ISCTE IUL
Instituto Universitário de Lisboa

Department of Information Science and Technology

Cyclist performance assessment based on WSN and Cloud technologies

Ana Catarina Duque Dias

A Dissertation presented in partial fulfillment of the Requirements for the Degree of

Master in Telecommunications and Computer Engineering

Supervisor:

Dr. Octavian Adrian Postolache, Assistant Professor

ISCTE-IUL

October 2018

Resumo

A mobilidade nas grandes cidades é um problema crescente e a utilização das bicicletas tem vindo a ser uma solução que, em conjunto com novos serviços de partilha, ajudam a motivar os utilizadores. Há também cada vez mais utilizadores a praticar desportos que envolvem a utilização da bicicleta. Foi neste contexto que a presente dissertação foi desenvolvida, um sistema de sensores distribuídos para monitorização de ciclistas. Com o suporte de uma rede de sensores sem fios ligada á internet e, utilizando um conjunto de sensores inteligentes como nós, é possível obter dados que vão ajudar o ciclista a melhorar o seu desempenho. O treinador consegue monitorizar e avaliar o desempenho para aperfeiçoar as sessões de treino.

A condição do estado de saúde é também monitorizada utilizando sensores de avaliação cardíaca e de respiratória. A informação proveniente dos nós da rede de sensores sem fios é carregada, através da ligação á internet, para a plataforma *Firebase*.

Foi desenvolvida uma aplicação móvel *Android*, que permite que os treinadores registem ciclistas, planeiem rotas e observem os resultados recolhidos pela rede. Com a inclusão destas tecnologias, o treinador e o ciclista podem analisar o desempenho de uma sessão e compara-lo com os resultados do treino anterior. Podem ser estabelecidas novas sessões de treino de acordo com as necessidades do atleta.

A eficácia do sistema proposto foi testada experimentalmente e os vários resultados foram incluídos nesta dissertação.

Palavras-chave: Monitorização de Ciclistas; Rede de Sensores sem Fios, *Internet of Things*; *Firebase*; Aplicação *Android.*

Abstract

Mobility in big cities is a growing problem and the use of bicycles has been a solution which, together with new sharing services, helps to motivate users. There are also more and more users practicing sports involving the use of bicycles. It was in this context that the present dissertation was developed, a distributed sensor system for monitoring cyclists. With the support of a wireless sensor network connected to the internet and, using a set of smart sensors as end-nodes, it is possible to obtain data that will help the cyclist to improve his performance. The coach can monitor and evaluate the performance to improve their training sessions.

The health status condition during training it is also monitored using cardiac and respiratory assessment sensors. The information from the nodes of the wireless sensor network is uploaded, via the internet connection, to the Firebase platform.

An Android mobile application has been developed, this allows trainers to register cyclists, plan routes and observe the results collected by the network. With the inclusion of these technologies, the coach and the athlete may analyze the performance of a session and compare it with the previous training results. New training sessions may be established according to the athlete's needs.

The effectiveness of the proposed system was experimentally tested and several results are included in this dissertation.

Keywords: Cyclist Monitoring; Wireless Sensor Network; Internet of Things; Firebase; Android Application

Acknowledgments

Firstly, I would like to express my sincere thanks to my advisor, Professor Octavian Postolache, for all the availability and support during the realization of this project.

To the Telecommunications Institute at ISCTE-IUL, thanks for providing all the material and resources needed for this dissertation.

My biggest thanks goes to my family for the unconditional support, especially to my parents, Zélia and Francisco, and my sister, Beatriz.

A big thanks to my boyfriend, Filipe, for all the help, support and for having a patience the size of the world.

A special thanks to my long-time friend, Carolina Dionísio, for all the encouragement and for, in some way, helping me through this project.

Finally, a huge thanks to all my friends and colleagues who accompanied me on this journey, especially to André Marques who was my great summer companion, to André Gloria for all help provided and to Miriam Batista for the discussions of ideas that really helped me complete this project.

Content

List of Figures

List of Tables

List of Acronyms

4G	4th Generation
ADC	Analog to Digital Conversion
ADC	Analog-to-Digital Converter
AHRS	attitude and heading reference system
API	Application Programming Interface
BaaS	Backend as a Service
BLE	Bluetooth Low Energy
BPM	Beats per Minute
CSV	Comma Separated Values
DB	Database
DCM	Direct Cosine Matrix
ECG	Electrocardiography
FFD	Full-Function Device
GPS	Global Positioning System
GUI	Graphical User Interface
HTTP	Hypertext Transfer Protocol
I2C	Inter-Integrated Circuit
ICSP	In-Circuit Serial Programming
IDE	Integrated Development Environment
IMU	Inertial Measurement Unit
IOS	iPhone Operating System
IoT	Internet of Things
JSON	JavaScript Object Notation
LTE	Long Term Evolution
NoSQL	Not Only Structured Query Language
P2P	Peer-to-Peer
PCB	Printed Circuit Board
PPG	photoplethysmogram
RFD	Reduced Function Device
RFID	Radio Frequency Identification

RPM	Respiration per Minute
SCG	Seismocardiography
SD	Secure Digital
SDK	Software Development Kit
SNR	Signal-to-Noise Ratio
SoC	System on Chip
SPI	Serial Peripheral Interface
SQL	Structured Query Language
TTFF	Time-to-First Fix
UART	Universal Asynchronous Receiver Transmitter
URL	Uniform Resource Locator
USB	Universal Serial Bus
WBAN	Wireless Body Area Network
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network
WPAN	Wireless Personal Area Networks
WSN	Wireless sensor network
XML	eXtensible Markup Language

Chapter 1 Introduction

1.1 Motivation

Nowadays there is an increasing need to apply technologies to everything around us, whether to make our daily basis easier or just for leisure. The Internet of Things (IoT) is a technological concept, with the potential to alter and replace various methods of classical sports training and information acquirement. The advantage of performing measurements of physical parameters using internet-connected devices is that the results of these measurements are immediately available in electronic format. This allows the users to access the information anyplace, anytime and to perform the evaluation of reached performances.

Athletes, whether professionals or amateurs, always want to evolve in the activities they practice, so there is a need to use the new technologies so that cyclists and their coaches can evaluate the progress in an objective way. In a project that took place in Sweden, the movement of a skier was analyzed, with the help of Smartphones the skier was able to develop his strategy ability to go faster [1]. Today, most professional athletes are carrying some sort of device, be it a smartwatch, an embedded sensor in their clothing, or sports gear that captures their performance [2]. Our work follows this theme, IoT based sport and athlete analysis, particularly cycling.

This proposal aims to use various technologies to increase performance and improve the practice of cycling considering also the health condition of the athletes. One of these technologies is the Wireless Sensor Network (WSN) which uses different wireless sensors placed on the bicycle and the cyclist to collect information during training, these are then processed to produce statistics. Using a data information system, it is possible to save and manage the previously mentioned information/statistics and the routes made. It is also possible to keep all the information related to the previous and the scheduled training. All this, shown in a mobile application that can be accessed by the cyclist and his coach. Using historical data analysis appropriate decisions can be considered based on models for each cyclist.

1.2 Objectives

In this dissertation, it is proposed to extend the research and the development of the hardware and software of a Smart Bike, in which several sensors have been considered attached to the bicycle and to the biker's equipment. The goal is to develop a multimodal system for cycling assessment considering some previous work developed by the IM Group of IT-IUL. Thus, were considered new sensors dedicated to the cyclist's physical performance monitoring and it was created a new system to store, analyze and visualize the information. A mobile application was developed to present the information coming from the sensors. With access to all these information, coaches can set goals for their workouts, refine their tactics and motivate cyclists to improve their performance [3]. The purpose of the system is to facilitate and improve the cyclist's training, however, it can be easily adapted to another type of bike users.

This system aims to introduce new metrics to evaluate the performance of the cyclist and improve their performance in this sport, complementing the traditional training of a cyclist.

The results obtained from the use of the sensors are kept in a data storage system. Then they are treated statistically to produce metrics that will be used by the athlete and his coach to evaluate the evolution after each training and to propose different types of data representation.

The resulting information can be visualized using a mobile application that can be accessed by users such as the cyclist and his coach.

1.3 Structure of the dissertation

Chapter 2 includes the literature review on Internet of Things, Wireless Sensor Networks and how these concepts can be used to monitor sports. Chapter 3 describes the developed system as a whole and how everything is connected. The embedded system of this project is described in Chapter 4. Chapter 5 includes the description of the database and the developed mobile application that helps the coaches analyzing the data. In Chapter 6 are the experimental results and its evaluation. The conclusions and future work are presented in Chapter 7. This dissertation includes also a User manual and a Technical manual, the first one describes how the system can be used and the second one shows the functionalities of the system.

Chapter 2 State of the art

This section presents the research area to develop the proposed work, such as the Internet of Things and Wireless Sensor Networks. It also presents some sports and health monitoring applications, smart sensors communication protocol, embedded systems suitable for this work, data storage solutions and the existing software development solutions for mobile applications.

2.1 Internet of Things

The Internet has drastically changed the way we live, by moving interactions between people to a virtual level in various contexts, ranging from professional life to social relations, IoT represents the next evolution of the internet [4] [5]. The internet takes a huge step forward in its ability to collect, analyze and distribute data. The number of objects connected to the Internet has been increasing [6], which makes this innovative technological concept rapidly entering people's lives.

IoT enables everyday objects to gather information around them, perform tasks, for example, detect there is nobody in the room a turn off the lights. These objects, communicate with each other, share information and coordinate decisions, exploring their adjacent technologies such as communication technologies or sensor networks. Its potential allows the development of a large number of applications, giving a new dimension allowing communication with and between intelligent objects, which leads to the vision of communications "anytime and anywhere" [5].

Internet of Things

Figure 2.1 - The overall picture of IoT

2.1.1 Internet of Things for Sports

The Internet of Things for Sports is a platform to analyze data from sensors that aims to evaluate the performance of athletes, amateurs or professionals. Sports and recreational activities are one of the fastest growing areas for consumer-oriented applications, but there are few manufacturers who develop smart sports products. This field of research lags far behind than other sectors in the area of IoT [1] [7].

With the technology currently available, athletes can get real-time results on their rhythm or movement, so they can intensify their physical activity with objective data and critical analysis of their workouts in order to increase their performance. The connected athlete uses sensors, for example, in shoes or other non-intrusive places, so that it does not interfere with his activity [8].

In Figure 2.2, is presented the ring structure around IoT Sports, this ring has four fundamental concepts: Interaction, Things, Process and Data [1].

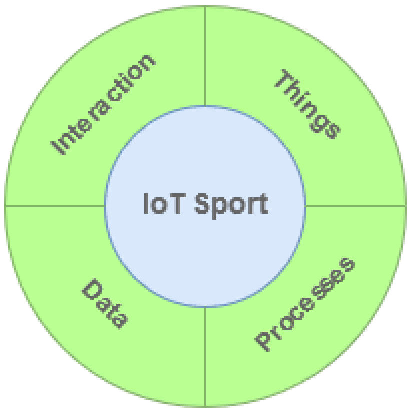


Figure 2.2 - ITPD ring around IoT Sports

Interaction represents the contact between the athlete and the sensor, for example, a force sensor on the athlete’s shoes that measure the force applied every time he jumps.

Things are the objects that can connect to the internet, in these case is the sensors that are going to make the measurements.

Processes mainly involve accumulating, communicating and analyzing collected data.

Data is collected from the sensors, it can be analyzed in real time or it can be stored in servers or in a Cloud to be processed later.

This concept was developed to automate the process of collecting information from sports training and to allow sensors, microcontrollers, cloud, athletes, trainers, medical terms to connect as a whole.

2.2 Wireless Sensor Network

A WSN includes sensing devices spatially distributed, typically used to monitor some type of event such as sound, temperature, pressure, vibration or motion [9]. These networks are composed of wireless nodes that have sensors [10]. In the past, these networks were generally wired. Wireless sensor networks consist, essentially, on low-power, light-weight sensor nodes [11].

Due to their fast development, WSNs are being used extensively in the scientific and technological fields. Several studies on medical, commercial and industrial [11] applications show successful results, increasing the interest of new users in this type of monitoring solutions. Nowadays, the use of these networks to collect and interpret data is facilitated.

As shown in Figure 2.3, some of the most common topologies found in WSNs are: *Peer-to-peer* (P2P, also known as mesh), *Star* and *Tree* [12].

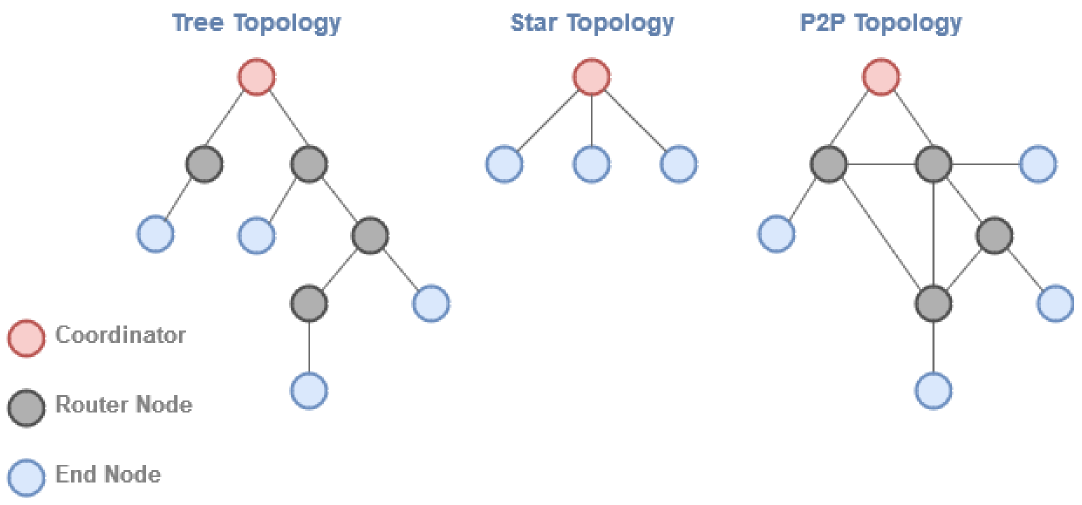


Figure 2.3 - WSN's most common topologies

In *Star* topology, usually, the nodes are within a hop distance of the coordinator, it sends and receives the messages to and from the nodes, the end-nodes do not communicate with each other. This topology has a lower power consumption than the others. ZigBee and Bluetooth are two examples of communication protocols that support star topology [13].

In both *Tree* and *P2P* topologies, multi-hops communications take place. In *Tree* topology, each node connects to the one that is higher up the tree, and finally to the coordinator. *Mesh* allows the transmission of data between the various nodes, as long as they are within the range, otherwise, the information is sent through intermediate nodes.

The great advantage of these two topologies is the easy error detection, on the other hand, the *Mesh* network becomes very large and expensive.

2.2.1 Wireless Body Area Network

A Wireless Body Area Network (WBAN) is a type of WSN where wireless sensor nodes are placed on the human body [14]. These sensors are set in different parts of the body and can be wearable or placed under the user's skin, each of which has specific requirements and is used for different purposes. The network communicates through, for example, Bluetooth Low Energy (BLE) protocol, with a coordinating node that is responsible for sending the data, in order to provide real-time monitoring [15].

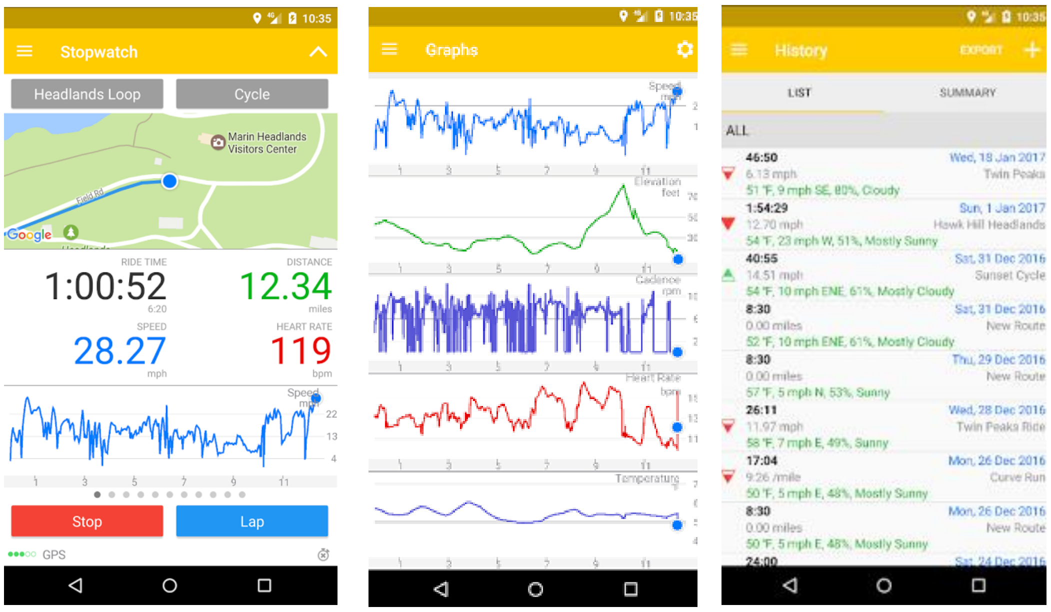
This type of network can be used to develop patient monitoring systems that offer greater flexibility to the medical team or to monitor the performance of athletes and assist them in their sporting activities [16].

2.3 Sensors and sports activities monitoring

Cycling is a very complete physical activity that involves a set of movements. These movements can be analyzed using several free applications, smart wristbands or through devices placed on the bike and on the cyclist. It is very important that these devices do not interfere with the athlete’s normal activity.

2.3.1 Mobile applications for sports

There are many sports applications on the market, such as Cyclemeter Cycling Running GPS which is considered “*the most advanced application for cyclists ever designed for a mobile device*” [17]. It is possible to register routes, speed and schedule training sessions. It is an application available on iPhone and Android devices. The application is quite limited because it only collects information through the available resources in the mobile phone, such as Global Positioning System (GPS). These type of applications cannot measure important factors such as the heart rate. In the following figure (Figure 2.4) [18], it is an example of this application.



46:50 — 6.13 mph — 51 °F, 9 mph SE, 80%, Cloudy — Wed, 18 Jan 2017 — Twin Peaks
1:54:29 — 12.70 mph — 54 °F, 23 mph W, 51%, Mostly Sunny — Sun, 1 Jan 2017 — Hawk Hill Headlands
40:55 — 14.51 mph — 54 °F, 10 mph ENE, 61%, Mostly Cloudy — Sat, 31 Dec 2016 — Sunset Cycle
8:30 — 0.00 miles — 52 °F, 10 mph ENE, 61%, Mostly Cloudy — Sat, 31 Dec 2016 — New Route
8:30 — 0.00 miles — 57 °F, 5 mph N, 53%, Sunny — Thu, 29 Dec 2016 — New Route
26:11 — 11.97 mph — 58 °F, 7 mph E, 49%, Sunny — Wed, 28 Dec 2016 — Twin Peaks Ride
17:04 — 9:26 /mile — 50 °F, 5 mph E, 48%, Mostly Sunny — Mon, 26 Dec 2016 — Curve Run
8:30 — 0.00 miles — 50 °F, 5 mph E, 48%, Mostly Sunny — Mon, 26 Dec 2016 — New Route
24:00 — Sat, 24 Dec 2016

Figure 2.4 - Cyclemeter Cycling Running GPS Application

Sports Tracker [19] is also a sports recording application, it is suitable for various sports, including cycling. As Cyclemeter Cycling Running GPS, the Sports Tracker is limited to the features of the mobile phone but has available accessories to complete the system, for example, heart rate sensors. This application is available on Android and iOS (iPhone Operating System) devices.

Strava [20] and Endomondo [21] are applications that promote the motivation and sports fun, both can be used not only in the smartphone but also in the computer. These are used to monitor physical activity, including cycling, they record information, collected through the modules available in the mobile phone, produce statistics with this information. In Endomondo application, there is still the possibility of having feedback through audio. Both work as social network of athletes, it is possible to see the training of virtual friends, comment, for example with motivational messages, and it is still possible to challenge other athletes or participate in challenges. *The Strava feed is full of inspiring activities, crazy adventures and interesting itineraries* [20]. Both applications are compatible with smartwatches which makes it possible to monitor heart rate. The Mountain Bike Runtastic [22] application is identical to the previous ones, but it also includes offline maps and the wind and weather conditions, which allows for a more detailed analysis. In Figure 2.5 [23], it is possible to see the layout of the application The Mountain Bike Runtastic.

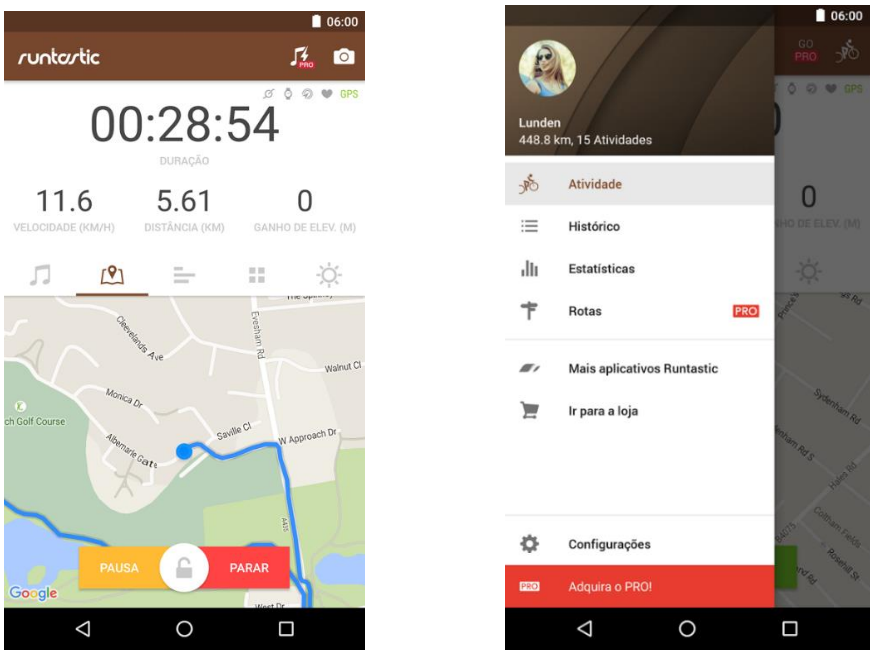


Figure 2.5 - Mountain Bike Runtastic Application

2.3.2 Sports monitorization

Coaches and athletes constantly strive to find effective ways to improve their sports performance. J. P. Broker and J. D. Crawley [24], provide a variety of examples of data acquisition, analysis, and feedback systems in an Olympic training environment. An instrumented boxing bag has been developed, containing accelerometers to measure the movements of the bag in response to the boxer's punches. This information was combined with the video-overlay process to be analyzed. Another example is the measurement of the mechanics of the pedals during the cycling activity, this activity was performed in the laboratory and aims to improve the pedaling technique. The bicycle pedals have two force transducers that measure applied forces. The specialized software provides real-time feedback so that cyclists have the opportunity to practice technique modifications and observe the results as they do so.

T. Ribeiro [25], presented a distributed system of sensors expressed by a WSN to evaluate and improve the performance of cyclists. This system implements the ZigBee communication protocol to perform data acquisition, processing and communication. All collected data is stored in a cloud platform and can be accessed through a mobile application. The hardware involved in the WSN is responsible for the acquisition, processing and sending of data to the server. So, in the WSN are the force sensors, Inertial Measurement Unit (IMU) boards and microcontrollers. The force sensors are placed on the shoes and gloves, the IMU board is attached to the cyclist's chest and it is used to measure the oscillations in the three plans of movement (yaw, pitch, roll). These sensors with the Arduino Fio, work as end-nodes and are responsible for data acquisition, the Arduino Mega works as coordinator and network center (star topology), all of these microcontrollers have a Xbee module coupled. On the coordinator is also a GPS shield and a Wi-Fi shield that will communicate with the Cloud.

An Android application was also developed to help cyclists and coaches visualize training results and to extract relations between measured values. The difference of this project for the present dissertation, besides detecting the interactions of the cyclist with his bicycle, is also to monitor his heart rate and respiration rate, use different types of representation of the data. A new Database (DB) system was implemented and a new mobile application was developed to support all the new types of information.

2.3.3 Health monitoring for sports

Medicine and sports science can be considered as two linked fields of research. Medicine is the science of diagnosing or monitoring the body for diseases and trying to get the patient's body back to normal. Sports science pulls the athlete to the limit and therefore, athletes should be monitored so that they do not reach the level that can lead to serious injuries [26]. Monitoring athletes by reading vital signs, and evaluating other signs like the force exerted during training is very important, as it allows to evaluate their performance, avoiding overtraining [11] and serious injuries. This monitoring can also be used during injury recovery [27].

In [28] the authors presented a wireless Electrocardiography (ECG) system to monitor athletes during their physical activity to prevent stroke or heart failure using dry-contact electrodes and ANT+ wireless technology. The experiments were performed in the laboratory using a compression vest, the electrodes were placed on the right shoulder and below the apex of the heart. Several tests were carried out: with the electrodes on the skin and over a cotton T-shirt while the subject was standing, walking, jumping and running. It was shown that it is possible to get signs on the upper part of the skin or through clothing. The authors concluded that the proposed system is a potential solution for future mobile health and sport monitoring applications.

2.3.4 Heart Rate and Respiration Rate measurement

As mentioned in the previous section it is very important to measure the vital signs of an athlete during his training session. Measuring heart rate is the simplest and cheapest way to determine the intensity of training so that athletes remain within the normal limits, either during the period of training or rest [28].

Measuring the rate of breathing is another very important factor, since changes detected in the respiratory rate can prevent serious diseases such as cardiac arrest. This means that the respiration rate is a very important indicator of a person's state of health.

According to the authors in [29], there are two different approaches to make these measurements: non-contact and contact. Contact is the most appropriated to this project because it is an outdoor and in moving sport and it becomes hard to monitor the athlete with non-contact methods, like temperature cameras.

By measuring the heart rate, respiration rate and combining them with the previous work will help coaches to give feedback to cyclists and help them improve their performance, avoiding excessive fatigue [30]. This is the reason why both these measurements are considered for this project.

Seismocardiography (SCG) is a non-evasive method to measure the vibrations of the human body caused by the heartbeat, this signal contains information regarding the cardiovascular and respiratory systems [31]. The Seismocardiography sensor is a high-sensitive triple-axis accelerometer. This technique allows to measure the breathing and heart rate but is not the most indicated for the heart rate when the subject is in movement [32]. This method is appropriated for when the subject is very still, when he’s sleeping for example. In this case, it would be really hard to distinguish when the subject is breathing from when the subject is moving.

2.4 Wireless Communication Protocols

As mention before one of the main components of IoT system is communication. There are several wireless and wired communication protocols, but since it is a wireless sensor network it is mandatory to use wireless protocols. The most common protocols used in short-range wireless networks communications are the IEEE 802.11 (Wi-Fi), IEEE 802.15.1 (Bluetooth) and IEEE 802.15.4 (ZigBee) standards. In Table 2.1 are presented the main characteristics of each of these protocols.

Table 2.1 - Main Wireless communication protocols characteristics [33]–[36]

Feature	Wi-Fi	Bluetooth	ZigBee
IEEE standard	IEEE 802.11	IEEE 802.15.1	IEEE 802.15.4
Max Signal Rate	54 Mbps	1 Mbps	250 kbps
Frequency	2.4 GHz; 5 GHz	2.4 GHz	2.4 GHz
Range	250 m	10 m	10 - 100 m
Nodes	Unlimited (ad hoc); 2007 (infrastructure)	7	> 65000
Typical Power Consumption	100 - 350 mA	1 - 35 mA	1 - 10 mA
Complexity	High	Medium	Low

2.4.1 IEEE 802.11 – Wi-Fi

Wireless Fidelity (Wi-Fi) includes the IEEE 802.11 standard, provides a wireless connection to devices within a Wireless Local Area Network (WLAN). Wi-Fi essentially uses an infrastructure network, which also supports ad-hoc networks in infrastructure mode. It allows users to browse the internet at broadband speeds when connected to an access point or are in ad-hoc mode [35], it also allows fast data transfer and can handle large amounts of data. Wi-Fi is the ideal protocol for a project that requires a quick connection between the device and the internet.

The Wi-Fi range depends on the version of Wi-Fi that the device is running, newer versions have more range than older ones, and physical obstacles in open spaces will have more range than indoors with walls or other interfering objects [33]. For example, the IEEE 802.11a has a range of 120 meters while the IEEE 802.11n is 250 meters.

In Table 2.2 are presented the advantages and disadvantages of the Wi-Fi protocol.

Table 2.2 - Advantages and disadvantages of the Wi-Fi protocol

Advantages	Disadvantages
- Decent coverage and outreach and can penetrate walls and other obstacles on the way - Adding or removing devices from a Wi-Fi network is a simple process	- High energy consumption - Radio waves in the network may interfere with other equipment

2.4.2 IEEE 802.15.1 – Bluetooth

Bluetooth, standard IEEE 802.15.1, is a network specification of the Wireless Personal Area Networks (WPAN) and uses the 2.4 GHz frequency which means they can operate on devices anywhere in the world. It is based on cheap and short-range wireless radio systems designed to replace some computer peripherals such as mouse, headphones, keyboards and printers [35]. Bluetooth devices can also be used for communications between portable computers, act as bridges between other networks.

This protocol not only defines a radio interface but a whole communication stack that allows the devices to find each other. A Bluetooth device can operate as a master or as a slave, up to 7 nodes plus the master and uses the star typology (described in section 2.2). Slaves only communicate with their master in a point-to-point way, under the master's control. In order to reduce power consumption, the slaves can switch from active mode to parked or standby mode [35], [36].

There is also the Bluetooth Low Energy that is designed specifically for low-power devices, which makes it a good communication protocol for the IoT area. However, BLE is not designed to transfer files but small chunks of data [34]. The reduced number of supported nodes made this protocol limited to applications that might need a greater number of nodes.

In Table 2.3 it is possible to see the main advantages and disadvantages of the Bluetooth protocol.

Table 2.3 - Advantages and disadvantages of the Bluetooth protocol

Advantages	Disadvantages
- One coordinator can control many slaves - Widely supported	- Only supports star topology - Very limited number of nodes

2.4.3 IEEE 802.15.4 – ZigBee

ZigBee over IEEE 802.15.4 standard is a technology that sets specifications for WPAN to support low power devices. This communication protocol provides self-organized, multi-hop and reliable networks with long battery life [35].

There are two types of devices that participate in ZigBee networks: the Full-Function Device (FFD) and the Reduced Function Device (RFD). FFD can communicate with RFDs and other FFDs but the RFD can only communicate with an FFD. The FFD can operate in three modes: as a network coordinator, as a router and as end-node. There is only one coordinator in the ZigBee network, it controls the network, delegates the network's functions devices, stores the security keys and makes bridges to other networks. The router works as an intermediary or to transmit data within the network, there may be several within the same network. The end-node cannot communicate directly with other end-nodes, it can only talk to the parent nodes (coordinator or router). These devices are designed to spend most of the time in suspension and wake up to transmit the data to the parent nodes. This type of network can have multiple end-nodes [35] [13].

Table 2.4 presents the advantages and disadvantages of the ZigBee protocol.

Table 2.4 - Advantages and disadvantages of the ZigBee protocol

Advantages	Disadvantages
- Low power - Supports star, tree and P2P topologies - Supports many slaves - One coordinator can control many slaves	- Requires additional equipment - Is incompatible with other network protocols

2.4.4 Remarks

Since it is not possible to perform tests with all the protocols presented, it was considered the description of the protocol, the advantages, disadvantages and the main differences between the protocols. Bluetooth can connect up to seven devices using the same frequency and has a range of only 10 m, which makes this technology a bit limited and not appropriate for this work. Wi-Fi is also not appropriate because it was designed for long connections and with a powerful power source, this protocol has a higher power consumption than Bluetooth and ZigBee. Considering that the proposed project encompasses a Wireless Sensor Network, it uses a limited power source, so it is necessary to use a low power protocol. Thus, the communication between the end-nodes and the coordinator will be made through the ZigBee protocol, IEEE 802.15.4 Standard.

2.5 Hardware components

The proposed system is based on a microcontroller, the chosen solution was considered after evaluation of different possible solutions according to the system requirements: signal acquisition, primary processing, data storage and data communication. These devices are used to build embedded systems and electronics projects. There are several boards to do this type of projects such as Raspberry Pi or BeagleBone microcomputers, but the most commonly used is Arduino [3], [11], [25], [37], since it is low-cost, has many libraries available and a really good online support.

2.5.1 Arduino platform

Arduino [38] is an open-source platform based on easy-to-use hardware and software, it is capable of signal acquisition, from sensors, digital I/O. Arduino is used in thousands of projects from simple and common things to complex scientific systems. Arduino board have an Atmel microcontroller that allows to program it and integrate with other circuits.

All boards are open-source enabling users to build them according to their needs. There are several good reasons to use Arduino:

Inexpensive: these boards are relatively inexpensive compared to other microcontroller platforms;

Cross-platform: unlike Arduino Software, most microcontroller systems are limited to Windows;

Simple and clear programming environment: the software is simple to use for beginners, but is flexible for more advanced users;

Open-source and extensible software: Projects can be expanded through the many C ++ libraries available online.

Within the Arduino, there are multiple choices to make this project. The choice of model falls on the dimensions, weight and price, although the Arduino being a low-cost component price between the various models differ. Table 2.5 shows the available Arduino models as well as the mentioned features.

Table 2.5 - Comparison between different Arduino's types *[39]*

Name	Analog In/Out	Digital IO/PWM	Dimensions	Weight	USB	Price
101	6/0	14/4	68.6 x 53.4 mm	34 gr.	Yes	28.65 €
Gemma	1/0	3/2	27.94 mm	2 gr.	Yes	Discontinued
LilyPad	6/0	14/6	50 mm	No Info	No	17.95€
LilyPad SimpleSnap	4/0	9/4	50 mm	No Info	No	17.95€
LilyPad USB	4/0	9/4	50mm	No Info	Yes	21.95€
Mega 2560	16/0	54/15	101.52 x 53.3 mm	37 gr.	Yes	35 €
Micro	12/0	20/7	48 x 18 mm	13 gr.	Yes	18 €
MKR1000	7/1	8/4	61.5 x 25 mm	32 gr.	Yes	30.99€
Pro	6/0	14/6	53.34 x 52.08mm	No Info	No	Discontinued
Pro Mini	6/0	14/6	18 x 33mm	2 gr.	No	Discontinued
Uno	6/0	14/6	68.6 x 53.4 mm	25 gr.	Yes	20 €
Zero	6/1	14/10	68 x 53 mm	12 gr.	Yes	39 €
Due	12/2	54/12	101.52 x 53.3 mm	36 gr.	Yes	34 €
Ethernet	6/0	14/4	68.6 x 53.3 mm	28 gr.	Yes	39.90 €
Leonardo	12/0	20/7	68.6 x 53.3 mm	20 gr.	Yes	18 €
Mega ADK	16/0	54/15	101.52 x 53.3 mm	36 gr.	Yes	43 €
Mini	8/0	14/6	30 x 18 mm	No Info	No	14€
Fio	8/0	14/6	28 x 65 mm	9g	No	21€
Nano	8/0	14/6	18 x 45 mm	7g	Yes	20€

2.6 Information Storage

For this work there is a need to store the information in a database, allowing a large amount of structured data to be collected. The local databases are on the device itself and the remote databases are, for example, on a server.

The main disadvantage of the first one is that it will use a lot of space on the device and the fact that the information is accessible only on the device itself, contrasting with the remote, which is best suited for this type of project because it can be accessed simultaneously by multiple users on multiple devices, but require an internet connection.

There are several software to develop databases, MySQL [40] is the most used, has customers such as Facebook, Google, Cisco Systems, among other world-renowned companies. The features that make it one of the most popular in the world are: portability, compatibility with various programming languages, excellent performance, low requirement for hardware resources.

The database must have some kind of security that ensures the privacy and integrity of the data since these DBs sometimes contain sensitive data from both users and companies.

With the development of the Internet and Cloud computing, it is necessary to have databases capable of storing and processing large amounts of data.

To solve these problems, new ways of data storage appeared, different from relational databases, which are called the NoSQL (Not Only Structured Query Language) database.

The main advantages of a NoSQL database are:

- Read and write data quickly;
- Support mass storage;
- Expandable;
- Low cost.

This type of information storage is different from relational databases, data from NoSQL databases are stored in a document in JavaScript Object Notation (JSON) or eXtensible Markup Language (XML) format and work with Key-Value (Figure 2.6).

When exchanging data with a server, the data can only be text. JSON files are text written in JavaScript object notation, which is why it is used as the syntax for storing and exchange data [41].

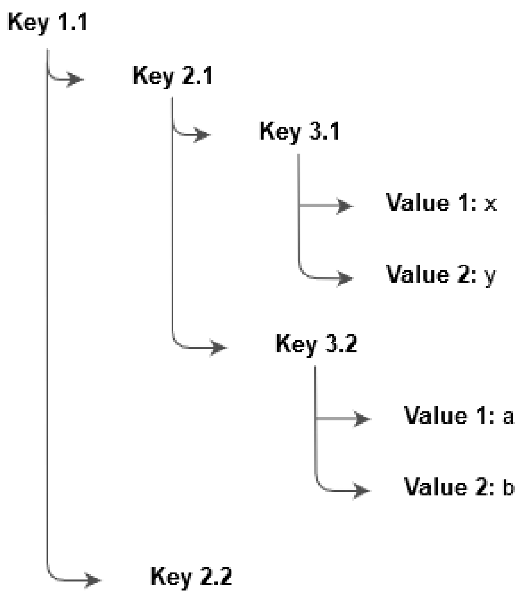


Figure 2.6 - NoSQL database structure

The database used in this dissertation is the Firebase Realtime Database which consists of a non-relational database that is in the Cloud provided by Firebase (which belongs to Google) [42], [43]. This platform does not require server-side code to access data and it is possible to add fields anytime without having to define them in the database model.

Firebase is a BaaS (Backend as a Service) platform that allows to develop Web and Mobile applications (Android and iOS) through Applications Programming Interfaces (API) and Software Development Kits (SDK), in which the entire backend is configured and managed by Firebase [43]. The Firebase SDK has functions that are listeners of the database, which makes Firebase real time. For example, when using it in a mobile application, every time anything changes in the database the methods are listening and immediately update the application's interface with the new data.

This system works with a JSON tree where the data is stored in nodes, which makes the search of the data easier. The Firebase Realtime Database already has an automatic synchronization system implemented, every time the data is changed all the devices receive the updates in a very short time [44].

2.7 Mobile Application

To accomplish this work, it is necessary to develop an application where it is possible to visualize the obtained data. To achieve this there were several options: computer application, mobile application or a web application. The decision was a mobile application, since the coach when accompanying the cyclist, will be constantly on outdoors and with a mobile application, he can follow the training in the field.

Android is an operating system developed by Google, based on Linux, opensource, designed for mobile devices such as smartphones and tablets. Android is the most widely used operating system, about 77% market share, globally, and about 75% in Portugal in August 2018, and it has been for at least the past year [45], [46] (Figure 2.7). These are the reasons why it is used native Android in this project and besides that, developing a mobile application for iOS requires a bigger investment in an Apple computer.

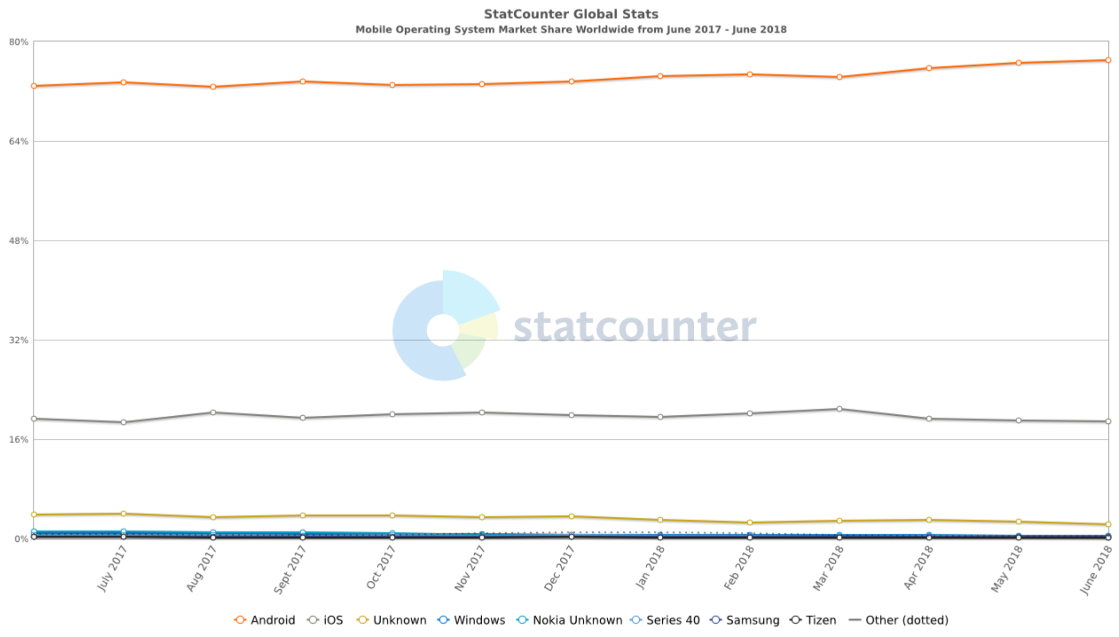


Figure 2.7 - Mobile Operating System Market Share Portugal

Android Studio is the Integrated Development Environment (IDE) for Android application development, is based on IntelliJ IDEA and is free to use [47]. Being an opensource Software and having many people to use, it has created a great community of help between developers. For these reasons, more and more people and companies are using this type of platform to develop applications.

Chapter 3 System Description

3.1 Overview

This work intends to develop a cyclist training analysis system, using multiple sensors, such as force and heart rate monitor sensors are considered, and various technologies to increase performance and improve the practice of cycling considering the health condition of the athletes.

Each training collects different data through sensors placed on the body of the cyclist and his bicycle. Both the coach and the cyclist have access to the results obtained in order to correct and improve each training session and, consequently, the performance of the athlete. Figure 3.1 illustrates the system architecture, which is divided into three parts.

