these latency times, the sensors and the smartphone used are the same, the only change is the module behind.

As the table above shows, the latency in the communication when using the Arduino Uno is greater. The Arduino Uno has a computational power smaller than the ESP32 and the BLE communications are faster than the ones from the Bluetooth classic module. The BLE module is an upgrade of the Bluetooth classic but that does not mean that the Bluetooth classic cannot be used. Nevertheless, both times are under 1 second and when considering view sensor data in a Serial Monitor, these types are more than acceptable and prove that using Bluetooth communications whether via Arduino Uno whether via ESP32 are viable.

5.2. WIFI Communications

While developing the IoT Lab Box and while exploring the WIFI communication, different features were created. Using the Arduino Uno and the ESP32, a webpage and development of the MQTT Protocol were implemented. Now, we must evaluate these features and determine if they are viable for our project. As in the previous topic, time is extremely important. On one hand, when implementing the webpage, we must understand how long the messages take to be displayed on the webpage. On the other hand, when using the MQTT Protocol, it is important to see how long the publishing and the subscription procedure take to see if using this protocol is viable or not.

Table 4 - WIFI communications latency - Webpage

Hardware device	Message latency – Webpage (average)
Arduino Uno + WIFI Shield	Approximate 5 seconds
ESP32 + WIFI	2.323 seconds

Looking at Table 4, we can have an overview of the total time the system takes to send the values to the webpage. To achieve these values, the procedure was repeated several hundred times and an average was calculated.

First, we analyze the Arduino Uno working together with the WIFI shield. It takes approximately 5 seconds, on average, to establish a connection to the Internet and to send the values from the sensors to the webpage. This is the average time calculated. Sometimes we observe a faster-sending procedure, while in other cases it takes a bit more. Since the implementation of a webpage when using the Arduino Uno does not allow for an automatic refresh, we must refresh the webpage manually. This is not ideal and can cause sometimes problems. If the user refreshes the webpage when the values are being sent, it can cause a crash in the system and the connection to the Internet is lost. Building a webpage when using an Arduino Uno is not a viable solution since the operation does not always perform flawlessly.

When considering the implementation of a webpage using the ESP32, the results are very positive. First, the connection between the ESP32 and the Internet is done very rapidly. The average time that the system takes to establish a connection to the Internet and initialize the webpage and send the sensor values to it, takes about 2.323 seconds. Since the ESP32 is cable of having a script allowing the webpage to refresh itself, the sending procedure of the values is done without a problem. The refresh function updates the webpage every two seconds. This refresh action can be reduced for 1 second for example, but while developing this project, a 2-second interval was the optimal one. Every two seconds, the webpage updates itself with the most recent values and they can be visualized. Using the ESP32 to create a webpage with Serial Monitor capabilities is extremely viable and recommended. No matter where the user is, by using the browser to access the IP address where the webpage is located, the values from the sensors can be visualized. For example, if the user wants to have an ESP32 with a specific sensor mapping the environment, the user can see the results in real-time from anywhere as long there is an Internet connection.

Since we can change the time, it takes to refresh the webpage, we can decrease it and develop other projects with it. In this project, we use the webpage to display sensor information, and the two-second interval is adequate to it but let’s imagine that we want to control a vehicle using as its brain an ESP32 and our webpage is also responsible for showing us the position of the vehicle. A 2-second update may be too much for it, since in 2 seconds the position changes and we may lose control of the vehicle. The ability to have a refresh action be configured and the power of the ESP32 makes this the most suitable and versatile hardware option.

Table 5 - WIFI communications latency - MQTT Protocol

Hardware device	Message latency – MQTT Protocol (average)
Arduino Uno + WIFI Shield	3 seconds
ESP32 + WIFI	0.884 seconds

Another implementation performed using the Internet was the MQTT protocol. The MQTT protocol was implemented with the Arduino Uno and the ESP32. Looking at Table 5 we have an overview of how long the system takes to Publish values. As explained in the previous section, when using the Arduino Uno with multiple sensors, the Subscribing procedure is not recommended since the Arduino Uno does not have the capacity for this task. Nevertheless, the measurement of the time was done using only one sensor in this case and the procedure was repeated several hundred times and the average was calculated.

