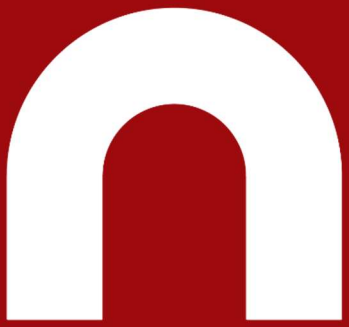


isec
Engenharia

DEPARTAMENTO DE INFORMÁTICA E SISTEMAS

Biomechanical Support System for Lower Limb Rehabilitation

Relatório de Trabalho de Projeto para a obtenção do grau de Mestre em Engenharia Informática

Especialização em Engenharia de Software

Autor
Emanuel Simões Ferreira

Orientador
César Paulo das Dores Páris

Co-Orientador
Luis Manuel Ferreira Roseiro

INSTITUTO POLITÉCNICO
DE COIMBRA

INSTITUTO SUPERIOR
DE ENGENHARIA
DE COIMBRA

Coimbra, Abril 2022

To my wife, always pushing me to new limits.

Acknowledgements

Thanks to ISEC for allowing me to work on this project that provided a compelling challenge and allowed me to expand my knowledge beyond the normal boundaries of software development.

I thank my adviser César Páris for providing this opportunity while providing the guidance and support necessary for it to come to fruition.

I thank my co-adviser Luis Roseiro for the support given and for always being open to provide access to materials, lab equipment or anything necessary that was within his reach for us to finish our work in time.

I would like to thank my wife for her continuous support during this endeavour even when things started to look hard to accomplish due to the pandemic and with the birth of our second child.

I thank my two children for their patience and forgiveness for the time that i wasn't with them because of the allotted time dedicated to finishing this academic chapter.

I thank my parents and sister for always believing in me and for their support, financial and otherwise, that allowed me to be the person i am today.

Abstract

When building rehabilitation platforms there is a certain tendency to build custom hardware that proves problematic to maintain or replacement parts are difficult to obtain, reducing their longevity and long-term utility.

With the evolution of current sensing technology, we are presented with an alternative to build viable rehabilitation technologies using off-the-shelf components for custom hardware or using commercially available products wherever possible.

Using known communication protocols, such as GATT, it is possible to interface with commercially available sensor technologies to build a platform that provides both longevity and customisation for the end user.

A set of frontend applications were developed using Angular providing a custom tailored responsive solution to managing rehabilitation patients and sessions while also providing visual stimulus to its users.

Those applications interface with the existing APIs, which were implemented using the .Net Core framework, enabling the connection of commercial and custom-built sensors that follow the Bluetooth SIG specification.

Lower limb rehabilitation is managed by a custom power meter that was built using off-the-shelf components and provided an opportunity to repurpose hardware from a similar project.

Further developments may add compatibility for other Bluetooth sensors, expanding its use beyond lower limb rehabilitation, thus increasing its longevity.

Keywords: Bluetooth, BLE, Sensors, Angular, Arduino, Bicycle

Resumo

Na construção de plataformas de reabilitação há uma certa tendência para a construção personalizada de hardware cuja manutenção se revela problemática, onde as peças de substituição são difíceis de encontrar, reduzindo a sua longevidade e utilidade a longo prazo.

Com a evolução da tecnologia atual de sensores, dispomos de uma alternativa para construir tecnologias de reabilitação viáveis, utilizando componentes do mercado para hardware personalizado ou utilizando produtos comercialmente disponíveis, sempre que possível.

Utilizando protocolos de comunicação conhecidos, como o GATT, é possível estabelecer uma interface com tecnologias de sensores disponíveis comercialmente para construir uma plataforma que proporcione tanto a longevidade e personalização para o utilizador final.

Foi desenvolvido um conjunto de aplicações front-end, utilizando o Angular, fornecendo uma solução à medida para a gestão de pacientes e sessões de reabilitação, ao mesmo tempo que proporciona um estímulo visual aos seus utilizadores.

Estas aplicações fazem interface com as APIs existentes, desenvolvidas utilizando a framework .Net Core, e que permitem a ligação de sensores comerciais e construídos à medida que seguem a especificação Bluetooth SIG.

A reabilitação dos membros inferiores é gerida por um medidor de potência personalizado que foi construído utilizando componentes comerciais e que proporcionou uma oportunidade de reutilização de hardware de um projeto semelhante.

Outros desenvolvimentos adicionais podem acrescentar compatibilidade para outros sensores Bluetooth, expandindo a sua utilidade para além da reabilitação dos membros inferiores, aumentando assim a sua longevidade.

Palavras-chave: Bluetooth, BLE, Sensores, Angular, Arduino, Bicicleta

Contents

List of Figures

List of Tables

Listings

Glossary

ANT+	A proprietary (but open access) multicast wireless sensor network technology designed and marketed by ANT Wireless (a division of Garmin Canada).
Arduino	An open-source hardware and software company, project, and user community that designs and manufactures single-board microcontrollers.
Bluetooth SIG	The standards organization that oversees the development of Bluetooth standards and the licensing of the Bluetooth technologies and trademarks.
Cadence	The rate at which a cyclist pedals (in revolutions per minute).
Chain	The chain is what turns the back wheel when you pedal.
Chain-ring	A ring with teeth for engaging the chain. When the term "chain-ring" is used in cycling, it usually refers to the front chain-ring. The rear one is usually referred to as "sprocket", or "cog".
Cleat	A part on the bottom of a shoe that attaches to the pedal of a bicycle and keeps the rider's foot in place.
Crank	Crank or Spider is the actual section that the chain rings bolt onto.
Hub	Wheel axle with bearings and cup that has spoke mounts, holds bicycle wheel, allowing it to turn freely.
Mockup	A way of designing user interfaces on paper or in computer images.
Paginator	Provides navigation for paged information, typically used with a table.
Sprocket	Ring with teeth, for engaging bicycle chain. When a term "sprocket" is used in cycling, it is usually meant the rear one, while the front one is referred to as "chain-ring".

Wi-Fi A wireless local area network that uses radio waves to connect computers and other devices to the Internet.

Acronyms

API	Application Programming Interface.
APP	Application.
ATT	Attribute Protocol.
BLE	Bluetooth Low Energy.
BMI	Body Mass Index.
BPM	Beats Per Minute.
COM	Communication Port.
CSC	Cycling Speed and Cadence.
DC	Direct Current.
ECG	Electrocardiogram.
GAP	Generic Access Profile.
GATT	Generic Attribute Profile.
GPIO	General Purpose Input/Output.
GPS	Global Positioning System.
HRM	Heart Rate Monitor.
HTML	HyperText Markup Language.
I2C	Inter-Integrated Circuit.
IDE	Integrated Development Environment.
IOT	Internet of Things.
JSON	JavaScript Object Notation.
LED	Light Emitting Diode.

NPM	Node Package Manager.
PC	Personal Computer.
PPG	Photophlethysmography.
REST	Representational State Transfer.
RPM	Rotations Per Minute.
SCL	Serial Clock.
SDA	Serial Data.
SIG	Special Interest Group.
URL	Uniform Resource Locator.
USB	Universal Serial Bus.
UUID	Universally Unique Identifier.
VIN	Voltage Input.
VRCTS	Virtual Reality-Cycling Training System.

Chapter 1

Introduction

1.1 Motivation

With the advances made in sensing technology in recent years, the opportunity for the development of new rehabilitation technologies can be addressed using off-the-shelf components for customised builds or commercially available products. However, these kinds of products, like commercial ones, are often cost-prohibitive, closed sourced, and not interchangeable and require custom hardware to function, thus limiting its uses and life expectancy.

During the development of this project we will attempt to address the issue of product compatibility and life expectancy by creating a customised platform that can be expanded whilst maintaining its original purpose, thus making it possible to add new uses in the future.

By using well-known communication protocols and taking advantage of the multitude of wireless sensors currently on the market, , it is possible to build a platform that provides compatibility for many sensors. In this way, its users can be allowed to add or remove these sensors to adapt the configuration to their needs or convert them entirely for other uses.

This document describes the steps taken to develop a platform to support physical-motor rehabilitation based on the measurement of the force exerted by the patient on the pedals of a stationary bicycle, measured using load cell sensors that are attached to both pedals, while also proving other sensor data that can be useful in the rehabilitation process.

1.2 Project Goals

The primary goal of this project is to develop a sensor platform, that will be implemented on a stationary bicycle, from which a supervisor can monitor and gather relevant data by means of an application, while a user is simulating a bicycle ride on a location of their choosing. The gathered data will then be used to improve upon the rehabilitation session of the user and to record or track its evolution.

A backoffice application will supply the means for rehabilitation supervisors to access relevant user data in real-time, provide a way to record that data for later reviewing and comparison with previous sessions and connect to available sensors using Bluetooth technology.

The front-end application will display a real-time visualization of sensor data and other relevant metrics, while providing the user with a simulation of a bicycle ride in an outdoor environment using previously recorded videos. The speed of the video is controlled by the speed recorded from the stationary bicycle setup and aims to provide further stimulus to the user thus contributing to its rehabilitation.

While the main goal of the platform will be to provide lower limb rehabilitation sensor data, by means of measuring the force applied by the user on an instrumented pedal, the platform will provide ways to integrate other sensors using BLE communication.

A communication API will mediate all communications between the array of existing sensors and the interface applications in real-time using the GATT protocol in order to maximize compatibility with existing commercial fitness and wellness sensors, since the main focus will be to expand the usefulness and lifespan of the developed platform.

1.3 Existing Platforms

With the bicycle industry being a growing one there are large number of software and hardware solutions in the market for enthusiasts and professionals alike.

While it is possible to use some commercial sensors for this project to measure cadence, heart-rate or speed, in order to expand on its longevity instead of having to develop custom hardware, most solutions are either specifically tailored for the rehabilitation and use custom made hardware that is not maintainable or compatible, or they are focused only on the fitness component as explained below.

Some sensors however, like the pedal power meters, are often very expensive and may require an custom made solution to reduce costs, while trying to maintain compatibility with its commercial equivalents.

Software applications like Zwift or Rouvy have a large compatibility to existing BLE sensors, provide the necessary stimulus with a virtual outdoor environment but lack the customization and recording functions needed for a rehabilitation platform.

Communication with bicycle sensors are usually done using either BLE or ANT+, however most computers don't support ANT+ communication without a special dongle. Also while there are many sensors that come with both protocols, usually low cost sensors only have BLE communication.

Communication with Sensors are often done using BLE Gateways, however they often do not provide any means of customization or data processing other than providing a IOT bridge between the devices and the software applications.

Commercially available rehabilitation platforms often tend to be too targeted and use proprietary hardware that is hard to maintain and don't provide any means to expand upon its use by adding other sensors or changing its primary goal.

1.4 Project Contributions

This project contributes with the following:

- Implementation of multiple commercial sensors to a stationary bicycle;
- Design and implementation of a custom pedal power meter;
- Development of a BLE communication API with customization and data preprocessing;
- Architecture and development of a rehabilitation oriented application;
- Architecture and implementation of a rehabilitation platform;
- Comparison with commercially available applications;
- Comparison with commercially available sensors.

1.5 Document Structure

The second chapter of this document will provide the necessary background information on existing software and hardware products and how they relate to the project, analysing compatibility, limitations and possible integration into the solution.

The third chapter addresses the requirement analysis that was conducted in order to gather the goals and overall objectives of the project and major futures to be developed.

The fourth chapter will provide an detailed view of the proposed solution, its architecture, while also providing the necessary information regarding its hardware implementations.

The fifth chapter will detail the development of the backoffice application detailing its major components and design features.

The sixth chapter will cover the development of the custom power meter, its firmware development and implementation.

The seventh chapter will cover the overall results of the developed solution while comparing it to commercially available products, evaluating results and validating them.

The eighth chapter will provide the conclusion to the project while also detailing the steps necessary to move the project forward.

Chapter 2

Related Work

2.1 Introduction

This section will describe the main components, software and hardware, that were studied because they were related to the project, had similar functionalities that would be interesting to implement, or were fully integrated in the development.

The main objective of this project was to address the problem of lower limb rehabilitation, so commercially available products from the bicycle industry were analysed to validate which type provided the best compatibility with the existing configuration, to obtain the force applied on the pedal by the lower limbs, while obtaining other metrics such as cadence, speed or heart rate.

A short review of the rehabilitation process and the benefits of using sensor data and biofeedback during rehabilitation sessions is provided.

For the software, the focus was on applications that offered similar functionality, based on existing requirements, regarding user motivation during training or rehabilitation and sensor data displayed or recorded, also analysing the communication used between the software and its related hardware.

The hardware analysed focused mainly on the sensors, used or considered for the project, while focusing mostly on off-the-shelf components.

In terms of communication between both components, hardware and software, several communications protocols were compared in order to validate the one that best suited the specifications.

2.2 Rehabilitation Process

Since the goal of this project is to provide a lower limb rehabilitation platform it is important to analyse what are the benefits of gathering sensor data from patients and

how the analysis of that data contributes to the overall rehabilitation process.

When compared to traditional methods, like getting manual readings of force with dynamometers, using visual scales, and non-instrumented workout machines, there are multiple advantages to using real-time sensor data collection from the patient during rehabilitation sessions. These advantages include the ability to provide graphic visualizations of the patients data, to allow rehabilitation supervisors to adjust future rehabilitation sessions in accordance to the patients evolution, as explained by Barbosa et al., 2015, and do detailed analysis and comparisons with the data gathered from the data collection devices.

Studies also corroborate the importance of providing the patient with real-time feedback of it's rehabilitation, either with games, metrics or other type of biofeedback, since it provides additional stimulus to the patient and improves the rehabilitation process when compared to usual therapies where the feedback is only provided verbally by the supervisor, as shown by Stanton et al., 2017.

By measuring the force applied by the lower limbs it is possible not only to quantify the function and overall strength of the lower limbs but also, by using force measurement devices for both legs, detect lower extremity asymmetries that can be used for studies of certain degenerative diseases such as Parkinson (Penko et al., 2014), and not just for rehabilitation.

2.3 Existing Applications

This section will discuss the cycling and fitness software applications that have been studied in order to identify their main features and rehabilitation potential. Although none of these applications fit the project requirements, which will be detailed in chapter 3, they still provided valuable information on features that were important for the solution to have in order to provide the best overall experience. The applications were analyzed based on, but not restricted to, the following parameters:

- User Motivation (whether it is a video, 3D environment or none)
- Metrics shown to the user
- Wireless sensor compatibility
- Cost
- Recording features
- Workout history

2.3.1 Zwift

Zwift is probably the most well known training application for cyclists, as it is used by professionals and enthusiasts alike. It features a monthly payment for its services and provides a virtual 3D environment for users to ride on while competing against other users from around the world (Zwift, 2021).

The application provides numerous metrics based upon the sensors attached by the user, like cadence, speed or heart rate, while also showing the distance traveled, terrain inclination or overhead map as shown on figure 2.1.

Figure 2.1: Zwift User Interface (Zwift, 2021)

Its compatible with BLE and ANT+ devices, not only supporting reading data from those devices but also controlling them, like for example when connected to smart trainers it's able to change its resistance in order to replicate hill climbs.

The movement of the avatar is controlled by power measurement, so a compatible trainer or power meter is necessary for the application to work, since it's data is necessary to extrapolate the speed of the avatar.

The application provides an helpful screen in order to pair compatible external sensors, set the main device that will supply power information and other useful configurations, as shown in figure 2.2.

During training, the Zwift application records data from the attached sensors and synchronises that data with its servers in order to maintain a complete database of user training information. At the end of each workout the user is able to view a report of relevant data gathered during the workout session, as shown in figure 2.3.

A recent version of the application also features the possibility to add direction control during the workout by using a sensor for the front wheel which allows for the detection of movement from the steering wheel of the bicycle.

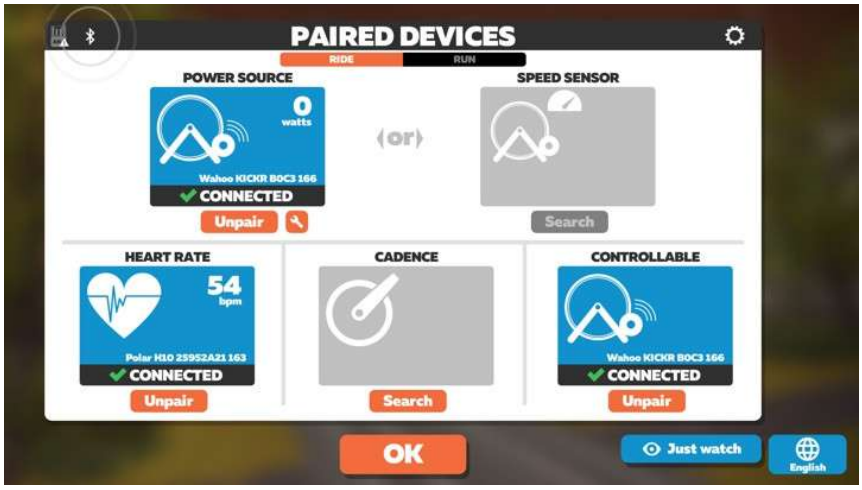


Figure 2.2: Zwift Sensor Configuration (Zwift, 2021)

Figure 2.3: Zwift Workout Report (Zwift, 2021)

Parameter	Exists	Detail
User Motivation	Yes	Virtual 3D environment with user avatars
Internet Connection	Yes	Allows multiplayer riding with people from around the world
Metrics Shown	Yes	Power, Speed, Cadence, Distance Traveled, Heart Rate and Event Related Data
Wireless sensor compatibility	Yes	Allows to connect both BLE and ANT+ devices
Multiple sensors	Yes	Supports Trainers, Cadence and Speed, Hear Rate, Power and Direction sensors
Recording Features	Yes	Session logging stores sensor data to be accessed in a later time
Personalized Hardware	No	It does not require custom hardware
Payment	Yes	Monthly or Annual subscription required

Table 2.1: Zwift Analysis

2.3.2 Rouvy

Rouvy is an application that lets you explore real places from around the world while training or racing indoors. The app uses augmented reality to place 3D models of

virtual cyclist on the real life locations in order to simulate a real life ride on them, while also allowing riding with friends thru the internet, as shown in figure 2.4 (Rouvy, 2021).

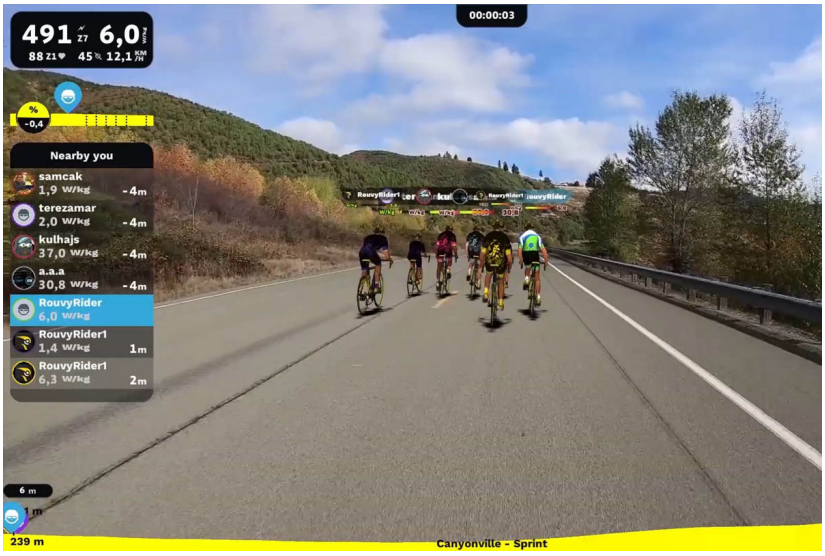


Figure 2.4: Rouvy User Interface (Rouvy, 2021)

Like Zwift it also features a monthly payment for its services and also supports multiple sensor connectivity using BLE or ANT+. It also provides the means for easy sensor configuration as shown in the figure 2.5.

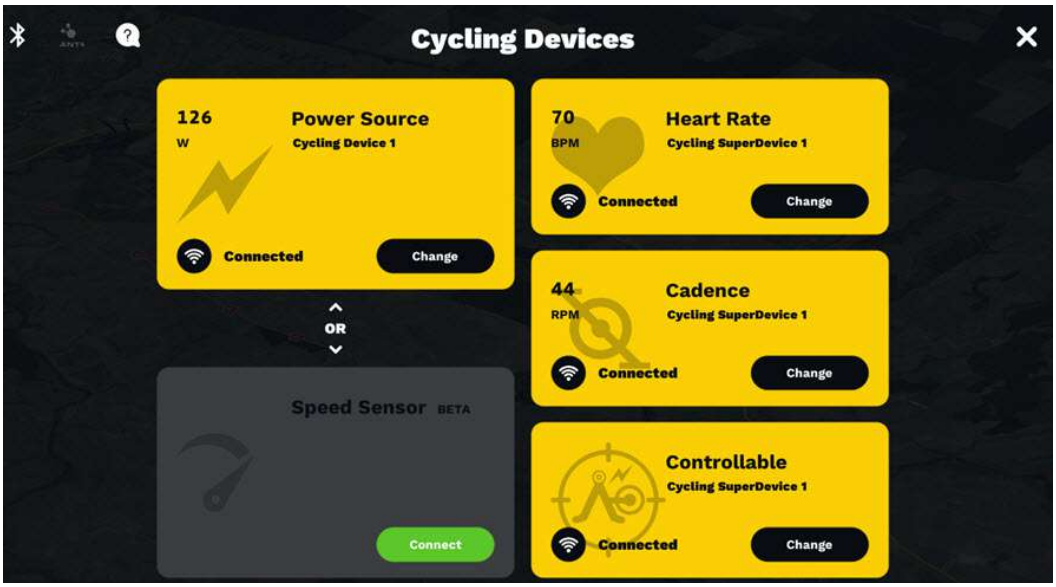


Figure 2.5: Rouvy Sensor Configuration (Rouvy, 2021)

Rouvy supports showing metrics with plot display like pulse rate, speed, cadence, power and even temperature. It also backs user-generated content, meaning a user can upload a video plus a GPS file and ride on them, which is a great motivator.

Users are allowed to participate in global events, some even based upon real-life races happening alongside real drivers.

All gathered data is stored to allow for future analysis and to provide the user with reports on their workout performance.

Parameter	Exists	Detail
User Motivation	Yes	Video of real-life locations and routes
Internet Connection	Yes	Allows participation in events and races from around the world
Metrics Shown	Yes	Power, Speed, Cadence, Distance Traveled, Heart Rate
Wireless sensor compatibility	Yes	Allows to connect both BLE and ANT+ devices
Multiple sensors	Yes	Supports Trainers, Cadence and Speed, Hear Rate and Power sensors
Recording Features	Yes	Data is recorded in training diaries alongside track dificulty and other usefull stats
Personalized Hardware	No	It does not require custom hardware
Payment	Yes	Monthly or Annual subscription required

Table 2.2: Rouvy Analysis

2.3.3 Bike Labyrinth

Bike Labyrinth is a bike tour simulation application oriented for rehabilitation and elderly care facilities. This application differs from the previous two for not being fitness or training oriented but more of a tool to provide an interactive cycling experience for elderly institutions or rehabilitation environments, thus providing a form of entertainment and improve cognitive function of the person or persons receiving care or treatment. The hardware setup consists in a custom hardware platform composed of two buttons and a speed sensor that can be attached to any stationary trainer or exercise bike. (BikeLabyrinth, 2021).

While the tour itself functions in a similar way to the Rouvy application, with videos from real life locations being the main focus, one particular feature that seemed important was the possibility of presenting the user with choices during the course of the tour, like turning left or right, which in turn changed the route presented.

The application however does not present any type of metric data besides the video itself, nor does it provide any recording capabilities. There is no 3D avatar, as in the Rouvy application, nor does it display any kind of information regarding the tour in progress.

2.3.4 SocialBike

SocialBike is an application for the physical and cognitive training of the elderly at home, developed as a academic project by Arlati et al., 2019 for the Istituto di Sistemi e Tecnologie Industriali Intelligenti per il Manifatturiero Avanzato in Italy, that aims to

Figure 2.6: Bike Labyrinth Setup (BikeLabyrinth, 2021)

Parameter	Exists	Detail
User Motivation	Yes	Video of real-life tours of known cities and places
Internet Connection	No	Local interaction only
Metrics Shown	No	No metrics are shown to the user
Wireless sensor compatibility	No	Only allows wired connection to cadence sensor
Multiple sensors	No	Only uses proprietary buttons and cadence sensor
Recording Features	No	No data is recorded during the tour
Personalized Hardware	Yes	Requires custom console and attached buttons and sensors
Payment	Yes	Hardware purchase and maintenance contract

Table 2.3: Bike Labyrinth Analysis

promote physical activity in patients of a certain age and social interaction in the form of multiplayer activities (Arlati et al., 2019).

The setup is composed of a cycle ergometer, a Cycle Ergometer 100 K by COSMED, a push button powered by an Arduino based micro-controller that connects to a PC that hosts the software. That PC is in turn connected to a screen that displays a virtual 3D environment, as shown in figure 2.7.

It supports the use of a heart rate sensor, connected by Bluetooth, in order to monitor the heart rate of the user, while the ergometer supplies cadence RPM values for the software to calculate the cycling velocity of an avatar that exists within the application.

One feature that stood out from this application was it's networking features, that allow patients to compete or collaborate in tasks within the developed application during workout exercises.

The application provides a series of cognitive tasks to the patients, that are played using the supplied push button, that involves marking previously designated targets within the 3D environment as they appear in the screen. An example of the ingame

Figure 2.7: SocialBike Setup (Arlati et al., 2019)

environment is shown in figure 2.8.

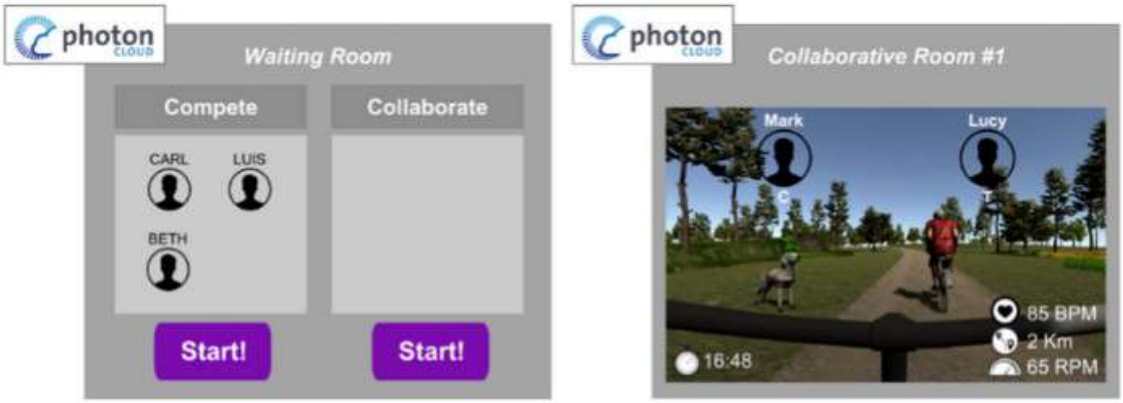


Figure 2.8: SocialBike Application (Arlati et al., 2019)

At the end of each workout exercise a score is given to the user, either a individual score or a global score depending on the task type being collaborative or not, and a full report of the workout statistics.

All private user data is synchronised to a cloud-based environment in order to be accessed and analysed later and to allow remote interaction with the user, as the configuration is designed to be used not in a medical environment, but in the user's home.

The SocialBike project is part of a larger study that aims to supply proof that biofeedback shows improvements at lower limb rehabilitation of stroke victims when compared to usual therapy methods (Stanton et al., 2017).

Parameter	Exists	Detail
User Motivation	Yes	3D environment developed using Unity
Internet Connection	Yes	Multiplayer tasks with other users and cloud networking
Metrics Shown	Yes	Time, User Information, Cadence, Distance Traveled and Heart Rate
Wireless sensor compatibility	Yes	Only with wireless Heart Rate sensors, and specific Ergometer
Multiple sensors	No	Only compatible with Heart Rate sensors
Recording Features	Yes	Data is recorded and synced to the cloud
Personalized Hardware	Yes	Requires custom Arduino based push button
Payment	No	Academic project

Table 2.4: SocialBike Analysis

2.3.5 Virtual Reality-Cycling Training System (VRCTS)

Since many studies indicated cycling as an effective mean of lower limb rehabilitation in stroke victims, the VRCTS project was developed, by Yin et al., 2016 for the National Yunlin University of Science and Technology, with the purpose of providing real-time feedback during rehabilitation programs in order to specifically train the affected side of the patient.

The setup is composed of a 3D rehabilitation game, a cycling device that includes a rotary encoder and a load cell on each pedal and a logging system for gathering data from the cycling device.