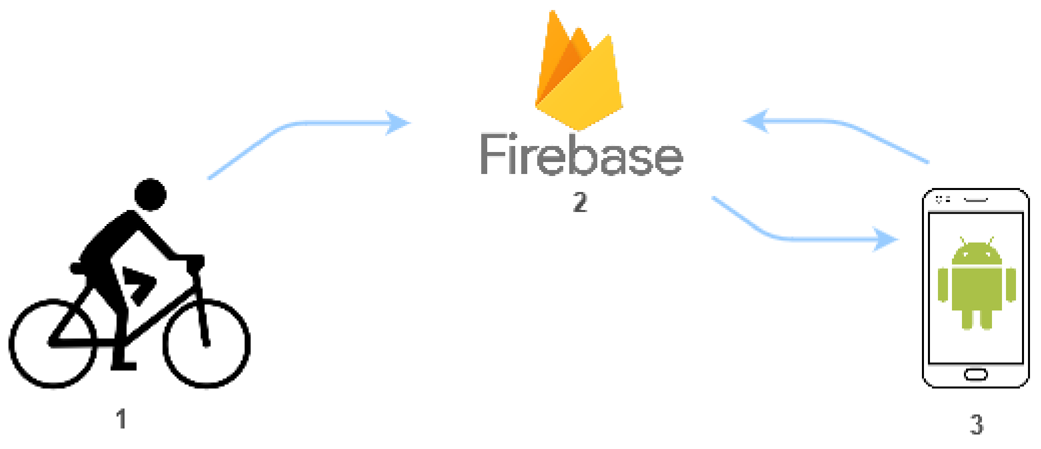


Figure 3.1 - System Architecture, 1-sensing, 2-storage, 3–data analyses and visualization

The first part (1) is dedicated to the training and data collection, in which the cyclist motion, the implied forces, the heart rate and the respiration rate are measured by the sensors during the training session. The second part (2) is represented by the storage of the data collected by the sensors, the settings and application data. The technology used for data storage is Firebase, developed by Google. The Firebase proved to be a very flexible, scalable and easy-to-use NoSQL database tool [42]. The last part (3) is dedicated to the mobile application associated with data analysis and data visualization. Using the developed mobile app every coach and cyclist may visualize the results associated with the performed exercises.

3.2 Users and Applications

The system allows different types of users: Administrator, Coach and Cyclist, any of the three can use the mobile application after they logged in, with a user and password. The administrator is on top of any user, he can create, edit or remove user accounts and also see the information of any coach or cyclist.

To accomplish the complete system, it was developed an embedded system to collect data from sensors and a mobile application to visualize the results, both of them communicate with the database via an internet connection:

Embedded System: expressed by several Atmel microcontrollers computation platforms (Arduino Mega and Arduino Fio), the microcontrollers were programmed to collect information from the sensors and send it to the Database.

Mobile Application: developed using Android Studio, it is used as a work tool for the coach but can also be used by athletes to consult their results. The Administrator uses the application to add new coaches and cyclists.

The main stakeholders in this system are:

- **Cyclist:** performs the training and can consult the application with the results and analysis done by his coach. Each cyclist has one and only one coach;
- **Coach:** Each coach may have one or more cyclists. This user supervises, plans, schedules and analyzes the training made, only of the cyclists that are associated with him. He also has permissions to create new cyclists through the application.

3.3 The LeadRide System

Figure 3.2 illustrates the flow of the developed system, the letters A-H represent each execution step. The first step is to register the information of coaches and cyclists by the administrator using the mobile application.

Coaches, if duly registered by the Administrator, may also register cyclists through the application. In the cyclist’s registration is requested a Radio Frequency Identification (RFID) tag so that the athlete can be identified when performing a workout. The tag can be found on the Secure Digital (SD) card that is in the coordinator. Each cyclist must have a tag used exclusively by him. The database also has a list of the tags, each tag is associated with a flag that allows to know if the tag is already used by any cyclist. In the registration form, only the unused tags appear for the coach or admin to choose. If it is the admin to register a cyclist, he has a spinner in the application to choose the coach of the new cyclist.

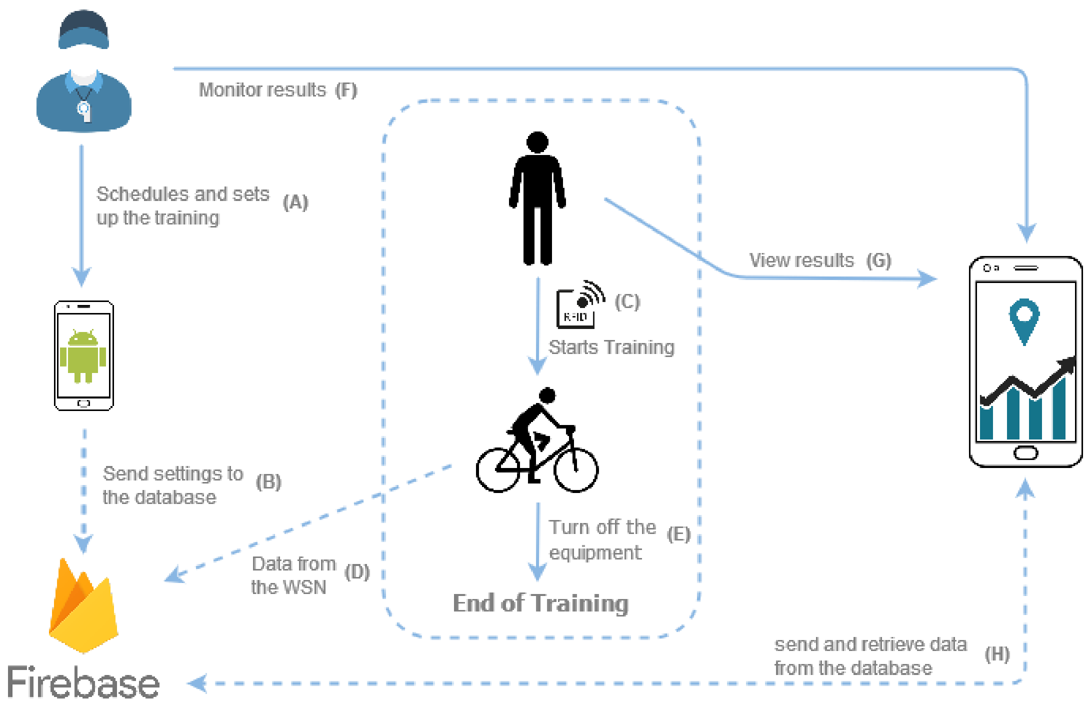

Monitor results (F)
Schedules and sets up the training (A)
Send settings to the database (B)
RFID
(C)
Starts Training
View results (G)
Data from the WSN (D)
Turn off the equipment (E)
End of Training
send and retrieve data from the database (H)
Firebase

Figure 3.2 - System workflow

The coach can schedule and set up the training session (A) with parameters such as:

- **Date**: day and time of the training;
- **Route**: giving two or three points in the map, the application draws a route.

When the coach wants to schedule a training section in the application, first he chooses a route then, before saving it in the database, he gives a date and time.

After submission of the training settings, these settings are sent and stored in the database (B). Once scheduled, the cyclist begins training by touching with his RFID card on the RFID Reader (Figure 3.3 [48]) (C), this card makes it possible to identify the athlete who will perform the training. Each cyclist will have an RFID tag with a unique number, as mentioned before, that will work as an identifier when he starts the training session. The tag that the cyclist passes when he starts the training will match with his registration in the database, so when he wants to see the results through the application the connection between the user and the training data can be made.

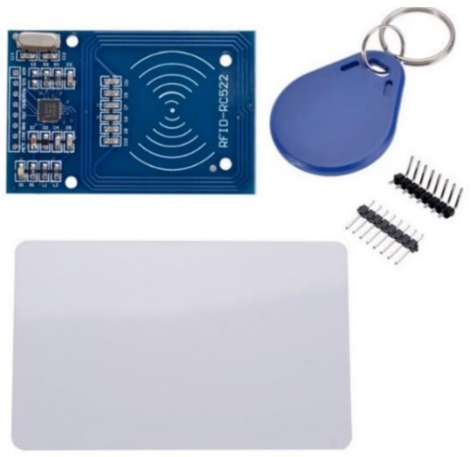


Figure 3.3 - RFID reader

The RFID tag will also trigger the data collection, the WSN only starts sending training data to the database (D) after the cyclist identification process is completed. To send data from the Arduino to Firebase it is used REST API with cURL command [44].

This system directly collects information about:

- The applied forces by the hands and feet;
- The angles in yaw, pitch and roll of the cyclist and the bicycle (using IMU);
- Heart rate and breathing rate;
- Coordinates along the training course.

Then it is possible to extract other information from this collected data, for example, the velocity, calories burned and the usage of each force sensor.

When the equipment is turned off (E) it means that the training is finished, the network stops sending information to the database.

When the biker completes, at least, one training session, the trainer can monitor the results (F) and leave some feedback, if necessary. The athlete can also view the results of training (G) and feedback left by his coach. The application accesses Firebase to get the correspondent data and to save the comments that the coach made (H). The coach and cyclist can only access the data through the username and password system. Both coach and cyclist can export the data to their smartphone or tablet, the mobile application has an option the export all the data from the charts to a Comma Speared Value (CSV) file. To exchange information between the Android Application and the database, it is used the Firebase SDK, which makes it very easy to use [44].

3.4 Main Hardware Components

3.4.1 Arduino Mega

The Arduino Mega (Figure 3.4 [49]) works as the coordinator of the WSN, it is a computation platform that receives the data collected from the end-nodes of the Zigbee Network. A RFID reader is connected to the Arduino Mega that identifies and authenticates the cyclists, collects the GPS coordinates of the route and also has a wi-fi shield with the Linux operating system. This allows to send the information of the sensors to the Firebase through the command line.

This Arduino is based on the ATmega2560 and it was chosen for coordinator because of the number of pins it has, as seen in section 2.5.1. These pins will be used to connect all the components that are in the coordinator (RFID reader, Yun shield and GPS shield). The Yun shield is a very important part of the system, this shield can be connected to a Wi-Fi network and, through that, connected to the database.

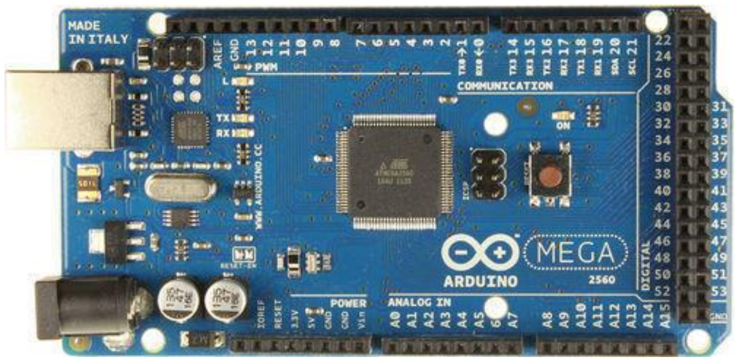


Figure 3.4 - Arduino Mega

3.4.2 Arduino Fio

This component works as an end-node of the ZigBee network (star topology). The Arduino Fio (Figure 3.5 [49]) platform presents one or more measurement channels associated to the sensors. This Arduino sends, through the ZigBee network, the collected information structured so that it can be read more easily by the coordinator.

This Arduino is one of the smallest, as discussed in Section 2.5.1, and the size matters a lot since the nodes cannot interfere with the cyclist's normal training, so this was the component chosen for the end-nodes. The Arduino Fio is a microcontroller board based on the ATmega328P and has an XBee socket built into the bottom of the board and, according to the official Arduino website, this board is intended for wireless applications [39]. This Arduino runs at 3.3V, it has connections for a Lithium Polymer battery and includes a charge circuit over USB.

Figure 3.5 - Arduino Fio

3.4.3 Android Smartphone or Tablet

In order to use the application, it is necessary to have a device with the Android operating system. In the developed application it is possible to visualize the map of the route carried out and charts resulting from the information collection of the sensors. Three different devices were used to develop and test the application: BQ Aquaris E5 FHD, Xiaomi Mi A1 and Huawei AGS-W09. Figure 3.6 shows the BQ Aquaris E5 FHD and in Figure 3.7 is the Huawei AGS-W09 which are physical devices.

Figure 3.6 - BQ Aquaris E5 FHD

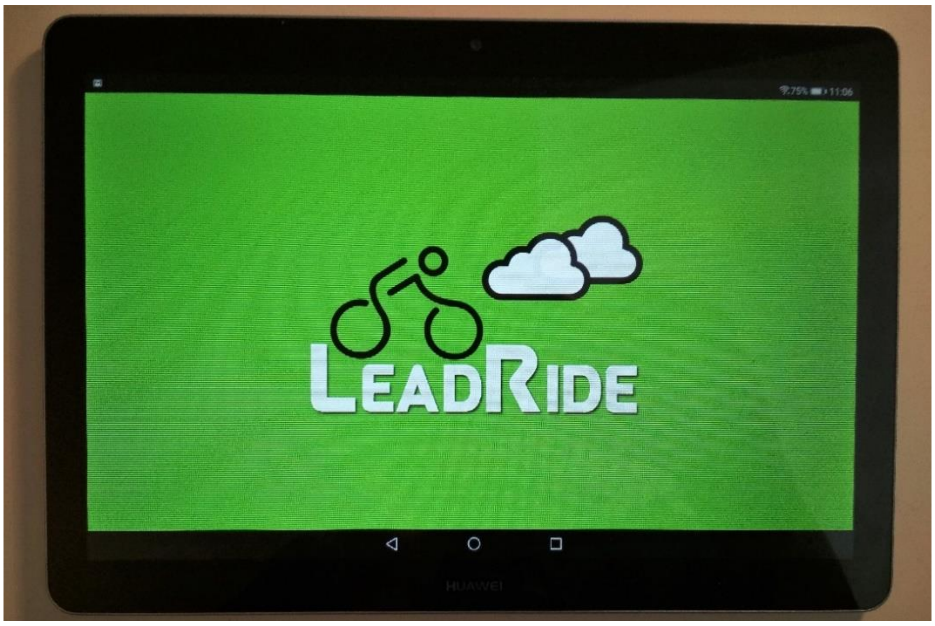


Figure 3.7 - Huawei AGS-W09

3.5 Final Assembly

Figure 3.8 shows the final assembly of the system, with the sensors, placed on the bicycle and on the cyclist.

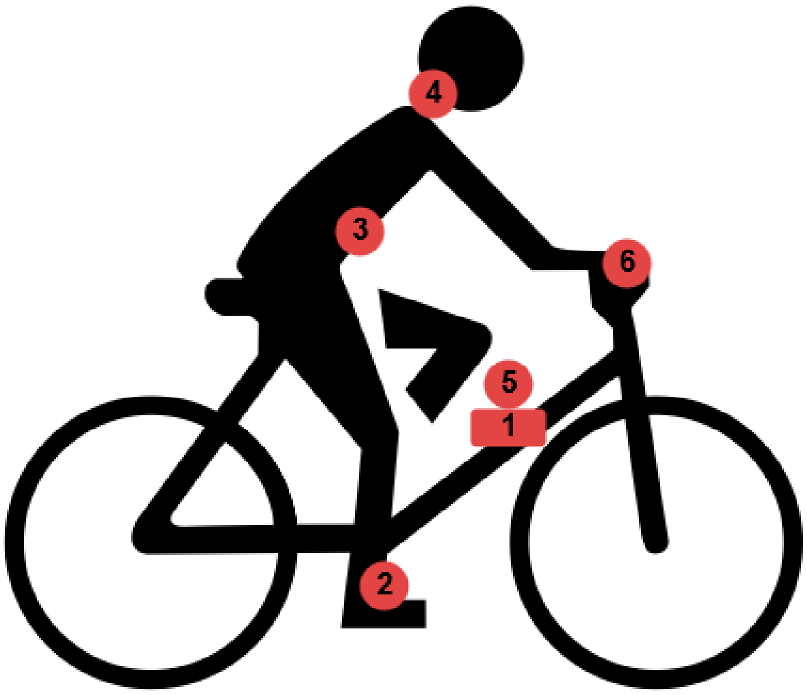


Figure 3.8 - Final assembly schematics

The coordinator (1) and one of the accelerometers (5) are placed on the bicycle frame and the second accelerometer is placed on the cyclist's upper body (3).

The gloves are placed in the hands of the cyclist (6) and measure the force that he makes in the bicycle brakes. The insoles are placed inside the shoes (2) in order to measure the force made when cycling. The strap that measures the breathing rate is placed around the chest (3) and the sensor that measures the heartbeat is placed in the cyclist's ear (4).

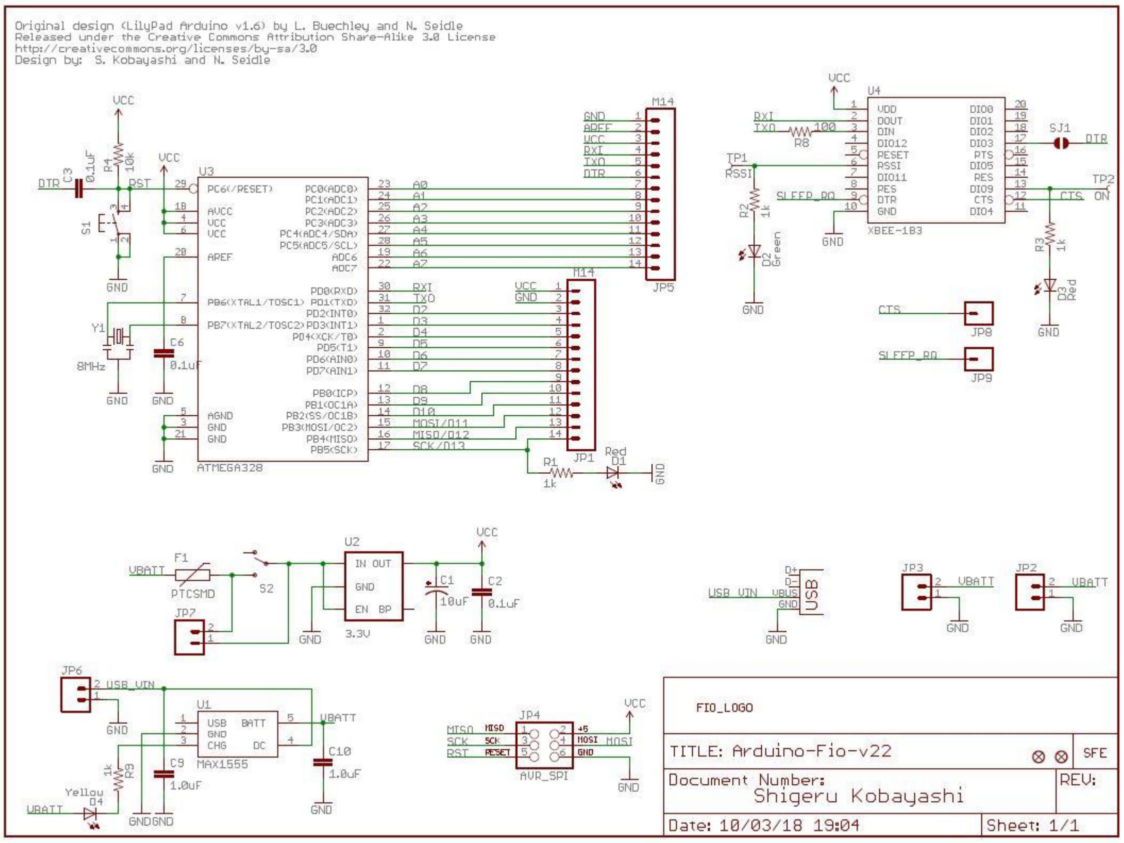
Chapter 4 Embedded System

Embedded systems are a combination of hardware, software and mechanical components designed to perform a specific function and incorporated as part of a complete device. Embedded systems are, typically, components in larger systems that are used to directly control or monitor that system [50]. These systems control many devices used on a daily basis [51].

This chapter describes the embedded system and the wireless sensor network used in this project, with components such as Arduino, sensors, sensor data processing and communications used to monitor the performance of cyclists. A WSN is composed by a coordinator, end-nodes and the communication between them.

4.1 End-nodes

The end-nodes of this system consist in sensors connected to the Arduino Fio (Figure 4.1 [49]) and the circuits of each end node are powered by lithium batteries.



Original design (LilyPad Arduino v1.6) by L. Buechley and N. Seidle
Released under the Creative Commons Attribution Share-Alike 3.0 License
http://creativecommons.org/licenses/by-sa/3.0
Design by: S. Kobayashi and N. Seidle
ATMEGA328
XBEE-1B3
MAX1555
AVR_SPI
TITLE: Arduino-Fio-v22
Document Number: Shigeru Kobayashi
Date: 10/03/18 19:04
Sheet: 1/1
REV:
SFE
FIO_LOGO

Figure 4.1 - Arduino Fio Schematics

Among the several sensors there are: force sensors, IMU boards, sensors that measure heart rate and breathing. In this project, it was developed and implemented seven end-nodes in the wireless network based on the Arduino Fio processing platform. The Arduino Fio makes the acquisition and primary processing of the sensor data, gets the data structured to send to the coordinator and then sends it.

4.1.1 Force applied by the cyclist

The force applied by the cyclist, in the breaks and pedals, was measured by FlexiForce force sensors [52]. In this project, the sensors are used on the cyclist's shoes and gloves. Each hand has two sensors, one in the index finger and the other in the middle finger, each foot has three sensors.

FlexiForce sensors are resistant to most environments, thin, flexible and can measure the force between two surfaces, which in this case will be between the fingers and the brakes and between the feet and the pedals. These sensors work as a resistance in an electrical circuit, when a force is applied to the sensor the resistance decreases and when the sensor is discharged that resistance increases. Table 4.1 shows the typical performance of the sensors [53].

Table 4.1 - Typical performance of the FlexiForce sensor

	Typical Performance
Linearity (Error)	< ±3% of Full Scale
Repeatability	< ±2.5%
Hysteresis	< 4.5% of Full Scale
Drift	< 5% per logarithmic time
Response Time	< 5 µsec
Operating Temperature	-9°C - 60°C

The sensors are constructed of two substrate layers, each one is composed by a polyester film and a conductive material (silver) followed by a layer of a pressure sensitive ink. The adhesive is used to laminate the two layers of substrate together to form the sensor [53]. The active detection area is a small circle at the end of the sensor, Figure 4.2 [52]. On the other end of the sensor, there is a weldable male square pin connector, which allows them to be easily incorporated into a circuit. The two external pins of the connector are active while the middle connector is inactive.

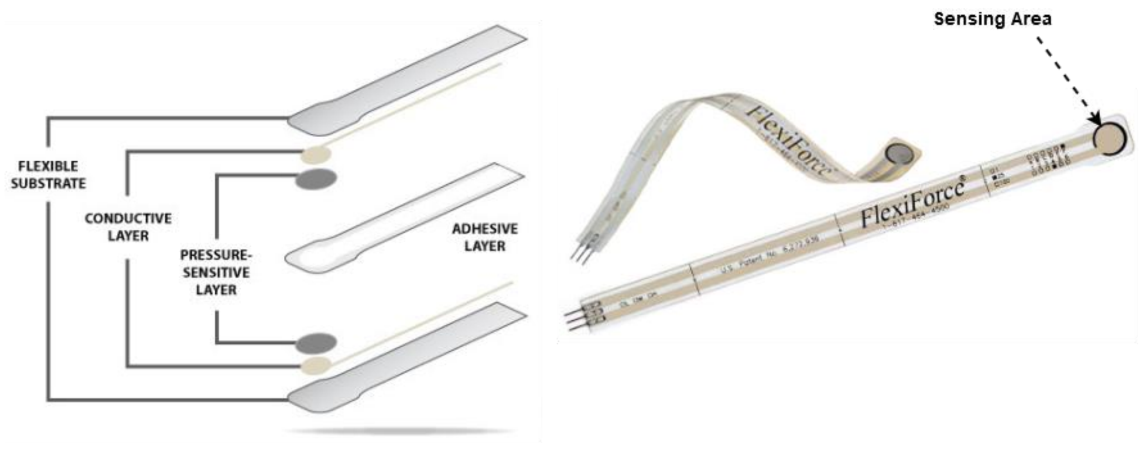


Figure 4.2 - FlexiForce sensor

It is necessary to use a conditioning circuit to get more accurate values, the circuit conditioning used is as follows (Figure 4.3). Circuit conditioning enables sensor measurements to be more effective and accurate, in this case, a signal amplifier was used in order to increase sensitivity and reduce signal-to-noise ratio (SNR)[54].

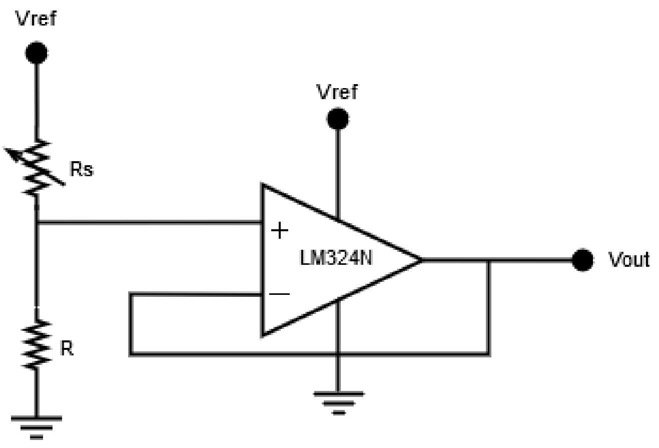


Figure 4.3 - Circuit conditioning for the force sensors

The output voltage equation (4.1) is:

$$Vout = \frac{R}{Rs + R} \times Vref \tag{4.1}$$

Where the parameters are defined as follows:

- *Vout*: is the output voltage [V];
- *Rs*: is the variable resistance that comes from the force sensor [Ω];
- *R*: is the reference resistor [Ω];
- *Vref*: is the reference voltage [V].

The $Vref$ used was 3.3 V which comes from the Arduino Fio and the reference resistance is 1MΩ.

To convert the Analog-to-Digital Converter (ADC) [55] values from the sensors to Newtons (N) the following formula (4.2) is used:

$$gain = \frac{KnownWeight}{ADCValue} = \frac{1500g}{131} \approx 11.45 \qquad (4.2)$$

The gain was calculated using equation (4.2) with the same known weight as the referenced project [56], and with the resolution of the ADC (ADCValue) of 1023, corresponding to the Arduino Fio [39]. In equation (4.2) it is used a known weight (in this case 1.5 kg) and the read ADC value in order to calibrate the sensor.

$$weight\ (in\ Newton) = gain\ \times ADC_{value}\ \times 0.001\ \times 9.80665 \qquad (4.3)$$

Equation (4.3) was used to calculate the force applied to each sensor. The $gain\ \times ADC_{value}$ is the weight in grams (equation (4.2)) but, in order to calculate the force in Newtons, it has to be in kilograms, so it is multiplied by 0.001, finally multiplied by the gravitational acceleration.

Figure 4.4 and Figure 4.5 shows the end-nodes containing the force sensors.

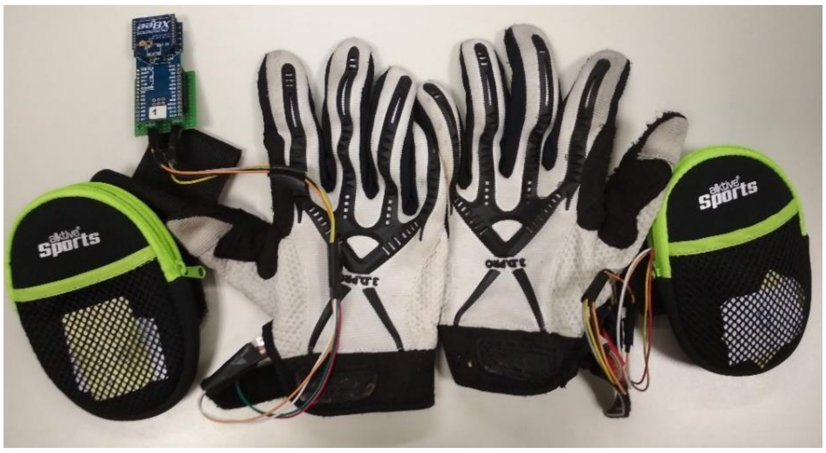


Figure 4.4 - Hands end-node with force sensors

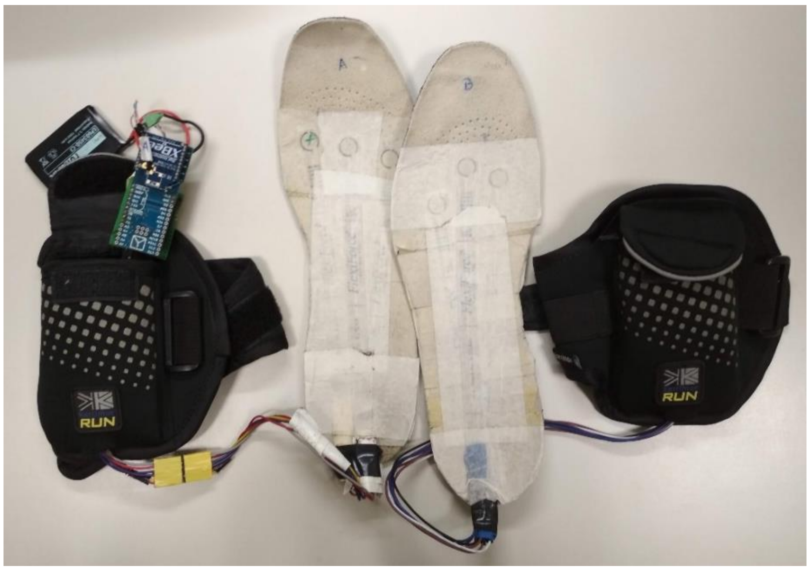


Figure 4.5 - Feet end-node with force sensors

4.1.2 Angles of the cyclist and the bicycle

An Inertial Measurement Unit was used to calculate the angles of the cyclist and the bike as well as the direction of the movement, the system uses two sensors one placed on the cyclist's chest to measure the upper body motion and the other one is placed on the bicycle frame. This IMU packs a L3GD20H 3-axis gyroscope, a LSM303D 3-axis accelerometer and 3-axis magnetometer onto a tiny board [57], Figure 4.6.

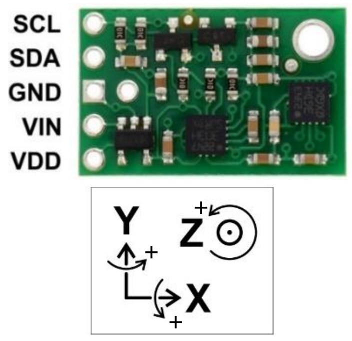


Figure 4.6 - MinIMU-9 v3 Gyro, Accelerometer, and Compass

The connection between the IMU board and the Arduino Fio is made through Inter-Integrated Circuit (I^2C) communication protocol. The I^2C works in the master-slave model, with at least one device acting as master, and the other devices acting as a slave. The function of the master is to coordinate the communication [58].

This sensor has nine independent readings of rotation, acceleration and magnetic, providing all the data needed to create an Attitude and Heading Reference System (AHRS).

The gyro can be used to very accurately track rotation on a short timescale, while the accelerometer and compass can help compensate for gyro drift over time by providing an absolute frame of reference. The respective axes of the two chips are aligned on the board to facilitate these sensor fusion calculations.

In the previous work, it was used the Direct Cosine Matrix Algorithm (DCM) (Figure 4.7) to calculate the angles between the cyclist and the bike [56].

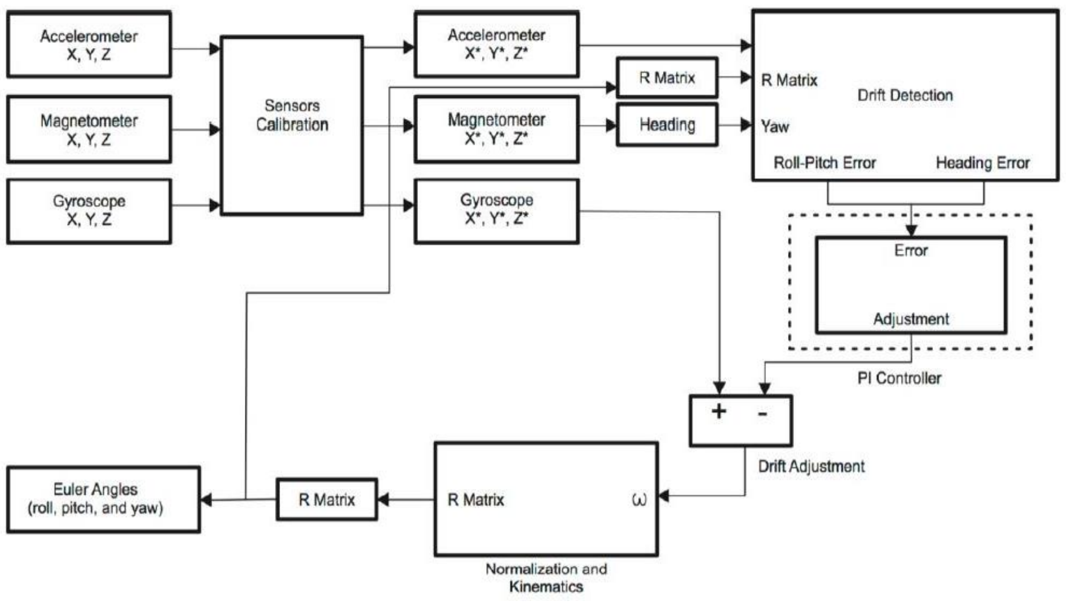


Figure 4.7 - DCM Algorithm

This algorithm calculates the orientation of a rigid body relative to earth rotation using rotational matrices that describe the orientation of one coordinate system relative to another. The columns of the matrix are unit vectors of a system, these vectors can be transformed into other systems when multiplied by the rotation matrix, equation (4.4).

$$R = \begin{bmatrix} r_{xx} & r_{xy} & r_{xz} \\ r_{yx} & r_{yy} & r_{zy} \\ r_{zx} & r_{zy} & r_{zz} \end{bmatrix} = rotation\,matrix \tag{4.4}$$

Certain types of vectors (directions, velocity, acceleration and translation) can be transformed between rotated reference frames with a 3x3 matrix.

Example, equation (4.5):

A_P = a vector A measured in the frame of reference of a plane;

A_G = a vector A measured in the frame of reference of the ground.

$$A_G = RA_P \tag{4.5}$$

The Euler angles describe the three consecutive rotations required to describe the orientation, the rotation matrices are related to these angles. The relationship between the cosine direction matrix and the Euler angles ψ, θ, φ is represented in Equation (4.6) [56].

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\psi & -\sin\psi \\ 0 & \sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} \tag{4.6}$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} X'' \\ Y'' \\ Z'' \end{bmatrix} \tag{4.7}$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X''' \\ Y''' \\ Z''' \end{bmatrix} \tag{4.8}$$

$$R = \begin{bmatrix} \cos\theta\cos\phi & \sin\psi\sin\theta\, cos\,\phi - \cos\psi\, sin\phi & \cos\psi\sin\theta\, cos\,\phi + \sin\psi\, sin\phi \\ \cos\theta\sin\phi & \sin\psi\sin\theta\sin\phi + \cos\psi\, cos\phi & \cos\psi\sin\theta sin\,\phi - \sin\psi\, cos\phi \\ -\sin\theta & \sin\psi\cos\theta & cos\psi\, cos\theta \end{bmatrix} \tag{4.9}$$

Equation (4.4) and (4.9) express how to rotate a vector measured in a frame of reference of a plane to the frame of reference of the ground (Figure 4.8 [59]), equation (4.4) is expressed in terms of direction of cosines while equation (4.9) is expressed in terms of Euler angles [60].

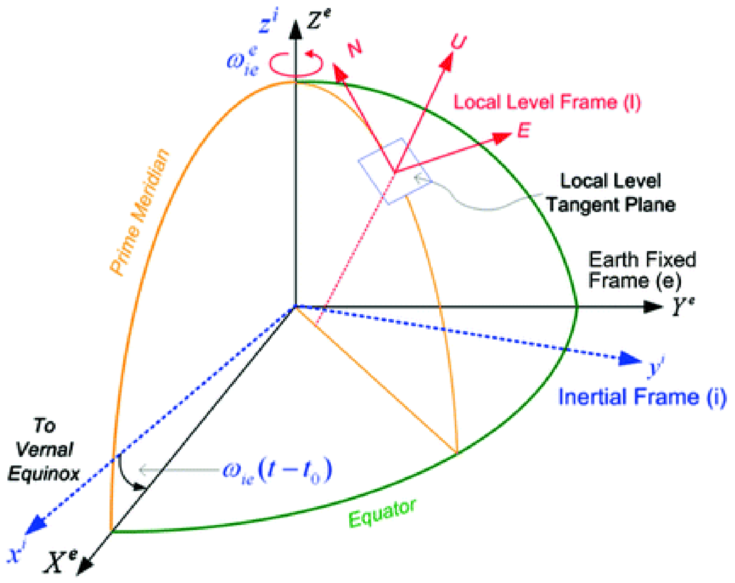


Figure 4.8 - Vector measured in the frame of reference of the plane to the frame of reference of the ground

The following equations allow to calculate Yaw (4.10), Pitch (4.11) and Roll (4.12):

$$\phi = atan2(r_{zy}, r_{xx}) \tag{4.10}$$

$$\psi = -\text{asin}(r_{zx}) \tag{4.11}$$

$$\theta = atan2(r_{zy}, r_{zz}) \tag{4.12}$$

Figure 4.9 shows the end-node with the IMU boards.

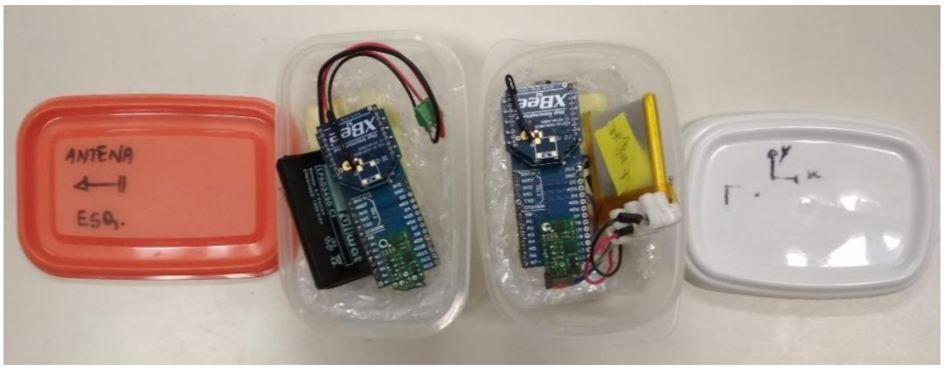


Figure 4.9 - End-nodes with IMU boards

The position and orientation of the cyclist are given by the set of three parameters previously spoken. The position of the sensor was oriented as shown in Figure 4.10.

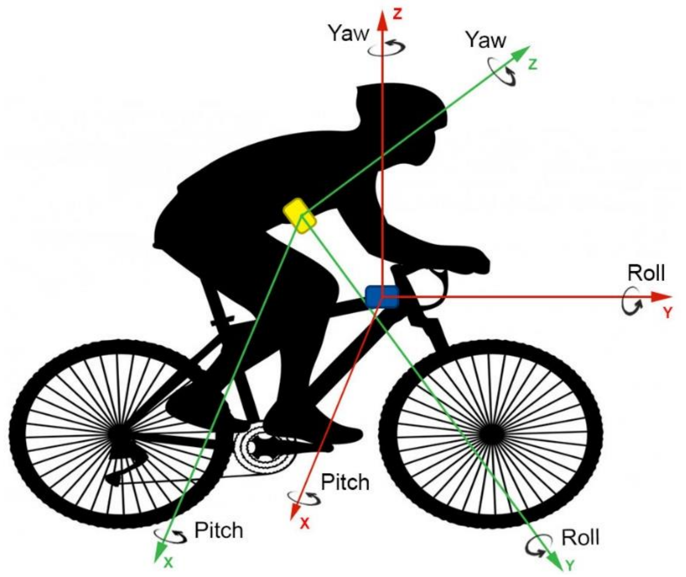


Figure 4.10 - Representation of the IMU axis

4.1.3 Heart Rate Monitor

In order to obtain the cyclist's pulse during training, the method known as Photoplethysmogram (PPG) was used, it is a non-invasive method and consists in the variation of the absorption of light, traditionally through the finger. This type of sensors uses light-based technology to sense the rate of the blood flow [29] through its volume, which is an effect of the contraction of the heart [61]. Figure 4.11 is the representation of the sensor working mode.

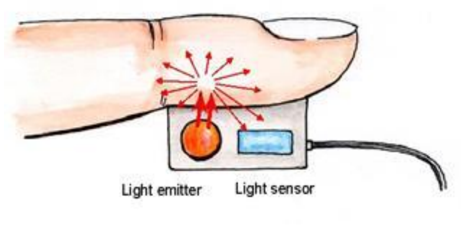


Figure 4.11 - PPG sensor working mode

In the case of this project, the used pulse sensor was the SEN-11574 (Figure 4.12 [62]). The sensor was placed on the ear, the force sensors were already included in the gloves. This heart rate sensor has its own library and, in this case, it was used to calculate the Beats Per Minute (BPM).

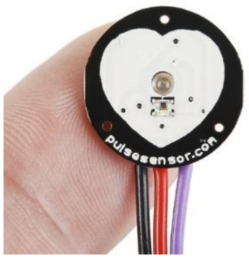


Figure 4.12 - Pulse Sensor SEN-11574

This sensor is connected to the Analog Port 1 in the Arduino Fio (a wire connected to the ground and the other one to the Arduino 3.3V).

4.1.4 Respiratory Rate

In order to get the cyclist’s breathing rate, it was used an electromechanical film sensor from EMFIT (Figure 4.13). An electromechanical film is a thin membrane which thickness is related to an electric voltage. These sensors are ribbon type sensors and consist of a ferroelectric film and 3 layers of electrodes on polyester films or aluminum or copper electrodes. The standard width and length of this type of sensor are 19mm and 6 meters, respectively, but they are prepared to be cut into smaller sensors [63]. In this case, the sensor was cut into a smaller one, about 30 cm.

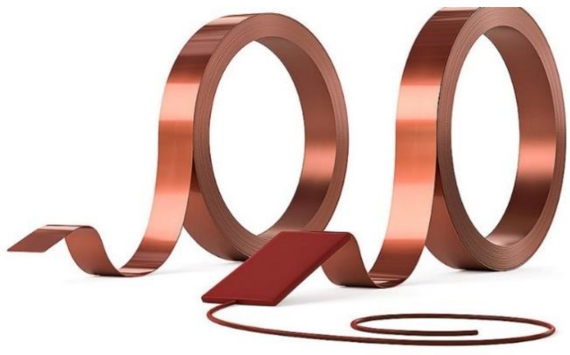


Figure 4.13 - EMFIT electromechanical film

It is necessary to use a conditioning circuit to get more accurate values, the circuit conditioning used is as follows (Figure 4.14).

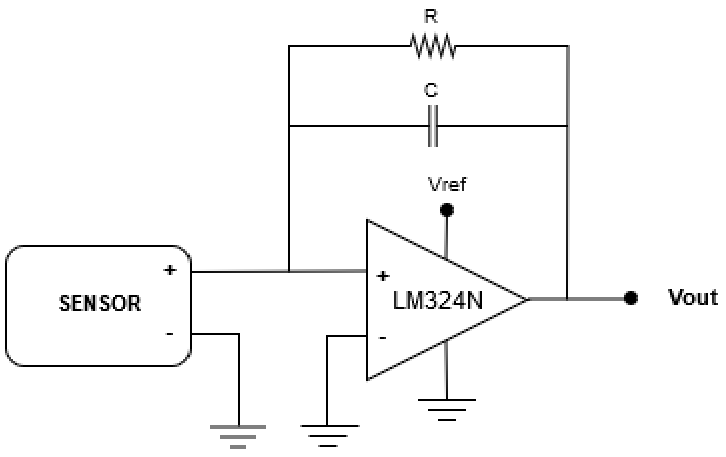


Figure 4.14 - Conditioning circuit for the electromechanical film

The thickness changes of these sensors generate charge, and this is how it is possible to calculate the cyclist's breathing rate. Each inhale and exhalation generates a non-zero voltage that is read and counted by the Arduino. The total count is divided by two, once a breath corresponds to one inspiration and expiration movement, the count is sent to the coordinator every minute. This sensor is attached to a strap around the cyclist's chest.

