Instituto Universitário de Lisboa

Department of Information Science and Technology

Smart System for Control and Monitoring of Swimming Pools

Gonçalo Alexandre Rodrigues Simões

A Dissertation presented in partial fulfillment of the Requirements

for the Degree of

Master in Telecommunications and Computer Engineering

Supervisor

Prof. Dr. Pedro Joaquim Amaro Sebastião, Assistant Professor

ISCTE-IUL

Co-Supervisor

Prof. Dr. Nuno Manuel Branco Souto, Assistant Professor

ISCTE-IUL

October 2019

Resumo

A Internet das Coisas surgiu como uma das maiores promessas tecnológicas, sendo cada vez mais identificada como um fator de dependência no quotidiano. Esta baseia-se na conexão de dispositivos capazes de proporcionar atividades de monitorização, controlo remoto ou desenvolvimento de ambientes inteligentes que visam reduzir a necessidade da intervenção humana.

Tendo em conta o uso incorreto de recursos como a água em atividades do dia-a-dia, cada vez mais existe uma grande preocupação e necessidade de acompanhar a qualidade da água para manter os seus utilizadores informados sobre possíveis situações de risco. Esta dissertação propõe a aplicação do conceito da Internet das Coisas com um novo esquema para monitorização e controlo da qualidade das piscinas, através de um sistema de baixo custo baseado numa rede sem fios de sensores e atuadores, reduzindo a requisição de recursos humanos na manutenção de piscinas. O principal objetivo deste sistema é proporcionar a poupança de recursos económicos e naturais aos seus utilizadores, contribuindo para um ambiente mais sustentável.

Foi desenvolvida uma aplicação Android, que possibilita a monitorização e controlo remoto de dados recolhidos das piscinas em tempo real, fornecendo uma análise de dados e a definição de limites para cada um dos parâmetros, de modo a notificar o utilizador quando os limites definidos forem excedidos. O controlo de alguns dispositivos é possível através de dois modos: manualmente e automaticamente.

A solução desenvolvida apresenta um sistema desenhado e implementado com uma arquitetura simples um alto nível de eficiência, com demonstrações práticas e resultados obtidos.

Palavras Chave: Internet das Coisas, Redes de Sensores e Atuadores, Piscina, ESP32, RFM95W, LoRa, Sustentabilidade, Eficiência.

Abstract

Internet of Things emerged as one of the biggest promises of evolutional technologies, being increasingly identified as a dependency of the quotidian life. It is based on the connection of many devices that aims to simplify daily activities providing monitorization activities, remote control or the development of smart environments that aims to reduce the need of human intervention.

Taking into consideration that the incorrect use of fresh water in daily activities is nowadays a big concern as well as the need to keep track of water quality in order to inform the user about potential risk situations, this dissertation proposes the appliance of IoT concept with a new scheme for monitoring and control of swimming pools quality through a low-cost system based on wireless sensor and actuator networks, which can reduce the requirements of human in the swimming pool maintenance. The main purpose of this system is to provide resources savings for the final user in financial and natural resources, contributing to a more sustainable environment.

An Android mobile application was developed, providing users to monitor and remotely control swimming pool's parameters in real time, providing an easier data analysis and the definition of thresholds to each parameter in order to notify the user when the imposed limits are exceeded. The remote control of some devices is possible to do in two modes: manually or automatically.

The developed solution presents a system designed and implemented with a simple architecture and high efficiency level, with practical demonstrations of the obtained results.

Keywords: Internet of Things, Wireless Sensor and Actuator Network, Swimming Pool, ESP32, RFM95W, LoRa, Sustainability, Efficiency.

Acknowledgements

I would like to thanks to my advisers, Professor Pedro Sebastião and Professor Nuno Souto, that were always available to supporting me in the development process of this project.

To Institute of Telecommunications of ISCTE-IUL for providing me a workspace and the necessary material for the development of the presented solution.

A special thanks to my family, my parents Ádela and Américo and to my brother Rodrigo that gave me motivation and unconditional support through my studies and always encouraged me for following my goals.

A huge thanks to my colleagues that were working hardly with me in the development of their dissertations in the last months and specially to André Glória for all the help provided and for sharing his knowledge in this area with me.

Finally, I would like to thank to my friend João Cardoso and satff of GesLoures swimming pools in Santo António dos Cavaleiros that provided us the possibility to implement this project in a real environment.

Contents

List of Figures

List of Tables

List of Acronyms

ACK	Acknowledgement
ADC	Analog-to-Digital Converter
AP	Access Point
BLE	Bluetooth Lower Energy
BSS	Basic Service Set
CSS	Chirp Spread Spectrum
DS	Distribution System
ESS	Extend Service Set
FFD	Full-Function Device
FSK	Frequency Shift-Keying
GFSK	Gaussian Frequency Shift-Keying
GMSK	Gaussian Minimum Shif-Keying
GPIO	General Purpose Input/Output
HTTP	Hypertext Transfer Protocol
IBSS	Independent Basic Service Set
IDE	Integrated Development Environment
IEEE	Institute of Electrical and Electronics Engineers
IoT	Internet of Things
LDR	Light Dependent Resistor
LoRa	Long Range
LoRaWAN	Long Range Wide Area Network
LR-WPAN	Low-Rate Wireless Personal Area Network
MQTT	Message Queue Telemetry Transport
MSK	Minimum Shift-Keying
OOK	On-Off Keying
OS	Operating System
PAN	Personal Area Network
PWM	Pulse-Width Modulation
QoS	Quality of Service
RF	Radio Frequency
RFD	Reduced-Function Device
RTC	Real Time Clock

RX	Receiver
SRAM	Static Random-Access Memory
TCP	Transmission Control Protocol
WLAN	Wireless Local Area Networks
WSN	Wireless Sensor Network
WSAN	Wireless Sensor and Actuator Network

Chapter 1 Introduction

1.1 Motivation and Framework

Information and communications technology, over the years, has been progressively present in human life, being a branch that is in constant evolution and an essential part in our everyday life.

With this evolution there is a growing number of physical objects and devices connected to the Internet, giving rise to the concept of the Internet of Things (IoT) [1]. IoT consists of a network of physical objects that are able of being connected to the Internet, identifying themselves, and communicating with each other to achieve a common goal. The main purpose of this technology is to exchange and update data between physical objects and hence to achieve an optimum performance [2].

When talking about IoT it is also useful to approach another concept called Wireless Sensor Networks (WSN), a technology that is inherent in the development of intelligent systems. Recently, IoT and WSN have been great bets in the development of monitoring and control systems by researchers [3]. This capability of monitoring and control covers a growing range of applications, among which some related to monitoring of swimming pools.

Swimming pool conditions, directly depend on how well their chemical properties are monitored [4]. Its maintenance requires the performance of some tests that may be more complex when performed by a human. Therefore, it is important to implement a sensor network that is able to perform those tests correctly and more precisely. To optimize its daily maintenance, instead of a person evaluate the water properties (pH, water level, temperature, water flow), machine room condition and outer space quality, comes up the proposed system, which main purpose is to control and monitoring swimming pools automatically.

Sustainability assumes an important role in preserving the planet Earth sources. Thus, it is increasingly important for the society to have sustainable attitudes in simple daily activities and particularly in the development of new technologies. In the specific case of swimming pools, it becomes relevant to monitor the water and energy consumption levels through a sensor network and, consequently, to adjust their

consumption, so that there is no waste of those resources for purposes deemed not essential for the correct operation and maintenance of swimming pools.

1.2 Objectives

The purpose of this thesis is to develop a control and monitoring system applied to swimming pools (public and private domain). This system is responsible for collecting some significant data from the point of view of maintenance, through a low-cost wireless sensor network, in order to evaluate if the swimming pool quality is within the desired parameters, whether at water level, piping leakage, environment quality or engine room efficiency.

The data obtained from the sensors will be stored and treated in such a way as to keep the user informed about the parameters considered relevant to the swimming pool quality. These will be presented to all the pool users and can only be controlled by a user with administrator permissions through an app (in case of private pools, by their owner and in case of public pools by their maintainer). When an anomaly is detected, the user will be notified and able to remotely control some actions on the swimming pool in order to stabilize the anomalous values that triggered the alert.

This work proposes a new system that can monitor and control the water properties (pH, chlorine, water level, temperature, water pressure), machine room and surrounding space quality bearing in mind the reduction of the consumption of natural and material resources, and the monetary saving by the final user.

1.3 Main Contributions

The main contributions of this dissertation are:

- Appliance of modern wireless communication protocols in WSAN;
- Design and implementation of a sensor and actuator network;
- Design and implementation of a mobile application for data analysis and device's control.

The results obtain through the development of this dissertation had contribute to the following scientific papers:

- G. Simões, C. Dionísio, A. Glória, P. Sebastião, and N. Souto, "Smart System for Monitoring and Control of Swimming Pools," in 5th IEEE World Forum on Internet of Things WF-IoT 2019;

- C. Dionisio, G. Simões, A. Glória, P. Sebastiao and N. Souto, “Distributed Sensing Solution for Home Efficiency Tracking”, 2019 IEEE 5th World Forum on Internet of Things (WF-IoT);
- A. Glória, C. Dionísio, G. Simões, P. Sebastião, and N. Souto, “WSN Application for Sustainable Water Management in Irrigation Systems”, in 5th IEEE World Forum on Internet of Things WF-IoT 2019;
- A. Glória, C. Dionísio, G. Simões and P. Sebastião, “LoRa Parameters Self Configuration for Low Power End Devices” in 22nd International Symposium on Wire-less Personal Multimedia Communications (WPMC 2019);
- A. Glória, C. Dionísio, G. Simões, J. Cardoso, P. Sebastião and N. Souto, “Water Management for Sustainable Irrigation System using Internet of Things,” in Transactions of the Institute of Measurement and Control.

1.4 Structure of the Dissertation

After this introduction chapter, Chapter 2 presents the State of the Art that resumes all the related topics with the development of this work describing some important definitions and comparations that had to be performed to provide the best choices for the system such as hardware and software levels. It also reviews some related works and the improvements of this approach. Chapter 3 aims to describe the system architecture, having a more detailed description of the software, hardware (nodes constitution) and communication protocols adopted for the system and how they fit in this solution. Chapter 4 presents the implementation in experimental and real environments such as the corresponding discussion of the obtained results. Finally, Chapter 5 presents the main conclusions obtained after the development of the system and future work that could be performed through what is also presented in this dissertation.

Chapter 2 State of the art

This chapter is based on a review of the possible approaches for the development of a smart system applied to swimming pools. In section 2.1 are described the relevance that IoT has nowadays and its main components. Section 2.2 describes the most common communication protocols used in IoT projects, finishing with remarks in conclusions are taken about the best protocol for this system. In 2.4 are presented all the hardware components that are necessary to build a complete smart system. This section compares all the hardware components with similar features in order to choose the best suited for the intended characteristics. Described in section 2.5 is the definition of a mobile application and the operating system that will be considered for the development of the mobile app. Section 2.6 presents some similar IoT projects applied to swimming pools and a distinction between the existing projects and the proposed system.

2.1 Internet of Things

At a time when the Internet was one of the major trends, technology gained a new base, called IoT [1]. This is a new paradigm that over the years has quickly gained ground in the modern branch of wireless communications [2]. The main strength of IoT is the high impact that this technology has on an increasingly wide range of applications in everyday life and on its potential users.

IoT enables physical objects to sense, think and perform some actions by communicating with each other, to share information and to coordinate decisions [3]. The main goal of this new concept is to reduce the need of human intervention in simple daily activities.

To gain a better insight into the real meaning of this concept, it is possible to decompose it in six main components, as can be seen in Figure 2.1 [1].

Figure 2.1 - IoT Main Components

Identification is essential for IoT to name and match services. Furthermore, addressing IoT objects is critical to differentiate between object ID (refers to its name) and its address. The addressing methods of IoT objects include IPv6 and IPv4.

Sensing is defined as the ability to collect data through physical objects/devices included in a network and send them to a database, ware house or cloud. The data collected is analyzed so that specific actions are taken to achieve certain objectives.

Communication technologies allow the connection of differentiated objects in order to provide certain services. The network nodes must operate, consuming little energy in the presence of communication links with losses and noise. Some of the most commonly used protocols in IoT are Wi-Fi, Bluetooth, ZigBee, LoRa, Z-ware, and LTE-Advanced.

Computation is described as the brain of IoT and it could be composed of processing units and software applications. There are some hardware platforms to run IoT such as Arduino, UDOO, FriendlyARM, Intel Galileo, Raspberry Pi, etc.

IoT services can be categorized under: identity-related services, information aggregation services, collaborative-aware services and ubiquitous services.

Semantic refers to the ability to extract intelligent knowledge through machines to provide certain services. This extraction includes the use and discovery of information and the recognition and analysis of data in order to make the right decision for the intended service.

2.2 Wireless Sensor and Actuator Network

Sensor networks have an important role in the development of an IoT system by gathering data from sensors, and then make some decisions to optimize the user experience on a specific branch.

Sensor networks are composed by three types of nodes [5]:

- Sensor Nodes compose the lowest level of the sensor network and are responsible for transmit the data to another node in the network. This type of nodes does not store or manipulate the gathered data.
- Data Nodes typically send data to a storage mechanism such as a data card, a database via computer, or directly to a visual output device. These nodes are an

excellent use for microcontroller and can be used to form autonomous or unattended sensor networks that record data for later archiving.

- Aggregation Nodes employ a communication device and a recording service or a gateway and there are no sensors in its constitution. They are used to collect data from sensor and data nodes.

With the uprising of more complex IoT project, a new type of networks appears, making use of a new type of nodes (Actuator nodes). The Wireless Sensor and Actuator Network (WSAN) follows a similar structure of WSN with an addition of Actuator Node (Figure 2.2), equipped with better processing capabilities and higher transmission power [6]. This type of nodes is composed by actuators that are responsible to perform, based on the gathered data from sensor nodes, to stabilized specific values when they are not in the required values [6].

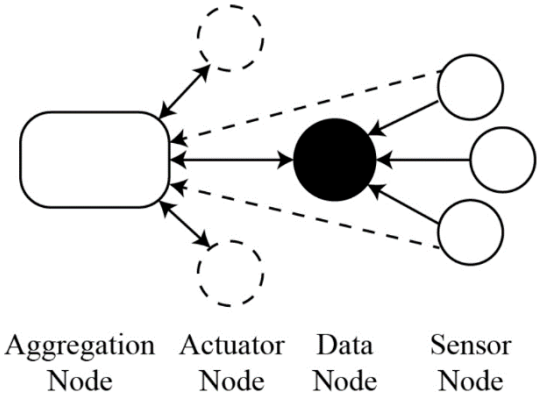

Figure 2.2 - Sensor and Actuator Network Nodes

2.3 Communication Protocols

As mentioned in Section 2.1, IoT greatly depends on the communication branch, being necessary to study which communication protocols best suits in the intended characteristics for the system that will be developed.

2.3.1 Wireless Communications

Wireless communications provide everything that is necessary to transfer data between devices, being a better alternative for wired communications. In this section some wireless communication protocols will be described such as Wi-Fi, Bluetooth, ZigBee and LoRaWAN.

To better compare the described protocols and decide which one fits in the proposed system, Table 2.1 [7] has some relevant parameters of the wireless protocols that will be analyzed.

Table 2.1 - Major Wireless Communication Protocols Characteristics

Feature	Wi-Fi	Bluetooth	ZigBee	LoRaWAN
Based Data Rate [Mbps]	11	1	0.25	0.11
Frequency [GHz]	2.4	2.4	2.4	0.433
Range [m]	1-100	10-100	10-100	2000
Nodes	32	7	65540	15000
Power Consumption [mA]	100-350	1-35	1-10	1-10
Complexity	High	Medium	Medium	Low
Security	WPA/WPA2	128 bit	128 bit	128 bit

2.3.1.1 IEEE 802.11 - Wi-Fi

Wireless Fidelity is a bidirectional communication protocol that allows devices to connect to a wireless local area network (WLAN) and the exchange of data. Wi-Fi includes IEEE 802.11 standards, and its architecture is composed of a set of components that interact with each other to provide a WLAN that supports station mobility, using basic service set (BSS), a set of mobile or fixed stations [8].

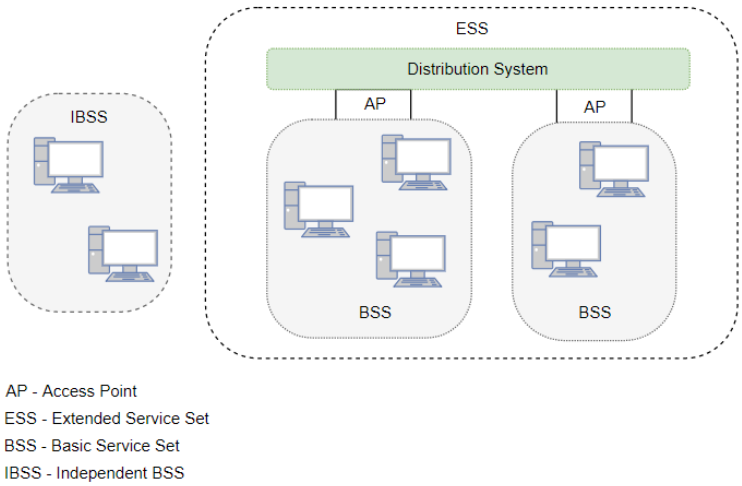

Figure 2.3 - IBSS and ESS Configurations of Wi-Fi Networks

Based on BSS, IEEE 802.11 also employs the independent basic service set (IBSS) and extend service set (ESS). The IBSS operation is possible when the devices

can communicate directly without any access point (AP). This type of operation is called ad hoc network, seeing that this type of LAN is formed without pre-planning for only the period that the LAN is needed. Instead of existing independently, is also possible to form an extended form of network composed of a set of BSSs. To interconnect the multiple BSSs is used the distribution system (DS), that with APs allows the creation of ESS network [8].

Table 2.2 shows some advantages and disadvantages of the IEEE 802.11-Wi-Fi protocol.

Table 2.2 - Advantages and Disadvantages of Wi-Fi

Advantages	Disadvantages
• Can penetrate obstacles on the way; • Simple process of add/remove nodes; • Secure connection provided by 128-bit AES encryption.	• High complexity; • Significant power consumption.

2.3.1.2 IEEE 802.15.1 – Bluetooth

Also known as the IEEE 802.15.1 standard, Bluetooth is based on a wireless radio system design for transmission over short and medium distances. Bluetooth devices operate in the 2.4 GHz frequency band, and it divides this band into 79 channels and employs channel hopping techniques [9].

Bluetooth devices can only operate in two different modes: master or slave. The master is the first device that provides the synchronization and can only control up to seven slaves. When a slave connects with the master it receives an address and clock, using this information to calculate the frequency hopping sequence [8].

The most recent version of Bluetooth is Bluetooth v4.0, commonly known as Bluetooth Lower Energy (BLE), which includes the low energy feature to reduce the power consumption problem [10].

Table 2.3 presents the advantages and disadvantages of Bluetooth.

Table 2.3 - Advantages and Disadvantages of Bluetooth

Advantages	Disadvantages
• Master can control up to seven slaves; • Doesn't need any configuration to start the data exchange.	• Limited to eight nodes (one master and seven slaves); • Only can be used with star topology; • Low communication range.

2.3.1.3 IEEE 802.15.4 – ZigBee

ZigBee was introduced in 2002 and it is based on IEEE 802.15.4 protocol. This communication protocol defines some specifications for low-rate wireless personal area network (LR-WPAN) for supporting simple devices that consumes the minimal energy as possible [11]. ZigBee also provides high reliability, security with both encryption and authentication services and it can handle up to 65540 nodes [7].

