

**PhysioAR – Smart sensing and Augmented Reality for Physical  
Rehabilitation**

João Pedro Duarte Monge

A Dissertation presented in partial fulfillment of the Requirements for the Degree of  
Master in Telecommunications and Computer Engineering

Supervisor:

Dr. Octavian Postolache, Assistant Professor, ISCTE-IUL.

October 2018



# Abstract

The continuous evolution of technology allows for a better analysis of the human being. In certain medical areas such as physiotherapy is required a correct analysis of the patient's evolution. The development of Information and Communication Technologies and recent innovations in the Internet of Things opens new possibilities in the medical field as systems of remote monitoring of patients with new sensors that allow the correct analysis of the health data of patients. In physiotherapy one of the most common problems in the application of treatments is the patient demotivation, something that today can be reduced with the introduction of Augmented Reality that provides a new experience to the patient. For this purpose, a system was developed that combines intelligent sensors with Augmented Reality application that will help monitor patient performance. This system is capable of monitoring lower limb movements acceleration, knee joint angle, patient equilibrium, muscular activity and cardiac activity using electromyography and electrocardiography with a wearable set of tools for easy utilization.

**keywords:** Augmented Reality; Serious Games; Physical Rehabilitation; Physiotherapy; Simblee.



# Resumo

A evolução contínua da tecnologia permite cada vez mais uma melhor análise do ser humano. Em certas áreas médicas, como a fisioterapia, é necessária uma correta análise da evolução do paciente. O desenvolvimento das Tecnologias de Informação e Comunicação, e as inovações no domínio de Internet das Coisas novas possibilidades no ramo da medicina, como sistemas de monitorização remota de pacientes com novos sensores que permitem a correta análise dos dados de saúde dos pacientes. Na fisioterapia um dos problemas mais comuns na aplicação dos tratamentos é a desmotivação do paciente, algo que hoje pode ser reduzido com introdução da aplicação da Realidade Aumentada que proporciona uma nova experiência ao paciente. Para isso nesta dissertação foi desenvolvido um sistema que combina sensores inteligentes com Realidade Aumentada que vai ajudar o paciente monitorizando o seu desempenho. Este sistema é capaz de monitorizar o ângulo do joelho, captar aceleração de movimentos dos membros inferiores, equilíbrio do paciente, atividade muscular e atividade cárdica usando electromiografia e electrocardiografia num conjunto wearable de fácil utilização.

**Palavras-chave:** Realidade Aumentada; Jogo Sérioo; Reabilitação Física; Fisioterapia; Simblee;



# Acknowledgments

I would like to thank my supervisor, Prof. Doctor Octavian Postolache for all the support given during this realization of this dissertation.

To the Institute of Telecommunications at ISCTE-IUL, for providing all the material and resources needed to develop this project.

To Académica de Santarém Football club for providing athletes and facilities for system testing.  
To physiotherapist Francisco Pratas for the help during a test session with his patients in Académica de Santarém Football Club.

To Lucas Viegas for support given during the developments.

And finally, to all my colleagues, family and friends for their constant support and motivation.

This research is supported by Instituto de Telecomunicações (IT-IUL) at ISCTE-IUL, Lisbon, Portugal and Fundação para a Ciência e Tecnologia (FCT) project PTDC/DTP-DES/6776/2014.

# Table of Contents

<b>Abstract</b> .....	<b>i</b>
<b>Resumo</b> .....	<b>iii</b>
<b>Acknowledgments</b> .....	<b>v</b>
<b>Table of Contents</b> .....	<b>vii</b>
<b>List of figures</b> .....	<b>x</b>
<b>List of Acronyms</b> .....	<b>xiv</b>
<b>1 Introduction</b> .....	<b>1</b>
1.1 Motivation and Overview.....	1
1.2 Research Questions and Objectives .....	2
1.3 Research methodology .....	2
1.4 Document structure .....	3
<b>2 Related work</b> .....	<b>4</b>
2.1 Sensors.....	4
2.1.1 Smart Sensors .....	4
2.1.2 Motion detection.....	5
2.1.3 Cardiac activity detection .....	9
2.1.4 Muscular activity detection .....	9
2.1.5 Wireless Sensor Networks (WSN) .....	10
2.1.6 Wireless Body Sensor Network (WBSN) .....	10
2.2 Virtual Environments .....	11
2.3 Applications.....	11
2.4 Physiotherapy rehabilitation systems .....	14
<b>3 System description</b> .....	<b>19</b>
3.1 Users and applications.....	20
3.2 PhysioAR System.....	21
3.3 Smart Sensors .....	23
3.3.1 Simblee Microcontroller.....	25



3.3.2	IMU (Inertial Measurement Unit) .....	26
3.3.3	ECG (Electrocardiography).....	27
3.3.4	EMG (Electromyography).....	29
3.3.5	RTC (Real-Time Clock).....	29
<b>4</b>	<b>Augmented Reality for Physiotherapy.....</b>	<b>30</b>
4.1	Apple ARKit.....	30
4.2	Adobe Fuse and Adobe Mixamo .....	31
4.3	Apple SceneKit.....	33
4.4	Game and sensors integration test .....	34
4.5	Stereoscopic view.....	35
<b>5</b>	<b>Smart Sensors Prototypes .....</b>	<b>37</b>
5.1	Motion smart sensor .....	37
5.1.1	Schematic.....	37
5.1.2	Acquisition Data .....	38
5.1.3	Prototype.....	40
5.1.4	Power Consumption .....	41
5.2	Motion + EMG Smart Sensor.....	42
5.2.1	Schematic.....	42
5.2.2	Acquisition Data .....	43
5.2.3	E-Textile shin pad electrodes .....	43
5.3	Motion + ECG Smart Sensor.....	45
5.3.1	Schematic.....	45
5.3.2	Power Consumption .....	46
5.3.3	E-Textile ECG electrode t-shirt.....	46
<b>6</b>	<b>System Backend .....</b>	<b>48</b>
6.1	Backend Application .....	48
6.1.1	Physiotherapist Login and Registration.....	49
6.1.2	Physiotherapist Main Page .....	53
6.1.3	Patient manager and register.....	53
6.1.4	Patient personal page .....	55
6.2	Cloud Server.....	58
<b>7</b>	<b>Mobile Application.....</b>	<b>60</b>

7.1	Login .....	60
7.2	Start Game screen.....	64
7.3	Starting game.....	66
<b>8</b>	<b>Results.....</b>	<b>71</b>
8.1	Preliminary tests .....	71
8.2	Laboratory tests .....	75
8.3	Clinical tests .....	83
<b>9</b>	<b>Conclusions and Future Work.....</b>	<b>86</b>
9.1	Conclusions .....	86
9.2	Future work .....	87
	<b>References.....</b>	<b>88</b>
	<b>Appendix A - Scientific Articles .....</b>	<b>91</b>
	<b>Appendix B – User Manual.....</b>	<b>97</b>
	<b>Appendix C – Technical Manual .....</b>	<b>116</b>

# List of figures

Figure 1.1- Dissertation planning diagram .....	3
Figure 2.1 - Kinect Sensor .....	5
Figure 2.2 - Tele rehabilitation system architecture.....	6
Figure 2.3 - Tele rehabilitation system description.....	7
Figure 2.4 - Results obtained .....	7
Figure 2.5 - Shimmer wearable IMU .....	8
Figure 2.6 - ECG signal sample.....	9
Figure 2.7 - System architecture .....	10
Figure 2.8 - Google fit application.....	12
Figure 2.9 - Apple Health application.....	12
Figure 2.10 - Strava application.....	13
Figure 2.11 - Blujay engage .....	14
Figure 2.12 - Gateway architecture.....	15
Figure 2.13 - Resulting angles of three patients .....	16
Figure 2.14 - "Collect Cubes" - Leap motion controller serious game interface.....	17
Figure 2.15 - Real world game implementation .....	17
Figure 3.1 - PhysioAR system architecture .....	19
Figure 3.2 - PhysioAR System flow .....	21
Figure 3.3 - User access card .....	22
Figure 3.4 - Smart Sensor Architecture.....	24
Figure 3.5 - Simblee microcontroller 7mm x 10mm x 2.2mm .....	25
Figure 3.6 - Lylipad Simblee board .....	26
Figure 3.7 - Flora 9DoF LSM9DS0.....	27
Figure 3.8 - AD8232 Functional block diagram .....	28
Figure 3.9 - Sparkfun AD8232 Development board.....	28

Figure 3.10 - Muscle Sensor V3 .....	29
Figure 3.11 - RTC Module .....	29
Figure 4.1 - ARCore example .....	30
Figure 4.2 - Apple ARKit example .....	31
Figure 4.3 - Mixamo character animation.....	32
Figure 4.4 - Xcode Scenekit development window .....	33
Figure 4.5 - Xcode SceneKit Editor.....	34
Figure 4.6 - First test football scenario .....	35
Figure 4.7 - Stereoscopic view.....	36
Figure 4.8 - PhysioAR Serious Game footage.....	36
Figure 5.1 - Smart Motion Sensor Schematic .....	37
Figure 5.2 - Roll, Pitch and Yaw Graph.....	38
Figure 5.3 - Acceleration vector of Shin movement .....	39
Figure 5.4 - Smart Motion Sensor Prototype .....	40
Figure 5.5 – Smart motion sensor power consumption .....	41
Figure 5.6 - Motion+EMG Schematic .....	42
Figure 5.7 - EMG E-Textile electrodes.....	44
Figure 5.8 - Motion+ECG Smart Sensor .....	45
Figure 5.9 - Smart ECG+Motion Sensor power consumption.....	46
Figure 5.10 - E-textile electrodes t-shirt .....	47
Figure 5.11 - Electrode cable plug .....	47
Figure 6.1 - Backend Login Page .....	49
Figure 6.2 - Backend user not exist message.....	50
Figure 6.3 - Backend - Wrong Password Message .....	50
Figure 6.4 - Physiotherapist registration page .....	51
Figure 6.5 - Physiotherapist registration failed.....	52
Figure 6.6 - Backend password recovery.....	52

Figure 6.7 - Backend Main page.....	53
Figure 6.8 - Backend patient registration.....	54
Figure 6.9 - Backend - Patient Manager .....	55
Figure 6.10 - Patient personal page/QRCode access card .....	55
Figure 6.11 - Backend train plan.....	56
Figure 6.12 - Backend train analysis.....	57
Figure 6.13 - Amazon Web Services.....	58
Figure 7.1 - Mobile App - Login Screen.....	60
Figure 7.2 - Registration succeeded email.....	61
Figure 7.3 - QRCode Login Screen .....	62
Figure 7.4 - Recover Password Screen .....	63
Figure 7.5 - Password recovery email.....	63
Figure 7.6 - Start Game Home Screen .....	64
Figure 7.7 - Progress screen.....	65
Figure 7.8 - Connecting sensors pop-up .....	66
Figure 7.9 – Bluetooth not enable.....	67
Figure 7.10 – Sensors connection failed.....	67
Figure 7.11 - Sensors connected successfully.....	68
Figure 7.12 - Countdown screen.....	68
Figure 7.13 - Google cardboard.....	69
Figure 7.14 - VR Universal smartphone box .....	69
Figure 7.15 - Game Screen .....	70
Figure 8.1 - Volunteer 1 results.....	72
Figure 8.2 - Volunteer 2 results.....	73
Figure 8.3 - Volunteer 3 results.....	74
Figure 8.4 - Participant 1 knee joint angles .....	75
Figure 8.5 - Participant 1 ECG .....	76

Figure 8.6 - Participant 1 EMG.....	76
Figure 8.7 - Participant 2 knee joint angles .....	77
<i>Figure 8.8 - Participant 2 EMG.....</i>	<i>78</i>
Figure 8.9 - Participant 2 ECG .....	78
Figure 8.10 - Participant 3 Knee joint angles .....	79
<i>Figure 8.11 - Participant 3 ECG.....</i>	<i>80</i>
Figure 8.12 - Participant 3 EMG.....	80
Figure 8.13 - Participant 4 knee joint angles .....	80
Figure 8.14 - Participant 4 EMG.....	81
Figure 8.15 - Participant 4 ECG .....	81
Figure 8.16 - Participants score .....	82
Figure 8.17 - Patient 1 results .....	83
Figure 8.18 - Patient 2 Results.....	84
Figure 8.19 - Patient 3 knee joint angles.....	84

# List of Acronyms

3G	3rd Generation
AR	Augmented Reality
AWS	Amazon Web Services
BPM	Beats Per Minute
ECG	Electrocardiogram
EMG	Electromyogram
IMU	Inertial Measurement Unit
IoT	Internet of Things
JS	JavaScript
LiPo	Lithium Polymer
OS	Operative System
OTA	Over the Air
SG	Serious Games
SoC	System On Chip
SS	Smart Sensors
VR	Virtual Reality
WBAN	Wireless Body Area Network
WLAN	Wireless Local Area Network

# 1 Introduction

## 1.1 Motivation and Overview

According to the World Health Organization, health is not just the absence of disease. It consists of the physical, mental, psychological and social well-being of the individual. It is a cumulative state, which must be promoted throughout life, in order to ensure that its benefits are fully enjoyed in later days [1] medicine is the science that has a fundamental role in the life of the human being, deals with the diagnosis, treatment, and prevention of diseases that affect the quality of life of the human being. It has branches in several areas such as biology, nursing, psychology, nutrition, physiotherapy, neurology, psychiatry among others in order to promote good - being of the human being. [2]

Unfortunately, the well-being of the human being is not always possible. The occurrence of road and work accidents, stroke attacks, physical disabilities, rare diseases as described in among other factors puts the humans in unexpected situations where they need medical care. Usually, these problems produce motor failure requiring a difficult and long process of physical rehabilitation or require constant monitoring of the patient which causes a certain dependence.

The process of rehabilitation or monitoring is a tedious process that plays a lot with the psychological state of the patient. These patients sometimes give up treatment making recovery more time consuming and difficult. [3] In order to deal with this problem, new platforms have emerged to speed up this process, some of them, such as described in [4] in which Internet of Things (IoT) new technologies are combined with Health monitoring technologies.

Internet Of Things uses the Wireless Sensor Network (WSN) technology to connect everything to the internet, this means that all our simple devices such as light switches, plugs, lamps, doors, vehicles among many things stay connected allowing to perform many tasks automatically and remotely [5]. But how does this apply to medicine? Good linking of the human body to the internet can allow monitoring parameters such as heart rate, respiratory rate, body temperature, movements among many other things which allows the creation of health monitoring system and alert in case of detection of health problems [4]. For this, there is a need for evolution of the devices that capture the vital signs so that they become comfortable, light and non-invasive which makes their use throughout the train possible and comfortable.



Development of wearable devices is required to respond to different challenges of IoT [6]. However, these devices require the use of clothing normally, which limits its use to the average user. In response to this problem, we think of the development of sensors that can be applied directly to the patient body, which allows the patient to wear any clothing. These sensors allow a continuous non-intrusive use for the patient. Besides giving freedom and greater efficiency (because being closer to the skin allows a better accuracy of the data captured), it covers a huge range of systems from applications in the sports, health and even virtual reality games.

## 1.2 Research Questions and Objectives

The objective of this dissertation is to develop a system for physical rehabilitation purposes using emerging technologies. The system is a set of smart/intelligent sensors and applications: A backend responsive web application and an interactive Augmented Reality mobile application. The objectives will be:

- Development of intelligent/smart sensors;
- Development of backend Application;
- Development of mobile Augmented Reality Application;
- Testing and implementation;

Some investigation questions:

- Will be the smart sensors easy for the patient to wear?
- Will be sufficient data and patients to correctly analysis the system?
- Will the Serious Game be effective in the physical rehabilitation area?
- Will the system adapt to the different types of patients?

## 1.3 Research methodology

The research method used in this dissertation consists of five distinct and sequential phases:

**1st Phase** – It consists of the clear definition of the objectives and the elaboration of a development plan. At the end of this phase, the planning of the system to be developed was completed, as well as the materials needed to elaborate on the project were clearly identified.

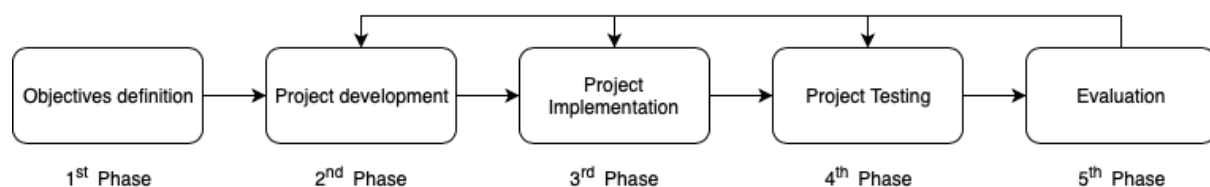
**2nd Phase** – It consists then in the development of the project. In this phase was taken advantage of all the material of the previous phase and was identified the details of all the components of the system as well as designed the logistics needed for the project.

**3rd Phase** – At this stage the implementation of the project begun with the materials elaborated in the previous phases.

**4th Phase** – At this stage of the project the entire system was tested. Professionals in the field of health monitoring supported testing and gave feedback. After testing the identified errors were corrected.

**5th Phase** – Finally, was the evaluation phase of the project.

**Note:** When its necessary to go back in any of the phases always was followed the order of the next phases without skipping any (Figure 1.1).



*Figure 1.1- Dissertation planning diagram*

## 1.4 Document structure

This document is organized as follow: Chapter 2 contains the related work on sensors and similar applications, chapter 3 includes the PhysioAR system description where the all system is described generally. Chapter 4 covers the Augmented Reality use in physiotherapy. Chapter 5 covers the development and design of the smart sensor. Chapter 6 cover all the cloud and backend parts of the PhysioAR system. Chapter 7 describes the mobile application which includes the Augmented Reality application. Chapter 8 presents some results and it discussion. Finally, Chapter 9 is the conclusions and future work.

## 2 Related work

This chapter presents the research on several relevant areas to this dissertation such as Wireless Body Area Network (WBAN), Smart Sensors for healthcare applications, data storage solutions, Mobile and Web Applications for healthcare, Virtual environments systems for physical rehabilitation and other solutions of relevant products in the market.

### 2.1 Sensors

**Sensor** is a device which measures or detects physical properties. There are several types of sensors in the market. In this chapter, a research into the relevant sensors for this project as well as some technologies available in healthcare is presented.

#### 2.1.1 Smart Sensors

A Smart Sensor is a sensor with some computational resources that empowers it to analyze data before sending it or perform other function that a simple sensor could not do.

An example of a smart sensor in the healthcare system is a smart sensor to detect the falls of the elderly [7]. Falls are a health hazard often presented in elderly population Andrew Sixsmith proposed a non-invasive system that detects human falls. In this work, an array of infrared detectors was used in combination with an embedded system and a neural network trained by Andrew to estimate vertical velocity and detect a fall. The infrared sensors array outputted thermal imaging which was able to detect human movement.

On this dissertation, we intend to use smart sensors to reduce computational resources on the server and also improve results, the work of Sixsmith that presents a smart sensor and its architecture for a healthcare system was used as a start for our smart sensors.

## 2.1.2 Motion detection

In physical rehabilitation, rigorous monitoring of body movements is required. There are many 3D motion detection systems available to choose from. It is necessary to analyze and identify the advantages that each one brings to this project. Here are some examples of sensors for motion detection:

**Non-Wearable** (Sensors that do not require contact with the user):

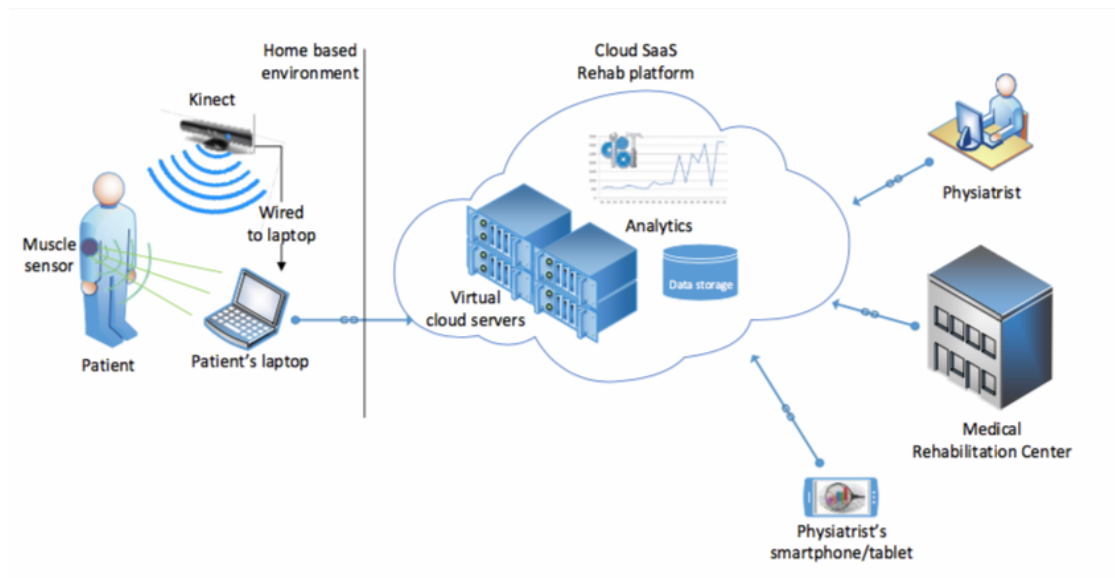
- **Microsoft Kinect sensor** (Figure 2.1) – The Kinect sensor incorporate advancing sensing hardware. It contains a depth sensor, the color camera a four-microphone array that provides full-body 3D motion capture, facial recognition, and voice recognition capabilities [8]. The low-cost and the SDK Microsoft provided to developers were the resources needed to adapt Kinect sensor to several different projects.



*Figure 2.1 - Kinect Sensor*

One good example of a project using the Kinect sensor is a Kinect Serious Game for physiotherapy [9]. In this work was presented a system that uses the Kinect sensor to acquire patient motion and a Serious Game to increase patient engagement and improve rehabilitation time.

Another example of using a Kinect sensor is a tele rehabilitation system presented by Sanja Vukicevic and Zoran Stamenkovic [10]. They presented a system that provides remote patient monitoring with the help of the Kinect they track body movements and EMG muscle sensors. They obtained good results from a combination of data on movements and muscular activity.



*Figure 2.2 - Tele rehabilitation system architecture*

As was mentioned before the motivation of patient is a problem in the rehabilitation process, they developed a serious game to augment the user experience. The system described contained IoT architecture as presented in Figure 2.2 that is close to what was developed in the present project.

Figure 2.2 describes the system architecture. The laptop acts as a gateway between the sensor (Kinect) to the cloud. The backend app stays in the cloud this provides easy updates and device compatibility since it is a web application any device with a compatible browser could access it.

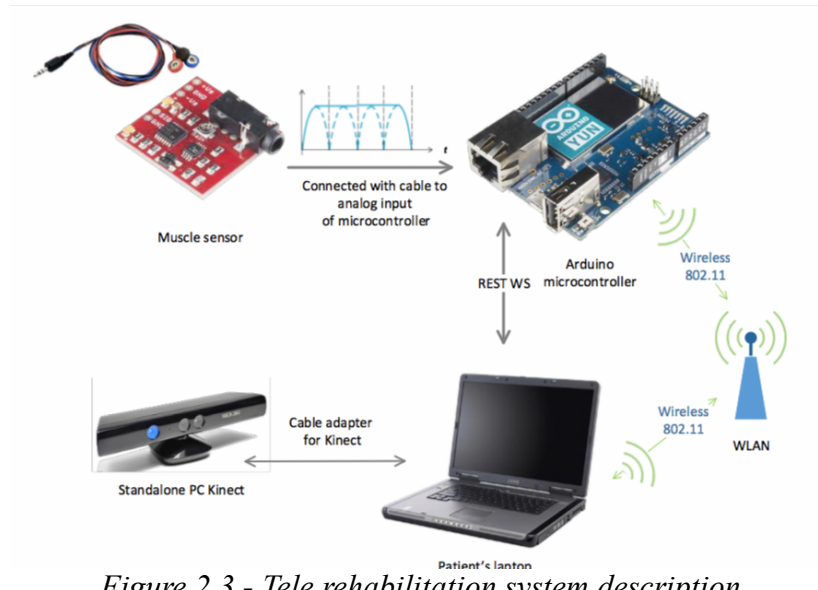


Figure 2.3 - Tele rehabilitation system description

Figure 2.3 describes the system components. A Muscle Sensor V3 board connected to Arduino Yun and transmitted data through a Wifi connection. With this system, the patient could execute the exercises in his home with the help of a virtual assistant which is a good thing if the patient lives in a remote area or if the clinical center is far away from his home.

The results obtained (Figure 2.4) in this work are positive showing a better hand movement, in a three-week playing the serious game, of a 60-year-old patient who suffered from right hemiparesis due to a stroke attack. The muscle activity was not recorded as the patient could not close his hand and therefore no contraction on the biceps was produced. There was proof of concept with this system showing us that is possible to obtain good results.

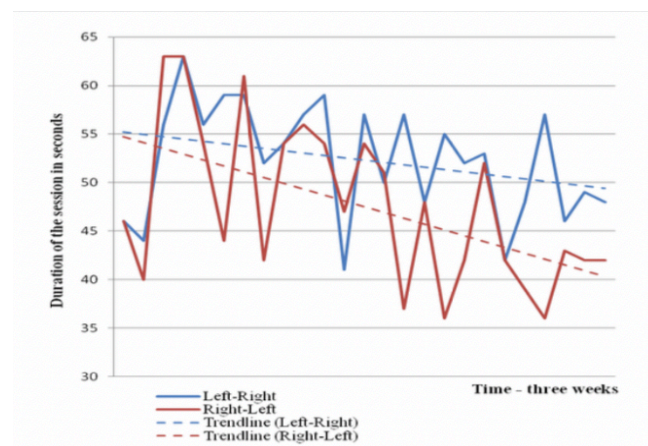


Figure 2.4 - Results obtained

As Kinect should be used only indoor was necessary to research more alternatives for outdoor/indoor like wearable sensors.

**Wearable** (Sensors that need to be attached to the user):

o Shimmer3 IMU - Shimmer is a commercial brand of sensors for healthcare applications they are lightweight, low-power, wirelessly enabled sensor platform which can be used as wearable devices [11]. The Shimmer3 IMU offers best data quality with integrated 9DoF inertial sensing via accelerometer, gyroscope, magnetometer and pressure sensor, each with selectable range. Shimmer3 also boasts an integrated motion processor for on-board 3D orientation estimation. 5 colored LEDs indicate device status and operating mode, as well as indicating Bluetooth streaming functionality [12]. In Figure 2.5 there is an example of a wearable shimmer3



*Figure 2.5 - Shimmer wearable IMU*

oXSensMt1–XSens brand is the leading innovators in 3D motion tracking. XSens Mt1 is a full-featured 3D Attitude and Heading Reference System (AHRS), Vertical Reference Unit (VRU) and Inertial Measurement Unit (IMU) and it is small enough to be used in wearable applications[13].

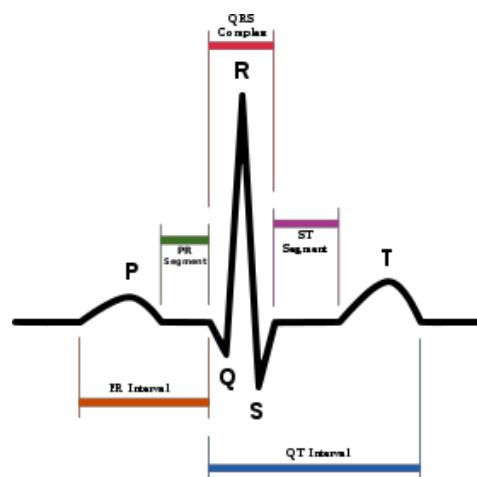
### 2.1.3 Cardiac activity detection

Cardiac activity is the principal vital sign of the human. In physical rehabilitation, it is important to understand the fatigue of the patient. The cardiac activity could indicate several situations like fatigue, exhaustion. Therefore, there are many reasons to include it in this project.

An electrocardiogram measures the bio-potential generated by electrical signals that control expansion and contraction of heart chambers. It can detect several heart disorders an example of the originated pulse is represented in Figure 2.6.

These technologies are used in hospitals and clinics to detect heart diseases.

In the work presented [14] represents a system that consists of a non-contact wearable ECG system for long-term monitoring featuring dry electrodes. The system used Bluetooth to send the ECG data. In the present project was added ECG continuous monitoring using E-textile electrodes similar to this project, but we intend to make them easier to wear.



*Figure 2.6 - ECG signal sample*

### 2.1.4 Muscular activity detection

Muscles can be very affected before physical rehabilitation, and for this reason, it is important to have a way to detect if they are improving their functionality. Clinics and hospitals have been using EMG to detect muscular and neural problems.



Electromyography is a diagnostic method that evaluates nervous or muscular problems. This technique uses surface electrodes to assess the ability of nerve cells to transmit electrical signals. It also can detect muscle contraction. In the work [15] is an example of a game for post-stroke rehabilitation using EMG data. A small robot guides the train and the patient muscles contraction act as controllers for the game.

### 2.1.5 Wireless Sensor Networks (WSN)

Innovations in technology bring smaller sensors and more efficient wireless communication. WSN's are networks of sensors that can communicate with each other or to the main device wirelessly. IoT will connect even more sensors to the internet creating a bigger network of sensors.

### 2.1.6 Wireless Body Sensor Network (WBSN)

This WBSN are specific human body related sensor network. In this dissertation is described a Wireless Body Area Network of EMG, ECG and Motion wearable sensors. An example of a WBSN for physical rehabilitation is the work purposed in by Emil Jovanov [16].

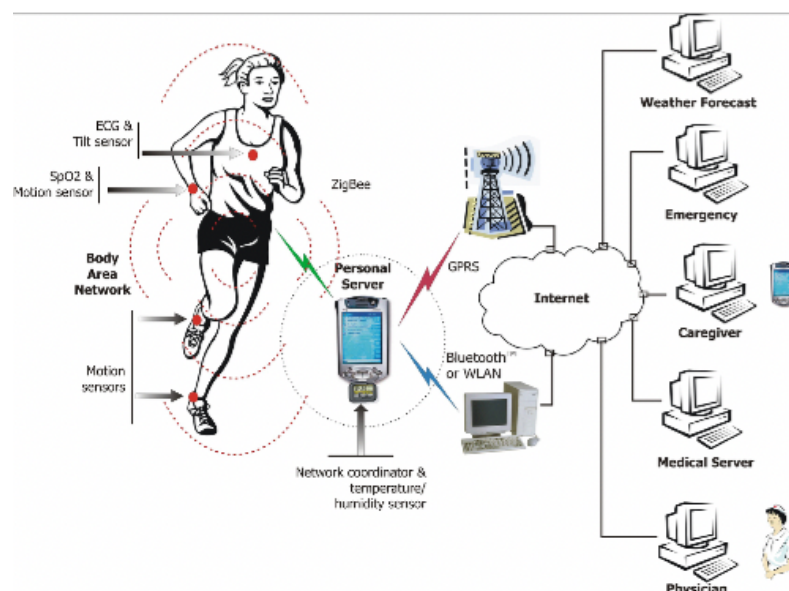


Figure 2.7 - System architecture

Figure 2.6 represents the architecture of the system. The figure contains the sensing units which include motion, ECG, and SpO2. The sensors communicate through Zigbee network coordinator. The data acquired by the sensors is then uploaded to the internet.

## 2.2 Virtual Environments

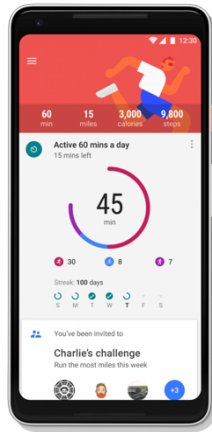
Motivation is a very important factor for rehabilitation success. Maintaining a patient motivated could be a hard job due to the repetitive and exhaustive nature of the rehabilitation exercises that physiotherapy demand. With this in mind in this dissertation will be considered several approaches that increase patient engagement:

- Video games can be used to help people improve their health. For example, Wii Fit Plus [17] uses a smart balance and guides the user during the train. Serious Games are games made for users to learn and even improve their skills rather than just pure fun or entertainment.
- Virtual Reality is an emerging technology that let the user enter a virtual world. In the work [18] a Virtual reality Serious Game for arm rehabilitation was purposed and consisted in a game where the patient has a virtual sword and has to hitboxes in order to win. The boxes are positioned in different levels with increased difficulty of movement.
- Augmented Reality consists of placing virtual objects in the real world. This could be a better approach vs the Virtual Reality for physical rehabilitation purposes since the patient sees the real world this reduced the risk to hit objects or fall. Augmented Reality based upper limb rehabilitation system [19] consisted in a recognizing form in the real world to place virtual objects above them and the user interacted with them by moving the forms.

## 2.3 Applications

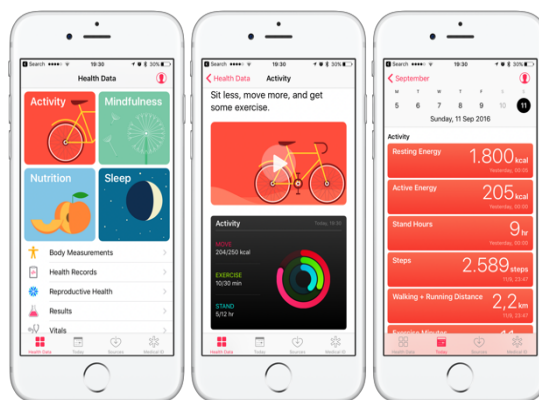
There are many applications available on the market that track human activity. Here are some examples of some of the most common ones:

- **Google fit** (Figure 2.8) - “From swimming to strolling, any activity that gets you moving makes an impact on your health. That is why Google Fit works with many of your favorite apps and health devices to give you credit for all your moves and provide a holistic view of your health.” [20]



*Figure 2.8 - Google fit application*

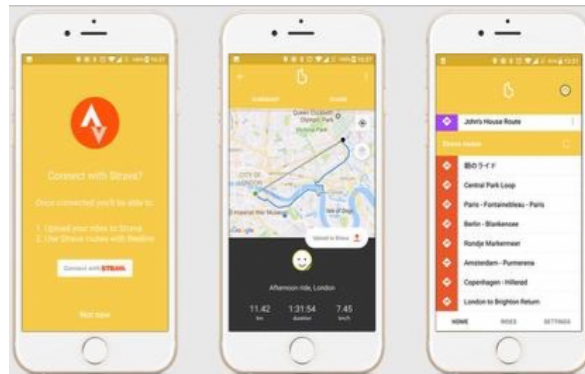
- **Apple Health** (Figure 2.9)- “The Health app highlights four categories: Activity, Sleep, Mindfulness, and Nutrition. Each plays an important role in your overall health — and in the app. Health suggests great apps from each category to get you going, and the Today view shows all your stats at a glance to help you stay on track. “ [21]



*Figure 2.9 - Apple Health application*

- **Strava** (Figure 2.10)- “Register your races, rides, and activities for free:
  - Activity control: during and after an activity, obtain key statistics such as distance, pace, speed, gain and calories burned, as well as an interactive map of your activity;
  - Personal Challenge: Participate in the monthly Challenges designed to encourage;

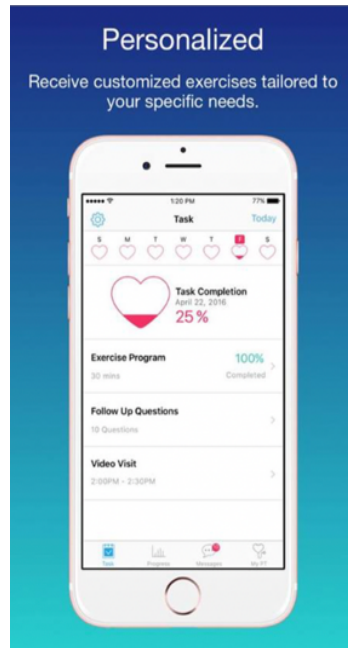
- Segments: Is there a climb or a long straight line? See your performance in certain sections of your activity with the Strava segments;” [22]



*Figure 2.10 - Strava application*

Although this is not a popular application is one of the unique application for physical rehabilitation purposes:

- **BlueJay Engage** (Figure 2.11) - “BlueJay Engage – Patient is your trusted companion for physical therapy and rehabilitation, guiding you through each step of your rehabilitation journey. This app allows you to:
  - Receive home exercise programs or HELP from your therapist right to your phone or iPad;
  - Chat with your therapist about using multimedia messages;
  - Log your progress to monitor pain and movement;
  - Browse the video library to create your own treatment plan to address your pain;” [23]



*Figure 2.11 - Blujay engage*

## 2.4 Physiotherapy rehabilitation systems

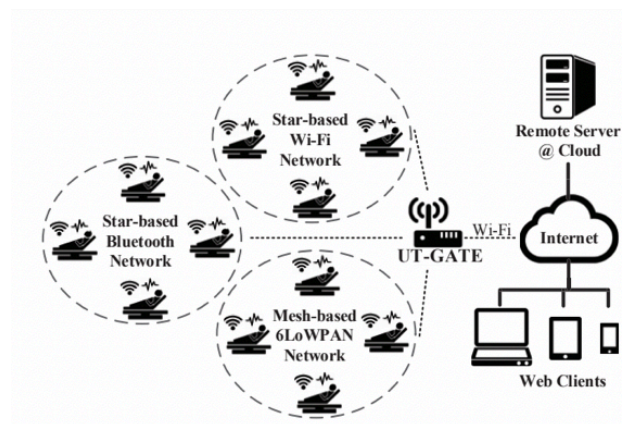
As more systems are emerging it becomes necessary to have a certain compatibility of devices. For example, different types of sensors have different communication protocols and in a healthcare system, it may be needed to connect them all. With that in mind, Rahmari [24] proposed a gateway with the objective of making a more compatible system for health use. In their work, it is possible to identify some of the problems pointed such as the necessity to create a system that is compatible with different network protocols such as wifi, Bluetooth, ZigBee etc. To solve the issue a gateway that will act between the medical sensor network and the backend app was developed with some important points in mind like:

- Data compression which is particularly important due to the quantity of data obtained;
- Data filtering was also pointed that is very important to correct sensor data and select only relevant data which can help decongesting networks;
- Local storage for backup access in case of the network going down - this is particularly important to have in mind since a hospital should not stop working if the network goes

down or it should not be very affected in terms of effectiveness since nowadays medics rely a lot on IT systems;

- Security was also pointed as a key point in an IoT healthcare system and was taken into consideration in the developed system. Since health data is critical and should be maintained private as the mandatory condition is important to understand the security flaws of the IoT systems and take the best security mechanism available to minimize them considering every precaution possible.

The architecture of the gateway is shown in Figure 2.12.



*Figure 2.12 - Gateway architecture*

As can be seen, the proposed gateway is named as UT-GATE and it is responsible for handling the data coming from different sensors types maximizing the compatibility of the systems then the gateway communicates through wifi with the remote server that contains the backend applications. This system is widely compatible, and the UT-GATE is responsible for several operations like data compression, data filtering. In this work, there is lots of information to take in consideration during an IOT HealthCare system development.

Another work to take in consideration is an Adaptative Mixed Reality Training System for stroke Rehabilitation presented in [25], proposing an Augmented/Mixed Reality Serious game for physical rehabilitation for a patient that suffer from hemiparesis due to a stroke attack. In this work, it is demonstrated that and Augmented Reality Serious Game is capable of capturing motivation of a patient using musical and visual environments and providing instant feedback to the patient to motivate him and help him self-correcting. This system used IR camera for detection upper limb motion detection. The major kinematic parameters obtained were measurement of goal completion, speed, trajectory, accuracy, velocity profile, range of joint

angles, joint coordination and compensatory shoulder and torso movements. The results obtained are in Figure 2.13.

