Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2020

Vasco Matos Maia

Monipart: sistema integrado de monitorização de participantes em experiências com aquisição móvel de biossinais

Monipart: integrated monitoring of participants in experiments with mobile acquisition of biosignals

Universidade de Aveiro Departamento de Eletrónica, Telecomunicações e Informática
2020

Vasco Matos Maia

Monipart: sistema integrado de monitorização de participantes em experiências com aquisição móvel de biossinais

Monipart: integrated monitoring of participants in experiments with mobile acquisition of biosignals

Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Informática, realizada sob a orientação científica do Doutor Ilídio Oliveira, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro

o júri / the jury

presidente / president	**Professor Doutor Joaquim Arnaldo Carvalho Martins** Professor Catedrático, Universidade de Aveiro
vogais / examiners committee	**Professor Doutor Rui Pedro de Magalhães Claro Prior** Professor Auxiliar, Faculdade de Ciências da Universidade do Porto
	Professor Doutor Ilídio Fernando de Castro Oliveira Professor Auxiliar, Universidade de Aveiro

agradecimentos / acknowledgements

I would like to take this opportunity to express my deep gratitude to all, each one in their unique way, who contributed to the development of this work.

Firstly, I would like to thank my mentor, Professor Ilídio Oliveira for his exemplary sense of duty, continued availability and for imparting me with his scientific and technical guidance throughout this process. Secondly, I would like to extend my gratitude to Professor Susana Brás for her unwavering support and willingness to share her knowledge with me during this project.

I would also like to thank all the wonderful people that helped me during the testing phase of the application, Mariana, Lara and Pedro, for giving up so much of your time and the much-appreciated feedback.

Lastly, my family for their unremitting dedication of 24 years and counting…

To all of you, thank you.

palavras-chave

fisiologia computacional, psicofisiologia, biossinais, computação móvel, programação web, microsserviços, full-stack

resumo

A realização de estudos experimentais com recolha de dados fisiológicos de grupos de participantes obriga frequentemente os investigadores a usar equipamentos proprietários, com poucas possibilidades de sair do laboratório ou adaptar os protocolos de recolha. A utilização de dispositivos móveis, em conjunto com sensores selecionados, pode proporcionar uma configuração de baixo custo e portátil para a recolha de dados fisiológicos, para investigação.
O Monipart é uma aplicação web que tira partido de uma solução já existente para a captura de biossinais baseada em dispositivos móveis e que a evolui no sentido de implementar um módulo web para a configuração e monitorização de experiências em tempo real. A aplicação também oferece um ambiente amigável para os investigadores fazerem a gestão dos seus estudos bem como a posterior análise da informação recolhida.

Para isso, a aplicação comunica com a solução de recolha preexistente, baseada em dispositivos móveis, recorrendo a mensagens (MQTT), para controlo e monitorização das experiências. A exploração dos dados obtidos é suportada pelo uso de uma base de dados orientada a séries temporais, para flexibilidade adicional no processamento de dados.
A aplicação desenvolvida permite uma fácil extensão via plugins para incluir novas funcionalidades (e.g.: visualizadores especializados para certos tipos de dados) que podem ser desenvolvidas por entidades externas ao projeto.
Com este trabalho, os investigadores podem agora planear e acompanhar experiências através de uma interface web, bem como ter um acesso imediato aos dados.

keywords

Computational fisiology, psychophysiology, bio-signals, mobile computation, web programming, microservices, full stack

abstract

The execution of experimental studies with the retrieval of fisiological data of groups of participants usually makes the researchers use proprietary equipment, with little to no possibility of leaving the laboratory or adapt the capture protocols. The use of mobile devices in coupled with a variety of selected sensors can provide a low cost, portable configuration for the collection of physiological data for investigation.
Monipart is a web application that takes advantage of na alredy existing solution for the capture of bio-signals based on mobile devices and improves it by implementing a web module for the parameterization and monitoring of experiments in real-time. The application also offers a friendly environment for the researcher to manage their studies as well as further analisys of the retrieved data.

In view of that, the application communicates with a pre-existing collection solution based in mobile devices that uses messages (MQTT) to control and monitor the experiments. The exploration of the collected information is supported by a database designed to handle time-series for added flexibility in the data processing.
The application can be easily extended via plugins that can include new functionalities (i.e. specialized viewers for certain types of data) that may be developed by external entities to the project.
With this application, the researchers now can plan and follow experiments via a web interface as well as have immediate access to the data.

Table of contents

Acronyms

AJAX – Asynchronous JavaScript and XML
API – Application Programming Interface
CSV – Comma Separated Values
ECG – Electrocardiogram
EDA – Electrodermal Activity
EEG – Electroencephalogram
EMG – Electromyography
GDPR – General Data Protection Regulation
HTML – Hypertext Markup Language
IEETA – Instituto de Engenharia Electrónica e Telemática de Aveiro
IP – Internet Protocol
JSON – JavaScript Object Notation
MQTT – Message Queuing Telemetry Transport
OSI – Open Systems Interconnection
REST – Representational State Transfer
SHA – Secure Hash Algorithm
SQL – Structured Query Language
SUS – System Usability Scale
TCP – Transmission Control Protocol
TSV – Tab Separated Values
UUID – Universally Unique Identifier
XML – Extensible Markup Language

List of figures

List of tables

1 Introduction

Data is arguably the most relevant factor when dealing with fact-based science. As such, tools serve an important purpose in order to aid researchers extract meaning from it, from the humble spreadsheet to the most abstract mathematical models.

On the other hand, psychology works towards understanding behaviour and how it relates to the effect of different physical and psychological stimuli, individual and societal dynamics and a range of other factors. The understanding of these phenomena can only be achieved by retrieving information and treating it in order to extract meaning from the collected information [1].

This work aims to do just that, help researchers manipulate and extract meaning from the data collected from experiments involving bio-signals involving participants in studies related to psychology by providing an information system that allows for an easier way to process the large quantity of information generated in this setting and help build a comprehensive picture of the behaviour to different stimuli.

1.1 Motivation

The motivation for this project was the necessity of providing a way for researchers to explore and navigate de information collected during experiments with participants as well as create, manage and monitor experiments in real time.

This project provides an information system that builds upon an already existing product that interfaces with sensors that capture a subject's vital signs which in turn was developed in line with the research group bitMob of IEETA [2], University of Aveiro in collaboration with the Department of Education and Psychology of University of Aveiro for the retrieval of physiological data.

Such a tool allows for the execution of psychologic experiments without the high costs associated with proprietary, professional grade, signal capture machinery whilst providing the researchers with a robust alternative that works outside of a laboratory environment, paving the way for more robust scenarios.

1.2 Objectives

The current work aims at expanding an existing mobile solution for the capture of bio-signals by providing the missing web front-end. The web environment should complement and integrate with the acquisition system to support end-users with the management of studies, participation of volunteers, and acquisition sessions and associated data.

In specific, the system should meet the following subgoals:

1. Monitor the experiment while it is taking place, using a web environment,
2. Explore the data acquisitions directly from the web portal,
3. Export signal data in various file formats,
4. Set up experiments beforehand (participant registry, selection of sensors, …),
5. Development of a plugin system for the analysis of the data,
6. Organize studies and sessions, such that they are readily accessible.

These features should greatly improve the monitoring, analysis and management of experiments capabilities of the current system and reduce the need for manually handle the information between various software tools.

1.3 Structure

This dissertation is divided into seven chapters.

Chapter 1: Introduction – Presents the problem this dissertation seeks to address as well as the motivation and the objectives;

Chapter 2: State of the art – Explains the current tendencies and the evolution of certain concepts in the scientific fields pertaining to the subjects related to this dissertation as well as market tendencies of similar products / services, establishing comparisons between them and outlining defining features;

Chapter 3: Use cases – Gives an overview of the functionality of the system, going into detail for each possible scenario;

Chapter 4: System architecture – Gives an overview of the system architecture and an in-depth explanation of each component;

Chapter 5: Implementation – Explains the implementation of each architecture component supported by their technical reasoning;

Chapter 6: Results and validation – Discusses the practices adopted to verify and validate the application;

Chapter 7: Conclusion – Contains the appraisal of the work as well as the difficulties felt during the development and future work.

2 State of the art

2.1 Methods for bio-signals acquisition in research applications

Experiments involving people in the field of psychology often require being conducted in a controlled environment and isolated from external factors, especially when working in the collection of data for future study. This makes for the necessity of acquiring highly specialised equipment, sometimes with huge costs and the downside of the following required process of training of the health professionals or researchers in order to use the equipment properly.

Other variable to have in mind is the reliability of the collected data from the participants, since part of the collected data relies on the creation of hypothetical scenarios and answering questionnaires that may produce unreliable or false information since it relies on the ability of the participant to recall important aspects of their behaviour or they may feel induced to answer a certain way [3].

However, since the beginning of the century we have been observing developments in the area of mobile computing and the development of sensors with an increasingly smaller footprint and by nature, less intrusive, that allows for a change of paradigm and offer alternative solutions to use in experiments with participants with the added benefit of mobility and a reduced learning curve with the equipment, at least in certain research use cases.

In the area of psychophysiology, getting data from the participants by the researchers via questionnaires is not enough considering psychophysiology is the branch of psychology that studies the relation between psychological and physiological phenomena [4]. The emergence of these new tools is pivotal in obtaining new useful information for the researchers and that better resemble the natural behaviour of the subjects, resulting therefore in more reliable data.

Due to the global adoption of smartphones and the reduction of the cost of buying these devices due to market forces, they are increasingly ubiquitous in the day-to-day with tendency to become more and more an extension of one's self [3].

The omnipresence of these devices opens the door for a new array of possibilities for the collection of information [5] useful for study and research in all sorts of research fields [3].

This scientific area, psychophysiology, in particular has a particular lack of solutions that allow the researchers to conduct experiments without being dependent from stationary equipment to do the required data.

According to this point of view, the production of a solution based in mobile devices that allows control, orchestration and processing the obtained signals from a single point in the context of these experiments, would tend to significantly benefit this kind of collections and the following study.

2.1.1 Diversity of bio-signals

Bio-signals refer to impulses or signals produced by living beings. These signals have a strict relation with the physiological behaviour of a given being and like so, they can be used to quantify specific situations.

In the field of psychophysiology, the capture of signals like electrocardiogram, electroencephalogram, electromyography or electrodermal activity is indispensable in order to enhance the understanding and the study of some phenomena, and in the way that they capture the heartrate, cerebral activity, muscular activity and the rate of produced sweat, respectively.

Regarding ECG (electrocardiogram), the most common way of acquiring this signal is the application of ten electrodes, six of which are placed near the heart and the remaining four are placed in the upper and lower limbs symmetrically [6] as exemplified in figure 1.