There are two device types that can operate on a ZigBee network: full-function device (FFD) and a reduced-function device (RFD). An FFD can operate as a PAN coordinator, a coordinator or a device, and it can communicate with RFDs or other FFDs. The RFD is used in applications that are extremely simple, such as a light switch. This type of devices may only associate with a single FFD at a time and typically do not need to send large amounts of data [8].

As can be seen in Table 2.4, ZigBee has some advantages and disadvantages associated.

Table 2.4 - Advantages and Disadvantages of ZigBee

Advantages	Disadvantages
• Supports a large number of nodes; • One coordinator is able to control many slave nodes; • Low-power consumption; • Secure connection provided by 128-bit AES encryption.	• Additional hardware is needed; • Incompatibility with other communication protocols.

2.3.1.4 LoRaWAN

LoRaWAN is a bidirectional communication protocol that uses LoRa physical layer, that enables the long-range communication link [7]. To provide a long-range and low power communication, LoRa uses the Chirp Spread Spectrum (CSS) modulation, which uses frequency chirps with linear frequency variation over time to encode information [12]. This modulation has the same low power characteristics as the FSK modulation, which is a type of modulation used in the major part of other wireless communication protocols. With LoRa it is possible to cover a full area with hundreds of square kilometres with a single gateway or base station [13].

This protocol is designed for sensor networks, where the sensors exchange packets with the server with a low data rate. A set of components are defined in the LoRaWAN: end-devices, gateways and the network servers. The end-devices are defined by the sensors that communicate with the gateways. The gateway is the intermediate that forward packets received from the end-devices to a network server. The network server de-duplicates and decodes the packets sent by the devices and then generates the packets that should be sent back to the devices [13].

Table 2.5 shows some advantages and disadvantages associated with LoRaWAN protocol.

Table 2.5 - Advantages and Disadvantages of LoRaWAN

Advantages	Disadvantages
• Low-power consumption; • Long communication range; • Low Complexity; • Secure connection provided by 128-bit AES encryption.	• A packet can only have a maximum size of 55 bytes.

2.3.1.5 Remarks

Based on Table 2.1, in section 2.3.1, and the description of the presented protocols, the protocol that best suits this system is LoRaWAN. One of the main goals of the system is to be power efficient and LoRaWAN is the one who has this characteristic combined with a low complexity level. Another parameter that is relevant

in LoRaWAN protocol is its large range, an essential feature for the swimming pool system, given that swimming pools are normally a confined environment where some nodes could be hundreds of meters apart (e.g. distance between an outdoor pool and its compensation tank).

2.3.2 Server-Side Communications

Since the wireless protocols displayed do not allow communication with the server, a targeted analysis is required. In this topic will be compared two protocols to establish the server-side communication: Hypertext Transfer Protocol (HTTP) and Message Queue Telemetry Transport (MQTT) protocol.

2.3.2.1 Hypertext Transfer Protocol

HTTP is an application level protocol for distributed, collaborative hypermedia systems [14]. This protocol has 3 main versions: http/1.0, http/1.1 and http/2.0.

This protocol uses TCP service from transport layer and follows the client/server model. The client is the navigator that requests, receive and shows objects in Web. The server Web is responsible for sending objects, responding to client requests.

In Figure 2.4 is represented the client/server model. The client starts a TCP connection (creating a socket) with the HTTP server and the server accepts the TCP connection. After the server acceptance it can be followed the messages exchange between the client and the server. When the messages exchange is finished the TCP connection is closed [14].

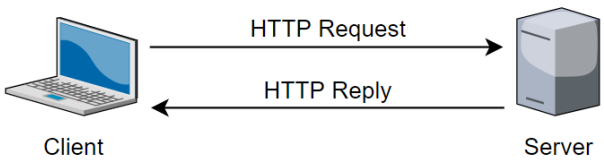

Figure 2.4 - HTTP Client/Server Model

2.3.2.2 Message Queue Telemetry Transport

MQTT is a communication protocol based on TCP which main goal is to connect devices to embedded networks with applications and middleware. The connection operation makes use of a routing mechanism and enables MQTT as a great

connection protocol for IoT communications, as it is able to provide routing for small, low power, low cost and low memory devices in low bandwidth networks [15].

This protocol uses the publish/subscribe standard, in order to guarantee transition flexibility and simplicity in its implementation. MQTT is based on 3 components [16]:

- The publisher, that to communicate via the MQTT protocol must define two main elements: message and topic. The message is the string data that the publisher pretends to share with subscribers via MQTT broker. The topic refers to a string that is used to filter and decide which subscribers should receive which message.
- The subscriber is the one who subscribes a specific topic in order to receive data from the publisher related with the subscribed topic.
- The MQTT broker is the central controller that forwards, filters and prioritizes published requests from the publishers to the subscribers.

In Figure 2.5 is represented the publish/subscribe process [17].

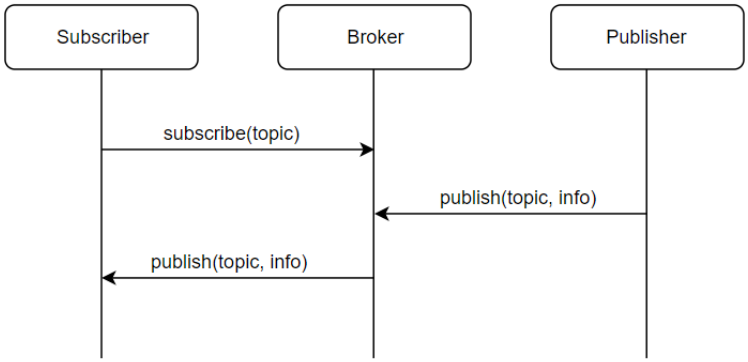

Figure 2.5 - MQTT Publish/Subscribe Process

MQTT supports three levels of Quality of Service (QoS) to ensures reliability of message transmission [17]. QoS level 0 only sends the message and does not check if it arrived at its destination. When the sent message has a considerable size, it is possible that it will be lost when any loss comes in the way. The QoS level 1 sends the message at least once and checks if the message arrived at its destiny by using a status check message (PUBACK). When the status check message is lost, the server will try to send the same message. In the QoS level 2 it is not possible to have a message loss, since it uses the 4-way handshake to send the message. However, due to the complexity of the 4-way handshake process it is possible to have longer end-to-end delays.

2.3.2.3 Remarks

In [18] the author performed some tests to compare the HTTP and MQTT protocols. According to the performed tests, due to MQTT increase in information density for each payload, a greatest efficiency gain can be reached. The advantage of using MQTT over HTTP occurs when a single connection is reused for sending multiple messages in which the average response time of MQTT is considerably lower when compared with HTTP. Considering that, the MQTT protocol is the one, which best suits the intended system.

2.4 Hardware

A smart system has in its constitution many hardware components, such as controlling platforms, radio modules to provide communication between the nodes that compose the system, and a set of sensors.

2.4.1 Controlling Platforms

The controlling platforms that will be analyzed in this topic are Arduino, ESP8266 and ESP32. These provide the reduction of the overall cost of the system.

2.4.1.1 Arduino

Arduino is an open source physical computing platform based on ATMega328 microcontroller board and a development environment that implements the Processing language [19]. The reason behind the Arduino development was to provide a platform to create devices and projects by integrating various types of sensors and actuators. Arduino has libraries and based on C++ and Integrated Development Environment (IDE) environment for proper programming interface [20].

Arduino IDE is a platform independent base and can run on multiple operating system platforms. This is a strong platform to develop projects on Arduino controllers, with multiple libraries for sensor already developed.

The most common Arduino boards type is Arduino Uno (Figure 2.6), that responsible for interfacing of different peripherals, sensors and wireless communication devices. These boards are composed of a 20 MHz clock oscillator, 32kB flash, 1kB

SRAM, 23 input/output programmable pins, six channels 10-Bit ADC and six PWM outputs [20].

Figure 2.6 - Arduino UNO board

2.4.1.2 ESP8266

ESP8266 (Figure 2.7) is a low-cost and energy efficient Wi-Fi module that can carry software applications. This is 32-bit single-core microcontroller that can provide a high performance and high integration performance. It also has an advanced power management which provides an energy efficient Wi-Fi based wireless sensor network. ESP8266 consumes only 60uA in deep sleep mode with RTC (real-time clock) still running and less than 1mA to stay connected to the access point [21].

The module has a good processing and storage capability that allows it to integrate GPIO ports sensors. Due to its highly integrated chip, this module allows very less external circuity. The module works on 3.3 V and has 17 GPIO pins with only one of them being analog [21].

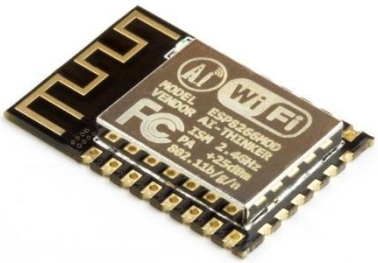

Figure 2.7 - ESP8266 Microcontroller

2.4.1.3 ESP32

ESP32 (Figure 2.8) is a single 2.4 GHz Wi-Fi and Bluetooth combo chip dual-core designed for mobile, wearable electronics and Internet of Things applications [22]. This microcontroller was introduced by Espressif System which is a successor to the ESP8266 microcontroller [23].

The reason that turns this microcontroller compatible with IoT applications is due to its low-cost and low-power system capabilities. In a low-power IoT sensor hub application, ESP32 has a power saving feature named Deep Sleep. This feature allows that ESP32 is only woken up periodically and when a specified condition is detected. To minimize the expended amount of energy it is also used a low duty-cycle. The output of the power amplifier is adjustable, contributing to an optimal trade-off between data rate, communication range and power consumption [23].

The ESP32 has 34 GPIO pins that can be assigned various functions by programming the appropriate registers. With this microcontroller it is possible to use 12 of the GPIO pins as analog pins [22].

Figure 2.8 - ESP32 Microcontroller

2.4.1.4 Remarks

Due to the low-power features provided by ESP8266 and ESP32, the Arduino platform isn't the best choice for the development of a low-cost system that is designed to be highly energy efficient. The main difference between ESP32 and ESP8266 microcontrollers is the number of cores. ESP32, being a dual-core microcontroller, it provides a division of tasks between both cores which results in a higher efficiency performance.

2.4.2 Radio Modules

Radio modules are extremely important in the system's communication branch. In this topic will be compared the RFM69HCW and RFM95W in order to choose the best which suits in the system.

2.4.2.1 RFM69HCW

The radio module illustrated in Figure 2.9 is able to operate over a wide frequency range. A big part of RF communication parameters is programmable and most of them can be dynamically set. RFM69HCW offers the advantage of programmable narrow-band and wide-band communications modes. It is optimized for low-power consumption while offering high RF output power and channelized operation [24].

Figure 2.9 - Radio Module RFM69HCW

The main features of the RFM69HCW module are[24]:

- +13dBm of power output capability;
- High sensivity, down to -120dBm at 1.2 kbps;
- Low RX current of 16 mA, with 100 nA register retention;
- Programmable Pout: -18 to +13 dBm in 1dB steps;
- FSK, GFSK, MSK, GMSK and OOK modulations.

2.4.2.2 RFM95W

Figure 2.10 - Radio Module RFM95W

RFM95W transceiver features the LoRa long range modem which provides long range spread spectrum communication and high interface immunity, whilst minimising power consumption. With RF's patented LoRa modulation technique it is possible to achieve a high sensitivity and provide significant advantages in both blocking and selectivity over conventional modulation techniques. This device (Figure 2.10[25]) can

support high performance FSK modes for systems and delivers exceptional noise, selectivity, receiver linearity and IIP3 for significantly lower current consumption than other devices[25].

The main features of this type of radio module are[25]:

- -20 dBm – 100 mW constant RF output vs. V supply;
- +14 dBm high efficiency power amplifier;
- Programmable bit rate up to 300 kbps;
- High sensivity, down to -148 dBm;
- Low RX current of 10.3 mA and 200nA of register retention;
- FSK, GFSK, MSK, GMSK, LoRa and OOK modulation;
- Built-in temperature sensor and low battery indicator.

2.4.2.3 Remarks

When compared both presented radio modules, RFM95W has a longer range than RFM69HCW. For this reason and for one of the key product features being a LoRa modem, the best choice to do is use RFM95W, due that the communication protocol that will be used in this system is LoRaWAN.

2.4.3 Sensors

A sensor is a device which main goal is to measure a specific parameter of some physical environments. Generally, sensors have a limited capacity and are low-cost components of wireless sensor networks.

To measure the pool's parameters are recommended the following sensors:

- DS18B20, a digital thermometer that allows temperature readings from -55ºC to +125ºC [26] and it is used to measure the water temperature;
- DHT22, a temperature and humidity sensor that allow temperature readings between -40ºC and +80ºC and humidity readings between 0% to 100% [27] (environment temperature and humidity);
- POW110D3B, a water flow sensor composed of a valve body, a water rotor and a hall-effect sensor that outputs the corresponding pulse signal [28];
- SEN01161, an analog pH meter kit, that allows pH readings from 0 to 14 [29];

- SEN0205, a liquid level sensor that operates using optical principles, has good sensitivity and no need for mechanical parts [30];
- LDR (Light Dependent Resistor).

2.5 Mobile Application

A mobile application is a computer program designed to run on mobile devices such as smartphones [31]. A mobile application could be developed for a specific operating system (OS) or for a set of operating systems. The operating systems that are mostly used are Android and iOS.

Android is a Linux based operating system that was launched by Google and it supports a large range of mobile applications. An android application can be full developed with Java programming language. To develop a mobile application for Android it could be used the Android studio.

iOS is an operating system developed by Apple and only present in its devices. Commonly known as iPhone operating system use a multi-touch interface in which simple actions are used to operate the device.

Due to the large number of users that have android devices when compared with iOS, being the aim to cover the largest amount of users, the mobile app will be developed for the android operating system.

The mobile app that will be developed will show the results gathered from the sensors and will provide the user to control the water pumps and lights of the swimming pool.

2.6 Related Work

Taking into consideration that the incorrect use of fresh water in recreational activities is nowadays a big concern as well as the need to keep track of water quality in order to inform the user about potential risk situations, swimming pool monitoring systems is something we can see in numerous automation projects already developed.

In [3], the authors developed an information and management system for swimming pools. This system consists of a sensor node interfacing with sensors (to measure pH, chlorine level, temperature, cleaner mobility and water level) and actuators, and a web-based user interface that can be accessed remotely for controlling the system. The project has accurate sensors, providing an optimum maintenance of the

swimming pools. However, the implemented sensors are extremely expensive, which creates a system that is not easily accessible to all users.

The authors in [4] developed a low-cost swimming pool automation system that, to keep the swimming pool in good conditions, performs tests on chlorine level and pH and automatically makes the necessary adjustments. The main goal of this project was to lower the energy consumption of the system by reducing automatically the functioning time of the water pump.

Some solutions already exist for the intended application, but none was based on the deployment of a WSN with multiples nodes spread across the pool to compare values in multiple locations. This system differs as it adopts a full WSN approach, in a modular architecture where new sensors can be added without compromising the network performance. The system is based on the use of low power consumption nodes and a communication protocol capable of connecting nodes spread along large areas with high reliability and low-power.

Chapter 3 System Architecture

The main goal of this work is to create a new a control and monitoring system applied to swimming pools, in both public and private domain. This system aims to use multiple sensors for collecting real time data, from maintenance relevant parameters, such as air temperature and humidity, water temperature and level, and pH level, and use these data to improve pool efficiency. Besides that, this new system has in count some other factors that can be significant to its users, such as natural and financial consumptions, in a way to promote a more sustainable environment with a circular economy fashion.

As mentioned in section 2.6, the market already has some solutions available, but none is capable of combining the maintenance requirements of swimming pools with a lower consumption of natural and material resources.

The system follows a common architecture of a Wireless Sensor and Actuator Network (WSAN). This type of networks is usually composed by a microcontroller, a communication system for inter and intra communication between the nodes and a set of sensors/actuators. A high-level view of the adopted system architecture is represented in Figure 3.1.

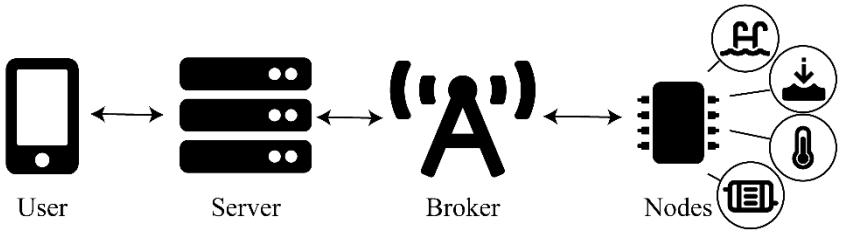

Figure 3.1 - System Architecture

The proposed solution, bearing in mind the WSAN fashion, is composed by a set of hardware nodes, responsible for collection of sensor data and/or turn on/off actuators, that communicate among each other using LoRa and with the server via MQTT over WiFi. Then, a set of software algorithms complements the network, not only to store and analyze the collected data, but also to provide the user a way to interact with the network, via an Android mobile application.

A more detailed description of the proposed architecture is divided into three main topics: communication protocols, hardware and software, each being detailed in the following sections.

3.1 Communication

As previously stated, communication is the most important part of a WSN, being responsible not only for transmitting the collected data, but also to keep the network connected, create reliability and guarantee that the system can perform the tasks that were specified. In a WSN, communication also helps improve the coverage of the implementation environment, mainly with new protocols, such as LoRa, allowing network to expand up to 5km.

The proposed system, composed by 3 different types of nodes, required that two protocols were used, since an intra-network communication, for communication between the network nodes, and an Internet communication, to send data to the servers, are needed.

As described in Section 2.3, where a study was conducted to decide the best communication protocol for each, LoRa, for node-to-node communication, and MQTT, for node-to-server communication, are the best choices to create the best solutions in terms of communication.

The details and implementation of each communication scheme is described below.

3.1.1 Node-to-Node Communication

Node-to-Node communication is how information is exchanged inside the WSN, from sensor information to actuator actions. Since nodes can be far from each other, and as some of the nodes are powered by batteries, a wireless communication capable of long-range transmissions with low power consumption is needed.

So, in order to connect all the network nodes, a LoRa peer-to-peer network was implemented, using the RFM95W radio transceiver. These modules, capable of using the LoRa protocol, allow the network to cover up to 2km with up to 256 nodes, with individual node addressing. Also, with only 70mA consumption while transmitting and a sleep mode, they check all the necessary specification in terms of node-to-node communication.

For configuration and usage, Arduino library RadioHead was used. This library enables the node addressing, with a unique address being chosen, guaranteeing that each node has its own address.

The nodes can communicate in two different ways, both encrypted by the library:

- By sending a message directly to a specific node, using its destination address;
- Sending a broadcast message to every node in the network, including the destination node ID inside the message.

In first scenario, the RadioHead library assures network fidelity by sending an automatic ACK response to the origin node that sent the message, using its ID. However, when a broadcast message is sent, the library cannot guarantee the network reliability. To cope with this limitation, each destination node will read the ID on the received message and if this parameter corresponds to its own ID, the node will accept the message, confirming that it was received by sending an ACK response to the origin node.

The presented system uses both communication methods: the first, when sensor nodes send the gathered data to the gateway and the second, when the gateway sends information to actuator nodes.

3.1.2 Node-to-Server Communication

Node-to-Server communication is how information containing sensor data leaves the network and actions are received. Since some sort of Internet connection is needed to connect to the server, and since ESP32 already as WiFi built-in, a way to transfer data in a reliable and low power way using WiFi was required.

So, in order to always have a connection between the server and the network, the MQTT protocol was adopted, due to its high communication reach and the low power consumption, when compared to other protocols, as described in Section 3.3.

As previously described, to implement the selected protocol, it was necessary to define specific topics for each way of communication, making it possible to distinguish between messages intended for the server and those intended for the broker:

- “tese/goncalo/in” – to send messages to the server and/or mobile application;
- “tese/goncalo/out” – messages whose destination is the network.