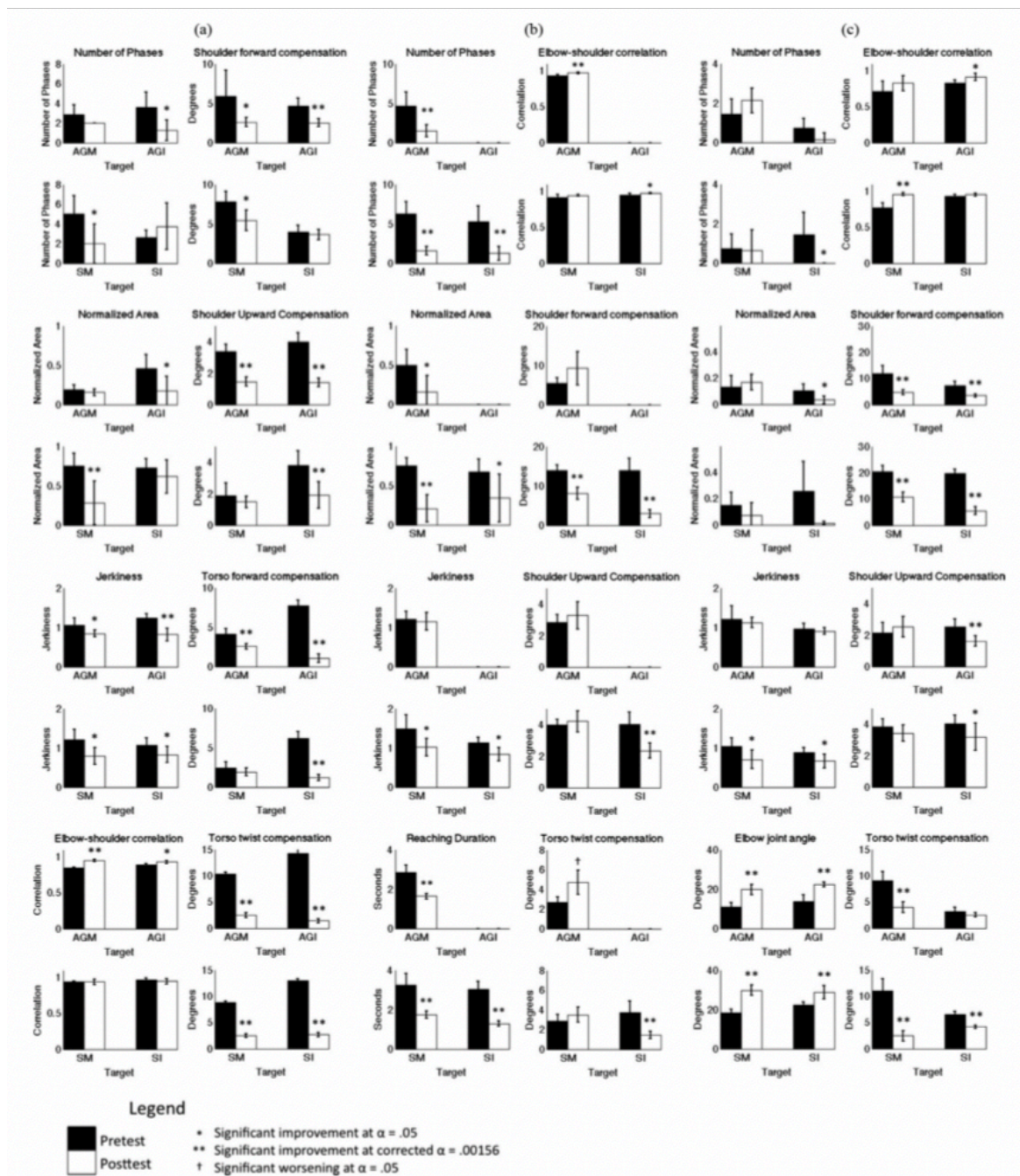
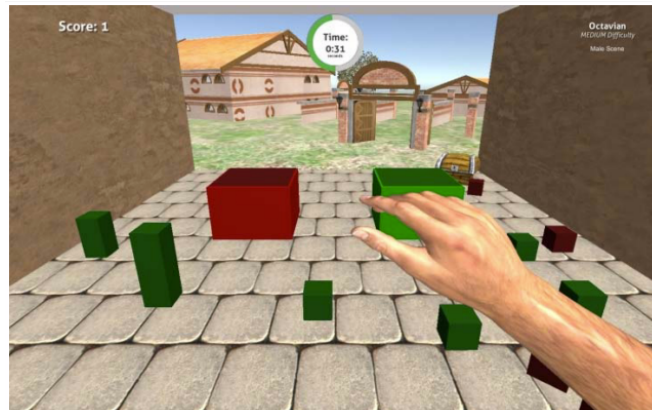


Figure 2.13 - Resulting angles of three patients

The results show an overall improvement in all three patient which carried the tests during 6 sessions of 75 minutes, for two weeks. However, for better understanding, the effectiveness of the games for patient exercising in motor rehabilitation, the motivation of the patient should be considered.

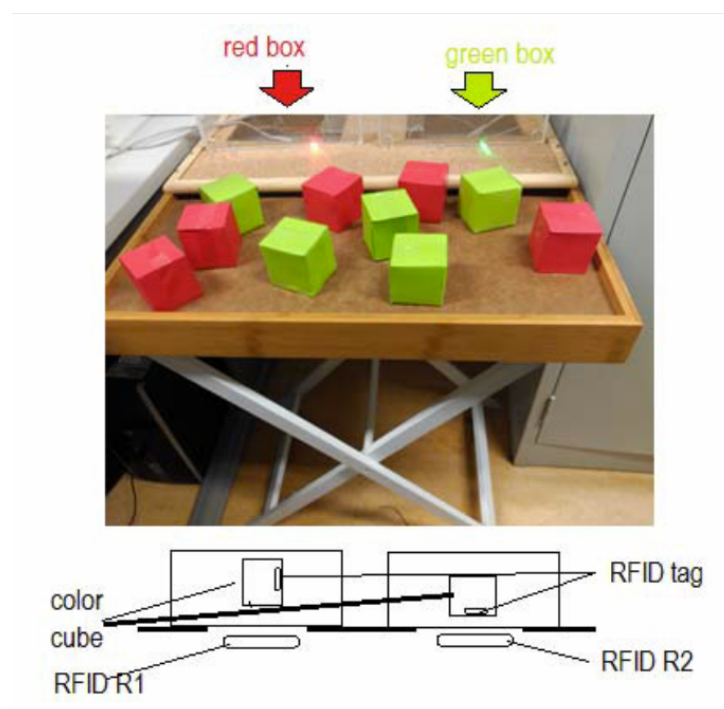
Serious Games was implemented by Filipe Lourenço, a member of our team to understand the role of real and virtual environment on the motivation for exercising during physical



*Figure 2.14 - "Collect Cubes" - Leap motion controller serious game interface*

rehabilitation [26]. The main objective was to compare both scenarios and see if the virtual one was similar or better than the real one. The virtual environment consists in picking up colored boxes and dropping them in a specific place as shown Figure 2.14.

A leap motion sensor was used to capture the hand movement in unobtrusively way and the game was produced using the Unity Engine. The patient obtained feedback by getting a score for successfully dropping the box in the right place. The effect on patients of exercising using



*Figure 2.15 - Real world game implementation*



serious game was measured using thermography of the hand at the end of the exercise obtained with a FLIR Camera.

For the real scenario, a replica of the game was made with real boxes and an RFID system to detect if the boxes were correctly dropped as shown in Figure 2.15.

The evaluation was made in a similar way to the ones in the virtual game using a thermography and then compared in final. This work shows new technologies adaptation to physical rehabilitation and that is possible to obtain relevant metrics from it.

An objective of this project proposal focuses on studying how the patients recover faster by analyzing and combining data originated from all patients training sessions. Therefore was needing to use data mining techniques due to the extensive amount of data originated by the sensors. In the work by Hian Chye Koh and Gerald Tan A [27] is described data mining algorithm for healthcare. In their work, it is mentioned several situations where data mining could improve healthcare such as healthcare management, fraud detection, and treatment effectiveness. Treatment effectiveness detection could lead to discovering which drugs work better on certain pathologies as also which are more cost-effective. One of the objectives of the present project was an analysis of the physiotherapy exercises that work better and produce more improvements. Data mining is a complex area and there is a need to understand the difficulty in the area as there is a need to obtain a certain amount of data to obtain clear results. To achieve this is required a lot of data from clinical examinations of patients.

In the present work new Serious Games that include Augmented Reality technologies was developed and tested to provide new ways of interaction between the patients and the rehabilitation scenarios which will improve physical rehabilitation outcome. An important part of the work is granted to the IoT healthcare systems, for physical rehabilitation and for improved quality of services.

### 3 System description

The PhysioAR is a system developed to assist patients and physiotherapist during rehabilitation of the lower limb and gait. The patient executes exercises following instructions configured by the physiotherapist which then receives the results of the patient performance. Figure 3.1 illustrates the physical system architecture divided into three blocks.

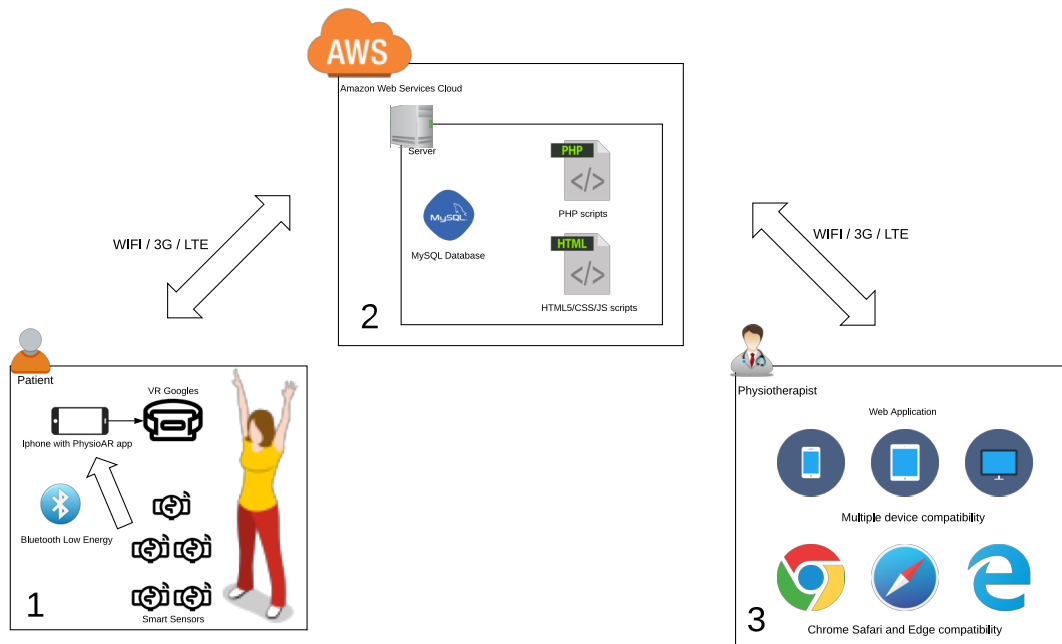


Figure 3.1 - PhysioAR system architecture

The first block (block “1”) is dedicated to the patient and is composed by smart sensors which include EMG, ECG and motion data acquisition units, an IOS smartphone and a VR Googles universal case. In block “1” patient has access to the Augmented Reality Application. Block “2” consists in the main server and it is hosted in AWS and contains a MySql database where all the patient information and results are stored. It contains also the PHP scripts to interact with the database and the Web Application scripts written in HTML5, JS (JavaScript) and CSS. Finally, block “3” is dedicated to the physiotherapist and explains how the access to the backend application is done. Any device with a compatible web browser (Chrome, Safari, Edge) allow the physiotherapist to access his patient's information and results either by a tablet, smartphone or computer. This chapter describes the operations between the system blocks and the system user interactions.

## 3.1 Users and applications

This system is intended to be used by two types of users: one is the *physiotherapist* and the other is the *patient*. The physiotherapist interacts with the system with a backend web application that allows managing his patients and configuration of the training plan. The patient uses the mobile application (that contains the augmented reality environment) to follow a training plan assigned by his physiotherapist.

Two applications were developed for the PhysioAR system:

- **Mobile Application:** Developed in Swift 4 and ARKit framework. This serves as a serious game for the patients to perform the training. The Serious Games are configurable by the physiotherapist and take in consideration metrics like the age of patients and his treatment requirements. These games are highly interactive and have a virtual assistant that guides the train. The patient uses smart sensors transmitting the motion of the lower limb as well as other vital data such as cardiac activity and muscular activation;

This requires an internet connection for database communication and data storage. This connection can be done by Wifi, 3G or 4G/LTE.

- **Backend Web Application:** Developed in HTML5, JavaScript and CSS to ensure maximum device compatibility. This app allows the Physiotherapist to manage his patients and assign a customized train plan to each one. It also allows to consult training sessions results and the overall results (actual user improvement by combining all training session results);

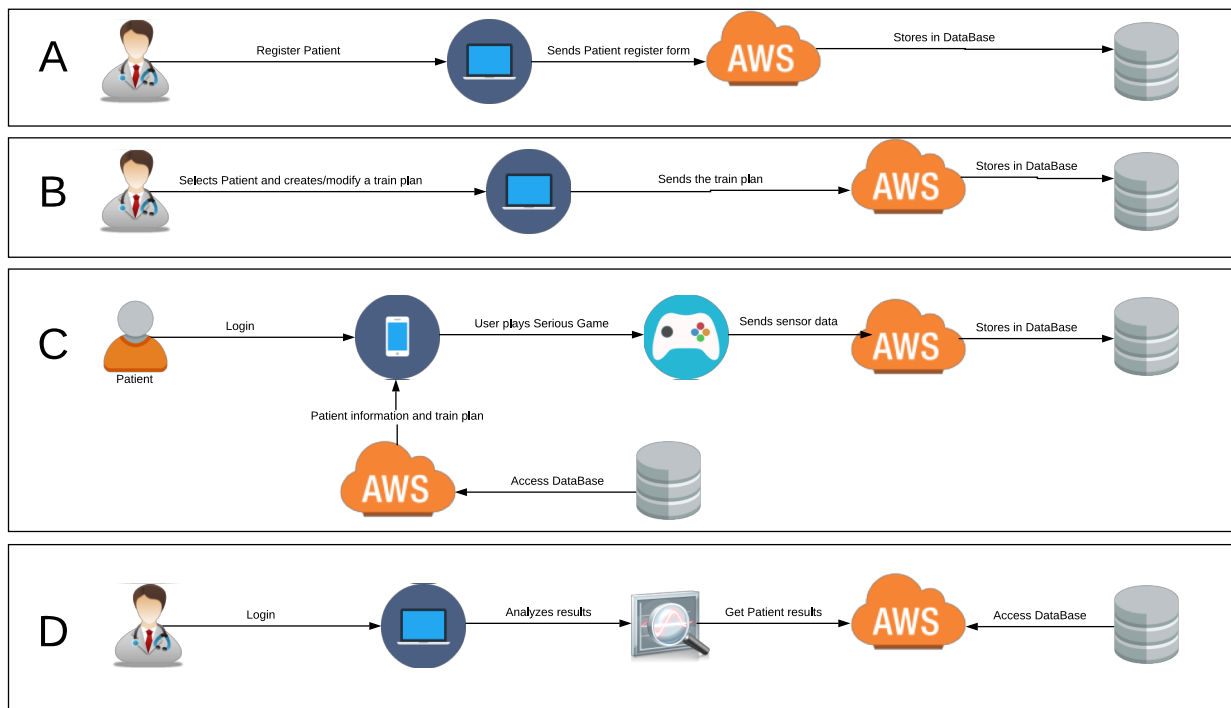
This requires an internet connection for database communication and data storage. This connection can be done by Wifi, 3G or 4G/LTE.

The system users detailed description:

- **Patient:** This is the user of this system which is under physical rehabilitation needs. This patient corresponds to the mobile application user and will perform the train assigned by the physiotherapist;
- **Physiotherapist:** This is responsible for monitoring his patients and inflicts the training plan. Using the Backend Web Application, they can analyze patient trains, change or insert patient data and add/register patients into the system.

## 3.2 PhysioAR System

Figure 3.2 illustrates some PhysioAR user interactions. A time flow is considered by the order of the blocks from A to D. Although not illustrated in the Figure 3.2Figure 1.1 the first step is the Physiotherapist registration which can be done in the login page accessible to everyone on the address <https://www.physioar.tk> (**note:** this is temporary domain that can be closed anytime at the end of this dissertation development time due to associated costs that will be no longer supported as this project reached the end). It requires a user photo (there is an option to take it directly in the browser using a webcam or smartphone camera), complete name, email, password, birthday, professional ID, professional certificate copy and finally a contact. After the physiotherapist does this step the system is ready to start registering patients.



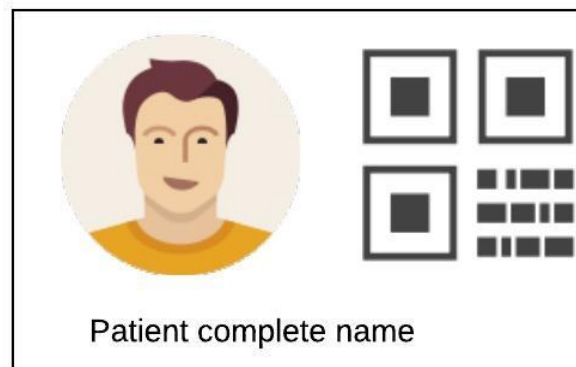
*Figure 3.2 - PhysioAR System flow*

Patients can be registered according to the flow presented in block “A”, physiotherapist login into his account and there is option to register a patient after a successful login. The patient registration fields needed to fill are:

- **Photo:** Webcam or camera access needed to take the photo directly in the application;
- **Name:** Complete name of the patient is required;
- **Password:** Patient can define his password;

- **Birthday:** Patient birthday;
- **Height:** Patient height is used to render the game according to the position of the VR goggles to the ground;

Concluding registration of a patient a QRCode is generated that provides an easy login option for a patient with less mobility. This QRCode is a cheaper option compared to RFID or NFC tags because it just requires software and no additional hardware. A physiotherapist can then print a card like the one presented in Figure 3.3 and deliver to his patient.



*Figure 3.3 - User access card*

The next block “B” represents the next step a physiotherapist is required to do. The physiotherapist should perform the patient complete analysis before assigning any plan to him and is responsible to insert a correct clinical status of the patient in the system. As the flow inside the block “B” indicates the user must select the patient from the patient manager page, and then after selecting it a redirection to that patient personal page occurs. In this page is possible to create the training plan. To do so the physiotherapist should select the exercises from a long palette of defined exercises and drag and drop them in the correct order in a timeline window. After dropping each exercise a pop-up shows in which the physiotherapist should indicate the duration of that particular exercise. After creating the train plan is also possible to modify it anytime and the changes are updated immediately.

After the patient is registered and a training plan is assigned to him, he can start using the mobile application and execute his training session. The block “C” represents the user interaction flow with the mobile application and serious game. The patient at first has to login in the application to do so there are two options, one is a username/password combination and the other is a QRCode, after successful login the user can view his progress or start playing the game.

Selecting the progress, the user can see last scored points otherwise selecting “start game” the game starts. After the user finishes the train session the game ends and the total score is shown.

After each train is completed the physiotherapist can access the system to analyze the patient results block “D” describes the flow of that interaction. As described the physiotherapist as to log in using his credentials, select the pretended patient from the patient manager page and then select the “train analysis” page where he can select the train by date or see the overall progress which combines the data from all trains. Several graphs are then displayed according to the option selected.

Is necessary to take in consideration that a blocks flow can be repeated several times and some blocks can be accessed without respecting the flow as presented. For example, the physiotherapist can modify the train at any time after creating it and can also access the patient results any time.

### **3.3 Smart Sensors**

The smart sensors are a critical part of this project as they will act as controllers for the serious game detecting motion of the lower limb and gait as well as heart activity and muscular activity. Smart sensors have great importance in the developed system. In this chapter, the information related to these is presented. Randy Frank describes what an SS (Smart Sensor or Intelligent Sensor) is by contrasting a non-intelligent sensor. In this contrast, we can read that an intelligent sensor is a sensor that has the ability to do something more such as conditioning of the signals or facilitate the connection to an MCU (Microcontroller Unit or Microcontroller) and it was this facility that became the main point of defining what a Smart Sensor is. In 1993 several IEEE 1451 standards were released which defined various aspects of what a Smart Sensor is [28].

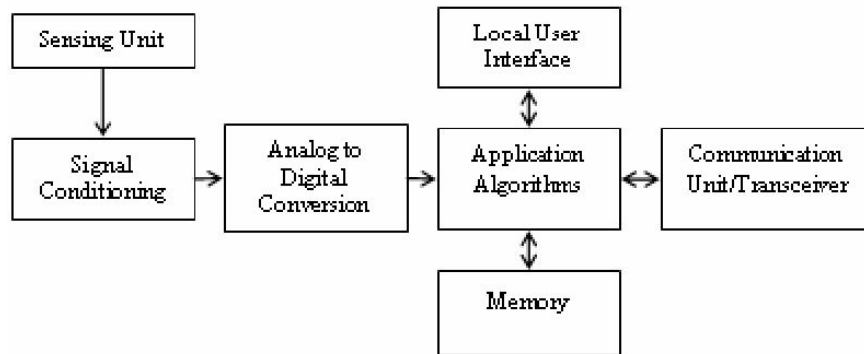


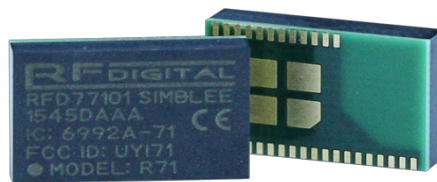
Figure 3.4 - Smart Sensor Architecture

Figure 3.4 illustrates a block diagram representing the various components of a Smart Sensor:

1. **Sensing Unit** - Is the unit responsible for collecting the intended information.
2. **Signal Conditioning** - Sometimes the sensors send data that needs to be packaged and this is the module responsible for this. As stated earlier this is one of the factors that differentiate smart from the ordinary one that does not have this ability to handle the data.
3. **Analog to Digital Conversion or analog to digital signal conversion** - This is the component responsible for passing analog data to digital so that it can be processed.
4. **Application Algorithms** - This component has the ability to compute the data after it has been transformed into digital. It is possible to select data as well as its treatment.
5. **Local User Interface** - This component is the component that allows the user to perform actions such as sensor calibration or simple data query. It is normally present in another device that uses the sensor communication component to access this
6. **Communication** - Component responsible for the transmission of data after processing by previous layers. This component should perform communication with an MCU in a simple, easy and universal way through serial communication, I2C or using the SPI protocol to be considered an intelligent sensor.
7. **Memory or Memory** - Some sensors have the capacity of memory so that in case of loss of communication there is no loss of data or even to save configuration files.

### 3.3.1 Simblee Microcontroller

As referred before a smart sensor needs an MCU and the one used in the prototype sensors for the presented work is a Simblee microcontroller (Figure 3.5). Its characteristics are favorable to wearable projects. Its small 7mm x 10mm x 2.2mm [29] enough to make it adaptable to any wearable project.



*Figure 3.5 - Simblee microcontroller 7mm x 10mm  
x 2.2mm*

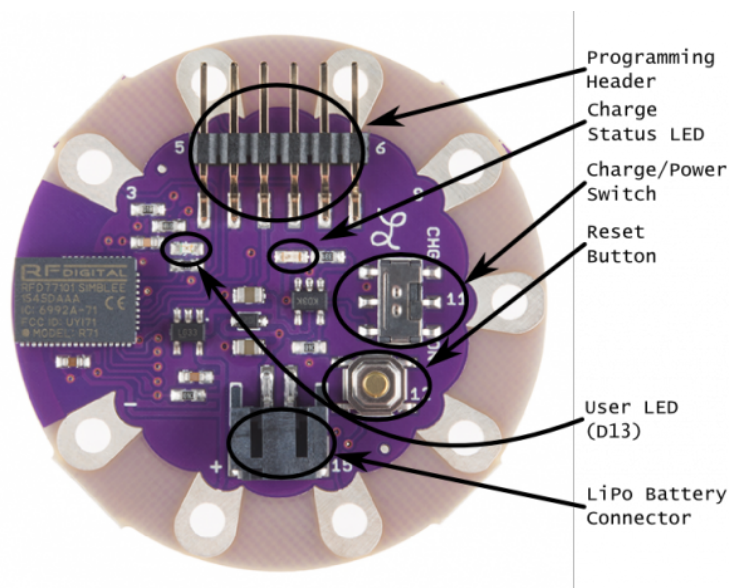
The Simblee has 29 GPIOs and SPI and I2C interfaces which makes it compatible with most sensors on the market. These characteristics make it the smallest Arduino compatible board on market to this date, which is compatible with Arduino IDE and has frameworks developed by its creators that allows the development of mobile apps for Android or IOS just with Arduino code.

These microcontrollers feature also low power consumption of just <4uA ULP with clock running (run for years on a coin cell) and 600nA ULP Sleep mode.

The ARM® Cortex M0 processor is the smallest of the ARM family with just 56 instructions but its powerful for this type of application[30].

The sparkfun company known for developing electronics for hobbyists and enthusiasts developed several ready to use boards with the Simple pre-soldered. The board chosen for this project was Lylipad Simblee board (Figure 3.6) with conductive thread ready sewable holes so it is very simple to add to a cloth.





*Figure 3.6 - Lilypad Simblee board*

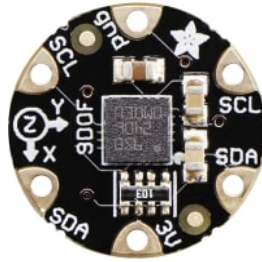
In Figure 3.6 is possible to observe some features that the Lilypad board has, for example a programming header. Although Simblee SoC has OTA (Over The Air) programming feature is very useful to have a programming header in hand for cases when OTA does not work or debug is necessary. Other features presented in Lilypad Simblee board are a battery jst connector that makes it easy to hookup a battery to it, user-led, charging circuit built-in the board.

This features combined, made a very complete board for the smart sensors that were prototyped in the present work.

### **3.3.2 IMU (Inertial Measurement Unit)**

In physical rehabilitation tracking body movements has great importance. In this work was used IMU modules that enabled this system to calculate lower limb important angles like the knee joint angles as well as acceleration of movements which can be a good indicator of patient progress.

In the present work was combined several sensing units to create a smart sensor network, in each one, was included an IMU board named Flora 9-DoF [31] (Figure 3.7).



*Figure 3.7 - Flora 9DoF  
LSM9DS0*

The Flora 9 Dof packs a small size with conductive thread holes for sewing purposes has 9 degrees of freedom making it capable of accurately determine body movements using and an array of these boards.

The reasons why this board was used in this project are presented in the next bullet points:

- **Size:** It a small board making it easy to adapt to this project;
- **Power consumption:** Low power consumption;
- **Interface:** I2C interface making it compatible with Simblee microcontroller;
- **E-Textile:** This board can be easily added to the wearable project.

There were other boards taken in consideration but in last the flora 9Dof was adequate for the present work.

### **3.3.3 ECG (Electrocardiography)**

Cardiac activity is one of the most important vital signs our bodies have. It is a sign that is used to detect living body and shows when a person is tired and when the human body is reaching its limits. The most common metric for heart rate is the Beats Per Minute (BPM) although this is not the only one that has importance for body functions evaluation BPM can be achieved easily by using a simple PPG sensor that uses light to detect a heartbeat. A more complex analysis of the heart is done by using an electrocardiogram (ECG) and this can reveal far more cardiac problems. In the past, this system was very big and impossible to carry but advances in this areas allowed a change. This change enabled new possibilities in health monitoring allowing that patients take a device with them and the doctor could analyze a more complete

set of data that could reveal more detailed on heart problems, these devices medical term is holter.

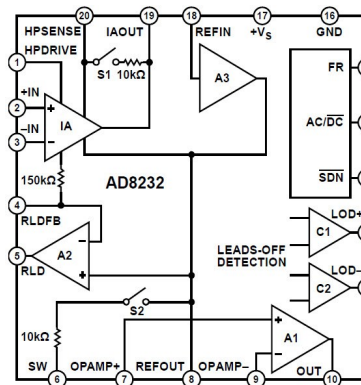


Figure 3.8 - AD8232 Functional block diagram

The Texas Instruments AD8232 [32] (Figure 3.8) it is a small chip used in the present work for electrocardiography. It is not the best in the market but it is simple and easy to hookup to simblee.

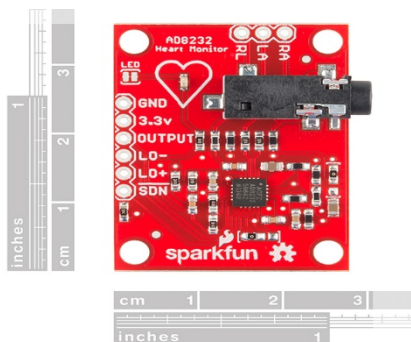


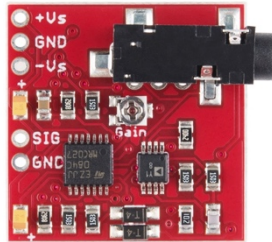
Figure 3.9 - Sparkfun AD8232

The data obtained that reveals patient fatigue and effort done during exercises executed represents good metrics for the physiotherapist to analyze.

Physiological and physical metrics obtained when patients exercising during physiotherapy sessions may indicate if patients are tired or make a great effort.

### 3.3.4 EMG (Electromyography)

Sometimes is necessary to detect if a muscle functionality is improving. Movements metrics can give some information on muscle activity, but not precise information on what muscle and what intensity of muscle exertion. Electromyography (EMG) give information on muscle activation.



*Figure 3.10 - Muscle Sensor V3*

### 3.3.5 RTC (Real-Time Clock)

In the present project was used five smart sensors each one sending 10 samples a second. The smartphone handles the reception of the packets but in this situation was identified packet delays so we introduced a real-time clock (RTC)[33] in each sensor and each packet was sent with the current time. In this way was possible to correct packet delay as by combining the data of the five sensors to make calculations of angles and other important metrics that could become very affected with delays.



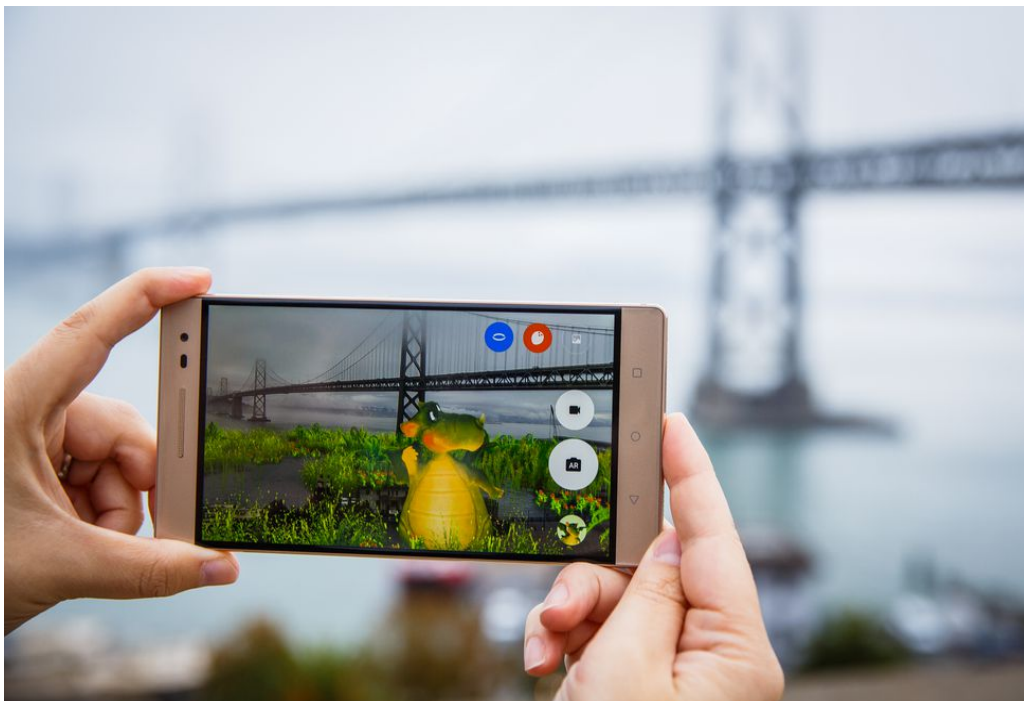
*Figure 3.11 - RTC Module*

## 4 Augmented Reality for Physiotherapy

As one of the main problems in physical rehabilitation is patient demotivation. To reduce these problems was introduced an interactive Augmented Reality environment. This environment includes a Serious Game. Serious Games are games with a purpose that is not only entertainment they can be educative or have another purpose rather than entertainment.

### 4.1 Apple ARKit

To develop Serious Games several game engines can be used, like Unity3D, Unreal Engine but both are complex. In the present work was opted for a simpler solution. Developed in IOS. Apple had created a framework called ARKit [34] for Augmented Reality apps that is simple to learn in a short time, easy to program and intuitive. Another option could be the Android ARCore Figure 4.1 but it was only released in March 2018 which coincided with ongoing developments of this project making it difficult to adapt to the Android language.



*Figure 4.1 - ARCore example*

Apple ARKit (Figure 4.2) is the framework responsible for the game engine. All the code for the game was done using it.



*Figure 4.2 - Apple ARKit example*

## 4.2 Adobe Fuse and Adobe Mixamo

In the course of the developments, was dealt with the complexity of the required exercises and would be difficult to explain them in the absence of a physiotherapist. Since giving up the possibility of remote training without the assistance of a health professional was not an option, the virtual assistant was created and is responsible for demonstrating the requested exercising tasks by giving visual feedback to the patient. It was also thought that the assistant should be suitable for each age group and gender so that the patients could feel more comfortable during the training session. Several types of assistants were created that are automatically assigned according to the age of the patient.

For the creation of the different virtual assistants, it was necessary to modulate them using CAD. The complexity of this task was understood and was chosen as a simple method that reduces development time. Mixamo and the Fuse Software produced by Adobe are easy to interface programs that allow to animate 3D and to create modular characters were chosen.

Mixamo allows exporting Collada files with dae extension, which is compatible with Apple SceneKit framework for creating 3D environments for the developed applications and for ARKit consecutively.

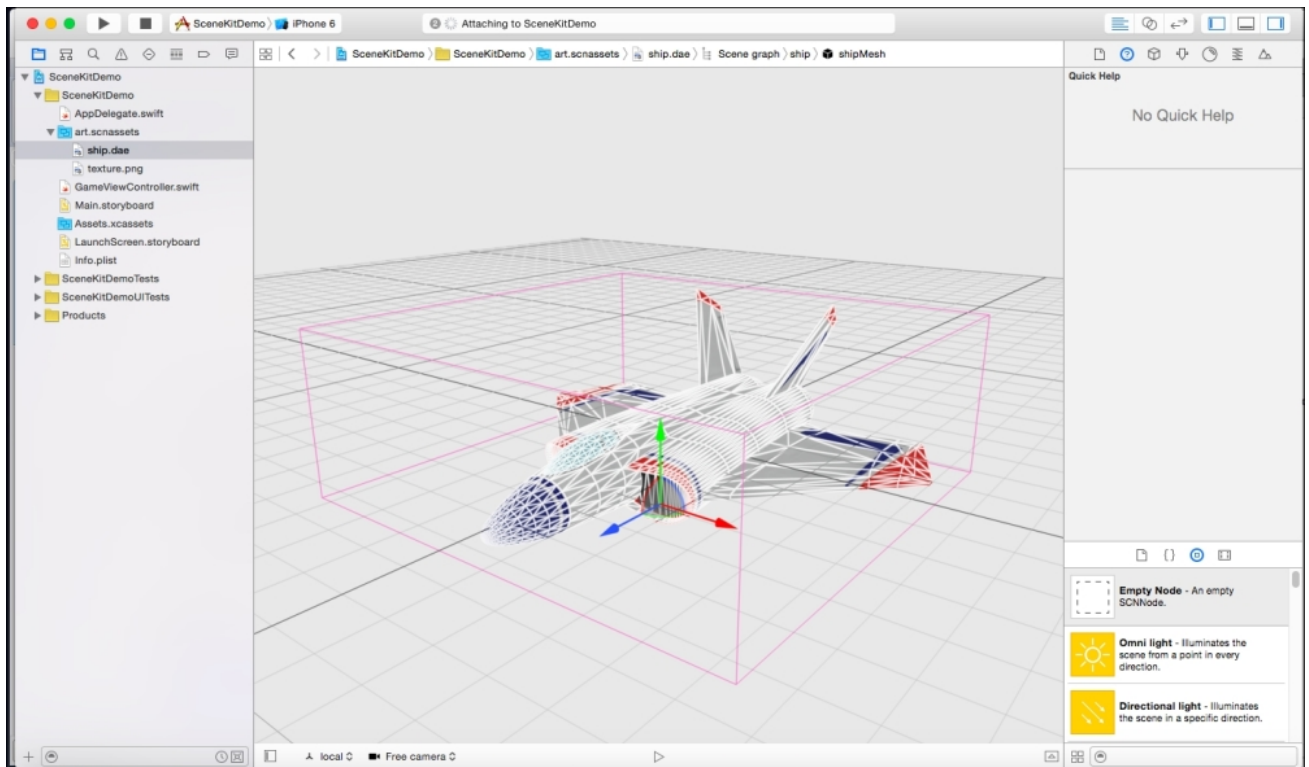


Figure 4.3 - Mixamo character animation

With Mixamo software it is very intuitive to rig and animate a 3D character. It is as simple as choose the animations from an extensive palette and assign them to the character and finally download the originated files, this process is all automatic and does not require any code or 3D modeling and animating knowledge.

## 4.3 Apple SceneKit

Apple has a framework for 3D objects creation and interaction which is called SceneKit. SceneKit was used in the present work to place the objects in a 3D environment and to make them move given them physics properties.



*Figure 4.4 - Xcode Scenekit development window*

“SceneKit combines a high-performance rendering engine with a descriptive API for import, manipulation, and rendering of 3D assets. Unlike lower-level APIs such as Metal and OpenGL that require you to implement in precise detail the rendering algorithms that display a scene, SceneKit requires only descriptions of your scene contents and the actions or animations you want it to perform.” [35].



## 4.4 Game and sensors integration test

After gathering all the tools necessary was tested the integration of the sensors with a virtual scenario to study real-time response in game. For doing this was created a simple scenario was a soccer goal is placed in front of the user and a ball. The user kicks the ball and the sensor translates the forces executed to the game which applies them to the virtual ball.

The Scenekit editor (Figure 4.3) lets us add physics to the scenarios although this should be done through coding respecting the good programming practices as these will be more susceptible to future bug corrections, there is an interface that allows drag and drop feature that makes it easier to add physics.