The rehabilitation game consists of a car placed in a track in which the speed and direction of the car is controlled by the cycling device. The cycling speed provides the movement of the car, converted from data supplied by the rotary encoder, while the direction is controlled by the force applied in either the left or left pedal, measured by a load cell in each pedal as shown in figure 2.9.

Figure 2.9: VRCTS Cycling Device (Yin et al., 2016)

The objective of the game is to focus rehabilitation on a specific leg of the stroke victim. To that effect the game controls the focus of the patient on that leg by changing

the track direction and making the patient turn the car to stay on track.

The direction is controlled by measuring the force applied to each pedal and measuring the differential between them. If the differential is positive it means the force applied by the right leg is stronger and the car turns right, on the contrary if the differential is negative the car turns left since the force of the left leg is stronger.

The interface of the game shows useful feedback information on the score, based upon the ability of the patient to stay on track and collect notes, number of times the patient crashed by not staying on track, round and a map with the track layout, as shown in figure 2.10.

Figure 2.10: VRCTS Cycling Device (Yin et al., 2016)

Parameter	Exists	Detail
User Motivation	Yes	3D rehabilitation game
Internet Connection	No	Local only
Metrics Shown	Yes	Time, Number of Crashes, Score, Round and Track layout
Wireless sensor compatibility	No	No wireless connections
Multiple sensors	No	No support for multiple sensors
Recording Features	Yes	Data logging of cycling force output and configuration parameters
Personalized Hardware	Yes	Custom cycling device with rotary encoder and load cell
Payment	No	Academic project

Table 2.5: VRCTS Analysis

2.4 Data Collection Hardware

The data collection hardware refers to the sensors that will be integrated in the rehabilitation platform to collect data from the patients when using it.

In the analysis conducted, the focus was on wireless technologies, to facilitate compatibility with multiple sensors, and commercial sensors that are used in the cycling industry.

2.4.1 Communication Protocols

Known communication protocols were analysed in order to choose the one that best fitted the project, taking into account energy requirements, compatibility with the sensors that will be used in the platform, external hardware required and known support from existing libraries.

2.4.1.1 Bluetooth Classic

First introduced in 1999 and adopted by over 1000 companies in its beginning Bluetooth (Haartsen, 2000) is now present in almost every smart device in the market. Similar to Wi-Fi it operates using radio waves, while not needing additional networking equipment, to communicate between two devices making it a popular choice for sending data between electronic devices.

Using Bluetooth it is possible to send pictures, videos, text or other types of data between smart devices or mobile phones over a maximum range of about 50 meters.

Bluetooth is maintained by the Bluetooth SIG, a special interest group created by leading manufacturers of the mobile industry, at the time, for the purpose of establishing Bluetooth as an industry standard, and maintaining it royalty-free through a necessary certification process for adopters (Haartsen, 2000).

2.4.1.2 Bluetooth Low Energy

Starting from version 4.0 Bluetooth specification introduced Bluetooth Low Energy (BLE), which like classic Bluetooth is also maintained by the Bluetooth SIG.

The main difference between BLE and standard Bluetooth is that it was designed to provide a significant lower power consumption, while providing less features. For example while Bluetooth can handle large amounts of data but consumes battery very quickly BLE is designed for applications that don't require large amounts of data and can extend its battery life to years if necessary.

The devices involved in the BLE communication are divided into two groups:

- Clients – Usually a smartphone or PC that accesses remote sensors to gather its data;
- Servers - A device that contains data and advertises its availability to other devices.

In the Bluetooth SIG specification, the groups are also often referred to as Central (Clients) and Peripheral (Servers).

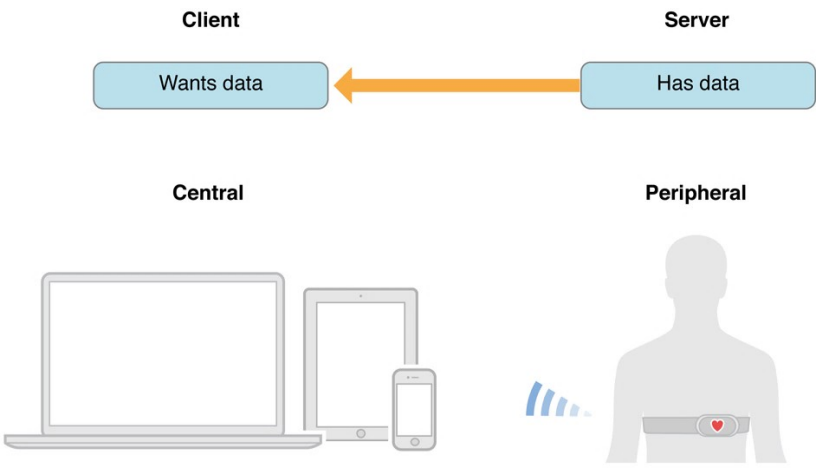


Figure 2.11: Client and Server Devices (Apple, 2013)

The role of the client is to discover, connect and interact with the servers, while the role of the server is to provide an interface for communication using services, as explained in more detail below.

The communication protocol adopted by the BLE specification is called Generic Attribute Profile (GATT), that is based on a previous existing protocol named Attribute Protocol (ATT). The GATT protocol can be viewed as a service provider, where each service has one or more characteristics that a device can read or subscribe to.

Each service is identified by a unique Bluetooth SIG-defined 16-bit UUID, as is each characteristic, which can be consulted in the Bluetooth specification documentation (Bluetooth, 2021a).

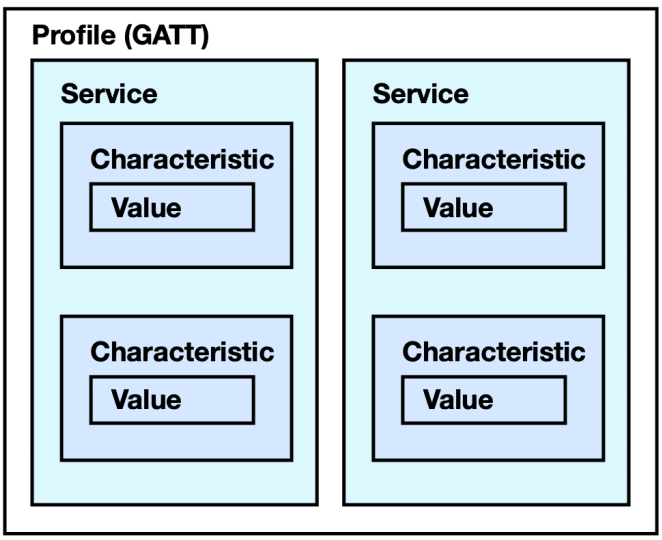


Figure 2.12: GATT Services

2.4.1.3 ANT+

ANT+, created by Garmin Inc, a popular manufacturer of smart products, is a robust and ultra-low-power communication protocol designed to be used in sensor technology. Its creators state that its devices battery can last as much as 3x when compared to other communication protocols like Bluetooth, and such claims are backed by the large number of devices and companies that use it in their products (Garmin, 2021c).

Similar to BLE it also operates in the 2.4Ghz band and is able to support multiple network topology such as Point to Point, Mesh, Star or Tree.

For a long time period the ANT+ protocol became the standard in many fitness and health devices as it was simpler to implement when compared to others, and provided good results. But due to the fact that smartphones and tablets all came equipped with Bluetooth and not many supported ANT+ the recent trend is for the devices to support both ANT+ and BLE protocols, in order for example to allow for the users to use its smartphone as a head unit, while maintaining ANT+ for legacy or specific products that use that protocol.

While this protocol showed much promise, it was discarded because of compatibility with existing devices, and other protocols requiring the purchase of a separate dongle in order to support ANT+ communication.

2.4.2 Sensor Types

The sensors were chosen based upon the following parameters:

- Commercial cost;
- Communications protocols;
- Potential Benefits in Rehabilitation;
- Compatibility;
- Availability.

2.4.2.1 Power Meter

Power meters are devices that measure the force applied by lower limbs of the cyclist while using the bicycle, usually by using strain gauges.

As explained by Cheung and Zabala (2017) there are many types of power meters available on the market as it is the most effective tool in the cycling industry to maximize athletic potential, including a different number of places on where to apply them, like the crank arm, the pedals, the axle, the bicycle wheel or even the spider of the crank arms as shown on figure 2.13. Due to the limitations of the stationary bike configuration on where to apply the sensors, a pedal-based product type was chosen. Since commercially available solutions such as the Garmin Vector (Garmin, 2021a), shown

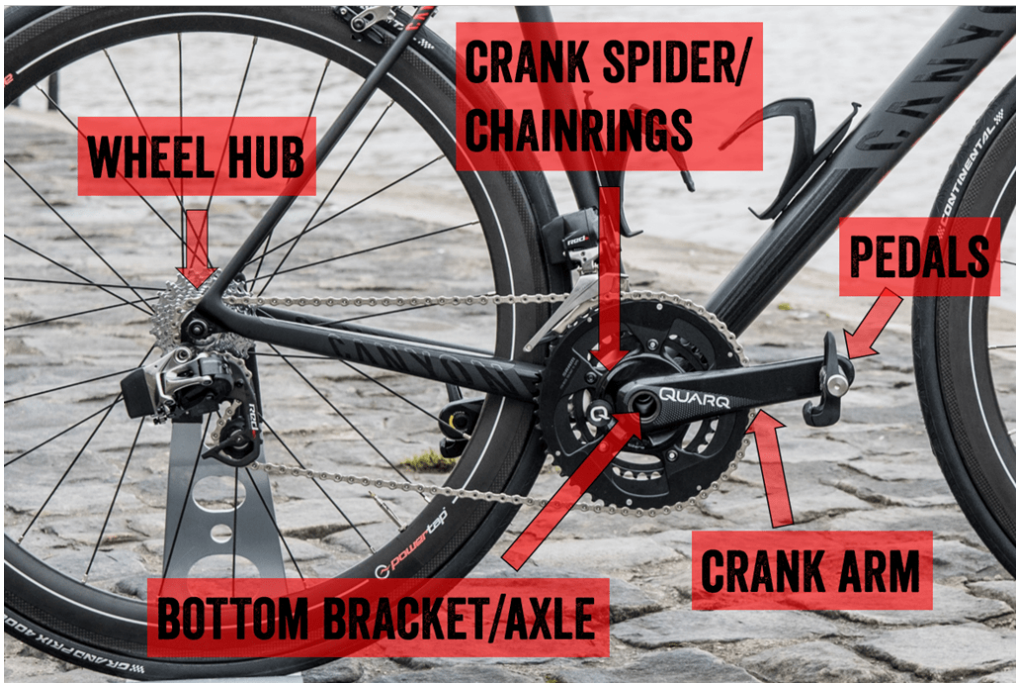


Figure 2.13: Power Meter Locations

Figure 2.14: Garmin Vector 3 (Garmin, 2021a)

in figure 2.14, come at a very high purchase value a custom-built solution was opted as the way forward. As explained above these devices usually use strain gauges in order to measure the force applied and the way that is achieved is by measuring the resistance of the strain gauge that varies in relation to the deformation of the material the strain gauge is applied to (Freddi et al., 2015).

Since the aim will be to build a custom solution, for the power meter, it was decided to use a load cell to measure the force applied to the pedal of the stationary bike. A load cell, as shown in figure 2.15, makes use of multiple strain gauges in order to accurately translate the force applied on it to an electrical signal which can be measured and standardized. The load cell can then be combined with an HX711 module, figure 2.16, to read and amplify the changes of resistance from the load cell and convert them to electrical signals that can be read with, for example, an Arduino or another chosen

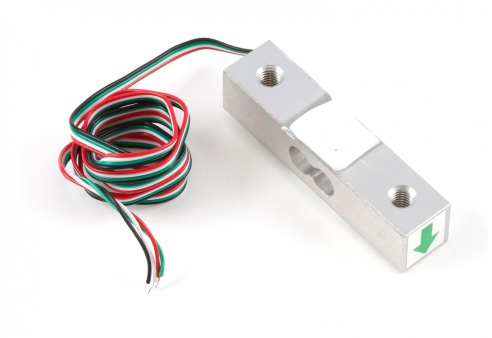


Figure 2.15: Load Cell

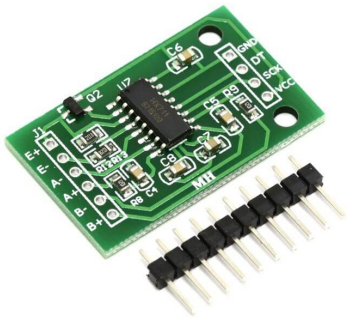


Figure 2.16: HX711 Module

micro-controller. Further development of this custom solution and its subsequent architecture is detailed in section 4.5.3.

2.4.2.2 Heart Rate

A heart rate sensor is a device that is able to measure the user's heart rate in real-time. The most popular types of devices are of the optical sensor type and of the electrode type. The optical heart rate sensors use a technique called photoplethysmography (Kamal et al., 1989) which involves measuring volumetric changes in blood circulation using infrared light. This is a very popular technique as it is non-invasive, has a relative low cost and is easy to implement. The way this works is a light emitting source, usually a light emitting infra-red diode, is paired with a photo-receiver that measures the variations in the light propagation, either by reflection or transmission in tissues and the bloodstream, as explained by Moraes et al. (2018).

In the reflection type the LED and the receiver are placed in the same side and the light that is not absorbed and reflected will be detected by the receiver. In the transmission type the LED and the receiver are placed in opposite sides and the non-absorbed light will be transmitted thru the skin and bloodstream and detected by the receiver on the other end (Moraes et al., 2018). From both types it is possible to measure a PPG signal that represents the changes in blood volume, as shown in figure 2.17 .

The electrode-based heart rate sensors are wearable devices that use electrodes in order to measure the heart rate of a user, most common one in the bicycle industry being of the chest-strap type as shown in in figure 2.18 .

They work by measuring electrical signals that the heart generates while beating, similar on how a medical electrocardiogram Electrocardiogram (ECG) works.

This is accomplished by placing the device's two electrodes in contact to the skin of the user, usually in the chest area, and those electrodes are able to pick up the small electrical signals that are generated when the heart muscles contracts. Those signals are then sent to a microprocessor placed inside the chest strap that monitors and calculates the heart rate. These devices, and other commercially available fitness

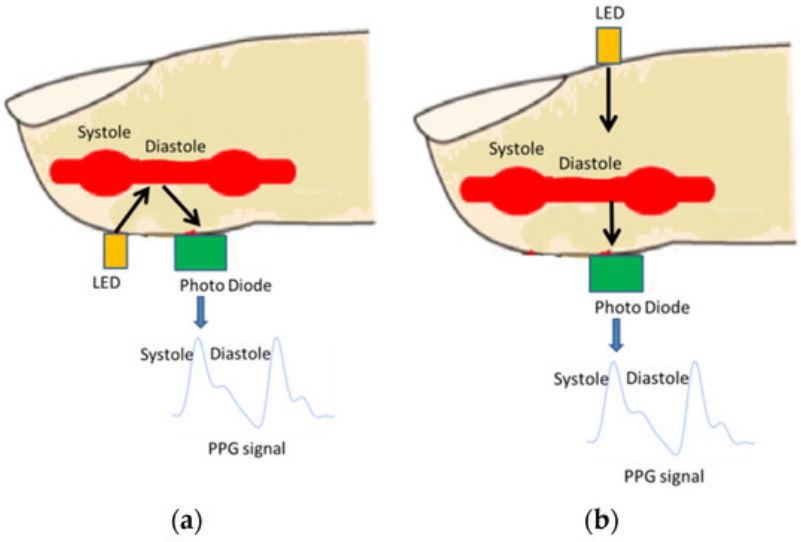


Figure 2.17: Illustration of a PPG by Reflection (a) and Transmission (b) (Liu et al., 2020)

Figure 2.18: HRM Chest Strap

wearables have been proven to have a very good accuracy when compared with medical grade ECG machines, as shown by Ge et al. (2016), and have an higher accuracy than the Photophlethysmography (PPG) type sensors.

Chest straps usually support either Bluetooth Low Energy (BLE) or ANT+ communication, making them an ideal candidate for the project, as they are also of relative low cost (Georgiou et al., 2018).

2.4.2.3 Cadence and Speed

The cadence and speed sensors are devices that measure the Rotations Per Minute (RPM) of the bicycles pedal, meaning the number of revolutions it makes in one minute, and the speed of the bicycle, by calculating it from the number of rotations of the bicycle's wheel and its diameter. Because they work in a very similar way, and some even share the same device for both functions, the explanation regarding their mode of operation will be provided as a whole.

The traditional models of these devices used a technique involving magnets that

were placed in different places of the bike in order to detect the passage of the magnet, using the principle of hall effect (Ramsden, 2006), and thus detecting rotations.

New models use an 3-axis accelerometer or a magnetometer in order to detect the rotation of the Crank arm, in the case of cadence, thus obtaining the RPM, and the rotation of the Hub in order to detect speed, and with it can also be calculated the distance traveled (Garmin, 2021b).

Since these devices use the same hardware for both features a single sensor can be used for both speed and cadence depending on the purpose it's needed for. However, if it is necessary to use both metrics at the same time, two devices must be installed.

The communication of the traditional models were done either by wire or using ANT+ communication in order to communicate with a bicycle computer.

New models, as shown on Figure 2.19 use both BLE and ANT+ in order to communicate with both smart devices and bicycle computers, thus making them a good candidate for the project, since they are easy to use and install.

Figure 2.19: Cadence/Speed Sensor

2.5 Summary

After having studied the state of the art of the different sensors that considered for the project, communication protocols and applications with similar functionalities, those that were found to meet the intended requirements and had what was needed to be integrated into the project were selected.

From the software applications reviewed none presented the intended requirements of were complete, i.e. being geared towards lower limb rehabilitation while providing other useful metrics, but they served to collect useful information regarding features, metrics, motivation and data recorded from the session.

As shown in Section 2.4.2 the sensors that provided useful metrics for the rehabilitation platform were analysed in order for the supervisor of said rehabilitation to have the best possible bio-metric feedback from the user.

Due to price constraints the only sensor that was opted to custom build is the power meter, an essential part of the setup since it is not possible to measure lower limb

rehabilitation without having a way to measure the force applied by the lower limbs. The other sensors are all off-the-shelf products, with BLE communication, relative low cost and used in the bike industry, to guarantee longevity. For the Heart Rate sensor a chest strap device was chosen and the cadence and speed sensor both share the same device, of the 3-axis accelerometer type as explained in Section 2.4.2.3.

In terms of communication as explained above in Section 2.4.1.2 the BLE communication protocol was chosen as the main means of communication between the host device and attached sensors, mostly due to the fact that it was present in all the devices tested, the Bluetooth SIG specification provides readily accessible documentation detailing the protocol and multiple libraries are available in known programming languages that are make it easy to develop applications for. Also, no dongle is necessary since most personal computers come equipped with Bluetooth network devices.

Taking inspiration from the Rouvy and Bike Labyrinth a virtual tour interface was opted as the way forward while also providing metrics from the sensors attached to the setup. Recording features will be added as they provide a useful way to recall and store the rehabilitation session and compare to other recorded sessions.

Chapter 3

Requirements Analysis

3.1 Introduction

This chapter will explain the overall view of the setup that will be provided to future users of the platform while listing the features necessary for it to provide a reliable solution to the problem at hand, providing a way to measure and quantify a lower limb rehabilitation process.

It will also be explained how the features of the application were validated during an interview with a rehabilitation professional and that provided useful information on improving the initially planned features as well as providing some for future developments.

The prototype phase will be where some potential solutions for the planned features will be tested in order to verify if they can be used in the project.

3.2 Vision and Scope

For the rehabilitation supervisor the platform aims to provide an easy way to manage and collect patient data, while also providing the necessary tools to analyse said data in an effective and intuitive manner. The supervisor will be able to manage patients details, gather patients personal information and record recurrent metrics that will be used in correlation with rehabilitation sessions.

The aim is also to configure sensor devices attached to the platform and add those sensors to existing displays that can be configured to the supervisor's needs.

The platform aims to be compatible with various commercial sensors by adhering to known communication protocols and existing specifications.

The supervisor will manage rehabilitation sessions while being able to record all

data being collected by the sensor devices, visualizing it's data in real-time and providing a practical analysis tool to previously recorded session history.

A custom power meter will be developed to measure the forced applied by the lower limbs of the user and provide those values in real-time to the developed platform.

The platform users will be provided with real-time biofeedback from their respective rehabilitation session, which includes metrics gathered from attached data acquisition sensor devices, while also providing a pleasant way to stimulate the workout with a previously recorded outdoor video.

3.2.1 Major Features

The initial set of features, described in table 3.1, of the planned rehabilitation platform were obtained from feedback gathered from previous projects developed by the academic community and other relevant published materials, in order to provide a proper set of tools to help in the rehabilitation process (Penko et al., 2014) (Barbosa et al., 2015).

Feature	Description
Supervisor Login	Provide the means for individual supervisor logins
Patient Management	Adding, Editing and Removing existing patients data
Supervisor Management	Adding, Editing and Removing existing supervisor accounts
Sensor Management	Add and Remove sensor devices to be used in sessions
Session Management	Create and Manage rehabilitation sessions
Session History	View history of existing rehabilitation sessions
Display Management	Add, Configure and Remove existing displays that will be used for sessions
Data Visualization	Provide a chart view of sensor data
Metrics Recording	Record patients relevant metrics data
Sensor Wireless Communication	Interface with existing sensor devices using wireless communication
Heart Rate Measurement	Provide support for heart rate sensors
Cadence and Speed Measurement	Provide support for cadence and speed sensors
Power Measurement	Provide support for cycling power sensors
User Interface	Provide biofeedback for rehabilitation session users
Power Meter	Provide a custom device that measures lower limb force

Table 3.1: Major Features

3.3 Requirements Validation

In order to build a platform that fits the needs required for lower limb rehabilitation, besides gathering information from academic and research related sources, an interview was conducted with the head nurse and manager of the department of physical medicine and rehabilitation of a central hospital. A From that interview it was possible to confirm the necessity of having a way to measure the force of the user but also that the application needed a way to record multiple metrics and offer comparative values to the supervisor. The following table indicates the proposed features, that were not part of the initial batch of planned features shown in section 3.2.1, that were added to the project and the justification for those that weren't.

Feature	Included	Description
Metrics recording	Yes	Weight, Height, Body Mass Index (BMI) recording methods added
O2 Saturation	No	No sensor available at the time, but can be added later due to existing BLE sensors of this type
Blood Glucose	No	No sensor available at the time, but can be added later due to existing BLE sensors of this type
Data Visualization	Yes	Although it had already been planned a way to display charts, the cut and note feature was included

Table 3.2: Proposed Features

As explained above two of the proposed sensors, besides the Heart Rate measurement, that although mentioned in the interview was already planned, were not included. This however does not prevent adding these types of sensors later, since BLE versions of these sensors exist commercially they should be compatible with the application with minor changes to add them, due to the fact that known and certified communication protocols will be used. These proposed changes will be incorporated in future developments of the application.

3.4 Prototype Development and Mockups

After gathering all the requirements information some prototype development was made in order to assure the validity of the proposed major features and to assist in the design of the support applications that would be developed.

Since the sensor integration was a critical part of the project a proof of concept API was developed in order to verify that the means existed to communicate with said sensors using BLE wireless communication.

The API that was developed using .Net Core, provided useful information on where to move forward with software development and was also presented at a conference for further validation (Ferreira et al., 2021).

A hardware test bench prototype was also created using a Arduino UNO R3 development board in order to test the possible components, described in chapter 4, that would be part of the custom power meter. For the testing of the various components the Arduino IDE provides a set of libraries that allows testing each component individually using sample code also provided by the libraries. Those samples include calibration programs for the force measuring sensor, various sensor readings and other useful applications.

Design mockups of the applications features were created to show potential users the design of the different management screens that were to be developed and of the user interface, a major component of the platform that provides biofeedback for the user, as shown in figure 3.1. This was done in order to gather feedback from the users of potentials changes that benefit the overall usability of the platform.

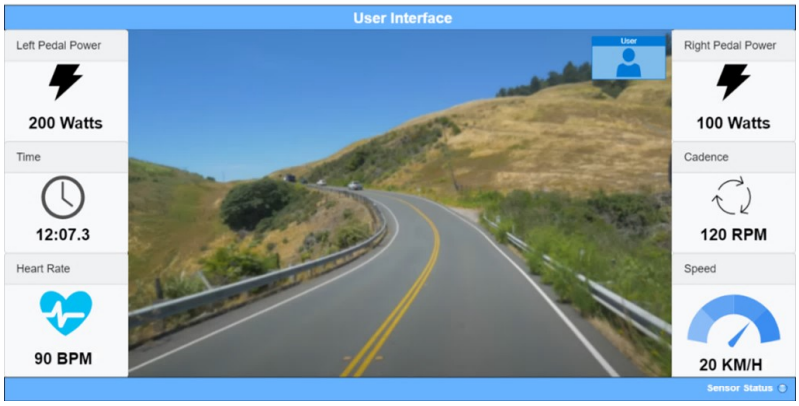


Figure 3.1: User Interface Mockup

3.5 Summary

After gathering project requirements from various sources, the focus shifted to being able to provide a viable set of features that would allow the construction of a robust rehabilitation platform.

These features were then validated with rehabilitation professionals in order to gather useful feedback that would allow the solution to be tailored to the needs of potential users. Finally, the prototyping phase of the project was addressed which helped gain insights to confidently move forward on the right path to develop a viable solution.

Chapter 4

Architecture

4.1 Introduction

This chapter describes the architecture of the platform, starting in Section 4.4 being given an overview of the architecture of the rehabilitation setup. The components of the libraries, programming languages and tools used during the hardware discovery process of the sensors used will also be detailed, as well as the API data exchange process.

The Sensor API will provide the means of interfacing the sensors, via Bluetooth BLE communication, with a host device while providing the necessary processed data to the front-end applications.

The SignalR Hub will be responsible for real-time notifications between the back-end and front-end applications.

The Backoffice API will be responsible for providing the methods to read, write and update the contents of the applications database.

The front-end applications will be divided between the user interface that will provide the necessary stimulus to the user during the session and the backoffice application used by the supervisor to record, configure and visualize data from the various sensors attached to the setup.

4.2 Project Solution

The projects solution was divided into multiple projects according to its purpose in order to better organize the development and promote separation of major functionalities, this way it is possible to change or update these functionalities without worrying about breaking other major modules.

The development was done following common programming designer patterns, promoting code reuse, wherever possible, and using the Bluetooth SIG specification as a reference for the development of the Bluetooth client methods, that will consume data provided by the sensors, and the server methods that are part of the custom power meter's firmware.

4.3 Development Environment

For the development of the applications, and the custom hardware firmware, two main Integrated Development Environments were chosen as shown in the table below. Although the Arduino IDE was used in the prototyping phase of the hardware development, it was later replaced by the Visual Studio Code since it also had the ability to program the development board firmware, and since Visual Studio Code was already being used to code the front-end applications, it was only logical to merge both developments under one IDE instead of using multiple ones.

IDE	Languages Used	Front-end	Back-end	Hardware
Visual Studio Enterprise 2019	C#	No	Yes	No
Visual Studio Code 1.62.3	Typescript, Html, CSS, Arduino Language	Yes	No	Yes
Arduino IDE 1.8.16	Arduino Language (C++)	No	No	Yes

Table 4.1: List of IDE's used

4.4 Software Components

As stated before the software part of the rehabilitation platform will be divided between two modules:

- The **front-end module** will host the user interface, that will be shown to the person using the bicycle, and the backoffice interface, that will be used by the supervisors to access and manage the session data.
- The **back-end module** will host database backoffice API, the signalR Hub that will enable push notifications and a Sensor API that will communicate with the hardware components via BLE communication, and with the front-end module thru a web based service, as shown in Figure 4.1.

The hardware components will consist of a personal computer that will host the front-end and back-end applications and will communicate with the sensors that are part of the bicycle setup. The machine will be connected to a display in order to show

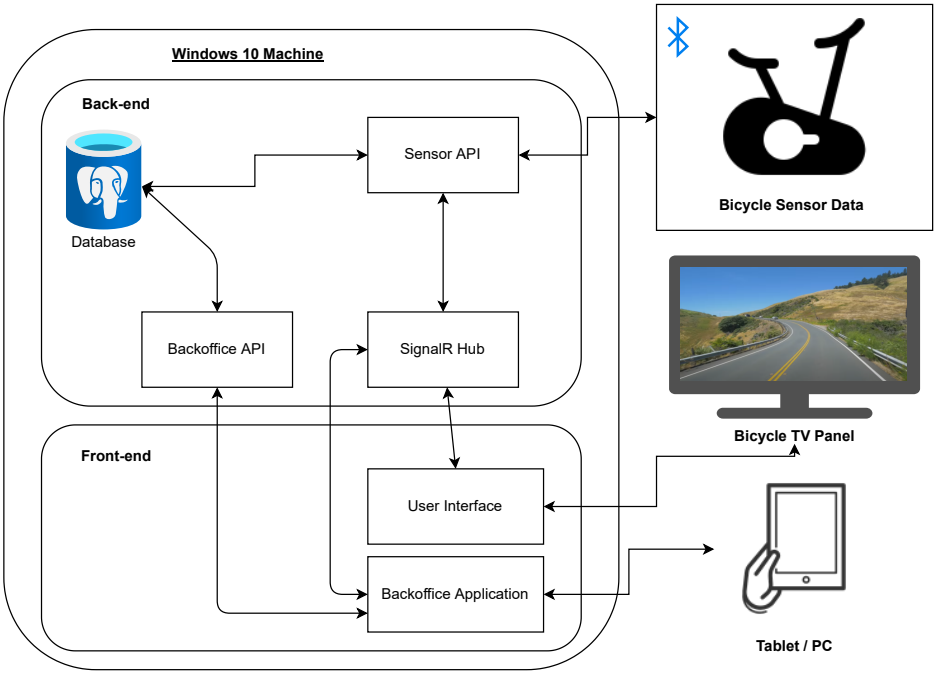


Figure 4.1: System Architecture

the user interface to the person using the bicycle setup and provide the means, thru a network connection, for the supervisor to access the administration application from an external device.