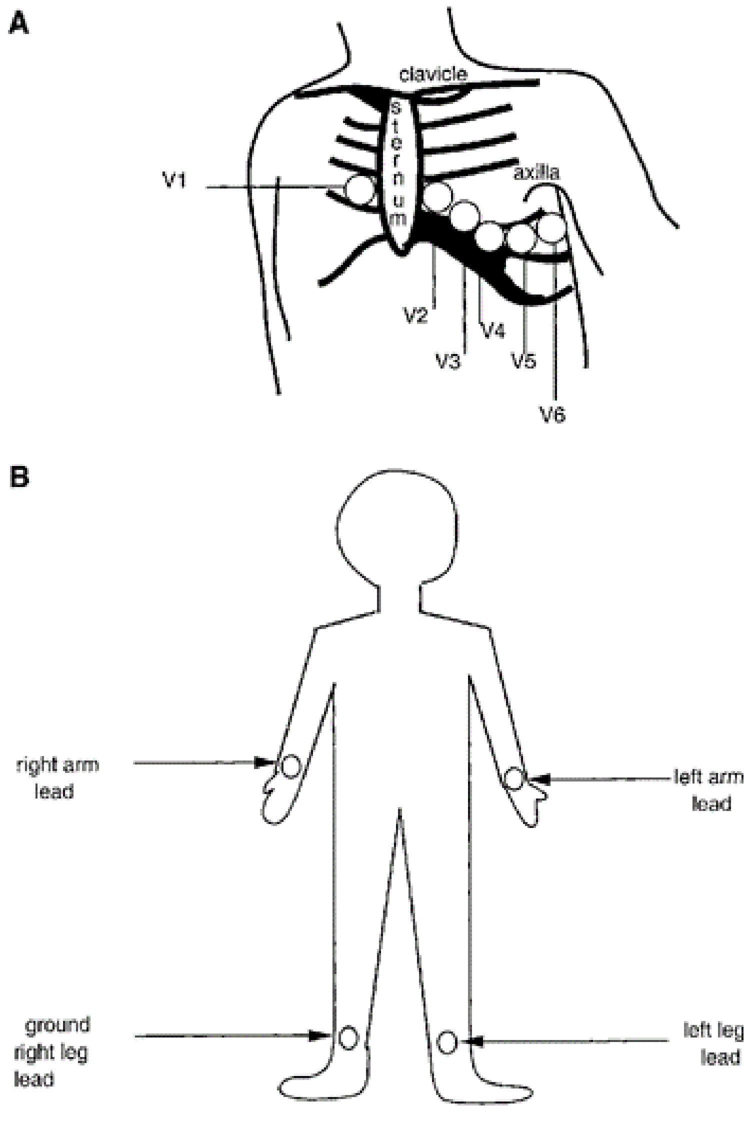


Figure 1. – Placement of the 10 electrodes to obtain the 12-lead electrocardiogram (ECG). A: placement of chest leads. B: placement of limb leads.[1]

These electrodes are then connected to specialized equipment that is not easily portable thus limiting the degree of freedom, and, therefore, the scope of experiments.

[1] https://www.adinstruments.com/blog/correctly-place-electrodes-12-lead-ecg

However, with the emergence of sensors capable of acquiring the same signals, we have been witnessing the emergence of solutions that offer greater mobility and convenience in the way of smartbands, armbands and various garments (wearables) [7].

Concerning EEG (electroencephalogram), it captures the cerebral activity of a person via the application of electrodes placed along the scalp, usually with a cap with the electrodes applied beforehand. The placement of the electrodes must comply with the 10-20 system, which is an internationally recognized method to describe the location of scalp electrodes, and can come in 32, 64 or 128-channel (as depicted in figure 2) variants, according to needs of the procedure.

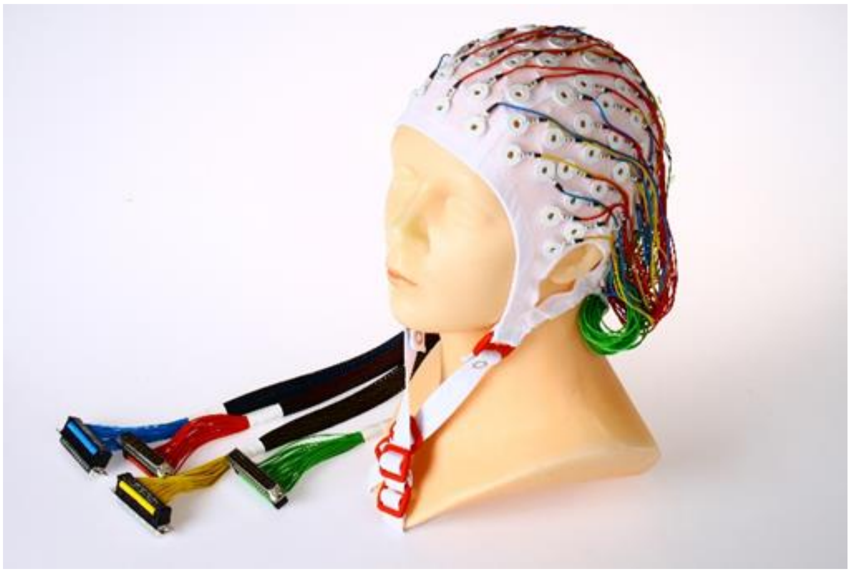


Figure 2. – Electroencephalogram cap featuring 128 channels[2]

Nowadays the collection is made essentially the same way. However, with the emergence of cheaper sensors that are equally as capable in conjunction with the advances in processing capacity of mobile devices, the development of new more economically sensible solutions tend to contribute to a broader adoption of this technology in support to the execution of these studies.

Regarding EMG (electromyography), the technologies used closely resemble the ones used by ECG, with the main difference being the useful measurable bandwidth and the placement of the electrodes. Consequently, the advances seen in technology that benefit EMG data collection, likewise benefit EMG signal collection and processing.

[2] https://brainbox-neuro.com/

Finally, the EDA (electrodermal activity) signal measures the production of sweat by the human body. In the field of psychophysiology and, more broadly, psychology, it is useful as a way of measuring the levels of stress a situation exerted on a subject.

The way this signal is obtained is by passing a small electric charge through the skin and measuring the change of impedance, meaning that the lower this value is, more sweat is produced. The sensors are usually placed on the extremity of the index and middle fingers in order to obtain a useful reading.

2.1.2 Solutions for bio-signals acquisition

With the previously mentioned advancements in technology, nowadays there are several applications that take advantage of mobile devices and sensors in order to deliver a solution oriented to signal processing and analysis [8].

However, while there has been an accelerated development in this area, the solutions that have arisen have been mostly focused on a particular type of signal processing, be it EMG, ECG, EEG or EDA and not many applications actually correlate two or more signals with each other.

As such, a researcher wanting to capture a plethora of bio-signals is not able to do so without having to resort to the more traditional equipment or using several devices/applications at once which greatly increases the complexity of the experiment, limit somewhat the subject’s ability to move freely and creates constraints regarding the orchestration of the devices and synchronization of the data.

There are already reference full-featured, certified-grade solutions designed to be used for this purpose, however there is an emergence of mobile solutions that by and large are good enough for certain research use cases but do not offer the same advanced features as a full-featured application.

Having done a survey of applications whose goal is to take advantage of sensors in a physiological monitoring setting, the following are an approximation of the current available solutions.

The application Kardia[3] is an Android and iOS application that allows a subject to keep track of their heartrate via a wireless sensor where the subject places the index and middle fingers of both hands which severely limits the ability to perform other actions.

Other applications like Sleeptime[4], make use of internal sensors of the device. This one uses the microphone in order to determine sleep patterns and movement during the sleep and presents a comprehensive analysis of the person's behaviour.

Whilst these two applications allow monitoring of vital signs, they are designed with an inexperienced and not medically trained end user, thus not really being suitable for a scientific setting.

The following two applications, on the other hand, are more geared towards a more advanced setup.

PurpleRobot[5] is an application that supports a broader range of sensors in order to create automated experiences with sensing and inference capabilities. This has been used mostly in order to detect depression.

AndWellness [12] is a platform used mostly for behavioural experiments divided in two main parts, the mobile device, capable of obtaining data through its sensors or external sensors and a web page that shows the collected data and allows for some treatment of the information, namely, finding relations between the data.

The following table gives a succinct overview of the discussed solutions and puts them in direct comparison with each other.

Name	***Platform***	***Signals***	***Obs.***
Kardia	Android, iOS	ECG (external sensor)	---
Sleeptime	Android, iOS	Audio	Comprehensive report and sleep pattern analysis
PurpleRobot	Android	GPS, Accelerom.	Open Source
AndWellness	Android	Microphone Accelerom. Proximity Sensor	Web page allows further study of the data

Table 1. – Comparative analysis of current consumer-grade applications for mobile devices

[3] https://www.alivecor.com/kardiamobile
[4] https://www.azumio.com/s/sleeptime/index.html
[5] https://github.com/cbitstech/Purple-Robot

The conclusions we can draw from the obtained results are that most applications take advantage of the internal sensors commonly found in most smartphones and tablets such as GPS, accelerometer, microphone and camera but offer little to no support for external sensors / devices, except some specialised applications.

Other aspect to take into account is that the vast majority of applications are developed for the Android operating system, however, a considerably smaller amount has support for iOS or both operating systems.

This way, a gap in the market for an application suitable for handling and taking advantage of external, low cost sensors has been identified. Also, the development of such application should be considered for iOS as well as Android considering that the market share of iOS devices is not negligible.

Regarding desktop computers, there are already some applications that fulfil these criteria namely the application OhMage, which also has a mobile version, that allows for the creation of studies, retrieval and analysis of participatory sensing data. The data gathered by this application are location, movement, sleep control and audio. The studies are questionnaires created in a web environment in which the data can be analysed or viewed. This application has been used in a variety of studies, research and medical applications regarding depression and behavioural issues.

Regarding sensors and wearables, separated from any kind of mobile application or otherwise, there are a few commercial third-party solutions currently worth considering.

BioPac is a North American company that manufactures sensors geared towards collecting physiological data and that is directed to the research and education markets [13]. They supply such products as eye tracking hardware that in the realm of psychology can be used in the study of autism as a way to measure the level of social attention and social motivation.

This is a closed, non-portable system largely used by the scientific community and is not open source, however, being considered the gold standard, all mobile applications developed to the same end should try to reach the same levels of quality, with the added benefit of portability and lower cost.

Other interesting wearable device worth considering is Vital Jacket. This jacket monitors signals like ECG, heartrate and body temperature using sensors embedded in the fabric. This device, developed by a Portuguese based company, has been used previously in clinical trials and in scenarios of search and rescue as well as forest fires, in order to monitor firefighters [14] during missions.

Also, there is a signal acquisition board, in a way, similar to an Arduino called BITalino.

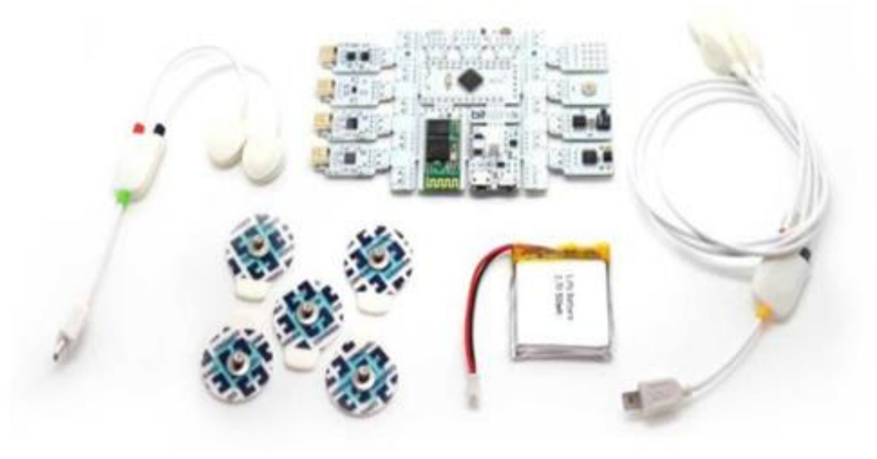


Figure 3. – BITalino board[6]

This is a microcontroller similar to an Arduino. However, is specially designed to capture bio-signals. This board can capture EMG, ECG, EEG and EDA and also contains an accelerometer and a light sensor. The capture rate is configurable and can be set to 1, 10, 100 or 1000Hz.