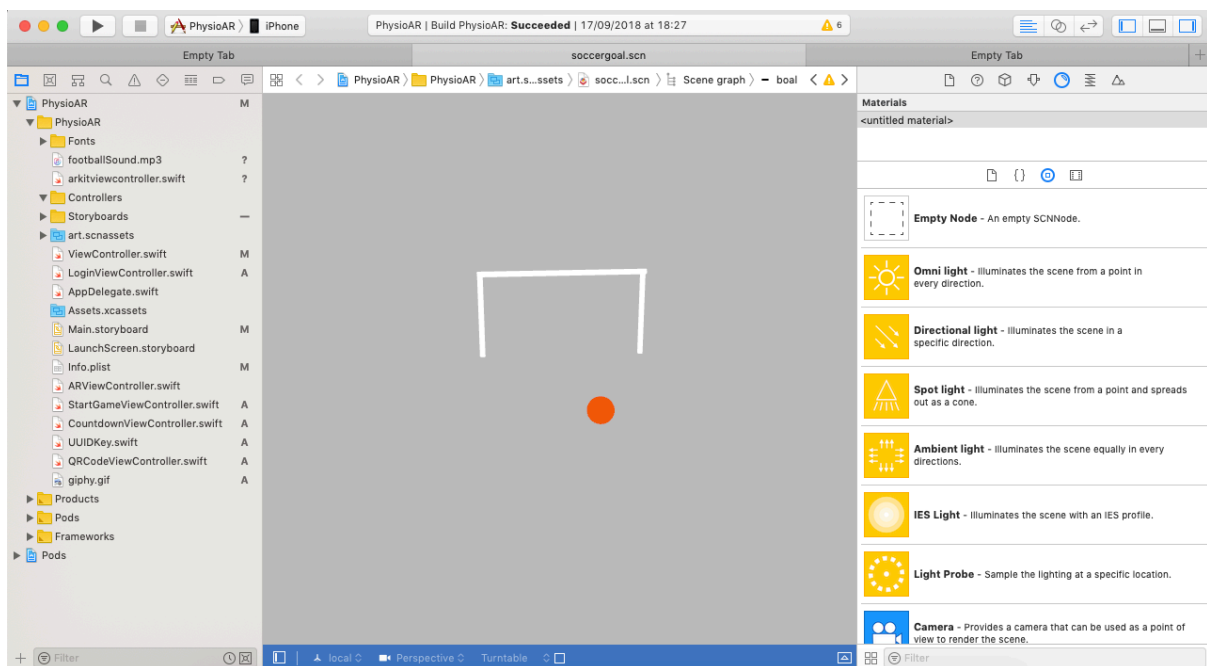


Figure 4.5 - Xcode SceneKit Editor

After completing the 3D soccer scenario and integrating a prototype sensor capable of measuring EMG and leg acceleration, was started the tests. In the first tests was calibrated the sensor and observed if its outputs were correct. The accelerometer that detects the velocity and the EMG was used to do the impulse indicating when the ball should start the movement. This test went well and demonstrated that the system was able to work in real time since the response of the sensors arrived in the expected time.



*Figure 4.6 - First test football scenario*

In Figure 4.6 the game screen created for the first test can be observed. As it was a preliminary test there was not much care with the aesthetic of the same but more with the functional part to realize how it would work when integrated with the sensors and the mobile application.

Concluding these tests demonstrated the functionalities of both mobile application, Apple Frameworks, and our smart sensor prototype. The combination of all elements of the system worked well and proved that the developed system worked with the technologies used which allowed to start the entire development process with some certainty that in the future everything would work correctly.

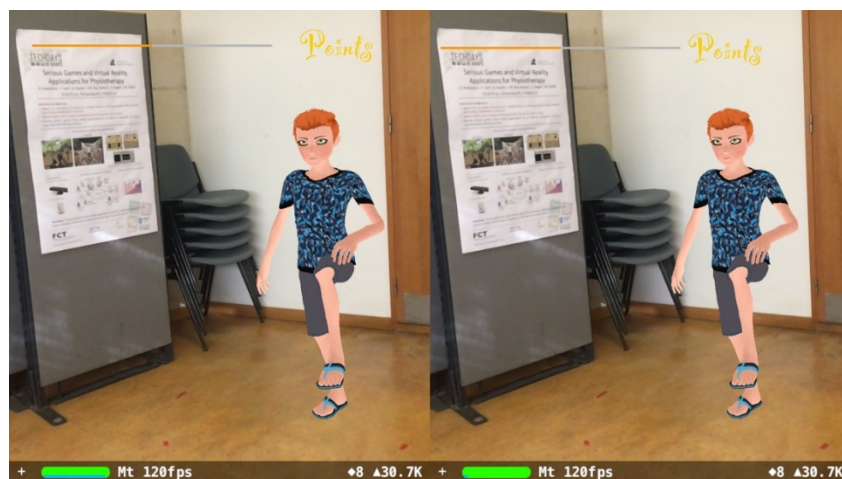
## 4.5 Stereoscopic view

Throughout the course of the research work was identified as a limitation of Apple frameworks for Augmented Reality, the fact that there is no possibility of easily placing the essential Stereoscopic view (Figure 4.7). In this project in order to free the patient as much as possible and not to stay with smartphone in hand during training (which is more common in the use of AR in mobile applications the user holds the smartphone by hand and sees what is around by looking at the screen), was included a virtual reality universal goggles with compatibility for smartphones up to 5.5 inches and has been developed stereoscopic view using the apple framework ARKit framework.



*Figure 4.7 - Stereoscopic view*

The result of the stereoscopic view developments can be seen in Figure 1.1 where it is already possible to see a screen of the mobile application after the completed developments. All the code to make this change of perspective was made using the language Swift 4 and was one of the biggest challenges of the present work because it was necessary to realize well how the human eye worked to realize the correct perspective of three-dimensional longitude.



*Figure 4.8 - PhysioAR Serious Game footage*

## 5 Smart Sensors Prototypes

This chapter will discuss the development of smart sensors, the tools used and the schematic of the circuits. The focus is put on the details of how communications are carried out and the reduction of energy consumption in order to prolongate the life of the batteries.

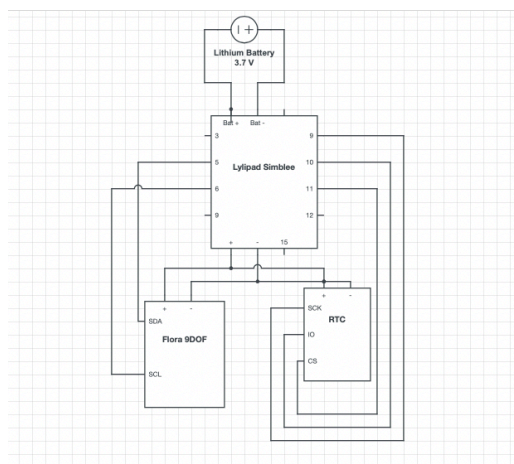
This system consists of three different types of sensors. A group of 3 sensors was used to detect the gait and some angles that could help the physiotherapist to better understand the evolution of his patient. In other development was used two sensing units in which one detect the movement (IMU) and an EMG unit that was used to detect the muscular activity. A prototype of a system that combined a motion sensor (IMU) and ECG was also realized.

### 5.1 Motion smart sensor

In this subchapter is described the development of the prototype of this smart sensor. It will also present the schematic of the circuit and also some of the code that runs on the microcontroller.

#### 5.1.1 Schematic

Figure 1.1 shows the schematic of the circuit that makes up the smart sensor. In the figure can be observed the three main components that are the Simblee microcontroller, the real-time clock module, and the IMU. The system is powered by a 3.7V lithium battery. This power supply is possible due to the existing voltage regulator on the microcontroller board that converts the 3.7V to 3.3V standard voltage.

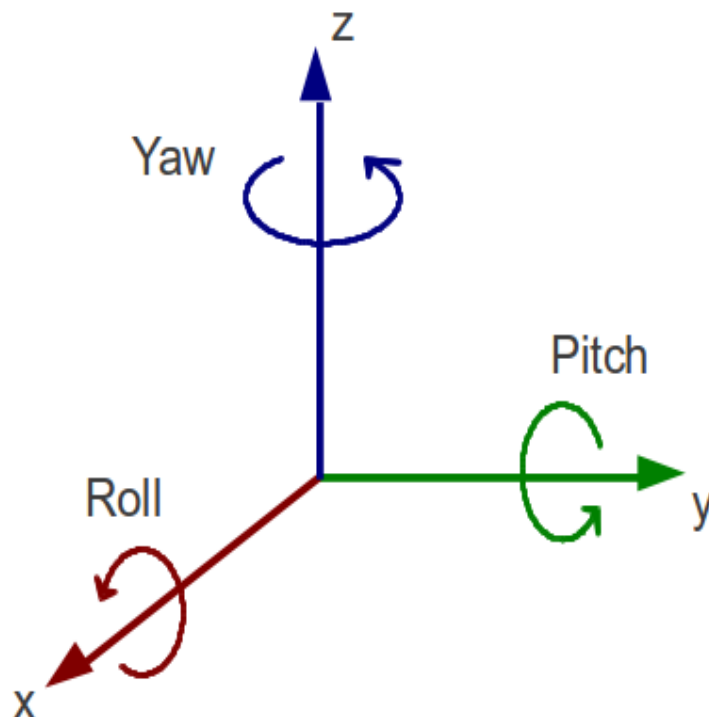


*Figure 5.1 - Smart Motion Sensor Schematic*

The IMU can return a lot of information related to the 9 axes, however, we select the relevant ones to fit in the small packet of 20 characters that is sent each time, this is a limitation of the communication protocol Low Energy Bluetooth that only allows 20 characters per packet.

### 5.1.2 Acquisition Data

The information selected was the roll angle, which corresponds to a rotation of the x-axis (Figure 5.2). In order for the system to function completely, it is necessary for the user to place the sensors correctly, a change in the orientation of the sensors completely changes the values by returning unexpected roll values, which damages all the final results as they combine data from different sensors the error of only one would influence all the results.



*Figure 5.2 - Roll, Pitch and Yaw Graph*



*Figure 5.3 - Acceleration vector of Shin movement*

Another metric considered important was an acceleration vector in the shin (1) that indicates the acceleration of the movements. To obtain this, an equation is needed which is done directly in the Simblee microcontroller. The equation is presented:

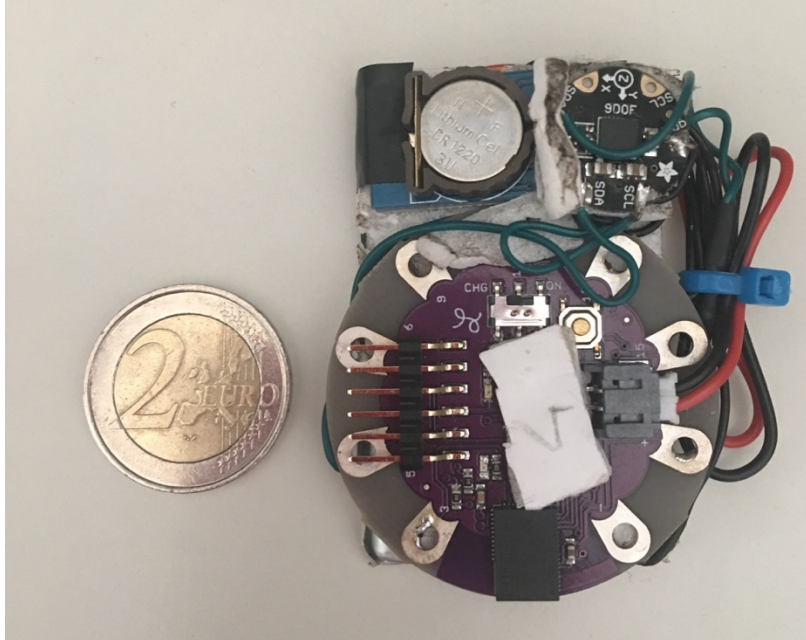
$$\text{Acceleration Vector} = \sqrt{\text{accelarationy}^2 + \text{accelarationz}^2} \quad (1)$$

The two-metrics referred above have enough information to give feedback on the movement of the lower limbs to the physiotherapist. They allow an accurate analysis of the movements of the lower limbs, increase precision in patient monitoring since it is possible to record all the information collected during the training, contrary to the traditional methods where there is no information storage of the train session.

In the system, the data are combined efficiently in interactive charts to allow a better analysis of the patient by the physiotherapist. It will be seen in more detail in Chapter 6.

As already mentioned, some problems were detected in the packet transmission, not with losses but delays. To solve this problem was introduced clock modules and the time is sent in the data packet, to avoid desynchronized packets. The integration of the module was quite simple since the module had a library for Arduino that simplified the process.

### 5.1.3 Prototype

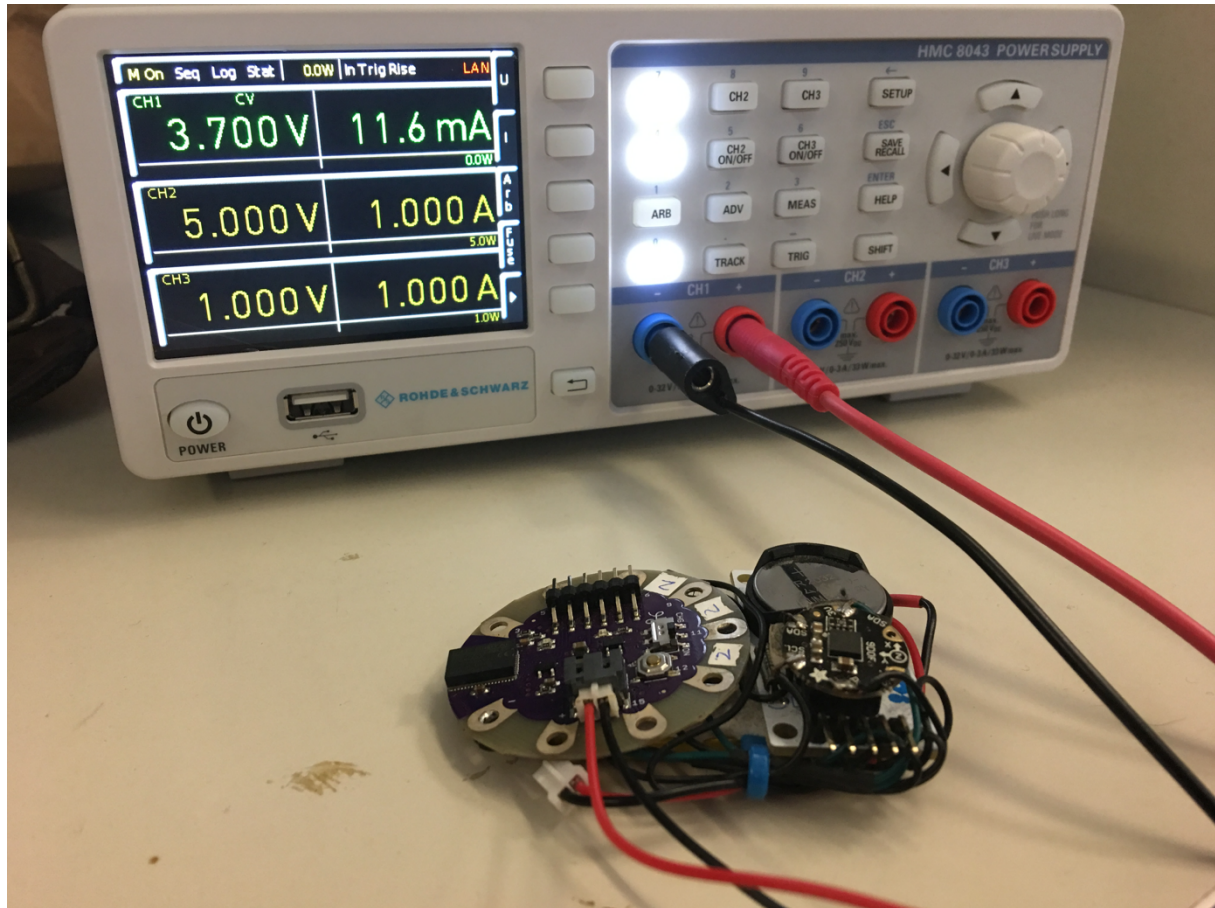


*Figure 5.4 - Smart Motion Sensor Prototype*

Finally, the prototype of the smart sensor can be seen in Figure 5.4. A coin of two euros was placed in the photo to show to serve as a comparison of dimension with the sensor. In the figure of the prototype, it is not possible to see the battery because it is glued underneath the microcontroller and the sensor.

### 5.1.4 Power Consumption

Power consumption was measured using a ROHDE&SCHWARZ HMC8043 Power supply with amperemeter.



*Figure 5.5 – Smart motion sensor power consumption*

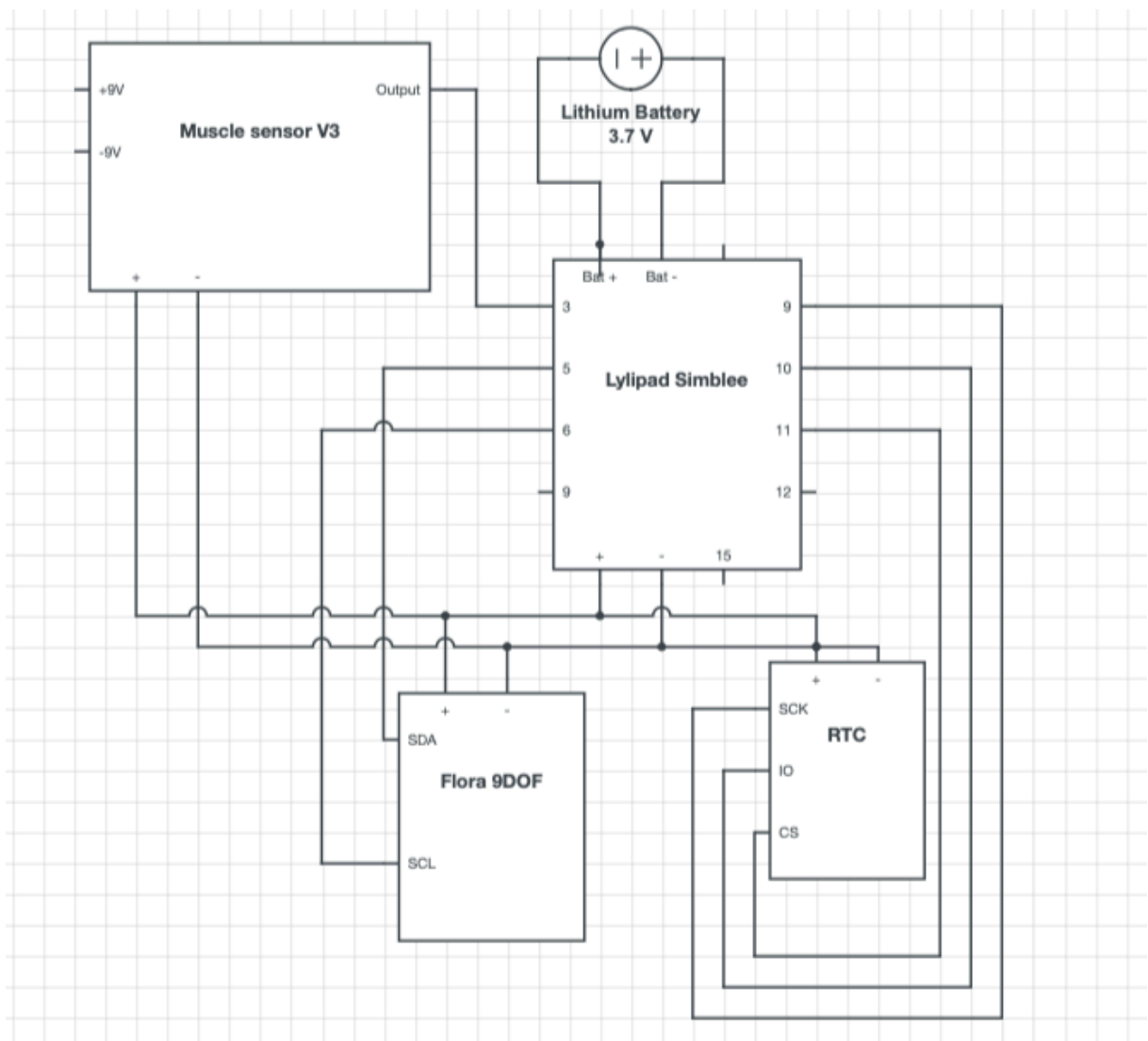
Figure 5.5 represents the moment when the power consumption was measured. This smart sensor consumes an average of 11.6 mA. This value means that the sensor is able to work for almost 48 hours and 27 minutes with 800mAh battery.



## 5.2 Motion + EMG Smart Sensor

In this subchapter is described as the process of developing the prototype of this smart sensor. It is also presented a schematic of the circuit.

### 5.2.1 Schematic



*Figure 5.6 - Motion+EMG Schematic*

Figure 5.6 describes the sensor schematic with the differences between this one and the motion sensor mentioned earlier is an EMG board presented. So, in this sub-chapter do not cover the motion part again since it is the same as chapter 5.1. This EMG board is the Muscle Sensor V3. It has an adjustable gain and requires batteries to amplify the output signal since this EMG

signal is very small. This board requires 9V batteries to make a +9 -9 power supply, and the gain is adjustable by a small screw.

**Note:** When setting the gain, the user should verify the output voltage as this can be higher than what Arduino supports.

### **5.2.2 Acquisition Data**

The data acquisition described here only cover the EMG as the information on motion acquisition is the same as before. The EMG signal is hard to adjust in the screw presented in the board which could lead to bad final output. These boards outputs a simple voltage signal that increases with muscular activity captured by the electrodes.

### **5.2.3 E-Textile shin pad electrodes**

The E-textile are urging in the market has the result of research in the area, although in current development state[36]. E-textile allow new developments of electronic e-textile sensors in wearable designs.

Common medical electrodes are disposable and have to be glued to the skin. Dry electrodes are becoming less common in clinics because of contamination risk as these are reusable, but for this project was developed an e-textile dry electrode shin pad as the risk of contamination is not a problem for this system. This shin pad was produced using an elastic fabric and a silver conductive fabric and common cloth buttons. The final prototype was capable of EMG acquisition.



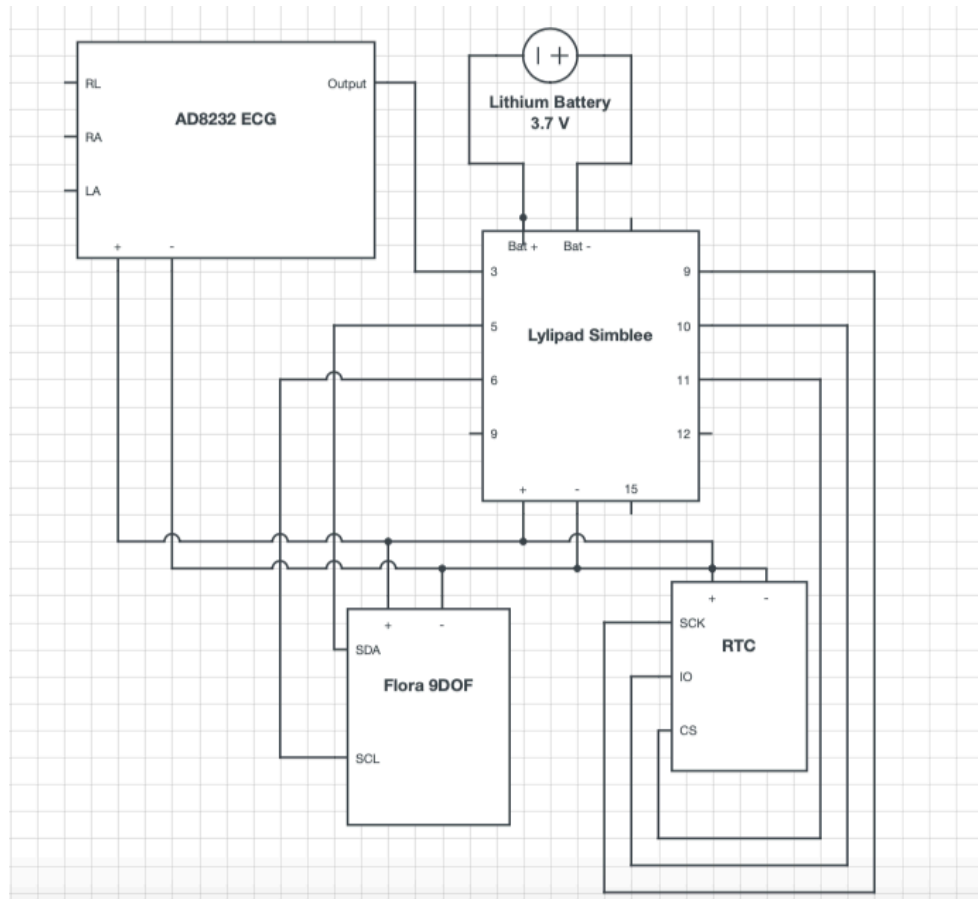
*Figure 5.7 - EMG E-Textile electrodes*

Figure 5.7 demonstrates the inside and outside part of the shin guard and the attached cable for connection to the acquisition board. The elastic fabric permits one size.

## 5.3 Motion + ECG Smart Sensor

In this subchapter is described and is presented the scheme of the prototype of the smart sensor.

### 5.3.1 Schematic



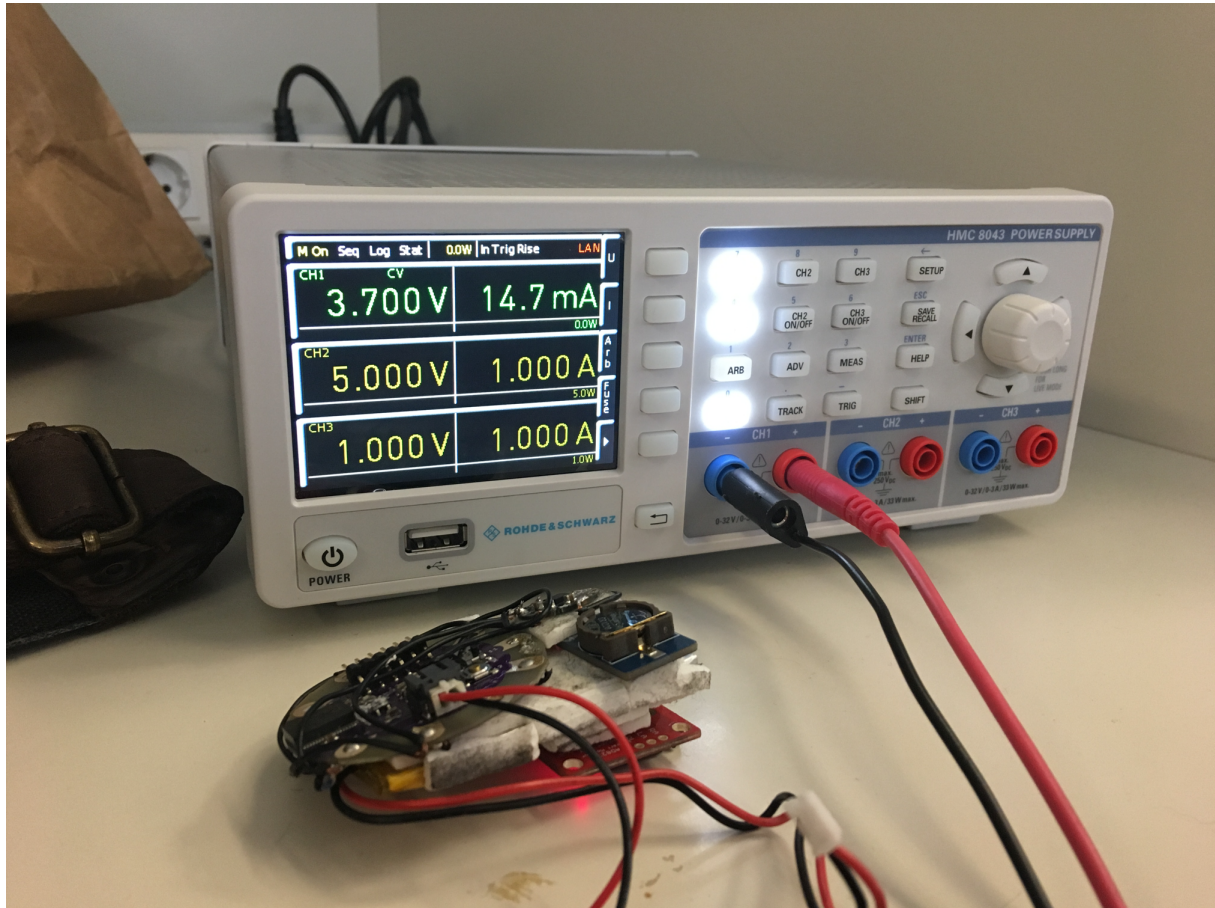
*Figure 5.8 - Motion+ECG Smart Sensor*

Figure 5.8 is presented the ECG and Motion smart sensor schematic. This schematic includes the AD8232 board which can acquire an ECG signal from three electrodes and output an ECG signal as a voltage.

The prototyped board worked well when the user was stable but not when in movement. The output signal also varies sometimes which turn it difficult to process using an algorithm.

### 5.3.2 Power Consumption

Power consumption was measured using a ROHDE&SCHWARZ HMC8043 Power supply with amperemeter.



*Figure 5.9 - Smart ECG+Motion Sensor power consumption*

Figure 5.9 represents the measuring of power consumption, the value obtained for the ECG+Motion was 14.7 mA. This sensor is powered by a 3.7V 1300mAh battery and its autonomy is 61 hours on continuous use.

### 5.3.3 E-Textile ECG electrode t-shirt

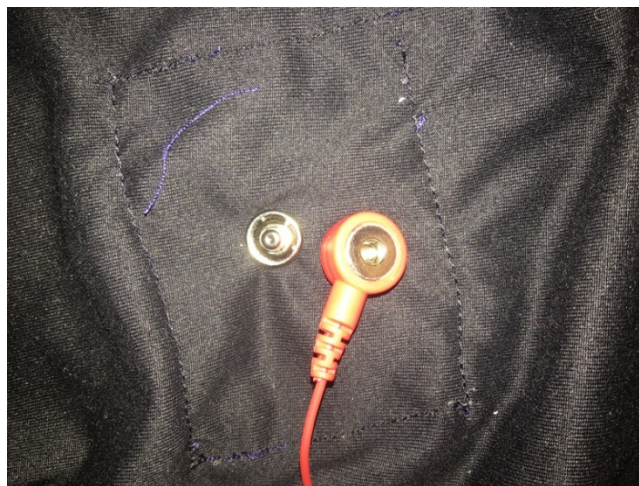
For acquisition of ECG signal was developed a t-shirt with dry electrodes sewed. This t-shirt is easy to wear and does not need to be glue or applied gel for conductivity like common methods.

Figure 5.10 demonstrates the inside and outside of the t-shirt. This t-shirt is a compression t-shirt designated for sports practice.



*Figure 5.10 - E-textile electrodes t-shirt*

A button was applied to the conductive fabric to plug the probe to the electrode as demonstrated in Figure 5.11.



*Figure 5.11 - Electrode cable plug*

## 6 System Backend

This chapter presents the backend application of the system aimed for the physiotherapist use and to what is done on the server side. In this chapter, the information on the server operations is merged with those from the application since both are hosted on an AWS instance. A great part of the present work was dedicated to server and backend application development. It is the backend of the system including the scripts responsible for the interactions with the database and the database itself, as the mobile application depends on this server to see and represent the patient information stored in it.

### 6.1 Backend Application

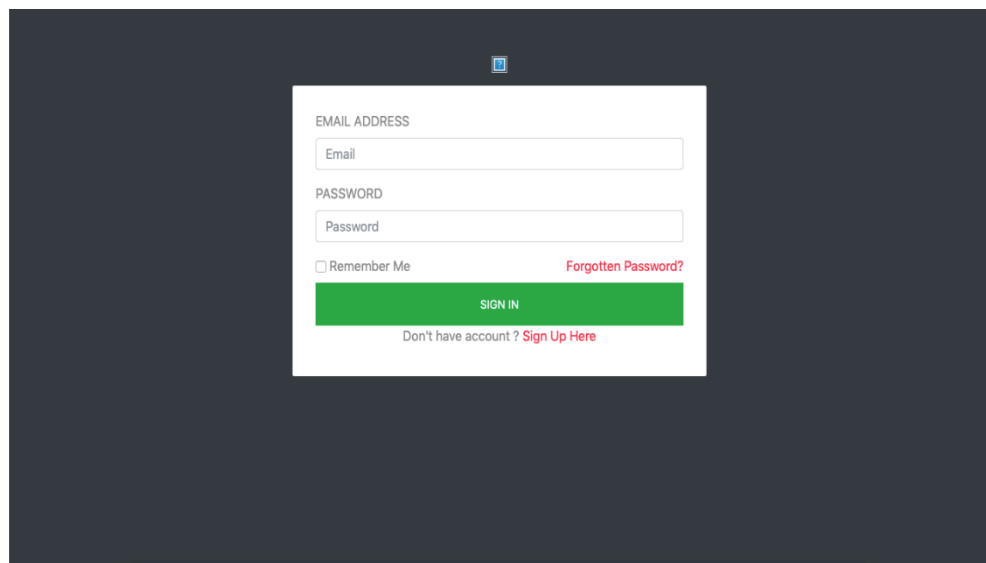
The backend application is unique for the user as a physiotherapist. In it, a certified physiotherapist can register and manage his or her patient and assigns them a training plan and analyze the training of their patients. This application was developed using HTML5 + CSS + JS to ensure greater device compatibility, a responsive design using libraries like the bootstrap was performed. By using these technologies, the application automatically changes depending on the screen size of the device, whether it is a smartphone, tablet, or computer adapting to screen size.

### 6.1.1 Physiotherapist Login and Registration

The first thing a physiotherapist should do when he starts using the physioAR system is to register in the system for that the physiotherapist should go to the address <https://www.physioar.tk> (note: the domain is maintained by paying the associated cost). Figure 6.1 represents the login page. On this page, the user can enter their credentials or go to the registration page or password recovery page.

It will now be explained what happens in these three situations:

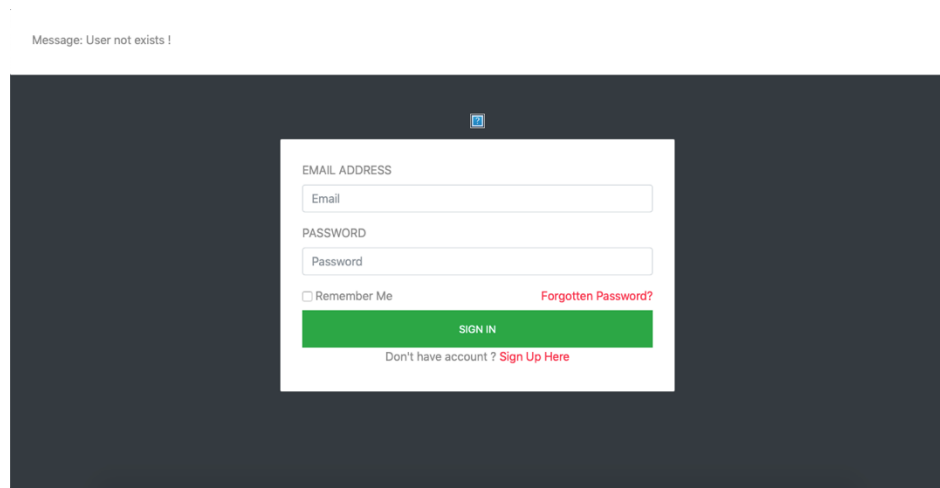
- **Login:** A registered user can log in using the email and password of registration by simply placing both data in the fields designated for this purpose. The application has an option to memorize the password and username for faster access in the future by clicking on the box remember me. After clicking on the Login button, the user can have three responses from the server:
  - Success Login: redirected to main page (see sub-chapter 6.1.3.);



*Figure 6.1 - Backend Login Page*

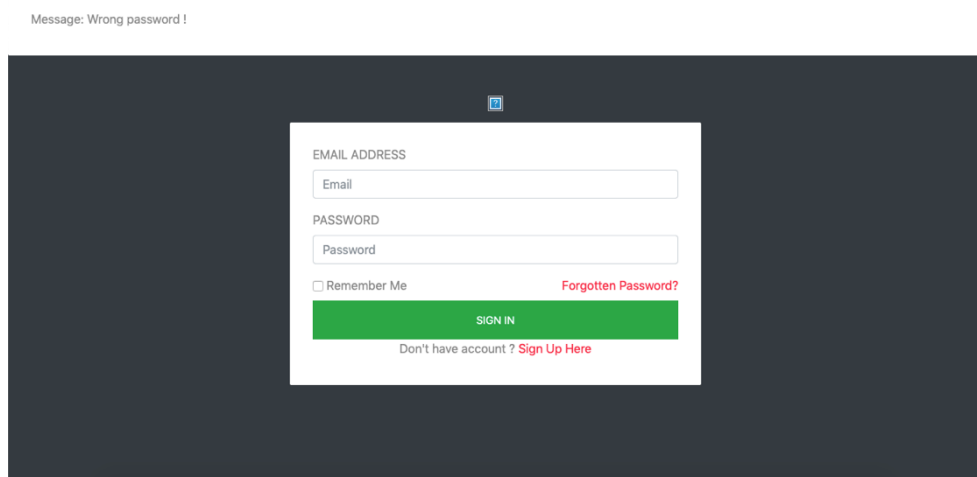
- Wrong Username: A message appears saying that the user does not exist. (Figure 6.2);





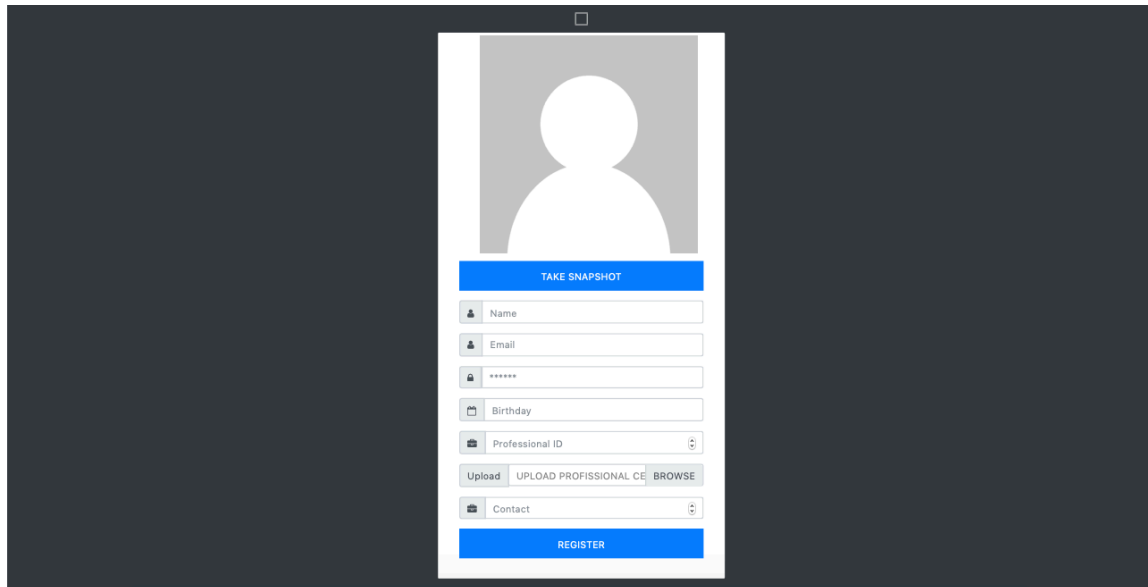
*Figure 6.2 - Backend user not exist message*

- Wrong password: A message is displayed showing that the password is incorrect (Figure 6.3);



*Figure 6.3 - Backend - Wrong Password Message*

- **Registration:** By selecting this option the user is redirected to another registration page (Figure 6.4).



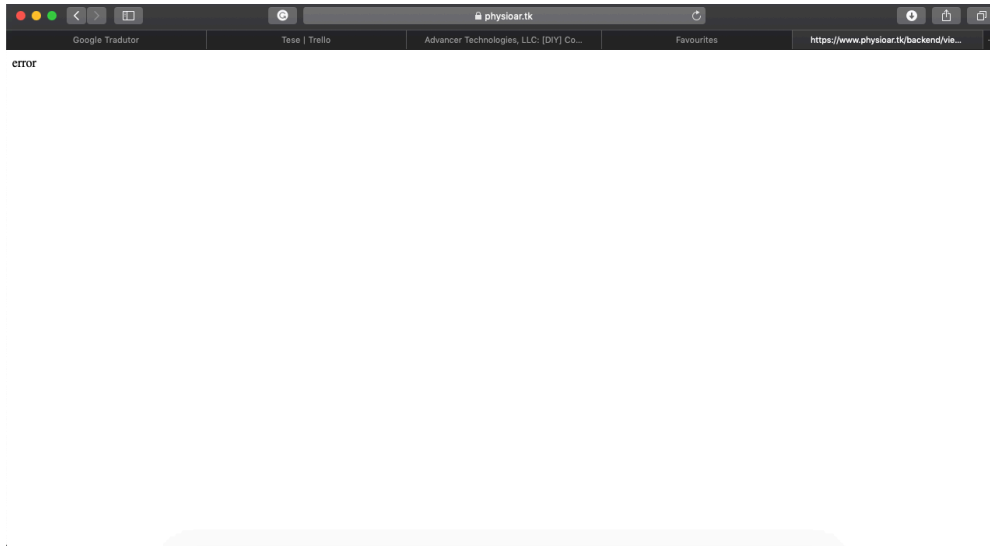
*Figure 6.4 - Physiotherapist registration page*

To register the user, it is necessary to fill in several fields: Photo, full name, email, password, birth date, professional id, file copy of the professional ID, contact.