The network, in specific the broker node, subscribe to the “tese/goncalo/out” topic and its constantly listening to new messages, for example action to turn on/off an actuator. Also, when a message is received through the LoRa network from another node, for example new sensor information, the broker publishes that message to the topic “tese/goncalo/in”.

The server and mobile application do the exact opposite, subscribing to the “tese/goncalo/in” to wait for new sensor information, and publishing to “tese/goncalo/out” when a new action is created.

3.2 Hardware

The proposed system relies on a network of hardware nodes, that when put together create a Wireless Sensor and Actuator Network capable of retrieving sensor information, perform actions and connect with a cloud server.

As mentioned, this architecture follows a common constitution of a WSAN, which is composed of four types of nodes: aggregation, data, sensor and actuator nodes. In the presented system there were only used three node types: aggregation, sensor and actuator nodes, existing a directly communication between the sensor nodes and the aggregation node.

Each node was created to be highly efficient, low cost and to work on a low-power fashion. For that, and as study on Section 2.4.1, the best microcontroller, communication module and additional hardware were chosen, as each node is described in detail in the following sections.

3.2.1 Sensor Nodes

The sensor node (as the name states) is the node responsible for collecting sensor information and then transmits this to the server to be analyzed. To be able to perform these tasks, the sensor node needs a microcontroller, a radio module and a set of sensors that can be changed from node to node, based on the different needs for the solution. Also, due to the nature of the application and the type of node, a way to power the node without cables is necessary.

Therefore, in order to cope with all these requirements, the sensor nodes are composed of an ESP32 microcontroller, a RFM95W radio module and a set of sensors capable to retrieving information.

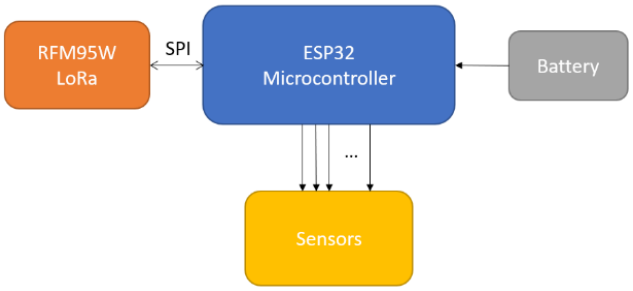

Figure 3.2 - Sensor Node Block Diagram

Since the node only needs to be awake for short periods, to collect the sensor information and transmit them, it can be powered by batteries, creating a true wireless node. This allows the node the ability to have a long lifespan without the need to recharge or replace the batteries. As represented in the Figure 3.3, when a sensor node wakes up, he asks each sensor to collect the current values that they are measuring and then he sends this information to the aggregation node via radio, entering in deep sleep mode right after receiving confirmation that the message was received.

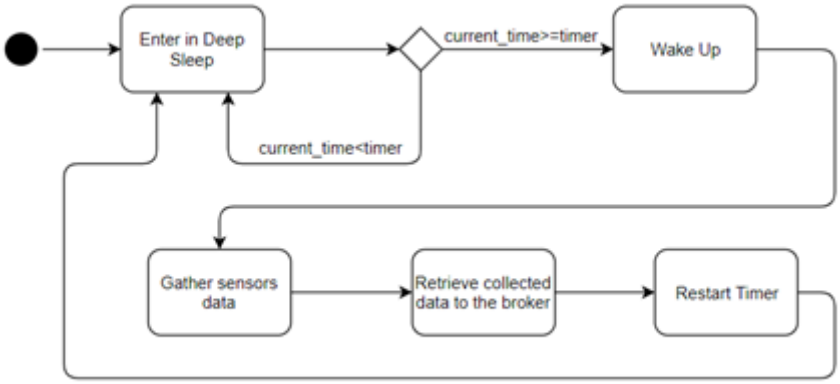

Figure 3.3 - Sensor Node Workflow

For the presented system were defined four sensor nodes, based on the function of each of the sensors used to implement in a real environment implementation:

- Node 1, in the compensation tank, an integrated component in common swimming pool systems, located away from the pool users, and whose purpose is just to control the pool quality. The node will be incorporate pH, water temperature;
- Node 2, in the pipe that provides water to the pool. This node consists of a water flow sensor to detect leaks if they occur;

- Node 3, in the swimming pool itself. The node will be composed of one water level sensor placed in the swimming pool borders;
- Node 4, in the swimming pool surroundings, indoor and outdoor. The node will include a humidity and temperature sensor to evaluate the environment humidity and temperature in the pool location and an LDR sensor to measure the luminance of the space.

3.2.2 Actuator Nodes

Actuator nodes are the one responsible for the remote control of the various actuators that compose the system. Their task is simply wait for new instructions and performing the actions. For that, a microcontroller, a radio module and a way to control high power devices are needed.

Being the most complex node in the system, mainly because of the hardware requirements to control high power actuators, as shown in Figure 3.4 its architecture is composed by an ESP32, an RFM95W and a relay system.

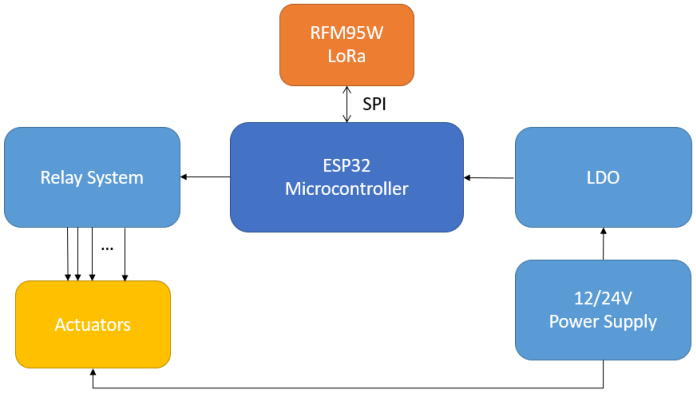

Figure 3.4 - Actuator Node Block Diagram

Actuator nodes can establish bidirectional communication with other nodes that compose the network and are always awake, since new actions can arrive at any moment. This is an important requirement given that they need to receive messages from the aggregation node with actions that were required by the pool's user and to guarantee the network fidelity by sending ACK messages to assure that the requested action was performed successfully.

A research was carried out on the various areas in which this type of nodes could act, which allowed defining two main areas:

- In the pool illumination, that through a light sensor, the system can automatically detect if the pool environment requires to turn the lights on or off, when this parameter achieves low values;
- On water pumps, which should be automatically shut down when water leaks are detected in the engine room or there is an anomalous water flow in the pipe where sensors are collecting data, regarding this parameter.

These nodes are powered directly from the socket since these nodes must always be connected and with the use of batteries, there would be a need to recharge them in short periods. Another reason for excluding batteries is that actuators require a constant voltage power source, which is a requirement that batteries cannot meet.

3.2.3 Aggregation Node

The aggregation node is the one responsible for maintaining the network connected and communicate with the server. It is responsible to receive the messages sent from the sensor nodes and send them to the server, and also receive the actions sent from the server and transmit them to the actuator node. For that, and since it is the only node that communicate with the server, the aggregation node is the only node in the system that uses two communication protocols.

As represented in Figure 3.5, the broker is only composed by an ESP32 and a RFM95W, being the ESP32 built-in Wi-Fi responsible for exchanging messages with the server, using the MQTT protocol.

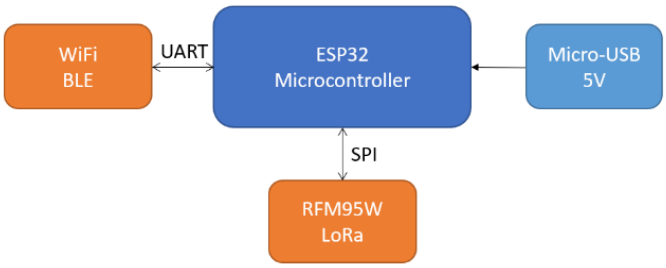

Figure 3.5 - Aggregation Node Block Diagram

The radio module is required to: send messages to the actuator nodes so that they can turn on or off its components; and receive messages from sensor nodes and then forward this information to the server and the mobile application.

When the broker receives an MQTT message with an action to turn off or on a specific actuator, the aggregation node is responsible for forwarding the same broadcast

message via radio, which contains the address of the destination node and the ID of the actuator to turn off/on. Thus, each node will receive that message and through the attached destination ID, they act on their components if their ID matches.

3.3 Software

In order to put the entire system together, a set of software scripts were developed and implemented in all modules of the system.

These scripts are responsible for handle, analyze and store the sensor information collected by the sensor network and also provide the user with a way to monitor and control the entire system. So, a set of server-side scripts was developed, as well as an Android application.

3.3.1 Mobile Application

To allow the users to monitor and control their swimming pools in real time, an Android mobile application was developed, in order to show all the gathered sensor data and also provide control over the actuators. To create a true real time application, the MQTT protocol was used, as was in the broker node of the WSAN.

This app will present all the stored data to the swimming pool users and also enable users with administrator permissions to remotely control the actuators, so as to stabilize the anomalous values detected.

Figure 3.6 presents a storyboard of the application screens.

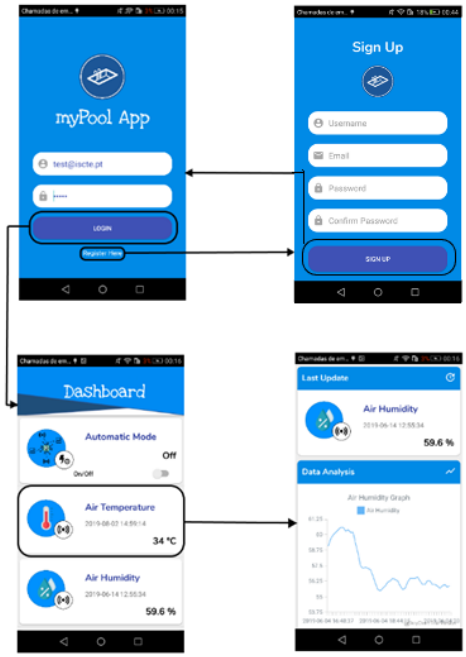

Figure 3.6 - Mobile Application Storyboard

3.3.1.1 LogIn Screen

When the application is launched is presented a LogIn screen (Figure 3.7) for the user's authentication. Username or email and the correspondent password are required to authenticate the user. These credentials are defined upon the user registration on the platform. Also, the registration button is shown in this screen.

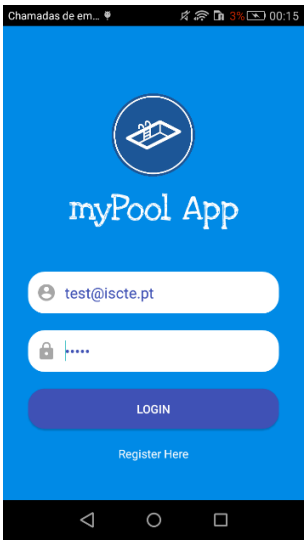

Figure 3.7 - LogIn Screen

3.3.1.2 Registration Screen

The registration screen (Figure 3.8) is shown when the user clicks on the "Registration Here" button, on the Login page. Some personal information is requested to the user such as username, email and password. After the successful registration, the user is redirected to the LogIn screen.

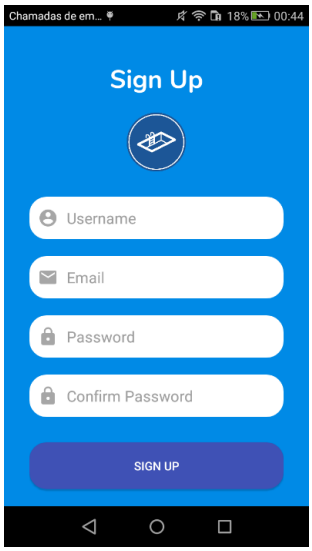

Figure 3.8 - Registration Screen

3.3.1.3 Dashboard Screen

This is the main screen of the application (Figure 3.9) where the control and monitoring features are provided to the pool's users.

Here, are shown all the information about sensors and actuators that compose the network. Based on its type of node, a different card is represented, with relevant information:

- For sensors it is possible to see the name, type of sensor, last value read and correspondent timestamp. When a sensor card is clicked, the user is redirected to the sensor details screen;
- For actuators is shown its type, current state (On/Off) and the timestamp for the last change, as well as a switch to change its state, that sends that action to the hardware component.

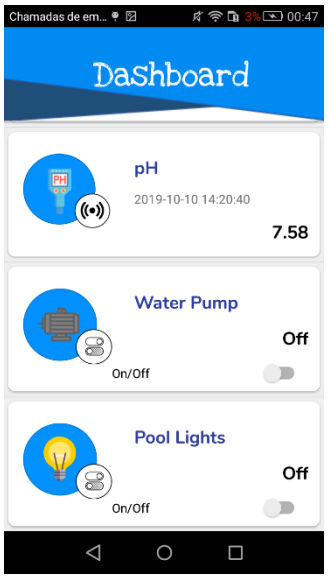

Figure 3.9 - Dashboard

To perform actuator's actions is followed a previous defined model based on a message exchange sequence to assure that this action was actually performed on the hardware before its representation in the database and mobile application. This model is represented in Figure 3.10.

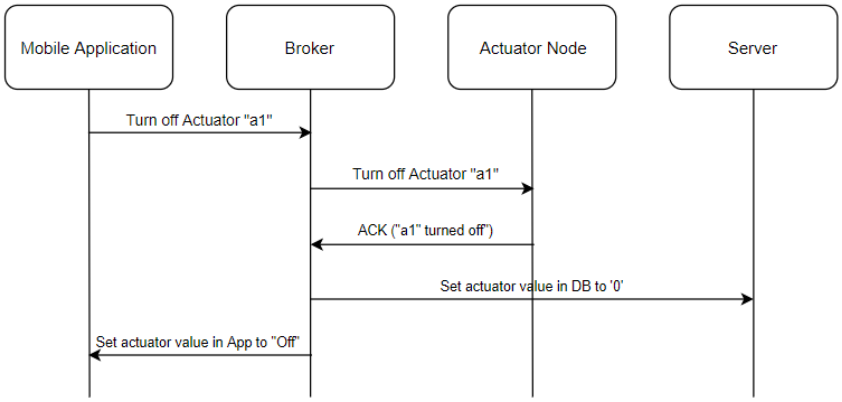

Figure 3.10 - Turn on/off Actuator's Workflow

3.3.1.4 Sensor Details Screen

This screen shows a more detailed information about a specific sensor that was selected by the user. This screen can be divided into three different sections:

1. The sensor card view showed in the previous screen is represented again in this new screen, with last read value (Figure 3.11 a));
2. An interactive graph is displayed, helping the user to do a better and easier analysis of the retrieved data of the selected sensor throughout timestamp. The interactive graph shows, by default, the last fifty values gathered by the sensor, allowing the user to select a specific time interval, using the inputs directly below, to provide the initial and final timestamp. Also, when a specific value is clicked, information about the timestamp and sensor value are shown (Figure 3.11 b));
3. The last section intends to define threshold values for the sensor (minimum and/or maximum) that will be used to notify the user when a retrieved value is not within the defined range (Figure 3.11 c)).

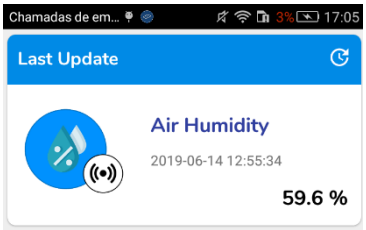

Figure 3.11 a) - Sensor Details - Last Update

Figure 3.11 b) - Sensor Details - Data Analysis

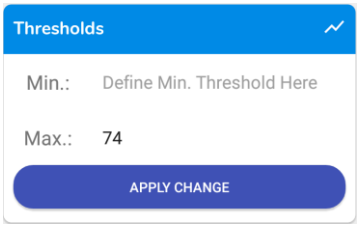

Figure 3.11 c) - Sensor Details - Thresholds

3.3.2 Support Scripts

3.3.2.1 Mobile Application

To assure the support of MQTT protocol and provide communication between the user and the system, the mobile application uses the Paho Java MQTT Client library for Android Platform.

The usage of this library starts with the connection of the application with the MQTT broker, based on client ID. Then the platform subscribes to one topic ("tese/goncalo/in") that allows the reception of messages designated to the mobile application. Each message that arrives to the platform starts with a tag and based on that tag, the application will perform a specific task:

- **Info tag –** a tag used to identify that the received message refers to new data that was collected by the sensor node. When this message is received, the application will refresh the value of the specified sensor in the dashboard and sensor details screen.
- **Notify tag –** Message received from the server that is sent when new data is received from sensor nodes and it exceeds the current thresholds. The reception of this message will trigger a new notification in user's mobile device.
- **Ack tag –** This tag is used to trigger a new action on a specific actuator. This message starts with the specified tag and ends with an "ok", informing the platform that the requested action by the user was performed and the current value of the actuator can be updated in the dashboard.

In Figure 3.12, is represented a flowchart illustrating the application workflow when a new message arrives to the platform.

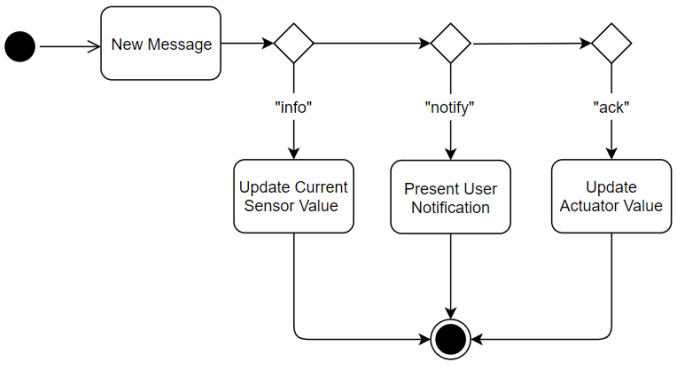

Figure 3.12 - Mobilie Application Workflow

Besides the topic's subscription, this library also allows the application to publish messages to a specified topic. This functionality was used to order state changes in the actuator components. For that was also attached a different tag ("actuator") to inform the server that the sent message intends to turn off/on the actuator with the specified ID. When a message of this type is sent, the application does not change the actuator state in the dashboard. As mentioned, this value is just updated when an ack message is received confirming that the requested action was truly performed in the hardware component.

3.3.2.2 Server

The server is the network's brain, being there where all the gathered data is processed and correctly stored in the database. This is the connection point between the user and the hardware components.

All the gathered information is stored on a MySQL database hosted on an online server, which treats the received data from the system and forwards the treated information to be correctly presented in the application. A more detailed representation of the relational model of the referred database is represented above in Figure 3.13.