The Arduino Uno takes a few seconds to establish a connection to the Internet and the MQTT protocol. Having these connections established, it begins to send values to the MQTT server. The 3 seconds measurement took into consideration the time the system took after being connected to the MQTT server to publish and to subscribe to the values from the topic.

The same measurement, considering the same factors, was applied to the ESP32. Here, we can see an improvement in the performance. Another improvement that plays a huge factor, when deciding which device to use, is the fact that the ESP32 allows for publishing and subscribing to multiple sensors without causing a system crash, and the same is not verified when using the Arduino Uno. These 0.884 seconds measured in the ESP32 refer to the time that the system took to publish the values after being connected to the Internet and the MQTT broker. The connection time to these two is very small and on average took about 0.2 seconds. When comparing the ESP32 to the Arduino Uno, we can see a big improvement, leading to the conclusion that the sESP32 is the viable solution to implement a system using the MQTT Protocol.

Even doe our system, when using the ESP32, is cable of performing the publish and subscribing action very quickly, for our project we do not need that type of speed. A timer was implemented in our code, to publish values periodically, allowing for a more stable publish and not overloading our devices. Nevertheless, knowing that our system can perform these tasks quickly, opens the door for future projects where speed is key, like controlling a vehicle, where sending commands at the highest speed possible is key.

5.3. ACIC laboratory results

After developing our IoT Lab Box system, the user performed the necessary adaptions to implement this project in the ACIC course, mainly the laboratory guides. This was done as an evaluation metric since it proves the scalability and versatility of the system. All the implementation details regarding this adaptation are detailed in section 4.3. of this document.

Having the adaptation of our system to the laboratory guides made, the ultimate test was testing it with people, mainly students and people who don't have a background in these subjects, giving us an unbiased opinion regarding the functionalities, the difficulty, and the performance of our system. However, this was not possible.

Due to the pandemic times, we live in, having different types of people come and test the system was not possible. The system was developed with the educational goal in mind and to give the possibility to everyone who wishes to learn more about the IoT world and to embrace new projects to have a system allowing them exactly that. Unfortunately, this did not happen. All the conclusions made regarding the performance and the advantages and disadvantages of our system were made by the author.

6. Conclusion

The IoT Lab Box was developed as an introduction to the IoT world. It was developed with several goals being the most important one, the educational. It is important to find ways to increase the interest of students in studying different subjects and when focusing on the IoT world, this project does that. With the development of different types of communication, mainly wireless ones, a new learning experience was created. For everyone, who does not have a background in these subjects, this project is a starting platform to start and explore this new world. The use of both Arduino Uno and ESP32 gives two platforms to develop the project and gives the possibility to conclusions and preferences to be created, regarding the module to use. By having Bluetooth and WIFI capabilities, alongside an android app, the base for development is all complete.

It was also proven, that this project has practical applications, using laboratory guides as a practical case study, demonstrating the adaptation capacity of this project. This project can be used in a laboratory context, as demonstrated, but also be used for students and other people to explore and try and develop their form of an IoT Lab box.

The evaluations results obtained were great and the limitations found were also positive, giving us an understanding of what is and isn't possible to do with this system.

6.1. Future work

Although the system is fully ready to be used, there is always room for the development of new features on top of what was done. While developing the project, ideas came up for the future of our project. Having a project that can be adapted for different purposes, in the future small laboratory case studies or big practical case studies can be applied, such as:

- Laboratory Case studies:
    - Using IoT Lab box to create a Face Recognition Door Lock System for the laboratory using either ESP32 or Arduino Uno.
    - Creating a Hand Sanitizer Dispenser using the IoT Lab Box components with a live update, via Webpage or the Bluetooth App, of the amount of sanitizer present.
    - Integrate the MQTT protocol with the IoT Lab Box smartphone App. Right now, the smartphone app is used as a mobile Serial Monitor and used to send some information to the box. Since this project has the MQTT protocol implemented, in the future the app could be also used to publish and to subscribe to topics. This would allow sending commands and messages from the app to the Arduino as the Bluetooth does.
    - This project has a strong link to courses in software and hardware engineering, but people and courses not related to these types of subjects can also benefit by using this project. For

example, by adding chemistry base sensors, like a pH Sensor to the IoT Lab Box, to view the pH of a solution using the mobile Serial Monitor providing an IoT Solution.