Figure 4.15 shows the end-node with the PPG sensor and the respiratory strap.

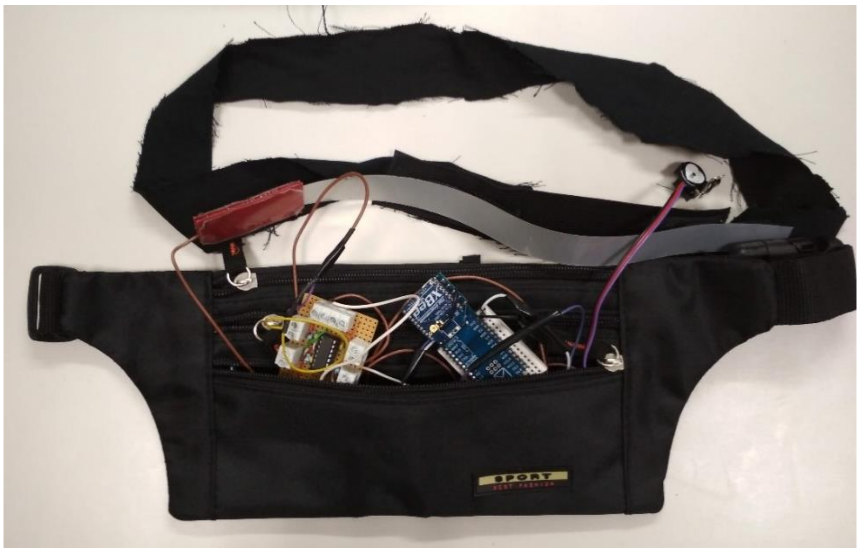


Figure 4.15 - End-node with the respiratory strap and the PPG sensor

4.2 Coordinator

To implement the WSN, an Arduino Mega (Figure 4.16) was used as the coordinator. The role of a coordinator is to manage the system: thus, it receives data from different end-nodes, sends information to the database, authentic cyclists and has a GPS module that records the coordinates of the training course.

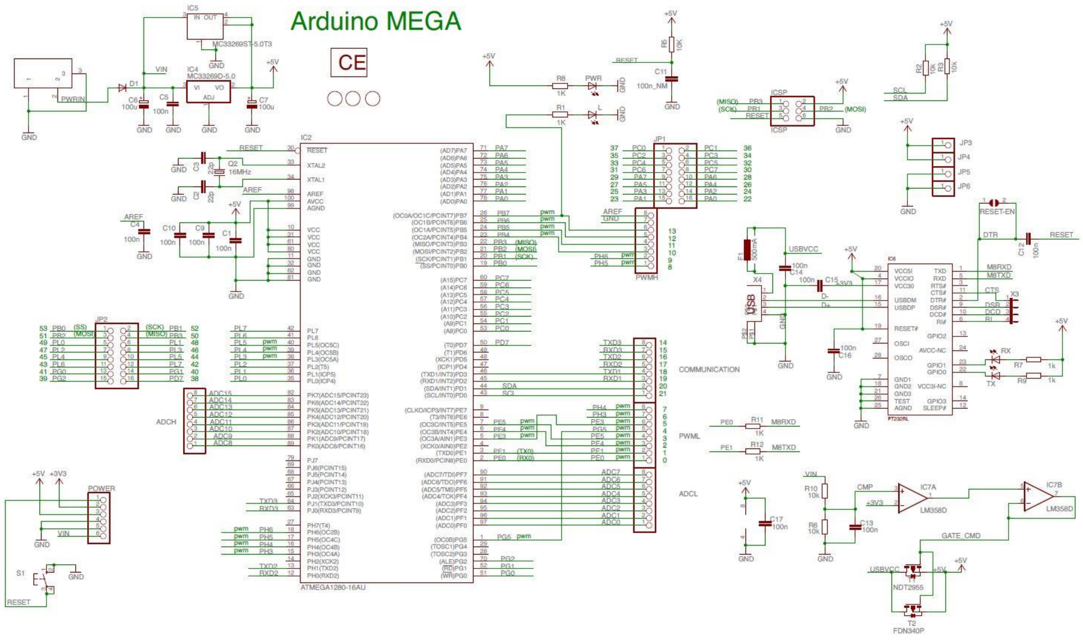


Figure 4.16 - Arduino Mega schematics

The coordinator can be found on the frame of the bike. In the coordinator, there are three shields and a sensor: GPS, Yun Shield, XBee shield and the RFID reader (Figure 4.17).

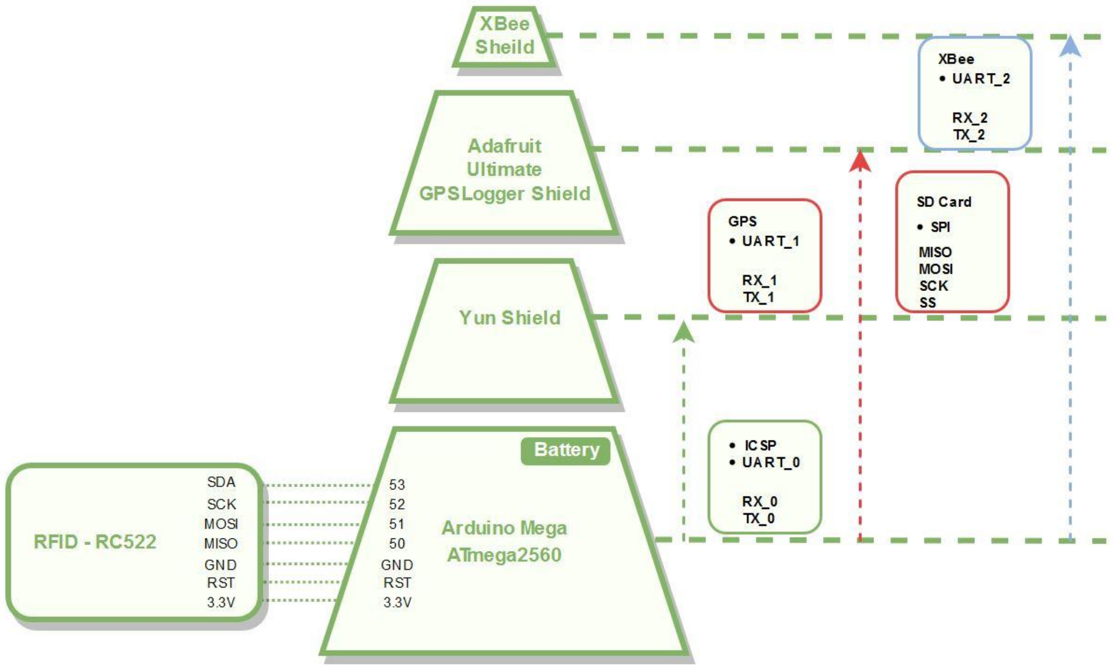


Figure 4.17 - Coordinator's assembly with the RFID reader, the Yun, GPS and XBee shields

4.2.1 GPS Shield

In order to get the information of the cyclist's location and the training route, a GPS shield was included in the coordinator. This shield is able to collect coordinates, date, time and speed information regardless of telephone or internet reception.

The shield (Adafruit Ultimate GPS Logger Shield, Figure 4.18) uses an embedded GPS module from GlobalTop PA6H with Mediatek MT3339 that achieves the industry's highest level of sensitivity (-165 dBm). It also has the lowest power consumption, for precise GPS signal processing to give the ultra-precise positioning under low receptive, high velocity conditions, instant Time-to-First-Fix (TTFF) [64]. It also supports a SD memory card, in which will be kept the RFID tags information, the Arduino program access this information to identify the cyclist that will perform the training.

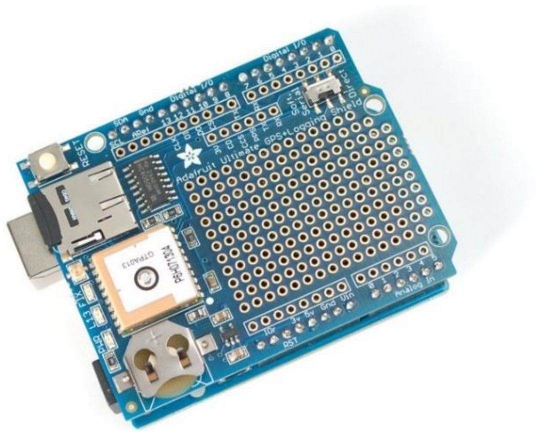


Figure 4.18 - Adafruit Ultimate GPS Logger Shield

4.2.2 Cyclist Identification

The identification of the cyclist and the beginning of the training is marked by the reading of a unique RFID tag that each cyclist has. The coordinator only starts sending the data to the database after this authentication. This is accomplished through an RFID reader, RFID-RC522, which is embedded in the coordinator, this RFID reader uses a 13.56 MHz frequency, which is considered a High Frequency, as it is indicated in the following table (Table 4.2).

Table 4.2 - RFID Frequencies

Frequency Band	**Frequency Range**	**The most commonly used frequencies in the RFID system**
Low Frequency (LF)	100 kHz - 500 kHz	125 kHz, 134.2 kHz
High Frequency (HF)	10 MHz - 15 MHz	13.56 MHz
Ultra-High Frequency (UHF)	400 MHz - 950 MHz	866 MHz - Europe, 915 - USA
Microwaves (µF)	2.4 GHz - 6.8 GHz	2.45 GHz, 3.0 GHZ

Radio frequency identification is a non-contact and automatic identification method using radio signals, electronically storing identification data in a tag. These tags contain transponders that emit messages that can be read by specialized readers and are divided into three types: active, passive and semi-passive [65].

Active tags have their own power source, which makes them more expensive. This type of tags can be constantly emitting a signal but may also remain inactive until they reach the range of an active receiver.

On the other hand, passive tags are very cheap and smaller, which are some of the reasons they are the basis of most RFID implementations, even though they have a much shorter range (approximately 10 meters). This type of tags can store energy from the signal of the reader, this energy is used to send the data that it has stored to the reader, these are the tags used in the project.

The semi-passive tags are like a cross between the previous two, have a battery but have no internal power source, they use, just like the passive, the energy of the reader to charge their battery.

4.2.3 Arduino’s Internet Connection

For this work, it is necessary to send the collected data from the sensor to the database, so it is required an internet connection, for that it was used the Iduino Yun Shield (Figure 4.19 [66]). This shield uses two communications protocols with the microcontroller, Serial Peripheral Interface (SPI) and Universal Asynchronous Receiver Transmitter (UART). The SPI interface is only used during the upload of the sketch coming from the Arduino IDE, so the Arduino SPI can still be used to connect other slaves [66].

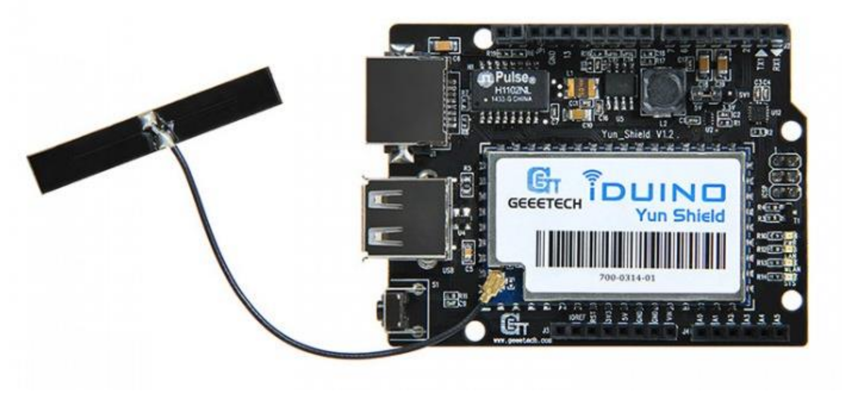


Figure 4.19 - Iduino Yun Shield

The Arduino Mega has an ATmega16u2 which provides a virtual com port to software on the computer. The UART between mega 2560 and mega16u2 will influence the Bridge feature with the Iduino Yun Shield, so, it is necessary to disconnect it by setting mega16u2 into reset mode. This intervention is made on Arduino mega, the utilization of the header In-Circuit Serial Programming (ICSP) is fundamental because it will possible to develop the applications through a Wi-Fi connection. Eliminating the need of the Universal Serial Bus (USB) cable connected to the microcontroller, for the insertion of new sketches [56]. These interventions are shown and explain with more detail in the Technical manual.

The Yun Shield has an open source Linux system that allows to send the data to Firebase through the command line, with one command is possible to send all the data at the same time which also saves battery and the resources of the Arduino.

4.3 Communications

For the transfer of data, wireless communication protocols were used since it is a wireless sensor network. To communicate within WSN was used the ZigBee protocol and for the communication between the WSN's coordinator and the remote database was used the Wi-Fi protocol.

4.3.1 WSN: End node and Coordinator

All end-nodes and the coordinator have a XBee radio that is based on the IEEE 802.15.4 Standard that was designed for star, tree or point-to-point communications. This type of communication typically operates within a 10m radius and has a low power consumption, which is an important factor for such systems, as explained in Section 2.4. In this project it was used the star typology, as represented in Figure 4.20, this was the typology chosen because the nodes are close to the coordinator and they only transmit information.

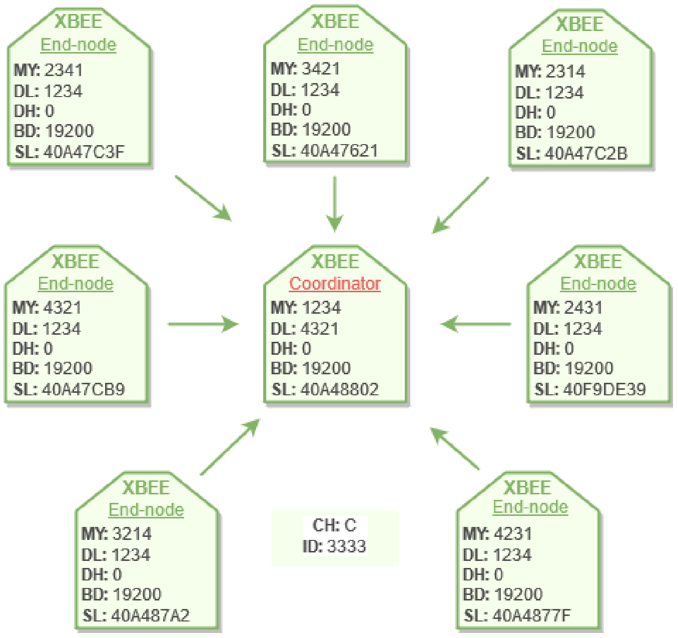


Figure 4.20 - Representation of the project's star topology

The main configurations of the XBee module are PAN ID, in order to identify the network, the Destination Address (High and Low) to distinguish the nodes within the network and the baud rate. These settings were made using the XCTU software [67]. Table 4.3 and Table 4.4 shows the configurations made on the coordinator and one of the end-nodes XBee. The remaining XBee settings are in the Technical manual.

Table 4.3 - Coordinator's XBee configuration

Coordinator	
Channel (CH)	C
PAN ID (ID)	3333
Destination Address High (DH)	0
Destination Address Low (DL)	4321
16-Bit Source Address (MY)	1234
Serial Number High (SH)	13A200
Serial Number Low (SL)	40A48802
Coordinator Enable (CE)	Coordinator [1]
Interface Data Rate (BD)	19200 [4]
API Enable (AP)	API enabled [1]

Table 4.4 - Respiration rate and heart rate end-node XBee configuration

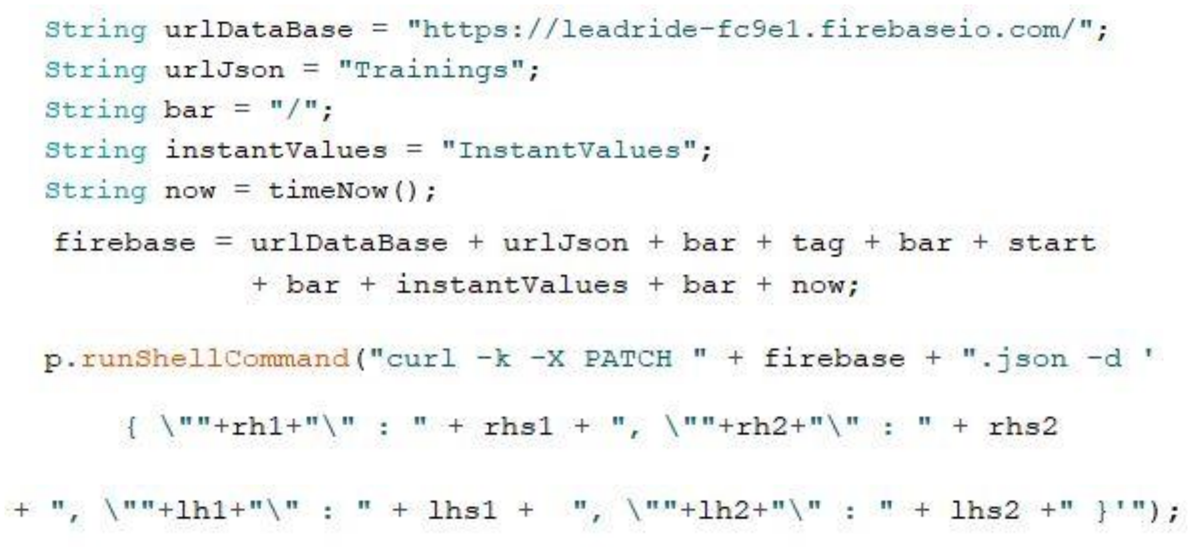
Respiration rate and heart rate end-node	
Channel (CH)	C
PAN ID (ID)	3333
Destination Address High (DH)	0
Destination Address Low (DL)	1234
16-Bit Source Address (MY)	4231
Serial Number High (SH)	13A200
Serial Number Low (SL)	40A4877F
Coordinator Enable (CE)	End Device [0]
Interface Data Rate (BD)	19200 [4]
API Enable (AP)	API enabled w/PPP [2]

In order to send the information to the coordinator, structured messages are formed with the values of the sensors and a flag, which allows distinguishing the end-node from which the message comes from. The values, inside the message, are always sent in the same order to be able to distinguish which sensor they correspond.

4.3.2 Coordinator – Database

The coordinator is also the WSN's gateway, and it includes a Yun shield (see subsection 4.2.3). Using a Wi-Fi modem, it is possible to connect to the database and send data. The router used is the Alcatel MW40V, an ideal portable device to allow access to the Internet from any location. The small-sized structure makes it easy to transport to any location and is a great asset to use in camping, nature walks or outdoor activities. This router features 4th Generation (4G) Long Term Evolution (LTE) network technology, delivering download speeds of up to 150 Mbps and upload speeds of up to 50 Mbps. Its 1800 mAh battery gives it up to seven hours of continuous operation [68], which means the battery lasts a whole training.

The following code (Figure 4.21) represents the Arduino code that was used to send the information from the coordinator to Firebase:



```
  String urlDataBase = "https://leadride-fc9e1.firebaseio.com/";
  String urlJson = "Trainings";
  String bar = "/";
  String instantValues = "InstantValues";
  String now = timeNow();
   firebase = urlDataBase + urlJson + bar + tag + bar + start
              + bar + instantValues + bar + now;

  p.runShellCommand("curl -k -X PATCH " + firebase + ".json -d '

       { \""+rh1+"\" : " + rhs1 + ", \""+rh2+"\" : " + rhs2

+ ", \""+lh1+"\" : " + lhs1 +  ", \""+lh2+"\" : " + lhs2 +" }'");
```

Figure 4.21 - Arduino code to send data to Firebase

This is an example of what was used, this particular part of the code sends the information about the force sensors on both hands, *rh1*, *rh2*, *lh1* and *lh2* are the names of the key in the database. The variables *rhs1*, *rhs2*, *lhs1* and *lhs2* are the actual values that the coordinator has received. The string *firebase* represents the path in the database where the data is being kept. The remaining sensor values are being sent to the database by the same method but with the respective keys, for example the key to yaw of the bycile measured by the IMU is *yawBicycle.*

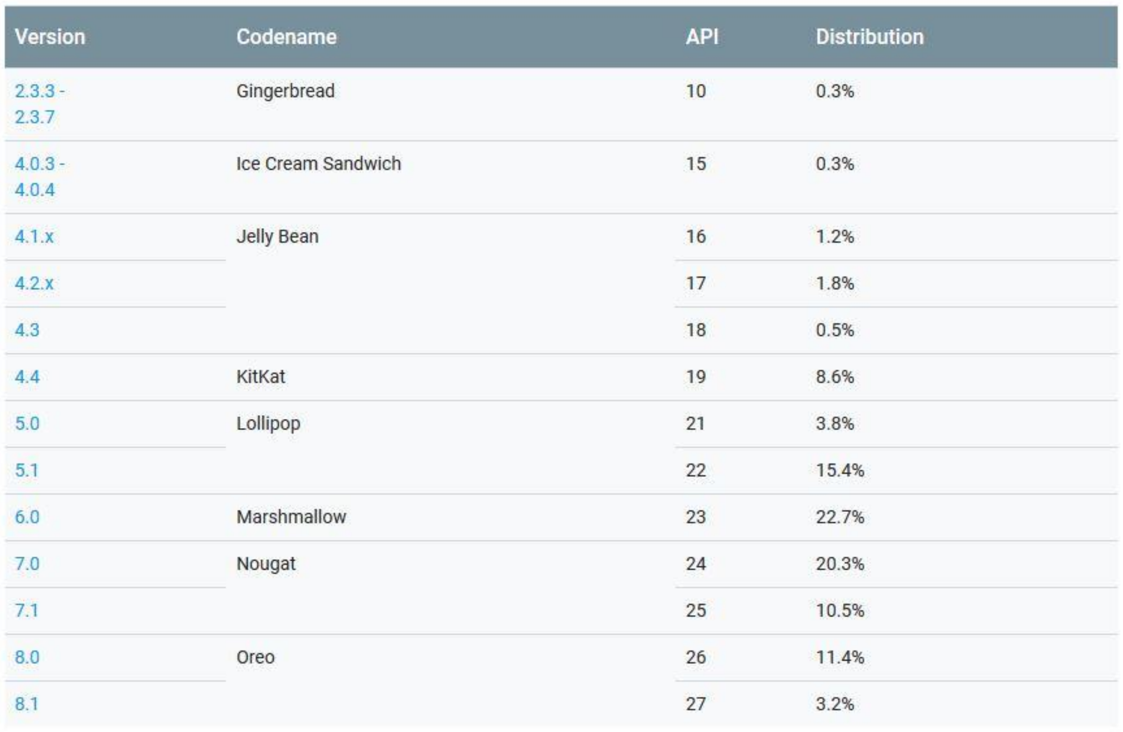
Chapter 5 Cloud and Mobile Application

5.1 Mobile Application

The mobile application of this system, developed for Android, aims to help the coach manage the results of his cyclist's training. The cyclist can also see the data from the different training.

To develop the application, it was used the official IDE (Android Studio) of the Android operating system from Google and the programming language used was Java.

In the official website of Android [69] is available information regarding the most used versions, in Figure 5.1 this information is represented, in studies completed on the 12th of September 2018, it was verified that the most used version was 6.0 (Marshmallow), so this was the target version chosen. The minimum version chosen was 5.0 (Lollipop).



Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.3%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.3%
4.1.x	Jelly Bean	16	1.2%
4.2.x		17	1.8%
4.3		18	0.5%
4.4	KitKat	19	8.6%
5.0	Lollipop	21	3.8%
5.1		22	15.4%
6.0	Marshmallow	23	22.7%
7.0	Nougat	24	20.3%
7.1		25	10.5%
8.0	Oreo	26	11.4%
8.1		27	3.2%

Figure 5.1 - Android's most used versions

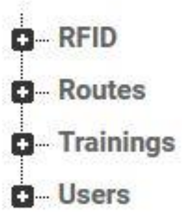
5.2 The data structure in Firebase

All information regarding cyclists, coaches and training results has to be centrally stored in a place where it can be accessed remotely by anyone with permission and anywhere. The *Firebase Realtime Database* platform is responsible for this functionality, this service generates a unique Uniform Resource Locator (URL) so that the user who created the database can use it.

Firebase has a dynamic structure, but it is essentially organized by nodes, each of which is identified by a unique key, through which data is accessed in the application. There may be numerous nodes in the database root and each of these nodes can have numerous children. The last node of the chain is where the values are stored, also those identified by a unique key.

By using Firebase, nothing is defined at the server level, the data is sent to Firebase with a certain structure that the platform can recognize. The structure used in this project will be explained in this section.

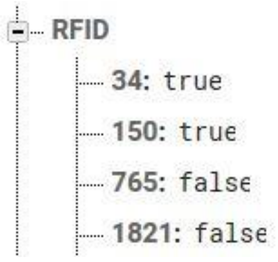
Figure 5.2 represents the main nodes of this system.



RFID
Routes
Trainings
Users

Figure 5.2 - Main nodes of the system

The first node, RFID node (Figure 5.3), stores all the RFID tags that may be considered for the cyclists, it also holds a Boolean (true or false) value that represents whether or not that tag is being used by any cyclist.



```
RFID
    34: true
    150: true
    765: false
    1821: false
```

Figure 5.3 - RFID node example

The Users node (Figure 5.4) has all users registered to the system: Admin, Coaches and Cyclists. Admin and Coaches are identified by their unique username, Cyclists are identified by a key composed of the letter C and its RFID tag. The mobile application has different activities to register the new users, so when a new user is registered he can be mapped into the correspondent node. In Table 5.1 are the attributes of each user.

Table 5.1 - Attributes by user type

	Admin	**Coach**	**Cyclist**
Name	√	√	√
Username	√	√	√
Password	√	√	√
Email	√	√	√
List of cyclists	-	√	-
RFID Tag	-	-	√
Birthday	-	-	√
Coach Username	-	-	√
Gender	-	-	√
Height	-	-	√
Weight	-	-	√

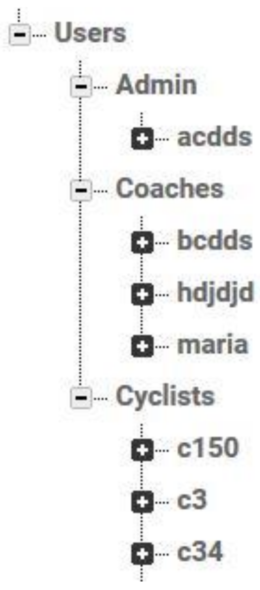
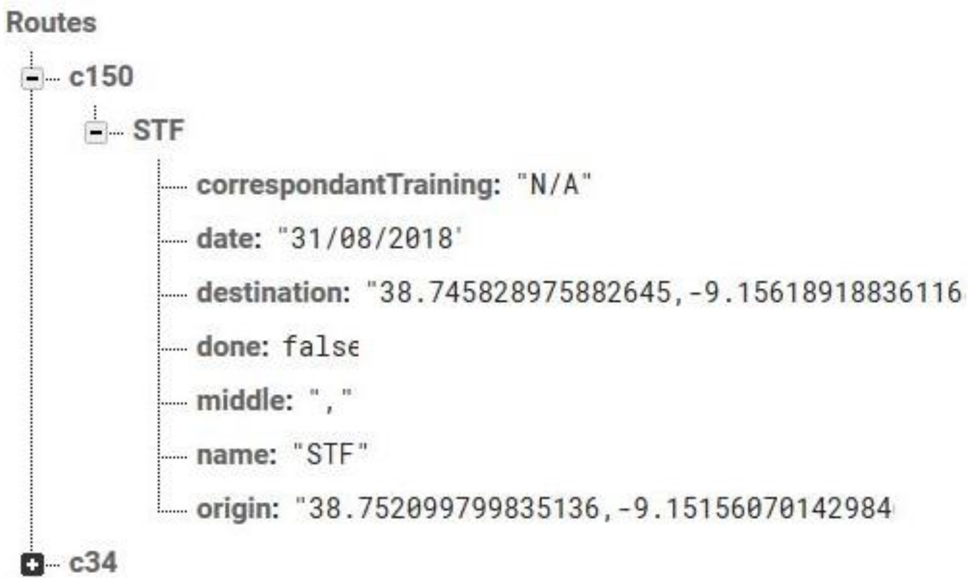


Figure 5.4 - Users node example

The second node (Figure 5.5) represents the planned or completed routes. The route identification is done by combining two keys: a key that identifies the cyclist and a unique name given by the user when creating the route.



```
Routes
⊟-- c150
    ⊟-- STF
        |-- correspondantTraining: "N/A"
        |-- date: "31/08/2018'
        |-- destination: "38.745828975882645,-9.15618918836116
        |-- done: falsε
        |-- middle: ","
        |-- name: "STF"
        └-- origin: "38.752099799835136,-9.15156070142984
⊞-- c34
```

Figure 5.5 - Routes node example

Finally, the Trainings node (Figure 5.6) contains the information from the sensors of each training, sent by the Arduino. The training of a cyclist's training is identified, once again, by his key and each training is identified by the start date and time. Within each training there are at least three different nodes: InstantValues, ValuesPerMinute and GPSValues, there may still be a fourth and a fifth node, TrainingData and/or Comments. The first three nodes are relative to the values that come from the Arduino, InstantValues are values that are sent as soon as they are collected (force sensors, yaw, pitch and roll), ValuesPerMinute are values that are sent every minute (heart rate and respiratory rate) and GPSValues are values that are also sent as soon as they are collected, but are relative to the data provided by the GPS shield (coordinates, altitude and speed). The TrainingData node is built after the user opens the training result in the application for the first time, this node contains information about, for example, the average speed and altitude, training duration, calories burned. The Comments node is only created if the Coach adds a comment.

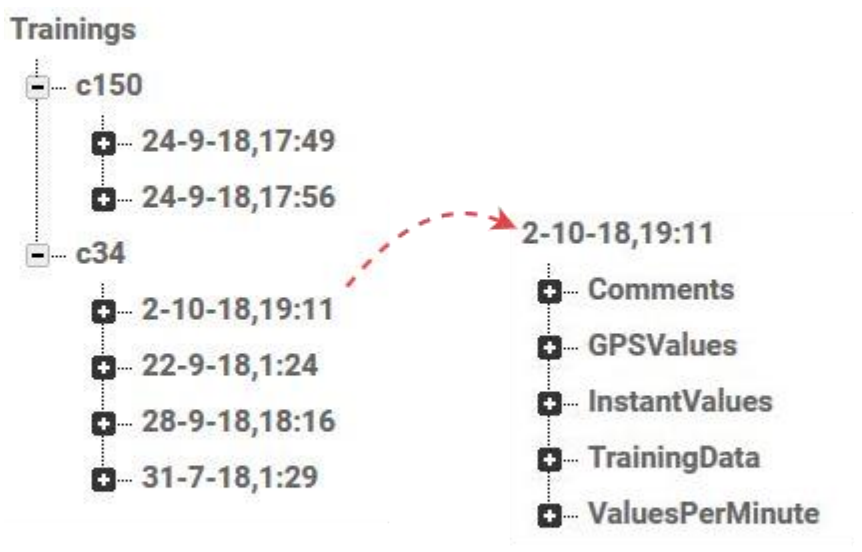


Figure 5.6 - Trainings node example

5.3 Communication with Firebase

In order for information to be exchanged, it is used the REST API on the Arduino side and the Firebase SDK on the Android application side.

5.3.1 REST API

Firebase uses the REST API and it allows to use any Firebase Realtime Database URL as an endpoint.

The REST API is a client-server protocol in which each Hypertext Transfer Protocol (HTTP) message contains all the information necessary to understand the request, as such, neither client nor server needs to record any communications state between the messages [70].

The REST API has a set of well-defined operations that apply to all information resources such as POST, GET, PUT, PATCH, and DELETE.

To save data in Firebase it is possible to use:

- PUT: writes, if it does not exist, and replaces, if it exists, data in a defined path or in the URL;
- PATCH: updates keys and data on a specific path without replacing the data;
- POST: Adds a list of data, whenever a POST request is used, a unique random key is generated to differentiate the data.

The PATCH request was used to send sensors data to the Cloud because the purpose is to add data and not replace, as it does when using PUT. POST is also not used because it is not necessary to have a random key. The data has a unique key, but it is not randomly generated, the keys are logically generated by the system as it was explained before. The Arduino code used to achieve this is in Figure 4.21 (Chapter 4).

To send the data to the Cloud the cURL command was used, this command is used to get or send files using a URL syntax [71], in this case is used in combination with PATCH, the URL is what contains the database of Firebase and is also placed at the end of the command a JSON format file with the data to be saved.

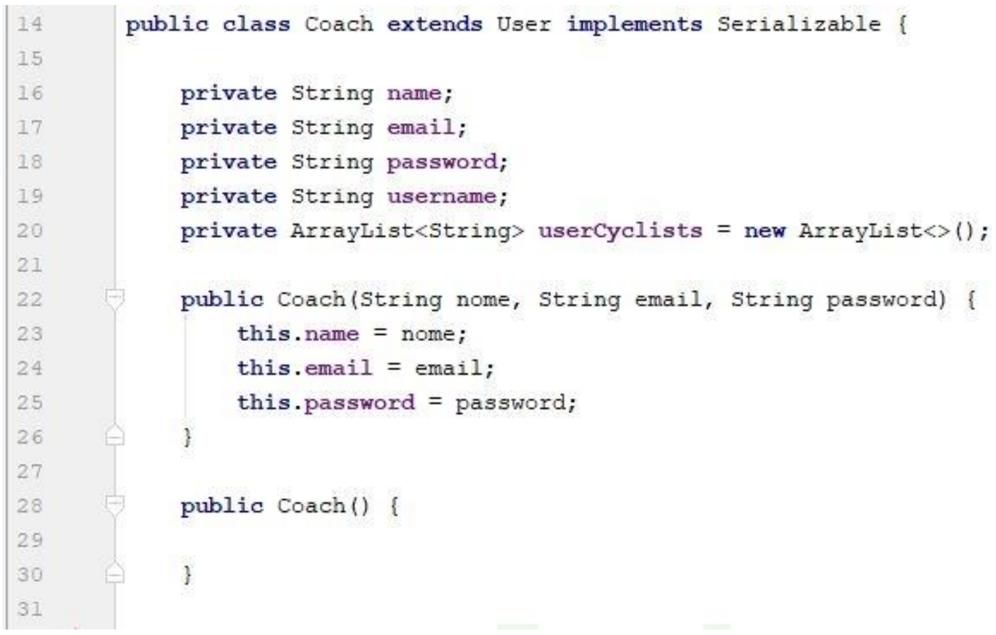
5.3.2 Firebase SDK

To use Android with Firebase the Firebase SDK was installed. The available features of this tool were used to get and send data to the database.

Firebase allows the following data types [44]:

- String;
- Long;
- Double;
- Boolean;
- Map<String, Object>;
- List<Object>.

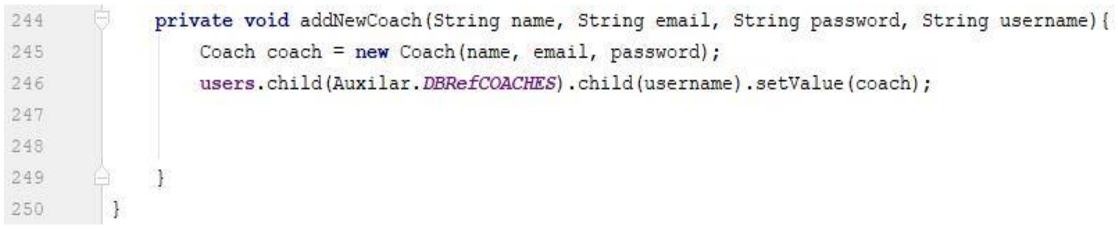
By allowing Object it means that it is possible to get and place directly custom java objects whose attributes must be of the types mentioned above. If java objects are used, they will be automatically mapped to child places in an organized way, the name of the object's attributes are the key to the mentioned property. Figure 5.7 is the class that represents the coach object.



```java
public class Coach extends User implements Serializable {

    private String name;
    private String email;
    private String password;
    private String username;
    private ArrayList<String> userCyclists = new ArrayList<>();

    public Coach(String nome, String email, String password) {
        this.name = nome;
        this.email = email;
        this.password = password;
    }

    public Coach() {

    }

```

Figure 5.7 - Representation of the Coach object

In Figure 5.8 is an excerpt of the code used to place an object in the database.



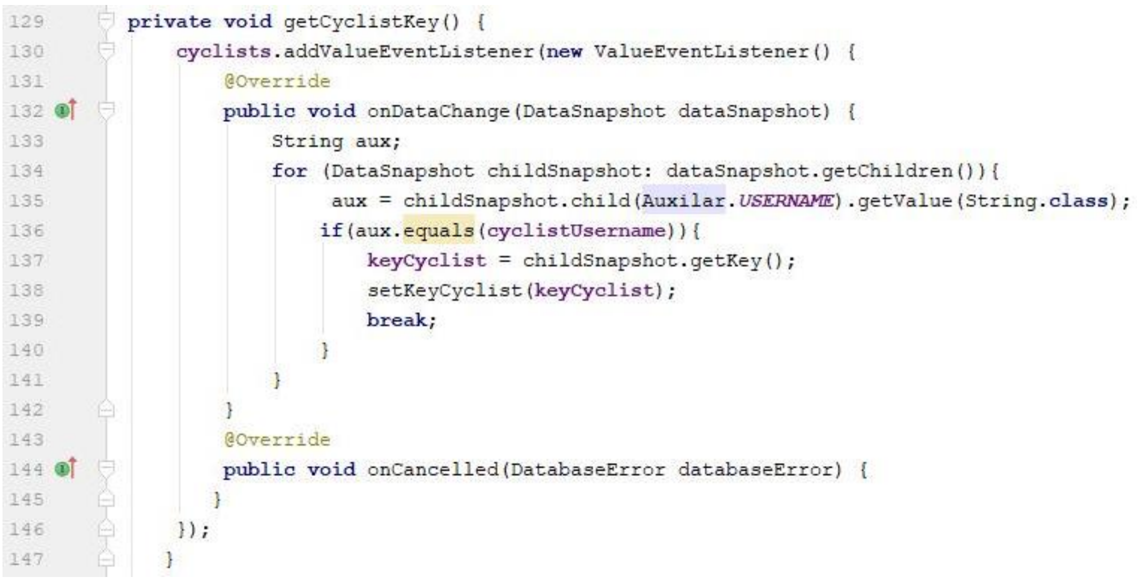
```java
    private void addNewCoach(String name, String email, String password, String username){
        Coach coach = new Coach(name, email, password);
        users.child(Auxilar.DBRefCOACHES).child(username).setValue(coach);


    }
}
```

Figure 5.8 - Placing an object in Firebase

The variable *users* is the reference for the database, the *child* function, which receives a key as a parameter, allows access to a certain node within the database, and it is possible to go through several children to reach the desired data, since that the various *keys* to get there, are known.

In the same way that it is possible to write to the database through objects, it is also possible to collect the data in the same way. Given a path and an object class, Firebase can map the stored values into the attributes of the given class and return them as an object. The *getValue("Class")* is the method that allows to do this.

To get the values from the Firebase it is necessary to implement a *listener* of this SDK, there are two types of listeners: *ValueEventListner* and *ChildEventListener*. As the names themself indicate, the first refers to the values of a given node while the second refers to the children of a given node. The *ValueEventListner* allows two types of procedures: *addListenerForSingleValueEvent* and *addValueEventListener*. Both allow to get data, but the first is only called once and the second is called whenever there is a change in the database, this allows the application to be in real time, as is the case of this application. Figure 5.9 is an excerpt from the code that was used to collect *Firebase* data.



```
private void getCyclistKey() {
    cyclists.addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(DataSnapshot dataSnapshot) {
            String aux;
            for (DataSnapshot childSnapshot: dataSnapshot.getChildren()){
                 aux = childSnapshot.child(Auxilar.USERNAME).getValue(String.class);
                if(aux.equals(cyclistUsername)){
                    keyCyclist = childSnapshot.getKey();
                    setKeyCyclist(keyCyclist);
                    break;
                }
            }
        }
        @Override
        public void onCancelled(DatabaseError databaseError) {
       }
    });
  }
```

Figure 5.9 - Collecting data from Firebase

As in Figure 5.8, *cyclists* is the reference to the database, the *child* function allows access to a node, the variable *dataSnapshot* represents the current data in the location given by the reference and by the children. After being in the desired location, the value of the Firebase is retrieved using the *getValue(String.class)* function that returns a String.

5.4 Mobile Application Structure

The developed application can have three different user types: Admin, Coach and Cyclist. Every user has a username and password to log in. According to the user who logged in the application behaves differently. The main differences are found in the features of adding and editing users and training.

Figure 5.10 shows the activity diagram of the administrator.