All available sensors will be connected wireless using the API that will provide compatibility to both custom-made and commercially available sensors.

There will be 4 major types of sensors used in the setup, the Heart Rate sensor, Cadence sensor, Speed sensor and Power sensor, as shown in Figure 4.2.

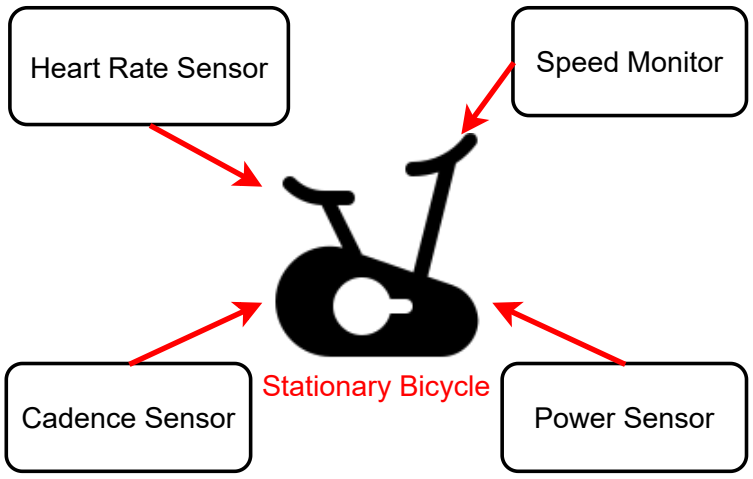


Figure 4.2: Bike Sensors

4.4.1 User Interface

To provide motivation to the user during the rehabilitation session the user interface will display a simulated ride on a real-life location. One of the main advantages of

using video recorded from real life locations, as opposed to 3D virtual environments, is that the experience can be custom tailored to the user according to their preferences, and thus improving his overall experience during the rehabilitation process.

The speed of the video is controlled by the pedal rotation speed recorded by the hardware, and thus adjusted accordingly, by using steps of 0.1 units, and within a range of 0 to 1 times the speed of the original recorded video. Background music can also be added to the video providing an additional stimulus to the user.

The main user interface will also display some metrics regarding the sensor readings taken from the available hardware.

The rehabilitation session is only started after the supervisor, thru its backoffice application detailed in Section 4.4.2, configure said session and assigns it to the user.

After the rehabilitation session is properly configured, metrics will start to be recorded as soon as changes in cadence are detected by the cadence sensor and transmitted back to the API.

4.4.2 Backoffice Application

This administration component serves the purpose of allowing the supervisors to manage the rehabilitation session, its users and hardware attached. The main purpose of this administration application will be to provide a way to record rehabilitation sessions, to provide personalized statistics regarding user's progress, to be able to access user management with individual profiles for each user, to allow data export and other features as detailed below.

Patient Management Allow the supervisor to add, remove, edit patient details.

Supervisor Management Allow a supervisor to add, remove, edit supervisor account details.

Sensor Management Allow the supervisor to scan for sensors, list the available sensors within range, add a sensor to platform or remove a existing sensor from the platform.

Display Management Allows the supervisor to customize the user interface module in regards to the information being shown on screen, adjust the metrics and add or remove sensor information.

Session Management Allows the supervisor to create new sessions, define its user and display environment and media.

Session History Allows the supervisor to visualize previous sessions, export, delete or compare user recorded metrics from previous rehabilitation sessions.

An example of the initial dashboard of the developed application is shown in figure 4.3.

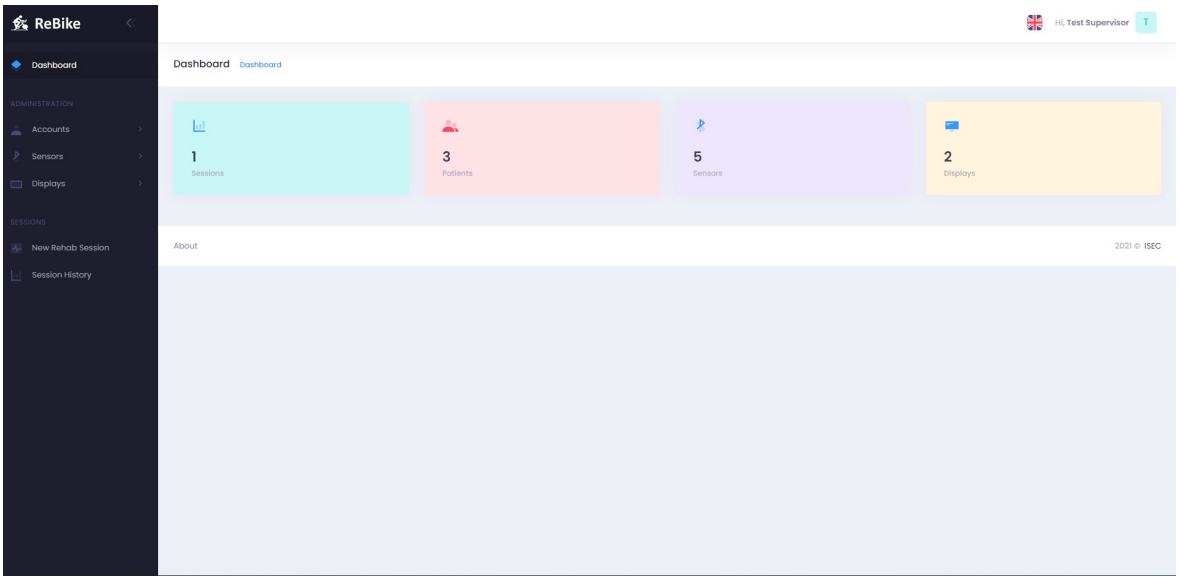


Figure 4.3: App Dashboard

4.4.3 Sensor API

The sensor API, a module that serves the primary purpose of interfacing with the sensor hardware, will allow users to scan for, and connect to, available sensors and process and return metric data to the user interface and backoffice applications.

This API will allow wireless connectivity to the available sensors using Bluetooth Low Energy, thus expanding compatibility of the existing configuration to off-the-shelf components, such as fitness trackers, and thus eliminating the need to develop custom hardware for other sensors that can be easily obtained commercially, or for the user to use their own.

This will be accomplished by following the rules and guidelines of the Bluetooth SIG specification, allowing the API to identify the sensor hardware properly thru its predesignated unique identifiers.

To verify compatibility of the chosen sensors the nRF Connect application was chosen for an initial exploratory analysis. This testing tool allows the discovery of the devices, connect to them, list their services and characteristics.

With this analysis it was possible to verify that the sensors advertised the proper services, like the Heart Rate Service as validated by the 0x180D UUID, and its corresponding characteristic, and that future developments could be pursued, as shown in figure 4.4. This was done for every sensor in order to validate that it was compatible with the API, it provided the necessary services, and that the correct values could be obtained from its advertised characteristics in order to be used in the applications.

Using the information provided by the analysis it was possible to create methods for the advertised services and characteristics, based on their unique identifiers, that would be used in the applications.

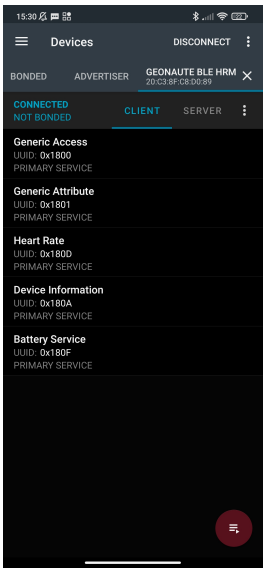


Figure 4.4: nRF Connect Application

4.4.4 SignalR Hub

The SignalR Hub module provides the means for real-time communication between the front-end and back-end applications by using the SignalR Microsoft library (Microsoft, 2021).

SignalR enables real-time functionality so that applications are are to push content to connected clients using websockets in real-time. This enables notification of front-end applications whenever new sensor data is available, the status of the user interface changes or there are important changes to objects in the database that need to be reflected in the client applications in order to maintain coherence.

In the case of the user interface application each client, known as Display, connects to the hub when it is started and subscribes to receive changes for each of the attached sensors that are part of the rehabilitation session created for that display. Each time new data is available from the attached sensors a push notification is sent to the client which receives that data and displays it in the user interface. At the same time the sensorhub stores that new sensor value in the database for future consumption.

The user interface also subscribes to receive changes to its current status, in order for the backoffice application, mainly the session management module, to be able to start, stop, restart or end the rehabilitation session that is currently running on the display.

As for the backoffice applications they also subscribe to a status change notification in order to reflect the current status of the rehabilitation's session display correctly if it's changed by another application or if the display disconnects from the sensor hub, thus stopping the rehabilitation session.

4.4.5 Backoffice API

The backoffice API is a Restful based API that is responsible for providing methods to access the database system. A web service will provide methods for reading, writing and updating the objects stored that are part of the developed applications. Those methods will in turn be used by the front-end applications.

The database is comprised of a PostgreSQL server that hosts all the backoffice objects data and stored sensor data.

PostgreSQL was chosen as the database because it's an open source relational database system that is easily integrated into the chosen programming language, is compatible with the EF Core library, has a good performance and is reliable as it has been in development for several years.

4.5 Hardware Components

As stated before the sensors are comprised of commercially available sensors and a custom built power sensor that will be attached to a stationary bicycle setup. The following topics will detail each of the commercial sensors chosen and the components used to build the custom power sensor.

4.5.1 Cadence and Speed Sensor

The device used in the setup for this feature is a Magene S3+ Speed/Cadence Dual Mode Sensor (Magene, 2020). The chosen sensor for obtaining cadence is also the same one used to obtain speed since the device combines both features into a single device.

To switch between the cadence and speed mode one must only replace the battery and the device will flash a green light when in speed mode or a red light when in cadence mode.

The device uses a geomagnetic sensor chip to detect movement and thus accurately calculate cadence or speed, as stated by its manufacturer. It provides communication with both BLE and ANT+ compatible devices and is of relative low cost.

According to the Bluetooth SIG specification (Bluetooth, 2012b) for this type of device, when using Bluetooth communication mode, a Cycling Speed and Cadence Service is advertised and from that service a list of mandatory and optional characteristics is provided as shown in table 4.2.

Apart from this service the device also advertises other optional generic services such as the Device Information Service, which provides information about the device like manufacturer name, serial, model and other relevant device information, and the Battery Service, that provides only one characteristic to get a reading of the device's battery level. The advertised services from the device, viewed using the nRF Connect app, are shown in figure 4.5.

Characteristic	Requirement	Properties
CSC Measurement	Mandatory	Notify
CSC Measurement Descriptor	Mandatory	Read, Write
CSC Feature	Mandatory	Read
Sensor Location	Optional	Read
SC Control Point	Optional	Write, Indicate
SC Control Point Descriptor	Optional	Read, Write

Table 4.2: Cycling Speed and Cadence Service

In this project, the parser for the CSC Measurement feature will be implemented to obtain the Cadence or Speed values. According to the Cadence and Speed Profile (Bluetooth, 2012a) this characteristic will provide the data to the collector, the name given to clients that consume the devices services, depending on the content present in a flags field that is part of the characteristic, as shown in table 4.3.

Bit Number	Definition
0	Wheel Revolution Data Present: 0: False 1: True
1	Crank Revolution Data Present 0: False 1: True
2-7	Reserved for Future Use

Table 4.3: CSC Measurement Flags

In the case of this device, which is a speed and cadence combo, the flags value will change when the device is either in Speed or Cadence mode according to its configuration instructions, referred to in the beginning of this section. If the device is in Speed mode the 0 bit number will be set to true and the Wheel Revolution Data will be present in the characteristics data, and in alternative if the device is in Cadence mode the 1 bit number will be set to true instead and the Crank Revolution Data will be present.

The complete description of the fields provided by the characteristic is presented in table 4.4, as detailed in the GATT Specification Supplement documentation (Bluetooth, 2021b).

Knowing the data structure of the characteristic, it is now possible to calculate the instantaneous Cadence and Speed values.

Since the device only reports the total cumulative revolutions and last event time, each time new data is received from the device, those values will be stored in order to compare them to the previous data and make the necessary calculations.

Therefore for each two consecutive measurements the following calculations can be made, shown in equation 4.1, in order to obtain the instantaneous speed value.

Field		Data Type	Size (in octets)	Description
Flags		struct	1	See table 4.3
Wheel Revolution Data Present if bit 0 of Flags field set to 1	Cumulative Wheel Revolutions	uint32	4	Unit: org.bluetooth.unitless
	Last Wheel Event Time	uint16	2	Base Unit: org.bluetooth.unit.time.second Represented values: M = 1, d = 0, b = -10 Unit is 1/1024th of a second
Crank Revolution Data Present if bit 0 of Flags field set to 1	Cumulative Crank Revolutions	uint16	2	Unit: org.bluetooth.unitless
	Last Crank Event Time	uint16	2	Base Unit: org.bluetooth.unit.time.second Represented values: M = 1, d = 0, b = -10 Unit is 1/1024th of a second

Table 4.4: CSC Measurement Characteristic

$$Speed = \frac{Cumulative\ Wheel\ Revolutions\ Difference \times Wheel\ Circumference}{Last\ Wheel\ Event\ Time\ Difference} \tag{4.1}$$

The Cumulative Wheel Revolutions Difference represents the difference between the current measured value of the total number of wheel revolutions minus the previously stored value of the previous measurement.

The Wheel Circumference is obtained by providing the diameter of the bicycle wheel and applying the following equation.

$$Wheel\ Circumference = \pi \times Wheel\ Diameter \tag{4.2}$$

The Last Wheel Event Time Difference is the difference between the current measured value of the timestamp provided by the device minus the previously stored value of the previous measurement.

In the same manner, the following calculation, shown in equation 4.3, can also be performed to obtain the instantaneous cadence value.

$$Cadence = \frac{Cumulative\ Crank\ Revolutions\ Difference}{Last\ Crank\ Event\ Time\ Difference} \tag{4.3}$$

The Cumulative Crank Revolutions Difference is obtained by subtracting the value of the current measured total number of crank revolutions from the previously stored value of the former measurement that was stored.

The Last Crank Event Time Difference is obtained by subtracting the value of the current timestamp minus the timestamp of the previous measurement.

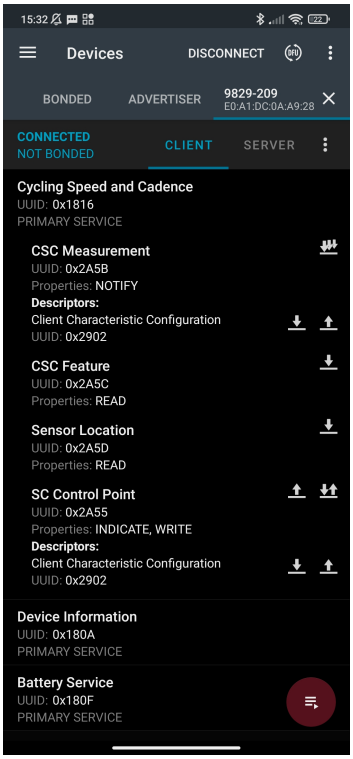


Figure 4.5: CSC Services

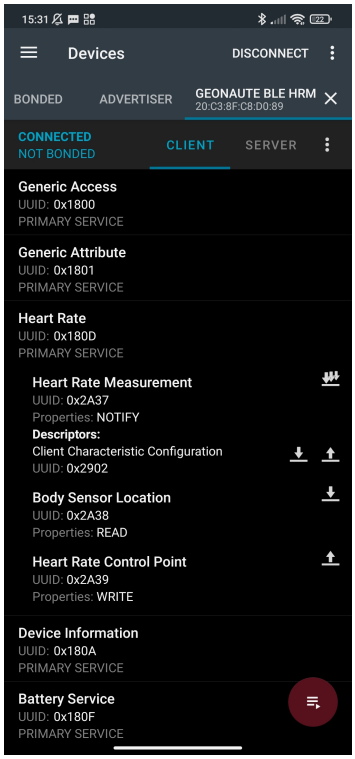


Figure 4.6: Heart Rate Services

4.5.2 Heart Rate Monitor Sensor

The heart rate monitor device used is a Geonaute Bluetooth HRM Chest Strap (Geonaute, 2015) that uses electrodes to measure the heart rate of its user, as explained in chapter 2.4.2.2, and communicates with client devices using Bluetooth LE.

As stated in its Bluetooth SIG specification (Bluetooth, 2011b), for this type of device, it advertises the Heart Measurement Service which provides a set of mandatory and optional characteristics as shown in table 4.5.

Characteristic	Requirement	Properties
Heart Rate Measurement	Mandatory	Notify
Heart Rate Measurement Descriptor	Mandatory	Read, Write
Body Sensor Location	Optional	Read
Heart Rate Control Point	Optional	Write

Table 4.5: Heart Rate Service

As the previous device this sensor also advertises the Device Information and Battery Services as shown in figure 4.6.

For the purpose of this project the a parser for the Heart Rate Measurement will be implemented so that it will be possible to obtain BPM values from the device. According to the Heart Rate Profile (Bluetooth, 2011a) this characteristic can supply different fields of data depending on what the manufacturer implements on any given device, besides the mandatory Heart Rate measurement value.

The collector is informed of the existence of optional fields and features by means of a flags field present in the characteristic's data. The possible contents of the flags field is shown in table 4.6.

Bit Number	Definition
0	Heart Rate Value Format: 0 = Heart Rate Value Format is set to uint8 1 = Heart Rate Value Format is set to uint16
1	Sensor Contact detected 0 = False 1 = True
2	Sensor Contact Supported 0 = False 1 = True
3	Energy Expended present: 0 = False 1 = True
4	RR-Intervals present: 0 = False 1 = True
5-7	Reserved for Future Use

Table 4.6: Heart Rate Measurement Flags

The Heart Rate Value Format indicates if the format of the Heart Rate Measurement value is an unsigned integer of 8-bit or 16-bit size.

The Sensor Contact bits indicates if the device supports the Sensor Contact feature, and if the feature exists if contact between the device and the users skin is detected.

The Energy Expended bit indicates if that feature is supported by the device and is present in the characteristic, which measures the accumulated energy expended measured in kilo Joules.

The RR-Interval checks whether if RR-Intervals field is available in the characteristic. It represents the time between two consecutive R waves in an ECG waveform, as shown in picture 4.7.

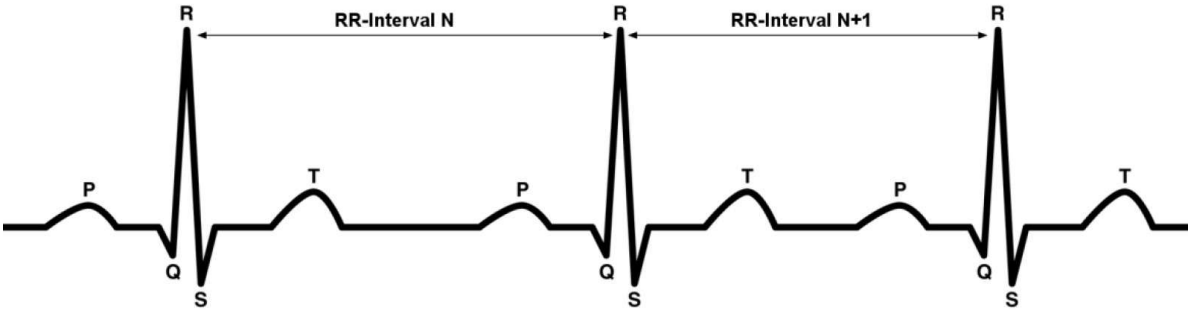


Figure 4.7: ECG RR-Interval Waveform (Bluetooth, 2021a)

According to the device chosen, development was focused only on obtaining the

Heart Rate measurement value since other features were not available. The complete set of fields that are part of the characteristic are shown in table 4.7.

Field	Data Type	Size (in octets)	Description
Flags	struct	1	See table 4.3
Heart Rate Measurement Value	If bit 0 of Flags field set to 0: uint8	If bit 0 of Flags field set to 0: 1	Unit: org.bluetooth.unitless
	If bit 0 of Flags field set to 1: uint16	If bit 0 of Flags field set to 1: 2	Base Unit: org.bluetooth.unit.time.second Represented values: M = 1, d = 0, b = -10 Unit is 1/1024th of a second
Energy Expended Present if bit 3 of Flags field set to 1	uint16	0 or 2	Unit: org.bluetooth.unit.energy.joule
RR-intervals Present if bit 4 of Flags field set to 1	uint16 Array	0 or n*2	See figure 4.7

Table 4.7: Heart Rate Measurement Characteristic

Since the Heart Rate Measurement raw value is already in BPM format no conversion is needed for this characteristic.

4.5.3 Custom Power Sensor

As explained in chapter 2.4.2.1 due to the high cost of the commercial power meter devices, and the goal of repurposing parts from an previous project, it was opted to build one using off-the-shelf components in order to maintain repair-ability while keeping the overall cost low.

A load cell is used in conjunction with several other components, such as a micro-controller, that are part of the pedal setup in order to achieve the goal of measuring the force of the lower limbs of its user and transmit that data wireless to the developed applications.

The power meters were built in a set of two individual pedals, since since it is necessary to measure the force applied by both legs of the user at the same time, while also providing the means to compare both measurements.

The assembled circuit board was then attached to the base of a Zeray ZP-108S cleat type pedal and connected to the previously developed load cell interface. Both pedals, left and right, are shown in figure 4.8.

In the next subsections, the components used to build the custom power pedal will be detailed, as well as their purpose in the overall construction and the overall diagram of the prototype circuit.

Figure 4.8: Assembled Pedals

4.5.3.1 Cycling Power Service

To make the pedal compatible with the applications and still maintain some form of compatibility with the Bluetooth SIG specification the firmware was developed following the existing specification for the commercial counterparts of pedal based power meters, and thus opening the possibility of the custom pedals to be used in other applications, if development continues.

The Bluetooth SIG specification (Bluetooth, 2016b), for a commercial device of this type, indicates that it must advertise the Cycling Power Service from which a whole set of mandatory and optional features are provided according to table 4.8.

Characteristic	Requirement	Properties
Cycling Power Measurement	Mandatory	Notify
Cycling Power Feature	Mandatory	Read
Sensor Location	Mandatory	Read
Cycling Power Control Point	Optional	Write, Indicate
Cycling Power Vector	Optional	Notify

Table 4.8: Cycling Power Service

Since the firmware development would be focusing on making it compatible with the Bluetooth SIG specification, the collector part of the development, integrated in the Sensor API, also followed that specification in order to achieve some form of compatibility with commercial products. This was made so that in the future, if the need would

arise, one could eventually replace the custom made power meter with a commercial counterpart without losing overall functionality.

Contrary to the previous sensor parser implementations, this one in particular will not focus primarily on the mandatory Cycling Power Measurement, but on the optional feature Cycling Power Vector instead. The reasoning for this is that the Cycling Power Measurement, while providing useful data, does not provide the raw value of the force being applied by the lower limbs on the pedals.

The optional characteristic however, is used to send raw data of torque or force, being force the unit of measure that matters, and therefore makes it the perfect candidate for development.

According to the Cycling Power Profile, (Bluetooth, 2016a) this characteristic, the Cycling Power Vector, provides the collector with various optional fields of data depending on what its defined in its flags field, whose contents are shown in table 4.9.

Bit Number	Definition
0	Crank Revolution Data Present 0: False 1: True
1	First Crank Measurement Angle Present 0: False 1: True
2	Instantaneous Force Magnitude Array Present 0: False 1: True (Note 1)
3	Instantaneous Torque Magnitude Array Present 0: False 1: True (Note 1)
4-5	Instantaneous Measurement Direction
6-7	Reserved for Future Use

Table 4.9: Cycling Power Vector Flags

Depending on the data defined as present in the flags field, the characteristics data presents itself as shown in table 4.10.

Since all fields are optional, the one field that that matters is the Instantaneous Force Magnitude Array, since it's that field of data that stores the force values measured by the device, in newtons.

The field uses an array in order to store raw force sensor values that are sampled between notification attempts. The number of values depend on the difference between the sampling rate and the notification interval.

Field		Data Type	Size (in octets)	Description
Flags		16-bit	2	See table 4.9
Crank Revolution Data Present if bit 0 of Flags field set to 1	Cumulative Crank Revolutions	uint16	2	Unit: org.bluetooth.unitless
	Last Crank Event Time	uint16	2	Base Unit: org.bluetooth.unit.time.second Represented values: M = 1, d = 0, b = -10 Unit is 1/1024th of a second
First Crank Measurement Angle Present if bit 1 of Flags field set to 1		uint16	2	Unit: org.bluetooth.unit.plane_-angle.degree
Instantaneous Force Magnitude Array Present if bit 2 of Flags field set to 1		sint16 Array	0–18	Unit: org.bluetooth.unit.force.newton
Instantaneous Torque Magnitude Array Present if bit 3 of Flags field set to 1		sint16 Array	0–18	Base Unit: org.bluetooth.unit.moment_-of_force.newton_metre Represented values: M = 1, d = 0, b = -5 Unit is 1/32 Newton meter

Table 4.10: Cycling Power Vector Characteristic

4.5.3.2 ESP32 Microcontroller

The microcontroller chosen for the custom pedal was a ESP32-WROOM-32E (Espressif, 2021). This development board provides multiple analog and digital pin for attaching components and it's easily programmable using known Integrated Development Environments commonly used for the Arduino boards.

It provides Bluetooth BLE and Wi-Fi connectivity, although only the Bluetooth communication was used in the project.

The board was chosen because of it's small form factor, and since initial prototypes were made using an Arduino Uno R3 board the knowledge and code developed could easily be ported to this device since both share the same IDE and libraries.

Besides Bluetooth Classic the board also supports Bluetooth 4.0, also known as BLE or Bluetooth Smart, which enables it to easily perform communication with client devices, and to develop this communication using the same protocols used in the commercial sensors of choice.

The board can be powered either using the on-board micro USB connector or by supplying 5V to the VIN pin, which in this case is done using a voltage regulator, as

explained in section 4.5.3.6.

In order to develop for this board the Arduino IDE allows for the installation of a ESP32 Arduino Core library, provided by Espressif, that supplies the basic libraries for communication with various peripheral devices that can be connected to the development board.

The Arduino ESP32 core library also enables firmware development for the device to act as a BLE server, as explained in section 2.4.1.2, and advertise it's services and characteristics to the clients.

For the board to communicate with the external components some external libraries were added to development as shown in table 4.11.

Library	Version	Description
Adafruit MPU6050	2.0.5	Adafruit library for the MPU6050 sensors
HX711_ADC	1.2.9	Library to interface with the HX711 for reading load cells / weight scales
Adafruit Unified Sensor	1.1.4	A unified sensor abstraction layer used by many sensor libraries
Arduino core for the ESP32	2.0.2	Provides BLE and peripherals connectivity

Table 4.11: Arduino Libraries Used

Logging is done using a serial console provided by Arduino IDE and uses an available COM port that is made available by the ESP32 when using the USB connector. The developed sketches can then output to that console in real-time in order to help with development and debugging operations.

Further details on the microcontroller's firmware development phase are explained in chapter 6.

4.5.3.3 Load Cell

In order to promote reusability the load cells used in this build were supplied by a previous project called Exobike (Santos et al., 2019). That project while having a similar goal, collecting data from rehabilitation sessions, lacked wireless communication and only focused on the mechanical part of development of the sensors.

To that effect only the mechanical interface part of the developed pedal from that project was used, which includes the load cell and the clipping shoe interface that will attach to the existing bicycles pedals using cleats, as shown in figure 4.10.

The load cell used in the developed interface is a planar load cell, Vetek 202WA-200lb, which according to its specifications can detect a maximum weight of 90,72 kg (Vetek, 2019).