The photo can be directly taken from the web browser which the user needs only a webcam or the smartphone camera or tablet depending on the device is using. The website <https://www.physioar.tk> of this system has an SSL valid certificate for accessing the user camera for security reasons (some browsers block apps without them).

After a successful registration, the user is redirected for the login page (Figure 6.1) where can login with the created credentials and receives a confirmation email.

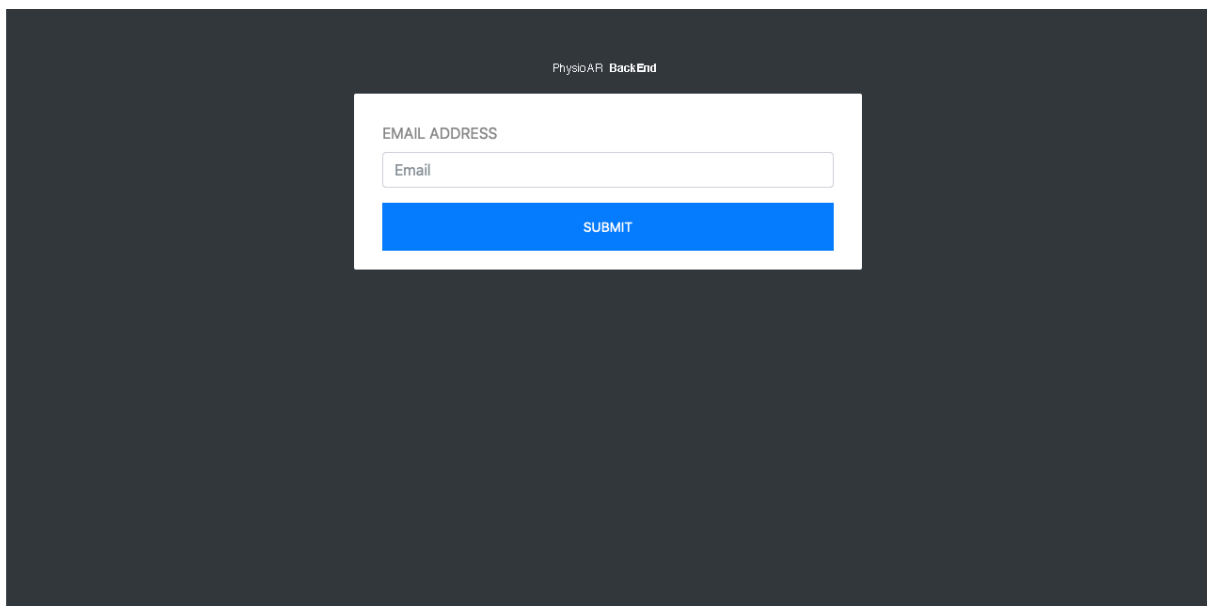
If it fails, the registration due to some error or a repeated email the user is redirected to a page that shows the error (Figure 6.5).



*Figure 6.5 - Physiotherapist registration failed*

- Password recovery:

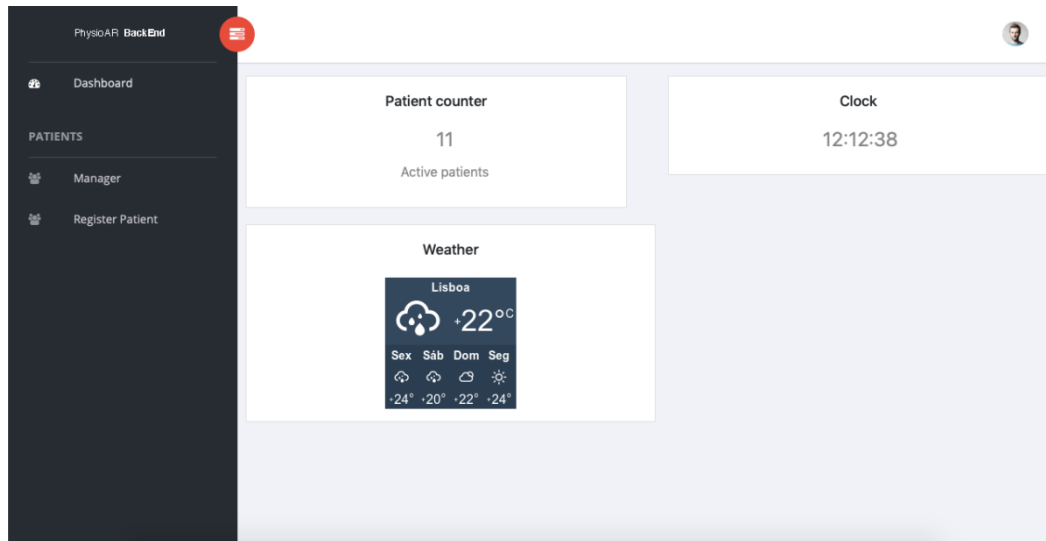
The option to recover password allows the user to recover access to his account when he forgets the password. This option sends an email with the new password. To reset the password the user should enter the email and submit the form (Figure 6.6).



*Figure 6.6 - Backend password recovery*

## 6.1.2 Physiotherapist Main Page

The main page of the physiotherapist is a simple page with just a few cards such as the clock, patient counter, and weather. The page can be seen in Figure 6.7.



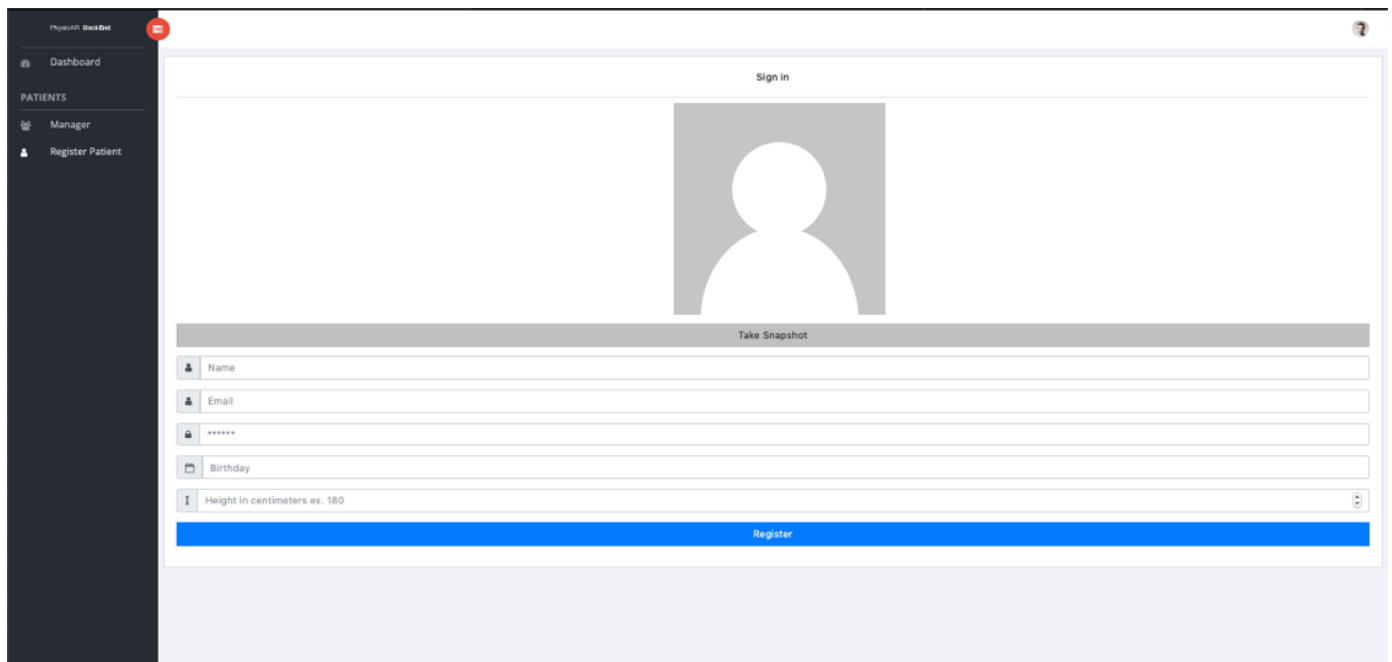
*Figure 6.7 - Backend Main page*

This page is accessed after successful login and allows the physiotherapist to access the patient management pages from the main page. For safety, an inactivity monitor logs out the physiotherapist if they do not touch the computer for a defined period of time.

As previously stated, this screen takes you to the Patient Management and Patient Registration pages that are covered in the next sub-chapter.

## 6.1.3 Patient manager and register

These are pages that allow the registration of users. This registration is done by registered physiotherapists in which the patient is automatically assigned to the physiotherapist who registers him.

The image shows a web application interface for patient registration. On the left is a dark sidebar with a 'PhysioAR BackEnd' header and a 'PATIENTS' section containing 'Manager' and 'Register Patient' options. The main content area has a 'Sign in' header and a large grey placeholder for a profile picture. Below this is a 'Take Snapshot' button. The registration form consists of five input fields: 'Name', 'Email', a password field with six asterisks, 'Birthday', and 'Height in centimeters ex. 180'. A blue 'Register' button is positioned at the bottom of the form.

*Figure 6.8 - Backend patient registration*

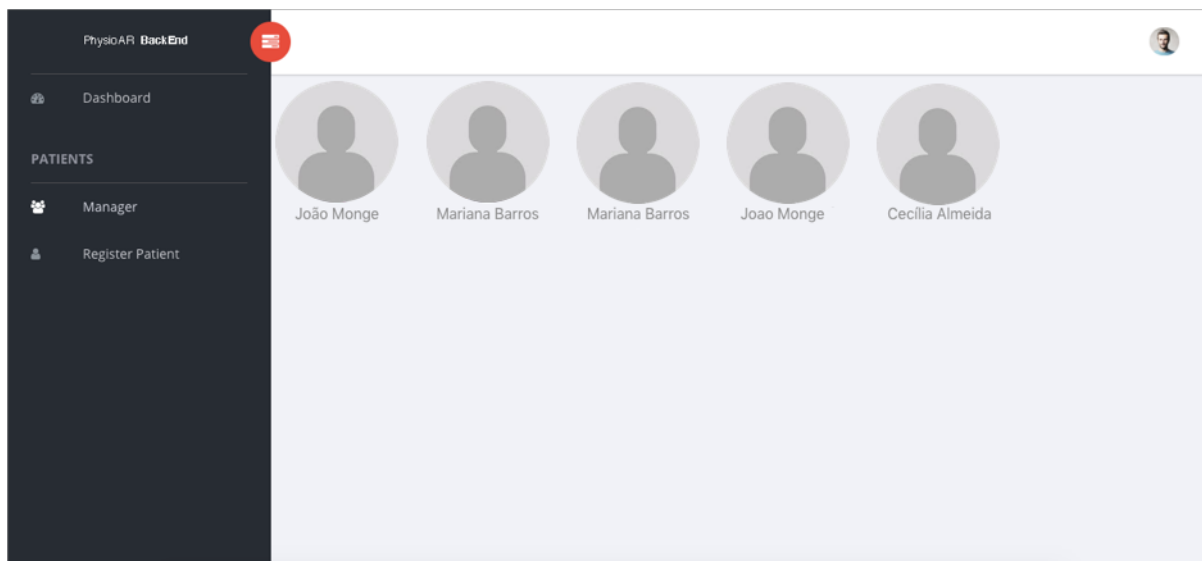
Figure 6.8 represents the screen responsible for creating a patient. On this screen, there is a form that contains the fields that are needed to fill to register the patient.

These fields are composed of photo, name, email, date of birth, patient height, and weight.

The height of the patient and the date of birth are important fields that must be placed correctly because they are two metrics used for rendering of the game, the height is for the 3D objects to be placed well and the date of birth is for adapting the virtual environment according to the age of the patient.

After a successful patient creation, the physiotherapist user is redirected to the patient manager page (Figure 6.9), where the user/physiotherapists can select the patient for which he wants to consult data.

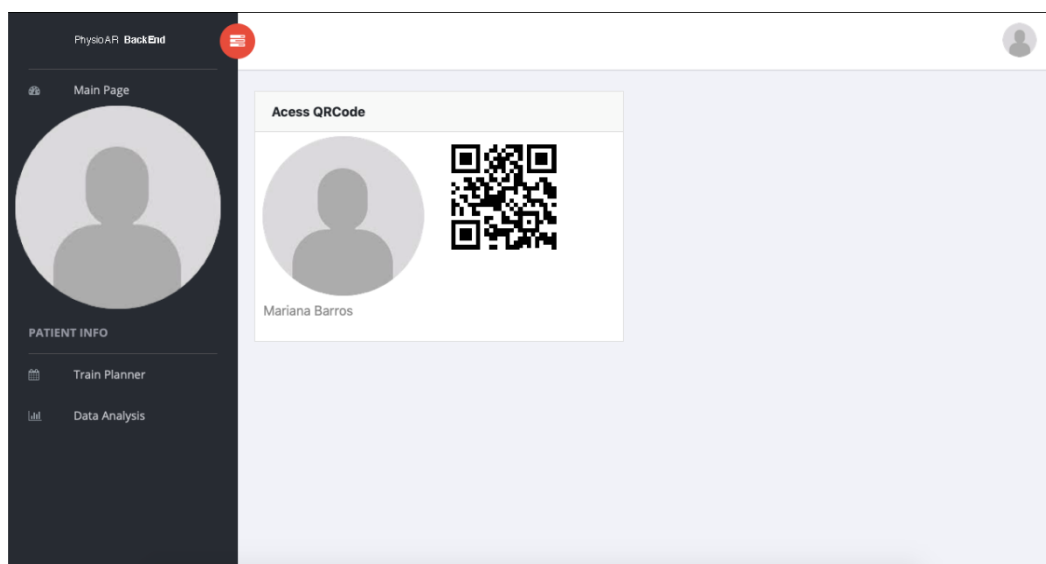
If the physiotherapist clicks on a patient on this page, he will be redirected to the patient personal page which is described in the next sub-chapter



*Figure 6.9 - Backend - Patient Manager*

### 6.1.4 Patient personal page

This page is where the physiotherapist can consult the entire training process of a particular patient. It is possible to assign a training plan and consult the results of patient training.



*Figure 6.10 - Patient personal page/QRCode access card*

The page in Figure 6.10 represents the patient main page where the access card with the QRCode is showed.

This screen is simple and allows access to more complex screens such as the screen that allows to create a training plan.

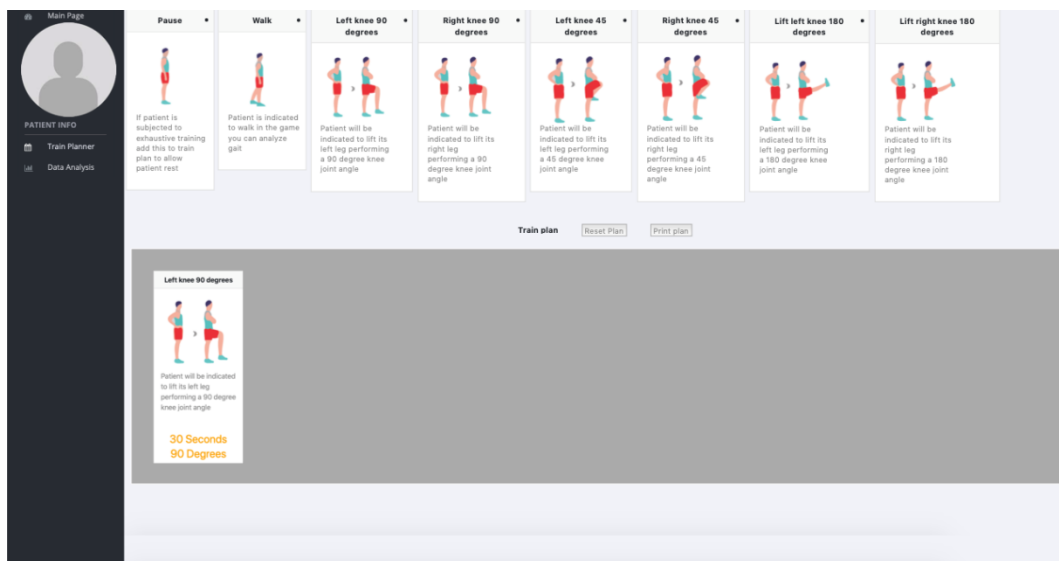


Figure 6.11 - Backend train plan

Figure 6.11 shows the train planner screen. It works as follows: the physiotherapist has a range of varied exercises to choose from. When selecting the desired one he should drag it to the bottom rectangle. For setting the exercise to be realized a pop-up window appears asking the thresholds of the exercise and duration. These parameters are used in the game for the user to score points when he reaches the threshold that the physiotherapist determined. Using this the system can motivate the patient with low and high difficulty as the threshold can be adapted to each patient by the physiotherapist.

After the patient has an assigned train plan he can start playing and in turn get results. These results can be consulted by clicking on Data analysis link on the left menu.

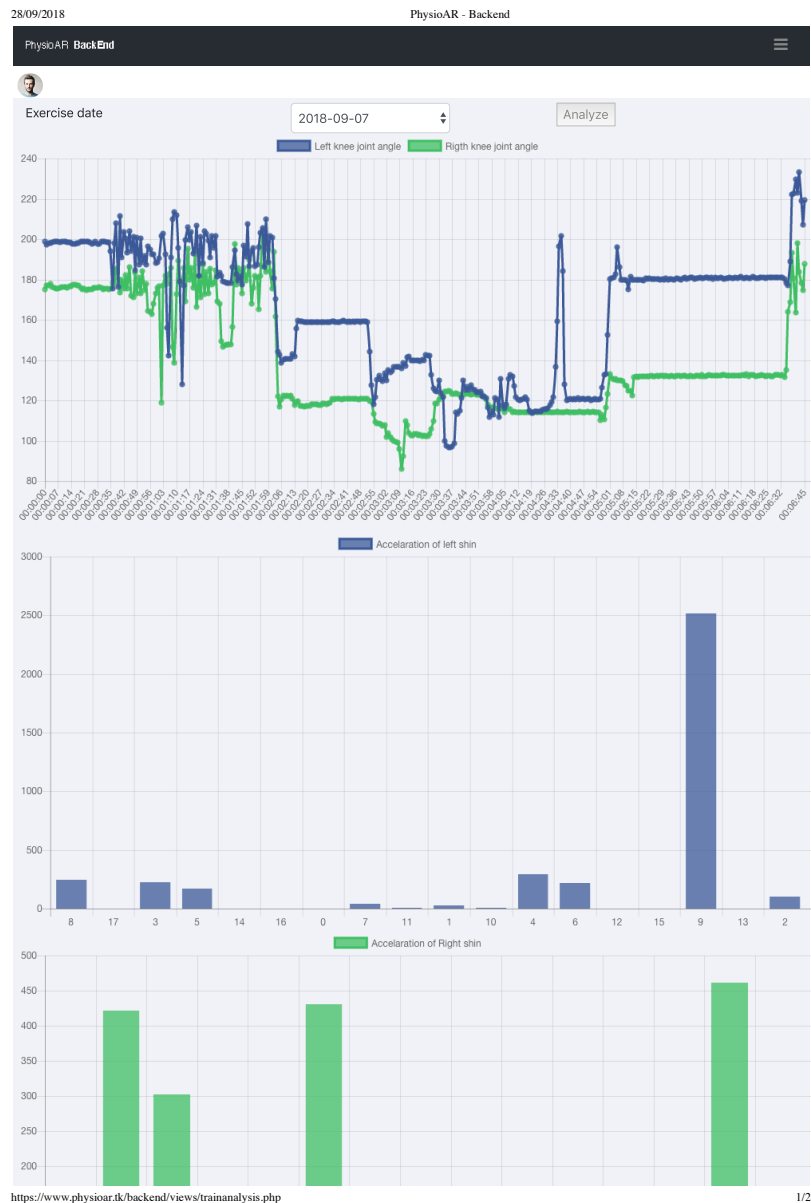


Figure 6.12 - Backend train analysis

In this screen (Figure 6.12) the physiotherapist can select the training by date or the overall option that combines all the results to receive feedback of overall progress.

It has several different graphs all showing different metrics of the patient gait and health data that allows a further analysis by the physiotherapist professional.



## 6.2 Cloud Server

As stated earlier this system was done thinking in account maximum device compatibility a server in the cloud was chosen to assure interconnectivity. This server allows greater flexibility and elasticity because it allows an increase of memory according to the need of the system reducing the costs when are not still enough users to support the price of a common server with greater capacity of storage. This sub-chapter give information on the server of this system.



*Figure 6.13 - Amazon Web Services*

In the present project, Amazon Web Services (Figure 6.13) was chosen as the cloud service provider. Amazon has a program for students given free access to some contents and features. AWS is choice was hampered by its free student plan and reputation in the cloud industry that led to the acquisition by this provider.

AWS offers different types of solutions some more oriented to data mining or Internet of Things (IoT). The solution chosen was the Elastic Cloud (EC2).

What amazon says about is product:

“Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. It is designed to make web-scale cloud computing easier for developers. Amazon EC2 simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon is proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change. Amazon EC2 changes the economics of computing by allowing you to pay only for capacity that you

actually use. Amazon EC2 provides developers the tools to build failure resilient applications and isolate them from common failure scenarios.” [37]

The citation above is available in amazon web page as the description of its product and can describe very well it is service EC2.

To configure an instance in the cloud one can, choose different operating systems depending on the purpose of the project. It is up to the user to decide. In the present project was opted for an Ubuntu Linux operative system. It is easy to configure the server from the image of the chosen operating system available on the amazon configuration page, after selecting the operating. It only takes a few minutes until everything is operational.

After getting the cloud server operational, it was needed to install some packages for our database and scripts. So, for server was installed Apache2 for database was MySQL and for the scripts PHP7.

The database of this system is made using MySQL and phpMyAdmin for administration. Both apps for mobile and web use the REST API developed to access the database. The REST API was developed in PHP code and is responsible for handling all database connections and requests.

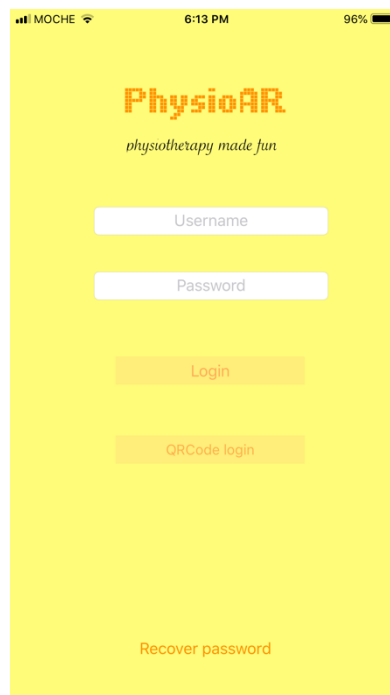
The server has also the code for the web application stored in it making it a very big component of this system responsible for the backend application and the database operations and storag

## 7 Mobile Application

In this chapter is described the mobile application. This application allows a patient to perform an independent training in case it is physically possible. However, certain patients may find themselves in a very bad state in which they cannot even move, and for this situation a presence of a physiotherapist is required. The application workflow will also be described.

### 7.1 Login

When the user opens the application the first screen that appears is the login screen. Figure 7.1 shows the login screen.

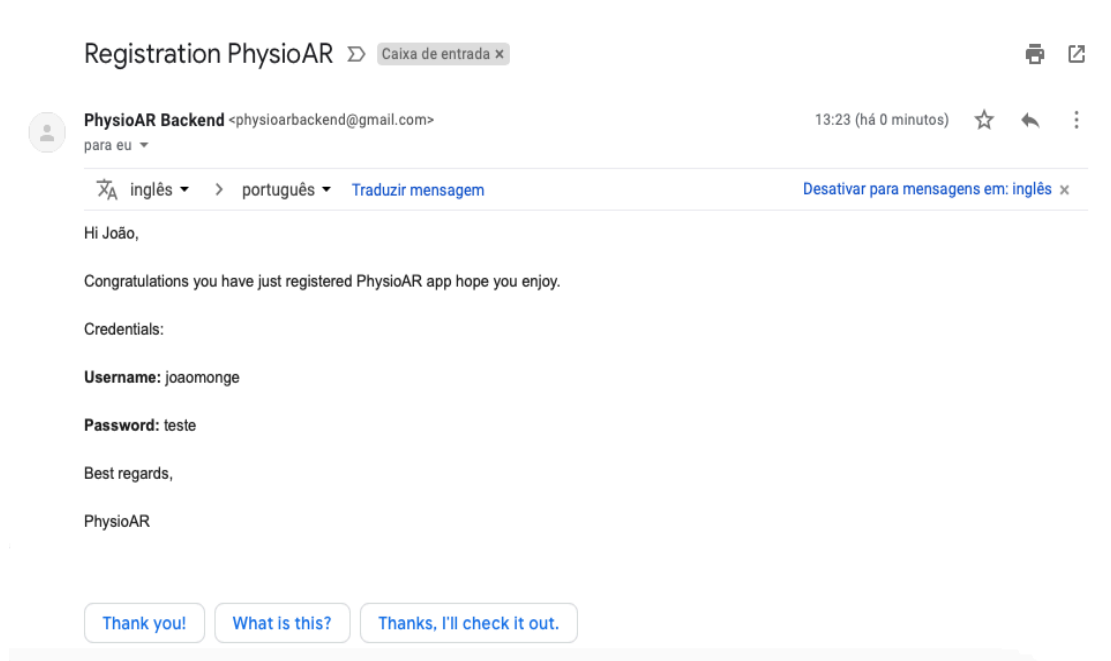


*Figure 7.1 - Mobile App - Login Screen*

The user has several options on the login screen. The user can enter the access credentials that is received in the user email when the physiotherapist make the registration of the user/patient. Another option is to scan the QRCode. This option allows the user to use the card that was given by the physiotherapist to access the application. Finally, it is also possible in the login screen to recover the password. It will now be described in more detail these three options:

- **Credentials login:** This is the primary way to access the app content. To do so after a physiotherapist register a patient an email is sent to the patient with the login and password. An example of the email sent is represented in Figure 7.2.

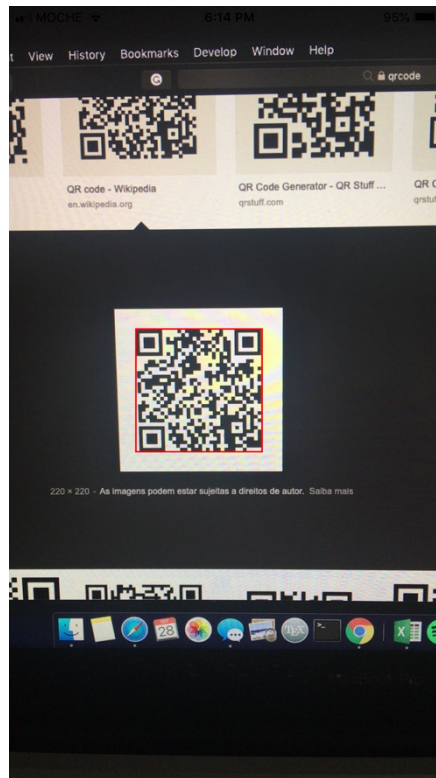
With the credentials received in the email the user can now access is account by filling the fields and pressing Login button, then if their valid the user logs successfully.



*Figure 7.2 - Registration succeeded email*

- **QRCode Login:** As this application is for various age groups and it is known that the older have more difficulties with the technology. This system contains an easier way to login using only the camera of the smartphone which prevents the user having to write using a touch keyboard. This option is possible if the physiotherapist has delivered the access card generated to his patient.

If the patient has the QRCode card in his possession, then he can click on the QRCode login button that will take the user to the screen shown in Figure 7.3.



*Figure 7.3 - QRCode Login Screen*

In this screen the camera of the smartphone is opened, and the user must point to his access card. If a QRCode is detected it is surrounded by a red rectangle. If QRCode belongs to a registered user, then a beep will sound, and the user is redirected to the next screen which is described in the next sub-chapter.

- **Recover password:** In this option allows the user to receive a new password that is sent by email. This feature allows users not to lose access to the account if they forget the password.

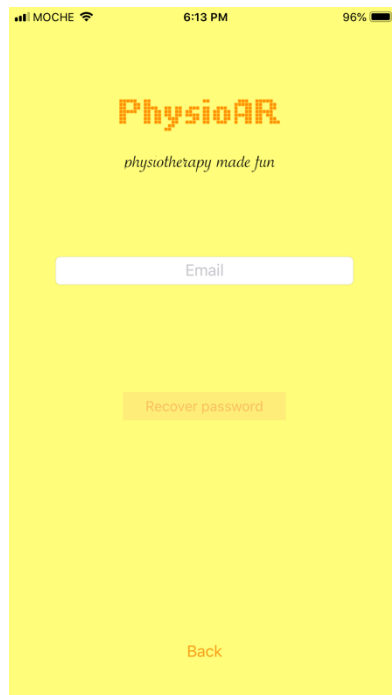


Figure 7.4 - Recover Password Screen

The email received after the user fills the email and presses recover password button is presented in the Figure 7.5.

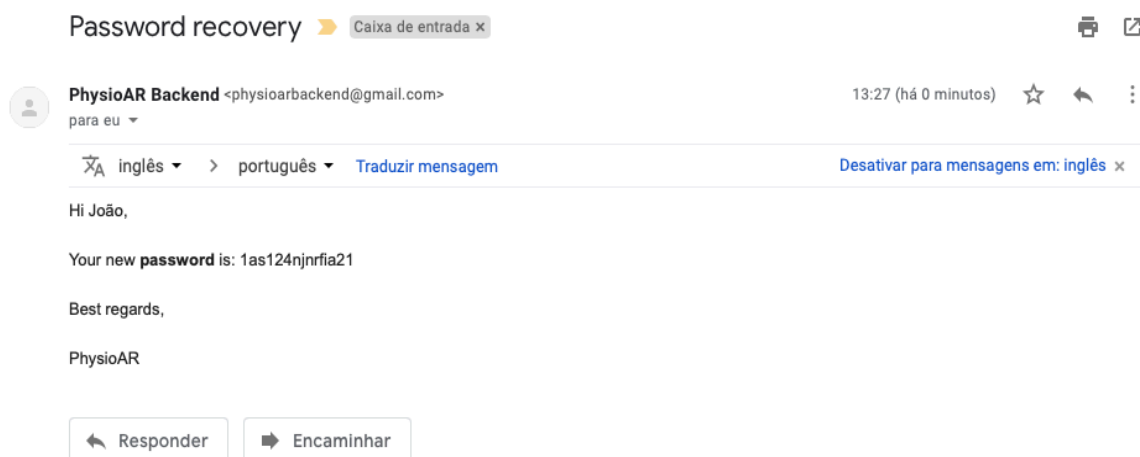
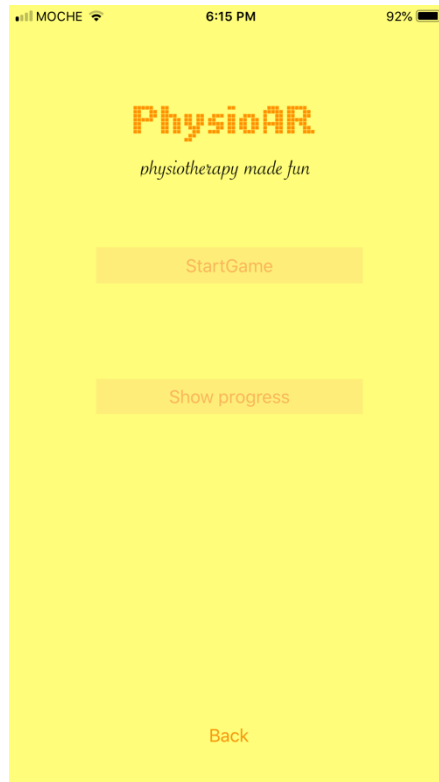


Figure 7.5 - Password recovery email

## 7.2 Start Game screen

After a successful login the user is redirected to the game home screen. Here the user has two options to start the game or see their progress.



*Figure 7.6 - Start Game Home Screen*

The screen shown in Figure 7.6 represents the start game screen the two buttons:

- **Start Game:** This button will take the user to the page described in the next sub-chapter 7.3.
- **Show progress:** This page will take the user to the screen shown in Figure 7.7.

In the progress screen the user can get a visual feedback of his progress.



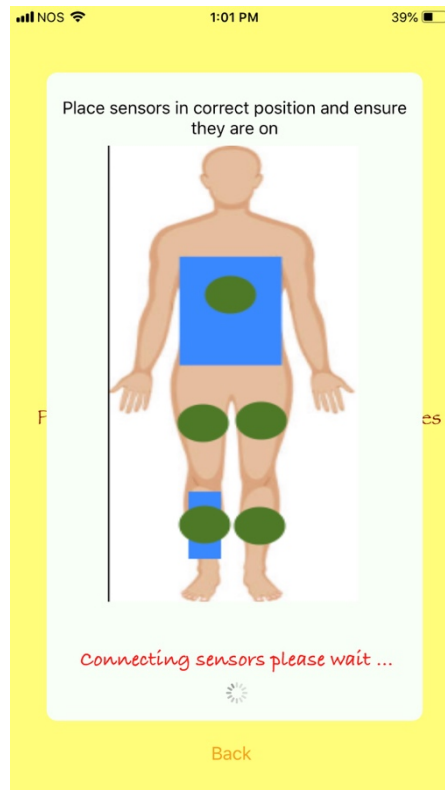
Figure 7.7 - Progress screen

- **Back:** This button takes the user to the Login page described before;
- **Show Graph Button:** This button generates the graphs for the selected date

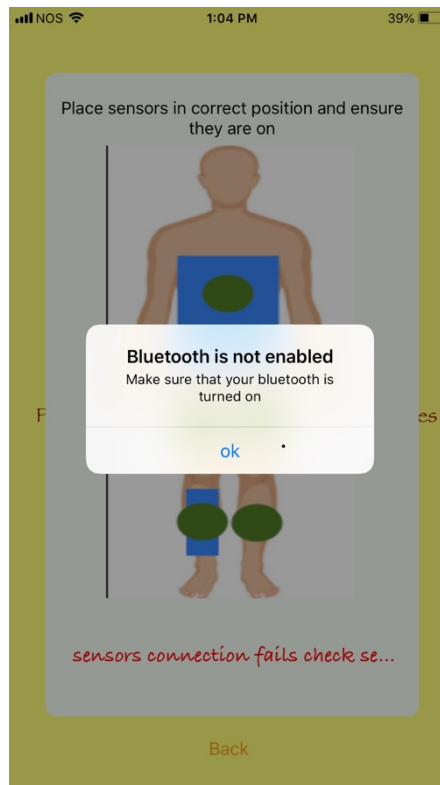


## 7.3 Starting game

This screen is responsible for loading the game and checking if the sensors are correctly connected. First a pop-up appears (Figure 7.8) and image is showed to indicate the user where he should place the sensors on his body. Now there are several options that could happen as a message that is displayed in case of the Bluetooth is disconnected (Figure 7.9).

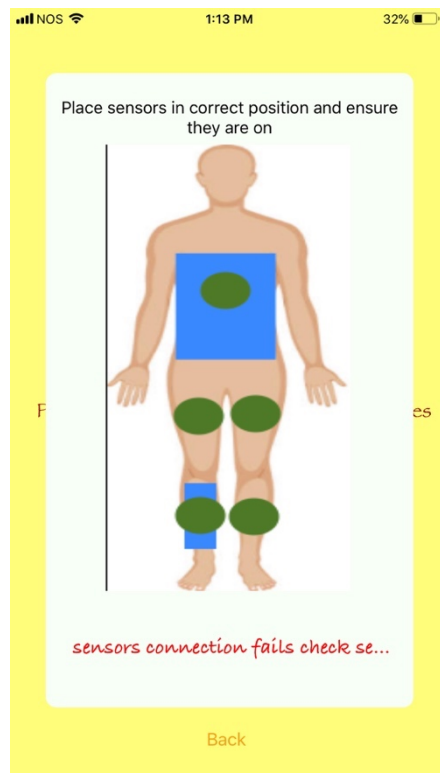


*Figure 7.8 - Connecting sensors pop-up*



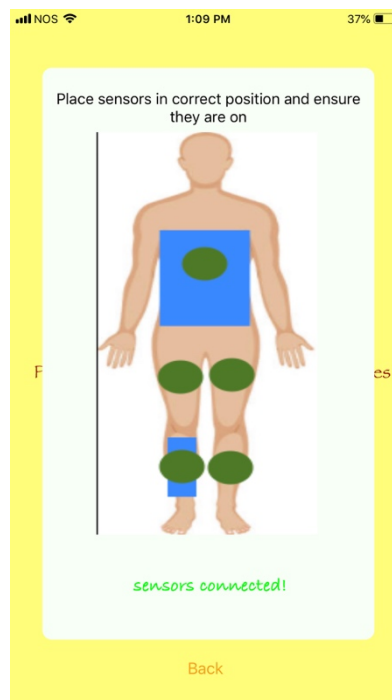
*Figure 7.9 – Bluetooth not enable*

Another option is that some sensor fails to connect, and message is also showed (Figure 7.10).



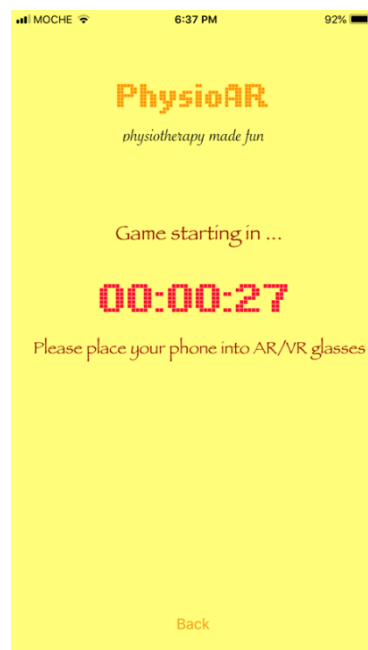
*Figure 7.10 – Sensors connection failed*

Finally, if everything is OK a message is shown (Figure 7.11).



*Figure 7.11 - Sensors connected successfully*

After 2 seconds this pop up hides and a countdown of 30 seconds initiates (Figure 7.12) meaning that the game starts loading.



*Figure 7.12 - Countdown screen*

This countdown was created to allow the time necessary for the user to place the smartphone into the VR goggles box.

This VR googles box become very popular when google decided to create a cardboard version that is like the one presented in Figure 7.13.



*Figure 7.13 - Google cardboard*

This cardboard was cheap effective and transformed almost any android smartphone into a VR headset. They had NFC chip that opens a app on the phone when it is inserted in the cardboard box.

In this project was used a plastic version of the box (Figure 7.14) to increase durability and to give a better final look to the system but the work principal is the same.



*Figure 7.14 - VR Universal smartphone box*

After the phone is inserted in the VR box and the timer finishes the game environment appears (Figure 7.15).



*Figure 7.15 - Game Screen*

The game will now run following the plan defined by the physiotherapist for the user. When it ends the user can exit the app. The data coming from the sensors will already be in the cloud.