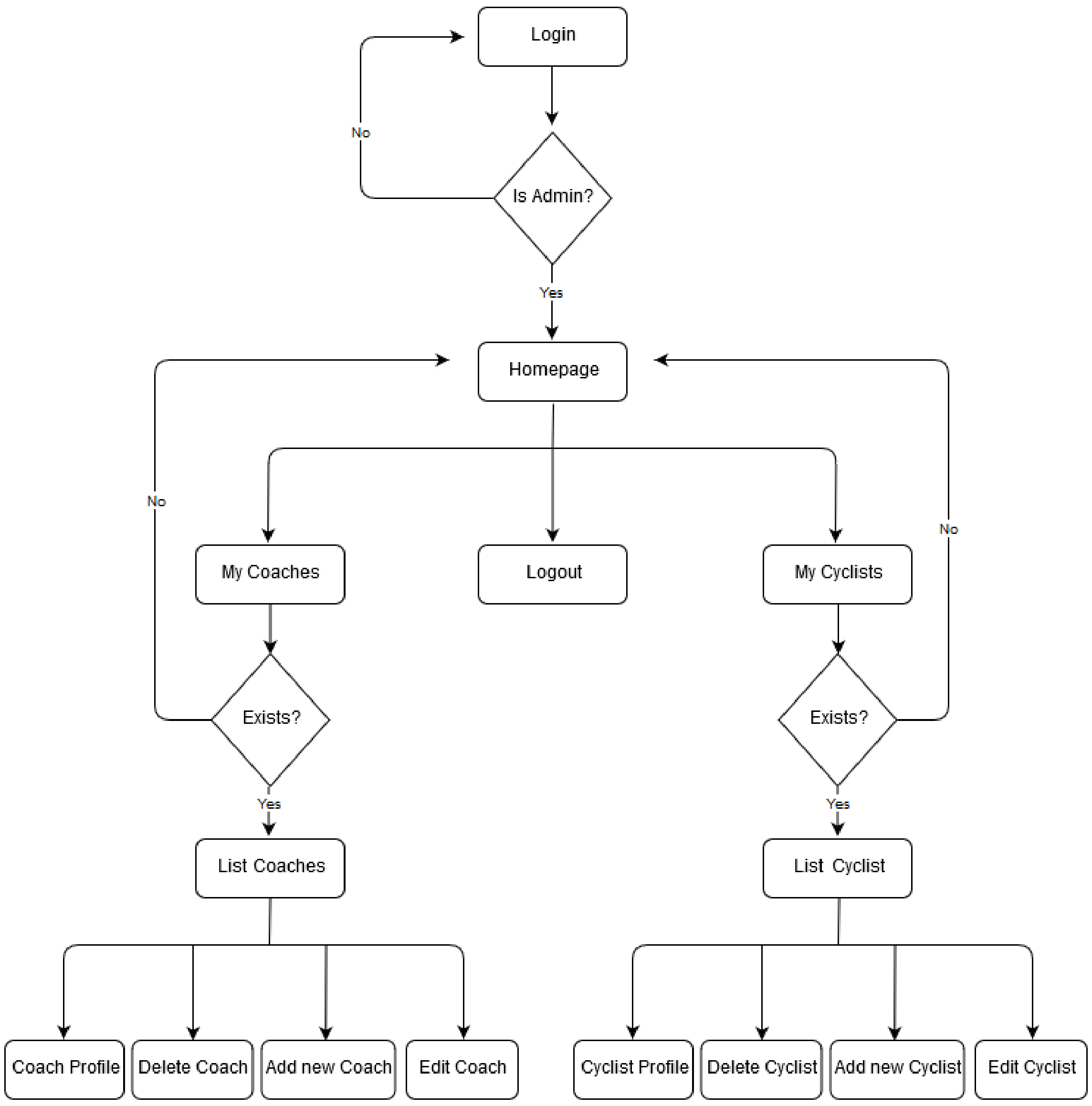


Figure 5.10 - Administrator's activity diagram

After a successful login, the admin will access his homepage where he has statistics about coaches, cyclists and trainings. It is also possible to access the complete list of coaches and cyclists registered in the database, from which the profile of each user can be accessed. The Admin has permissions to create cyclists and coaches.

Figure 5.11 illustrates the activity diagram coach.

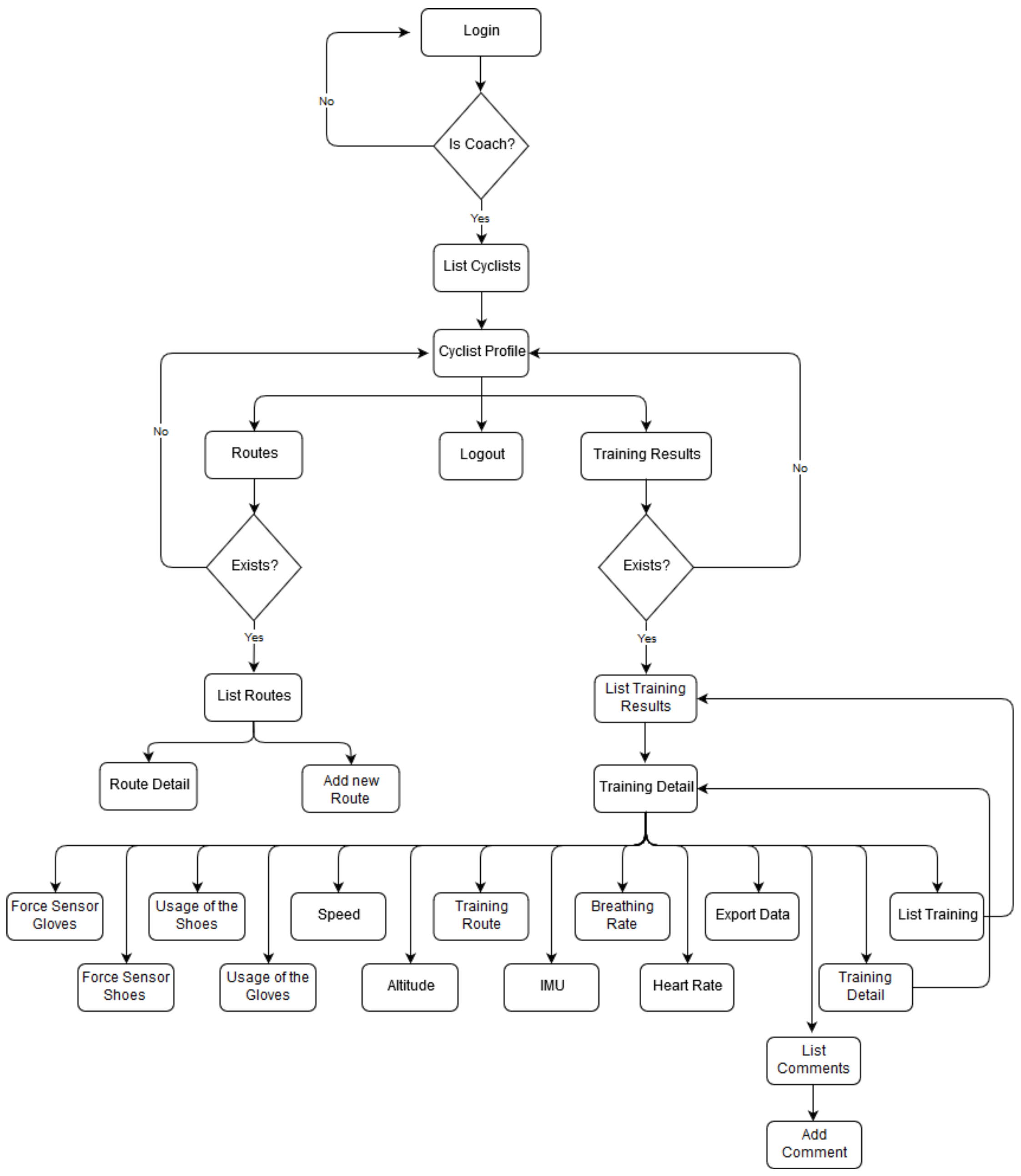


Figure 5.11 - Coach's activity diagram

Like Admin, the coach will also have to log in successfully in order to access the application. The coach accesses the complete list of cyclists assigned to him and then their profiles. When the coach is in the profile of a cyclist he has two options: access the planned routes or the list of trainings performed. In the List Routes activity, it is possible to see the map with the planned route or to plan a new route. By accessing a training in the training list, the coach can then see the results of the training, with statistical and graphical information, and the map of the training route. The coach can also comment the performed training.

In Figure 5.12 is represented the activity diagram for the cyclist.

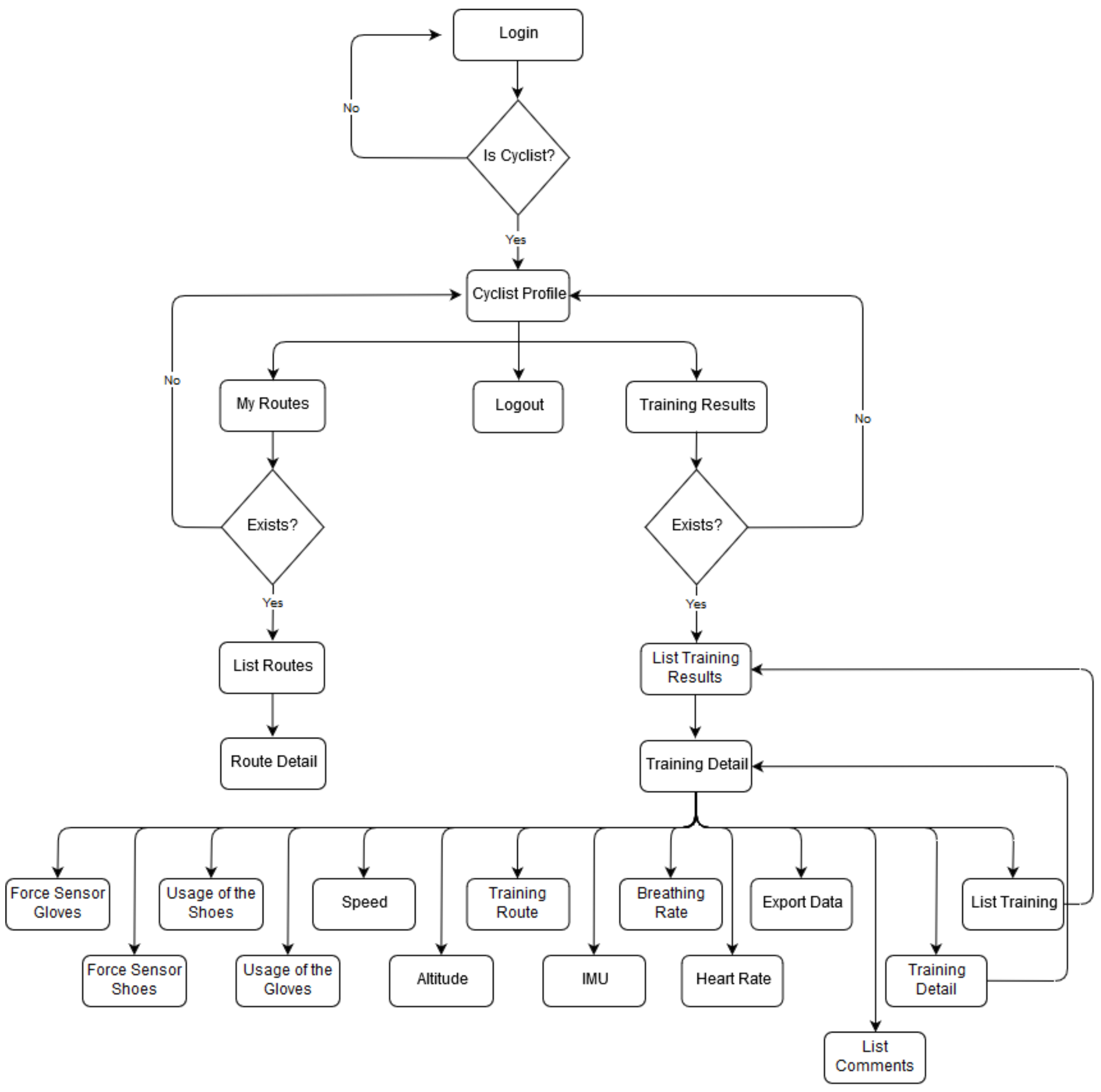


Figure 5.12 - Cyclist's activity diagram

After logging in, the cyclist accesses his profile and, from there, he has the same features as the coach, except for adding comments or new routes.

5.5 Application Design and Implementation

The application that supports this project was developed from scratch. In order for the user to have the best experience using it, the Android standards were used for the creation and development of the Graphical User Interface (GUI).

The main features of the application are described in this section. Figure 5.13 shows the logo of the application.

Figure 5.13 - Application logo

5.5.1 Authentication

All users of the system must be properly registered in the database and, in order to access the application, they must log in with their user and their password. In Figure 5.14, the login activity is represented.

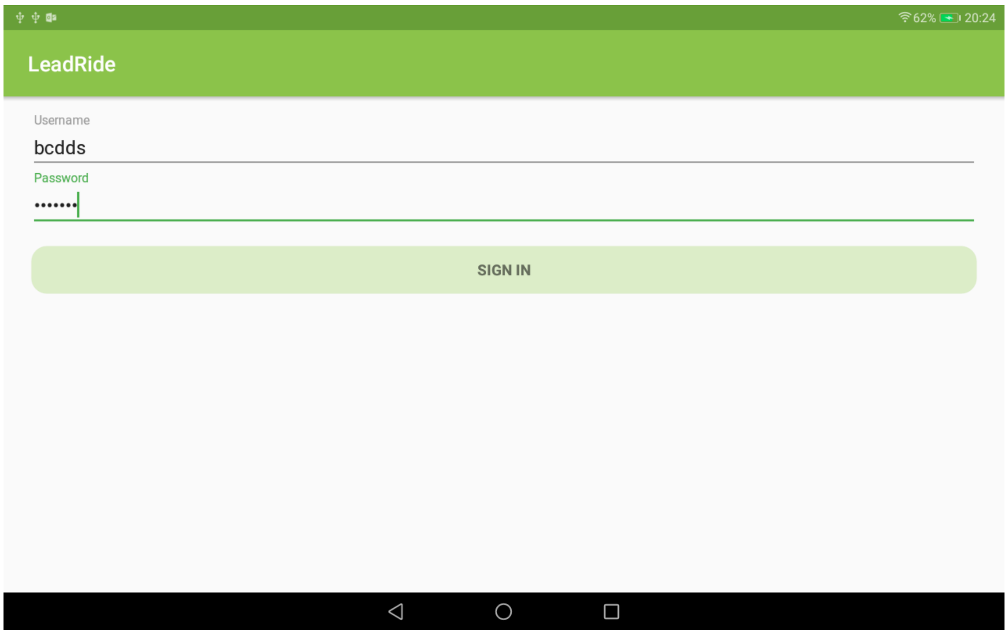
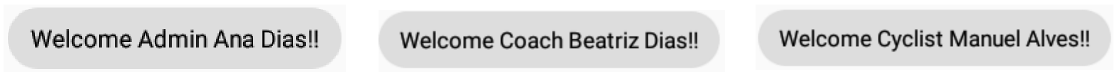


Figure 5.14 - Login Activity

New users are added by existing users, namely by the Admin or Coach user type. The Admin can create any type of user (Coach or Cyclist), the Coach can also create his own cyclists. When a user logs in, their user type is automatically detected, as shown in Figure 5.15.



Welcome Admin Ana Dias!!

Welcome Coach Beatriz Dias!!

Welcome Cyclist Manuel Alves!!

Figure 5.15 - Toast launched depending on the user type

5.5.2 Encryption

The password field is encrypted to ensure user protection. The Encryption Library [72] was used in conjunction with Android Studio, this library allows to encrypt and decrypt data through the application.

The password is never decrypted, when a new user is registered, his password is encrypted on the application side and stored in the database. When the user tries to log in, the text in the password field is encrypted and compared to the one in the database.

5.5.3 Data Chart

In order to analyze the training results, line and bar charts were created using the MP Android Chart Library [73]. This library allows for various types of graphics and various types of settings.

The developed application has different types of graphics:

- **Multiple line chart**: hand's and feet's force, and the IMU chart;
- **Single line chart**: speed, altitude, heart rate and breathing rate;
- **Bar chart**: percentage of usage of the hand and feet sensors.

All of these graphics have the zoom in, zoom out, scroll right and scroll left functionality for more detailed observation of the data. Figure 5.16 and Figure 5.17 shows two examples of the application charts.

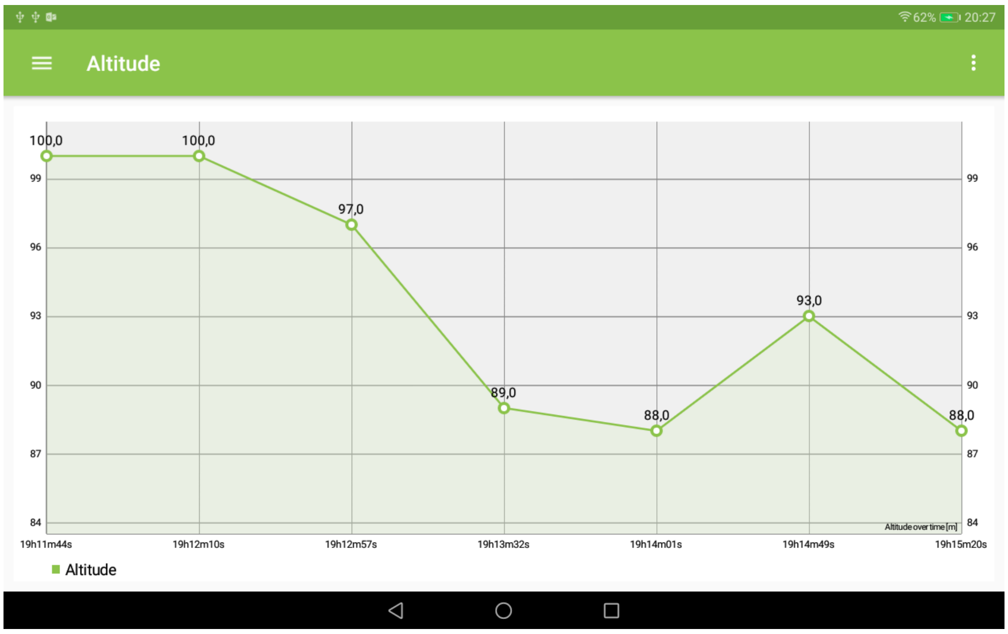


Figure 5.16 - Altitude - single line chart for a training session

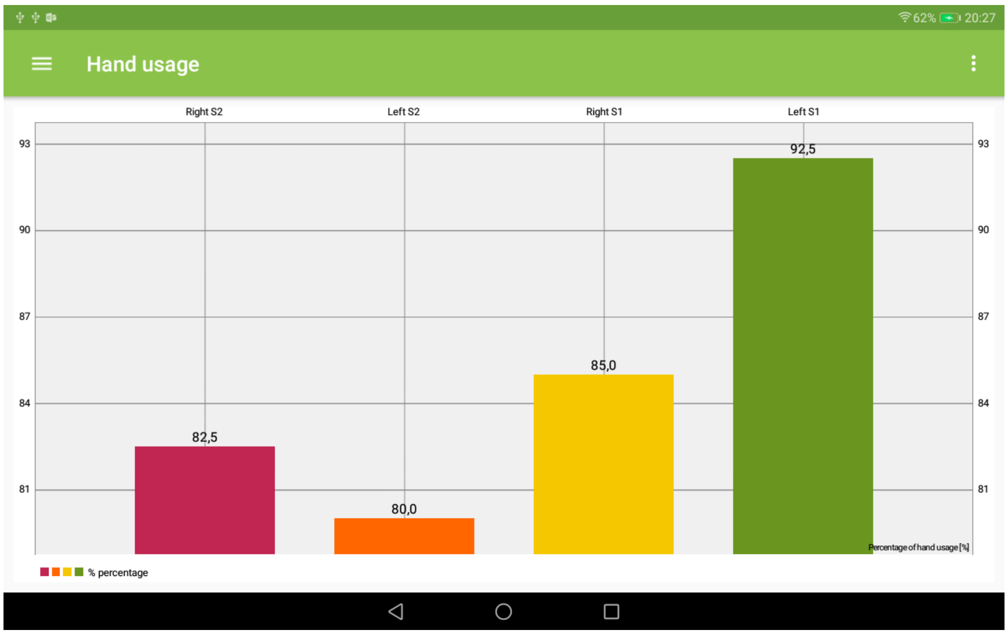


Figure 5.17 - Hand usage - bar chart for a training session

5.5.4 Maps

The application allows to view and create training routes and, in order for this to be possible, it was used the *Maps SDK for Android* [74] and the *Directions API* [75]. The first one is responsible for the maps view, works with coordinates, location and markers. By providing an origin coordinate, a destination coordinates and transport (cycling, driving or walking) *Directions API* is responsible for drawing a path between those two coordinates.

The application has been implemented to allow multiple coordinates because the goal is to draw the rote and not the closest or fastest path between two points. In the case of route planning, the user can choose up to three points on the map to draw a route. About the performed route, the application allows as many coordinates as the number recorded in the database.

Figure 5.18 represents an example of a route performed during a training.

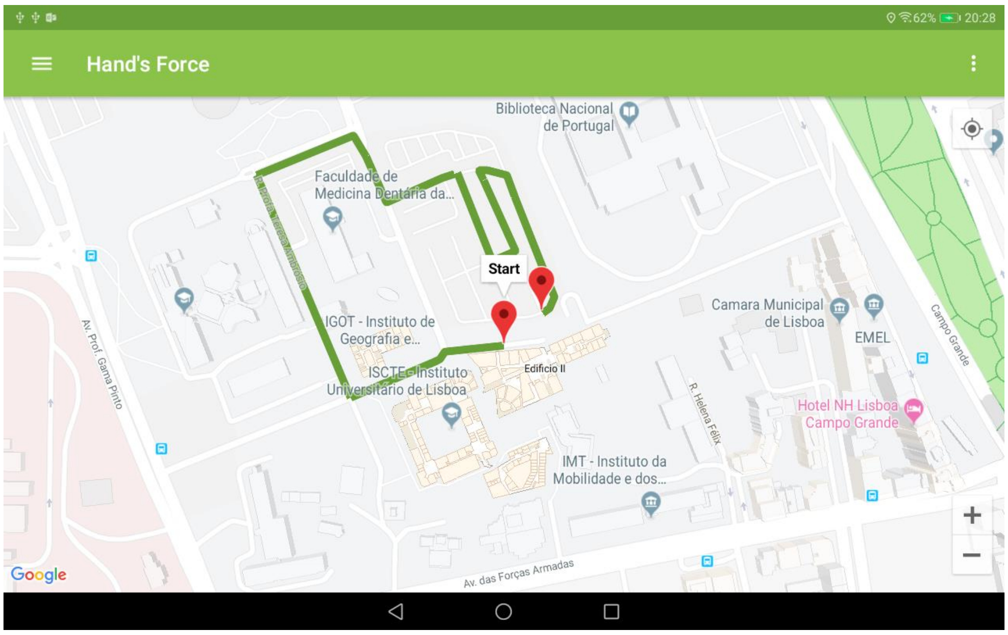

Hand's Force
Start
Biblioteca Nacional de Portugal
Faculdade de Medicina Dentária da...
IGOT - Instituto de Geografia e...
ISCTE - Instituto Universitário de Lisboa
Edifício II
IMT - Instituto da Mobilidade e dos...
Camara Municipal de Lisboa
EMEL
Hotel NH Lisboa Campo Grande
Av. Prof. Gama Pinto
R. Helena Félix
Campo Grande
Av. das Forças Armadas
Google

Figure 5.18 - Training route of 2nd October 2018 - *Maps SDK* and *Directions API*

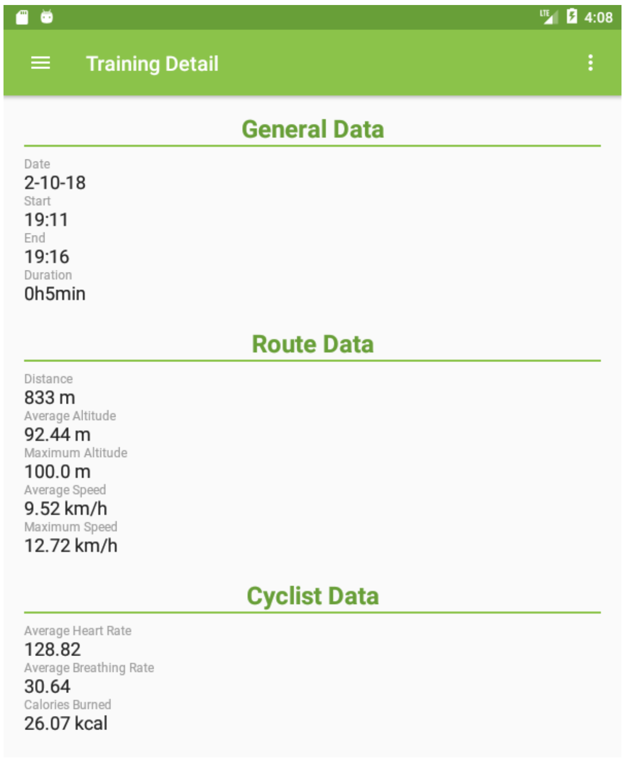
5.5.5 General Information

The application has an activity with the general information about the training (Figure 5.19). Some of this information is obtained directly from the database and other is calculated. This activity contains the following information:

- Date, start and end time, and duration;
- Traveled distance, average and maximum altitude and speed;
- Average heart and respiration rate;
- Calories burned;

The calories burned were calculated through the following equation (5.1).

$$Calories[kcal] = index \times weight\ [kg] \times duration\ [min] \tag{5.1}$$


4:08

Training Detail

General Data

Date
2-10-18
Start
19:11
End
19:16
Duration
0h5min

Route Data

Distance
833 m
Average Altitude
92.44 m
Maximum Altitude
100.0 m
Average Speed
9.52 km/h
Maximum Speed
12.72 km/h

Cyclist Data

Average Heart Rate
128.82
Average Breathing Rate
30.64
Calories Burned
26.07 kcal

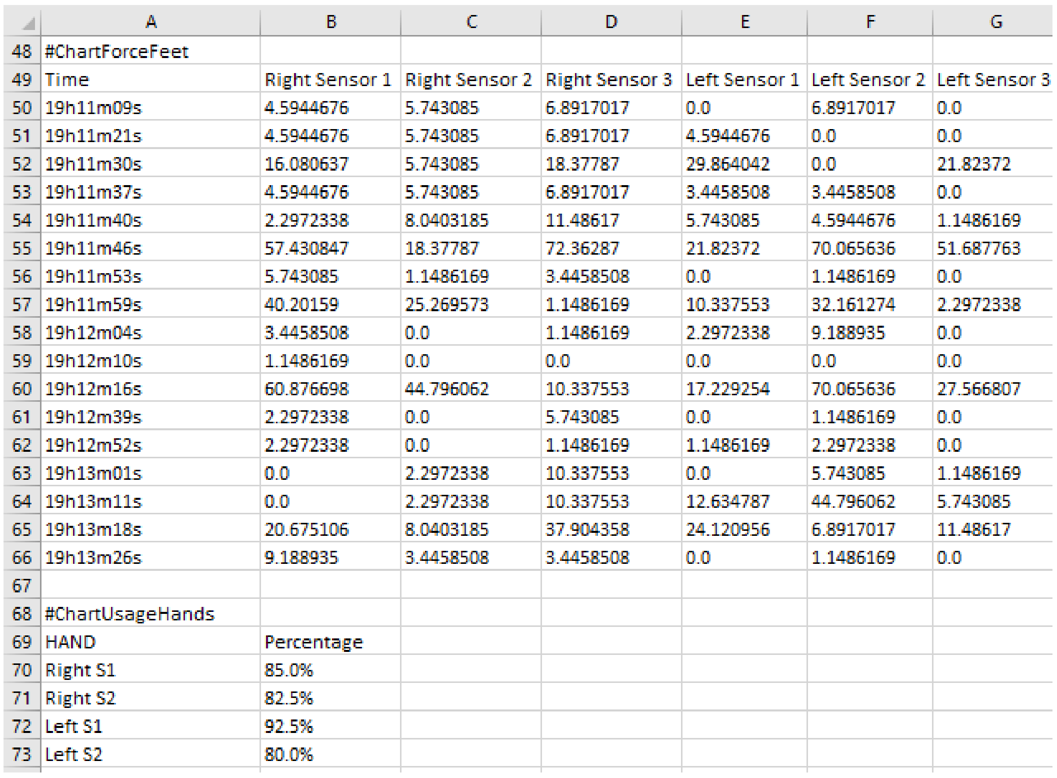
Figure 5.19 - Training's general information

5.5.6 Exporting the data

The application has a feature that allows to export the data of the charts to a CSV file format, which is compatible with Microsoft Excel. This feature allows to analyze the results offline. The data was organized into files with a specific structure as follows:

- The file's name contains the cyclist's name and the start date and time of the considered training, for example, "Manuel Alves_2-10-18_19:11.csv";
- Every chart is identified with the symbol "#" followed by the name of the chart, for example "#ChartForceHands".

Figure 5.20 shows an example of a part of an exported CSV file from the mobile application that was imported to the Microsoft Excel. This example shows the data of the force sensors of the feet and the data from the hand usage chart. The user can export the data when he has chosen a training and the button is available in the training menu.



	A	B	C	D	E	F	G
48	#ChartForceFeet						
49	Time	Right Sensor 1	Right Sensor 2	Right Sensor 3	Left Sensor 1	Left Sensor 2	Left Sensor 3
50	19h11m09s	4.5944676	5.743085	6.8917017	0.0	6.8917017	0.0
51	19h11m21s	4.5944676	5.743085	6.8917017	4.5944676	0.0	0.0
52	19h11m30s	16.080637	5.743085	18.37787	29.864042	0.0	21.82372
53	19h11m37s	4.5944676	5.743085	6.8917017	3.4458508	3.4458508	0.0
54	19h11m40s	2.2972338	8.0403185	11.48617	5.743085	4.5944676	1.1486169
55	19h11m46s	57.430847	18.37787	72.36287	21.82372	70.065636	51.687763
56	19h11m53s	5.743085	1.1486169	3.4458508	0.0	1.1486169	0.0
57	19h11m59s	40.20159	25.269573	1.1486169	10.337553	32.161274	2.2972338
58	19h12m04s	3.4458508	0.0	1.1486169	2.2972338	9.188935	0.0
59	19h12m10s	1.1486169	0.0	0.0	0.0	0.0	0.0
60	19h12m16s	60.876698	44.796062	10.337553	17.229254	70.065636	27.566807
61	19h12m39s	2.2972338	0.0	5.743085	0.0	1.1486169	0.0
62	19h12m52s	2.2972338	0.0	1.1486169	1.1486169	2.2972338	0.0
63	19h13m01s	0.0	2.2972338	10.337553	0.0	5.743085	1.1486169
64	19h13m11s	0.0	2.2972338	10.337553	12.634787	44.796062	5.743085
65	19h13m18s	20.675106	8.0403185	37.904358	24.120956	6.8917017	11.48617
66	19h13m26s	9.188935	3.4458508	3.4458508	0.0	1.1486169	0.0
67							
68	#ChartUsageHands						
69	HAND	Percentage					
70	Right S1	85.0%					
71	Right S2	82.5%					
72	Left S1	92.5%					
73	Left S2	80.0%					

Figure 5.20 - Exported CSV file example

This functionality has been included so that coaches have the possibility to do other types of analysis in addition to those that the application allows.

Chapter 6 Results and Discussion

This chapter includes the experimental results of the developed system. Tests were carried out on the consumption of the system and the system itself, collecting real data from the sensors implemented. The test of the system consumption was made using a power supply with an ammeter. The accuracy of the values read by the heart rate sensor were also tested.

Preliminary tests were carried out with only two sensors and then tests were performed on the complete system, with all sensors, connected to the database and mobile application.

6.1 Validation of the heart rate measurement capabilities

The accuracy of the pulse sensor has been proven through the use of medical equipment, Pulse Oximeter P-OX100 [76] (Figure 6.1).

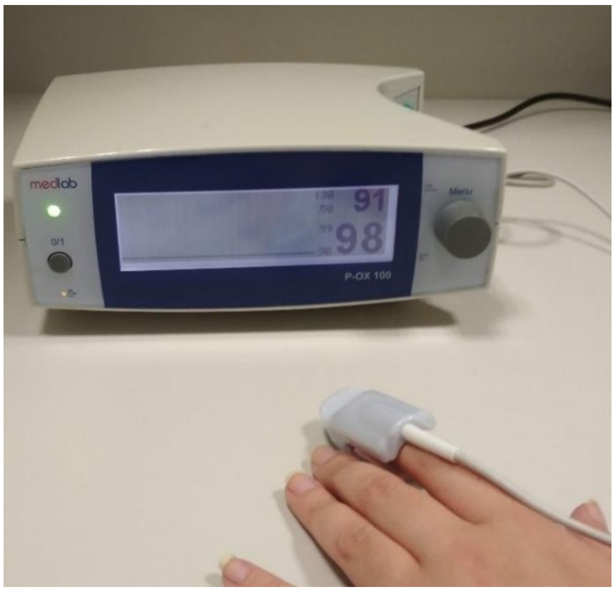


Figure 6.1 – Pulse Oximeter, P-OX100

Two tests were performed, by two subjects, one at rest and one immediately after physical activity, in which both sensors extracted values at exactly the same time. Table 6.1 shows the measurements taken.

Table 6.1 - Comparison between medical device and Pulse Sensor

	Pulse Oximeter	Pulse Sensor
In rest (Subject 1)	85	83
In rest (Subject 2)	82	80
After physical activity (Subject 1)	151	148
After physical activity (Subject 2)	124	122

The values are very similar, which proves the sensor's accuracy.

6.2 System Results

Several tests were carried out in laboratory and in the field conditions. The obtained results correspond to different sensors signals associated with cyclist training. Thus, in Figure 6.2 are presented the names associated to the considered force sensors and their setup on the hand and feet level.

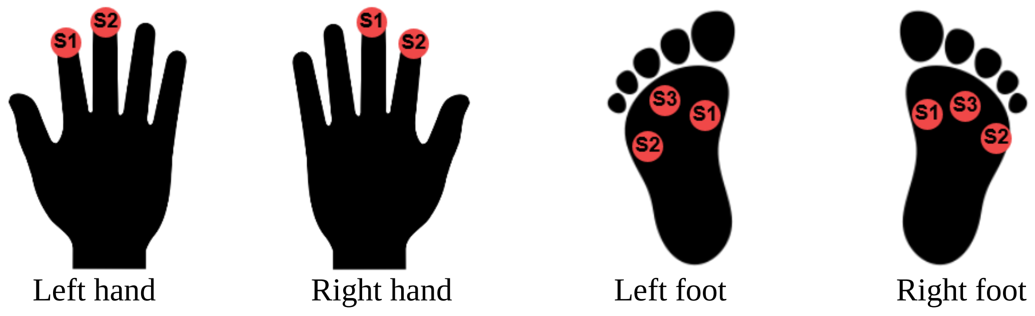


Figure 6.2 - Nomenclature of the sensors

6.2.1 Laboratory Tests

In the first phase, laboratory tests were performed in real time with only two end-nodes and the coordinator. These initial results have been published the article that is in Appendix A. In this test, the goal is to prove that it is possible to collect data from sensors, sending this data to the coordinator and, in turn, to the firebase.

The test was performed by a 23-year-old female subject. The participant is healthy and performed two training sessions of five minutes each, always performing the same straight-line course with a rest interval of two minutes. These tests were performed inside.

Figure 6.3, Figure 6.4 and Figure 6.5 show the results of this test.

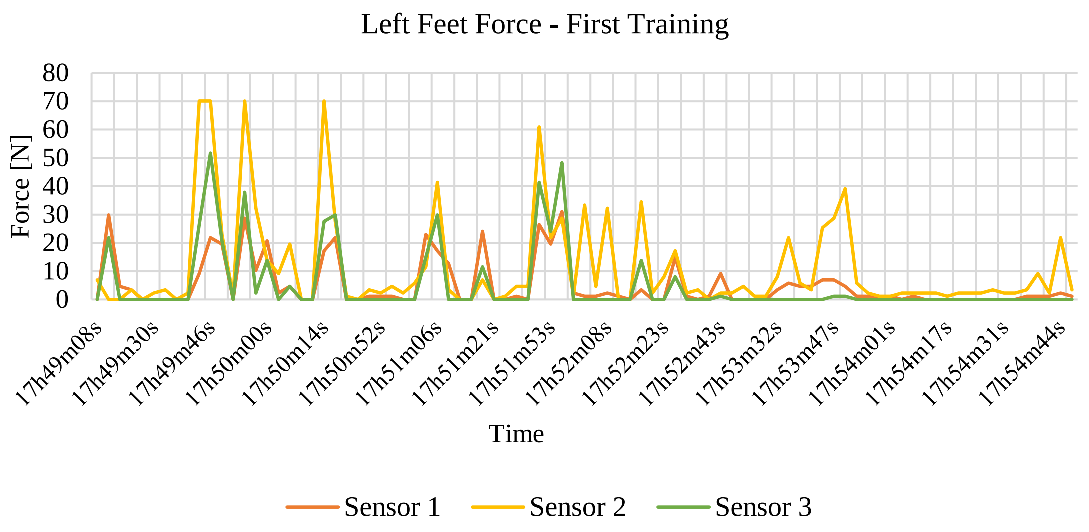


Figure 6.3 – Laboratory Test - First Training

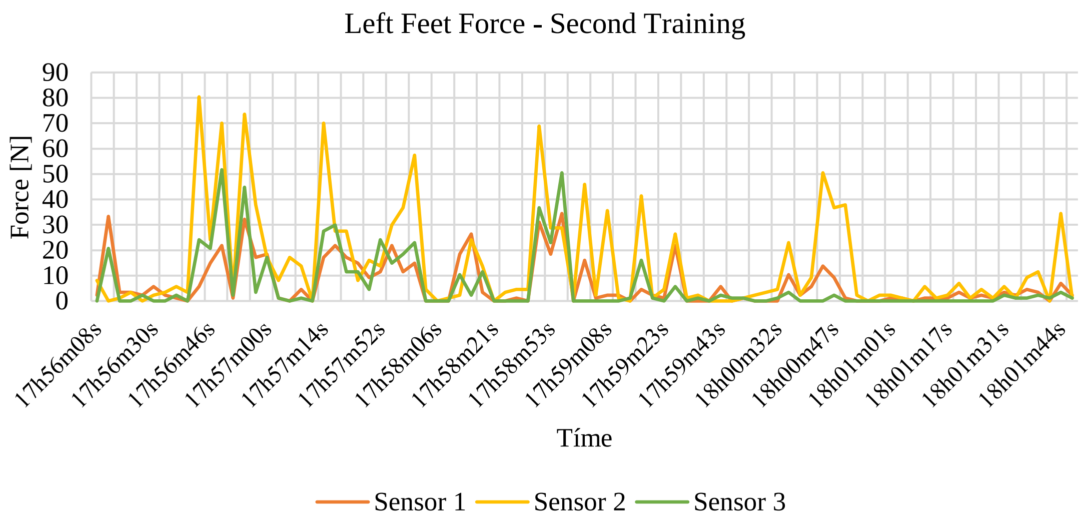


Figure 6.4 - Laboratory Test - Second Training

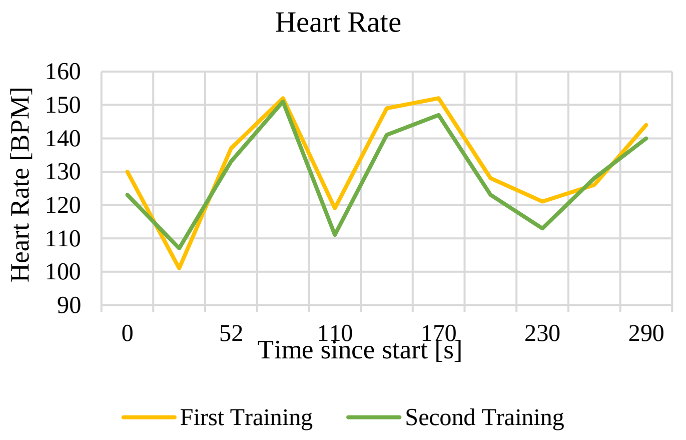


Figure 6.5 - Laboratory Test - BPM for both trainings

The results are from the left foot, obtained through three *Flexiforce* sensors, and the heartbeat, obtained through the PPG sensor placed in the ear. This data is sent from the end-nodes to the coordinator and then it is automatically placed in the firebase database. The information was exported from Firebase with JavaScript Object Notation (JSON) format and later imported into Microsoft Excel from where it was possible to obtain these charts.

The data obtained by the Pulse sensor is directly used to draw the chart, unlike the force chart. The values (val) from the force sensors needs to be converted into Newtons through the equation (6.1).

$$Force\ [N] = \frac{val \times 1023}{100} \times 11{,}45 \times 0{,}001 \times 9{,}80605 \qquad (6.1)$$

Through the charts in Figure 6.3, Figure 6.4 and Figure 6.5, it is already possible to compare the trainings. Considering that there are two trainings in a row, it is expected that in the second training the effort will be superior, this is visible through the increase of both strength and BPM values.

6.2.2 Field Tests

In this phase, tests were performed in a real environment with the objective of testing the system as a whole, the comfort and usability of the equipment. These tests were performed by a 20-year-old female subject. The three trainings made by this subject were on the same route, two of the trainings in the same day (training A and training B) and the third in the next day (training C).

Figure 6.6, Figure 6.7, Figure 6.8 and Figure 6.9 show the assembly of the sensors in the cyclist's body and coordinator on the bicycle and on the cyclist. Figure 6.10 shows the coordinator's final assembly.

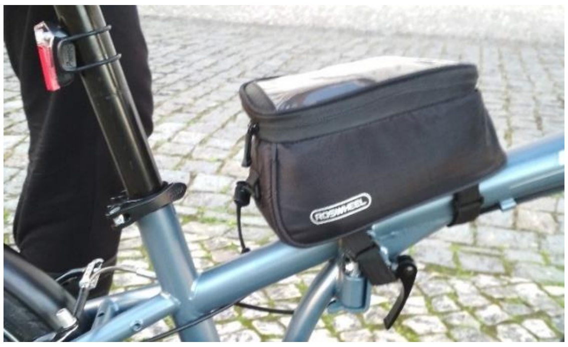


Figure 6.6 - Coordinator's assembly on the bicycle

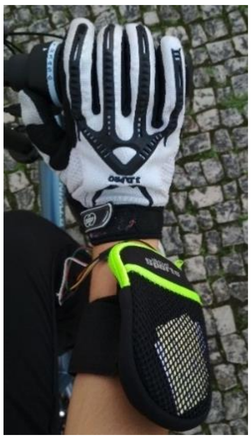


Figure 6.7 - Hand's end-node

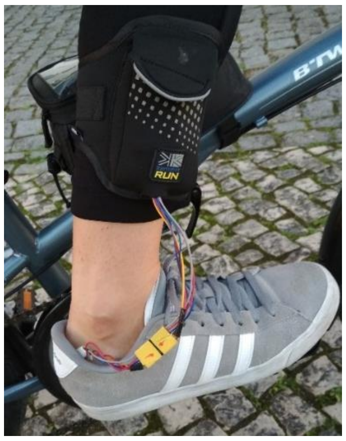


Figure 6.8 - Feet's end-node

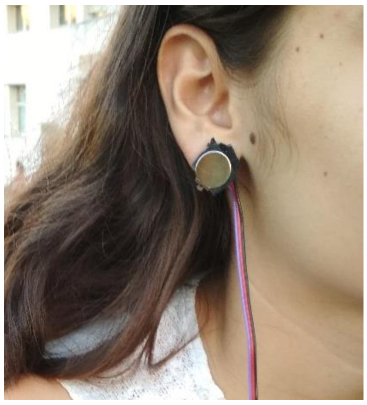


Figure 6.9 - Pulse sensor end-node

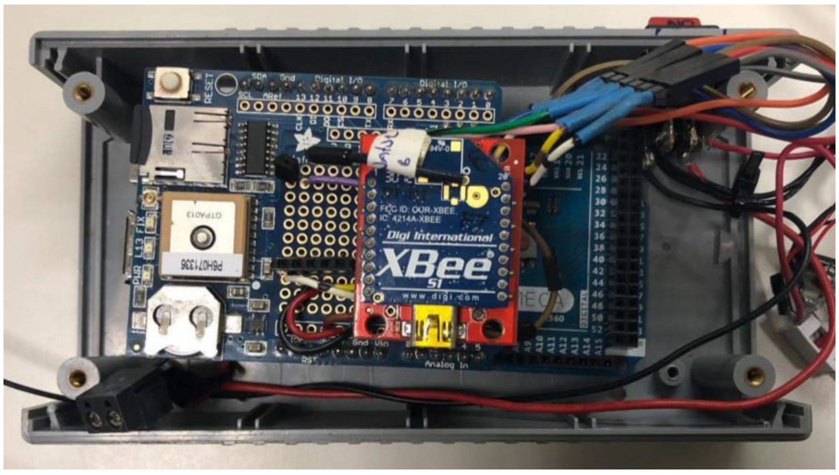


Figure 6.10 - Coordinator final assembly

From the training collected, it can be retrieved the route the cyclist did, as shown in Figure 6.11. The blue circle in the image will be referenced in all the charts, with a similar symbol in blue, in order to have an idea of which zone of the route the chart values correspond.