Figure 4.9: ESP32 Development Board

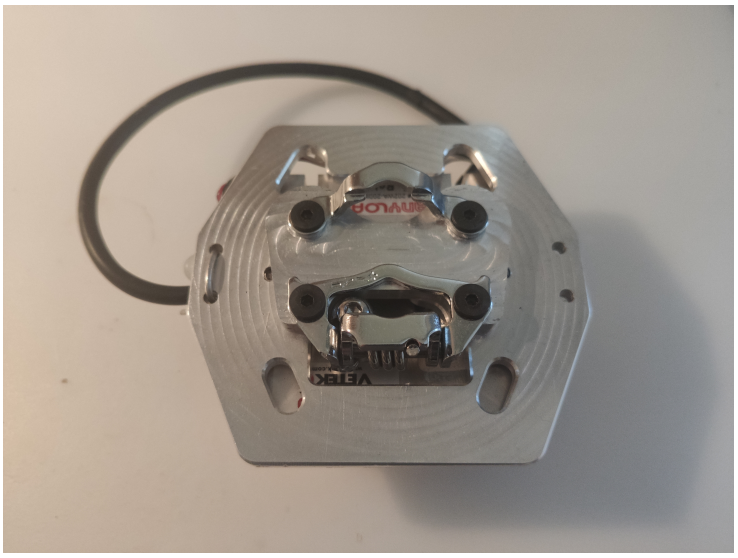


Figure 4.10: Exobike Load Cell

These types of load cells use strain gauges in order to detect material deformation, in a Weightstone Bridge configuration, as shown in figure 4.11. By detecting the small variations in resistance that happen when there is a physical change in the load cell, it is then possible to measure, with a previously calculated calibration factor, the weight that is being applied on the load cell.

The load cell is attached to an amplifier board that enables the microcontroller to detect these changes in the resistance, as explained in section 4.5.3.4.

4.5.3.4 HX711

The HX711 (Avia, 2021) is a 24-bit analog to digital converter that serves as a load cell amplifier allowing it to connect to the chosen microcontroller and, as explained in the previous section, detect changes in the resistance of the load cell and thus measuring

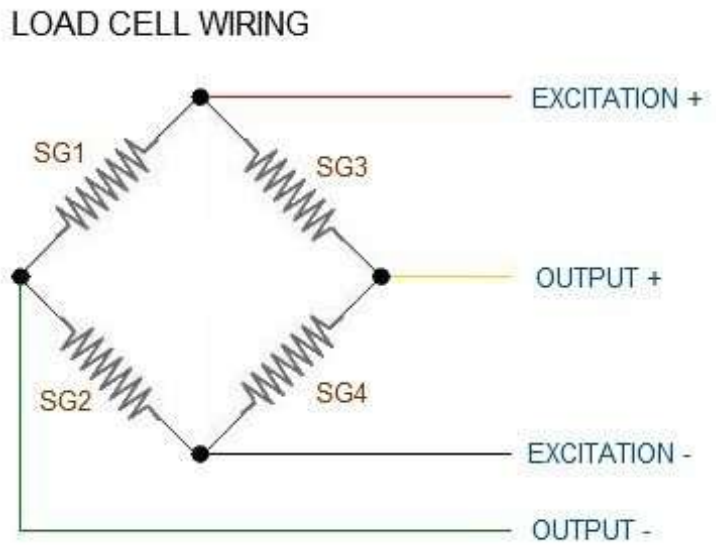


Figure 4.11: Load Cell Wheatstone Bridge

the weight applied on it. The HX711 uses a four wire connection, two for the interface communication, for clock and data, and two for power, for ground and voltage input. The interface pins can be connected to any of the GPIO pins of the ESP32 development board and there are various libraries available for the microcontroller that allow for calibrating and testing the readings from the load cell.

The HX711 requires an initial calibration in order to accurately measure the weight reading from the loadcell. In an initial exploratory phase the HX711 was connected to an Arduino UNO R3 development board in order to test the HX771, with the loadcell attached, using libraries provided by the Arduino IDE and obtain its calibration factor. That calibration factor was in turn used in the sub-sequential developments of the microcontrollers firmware that the HX711 would be connected to.

Calibration was done using calibrated weights of various lengths in order to validate weight detection accuracy.

4.5.3.5 MPU6050

The MPU6050 is an integrated circuit that provides a high precision gyroscope and an accelerometer sensor, alongside a temperature sensor that can measure readings between -40 e +85 degrees according to its datasheet (InvenSense, 2013).

The MPU6050 communicates with the microcontroller by means of a I2C interface, using the SCL e SDA pins. It also provides a secondary I2C interface that allows the connection to other external sensors, like a magnetometer, in order to create a full guidance system, however in the setup those pins are not used at the time. It also uses two pins for power, ground and voltage input, that are attached to the 5v provided by the power supply.

The purpose of using this sensor in the setup is to be able to measure orientation and velocity of the pedal in order to integrate those values in the power measurement

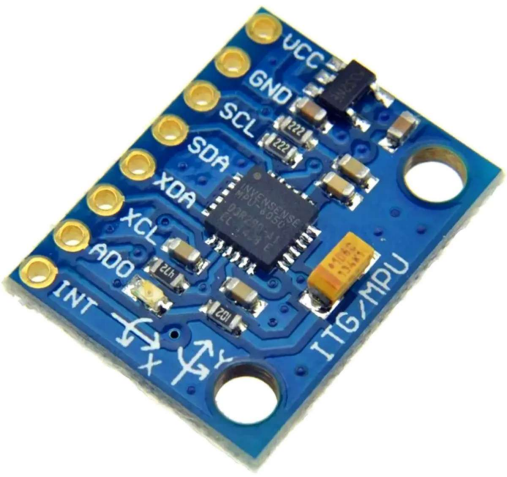


Figure 4.12: MPU6050 Gyro + Accelerometer

calculations.

According to Goethel et al., 2018 using a triaxial accelerometer, like the one present on this circuit, it is possible to develop an algorithm that can calculate the angle of the pedal on the bicycle, as shown in figure 4.13, based upon the Y and Z axis.

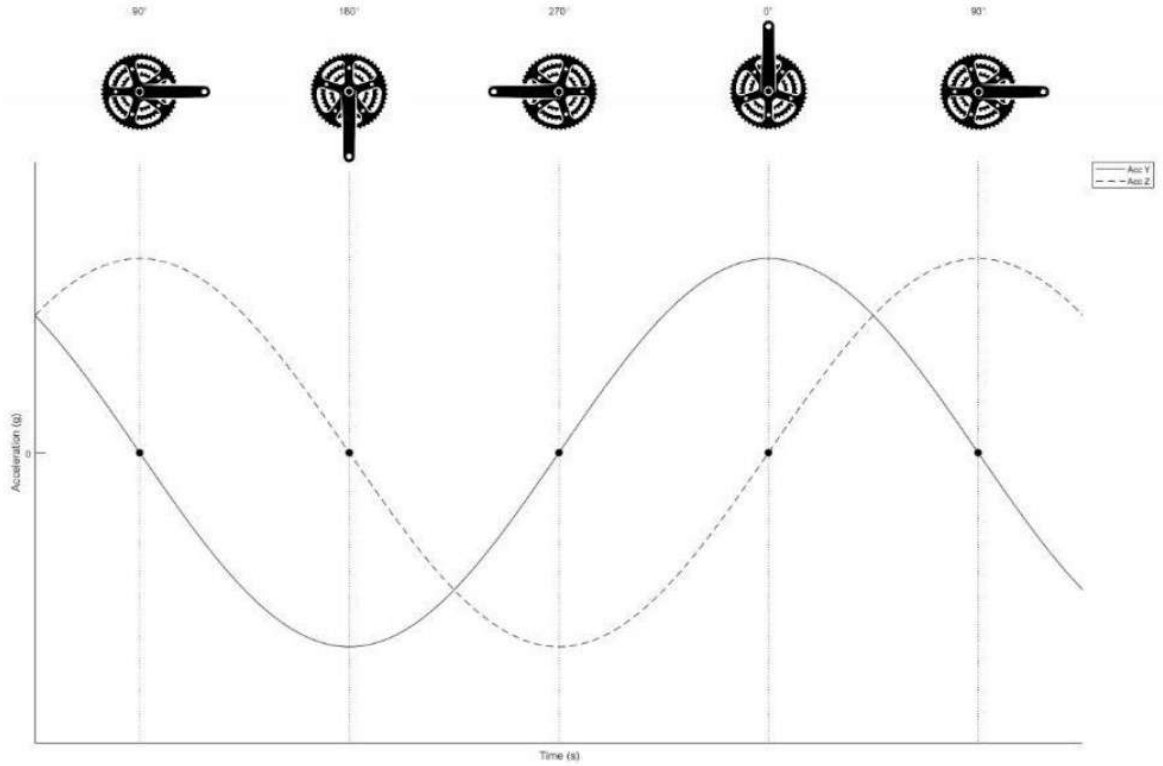


Figure 4.13: Pedal Position Identification (Goethel et al., 2018)

Knowing the crank angle is important for the calculation of the power values since this allows the cadence of the pedal rotation to be determined.

4.5.3.6 Power Supply

The power used by the components is supplied by a 3xAA battery holder that provide an average of 4.5v to the circuit. Since the components either needs an regulated 3.3v power supply or a 5v one, a small DC to DC voltage regulator step up boost converter board was used to input the provided voltage from the battery holder and boost the output up to correct 5v needed by the circuit.

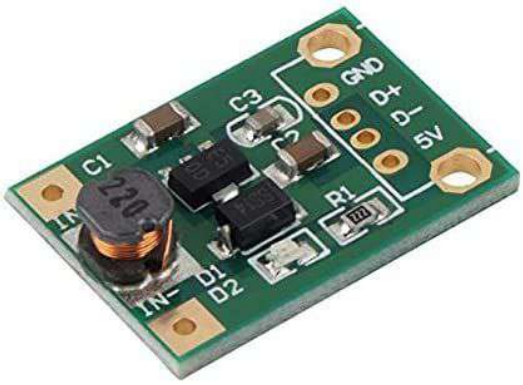


Figure 4.14: DC to DC Voltage Regulator

4.5.3.7 Circuit Diagram

As shown in figure 4.15 a circuit was designed in order to correctly attach each of the components to the proper pins of the microcontroller and power supply. After the circuit design a prototype board was assembled in order to enable its attachment, with all the necessary components, to the bicycles pedal.

The circuit also includes two LEDs, a green LED used to indicate when a Bluetooth client connection is made to the device and a yellow LED used to indicate calibration mode, and a push button to enable recalibration of the HX711 module. The loadcell is not included in the diagram since its externally connected thru a 4 pin socket.

After the prototype board was assembled, and confirmation was made that all the components were working as expected, the board was attached to the bicycle pedal so that the necessary firmware for the custom pedal power meter could be developed. The prototype board with all the necessary components attached is shown in figure 4.16.

4.6 Summary

After an initial analysis of the major features required to build the rehabilitation platform, this analysis was confirmed after consulting an expert in the field, a head nurse from the rehabilitation department of the Coimbra Hospital and Universitary Center.

That interview served as a confirmation of the initial identified features as well as the proposal to include others that were not part of the initial planning, and although

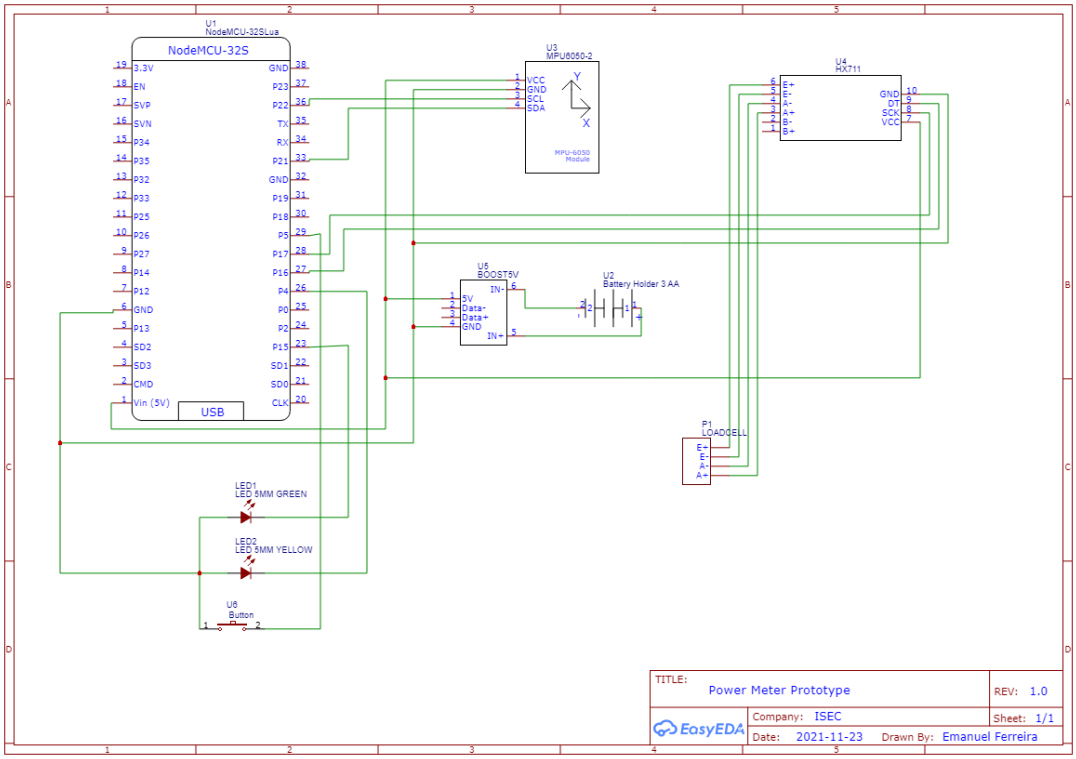


Figure 4.15: Power Pedal Circuit

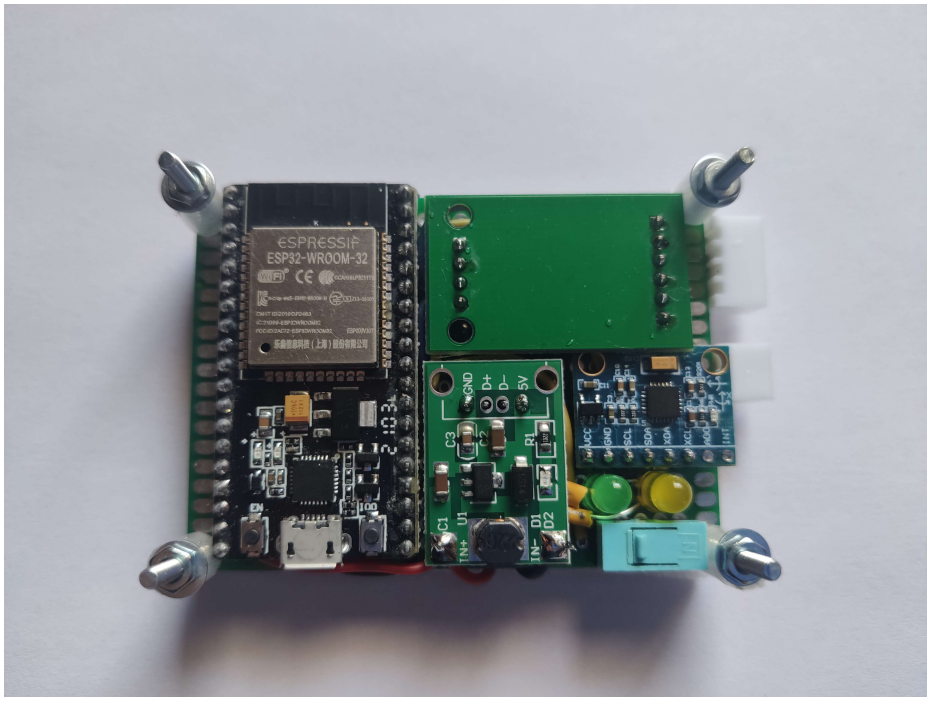


Figure 4.16: Power Pedal Prototype Board

some of those features could not be included at the time, due to lack of hardware available, the ones that were within reach were implemented.

In section 4.4 an overview of the various modules that make up the projects solution was presented, detailing their purpose and how they interact with each other. Also explained were the main features that are part of each module and how their methods

were designed using external tools such as nRF Connect.

In section 4.5 the sensors that are part of the platform were detailed and details were given about the design and construction of the custom power pedal, an essential part of the project since it is this device that will allow the measurement of the force applied by the users' lower limbs.

Moving forward in chapter 5 the software development phase of the project will be detailed which includes all the backoffice applications developed and the firmware developed for the custom power pedal, as well as detailing the libraries and other tools and technologies used during development, and the implementation of all the hardware for configuring the stationary bike.

Chapter 5

Backoffice Applications

5.1 Introduction

This section will present the steps taken to develop both the backoffice applications and the motivational user interface.

For the front-end applications, it was decided to use Angular as a single page development framework in order to optimize development and have a responsive layout.

As for the back-end applications, since tests were previously performed using Microsoft Bluetooth framework, it was decided to develop them using the .Net Core 3.1 framework.

The next sections will give an overview of the processes used in developing the various modules that comprise the projects solution, explaining the frameworks used, libraries and other relevant decisions that were taken during the development phase.

5.2 Angular Front-end Application

The Angular development platform, not to be confused with AngularJS, is a typescript development environment that has a large collection of libraries that handle various features so that it is possible to build fast, responsive and optimize applications, being enterprise size projects or small sized projects (Angular, 2021a).

The main building blocks of angular applications are called components, which consists of a HTML template for the page, a typescript class that defines what happens on the page, and a CSS style for the template.

All components were created using Angular CLI, which is a command line tool used to initialize, create and maintain angular projects. This tool allows easy generation of new components, modules, workspaces and so on, while maintaining the correct folder

structure and its dependencies. The Angular project uses material design components and is based upon the metronic theme (KeenThemes, 2021).

Material design is a design system made by Google that follows set of rules and guidelines in order to provide high quality and pleasant looking components across all platforms. It provides a large amount of readily available design components that are easy to use and facilitate the building of reactive and elegant single page applications (Angular, 2021b).

Navigation between components is called routing, and is enabled by using the angular route module, which in turn enables angular to travel between components. By using the angular route module methods, are also allowed to pass information between components, access to query parameters from the URL, and preventing unauthorized access to components by using a route guard.

The next sections will detail the development of each of the single page applications that are part of the front-end modules. All libraries detailed in the next sections were installed using Node Package Manager (NPM) that is a part of the Node.JS application. NPM is a tool designed to easily install or update project libraries and dependencies by using simple command line instructions, that are readily available from a central repository (NPM, 2021).

5.2.1 User Interface Module

The user interface module is to be used as the principal motivator behind the rehabilitation's session. Its goal is to provide stimulus to the user of the rehabilitation platform by allowing him to navigate a real life location in accordance to the cadence readings of the bicycles pedals, while also recording data from the other sensors in use.

Each user interface is directly associated to a Display configured in the backoffice module, and it's that Display object that determines the sensors that will be in use, the Display's name and unique identifier.

The display configured is later associated with a session that will detail its user and rehabilitation settings.

The user interface is composed of a component that receives a Display identifier as an input value and retrieves from the database all the data related to a Display that shares the unique identifier, if it exists. The Display identifier is obtained from the URL query as shown below, using the angular route module.

```
1 ngOnInit(): void {
2 this.sub = this.route.params.subscribe(params => {
3   this.displayId = params['displayId'];
4 });
5 this.requestDisplayData(+this.displayId);
6 ...
```

Listing 5.1: Routing Parameters

The ngOnInit function is a callback method, part of the angular core module, that is used to initialize the component before the page is loaded and is invoked after angular sets the components input properties.

After obtaining the Display identifier from the URL, its value will be passed to a function that will request the display data from the database, in order to update the contents of the page.

During the component initialization, a connection to the signalR Hub was also established, detailed in section 5.3.2, in order to receive push notifications from the server. Those push notifications will allow the user interface to update data in real-time regarding the sensors or changes to the session.

The connection to the signalR Hub is done using a library made available by Microsoft for Angular called @microsoft/signalR. This library enables the configuration of the connection properties and establish a connection to the signalR Hub server, while also assuring that the connection is automatically reconnected in case the connection is dropped.

```
public startConnectionDisplay(displayId: number) {
    this.hubConnection = new signalR.HubConnectionBuilder()
                         .withUrl(this.HUB_URL + '?displayId=' + displayId)
                         .withAutomaticReconnect()
                         .build();
    this.hubConnection
  .start()
  .then(() => console.log('Display Connection started'))
  .catch(err => console.log('Error starting display connection: ' + err))
  }
```

Listing 5.2: SignalR Hub Connection

After successfully establishing a connection to the signalR Hub, a registration is made to receive push notifications for each of the sensors and for new data for the display, in case any changes are made on the server side.

This is done by registering a handler that will be invoked when the hub method with the same name is invoked.

The example below shows the "hubConnection.on"method registering the handler, while the "getSensorCadenceData"parameter indicates the name of the hub method to be listened to.

After the handler is successfully triggered, the newly received data is stored and a change detection method is triggered to inform the component page that new data is available.

```
public addCadenceSensorDataListener() {
    this.hubConnection.on('getSensorCadenceData', (data) => {
      this.cadenceData = data;
      this.cadenceDataChange.next(this.cadenceData);
      console.log('Cadence Data: ', data);
```

```
6       });
7   }
```

Listing 5.3: SignalR Connection Handler

The video portion of the user interface component is handled by the @videogular/ngx-videogular library, which is a HTML5 video player wrapper for Angular, that provides an API service from where all aspects of video playback can be controlled.

The video to play during the rehabilitation session will be derived from the session data. After the UI receives a new push notification from the server indicating that the display data has changed a UI component method requests new session information from the server using the Database API.

The component will also use the session data to populate all the information that composes the user interface such as the patients name and picture, update session and display status and connect to the sensors that are enabled for said session as shown in figure 5.1.

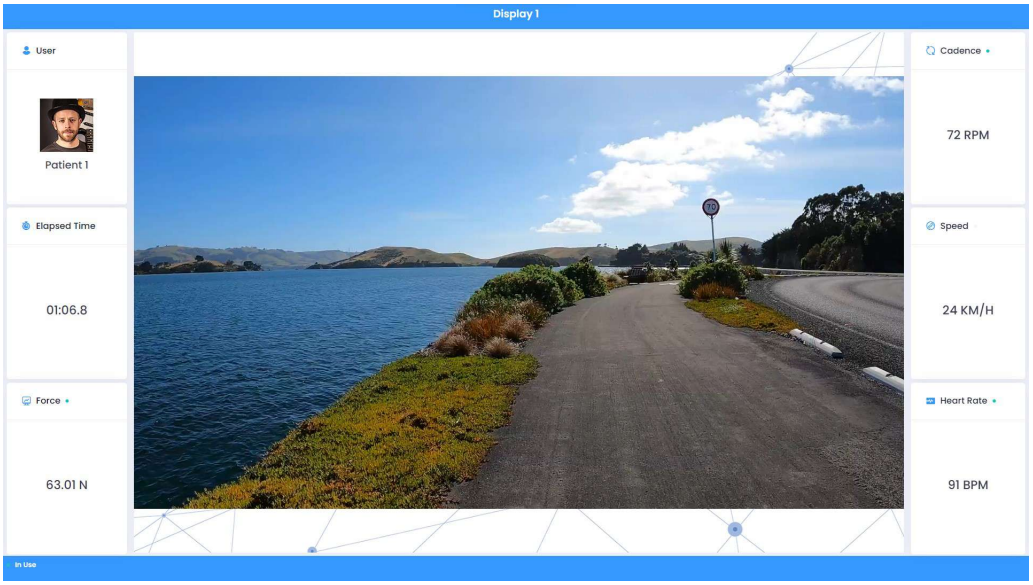


Figure 5.1: User Interface

On the left side is the patient that will be part of the session, below is a countdown timer that starts when the session is started, and at the bottom is the power sensor data.

On the right side is the cadence sensor data, followed by the speed sensor data, and at the bottom is the heart rate sensor data.

The video player, presented at the center of the user interface, is only presented if the session that was received from the server has already been started. If not it will only inform the user to wait for the session to start.

Following the start of the session by the supervisor, the user is informed to start the workout to begin the rehabilitation session, while the workout starting is directly dependant on the cadence sensor data.

When a new cadence value has been detected by the user interface, the video will start playing, and the speed of the video will be adjusted according to the values presented in table 5.1.

Video Speed	Cadence Minimum	Cadence Maximum
0x	0	0
0.30x	1	24
0.70x	25	49
0.90x	50	99
1x	100	-

Table 5.1: Cadence to Video Speed Values

5.2.2 Backoffice Module

The backoffice module is comprised of all the components that make up the backoffice application. Each of these components exist in order to manage some part of the applications data structure.

The next sections will detail the purpose of those components and how they are linked together.

5.2.2.1 Patient Management

The patient management component provides the supervisor the means to add, update or remove existing patients from the database, as shown in figure 5.2.

Upon initialization the patient management component requests from the Database API the list of the available patients. A material design table lists the patients showing its picture, name, gender, date of birth, contact and other relevant information.

The are two ways to filter the patients list, either by the patients name, using the search box provided, or by status using a combo box.

A paginator and sorting service was also created in order to allow the user to search thru pages of patients, if the count is above a threshold value, and to allow sorting the tables columns by name, date of birth or other existing table parameters.

The sorting function receives the column as a parameter order the data server side according to ascending or descending order.

The paginator service is also requested server side, and the correct paged list is returned from the server.

5.2.2.2 Supervisor Management

The supervisor management component provides the supervisor the means to manage user logins of other supervisors that will use the backoffice application.

The component allows supervisors to create, update or remove existing supervisors, update their login details, as well as managing its personal data, as shown in figure 5.3.

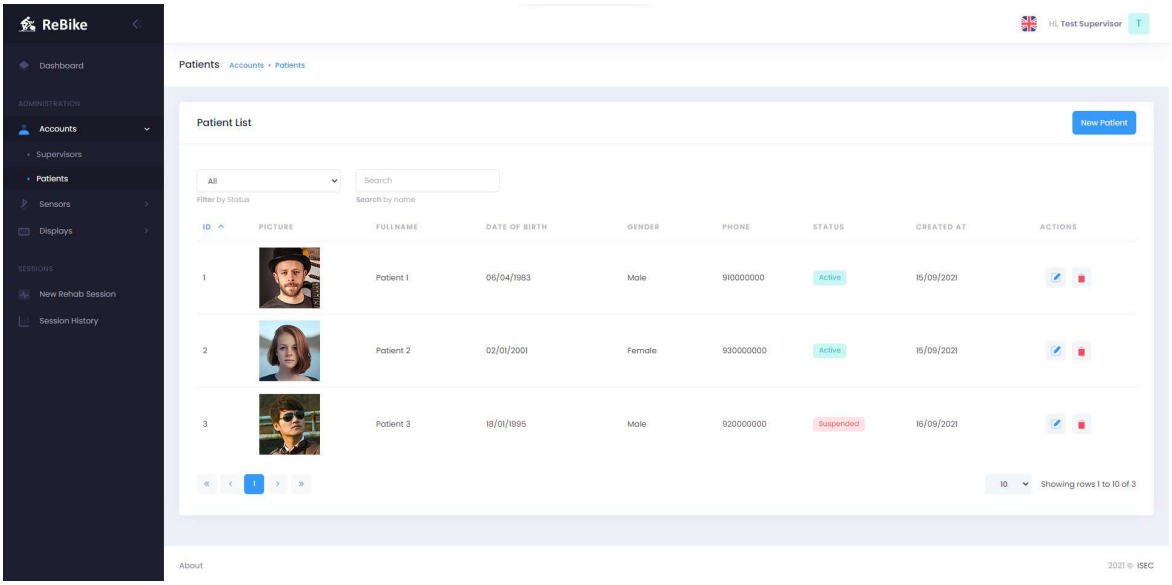


Figure 5.2: Patient Management Screen

As the previous component the supervisor page also uses a material design table to list the existing supervisors, which is populated during initialization of the component, and the table also add ways to filter the supervisors list by name and status.

The paginator and sorting service created for the patient management component is also present and handles requests on the server side.