This comes in several different configurations such as the board on its own where you can only use its internal sensors, the board with modules capable of connecting to an array of different sensors and a version where all individual blocks are outside the board, allowing you to create a custom board tailored to your needs.

[6] http://bitalino.com

2.2 Web development strategies

This project is composed of several different parts, one being a web platform with which the users of the system will interact with.

As such, an understanding of the evolution of web development strategies and architectures are necessary in order to choose the technologies best suited to handle the requirements of the system. The following sections of this chapter go into detail about the tendencies and evolution of the technologies in this area.

2.2.1 Web application architectures

Web application architectures are constantly evolving to meet the needs of an increasingly connected world. These changes are intrinsically related to the advances made in computational power and the need for managing larger and larger amounts of data.

There has been a shift from a static environment to a more dynamic one, allowing for an increase in complexity. Nowadays, given the current state of technology, it's feasible the creation of web-based real-time monitoring systems that benefit from being device and operating system agnostic.

There are several architectures currently in widespread use, detailed below, each with their specific use cases and peculiarities. The next sections detail each of these architectures and their utility in the current web landscape.

Server-side HTML web applications

Server-side HTML web applications (Figure 4) are quite ubiquitous in the current web landscape. The basic concept of this architecture is that the server generates HTML content and sends it to the client as a full-fledged HTML page. This architecture is also known as "Web 1.0" and currently dominates web development.

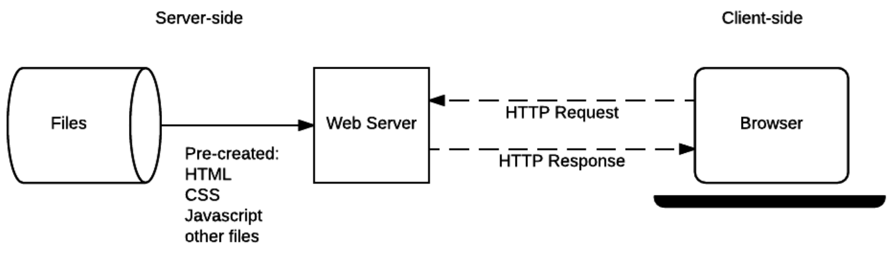


Figure 4. – Example of server-side HTML web application architecture[7]

However, one flaw associated with this architecture is that a large amount of information is exchanged between the server and the client, and the client is forced to wait for the entire page to load, even when dealing with something as simple as user interaction whereas only part of the page reloading would suffice. This severely limits the scope and complexity of applications and their intended use, as this technology is out of question when dealing with frugal data plans such as those provided by cellphone carriers.

This architecture also makes it impossible to send instant data updates or changes in real time.

Regarding scalability, there comes a time when, if the demand is high enough, there is the need to implement some sort of load balancer and replication across several nodes (horizontal scalability), however not much complexity is involved as each node can operate independently of each other. In the other hand, vertical scalability is always possible. However, there is a theoretical ceiling to the performance gains.

Concerning performance, this aspect is directly related to scalability. However, the performance yield of a system is comparatively low considering the large amount of information that must be transferred per user containing HTML, design and business logic.

One advantage of this kind of architecture is that all application behaviour logic resides in the server side, thus posing an advantage when dealing with security issues. However, regarding any client-server architecture, a secure channel of communication must be privileged as to assure the information exchanged is not tampered.

[7] https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Introduction

AJAX based web applications

This architecture is an evolution of server-side HTML applications, with the main differentiator being that the page can be partially or completely composed of independent units that each communicate with the server and react accordingly.

Data can be exchanged to these units from the server as HTML or JSON and be integrated into the context of the page. This opens the opportunity for pages to be partially changed without the need to reload the entire page.

This technology reduces the amount of data transferred on the whole, increasing responsiveness regarding the previous architecture. However, the use of multiple AJAX widget per page will cause longer loading time for the first full loading of the page since in this architecture, time is spent on retrieving the information from the server for the widget whereas in the previous architecture, the information was promptly served and even faster if the information is cached.

However, after the first load of the page, the rest of the experience is much more seamless and pleasant to use.

Overall, the time spent on HTML generation is lower and traffic usage is substantially lowered, specially if the data being exchanged is done so in JSON format. This, however, adds an extra layer of complexity, as this information needs to be processed and translated into its HTML counterpart in the page.

This processing falls under the responsibility of the client. This means that computing requirements are slightly increased and some of logic is shifted to the client, which can be tampered by a malicious actor.

Service-oriented single-page web applications

In this paradigm, an HTML page is retrieved from the server. This page is a container for JavaScript code which addresses a web service and fetches business data only. This data is then used by the application which generates the HTML (Figure 5).

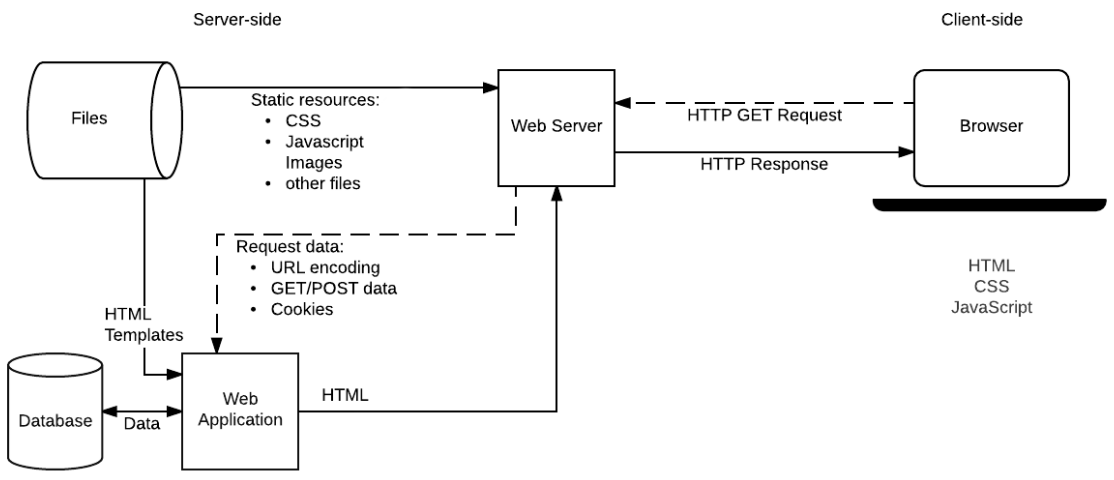


Figure 5. – Example of service-oriented single-page web application architecture[8]

This architecture ensures that the volume of data exchanged is minimal, but if there is the need of performing computationally heavy tasks on the data, this responsibility should be deferred to the web service, since JavaScript is single-threaded and client hardware is not expected to handle heavy computational loads.

Considering the web logic is entirely on the client side, this means that there is not content generation on the server, meaning that in terms of scalability, only the web services that give the business data are required to scale.

With the shift of the logic to the client, this means that the system is susceptible to tampering and thus the implementation of a preventative architecture on the web services is a must. Such an architecture should never trust the client and like so, should implement a set of checks and balances to validate all received information.

[8] https://developer.mozilla.org/en-US/docs/Learn/Server-side/First_steps/Introduction

2.2.2 Review of selected frameworks for web development

According recent polls[9] [10] the three main front-end frameworks being used for web development are React.js, Vue.js and Angular. This information greatly influenced the selection process for what framework to use in this project.

React is the front-end framework created and developed by Facebook. The team was interested in obtaining high performance by building an effective UI.

To do so, React implements two crucial features:

1. It works with the virtual DOM and applies abstract copies of the real DOM, therefore it updates all the changes of the user but has no effect on other parts of the interface,
2. When applying any updates to a page, only the chosen components will be changed.

Vue.js is the web framework for building user interfaces. It can be used for both represented components and complete single-page applications. It includes component file layout and logical structure. This framework deals with two-way reactive data-binding and doesn’t demand any extra libraries.

Angular is an all-encompassing solution and truly a framework instead of a just suite of libraries. Developers can focus more on accomplishing their tasks rather than looking for libraries and solutions for their tasks. It is based on TypeScript, which brings all of the advantages it provides: arrow functions, async/await, class syntax, etc. The introduction of well-known OOP practices makes the transition much easier for shifters from languages like C# and Java.

The aggregation of the pros and cons of all these three frameworks (table 2) and considering the overall support from the developer community, React was the chosen framework for the development of this project.

	React.js	***Vue.js***	***Angular***
Pros	• Fast DOM manipulation, • Reusable components, • Open source, • Large developer community	• Simplicity, • Extensive documentation, • Reusable components, • Extensions	• Component-based architecture, • High performance, • Large ecosystem
Cons	• Large learning curve	• Smaller developer community, • Too much flexibility can lead to code irregularities	• Very complex, • Large learning curve, • Poor documentation

Table 2 – Comparison between the three major front-end frameworks

9 https://insights.stackoverflow.com/survey/2019#technology
10 https://2019.stateofjs.com/front-end-frameworks/

3 Use cases

The main objective of this project is to create a system that builds upon an existing mobile solution for the capture of physiological data, provides a web interface for setting up and managing experiments, and provides core analysis and visualization to aid the researcher without the need for external programs.

Given the partnership between IEETA's bitMob research team and the Department of Education and Psychology of University of Aveiro, a number of requirements have been put forward, the main being the possibility of using this tool to visualize and export segments of a signal.

The participants are generally volunteers that may or may not be subjected to a selection process where they are evaluated for their suitability with a questionnaire or during an interview.

Having had the opportunity to accompany several experiments (in the context of a project conducted in connection to the Integrated Masters in Biomedical Engineering degree), in which this system was being used, the way the data collection is done can vary from study to study but by and large they can be separated in two distinct groups, a brief questionnaire done before the signal capture is initiated and by taking some notes during the experiment and, in the other hand, using tools to gauge the reaction and physical response of a subject during the experiment.

3.1 Selected workflows in research studies with physiology acquisition

When learning the way the research studies are conducted and the workflows in this domain, there are some aspects that the system ought to represent carefully in order to preserve those workflows.

In a typical study, there may be more than one researcher associated with it, with one being designated the responsible researcher, that will define a protocol and mostly conduct the experiments that will collect the data for further study.

The other researchers may also have access to the information and be able to analyse it independently from one another.

When defining a study, it can either be conducted with several participants where each session is independent from each other or be conducted with several groups of participants where during a session, all the members of a group may be monitored simultaneously, namely studies where group dynamics or conflict resolution are the focus of the assessment.

For various reasons, such as not influencing the outcome of the experiment, the participants do not have access to the collected information, at least during the study itself. An average study will have the participant answer a questionnaire at the start of the experiment in order to gauge their suitability for the study and existing preconditions that might influence the results. After that they are subject to some sort of stimuli during which their physiological markers are recorded.

Finally, the participant may or may not be asked to fill in another questionnaire at the end of the study in order to evaluate the participants perspective of the study.

The developed system does cater to these workflows in regard to the researchers and what information they are able to access as well as the definition of new studies with the various possible arrangements of participants.

3.2 Existing mobile solution

The previously developed solution is a system composed mainly by mobile components to support the collection and aggregation of physiological data using mobile devices[12].

The solution makes it easy to extend the array of supported sensors and monitor ongoing group experiments with several participants. The system, Vitals Recorder, includes two main modules: an Android application for the collection of physiological data, running on a smartphone associated with an individual (VR-Unit) and a monitoring application that runs on a tablet associated with the coordinating researcher (VR-Remote).

In this module, the researcher can manage the group of participants, mark events of interest and inspect, in real time, the data of the various participants. Data from different sessions is consolidated in a backend, which allows previewing on the web and export to several formats.