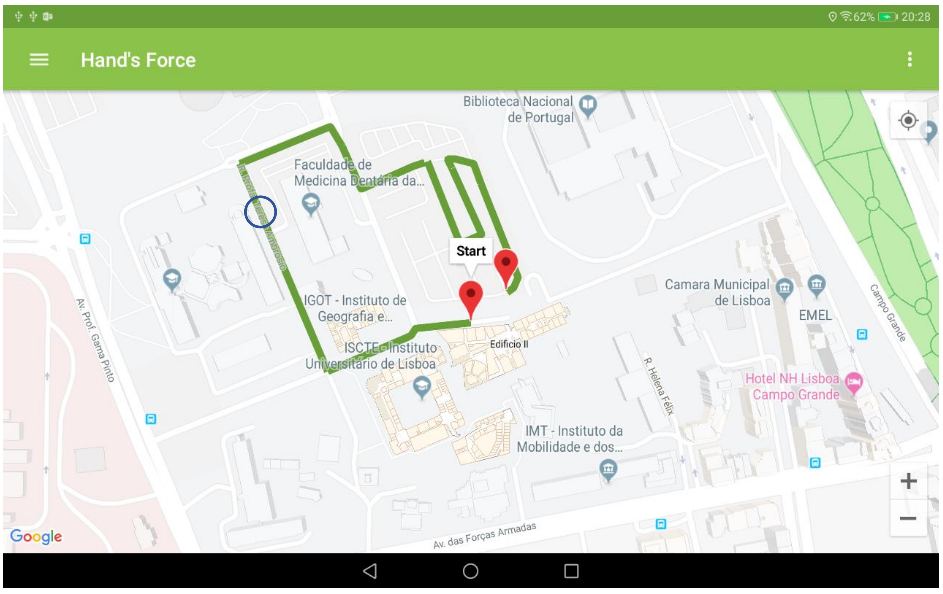

Hand's Force
Start
Biblioteca Nacional de Portugal
Faculdade de Medicina Dentária da...
IGOT - Instituto de Geografia e...
ISCTE- Instituto Universitário de Lisboa
Edifício II
Camara Municipal de Lisboa
EMEL
Hotel NH Lisboa Campo Grande
IMT - Instituto da Mobilidade e dos...
Av. Prof. Gama Pinto
R. Helena Félix
Campo Grande
Av. das Forças Armadas
Google

Figure 6.11 - Real training route for a 5 minutes training

The altitude is a very important factor in the training because it is directly related to all of the other variables collected. It influences the speed of the cyclist, the force applied on the sensors, the heart and respiration rate. The altitude chart is shown in Figure 6.12.

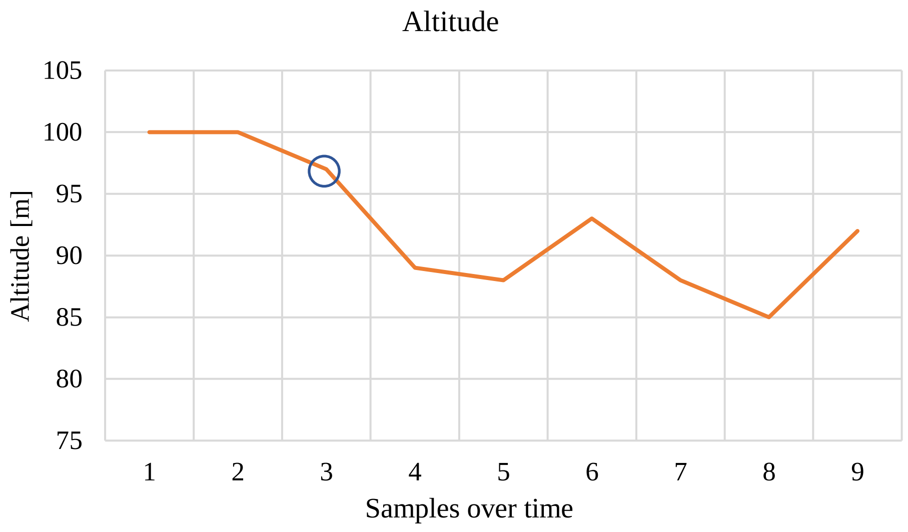


Figure 6.12 - Altitude chart for a 5 minutes training and the blue symbol represents point in the route where the cyclist is

As the coordinates and the altitude, the speed of the training is obtained by the GPS shield. In this case, the speed comes in knots and so it was converted to km/h with the following equation (6.2).

$$Speed\ [km/h] = Speed\ [knots] \times 1{,}852 \qquad (6.2)$$

Figure 6.13 shows the speed chart with lines of the training.

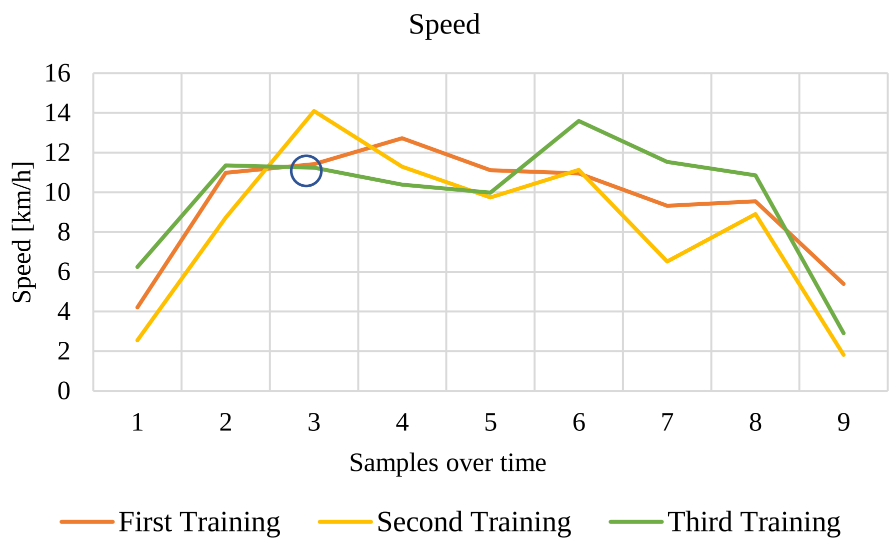


Figure 6.13 - Speed chart for a 5 minutes training and the blue symbol represents point in the route where the cyclist is

Analyzing the figure, it is possible to see that the velocity decreases when comparing the first training with the second training. Despite being the same route these two training only have a two-minute interval which is not enough for the cyclist to rest. As a consequence of the athlete being more tired, his speed is lower.

In all the three trainings, the speed in the initial phase increases and in the end decreases, this is the expected behavior. The starting speed tends to increase whereas at the end of training the speed tends to zero, this proves that the values are being measured correctly.

Comparing with the altitude chart, when considering sample from 1 to 4 it can be observed that the altitude decreases which make the speed increase. The exact opposite happens at the end of the training (sample 8 and 9).

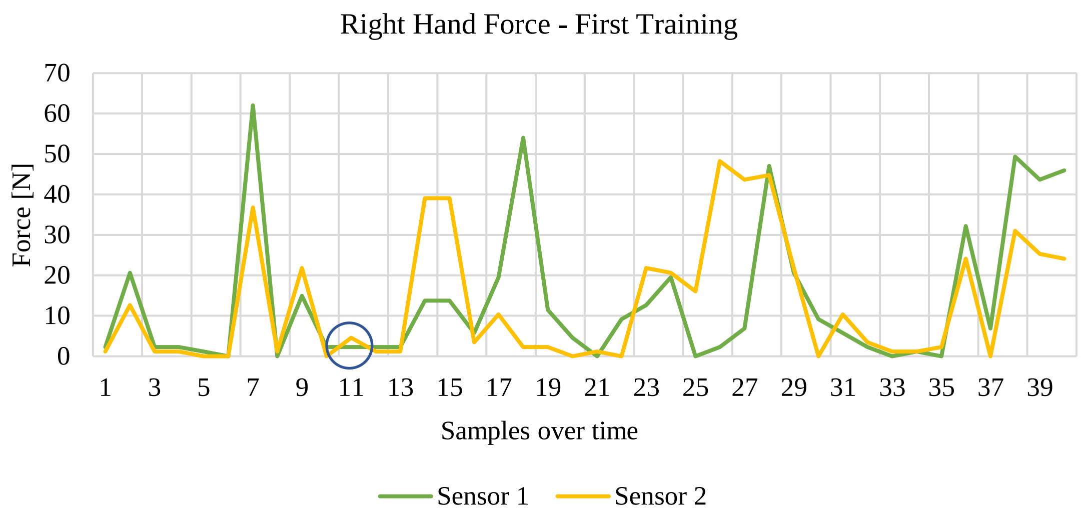


Figure 6.14 - Right-hand force chart for a 5 minutes training - First training (A) and the blue symbol represents point in the route where the cyclist is

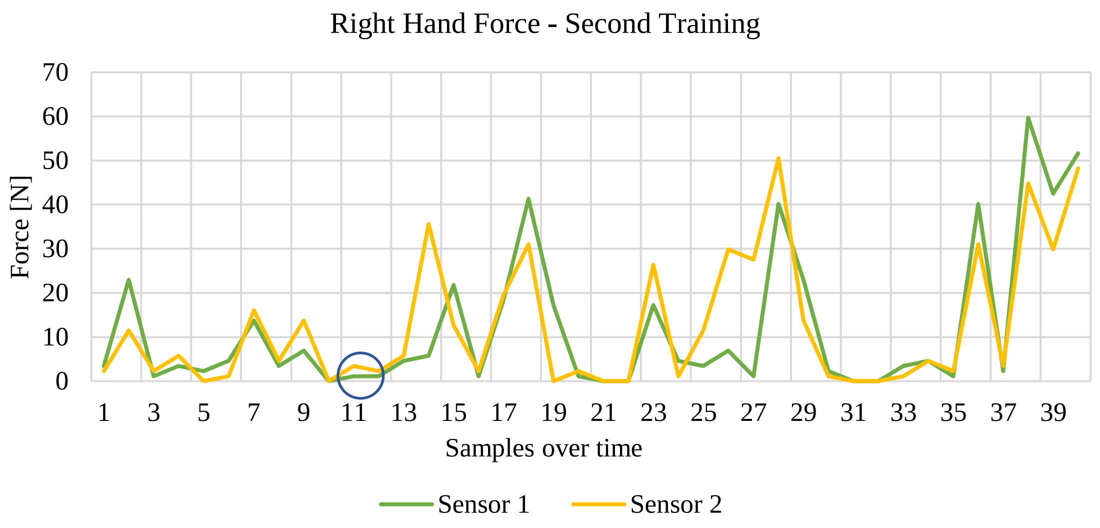


Figure 6.15 - Right-hand force chart for a 5 minutes training - Second training (B) and the blue symbol represents point in the route where the cyclist is

The charts in Figure 6.14 and Figure 6.15 in represent the training A and B. By observing them, it is possible to verify that the forces collected by sensor 1 (S1 - middle finger) and sensor 2 (S2 - indicator) show very similar but not equal behavior. This happens because they are not being measured on the same finger, so the force exerted will be different. It is also possible to observe that the force exerted on the sensor 1 is, in most cases, higher.

There are peaks of force that are explained by sharp curves or breaking along the course.

As in the first training, the values of S1 and S2 of the second training (Figure 6.15) are similar and follow the same behavior, this helps to prove the credibility of the developed system.

Figure 6.16 and Figure 6.17 show the results of the left hand for training A and B.

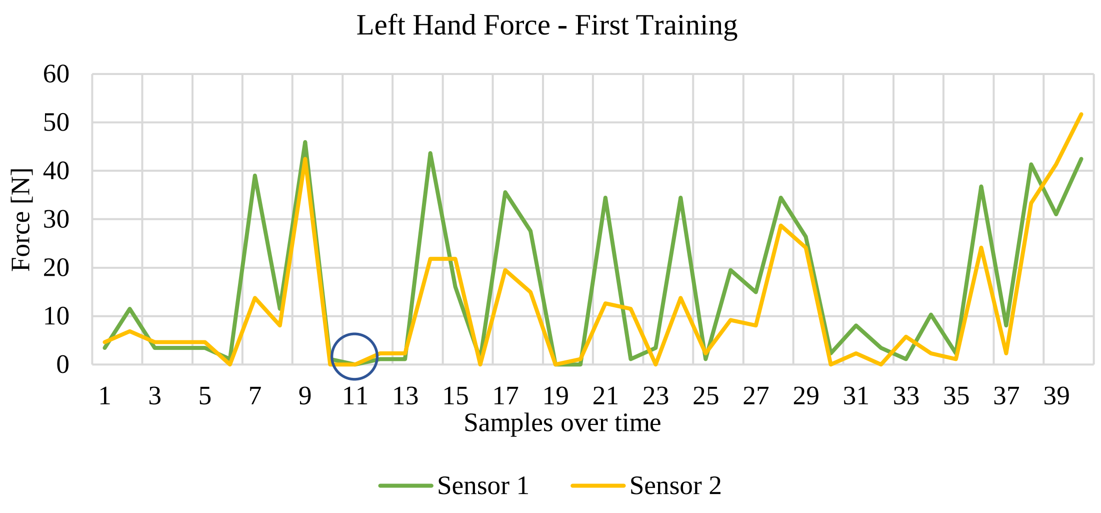


Figure 6.16 - Left-hand force chart for a 5 minutes training - First training (A) and the blue symbol represents point in the route where the cyclist is

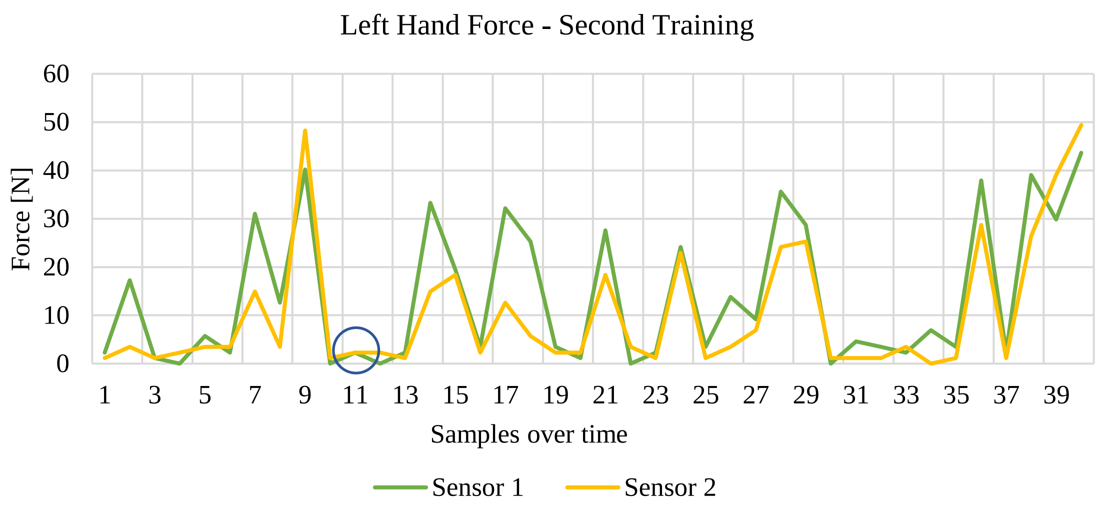


Figure 6.17 - Left-hand force chart for a 5 minutes training - Second training (B) and the blue symbol represents point in the route where the cyclist is

Analyzing these charts, it can be concluded that this hand is more used by the cyclist since there are more peaks comparing to the other hand. As the right hand, both sensors of the two trainings show similar behaviors.

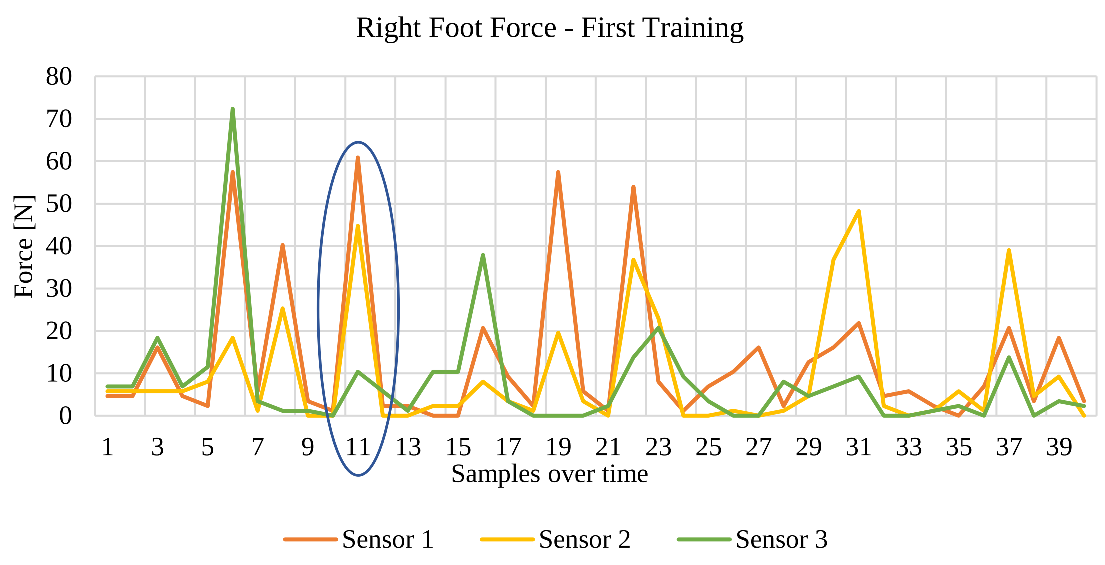


Figure 6.18 - Right foot force chart for a 5 minutes training - First training (A) and the blue symbol represents point in the route where the cyclist is.

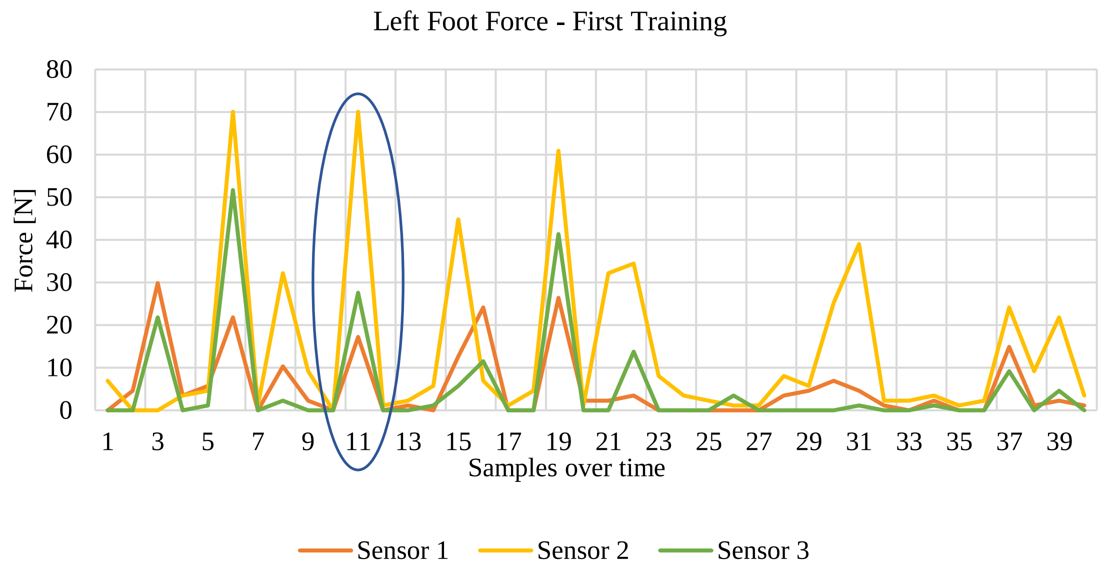


Figure 6.19 - Left foot force chart for a 5 minutes training - First training (A) and the blue symbol represents point in the route where the cyclist is.

Figure 6.18 and Figure 6.19 show the charts of the first training for the feet force sensors. As the hand's chart, the behavior between the several sensors is similar but almost never equal. This chart has more peaks than the hand's chart because the cyclist pedals more than the brakes.

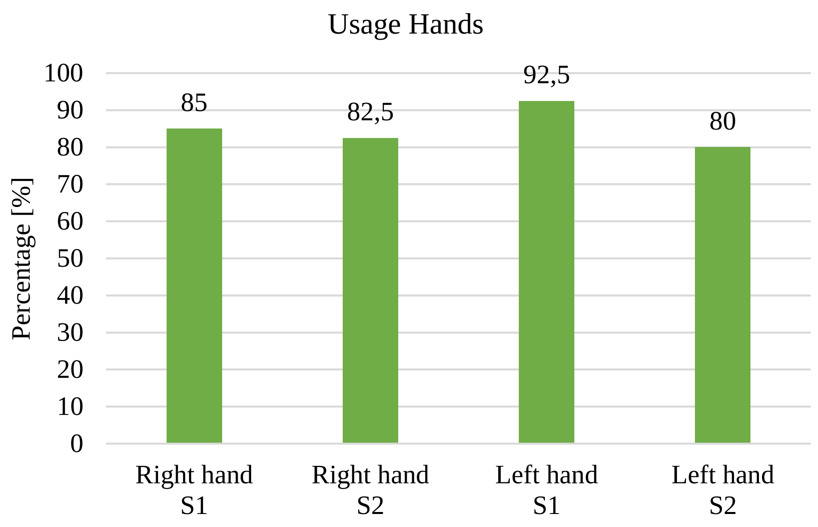


Figure 6.20 - Usage of the hand for a 5 minutes training.

The chart in Figure 6.20 shows the hand usage, this is calculated by counting the number of times the sensor is used, meaning, the value read by the sensor is different from zero.

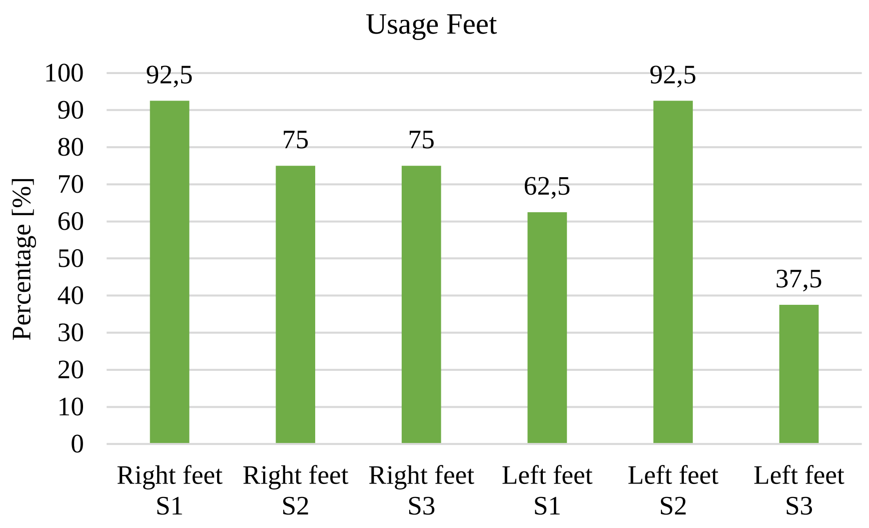


Figure 6.21 - Usage of the foot for a 5 minutes training.

The chart of Figure 6.21 was calculated in the same way as the previous chart, and it comes to prove that the most of times there are values to read, different from zero, by the sensors. This means that even though the cyclist is not always pedaling, he ends up always being in touch with the pedals and ends up making force, however small. The same can be concluded for the chart in Figure 6.20.

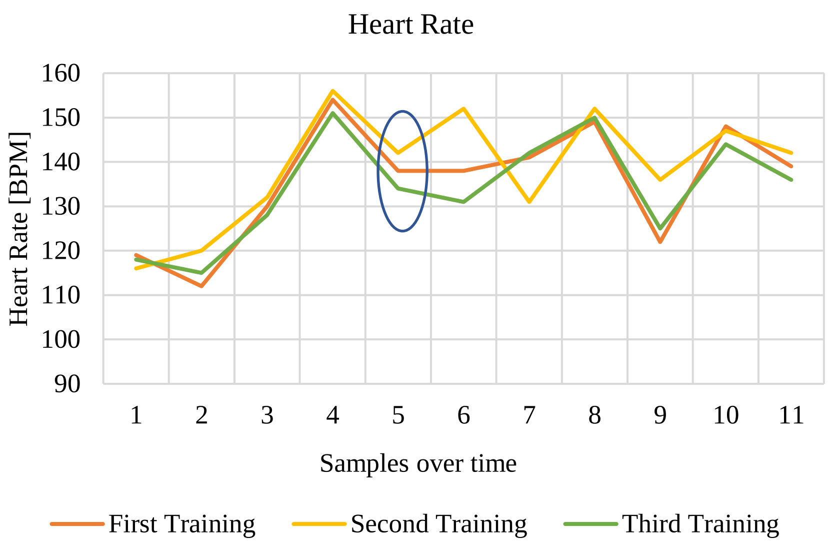


Figure 6.22 - Heart rate chart for a 5 minutes training and the blue symbol represents point in the route where the cyclist is.

The chart in Figure 6.22 shows really well the variations that are being analyzed, it is possible to see how similar are the values and the behavior of the first (A) and the last (C) training. It is also possible to observe that the heartbeats of training B have differences of values much greater than the other two trainings, which is the expected performance.

The normal heart Beatings Per Minute (BPM) of a resting adult is between 60 and 90 BPM [77], and moderate physical activity may reach 160 BPM. Comparing these values with the chart's values prove the accuracy of the PPG sensor.

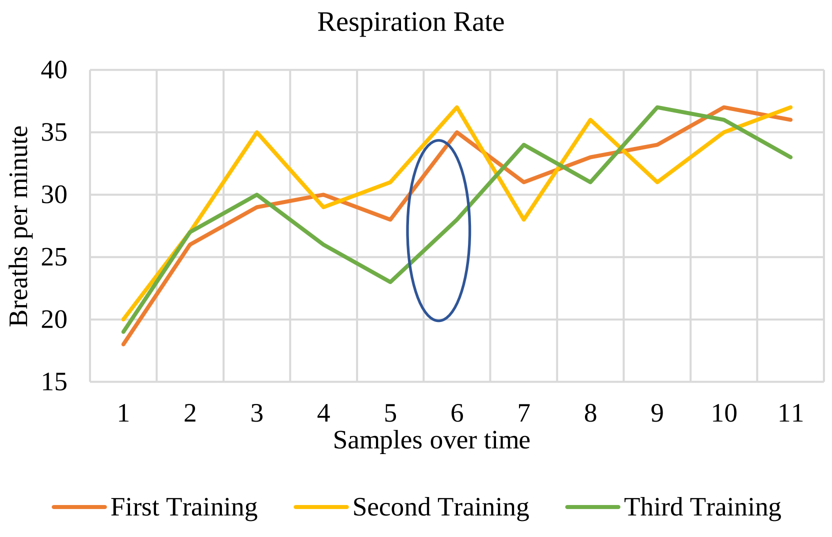


Figure 6.23 - Respiration rate chart for a 5 minutes training

By analyzing the chart in Figure 6.23, it is possible to see that, once again, the training A and training C have similar values, they start with normal values of respiration and those values increase along the training. The data of the second training starts with superior values comparing with the first, it also increases, and it is much more unstable.

The typical respiratory rate for a resting adult is about 12 to 20 breathing per minute, those numbers are higher when the person is in physical activity.

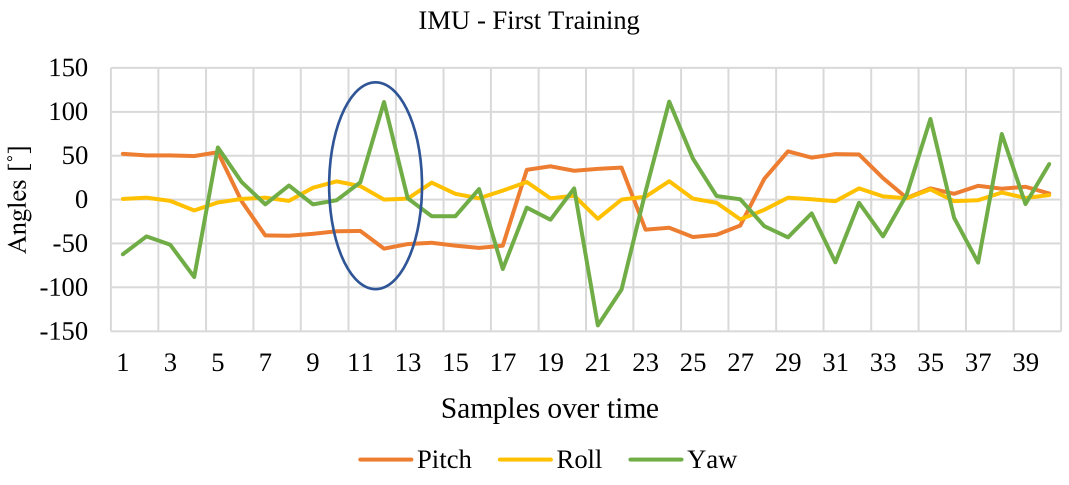


Figure 6.24 – Bicycle IMU chart for a 5 minutes training – First training (A) and the blue symbol represents point in the route where the cyclist is

Figure 6.24 shows the IMU chart, which represents the variation of the movement of the bicycle in yaw, pitch and roll. Yaw characterizes the movement of changing direction. Pitch represents the change of inclination when the cyclist is going down the pitch values is negative. The opposite happens when the cyclist is going up. Finally, the roll values show the inclination of the bicycle towards the floor.

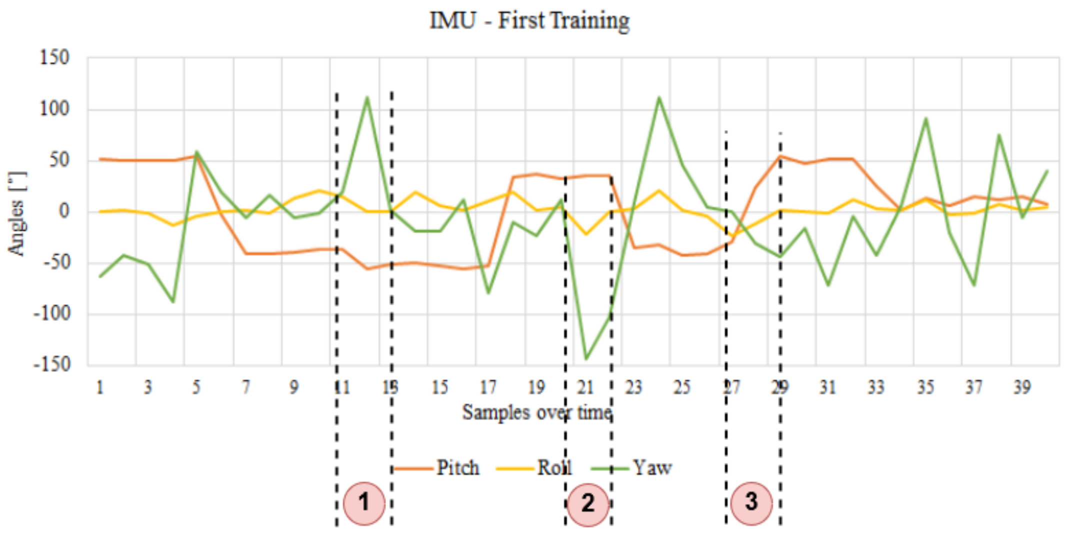


Figure 6.25 - Analysis of yaw, pitch and roll and the blue symbol represents point in the route where the cyclist is

Figure 6.25 is the analysis of yaw, pitch and roll. In part 1, it is going to be analyzed the yaw, part 2 the roll and part 3 the pitch. These measurements are important to know the position of the cyclist in its direction of movement, inclination and direction change. This allows the coach to know if the cyclist is making the right movement. For example, for high competition athletes, their movement is important in order to take advantage of the terrain or the wind.

The values in part 1 show that the cyclist made a turn to the left because there is a positive value. Observing the complete chart, it is possible to see that the roll does not change much, there are only two small peaks, one of them is the part 2. With this, it can be concluded that the performed route does not have many narrow curves.

In part 3 the pitch is being analyzed, it is verified that the values increase, this means that the subject was going up.

6.2.3 Data Visualization

Figure 6.26 and Figure 6.27, allows to compare the data visualization of the Android application and the chart generated by the Microsoft Excel. It is possible to see that both of the charts display the data correctly according to the received information.

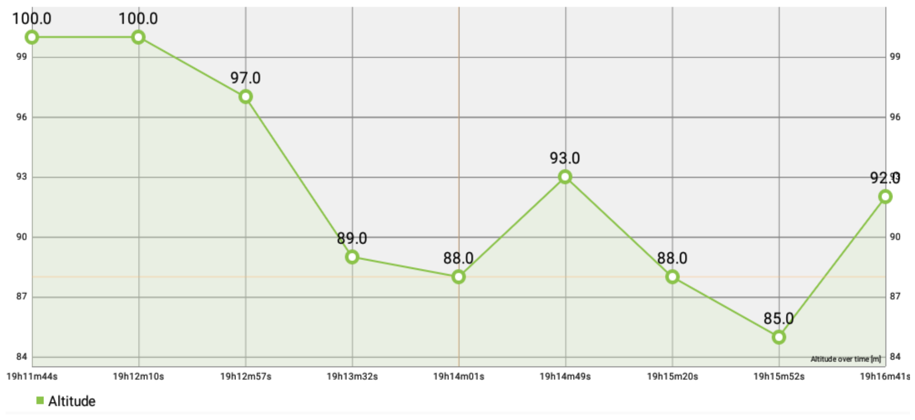


Figure 6.26 - Application Chart – Altitude for a 5 minutes training.

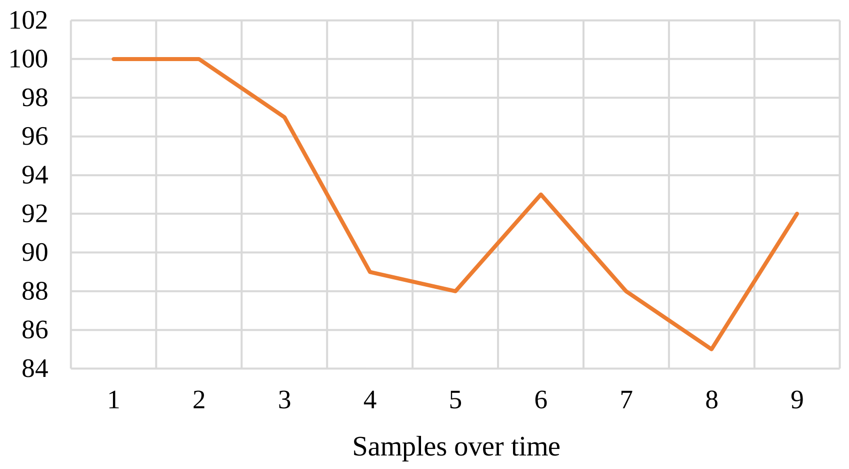


Figure 6.27 - Microsoft Excel Chart Altitude for a 5 minute training.

6.3 System Autonomy

This section describes the consumption of the developed system. To obtain these results, a power supply with an ammeter (Figure 6.28) was used, this allowed to measure the individual current of each end-node.

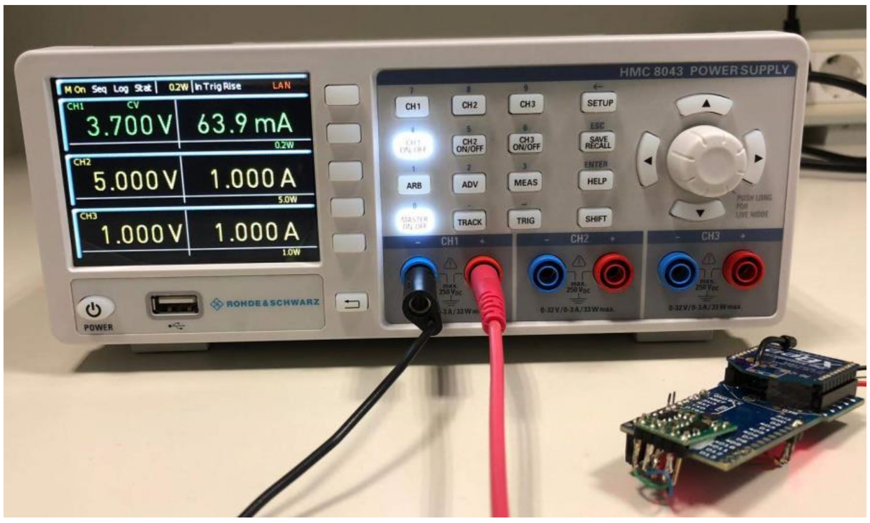


Figure 6.28 - Power supply with an ammeter.

Equation (6.3) was used to calculate the consumption.

$$Consumption\ [h] = \frac{\text{battery capacity [mAh]}}{current\ [mA]} \tag{6.3}$$

The batteries used are 1050 mAh. Table 6.2 includes the measurements and calculations described above.

Table 6.2 - Current and Autonomy of the end-nodes.

End-node	Current [mAh]	Autonomy [h]
Right Hand	56,8	18,49
Left Hand	57,2	18,36
Right Feet	80,1	13,11
Left Feet	59,1	17,77
IMU Bicycle	64,6	16,25
IMU Cyclist	64,3	16,31
Respiration and PPG sensor	57,7	18,19

The coordinator has two batteries of 5200 mA, the equation used is the one in (6.3). Table 6.3 shows these measurements.

Table 6.3 - Current and Autonomy of the coordinator.

	Current [mAh]	Autonomy [h]
Coordinator	279,7	37,18

Chapter 7 Conclusions and Future Work

7.1 Conclusions

An IoT system for cyclist activity monitoring including the performance evaluation was designed and implemented, which was the main goal of this dissertation.

The system capabilities were tested and the obtained data allows to compare the results coming from different trainings, that may be used to extract differences between the training sessions. With the analyzed data it is possible for the cyclist to have feedback, based on the following values: how many times the cyclist braked, his heart rate, his breathing rate and his maximum speed. Thus, the athlete becomes more motivated and may consider a different strategy to obtain better results.

The system provides also metrics that may be evaluated by the coach through charts and training's summary. There are charts about the force applied to the breaks and the pedals, the altitude of the training course, heart rate, breathing rate, IMU values and speed. Thus, the coach may find out where the athletes are failing and what points to improve.

Another objective was to extend the capabilities of the previous smart bike prototype, which was also achieved. New sensors, important to the performance of the cyclist, were added, such as the heart rate and breathing rate sensors. A different solution for the database was considered, using NoSQL with Firebase it was possible to have a more modern, flexible and real-time database. A new mobile application was developed with more information about the training, such as new charts with the heart and rate breathing rate and the possibility of planning a training route. In terms of data access, the latency is very low, either sending or receiving the data, either from the Arduino or the mobile application.

At the technological level, the system presents a user-friendly graphical interface including statistical charts for easy data visualization. The coach can follow the cyclist's evolution for days, months or years considering the implemented IoT architecture.

It was proven the accuracy of the values collected, by comparing the breathing and heart rate values obtained with the reference values. It was also tested the veracity of the pulse sensor by comparing its values to a medical device. It was also evaluated the behavior of the several sensors along the performed trainings.

To summarize, this system can acquire data from the sensors and send it to the real-time database. This database keeps the history of the trainings and then the coach or the cyclist can access it through the mobile application.

7.2 Contributions

This dissertation's main contributions are:

- A practical approach regarding communication protocols for WSN;
- Design and implementation of the sensor network for the monitoring of cyclists;
- The development of a data storage system based on a non-relational database using Firebase;
- Development and implementation of a mobile application for data visualization.

The developed work and the initial results were included in the article and the poster in Appendix A, which were accepted and presented at the IEEE EPE 2018 international conference, October 18-19, Iasi, Romania. The article will be published in IEEEXplorer.

Dias A. and Postolache O., "Cyclist Performance Assessment based on WSN and Cloud Technologies", EPE 2016, IASI Romania 2018 International Conference on Electrical an Power Engineering.

Appendices B and C also include a User Manual (B) and a Technical Manual (C) to help to understand better the design and implementation of the developed project.

7.3 Future Work

Despite being a fairly complete system, improvements can still be made, both software and hardware. Further testing of the system may be performed in different situations and conditions than those presented in Chapter 6.

The implementation of a central switch to turn on / off all end-nodes.

The use of a shield that uses an internet card instead of wi-fi are some of the improvements that can be made.

The induction of new metrics, for example, relating the results with the weather conditions.

Smaller PCB circuits can be created so that it is not uncomfortable.

The system can also be improved at the level of data processing, instead of being processed in the application could be processed at the level of Firebase, avoiding spending resources of the mobile phone. Firebase has a feature that allows this, by writing functions on the database level with the Firebase *Cloud Functions*.

References

[1] P. P. Ray, “Internet of Things for Sports (IoTSport): An architectural framework for sports and recreational activity,” *Int. Conf. Electr. Electron. Signals, Commun. Optim. EESCO 2015*, 2015.

[2] J. Rodian, “StiboSystems.” [Online]. Available: http://blog.stibosystems.com/how-data-and-iot-are-changing-sports. [Accessed: 10-Dec-2017].

[3] S. K. Gharghan, R. Nordin, and M. Ismail, “Design Consideration of an Energy Efficient Wireless Sensor Network for High Performance Track Cycling,” *2014 Int. Conf. Inf. Sci. Appl.*, p. 5, 2014.

[4] D. Evans, “The Internet of Things - How the Next Evolution of the Internet is Changing Everything,” *CISCO white Pap.*, no. April, pp. 1–11, 2011.

[5] S. Li, L. Da Xu, and S. Zhao, “The internet of things: a survey,” *Inf. Syst. Front.*, vol. 17, no. 2, pp. 243–259, 2015.

[6] A. Al-fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, “Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications Ala,” vol. 17, no. 4, pp. 2347–2376, 2015.

[7] P. P. Ray, “Generic Internet of Things architecture for smart sports,” *2015 Int. Conf. Control Instrum. Commun. Comput. Technol. ICCICCT 2015*, pp. 405–410, 2016.

[8] Cisco, “The Internet of Everything and the Connected Athlete : This Changes … Everything,” pp. 1–10, 2013.

[9] D. Springer, “Network-Centric Service Oriented Enterprise,” 2008, pp. 157–209.

[10] A. Deshpande, C. Montiel, and L. McLauchlan, “Wireless Sensor Networks - A Comparative Study for Energy Minimization Using Topology Control,” *2014 Sixth Annu. IEEE Green Technol. Conf.*, pp. 44–48, 2014.