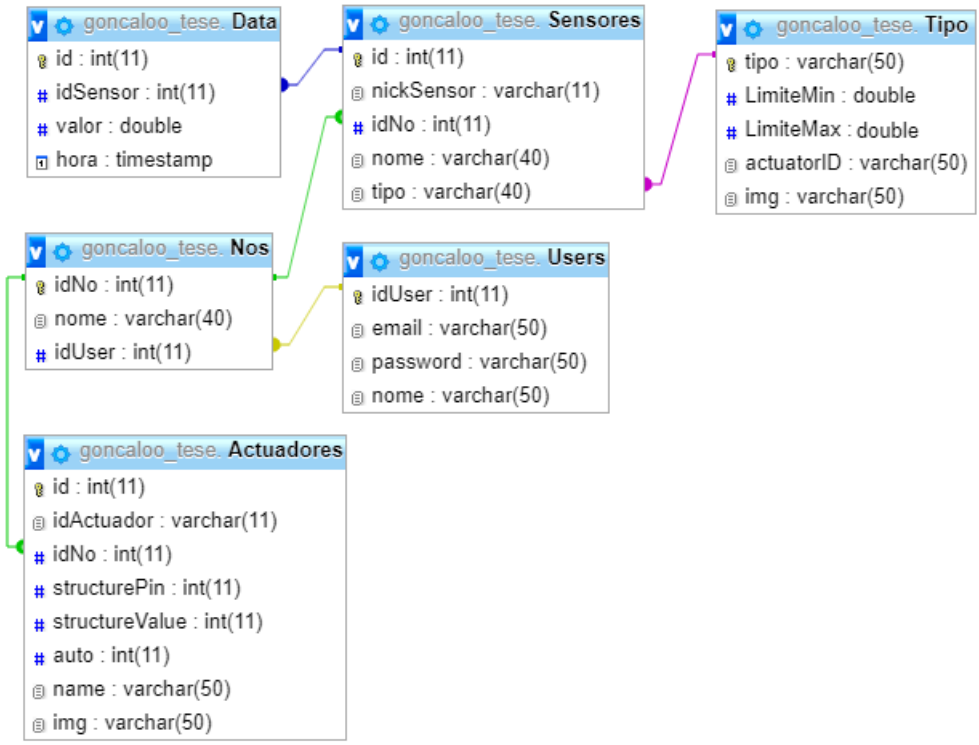

Figure 3.13 - Database Relational Model

On the server a python script, that implements the Paho MQTT Client library to allow MQTT messages to be sent and received, is running. As in the Android application, this script can subscribe to one or more topics, and/or to publish messages that will be sent to the mobile application or to the actuators.

For data storage, the server subscribes to the topic “thesis / Goncalo / in” and the info tag, as in the Android application, is used to indicate that the message is intended for data storage.

For each message received, after storing the new information, the script queries the “Type” table in the database to see if the received sensor values are within the user-defined threshold values. If the value exceeds the limit, a message with the “notify” tag is sent to the application to inform the user that the specific sensor has exceeded its limitations. Also, it is checked if the automatic mode is activated. If this condition is met, in addition to the notification, the server analyzes the data type that is out of range and based on this variable, it decides to turn on or off the actuator responsible for manipulating this data type to normalize this value without the user having to perform any action from the app. This workflow is represented in Figure 3.14.

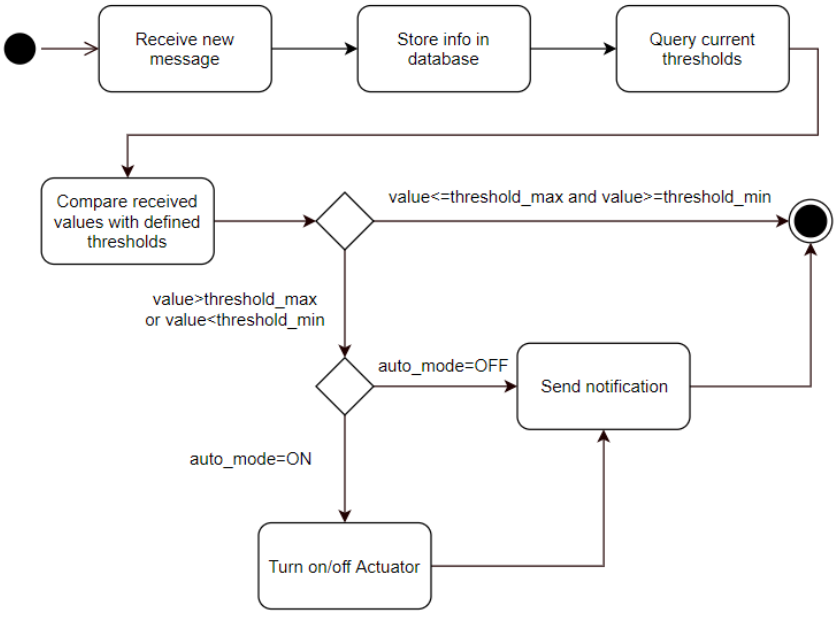

Figure 3.14 - Message analysis with info tag (Server)

Chapter 4 Implementation and Results

The demand for technologies capable of monitoring data via mobile devices has been growing exponentially these days. The developed network allows the users to have a sustainable and economical way to monitor the swimming pool parameters and to control its components through their mobile device.

This chapter describes how the system architecture, designed in Chapter 3, was implemented and tested in experimental and real environment.

A prototype of the system was assembled, capable of simulating the swimming pool behavior under ideal conditions, where the distances between the nodes were not significant. In this phase, the variation of the water level was simulated to verify that the sensor can detect this parameter properly.

During this implementation step, the pH sensor calibration was performed as well as actuator behavior tests, based on the user-defined thresholds in the application.

The tests were divided into two main phases: sensors and communication tests, and actuator tests. The first step aims to assure that the set of selected sensors can read data correctly and make sure that the collected data follows the communication flow until the data storage on the database. The actuator tests aim to evaluate the actuators behavior when there is a user action to turn on or off a specific actuator and when the automatic mode is activated.

Tests were performed on messages exchanged between the nodes (via radio) and between the nodes and the server (MQTT).

4.1 System Prototype

This step aims to implement all the software developed and assemble a simplified model of the system without external factors that can interfere with its performance, such as the distance between the nodes and adverse weather conditions, that negatively influence the achievement of abnormal results by the sensors.

Figure 4.1 shows the prototype of the system used to perform tests, where the distance between the nodes is very small compared to what happens in a real environment.

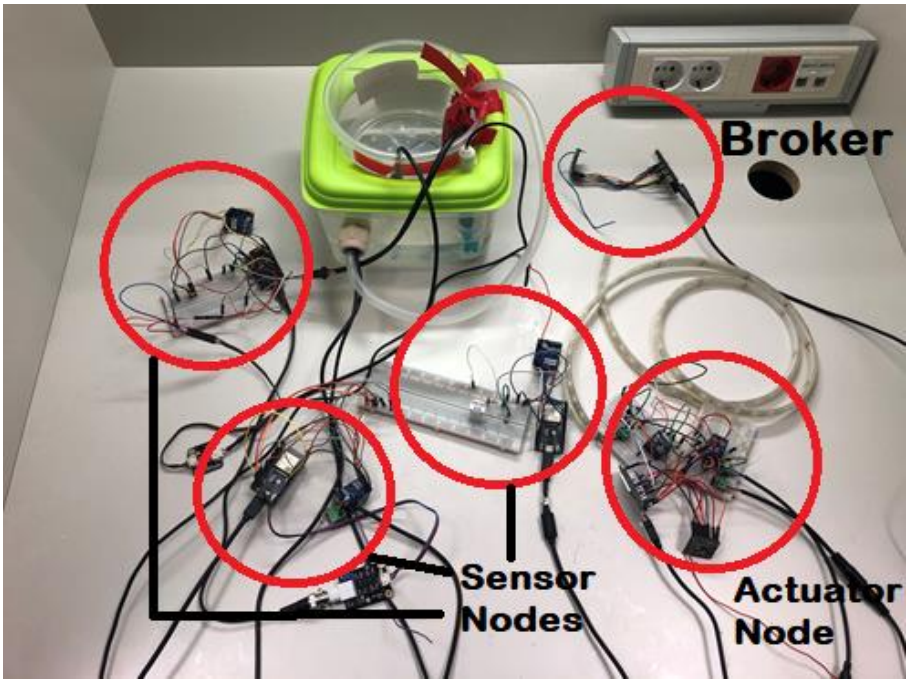

Figure 4.1 - System Prototype

The following sections describe the components of the system used to perform tests in an experimental environment and to analyze the respective consumptions.

4.1.1 Actuator Node

As can be seen in Figure 4.2, the actuator node consists of three main components: an ESP32 board, an RFM95W radio module, and a relay system. The last one makes it possible to switch off/on devices that have a higher voltage through a lower current.

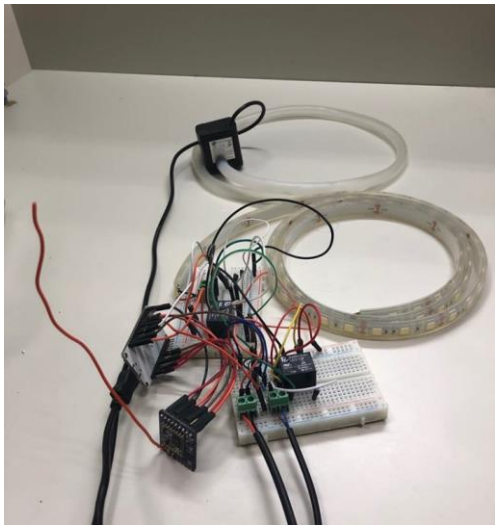

Figure 4.2 - Actuator Node

Two devices were connected to this node, each one destined to a different area of operation:

- Water pump, to allow the variation of the water level in the container;
- White LED strip, to simulate the behavior of turning on and off the pool lights.

In this node, the software was implemented to enable the reception of messages forwarded by the aggregation node regarding actions that the user performed in the mobile application or actions that have been automatically indicated by the server when the thresholds are exceeded, and the automatic mode is activated.

4.1.2 Sensor Node

The composition of the sensor node is based on an ESP32 board, an RFM95W radio module for sending messages to the aggregation node and a set of sensors for data collection.

In the following figure (Figure 4.3) the set of sensors that compose the node are: pH and water temperature.

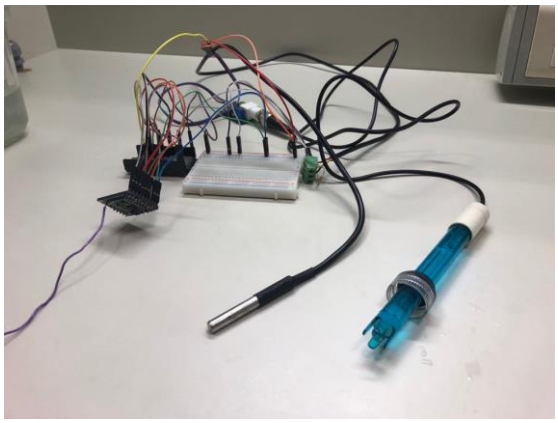

Figure 4.3 - Sensor Node

For the pH sensor, readings have been taken to calibrate the sensor and check whether reading this parameter makes it possible to distinguish between an acidic, neutral and basic solution correctly.

Thus, three containers were used, each with one of the three types of the solution indicated. In the first one, bleach (basic solution) was used, in the second water (neutral solution), whose pH is indicated on the bottle for use as a reference value and in the third vinegar (acid solution).

For each case, the pH values were read over 10 minutes, with an interval of 1 minute between each reading. At the end of the first 5 minutes, in the case of the acidic and basic solution, water was gradually added to ensure that the sensor could detect the variation of this parameter. In the case of the neutral solution, bleach was added for the same purpose.

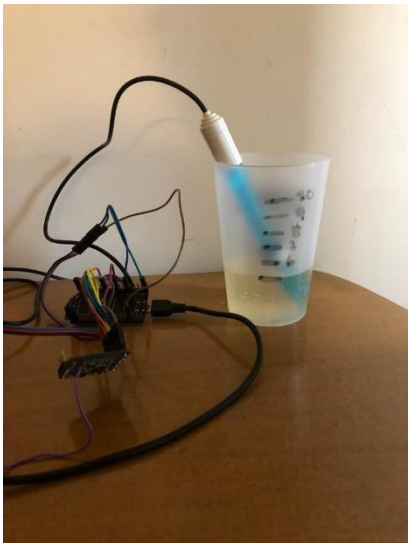

Figure 4.4 - pH Sensor Tests

The results of these calibration tests are represented in the Table 4.1:

Table 4.1 - pH Sensor Readings

Minute	pH		
	Vinegar	Water	Bleach
1	2.89	6.63	12.96
2	2.87	6.66	12.93
3	2.87	6.68	12.97
4	2.88	6.64	12.96
5	2.89	6.67	13.01
6	3.12	7.11	12.46
7	3.46	7.46	12.19
8	3.74	7.93	11.99
9	3.93	8.15	11.73
10	4.05	8.49	11.62

These results show that the pH sensor readings follow the real variation of the solution. For the vinegar case, where it was added water with pH of 6.66 the solution's

pH was increasing, reflecting a progressively neutralization of the solution. The opposite behavior can be seen in the bleach case that starts with a pH that corresponds to a basic solution and it has decreased after adding water to the cup.

In the water case, being a neutral solution, it was also expected that its pH should increase when adding bleach to the cup.

With this calibration and with the obtained results it was possible to assure that the pH readings were correct, once that its increasing and decreasing was reflected for the three tested cases, according to what was expected.

4.1.3 Aggregation Node

In Figure 4.5 is represented the aggregation node used in the developed system, composed only of an ESP32 board and an RFM95W radio module.

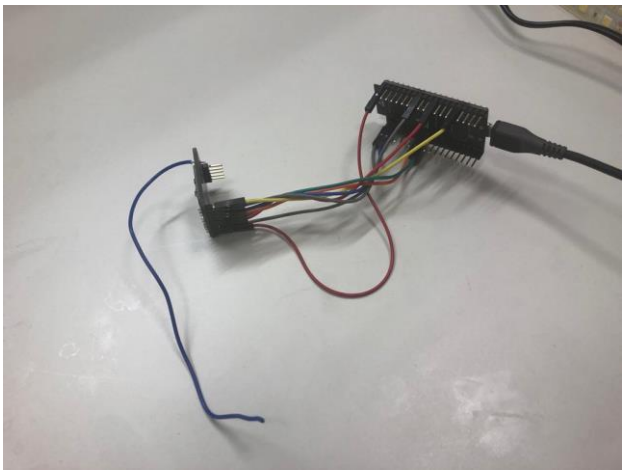

Figure 4.5 - Aggregation Node

This node, despite being simpler in terms of its components, is the node with the most important role in the implemented network, since it is this one that allows the connection of the WSAN network with the server and the mobile application.

Its main role is to forward the messages to its recipients based on the content of those messages. It receives radio messages from sensor nodes with the collected data and messages from actuator nodes that are intended to confirm that the actions indicated by the user and/or server have been correctly performed on the intended components. These messages are then forwarded to the server and/or mobile application.

On the other hand, it also receives messages from the server and the application through the MQTT protocol, aimed at performing actions to turn on/off the actuators, forwarding them to the respective actuator node.

In the case of the MQTT protocol, the HiveMQ[32] platform was used to simulate the sending of messages intended for network nodes and to ensure that the messages sent by the nodes were being sent correctly. For this purpose, the server data needed to establish the connection was entered. Subsequently, the topic "thesis/Goncalo/in" was subscribed to receive the messages sent by the broker (Figure 4.6) and the transmission of messages to the topic destined to the network nodes, "thesis/goncalo/out", was simulated.

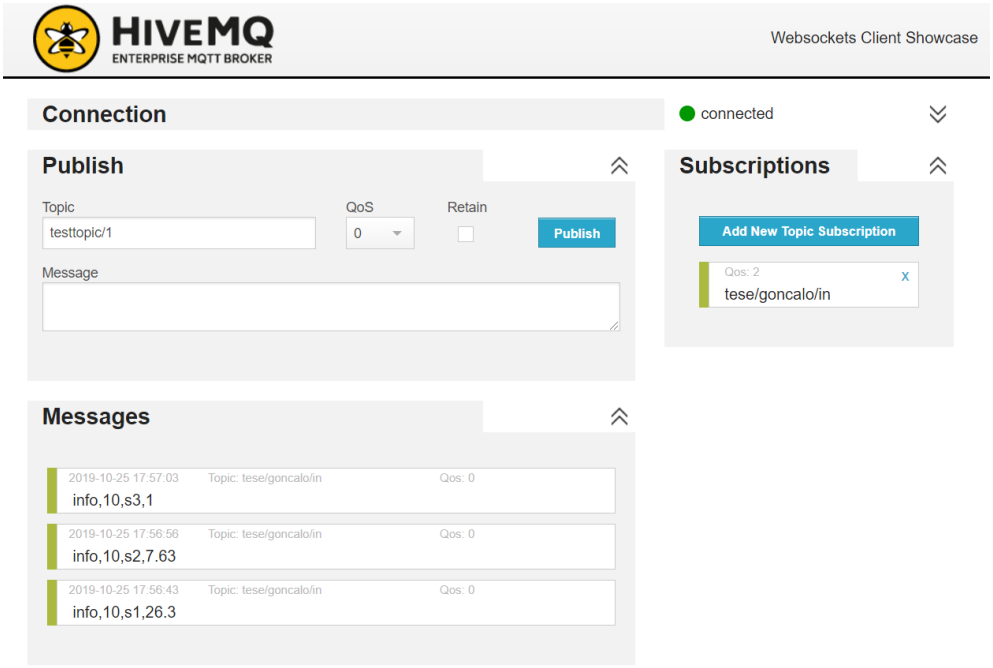

Figure 4.6 - HiveMQ Platform

4.2 System Consumption

Being one of the main objectives of the system, the development of a solution that has low energy consumption, it was necessary to analyze the consumption of the components that make up the nodes. Since only the sensor nodes can be powered by batteries, it will be necessary to predict their duration to ensure that they do not have to be changed and/or recharged in a short period. As illustrated in Figure 4.7, a digital multimeter was used to measure the consumptions.

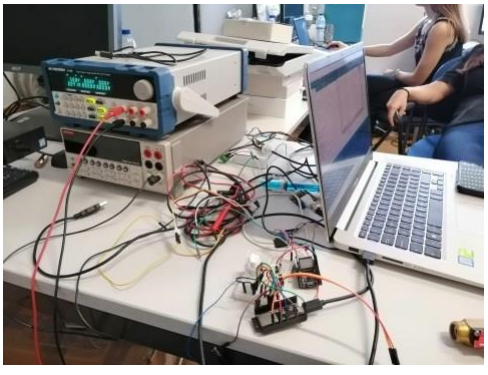

Figure 4.7 - Consumptions Measurements

After the assembly of each node and implementation of the respective software, the consumptions were measured, for each of the sensor node components.

Table 4.2 - Component's Consumption and Cost

	Power Consumption [mA]	**Cost [€]**
ESP32	44	8,73
RFM95W On	4	17,42
RFM95W sending message	36	
Temperature and Humidity Sensor (DHT22)	2	8,69
Luminosity (LDR)	1.2	0,43
Water Temperature	1.5	6,03
Water Level	11	7,77
pH	12	25,76
Water Flow	3	5,15

Based on the obtained consumption values for each component, the node consumption was calculated. For that, was considered that each sensor node in one cycle takes 2 seconds reading data and sending the value to the aggregation node and the enter in deep sleep mode for a period of 4 minutes and 58 seconds.

Knowing that the capacity of one battery ($Cap_{Battery}$) is equal to 1500 mAh, the first step is to convert this value to mAs using the following formula:

$$Cap_{Battery} = 1500\ \text{mAh} \times 3600\ \text{s} = 5400000\ \text{mAs}$$

After that it was necessary to consider the consumptions of each node when it is reading sensors data and sending ($C_{reading\ \&\ sending}$)that to the aggregation node via radio. For this was measured the consumption of an ESP32 with an RFM95W sending a message that assumed the value of 80 mA. Besides this value, it was necessary to consider the sensors consumptions represented in Table 4.2. For this step was considered the formula, having in count the process duration of collecting sensors data and sending it via radio:

$$C_{reading\ \&\ sending} = (80\ \text{mA} + sensor'consumptions) \times 2\ \text{s}$$

The next step is intended for the calculations of the node's consumptions in deep sleep mode ($C_{Deep\ Sleep}$) for the defined period of 4 minutes and 58 seconds. Once the shield's consumption when this mode is activated is 100µA the next formula was applied:

$$C_{Deep\ Sleep} = 100\ \mu A \times (5\ \text{min} \times 60 - 2) = 29.8\ \text{mAs}$$

Based on the application of the previous formulas it was necessary to determinate the number of possible cycles that the battery guaranties based on its capacity, the consumption of the nodes collecting and sending data, and the consumption when they are in deep sleep mode through the next formula:

$$Num\ Cycles = \frac{Cap_{Battery}}{C_{reading\ \&\ sending} + C_{Deep\ Sleep}}$$

After knowing this value, it is possible to calculate the duration of the batteries for each sensor node through the formula presented below. It was necessary to consider the number of cycles that were performed during a period of one hour, having in count the 2 s (collecting and sending sensors data) and 4 min. and 58 s (deep sleep) that corresponds to 12 cycles.

$$Days = \frac{Num\ Cycles}{12 \times 24\ hours}$$

The following table represents the results of the application of the descripted formulas for each node. The nodes numbering in the table corresponds to what is described in Section 3.2.1.