However this system presents some limitations that this project aims to eliminate, namely, being able to manage and configure experiments and sessions, have ready access to the collected data without having to use third party programs, provide a suite of tools that perform basic statistical analysis and exporting the information to be used in other environments. This project offers all these missing features from the previous solution in a cohesive web interface.

The previous system only allows for the capture and storage of information in a Dropbox folder, with all the processing being done manually afterwards and by means of third-party software.

3.3 New use cases

The application developed for this project needs to cater to the given specifications while supporting all aforementioned interactions between the system and the various actors (table 3) that play a role in it.

Actors	***Description***
Participant	Is the user of the system that participates in the experiments. This actor has a very limited interaction with the system, and may not access or manipulate any information regarding the studies / sessions.
Researcher	Is the user who sets up the experiments and associates participants or other researchers to it. This actor also uses the system for tracking experiments in real time and analyse the retrieved information.
VitalsRecorder	Is the Android monitoring solution that captures and sends data to the system in real-time.

Table 3 – Specification of the intervening actors of the system

The use case diagram below (figure 6), illustrates all the different scenarios that can take part in the system and how each actor relates to it.

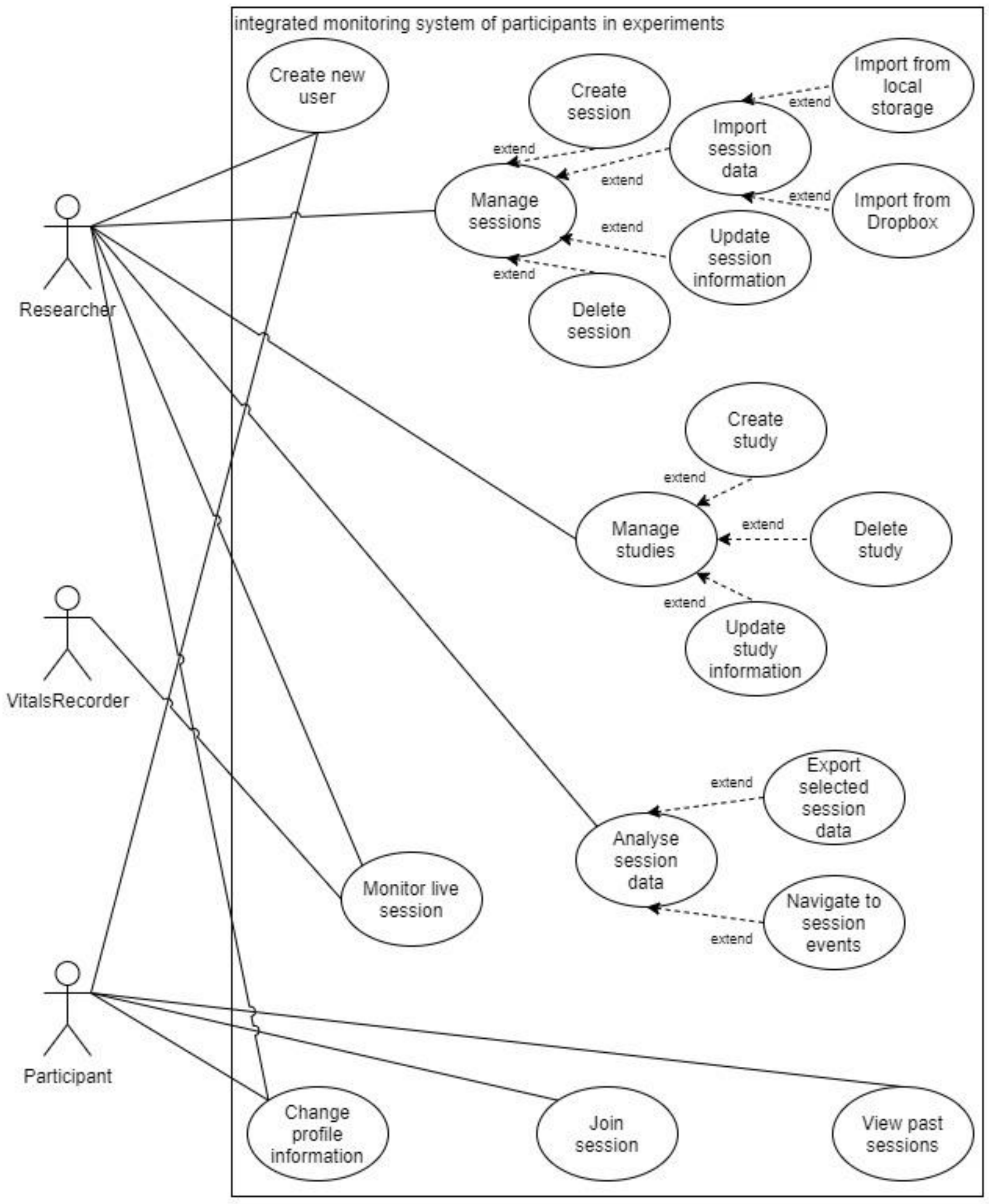


Figure 6 – Use case diagram of the system

Create user

Both the researcher and the participant actors can perform this action at the login screen.

When creating a new user, the type of user must be specified, either participant or researcher. Other information like profile picture, email, password and date of birth must also be supplied or fallback to default values.

Change profile information

Once authenticated, both actors are capable of changing their personal information in their profile page. These changes will be permanent and will not affect previous studies / sessions they’ve been part of.

Join session

A participant may be able to join an ongoing session to which they have been invited. This functionality can be bypassed by the already existing capabilities of the Vitals Recorder application. This step ought not to be essential in order to conduct a capture session.

At no point during the session any information related to the capture and the values therein should be made available to the participant.

View past sessions

A participant may be able to view a listing of past sessions they have been invited to participate. Similar to the scenario where the participant joins an ongoing session, no information related to the captured data should be made available to the participant.

Manage studies

A researcher may be able to perform actions that directly impact the studies themselves such as create, delete or update study information.

This scenario should be allowed for all researchers associated to a study, not just the main researcher that initially created the study.

Manage sessions

A researcher may be able to perform actions that directly impact the sessions of a given study such as create, delete, update information or import data for external sources.

This scenario should be allowed for all researchers associated to a study, not just the main researcher that initially created the study.

Import session data

In this scenario, the system may support the sourcing of data from either local storage or a dropbox folder previously known to the system, in the format exported by the Vitals Recorder application.

Monitor live session

In this scenario the researcher may be able to watch in real time the data captured by the sensors during a live experiment. Due to the transient nature of the task, it’s not expected to do any kind of signal analysis or manipulation during this time.

Analyse session data

The researcher has at their disposal a suite of tools that allows them to do some sort of signal analysis, gather some basic statistics and export the data to a number of file formats.

4 Monipart system architecture

4.1 System overview

The proposed system is divided into five main components. The first is the frontend which is tasked with the interaction with the user. The frontend communicates mainly with the backend via HTTP and web sockets live data. Regarding the import of existing sessions, the architecture allows for communication with the user's device's local storage or connect to a remote Dropbox folder.

The second component is the backend. This serves as support system for the frontend and mediates the communication between all the other components via a REST API and web sockets in the case of live session data. The main purpose of this component is to provide the frontend with information from both databases as well as the message broker and manipulate both databases.

There are two databases that are used to retrieve and store data in a persistent manner, with the main difference between the two being one is optimised to work with timeseries data, therefore, all session measurements taken during a live session or imported are stored in this database, whilst all other information pertaining to the rest of the system including session information is stored in the other.

Finally, the message broker is used to relay the messages coming from the sensors and distribute it to its subscribers with very low latency. In this case the subscriber is the backend, which, in turn, relays the information to the timeseries database and the frontend.

Moreover, all these components are designed with docker deployment in mind, and as such, the system can be deployed in a few minutes with little configuration needed via docker-compose.

The following image gives an overall picture of the architecture of the system and its main components as well as the technologies adopted for the development of each part (figure 7).

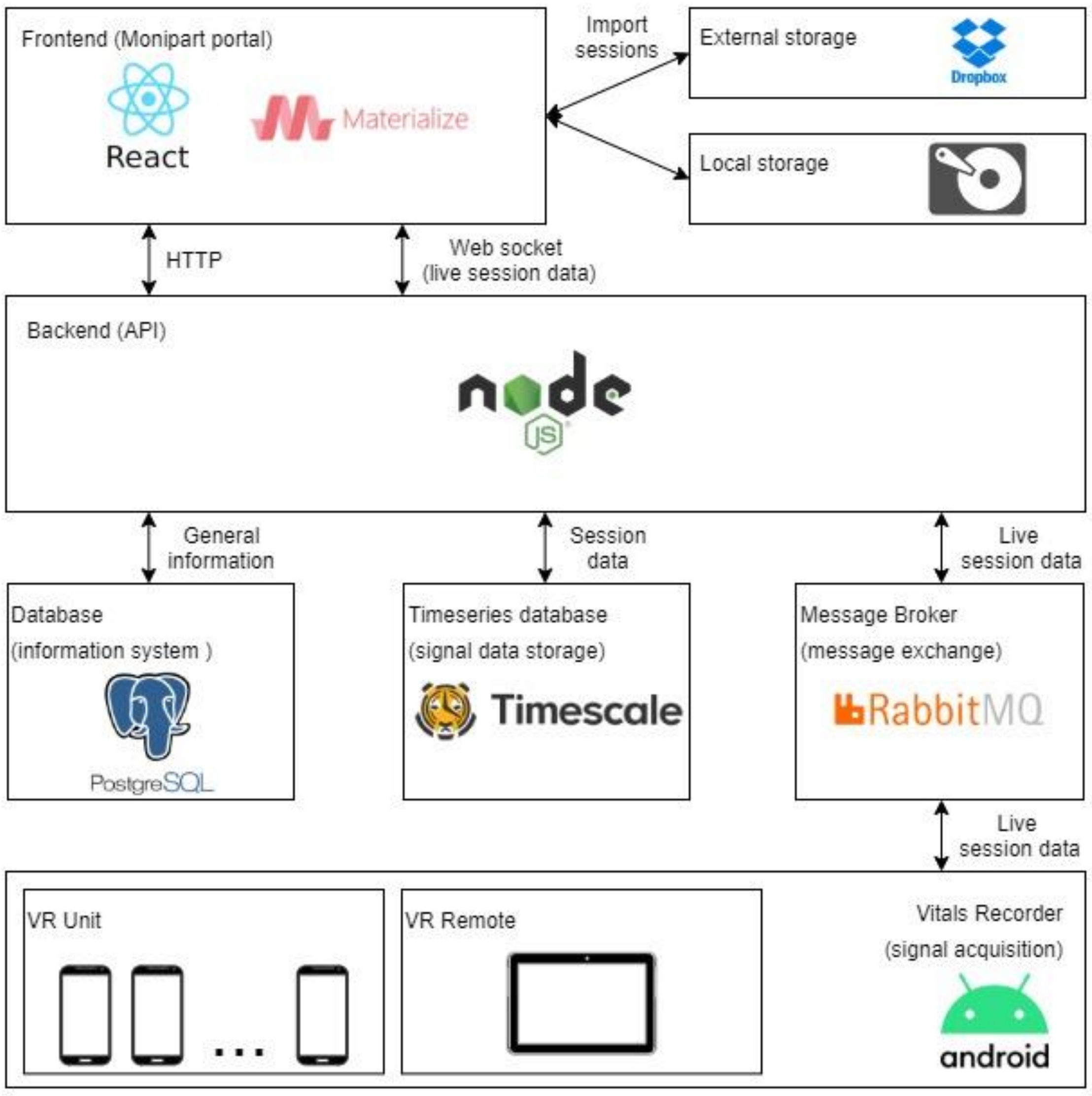


Figure 7 – System architecture overview and technologies employed

Regarding the technologies used for each component, they were chosen based on their suitability for the task at hand and specifically regarding the frontend and the backend, they were chosen partly by their increasing market share and wide adoption rate.