[11] G. Kiokes *et al.*, “Performance evaluation of a communication protocol for vital signs sensors used for the monitoring of athletes,” *Int. J. Distrib. Sens. Networks*, vol. 2014, 2014.

[12] D. G. Reina, S. L. Toral, F. Barrero, N. Bessis, and E. Asimakopoulou, “The Role of Ad Hoc Networks in the Internet of Things: A Case Scenario for Smart Environments,” vol. 460, no. February, pp. 89–113, 2013.

[13] J. S. Lee and Y. C. Huang, “ITRI ZBnode: A ZigBee/IEEE 802.15.4 platform for wireless sensor networks,” *Conf. Proc. - IEEE Int. Conf. Syst. Man Cybern.*, vol. 2, pp. 1462–1467, 2007.

[14] H. Kemis, N. Bruce, W. Ping, T. Antonio, L. B. Gook, and H. J. Lee, “Healthcare monitoring application in ubiquitous sensor network: Design and implementation based on pulse sensor with arduino,” *Inf. Sci. Serv. Sci. Data Min. (ISSDM), 2012 6th Int. Conf. New Trends in.*, pp. 34–38, 2012.

[15] R. Negra, I. Jemili, and A. Belghith, “Wireless Body Area Networks: Applications and Technologies,” *Procedia Comput. Sci.*, vol. 83, pp. 1274–1281, 2016.

[16] J. Yusuf Khan, M. R. Yuce, G. Bulger, and B. Harding, “Wireless body area network (WBAN) design techniques and performance evaluation,” *J. Med. Syst.*, vol. 36, no. 3, pp. 1441–1457, 2012.

[17] “iTunes Apple - Cyclemeter Cycling Running GPS.” [Online]. Available: https://itunes.apple.com/br/app/cyclemeter-cycling-running-gps/id330595774?mt=8. [Accessed: 10-Dec-2017].

[18] "GooglePlay - Cyclemeter GPS - Cycling, Running, Mountain Biking." [Online]. Available: https://play.google.com/store/apps/details?id=com.abvio.meter.cycle&hl=pt_PT. [Accessed: 10-Dec-2017].

[19] "Sports Tracker." [Online]. Available: http://www.sports-tracker.com/. [Accessed: 10-Dec-2017].

[20] "Strava." [Online]. Available: https://www.strava.com/. [Accessed: 03-Jan-2018].

[21] "Endomondo." [Online]. Available: https://www.endomondo.com/. [Accessed: 03-Jan-2018].

[22] "Mountain Bike Runtastic." [Online]. Available: https://www.runtastic.com/en/apps/mountainbike. [Accessed: 03-Jan-2018].

[23] "GooglePlay - The Mountain Bike Runtastic." [Online]. Available: https://play.google.com/store/apps/details?id=com.runtastic.android.mountainbike.lite&hl=pt_PT. [Accessed: 03-Jan-2018].

[24] J. P. Broker and J. D. Crawley, "Advanced Sport Technologies: Enhancing Olympic Performance," no. Figure 1, pp. 323–327, 2001.

[25] T. Ribeiro, O. Postolache, and P. Passos, "Performance assessment for mountain bike based on WSN and cloud technologies," *Proc. 2016 Int. Conf. Expo. Electr. Power Eng. EPE 2016*, no. Epe, pp. 380–386, 2016.

[26] S. Armstrong, "Wireless connectivity for health and sports monitoring: a review," 2007.

[27] C. G. D. I. Fotiadis, A. Lika, and A. Stafylopatk, "A Weareble Intelligent System for Monitoring Health Condition and Rehabilitation of Running Athletes," pp. 276–279.

[28] N. S. A. Zulkifli, F. K. C. Harun, and N. S. Azahar, "XBee Wireless Sensor Networks for Heart Rate Monitoring in Sport Training," 2012.

[29] F. Q. Al-Khalidi, R. Saatchi, D. Burke, H. Elphick, and S. Tan, "Respiration rate monitoring methods: A review," *Pediatr. Pulmonol.*, vol. 46, no. 6, pp. 523–529, 2011.

[30] A. Baca, P. Kornfeind, E. Preuschl, S. Bichler, M. Tampier, and H. Novatchkov, "A server-based mobile coaching system," *Sensors*, vol. 10, no. 12, pp. 10640–10662, 2010.

[31] K. Pandia, O. T. Inan, G. T. A. Kovacs, and L. Giovangrandi, "Extracting respiratory information from seismocardiogram signals acquired on the chest using a miniature accelerometer," *Physiol. Meas.*, vol. 33, no. 10, pp. 1643–1660, 2012.

[32] A. Dinh, Y. Choi, and S.-B. Ko, "A Heart Rate Sensor Based On Seismocardiograpfy For Vital Sign Monitoring Systems," pp. 665–668, 2011.

[33] "Connectivity of the Internet of Things." [Online]. Available: https://learn.sparkfun.com/tutorials/connectivity-of-the-internet-of-things. [Accessed: 12-Jan-2018].

[34] "11 Internet of Things (IoT) Protocols You Need to Know About." [Online]. Available: https://www.rs-online.com/designspark/eleven-internet-of-things-iot-protocols-you-need-to-know-about. [Accessed: 12-Jan-2018].

[35] J. S. Lee, Y. W. Su, and C. C. Shen, "A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi," *IECON Proc. (Industrial Electron. Conf.*, vol. 4, no. 6, pp. 46–51, 2014.

[36] E. Ferro and F. Potortì, "Bluetooth and Wi-Fi Wireless Protocols: a survey and a comparison," no. February, pp. 12–26, 2005.

[37] S. K. Gharghan, R. Nordin, and M. Ismail, "Energy-efficient ZigBee-based wireless sensor network for track bicycle performance monitoring," *Sensors*

(Switzerland), vol. 14, no. 8, pp. 15573–15592, 2014.
[38] "What is Arduino?" [Online]. Available: https://www.arduino.cc/en/Guide/Introduction. [Accessed: 19-Jan-2018].
[39] "Arduino Compare board specs." [Online]. Available: https://www.arduino.cc/en/Products/Compare. [Accessed: 11-Jan-2018].
[40] "MySQL." [Online]. Available: https://www.mysql.com/. [Accessed: 22-Jan-2018].
[41] "JSON Introduction." [Online]. Available: https://www.w3schools.com/js/js_json_intro.asp. [Accessed: 26-Oct-2018].
[42] A. L. Gonçalves, "Desenvolvimento de um aplicativo Android utilizando banco de dados não-relacional para organização e controle de presença de um time de futebol," 2016.
[43] A. Sganzerla and R. Lummertz, "Direto ao Ponto – App colaborativo do transporte coletivo usando o Firebase."
[44] "Documentação | Firebase." [Online]. Available: https://firebase.google.com/docs/. [Accessed: 19-Jul-2018].
[45] J. Ulrich, "Android will keep dominating iOS (despite iPhone X)," 2017. [Online]. Available: https://thenextweb.com/contributors/2017/12/29/android-will-keep-dominating-ios-despite-iphone-x-6-tech-predictions-2018/. [Accessed: 19-Jul-2018].
[46] "Mobile Operating System Market Share Worldwide | StatCounter Global Stats," 2018. [Online]. Available: http://gs.statcounter.com/os-market-share/mobile/worldwide/#monthly-201706-201806. [Accessed: 18-Sep-2018].
[47] "Conheça o Android Studio | Android Developers." [Online]. Available: https://developer.android.com/studio/intro/. [Accessed: 19-Jul-2018].
[48] "LEITOR RFID RC522 + CARTÃO + TAG 13.56MHZ." [Online]. Available: http://automatizacg.com/shop/leitor-rfid-rc522-cartao-tag-13-56mhz. [Accessed: 19-Jul-2018].
[49] "Arduino - Products." [Online]. Available: https://www.arduino.cc/en/Main/Products. [Accessed: 19-Jul-2018].
[50] G. Liebel, N. Marko, M. Tichy, A. Leitner, and J. Hansson, "Model-based engineering in the embedded systems domain: an industrial survey on the state-of-practice," *Softw. Syst. Model.*, vol. 17, no. 1, pp. 91–113, 2018.
[51] M. Barr and A. Massa, *Programming Embedded Systems: With C and GNU Development Tools*. 2006.
[52] "Force Sensors | Tekscan." [Online]. Available: https://www.tekscan.com/product-group/embedded-sensing/force-sensors. [Accessed: 23-Jul-2018].
[53] Tekscan, "FlexiForce ® Sensors | User Manual."
[54] "O que é condicionamento de sinal? - National Instruments." [Online]. Available: http://www.ni.com/white-paper/10630/pt/. [Accessed: 26-Oct-2018].
[55] SparkFun Electronics, "Analog to Digital Conversion," p. 470, 2010.
[56] T. Ribeiro, O. Postolache, and P. Passos, "Performance assessment for mountain bike based on WSN and cloud technologies," ISCTE-IUL, 2016.
[57] "Pololu - MinIMU-9 v3 Gyro, Accelerometer, and Compass (L3GD20H and LSM303D Carrier)." [Online]. Available: https://www.pololu.com/product/2468. [Accessed: 25-Jul-2018].
[58] "I2C - learn.sparkfun.com." [Online]. Available: https://learn.sparkfun.com/tutorials/i2c. [Accessed: 26-Oct-2018].
[59] N. A., K. T.B., and G. J., *No Basic Navigational Mathematics, Reference Frames*

and the Earth's Geometry. In: Fundamentals of Inertial Navigation, Satellite-based Positioning and their Integration. 2013.

[60] Z. Lin, X. Linlin, L. Fanming, and C. Yi, "Application of UKF for MEMS IMUs and fluxgate sensors based attitude and heading reference system of carriers," *ICIEA 2007 2007 Second IEEE Conf. Ind. Electron. Appl.*, vol. 0, pp. 2278–2283, 2007.

[61] A. Ave, H. Fauzan, S. R. Adhitya, and H. Zakaria, "Early detection of cardiovascular disease with photoplethysmogram(PPG) sensor," *Proc. - 5th Int. Conf. Electr. Eng. Informatics Bridg. Knowl. between Acad. Ind. Community, ICEEI 2015*, pp. 676–681, 2015.

[62] "Pulse Sensor - SEN-11574 - SparkFun Electronics." [Online]. Available: https://www.sparkfun.com/products/11574. [Accessed: 26-Jul-2018].

[63] "EMFIT | R-Series sensor." [Online]. Available: https://www.emfit.com/copy-of-l-series. [Accessed: 04-Sep-2018].

[64] GlobalTop Technology Inc., "FGPMMOPA6H GPS Standalone Module Data Sheet," no. 16, pp. 1–37, 2011.

[65] R. Weinstein, "RFID: A technical overview and its application to the enterprise," *IT Prof.*, vol. 7, no. 3, pp. 27–33, 2005.

[66] "Iduino Yun Shield - Geeetech Wiki." [Online]. Available: http://www.geeetech.com/wiki/index.php/Iduino_Yun_Shield. [Accessed: 26-Jul-2018].

[67] "Set up your XBee devices." [Online]. Available: https://www.digi.com/resources/documentation/digidocs/90001526/containers/cont_setup_devices.htm?tocpath=Set up your XBee devices%7C_____0. [Accessed: 31-Jul-2018].

[68] "Ajuda Download do manual - Alcatel MW40V | NOS." [Online]. Available: http://www.nos.pt/particulares/ajuda/equipamentos-servicos/Pages/ajuda-equipamentos.aspx?page=device/nos/alcatel-mw40v/topic/manual-do-utilizador/download-do-manual/1. [Accessed: 01-Aug-2018].

[69] "Painéis | Android Developers." [Online]. Available: https://developer.android.com/about/dashboards/#OpenGL. [Accessed: 18-Sep-2018].

[70] H. G. C. Ferreira, E. Dias Canedo, and R. T. De Sousa, "IoT architecture to enable intercommunication through REST API and UPnP using IP, ZigBee and arduino," *Int. Conf. Wirel. Mob. Comput. Netw. Commun.*, pp. 53–60, 2013.

[71] S. Powers, J. Peek, T. O'Reilly, M. Loukides, and M. Kosta Loukides, *Unix Power Tools*, 3rd ed. 2002.

[72] "GitHub - simbiose/Encryption: Encryption is a simple way to encrypt and decrypt strings on Android and Java project." [Online]. Available: https://github.com/simbiose/Encryption. [Accessed: 14-Oct-2018].

[73] P. Jahoda, "MPAndroidChart: A powerful Android chart view / graph view library." [Online]. Available: https://github.com/PhilJay/MPAndroidChart. [Accessed: 14-Oct-2018].

[74] "Overview | Maps SDK for Android | Google Developers." [Online]. Available: https://developers.google.com/maps/documentation/android-sdk/intro. [Accessed: 14-Oct-2018].

[75] "Developer Guide | Directions API | Google Developers." [Online]. Available: https://developers.google.com/maps/documentation/directions/intro. [Accessed: 14-Oct-2018].

[76] "P-OX100." [Online]. Available: http://www.medlab-

gmbh.de/english/products/patientmonitors/pox100/index.html. [Accessed: 26-Oct-2018].

[77] A. Shaffler and N. Menche, *Medicina Interna e Cuidados de Enfermagem*.

Appendix A - Scientific Articles

Article: **Cyclist Performance Assessment Based on WSN and Cloud Technologies**. This article has been accepted and presented at the IEEE EPE 2018 international conference, October 18-19, Iasi, Romania and will be published in IEEEXplorer.

10th ANNIVERSARY
EPE 2018
10th INTERNATIONAL CONFERENCE AND EXPOSITION ON ELECTRICAL AND POWER ENGINEERING
October 18-19, 2018
IASI, ROMANIA
"Gheorghe Asachi" Technical University of Iasi
Faculty of Electrical Engineering
IEEE
Romania Section
SETIS

Organized by the Faculty of Electrical Engineering of Iasi and SETIS Association, the EPE Conference is now a tradition, being confirmed as an important international event in the electrical engineering area. Starting in 1999 with the 1st edition, EPE reached today the 10th anniversary.

Cyclist Performance Assessment based on WSN and Cloud Technologies

Ana Catarina Dias,
Instituto de Telecomunicações, IT-IUL
ISCTE-IUL, Lisbon, Portugal
acdds@iscte-iul.pt

Octavian Postolache ISCTE-IUL
Instituto de Telecomunicações, IT-IUL
Lisbon, Portugal
opostolache@lx.it.pt

***Abstract* -The mobility in the big cities become a big issue and the usage of bicycles represent one of the solutions together new services such as Bike Share services that motivate even more the users. At the same time, more users are practicing the sports that involve the bike usage. In this context, the article distributed sensing system for cyclist's training monitoring. With the support of a Wireless Sensor Network (WSN), connected to the Internet and using a set of smart sensors as WSN nodes mounted on the bicycle and the cyclist himself, it is possible to obtain data that will help to improve the cyclist's performance. The coach can monitor and evaluate the performance to improve their training sessions. The health status condition during training it is also monitored using cardiac assessment sensor. The information coming from the WSN nodes is uploaded, through an internet connection, to the Firebase platform. The data can be accessed through an Android mobile application, that also allows the configuration of the training route. With the inclusion of these technologies, the coach and the athlete may analyze the performance and compare with the previous training results. The coach may establish a new training session according to the athlete needs. The effectiveness of the proposed system was experimentally tested and several results are included in the paper.**

Keywords—cyclist monitoring; wireless sensor network; IoT, Firebase; Android application

I. Introduction

Nowadays there is an increasing need to apply technologies to everything around us whether to make our daily basis easier or just for leisure. The Internet of Things (IoT) is a technological concept with a great development, with the potential to alter and replace various methods of classical sports training and information acquirement. The advantage of performing measurements of physical parameters using internet-connected devices is that the results of these measurements are immediately available in electronic format, which allows the users to access the information anyplace and anytime.

Athletes, whether professionals or amateurs, always want to evolve in the activities they practice. There is a need to use the new technologies so that cyclists and their coaches can evaluate the progress in an objective way. In a project that took place in Sweden, the movement of a skier was analyzed, with the help of Smartphones the skier was able to develop his strategy ability to go faster [1]. Today, most professional sports athletes are carrying some sort of device, be it a smartwatch, an embedded sensor in their clothing, or sports gear that captures their performance [2]. Our work follows this theme, IoT based sport and athlete analysis, particularly cycling.

This work intends to develop a complete analysis system, using multiple sensors, like force and heart rate monitor sensors, and various technologies. The goal is to increase performance and improve the practice of cycling considering also the health condition of the athletes. One of these technologies is the Wireless Sensor Network which uses different wireless sensors placed on the bicycle and the cyclist to collect information during training. This information is then processed to produce statistics.

The goal of the work is to develop a complete system for cycling assessment considering some previous work developed by IT-IUL research group, adding new sensors dedicated to the cyclist's physical performance monitoring. A mobile application will also be developed to present the information acquired by the sensors attached on the bicycle and on the athlete. With access to all these information coaches can set goals for their workouts, refine their tactics and motivate cyclists to improve their performance [3].

II. Related Work

Cycling is a very complete physical activity that involves a set of movements. These movements can be analyzed using several free applications, smart wristbands or through devices placed on the bike and on the cyclist.

There are many sports applications on the market, such as Cyclometer Cycling Running GPS which is considered "*the most advanced application for cyclists ever designed for a mobile device*" [4]. In this application, it is possible to register routes, speed and schedule training sessions. It is an application available on iPhone and Android devices, but it is quite limited because it only collects information through the available resources in the mobile phone. Sports Tracker [5] is also a sports recording application, is suitable for various sports, including cycling. It is limited to the features of the mobile phone but has available to acquire accessories, for example, heart rate sensors.

In [6], the authors developed a heart rate monitoring system in order to determine the exercise intensity of a training session or race. This system has the ability to monitor several athletes simultaneously. It was used the Garmin heart rate monitor that was built with ANT+ communication protocol for wireless monitoring. The heart rate strap is activated once the athletes start moving. The receiver establishes the connection using nRF24AP1 and an Arduino-Nano, the data sent by the Garmin heart rate monitor is received by nRF24AP1, through a peer to peer network. The collected data is transmitted to the centralize center or to the coordinator wirelessly via the XBee Mesh Network. The data is sent attached to a unique ID and is distinguished using LabVIEW before the results are displayed.

The authors in [7], have researched and presented which wireless communication protocol best suited for the specific application of vital signs sensors monitoring athletes. This type of monitoring is very important, both for the athlete as for the coach to avoid overtraining. The values and performance of the ZigBee standard were simulated, evaluated and analyzed, using the star topology. The use of the wireless network allows to have several sensors being monitored at the same time, this increases the number of data acquired which contributes to a better view of the athletes' training. The OPNET Simulation Platform allows the study of devices, applications, protocols and it was the authors' choice to carry out the simulations. The main objective was the development and study of the communication network between the athletes (end-node) and the coach (coordinator). The authors concluded, from the results of the simulation that the ZigBee protocol, is the most suitable for monitoring athletes or similar applications.

The authors in [8] presented a distributed system of sensors expressed by a WSN to evaluate and improve the performance of cyclists. This system implements the ZigBee communication protocol to perform data acquisition, processing and communication. All collected data is stored in a cloud platform and can be accessed through a mobile application. The hardware involved in the WSN is responsible for the acquisition, processing and sending of data to the server. So, in the WSN are the force sensors, inertial measurement boards and microcontrollers. The force sensors are placed on the shoes and gloves, the Inertial Measurement Unit (IMU) board is attached to the cyclist's chest and it is used to measure the oscillations in the three plans of movement (yaw, pitch, roll). These sensors use the Arduino Fio, they work as end-node and are responsible for data acquisition. The Arduino Mega work as coordinator and network center (star topology), all of these microcontrollers have a XBee module coupled. On the coordinator is also a Global Positioning System (GPS) shield and a Wi-Fi shield that will communicate with the server. An Android application was also developed to help cyclists and coaches visualize training results and to extract relations between measured values. The difference of this work, besides detecting the interactions of the cyclist with his bicycle, is also to monitor his physical state, use different types of representation of the data. The data storage is in a different, more modern and a real-time storage system which uses a non Structured Query Language (NoSQL) syntax. All of the collected data can also be analyzed through a mobile application.

III. System Description

The system architecture includes three blocks (Fig.1). The first block is represented by the bike and the cyclist, where the wireless sensor network is placed, this block is an embedded system. Each end-node on the bike and the cyclist will send the acquired data to the coordinator, then it sends the data to the database, the coordinator is also on the bike. This takes us to the second block, the server receives the information from the WSN and makes it available to be accessed by the application user, either the trainer or the cyclist. The third block represents the mobile application which, with access to the server, gives the user the information collected. The application interprets and relates this information graphically and schematically so that it can be understood by users.

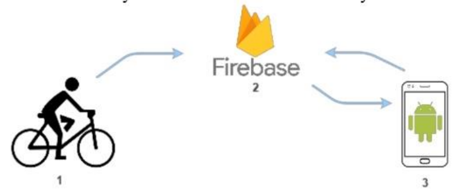


Fig. 1. System Architecture.

A. Hardware

The nodes of this embedded system consist in sensors connected to the Arduino Fio and the circuits of each end node are powered by lithium batteries. Among the several sensors we have: force sensors, IMU boards, sensors that measure heart rate and breathing. On this project, it was developed and implemented seven nodes in the wireless network based on the Arduino Fio processing platform. The Arduino Fio makes the acquisition and primary processing of the sensor data and gets it ready to send to the coordinator.

In this project, the force sensors are used on the cyclist's shoes and gloves. The force is measured by FlexiForce force sensors [9], which work as a resistance in an electrical circuit, when a force is applied to the sensor the resistance decreases and when the sensor is discharged that resistance increases. It was necessary to use a conditioning circuit since the acquisition module only reads voltage values, the circuit conditioning used is as follows (Fig 2.).

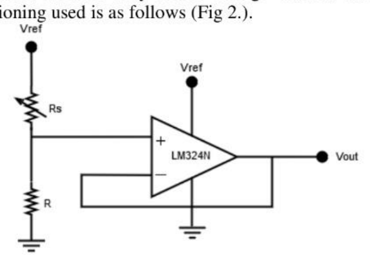


Fig. 2. Force sensor circuit's conditioning.

The output voltage equation (1) is:

$$Vout = \frac{R}{Rs + R} \times Vref \qquad (1)$$

Where *Vout* is the output voltage [V], *Rs* is the variable resistance that comes from the force sensor [Ω], *R* is the reference resistor [Ω] and *Vref* is the reference voltage [V].

To convert the Analog-to-Digital Converter (ADC) values [10] from the sensors to Newtons [N] the following formulas are used:

$$gain = \frac{KnownWeight}{ADCValue} = \frac{1500g}{131} \approx 11.45 \quad (2)$$

$$weight\ (in\ Newton) = gain\ \times ADC_{11.45value}\ \times 0.001\ \times 9.80665 \quad (3)$$

In equation (2) it is used a known weight (in this case 1.5 kg) and the read ADC value in order to calibrate the sensor. Equation (3) represents the calculation of the force applied to each sensor.

An IMU board was used to calculate the angles between the cyclist and the bike as well as the direction of the movement. The IMU packs a L3GD20H 3-axis gyro, a LSM303D 3-axis accelerometer and 3-axis magnetometer onto a tiny board. In the previous work, it was used the Direct Cosine Matrix Algorithm (DCM) to calculate the angles between the cyclist and the bike [11]. This algorithm allows to calculate the orientation of a rigid body relative to earth rotation using rotational matrices that describe the orientation of one coordinate system relative to another.

In order to obtain the cyclist's pulse during training, the method known as Photoplethysmogram (PPG) was used. It is a non-invasive method and consists in the variation of the absorption of light, traditionally through the finger. This type of sensors uses light-based technology to sense the rate of the blood flow [12] through its volume, which is an effect of the contraction of the heart [13]. Fig. 3 is the representation of the sensor working mode.

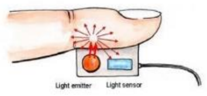


Fig. 3. PPG sensor working mode.

It is also important to get the cyclist's respiration rate, in order to do that it was used an electromechanical film which is a thin membrane which thickness is related to an electric voltage. These are ribbon type sensors and consist of a ferroelectric film and 3 layers of electrodes on polyester films or aluminum or copper electrodes [14].

The thickness changes of these sensors generate charge, and this is how it is possible to calculate the cyclist's breathing rate. Each inhalation and exhalation generate a non-zero voltage that is read and counted by the Arduino. The total count is divided by two, once a breath corresponds to one inspiration and expiration movement, the count is sent to the coordinator every minute.

It is necessary to use a conditioning circuit since the acquisition module only reads voltage values, the circuit conditioning used is as follows, in Fig. 4.

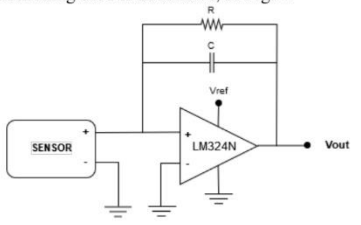


Fig. 4. Respiration rate sensor circuit's conditioning.

The Arduino Mega was used the system coordinator. It receives data from different end-nodes, sends information to the database, authentic cyclists and has a GPS module that records the coordinates of the training course. The coordinator has three shields and a sensor: GPS, Yun and Xbee shield and a Radio-Frequency Identification (RFID) reader, it can be found on the frame of the bike.

The GPS Shield is able to get the coordinates, date, time, altitude and speed data, this shied supports a SD card which contains the tags available for the cyclists to use. The identification of the cyclist is accomplished by, at the beginning of the training, an RFID reader. This module reads the tag and compares it the tags in the file in the Secure Digital Card (SD) that was mentioned before. The RFID reader used in this project was the RFID-RC522, it uses a 13.56 MHz frequency and this reader has its own library for the Arduino, MFRC522 library.

B. Communication

For the transfer of data, wireless communication protocols were used since it is a wireless sensor network. To communicate within WSN was used the ZigBee protocol and for the communication between the WSN and the database was used the Wi-Fi protocol.

All end-nodes and the coordinator have a XBee radio that is based on the IEEE 802.15.4 Standard that was designed for star, tree or point-to-point communications. In this project, it was used the star typology, as represented in Fig.5. This was the typology chosen because the nodes are close to the coordinator and they only transmit information.

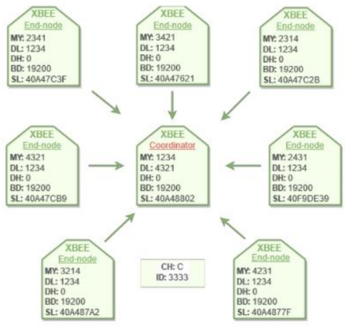


Fig. 5. Project's star topology.

The coordinator is the WSN gateway, with the Yun shield and a Wi-Fi modem, it is possible to connect to the database and send data. Since the Wi-Fi shield has a Linux operating system it is possible to send the information through the command line.

C. Database and Mobile Application

The application of this system, developed for Android. The application aims to help the coach to manage the results of his cyclist's trainings, the cyclist can also check the data from the different training.

All information regarding cyclists, coaches and training results has to be centrally stored in a place where it can be accessed remotely by anyone, with permission, and anywhere. The Firebase Realtime Database platform is responsible for this service. Firebase is structured on nodes, each of the nodes are identified by a unique key through which they are accessed in the application. Nothing is defined at the server level, data is sent both by the application and by the Arduino, with a certain structure that is recognized by the Firebase. Fig. 6 represents the main nodes in Firebase, used in this system.

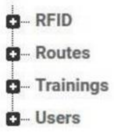


Fig. 6. Main nodes in Firebase.

The application is designed for the admin, coaches, cyclists, having different functionalities depending on the type of user. The access is done through a username and password system, to prevent access to data to unauthorized people.

The main features of the admin are:

- Register and delete coaches and cyclists;
- Get the main statistics of the system (for example, the total of coaches, cyclists and training).

The coach can:

- Register cyclists;
- Schedule trainings, by giving a route and a date;
- Monitor results of each of his cyclists;
- Compare with previous trainings;
- Give feedback.

The features of a cyclist are:

- View scheduled trainings;
- View training results;
- Compare with previous trainings;
- Have feedback from his coach.

The application is a very important element in this system because it is through it that the results obtained by the athletes are observed. These results are important to verify if there are improvements in the athlete's performance. To see if the training is appropriate or if it is necessary to change the type of training performed, considering his physical limitations or needs.

When the user is in the results activity he can access the training, data collected by the sensors. In Fig. 7 and Fig. 8 are examples of two types of charts with training results. The first one is the force applied by the cyclist in the hands. The second chart is the percentage of usage for the three sensors applied on each foot. For example, analyzing this chart it is possible to see that the most used sensors are the sensor 1 and 3 from the right foot.

D. Fluxogram

After the previous description, it is possible to understand how the real system works, as is represented in Fig. 9. From the configuration to the execution and the visualization of the results. The main stakeholders are the coach and the cyclist.

The coach can schedule and set up the training session (A) with parameters such as the date and route. After submission, these settings are sent to the server and stored in the database (B).

The cyclist starts the training session through an RFID card (C), this card makes it possible to identify the athlete who will perform the training. The RFID card will also trigger the start of data collection by WSN, which will happen as the cyclist performs the training (D). The training ends by turning off the equipment (E).

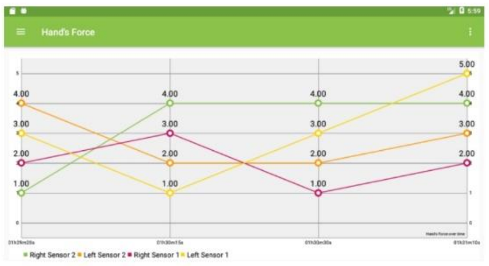


Fig. 7. Hand's force sensors result.

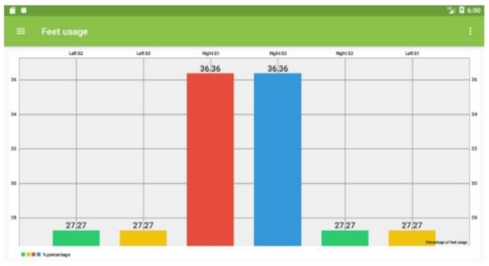


Fig. 8. Percentage of the feet's sensor usage.

When the biker has already completed at least one training session, the trainer can monitor the results (F) and leave some feedback, if necessary. The athlete can also view the results of training (G) and feedback left by his coach. The application accesses the server both to get the respective data and to save the comments that the coach can make (H).

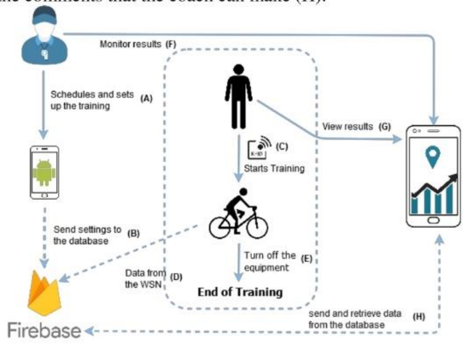

Monitor results (F)
Schedules and sets up the training (A)
View results (G)
(C)
Starts Training
Send settings to the database (B)
Turn off the equipment (E)
Data from the WSN (D)
End of Training
send and retrieve data from the database (H)
Firebase

Fig. 9. System Workflow.

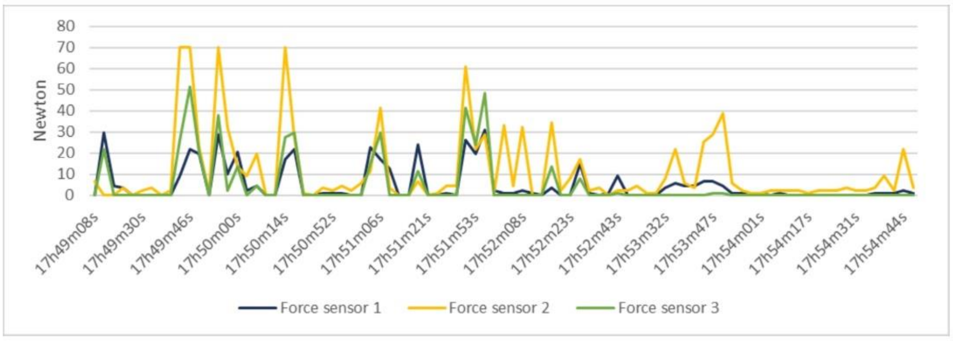


Fig. 10. Feet's force sensor - First Training.

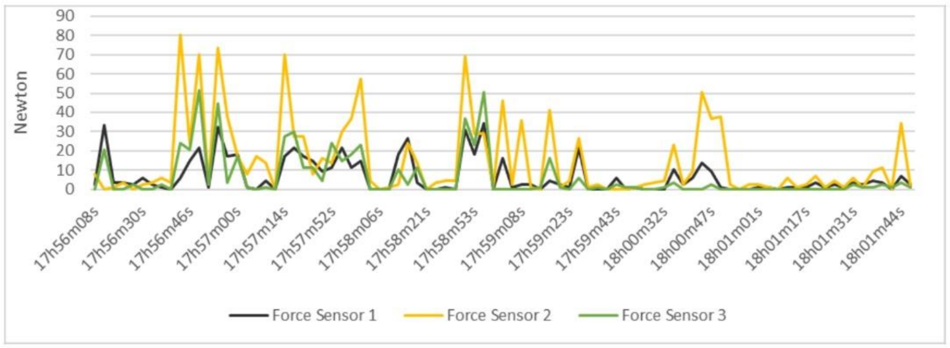


Fig. 11. Feet's force sensors - Second training.

IV. Results and Discussion

The first results of system tests were performed in the laboratory and in real time, by a female subject. The participant is healthy and performed two training sessions of five minutes each, always performing the same course, with a rest interval of two minutes. The experimental procedure focuses on checking the differences in the effort as a consequence of two equal workouts, namely the variation of the heart rate and the force exerted in each one of the exercises. In the graph of Fig. 10, it is possible to see the results of the left foot force sensors, from the first training, and Fig. 11 for the second training. In Fig. 12 it is possible to see the graphs corresponding to the heart rate per minute, for the first and second training.

The results analyzed are the left foot sensors obtained through three *Flexiforce* sensors in the left foot and the heart beats obtained through the PPG sensor placed in the ear. This data was automatically placed in the firebase database. Then it was exported with JavaScript Object Notation (JSON) format and later imported into Microsoft Excel from where it was possible to obtain the charts.

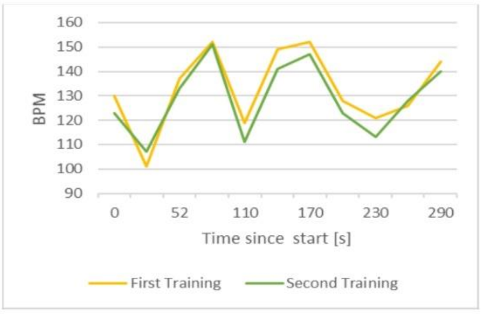


Fig. 12. Heart Beat per minute.

By analyzing the result charts, it is possible to see that the effort when pedaling is higher in the second training. In the first training the values do not exceed 70 Newtons whereas in the second training they reach 80 Newtons. Like the force, the heart rate of the second workout also increases slightly compared to the first. The results present this behavior since the participant only had two minutes to rest. Despite being short workouts, it proves to be insufficient time to recover either the leg muscles or the normal heartbeat.

V. Conclusions and Future Work

An IoT system for cyclist activity monitoring including the performance evaluation was designed and implemented. The system capabilities were tested and the obtained data allows to compare the results coming from different trainings may be used to extract differences between different training sessions. Based on the analyzed data a feedback is provided to the cyclist. Thus, the athlete becomes more motivated and may consider a different strategy to obtain better results.

The system provides also metrics that may be evaluated by the coach. Thus, the coach may find out where the athletes are failing and what points to improve.

At the technological level the system presents a user friendly graphical interface including statistical graphs for easy data visualization. The trainer can follow the cyclist's evolution for days, months or years considering the implemented IoT architecture.

The normal heart Beatings Per Minute (BPM) of a resting adult is between 60 and 90 BPM [15], and moderate physical activity may reach 160 BPM. These values prove the accuracy of the system in the collection of information.

To summarize, this system can acquire data from the sensors, send it to the a Realtime database and them the coach or the cyclist can access it through the application. In terms of data access, the latency is very low, either sending or receiving the data, about less than 50 milliseconds.

In the future, the authors intend to do more tests to the system, with participants of different genders and physical condition and follow up of subjects for several days to analyze their cycling performances.

References

[1] P. P. Ray, “Internet of Things for Sports (IoTSport): An architectural framework for sports and recreational activity,” Int. Conf. Electr. Electron. Signals, Commun. Optim. EESCO 2015, 2015.

[2] J. Rodian, “StiboSystems.” [Online]. Available: http://blog.stibosystems.com/how-data-and-iot-are-changing-sports. [Accessed: 10-Dec-2017].

[3] S. K. Gharghan, R. Nordin, and M. Ismail, “Design Consideration of an Energy Efficient Wireless Sensor Network for High Performance Track Cycling,” 2014 Int. Conf. Inf. Sci. Appl., p. 5, 2014.

[4] “iTunes Apple - Cyclemeter Cycling Running GPS.” [Online]. Available: https://itunes.apple.com/br/app/cyclemeter-cycling-running-gps/id330595774?mt=8. [Accessed: 10-Dec-2017].

[5] “Sports Tracker.” [Online]. Available: http://www.sports-tracker.com/. [Accessed: 10-Dec-2017].

[6] N. S. A. Zulkifli, F. K. C. Harun, and N. S. Azahar, “XBee Wireless Sensor Networks for Heart Rate Monitoring in Sport Training,” 2012.

[7] G. Kiokes et al., “Performance evaluation of a communication protocol for vital signs sensors used for the monitoring of athletes,” Int. J. Distrib. Sens. Networks, vol. 2014, 2014.

[8] T. Ribeiro, O. Postolache, and P. Passos, “Performance assessment for mountain bike based on WSN and cloud technologies,” Proc. 2016 Int. Conf. Expo. Electr. Power Eng. EPE 2016, no. Epe, pp. 380–386, 2016.

[9] Tekscan, “FlexiForce ® Sensors | User Manual.”

[10] SparkFun Electronics, “Analog to Digital Conversion,” p. 470, 2010.

[11] T. Ribeiro, O. Postolache, and P. Passos, “Performance assessment for mountain bike based on WSN and cloud technologies,” ISCTE-IUL, 2016.

[12] F. Q. Al-Khalidi, R. Saatchi, D. Burke, H. Elphick, and S. Tan, “Respiration rate monitoring methods: A review,” Pediatr. Pulmonol., vol. 46, no. 6, pp. 523–529, 2011.

[13] A. Ave, H. Fauzan, S. R. Adhitya, and H. Zakaria, “Early detection of cardiovascular disease with photoplethysmogram(PPG) sensor,” Proc. - 5th Int. Conf. Electr. Eng. Informatics Bridg. Knowl. between Acad. Ind. Community, ICEEI 2015, pp. 676–681, 2015.

[14] “EMFIT | R-Series sensor.” [Online]. Available: https://www.emfit.com/copy-of-l-series. [Accessed: 04-Sep-2018].

[15] A. Shaffler and N. Menche, Medicina Interna e Cuidados de Enfermagem.

Appendix B - User Manual

ISCTE IUL
Instituto Universitário de Lisboa

Department of Information Science and Technology

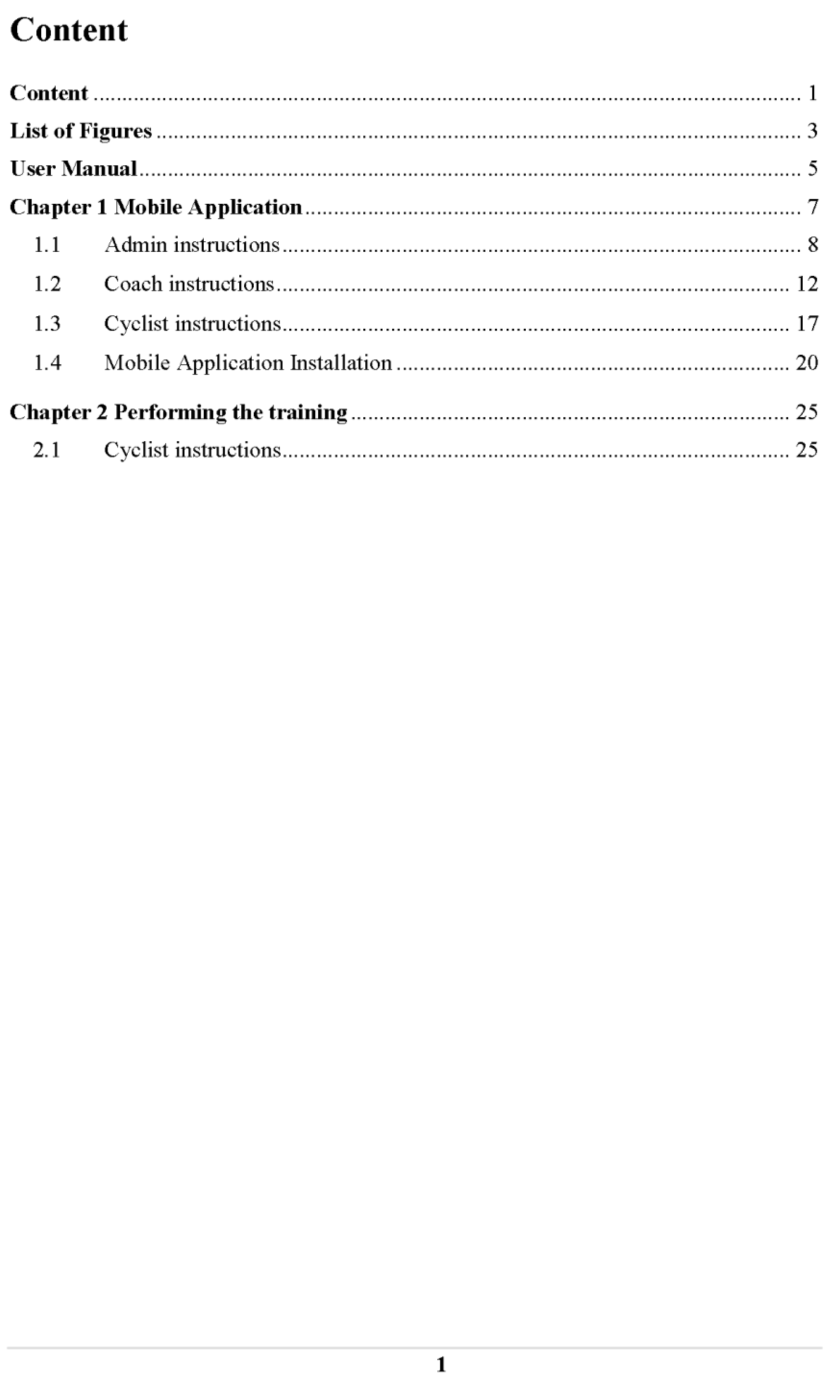
User Manual

Cyclist performance assessment based on WSN and Cloud technologies

Ana Catarina Duque Dias

Supervisor:
Dr. Octavian Adrian Postolache, Assistant Professor
ISCTE-IUL

October 2018



Content

List of Figures

User Manual

This manual aims to present the features of the developed system and how it should be used. The system has three types of users, the Admin whose function is to manage the users in the application. The Coach, that whose job is to analyze the training results and give feedback to the cyclists. Finally, the Cyclist who performs the training and also has access to the results in the application.