Table 4.3 - Consumptions Results

Node	$Cap_{Battery}$ [mAs]	$C_{reading\ and\ sending}$ [mAs]	$C_{Deep\ Sleep}$ [mAs]	*Number of Cycles*	*Days*
1	5400000	187	29.8	24907	**86**
2		166		27579	**95**
3		182		25495	**88**
4		164		27863	**96**

As can be seen through the obtained results, the way that the sensors are distributed through the set of nodes can guarantee that the system can work without the need of recharging the sensor nodes batteries for a minimum period of 86 days. Analyzing that through an entire year, the user just needs to recharge them four times per year.

4.3 Results in Laboratory

Based on the descripted implementation, tests were performed on the developed system prototype. The main goals of this step are:

- To check if there are being obtained credible results by the sets of sensors and if all the gathered data is being sent by the sensor nodes, forwarded by the gateway, received by the server and correctly stored in the database;
- Actuators behavior and how they can affect the sensor readings, based on the user actions and based on the defined thresholds (automatic mode).

4.3.1 Sensor Tests

The system's sensors were being tested in experimental environment during a period of 10 days. As result of the experimental tests 11366 results were obtained from all the sensors.

In the following figures can be seen the obtained results of each one of the parameters that were defined as relevant for swimming pool's maintenance (air temperature, humidity, water temperature, water level, pH, water pressure). The values of each parameter are represented as a function of date and time.

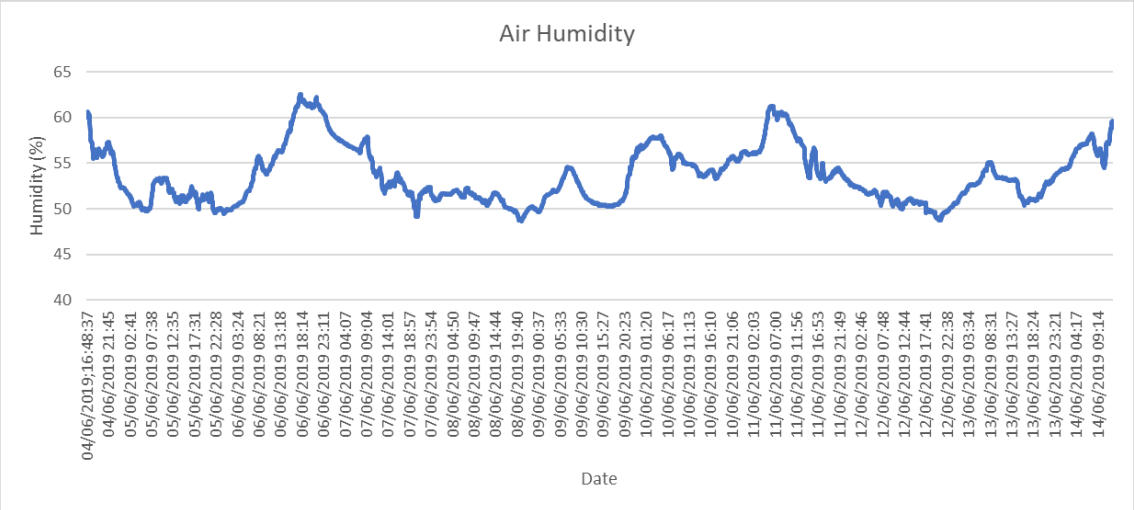

Figure 4.8 - Air Humidity Readings

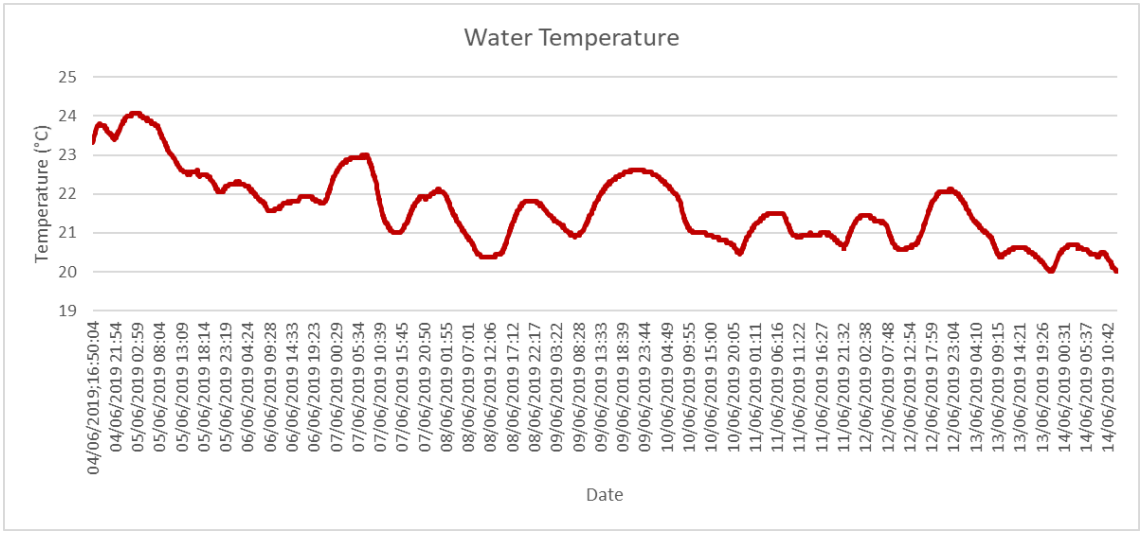

Figure 4.9 - Water Temperature Readings

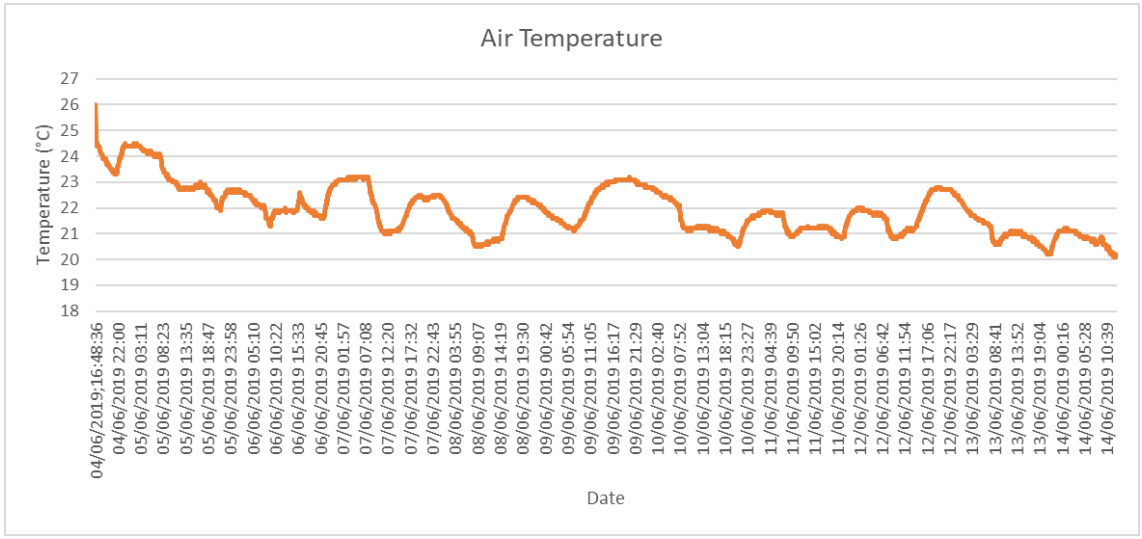

Figure 4.10 - Air Temperature Readings

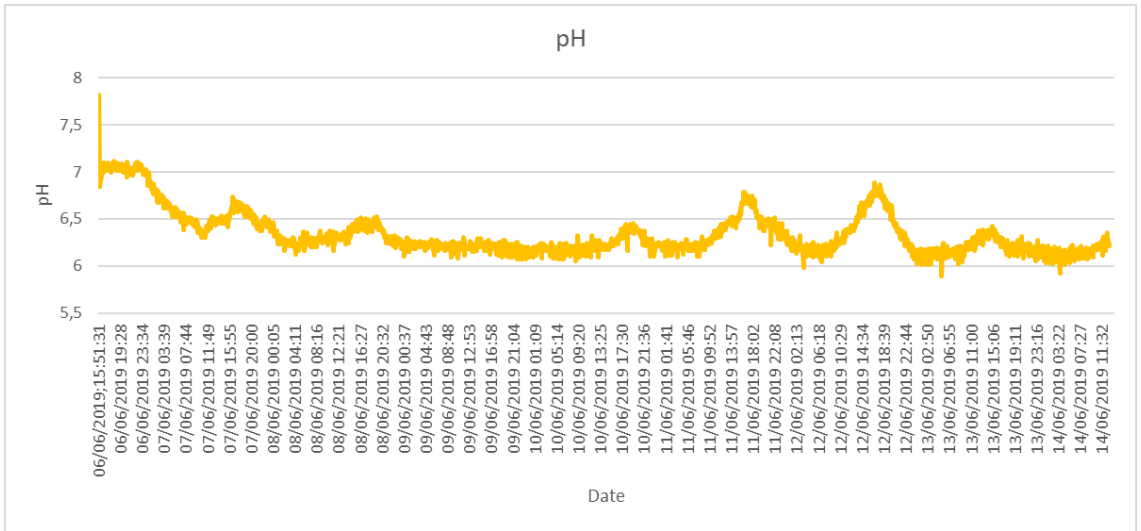

Figure 4.11 - pH Readings

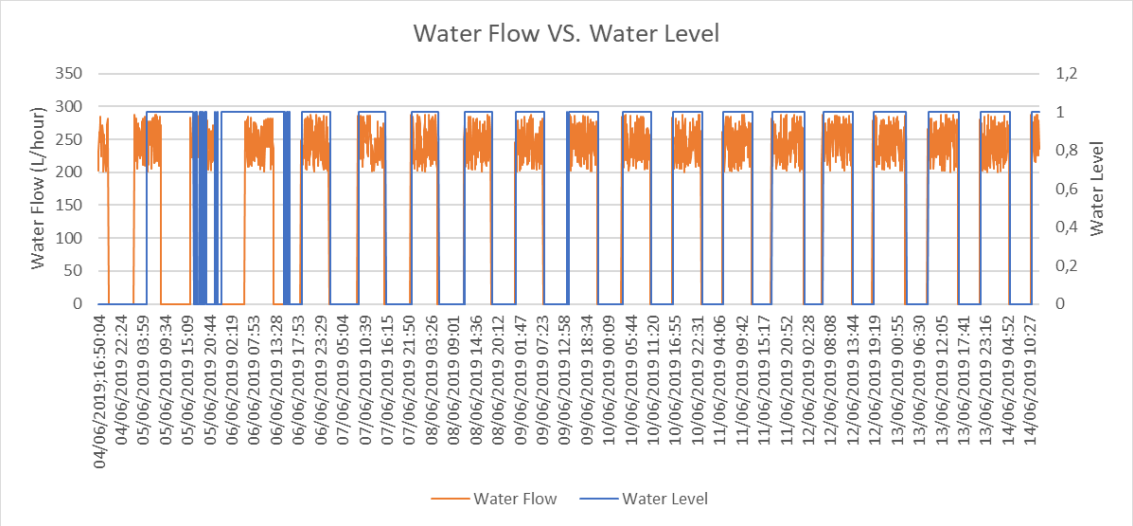

Figure 4.12 - Water Flow and Water Level Readings

The results were obtained from the implementation of the prototype inside the IT (Telecommunications Institute). The data was retrieved by the sensor nodes and sent to the aggregation via radio and then forwarded to the server via MQTT. The server has a python script running that subscribes the defined topic: “tese/goncalo/in”. When the server receives a message from the broker, he processes the retrieved data and store it in the database.

To generate the presented charts some queries were executed in the database to filter data based on the sensor ID. As result of each query were obtained the correspondent number of entries that were exported as a csv file. This file was later opened with Microsoft Excel that allowed the generation of the presented charts.

The exported data was important to extract some conclusions from the data behavior. Comparing the air temperature and water temperature charts it is possible to conclude that both graphs present the same behavior, once the water temperature is affected by the temperature of the environment where the system is located.

Watching the pH chart is possible to note that the gathered values oscillate between 6 and 7,5. Once the water pH assumes an average value of 7 it expected that this parameter fluctuates around the mentioned value.

Finally, it is also possible to establish a relation between the water level variation and the water flow readings. Initially, some problems occurred with the water level readings that were not detecting correctly water inside the recipient. After this detection there were performed some adjustments in this sensor to correct this reading failures that were caused by some problems with cable connections with the shield. After this readjustment, the water level changes when the implemented water pump is

turned off or on for a period of 30 s. In that case it can be seen that when the water flow is 0, the water level does not become to 0 immediately since the container still has a bit of water left when the water flow is 0. For that reason, there is a delay in the water detection by the water level sensor and the water flow readings.

4.3.2 Actuator Tests

Besides the sensor test it was necessary to analyze it the system's actuators can have a positive impact on manipulating the environment conditions such as water level and luminance. Once the actuator's impact in the system could be seen immediately after putting them on or off, these tests were performed in a shortest time when compared with the sensor readings tests.

These tests were performed in a short period just to check if the actuator's node can deal with the user actions sent from the application and is capable of turning them on or off. Each performed action was spaced by 1 min.

Firstly, it was necessary to make sure that the user can put on or off each actuator through the mobile application. There were measured again the luminance, water flow, luminance and water level, once these are the parameters that the actuators are supposed to act. Below are represented the actuator states in function of time and the data collected by the sensors (water level, luminance and water level) when they were turned on through user actions given in the mobile application.

Below are represented the actuator states in function of time (Figure 4.13 and 4.14) and Logcat outputs that are reported every time the user clicks on a switch of the actuator CardView (Figure 4.15).

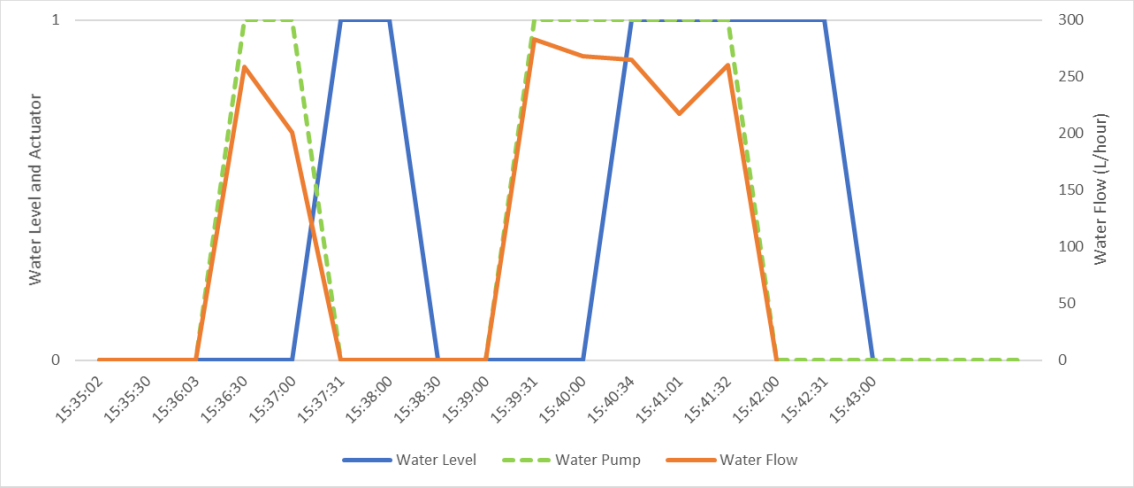

Figure 4.13 - Water Level, Water Flow and Water Pump Actuator Values

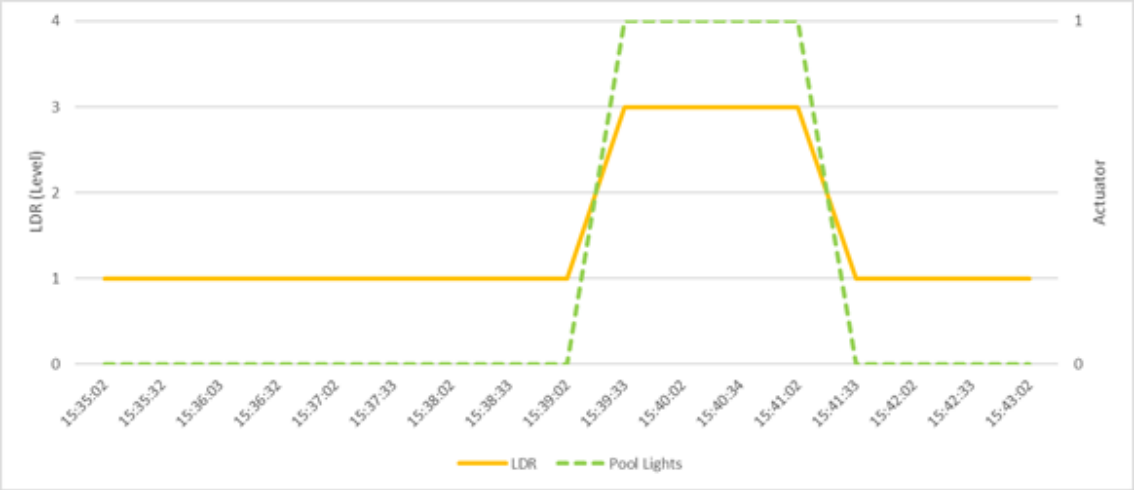

Figure 4.14 - LDR and Pool Lights Actuator Values

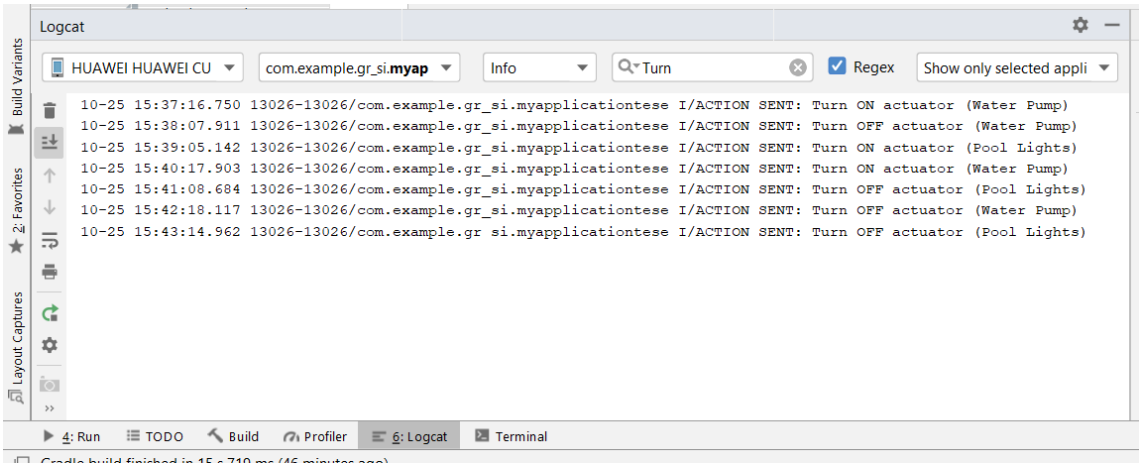

Figure 4.15 - Logcat Outputs

Through the represented charts and Logcat outputs it is possible to conclude that the actuator's state (on/off) manipulated through the mobile application had an impact on sensors readings.