4.2 Main architectural modules

4.2.1 Frontend

The frontend is the part of the system with which the user interacts and like so, several things have to be taken in consideration when projecting such a system. A user interface should be able to present the information in a straightforward and unambiguous way such that it remains simple, intuitive, consistent and predictable.

As such, the system was designed following a set of rules that are as follows:

- **Responsiveness:** The pages and their components adapt to a range of screen sizes without reducing functionality, allowing for an effective use in a plethora of devices;

- **Consistency:** All the pages follow the same design language and ensure the availability of the same functionalities across the entire platform, such as user settings and logout;

- **Simplicity:** The terms used across the different interface elements follow a simple and non-technical language pattern;

- **Feedback:** When executing several commands, the interface communicates the success or warnings of said actions such as missing fields in forms, deleting studies/sessions, importing data…

The interface is divided in two contrasting parts, the participant interface and the researcher interface. While the participant interface is only used to pair the devices with the experiment and can be bypassed altogether using the already existing mobile application, the researcher interface is much more the focus of this system.

There the researcher has at their disposal a series of tools and mechanisms to handle the creation and manipulation of studies and sessions while allowing access to other researchers as they see fit, live monitoring of ongoing sessions and basic signal analysis tools.

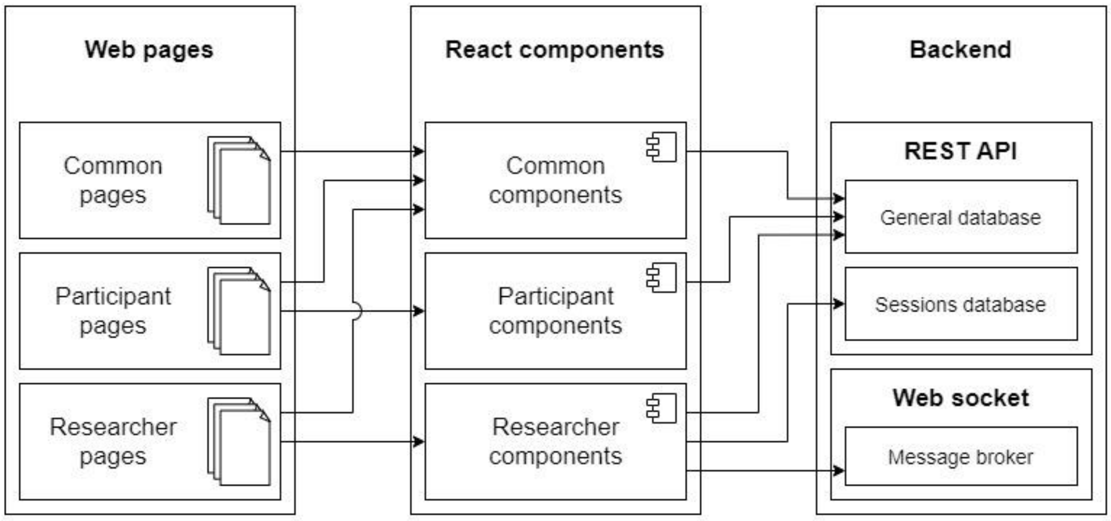


Figure 8 – Logical organization of the frontend

As depicted in the diagram (figure 8), the logical organization of the frontend is laid out in a somewhat concise manner. There are two big distinct categories, web pages and components. The former is nothing more than an agglomeration of components that can exist on their own or cooperate with each other with some logic.

Both web pages and the react components are divided in three separate groups: common, participant and researcher. This ensures that there is a logical separation of concerns and that each component is only has access to the strictly necessary information.

If a user tries to access a page they don’t have permission, the system in will not show the requested page and will show an access denied message instead. Also, if a user tries to load a non-existent page, the system will fallback to a default page with an error message.

4.2.2 Backend

The backend acts as an abstraction layer between the frontend and the multiple sources of information by providing a uniform interface and a layer of security that mediates unwarranted database calls by third parties.

The backend of the project is divided into two distinct parts: a REST API that relays information from the various databases to the frontend via JSON objects and can also serve files and images in base64 encoded format and a web socket that relays information in real time from the message broker to the frontend.

REST is an architectural approach to the communication between a server and a client over HTTP and is often favoured over its more robust counterpart SOAP because of the lower bandwidth use that makes it more suitable for efficient internet usage.

In the case of the implemented system, all API requests with the exception of a few such as login, logout and some others, are routed via a middleware that checks for login authenticity via an access token that is generated for the user at login and deleted at logout.

This token must be sent in every API call in order to obtain a response, otherwise the system assumes the user is not logged in and bars the access. Furthermore, even if the user is a valid user and is logged in, the information will only be served if the requester has the needed permissions for such a request. This ensures only researchers have access to the data collected during experiments and that they only have access to studies / sessions they have been associated.

4.2.3 MQTT message broker

The message broker acts as a buffer between the Android application that retrieves the sensor information from the experiment subject and the backend which in turn stores the information permanently in the sessions database and reroutes the information to the frontend. As such, a MQTT message broker is the tool most suitable for the task.

The MQTT protocol is a lightweight publish-subscribe network protocol that transports messages between devices. The protocol is part of the application layer of the OSI model and usually runs over TCP / IP but can be supported by any network protocol that provides ordered, lossless and bi-directional connections [15].

The publish-subscribe architecture works similarly to a notification system where subscribers are independent of other subscribers. When publishing messages, it is not required for publishers to specify a consumer nor does the consumer have to specify anything about the publishers. Instead, the messages are manipulated in such a way that the consumer subscribes to receive only certain types of messages.

Using this architecture, the detachment of the various components of the system is possible due to the fact that it allows for multiple subscribers.

This protocol defines two types of entities: a message broker and clients. The broker is a server that receives all messages from the clients and routes them to the appropriate destination clients. A client can be any device that runs an MQTT library and connects to the broker over a network.

There are several MQTT Message Broker solutions in the market with Apache Kafka[11] and RabbitMQ[12] having the biggest market share. Considering the demands of this project, RabbitMQ outperforms Apache Kafka in terms of throughput, however, if scalability were to be a determining factor, Apache Kafka would be the most suitable choice [15].

The way RabbitMQ works ranges from the very simple and rudimentary to the very robust and complex. RabbitMQ is composed of a few simple building blocks which include publishers, consumers, queues, topics and exchanges [15]. The most basic setup possible is to have a publisher, a queue and a consumer (figure 9).

[11] https://kafka.apache.org/
[12] https://rabbitmq.com/

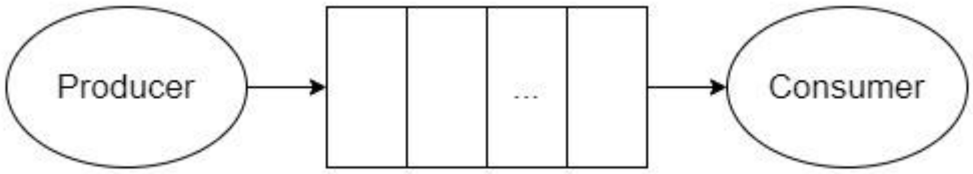


Figure 9 – Example of simple producer, queue and consumer setup

Since queues work by employing load balancing algorithms in order to distribute the messages among the subscribers, this approach is not suitable to have several consumers take the message and process it independently of each other.

This can be solved by using exchanges. There are different types of exchanges (table 4) which rule the way it treats messages, but the concept of operation is relatively simple, on one side it receives messages from producers and the other side it pushes them to queues (figure 10).

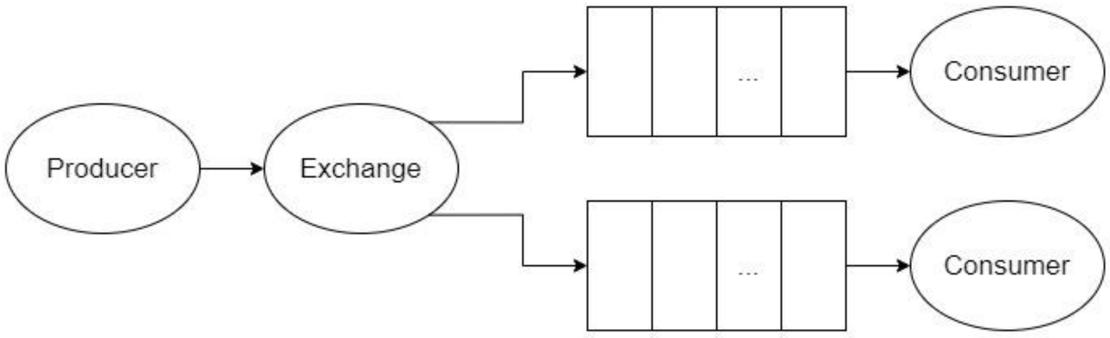


Figure 10 – Example of RabbitMQ fanout exchanger functionality with multiple queues

The fanout exchanger broadcasts the message sent by the producer to all queues it knows about. This way, each consumer is assured to receive every message instead of just one of the consumers, because no matter which load balancing algorithm the queue uses, (round-robin by default), all messages are sent to the one consumer.

Types	***Routing criteria***
Direct	Routes messages with a routing key equal to the routing key declared by the binding queue.
Fanout	Routes messages to all bound queues indiscriminately. If a routing key is provided, it will simply be ignored.
Topic	Routes messages to queues whose routing key matches all, or a portion of a routing key.
Headers	Routes messages based upon a matching of message headers to the expected headers specified by the binding queue.

Table 4 – Specification of the different types of RabbitMQ exchange types

The broker can also organize information in topics that the clients can publish messages and subscribe to receive messages sent by other clients. Basically, a message sent with a particular routing key will be delivered to all the queues that are bound with a matching binding key (figure 11).

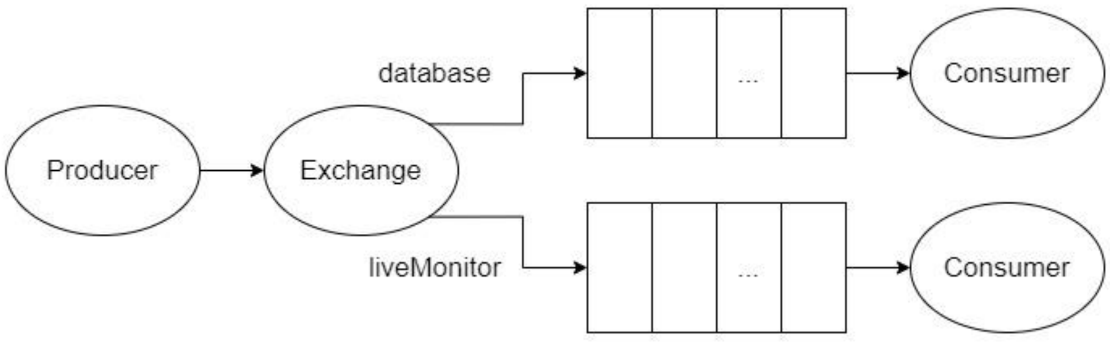


Figure 11 – Example of RabbitMQ topic exchange with two distinct routing keys

As per the above figure, the messages will be routed according to their routing key only to the queues with matching binding keys. These binding keys can be more robust with the use of wildcards, allowing for more complex systems.

In summary, the use of a message broker is integral in establishing a message passing mechanism that works in real time and has the flexibility to articulate the communication between all the different components of the system.