- Bigger Practical Cases:
    - By having a project with WIFI and Bluetooth capabilities, we can think of bigger projects to implement and escape the inside laboratory projects. For example, by connecting air quality sensors through the university campus, we could have multiple IoT Lab Boxes mapping a bigger environment, and using the mobile Serial Monitor, it would be possible to indicate the Air Quality level present in our University.
    - Considering the Tagus Park IST campus, the implementation of an Ultrasonic Range sensor into our project could be used to monitor available parking spots. The IoT Lab Box would then send the sensor information via the Internet to the local webpage, and students, professors and more could always know the occupation level of the campus garage.

7. References

[1] Alberto Manuel Ramos da Cunha, "Applications and Computation for the Internet of Things," [Online]. Available: https://fenix.tecnico.ulisboa.pt/disciplinas/ACPIC7/2020-2021/1-semestre. [Accessed 10 2020].

[2] Renato Jorge Caleira Nunes and Alberto Manuel Ramos da Cunha, "Ambient Intelligence," [Online]. Available: https://fenix.tecnico.ulisboa.pt/disciplinas/AI5146/2020-2021/2-semestre. [Accessed 10 2020].

[3] Hugo Nicolau and Valentina Nisi, "Internet of Things Interaction Design," [Online]. Available: https://fenix.tecnico.ulisboa.pt/disciplinas/DIIC13/2020-2021/1-semestre. [Accessed 10 2020].

[4] João Coelho Garcia and Luís Pedrosa, "Mobile and Ubiquitous Computing," [Online]. Available: https://fenix.tecnico.ulisboa.pt/disciplinas/CMov46/2020-2021/2-semestre. [Accessed 10 2020].

[5] O. H. Graven and J. Bjørk, "The use of an Arduino pocket lab to increase motivation in Electrical engineering students for programming," 2016 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), 2016, pp. 239-243, doi: 10.1109/TALE.2016.7851800," 10 2016. [Online]. Available: https://ieeexplore.ieee.org/document/7851800.

[6] UTAD, "Temperature Control Laboratory," [Online]. Available: https://utadtv.utad.pt/info/academia/estudantes-engenharia-electrotecnica-da-utad-aulas-praticas-ajuda-laboratorio-portatil/. [Accessed 10 2020].

[7] D. Călinoiu, R. Ionel, M. Lascu and A. Cioablă, " "Arduino and LabVIEW in educational remote monitoring applications," 2014 IEEE Frontiers in Education Conference (FIE) Proceedings, 2014, pp. 1-5, doi: 10.1109/FIE.2014.7044027," 22-25 October 2014. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7044027. [Accessed 11 2020].

[8] P. Jamieson, "Arduino for Teaching Embedded Systems. Are Computer Scientists and Engineering Educators Missing the Boat?," April 2021. [Online]. Available: https://www.researchgate.net/publication/264847224_Arduino_for_Teaching_Embedded_Systems_Are_Computer_Scientists_and_Engineering_Educators_Missing_the_Boat.

[9] Perenc, Izabela & Jaworski, Tomasz & Duch, Piotr, "Teaching programming using dedicated Arduino Educational Board. Computer Applications in Engineering Education. 27. 10.1002/cae.22134," June 2019. [Online]. Available: https://www.researchgate.net/publication/334110064_Teaching_programming_using_dedicated_Arduino_Educational_Board. [Accessed 10 2020].