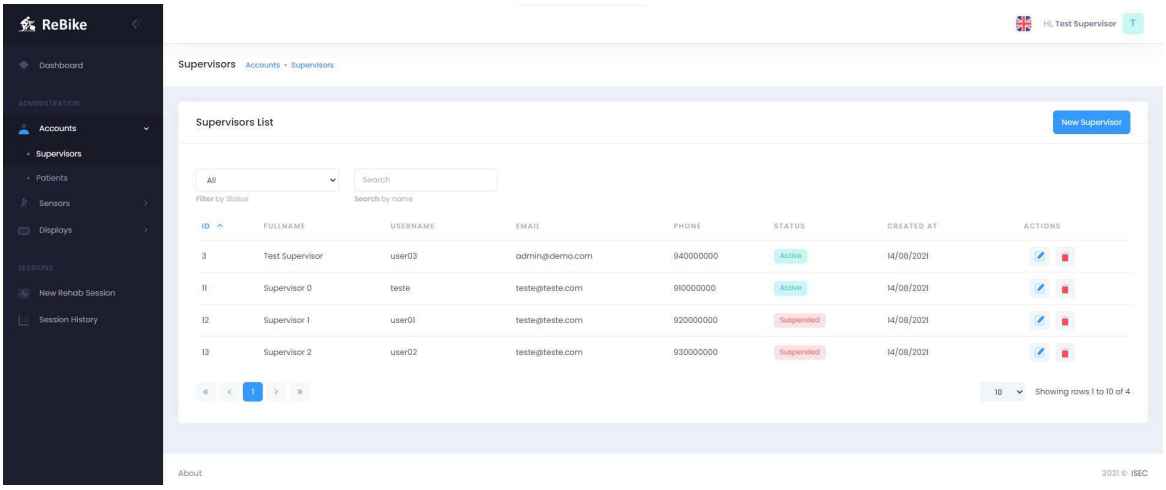


Figure 5.3: Supervisor Management Screen

5.2.2.3 Sensor Management

Sensor management is divided between two component pages. One that allows the supervisor to scan for Bluetooth LE devices in range, and add them to the list of available sensors if needed, and another that manages sensors available.

The scan component allows the supervisor to scan for new devices and any Bluetooth LE device will be added to the list of available devices. The component uses methods from the Sensor API as its service for all of its functions.

The supervisor can then pick a device from the list and choose to add them to the database of available sensors for future use in the application, if it still not exists, as shown in figure 5.4.

The page contains a button to start and stop the scan, that calls those methods from the Sensor API service accordingly. After the scan is started and in progress the component will activate a timer that will request the scanned sensor list from the server every 5 seconds until the scan is stopped. If the supervisor decides to add a new sensor

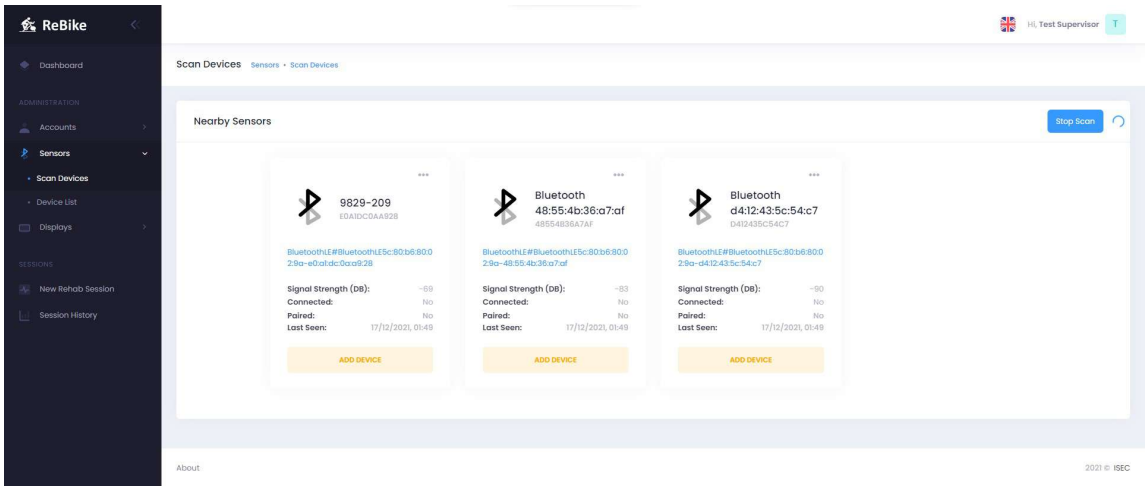


Figure 5.4: Sensor Scanning

to the database, and it does not already exist, the component will redirect the view to the list of available sensors component, as shown in figure 5.5.

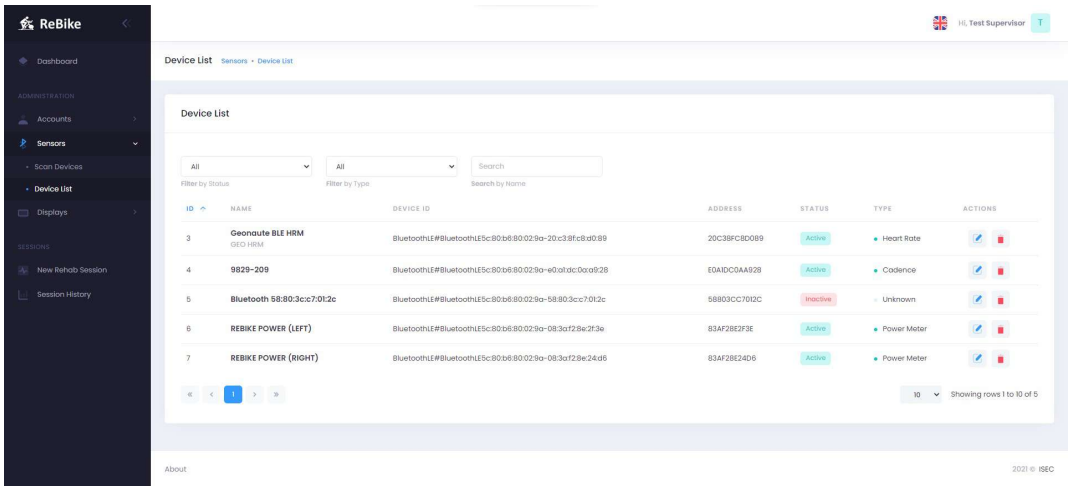


Figure 5.5: Available Sensor List

The list of available sensors lets the supervisor edit the sensors or remove them from use.

A friendly name can be added to an available sensor so that it's easily identified when used throughout the application. If the friendly name is present it will be used instead of the sensor's default name.

The sensor can be disabled if temporarily not present in order for it to not be available for selection where used.

5.2.2.4 Display Management

The display management page allows the supervisor to add new displays, that will make use of the user interface module, and manage existing ones.

After the display is properly configured it can then be associated to a newly created session, that will allow the supervisor, after properly creating the session, to control the session's status and therefore change was is currently happening in the display.

The displays connection can assume various statuses depending on its current activity, a list of possible status is shown in table 5.2.

Status	Description
Disconnected	Display is not connected to the server
Connected	Display is connected and waiting session
Standby	Display is connected and waiting session to start
In Use	Display is connected and in session
Error	Display connection error as occurred

Table 5.2: Display Connection Status

Each sensor is directly associated with one display, in order to not allow the use of a sensor on multiple locations at the same time.

The user can add a name and description that best defines the display, and disable it if needed, as shown in figure 5.6.

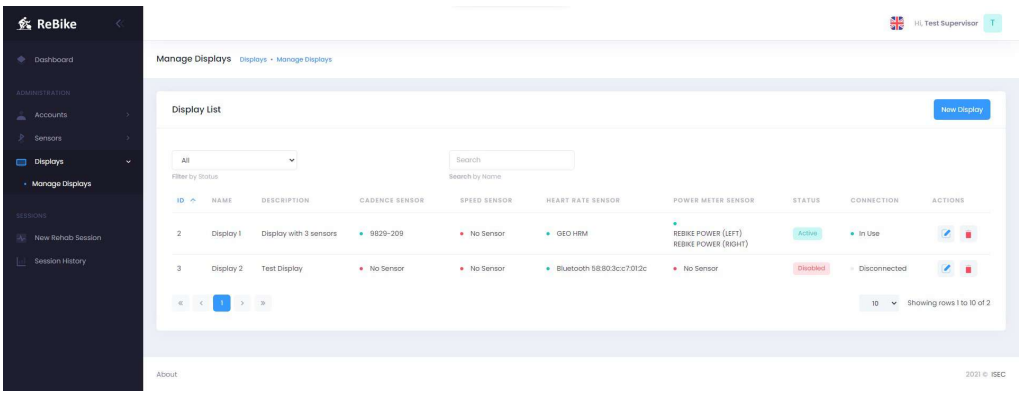


Figure 5.6: Display Management Screen

5.2.2.5 Session Management

The session management is divided between two components, one for creating a new session and another for managing existing sessions, active or otherwise.

The new session component uses a material design linear stepper that provides a wizard like workflow in order to divide the session creation into easy to understand steps, as shown in figure 5.7.

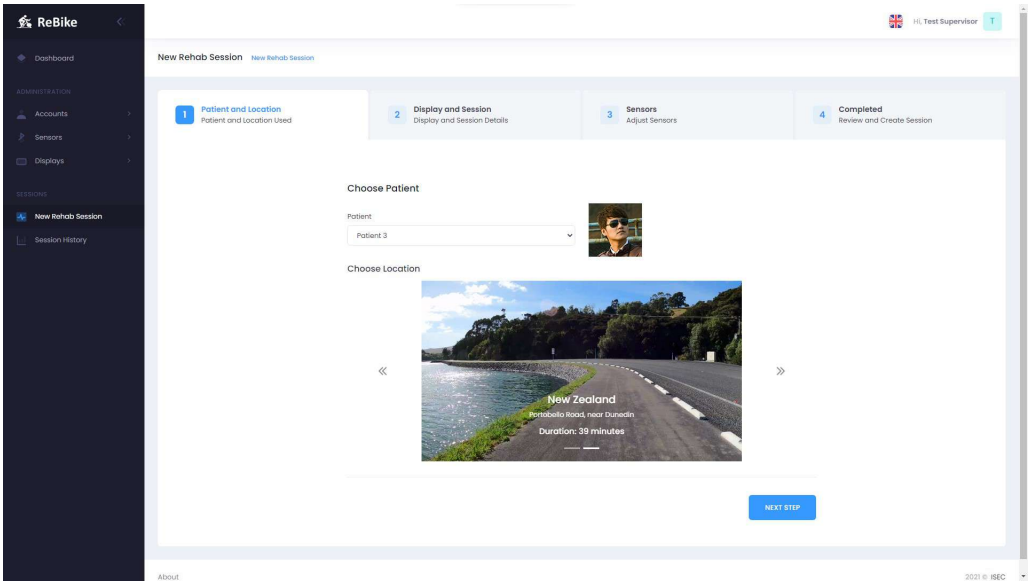


Figure 5.7: New Session Wizard

The first step allows the supervisor to choose the patient that will be attending the rehabilitation session and the workout video to display during said session. Only patients that aren't already enrolled in an active session appear in the selection box, and active is meant that the session exists for this patient and hasn't been ended or archived.

The second step allows the supervisor to choose which display will be used for the rehabilitation session, write a small description about the session and input its duration. The duration of the session cannot be more than the duration chosen in the previous step.

Contrary to the previous step, where the patient cannot be chosen from the list if it already exists in an active session, the display of the session can be selected even if a session for that display already exists, to allow the supervisor to create all the daily sessions for its patients before hand. The display is later protected from multiple uses by only allowing one session to be started at once.

The third step is related to the sensors and allows the supervisor to fine tune the sensors that are attached to the display. This can range from disabling a sensor from recording, or adjust configuration values for the sensors.

The last step is only a review step for the supervisor to confirm all the inputted data before the session is created and stored in the database.

After a session is successfully created the component redirects the page to the session history component for the supervisor to be able to act on it.

The session history component will list all the active and archived sessions for the supervisor to manage, in case of active sessions, or analyse the ones that have already ended, as shown in figure 5.8.

The table displays columns for the session's start and end time, the session's description, its duration, current status, patient name, display name and display connection

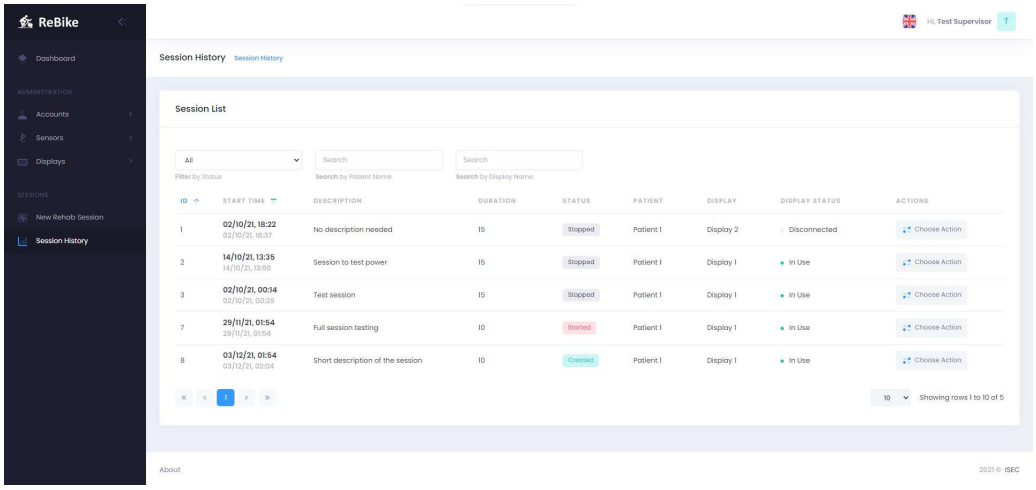


Figure 5.8: Session History

status.

The display connection list of possible status was detailed in section 5.2.2.4, while the possible status for the session are listed in table 5.3.

Status	Description
Created	A new session was created and is ready to start
Started	The session was started and is waiting display initialization
In Session	The rehabilitation session is in use
Stopped	The session has ended or was stopped
Ended	The session has ended and cannot be changed

Table 5.3: Session Status

The session when started waits from input from the display that the session was assigned to. If indication is received that the workout has been initialized the status of the session changes to "In Session", and only stops when the duration as expired or if stopped manually by the supervisor.

A stopped session can be restarted, erasing all the previous data recorded and changing its status to "Started", or ended if the user wants to archive the session and store all data, thus ending the session and freeing the patient for new sessions.

The session list allows the supervisor to filter the list by patient name, display name and session status. it also supports column sorting and pagination, as with previously detailed lists.

While the session is in session, stooped or ended a button is available for the supervisor to view and analyse data gathered during the session, called session detail.

The session detail component shows data from the session, alongside patient details, the session description, a placeholder for taking notes during the session and data visualization in the form of charts.

The chart visualization used in the session detail component uses the @amcharts/amcharts4 library for angular. This library made it possible to build a fast and flexible way to visualize the data, while providing access to multiple types of charts, with

various useful functions like, zoom, sectioning, range sliders and data extraction, as requested in a previous chapter.

This provides the supervisor a way to easily visualize the data acquired from the sensors, during and after the session, in order to best manage the rehabilitation process.

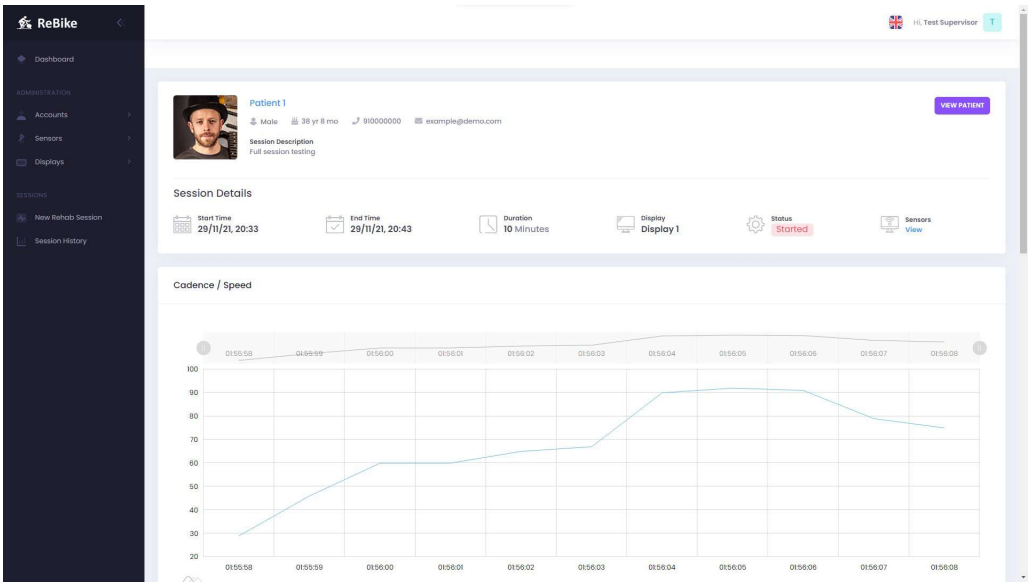


Figure 5.9: Session Detail

5.3 Back-end Applications

The back-end applications, developed using the .Net Core framework, are a set of services to be used by the front-end applications.

5.3.1 Sensor API

In this section will be describe the Windows API Runtime methods that were used to develop the API, and the difficulties encountered.

While the development was based on the Microsoft Docs (formerly MSDN) documentation and examples, which were only focused on developing UI applications based on the UWP platform, there was no reference on how to use it on the .Net Core framework other than a release notice of the available Nuget legacy package.

Using the project basis described in a previous topic, the development was divided based on the following topics, considering the methods provided by the Runtime API:

- Device Discovery and Connectivity
- Listing Services and Characteristics available
- Mapping UUID with Bluetooth SIG descriptions
- Reading Values and Data Conversion

The discovery process was quite simple to implement, with only a few details changed from the original documentation to accommodate the fact that a synchronous operation was being performed on top of an asynchronous discovery process in the web API.

Since there was a need to control the start of scanning, results callback and stopping, a singleton was created globally, called "watcher", to store the object between API requests, the object was then declared and the corresponding EventHandler in which all results will be stored, for later retrieval.

```
1 watcher = new BluetoothLEAdvertisementWatcher();
2 watcher.Received += WatcherOnReceived;
```

Listing 5.4: AdvertisementWatcher Sample

Each time a new device is detected, the callback function receives the event arguments, from which the Bluetooth address of the newly discovered device can be retrieved. Then by use of the "FromBluetoothAddressAsync" method, an object can be populated with more detailed information, such as its name, if it is already paired, or the signal strength.

Although it is known that using a singleton goes against the good practices of the restful being stateless, there is a believe that it was a necessary step, and solutions to this problem will be addressed in future versions.

After achieving discovery of the nearby device's, the obtained device identification was used to retrieve the devices' services by using the "GetGattServicesAsync" method from a previously populated object with the device information.

```
1 var device_info = await BluetoothLEDevice.FromIdAsync(deviceId);
2 ...
3 var gatt_services = await device_info.GetGattServicesAsync();
```

Listing 5.5: Service Gathering Sample

After obtaining a list of services, a start was made to populate each of them with its corresponding list of available characteristics, but initially encountered an issue with some deprecated methods being used in the provided examples, such as "GetAllCharacteristics". A corresponding async method is provided instead "GetCharacteristicsAsync".

Before the retrieval operations it is also important to verify if access to the device is available by using the "RequestAccessAsync" method.

Since the device only provides the services and characteristics by means of an identifying UUID, a mapping mechanism was devised based on the provided information from the Bluetooth SIG documentation.

The Bluetooth SIG used to provide those values in a practical XML format but that has since been discontinued and is now provided in a pdf file (Bluetooth, 2021a).

An example of the output of a characteristic, correctly mapped to the corresponding descriptions and names, is shown in listing 5.6, in JSON format.

```
{
  "service": {
    "device": {
      "broadcastTime": "2021-09-14T16:26:48.1910888+00:00",
      "address": 246985786042664,
      "name": "9829-209",
      "signalStrengthInDB": -59,
      "connected": false,
      "canPair": true,
      "paired": false,
      "deviceId": "BluetoothLE#BluetoothLE5c:80:b6:80:02:9a-e0:a1:dc:0a:a9
          :28"
    },
    "uuid": "00001816-0000-1000-8000-00805f9b34fb",
    "name": "Cycling Speed and Cadence",
    "description": "This service exposes speed-related and cadence-related
        data from a Cycling Speed and Cadence sensor intended for fitness
        applications."
  },
  "uuid": "00002a5b-0000-1000-8000-00805f9b34fb",
  "properties": 16,
  "protectionLevel": 0,
  "attributeHandle": 25,
  "name": "CSC Measurement",
  "description": "The CSC Measurement characteristic is used to send speed-
     related data and/or cadence-related data.",
  "readDisplayFormat": 2,
  "writeDisplayFormat": 1,
  "notificationDisplayFormat": 2
},
...
```

Listing 5.6: Characteristics JSON Sample

After gathering all the device's services and characteristics a method was implemented to read a value from a given characteristic uuid, in this case the heart rate measurement.

That particular method receives as its input values the device identification, the service uuid and the characteristic uuid, and returns the correctly formatted value.

According to the Bluetooth SIG specification there are multiple types of data-transfers as shown in figure 5.10. In the example above, of the CSC Measurement, it will be of a Notify type since the device will inform the client each time it has a new value.

Initially for development purposes this asynchronous method was converted to a synchronous one with only one single value being returned, due to limitations of the Restful API in the initial proof of concepts.

However, as explained in section 5.3.2, an alternative was found for the notifications

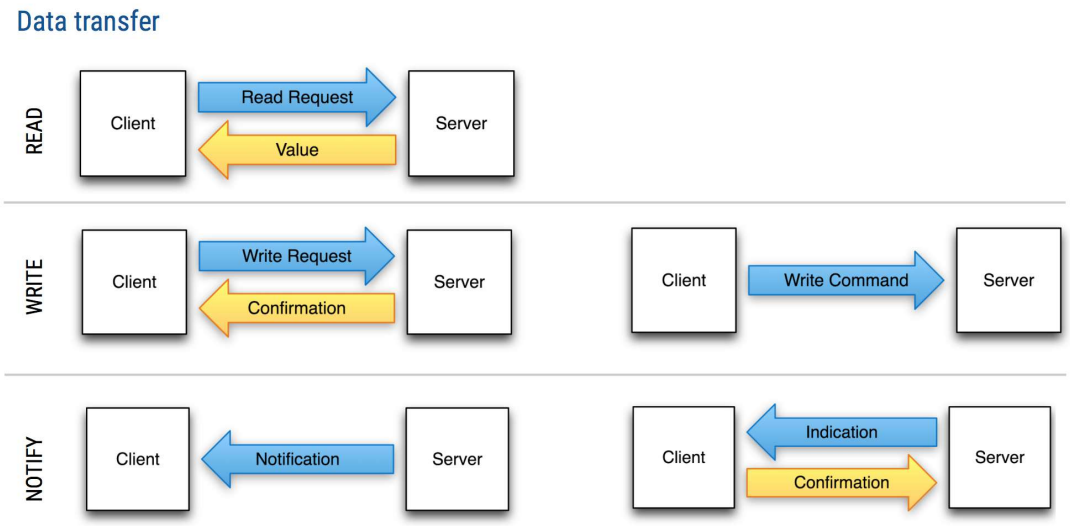


Figure 5.10: GATT Data Transfer Types

by using the signalR library, which enabled the power to push notifications to the clients as new data became available.

The resulting values obtained from the API methods follows the format specified in the corresponding XML file from the Bluetooth SIG specification and needs to be adjusted accordingly.

Further details on how to handle these values, properties and descriptors is described in the Microsoft and Bluetooth SIG documentation and examples. Each service registered in the Bluetooth SIG documentation has a set of service and profile documents that act as guidelines for developing client and server applications for that particular service.

5.3.2 SignalR Hub

The implementation of the signalR framework in the project, as previously explained in section 4.4.4, came from the necessity of having a way for the back-end applications to push information in real-time to the front-end applications as it becomes available without having to wait for new requests.

The signalR library, using websockets, allows for clients to make a permanent connection to the server, contrary to the http connections to web services, like the Database API, that have a short lifespan and require reconnection each time a request is made.

This works similar to a chat room, where messages can be pushed to all the connected clients or send messages to specific ones, while monitoring the clients connection.

As shown in listing 5.7 each time a Display connects to the signalR Hub an override is made to the method provided by the library in order to add more features to the connection method.

In this way the Display is added to a list of connected displays and its corresponding connection id, the information that the Display is now connected is stored in the database, and is broadcast to the newly connected client that it can update its displayed data.

This will make the connected display, as shown in section 5.2.1, which previously subscribed to the "requestUpdateDisplayData"method, receive the broadcast message that tells it that it must request new display data from the server.

It will also send a broadcast message to all connected Backoffice clients that subscribed to the "requestUpdateTables"method, which display changed data, and that they should update the table data, if they are viewing that particular component that subscribed to this method, to match the changes made to the database.

```
1  public override Task OnConnectedAsync()
2  {
3      var displayId = Context.GetHttpContext().Request.Query["displayId"];
4      if(Convert.ToInt32(displayId) > 0)
5      {
6          _connections.Add(displayId, Context.ConnectionId);
7          var display = _databaseService.UpdateDisplayStatus(Convert.ToInt32(
               displayId), "connected", Context.ConnectionId).Result;
8          Clients.Client(Context.ConnectionId).SendAsync("
               requestUpdateDisplayData", display);
9          Clients.All.SendAsync("requestUpdateTables");
10     }
11     return base.OnConnectedAsync();
12 }
```

Listing 5.7: SignalR Connection

If the client disconnects for some reason from the Hub, that library method is also overridden, removing the Display from the previous list, updating its status to disconnected in the database and broadcasting that update to the connected Backoffice clients using, as before, the "requestUpdateTables"method.

Similar methods were also created in the signalR Hub to have full control over the Session being used on any given Display in real-time, such as notifying the Display that the supervisor as started a particular Session, as shown in listing 5.8, that the Session was stopped, restarted or ended.

```
1  [HubMethodName("StartSession")]
2  public Task StartSession(int displayId)
3  {
4      var display = _databaseService.UpdateDisplayStatus(Convert.ToInt32(
           displayId), "sessionstart", null).Result;
5
6      Clients.All.SendAsync("requestUpdateTables");
7      return Clients.Client(display.ConnectionId).SendAsync("
           requestUpdateDisplayData", display);
```

```
8 }
```

Listing 5.8: SignalR Start Session

Every time the Display receives those subscribed messages it acts accordingly, requesting new Session data from the database for it to display the necessary information on screen and start the rehabilitation Session, stopping the Session while displaying a message informing the user or ending the Session displaying a message that the display is waiting for a new Session to be started.

The most important part of the signalR Hub is the broadcast of sensor data in real-time to the Display, since that was its primary goal when its development was decided.

After the Display receives the message from the signalR Hub informing it that there is a new Session for it, it will gather from the Session information which sensors, that are part of the Display, it must subscribe to.

For each subscribed sensor, it will then register an event handler method for the selected feature that will trigger each time there is new data available from the subscribed sensor. That event handler method will then process the new data in its proper format, according to the Bluetooth SIG specification, broadcast the value to the connected Display and store the newly formatted value in the database.

In the eventual case the Display disconnects from the signalR Hub during a session, then remove the event handler and terminate the data storage procedure in order to prevent it from continuously flooding the database with new data without the client being active.

5.3.3 Database API

The developed Database API is a Restfull API that provides methods for applications to use HTTP requests to access and use the database data, providing the front-end applications access to the database.

This type of service was decided because of compatibility of REST with the front-end Angular applications.

The API makes use of the Microsoft Entity Framework Core library to provide data access using a context object that is declared upon the API initialization.

The Entity Framework Core library provides compatibility with multiple database providers including the one used by these PostgreSQL applications, using the "Npgsql EntityFrameworkCore PostgreSQL"package that is available via the Nuget package manager in the IDE.

A controller was created for each of the existing tables, and in each there are methods to create, update, and delete data.

Since the front-end table functions, like sorting, filtering, paginating and search terms are made server side, a single method serves the table for those purposes. This

was done to optimize performance since in this way the database retrieves only the requested necessary data from the server.

5.4 PostgreSQL Database

The database design followed the rules of a relational database by which each tables relates to at least one other. For the design of the database, by using the DbSchema tool along with the pgAdmin application for managing each tables data.

The users table is used to store data from supervisors, alongside login information, personal details and authentication tokens.

The patients table stores each patients personal data and picture, to be used in the rehabilitation sessions.

The patient measurements table stores the information about measurements taken from the patient at any given time. This provides information for the rehabilitation session of the patients evolution between sessions using numerous metrics like weight, body mass index and height.

The workout medias table stores the information relating to the media files used in the user interface front-end application. Those videos are shown and controlled by user during the rehabilitation session and chosen upon its creation.

The rehab sessions table stores all the information relating to a rehabilitation session, like its start and end time, duration and current status, and each session is directly linked to a patient, a supervisor, a display and a workout media.

The displays table stores the information regarding the various monitors used in rehabilitation sessions, and is directly linked to the sensors that will be used on that display.

The sensors table stores information regarding the available sensors, that can be used by each display, and each sensor is linked to a known type.

The sensor types table stores information regarding the numerous types of sensors available, being cadence, power, heart rate, speed or unknown type.

The sensor values table stores the information gathered by the sensors during a rehabilitation session, along with a timestamp of its acquisition date and time, and each sensor reading is linked to a sensor and a specific session.

The current database layout and its relationships, along with the descriptions of its tables and columns, is available in appendix B.

5.5 Summary

After a initial analysis of the tools and design of the project structure development was made using two separate integrated development environments that best encompassed the tasks at hand.