4.3 Domain model

The domain model developed for this project is geared towards supporting both an information system for experimental studies and the associated metadata and in the other hand, the handling of millions of entries that describe such experiments.

With those requirements in mind, there were created two distinct models that deal with the specificities of each domain.

4.3.1 The experiments information system

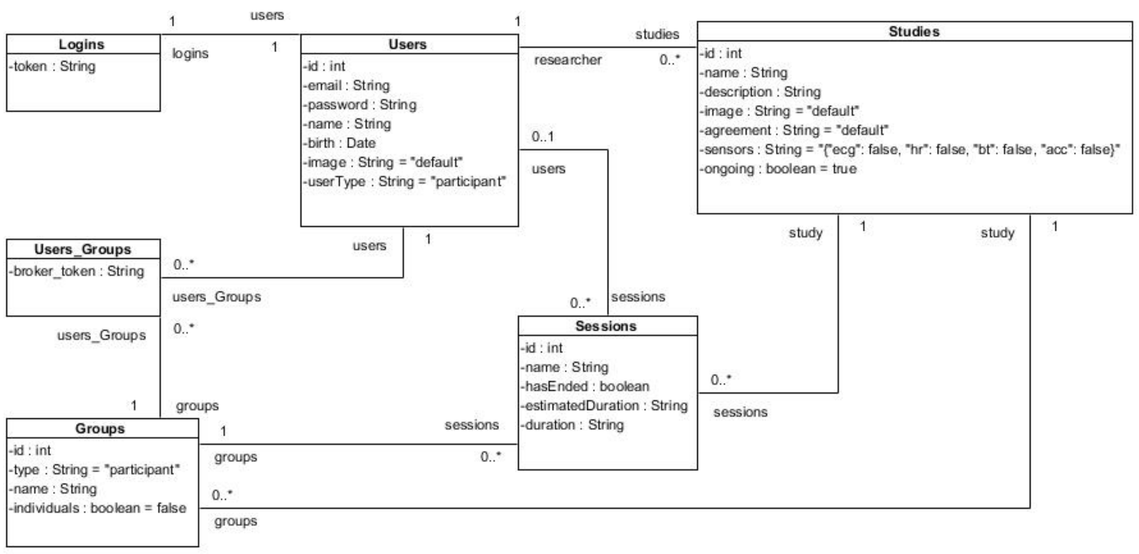


Figure 12 – Domain model of general database information system

The model depicted above (figure 12) handles all the information concerning the metadata of studies / sessions as well as user registration and authentication. Concerning the storage of user credentials, such as passwords, only a hashed version of the value is stored, minimizing the immediate impact in case of an information breach.

A user, which can either be a researcher or a participant as defined by their user type, is assigned a token in the logins table while the user is logged in and is valid for the entire duration of the user session. This token is deleted when the user logs out. This mechanism ensures the validity of the API calls made to the backend.

Studies can be created by researchers and which are associated as the main researcher of said study. A study can have a group of invited researchers. These researchers have the same access and control over the study as the main researcher.

When creating a study, the association of participants can be made one of two ways:

1. The researcher can specify one or more participants as individuals; or

2. The researcher can specify one or more groups of participants, each consisting of one or more participants.

Either way, in order for a study to be created, there needs to be one or more people associated as participants.

The sessions are created in direct relation to a study. A session must be associated to either one and only one participant from the pool of participants from the study or one and only one group of participants from the pool of groups from the study. This way of managing the association between participants and sessions allow for experiments where sessions are conducted with each participant at a time or with multiple participants at a time.

4.3.2 Timeseries database for physiological data

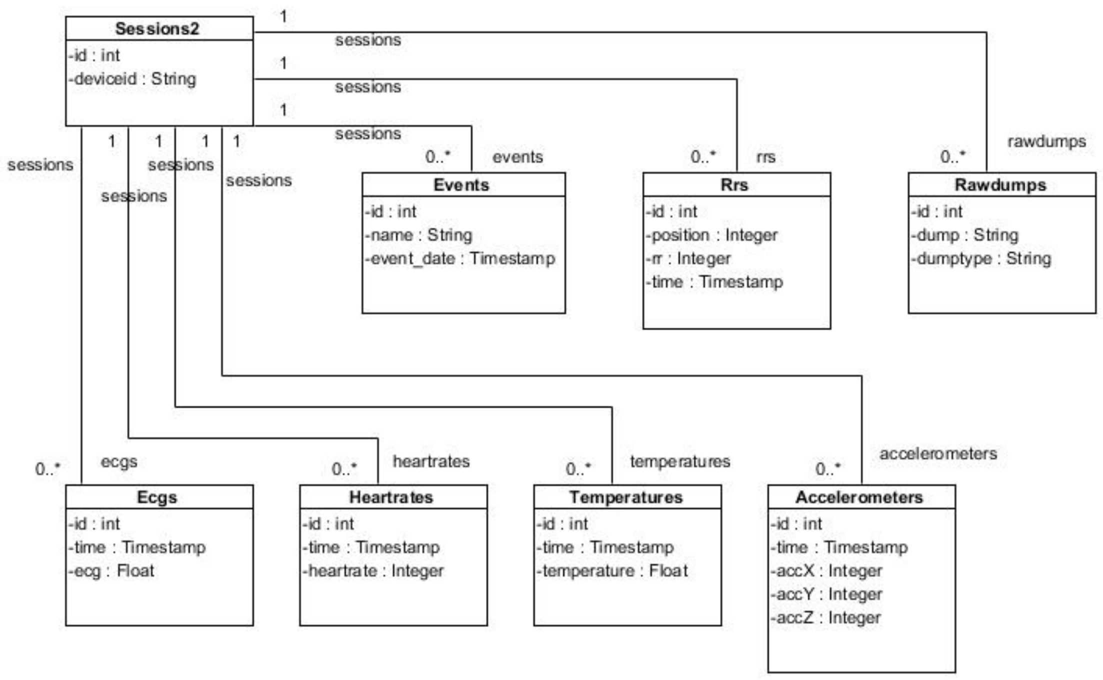


Figure 13 – Domain model of the timeseries information system

The model depicted above (figure 13) handles all the information that actually make up a session, all the retrieved measurements from the sensors such as ECG, heartrate, temperature, accelerometer information as well as points of interest during the experiment or events.

This model was conceived with the goal of being able to manipulate and retrieve large amounts of information by time.

The data stored by this model can be related to the system information database as the entries in the table sessions in this model share the same identifier as the sessions in the other model.

4.4 Plugin support

In computing, a plugin is an external component that adds a new feature to an alredy existing piece of software. These components can vary from cosmetic enhancements to more complex logic that further expands the utility of a program.

In order to increase the flexibility and expandability of the system and make it easy for developers external to the project to contribute with new features, the application has a plugin system that integrates the new code seamlessly with the current system.

This plugin system was conceived with the ability to operate over the various signals in mind and as such, all the information regarding the collected signal data is made available for the developer to interface with and use it as they see fit.

These plugins must be written as a Reactjs component in order for the system to be able to load and interface with them properly. However, being written in Javascript, they benefit from a multitude of external packages that can be used and the developers are not limited to simple, single-file add-ons, and are able to impart a great deal of logic and complexity in their additions.

Also, the developed system, in this current iteration allow for plugins to interact with external systems and APIs unknow to the application and use them to their benefit.

4.5 Data protection

The collection of sensitive information that may identify a test subject or reveal otherwise private information, forces the application to adhere to the counsel of a relevant ethics council regarding how the information is processed.

This system was designed with General Data Protection Regulation (GDPR) compliance in mind, therefore the way the various components of the system interact with each other and the features they offer are geared towards the protection of sensitive information.

The system also offers the possibility of deleting partial or all collected information from a participant at the researcher’s discretion or at the request of the participant, and each participants must agree to the collection and storage of relevant information before partaking in a study.

Architecturally, some steps were taken to ensure the collected data remains safe and protected from unwarranted access by implementing the following measures:

- The data pertaining to the collected signals are kept separated from the information system both logically as well as physically, depending on how the application is deployed,

- The Docker container where this information resides has no exposed ports to the exterior, meaning that only other containers in the same virtual network can connect to it, namely the backend. Moreover, the access to this information via the backend is done through an API that requires a valid login and the correct level of clearance.

The application does not require a public cloud in order to be deployed, all systems can run “on premises” with no need of external systems. The dropbox feature of the application is not considered essential for its operation and is considered a facultative step where no identifiable information is stored.

5 Implementation

In this chapter the methods and reasoning for how each component of the application has been implemented will be thoroughly detailed along with a discussion of alternative ways it could have been done. The screenshots taken to illustrate the various components of the system depict fictional studies and all the captured signals shown were obtained from volunteers for the purpose of system validation and were afterwards deleted.

5.1 Typical deployment scenario

In order to make the application able to be deployed in a cloud environment isolated from other applications and operating system agnostic, a container-based approach is the most suitable for the task. Containers allow for the creation of consistent environments, isolated from other applications and it can bundle an application with all its software libraries and dependencies. These containers can be run in most systems while assuring the application environment remains unchanged [15].

Containers share their operating system, running as isolated processes regardless of the host operating system. In contrast with the virtual machines approach, where the entire hardware stack is virtualized, this virtualization at the OS-level provides a sandboxed view of the OS logically isolated from other applications [16] (figure 14).

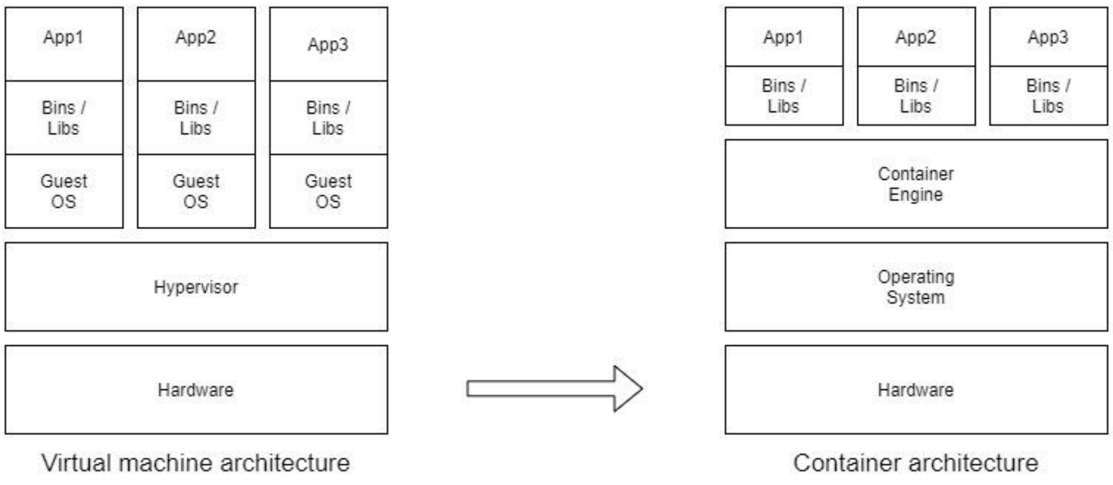


Figure 14 – Comparison between virtual machine and container deployment architecture

A well-known and widely adopted container deployment solution is Docker[13], an open-source platform built to automate the deployment of applications in containers. Within this project scope, the main advantage of deploying the application in Docker containers is the fact that it provides the ability to deploy applications anywhere, including a self-hosted solution or the cloud.

Since containers are independent of the underlying operating system and hardware, they can be easily moved between hosting solutions with minor changes. Furthermore, cloud computing providers such as Amazon Web Services[14] or Microsoft Azure[15] offer Docker support for cloud deployment of containers, eliminating completely the need of managing the hosting hardware along with all the infrastructure around it and delegating that responsibility to a third party.