## 8 Results

In this chapter the experiments of this system are presented. A comparative analysis of the results obtained among different volunteers was made, with a set of 9 volunteers who experimented the system in different phases. Firstly, are presented preliminary results obtained by the system those results that were included in the scientific paper. Then laboratory tests with young volunteers of different ages. Finally, the data obtained from a of three soccer players who were performing physiotherapy is analyzed.

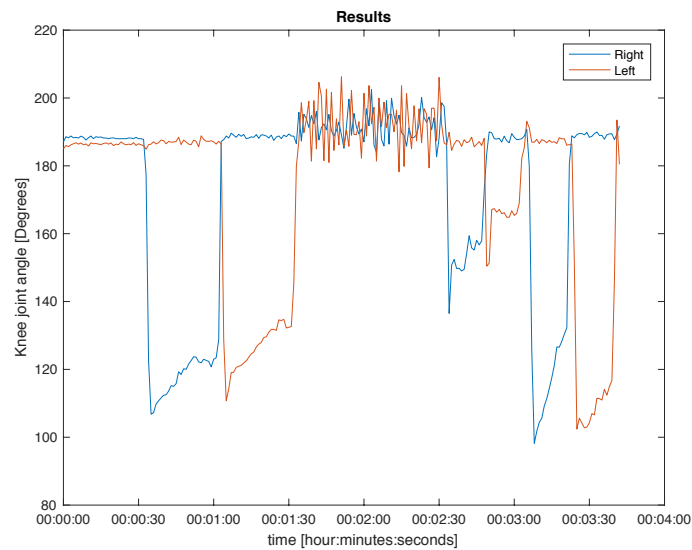
### 8.1 Preliminary tests

The methods and results for these preliminary tests were presented in a scientific paper that was submitted for a conference. These tests were carried out for analysis of efficacy of motion detection sensors in the lower limbs, thus excluding the electromyogram and the electrocardiogram since the e-textile electrodes were not yet ready at this time. The tests demonstrate the good capability of the system in terms of the correct gait analysis but also prove that the sensors are able to communicate with the server and with the Augmented Reality application.

Three volunteers were invited to participate to the tests. The volunteers consisted of a 23-year-old girl (volunteer 1), a 23-year-old boy (volunteer 2) and a 69-year-old lady (volunteer 3). The three volunteers were registered in the application and were given each one a QRCode that allowed access to the mobile application and in turn start the training.

The same training plan was also given to everyone so that a comparison could be made between the different participants.

The obtained results were processed through Matlab software and represent the angle of the knee joint during the execution of the training session.

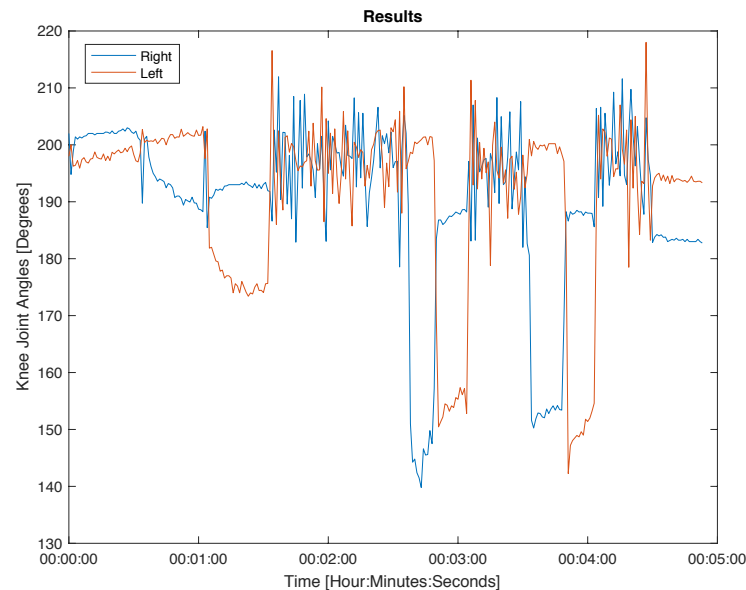


*Figure 8.1 - Volunteer 1 results*

In the Figure 8.1 are the results of the 23 years old female volunteer. By signal analysis the information that can be extracted is:

- **Equilibrium:** The signal indicates good equilibrium since low noise is presented in the graph and noise is created with the slight movements;
- **Both legs performed equal:** It possible to observe that both legs had an almost equal performance. In 02:45 to 03:00 comparing with 02:30 to 02:45 it is possible to observe a relevant difference performing the same exercise in one leg and the other. Both legs working equal can suggest that the person has no equilibrium problems;
- **Good flexion:** The angle is as a metric to determine the flexibility of the participant. In this case the participant suggests a good flexion;

Considering these points, it was possible to conclude that this participant is in good shape which in fact matched to the reality of the participant.



*Figure 8.2 - Volunteer 2 results*

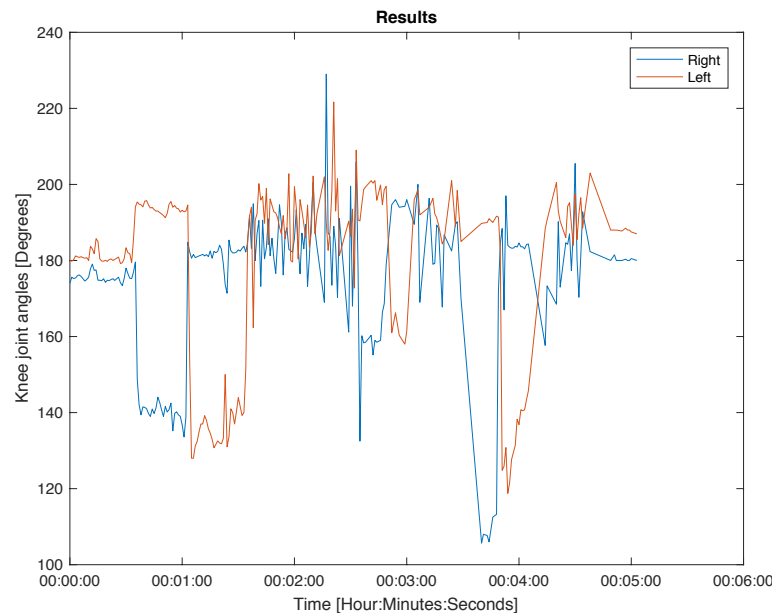
The next volunteer was one male and the results are represented in Figure 8.2. Now comparing the results with those from volunteer 1 (Figure 8.1) a great difference in the noise of the obtained data appears. This is justified by the fact that this volunteer has less ability to balance than the previous one which causes more noise because it makes small movements which originate it.

Now comparing to volunteer 1:

- **Less equilibrium capabilities:** As stated before this volunteer showed less capabilities of maintaining equilibrium;
- **Both legs performed equal:** As similar to volunteer 1 both legs had a similar performance in achieving the same angle;
- **Knee flexion:** Comparing with volunteer 1 the obtained angle is smaller however it is considerable reasonable for a healthy person.



By comparing volunteer 1 and 2 it is possible to identify several differences performing the same train. Volunteer 1 had better results than volunteer 2. This suggests that the system is capable of identifying differences in the gait by performing a real-time analysis during training session. As the purpose of these tests was to test the operating principle of this system was concluded at that time that the system is fully capable of achieving its purpose.



*Figure 8.3 - Volunteer 3 results*

A final test was carried out with volunteer 3, Figure 8.3 represents the results of the volunteer. The differences among the three volunteers are clear represented in their results. Comparing volunteer 3 to volunteer 1 the difference is very big in both flexion and equilibrium performance, the volunteer 1 obtained the best results. Now comparing volunteer 3 to volunteer 2 the difference is not so big but it is clear that the volunteer 2 had better results.

Some highlights of the results of volunteer 3 are:

- **Worst equilibrium:** Volunteer 3 showed the worst equilibrium among the three although that does not mean that Volunteer 3 has any visible or relevant difficulty daily basis, only indicates the age factor and its consequence in the gait of the human being
- **Both legs performed equal:** In all other volunteers it has been observed that both legs function equally, thus not indicating problems in one of the limbs alone.
- **Knee Flexion:** Volunteer 3 showed good flexion results.

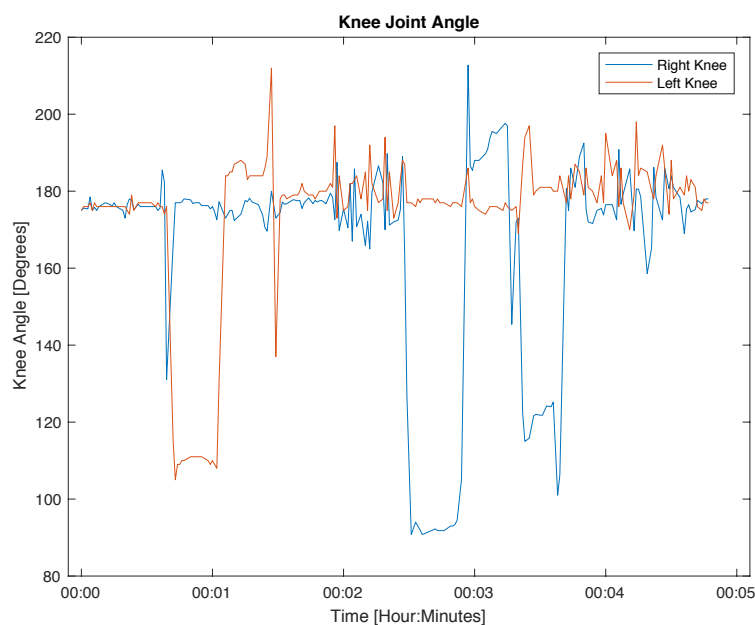
The main purpose of these tests was to see if the system would work using the knee joint angles as the primary metric for assessing user performance. We concluded at the end of the tests that it would be a good metric revealing important factors like flexion, balance and differences between each leg behavior. The fact that the volunteers were of different genders and ages led different results even though all the volunteers were healthy. After these tests the system was developed to its final state and other metrics were included such as EMG and ECG. The next sub-chapter will present the test carried out after the system was complete.

## 8.2 Laboratory tests

Evaluation of efficacy of the prototyped system was carried out. Four participants were selected from an 8 - year - old boy, a 15 - year - old girl and two youngsters from 23. These tests were performed in the laboratory and all participants were healthy. Contrary to the preliminary tests the training plan was not the same for all the participants but one different for each one and also the duration of the sessions was not the same.

In these tests the EMG, ECG and knee angle data were included in the tests.

The first participant was a 23-year-old healthy girl.

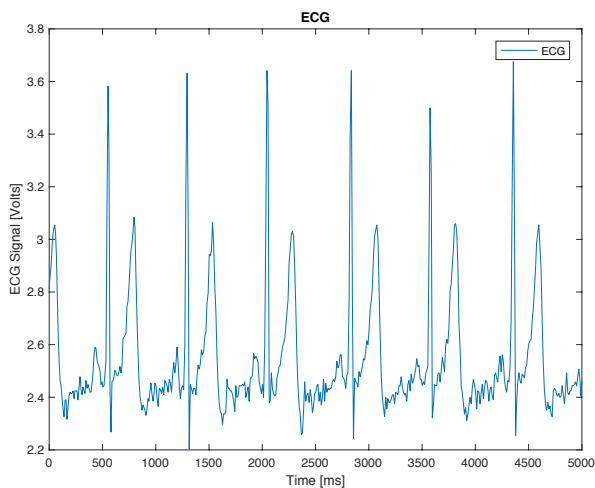


*Figure 8.4 - Participant 1 knee joint angles*

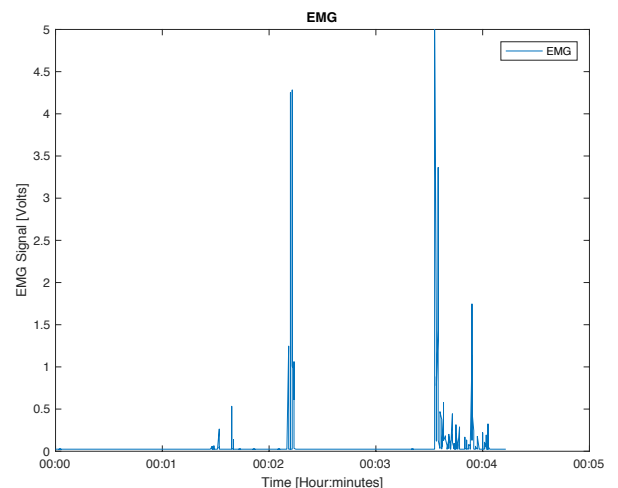
The graph in Figure 8.4 is like the previous ones obtained in the preliminary tests and as referred previously contains useful data for the evaluation of balance and flexion of the lower limb. An analysis of the graph:

- **Equilibrium:** Low noise in output and low difference between both legs reveal that this participant has good equilibrium;
- **Knee flexion:** The achieved angles suggest that this patient has good flexion on both legs although right leg showed better performance than left.

The output graphs from participant 1 are showed in Figure 8.6 and in Figure 8.5:



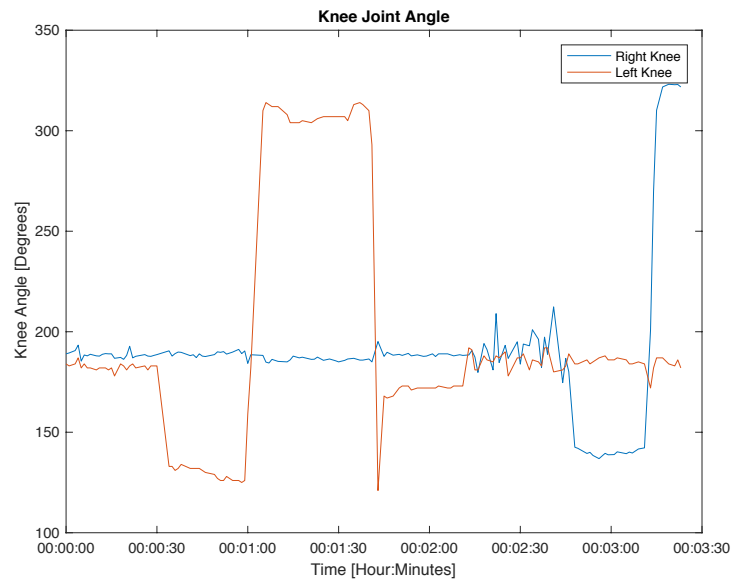
*Figure 8.5 - Participant 1 ECG*



*Figure 8.6 - Participant 1 EMG*

Figure 8.6 represents the EMG signal captured from the participant right calve muscle during the whole train session. This metric is important since it allows to detect how much effort the muscle is making to maintain a certain position. More information on muscular activity can be obtained by combining Figure 8.4 graph and Figure 8.6. Analyzing the EMG graph, can be seen that participant 1 calve was in motion when the participant walked at 00:02:00 and when performed the exercise between 00:03:30 and 00:04:00.

Figure 8.5 represents part of the ECG signal captured. ECG signal can be used to assess heart rate (beats per minute - BPM) but it is also possible to analyze several other components of the ECG wave like PR interval, QRS complex, QT interval. Other metrics would require a better acquisition board since this produces noise that makes the signal unclear. During the session participant 1 has an average heart rate of 72 BPM extracted from the ECG signal which is normal for the age of the participant and for executed exercises.



*Figure 8.7 - Participant 2 knee joint angles*

The second participant was a healthy male with 23 years old. The results obtained are showed below:

Figure 8.7 shows the knee joint angle during the train session. The results obtained indicate:

- **Good Equilibrium:** The signal has low level of noise originated form slight movements which is a good indicator that the participant has good equilibrium.
- **Good Flexion:** The angles achieved suggest that the participant has good knee flexion on both sides. This is observable by looking at minimum and maximum angles in the graph of Figure 8.7.

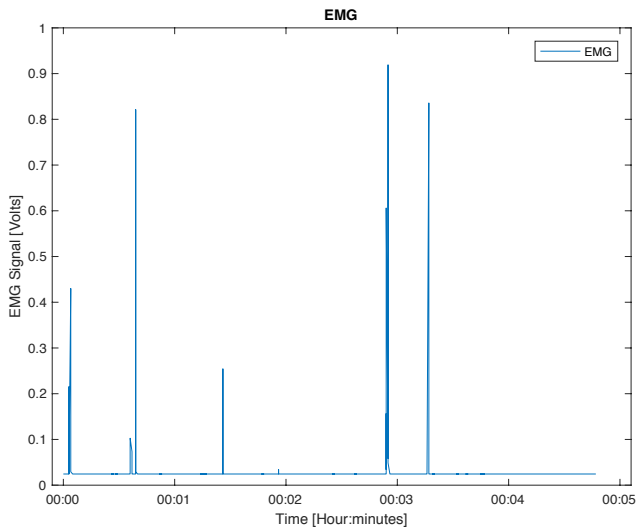


Figure 8.8 - Participant 2 EMG

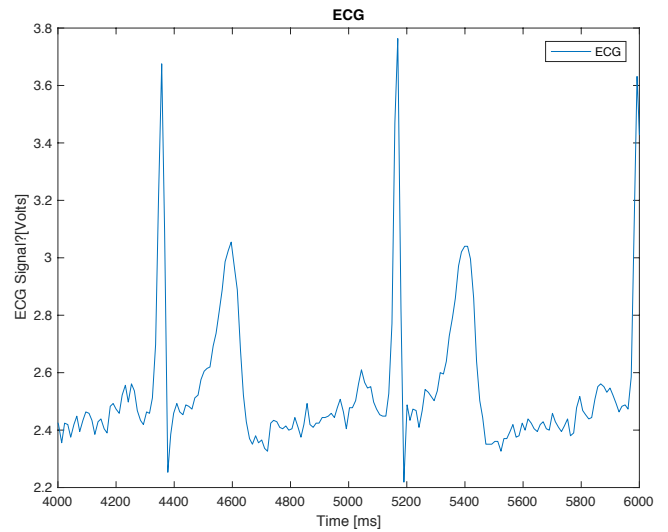
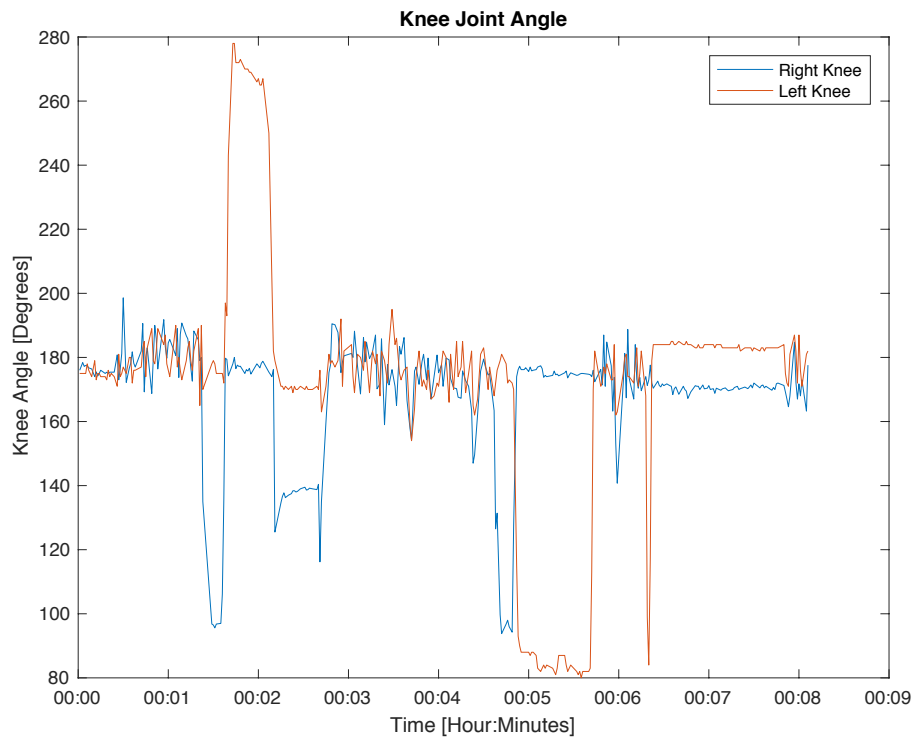


Figure 8.9 - Participant 2 ECG

An example of ECG acquired from that volunteer is represented in Figure 8.8. The average heart rate of this volunteer was 60 BPM mean which is normal because this participant is an athlete. The EMG results (Figure 8.9) showed that although exercising left leg the participant used is right calve muscle maybe to maintain equilibrium, and in the end of the train it is also possible to observe some activity on is right leg that can be observer in both EMG and Knee Joint Angle graphs.

The next participant is a female with 15 years old. The train had a duration of 8 minutes and the participant was healthy and in good shape.



*Figure 8.10 - Participant 3 Knee joint angles*

Figure 8.10 contains patient 3 results, this participant performed one of the longest trains. The output is therefore very rich. The data acquired suggest that:

- **Equilibrium:** The results obtained indicate that this participant had no equilibrium problems. The oscillation in values during 00:00 to 00:01 and 00:03 to 00:05 and in 00:06 the participant was induced by walking;
- **Knee flexion:** The angles obtained indicate that the participant has good flexion in knees;
- **Legs performance difference:** The legs showed low difference during the train which indicates that gait function for the volunteer was normal.

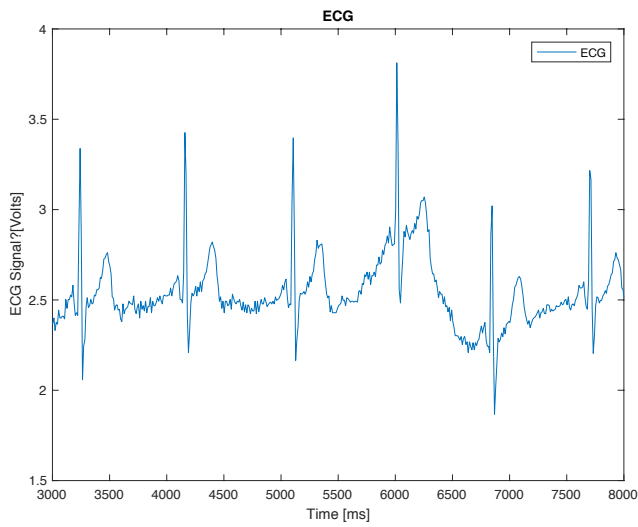


Figure 8.11 - Participant 3 ECG

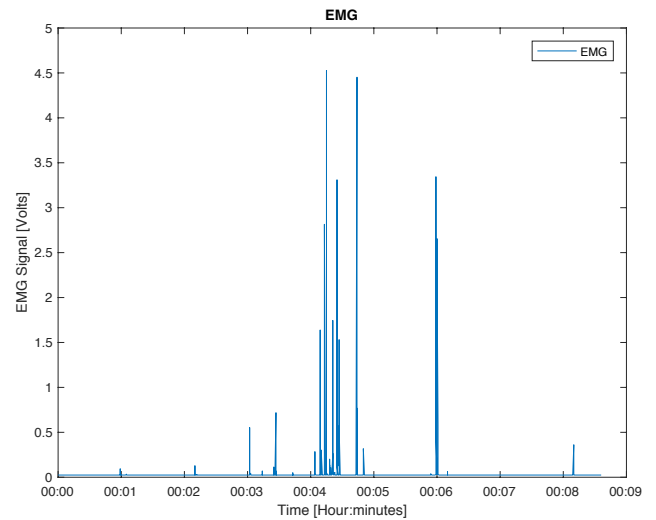


Figure 8.12 - Participant 3 EMG

Participant 3 EMG and ECG results are showed in Figure 8.11 and Figure 8.12 respectively. The EMG results show that participant 3 is using the right calve muscle when walking and when performing some exercises. The extracted average heart rate from ECG signal was 72 BPM.

The last participant was 8-year-old boy. This participant was the younger, so the duration of the train was very low at about one minute and half.

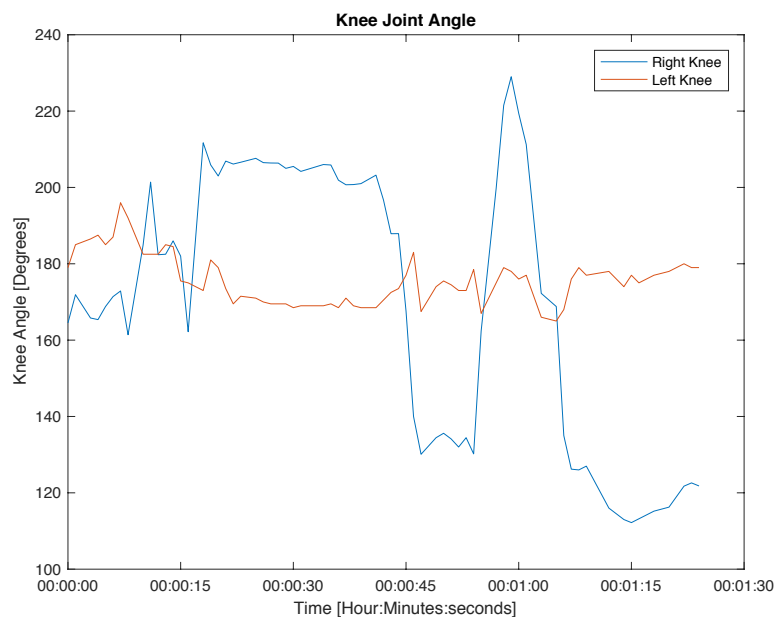


Figure 8.13 - Participant 4 knee joint angles

The results in Figure 8.13 show:

- **Equilibrium:** The kid had more struggle maintain equilibrium which is normal considering his age. This struggle can be seen in the graph in the form of noise but also in duration when the kid lifts his right leg at 00:00:45 to 00:01:00. It was supposed to maintain his leg up for 15 seconds, but it only endured 10 seconds;
- **Knee flexion:** The results demonstrated that the kid had good knee flexion by observing maximum and minimum reached angles;

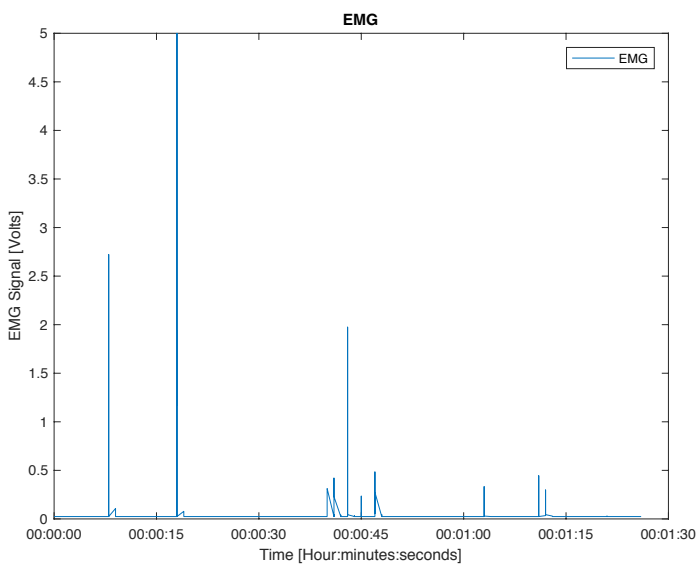


Figure 8.14 - Participant 4 EMG

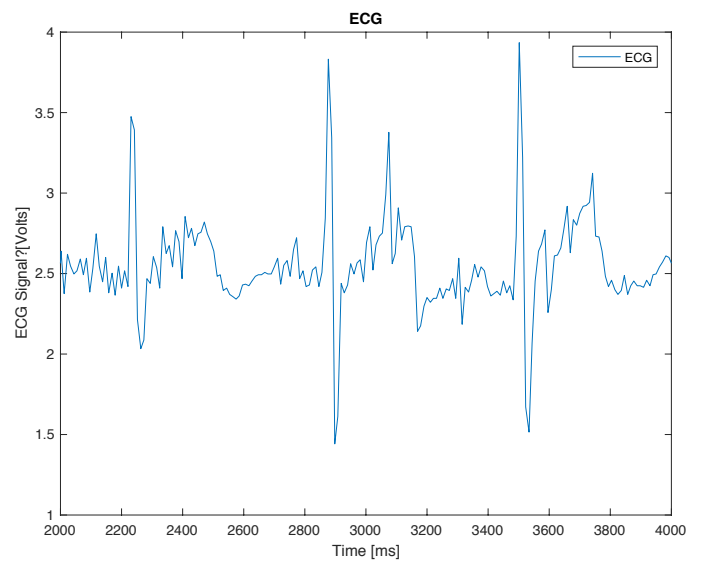
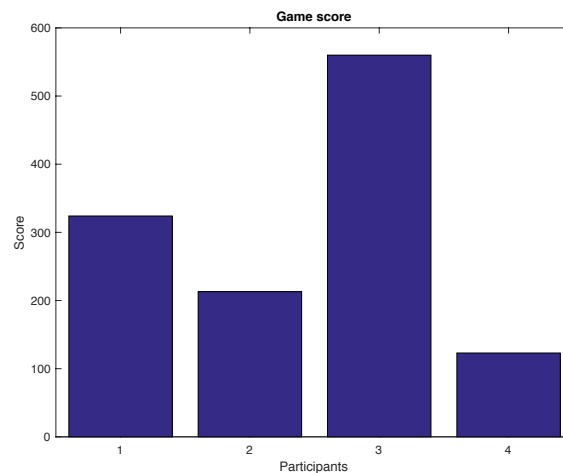


Figure 8.15 - Participant 4 ECG



For the participant 4 EMG signal showing muscular activity in the right calve was obtained (Figure 8.14) during exercise his right leg. The average heart rate obtained from the ECG was 90 BPM which is slightly high but can be explained due to the fact that the children has great fun with the game which explains the heart rate increase.



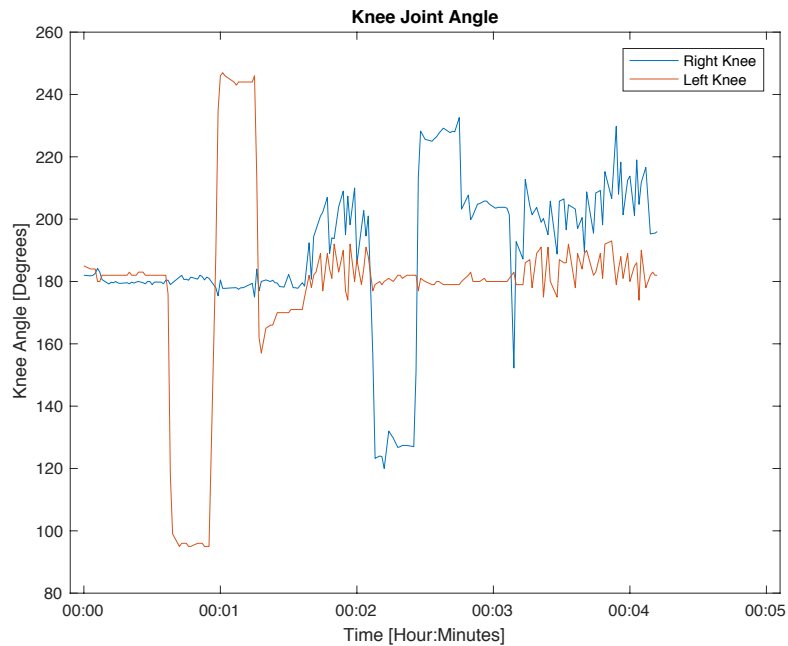
*Figure 8.16 - Participants score*

It was not possible to extract significative results on efficacy of the system as the three participants did not perform the same exercises and even the scores obtained in game were influenced by the duration of it. Although the scores obtained are shown in Figure 8.16 they do not represent anything rather than a motivation. A threshold was set for the acquired angle and every time the user passes it points are obtained. This score system was made to increase motivation.

All 4 participants enjoyed the train session and considered these an innovate way for physical rehabilitation. Participant 4 was the person more motivated to use the Augmented Reality Headset which indicates that this system could be very useful for motivating children to perform their train.

## 8.3 Clinical tests

These tests were carried out under the supervision of the physiotherapist Francisco Pratas and with his patients. The patients were football players at Académica de Santarém Football Club which were performing physical rehabilitation due to injuries caused by the sport. The physiotherapist allow the test in three patients all in final stage of recovery process, patient 1 was in rehabilitation of his right leg after a fracture, patient 2 and patient 3 had muscle injuries.



*Figure 8.17 - Patient 1 results*

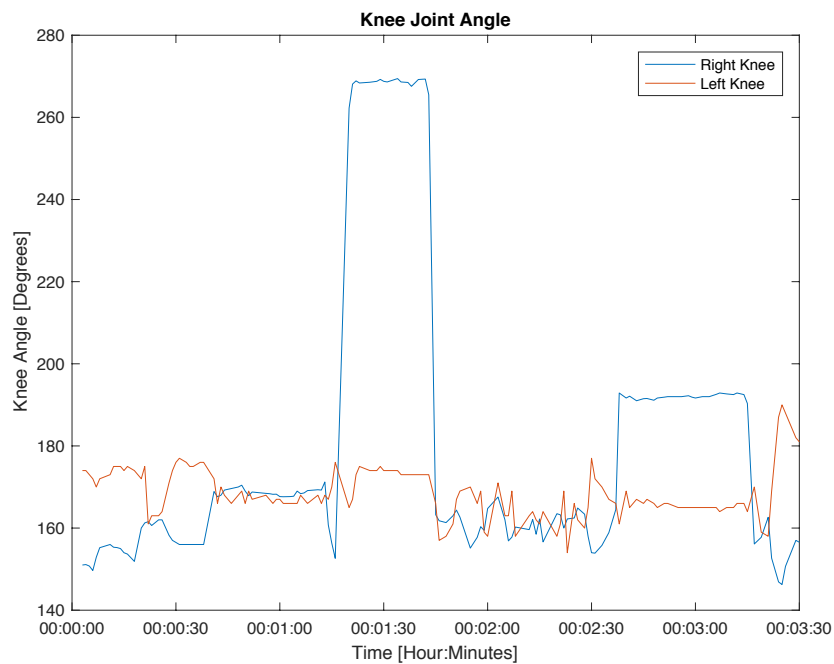
These tests did not include ECG and EMG only gait analysis.

The results for patient 1 are showed in Figure 8.17. The patient 1 had broken his right leg a few months before the test but was almost recovered when this test was performed.

The results indicate:

- **Equilibrium:** The patient was able to maintain equilibrium easily as results show no noise when the patient was performing exercises.
- **Difference between legs:** The results show clearly a big difference between legs of 20 degrees due to his broken leg. The 20 degrees indicate that the patient is almost recovered from such injury;
- **Flexion:** As expected from a football player the patient revealed good flexion in the tests;

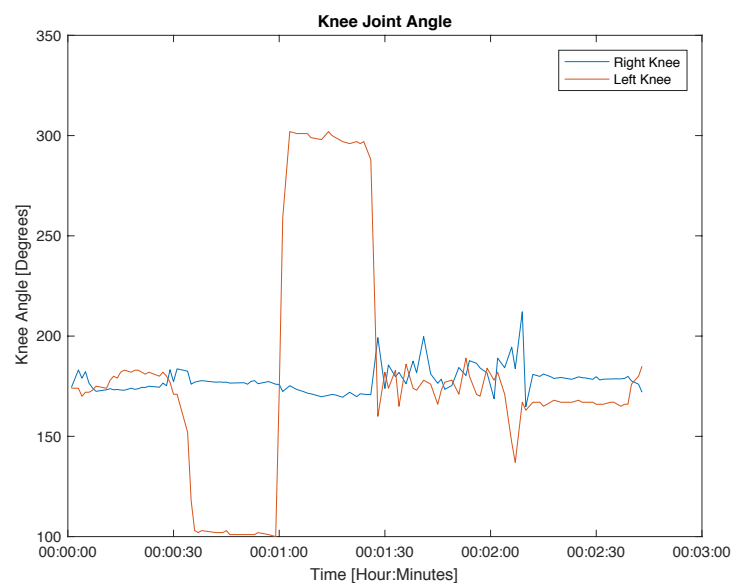
Patient 2 had a fresh muscle injury, so the tests were carried out only on his left leg due to physiotherapist advisement.



*Figure 8.18 - Patient 2 Results*

In Figure 8.18 is represented the patient 2 results. A further analyze indicates:

- **Equilibrium:** The results indicate that this patient had more difficulty maintaining the equilibrium than the other patients in this test.
- **Knee flexion:** The results suggest good flexion on his right leg.



*Figure 8.19 - Patient 3 knee joint angles*

Finally, patient 3 results are presented in Figure 8.19. A further analyze:

- **Equilibrium:** The results show no problems with equilibrium analyzing the noise of the obtained data;
- **Knee flexion:** Very good knee flexion was observed by analyzing maximum and minimum angles;

The physiotherapist feedback on backend application was that it has an intuitive interface and it was very useful to keep the history of the patient training.

Therefore, the tests were carried out in three different phases one was preliminary tests, final tests and clinical tests. The system needs to be tested in long term to see its reliability and performance of the use of Augmented Reality with physiotherapy.

## 9 Conclusions and Future Work

In this chapter are presented the conclusions and future work.

### 9.1 Conclusions

The MSc thesis presents a system for interactive physiotherapy that intends to mitigate one of the biggest problems that patients go through, the lack of motivation. Thus, was developed PhysioAR that represents a set of tools that help both the patients and the physiotherapists. These tools include two distinct applications one that can mainly be used by the physiotherapists and other that can be used by patients. The application of the physiotherapist allows him to make a management of all his patients as well as create personalized training for each one, and to analyze the patient performance during training sessions. On the other hand, the patient has a mobile application and a set of sensors that allow him to enter a world of Augmented Reality and to visualize on-line his actions through appropriate metrics. With this the patient is assisted during his training by a 3D avatar (virtual coach) that will help him throughout the process. Extended information is obtained by the sensors which allow to monitor of the patient movements and communicate with the physiotherapist mobile application. The whole objective of the project proposal was reached, however new challenges were identified regarding the usage of AR on the particular field of Physical rehabilitation.

## 9.2 Future work

This system could be optimized in some of aspects:

- Long term testing the system with patients could lead to improvements because it can reveal more accurate data and the introduction of data analyze algorithms in the system that can reveal more details about user evolution;
- The smart sensors could be redesigned in order to make them smaller and more efficient as new wearable technologies are emerging every day. Improvements on sensors can enhance user experience;
- Data analysis algorithms for progress analyzing requires developments;
- User localization algorithms that give an accurate position of the user in the virtual environment could
- An Android AR Core version of this system to make it compatible with the majority of mobile devices on the market;
- Optimizing sensors for data transmission in real time without delays that could let real-time remote analysis of users;

## References

- [1] W. H. Organization, “Men Ageing And Health,” *J. ISSAM*, vol. 233, no. June, pp. 1–63, 2001.
- [2] “Medicine.” [Online]. Available: <https://en.wikipedia.org/wiki/Medicine>. [Accessed: 11-Jan-2018].
- [3] M. Shaughnessy, B. M. Resnick, and R. F. Macko, “Testing a model of post-stroke exercise behavior,” *Rehabil. Nurs.*, vol. 31, no. 1, pp. 15–21, 2006.
- [4] G. Yang *et al.*, “A Health-IoT platform based on the integration of intelligent packaging, unobtrusive bio-sensor, and intelligent medicine box,” *IEEE Trans. Ind. Informatics*, vol. 10, no. 4, pp. 2180–2191, 2014.
- [5] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of Things (IoT): A vision, architectural elements, and future directions,” *Futur. Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, 2013.
- [6] A. Darwish and A. E. Hassanien, “Wearable and Implantable Wireless Sensor Network Solutions for Healthcare Monitoring,” *Sensors*, vol. 11, no. 6, pp. 5561–5595, 2011.
- [7] A. Sixsmith and N. Johnson, “A smart sensor to detect the falls of the elderly,” *IEEE Pervasive Comput.*, vol. 3, no. 2, 2004.
- [8] Z. Zhang, “Microsoft kinect sensor and its effect,” *IEEE Multimed.*, vol. 19, no. 2, pp. 4–10, 2012.
- [9] N. Duarte, O. Postolache, and J. Sharcanski, “KSGphysio – Kinect Serious Game for Physiotherapy,” *Int. Conf. Expo. Electr. Power Eng.*, no. Epe, pp. 16–18, 2014.
- [10] S. Vukicevic, Z. Stamenkovic, S. Murugesan, Z. Bogdanovic, and B. Radenkovic, “A new telerehabilitation system based on internet of things,” *Facta Univ. - Ser. Electron. Energ.*, vol. 29, no. 3, pp. 395–405, 2016.
- [11] K. J. O’Donovan, B. R. Greene, D. McGrath, R. O’Neill, A. Burns, and B. Caulfield, “SHIMMER: A new tool for temporal gait analysis,” *Proc. 31st Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. Eng. Futur. Biomed. EMBC 2009*, pp. 3826–3829, 2009.