The frameworks and programming languages that were chosen were, after conducting a series of proofs of conceptual designs, the ones that have been considered most suitable for what is needed, in order to speed up development and at the same time provide a nice aesthetic, hence the reasoning for the choice of material and angular design for the front-end applications.

Although there were some issues encountered during the development of the back-end applications, mainly the Bluetooth part of the project, because of the design of Microsoft Bluetooth's framework mainly focusing on UWP applications, the end result worked as expected, although it would still needs some improvement for the sake of optimization.

In section 5.2 it details the development of these front-end applications in which it became possible for the supervisor to configure every aspect of the rehabilitation setup, providing ways to record as much data from the patients as needed, while also developing tools for the supervisor to analyse that data, compare it or extract it for further study.

Also described was the development of the APIs, the backbone of the backoffice application, such as the signalR Hub which provided the means to push data in real time, something that was essential due to the nature of data logging services and the need to have live vouchers being recorded in the user interface.

In section 5.4, the database structure was highlighted, while explaining the decisions made and the purpose of each table. In the next section the development steps of the custom power meter will be explained.

Chapter 6

Power Meter

6.1 Introduction

This section will detail the firmware that will be loaded into the microcontroller that is part of the custom power pedal setup.

The firmware part of the custom pedal development was made using the Arduino programming language in order to have access to the large number of libraries that are available for that environment which allowed a large number of isolated tests to be performed on every component, using the appropriate libraries, before building the final product.

Finally, the implementation of the various sensors on the stationary bike will be detailed, while explaining the tests done on the platform after it is assembled.

6.2 Custom Power Meter

The custom power meter is the principal sensor in the rehabilitation setup since it will be responsible to measure the force applied by the lower limbs of the user.

As shown in section 4.5.3 the software development part of this sensor was made for a ESP32 device, a microcontroller that's compatible with Arduino sketches and libraries.

A Arduino sketch, the name that Arduino uses for the code files, is where the coding of the functions that will interact with the external devices connected to the microcontroller is done, this code will be compiled to machine language that the Arduino compatible device can understand and uploaded to the ESP32 microcontroller.

The Arduino code is written in C++, alongside some special functions and methods. There are two mandatory functions that are a part of every Arduino sketch, which is setup() and loop().

The setup() function is used to configure the various libraries and do a initial setup of the code that will be run once the device is powered on.

The loop() function, as the name implies, is a function that after the initial setup function is executed, will be run in a loop and is where the main logic will need to be implemented.

As shown in the sample code of listing 6.1 it starts by defining the libraries needed for the program to interact with the external modules, such as the HX711 and the MPU6050. The microcontroller is also told which pins these modules, leds and buttons are attached to. The libraries that will enable to control the Bluetooth connection of the ESP32 are also added.

```
#include <Wire.h> // I2C Library
#include <math.h>
// Libraries for Bluetooth BLE
#include <BLEDevice.h>
#include <BLEServer.h>
#include <BLEUtils.h>
#include <BLE2902.h>
// Libraries for HX711
#include <HX711_ADC.h>
#if defined(ESP8266)|| defined(ESP32) || defined(AVR)
#include <EEPROM.h>
#endif
// Library MPU6050
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
// Pin Definitions
#define LOADCELL_DOUT_PIN 16
#define LOADCELL_SCK_PIN 17
#define INTERNAL_LED_PIN 2
#define YELLOW_LED_PIN 4
#define TAR_BUTTON_PIN 5
#define GREEN_LED_PIN 15
// BLE Services Definitions
#define cyclingPowerService BLEUUID((uint16_t)0x1818) // UUID From
    Bluetooth SIG
```

Listing 6.1: Arduino Sketch Sample

Although it is necessary to define the loadcell amplifier pins, for the MPU6050 it is connected to the SDA and SCL pins of the microcontroller, so there is no need to define them, since the library used assumes those pins by default. Also, the UUID of the Bluetooth service that will be implemented in the device, the Cycling Power Service, will be defined, according to its Bluetooth SIG specification.

As previously explained in section 4.5.3.2 there are 3 mandatory required characteristics that must be implemented in the Cycling Power service for it to be compatible with existing applications, so initially these characteristics were defined together with

the Cycling Power Vector since it is this optional characteristic that will allow the torque applied to the pedal to be read, as shown in listing 6.2, using their respective UUID's and properties.

```
1 // Defines BLE Characteristcs
2 BLECharacteristic cyclingPowerLocationCharacteristic(BLEUUID((uint16_t)0
      x2A5D), BLECharacteristic::PROPERTY_READ);
3 BLECharacteristic cyclingPowerMeasurementCharacteristic(BLEUUID((uint16_t)0
      x2A63), BLECharacteristic::PROPERTY_NOTIFY);
4 BLECharacteristic cyclingPowerVectorCharacteristic(BLEUUID((uint16_t)0x2A64
      ), BLECharacteristic::PROPERTY_NOTIFY);
5 BLECharacteristic cyclingPowerFeatureCharacteristic(BLEUUID((uint16_t)0
      x2A65), BLECharacteristic::PROPERTY_READ);
```

Listing 6.2: Cycling Power Service Characteristics

After detailing the libraries, inputs and outputs that are going to be used in the program, implementation of the Bluetooth LE connection of the device has begun.

For the initialization of the Bluetooth LE a separate function was made to initialize all the needed information for the Bluetoth LE, this included defining its name, that will appear when scanning for the device using proper tools, and initialization of BLE in server mode, as it will be acting as a service provider for other clients to connect.

After the initial initialization, the above service definition was used to initialize a new service using the "createService()"function present in the Arduino BLE library, which will return an object for the created service, so that the necessary characteristics can be added to the objects from the listing 6.2, using the "addCharacteristic()"function of the library.

After the assignment of each service and feature, the Bluetooth LE library service will be activated and the advertisement of the newly created service and features will start. The above Bluetooth LE initialization function is called during the initial setup procedure that will be described in the next section.

6.2.1 Initial Setup

The setup function is one of the main methods of the Arduino sketch, being the one that provides the initialization of all the libraries and components that make part of the application code.

During this initial setup one starts by initializing the loadcell amplifier library and defining its calibration value, that was obtained in an early stage of testing. This calibration value, known as the calibration factor, was taken into account the variations in the load cell to give the weight accurately applied to the load cell. An initial tare reading of the load cell was also taken in order to ensure that the initial applied force is close to 0, without any weight applied to the pedal.

After initializing the HX711, it started by initializing the Bluetooth LE service, using the function described above, and also the MPU6050.

Upon exiting the setup function the program moves to the main program logic that will run continuously in a loop.

During the initialization the static features that are part of the Power Cycling service were also configured, as described in the following sections. Another service has also been initialized, the Device Information Service, that provides potential clients with useful information regarding the device as shown in table 6.1.

Characteristic	Requirement	Properties
Manufacturer Name String	Optional	Read
Model Number String	Optional	Read
Serial Number String	Optional	Read
Hardware Revision String Point	Optional	Read
Firmware Revision String	Optional	Read
Software Revision String	Optional	Read

Table 6.1: Device Information Service

6.2.1.1 Cycling Power Feature

According to its specification the Cycling Power Feature is a characteristic that is used to described the supported features of the BLE Server.

The value of this characteristic is of a static type, meaning it cannot be changed by the client, and is defined by a set of bits where each bit, when set to 1, matches a supported feature. It also a few bits reserved for future use and that are always set to 0.

The size of the characteristic is 4 bytes, corresponding to a bit map of 32 bits, each corresponding to a possible feature. The table of feature bits description and location is detailed in the GATT Specification Supplement documentation (Bluetooth, 2021b).

In the development a static 4 byte variable was initialized with only bit 2 and 3 are set to 1 while all the other bits are set to 0. The bit number 2 indicates the support for Wheel Revolution Data and bit number 3 indicates support for the Crank Revolution Data, indicating to the client that the device support data collection for those features.

The value of the variable along with its size is then set using the characteristic object created for it. Since this characteristic is of the read type there is no need to send notification to the client.

6.2.1.2 Sensor Location

The Sensor Location characteristic is used to inform the client of the device's physical location.

In order to implement this characteristic a static variable of 1 byte with the 7 bit set to 1, when implementing the Left Pedal, or the 8 bit set to 1, when implementing the Right Pedal, and all the other bits set to 0. Each of the active bits of the bit map indicates a single sensor location.

The list of possible sensor locations is available in the GATT Specification Supplement (Bluetooth, 2021b).

Similar to the previous characteristic the value of the variable along with its size is also set using the characteristic object created during initialization and since this characteristic is also of the read type there is no need to send notification to the client.

6.2.2 Main Logic Loop

This function is where notification is made to the connected clients whenever new sensor data is available.

In order to control the loop cycle, the communication is delayed to once every 100ms, which equals a sampling rate of 10Hz from the sensors. To achieve this number, several frequencies were tested and it was concluded that this was the best possible value, since the sampling rate of the HX711 is also limited to 10 samples per second.

In each loop a check is made on the existence of a connected client before starting sampling from the attached modules. If a client is connected, a new sample is requested from the HX711 library, which will correspond to a new weight reading from the load cell.

If a new sample is available, a calculation is then performed, adjusting each characteristic to its appropriate value, and a notification is sent to the customer.

6.2.2.1 Cycling Power Vector

The Cycling Power Vector feature is used to send raw force data to the connected client. Since the focus was present on obtaining the force values that the user applies to the pedal during its rotation, it will be force, not torque, that will be developed from the sections of the feature.

In order to develop that section, first the Cycling Power Feature bit was enabled indicating that the Sensor Measurement Context is force-based, as explained during the configuration phase, so the Instantaneous Force Magnitude Field will be implemented according to its specification.

The Instantaneous Force Magnitude Array is a variable length field that stores one or more readings of the raw force applied to the loadcell, expressed in Newtons and with a resolution of 1 Newton.

Since the library returns the sensor measurements in grams, and 1g equals to 0.009807 Newtons, the following conversion was applied:

$$Force(N) = Weight(g) \times 0.009807 \tag{6.1}$$

The returned values were then rounded to the appropriate resolution and each sampled measurement from the loadcell, in accordance to the configured sample rate, is added to the array.

The characteristic also includes a flags field, with a size of 1 byte, in which each bit tells the client which data is present in the received characteristic value.

Further details of the characteristics data structure is available in section 4.

6.2.2.2 Cycling Power Measurement

The Cycling Power Measurement is a characteristic used to supply information to the client regarding power, speed or cadence.

As the previous characteristic it also includes a flags field, but with a size of 2 bytes, to tell the client what fields of data are present.

Since the primary goal of the custom power meter was to develop the force sensor readings, not all fields of this characteristic were used or fully develop for this project, and were primarily used as a proof of concept.

The Instantaneous Power Field is a mandatory field of this characteristic since it stores the value of the power measured by the device.

Since samples of the force in Newtons are already available from the previous characteristic, this value can be used to calculate the Power using the known length and cadence of the crank arm.

$$Power(W) = Force(N) \times CrankLength(m) \times Cadence(1/s) \tag{6.2}$$

Ideally the Power calculations should use the angle of the crank in order to measure the effective force applied to the pedal, however, during the exploratory analysis a practical way to extrapolate the location of the pedal crank at a given time was not found, as it is currently only possible to detect its cadence by completing one full rotation.

A possible approach to this problem would be to apply a rotary encoder to the axis of the pedal that would give readings of its location as it travels while during a complete rotation, and thus calculating the angle and its angular velocity. A rotary encoder is a device that converts continuous movement into electrical impulses, and could easily be used by the chosen microcontroller.

6.3 Stationary Bicycle Setup

This section will detail the implementation of the sensors in the stationary bicycle setup, which includes the commercial sensors for which their respective installation procedures were followed, as described in annex III and II.

A picture of the rehabilitation platform with sensors installed is shown in figure 6.1, missing only the heart rate monitor that is attached to the user.

All tests, whose sample results are shown in chapter 7, were conducted using this setup while recording data from the sessions using the backoffice application and

Figure 6.1: Stationary Bicycle Setup

provided APIs. Initial testing was conducted to individual sensors to assure they were working correctly before conducting trials will all sensors attached.

Due to the fact that the available stationary bicycle lacked a location to correctly implement the speed sensor, due to the flywheel being enclosed but also the lack of space between the flywheel and the frame, it was not installed as part of the final setup and thus its values are simulated upon the characteristics of a regular bicycle. Assuming a 0.4 to 1 gear ratio and a 29 inch wheel diameter of an average bicycle it can be extrapolate that for each crank rotation it would have theoretically traveled approximately 5.7 meters. That value is then used with in conjunction with the Cadence value to calculate the theoretical speed value.

$$Speed(Km/h) = Cadence(RPM) \times 60(Minutes) \times 0.0057(Km) \tag{6.3}$$

However both the APIs and backoffice application were made compatible with the sensor and when attached the values are recorded and displayed properly when the device is present and in use, so the theoretical value is only used in the Display when the sensor is not present or disabled. This would occur in the likely scenario of the stationary bicycle setup being replaced by a regular bicycle on a stationary setup.

During testing it was noticed that there was some interference when using the cadence sensor in a position closer to the pedal which caused improper readings. This was assumed to be because this type of sensor uses a magnetometer and since stationary bicycles use a resistance system on the flywheel that is based on magnets, that are lowered upon the flywheel in order to induce a resistance force, the presence of those magnets could be interfering with the sensor readings. By placing the sensor closer to the axle the interference was minimized.

6.4 Summary

Using the knowledge gained from the initial proof of concept test of each individual component, it was possible to develop a set of pedals that met the goal of measuring lower limb strength.

It was shown how the custom power meter firmware was developed using Arduino based libraries for an ESP32 microcontroller, following the Bluetooth SIG specification where possible so that the power meter would be compatible with other applications.

Although some features of the Cycling Power Service were developed, there is still room for improvement in order to make the pedal a fully compatible device according to its specification, since only the parts necessary for the project at hand were developed.

The chapter 7 will show the tests performed on the rehabilitation platform, as well as the data validation that was done in order to verify the validity of the data recorded and presented.

Chapter 7

Results and Discussion

7.1 Introduction

This chapter aims to describe the experiments and tests made to determine the viability of the developed backoffice application, User Interface application and custom power meter as usable rehabilitation platform.

As described in the previous chapter initial tests were done on individual sensors before moving on to viability tests with all the sensors attached to the platform. The next section will explain the methodology used in the experiments and ends with a discussion of results.

7.2 Functional Testing

The functional testing focused on the backoffice application in order to validate the usability and usefulness of the developed components. The end-users navigated the application freely while performing a set of provided tasks while answering a survey. The feedback of the users were then taken into consideration for necessary corrections or improvements of the application.

7.2.1 Methodology

A group of potential end-users were asked to perform a set of predetermined tasks that involved interaction with the application, simulating a real life environment on which the application would be used.

Those tasks ranged from patient management, sensor management, session management and data analysis. Data validation tests were also performed on relevant fields.

The users were then asked about the difficulty of each of the tasks, problems encountered, if any, and suggestions in order to gather feedback for further improvements.

The users were selected from a mix of people with no previous knowledge of the field and some users that work or are in direct relation to the field of medicine.

The test cases followed the usual functional test template design in which steps and expected results were described to the user, as shown in figure 7.1.

Test Case ID	Test #01	Test Case Description	Test Backoffice Login Funcionality		
Created By	Emanuel	Project	Rebike Backoffice Application		
Tester's Name	Tester 01	Date Tested	02/12/2021	Test Case (Pass/Fail)	Pass

ID	Prerequisites:
1	Using Chrome Browser
2	
3	
4	

ID	Test Data
1	Email = admin@demo.com
2	Password = demo
3	
4	

Test Scenario	Verify on entering valid email and password, the supervisor can login

Step	Step Details	Expected Results	Actual Results	Pass / Fail / Not executed / Suspended
1	Navigate to https://localhost:4200	Site should open and ask the user to input login details	Site opened as expected and requested credentials	Pass
2	Enter Invalid Email	Message is displayed of wrong email format	Error message indicating wrong email	Pass
3	Enter Valid Username And Password	Site accepts credentials	Worked as expected	Pass
4	Press Sign In Button	Site allows login	Credentials accepted	Pass

Figure 7.1: Test Case Template

7.2.2 Results

Besides improving overall quality of the application, by detecting common mistakes made during development like missing validations or misspelling errors, functional tests proved to be an effective tool to gather feedback on changes needed to make the site more responsive and intuitive.

In general all test cases passed successfully, although some had to be repeated in order to correct minor errors detected. Most errors being detected accounted to one step not allowing further tests to proceed before the issue or problem detected being solved.

In terms of usability all test users queried shared the opinion that the design was intuitive and responsive, although all of them pointed out the need for the creation of a proper user manual describing the features in more detail. Since functional tests were shadowed by developers, that was not the issue during testing.

One test case in particular, regarding the creation of the workout session, did not reach consensus. Nearly half of the inquired users asked for the need to schedule future sessions, since the current configuration of the application only allows one active session per patient.

This would allow supervisors to add multiple sessions that would be done by the same patient in, for example, the following week. On the other side some users had the opinion that multiple sessions would be confusing since the page shares the created sessions with previous session history.

Future developments should consider separating the session history from active sessions in order to allow for the schedule implementation.

The approached users that had some connection or worked in the field where this kind of rehabilitation application would be used all concurred that it filled a existing gap in terms of rehabilitation data recording and processing, since most hospitals lack the tools needed to perform and record these kind of sessions, as it is mostly done manually.

The data recorded and its visualization methods were appraised as very useful and an essential part of the rehabilitation process, and there is much room for improvement like all developments, although some concerns regarding data privacy were raised.

The real-time control of the user interface was also tested and worked as expected. Suggestions regarding the user interface focused mostly on the possibility of customization and adding new sensors, something that was already considered for future developments.

7.3 Workout Session

The Workout Session tests aimed to provide information on the validity of our sensor data gathering solution, while also testing our user interface and evaluating the impact of providing biofeedback during said workout.

7.3.1 Methodology

Each user was provided with a 5 minute workout to complete at their own pace, while changing the resistance between sessions. A second round of 5 minute workouts was provided but with one pedal being handicapped in order to simulate an injury.

All tests were made with both custom power meters and the cadence sensor attached to the left crank of the stationary bicycle, while also using the heart rate monitor on the user participating in the workout.

Resistance is controlled manually on the bicycle by rotating a knob, shown in figure 7.2, that allows for 7 full clockwise rotations. Each rotation lowers a set of magnets onto a 8Kg flywheel in order to make it harder for the user to rotate the pedals, and thus needing to apply more force, thus a level of 1 being no resistance and 8 being full resistance applied.

Figure 7.2: Bike Resistance Knob

7.3.2 Results

A set of testing sessions was done to assess that the developed API was able to record data from all the attached sensors at the same time. Due to proper cleat compatible shoes being unavailable at the time of testing only down-force was measured since the user wore normal shows without being physically attached to the pedals, as would occur with the cleat devices.

The subjects were asked to stay above a cadence of 60 RPM and try to keep a regular pace during the whole session, despite any resistance applied to the flywheel. Below some of the sampling sessions will be shown in order to discuss the potential analysis that will be done with the gathered data.

During the first session the API was able to collect, during the 5 minute workout, a total of 6884 samples from the sensors attached.

The workout video played as expected and provided the proper incentive by displaying an pleasant scenery and the metrics of data being recorded in real time.

Description	Average	Min	Max	Unit of Measure
Resistance	8	8	8	-
Cadence	69	37	73	RPM
Heart Rate	140	103	152	BPM
Left Pedal Force	97	2	208	Newtons
Right Pedal Force	87	0	176	Newtons

Table 7.1: First Session Statistics

After the session the supervisor was able to analyse the data using the provided charts in a detailed view page of the workout session.

As shown in figure 7.3 it can be verified that the Cadence data was correctly collected from the sensor at a rate of 2 samples per second or once every 500ms.

As for the heart rate data the collection rate was of 1 sample per second, and an increase in the user's heart rate can be observed as the duration of the workout increases, as shown in figure 7.4, which is a consistent behaviour since exercise increases heart rate.

Figure 7.3: First Session Cadence Data

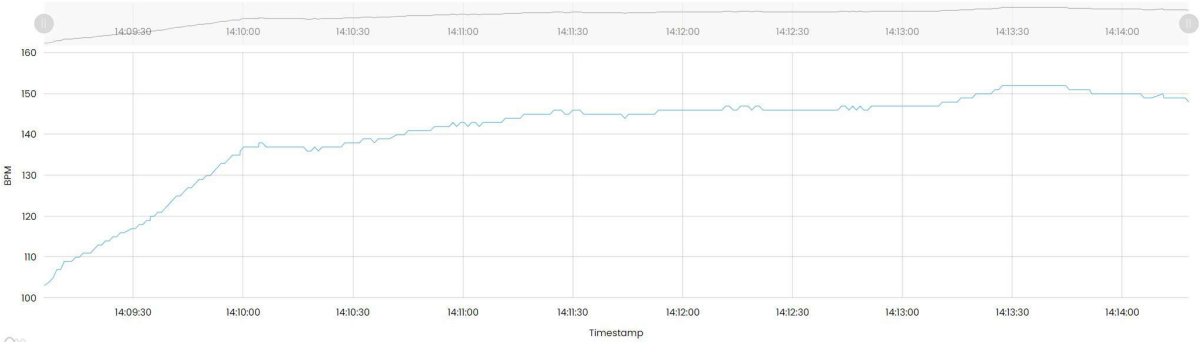


Figure 7.4: First Session Heart Rate Data

Data from both of our custom pedals were also collected at a rate of 100ms, or 10 samples per second, in order to obtain a large number of force values during a full pedal crank rotation.

The data collected allowed the supervisor to be able to analyse the force applied by both legs simultaneously and it was even possible to overlap both leg datasets in order to correctly visualize the alternating force between both legs in order to compare symmetry, as shown in a zoomed sample shown in figure 7.5.

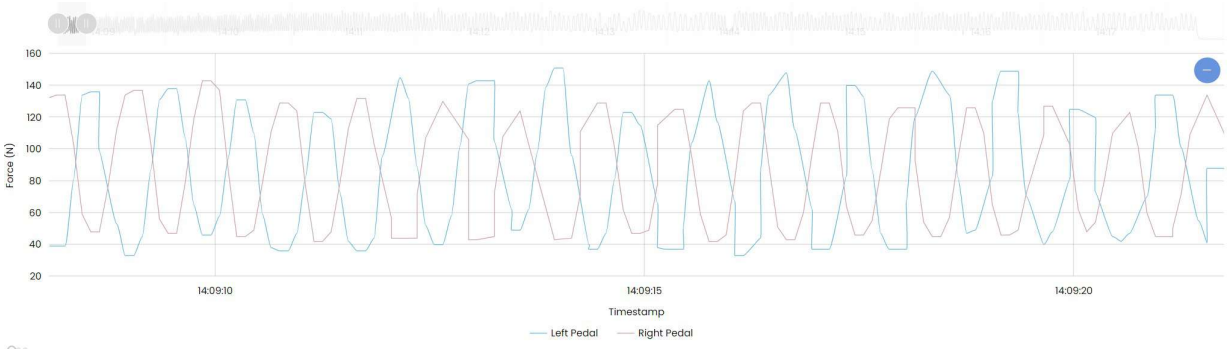


Figure 7.5: First Session Force Data

The force ranges detected by the device, between around 0 to 208 newtons, may seem low when compared to regular cycling values, but since in a recumbent bike the weight of the user is not fully supported by the pedals those values fall within expected parameters when compared to other sourced material (Boon et al., 2014).

The average initial weight not starting from 0, between rotations, was also expected since our force readings were not calibrated taking into account that the user starts the exercise with their feet already pressed upon the pedals, thus inducing the initial force values detected.

After the first session was concluded a second session was created by the supervisor while simulating an injury on the right leg.

A small plastic object was placed under the right feet of the user to cause discomfort and force the user to compensate the simulated injury, and to verify if that discrepancy could be detected. Due to the nature of this session the duration was reduced to 3 minutes since the plastic piece caused pain during longer intervals of exercise.

As shown in table 7.2 the session was conducted using the same resistance as before in order to offer a comparison.

Description	Average	Min	Max	Unit of Measure
Resistance	8	8	8	-
Cadence	66	30	69	RPM
Heart Rate	119	95	134	BPM
Left Pedal Force	104	25	210	Newtons
Right Pedal Force	86	35	140	Newtons

Table 7.2: Second Session Statistics

Although the injury didn't detail a large decrease in the average force applied by the injured limb, of only 1% in comparison with the previous session, a 25% decrease in the maximum force applied by the injured limb was nevertheless noted.

The left pedal detected an increase in 7% which can possibly be explained by the user compensating the disconfort of the right leg by applying more force to the left pedal. In the cadence values a decrease of 5% in the average cadence was also noted.

The most noticeable difference can be verified when comparing the differences between both legs within the same session.

While in the previous session the user already demonstrated a difference of 11% more force being applied on the left pedal, when compared to the right pedal, that difference became even more apparent in the second session since that difference increased to 20%.

The difference is even more noticeable when using the developed analysis charts in which it is technically possible to visually observe the difference between both forces in the sample shown in figure 7.6.

As for the Cadence and Heart Rate charts no changes worth mentioning were detected as shown in figure 7.7 and 7.8, since all data was collected properly.

Overall the tests conducted allowed to further validate the usefulness of the developed applications and custom power meter and to gather ways to improve it's performance to that it can be scaled to a larger test group in the future.

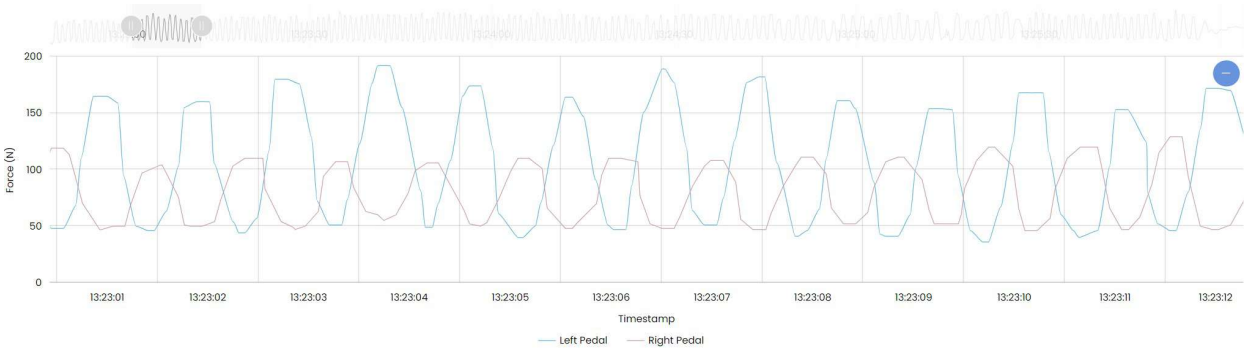


Figure 7.6: Second Session Force Data

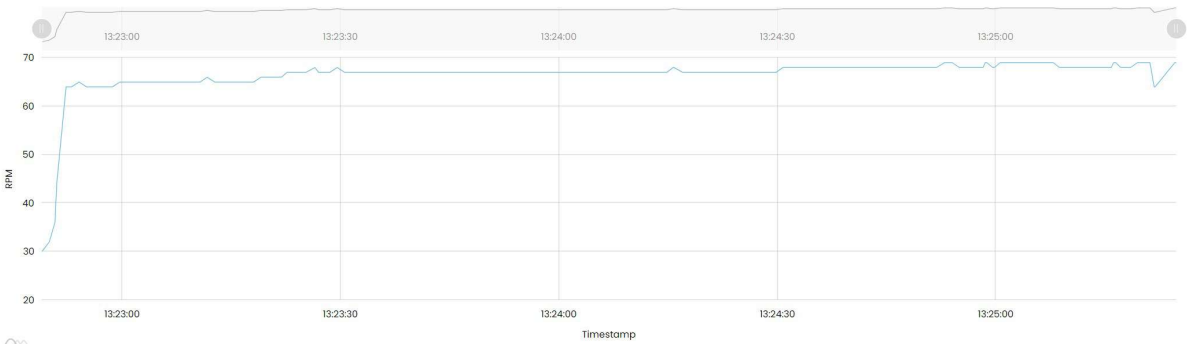


Figure 7.7: Second Session Cadence Data

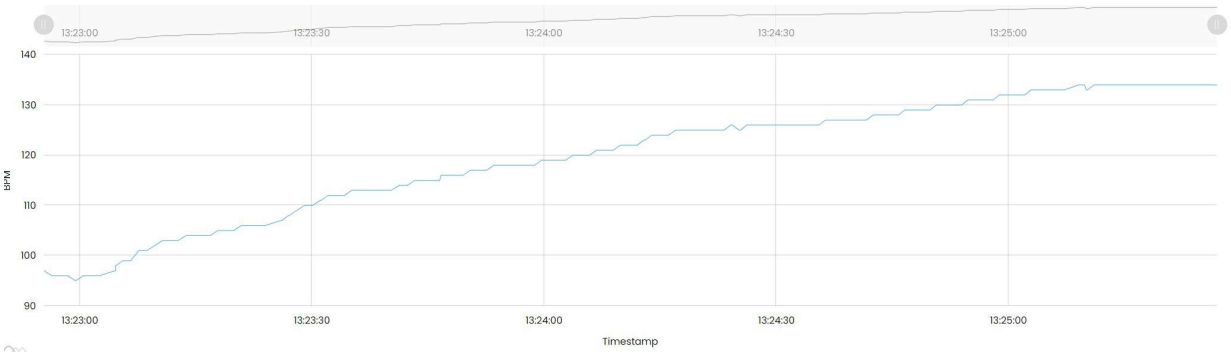


Figure 7.8: Second Session Heart Rate Data

7.4 Commercial Applications

Using commercial applications, like the previously mentioned nRF Connect, it is feasible to test the developed power meter and validate that it correctly displayed the implemented services and characteristics similar to its commercial counterparts.