In Figure 4.13, the water pump was activated two times. For each occurrence of turning it on, it can be seen that the water flow sensor immediately detects water passing through it and measures the correspondent values. Once the water level was placed in a height reference, it doesn't detect water immediately when the actuator is activated. The water level sensor, only detects water when the water inside the container achieves that height reference, being the reason of the delay between the actuator's activation and the water level readings.

The LDR sensor readings were adapted to levels of luminance (1-low luminance, 2-medium luminance and 3-high luminance) to turn easily the data analysis to users of the system.

In Figure 4.14, the user only put the lights on once. The LDR sensor was placed close of the lights to easily detect the difference between having the lights on or off. In this figure it's represented that when the user turned the lights on, the LDR sensor readings had grown up drastically.

After the user actions tests to turn on or off actuators, the automatic mode was tested. In this case, the need of turn on or off is detected by the server, having in count the defined threshold values and the new data collected sent by the sensors. To correctly detect the need of turn on or off the lights of the pool it was necessary to place the LDR sensor a little bit away from the light's location.

First is important to have in count that the defined thresholds were level 1 for the LDR sensor (minimum value); Once the water level sensor is responsible for detecting the presence of water and it makes digital readings, assuming values of 0 when the recipient has no water, and 1 otherwise, the sensor is placed in a height reference, and when it doesn't detect water on that height is supposed to turn on the water pump to fill the recipient until the water achieves the water level sensor height.

Below are shown the obtained results for the automatic mode tests.

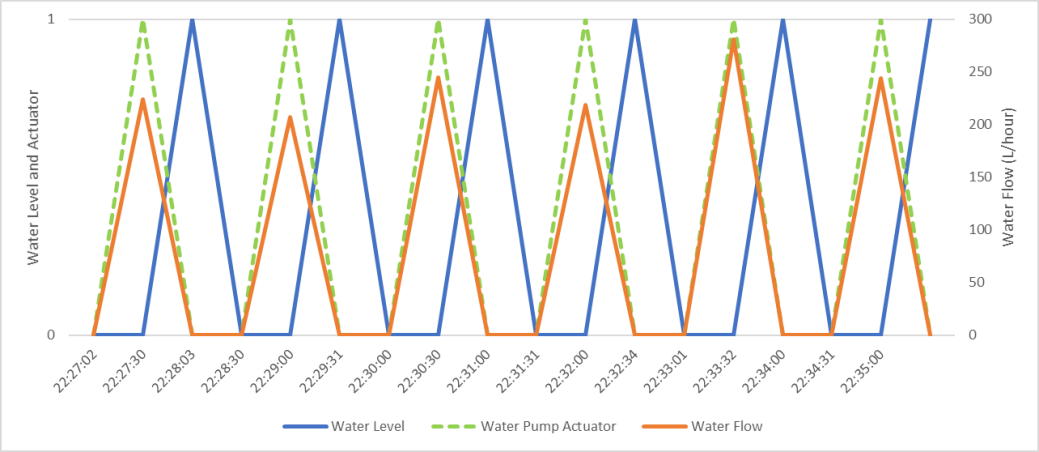

Figure 4.16 - Automatic Tests for Water Pump Actuator

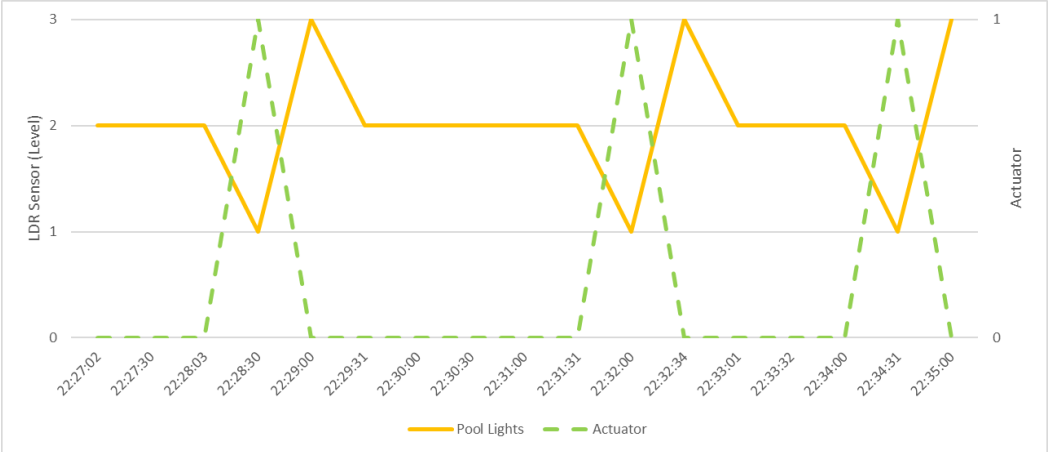

Figure 4.17 - Automatic Tests for Lights Actuator

Through the represented charts it is possible to see that when the automatic mode was activated the obtained results were the expected.

For the water level sensor, all the times that the sensor does not detected the presence of water inside the recipient, the server automatically turned on the water pump. When the server obtained the information that water achieved the height reference where the sensor is located it turned off the water pump. For each time that the water pump was turned off, the water flow sensor read values over 0 L/hour.

In case of the space's luminance, the water level was covered 3 times in the tests period to allow readings of level 1, and all the times that it was done the server when received this information automatically turned the lights on.

4.4 Nodes Implementation in Real Environment

Completed the system implementation in an experimental environment, where it was possible to perform some necessary tests to verify the reliability of the network in the messages exchange and ensure the correct reading and processing of data to be presented later to the user, it was then necessary to do a real case scenario implementation.

This implementation phase assumes a very important role in the usefulness of the developed system, since it will be in a real environment that it will be verified its efficiency in the accusation of anomalies that can occur during swimming pools maintenance, through the correlation of several types of collected data.

For the implementation in a real environment it was necessary to first distribute the sensor nodes to the correspondent location accordingly to what was previously defined. After that some tests were performed in the implemented sensor nodes and the collected data was lately exported from database and imported in Microsoft Excel for analyses these values, correlate them and assure that there were no problems in the pool's maintenance.

In the presented case, the scenario used for tests were the GesLoures swimming pools in Santo António dos Cavaleiros. As can be seen in Figure 4.18[33], distinct zones were defined for the location of each node according to what is detailed in section 3.2.1. Being a case of municipal swimming pools, it was not possible to perform water pressure measurements, nor the implementation of actuators capable of manipulating

the collected data, since no authorization was obtained for intervention in the engine room to implement these components.

Figure 4.18 - Implementation Zones

In zone 1, the node that allows the collection of humidity and air temperature values was implemented. The system was implemented in indoor pool and places like this should maintain a humidity level between 55% and 75%. The air temperature assumes a very important role for conclusions regarding the quality and performance of the pool support systems. Whenever the difference between air temperature and water temperature is outside the range [-2,+2] (ºC), there is a risk that the diluted chlorine in the pool water will evaporate and create a toxic atmosphere that could be harmful to the users of the space. Therefore, the maintenance manager must be quickly notified when a difference outside of the indicated values is detected.

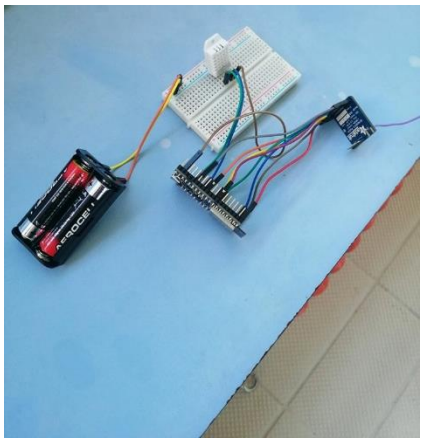

Figure 4.19 - Sensor Node (Zone 1)

In zone 2, the sensors for reading pH and water temperature, located in the pool borders, were implemented. As is described in Section 3.2.1, this node should be placed in the compensation tank, once this is where most of the water analises are ussually done, since all the pool water goes throught there before being filtered and inserted back in the pool. The choice of this place is justified by the fact that this is an area that is considerably away from the pool users. Once that it wasn't allowed to place this node inside the compensation tank, it was placed in the pool borders.

Figure 4.20 - Sensor Node (Zone 2)

Zone 3 is located in the pool itself, where the readings of the water level, pH and water temperature are made. The water level sensor is placed in a reference height and allows to verify if the pool level is lower than supposed to and notifies to the user that he should activate the pumps until this value is normalized. The reference value for the water temperature is 27.5 ºC. Whenever this value increases or decreases 0.5 ºC it is possible to conclude that there is a problem in the engine room related to the water heating process.

Figure 4.21 - Sensor Node (Zone 3)

Finally, in zone 4 is where the gateway is located, which is the node responsible for forwarding the collected data by the sensor nodes to the server by MQTT.

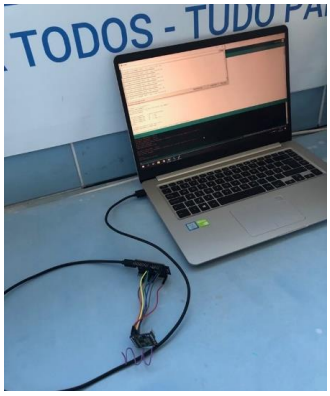

Figure 4.22 - Aggregation Node (Broker)

4.5 Results in Real Environment

After the successful implementation of the sensors nodes in the defined locations, the system was collection data since 9:00 am until 15:30 approximately. For an ideal test case in a real environment case the tests period should be larger. Once the infrastructures that were possible to made tests were a public space and there were going on swimming trainings, there was not possible to let the system collecting data through more than a day.

Through the obtained results in the experimental environment and talking with the responsible for swimming pool's maintenance of GesLoures it was possible to conclude that there was no need to collect data through periods of 5 min. For that reason, each cycle of data collection was spaced by 10 min. due to the insignificant variation of data when this interval was 5 min.

In the following figures are represented all extracted data from database in function of the time.

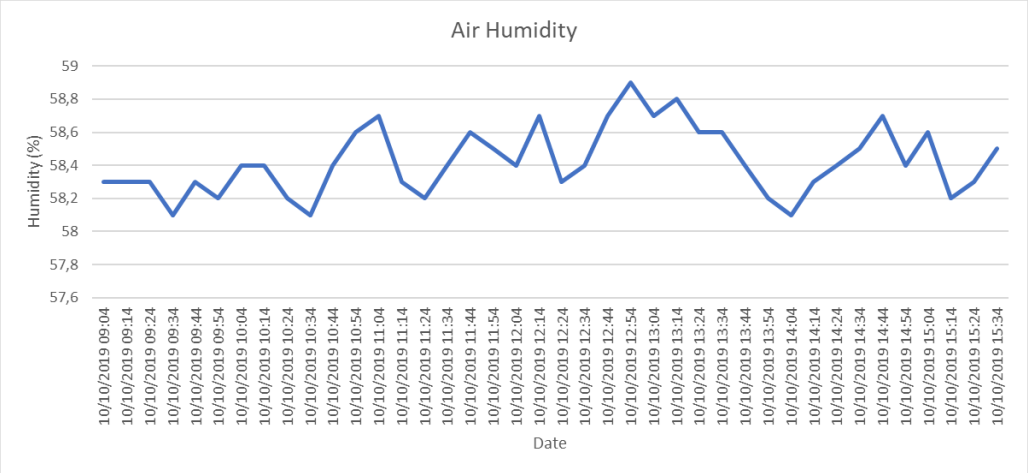

Figure 4.23 - Air Humidity Readings

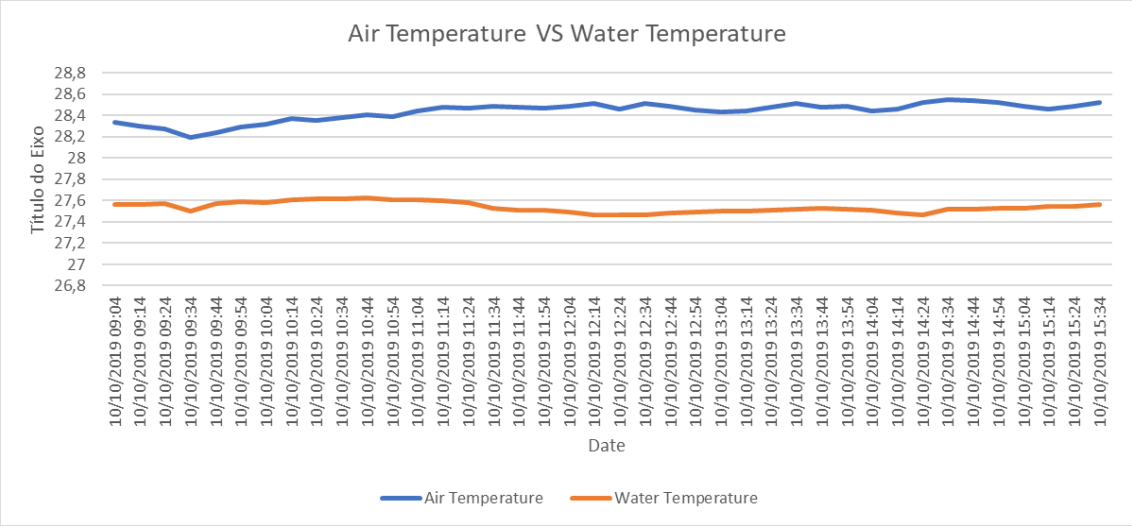

Figure 4.24 - Air Temperature and Water Temperature

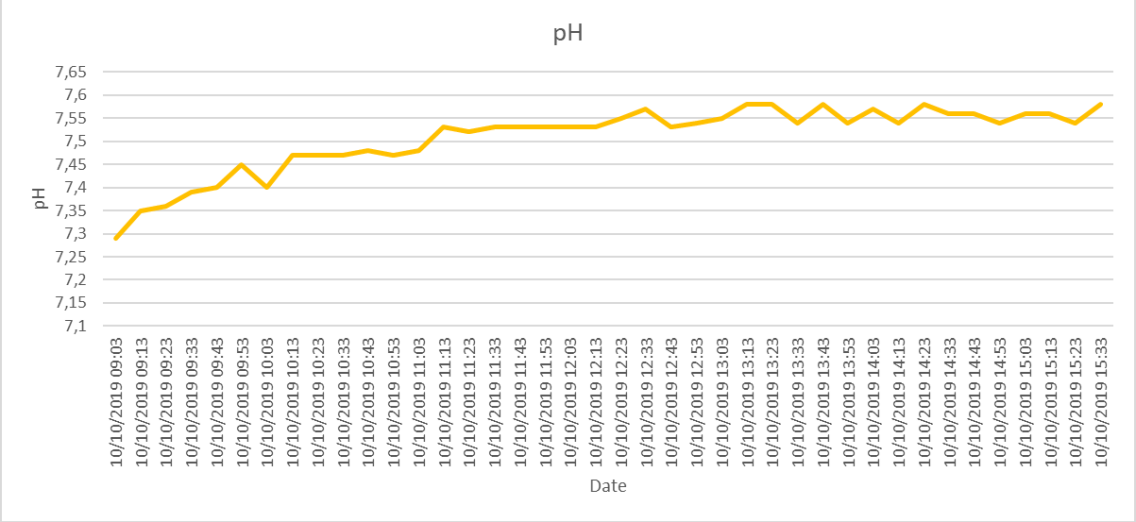

Figure 4.25 - pH Readings

In the first chart, relative to humidity values in the pool surroundings it is possible to see that its values fluctuate approximately between 58% and 59%. As mentioned in the last section in this environment the humidity values always must be between 55% and 75%. The results show that the collected values respect this range. This fact was previously expected once the environment was already a controlled environment due to imposed legislation ranges watching over the public health.

For air temperature and water temperature was necessary to show both values in the same chart due to the correlation that must be performed between these values to make possible the problems detection. Basing on the presented values it is evident that the difference between these values never achieves a value of 1ºC. With this analysis it is possible to conclude that this environment was ensuring that the atmosphere around

the pool was no danger for its users, not presenting the risk of chlorine particles evaporate in the air.

Watching the water temperature can be seen that its variation was never over 0.2ºC, indicating that the water heating system present in the machine room is working fine.

In poll's maintenance is also important to make sure that the water's pH is always as neutral as it is possible, never passing much over or under 7. Analyzing the correspondent chart, the pH of the pool was always accomplishing this condition. This criterion is important to never cause damages in its users. These damages could achieve very serious problems on the skin as well as in case of pool's water inhalation.

The next chart shows that the water level sensor was always detecting water in its high reference.

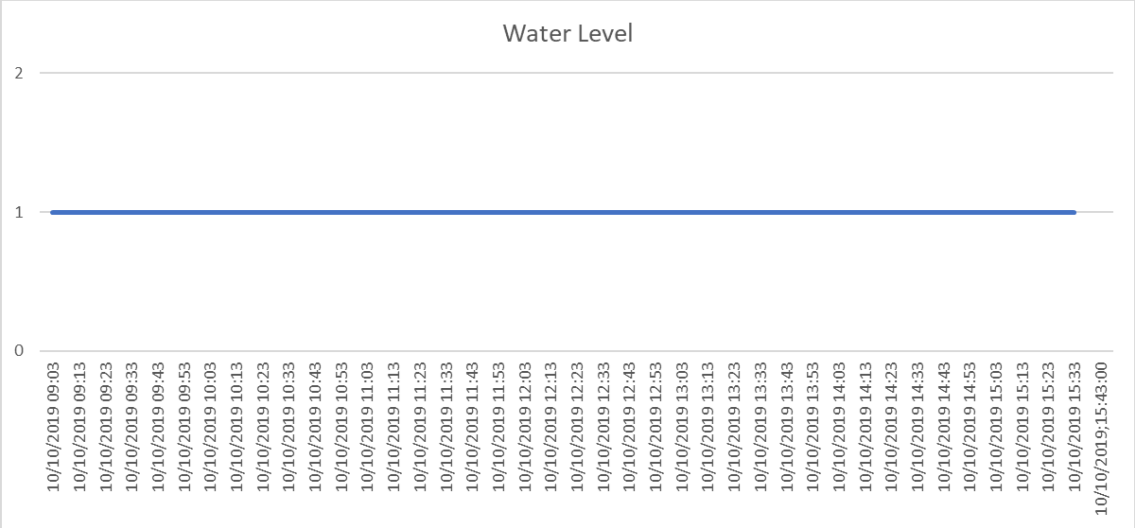

Figure 4.26 - Water Level Readings

Chapter 5 Conclusions and Future Work

5.1 Conclusions

This project presents an IoT solution for a system to control and monitoring swimming pools through a mobile application. The main goals of this solution are to present a system that can contribute to a more sustainable environment, having in count its costs, and energy efficiency.

The system was designed basing on hardware and communication protocols comparing them to select the best things that suit in the intended system.

The type of network used was a WSAN, once the presented system aims to have in its constitution sensing and actuator components. For the node's communication RFM95W radio modules were used, due to its high range and the fact that allows the use of LoRa in this type of communication. The nodes communication was implemented and tested with the use of RadioHead library that provides the node addressing and network fidelity using ACK messages. With the performed tests was concluded that the selected performed messages exchange with no failures.

For the node-to-server communication the selected protocol was MQTT protocol, due to its simplicity scheme of subscribe and publish methods. For that were defined to topics, each one for one way of communication. The use of this protocol was also tested since the sending of messages till the storage of the sent data in MySQL database. As the other type of communication this protocol was implemented and tested, and in case of experimental environment tests were possible to conclude that it performed its role as expected providing the messages to arrive to the server and mobile application and its storage in the database.