- [12] “Shimmer3 IMU.” [Online]. Available: <https://shimmersensing.com/products/shimmer3-imu-sensor>. [Accessed: 09-Sep-2018].
- [13] G. Ins, “MTi 1-series Data Sheet,” no. July, 2018.
- [14] S. Majumder *et al.*, “Non-Contact Wearable Wireless ECG Systems for Long Term Monitoring,” vol. 3333, no. c, pp. 1–18, 2018.
- [15] S. Xing and X. Zhang, “EMG-driven computer game for post-stroke rehabilitation,” *2010 IEEE Conf. Robot. Autom. Mechatronics, RAM 2010*, pp. 32–36, 2010.
- [16] E. Jovanov, A. Milenkovic, C. Otto, and P. C. De Groen, “A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation,” *J. Neuroeng. Rehabil.*, vol. 2, pp. 1–10, 2005.
- [17] “Wii Fit.” [Online]. Available: <https://www.nintendo.com/games/detail/wii-fit-u-packaged-version-wii-u#game-info>. [Accessed: 10-Sep-2018].
- [18] M. K. Tageldeen, I. Elamvazuthi, N. Perumal, and T. Ganesan, “A virtual reality based serious games for rehabilitation of arm,” *2017 IEEE 3rd Int. Symp. Robot. Manuf. Autom.*, pp. 1–6, 2017.
- [19] W. Ying, “Augmented reality based upper limb rehabilitation system,” 2017.
- [20] “Google Fit.” [Online]. Available: <https://play.google.com/store/apps/details?id=com.google.android.apps.fitness>. [Accessed: 10-Sep-2018].
- [21] “Apple Health.” [Online]. Available: <https://www.apple.com/ios/health/>. [Accessed: 10-Sep-2018].
- [22] “Strava.” [Online]. Available: <https://www.strava.com/features>. [Accessed: 10-Sep-2018].
- [23] “Bluejay App.” [Online]. Available: <https://itunes.apple.com/us/app/bluejay-engage-patient/id1161891845?mt=8>. [Accessed: 10-Sep-2018].
- [24] A. M. Rahmani *et al.*, “Smart e-Health Gateway: Bringing intelligence to Internet-of-Things based ubiquitous healthcare systems,” *2015 12th Annu. IEEE Consum. Commun. Netw. Conf. CCNC 2015*, no. June, pp. 826–834, 2015.
- [25] M. Duff *et al.*, “An adaptive mixed reality training system for stroke rehabilitation,” *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 18, no. 5, pp. 531–541, 2010.



- [26] O. Potolache, F. Lourenco, J. M. Dias Pereira, and P. S. Girao, “Serious game for physical rehabilitation: Measuring the effectiveness of virtual and real training environments,” *I2MTC 2017 - 2017 IEEE Int. Instrum. Meas. Technol. Conf. Proc.*, 2017.
- [27] H. C. Koh and G. Tan, “Data mining applications in healthcare,” *J. Healthc. Inf. Manag.*, vol. 19, no. 2, p. 65, 2011.
- [28] R. FRANK, “What is a smart sensor?” [Online]. Available: <https://www.sensortips.com/hot-topic/sensor-specific-software/what-is-a-smart-sensor/>. [Accessed: 12-Jan-2018].
- [29] RF Digital Corp., “RFD77101 DATASHEET v2.2,” vol. 43, no. 0, pp. 0–6.
- [30] “Arm Cortex M0.” [Online]. Available: <https://developer.arm.com/products/processors/cortex-m/cortex-m0>. [Accessed: 23-Sep-2018].
- [31] “Flora 9DoF.” [Online]. Available: <https://www.adafruit.com/product/2020>. [Accessed: 27-Sep-2018].
- [32] A. Devices, “AD8232 Datasheet,” 2013.
- [33] F. Application *et al.*, “REAL TIME CLOCK Patent,” *U.S. Pat. 5976719*, no. 19, 1999.
- [34] Apple inc., “Apple ARKit Documentation.” [Online]. Available: <https://developer.apple.com/documentation/arkit>. [Accessed: 29-Oct-2018].
- [35] Apple inc, “SceneKit.” [Online]. Available: <https://developer.apple.com/documentation/scenekit>. [Accessed: 02-Oct-2018].
- [36] L. M. Castano and A. B. Flatau, “Smart fabric sensors and e-textile technologies: A review,” *Smart Mater. Struct.*, vol. 23, no. 5, 2014.
- [37] “AWS EC2.” [Online]. Available: <https://aws.amazon.com/ec2/>. [Accessed: 04-Oct-2018].

## Appendix A - Scientific Articles

Article

Article: Augmented Reality and Smart Sensors for Physical Rehabilitation.

This article has been accepted and presented at the IEEE EPE 2018 international conference, October 18-19, Iasi, Romania and will be published in IEEE Explorer.



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.

# Augmented Reality and Smart Sensors for Physical Rehabilitation

João Monge  
ISCTE-IUL  
Lisbon, Portugal  
[jpdme@iscte-iul.pt](mailto:jpdme@iscte-iul.pt)

Octavian Postolache  
Instituto de Telecomunicações / ISCTE-IUL  
Lisbon, Portugal  
[opostolache@lx.it.pt](mailto:opostolache@lx.it.pt)

**Abstract**—The smart physical rehabilitation becomes a new reality and challenge regarding the technology adoption by the users and high costs. In this context, the paper presents a smart physical rehabilitation system that combines augmented reality serious games and wearable sensor network to improve the patient engagement during physical rehabilitation. The implemented system provides also health status assessment acquiring vital data to be analyzed by clinical professionals. The wearable sensing is based on the *Simblee* platform that captures kinematic and dynamic quantities associated with lower limb motion using an IMU as so as the muscular activation and cardiac activity. The Healthcare IoT compatibility is assured by an augmented reality platform based on iOS smart phone that communicates with wearable wireless sensor network and after primary processing of the data it uploads the analyzed in an AWS cloud based web platform.

**Keywords**—wearable smart sensors; augmented reality; Healthcare IoT; Cloud Computing; Serious Game

## I. INTRODUCTION

The physical rehabilitation of suffered an accident or a stroke attack is a long and intensive process in order to upturn to normal condition or at least an acceptable state of physical motion. This process entails high cost of both time and money for the patient, but this cost is generally increased when the patient feels demotivated which leads to extended rehabilitation periods.

To decrease the rehabilitation time by highly motivated physical training we developed a system that aims to mitigate this problems using state of the art technology such as Augmented Reality and wearable wireless sensor network to create an Internet of Things physical rehabilitation system that provides in home self-training capabilities. Although this system will always require a clinical professional supervision, it reduces that need and enables long distance assisted rehabilitation providing the possibility of treatment in remote areas with low assessment. To mitigate the other problem mentioned we also included in this system a serious game based on an Augmented Reality platform that helps the patient to stay motivated rewarding him with achievements for successful exercises done and also displaying real-time health data.

SG (Serious Game) means a computer game with extended propose that exclusively entertaining [1]. Digital games have the ability to engage both children and adults capturing their attention [2]. Nintendo Wii that was launched back in 2006

changed the way of the people interaction with games complementing physical motion as a way to control the game. This console was an innovation and demonstrated that common users enjoyed this way of playing and interact with consoles by using body motion as a game controller [3].

AR (Augmented Reality) systems merge computer-generated virtual objects with real world objects scenarios instead of an all virtual world as in VR (Virtual Reality) [4]. This is possible by tracking and positioning virtual objects integrating them in real world [5]. In this system we aimed to use the acquired data from several sensors in an AR Scenario. Thus three different sensing devices were used; a set of 5 IMU sensors is used to capture upper limb motion as well as equilibrium of the body during a training session, an ECG sensor that is used to extract the cardiac activity information such as heart rate which will reveal fatigue of the patient enabling us to detect when the user needs to pause, and finally the third one will be 2 EMG sensors for muscular contraction and movement detection which will show if the exercises are being done correctly.

Different works are reported in the literature [6]-[8]. Our team has been developing several serious games applications for physical rehabilitation with great success in clinics which confirms the capabilities of these systems for this purpose. The conclusions obtained by our team have been revealing the capabilities of this types of systems for physical rehabilitation although this will be our first work in Augmented Reality developed by our team several Serious Games systems in VR scenario had been developed and proven their capabilities in physical rehabilitation area [9]-[14]. Special attention is granted to IoT Healthcare compatibility of this system.

In this context the present work provides a solution for physical rehabilitation expressed by a set of serious game in an Augmented Reality scenario. A set of technologies including Apple ARKit, Apple iPhone, *Simblee*, ECG, EMG and IMU are used to materialize different type of interaction with different scenarios and levels of difficulties that are imposed set by a clinical professional on a backend web based application for a bigger range of device compatibility.

This paper is organized as follows: section two presents the related work; section three presents the system description, hardware used and software; section four shows first

experimental results done to the system components; section five includes the conclusions and acknowledgments.

### II. RELATED WORK

Nowadays, wireless body area networks are frequently applied as part of healthcare ecosystems. One good example is presented in [15] in 2005 and consists in a multi-tier tele-medicine system and a WBAN network for computed assisted rehabilitation applications and ambulatory monitoring performing real-time analysis of sensor data. More recently in 2010 this work [16] gives examples of state-of-art wireless sensor network systems designs for health care systems with considerations in unobtrusiveness, scalability, energy efficiency, security which are important considerations.

Also in 2010 an Adaptive Mixed Reality Training System for Stroke Rehabilitation was reported [17] which consisted by a system for helping people who have hemiparesis. This system provided real-time, multimodal, customizable, and adaptive feedback generated from the movement patterns of the subjects' affected arm and torso during reaching to grasp and provided feedback via an innovative visual and musical forms that are part of a stimulating, enriched environment for physical rehabilitation of subjects and training assessment based on a multimodal sensory-motor integration

In 2017 a hands rehabilitation system that uses Leap Motion Controller as natural user interface for a serious game was presented [9] the remote detection of the motion being captured with high accuracy but no information about the muscle activity was extracted.

A gait rehabilitation monitor that used wearable non-intrusive sensors Shimmer3 IMU was also developed in 2017 to capture patient motion patterns and was reported by our team [18].

### III. SYSTEM DESCRIPTION

The general architecture of the implemented system is presented in Fig. 1. It includes an iPhone 6s Plus as the main computation platform and a set of smart wearable sensors as nodes for data acquisition during a rehabilitation session.

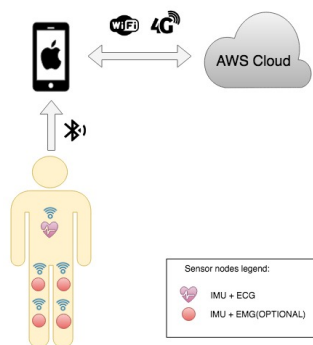


Fig. 1. System's architecture.

### A. System Hardware

- Smart Sensors

The smart sensors prototype uses a microcontroller inserted in a protoboard ready for development of wearable solutions in combination with sensor data acquisition boards. The hardware components are following presented.

#### 1) Microcontroller *Simblee Lilypad*

*Simblee* System on Chip [19] is characterized by small size of 10 mm x 7 mm x 2.2 mm (smaller than its original version RFDuino [20]), ARM cortex M0 processor, embedded Bluetooth Low Energy (BLE) communication and several interfaces: 29 GPIOs, I2C and SPI that are partially used in the present implementation.

*Lilypad Simblee* board presents a power regulator which provide the direct connectivity to a 3.7 V lithium-ion battery and a wearable friendly design. This wearable design was the key point for the choice of this board to this project as we needed wearable smart sensors. This board has a circular shape with e-sewable connectors for conductive thread which was used in this project for e-textile electrodes. This board we'll used in this project as the processing unit for the sensors.

#### 2) Inertial Measurement Unit (IMU)

Flora LSM9DS0 9DoF IMU, has an accelerometer, a gyroscope and a magnetometer, with 3 axes each one, and it's widely used as a motion detector in GPS navigation correction, virtual reality (VR) and robotics. The IMU presents a I2C communication interface and connected to the *Lilypad Simblee* providing information about patient's orientation and displacement.

#### 3) ECG - Heart Rate Sensor AD8232

Cardiac assessment is carried out using an ECG that is a common solution used in different health status monitoring systems [21].

The AD8232 ECG board characteristics are: 10 GOhm input impedance, 100 V/V gain. The board includes two pole high-pass followed by a two pole low pass filter for noise reduction of the noisy ECG signals.

#### 4) EMG – Advancer Technologies Muscle Sensor V3

Electromyography is a technique for muscle electrical activity measurement. The effectiveness of training in AR is evaluated in objective way using EMG module included in the system. This EMG module includes signal conditioning that provides functionalities such as amplifying, rectifying and filtering of the EMG signals that made the signal appropriate to be acquired by microcontroller ADC's.

- Computational modules

#### 1) iPhone 6s Plus

This smartphone has a 5.5' screen and a dual core 1.84 GHz processor which gives enough power that will be needed in this project in order to run the serious game in an augmented reality platform. It has also Bluetooth Low energy technology interface that is essential for the communication with the *Simblee*

microcontroller. It presents also communications capabilities such as 4G LTE, 3G, GSM and WIFI that makes an Internet connectivity possible in several conditions making it easier to connect to our Cloud App.

### 2) AWS Cloud Server

Amazon have been developing state of art cloud services suitable for a lot purposes with a good focus in IoT area.

In the proposed system the iPhone acts as a gateway between the sensors and the AWS Server.

For data storage we use an AWS Cloud Instance to store the system database, host the *WebApp* and reduce computational resources usage on data analyses on device and sensors to reduce battery consumption and CPU use on the iPhone and smart sensors. Our cloud server is Ubuntu based and as a *MySQL* database in it.

Using this type of cloud storage gives the benefits of secure data storing without facing the dangerous of losing the data. It's also more flexible than other types of servers making the system able to adapt to the type of app user growing.

Amazon also provides big data software which can be useful in the future of this system for discovering recover patterns.

With this cloud server we're able to access the *WebApp* via internet anywhere.

Amazon also provides a great layer of security which is very important IoT systems like this since we have sensible patient health data. iPhone will communicate to it as well as any device that accesses our *WebApp*.

### B. Software

The system software is expressed by three components. The first component is a Swift 3 APP for IOS platform the second is expressed by web based app with a wide compatibility of devices that act as a backend app which is directly connected via Internet with Swift 3 APP. The last one is the embedded software running on sensors nodes as referred before. The nodes perform the primary computation of the data before sending to swift 3 APP. The software interactions diagram is presented in Fig. 2.

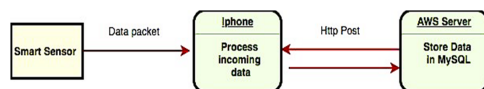


Fig. 2. Data handling diagram.

### 1) System App

The system App is running on iPhone so the APP was developed using Apple Swift 3 language considering that Apple has its own Augmented Reality Framework that was used for the game engine.

The game in this app and loads the patient information and his train plan after a successful login. To login the user as two options, one is to use username and password combination and the other is a QRCode which facilitates login for patients with less mobility.

### 2) Backend web App

The backend of the system was created to provide support for clinical professional related the set up the patient training as well as to consult the patient progress.

### 3) Sensors Embedded Software

The sensors embedded software was written in Arduino language and is responsible for acquisition and primary processing of the signals from measurement channels and later to send it via Bluetooth in an effective way.

- Prototypes

In Fig. 3 is possible to see the first smart sensor and wearable cases.

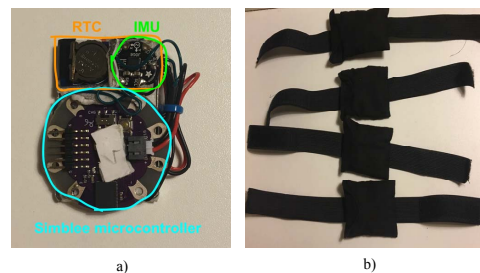


Fig. 3. System prototype: a) smart sensor prototypes; b) wearable sensor cases.

In Fig. 3a is showed the configuration of the smart sensor with the *LilyPad* board, real time clock module and the flora IMU. There's also a lithium battery in the bottom which doesn't appear in the image. In Fig. 3b there's the sensor cases which we're designed using elastic fabric and adhesive.

These systems we'll be composed by 5 of these smart sensors although in the first tests were only used 4, because we wanted to test if we can accurately measure the knee joint angle and prove that it will work as a metric for comparison between a healthy patient and one patient with gait problems.

## IV. PRELIMINARY RESULTS AND DISCUSSIONS

As this is a project under development these are preliminary results obtained in laboratory tests. The system isn't yet fully concluded as there is being developed EMG and ECG with E-Textile electrodes.

This test was driven by three volunteers: a male 23 years, a female 23 years and another female of 69 years. In this tests there possible to observe the knee joint angle during the execution off a train plan equal to the three volunteers. The train plan is created in the cloud backend app and its generated print version with a QRcode that gives patient access to the mobile app. In Fig. 4 there's a sample of the printed version of the plan used in these tests is also possible to read the QRcode which is mainly used for the patient as an easy way to login into the mobile application. This train plan is only a complement because all the train is guided by our Augmented Reality Serious Game where a virtual assistant (see Fig. 5) helps showing the movements that are in the train plan.

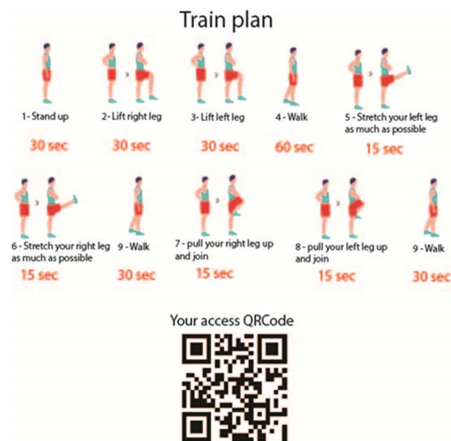


Fig. 4. Training plan and login QRCode.

During exercises patients are scoring points which are showed in Fig. 5 as a load bar which increases if the patient successfully executes the exercises. This provides visual feedback to the user; also in the game we'll implement in the final version sound alerts so patient get an instant feedback of their performance.



Fig. 5. Graphical User Interface – In-Game footage.

All three volunteers we're instructed to follow the plan in the Fig. 4 that consisted in movements of the lower limb for a period of defined time.

The results obtained were based on the four IMU modules, two for each lower limb, that were used to measure the left and right knee joint angle using the values acquired from a pair of IMU sensors:

$$Knee\ Joint\ Angle = roll1 + (180 - roll 2) \quad (1)$$

In Fig. 6, Fig. 7 and Fig. 8 are presented several results obtained during the training with three volunteers (volunteer 1: female, 23 years old; volunteer 2: male 23 years old; volunteer 3: female 69 years old).

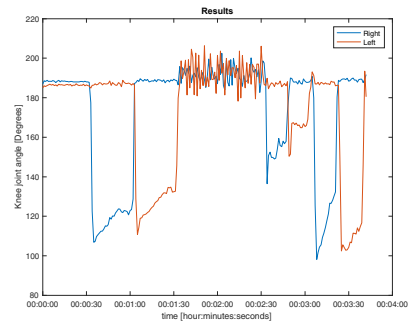


Fig. 6. Volunteer 1 training results.

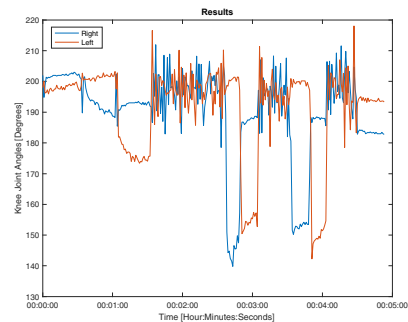


Fig. 7. Volunteer 2 training results.

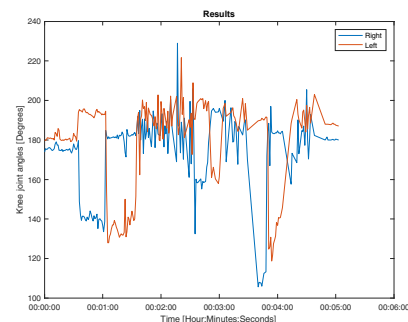


Fig. 8. Volunteer 3 training results.

The volunteers' motor conditions were different from volunteer to volunteer thus 69 years old female has the left knee affected and the 23 years old male has small equilibrium impairments. These conditions affected the results as expected showing clearly these two problems.

Observing the results presented in Fig. 6 it can be underlined the fact that in this case the volunteer showed a good equilibrium performing all exercises as well as appropriate knee flexion that conducted to a good score.

In Fig. 7 is presented the results for a volunteer that showed more difficulties maintaining the equilibrium but showed good knee flexion results scoring fewer points than the first staying in the middle of the qualification in the game.

Finally, in Fig. 8 shows that last volunteer, was the oldest one showed good equilibrium but less flexion obtaining the lowest score between the testers.

With these results it's possible to detect flexion and equilibrium during the execution of the exercises. The long term combination of these results will also show the progress obtained during the execution of the exercises by comparing historical training sessions with the new ones on the cloud backend app.

The volunteers also enjoyed to play the Augmented Reality Game considering a new way to make physical rehabilitation.

#### V. CONCLUSIONS

In this paper a first version of the system and preliminary tests regarding the usage of augmented reality in physical rehabilitation was presented. The obtained results and the volunteers' feedback about the AR serious game underline the capabilities of the system to extract objective information about physical rehabilitation. The system could be used as complementary tool for physical rehabilitation of lower limb motivating the users through the serious game but also through the feedback that it is provided during training sessions. Additional developments are being done including E-Textile electrodes for measurement of cardiac and muscular activity through ECG and EMG. As future work new tests will be carried out in a clinical environment with real patients.

#### ACKNOWLEDGMENT

This research is supported by Instituto de Telecomunicações (IT-IUL) at ISCTE-IUL, Lisbon, Portugal and Fundação para a Ciência e Tecnologia (FCT) project PTDC/DTP-DES/6776/2014. We also thanks to volunteers and physiotherapists that collaborate with us during the system conception and system validation.

#### REFERENCES

[1] P. Rego, P. M. Moreira, and L. P. Reis, "Serious Games for Rehabilitation A Survey and a Classification Towards a Taxonomy," 5th Iber. Conf. Inf. Syst. Technol., no. November 2015, pp. 1–6, 2010.

[2] C. Watters et al., "Extending the use of games in health care," Proc. Annu. Hawaii Int. Conf. Syst. Sci., vol. 5, no. C, pp. 1–8, 2006.

[3] C. Kirk, "Wii Revolution." [Online]. Available: [http://www.slate.com/articles/business/the\\_pivot/2012/10/nintendo\\_wii\\_the\\_console\\_that\\_beat\\_the\\_xbox\\_and\\_playstation\\_and\\_saved\\_nintendo.html](http://www.slate.com/articles/business/the_pivot/2012/10/nintendo_wii_the_console_that_beat_the_xbox_and_playstation_and_saved_nintendo.html). [Accessed: 29-Mar-2018].

[4] R. T. Azuma, "A Survey of Augmented Reality," vol. 4, no. August, pp. 355–385, 1997.

[5] V. Lampret, "A Survey of Augmented Reality Technologies, Applications and Limitations," Int. J. Virtual Real., vol. 9, no. 2, pp. 1–20, 2010.

[6] F. M. Dinis, A. S. Guimaraes, B. R. Carvalho, and J. P. P. Martins, "Virtual and augmented reality game-based applications to civil engineering education," 2017 IEEE Glob. Eng. Educ. Conf., no. April, pp. 1683–1688, 2017.

[7] M. K. Tageldeen, I. Elamvazuthi, N. Perumal, and T. Ganesan, "A virtual reality based serious games for rehabilitation of arm," 2017 IEEE 3rd Int. Symp. Robot. Manuf. Autom., pp. 1–6, 2017.

[8] W. Ying, "Augmented reality based upper limb rehabilitation system," 2017.

[9] O. Potolache, F. Lourenco, J. M. Dias Pereira, and P. S. Girao, "Serious game for physical rehabilitation: Measuring the effectiveness of virtual and real training environments," I2MTC 2017 - 2017 IEEE Int. Instrum. Meas. Technol. Conf. Proc., 2017.

[10] R. Oliveira, "Physical Rehabilitation based on Kinect Serious," pp. 0–5, 2017.

[11] O. Postolache, P. S. Giro, A. Lpez, F. J. Ferrero, J. M. D. Pereira, and G. Postolache, "Postural balance analysis using force platform for K-theragame users," 2016 IEEE Int. Symp. Med. Meas. Appl., pp. 1–6, 2016.

[12] V. Viegas, O. Postolache, J. M. D. Pereira, and P. M. S. Girao, "NUI therapeutic serious games with metrics validation based on wearable devices," Conf. Rec. - IEEE Instrum. Meas. Technol. Conf., vol. 2016–July, pp. 1–6, 2016.

[13] N. Duarte, O. Postolache, and J. Sharcanski, "KSGphysio – Kinect Serious Game for Physiotherapy," Int. Conf. Expo. Electr. Power Eng., no. Epe, pp. 16–18, 2014.

[14] R. N. Madeira, N. Correia, A. C. Dias, M. Guerra, O. Postolache, and G. Postolache, "Designing personalized therapeutic serious games for a pervasive assistive environment," 2011 IEEE 1st Int. Conf. Serious Games Appl. Heal. SeGAH 2011, 2011.

[15] E. Jovanov, A. Milenkovic, C. Otto, and P. C. De Groen, "A wireless body area network of intelligent motion sensors for computer assisted physical rehabilitation," J. Neuroeng. Rehabil., vol. 2, pp. 1–10, 2005.

[16] H. Alemdar and C. Ersoy, "Wireless sensor networks for healthcare: A survey," Comput. Networks, vol. 54, no. 15, pp. 2688–2710, 2010.

[17] M. Duff et al., "An adaptive mixed reality training system for stroke rehabilitation," IEEE Trans. Neural Syst. Rehabil. Eng., vol. 18, no. 5, pp. 531–541, 2010.

[18] P. Leite and O. Postolache, "Gait rehabilitation monitor," 2017 E-Health Bioeng. Conf. EHB 2017, pp. 438–441, 2017.

[19] RF Digital Corp., "RFD77101 DATASHEET v2.2," vol. 43, no. 0, pp. 0–6.

[20] P. Projects, "RFduino." pp. 1–18, 2013.

[21] S. Majumder et al., "Non-Contact Wearable Wireless ECG Systems for Long Term Monitoring," vol. 3333, no. c, pp. 1–18, 2018.

# Appendix B – User Manual



Departamento de Ciências e Tecnologias da Informação

## User Manual

PhysioAR –Smart sensing and Augmented Reality for Physical Rehabilitation

João Pedro Duarte Monge

Supervisor:

Dr. Octavian Postolache, Assistant Professor, ISCTE-IUL

October 2018



# Table of Contents

<b>Table of Contents</b> .....	<b>ii</b>
<b>List of figures</b> .....	<b>iii</b>
<b>1. Mobile Application</b> .....	<b>4</b>
1.1. Installation .....	4
1.2. User instructions.....	7
1.2.1. Login.....	7
1.2.2. Recover password.....	8
1.2.3. Connect Sensors .....	8
1.2.4. Play game.....	10
<b>2. Backend Application</b> .....	<b>11</b>
2.1. Access the application.....	11
2.2. Registration .....	11
2.3. Login.....	12
2.4. Patient Manager .....	13
2.5. Patient Registration.....	14
2.6. Train planner .....	14
2.7. Patient data analysis .....	15
<b>3. Smart Sensors</b> .....	<b>16</b>
3.1. Powering on .....	16
3.3. Charging the batteries .....	17
3.4. Placing the sensors .....	18

## List of figures

Figure 1 - Xcode open project.....	4
Figure 2 - XCode signing project.....	5
Figure 3 - Xcode build and install.....	5
Figure 4 - Open PhysioAR app .....	6
Figure 5 - Mobile App - Login .....	7
Figure 6 - Mobile App - QRCode Login .....	7
Figure 7 - Mobile App - Recover Password.....	8
Figure 8 - Mobile App - Bluetooth enable .....	8
Figure 9 - Mobile App - Start Game .....	9
Figure 10 - Mobile App - Countdown.....	9
Figure 11 - Insert device onto VR Googles .....	10
Figure 12 - Mobile App - Game .....	10
Figure 13 - Backend registration .....	11
Figure 14 - Backend Login .....	12
Figure 15 - Backend Main page .....	12
Figure 16 - Patient manager .....	13
Figure 17 - Patient main page.....	13
Figure 18 - Patient registration.....	14
Figure 19 - Patient train plan .....	14
Figure 20 - Train analysis.....	15
Figure 21 - Smart sensor powering on .....	16
Figure 22 - Smart sensor charging.....	17
Figure 23 - Sensor case.....	18
Figure 24 - Electrodes plug.....	18
Figure 25 - E-Textile EMG shinguard .....	19
Figure 26 - E-Textile ECG t-shirt .....	19

# 1. Mobile Application

This chapter contains the instructions for the mobile application including installation and use instructions.

## 1.1. Installation

This app wasn't launched in app store. It is required XCode and an Apple Developer account for installation.

The steps to perform an installation are the follow:

- First open the PhysioAR.xcodeproj project with Xcode (Figure 1):



Figure 1 - Xcode open project

- Sign the project with a developer account (Figure 2);

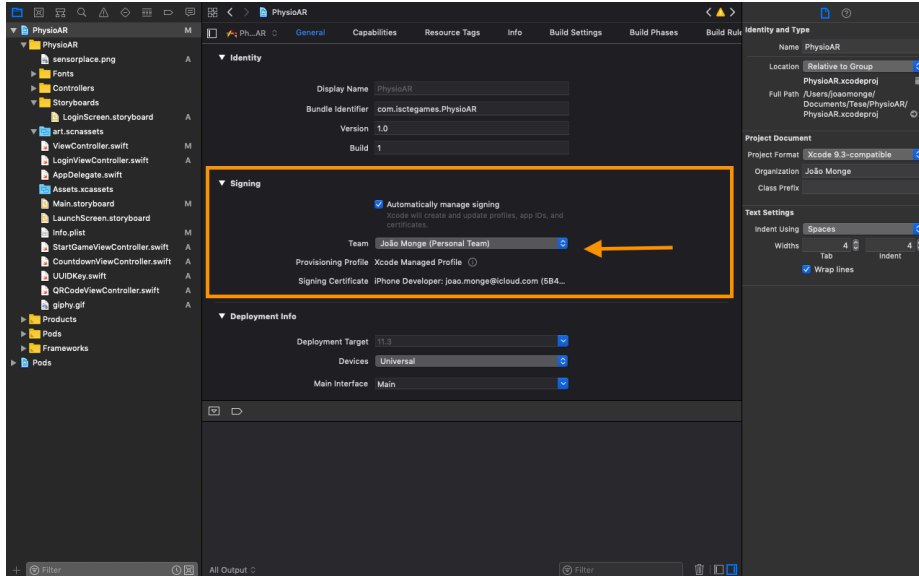


Figure 2 - XCode signing project

- Build and Install (Figure 3) **Note:** Only Iphone 6s or newer model are compatible;

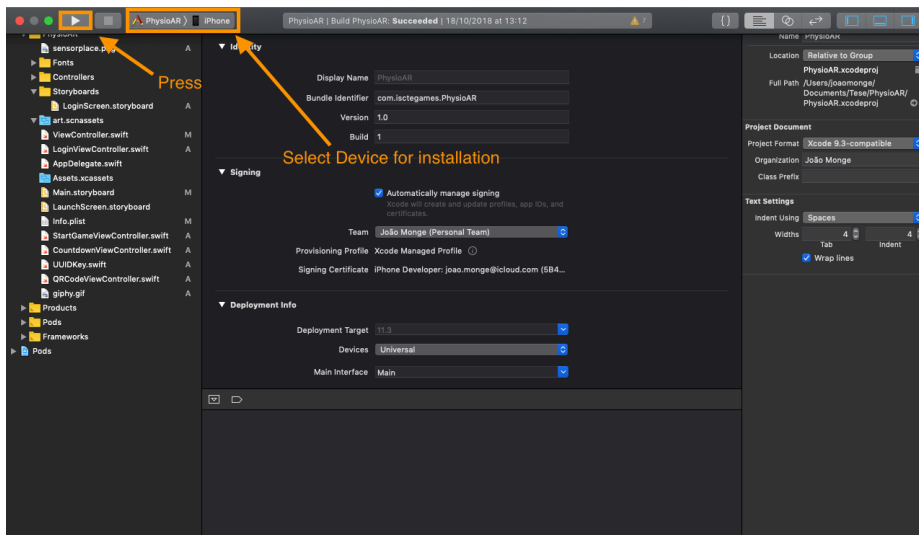


Figure 3 - Xcode build and install

- Perform the same steps for PhysioGame.xcodeproj;

After the two applications are installed successfully, the apps become available on the user device. To start the application the user should tap the PhysioAR application Icon (Figure 4).

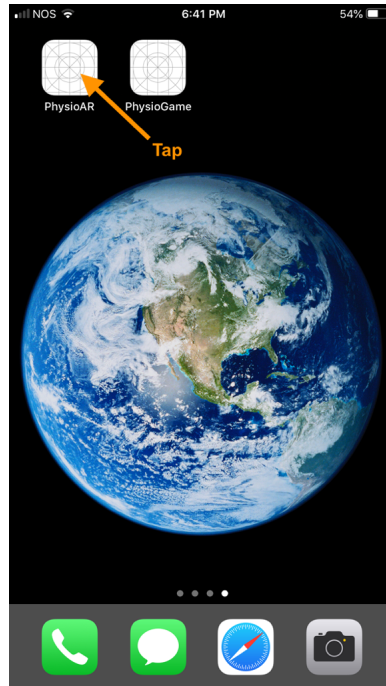


Figure 4 - Open PhysioAR app

## 1.2. User instructions

This chapter describes how to use the mobile application. The instructions presented in this chapter admit that the applications are already installed (Chapter 1.1).

### 1.2.1. Login

After opening PhysioAR Application is necessary to login. There are two options:

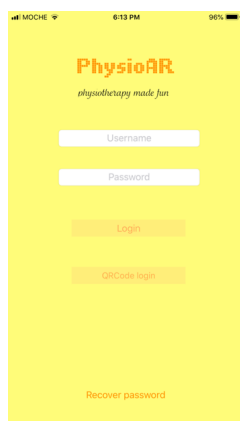


Figure 5 - Mobile App - Login

- One is to use a username/password combination and press login button (Figure 5);
- Press the QRCode Login button to login with the access QRCode (Figure 6);

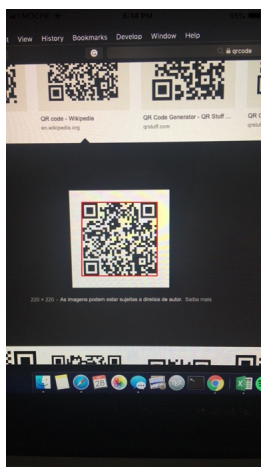


Figure 6 - Mobile App - QRCode Login

### 1.2.2. Recover password

- For password recovery press Recover Password button;

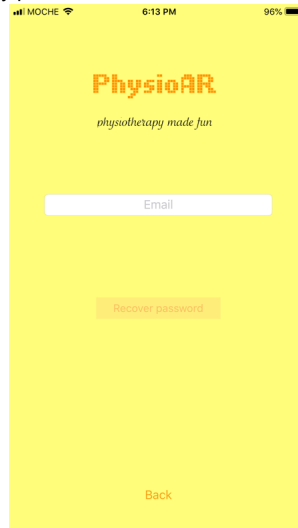


Figure 7 - Mobile App - Recover Password

### 1.2.3. Connect Sensors

**Note:** To connect the sensors enable bluetooth on device. Press the Bluetooth icon until its blue (Figure 8).

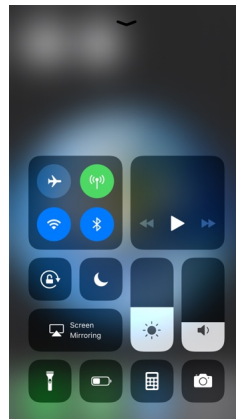


Figure 8 - Mobile App - Bluetooth enable

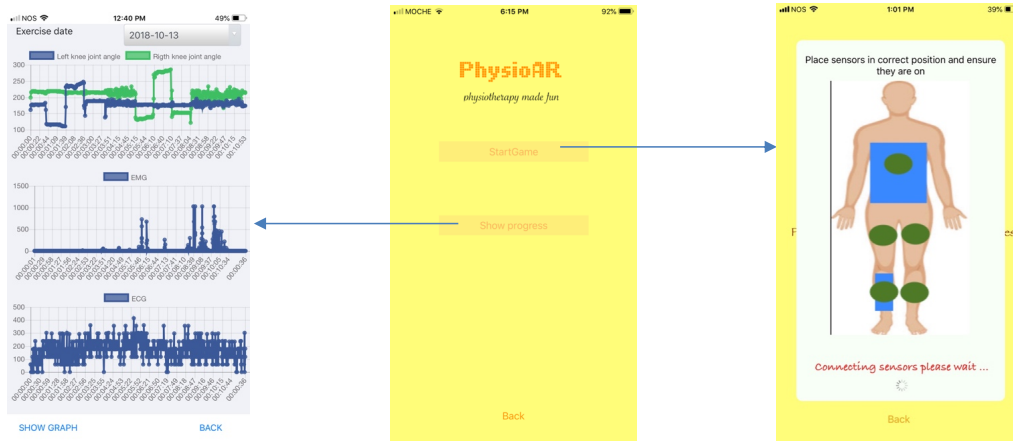


Figure 9 - Mobile App - Start Game

After the login the user as two options (middle screen Figure 9):

- Start Game button: If sensors are all powered on (see 3. Smart Sensors) and Bluetooth enable the game will start after a countdown (Figure 10);

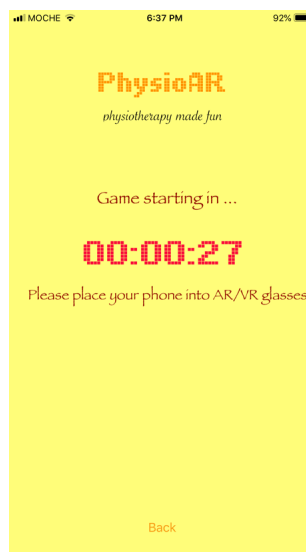


Figure 10 - Mobile App - Countdown

- Show Progress button: If user presses this button the results are showed (Left screen Figure 9);



### 1.2.4. Play game **Figure 1**

The game starts after the countdown ends, this countdown is for the user insert the device onto the VR Googles.



Figure 11 - Insert device onto VR Googles

Figure 11 describes how to place the device onto the VR Googles in two steps. First place on the case adapter and second place the adapter on the goggles.