7.4.1 Methodology

First the custom power meters were tested using the nRF Connect application in order to verify that all the developed services and characteristics were displayed.

The nRF connect scanned for the devices and both were found and available. Each of the custom devices were connected to individually and data retrieval was attempted on both of them.

After using the exploratory tool, commercial fitness oriented applications were tested to evaluate if the developed device would behave and be detected as if it was a commercial device.

The application chosen was the Rouvy app, as it provided a free trial that allowed the application to fully function during testing.

The tests were conducted only using the custom power meter and with no other external sensors connected.

7.4.2 Results

Using the nRF Connect application it was realized that both pedals advertised the correct services that were made available during development.

The Device Information service provides applications with relevant data about manufacturer, serial, and software and firmware versions as expected.

The Cycling Power Service also appear correctly configured, featuring the Cycling Power Measurement, Cycling Power Vector and Location characteristics.

The characteristics that provided regular notifications when subscribed to, like the Cycling Power Vector, also performed well, correctly sending new notifications to the client as regular intervals.

After the initial assessment using the provided app, the testing of the devices on the fitness apps then proceeded.

The application tested, the Rouvy app, provided a configuration menu on which compatible nearby devices are scanned so they can be added to the application.

An initial scan revealed that both custom pedals were detected by the application and appeared as valid power meters that could be added, in order to be used. Selecting the custom device, before adding it, also provided extra information regarding it's Device Information Service, as shown in figure 7.9.

Since both devices appeared separately, only one device can be used at any given time. However this behaviour is normal and was expected since it was already known that the Bluetooth SIG specification provided a solution to this issue and it is documented in the Cycling Power Service specification (Bluetooth, 2016b).

The solution, referred in the specification as a Distributed Power System, involves making a pair of devices communicate with each other, using known protocols, and then having one of the devices act as the main device that retrieves and calculates the sum of total measurements made by both devices. That data is then transmitted by the same device to the applications, while the other remains hidden. However at this time that feature was not implemented in our project.

The initial attempt to add the devices prompted an error reporting that the devices could not be used. Searching the logs of said application the problem was tracked down to an incorrect configuration of the bit pattern in the Cycling Power Feature, thus

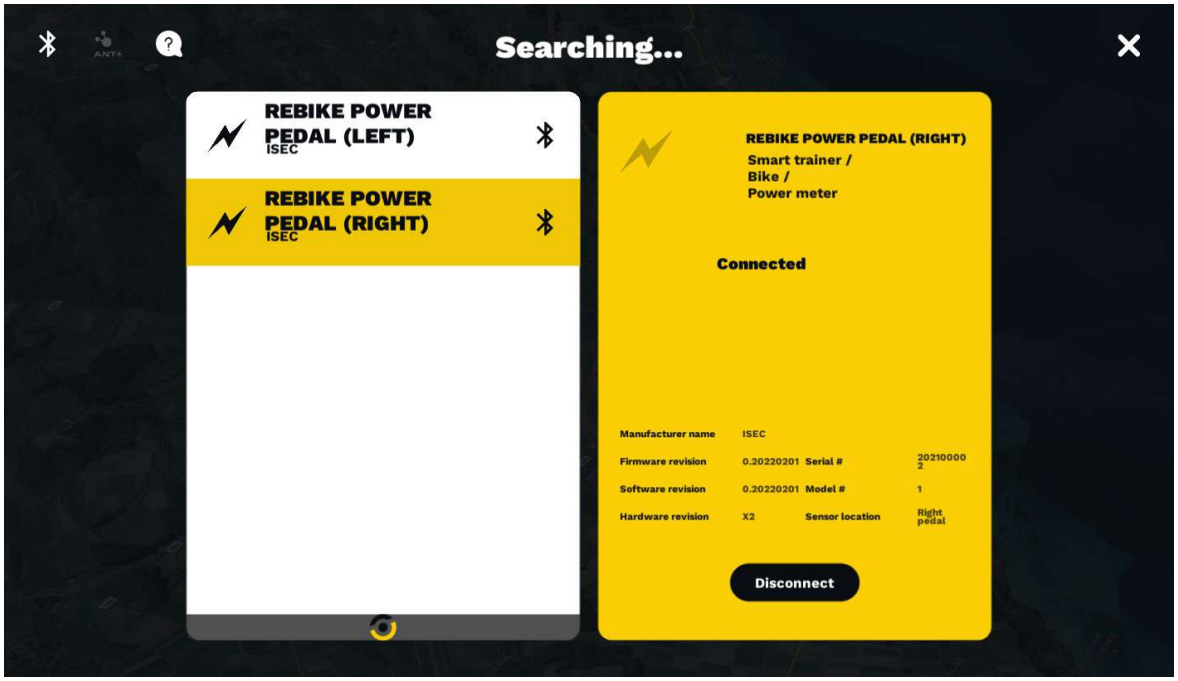


Figure 7.9: Rouvy Custom Scanning

validating that the Rouvy application only allowed correctly configured devices to be added, although initially detected in the scan.

After correcting the bit mapping the application then allowed the addition of one of the power meters, so that it could be used for the workout.

Since the custom devices also advertise that it provides crank revolution data, as explained in section 4.5.3, the application assumes that it can obtain RPM values from it and also adds the device as a Cadence sensor, as shown in figure 7.10.

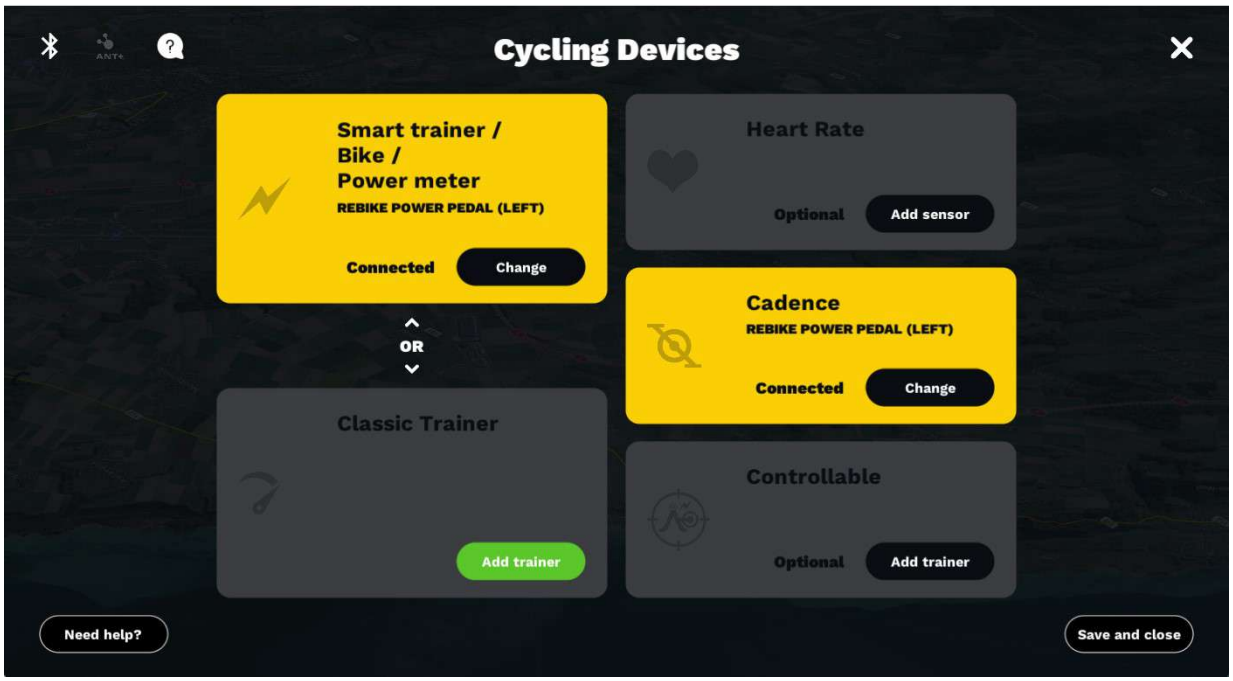


Figure 7.10: Rouvy Sensor List

As only one device can be used at a time in the application, as explained above, the power readings for the workout are taken from only the Left or Right pedal while the other is estimated and an average value is calculated. This works similar to other commercial devices that give the possibility of only buying one instrumented pedal, to save costs, without needing a complete set to be able to workout.

Chapter 8

Conclusion

While there are many software applications available in the bicycle industry, for training-oriented purposes, there are not many specifically geared towards the rehabilitation process. In addition, many require custom-made hardware that makes them difficult to replace in the future.

The proposed setup, while still using some custom-developed hardware, aims to provide compatibility to commercially or user designed devices using well-known and open-source communication protocols, thus expanding its future uses.

Currently available sensors can be added or replaced according to user specifications, while the interface aims to be as customizable as possible. Compatibility for new sensors is also easy to develop since known communication protocols were used, opening up the options for new devices.

The developed user interface provided the necessary feedback to make the rehabilitation sessions more appealing for the user and provides the proper motivation during the workout with useful metrics for both user and supervisor alike.

The backoffice application was built around the identified needs of users in the field and provided the necessary tools for supporting and analysing rehabilitation sessions while collecting data from the various sensors attached to the platform. The supervisors are provided with various charts and statistics that will help them on better performing their tasks, while also being able to fully manage the rehabilitation environments, it's users and other relevant items.

As learned from the interview with an expert in the field the market still lacks platforms that provide support for the rehabilitation process, so there is still room in the market for these types of platforms. The appeal is even greater when the fact that it was not designed for a singular purpose is considered.

In general the objectives proposed for this project were accomplish, although there

is still room for improvement in the custom power meter device and backoffice applications as it will be explained in the next section.

8.1 Future Developments

Future developments and research will aim to expand upon that compatibility, not limiting its uses to lower limb rehabilitation only but also to other types of rehabilitation.

Since the platform is designed for the sensors, and not directed specifically to the stationary bicycle setup, one could for example take some of the sensors and apply them to a normal bicycle or other types of rehabilitation machines used in normal rehabilitation procedures.

One could for example apply a treadmill to the setup by attaching a Heart Rate Monitor and a Running Speed and Cadence sensor to its users and use the developed platform to record data while users perform workout routines on that device.

The goal of this would be to give the platform other possible uses, besides managing a stationary bicycle rehabilitation setup, in order to expand its usefulness and in this way increase its longevity, by applying sensors to other existing rehabilitation hardware, such as the treadmill example given above.

The project was aimed to build an adaptive platform and be open to expansion to other purposes as much as possible without compromising its main goal. Thus expanding the types of sensors currently compatible with the application would help achieve this.

As for the power meter since it was constructed as a prototype, further study and development is needed in order to make it a fully compatible device with the Bluetooth SIG specification and other enhancements can be made to the hardware used.

For example, there is the need to address the issues of power consumption that have not been properly calculated or measured, some of the off-the-shelf components have small form factor siblings that can be used to reduce the size of the overall setup, and some services are still lack mandatory characteristics that were not immediately needed for this project, since the primary focus was on obtaining the force measurement, and therefore, although explored to some extent they were not developed.

However it is our belief that our initial prototype build provides a proper development platform that can evolve to a fully featured device, since the necessary components are present.

8.1.1 Companion App

As the approach taken was to use only wireless communication with the sensors, it would be possible to attach the sensors to a road bike and record data from that type of session, without the need for stationary mounting.

However since the theoretical range of Bluetooth LE is 100 meters, the range would be limited to for example cycling in an indoor pavilion.

To solve this problem a mobile Companion App for the backoffice application could be developed, in Android or IOS, in order for it to connect to the sensors and record data. The App would use the Bluetooth of the device to connect to the sensors and act in the same way as the user interface app, displaying live data from the sensors while recording said data locally on the mobile device.

The data that would be stored by the app, in the mobile device carried by the user, would then sync its session data, either in real-time or at the end of the session, back to the backoffice Application.

Each user of the app would need to be registered in the backoffice application in order for it to be paired with a patient profile, and data recording would only be started when a new session would be created specifically for that user's device by the supervisor, similar to what is already done in the user interface display.

This way the user could have the freedom to use the bicycle outdoor or indoor without having to worry about the limitation of the data acquisition range.

8.1.2 Adding New Sensors

Since the data acquisition process of the development followed the Bluetooth SIG specification, new sensors, as long as they are compatible with Bluetooth LE, can easily be added as compatible.

The process would involve identifying sensors that are compatible to a Bluetooth SIG profile, and from that documentation create the methods that will process the data from the sensors in order for them to be recorded alongside the other types of sensors we already support.

Two possible candidates were already identified in the requirements interview as shown in section 3.3, such as the O2 Saturation and Blood Glucose.

Although the O2 Saturation would be a possible addition to the rehabilitation session itself, use of the Blood Glucose would be reserved to a pre-session and post-session analysis.

In table 8.1 below we identified the profiles of the requested sensors and added two others that we think would be useful to add in the future.

8.1.3 Custom Power Meter

The custom developed power meter, whose firmware was developed to be compatible with the Bluetooth SIG specification, did not implement all the necessary features required for it to function as fully compatible device.

Since the focus was only on developing the parts that suited the needs of the project, such as the measuring of force, there are other parts of the firmware that still lack

Bluetooth Specification	Description
Glucose Profile	Would allow the platform to measure the glucose level of patients, before and after a rehabilitation session, using devices that implement the Glucose Service
Pulse Oxymeter Profile	Would allow for the recording of SpO2, percent oxygen saturation of hemoglobin, and Pulse Rate of patients during rehabilitation sessions
Weight Scale Profile	Would allow for the measuring of Weight and Body Mass Index from the patient, before and after the rehabilitation session
Running Speed and Cadence Profile	Would allow to measure cadence and speed in an running scenario, such as a treadmill

Table 8.1: Future Sensor Profiles

development or are at a very early stage and were mostly used as a proof of concept.

The MPU6050 sensor present in that device was mainly used for detecting cadence using the triaxial accelerometer, but its algorithm still needs improvement and was only developed as a proof of concept since it was not directly used in the data collection applications.

The same applies to the gyroscope part of the sensor that was initially planned to be used in detecting the angle of the pedal, and as such should be investigated and developed further.

Signal processing, such as low pass filtering, can also be applied to the raw signals from the sensors in order to improve overall quality of the data provided by the device.

A proper battery consumption study should also be conducted in order to measure the battery life of the device during normal operation, and consider replacing the current battery supply with a lithium-ion alternative in order to reduce overall size of the build.

Implementation of the wireless communication between both pedals is also important for developing the Distributed Power System described in 6.

8.1.4 Implementation and Usability Testing

While initially planned for the project to be fully implemented on the supporting hospital that was initially assigned to this project, such interaction was not possible due to pandemic concerns and inherent restrictions that were in place at the time of development.

Usability tests were planned to be conducted with the hospital staff in a real life environment and as such were not conducted. Since we think those tests are crucial for completing and improving upon the developed applications and hardware it recommended for future developments.

The full project was also not implemented in a hospital environment in order to test the proposed setup with real patients that are part of a rehabilitation process and as such is also important to conduct further testing in that manner.

8.1.5 Final Comments

The project provided a unique perspective of developing hardware oriented front-end applications and some parts even allowed for the sharing of some of knowledge obtained during development, in the form of a presentation during the Coimbra Health School Annual Meeting of 2021 (Ferreira et al., 2021), and the development of a full article currently accepted, but not yet published, for presentation on the CISTI'2022 - 17th Iberian Conference on Information Systems and Technologies (CISTI2022, 2022).

The development of the custom hardware provided useful knowledge on how to develop such devices as well as improved our notions of microcontroller development platforms, Bluetooth specifications and circuit prototyping, while also turning out to be a viable low cost alternative to its commercial counterparts.

That also presented a challenge since development focus had to be divided between developing both hardware and software components and as such it is our opinion that further developments of the project should be divided accordingly, between backoffice applications and firmware development, as each have some degree of complexity that went beyond the initially planned scope of the project and allocated time.

Because of that challenge however, developing this project provided the necessary motivation to investigate this field of study further and it is our hope that we can participate in more projects within this area of development, so that we can continue to help build better tools for the field of rehabilitation.

Since it was our goal to reach a level of development that the setup was mature enough for field testing, it is our hope that the project will be continued since it has the potential to become a useful tool in a real rehabilitation environment.

Bibliography

Angular (2021a). *Angular Features*. URL: https://angular.io/features (visited on 09/02/2021).

— (2021b). *Material Design*. URL: https://material.angular.io/ (visited on 09/02/2021).

Apple, I. (2013). *Bluetooth Programming - Overview*. URL: https://developer.apple.com/library/archive/documentation/NetworkingInternetWeb/Conceptual/CoreBluetooth_concepts/CoreBluetoothOverview/CoreBluetoothOverview.html (visited on 09/06/2021).

Arlati, S., V. Colombo, D. Spoladore, L. Greci, E. Pedroli, S. Serino, P. Cipresso, K. Goulene, M. Stramba-Badiale, G. Riva, A. Gaggioli, G. Ferrigno, and M. Sacco (2019). "A Social Virtual Reality-Based Application for the Physical and Cognitive Training of the Elderly at Home." In: DOI: 10.3390/s19020261. URL: www.mdpi.com/journal/sensors.

Avia (2021). *24-Bit Analog-to-Digital Converter (ADC) for Weigh Scales*. URL: https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/hx711_english.pdf (visited on 09/02/2021).

Barbosa, D., C. P. Santos, and M. Martins (2015). *The application of cycling and cycling combined with feedback in the rehabilitation of stroke patients: A review*. DOI: 10.1016/j.jstrokecerebrovasdis.2014.09.006.

BikeLabyrinth (2021). *What is Bike Labyrinth? - Bikelabyrinth.com*. URL: https://www.bikelabyrinth.com/product/ (visited on 09/02/2021).

Bluetooth (2011a). *Heart Rate Profile*. URL: https://www.bluetooth.com/specifications/specs/heart-rate-profile-1-0/ (visited on 09/02/2021).

— (2011b). *Heart Rate Service*. URL: https://www.bluetooth.com/specifications/specs/heart-rate-service-1-0/ (visited on 09/02/2021).

— (2012a). *Cycling Speed and Cadence Profile*. URL: https://www.bluetooth.com/specifications/specs/cycling-speed-and-cadence-profile-1-0/ (visited on 09/02/2021).

Bluetooth (2012b). *Cycling Speed and Cadence Service*. URL: https://www.bluetooth.com/specifications/specs/cycling-speed-and-cadence-service-1-0/ (visited on 09/02/2021).

— (2016a). *Cycling Power Profile*. URL: https://www.bluetooth.com/specifications/specs/cycling-power-profile-1-1/ (visited on 09/02/2021).

— (2016b). *Cycling Power Service*. URL: https://www.bluetooth.com/specifications/specs/cycling-power-service-1-1/ (visited on 09/02/2021).

— (2021a). *Assigned Numbers | Bluetooth® Technology Website*. URL: https://www.bluetooth.com/specifications/assigned-numbers/ (visited on 09/06/2021).

— (2021b). *GATT Specification Supplement*. URL: https://www.bluetooth.org/DocMan/handlers/DownloadDoc.ashx?doc_id=524815 (visited on 09/02/2021).

Boon, K., P. Klap, J. Van Lanen, G. Letsoin, and A. Jansen (June 2014). "Does pedalling on a recumbent bicycle influence the cyclist's steering behaviour?" In: *Elsevier*. DOI: 10.1016/j.proeng.2014.06.112. URL: http://resolver.tudelft.nl/uuid:79813eaa-8069-4baa-a5a4-cb2a4fb4f4c1.

Cheung, S. S. and M. Zabala (2017). *Cycling Science*. Sport Science Ser. Champaign, IL: Human Kinetics. ISBN: 9781492581079 9781450497329.

CISTI2022 (2022). *CISTI'2022 - 17th Iberian Conference on Information Systems and Technologies*. URL: http://cisti.eu/index.php?lang=en (visited on 02/02/2022).

Espressif (2021). *ESP32-WROOM-32E*. URL: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf (visited on 09/02/2021).

Ferreira, E., L. Roseiro, and C. Páris (Aug. 2021). "Universal API for wellness and fitness BLE sensors." In: *European Journal of Public Health* 31.Supplement$_2$. ckab120.063. ISSN: 1101-1262. DOI: 10.1093/eurpub/ckab120.063. URL: https://doi.org/10.1093/eurpub/ckab120.063.

Freddi, A., G. Olmi, and L. Cristofolini (Mar. 2015). "Introduction to the Application of Strain Gages." In: pp. 23–100. DOI: 10.1007/978-3-319-06086-6_2.

Garmin (2021a). *Garmin Vector 3*. URL: https://www.garmin.com/pt-PT/p/573589 (visited on 09/02/2021).

— (2021b). *How do Garmin Speed Sensors Measure Speed?* URL: https://support.garmin.com/en-US/?faq=9NL91YJSJd3Tnyif9jRSy6 (visited on 09/02/2021).

— (2021c). *The Wireless Sensor Network Solution - THIS IS ANT*. URL: https://www.thisisant.com/ (visited on 09/12/2021).

Ge, Z., P. W. C. Prasad, N. Costadopoulos, A. Alsadoon, A. K. Singh, and A. Elchouemi (2016). "Evaluating the accuracy of wearable heart rate monitors." In: *2016 2nd International Conference on Advances in Computing, Communication, Automation (ICACCA) (Fall)*, pp. 1–6. DOI: 10.1109/ICACCAF.2016.7748986.

Geonaute (2015). *Geonaute Bluetooth HRM Chest Strap*. URL: https://customercare.geonaute.com/hc/pt/sections/201651632-Banda-Cardio-Bluetooth-SMART (visited on 09/02/2021).

Georgiou, K., A. Larentzakis, N. Khamis, G. Alsuhaibani, Y. Alaska, and E. Giallafos (Feb. 2018). "Can Wearable Devices Accurately Measure Heart Rate Variability? A Systematic Review." In: *Folia medica* 60. DOI: 10.2478/folmed-2018-0012.

Goethel, M., P. Franco-Alvarenga, R. Canestri, C. Brietzke, R. Asano, and F. Pires (July 2018). "Pilot test of the resaerch project "Brain Pacemaker"." In: DOI: 10.13140/RG.2.2.32409.93286.

Haartsen, J. (2000). "The Bluetooth radio system." In: *IEEE Personal Communications* 7.1, pp. 28–36. DOI: 10.1109/98.824570.

InvenSense (2013). *MPU-6000 and MPU-6050 Product Specification*. URL: https://invensense.tdk.com/wp-content/uploads/2015/02/MPU-6000-Datasheet1.pdf (visited on 09/02/2021).

Kamal, A. A., J. B. Harness, G. Irving, and A. J. Mearns (Apr. 1989). "Skin photoplethysmography — a review." In: *Computer Methods and Programs in Biomedicine* 28 (4), pp. 257–269. ISSN: 0169-2607. DOI: 10.1016/0169-2607(89)90159-4.

KeenThemes (2021). *Metronic*. URL: https://keenthemes.com/products/categories/free-templates (visited on 09/02/2021).

Liu, S.-H., R.-X. Li, J.-J. Wang, W. Chen, and C.-H. Su (2020). "Classification of Photoplethysmographic Signal Quality with Deep Convolution Neural Networks for Accurate Measurement of Cardiac Stroke Volume." In: *Applied Sciences* 10.13. ISSN: 2076-3417. DOI: 10.3390/app10134612. URL: https://www.mdpi.com/2076-3417/10/13/4612.

Magene (2020). *S3+ Speed/Cadence Dual Mode Sensor*. URL: https://magenefitness.com/product_s3.html (visited on 09/02/2021).

Microsoft (2021). *Real-time ASP.NET with SignalR*. URL: https://dotnet.microsoft.com/apps/aspnet/signalr (visited on 09/02/2021).

Moraes, J. de, M. Id, Rocha, G. Vasconcelos, J. E. Vasconcelos Filho, V. Hugo, V. Albuquerque, and A. Alexandria (June 2018). "Advances in Photopletysmography Signal Analysis for Biomedical Applications." In: *Sensors* 18. DOI: 10.3390/s18061894.

NPM (2021). *NPM Repository*. URL: https://www.npmjs.com/ (visited on 09/02/2021).

Penko, A. L., J. R. Hirsch, C. Voelcker-Rehage, P. E. Martin, G. Blackburn, and J. L. Alberts (Dec. 2014). "Asymmetrical pedaling patterns in Parkinson's disease patients." In: *Clinical Biomechanics* 29 (10), pp. 1089–1094. ISSN: 0268-0033. DOI: 10.1016/J.CLINBIOMECH.2014.10.006.

Ramsden, E. (2006). "Chapter 9 - Application-Specific Sensors." In: *Hall-Effect Sensors (Second Edition)*. Ed. by E. Ramsden. Second Edition. Burlington: Newnes, pp. 177–185. ISBN: 978-0-7506-7934-3.

Rouvy (2021). *Rouvy Indoor Cycling & Running*. URL: https://rouvy.com/ (visited on 09/02/2021).

Santos, J., L. Roseiro, and P. Ferreira (Sept. 2019). "ExoBike Mechanical Component Development." Master's thesis. Coimbra Institute of Engineering.

Stanton, R., L. Ada, C. M. Dean, and E. Preston (Jan. 2017). "Biofeedback improves performance in lower limb activities more than usual therapy in people following stroke: a systematic review." In: *Journal of Physiotherapy* 63 (1), pp. 11–16. ISSN: 1836-9553. DOI: 10.1016/J.JPHYS.2016.11.006.

Vetek (2019). *Vetek Load Cell 202WA Datasheet*. URL: https://www.vetek.com/Dynamics/Documents/cc8bc1dc-e019-4acc-a6ea-581ccdfc8521/Datasheet_202WA_V2.pdf (visited on 09/02/2021).

Yin, C., Y.-H. Hsueh, C.-Y. Yeh, H.-C. Lo, and Y.-T. Lan (2016). "A Virtual Reality-Cycling Training System for Lower Limb Balance Improvement." In: DOI: 10.1155/2016/9276508. URL: http://dx.doi.org/10.1155/2016/9276508.

Zwift (2021). *Zwift Home Training App*. URL: https://www.zwift.com/ (visited on 09/02/2021).

Appendix 1: CHUC Interview

Requirements Interview

(Translated from Portuguese)

1. **Company Name**

   Central Hospital

2. **Department**

   Department of Physical Medicine and Rehabilitation

3. **Position**

   Manager – Head Nurse

4. **Does the department do internal research on the topic of rehabilitation or consult with outside sources?**

   Not at the moment, however, is it something that is being considered for the future since it is always good to promote investigation projects in order to develop new solutions.
   At the moment we are implementing a new project, that derived from external research, in the area of cardiac rehabilitation, something that was non existent in the central area of Portugal and will open the possibility to integrate new hardware developed for that purpose.

5. **How often do people require rehabilitation?**

   Usually that varies from case to case, depending on existing pathologies, but people usually require multiple sessions and not just one.

6. **What professionals are involved in the rehabilitation process?**

   Usually its only nurses and therapists, sometimes accompanied by the attending doctor for possible emergencies or accidents that may occur depending on the patient.

7. **In the context of rehabilitation what initial data from the patient do you consider essential to register before a rehabilitation session?**

   Usually name, age, weight, height, body mass index and existing pathologies.

8. **Is there a physical evaluation of the patient before each session?**

   Yes periodically, usually every month or so, new measurements are obtained to compare to previous values and assess the evolution of the patient throughout the rehabilitation process.
   Usually those measurements can also involve not just the patient's weight or BMI but also force applied by limbs and other body measurements.

9. **What kind of patients often require lower limb rehabilitation?**

   Usually stroke victims, paralysis, diabetes, among other pathologies.

10. **What are the main focused points of study when addressing lower limb rehabilitation?**

Usually measuring by hand, the force applied by the lower limbs, using known MRC defined scales, muscle mass variations and anthropometry.

11. **What kind of biometric data is considered relevant in a rehabilitation context?**

    Heart rate measurement, O2 saturation, blood glucose, body measurements, weight changes and limb or overall strength.