After communications tests, it followed the hardware tests with the use of ESP32 microcontrollers, a set of sensors and actuators. The use of these shields allowed the use of Deep Sleep feature on sensor nodes. This feature highly contributed to a lower energy spent when there is not expected the collection of data by the sensors.

The sensors selected for the intended system were previously calibrated and tested individually before its implementation in experimental environment. The more detailed performed calibration was for pH sensor since its importance for the user's safety with use of three solution types. The sensors readings obtained in this step

showed that the sensor was getting correct data and can keep up with pH variations, being a great choice for its purpose.

The actuator tests were performed after the design and implementation of the mobile application. The developed Android application has a user-friendly design that brings an intuitive use of tis features, even for users with lower software knowledge.

After the conclusion of the app development experimental tests were conducted in an environment, with the implementation of all the selected hardware and software. In the initial days of tests, the water level sensor had some problems, that we suspect being of bad cable's contact with the shield, once this never occurred again after re-assemble this sensor. Beside this, not significant error readings the system had a good performance in the experimental environment extracting correct data, storing it in the database and presenting it to the user in the mobile application.

The system was also implemented and tested in GesLoures pool's in Santo António dos Cavaleiros. Once it is a public space, the water flow sensor and the actuators, whose implementation required interventions in pool's infrastructures could not be implemented in this test case. The performed tests period was significantly lower when compared to experimental environment tests. These tests make possible to confirm that the system had a good performance even when the environment characteristics were not ideal, such as node distances, and temperature or humidity levels.

In resume, the presented system had accomplished the main goals of this dissertation which were contribute to a more sustainable environment by the monitoring and control of water use, to lower energy spent when there is no need to collect data from sensor nodes and with the use of batteries in sensor nodes that can keep activated through long periods with no need to be recharged.

5.2 Future Work

Despite of the developed system presents great features to pool's users, it is still possible to make some improvements and optimizations:

- **Online Web Site** – To allow the pool's monitorization and control through their computers with no need of using their smartphones. This site could present extra functionalities to its users that can contribute to a better and monitoring of these infrastructures;

- **iOS application** – Despite of the amount of smartphone users have android devices, could be interesting to develop the same application for iOS devices, increasing the specter of users that adhere to the present system for pool's control and monitoring;
- **Increase sensor types** – The developed system provides data monitoring that is essential to correct control this type of environments. Increase the sensor types used in the system could provide to its users the monitoring of other parameters that should be relevant, for example water turbidity;
- **Implementation in other water infrastructures** – The system allows the extension of nodes number without affect its performance. So that it could be implemented and tested in aquatic parks, where the pool's number is considerably higher compared to the environments where it was implemented.

Appendix A – Scientific Paper

Smart System for Monitoring and Control of Swimming Pools

Gonçalo Simões[1,2], Carolina Dionísio[1,2], André Glória[1,2], Pedro Sebastião[1,2], Nuno Souto[1,2]
[1] ISCTE – IUL, Av. Forças
[2] Instituto de Telecomunicações, Av. Rovisco Pais, 1, Lisbon, Portugal
garss@iscte-iul.pt, cadoo@iscte-iul.pt, afxga@iscte-iul.pt, pedro.sebastiao@iscte-iul.pt, nuno.souto@iscte-iul.pt

***Abstract*— This paper proposes a new scheme for monitoring and controlling the swimming pool's quality through a low-cost system based on wireless sensor networks, which can reduce the requirements for human intervention in the swimming pool maintenance. The main purpose of this system is to provide resources savings for the final user in financial and natural resources, contributing to a sustainable environment. This article also presents a mobile application for interacting with the proposed system which enables users with administrator permissions to control some actions in the swimming pool, in order to stabilize some required parameters for its good quality.**

Keywords—Internet of Things, Wireless Sensor Network, Swimming Pool, ESP32, LoRa, Sustainability

I. Introduction

Information and communications technology, over the years, has been progressively present in human life, being a branch that is in constant evolution and an essential part in our everyday life. With this evolution there is a growing number of physical objects and devices connected to the Internet, giving rise to the concept of the Internet of Things (IoT) [1]. IoT consists of a network of physical objects that are able of being connected to the Internet, identifying themselves, and communicating with each other to achieve a common goal. The main purpose of this technology is to exchange and update data between physical objects and hence to achieve an optimum performance [2].

When talking about IoT it is also useful to approach another concept called Wireless Sensor Networks (WSN), a technology that is inherent in the development of smart systems. Recently, IoT and WSN have been great bets in the development of monitoring and control systems by researchers [3]. This capability of monitoring and control covers a growing range of applications, among which some related to monitoring of swimming pools.

Swimming pool conditions directly depend on how well their chemical properties are monitored [4]. Its maintenance requires the performance of some tests that may be more complex when performed by a human. Therefore, it is important to implement a sensor network that is able to perform those tests correctly and more precisely. To optimize its daily maintenance, instead of manually evaluate the water properties (pH, chlorine, water level, temperature), machine room condition and surrounding space quality, comes up the proposed system, which main purpose is to control and monitoring swimming pools automatically.

Sustainability assumes an important role in preserving the planet Earth sources. Thus, it is increasingly important for the society to have sustainable attitudes in simple daily activities and particularly in the development of new technologies. In the specific case of swimming pools, it becomes relevant to monitor the water and energy consumption levels through a sensor network and, consequently, to adjust their consumption, so that there is no waste of those resources for purposes deemed not essential for the correct operation and maintenance of swimming pools.

Taking into consideration that the incorrect use of fresh water in recreational activities is nowadays a big concern as well as the need to keep track of water quality in order to inform the user about potential risk situations, swimming pool monitoring systems is something we can see in numerous automation projects already developed.

In [3], the authors developed an information and management system for swimming pools. This system consists of a sensor node interfacing with sensors (to measure pH, chlorine level, temperature, cleaner mobility and water level) and actuators, and a web-based user interface that can be accessed remotely for controlling the system. The project has really accurate sensors, providing an optimum maintenance of the swimming pools. However, the implemented sensors are extremely expensive, which creates a system that is not easily accessible to all users.

The authors in [4] developed a low-cost swimming pool automation system that, to keep the swimming pool in good conditions, performs tests on chlorine level and pH and automatically makes the necessary adjustments. The main goal of this project was to lower the energy consumption of the system by reducing automatically the functioning time of the water pump.

Some solutions already exist for the intended application but none was based on the deployment of a WSN with multiples nodes spread across the pool to compare values in multiple locations. Our proposal differs as it adopts a full WSN approach, in a modular architecture where new sensors can be added without compromising the network performance. The system is based on the use of low power consumption nodes and a communication protocol capable of connecting nodes spread along large areas with high reliability and low power.

Saying that, in this paper the authors propose a new system that can monitor and control the water properties (pH, chlorine, water level, temperature, water pressure), machine room and surrounding space quality bearing in mind the reduction of the consumption of natural and material resources, and the monetary saving by the final user.

II. Proposed System Architecture

The purpose of this paper is to develop a control and monitoring system applied to swimming pools (public and private domain). This system is responsible for collecting some significant data from the point of view of maintenance,

978-1-5386-4980-0/19/$31.00 ©2019 IEEE

through a low-cost WSN, in order to evaluate if the swimming pool quality is within the desired parameters, whether at water level, piping leakage, environment quality or engine room efficiency.

The data obtained from the sensors will be stored and treated in such a way as to keep the user informed about the parameters considered relevant to the swimming pool quality. These will be presented to all the pool users and can only be controlled by a user with administrator permissions through an app (in case of private pools, by their owner and in case of public pools by their maintainer). When an anomaly is detected, the user will be notified and able to remotely control some actions on the swimming pool in order to stabilize the anomalous values that triggered the alert.

The proposed architecture follows the same structure of a common WSN. This type of networks can be composed by a microcontroller, a communication system for inter and intra communication between the nodes and a set of sensors [5]. A high-level view of the adopted system architecture can be seen in the Figure 1.

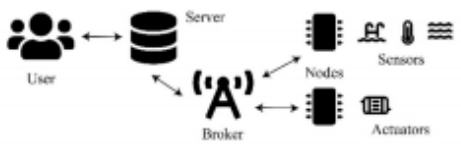

Fig. 1. Proposed system architecture

A. Communication

The system architecture is composed of a single gateway (broker), a set of sensor nodes (each node can support a maximum of eight sensors) and actuator nodes. The gateway is in constant communication with an online server through the Message Queuing Telemetry Transport (MQTT) protocol via WiFi connection. The MQTT protocol is a protocol well-suited for IoT communications, as it is able to provide routing for small, low power, low cost and low memory devices in low bandwidth networks [1].

Although a swimming pool is normally a confined environment, some nodes can be hundreds of meters apart (distance between an outdoor pool and its compensation tank). This combined with the need to have power efficient nodes implies that is useful to select a communication protocol able to meet these requirements. As can be seen in [6], one of the wireless communication protocols that best suits the mentioned characteristics is LoRa, since it is a bidirectional communication protocol that can provide a low power and long range communications, meeting all the requirements for the proposed system and presenting better characteristics when facing, for example, WiFi, Bluetooth, ZigBee or LTE. Therefore, the sensor/actuator nodes will communicate with the broker through a peer-to-peer LoRa connection.

B. Hardware

The proposed sensor network has three types of nodes in its constitution: sensor nodes, actuator nodes and a gateway.

The first has the capability to sense, process and communicate data by themselves [7]. These are mainly composed of a set of sensors that will be periodically sensing a specific type of data. These nodes will only be awake for short periods of time, as they will be powered by batteries, as shown in Figure 2.

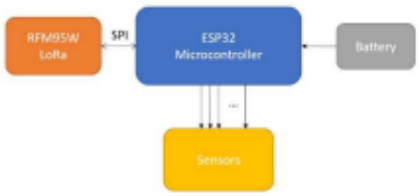

Fig. 2. Sensor node block diagram

Actuator nodes architecture is represented in Figure 3. They are responsible for the system control, having the capability to control and manipulate the system conditions. Particularly, in the presented system, these nodes will provide the user the ability to control the water pumps and pool's chemical conditions.

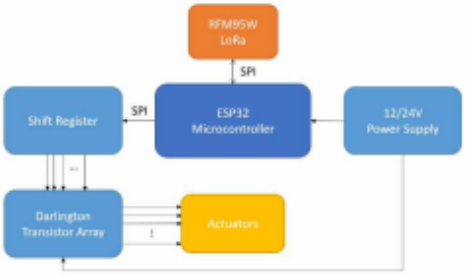

Fig. 3. Actuator node block diagram

The gateway is composed of a microcontroller and a communication module, as shown in Figure 4. This node does not read the sensors data and have the responsibility of collecting data from all the sensor nodes and sending them to the server.

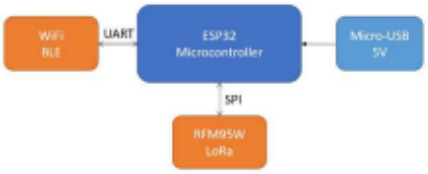

Fig. 4. Gateway block diagram

Each one of the described nodes, despite its different functionalities, will be using the same hardware, which is composed of:

- ESP32 - a microcontroller compatible with IoT applications due to its low cost, low power system with integrated WiFi and dual-mode Bluetooth capabilities [8]. Besides that, it has power saving features, among which there is one named Deep Sleep. This feature will be applied to the sensor nodes so that they only "wake up" when they need to send data to the broker.

- RFM95W - LoRa transceiver that provides ultra-long-range communication and high interference immunity whilst minimising current consumption with a sleep function which contributes for a lower power consumption [9].

The selected hardware was chosen due to its high performance and characteristics. Regarding the microcontroller, the dual-core, low consumption in Deep Sleep Mode, and multiple communication protocols available were decisive when facing other common microcontrollers for IoT projects, such as the ESP8266, Arduino Uno or Raspberry Pi. In terms of communication modules, the RFM95W is the most common LoRa transceiver, allowing not only a local network but also LoRaWAN connectivity, something that the RFM69, RFM98 or RN2483 are not capable.

To accomplish the power saving criteria, the minimum time interval for sensor data collection will be analysed so that the sensors do not have to send constant data in real time, and thus can have lower energy requirements.

C. *Software*

The collected data will be treated and stored in an online server that sends these values to an Android application (Figure 5). This app will present all the stored data to the swimming pool users. Besides the data representation, the app will enable users with administrator permissions to remotely control the actuators so as to stabilize the anomalous values detected.

Fig. 5. Android application

The whole process, starting from the data collection to its presentation in the android application is represented in Figure 6.

The system nodes communicate with each other through a peer-to-peer LoRa connection, using the RFM95W modules. This system will use the Arduino library RadioHead [10], that enables the nodes addressing.

The nodes can communicate in two different ways, both encrypted by the API:

- Send directly a message to a specific node, using the destination address;
- Sending a broadcast message with the destination node ID to every node in the network;

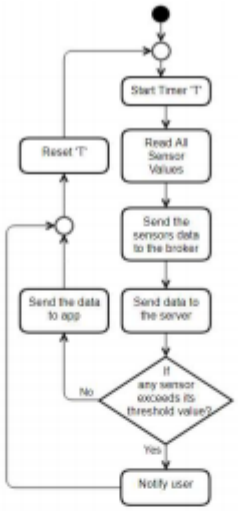

Fig. 6. Flowchart of software implementation

In the first scenario, the RadioHead library assures the network fidelity by sending an automatic ACK response to the origin node that sent the message, using its ID. However, when broadcast message is sent, the library cannot guarantee the reliability of the network. To cope with this limitation, each destination node will read the ID on the message, and if this parameter corresponds to its ID, the node will accept the message, confirming that it was received by sending an ACK response to the origin node. Our proposed system uses both communication methods, the first when sensor nodes send information to the gateway and the second when the gateway sends information to the sensor node.

The use of an ACK response will improve some key features in IoT systems, such as network efficiency and reliability.

III. Proposed System Implementation

Designed and presented the system architecture capable of control and monitoring swimming pools, will be realized the following implementation to evaluate the system architecture.

The proposed system implementation consists of three sensor nodes to be applied as follows:

1. In the compensation tank, an integrated component in common swimming pool systems, located away from the pool users, and whose purpose is just to control the pool quality. The node will be incorporate chlorine level, pH, temperature and water pressure sensors;
2. In the swimming pool itself. The node will be composed of two water level sensors placed in the swimming pool borders;
3. In the swimming pool surroundings, indoor or outdoor. The node will include a humidity and temperature sensor to evaluate the environment humidity and temperature in the pool location.

To measure the mentioned parameters, the following sensors will be used:

- DS18B20, a digital thermometer that allows temperature readings from -55ºC to +125ºC [11] and will be used to measure the water temperature;
- DHT22, a temperature and humidity sensor that allow temperature readings between -40ºC and +80ºC and humidity readings between 0% to 100% [12] (environment temperature and humidity);
- POW110D3B, a water flow sensor composed of a valve body, a water rotor and a hall-effect sensor that outputs the corresponding pulse signal [13];
- SEN01161, an analog pH meter kit, that allows pH readings from 0 to 14 [14];
- SEN0165, an analog ORP meter, that takes into account the concentration and the activity of the chlorine in the water and provides a measurement of the effectiveness of the chlorine [15];
- SEN0205, a liquid level sensor that operates using optical principles, has good sensitivity and no need for mechanical parts [16];
- SEN0189, a turbidity sensor, that detects water quality by measuring the levels of turbidity. It uses light to detect suspended particles in water by measuring the light transmittance and scattering rate [17].

When the collected data from the sensors exceeds the threshold values, the system will notify the user through his smartphone, so that the he can take control of the swimming pool to stabilise the values that triggered the alert. For that, an actuator node will be implemented alongside the pumps and chemicals controllers.

The proposed system is flexible in the sense that new nodes can be added to create more rigorous analyzes, without compromising the network capabilities.

IV. Conclusions and Future Work

This paper presents a proposal for a system that provides users the capability to effectively control and monitor their swimming pools. The proposed solution has the main goal of promoting a sustainable environment, reducing the consumption of natural and financial resources for the final user and thus achieve a high-efficiency level. The proposed architecture is based on WSNs, combining low-cost and low power hardware with long range communication modules, that allows the system to be more efficient and run on batteries for extensive time periods.

For future work, the system needs to be implemented and tested in real case environments in order to analyse its performance and thus conclude if it is possible to reach the desired efficiency level.

Although this system was designed for swimming pools, other water features, such as artificial fountains or lakes, can be used as additional implementation scenarios.

Acknowledgements

This work was supported in part by ISCTE – Instituto Universitário de Lisboa from Portugal under the project ISCTE-IUL-ISTA-BM-2018 and by the FCT – Fundação para a Ciência e Tecnologia and Instituto de Telecomunicações under project UID/EEA/50008/2013

References

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Commun. Surv. Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[2] H. N. Saha, A. Mandal, and A. Sinha, "Recent trends in the Internet of Things," *2017 IEEE 7th Annu. Comput. Commun. Work. Conf. CCWC 2017*, pp. 15–18, 2017.

[3] J. M. Marais, D. V. Bhatt, G. P. Hancke, and T. D. Ramotsoela, "A web-based swimming pool information and management system," *IEEE Int. Conf. Ind. Informatics*, pp. 980–985, 2017.

[4] R. Gouws and A. Nieuwoudt, "Design and cost analysis of an automation system for swimming pools in South Africa," *(DUE), 2012 Proc. 20th*, no. 2, pp. 9–15, 2012.

[5] G. S. Menon, M. V. Ramesh, and P. Divya, "A low cost wireless sensor network for water quality monitoring in natural water bodies," *GHTC 2017 - IEEE Glob. Humanit. Technol. Conf. Proc.*, vol. 2017–Janua, pp. 1–8, 2017.

[6] A. Gloria, F. Cercas, and N. Souto, "Comparison of communication protocols for low cost Internet of Things devices," *South-East Eur. Des. Autom. Comput. Eng. Comput. Networks Soc. Media Conf. SEEDA-CECNSM 2017*, 2017.

[7] K. R. R. Babu, S. J. George, and P. Samuel, "Optimal sensor selection from sensor pool in IoT environment," *Proc. 2016 2nd Int. Conf. Appl. Theor. Comput. Commun. Technol. iCATccT 2016*, pp. 697–702, 2017.

[8] S. B. Biswas, "Solar Water Pumping System Control Using a Low Cost ESP32 Microcontroller," *2018 IEEE Can. Conf. Electr. Comput. Eng.*, pp. 1–5, 2018.

[9] Adafruit, "RFM95_96_97_98W." [Online]. Available: https://cdn-learn.adafruit.com/assets/assets/000/031/659/original/RFM95_96_97_98W.pdf?1460518717. [Accessed: 04-Dec-2018].

[10] Adafruit, "RadioHead." [Online]. Available: https://github.com/adafruit/RadioHead. [Accessed: 05-Dec-2018].

[11] M. Integrated, "DS18B20 Datasheet." [Online]. Available: https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf. [Accessed: 07-Dec-2018].

[12] SparkFun, "Humidity and Temperature Sensor - DHT22." [Online]. Available: https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf. [Accessed: 07-Dec-2018].

[13] Chipdip, "G 1/2 Water Flow sensor." [Online]. Available: https://lib.chipdip.ru/583/DOC000583441.pdf. [Accessed: 10-Dec-2018].

[14] DFRobot, "PH meter(SKU: SEN0161)." [Online]. Available: https://www.dfrobot.com/wiki/index.php/PH_meter(SKU:_SEN0161). [Accessed: 10-Dec-2018].

[15] DFRobot, "Analog ORP Meter(SKU:SEN0165)." [Online]. Available: https://www.dfrobot.com/wiki/index.php/Analog_ORP_Meter(SKU:SEN0165). [Accessed: 10-Dec-2018].