[10] Uyanık, İsmail & Catalbas, Bahadir, "A low-cost feedback control systems laboratory setup via Arduino-Simulink interface. Computer Applications in Engineering Education. 26. 10.1002/cae.21917," February 2018. [Online]. Available: https://www.researchgate.net/publication/323325416_A_low-cost_feedback_control_systems_laboratory_setup_via_Arduino-Simulink_interface. [Accessed 10 2020].

[11] J. Sarik and I. Kymissis, ""Lab kits using the Arduino prototyping platform," 2010 IEEE Frontiers in Education Conference (FIE), 2010, pp. T3C-1-T3C-5, doi: 10.1109/FIE.2010.5673417," 27-30 October 2010. [Online]. Available: https://ieeexplore.ieee.org/document/5673417. [Accessed 12 2020].

[12] W. J. Esposito, F. A. Mujica, D. G. Garcia and G. T. A. Kovacs, ""The Lab-In-A-Box project: An Arduino compatible signals and electronics teaching system," 2015 IEEE Signal Processing and Signal Processing Education Workshop (SP/SPE), 2015, pp. 301-306, doi: 10.1109/DSP-SPE.2015.7369570," 9-12 August 2015. [Online]. Available: https://ieeexplore.ieee.org/document/7851800. [Accessed 12 2020].

[13] M. Caţă, ""Smart university, a new concept in the Internet of Things," 2015 14th RoEduNet International Conference - Networking in Education and Research (RoEduNet NER), 2015, pp. 195-197, doi: 10.1109/RoEduNet.2015.7311993," 10 2020. [Online]. Available: https://ieeexplore.ieee.org/document/7311993.

[14] D. Sullivan, W. Chen and A. Pandya, "Design of remote control of home appliances via Bluetooth and Android smart phones," 2017 IEEE International Conference on Consumer Electronics - Taiwan (ICCE-TW), 2017, pp. 371-372, doi: 10.1109/ICCE-China.2017.7991150," 12-14 June 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7991150. [Accessed 11 2020].

[15] S. Banerjee, S. Laskar, A. Chowdhury, S. Sarkar, A. Roy and A. Das, ""Real-Time Monitoring and Control of Consumed Power for Household Appliances using Arduino Uno through Bluetooth and Android Application," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), 2019, pp. 529-533, doi: 10.1109/," 23-25 April 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8862528. [Accessed 12 2020].

[16] S. Majumder, M. A. Rahman, M. S. Islam and D. Ghosh, " "Design and Implementation of a Wireless Health Monitoring System for Remotely Located Patients," 2018 4th International Conference on Electrical Engineering and Information & Communication Technology (iCEEiCT), 2018, pp. 86-91, doi: 10.1109/CEEICT.2018.," 13-15 September 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8628077. [Accessed 11 2020].

[17] Kubínová, Štěpánka & Slegr, Jan, "ChemDuino: Adapting Arduino for Low-Cost Chemical Measurements in Lecture and Laboratory. Journal of Chemical Education. 92. 150807151714005. 10.1021/ed5008102," August 2015. [Online]. Available: https://www.researchgate.net/publication/282485234_ChemDuino_Adapting_Arduino_for_Low-Cost_Chemical_Measurements_in_Lecture_and_Laboratory. [Accessed 12 2020].

[18] A. Garrigós, D. Marroquí, J. M. Blanes, R. Gutiérrez, I. Blanquer and M. Cantó, ""Designing Arduino electronic shields: Experiences from secondary and university courses," 2017 IEEE Global Engineering Education Conference (EDUCON), 2017, pp. 934-937, doi: 10.1109/EDUCON.2017.7942960," 10 2020. [Online]. Available: https://ieeexplore.ieee.org/document/7942960.

[19] "Author's GITHUB," [Online]. Available: https://github.com/santosandre123/IoTLabBox.

[20] T. M. Siham Sayeed, M. T. Rayhan and S. Chowdhury, "Bluetooth Low Energy (BLE) based portable medical sensor kit platform with cloud connectivity," 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2), 2018, pp. 1-4, doi: 10.1109/IC4ME2.2018.84656," 8/9

February 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8465645. [Accessed 02 2021].