12. **What type of equipment is used for lower limb rehabilitation?**

    Besides the usual treadmill or stationary bicycle the hospital does not have specific rehabilitation oriented platforms that provide any kind of live measurements. Rehabilitation measurements are usually conducted using small manual devices, like dynamometers, hand grip force measurements devices or similar, or by touch.

13. **When you imagine a rehabilitation-oriented platform what functionalities come to mind?**

    It would be important to have the ability to record the previously talked about metrics, but also to have the means for easy consultation and comparison with previously obtained values.
    In our current situation we can only do this kind of studies using excel spreadsheets and by taking previous measurements manually from the existing hospital software that we use to record patient data (S Clinico).

14. **Have you ever used specialized hardware or software designed for rehabilitation?**

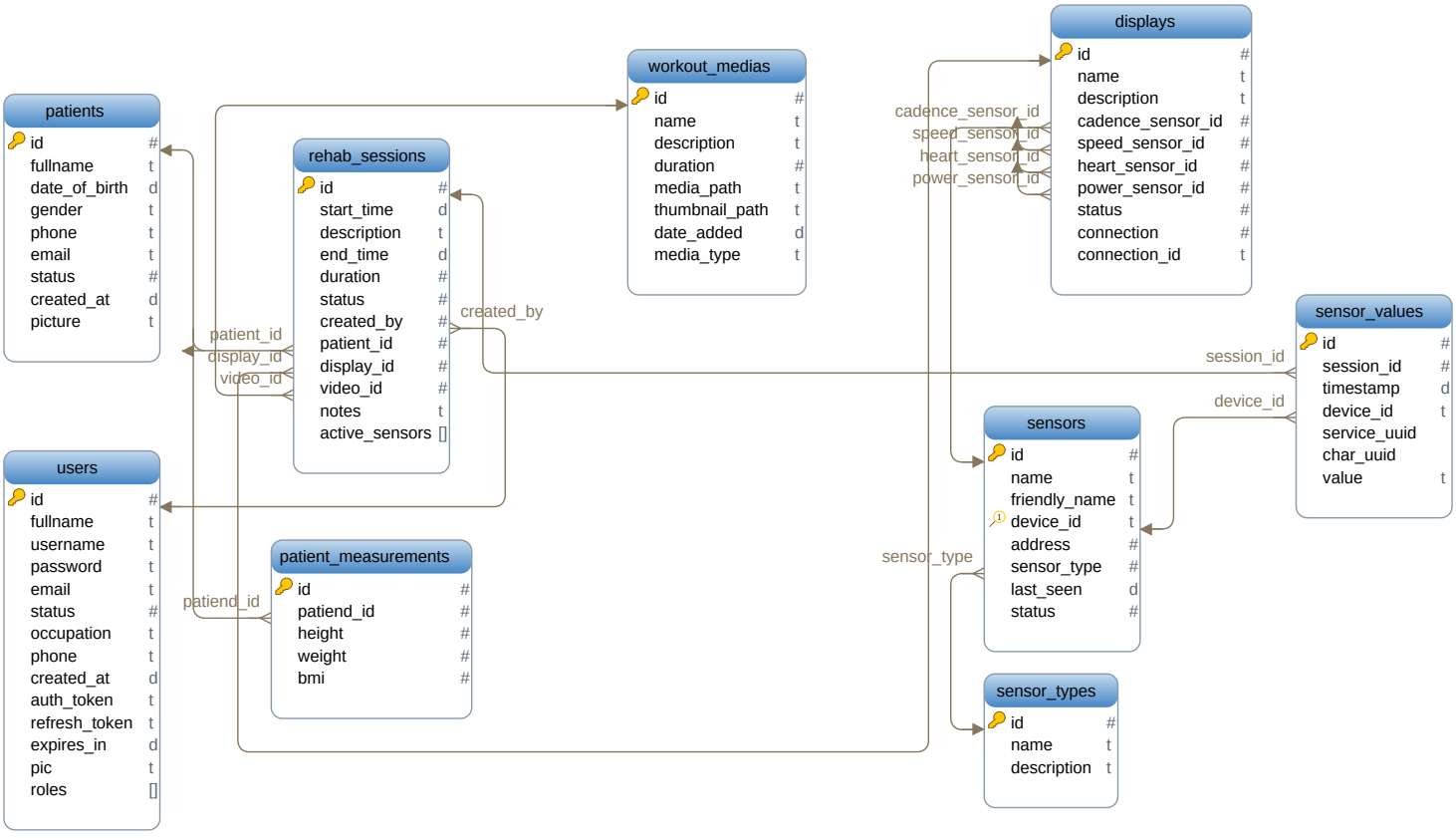
    No, the software that we use is the generic patient data recording software that every hospital department uses, and its not oriented for a specific purpose.
    As stated before there is no dedicated rehabilitation platform or software, but now with the introduction of the cardiac rehabilitation project we will have, although limited, access to log data and specialized equipment.
    However since the project as yet to be delivered we are still unsure what features it will have or if it will possess any type of data processing and extraction.

15. **What type of data visualization would you expect to have in that kind of software?**

    Besides the usual table layout, chart visualization would be nice.
    It would also be nice to be able to make specific cuts to the chart visualization to extract specific sections for future analysis.

Appendix B

Appendix 2: Backoffice Database


displays
id #
name t
description t
cadence_sensor_id #
speed_sensor_id #
heart_sensor_id #
power_sensor_id #
status #
connection #
connection_id t
workout_medias
id #
name t
description t
duration #
media_path t
thumbnail_path t
date_added d
media_type t
patients
id #
fullname t
date_of_birth d
gender t
phone t
email t
status #
created_at d
picture t
rehab_sessions
id #
start_time d
description t
end_time d
duration #
status #
created_by #
patient_id #
display_id #
video_id #
notes t
active_sensors []
sensor_values
id #
session_id #
timestamp d
device_id t
service_uuid
char_uuid
value t
sensors
id #
name t
friendly_name t
device_id t
address #
sensor_type #
last_seen d
status #
users
id #
fullname t
username t
password t
email t
status #
occupation t
phone t
created_at d
auth_token t
refresh_token t
expires_in d
pic t
roles []
patient_measurements
id #
patiend_id #
height #
weight #
bmi #
sensor_types
id #
name t
description t
cadence_sensor_id
speed_sensor_id
heart_sensor_id
power_sensor_id
patient_id
display_id
video_id
created_by
session_id
device_id
sensor_type
patiend_id

Default Layout

Table displays

* Pk	id	integer GENERATED BY DEFAULT AS IDENTITY	Unique display identifier
*	name	varchar(50)	Name of the display
	description	varchar(200)	Small description of the display
	cadence_sensor_id	integer	ID of the cadence sensor
	speed_sensor_id	integer	ID of the speed sensor
	heart_sensor_id	integer	ID of the heart rate sensor
	power_sensor_id	integer	ID of the power sensor
*	status	integer	Status of the display
*	connection	integer	Status of the display connection
	connection_id	varchar(200)	ID of the connection to the SignalR Hub
Indexes			
Pk	pk_displays_id	id	
Foreign Keys			
	fk_displays_sensors (cadence_sensor_id) ref sensors (id)		
	fk_displays_sensors_0 (speed_sensor_id) ref sensors (id)		
	fk_displays_sensors_1 (heart_sensor_id) ref sensors (id)		
	fk_displays_sensors_2 (power_sensor_id) ref sensors (id)		

Table patient_measurements

Table that stores patient measurement data like weight, height, and other relevant data.

* Pk	id	integer GENERATED BY DEFAULT AS IDENTITY	Unique identifier
*	patiend_id	integer	Patient unique identifier
*	height	integer	Patient height in centimeters
*	weight	integer	Patient weight in KG
*	bmi	integer	Body Mass Index (BMI = mass (kg) / height x height (m))
Indexes			
Pk	pk_patient_measurements_id	id	
Foreign Keys			
	fk_patient_measurements_patients (patiend_id) ref patients (id)		

Table patients

Table that stores patients personal information

* Pk	id	integer GENERATED BY DEFAULT AS IDENTITY	
*	fullname	varchar(100)	Name of the Patient
*	date_of_birth	date	Data of birth of the patient
*	gender	varchar(50)	Gender of the patient (Male, Female or Other)
	phone	varchar(50)	Patient's phone number
	email	varchar(200)	Email address of the patient
*	status	integer	Status of patient profile
*	created_at	date	Patient profile creation date
	picture	varchar(500)	

Table patients

Indexes			
Pk	pk_patients_id	id	

Table rehab_sessions

Table that stores the rehabilitation session data.

* Pk	id	integer GENERATED BY DEFAULT AS IDENTITY	Unique identifier
*	start_time	timestamp	Session started recorded time.
*	description	varchar(200)	Simple description of the rehabilitation session
	end_time	timestamp	Session stopped recorded time.
*	duration	integer	Session max duration in minutes
*	status	integer	Session status identifier.
*	created_by	integer	ID of the user that created the session.
*	patient_id	integer	The patient id from which the session refers to.
*	display_id	integer	ID of the display used in the rehab session
*	video_id	integer	ID of the workout video used in the session
	notes	varchar(600)	Current session notes
*	active_sensors	integer[]	ID's of the sensors used
Indexes			
Pk	pk_rehab_session_id	id	
Foreign Keys			
	fk_rehab_session_patients (patient_id) ref patients (id)		
	fk_rehab_session_users (created_by) ref users (id)		
	fk_rehab_sessions_displays (display_id) ref displays (id)		
	fk_rehab_sessions_workout_media (video_id) ref workout_medias (id)		

Table sensor_types

List of types of sensors

* Pk	id	integer GENERATED BY DEFAULT AS IDENTITY	
*	name	varchar(100)	Type of Sensor
*	description	varchar(200)	Description of type of sensor
Indexes			
Pk	pk_sensor_types_id	id	

Table sensor_values

* Pk	id	serial	Unique identifier
*	session_id	integer	ID that identifies the session from which the sensor data belongs to
*	timestamp	timestamp	Date and time of sensor data
*	device_id	varchar	Unique device identifier
*	service_uuid	uuid	Service UUID
*	char_uuid	uuid	Characteristic UUID
*	value	varchar	Sensor data values
Indexes			
Pk	pk_sensor_data_id	id	
Foreign Keys			

Table sensor_values

	fk_sensor_data_rehab_sessions (session_id) ref rehab_sessions (id)		
	fk_sensor_data_sensors (device_id) ref sensors (device_id)		

Table sensors

* Pk	id	integer GENERATED BY DEFAULT AS IDENTITY	Unique identifier
*	name	varchar(200)	The name of the device
	friendly_name	varchar(100)	A friendly description of the device for user interface purposes
* Unq	device_id	varchar(200)	Unique identifier of the device
*	address	bigint	Hardware address of the device (in numeric format)
*	sensor_type	integer	Numeric value that identifies the type of sensor
*	last_seen	date	Date time of last time device was scanned for
*	status	integer	Numeric value to indentify current status of the device
Indexes			
Pk	pk_sensors_id	id	
Unq	unq_sensors_device_id	device_id	
Foreign Keys			
	fk_sensors_sensor_types (sensor_type) ref sensor_types (id)		

Table users

Stores the login data and other relevant information of the backoffice users.

* Pk	id	integer GENERATED BY DEFAULT AS IDENTITY	Unique identifier
*	fullname	varchar(100)	User's name.
*	username	varchar(50)	User's login name.
*	password	varchar(50)	User's password.
*	email	varchar(300)	User's email address
*	status	integer	Status of the user account.
	occupation	varchar(100)	User occupation
	phone	varchar(50)	User phone number
*	created_at	timestamp	Timestamp of when the user was created.
*	auth_token	varchar(50)	
*	refresh_token	varchar(50)	
*	expires_in	date	
	pic	varchar(200)	User picture
	roles	integer[]	Roles of the user
Indexes			
Pk	pk_users_id	id	

Table workout_medias

Table that stores information regarding the workout motivational videos

* Pk	id	integer GENERATED BY DEFAULT AS IDENTITY	Unique identifier
*	name	varchar(100)	Video name
*	description	varchar(200)	Video description
*	duration	integer	Video duration in minutes

Table workout_medias			
*	media_path	varchar(100)	Path to the media location
*	thumbnail_path	varchar(100)	Filename of the video's thumbnail
*	date_added	date	Timestamp of date the video was added to the library
*	media_type	varchar(20)	Type of media (video or music)
Indexes			
Pk	pk_workout_media_id	id	

ANNEX I

ANNEX 1: PROJECT PROPOSAL

DEPARTAMENTO DE ENGENHARIA
INFORMÁTICA E DE SISTEMAS

PROPOSTA DE ESTÁGIO

Ano Letivo de 2020/2021

Mestrado em Informática e Sistemas (Desenvolvimento de Software)

TEMA

Sistema Biomecânico de Apoio à Reabilitação dos Membros Inferiores

SUMÁRIO

Este trabalho de projeto propõe o desenvolvimento de uma plataforma simples de apoio à reabilitação físico-motora baseada na medição da força exercida pelo paciente nos pedais de uma bicicleta estática.

1. ÂMBITO

O trabalho propõe a utilização de dois pedais instrumentados, base principal do sistema, podendo, contudo, evoluir para a introdução de mais sensores, como por exemplo uma unidade de massa inercial.

A diferença na força exercida pelos membros inferiores na pedalada é uma quantificação que pode ser usada, tanto como indicador da evolução do processo de reabilitação como na definição de jogos que permitam, numa linha de *biofeedback*, motivar o paciente.

O trabalho tem a cobertura e acompanhamento do Laboratório de Biomecânica Aplicada, havendo um suporte do Hospital Rovisco Pais e de uma equipa médica deste hospital, que colaboram em todas as etapas de desenvolvimento.

2. OBJECTIVOS

O presente projecto pretende atingir os seguintes objetivos genéricos:

- Análise das soluções existentes no mercado em termos de sensores para medição da força exercida.
- Integração de equipamentos sensoriais na bicicleta.
- Desenvolver uma plataforma de auxílio remoto a integração e utilização desse mesmo *hardware* sob forma de um assistente (remoto ou virtual).
- Integração de jogos que possam ser usados para motivação do paciente.

3. PROGRAMA DE TRABALHOS

O estágio consistirá nas seguintes actividades e respectivas tarefas:

- **T1** – Pesquisa bibliográfica sobre o estado de arte e principais conceitos;
- **T2** – Produção de documentação e definição dos casos de estudo.
- **T3** – Aplicação em casos de estudo com análise, avaliação e proposta de melhoria;
- **T4 -** Desenvolvimento de soluções, baseados no estudo realizado em T1, para os casos de estudo analisados;
- **T5** – Realização de testes e análise de resultados;
- **T6** – Escrita de documentação.

4. CALENDARIZAÇÃO DAS TAREFAS

As Tarefas acima descritas, incluindo os testes de validação de cada módulo, serão executadas de acordo com a seguinte calendarização:

O plano de escalonamento dos trabalhos é apresentado em seguida:

	Meses									
Tarefas	N	N+1	N+2	N+3	N+4	N+5	N+6	N+7	N+8	
T1										
T2										
T3										
T4										
T5										
T6										

5. RESULTADOS

Os resultados do estágio serão consubstanciados num conjunto de documentos a elaborar pelo estagiário de acordo com o seguinte plano:

- *Relatório final de projeto (Tarefa T6)*

6. LOCAL DE TRABALHO

Laboratório de Biomecânica - ISEC.

7. METODOLOGIA

Reuniões periódicas com os orientadores.

DEPARTAMENTO DE ENGENHARIA
INFORMÁTICA E DE SISTEMAS

8. ORIENTAÇÃO

ISEC:

César Paris (cparis@isec.pt)
Professor Adjunto (DEIS-ISEC)

Laboratório de Biomecânica - ISEC:

Luís Roseiro (lroseiro@isec.pt)
Professor Coordenador (DEM-ISEC)

ANNEX II

ANNEX 2: MAGENE MANUAL

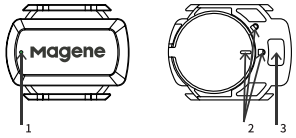
Magene

Bike Speed and Cadence 2-in-1 Sensor

Model:S3+

Magene Dual Mode Sensor Introduction

The product supports both Bluetooth 4.0 and ANT+ protocols and can be used as both a speed sensor or a cadence sensor.
The sensor can be used by most smartphone apps and GPS Head units. Re-installing the battery will switch modes between cadence and speed sensor.
When the battery is installed, the green light flashes to indicate that it is in speed mode, or the red light flashes to indicate the sensor is in cadence mode.



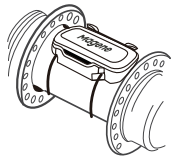
1. Red and green mode indicator light (visible only when battery is first installed)
2. Battery compartment locked / unlocked indicator
3. Silicone gasket mounting position (Cadence sensor mode)

Using the Sensor (1)

* The new sensor comes with a battery insulation sheet that must be removed from the battery compartment before use.

Speed sensor mode

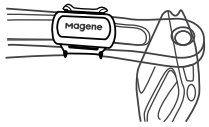
1. Reinstall the battery and the green light will illuminate to indicate that the sensor is in speed sensor mode.
2. Install the sensor on the front hub using a rubber ring.
3. Spin the wheel and search for the sensor using ANT+ or Bluetooth device.



Using the Sensor (2)

Cadence sensor mode

1. Remove and reinstall the battery and the red indicator lights up to indicate that the sensor is in cadence mode.
2. Install the flat silicone gasket on the bottom of the sensor and install the sensor on the inside of the left crank using a rubber ring.
3. Rotate the crank and search for the cadence sensor using an ANT+ or Bluetooth device.



Note: After installing the sensor, please make sure that the sensor and rubber ring do not rub against the shoes or bicycle during riding, so as to avoid damage or loss of the sensor during use.

Device Connection Instructions

Indicator Light	Status
Flashing Green Light	Speed data is being broadcast over bluetooth.*
Flashing Red Light	Cadence data is being broadcast over bluetooth.*
Alternatively Flashing Red & Green	Device Battery Low*

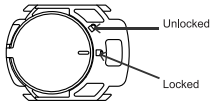
1. The sensor will only start sending Bluetooth and ANT+ broadcasts after it is properly installed and woken-up. Then, you can use the corresponding device or APP to search and connect.
2. When using the Bluetooth protocol, you can only connect to one device or APP concurrently. Please disconnect the previous device or APP when you want to change it.
3. When using the ANT+ protocol, it can be connected to multiple devices at the same time.
4. When using a smartphone app, you need to search for the sensor in the app, searching through the phone system's bluetooth settings is invalid
5. After the sensor is removed, it will automatically enter the sleep state for 1 minute to save power.

Product Specification

Accessories: Sensor ,Silicone pads ,Rubber band , CR2032 Battery	
Weight: 9g	Batterylife: 500h
Temperature: -20°C~50°C	Water Proof: IP66
Protocol: Bluetooth 4.0&ANT+	Sensor Scale: 38.3*29.8*8.9mm
Maximum Speed: 700c 110km/h	

* Actual batterylife depends on using environment.

Battery Replacement



1. Open the battery compartment by turning the position mark on the battery cover counterclockwise from the lock position to the unlocked position.

2. Place the new battery into the battery compartment and press the battery cover into position with the marker aligned with the unlocked indicator (as shown below). After the battery cover is fully pressed, turn the battery cover clockwise to align the indicator to the locked position.

FAQ (1)

1. Which apps and GPS head-units are compatible with the sensor?

A: The sensor is compatible with all devices that support the ANT+ standard protocols, such as; Garmin, Bryton,iGPSPORT, Zwift, Onelap, BKool, TACX, etc and many other virtual training software. When using Bluetooth it's compatible with magene bike computer, XingZhe, etc.

2. Why is the sensor not discovered by other equipment when it is not used for a long time?

A: In order to save power, the sensor will go to sleep when it detects no data for 1 minute. Normal broadcasting will resume when the device is used.

3. Why doesn't the indicator light illuminate when reinstalling the battery?

A: 1) It's possible the battery connector is covered with foreign contaminate or the spring is not pushed up; 2)If the connector is clean and sprung-up, replace the battery with a brand new battery.(battery model is CR2032-3V);
(If you are still unable to solve your problem, please contact online technical support).

FAQ (2)

4. Why can't the newly purchased Magene sensor be found by other equipment?You should check;

1) Check the device is in the correct mode. Red: Cadence, Green: Speed.
2) Whether the software is compatible.
3) Whether there are any inductive magnets causing interference.
4) When searching for the device using a GPS headunit, select the "speed" or "cadence" option, do don't select the "Speed/Cadence"(combined) option.
(If you are still unable to solve your problem, please contact online technical support).

5. Why does a GPS head unit connected to the sensor in speed mode not display speed data?

This is because the head unit is set to prefer speed data from the GPS data. Therefore if you have not obtained a GPS position lock, there will be no speed data. Please change the settings of the head unit to prefer speed data from a speed sensor.

FAQ (3)

6. Why does a GPS head unit connected to the sensor in speed mode not display cadence data?

1) Determine whether the working mode of the device is the cadence mode; 2) check whether the connection with the software is normal;3) whether there is an induction magnet left around;4) if the battery is dead, replace it with a new one. (If you are still unable to solve the problem, please contact online technical support)

7. Is there any delay in the data of the sensor?

The sensor uses geomagnetic sensor measurement data, abandoning the traditional magnet sensing scheme - the installation is more convenient, but there is a certain delay in calculating the data, but the main reason for the data display delay is that the bicycle GPS head unit uses an averaging algorithm to smooth the data.

8. How many hours can the Magene cadence sensor be used?

The battery life is about 500 hours (there will be differences due to the influence of temperature and use environment)

Firmware Updates

Install the Magene Utility app on a Bluetooth-enabled smartphone and connect to the sensor to get the latest firmware.

iOS Download

Android Download

E-mail：support-cn@magene.cn
Telephone：400-662-8297

For more information, please visit:
http://www.magenefitness.com

Multi-language Manual

Manual in multiple languages
can be available from the following link:
https://magenefitness.com/pages/manual

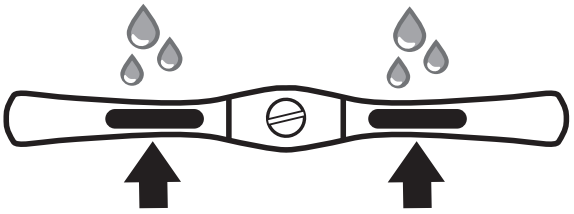
ANNEX III

Annex 3: Geonaute Manual

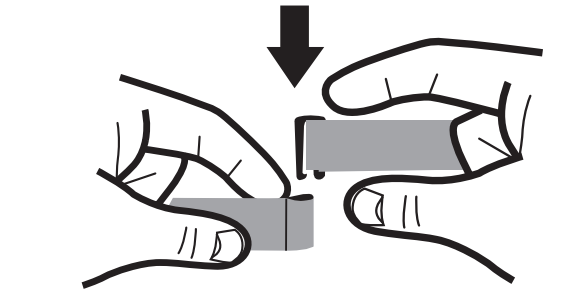
PT Bluetooth® SMART belt

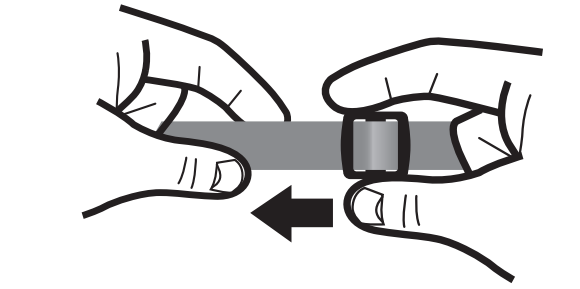
1 Posicionamento do cinto

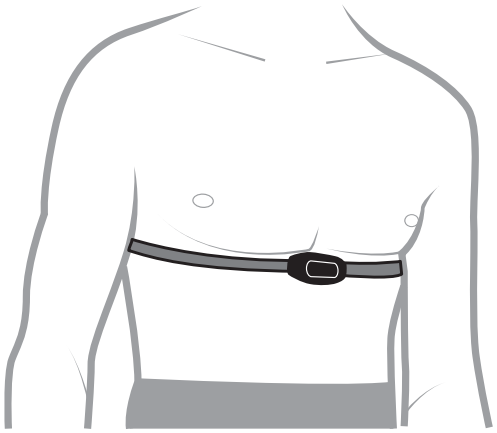
- Humedeça os sensores do cinto.



- Coloque o cinto no torso e aperte.







2 Utilização com um Smartphone

- Depois de o cinto ser colocado, ative o Bluetooth® nos parâmetros do seu telefone.

Os acessórios Bluetooth® Smart não aparecem na lista dos periféricos Bluetooth®. É, pois, normal que não a veja na lista.

- Inicie a aplicação desportiva à escolha.

Certifique-se de que está é compatível com um cinto Bluetooth® Smart.

- Na primeira utilização, deve iniciar um emparelhamento entre o seu cinto e a aplicação. O emparelhamento permite a deteção do cinto pela aplicação.
  O emparelhamento faz-se geralmente nos parâmetros da aplicação.

Para mais informações, consulte as instruções de utilização da aplicação.

- Depois de o cinto ser detetado, pode iniciar a atividade.

3 Utilização com o ONmove 200 e outros produtos Bluetooth® Smart ready

- O cinto Bluetooth® Smart pode ser usado com todos os produtos BluetoothSmart ready compatibles (smartwatch, relógio GPS, etc.).

Para mais informações, consulte as instruções de utilização do produto.

Bluetooth® SMART

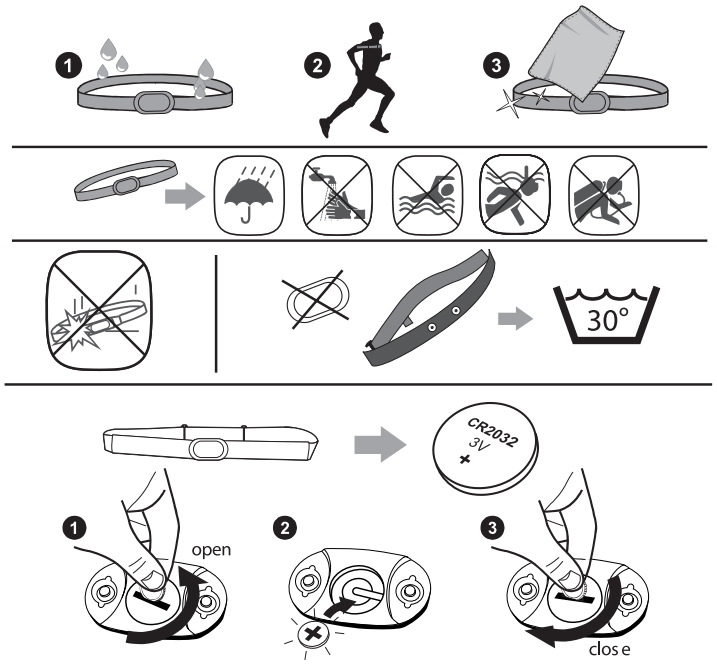
COMPATIBILIDADE

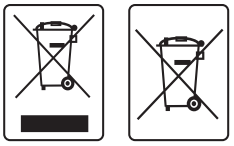
- Smartphones : support.geonaute.com/bluetooth
  > Apple : iPhone 4s, iPhone 5, iPhone 5s, iPhone 5c
  > Android : apenas versão 4.3

Certifique-se de que a aplicação é simultaneamente compatível com o seu sistema operativo e os cintos do cardiofrequencímetro Bluetooth® Smart.

- Relógio GPS Geonaute ONmove 200
- Produtos Bluetooth® Smart Ready

PRECAUÇÃO DE UTILIZAÇÃO





Este produto e as pilhas que contém não podem ser eliminadas com os resíduos domés ticos. Estão sujeitos a uma triagem selectiva específica. Deposite as pilhas, bem como o seu produto electrónico em fim de vida útil, num local de recolha autorizado para os reciclar. Esta reciclagem do seu lixo electrónico permitirá a protecção do ambiente e da sua saúde.

GARANTIA

A OXYLANE garante ao comprador inicial deste produto que este está isento de defeitos ligados aos materiais ou ao fabrico. Este produto tem uma garantia de dois anos a contar da data de compra. Guarde a factura em lugar seguro, dado que é a sua prova de compra.

A garantia não cobre:

- Os danos devidos à má utilização, ao não respeito das precauções de emprego ou aos acidentes, nem a uma manutenção incorreta ou à utilização comercial do produto.
- Os danos provocados por reparações efetuadas por pessoas não autorizadas pela OXYLANE.
- As pilhas, as caixas fissuradas ou quebradas ou que apresentam vestígios de choques.

Durante o período de garantia, o aparelho é reparado gratuitamente por um serviço autorizado ou substituído a título gracioso (de acordo com o distribuidor).

GEONAUTE.COM

oxylane

OXYLANE
4 Boulevard de Mons – BP 299
59665 Villeneuve d'Ascq cedex – France

CE