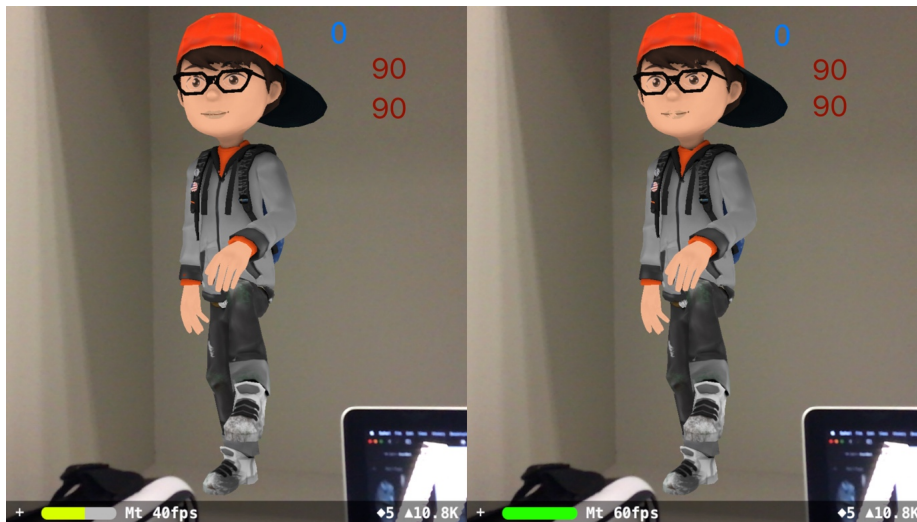


Figure 12 - Mobile App - Game

After the game finishes loading time user is able to see the screen represented in Figure 12. The blue number on the right up corner represents the points achieved and the red represent the current knee angle on both knees.

## 2. Backend Application

This chapter described the backend application intended for physiotherapist use.

### 2.1. Access the application

This application is web based and requires no installation. The user needs a compatible browser ( Safari, Chrome, Edge) and any device like a laptop, tablet or smartphone should work. To access the application, go to <https://www.physioar.tk> .

### 2.2. Registration

The first time a user/physiotherapist uses this system, he must perform the registering process. To do so on the main page the user should click on the sign up link and the page represented on Figure 13 appears.

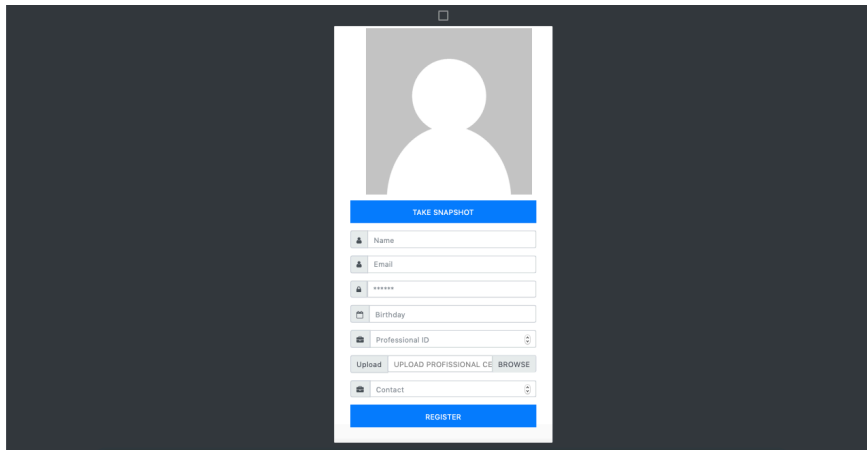


Figure 13 - Backend registration

The user should fill all the fields requested as they are required and take a photo or submit one. After filling the fields the user should press register and the process its complete if shows no errors and redirects to Login page.

## 2.3. Login

To perform a login the user should register in the application and then use the email/password to sign in (Figure 14).

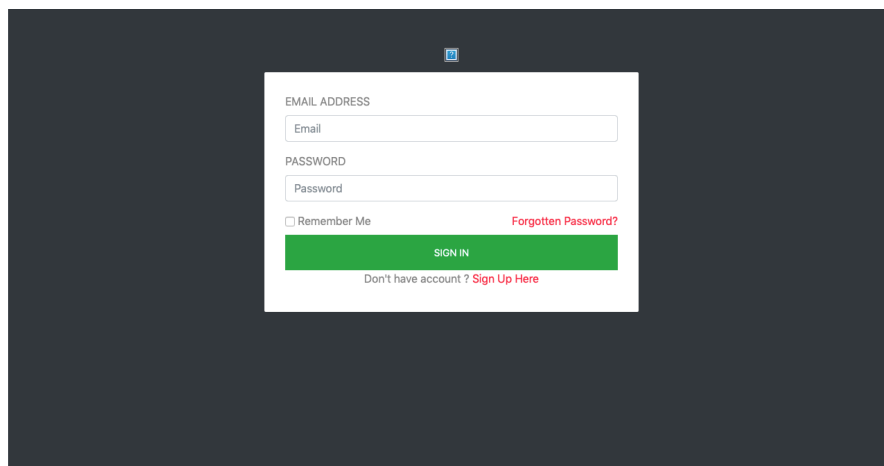


Figure 14 - Backend Login

If the login is successful, the page represented in Figure 15 appears. In it is possible to perform several tasks like access the patient manager and register pages.

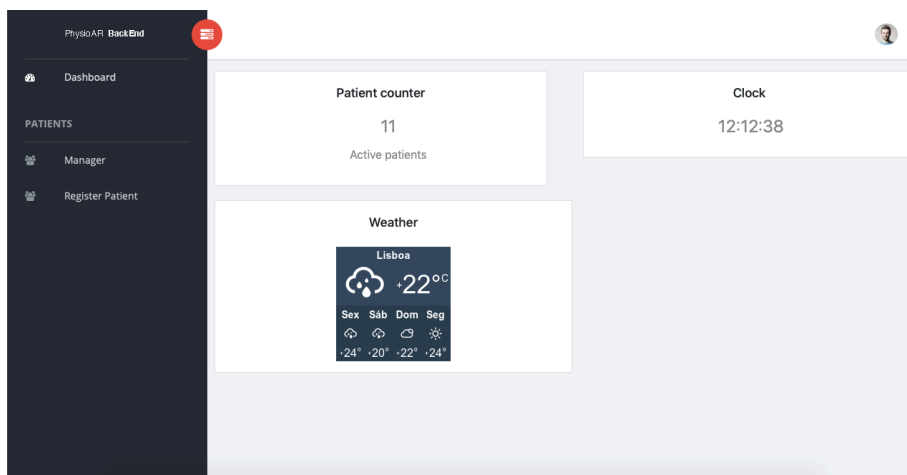


Figure 15 - Backend Main page

## 2.4. Patient Manager

In the page represented in Figure 16 is possible to see all the registered patients, and click on them to access their personal page.

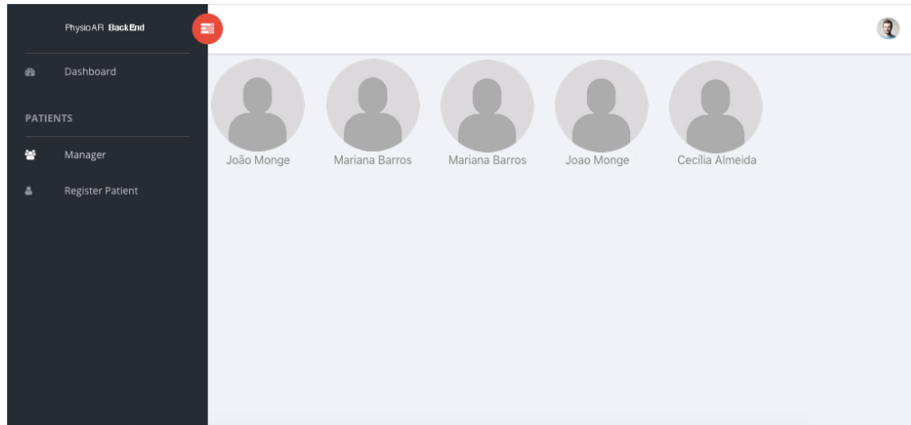


Figure 16 - Patient manager

If the user clicks on a particular patient his personal page should show up. In this page the user can access the train planner and the data analysis pages by clicking on the buttons on the left menu.

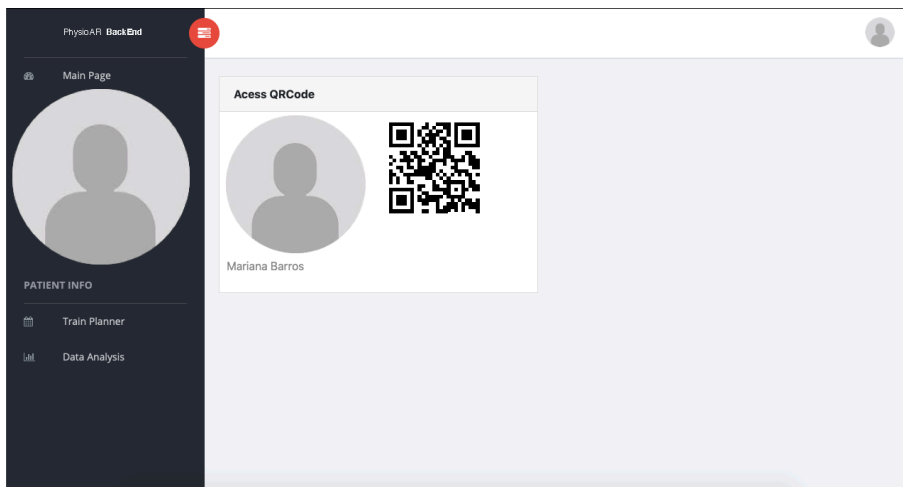


Figure 17 - Patient main page

## 2.5. Patient Registration

To register a patient the user should fill the fields represented in Figure 18 and press the button register. If the registration is successful, the registered patient appears in the manager page.

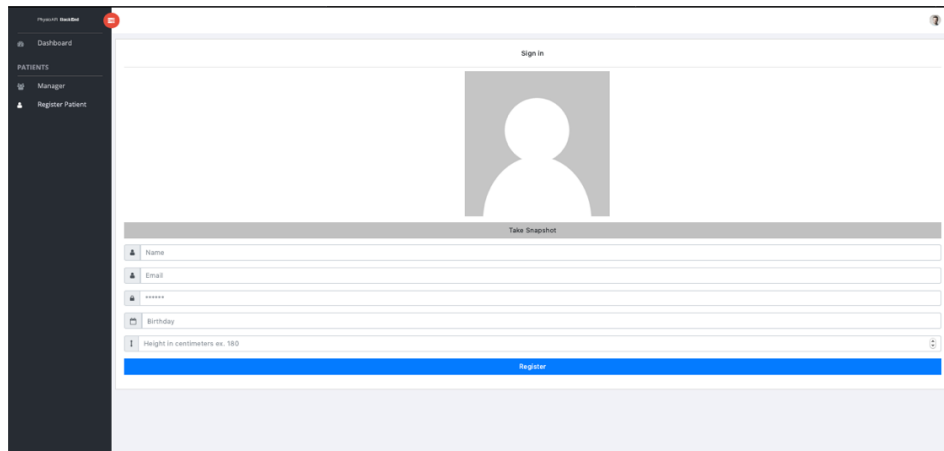


Figure 18 - Patient registration

## 2.6. Train planner

To create a new train plan the user should drag the desired exercises to the grey rectangle represented in Figure 19 and attribute a duration and a threshold (represents the angle that the patient should reach



Figure 19 - Patient train plan

## 2.7. Patient data analysis

To access the patient data analysis page represented in Figure 20 there is a button on patient personal page.

To select the pretended train the user should select the day and press analyze and the corresponding results appear. To interact with the graphs pass the cursor above the desired point.



Figure 20 - Train analysis

## 3. Smart Sensors

This chapter describes the user instructions for the smart sensors.

### 3.1. Powering on

To power on the sensors toggle the switch to on a led should light up (Figure 21). **Note:** If the sensor doesn't turn on make sure it has battery.

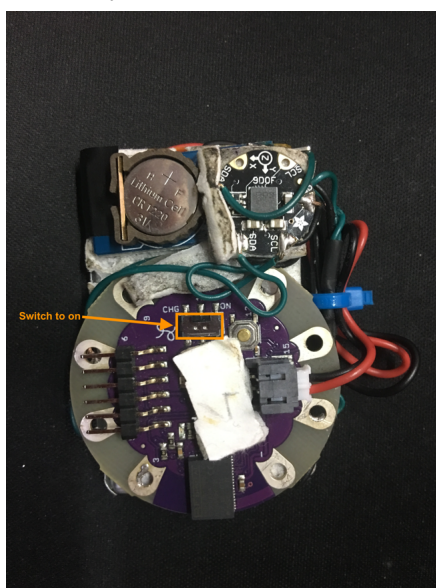


Figure 21 - Smart sensor powering on

### 3.3. Charging the batteries

To charge the smart sensors (Figure 22):

1. Switch to charge mode;
2. Connect the FTDI interface;
3. A red led lights up indicating that the sensor is charging;
4. The led turns off when fully charged;

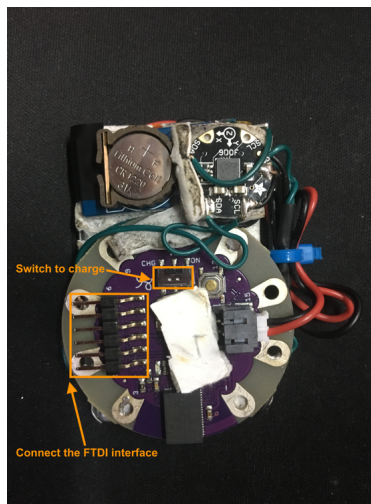


Figure 22 - Smart sensor charging



### 3.4. Placing the sensors

Place the sensors in the case as represented in Figure 23.



Figure 23 - Sensor case

The sensors should be distributed in the following order (all sensors have an id number):

- Sensor 1 on right thigh;
- Sensor 2 on left thigh;
- Sensor 3 on right shin with E-textile shinguard (Figure 25) plugged on 2.5mm jack present in the smart sensor (Figure 24);

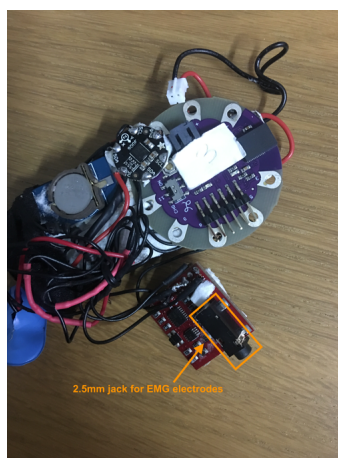


Figure 24 - Electrodes plug



Figure 25 - E-Textile EMG shinguard

- Sensor 4 on left shin;
- Sensor 5 on torso with E-textile t-shirt ( ) plugged on 2.5mm jack present in smart sensor;



Figure 26 - E-Textile ECG t-shirt

## **Appendix C – Technical Manual**



Departamento de Ciências e Tecnologias da Informação

## Technical Manual

PhysioAR –Smart sensing and Augmented Reality for Physical Rehabilitation

João Pedro Duarte Monge

Supervisor:

Dr. Octavian Postolache, Assistant Professor, ISCTE-IUL

October 2018

## Table of Contents

<i>Table of Contents</i> .....	<b>2</b>
<b>1. Database</b> .....	<b>4</b>
1.1. DB creation script .....	4
1.2. Tables description .....	11
<b>2. REST API</b> .....	<b>12</b>
2.1. API methods description .....	12
2.2. REST API .....	21
<b>3. Server</b> .....	<b>21</b>
<b>4. Web Application</b> .....	<b>22</b>
4.1. Code description .....	22
4.2. Build and deploy .....	23
<b>5. Mobile Application</b> .....	<b>23</b>
5.1. PhysioAR .....	24
5.2. PhysioGame .....	25
<b>6. Smart Sensors</b> .....	<b>38</b>
6.1. Code description .....	38

# Figures

Figure 5.1 - Mobile folder .....23

Figure 5.2 - Mobile apps .....24

Figure 5.3 - PhysioAR mobile code location .....24

Figure 5.4 - PhysioGame Source folder.....25

## 1. Database

The database of this project was build using MySQL. The creation script is on sub-chapter 1.1. and the tables description is on 1.2.

### 1.1. DB creation script

To create a database compatible with this system run the following script with MySQL:

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `PhysioAR`
--

-----

--
-- Stand-in structure for view `Acceleration_Calves`
--
CREATE TABLE `Acceleration_Calves` (
  `accel_calves` int(20)
, `occurrences` bigint(21)
, `trainID` int(20)
, `sensorID` varchar(20)
);

-----

--
-- Stand-in structure for view `AVG_IMU_ANGLES`
--
CREATE TABLE `AVG_IMU_ANGLES` (
  `roll_avg` decimal(23,4)
, `pitch_avg` decimal(23,4)
, `yaw_avg` decimal(23,4)
, `time` time
, `sensorID` varchar(20)
, `trainID` int(20)
);

-----
```

```
--  
-- Table structure for table `Exercise`  
--  
  
CREATE TABLE `Exercise` (  
  `Id` int(11) NOT NULL,  
  `Name` varchar(40) NOT NULL,  
  `Description` varchar(40) NOT NULL,  
  `AppInfoLink` varchar(40) NOT NULL,  
  `VideoLink` varchar(40) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
-----  
  
--  
-- Table structure for table `exerciseLink`  
--  
  
CREATE TABLE `exerciseLink` (  
  `id` int(11) NOT NULL,  
  `duration` int(11) NOT NULL,  
  `exerciseID` int(11) NOT NULL,  
  `planID` int(11) NOT NULL,  
  `angle` int(11) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
-----  
  
--  
-- Stand-in structure for view `KneeJointAngles`  
--  
CREATE TABLE `KneeJointAngles` (  
  `kneejoint_Angle_Rigth` decimal(25,4)  
  ,`kneejoint_Angle_Left` decimal(25,4)  
  ,`time` time  
  ,`trainID` int(20)  
  );  
  
-----  
  
--  
-- Stand-in structure for view `Knee Joint Left`  
--  
CREATE TABLE `Knee Joint Left` (  
  `kneejoint_angle_Left` decimal(25,4)  
  ,`time` time  
  ,`trainID` int(20)  
  );  
  
-----
```



```
--  
-- Stand-in structure for view `Knee Joint Righth`  
--  
CREATE TABLE `Knee Joint Righth` (  
  `kneejoint_angle_Righth` decimal(25,4)  
  , `time` time  
  , `trainID` int(20)  
);  
  
-----  
  
--  
-- Table structure for table `plan`  
--  
CREATE TABLE `plan` (  
  `id` int(11) NOT NULL,  
  `userID` int(11) NOT NULL,  
  `container` longtext CHARACTER SET utf8 COLLATE utf8_bin  
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
-----  
  
--  
-- Table structure for table `professional`  
--  
CREATE TABLE `professional` (  
  `id` int(11) NOT NULL,  
  `Name` varchar(40) NOT NULL,  
  `ProfessionalID` int(10) NOT NULL,  
  `Photo` varchar(40) NOT NULL,  
  `Contact` int(40) NOT NULL,  
  `pass` varchar(40) NOT NULL,  
  `email` varchar(40) NOT NULL,  
  `birthday` date DEFAULT NULL  
  ) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
-----  
  
--  
-- Table structure for table `sensorData`  
--  
CREATE TABLE `sensorData` (  
  `id` int(20) NOT NULL,  
  `roll` int(20) DEFAULT NULL,  
  `pitch` int(20) DEFAULT NULL,  
  `yaw` int(20) DEFAULT NULL,  
  `time` time NOT NULL,
```

```
`emg` int(20) DEFAULT NULL,  
`bpm` int(20) DEFAULT NULL,  
`ecg` int(20) DEFAULT NULL,  
`sensorID` varchar(20) NOT NULL,  
`trainID` int(20) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
-----  
  
--  
-- Table structure for table `train`  
--  
  
CREATE TABLE `train` (  
  `id` int(20) NOT NULL,  
  `date` date NOT NULL,  
  `Description` varchar(200) DEFAULT NULL,  
  `userID` int(20) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
-----  
  
--  
-- Table structure for table `user`  
--  
  
CREATE TABLE `user` (  
  `Id` int(4) NOT NULL,  
  `name` varchar(50) NOT NULL,  
  `pass` varchar(50) NOT NULL,  
  `birthday` date DEFAULT NULL,  
  `email` varchar(50) NOT NULL,  
  `faceimage` varchar(40) NOT NULL,  
  `associatedProfessional` int(11) NOT NULL,  
  `qrcode` varchar(20) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=latin1;  
  
-----  
  
--  
-- Structure for view `Acceleration_Calves`  
--  
DROP TABLE IF EXISTS `Acceleration_Calves`;  
  
CREATE ALGORITHM=TEMPTABLE DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW  
`Acceleration_Calves` AS select `sensorData`.`pitch` AS  
`accel_calves`,count(`sensorData`.`pitch`) AS `occurences`,`sensorData`.`trainID`  
AS `trainID`,`sensorData`.`sensorID` AS `sensorID` from `sensorData` where  
((`sensorData`.`sensorID` = 'sensor3') or (`sensorData`.`sensorID` = 'sensor4'))  
group by `sensorData`.`pitch`,`sensorData`.`trainID`,`sensorData`.`sensorID` ;
```

```
-----  
  
--  
-- Structure for view `AVG_IMU_ANGLES`  
--  
DROP TABLE IF EXISTS `AVG_IMU_ANGLES`;  
  
CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW  
`AVG_IMU_ANGLES` AS select avg(`sensorData`.`roll`) AS  
`roll_avg`,avg(`sensorData`.`pitch`) AS `pitch_avg`,avg(`sensorData`.`yaw`) AS  
`yaw_avg`,`sensorData`.`time` AS `time`,`sensorData`.`sensorID` AS  
`sensorID`,`sensorData`.`trainID` AS `trainID` from `sensorData` group by  
`sensorData`.`time`,`sensorData`.`sensorID`,`sensorData`.`trainID` ;  
  
-----  
  
--  
-- Structure for view `KneeJointAngles`  
--  
DROP TABLE IF EXISTS `KneeJointAngles`;  
  
CREATE ALGORITHM=TEMPORARY DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW  
`KneeJointAngles` AS select `kr`.`kneejoint_angle_Rigth` AS  
`kneejoint_Angle_Rigth`,`kl`.`kneejoint_angle_Left` AS  
`kneejoint_Angle_Left`,`kl`.`time` AS `time`,`kl`.`trainID` AS `trainID` from  
(`Knee Joint Left` `kl` left join `Knee Joint Rigth` `kr` on((`kr`.`time` =  
`kl`.`time`))) where (`kr`.`trainID` = `kl`.`trainID`);  
  
-----  
  
--  
-- Structure for view `Knee Joint Left`  
--  
DROP TABLE IF EXISTS `Knee Joint Left`;  
  
CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW  
`Knee Joint Left` AS select (`t1`.`roll_avg` + (180 - `t2`.`roll_avg`)) AS  
`kneejoint_angle_Left`,`t1`.`time` AS `time`,`t1`.`trainID` AS `trainID` from  
(`AVG_IMU_ANGLES` `t1` left join `AVG_IMU_ANGLES` `t2` on((`t1`.`time` =  
`t2`.`time`))) where ((`t1`.`sensorID` = `sensor2`) and (`t2`.`sensorID` =  
`sensor4`) and (`t1`.`trainID` = `t2`.`trainID`));  
  
-----  
  
--  
-- Structure for view `Knee Joint Rigth`  
--  
DROP TABLE IF EXISTS `Knee Joint Rigth`;  
  
CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL SECURITY DEFINER VIEW  
`Knee Joint Rigth` AS select (`t1`.`roll_avg` + (180 - `t2`.`roll_avg`)) AS
```

```
`kneejoint_angle_Rigth`, `t1`.`time` AS `time`, `t1`.`trainID` AS `trainID` from
(`AVG_IMU_ANGLES` `t1` left join `AVG_IMU_ANGLES` `t2` on((`t1`.`time` =
`t2`.`time`))) where ((`t1`.`sensorID` = 'sensor1') and (`t2`.`sensorID` =
'sensor3') and (`t1`.`trainID` = `t2`.`trainID`));

--
-- Indexes for dumped tables
--

--
-- Indexes for table `Exercise`
--
ALTER TABLE `Exercise`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `exerciseLink`
--
ALTER TABLE `exerciseLink`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `plan`
--
ALTER TABLE `plan`
  ADD PRIMARY KEY (`id`),
  ADD UNIQUE KEY `userID` (`userID`);

--
-- Indexes for table `professional`
--
ALTER TABLE `professional`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `sensorData`
--
ALTER TABLE `sensorData`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `train`
--
ALTER TABLE `train`
  ADD PRIMARY KEY (`id`);

--
-- Indexes for table `user`
--
ALTER TABLE `user`
  ADD PRIMARY KEY (`id`),
```

```
ADD KEY `associatedProfessional` (`associatedProfessional`) USING BTREE;

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `Exercise`
--
ALTER TABLE `Exercise`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=12;
--
-- AUTO_INCREMENT for table `exerciseLink`
--
ALTER TABLE `exerciseLink`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=548;
--
-- AUTO_INCREMENT for table `plan`
--
ALTER TABLE `plan`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=24;
--
-- AUTO_INCREMENT for table `professional`
--
ALTER TABLE `professional`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=11;
--
-- AUTO_INCREMENT for table `sensorData`
--
ALTER TABLE `sensorData`
  MODIFY `id` int(20) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=354600;
--
-- AUTO_INCREMENT for table `train`
--
ALTER TABLE `train`
  MODIFY `id` int(20) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=41;
--
-- AUTO_INCREMENT for table `user`
--
ALTER TABLE `user`
  MODIFY `id` int(4) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=36;
--
-- Constraints for dumped tables
--

--
-- Constraints for table `user`
--
ALTER TABLE `user`
  ADD CONSTRAINT `user_ibfk_1` FOREIGN KEY (`associatedProfessional`) REFERENCES
`professional` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

```
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;  
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;  
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

## 1.2. Tables description

### Tables:

- **Exercise**
  - Description: List of exercises available. To add new exercises manually insert with sql command. This exercises appear in the physiotherapist train planner page.
  - Relations: This table relates to exerciseLink table as a foreign key.
- **ExerciseLink**
  - Description: This table links the exercise with a user and add a duration and threshold to it. The data in this table is originated when a physiotherapist assigns an exercise to a user.
  - Relations: Relates to plan table.
- **Plan**
  - Description: The plan table is where every user created plan is stored.
  - Relations: The plan relates to user and professional tables.
- **Professional**
  - Description: This is where the physiotherapist user information is stored
  - Relations: Relates to plan and user.
- **sensorData**
  - Description: This table is where the data coming from the sensors is stored.
  - Relations: This relates to a train.
- **Train**
  - Description: The train is responsible for storing data relative to a train that a user executed in the mobile app.
  - Relations: Relates to sensorData and to plan.
- **User**
  - Description: This table is where the information of the user is stored.
  - Relations: Relates to professional, plan.

### Views:

- **KneeJointAngles:** This view combines data from different sensors and computes the knee angle.

## 2. REST API

The apps interaction with the database is done using a REST API. In chapter 2.1. the REST API Script description and in 2.2. the initialization procedures.

This REST API is written in PHP and is accessed through HTTP Post method.

**IMPORTANT NOTE:** The current address for this application is <https://www.Physioar.tk/> if this address is changed is necessary to change the code on the mobile application to accommodate the new address in the requests of this API.

### 2.1. API methods description

Database configuration script located in **PhysioAR/Server/backend/config/db.php**: This is the script where the database credentials and location should be inserted. To create a database consult chapter 1.

```
<?php
/**
 * Configuration for: Database Connection
 *
 * DB_HOST: database host, usually it's "127.0.0.1" or "localhost", some servers
 also need port info
 * DB_NAME: name of the database. please note: database and database table are not
 the same thing
 * DB_USER: user for your database. the user needs to have rights for SELECT,
 UPDATE, DELETE and INSERT.
 * DB_PASS: the password of the above user
 */
define("DB_HOST", "13.58.239.160");
define("DB_NAME", "PhysioAR");
define("DB_USER", "root");
define("DB_PASS", "*****");
#####
```

Login Application API located in **PhysioAR/Server/backend/loginapp.php**: This is the API used by the PhysioAR mobile application to check the correct login of a user.

```
<?php
require_once("config/db.php");

session_start();
if(isset($_POST["email"]) && isset($_POST["password"])){

    $email = trim($_POST["email"]);
    $password = trim($_POST["password"]);
```

```
$db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);

// check if user or email address already exists
$sql = "SELECT * FROM user WHERE email= '$email.'";
$result = mysqli_query($db_connection,$sql);

if(mysqli_num_rows($result)==1){
    $row=mysqli_fetch_array($result);

    if(trim($row["pass"]) == $password){
        $_SESSION["UserID"] = $row["Id"];
        $_SESSION["id"] = $row["associatedProfessional"];

        echo "login accepted";

    }
    else{
        echo "Invalid password";
    }
}
else{
    echo "User doesnt exists!";
}
}
else{
    echo "invalid post data";
}

#####
QRCode login API located in PhysioAR/Server/backend/loginqrcode.php: This API is
responsible to verify if the qrcode scanned on the application matches a registered
user on the database.

<?php
require_once("config/db.php");

session_start();

if(isset($_POST["qrcode"])){
    $qrcode = trim($_POST["qrcode"]);
    $qrcode = rtrim($qrcode,"\n");
    $qrcode = rtrim($qrcode,"\" ");
    $qrcode = trim($qrcode);

    $db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);
```



```
// check if user or email address already exists
$sql = "SELECT * FROM user WHERE qrcode = " . $qrcode. " ";
$result = mysqli_query($db_connection, $sql);

if(mysqli_num_rows($result)==1){
    $row = mysqli_fetch_array($result);
    $_SESSION['UserID'] = $row['Id'];
    $_SESSION['id'] = $row['associatedProfessional'];
    echo "loggin accepted,userid=".$_SESSION['UserID']."";
}
else{
    echo "loggin refused";
}
}
else{
    echo "invalid post data";
}
#####
Reset plan located in PhysioAR/Server/backend/resetPlan.php:This API is used for
resetting a train plan through the backend application.

<?php

require_once("config/db.php");
session_start();

$db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);
$sql = "SELECT * FROM plan WHERE userID = '".$_SESSION['userID']."'";
$result = mysqli_query($db_connection, $sql);
$planId;
if(mysqli_num_rows($result)==1){
    $row = mysqli_fetch_array($result);
    $planId = $row['id'];
    $sql2 = "DELETE FROM `exerciseLink` WHERE planID = '".$planId.'";";
    $result2 = mysqli_query($db_connection, $sql2);
    $sql6 = "UPDATE `plan` SET `container`='';";
    $result6 = mysqli_query($db_connection, $sql6);
}

?>
#####
Update plan API located in PhysioAR/Server/backend/updatePlan.php: This API is
responsible for both creation and update of the train plan.

<?php
require_once("config/db.php");
session_start();

if(isset($_POST['listExercises']) && isset($_SESSION['id'])){
    $db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);
```

```

$sql = "SELECT * FROM plan WHERE userID = '".$_SESSION['userID']."'";
$result = mysqli_query($db_connection,$sql);
$planId;
if(mysqli_num_rows($result)==1){
    $row = mysqli_fetch_array($result);
    $planId = $row['id'];
    $sql2 = "DELETE FROM `exerciseLink` WHERE planID = '".$_SESSION['planID']."'";
    $result2 = mysqli_query($db_connection,$sql2);
}
else{
    $sql1 = "INSERT INTO `plan`(`userID`) VALUES ('".$_SESSION['userID']."'");
    $result1 = mysqli_query($db_connection,$sql1);
    $row = mysqli_fetch_array($result1);
    $planId = $row['id'];
}
foreach($_POST['listExercises'] as $plannedExercise){
    $id = $plannedExercise['id'];
    $dur = $plannedExercise['dur'];
    $ang = $plannedExercise['angle'];

    $sql3 = "SELECT Id FROM Exercise WHERE Name LIKE '".$_SESSION['id']."'";
    $result3 = mysqli_query($db_connection,$sql3);
    $row = mysqli_fetch_array($result3);

    $sql4 = "INSERT INTO
`exerciseLink`(`duration`,`angle`,`exerciseID`,`planID`) VALUES
('".$dur."','".$ang."','".$row['Id']."','".$planId."'");
    $result4 = mysqli_query($db_connection,$sql4);

}
}

if(isset($_POST['str'])){
    echo "responding";
    $db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);
    $sql5 = "SELECT * FROM plan WHERE userID = '".$_SESSION['userID']."'";
    $result5 = mysqli_query($db_connection,$sql5);
    $row = mysqli_fetch_array($result5);
    $planId = $row['id'];

    $sql6 = "UPDATE `plan` SET `container`='".$_SESSION['str']."' WHERE id =
".$_SESSION['planID']."'";
    $result6 = mysqli_query($db_connection,$sql6);
    if ( false=== $result6 ) {
        echo mysqli_error($db_connection);
    }
}
}

```

```
?>
#####

Get points API located in PhysioAR/Server/backend/getPoints.php: This API method
returns the current points of the user.

<?php
require_once("config/db.php");
session_start();

if(isset($_POST['UserID']) && isset($_POST['start']) && isset($_POST['end']) &&
isset($_POST['threshold'])){
    $db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);
    $sql = "SELECT id FROM train WHERE userID = '".$_POST['UserID']."' AND date =
CURRENT_DATE()";
    $result = mysqli_query($db_connection,$sql);
    if(mysqli_num_rows($result)==1){

        $row = mysqli_fetch_array($result);
        $trainId = $row['id'];

        $sql2 = "SELECT count(*) as multiplier FROM KneeJointAngles WHERE trainID =
'".$trainId."' AND (time >= '".$_POST['start']."' AND time < '".$_POST['end']."' )
AND kneejoint_Angle_Rigth > '".isset($_POST['threshold'])."'";
        $result1 = mysqli_query($db_connection,$sql2);
        if(mysqli_num_rows($result1)==1){

            $row = mysqli_fetch_assoc($result1);
            echo $row['multiplier'];
        }else{
            echo 0;
        }
    }
    else{
        echo 0;
    }
}

?>
#####

Get exercises API located in PhysioAR/Server/backend/views/getExercises.php: This
API method returns the exercises of a determinate user.

<?php
require_once("../config/db.php");
session_start();

$exercisesString = "";
if(isset($_POST['userid'])){
```

```

        $_SESSION['UserID'] = $_POST['userid'];
    }
    if(isset($_SESSION['UserID'])){
        $db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);
        // get plan id
        $sql = "SELECT id FROM plan WHERE userID='".$_SESSION['UserID']."'";
        $result = mysqli_query($db_connection,$sql) or die;
        $row = mysqli_fetch_array($result);
        $planId= $row['id'];

        //Getting exercises ordered by id
        $sql1 = "SELECT * FROM exerciseLink WHERE planID='".$planId."'";
        $result2 = mysqli_query($db_connection,$sql1) or die;

        while ($row = mysqli_fetch_array($result2)){
            $returned = "(";
            $returned =
$returned.$row['exerciseID'].",".$row['duration'].",".$row['angle'].")";
            $exercisesString = $exercisesString.$returned;
        }

        $exercisesString=$exercisesString."";
        echo $exercisesString;
    }

?>
#####

EMG data API located in PhysioAR/Server/backend/emgData.php: This API method
returns the user EMG data from a train session.

<?php
header('Content-Type: application/json');
require_once("../config/db.php");

session_start();

if( isset($_SESSION["id"]) && isset($_GET["trainID"])){
    $db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);

    $sql_sensorLeft = "SELECT time,yaw as emg FROM `sensorData` WHERE
sensorId='sensor3' and trainID =".$_GET["trainID"].'";

    $result = mysqli_query($db_connection,$sql_sensorLeft);

    $data = array();

```

```
        foreach ($result as $row) {
            $data[] = $row;
        }

        //free memory associated with result
        $result->close();

        //close connection
        $db_connection->close();

        //now print the data
        echo json_encode($data);
    }
#####

ECG data API located in PhysioAR/Server/backend/ecgData.php: This API method
returns the user ECG data from a train session.

<?php
header('Content-Type: application/json');
require_once("../config/db.php");

session_start();

if( isset($_SESSION["id"]) && isset($_GET["trainID"])){
    $db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);

    $sql_sensorLeft = "SELECT time,yaw as ecg FROM `sensorData` WHERE
sensorId='sensor5' and trainID = '".$_GET["trainID"]."'";

    $result = mysqli_query($db_connection,$sql_sensorLeft);

    $data = array();
    foreach ($result as $row) {
        $data[] = $row;
    }

    //free memory associated with result
    $result->close();

    //close connection
    $db_connection->close();

    //now print the data
    echo json_encode($data);
}
```

```
#####
```

Knee joint data API located in **PhysioAR/Server/backend/kneejointdata.php**: This API method returns the user knee joint data from a train session.

```
<?php
header('Content-Type: application/json');
require_once("../config/db.php");

session_start();

if( isset($_SESSION["id"]) && isset($_GET["trainID"])){
    $db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);

    $sql_sensorLeft = "SELECT * FROM `KneeJointAngles` WHERE trainID
    =".$_GET["trainID"]." ORDER BY `KneeJointAngles`.`time` ASC;";

    $result = mysqli_query($db_connection,$sql_sensorLeft);

    $data = array();
    foreach ($result as $row) {
        $data[] = $row;
    }

    //free memory associated with result
    $result->close();

    //close connection
    $db_connection->close();

    //now print the data
    echo json_encode($data);
}
```

```
#####
```

Register Patient API located in **PhysioAR/Server/backend/register\_patient.php**: This API method is for user/patient registration.

```
<?php
require_once("/var/www/html/backend/config/db.php");
session_start();
```

```
if(isset($_POST['face_image'])){
$_SESSION['face_image'] = $_POST['face_image'];
echo "image received";
}
if(isset($_POST['name']) && isset($_POST['email']) && isset($_POST['password']) &&
isset($_POST['birthday']) && isset($_POST['height']) &&
isset($_SESSION['face_image']) ){

$data = $_SESSION['face_image'];

list($type, $data) = explode(':', $data);
list($, $data) = explode('.', $data);
$data = base64_decode($data);
$random = rand(10000000,99999999);

$qrcode = rand(10000000,99999999);

$image_path = '/var/www/html/backend/userimages/'.$random.'.jpg';
file_put_contents($image_path, $data);

$db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);
$uploadDate = date("Y-m-d", strtotime($_POST['birthday']));

$sql1 = "INSERT INTO `user`(`name`, `pass`, `birthday`, `email`, `faceimage`,
`associatedProfessional`, `qrcode`) VALUES
('".$_POST['name']."','".$_POST['password']."','".$uploadDate."','".$_POST['email']
."','".$random."','".$_SESSION['id']."','".$
".qrcode."');";
$result1 = mysqli_query($db_connection,$sql1) or die;
header("Location: https://www.physioar.tk/backend/views/usermanager.php"); /*
Redirect browser */
exit();
}else{
echo "error";
}

?>
#####

Register Professional API located in
PhysioAR/Server/backend/register_professional.php: This API method is for
user/physiotherapist registration.