5

Chapter 1 Mobile Application

Each user access different functionalities of the application. In this chapter the first section describes the Admin features, the second section demonstrates the functionalities of the Coach.

The login page is the same to all user and is it represented in Figure 1.1. Each user has his own username and password in order to enter the application. The application requires an internet connection to work properly.

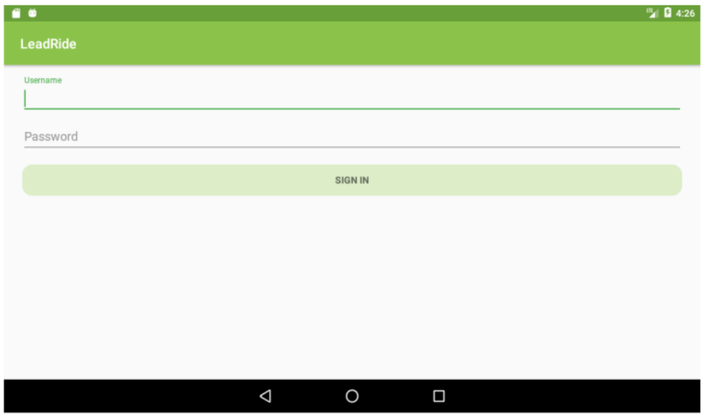


Figure 1.1 - Login page

If the credentials are not correct the application displays a message with an error, Figure 1.2. The application requires an internet connection to work properly.

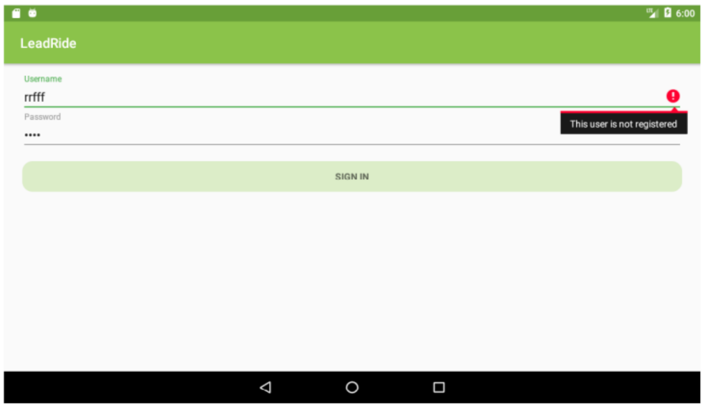
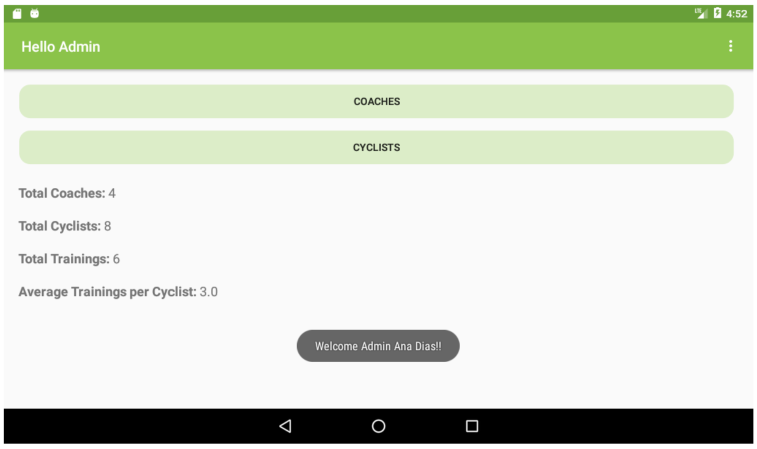


Figure 1.2 - Error message in the Login page

1.1 Admin instructions

This subchapter shows the application features for the Admin user type, when this user logs in his type is automatically detected. After a successful login, the application shows the corresponding homepage, Figure 1.3. The figure also shows a toast message that identifies the user as Admin type.



Total Coaches: 4

Total Cyclists: 8

Total Trainings: 6

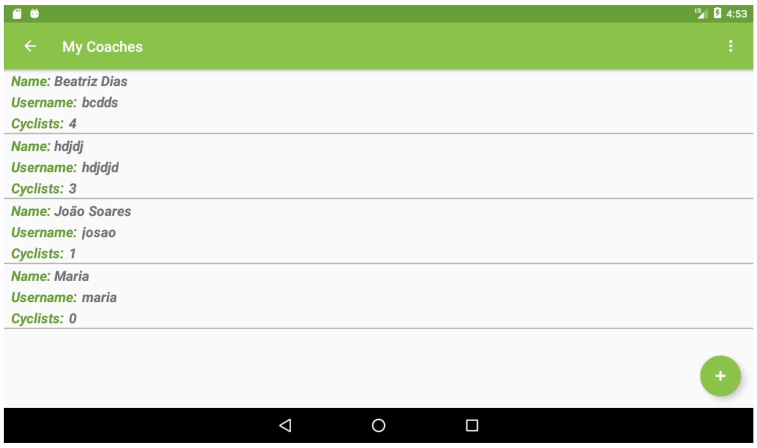
Average Trainings per Cyclist: 3.0

Welcome Admin Ana Dias!!

Figure 1.3 – Admin's homepage

In the homepage, the user can see some statistic information about the application data and, from here, the admin can see the list of all of the coaches and cyclists, by clicking the correspondent button.

In Figure 1.4, is the activity with the list of all coaches registered in the database, only the admin has access to this type of data.



Name: Beatriz Dias
Username: bcdds
Cyclists: 4

Name: hdjdj
Username: hdjdjd
Cyclists: 3

Name: João Soares
Username: josao
Cyclists: 1

Name: Maria
Username: maria
Cyclists: 0

Figure 1.4 - List of coaches

From this activity, the user can create new coaches (Figure 1.5), by clicking the add button in the bottom right corner, and access the coach's profile (Figure 1.6), by clicking in a list item.

When the admin wants to add a new coach, he needs to fill all the fields in the form.

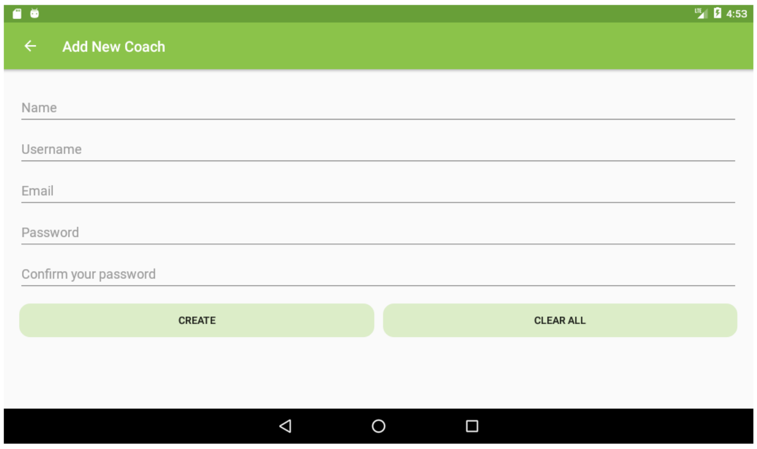
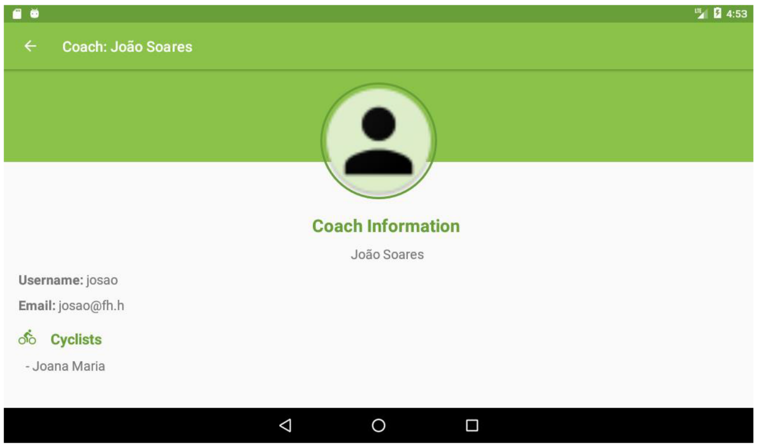


Figure 1.5 - Add new coach activity


Coach: João Soares

Coach Information

João Soares

Username: josao

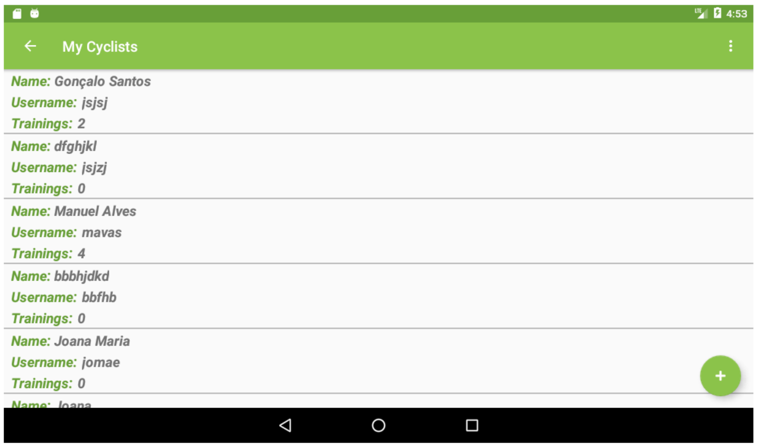
Email: josao@fh.h

Cyclists

- Joana Maria

Figure 1.6 - Coach's profile

In Figure 1.7 is the activity shown when the user clicks on the cyclists' button on the homepage activity (Figure 1.3).


My Cyclists

Name: *Gonçalo Santos*
Username: *jsjsj*
Trainings: *2*

Name: *dfghjkl*
Username: *jsjzj*
Trainings: *0*

Name: *Manuel Alves*
Username: *mavas*
Trainings: *4*

Name: *bbbhjdkd*
Username: *bbfhb*
Trainings: *0*

Name: *Joana Maria*
Username: *jomae*
Trainings: *0*

Name: *Joana*

Figure 1.7 - List of cyclists

As in the list of trainers here is also possible to consult the profile of each cyclist by clicking on a list item and add new cyclists. The Admin has access to all the cyclists registered in the database. In order to add new cyclists, the Admin has to provide a coach for them, this is an extra field in the form that gets the coach list from the database and shows them in a spinner.

From every activity that has the symbol in Figure 1.8 in the right upper corner, it is possible for the user to log out (Figure 1.10). If the user is in the homepage activity it is also possible to edit his information of fields such as name, email and password, as shown in Figure 1.9.

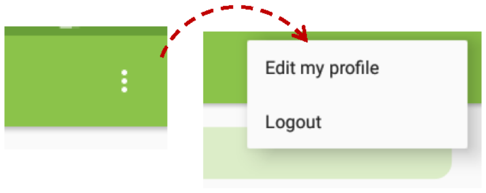
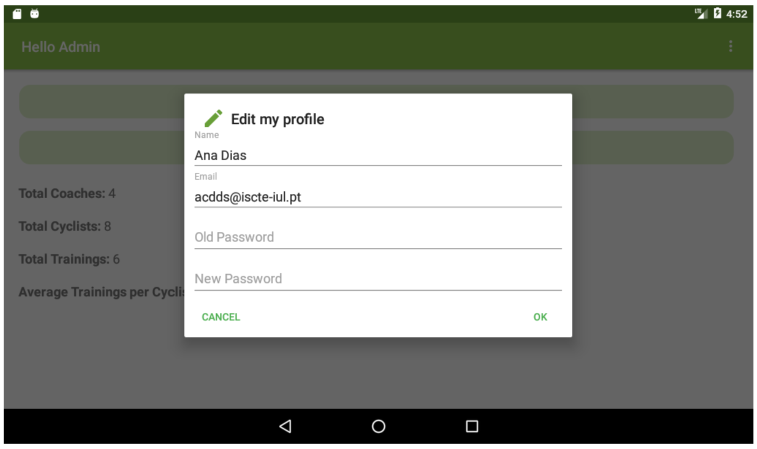


Figure 1.8 - Options button



Edit my profile

Name
Ana Dias

Email
acdds@iscte-iul.pt

Old Password

New Password

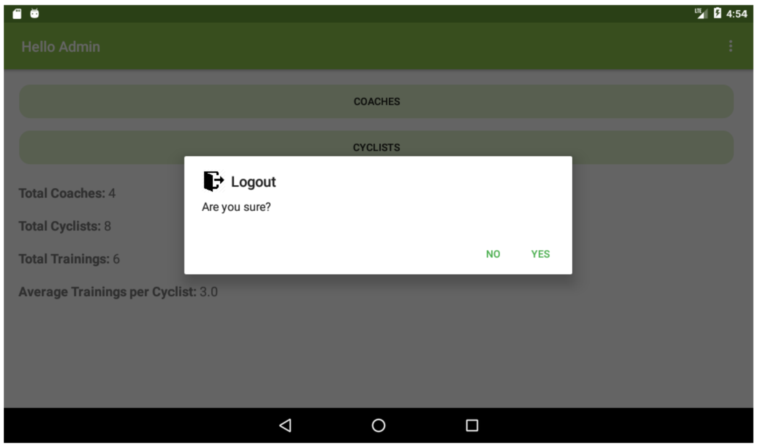
Total Coaches: 4

Total Cyclists: 8

Total Trainings: 6

Average Trainings per Cycli

Figure 1.9 - Admin's edit profile dialog


Hello Admin

COACHES

CYCLISTS

Total Coaches: 4

Total Cyclists: 8

Total Trainings: 6

Average Trainings per Cyclist: 3.0

Logout

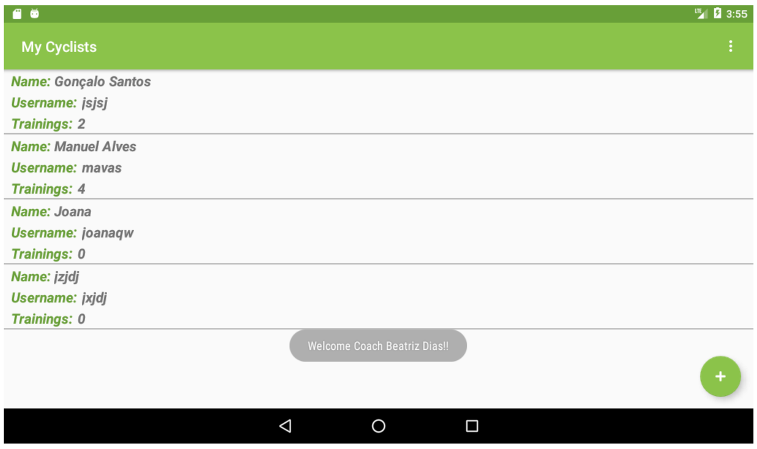
Are you sure?

NO YES

Figure 1.10 - Logout dialog

1.2 Coach instructions

The Coach's main features are shown in this subchapter. Like the Admin, the type of user is automatically detected once the user logs in. Figure 1.11 shows the coach's homepage and the toast message that identifies the user type.


My Cyclists

Name: Gonçalo Santos
Username: jsjsj
Trainings: 2

Name: Manuel Alves
Username: mavas
Trainings: 4

Name: Joana
Username: joanaqw
Trainings: 0

Name: jzjdj
Username: jxjdj
Trainings: 0

Welcome Coach Beatriz Dias!!

Figure 1.11 - Coach's homepage

The coach's homepage shows the list containing only his cyclists, as the admin, the user can click on one cyclist in the list to access his profile or add new cyclists by clicking the button in the bottom right corner. In Figure 1.12 is the form to register a new cyclist, all files are required, if the fields are empty or have errors, the application shows the correspondent messages (Figure 1.13). The field "*Select the RFID Tag*" only shows the available tags.

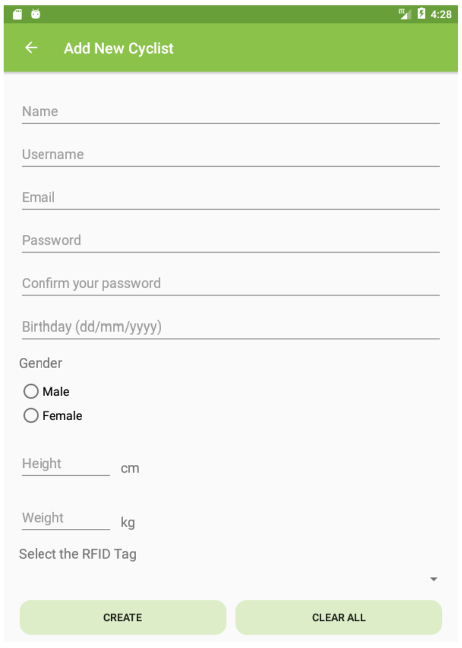
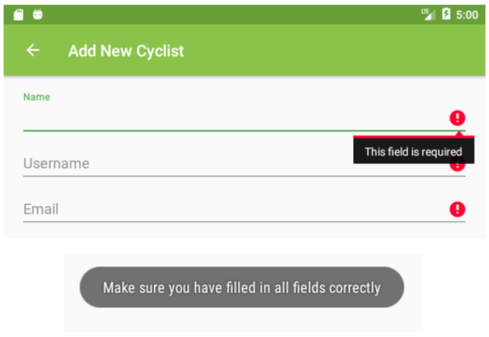


Figure 1.12 - Add new cyclist activity

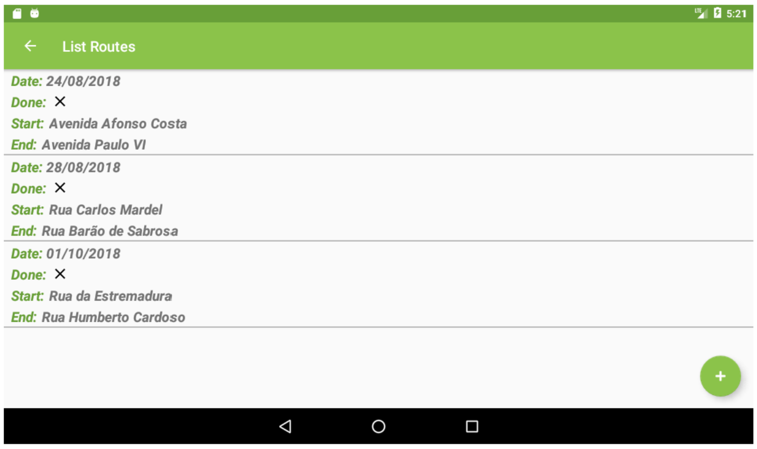


This field is required

Make sure you have filled in all fields correctly

Figure 1.13 - Add new cyclist error messages

By clicking on an element of the list in Figure 1.11, the application displays the cyclist's profile, which is also the cyclist's homepage, in this page the user has two buttons *Routes* and *Results*. From here the coach can access the list with the cyclist's trainings or the list of planned routes (Figure 1.14).



List Routes

Date: 24/08/2018
Done: ✕
Start: Avenida Afonso Costa
End: Avenida Paulo VI

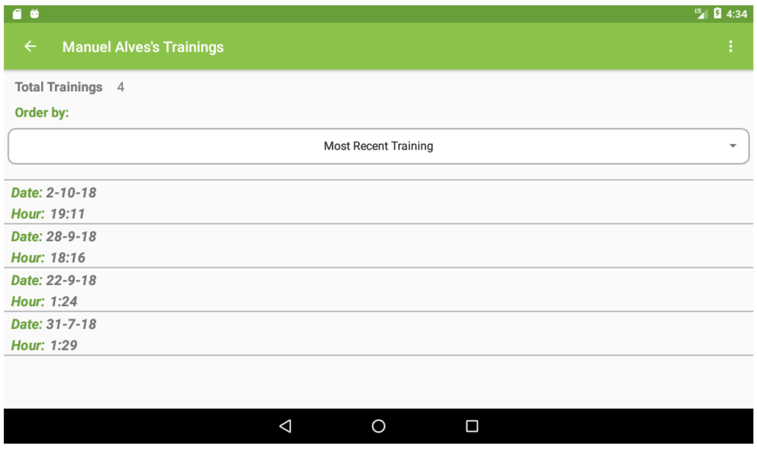
Date: 28/08/2018
Done: ✕
Start: Rua Carlos Mardel
End: Rua Barão de Sabrosa

Date: 01/10/2018
Done: ✕
Start: Rua da Estremadura
End: Rua Humberto Cardoso

Figure 1.14 - List of routes

From this activity, the coach can see the planned route or create new ones. When creating new routes, the coach can select up to three points in the map, give a date for the training he's planning and a name.

When the user is in the cyclist profile he can also access the training list (Figure 1.15) through the *Results* button.



Manuel Alves's Trainings

Total Trainings 4
Order by:
Most Recent Training

Date: 2-10-18
Hour: 19:11

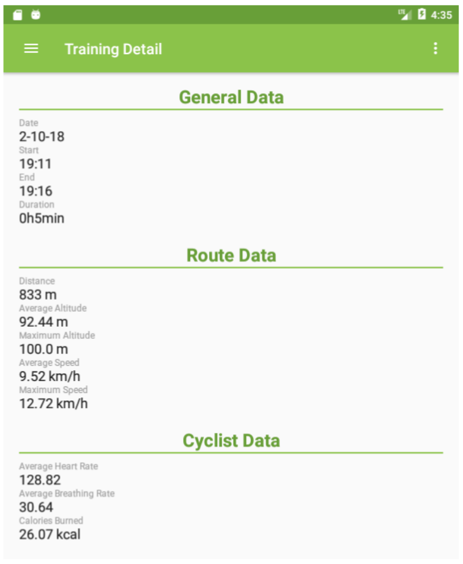
Date: 28-9-18
Hour: 18:16

Date: 22-9-18
Hour: 1:24

Date: 31-7-18
Hour: 1:29

Figure 1.15 - List of performed trainings

This activity shows the total number of trainings and the list of trainings, it has the functionality of ordering the training from the most recent to the oldest and vice-versa. By selecting one of the trainings, the coach can see all the collected data from the sensors, organized and treated through calculations and charts. The homepage of the results is in Figure 1.16.


4:35

Training Detail

General Data

Date
2-10-18
Start
19:11
End
19:16
Duration
0h5min

Route Data

Distance
833 m
Average Altitude
92.44 m
Maximum Altitude
100.0 m
Average Speed
9.52 km/h
Maximum Speed
12.72 km/h

Cyclist Data

Average Heart Rate
128.82
Average Breathing Rate
30.64
Calories Burned
26.07 kcal

Figure 1.16 – Results' homepage

By displaying the menu (Figure 1.17) on the upper left corner of the previous figure the user can access the charts with the results, the training route, export the data, add comments and go back to the list of trainings or to the results homepage. In Figure 1.18 is an example of a chart, the altitude chart.

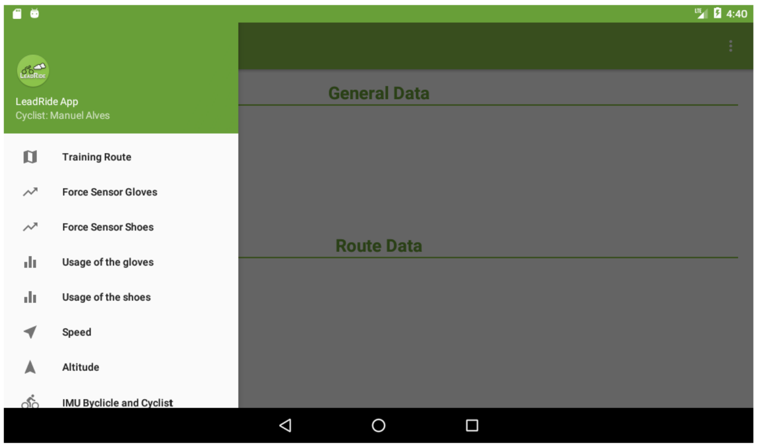


Figure 1.17 - Results' menu

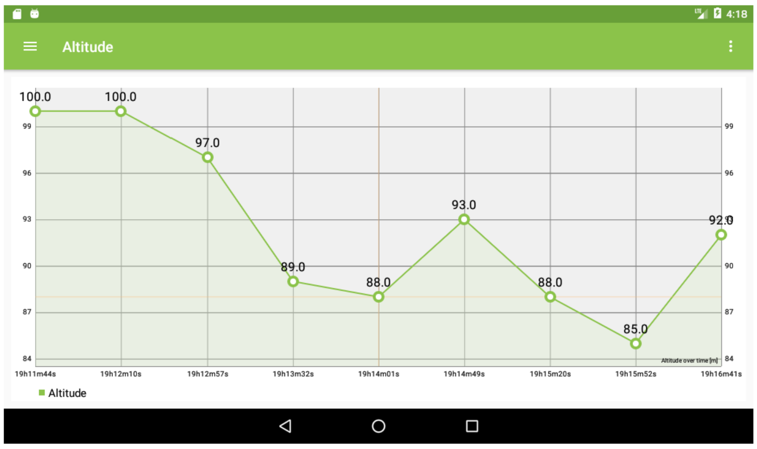
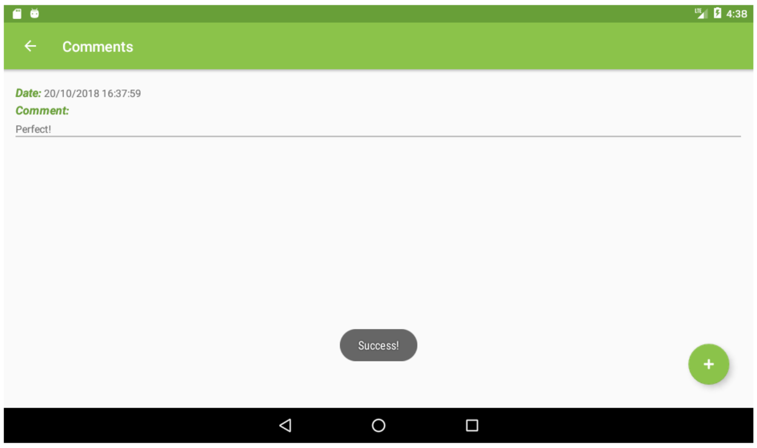


Figure 1.18 - Altitude chart

The coach is the only one who has permissions to add comments of the trainings, this feature can be accessed through the comments list on the menu, Figure 1.19.



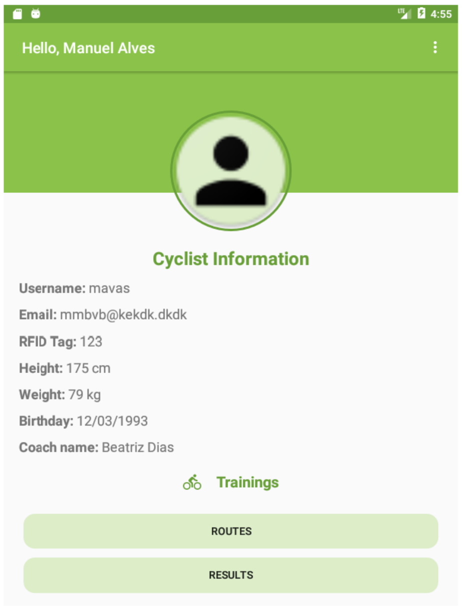
Date: 20/10/2018 16:37:59
Comment:
Perfect!

Figure 1.19 - Comments feature

The figure shows the list of comments immediately after the coach added one, it shows the commented added and the *Success* message.

1.3 Cyclist instructions

This subchapter describes the features of the cyclist type of user, as the other two types, the type of this one is also detected immediately after they log in successfully, it is also shown the correspondent homepage, Figure 1.20.

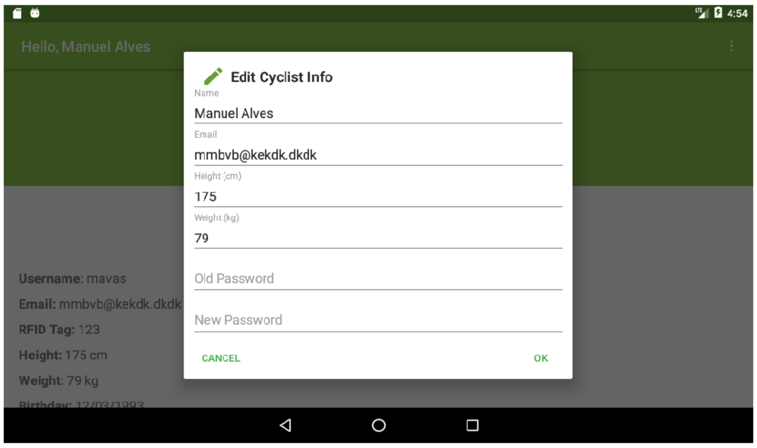

Hello, Manuel Alves

Cyclist Information

Username: mavas
Email: mmbvb@kekdk.dkdk
RFID Tag: 123
Height: 175 cm
Weight: 79 kg
Birthday: 12/03/1993
Coach name: Beatriz Dias

Figure 1.20 - Cyclist's homepage

From this activity, the user can edit his name, email height, weight and password, as shown in Figure 1.21.


Edit Cyclist Info

Name
Manuel Alves
Email
mmbvb@kekdk.dkdk
Height (cm)
175
Weight (kg)
79
Old Password
New Password

Figure 1.21 - Cyclist's edit profile dialog

When the cyclist is on the homepage he has the same options as the coach, he can see the planned routes but cannot add new ones. He can also access the list of performed, trainings, from which he can see your results. The results homepage (Figure 1.16) and menu (Figure 1.17) are the same for both users, the only difference is that the cyclist cannot add comments, he can only see them.

Figure 1.22 shows one of the results that both the cyclist and the coach can see, in this case, the performed route.

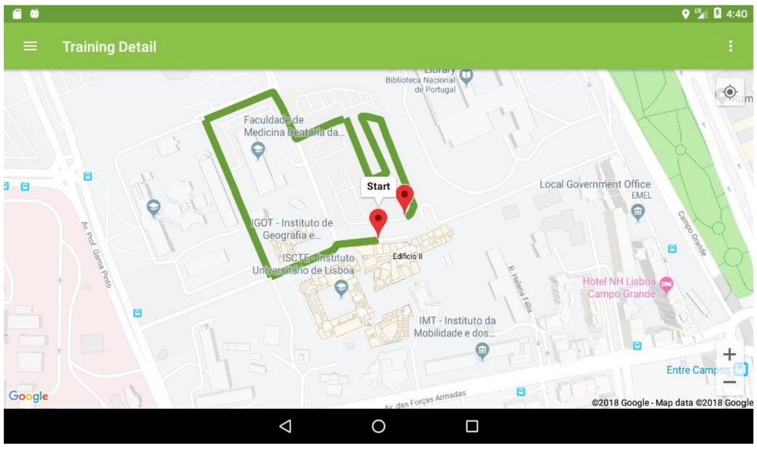

4:40
Training Detail
Start
Faculdade de Medicina Dentária da...
IGOT - Instituto de Geografia e...
ISCTE - Instituto Universitário de Lisboa
Edifício II
IMT - Instituto da Mobilidade e dos...
Local Government Office EMEL
Hotel NH Lisboa Campo Grande
Biblioteca Nacional de Portugal
Entre Campos
Google
©2018 Google - Map data ©2018 Google

Figure 1.22 - Route performed

Figure 1.23 shows the behavior of the application when the user is exporting the data, after this is completed, a file with all the training data is created, the name of the file is the cyclist's key, date and time of the training.

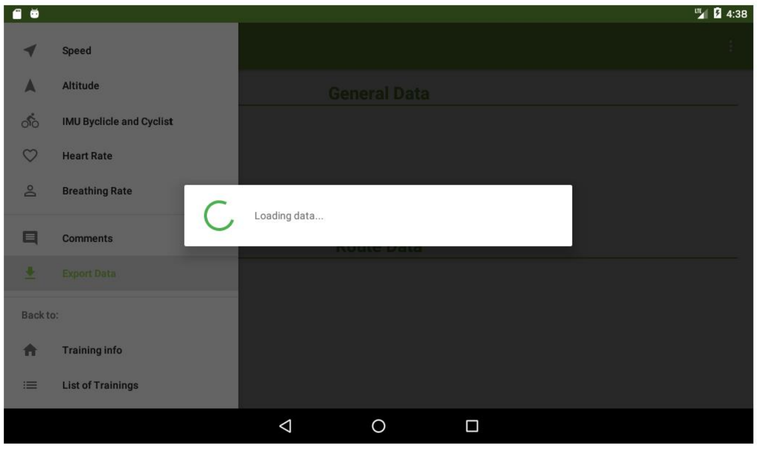


Figure 1.23 - Exporting the training data

1.4 Mobile Application Installation

This section shows how to install the mobile application developed to support this system.

To get the required files it is necessary to access the pen drive the contents of the project, as shown in Figure 1.24.

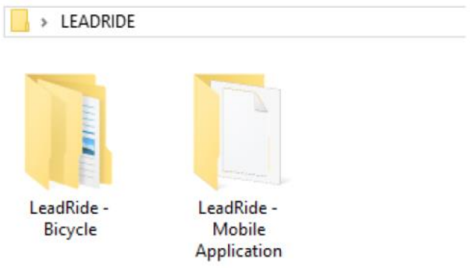


Figure 1.24 - Pen drive's main folders

In the folder ***LeadRide – Mobile Application***, there is an APK file (Figure 1.25) that must be installed on a mobile device to use the LeadRide mobile application. The APK file should be downloaded to the smartphone.

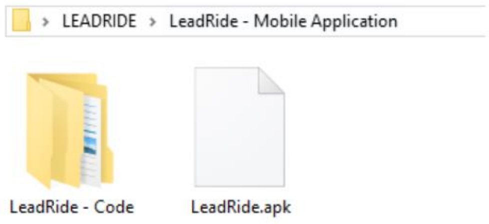


Figure 1.25 - Contents of the *LeadRide – Mobile Application* folder

The application was developed to run on devices with version 5.0 (Lollipop) or higher, and its target is version 6.0 (Marshmallow). The mobile device where the application is to be installed should have the "Unknown Sources" option active:

- Go to "Settings" (Figure 1.26).

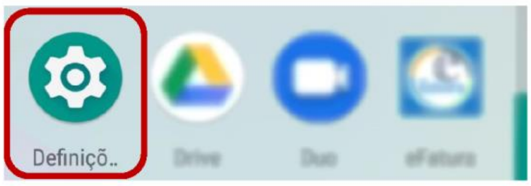


Figure 1.26 - Device settings

- Tap "Additional Settings" (Figure 1.27).

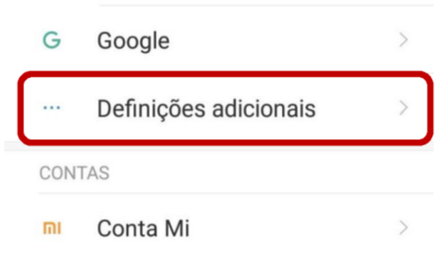


Figure 1.27 - Device Settings - Additional Settings Option

- Check the "Privacy" box (Figure 1.28).

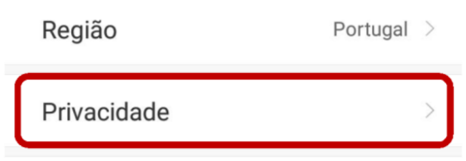


Figure 1.28 - Privacy option

- Check the "Unknown sources" box (Figure 1.29).

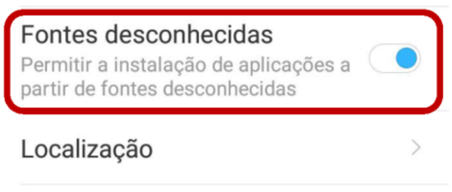

Fontes desconhecidas
Permitir a instalação de aplicações a partir de fontes desconhecidas
Localização

Figure 1.29 - Unknow sources option

To manually install the APK file:

- Tap "File manager" (Figure 1.30).

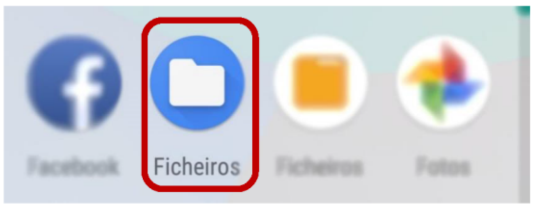


Figure 1.30 – Device's File Manager

- Use the File manager to locate the APK application and tap the APK file to open it (Figure 1.31).

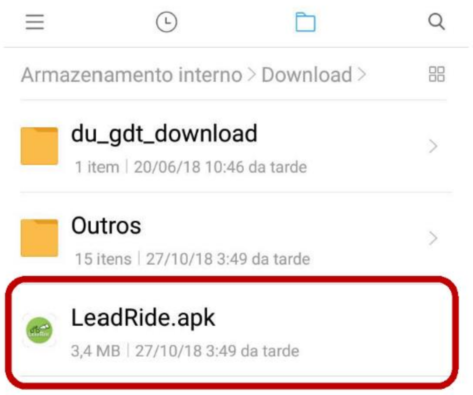
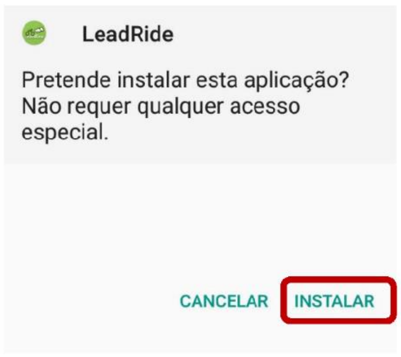

Armazenamento interno > Download >
du_gdt_download
1 item | 20/06/18 10:46 da tarde
Outros
15 itens | 27/10/18 3:49 da tarde
LeadRide.apk
3,4 MB | 27/10/18 3:49 da tarde

Figure 1.31 - LeadRide.apk

- Tap "Install" to complete the installation (Figure 1.32).


LeadRide

Pretende instalar esta aplicação? Não requer qualquer acesso especial.

Figure 1.32 – Install the LeadRide application

After installing the application is ready to be used by cyclist, coach and administrator (Figure 1.33).

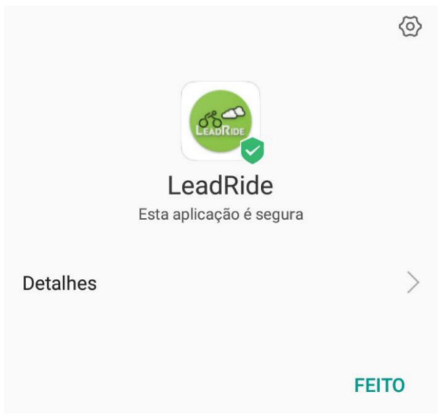


Figure 1.33 - Installation Complete

Chapter 2 Performing the training

This chapter describes the necessary procedures to perform the training. In this case, the cyclist is the only stakeholder.

2.1 Cyclist instructions

To perform a training session the cyclist must turn on the equipment. It is advisable for the cyclist to turn on the microcontrollers of the end-nodes first and finally the coordinator. There are seven microcontrollers like the one in Figure 2.1, and all of them have a switch on/off (indicated on the right side of the figure). To check if it is on, the light, marked on the left side of the figure, lights up.

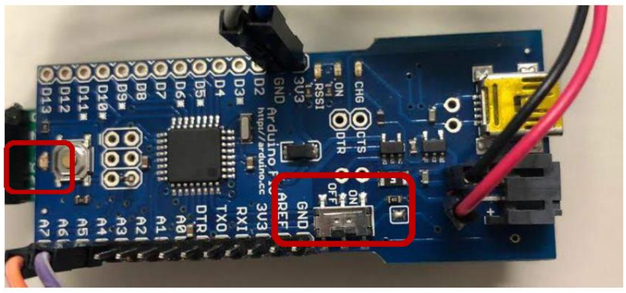


Figure 2.1 - Microcontroller - switch on/off

Finally, the cyclist has to turn on the coordinator (Figure 2.2), its batteries (Figure 2.3) and pass the RFID tag or card to start the training.

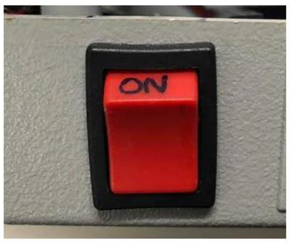


Figure 2.2 - Coordinator - switch on/off

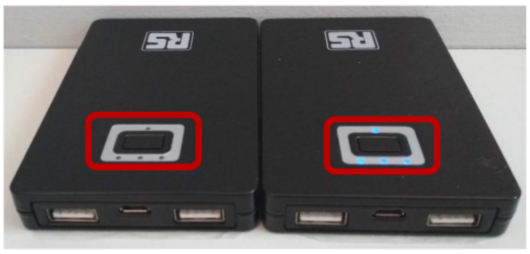


Figure 2.3 - Coordinator's batteries

At the end of the training, the cyclist must do the opposite procedure, in order to turn off the equipment.

Appendix C - Technical Manual

Instituto Universitário de Lisboa

Department of Information Science and Technology

Technical Manual

Cyclist performance assessment based on WSN and Cloud technologies

Ana Catarina Duque Dias

Supervisor:

Dr. Octavian Adrian Postolache, Assistant Professor

ISCTE-IUL

October 2018

Content

List of Figures

List of Tables

Technical Manual

This manual aims to describe the technical features of the system. Chapter 1 describes the embedded system, the communications and the programming of the microcontrollers. Chapter 2 explains the main functions of the mobile application.

.

Chapter 1 Embedded System

This chapter intends to explain how the end-nodes communicate with the coordinator and how the coordinator communicates with the database. Also, how the different shields are connected to each other and how to configure them.

1.1 Communications

All the communications between the end-nodes and the coordinator are made through the ZigBee protocol based on the IEEE 802.15.4.

1.1.1 XBee Configuration

The XBee modules have two different work modes: Transparent Mode (AT) and API mode. AT mode is used for point-to-point communications between two XBee. The API mode allows the communication between more than two XBee and the exchange of structured message packages. In this project, the API mode was used.