The way each container is created is via a Dockerfile. This file is a text file that specifies all the required commands to build a given image, either from an existing one or from scratch and can specify additional resources to be loaded depending on the needs of the application. This way, containers are be kept immutable while the Dockerfile performs the needed operations at build time. On the other hand, by establishing a consistent development environment it reduced the amount of time spent in system configuration since it only needs to be done once for all platforms.

[13] https://docker.com/
[14] https://aws.amazon.com/
[15] https://azure.microsoft.com/

In order to build and deploy multiple containers at the same time, Docker compose can orchestrate this process with a single file. This file contains directives to build each of the Dockerfiles, mount volumes, expose ports and create networks.

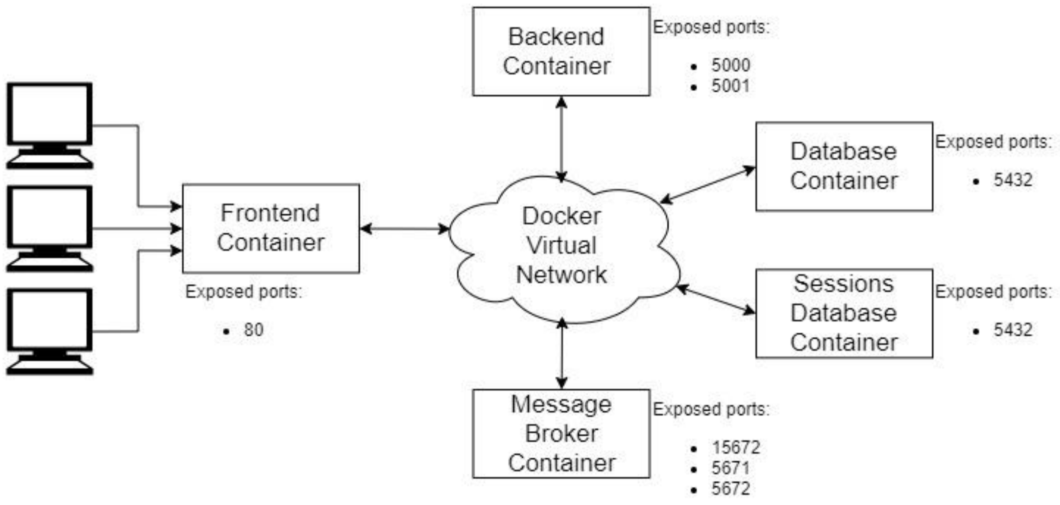


Figure 15 – Application container deployment scheme

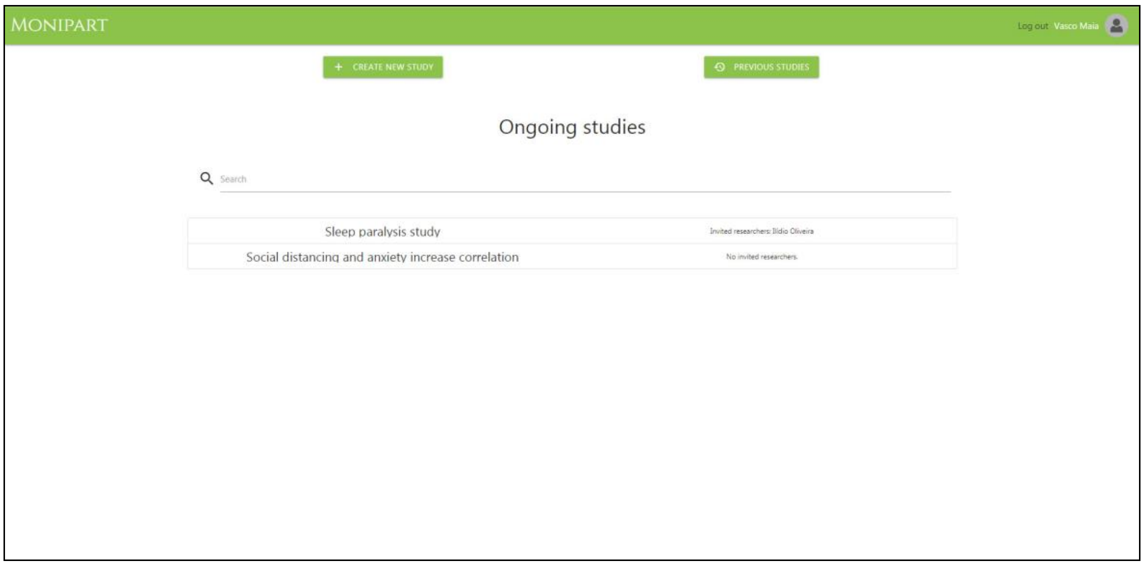
The figure above (figure 15) illustrates the topology of the deployed application and its various containers interconnected via a docker network.

However, in an effort to make the process of deployment even more straightforward, before building the images with Docker compose, a script was created that pulls all the latest versions of the different components of the application from their respective github repositories, ensuring that every time the application is deployed, and no matter where the application is deployed, it runs the latest version with all the latest features

5.2 Web portal

5.2.1 Researcher area

This is the area where the researcher is redirected after logging in (figure 16). From here they can either create new studies or open existing studies, both studies that are ongoing or studies that have ended.



Ongoing studies

Sleep paralysis study	Invited researchers: Ildio Oliveira
Social distancing and anxiety increase correlation	No invited researchers.

Figure 16 – Landing page of the users of type "researcher"

All these listing will display the studies the researcher has created but also studies created by other researchers that have included them as an invited researcher during the creation of the study.

Study creation

This part of the system allows for the creation of studies. This process involves a fair amount of configuration and in order to organize the information there are four topical sections:

- **Study details:** In this section, all the generic information regarding the study can be entered such as a study name, a description and a thumbnail;

- **Invite researchers:** This field is optional, however if there are other researchers registered in the platform with which the researcher creating this study wishes to share the contents of the study, they can add other researchers to this list. In order to find researchers one can use either their partial name or partial email;

- **Participant selection:** In order to create a study, there needs to be at least one participant associated. Subsequent sessions created within this study can only choose from these selected participants. In this step the researcher can either select several participants as individuals, meaning that each session can only be done with one participant at a time, or create groups of participants (figure 17) where participants are grouped together and can be part of sessions simultaneously;

- **Agreement consent:** The researcher may upload a pdf file containing the agreement the participants must accept when joining a session.

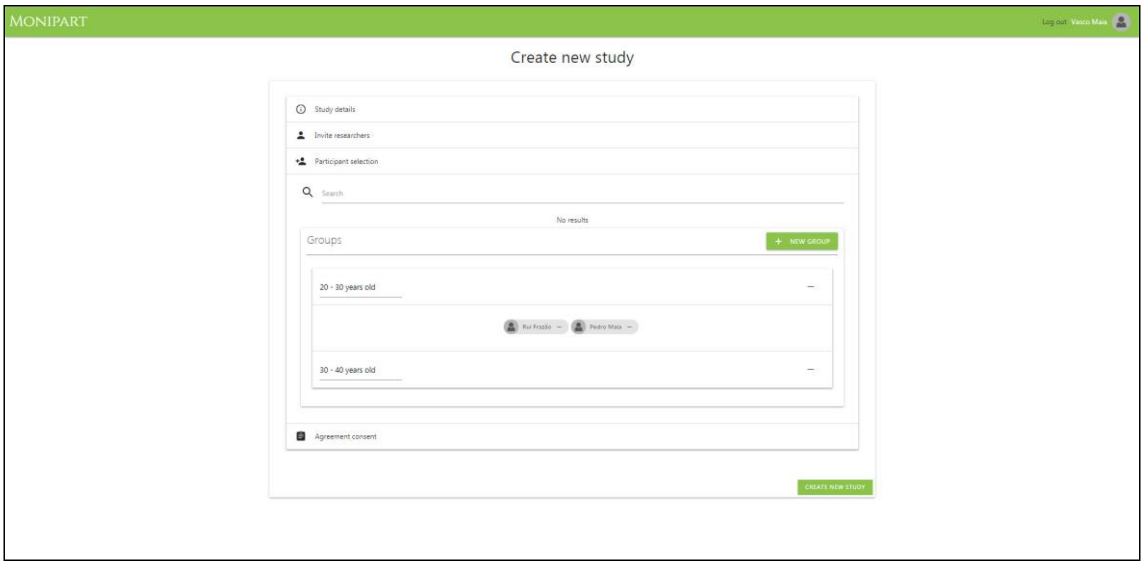


Figure 17 – Example of the creation of a study where there are two groups of participants

Study page

The study page aggregates all the information pertaining to a given study and allows the creation and access of sessions.

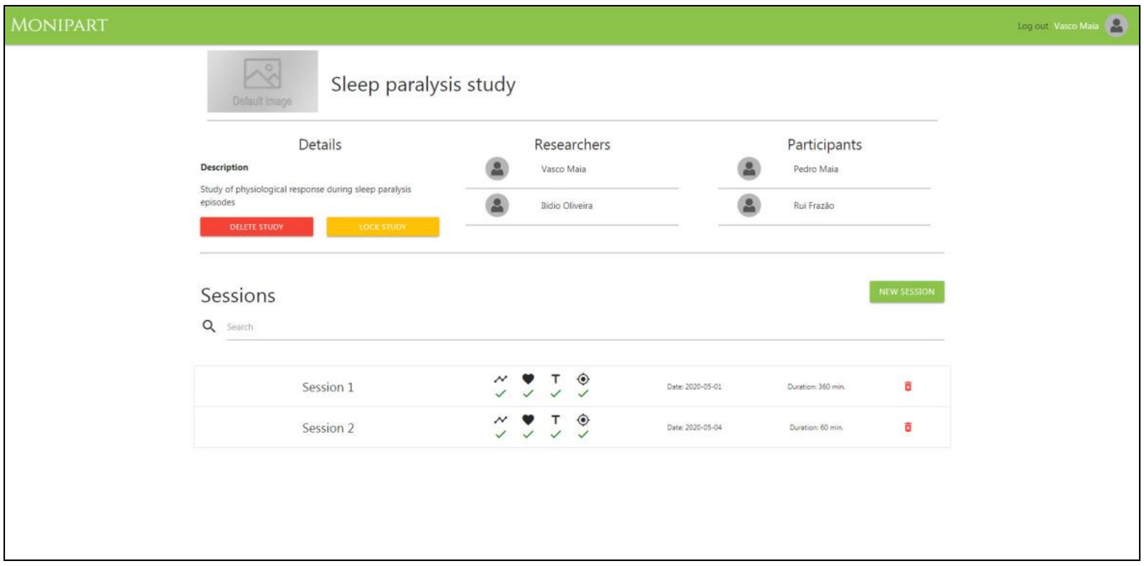


Figure 18 – Study page from the researcher's point of view

In this page (figure 18) the researcher is presented with information such as the invited researchers that also have access to the study and the pool of participants that make part of the study.

From here the study can be locked, meaning that no further sessions can be created under this study and it moves to the "previous studies" section in the landing page, allowing the study to still be accessible and all the data therein, or the study can also be deleted. The later option completely removes all traces of the study from the system including all the sessions associated with the study.