<?php
require_once("/var/www/html/backend/config/db.php");
session_start();

if(isset($_POST['face_image'])){
$_SESSION['face_image'] = $_POST['face_image'];
```

```
echo "image received";
}
if(isset($_POST['name']) && isset($_POST['email']) && isset($_POST['password']) &&
isset($_POST['birthday']) && isset($_POST['professionalID']) &&
isset($_POST['contact']) && isset($_SESSION['face_image']) ){

    $data = $_SESSION['face_image'];

    list($type, $data) = explode('.', $data);
    list($data) = explode('/', $data);
    $data = base64_decode($data);
    $random = rand(10000000,99999999);
    $image_path = '/var/www/html/backend/professionalphotos/'.$random.'.jpg';
    file_put_contents($image_path, $data);

    $db_connection = new mysqli(DB_HOST, DB_USER, DB_PASS, DB_NAME);
    $uploadDate = date("Y-m-d", strtotime($_POST['birthday']));

    $sql1 = "INSERT INTO `professional` (`Name`, `ProfessionalID`, `Photo`, `Contact`,
`pass`, `email`, `birthday`) VALUES ('".$_POST['name']."',
".$_POST['professionalID']."', '".$_$random."', '".$_POST['contact']."',
".$_POST['password']."', '".$_POST['email']."', '".$_uploadDate."');";
    $result1 = mysqli_query($db_connection,$sql1) or die;
    header("Location: https://www.physioar.tk/backend"); /* Redirect browser */
    exit();
}else{
    header("Location: https://www.physioar.tk/backend/views/page-
register_error.html");
    exit();
}

?>
```

## 2.2. REST API

The REST API is placed in the backend folder and shares the same folder as the backend application. To access the methods use HTTP post request with the necessary parameters.

## 3. Server

The server used for this project was Ubuntu 16.04 with Apache2 and PHP7 installed. This server was hosted in Amazon Web Services EC2 instance. Please use the same server operative system and packages for better compatibility.



## 4. Web Application

All the code of the web application is located in **PhysioAR/Server/backend**. This application is written in HTML5, JS, CSS3 and PHP for user session and database access.

### 4.1. Code description

As this is a Web Application the pages will be described below.

**PhysioAR/Server/backend/index.php:** Verifies if user is logged and redirects according to that. If logged in user goes to main page else user goes to login page.

```
#####  
  
<?php  
// include the configs / constants for the database connection  
require_once("config/db.php");  
  
// load the login class  
require_once("classes/Login.php");  
  
$login = new Login();  
  
// ... ask if we are logged in here:  
if ($login->isUserLoggedIn() == true) {  
    // the user is logged in. you can do whatever you want here.  
    // for demonstration purposes, we simply show the "you are logged in" view.  
    //echo "<pre>";  
    //echo "SHOWING MAIN PAGE";  
    //echo "</pre>";  
  
    include("views/physioar.php");  
}  
else {  
    // the user is not logged in. you can do whatever you want here.  
    // for demonstration purposes, we simply show the "you are not logged in" view.  
    include("views/page-login.html");  
}  
#####
```

**PhysioAR/Server/backend/views/page-register.html:** This page contains the user registration form.

**PhysioAR/Server/backend/views/page-login.html:** This page contains the user login form.

**PhysioAR/Server/backend/views/physioar.html:** This is the user main page.

**PhysioAR/Server/backend/views/pages-forget.html:** This is the password recovery page.

**PhysioAR/Server/backend/views/trainanalysis.php:** This page contains the chart.js graphs and user train analysis.

**PhysioAR/Server/backend/views/trainplanner.php:** This is the page for physiotherapist assigning a train plan to the user.

**PhysioAR/Server/backend/views/usermanager.php:** This is the page for the physiotherapist to select a user/patient.

**PhysioAR/Server/backend/views/userpersonalpage.php:** This is the user/patient main page.

**PhysioAR/Server/backend/views/userRegistration.php:** This is the user/patient registration page.

## 4.2. Build and deploy

This application doesn't require a build. To deploy just place on server folder and access the address.

## 5. Mobile Application

This system is composed by two mobile application both written in Swift 4. Both applications are accessible in the Mobile folder under the PhysioAR Main folder. One app is the PhysioAR and the other is the PhysioGame.

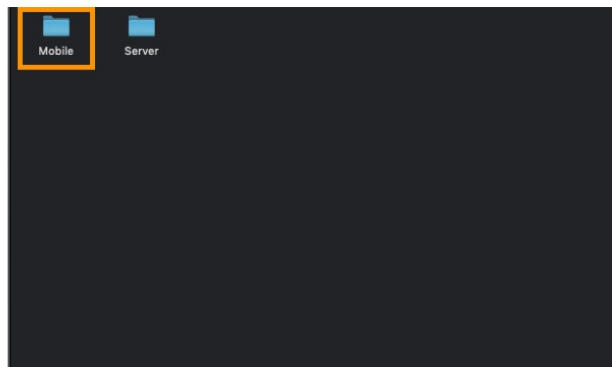


Figure 5.1 - Mobile folder

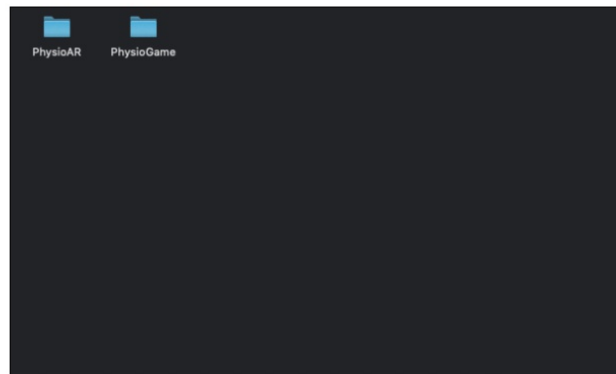


Figure 5.2 - Mobile apps

### 5.1. PhysioAR

The physioAR app is responsible for login and user data loading. Is also responsible for connecting the sensors.

The code in swift can be accessed in the physioAR folder as described in Figure 5.3

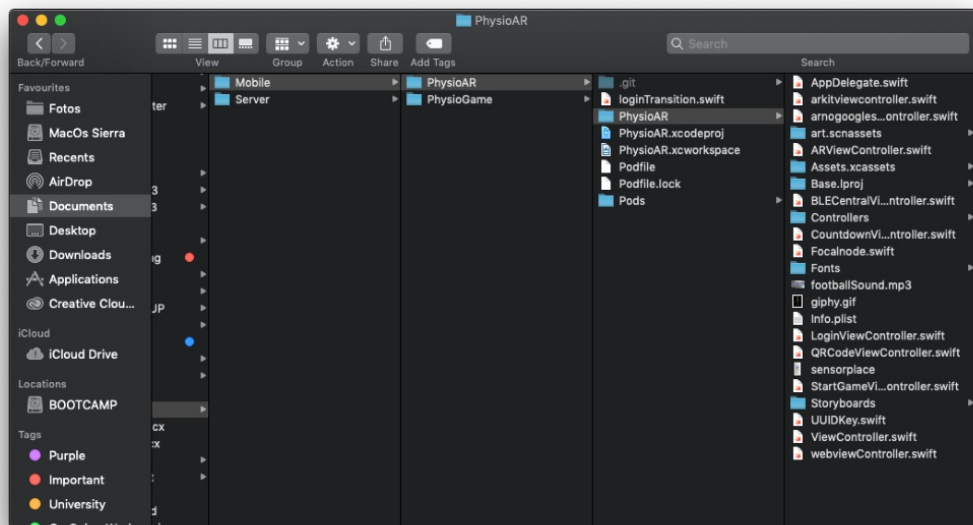


Figure 5.3 - PhysioAR mobile code location

**CountdownViewController.swift:** Contains the countdown timer screen code.

**LoginViewController.swift:** Contains the login code.

**QRCodeViewController.swift:** Contains the QRCode scanner code.

**StartGameViewController.swift:** Contains the start game screen code.

**webViewController.swift:** Contains the progress screen code.

## 5.2. PhysioGame

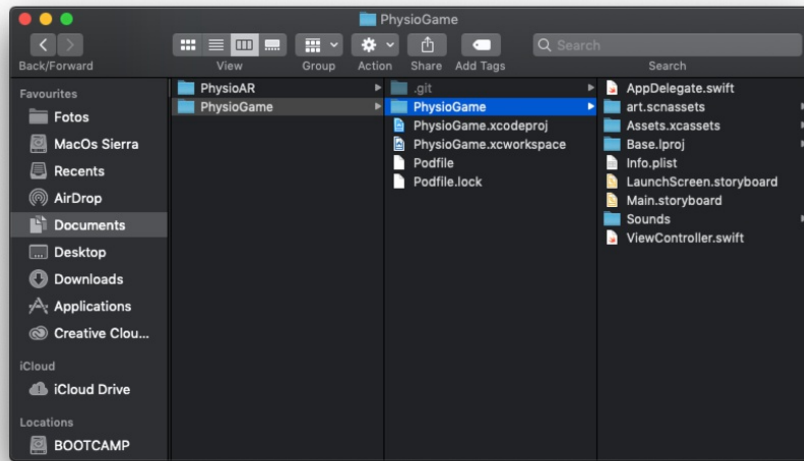


Figure 5.4 - PhysioGame Source folder

**ViewController.swift:** Contains the Augmented Reality application code.

**ViewController.swift**

```
import UIKit
import SceneKit
import ARKit
```

```
class ViewController: UIViewController, ARSCNViewDelegate {

    @IBOutlet weak var points: UILabel!
```

```
@IBOutlet weak var points2: UILabel!

@IBOutlet weak var AR_Right_Right: UILabel!
@IBOutlet weak var AR_Left_Right: UILabel!

@IBOutlet weak var GameInfo: UILabel!
@IBOutlet weak var GameInfo2: UILabel!

@IBOutlet weak var AR_Right_Left: UILabel!
@IBOutlet weak var AR_Left_Left: UILabel!

@IBOutlet weak var sceneView: ARSCNView!

@IBOutlet weak var sceneView2: ARSCNView!

var timer: Timer!
var timer2: Timer!

var animations = [String: CAAAnimation]()
var idle: Bool = true

struct Exercise{
    let id: Int
    let dur: Int
    let angle: Int
}

var exerciseList = Array<Exercise>()
var result = Array<Substring>()
var userid = String()
var date: Date!
var audioSource: SCNAudioSource!

override func viewDidLoad() {
    super.viewDidLoad()
    self.GameInfo.text="Train Starting"
    self.GameInfo2.text="Train Starting"

    audioSource = SCNAudioSource(fileName: "art.scnassets/audio/pacman.mp3")!
    userid = UIPasteboard.general.string!

    // As an environmental sound layer, audio should play indefinitely
    audioSource.loops = true
    // Decode the audio from disk ahead of time to prevent a delay in playback
    audioSource.load()

    getExerciseList()
}
```

```
while(exerciseList.isEmpty){
}

// Set the view's delegate
sceneView.delegate = self

// Show statistics such as fps and timing information
sceneView.showsStatistics = true

// Create a new scene
let scene = SCNScene()

// Set the scene to the view
sceneView.scene = scene

// Set up SceneView2 (Right Eye)
sceneView2.scene = scene
sceneView2.showsStatistics = sceneView.showsStatistics
sceneView2.isPlaying = true // Turn on isPlaying to ensure this ARSCNView
receives updates.
loadAnimations(path:"art.scnassets/idleFixed.dae")
//in your viewDidLoad
playSound(sound: "pacman", format: "mp3")
timer2 = Timer.scheduledTimer(timeInterval: 4, target: self, selector:
#selector(self.getLastestKneeAngles), userInfo: nil, repeats: true)

trainPlan(order: "start");

}
var current = 0
func trainPlan(order: String){
self.GameInfo.text=""
self.GameInfo2.text=""
print("trainPlan call")

let end = exerciseList.endIndex
if(order == "start"){

date=Date()
```

```
        timer = Timer.scheduledTimer(timeInterval: 1, target: self, selector:
#selector(self.updateAnimation), userInfo: exerciseList[0], repeats: true)
        current=current + 1
    }
    if(order == "middle"){
        if(current<end){
            date=Date()
            timer = Timer.scheduledTimer(timeInterval: 1, target: self,
selector: #selector(self.updateAnimation), userInfo: exerciseList[current],
repeats: true)
            current=current + 1
        }
        else{
            self.GameInfo.text="Train finished"
            self.GameInfo2.text="Train finished"
        }
    }
}

// must be internal or public.
@objc func updateAnimation(timer:Timer) {
    let exe = timer.userInfo as! Exercise
    playAnimation(key: convertIdToString(id: exe.id))

    if((Date().timeIntervalSince(date))>Double(exe.dur)){
        updatePoints(dur: exe.dur, threshold:exe.angle)
        self.timer.invalidate()
        trainPlan(order: "middle")
    }
}

func convertIdToString(id: Int) -> String{
    switch id {
    case 3:
        GameInfo.text="Pause"
        GameInfo2.text="Pause"
        return "pause"
    case 4:
        return "walk"
    case 6:
        return "knee90left"
    case 7:
        return "knee90right"
    case 8:
        return "knee45left"
    case 9:
        return "knee45right"
    case 10:
        return "knee180left"
    case 11:
```

```
        return "knee180right"
    default:
        return "pause"
    }
}

override func touchesBegan(_ touches: Set<UITouch>, with event: UIEvent?) {
    let location = touches.first!.location(in: sceneView)

    // Let's test if a 3D Object was touch
    var hitTestOptions = [SCNHitTestOption: Any]()
    hitTestOptions[SCNHitTestOption.boundingBoxOnly] = true

    let hitResults: [SCNHitTestResult] = sceneView.hitTest(location, options:
hitTestOptions)

    if hitResults.first != nil {
        if(idle) {
            playAnimation(key: "knee45left")
        } else {
            stopAnimation(key: "knee45left")
        }
        idle = !idle
        return
    }
}

override func viewWillAppear(_ animated: Bool) {
    super.viewWillAppear(animated)

    // Create a session configuration
    let configuration = ARWorldTrackingConfiguration()

    // Run the view's session
    sceneView.session.run(configuration)
}

override func viewWillDisappear(_ animated: Bool) {
    super.viewWillDisappear(animated)

    // Pause the view's session
    sceneView.session.pause()
}

override func didReceiveMemoryWarning() {
    super.didReceiveMemoryWarning()
    // Release any cached data, images, etc that aren't in use.
}
```



```
    // MARK: - ARSCNViewDelegate

    /*
     // Override to create and configure nodes for anchors added to the view's
     session.
     func renderer(_ renderer: SCNSceneRenderer, nodeFor anchor: ARAnchor) ->
     SCNNode? {
         let node = SCNNode()

         return node
     }
    */

    func session(_ session: ARSession, didFailWithError error: Error) {
        // Present an error message to the user
    }

    func sessionWasInterrupted(_ session: ARSession) {
        // Inform the user that the session has been interrupted, for example, by
        presenting an overlay
    }

    func sessionInterruptionEnded(_ session: ARSession) {
        // Reset tracking and/or remove existing anchors if consistent tracking is
        required
    }

    // UPDATE EVERY FRAME:
    func renderer(_ renderer: SCNSceneRenderer, updateAtTime time: TimeInterval) {
        DispatchQueue.main.async {
            self.updateFrame()
        }
    }

    func updateFrame() {

        // Clone pointOfView for Second View
        let pointOfView : SCNNode = (sceneView.pointOfView?.clone())!

        // Determine Adjusted Position for Right Eye
        let orientation : SCNQuaternion = pointOfView.orientation
        let orientationQuaternion : GLKQuaternion =
        GLKQuaternionMake(orientation.x, orientation.y, orientation.z, orientation.w)
        let eyePos : GLKVector3 = GLKVector3Make(1.0, 0.0, 0.0)
        let rotatedEyePos : GLKVector3 =
        GLKQuaternionRotateVector3(orientationQuaternion, eyePos)
        let rotatedEyePosSCNV : SCNVector3 = SCNVector3Make(rotatedEyePos.x,
        rotatedEyePos.y, rotatedEyePos.z)
    }
}
```

```
        let mag : Float = 0.066 // This is the value for the distance between two
        pupils (in metres). The Interpupillary Distance (IPD).
        pointOfView.position.x += rotatedEyePosSCNV.x * mag
        pointOfView.position.y += rotatedEyePosSCNV.y * mag
        pointOfView.position.z += rotatedEyePosSCNV.z * mag

        // Set PointOfView for SecondView
        sceneView2.pointOfView = pointOfView
    }

    func getExerciseList(){
        let url = URL(string:
"https://www.physioar.tk/backend/views/getExercises.php")!
        var request = URLRequest(url: url)
        request.httpMethod = "POST"
        request.setValue("application/x-www-form-urlencoded", forHTTPHeaderField:
"Content-Type")

        //creating the post parameter by concatenating the keys and values from
text field
        let postParameters = self.userid

        request.httpBody = postParameters.data(using: String.Encoding.utf8)

        //creating a task to send the post request
        let task = URLSession.shared.dataTask(with: request){
            data, response, error in
            guard let data = data, error == nil else {
// check for fundamental networking error
                print("error=\(error)")
                DispatchQueue.main.async(execute: {
                    let alert = UIAlertController(title: "Ups something gone
wrong", message: "Check internet connection", preferredStyle: .alert)
                    alert.addAction(UIAlertAction(title: "Back", style: .cancel,
handler: nil))
                    self.present(alert, animated: true)
                })
                return
            }

            if let httpStatus = response as? HTTPURLResponse, httpStatus.statusCode
!= 200 {
                // check for http errors
                print("statusCode should be 200, but is \(httpStatus.statusCode)")
                print("response = \(response)")
                DispatchQueue.main.async(execute: {
                    let alert = UIAlertController(title: "Ups something gone
wrong", message: "Make sure you have a good internet connection it seems like
server issues", preferredStyle: .alert)

```

```
                alert.addAction(UIAlertAction(title: "Back", style: .cancel,
handler: nil))
                self.present(alert, animated: true)
            })
        }

        let responseString = String(data: data, encoding: .utf8)

        self.result=self.exerciseToArray(exeList: responseString!)
        self.saveExercises(exercises: self.result)

    }
    task.resume()
}

func exerciseToArray(exeList: String) -> Array<Substring>? {
    if(exeList.range(of: "{") != nil && exeList.range(of: "}") != nil){
        var result = String()
        result = exeList.replacingOccurrences(of: "{", with: "")
        result = result.replacingOccurrences(of: "}", with: "")
        return result.split(separator: ";")
    }
    return nil
}

func saveExercises(exercises: Array<Substring>){
    for exe in exercises{
        var temp = String()
        temp=exe.replacingOccurrences(of: "(", with: "")
        temp=temp.replacingOccurrences(of: ")", with: "")
        let exes=temp.split(separator: ",")
        let id=Int(String(exes[0]))
        let dur=Int(String(exes[1]))
        let ang=Int(String(exes[2]))

        let result1 = Exercise.init(id: id!, dur: dur!,angle: ang!)
        exerciseList.append(result1)
    }
}

//Animations Mixamo
func loadAnimations(path: String) {
    // Load the character in the idle animation
    print(path)
    let idleScene = SCNScene(named: path)

    // This node will be parent of all the animation models
    let node = SCNNode()

    // Add all the child nodes to the parent node
```

```
for child in idleScene!.rootNode.childNodes {
    node.addChildNode(child)
}

// Set up some properties
node.position = SCNVector3(0, -10, -30)
node.scale = SCNVector3(0.1, 0.1, 0.1)

//Remove any node
sceneView.scene.rootNode.enumerateChildNodes { (node, stop) in
    node.removeFromParentNode() }

// Add the node to the scene
sceneView.scene.rootNode.addChildNode(node)

// Load all the DAE animations
loadAnimation(withKey: "walk", sceneName: "art.scnassets/walkingFixed",
animationIdentifier: "walkingFixed-1")
loadAnimation(withKey: "knee180left", sceneName:
"art.scnassets/knee180Fixed", animationIdentifier: "knee180Fixed-1")
loadAnimation(withKey: "knee180right", sceneName:
"art.scnassets/knee180Fixed", animationIdentifier:"knee180Fixed-1")

}

func loadAnimation(withKey: String, sceneName:String,
animationIdentifier:String) {
    let sceneURL = Bundle.main.url(forResource: sceneName, withExtension:
"dae")
    let sceneSource = SCNSceneSource(url: sceneURL!, options: nil)

    if let animationObject =
sceneSource?.entryWithIdentifier(animationIdentifier, withClass: CAAnimation.self)
{
        // The animation will only play once
        animationObject.repeatCount = 1
        // To create smooth transitions between animations
        animationObject.fadeInDuration = CGFloat(1)
        animationObject.fadeOutDuration = CGFloat(0.5)

        // Store the animation for later use
        animations[withKey] = animationObject
    }
}

func playAnimation(key: String) {
    // Add the animation to start playing it right away
    if(key == "walk" || key == "knee180left" || key == "knee180right"){
        loadAnimations(path: "art.scnassets/idleFixed.dae");
    }
}
```

```
        sceneView.scene.rootNode.addAnimation(animations[key]!, forKey: key)
    }
    if(key=="knee90left" || key=="knee90right"){
        fixAnimations(path: "art.scnassets/knee90Fixed.dae")
    }

    if(key=="knee45left" || key=="knee45right"){
        fixAnimations(path: "art.scnassets/knee45Fixed.dae")
    }

    if(key=="pause"){
        loadAnimations(path: "art.scnassets/idleFixed.dae");
    }
}

func fixAnimations(path: String){
    // Load the character in the idle animation
    let idleScene = SCNScene(named: path)!

    // This node will be parent of all the animation models
    let node = SCNNode()

    // Add all the child nodes to the parent node
    for child in idleScene.rootNode.childNodes {
        node.addChildNode(child)
    }

    // Set up some properties
    node.position = SCNVector3(0, -10, -30)
    node.scale = SCNVector3(0.1, 0.1, 0.1)

    // Add the node to the scene
    sceneView.scene.rootNode.enumerateChildNodes { (node, stop) in
        node.removeFromParentNode() }

    sceneView.scene.rootNode.addChildNode(node)
}

func stopAnimation(key: String) {
    // Stop the animation with a smooth transition
    if(key != "pause"){
        sceneView.scene.rootNode.removeAnimation(forKey: key, blendOutDuration:
CGFloat(0.5))
    }
}
}
```

```
var player: AVAudioPlayer?

func playSound(sound : String, format: String) {
    guard let url = Bundle.main.url(forResource: sound, withExtension: format)
else { return }
    do {
        try AVAudioSession.sharedInstance().setCategory(.playback, mode:
.default)
        try AVAudioSession.sharedInstance().setActive(true)

        player = try AVAudioPlayer(contentsOf: url, fileTypeHint:
AVFileType.mp3.rawValue)

        guard let player = player else { return }
        player.play()
    } catch let error {
        print(error.localizedDescription)
    }
}

@objc func getLastestKneeAngles() {
DispatchQueue.global(qos: .background).async{
    let url = URL(string:
"https://www.physioar.tk/backend/getRecentKneeAngle.php")!
    var request = URLRequest(url: url)
    request.httpMethod = "POST"
    request.setValue("application/x-www-form-urlencoded", forHTTPHeaderField:
"Content-Type")

    //creating the post parameter by concatenating the keys and values from
text field
    let split = self.userid.split(separator: "=");
    let postParameters = "UserID="+split[1];
    request.httpBody = postParameters.data(using: String.Encoding.utf8)

    //creating a task to send the post request
    let task = URLSession.shared.dataTask(with: request){
        data, response, error in
        guard let data = data, error == nil else {
// check for fundamental networking error
            print("error=\(error)")
            return
        }

        if let httpStatus = response as? HTTPURLResponse, httpStatus.statusCode
!= 200 {
            // check for http errors
            print("statusCode should be 200, but is \(httpStatus.statusCode)")
            print("response = \(response)")
        }
    }
}
```

```
        let responseString = String(data: data, encoding: .utf8)

        if(!(responseString == ",") && !(responseString?.isEmpty!)){
            let results=responseString?.split(separator: ",")

            DispatchQueue.main.async {
                self.AR_Left_Left.text=String(format: "%.0f",
                Double((results![0]))!)
                self.AR_Right_Left.text=String(format: "%.0f",
                Double((results![0]))!)
                self.AR_Left_Right.text=String(format: "%.0f",
                Double((results![1]))!)
                self.AR_Right_Right.text=String(format: "%.0f",
                Double((results![1]))!)

            }

        }

        task.resume()
    }
}

//SELECT count(*) FROM KneeJointAngles WHERE trainID = 17 AND (time >=
'00:00:30' AND time < '00:01:00') AND kneejoint_Angle_Rigth > 164.0
var last = 30;
func updatePoints(dur: Int,threshold: Int){
    DispatchQueue.global(qos: .userInitiated).async{
        //timer2.invalidate()
        let start = self.secondsToHoursMinutesSeconds(seconds: self.last)
        let end = self.secondsToHoursMinutesSeconds(seconds:(self.last+dur))
        self.last = self.last+dur

        let url = URL(string: "https://www.physioar.tk/backend/getPoints.php")!
        var request = URLRequest(url: url)
        request.httpMethod = "POST"
        request.setValue("application/x-www-form-urlencoded", forHTTPHeaderField:
"Content-Type")

        //creating the post parameter by concatenating the keys and values from
text field
        let split = self.userid.split(separator: "=")
```

```
        let
postParameters="UserID="+split[1]+"&start="+start+"&end="+end+"&threshold="+String(
threshold)

        // print(postParameters)
        request.httpBody = postParameters.data(using: String.Encoding.utf8)

        // request.log()
        //creating a task to send the post request
        let task = URLSession.shared.dataTask(with: request){
            data, response, error in
            guard let data = data, error == nil else {
// check for fundamental networking error
                print("error=\(error)")
                return
            }

            if let httpStatus = response as? HTTPURLResponse, httpStatus.statusCode
!= 200 {
                // check for http errors
                print("statusCode should be 200, but is \(httpStatus.statusCode)")
                print("response = \(response)")
            }

            let responseString = String(data: data, encoding: .utf8)
            print(responseString!)
            if(Int(responseString!)!>0){
                DispatchQueue.main.async{
                    self.points.text = String(Int(self.points.text!)! +
Int(responseString!)! * 10)
                    self.points2.text = String(Int(self.points2.text!)! +
Int(responseString!)! * 10)
                    //self.timer2=Timer.scheduledTimer(timeInterval: 1, target:
self, selector: #selector(self.getLastestKneeAngles), userInfo: nil, repeats: true)

                }
            }
        }

        task.resume()
    }

func secondsToHoursMinutesSeconds (seconds : Int) -> String {
    var result = String(seconds / 3600)
    result.append(":")
    result.append(String((seconds % 3600) / 60))
}
```



```
        result.append(":")
        result.append(String((seconds % 3600) % 60))
        return result
    }
}
```

## 6. Smart Sensors

This chapter contains the sensors code and a description. FTDI 5V is required to upload the code to Simblee.

### 6.1. Code description

**Motion Smart Sensor:** This code sends a Bluetooth Low Energy packet with IMU roll,pitch and tilt angles and the current time.

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LSM9DS0.h>
#include <Adafruit_Simple_AHRS.h>
#include <MyRealTimeClock.h>
#include <SimbleeBLE.h>

// Create RTC object
MyRealTimeClock myRTC(9, 11, 12); // Pin assigned

// Create LSM9DS0 board instance.
Adafruit_LSM9DS0    lsm(1000); // Use I2C, ID #1000

// Create simple AHRS algorithm using the LSM9DS0 instance's accelerometer and
magnetometer.
Adafruit_Simple_AHRS ahrs(&lsm.getAccel(), &lsm.getMag());

//Used to show if BLE is waiting for connection
int led = 13;

// Function to configure the sensors on the LSM9DS0 board.
// You don't need to change anything here, but have the option to select different
// range and gain values.
void configureLSM9DS0(void)
{
    // 1.) Set the accelerometer range
    lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_2G);
```

```
//lsm.setupAccel(lsm.LSM9DS0_ACCELRange_4G);
//lsm.setupAccel(lsm.LSM9DS0_ACCELRange_6G);
//lsm.setupAccel(lsm.LSM9DS0_ACCELRange_8G);
//lsm.setupAccel(lsm.LSM9DS0_ACCELRange_16G);

// 2.) Set the magnetometer sensitivity
lsm.setupMag(lsm.LSM9DS0_MAGGAIN_2GAUSS);
//lsm.setupMag(lsm.LSM9DS0_MAGGAIN_4GAUSS);
//lsm.setupMag(lsm.LSM9DS0_MAGGAIN_8GAUSS);
//lsm.setupMag(lsm.LSM9DS0_MAGGAIN_12GAUSS);

// 3.) Setup the gyroscope
lsm.setupGyro(lsm.LSM9DS0_GYROSCALE_245DPS);
//lsm.setupGyro(lsm.LSM9DS0_GYROSCALE_500DPS);
//lsm.setupGyro(lsm.LSM9DS0_GYROSCALE_2000DPS);
}

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

  //Setting ble visible name IMPORTANT: DONOT CHANGE IOS USES TO CONNECT
  SimbleBLE.deviceName = "sensor3";

  //Initialise BLE
  SimbleBLE.begin();

  // Initialise the LSM9DS0 board.
  if(!lsm.begin())
  {
    // There was a problem detecting the LSM9DS0 ... check your connections
    Serial.print(F("Oops, no LSM9DS0 detected ... Check your wiring or I2C
ADDR!"));
    while(1);
  }

  // Setup the sensor gain and integration time.
  configureLSM9DS0();

  myRTC.setDS1302Time(0,0,0,0,0,0);

}

void loop(void)
{
  sensors_vec_t orientation;
  sensors_event_t accel, mag, gyro, temp;

  lsm.getEvent(&accel, &mag, &gyro, &temp);
```

```
// Use the simple AHRS function to get the current orientation.
if (ahrs.getOrientation(&orientation))
{
    myRTC.updateTime();

    //Changed to accomodate textile design changes and minimize problems

    int roll = (int) orientation.pitch;
    int pitch = (int) orientation.roll;
    int tilt = (int) orientation.heading;

    String timestamp = String( String(roll) + "," + String((int)
sqrt(pow(accel.acceleration.y,2) + pow(accel.acceleration.z,2))) + ","
+String(analogRead(3)) + "," + String(myRTC.hours) + ":" + String(myRTC.minutes)
+":" + String(myRTC.seconds));

    char PostData[timestamp.length()];

    //Serial.println(timestamp.length());
    Serial.println(analogRead(3));
    sprintf(PostData,"%d,%d,%d,%d:%d:%d", roll, (int)
sqrt(pow(accel.acceleration.y,2) +
pow(accel.acceleration.z,2)), analogRead(3), myRTC.hours, myRTC.minutes, myRTC.seconds)
;

    while(!SimbleeBLE.send(PostData,timestamp.length()));
}

delay(100);
}

void SimbleeBLE_onConnect()
{
myRTC.setDS1302Time(0,0,0,0,0,0);
}

void SimbleeBLE_onAdvertisement(bool start)
{
// turn the green led on if we start advertisement, and turn it
// off if we stop advertisement

if (start)
    digitalWrite(led, HIGH);
else
    digitalWrite(led, LOW);
}
```

ECG/Motion Sensor:

```
#include <SPI.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LSM9DS0.h>
#include <Adafruit_Simple_AHRS.h>
#include <MyRealTimeClock.h>
#include <SimbleeBLE.h>

const int numReadings = 10;

int readings[numReadings]; // the readings from the analog input
int readIndex = 0; // the index of the current reading
int total = 0; // the running total
int average = 0; // the average

int inputPin = 3;

int thresholdOn = false;
int bpm = 0;
int startMillis = 0;
int currentMillis = 0;
int oldBeat=0;
// Create RTC object
MyRealTimeClock myRTC(9, 11, 12); // Pin assigned

// Create LSM9DS0 board instance.
Adafruit_LSM9DS0 lsm(1000); // Use I2C, ID #1000

// Create simple AHRS algorithm using the LSM9DS0 instance's accelerometer and
magnetometer.
Adafruit_Simple_AHRS ahrs(&lsm.getAccel(), &lsm.getMag());

//Used to show if BLE is waiting for connection
int led = 13;

// Function to configure the sensors on the LSM9DS0 board.
// You don't need to change anything here, but have the option to select different
// range and gain values.
void configureLSM9DS0(void)
{
  // 1.) Set the accelerometer range
  lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_2G);
  //lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_4G);
  //lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_6G);
  //lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_8G);
  //lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_16G);

  // 2.) Set the magnetometer sensitivity
  lsm.setupMag(lsm.LSM9DS0_MAGGAIN_2GAUSS);
  //lsm.setupMag(lsm.LSM9DS0_MAGGAIN_4GAUSS);
  //lsm.setupMag(lsm.LSM9DS0_MAGGAIN_8GAUSS);
}
```

```
//lsm.setupMag(lsm.LSM9DS0_MAGGAIN_12GAUSS);

// 3.) Setup the gyroscope
lsm.setupGyro(lsm.LSM9DS0_GYROSCALE_245DPS);
//lsm.setupGyro(lsm.LSM9DS0_GYROSCALE_500DPS);
//lsm.setupGyro(lsm.LSM9DS0_GYROSCALE_2000DPS);
}

void setup(void)
{
  Serial.begin(9600);
  startMillis=millis();

  //Setting ble visible name IMPORTANT: DONOT CHANGE IOS USES TO CONNECT
  SimbleeBLE.deviceName = "sensor5";

  //Initialise BLE
  SimbleeBLE.begin();

  // Initialise the LSM9DS0 board.
  if(!lsm.begin())
  {
    // There was a problem detecting the LSM9DS0 ... check your connections
    Serial.print(F("Oops, no LSM9DS0 detected ... Check your wiring or I2C
ADDR!"));
    while(1);
  }

  // Setup the sensor gain and integration time.
  configureLSM9DS0();

  myRTC.setDS1302Time(0,0,0,0,0,0);

  for (int thisReading = 0; thisReading < numReadings; thisReading++) {
    readings[thisReading] = 0;
  }

}

void loop(void)
{
  sensors_vec_t orientation;
  sensors_event_t accel, mag, gyro, temp;

  lsm.getEvent(&accel, &mag, &gyro, &temp);

  // Use the simple AHRS function to get the current orientation.
```

```
if (ahrs.getOrientation(&orientation))
{
    myRTC.updateTime();

    //Changed to accomodate textile design changes and minimize problems

    int roll = (int) orientation.pitch;
    int pitch = (int) orientation.roll;
    int tilt = (int) orientation.heading;

    int ecg = ecgRead();
    //Serial.println(ecg);
    if(ecg>540 && thresholdOn == false){

        thresholdOn = true;
        bpm++;

    }else if(ecg<540 && thresholdOn == true){

        thresholdOn = false;

    }

}

    int beats = (bpm * 60000)/(millis()-startMillis);

    String timestamp = String( String(roll) + "," + String((int)
sqrt(pow(accel.acceleration.y,2) + pow(accel.acceleration.z,2))) + "," + beats +
"," + String(myRTC.hours) + ":" + String(myRTC.minutes) + ":" +
String(myRTC.seconds));

    char PostData[timestamp.length()];

    //Serial.println(timestamp.length());

    sprintf(PostData,"%d,%d,%d,%d:%d:%d",roll,(int)
sqrt(pow(accel.acceleration.y,2) +
pow(accel.acceleration.z,2)),beats,myRTC.hours,myRTC.minutes,myRTC.seconds);

    if(millis()-startMillis>1000){
        Serial.println(PostData);
        sendPacket(PostData,timestamp.length());
        startMillis=millis();
        bpm=0;
    }
}
```

```
    }

    delay(1);
}

void sendPacket(char PostData[],int time){

    //Serial.println(PostData);
    while(!SimbleeBLE.send(PostData,time));
}

int ecgRead(){
    total = total - readings[readIndex];
    // read from the sensor:
    readings[readIndex] = analogRead(inputPin);
    // add the reading to the total:
    total = total + readings[readIndex];
    // advance to the next position in the array:
    readIndex = readIndex + 1;

    // if we're at the end of the array...
    if (readIndex >= numReadings) {
        // ...wrap around to the beginning:
        readIndex = 0;
    }

    // calculate the average:
    average = total / numReadings;

    // send it to the computer as ASCII digits
    return average;
}

void SimbleeBLE_onConnect()
{
myRTC.setDS1302Time(0,0,0,0,0,0,0);
}

void SimbleeBLE_onAdvertisement(bool start)
{
    // tum the green led on if we start advertisement, and turn it
    // off if we stop advertisement

    if (start)
        digitalWrite(led, HIGH);
}
```

```
    else
        digitalWrite(led, LOW);
}

EMG/Motion Sensor:

#include <SPI.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LSM9DS0.h>
#include <Adafruit_Simple_AHRS.h>
#include <MyRealTimeClock.h>
#include <SimbleeBLE.h>

const int numReadings = 10;

int readings[numReadings]; // the readings from the analog input
int readIndex = 0;         // the index of the current reading
int total = 0;             // the running total
int average = 0;          // the average

int inputPin = 3;

// Create RTC object
MyRealTimeClock myRTC(9, 11, 12); // Pin assigned

// Create LSM9DS0 board instance.
Adafruit_LSM9DS0 lsm(1000); // Use I2C, ID #1000

// Create simple AHRS algorithm using the LSM9DS0 instance's accelerometer and
magnetometer.
Adafruit_Simple_AHRS ahrs(&lsm.getAccel(), &lsm.getMag());

//Used to show if BLE is waiting for connection
int led = 13;

// Function to configure the sensors on the LSM9DS0 board.
// You don't need to change anything here, but have the option to select different
// range and gain values.
void configureLSM9DS0(void)
{
    // 1.) Set the accelerometer range
    lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_2G);
    //lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_4G);
    //lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_6G);
    //lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_8G);
    //lsm.setupAccel(lsm.LSM9DS0_ACCEL_RANGE_16G);

    // 2.) Set the magnetometer sensitivity
    lsm.setupMag(lsm.LSM9DS0_MAGGAIN_2GAUSS);
}
```



```
//lsm.setupMag(lsm.LSM9DS0_MAGGAIN_4GAUSS);
//lsm.setupMag(lsm.LSM9DS0_MAGGAIN_8GAUSS);
//lsm.setupMag(lsm.LSM9DS0_MAGGAIN_12GAUSS);

// 3.) Setup the gyroscope
lsm.setupGyro(lsm.LSM9DS0_GYROSCALE_245DPS);
//lsm.setupGyro(lsm.LSM9DS0_GYROSCALE_500DPS);
//lsm.setupGyro(lsm.LSM9DS0_GYROSCALE_2000DPS);
}

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

  //Setting ble visible name IMPORTANT: DONOT CHANGE IOS USES TO CONNECT
  SimbleBLE.deviceName = "sensor3";

  //Initialise BLE
  SimbleBLE.begin();

  // Initialise the LSM9DS0 board.
  if(!lsm.begin())
  {
    // There was a problem detecting the LSM9DS0 ... check your connections
    Serial.print(F("Oops, no LSM9DS0 detected ... Check your wiring or I2C
ADDR!"));
    while(1);
  }

  // Setup the sensor gain and integration time.
  configureLSM9DS0();

  myRTC.setDS1302Time(0,0,0,0,0,0);

  for (int thisReading = 0; thisReading < numReadings; thisReading++) {
    readings[thisReading] = 0;
  }

}

void loop(void)
{
  sensors_vec_t orientation;
  sensors_event_t accel, mag, gyro, temp;

  lsm.getEvent(&accel, &mag, &gyro, &temp);
```

```
// Use the simple AHRS function to get the current orientation.
if (ahrs.getOrientation(&orientation))
{
    myRTC.updateTime();

    //Changed to accomodate textile design changes and minimize problems

    int roll = (int) orientation.pitch;
    int pitch = (int) orientation.roll;
    int tilt = (int) orientation.heading;

    int ecg = analogRead(3);
    Serial.println(ecg);

    String timestamp = String( String(roll) + "," + String((int)
sqrt(pow(accel.acceleration.y,2) + pow(accel.acceleration.z,2))) + "," +String(ecg)
+ "," + String(myRTC.hours) + ":"+ String(myRTC.minutes) +":" +
String(myRTC.seconds));

    char PostData[timestamp.length()];

    //Serial.println(timestamp.length());

    sprintf(PostData,"%d,%d,%d,%d:%d:%d", roll, (int)
sqrt(pow(accel.acceleration.y,2) +
pow(accel.acceleration.z,2)),ecg,myRTC.hours,myRTC.minutes,myRTC.seconds);
    //Serial.println(PostData);

    while(!SimbleeBLE.send(PostData,timestamp.length()));
}

delay(200);
}

void SimbleeBLE_onConnect()
{
myRTC.setDS1302Time(0,0,0,0,0,0);
}

void SimbleeBLE_onAdvertisement(bool start)
{
    // turn the green led on if we start advertisement, and turn it
    // off if we stop advertisement

    if (start)
        digitalWrite(led, HIGH);
    else
        digitalWrite(led, LOW);
}
```

}