In the tables below, are the configurations of the XBee modules made through the XCTU software:

Table 1.1 - Coordinator's XBee configuration

Coordinator	
Channel (CH)	C
PAN ID (ID)	3333
Destination Address High (DH)	0
Destination Address Low (DL)	4321
16-Bit Source Address (MY)	1234
Serial Number High (SH)	13A200
Serial Number Low (SL)	40A48802
Coordinator Enable (CE)	Coordinator [1]
Interface Data Rate (BD)	19200 [4]
API Enable (AP)	API enabled [1]

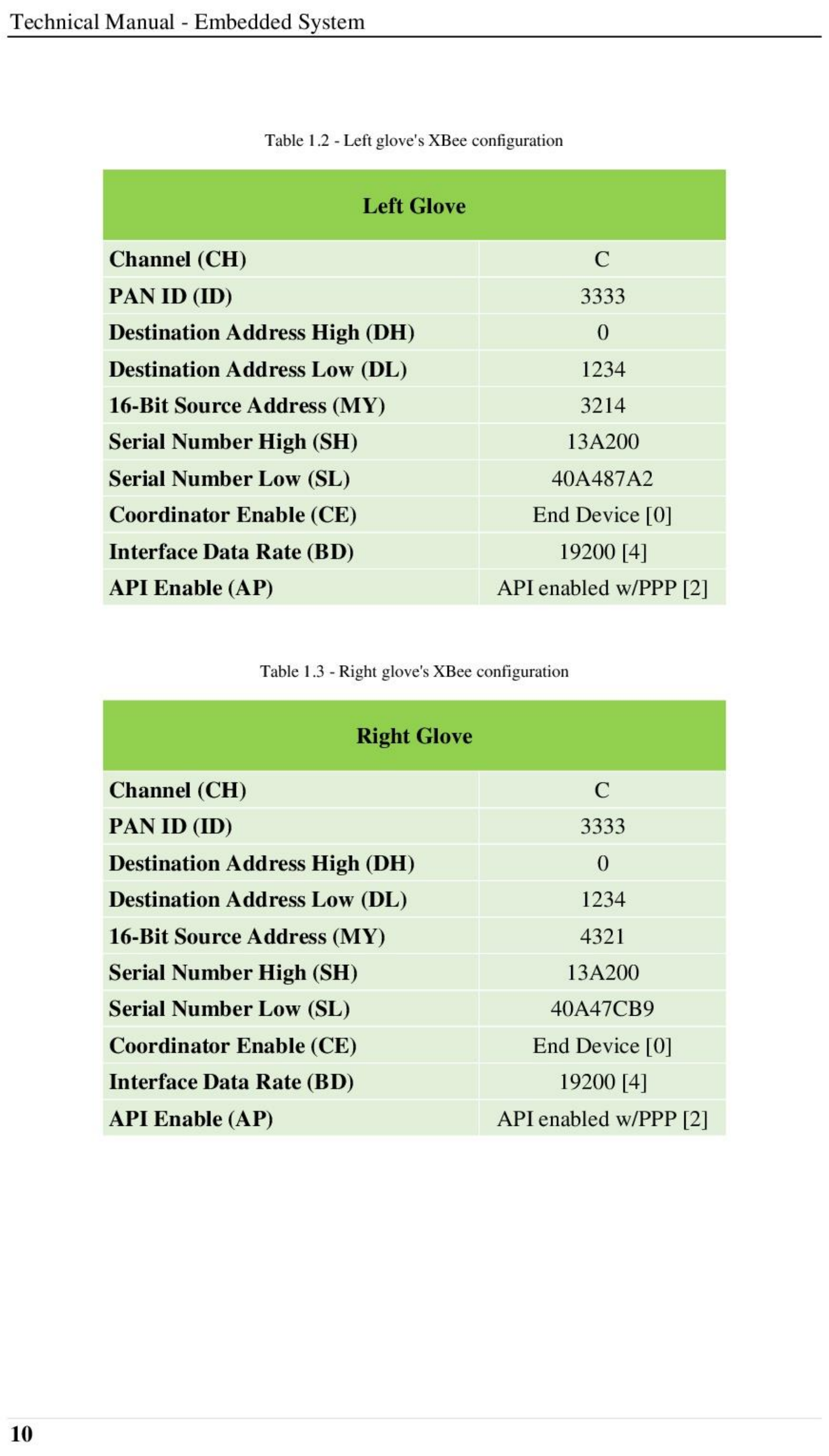


Table 1.2 - Left glove's XBee configuration

Left Glove	
Channel (CH)	C
PAN ID (ID)	3333
Destination Address High (DH)	0
Destination Address Low (DL)	1234
16-Bit Source Address (MY)	3214
Serial Number High (SH)	13A200
Serial Number Low (SL)	40A487A2
Coordinator Enable (CE)	End Device [0]
Interface Data Rate (BD)	19200 [4]
API Enable (AP)	API enabled w/PPP [2]

Table 1.3 - Right glove's XBee configuration

Right Glove	
Channel (CH)	C
PAN ID (ID)	3333
Destination Address High (DH)	0
Destination Address Low (DL)	1234
16-Bit Source Address (MY)	4321
Serial Number High (SH)	13A200
Serial Number Low (SL)	40A47CB9
Coordinator Enable (CE)	End Device [0]
Interface Data Rate (BD)	19200 [4]
API Enable (AP)	API enabled w/PPP [2]

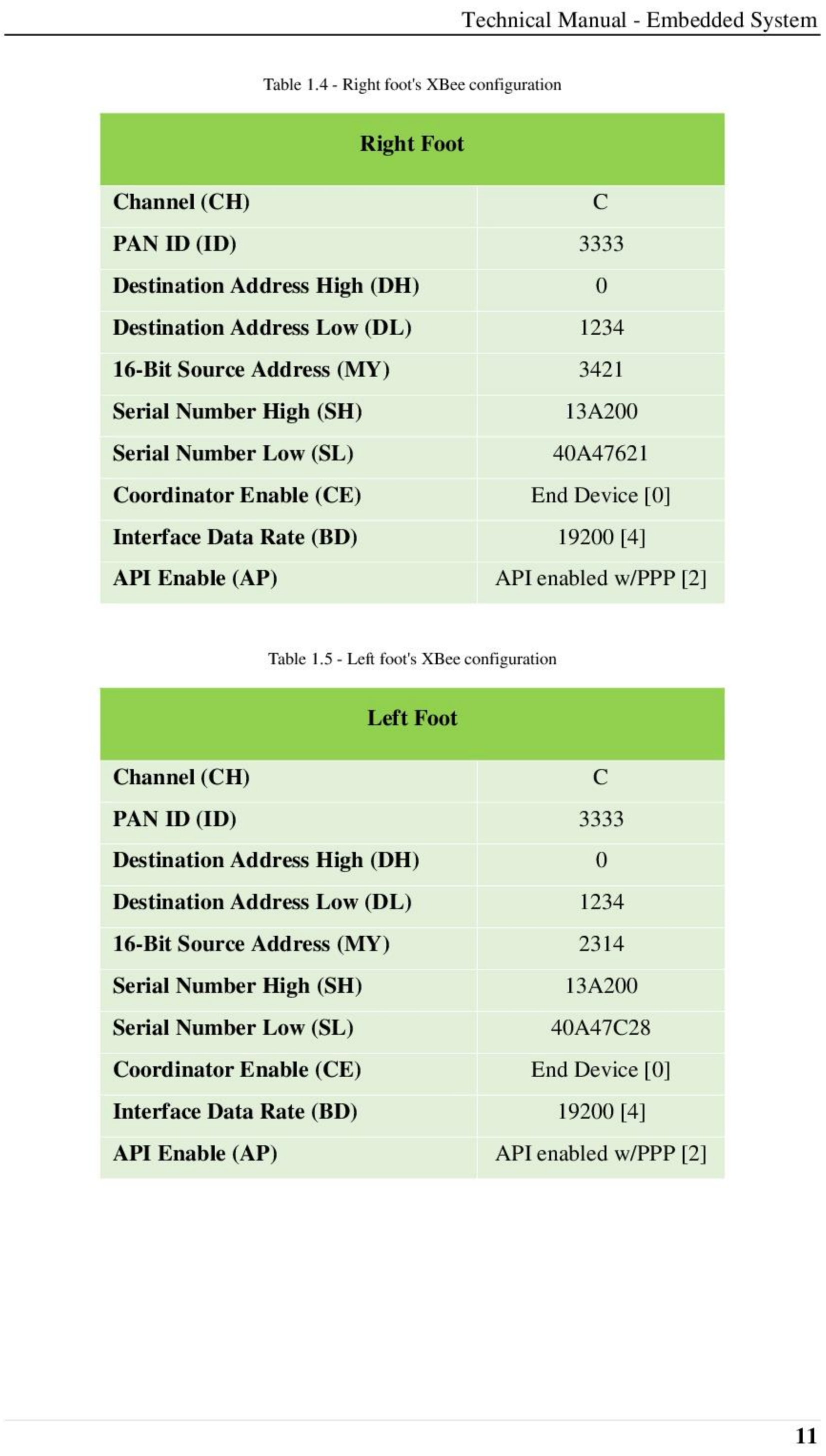


Table 1.4 - Right foot's XBee configuration

Right Foot	
Channel (CH)	C
PAN ID (ID)	3333
Destination Address High (DH)	0
Destination Address Low (DL)	1234
16-Bit Source Address (MY)	3421
Serial Number High (SH)	13A200
Serial Number Low (SL)	40A47621
Coordinator Enable (CE)	End Device [0]
Interface Data Rate (BD)	19200 [4]
API Enable (AP)	API enabled w/PPP [2]

Table 1.5 - Left foot's XBee configuration

Left Foot	
Channel (CH)	C
PAN ID (ID)	3333
Destination Address High (DH)	0
Destination Address Low (DL)	1234
16-Bit Source Address (MY)	2314
Serial Number High (SH)	13A200
Serial Number Low (SL)	40A47C28
Coordinator Enable (CE)	End Device [0]
Interface Data Rate (BD)	19200 [4]
API Enable (AP)	API enabled w/PPP [2]

Table 1.6 - IMU Cyclist's XBee configuration

IMU Cyclist	
Channel (CH)	C
PAN ID (ID)	3333
Destination Address High (DH)	0
Destination Address Low (DL)	1234
16-Bit Source Address (MY)	2341
Serial Number High (SH)	13A200
Serial Number Low (SL)	40A47C3F
Coordinator Enable (CE)	End Device [0]
Interface Data Rate (BD)	19200 [4]
API Enable (AP)	API enabled w/PPP [2]

Table 1.7 - IMU Bicycle's XBee configuration

IMU Bicycle	
Channel (CH)	C
PAN ID (ID)	3333
Destination Address High (DH)	0
Destination Address Low (DL)	1234
16-Bit Source Address (MY)	2431
Serial Number High (SH)	13A200
Serial Number Low (SL)	40F9DE39
Coordinator Enable (CE)	End Device [0]
Interface Data Rate (BD)	19200 [4]
API Enable (AP)	API enabled w/PPP [2]

Table 1.8 - Respiration and heart rate's XBee configuration

Respiration and heart rate	
Channel (CH)	C
PAN ID (ID)	3333
Destination Address High (DH)	0
Destination Address Low (DL)	1234
16-Bit Source Address (MY)	4231
Serial Number High (SH)	13A200
Serial Number Low (SL)	40A4877F
Coordinator Enable (CE)	End Device [0]
Interface Data Rate (BD)	19200 [4]
API Enable (AP)	API enabled w/PPP [2]

1.1.2 Sending data to the coordinator

The method in Figure 1.1 shows the loop of the heart and respiration end node.

```
void loop() {
    byte i = 0;
    int myBPM;
    now = millis();
    if (millis() - start > 15000) {
        while(millis() - now < 60000){
            analogValue0 = analogRead(0);
            val0 = map(analogValue0, 0, 1023, 0, 100);
            myBPM = pulseSensor.getBeatsPerMinute();
            if(val0 != 0){
                i++;
            }
        }
        payload[0] = flag1;
        byte resp = 0;
        resp = i/2;
        char flag1 = 'g';
        payload[1]=0;
        payload[1] = resp;
        payload[2]=0;
        payload[2] = myBPM;
        xbee.send(tx);
}   delay(200);
}
```

Figure 1.1 - *loop()* function of the heart and respiration rate end-node

In the code part, it is possible to see that the values began to be read fifteen seconds after the microcontroller is turned on and that the values are sent, through *xbee.send(tx)*, every minute.

1.1.3 Receiving data in the coordinator

Figure 1.2 shows the part of the code where the coordinator receives a package coming from the ZigBee network.

```
xbee.readPacket();
    if (xbee.getResponse().isAvailable()) {
        // got something
        if (xbee.getResponse().getApiId() == RX_16_RESPONSE ||
 xbee.getResponse().getApiId() == RX_64_RESPONSE) {
            // got a rx packet
            if (xbee.getResponse().getApiId() == RX_16_RESPONSE)
 {
                xbee.getResponse().getRx16Response(rx16);
                option = rx16.getOption();
                data = rx16.getData(0);
            } else {
                xbee.getResponse().getRx64Response(rx64);
                option = rx64.getOption();
                data = rx64.getData(0);

            }
        switch (incomingByte = char(rx16.getData(0)) ) {
```

Figure 1.2 - Receiving data in the coordinator

After the coordinator receives the package, the data contained in it is read and processed according to its information. All the end-nodes send a flag in the message that identifies the end-node where they come from.

1.1.4 Sending data to Firebase

In Figure 1.3 is part of the coordinator's code, this is the function that sends the values of the force sensors and accelerometers to the Firebase Database. This function is called whenever new values arrive at the coordinator.

```
void sendData(){


    now = "";
    now = timeNow();

    firebase ="";
    firebase = urlDataBase + urlJson + bar + tag + bar + inicio + bar + instantValues + bar + now;
    firebaseCoord = urlDataBase + urlJson + bar + tag + bar + inicio + bar + gpsValues + bar + now;

    readGPS();

    //hands:
    rhs1 = String(r_hand_sensor1);
    rhs2 = String(r_hand_sensor2);
    lhs1 = String(l_hand_sensor1);
    lhs2 = String(l_hand_sensor2);
    //Feet:
    rfs1 = String(r_feet_sensor1);
    rfs2 = String(r_feet_sensor2);
    rfs3 = String(r_feet_sensor3);
    lfs1 = String(l_feet_sensor1);
    lfs2 = String(l_feet_sensor2);
    lfs3 = String(l_feet_sensor3);

    //ACC Bicycle:
    pbs = String(pitchBicycle);
    ybs = String(yawBicycle);
    rbs = String(rollBicycle);

    //ACC Cyclist:
    pcs = String(pitchCyclist);
    ycs = String(yawCyclist);
    rcs = String(rollCyclist);

    lats = String(GPS.latitudeDegrees, 4);
    lons = String(GPS.longitudeDegrees, 4);
    alts = String(GPS.altitude);
    spes = String(GPS.speed);


    String hands = "Hands";

    //hands
    p.runShellCommand("curl -k -X PATCH " + firebase + ".json -
d '{ \""+rh1+"\" : " + rhs1 + ", \""+rh2+"\" : " + rhs2 + ",
\""+lh1+"\" : " + lhs1 +  ", \""+lh2+"\" : " + lhs2 +" }'");

    //feet
    String feet = "Feet";
    p.runShellCommand("curl -k -X PATCH " + firebase + ".json -
d ' { \""+rf1+"\" : " + rfs1 +  ", \""+rf2+"\" : " + rfs1 + ",
\""+rf3+"\" : " + rfs3 + ", \""+lf1+"\" : " + lfs1 + ",
\""+lf2+"\" : " + lfs2 +   ", \""+lf3+"\" : " + lfs3 +" }'");

    //ACC
    String acc = "ACC";
    p.runShellCommand("curl -k -X PATCH " + firebase + ".json -
d ' { \""+pc+"\" : " + pcs + ", \""+yc+"\" : " + ycs + ",
\""+rc+"\" : " + rcs +  ", \""+pb+"\" : " + pbs +  ",
\""+yb+"\" : " + ybs +  ", \""+rb+"\" : " + rbs + "}'");
}

```

Figure 1.3 - *sendData()* Function

1.2 Coordinator

The coordinator can be found on the frame of the bike. In the coordinator, there are three shields and a sensor: GPS, Yun Shield, XBee shield and the RFID reader. Figure 1.4 is the constitution of the coordinator.

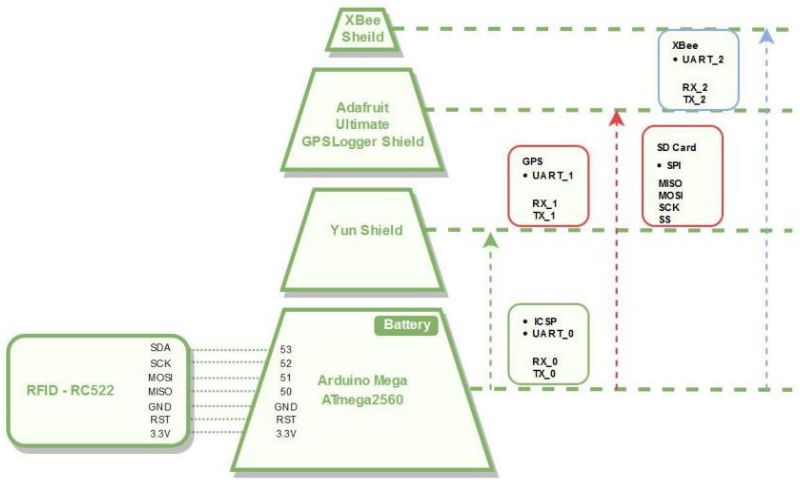


Figure 1.4 - Coordinator's assembly

1.2.1 RFID

The RFID is connected to the Arduino through the SPI protocol, as shown in Figure 1.4. This module is used to identify the cyclist and to trigger the coordinator to start collecting data and sent it to the database. The sensor used in this project is the MFRC522 with the MFRC522 library. To program the Arduino Mega via Wi-Fi, it is necessary to disconnect this module from the Arduino.

1.2.2 Yun Shield

The Yun shield is a very important part of this project because it is through it that is possible to send the data to the database. This shield communicates with the microcontroller through two communication protocols, ICSP and UART.

The UART between the mga2560 and the mega16u2 will influence the Bridge feature of the Yun shield, so it is necessary to disconnect it by setting the mega16u2 into reset mode, shown in Figure 1.5.

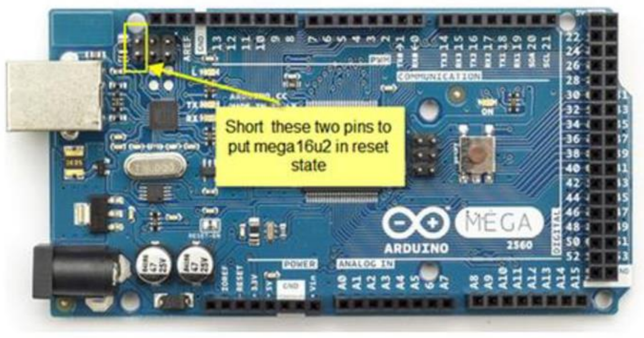


Figure 1.5 - Reset mode for the mega16u2

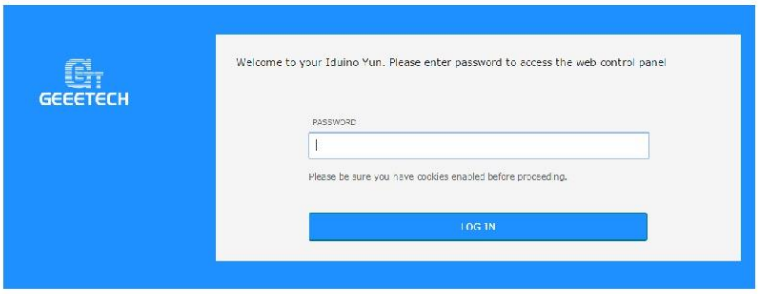
The Yun shield also allows to develop through a Wi-Fi connection, eliminating the need to use a USB cable.

To program the microcontroller with the Yun shield it is necessary to add the "Arduino Mega 250-Iduino Yun" board type (Figure 1.6) in the file: *C:\Program Files (x86)\Arduino\hardware\arduino\avr\boards.txt*.

```
mega2560Yun.name=Arduino Mega 2560 -- Iduino Yún
mega2560Yun.upload.via_ssh=true
mega2560Yun.vid.0=0x2341
mega2560Yun.pid.0=0x0044
mega2560Yun.vid.1=0x2341
mega2560Yun.pid.1=0x003f
mega2560Yun.upload.tool=avrdude
mega2560Yun.upload.protocol=arduino
mega2560Yun.upload.maximum_size=258048
mega2560Yun.upload.maximum_data_size=8192
mega2560Yun.upload.speed=57600
mega2560Yun.upload.disable_flushing=true
mega2560Yun.upload.use_1200bps_touch=true
mega2560Yun.upload.wait_for_upload_port=true
mega2560Yun.bootloader.tool=avrdude
mega2560Yun.bootloader.low_fuses=0xff
mega2560Yun.bootloader.high_fuses=0xd8
mega2560Yun.bootloader.extended_fuses=0xfd
mega2560Yun.bootloader.file=stk500v2/stk500boot_v2_mega2560.hex
mega2560Yun.bootloader.unlock_bits=0x3F
mega2560Yun.bootloader.lock_bits=0x0F
mega2560Yun.build.mcu=atmega2560
mega2560Yun.build.f_cpu=16000000L
mega2560Yun.build.board=AVR_MEGA2560
mega2560Yun.build.core=arduino
mega2560Yun.build.variant=mega
```

Figure 1.6 - Arduino Mega 250-Iduino Yun board type

To use the shield it is necessary that the computer is on the same network as the Yun shield and for this, it is necessary to configure it. The first thing to do is connect to the network, generated by the shield, to the computer as if it were a normal wi-fi network. Then, access the Webpage by entering http://192.168.240.1/, a login page will appear, Figure 1.7, the default password is “iduino”.

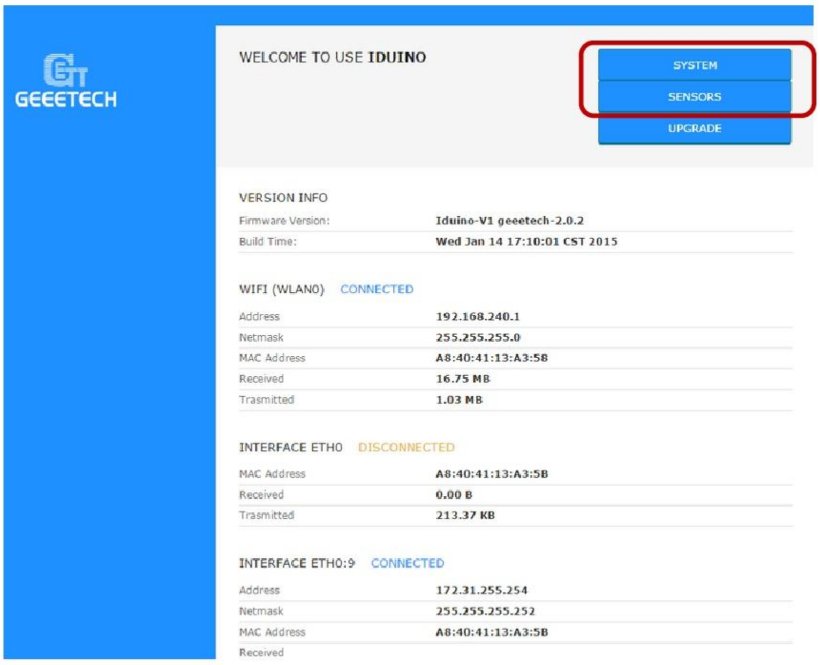


Welcome to your Iduino Yun. Please enter password to access the web control panel

Please be sure you have cookies enabled before proceeding.

Figure 1.7 - Yun shield login page

After the login, a page will appear with the wi-fi status, Figure 1.8.



WELCOME TO USE **IDUINO**

VERSION INFO

Firmware Version:	Iduino-V1 geeetech-2.0.2
Build Time:	Wed Jan 14 17:10:01 CST 2015

WIFI (WLAN0) CONNECTED

Address	192.168.240.1
Netmask	255.255.255.0
MAC Address	A8:40:41:13:A3:5B
Received	16.75 MB
Trasmitted	1.03 MB

INTERFACE ETH0 DISCONNECTED

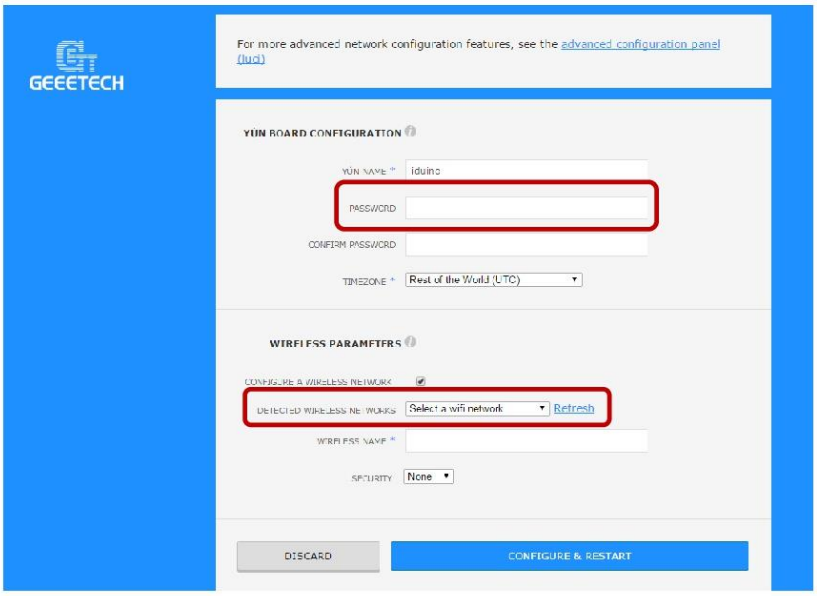
MAC Address	A8:40:41:13:A3:5B
Received	0.00 B
Trasmitted	213.37 KB

INTERFACE ETH0:9 CONNECTED

Address	172.31.255.254
Netmask	255.255.255.252
MAC Address	A8:40:41:13:A3:5B
Received	

Figure 1.8 - Yun shield webpage

By clicking on the System button, shown in the figure, it is possible to set a new password and configure the wi-fi parameters, Figure 1.9.



For more advanced network configuration features, see the advanced configuration panel (luci)

YUN BOARD CONFIGURATION

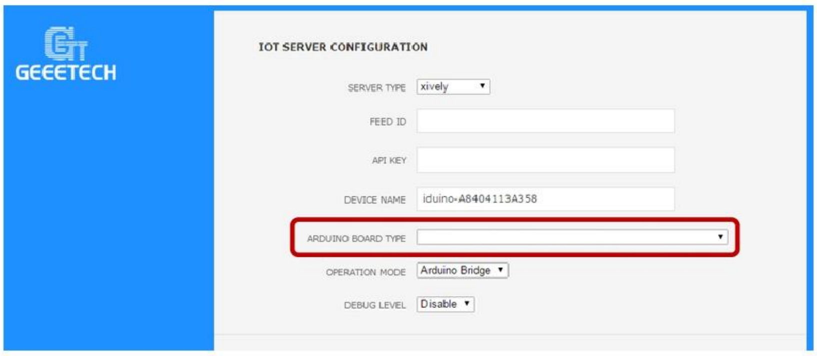
YUN NAME * iduino
PASSWORD
CONFIRM PASSWORD
TIMEZONE * Rest of the World (UTC)

WIRELESS PARAMETERS

CONFIGURE A WIRELESS NETWORK ✓
DETECTED WIRELESS NETWORKS Select a wifi network Refresh
WIRELESS NAME *
SECURITY None

Figure 1.9 - Yun shield configure page

It is also important to configure the Arduino board that will be used by clicking on the Sensors button (Figure 1.8). A page like the one in Figure 1.10 will appear, on this page, it should be selected the Arduino Mega 2560.



IOT SERVER CONFIGURATION

SERVER TYPE xively
FEED ID
API KEY
DEVICE NAME iduino-A8404113A358
ARDUINO BOARD TYPE
OPERATION MODE Arduino Bridge
DEBUG LEVEL Disable

Figure 1.10 - Yun shield configuration of the Arduino board

1.2.3 GPS Shield

The GPS shield collects the data such as coordinates, altitude, speed and time. This shield also supports a SD card that contains the RFID tags. The GPS operates independently of any telephonic or internet connection. This shield needs to be outdoors to work properly.

1.3 Arduino Files

Table 1.9 shows the Arduino files used to program each end-node and coordinator.

Table 1.9 - Arduino Files

File Name	Location	Description
ArduinoFioMinIMU9 AHRS_bicycle	**End-node** IMU bicycle	Contains the calculations for the DCM algorithm; Reads the accelerometer, the magnetometer and the gyroscope; Sends the structured data to the coordinator.
ArduinoFioMinIMU9 AHRS_cyclist	**End-node** IMU bicycle	
coordinator_wi-fi_c_RFID_auth	**Coordinator**	Contains the RFID authentication; reads the GPS values; Processes the information that comes from the ZigBee network; Sends the information to the database.
end-node_leftFeet	**End-node** left feet	Reads the analog values from the Flexiforce sensors; Sends the structures data to the coordinator.
end-node_leftHand	**End-node** left hand	
end-node_rightFeet	**End-node** right feet	
end-node_rightHand	**End-node** right hand	
HRM_Resp	**End-node** pulse sensor and respiration strap	Counts the number of respirations per minute; reads the values sent by the pulse sensor; Sends the structures data to the coordinator.

1.3.1 Libraries

In the programming of the end-nodes and the coordinator were used support libraries. All the libraries used are in Table 1.10.

Table 1.10 - Support Libraries

Library	Description	Location
Adafruit_GPS	Library to use the GPS shield	https://github.com/adafruit/Adafruit_GPS
Bridge	Library to use the Yun shield	https://www.arduino.cc/en/Reference/YunBridgeLibrary
L3G	Library to get the values from the gyroscope	https://github.com/pololu/l3g-arduino
LSM303	Library to get the values from the accelerometer and magnetometer	https://github.com/pololu/lsm303-arduino
MFRC522	Library to use the RFID	https://github.com/miguelbalboa/rfid
SD	Library to use the SD card in the shield	https://github.com/adafruit/SD
Xbee-Arduino_Library	Library to send and receive data through the XBee	https://github.com/andrewrapp/xbee-arduino
PulseSensorPlayground	Library to read the BPM values from the pulse sensor	https://github.com/WorldFamousElectronics/PulseSensorPlayground

Chapter 2 Mobile Application and Database

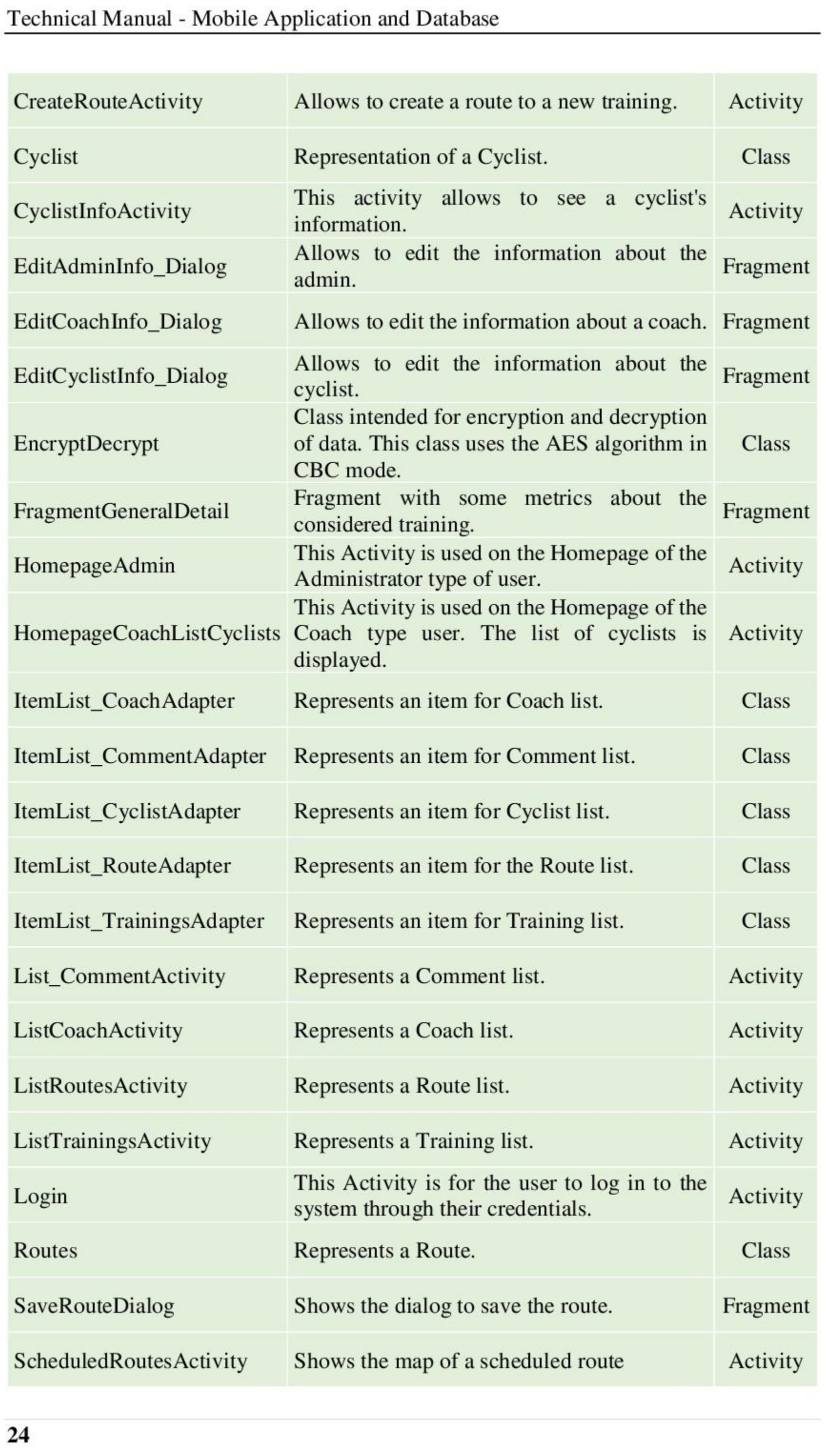
This chapter aims to explain, in terms of programming, the main functionalities and classes of the mobile application. The application was developed in Android Studio with version 3.1.4. The application was developed to run on devices with version 5.0 (Lollipop) or higher and the target version is 6.0 (Marshmallow). To run the application, the devices need to have internet access.

2.1 Java Classes

An Android application is composed by Activities that provide a screen for the user to interact with. Each Activity receives a window that displays a user interface and deals with the actions that the user makes. Activities can also support multiple Fragments that allows to change a part of the interface according to an action. Table xx shows the Java classes developed to the mobile application, description and respective category.

Table 2.1 - Classes used in the mobile application and their description

Class Name	Description	Category
AddCoachActivity	Allows to add a coach.	Activity
AddCyclistActivity	Allows to add a cyclist.	Activity
Admin	Represents an Administrator.	Class
Auxiliar	Has auxiliary strings and methods common to the whole application.	Class
AuxiliarResults	Has auxiliary strings and methods used in the training results Activities or Fragments.	Class
Charts	The fragment that represents the data in the charts.	Fragment
ChartStructeredData	Represents the data of the charts.	Class
Coach	Represents a Coach.	Class
CoachInfoActivity	Allows to see the information about a coach.	Activity
Comment	Representation of a Comment.	Class
CreateCommentActivity	Allows to add a new comment.	Activity



CreateRouteActivity	Allows to create a route to a new training.	Activity
Cyclist	Representation of a Cyclist.	Class
CyclistInfoActivity	This activity allows to see a cyclist's information.	Activity
EditAdminInfo_Dialog	Allows to edit the information about the admin.	Fragment
EditCoachInfo_Dialog	Allows to edit the information about a coach.	Fragment
EditCyclistInfo_Dialog	Allows to edit the information about the cyclist.	Fragment
EncryptDecrypt	Class intended for encryption and decryption of data. This class uses the AES algorithm in CBC mode.	Class
FragmentGeneralDetail	Fragment with some metrics about the considered training.	Fragment
HomepageAdmin	This Activity is used on the Homepage of the Administrator type of user.	Activity
HomepageCoachListCyclists	This Activity is used on the Homepage of the Coach type user. The list of cyclists is displayed.	Activity
ItemList_CoachAdapter	Represents an item for Coach list.	Class
ItemList_CommentAdapter	Represents an item for Comment list.	Class
ItemList_CyclistAdapter	Represents an item for Cyclist list.	Class
ItemList_RouteAdapter	Represents an item for the Route list.	Class
ItemList_TrainingsAdapter	Represents an item for Training list.	Class
List_CommentActivity	Represents a Comment list.	Activity
ListCoachActivity	Represents a Coach list.	Activity
ListRoutesActivity	Represents a Route list.	Activity
ListTrainingsActivity	Represents a Training list.	Activity
Login	This Activity is for the user to log in to the system through their credentials.	Activity
Routes	Represents a Route.	Class
SaveRouteDialog	Shows the dialog to save the route.	Fragment
ScheduledRoutesActivity	Shows the map of a scheduled route	Activity

SplashScreen	Android application SplashScreen.	Activity
TrainingData	Representation of a Training.	Class
TrainingDetailActivity	Activity that gets the result of a training a contains the general detail and charts fragments	Activity
User	Representation of a User.	Class

2.2 Programming Application Main Features

This section explains how some of the application main features were programmed. Each function or code parts are commented for better understanding.

2.2.1 Authentication

The following code (Figure 2.1) is the listener for the Login button, which triggers the user and password validation. The *singIn()* method makes the validation, this method receives as an argument the username and the encrypted password, so they can be compared to the fields in the database.

```
//listener do Botao login
    btn_singIn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            if(Auxilar.isNetworkAvailable(Login.this)){
                if(!edt_username.getText().toString().matches("")){
                    singIn(edt_username.getText().toString(), new
EncryptDecrypt().encryptedValue(edt_password.getText().toString()));
                }
                else{

edt_username.setError(getString(R.string.error_field_required));
                }
            }
            else {
                Toast.makeText(Login.this,
getString(R.string.error_internet),Toast.LENGTH_SHORT).show();
            }
        }
    });
}
```

Figure 2.1 - Login Button listener

2.2.2 Encryption

Figure 2.2 shows the EncryptDecrypt class contains the function to encrypt strings, in this case, the password.

```java
public class EncryptDecrypt {

    private final String key = "KEY";
    private final String salt = "SALT";
    private final byte[] iv = new byte[16];

    private final Encryption encryption;

    // Constructor
    public EncryptDecrypt() {
        this.encryption = Encryption.getDefault(key, salt, iv);
    }

    // Returns a String with the encrypted value
    public String encryptedValue(String value) {
        return encryption.encryptOrNull(value);
    }

    // Returns a String with the decrypted value
    public String decryptedValue(String value) {
        return encryption.decryptOrNull(value);
    }
}
```

Figure 2.2 - EncryptDecrypt Java class

2.2.3 Data Chart

Figure 2.3 is the code to populate the line chart. The procedure is called every time that a line needs to be drawn, for example, if the chart has four lines the procedure is called four times.

```java
private void populateLineChart(){

    final ArrayList<String> timeList = new ArrayList<>();
    timeList.addAll(chartStructeredData.get(0).getxTime());

    IAxisValueFormatter formatter = new IAxisValueFormatter() {
        @Override
        public String getFormattedValue(float value, AxisBase axis)
{

            return timeList.get((int) value);
        }
    };

    XAxis xAxis = lineChart.getXAxis();
    xAxis.setPosition(XAxis.XAxisPosition.BOTTOM);
    xAxis.setGranularity(1f); // minimum axis-step (interval) is 1
    xAxis.setValueFormatter(formatter);
    xAxis.setTextSize(10f);

    Description d = new Description();
    d.setText(AuxiliarResults.mapDescription.get(chartType));
    lineChart.setDescription(d);

    LineData lineData = new LineData(dataSetList);
    lineChart.setData(lineData);

    lineChart.setDrawGridBackground(true);
    lineChart.setBackgroundColor(Color.parseColor("#ffffff"));
    lineChart.setVisibleXRangeMaximum(6);
    lineChart.setExtraOffsets(11, 0, 11, 0);
    lineChart.invalidate();
    Legend legend = lineChart.getLegend();
    legend.setTextSize(15f);
    legend.setWordWrapEnabled(true);
}
```

Figure 2.3 – *populateLineChart()* procedure

2.2.4 Maps

Figure 2.4 shows the function that returns the string of the URL needed to work with the directions API, the function receives two Strings: origin and destination coordinates. To draw a route combining more than two points this function is called as many times as needed. When this code is called a HTTP connection is made to request the direction to the Directions API.

```
private String getRequestUrl(LatLng origin, LatLng destination) {
    String str_origin = "origin=" + origin.latitude + ","
            + origin.longitude;
    String str_destination = "destination=" + destination.latitude
            + "," + destination.longitude;
    String sensor = "sensor=false";
    String mode = "mode=bicycling";
    String param = str_origin + "&" + str_destination + "&"
            + sensor + "&" + mode;
    String output = "json";
    String url = "https://maps.googleapis.com/maps/api/directions/"
            + output + "?" + param + "&key=" + Auxilar.KEYMAPS;
    return url;
}
```

Figure 2.4 - Request URL function

2.2.5 General Information

Figure 2.5 and Figure 2.6 show two of the calculation made to the activity of the general information, the burned calories and distance. This activity has essentially averages, maximum values, the calories burned, distance and duration.

```java
public static String calculateCaloriesBurned(Double peso, String
tempo){
    double calories;
    String h = tempo.split("h")[0];
    String m = tempo.split("h")[1].split("min")[0];
    int time = Integer.parseInt(h)*60 + Integer.parseInt(m);
    calories = 0.066 * peso * time;
    return calories+"";

}
```

Figure 2.5 - Calories burned method

```java
public static String getDistance(ArrayList<Location> coordinates){
    float distance = 0;
    for (int i = 1; i < coordinates.size(); i++){
        distance += coordinates.get(i-
1).distanceTo(coordinates.get(i));
    }
    String total;
    DecimalFormat df = new DecimalFormat("#.00");
    if(distance>1000){
        double aux = distance / 1000;
        total = df.format(aux) + " km";
    } else {
        total = Math.round(distance *100) /100 + " m";
    }
    return total;
}
```

Figure 2.6 - Calculate distance method

2.2.6 Exporting the data

The inner class in Figure 2.7 extends an *AsyncTask* and it is used to wait for all the necessary data from Firebase to be loaded, this happens because the Firebase functions are asynchronous. Using the AsyncTask it is necessary to override some of the class methods, in this case, it was used the *onPreExecute*, *doInBackground* and *onPostExecute.*

The *onPreExecute* runs before the Thread start, here it is possible to show the user a loading message, for example.

The *doInBackground* is responsible for all the processing and it runs on a separated Thread. This is where the getOneLineData() method is called and the information is processed, the result is processed by the *onPostExecute* method.

The method *onPostExecute* receives the return of the *doInBackground* method. Its execution takes place on the same Thread as the user interface and it is here that the final processing regarding the obtained result is made.

```
private class AsyncTaskGetData extends AsyncTask<HashMap<String,
String>, String, ArrayList<ChartStructeredData>>{

    @Override
    protected void onPreExecute() {
        super.onPreExecute();

    }

    @Override
    protected ArrayList<ChartStructeredData>
doInBackground(HashMap<String, String>... strings) {
        ArrayList<ChartStructeredData> chartList =
getOneLineData(strings[0]);

        return chartList;
    }

    @Override
    protected void onPostExecute(ArrayList<ChartStructeredData>
chartStructeredData) {
        super.onPostExecute(chartStructeredData);

        if(optionToExport == false){
            fragment = Charts.newInstance(chartStructeredData);
            executeFragmentTransaction();
        }
    }
}
```

Figure 2.7 - Async Task to get the data

After all the data is obtained the exportDataToCSVFile() method is called to export to the file.

2.3 Database access

To access the database, it is necessary to ask the database owner for permission, it is possible to do this by sending an email to catarinadduque@gmail.com.After authorization is granted, access is made through https://leadride-fc9e1.firebaseio.com/.