[16] DFRobot, "Liquid Level Sensor-FS-IR02 SKU: SEN0205." [Online]. Available: https://www.dfrobot.com/wiki/index.php/Liquid_Level_Sensor-FS-IR02_SKU:_SEN0205. [Accessed: 10-Dec-2018].

[17] DFRobot, "Turbidity sensor SKU: SEN0189." [Online]. Available: https://www.dfrobot.com/wiki/index.php/Turbidity_sensor_SKU:_SEN0189. [Accessed: 10-Dec-2018].

Appendix B – User & Technical Manual

Department of Information Science and Technology

myPool System – User & Technical Manual

Gonçalo Alexandre Rodrigues Simões

October, 2019

Contents

List of Figures

List of Tables

User & Technical Manual

The developed software was an Android mobile application which main goal is to provide user to monitor in real time swimming pool's parameters such as air temperature, humidity, water temperature, pH, water level, water flow and luminance of the space. It provides the user to remotely control some electric components included in swimming pool systems by turning them on or off. To use this application, it is required to connect the device to the Internet.

1.1 Mobile Application

1.1.1 LogIn

When the application is launched the LogIn screen is presented to the user. Each user has a username, e-mail and password. In this screen user's authentication is required to access to all the application functionalities.

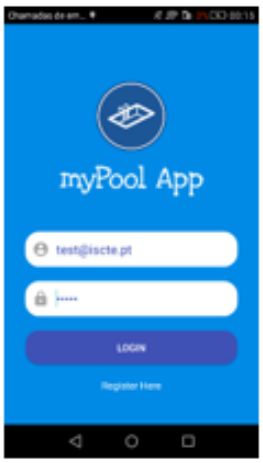

Figure 1.1 - LogIn Screen

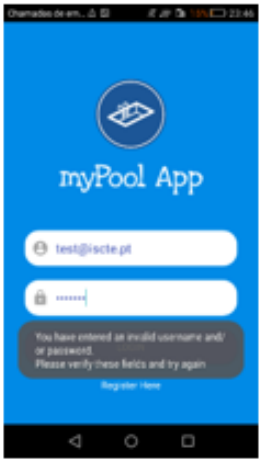

Figure 1.2 - LogIn Screen Error

If the LogIn fields were filled with invalid credentials that are not stored in the database, it is presented an error message to alert the user to verify the inserted inputs and try again to perform the authentication.

In case of being the first access to the application, the user needs to click on "Register Here" button to perform its registration.

1.1.2 Registration

For user's registration username, email and password are required. To confirm that the user knows the inserted password it is also required its confirmation. If the inserted inputs on both password fields does not match, a warning message is launched. After a successful registration, the user is redirected to the LogIn screen.

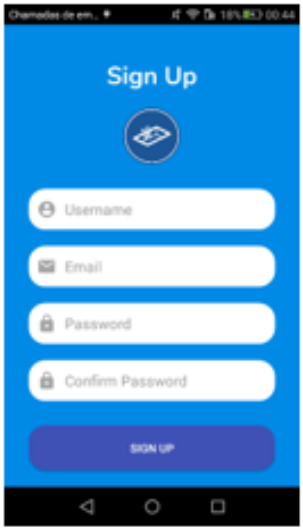

Figure 1.3 - Registration Screen

1.1.3 Dashboard

Dashboard is presented after a successful authentication on LogIn screen. It shows an overall view of the collected data by the system and the current values of each sensor, actuator (on/off) and automatic mode.

This screen presents a list of three types of card views:

Sensor card view which contains the name of the parameter that is being presented (e.g. air temperature), the current value and date and time when this value was read by the correspondent sensor.

Figure 1.4 - Sensor CardView

5

Actuator card view shows the component's name that the user can turn on or off in the swimming pool environment, and a switcher to perform that action. The state of the button only changes in the application when a confirmation is received from the correspondent actuator.

Figure 1.5 - Actuator CardView

Automatic mode is similar to actuator's card view, showing its title and a switch to activate or deactivate this mode.

Figure 1.6 - Automatic Mode CardView

1.1.4 Sensor Details

Sensor Details screen presents all the information correspondent to one sensor, providing an easier analysis of the collected values.

On top it shows the same card view presented in the dashboard screen with the current value of the sensor and date and time of its collection.

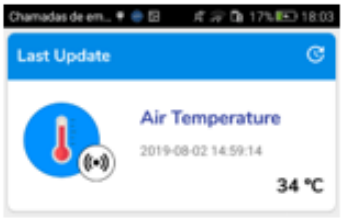

Figure 1.7 - Last Update Section

Next to this, the "Data Analysis" view is shown, which provides a more detailed analysis of all the collected data between a specific period through a graph. When users click on the graph is shown the sensor value for a specific date and time. When this screen is launched, the chart represents the last 50 values of the sensor. After that, it is possible to filter the data representation by filling the correspondent inputs ("From" and "To") and

then click on the "Apply Filter" button. These inputs are chosen through a calendar that is presented when the field is clicked.

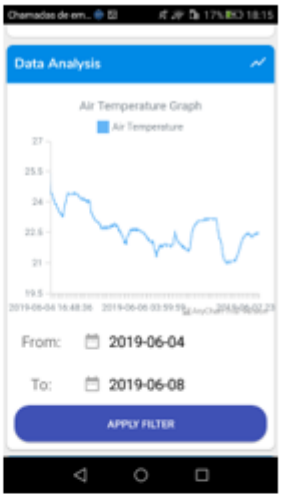

Figure 1.8 - Data Analysis Section

Figure 1.9 - Calendar Picker

At the end are presented the current thresholds for the sensor that was clicked on the dashboard: minimum and maximum. These thresholds main goal is to notify the user when the collected data exceeds those values. They are also important to the automatic mode activation, once it is through this values that the server can take decisions to turn on or off the correspondent actuators.

The threshold values can be changed by the user deleting the current value of the field that is intended to change and click on "Apply Change" button (Figure 1.10).

Figure 1.10 - Thresholds Section

1.1.5 Notifications

As described before the application notifies the user when a new value is collected, and it exceeds the defined thresholds for that sensor. Independently of having automatic mode activated or not the user will always be notified when a threshold value is exceeded.

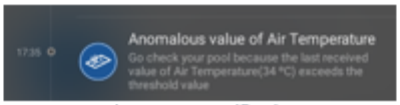

Figure 1.11 - Notification

1.2 Server

For Backoffice development a server was used, a database, PHP scripts and Python scripts.

The used server is responsible for store the collected data by the system nodes, the user's accounts and update data retrieved from the mobile application.

Data visualization through the mobile application is possible with the use of this server by sending requests to retrieve data from the database. Data is retrieved by php scripts that are hosted in the server, existing one script for each type of request sent by the mobile application.

The use of Python scripts running in the server allows it to receive data retrieved from the system nodes through MQTT protocol. Were developed two Python scripts, one for each topic used in MQTT:

- Tese/goncalo/in;
- Tese/goncalo/out;

1.2.1 Database

To correctly create an efficient database, a study was performed to gather all the data that is needed to store for the developed system. After this study, a database was created. The correspondent UML diagram is represented in Figure 1.12.

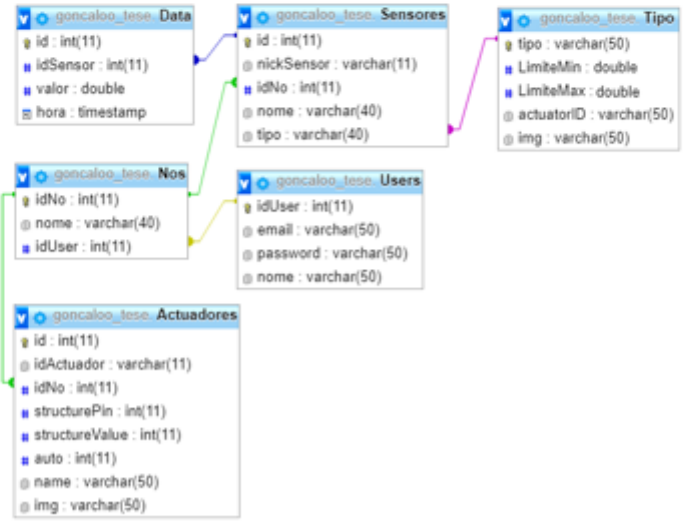

Figure 1.12 - Database Relational Model

The database constitution is based on six tables:

- **Users** – This table is where is represented all the user's information given in the sign-up process. The idUser is automatically generated when a new user is registered, and it is the primary key of this table. The password column is encrypted to maintain the user's account as safety as possible.
- **Nos** – It is where each system's node is registered. The idNo corresponds to the defined id in the Arduino software hosted in each shield node. The nome column matches to the location of the node (e.g. compensation tank). This table also has foreign key that corresponds to user's id. This way it is possible to distinct the nodes of each user of the mobile application.
- **Actuadores** – It is where the actuators information is represented. Each actuator is always associated to one node. The columns idActuator, idNo, structurePin and structureValue corresponds to what is defined in the Arduino software. The structurePin is the shield pin where the actuator is connected and the structureValue corresponds to its current value (0[Off]or 1[On]). The auto column only assumes 0 or 1 value and it is responsible for store if the system's actuators are with the automatic mode activated or not. The img and name are an auxiliary information to match the actuator's representation on the mobile application with the correspondent icon and title.
- **Sensores** – This table represents each sensor that compose each sensor node in the system. The nickSensor and idNo are defined in the software implemented in the system nodes. Column tipo is a foreign key from the Tipo table that is the type of data collected from the sensor.
- **Tipo** – Represents each type of data collected by the sensors. Each sensor has a type associated. For each type exists a maximum and minimum threshold. This table allows the storage of the defined thresholds for each sensor. Each type could have an actuatorID associated if there exists one actuator in the system that could have an impact in the swimming pool environment.
- **Data** – This table stores all the collected information by the sensors. It stores the value collected by the sensor, its ID and the timestamp of when this value was collected.

1.2.2 PHP Scripts

PHP scripts are necessary to retrieve data stored in the database for its representation in the mobile application and to perform the user's login and sign up.

In Figure 1.13 are represented all the PHP scripts developed for the presented system.

/var/www/html/goncaloo/files/

Nome	Tamanho	Alterado	Direitos	Dono
..		15/03/2019 21:56:49	rwxr-xr-x	nobody
getLastSensorValues.php	1 KB	17/10/2019 22:36:34	rwxr-xr-x	goncalo
getFilterSensorData.php	1 KB	01/08/2019 18:22:55	rwxr-xr-x	goncalo
modifyThreshold.php	1 KB	01/08/2019 12:06:09	rwxr-xr-x	goncalo
getLastActuatorValues.php	1 KB	31/07/2019 13:15:51	rw-r--r--	goncalo
getAllSensorData.php	1 KB	19/07/2019 15:33:24	rwxr-xr-x	goncalo
login.php	1 KB	11/07/2019 11:50:09	rwxr-xr-x	goncalo
signup.php	1 KB	04/07/2019 16:25:11	rwxr-xr-x	goncalo
connectDB.php	1 KB	04/07/2019 16:01:54	rw-r--r--	goncalo

Figure 1.13 - PHP Scripts

- **connectDB.php** – This script is responsible to establish the connection to the database. All the php scripts that need to access to the database have to import this script;
- **signup.php** – Responsible to store the user's information in the database (username, email and password);
- **login.php** – Responsible to receive the username (or email) and password combination and verify if these values have a match. In case of the given values does not match the script retrieves an error code;
- **getLastSensorValues.php** – This script retrieves the last values collected for each sensor;
- **getLastActuatorValues.php** - This script retrieves the current values of each actuator;
- **getAllSensorData.php** – Responsible for retrieve the last 50 values of a specific sensor. The sensor specification is sent by the mobile application when the user accesses to the Sensor Details screen;
- **getFilerSensorData.php** – Responsible for retrieve the sensor values for a specified period for its representation in the mobile application;
- **modifyThresholds.php** – This script updates the current thresholds for a specific sensor.

1.2.3 Python Scripts

As described in section 1.2, there were developed two python script, one for each topic used for MQTT protocol:

- mqtt-server.py (for tese/goncalo/in) – This script is responsible for storing data collected from the sensors, process this information to send notifications to the mobile application users. It also updates the values of auto mode in the database;
- server-mqtt.py (for tese/goncalo/out) – This script takes decisions for the actuators actions when the automatic mode is activated.

1.3 Arduino IDE

The selected hardware for the system is compatible with Arduino IDE which can be downloaded at www.arduino.cc/en/main/software.

For the implementation of some components of the system, the following libraries were used:

- DallasTemperature;
- DFRobot_ESP_PH_BY_GREENPONIK-master;
- DHT sensor library;
- FlowMeter;
- OneWire;
- PubSubClient;
- RadioHead-master;
- StringSplitter.

In Table 1.1, are described the developed code files using Arduino IDE for each node of the network:

Table 1.1 - Arduino Files

File	Description
broker.ino	Forwarding messages between sensor/actuator nodes and the server or mobile application
compensation_tank.ino	Obtaining data through water temperature and pH sensors
water_flow.ino	Obtaining water flow data
swimming_pool.ino	Obtaining water level information
surroundings.ino	Reading humidity and air temperature, and luminance values
actuators.ino	Turn on or off actuators through received messages from broker

References

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Commun. Surv. Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[2] H. N. Saha, A. Mandal, and A. Sinha, "Recent trends in the Internet of Things," *2017 IEEE 7th Annu. Comput. Commun. Work. Conf. CCWC 2017*, pp. 15–18, 2017.

[3] J. M. Marais, D. V. Bhatt, G. P. Hancke, and T. D. Ramotsoela, "A web-based swimming pool information and management system," *IEEE Int. Conf. Ind. Informatics*, pp. 980–985, 2017.

[4] R. Gouws and A. Nieuwoudt, "Design and cost analysis of an automation system for swimming pools in South Africa," *(DUE), 2012 Proc. 20th*, no. 2, pp. 9–15, 2012.

[5] C. Bell, *Beginning Sensor Networks with Arduino and Raspberry Pi*, 1st editio., no. 4. 2013.

[6] D. Elmazi, M. Cuka, T. Oda, M. Ikeda, and L. Barolli, "Effect of node density on actor selection in wsans: A comparison study for two fuzzy-based systems," *Proc. - Int. Conf. Adv. Inf. Netw. Appl. AINA*, pp. 865–871, 2017.

[7] A. Gloria, F. Cercas, and N. Souto, "Comparison of communication protocols for low cost Internet of Things devices," *South-East Eur. Des. Autom. Comput. Eng. Comput. Networks Soc. Media Conf. SEEDA-CECNSM 2017*, 2017.

[8] K. Pothuganti and A. Chitneni, "A Comparative Study of Wireless Protocols :," *IECON Proc. (Industrial Electron. Conf.*, no. January 2014, pp. 46–51, 2014.

[9] A. Benlghazi, E. Chadli, and D. Moussaid, "Bluetooth technologie for industrial application," *2014 5th Int. Conf. Inf. Commun. Syst. ICICS 2014*, pp. 1–5, 2014.

[10] Z. M. Lin, C. H. Chang, N. K. Chou, and Y. H. Lin, "Bluetooth Low Energy (BLE) based blood pressure monitoring system," *Proc. 2014 Int. Conf. Intell. Green Build. Smart Grid, IGBSG 2014*, pp. 1–4, 2014.

[11] A. Hafeez, N. H. Kandil, B. Al-Omar, T. Landolsi, and A. R. Al-Ali, "Smart home area networks protocols within the smart grid context," *J. Commun.*, vol. 9, no. 9, pp. 665–671, 2014.

[12] A. Augustin, J. Yi, T. Clausen, and W. M. Townsley, "A study of Lora: Long range & low power networks for the internet of things," *Sensors (Switzerland)*, vol. 16, no. 9, 2016.

[13] LoRa Alliance, "A technical overview of LoRa ® and LoRaWAN ™ LoRaWAN ™ What is it?," no. November, 2015.

[14] R. Fielding *et al.*, "RFC 2616: Hypertext transfer protocol–HTTP/1.1, June 1999," *Status Stand. Track*, vol. 1, no. 11, pp. 1829–1841, 1999.

[15] P. Alqinsi, I. Joseph, and M. Edward, "IoT-Based UPS Monitoring System Using MQTT Protocols," *2018 4th Int. Conf. Wirel. Telemat.*, pp. 1–5, 2018.

[16] N. Tantitharanukul, K. Osathanunkul, K. Hantrakul, P. Pramokchon, and P. Khoenkaw, "MQTT-Topics Management System for sharing of Open Data," *2nd Jt. Int. Conf. Digit. Arts, Media Technol. 2017 Digit. Econ. Sustain. Growth, ICDAMT 2017*, pp. 62–65, 2017.

[17] S. Lee, H. Kim, D. K. Hong, and H. Ju, "Correlation analysis of MQTT loss and delay according to QoS level," *Int. Conf. Inf. Netw.*, pp. 714–717, 2013.

[18] Charlie Wang, "HTTP vs. MQTT: A tale of two IoT protocols," 2018. [Online].

Available: https://cloud.google.com/blog/products/iot-devices/http-vs-mqtt-a-tale-of-two-iot-protocols.
[19] A. A. Galadima, "Arduino as a learning tool," *Proc. 11th Int. Conf. Electron. Comput. Comput. ICECCO 2014*, pp. 1–4, 2014.
[20] A. Nayyar and E. V. Puri, "A Review of Arduino Board's, Liypad's & Arduino Shields," pp. 1485–1492, 2016.
[21] ESP8266 Datasheet, "ESP8266EX Datasheet," *Espr. Syst. Datasheet*, pp. 1–31, 2016.
[22] E. Systems, "ESP32 Datasheet," 2018.
[23] S. B. Biswas, "Solar Water Pumping System Control Using a Low Cost ESP32 Microcontroller," *2018 IEEE Can. Conf. Electr. Comput. Eng.*, pp. 1–5, 2018.
[24] Hope RF, "RFM69HW HopeRF Module," pp. 1–79.
[25] Adafruit, "RFM95_96_97_98W." [Online]. Available: https://cdn-learn.adafruit.com/assets/assets/000/031/659/original/RFM95_96_97_98W.pdf?1460518717. [Accessed: 04-Dec-2018].
[26] M. Integrated, "DS18B20 Datasheet." [Online]. Available: https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf. [Accessed: 07-Dec-2018].
[27] SparkFun, "Humidity and Temperature Sensor - DHT22." [Online]. Available: https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf. [Accessed: 07-Dec-2018].
[28] Chipdip, "G 1/2 Water Flow sensor." [Online]. Available: https://lib.chipdip.ru/583/DOC000583441.pdf. [Accessed: 10-Dec-2018].
[29] DFRobot, "PH meter(SKU: SEN0161)." [Online]. Available: https://www.dfrobot.com/wiki/index.php/PH_meter(SKU:_SEN0161). [Accessed: 10-Dec-2018].
[30] DFRobot, "Liquid Level Sensor-FS-IR02 SKU: SEN0205." [Online]. Available: https://www.dfrobot.com/wiki/index.php/Liquid_Level_Sensor-FS-IR02_SKU:_SEN0205. [Accessed: 10-Dec-2018].
[31] M. J. Cheng, S. W. Hung, H. H. Tsai, and P. W. Chen, "The adoption intentions of mobile applications," *2016 IEEE/ACIS 15th Int. Conf. Comput. Inf. Sci. ICIS 2016 - Proc.*, pp. 1–3, 2016.
[32] "HiveMQ." [Online]. Available: http://www.hivemq.com/demos/websocket-client/ [Accessed: 19-Feb-2018].
[33] "Piscinas Municipais GesLoures Santoantónio dos Cavaleiros." [Online] Available: https://gesloures.pt/santo-antonio-dos-cavaleiros/ [Accessed: 22-Sep-2019].