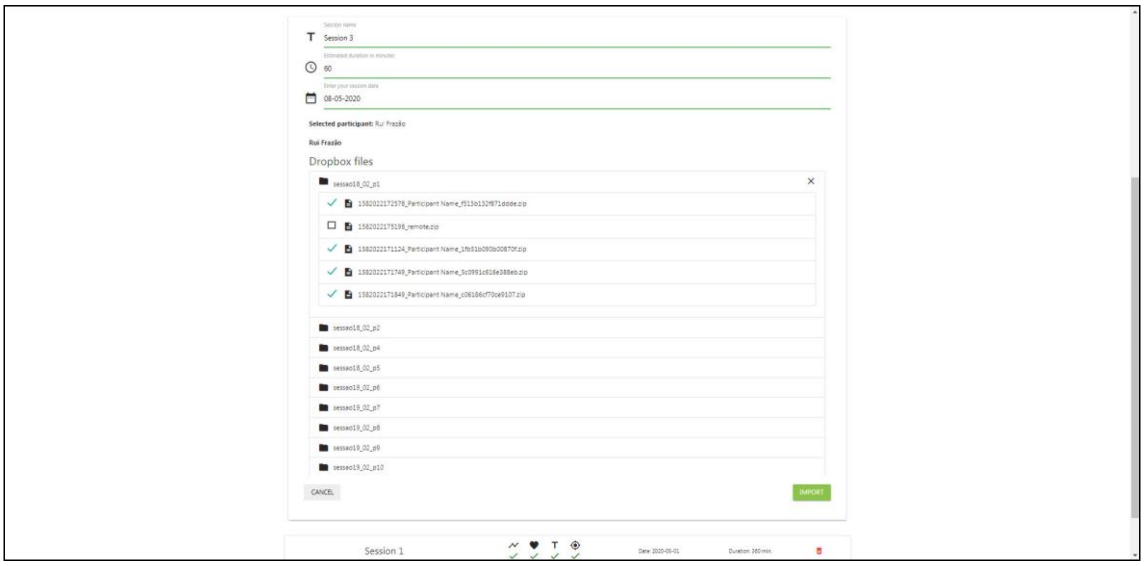
Session creation

The sessions creation process can be made one of three ways:

- **Import local files:** The session information can be imported from a single or a group of .zip files that pertain to the collected data from the mobile application, stored locally in the user's device;
- **Import from dropbox:** The session information can be imported from a single or a group of .zip files that pertain to the collected data from the mobile application, stored in a dropbox folder in the cloud (figure 19). This folder cannot be changed by the user via the interface but can be easily configured via a .json file;

- **Create new session:** As opposed to the previous session creation methods, this method creates a session that can receive information in real-time from the sensors.

The creation of a session via local files or a dropbox folder are supported as a way to import tests that have been done previously to the development of this tool and still benefit from the functionality it offers.


Session name
Session 3
Estimated duration in minutes
60
Enter your session date
08-05-2020

Selected participant: Rui Frazão

Rui Frazão

Dropbox files

- sessao18_02_p1
  - ✓ 1582022172578_Participant Name_f513b132f871ddde.zip
  - ☐ 1582022175198_remote.zip
  - ✓ 1582022171124_Participant Name_1fb51b090b00870f.zip
  - ✓ 1582022171749_Participant Name_5c0991c616e388eb.zip
  - ✓ 1582022171849_Participant Name_c06186cf70ce9107.zip
- sessao18_02_p2
- sessao18_02_p4
- sessao18_02_p5
- sessao19_02_p6
- sessao19_02_p7
- sessao19_02_p8
- sessao19_02_p9
- sessao19_02_p10

Date: 2020-05-01
Duration: 360 min.

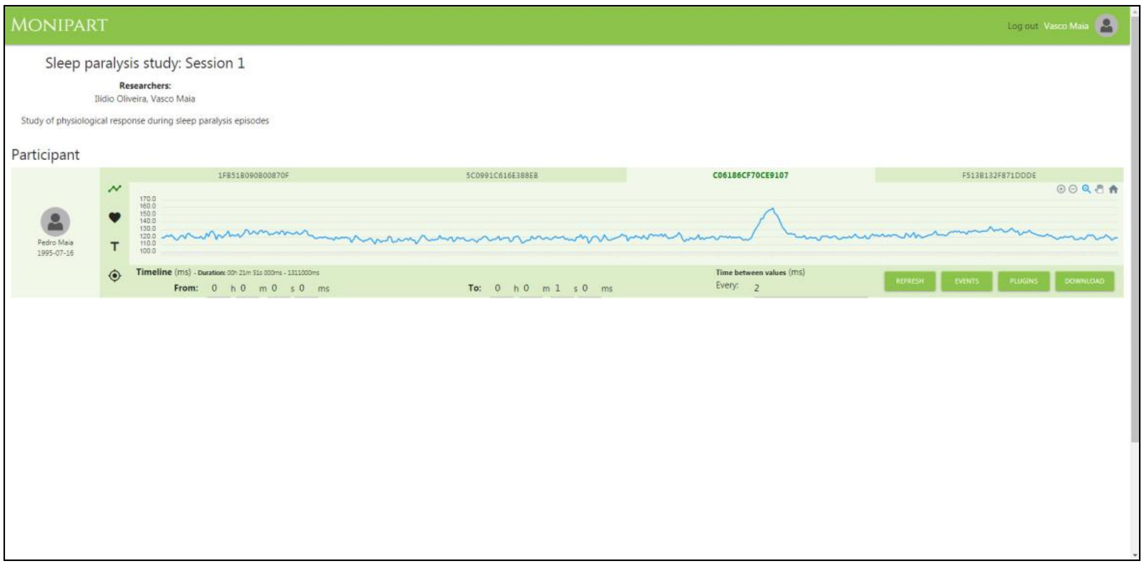
Figure 19 – Example of an import of a study from the Dropbox folder

Irrespective of which method is used, certain information must be provided such as a name, date, estimated duration and the participants that are being evaluated during the session. The participants must be chosen from the pool of participants defined when creating the study or, in the case of participant groups, one group must be chosen.

A session can only be conducted with either one participant or one group of participants. No more than one participant can be part of a session unless its inserted in a group and no multiple groups can be part of a session simultaneously.

Session page

The session page is where presumably the researcher will spend most of its time and is also the part of the application most fraught with features related to the signal analysis and processing.



Sleep paralysis study: Session 1

Researchers:
Ilidio Oliveira, Vasco Maia

Study of physiological response during sleep paralysis episodes

Timeline (ms) - **Duration:** 00h 21m 51s 000ms - 1311000ms

From: 0 h 0 m 0 s 0 ms **To:** 0 h 0 m 1 s 0 ms

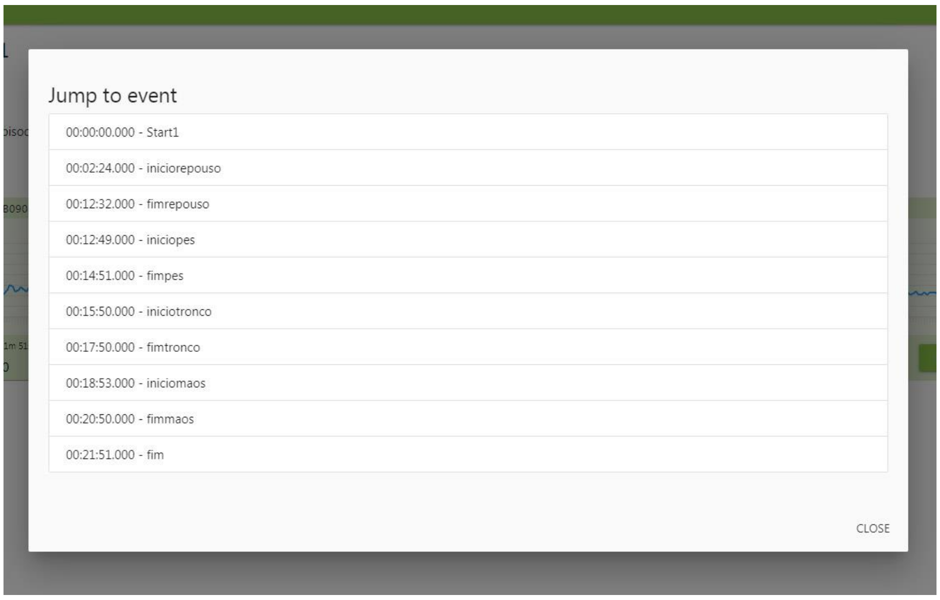
Time between values (ms)
Every: 2

Figure 20 – Example of the study page from the researcher's point of view with one participant, displaying the ECG signal of the third device between the interval 0h:0m:0s:0ms and 0h0m1s0ms

The layout of the page (figure 20) is straightforward, with the study / session details at the top and a timeline for each participant below that. In this timeline the researcher can switch between the various types of signals such as ECG, heartrate, temperature or accelerometer data for each capture device (sensor box).

The signal can be visualized in intervals of time specified by the researcher and manipulated with tools such as zoom and drag that offer a better way of analysing the information and the relation to its surroundings.

The timeline also provides some features that ease navigation such as the ability to jump to one of the many events registered during the session (figure 21) such as the start or the end of periods of rest and so on.

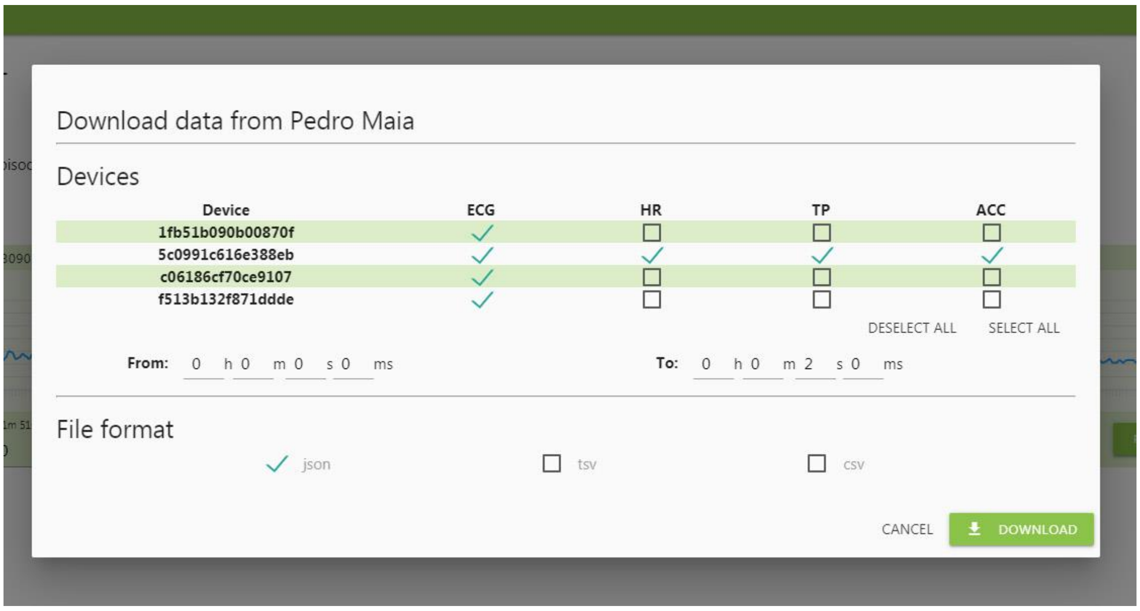

Jump to event

00:00:00.000 - Start1
00:02:24.000 - iniciorepouso
00:12:32.000 - fimrepouso
00:12:49.000 - iniciopes
00:14:51.000 - fimpes
00:15:50.000 - iniciotronco
00:17:50.000 - fimtronco
00:18:53.000 - iniciomaos
00:20:50.000 - fimmaos
00:21:51.000 - fim

CLOSE

Figure 21 – Example of the event jumping functionality

Furthermore, all the data in the system, not just the one currently being displayed, can be downloaded in a variety of file formats for external use (figure 22). This can be achieved through the download button where information such as which devices and which signals to download and the desired interval and the file format. Currently the system allows the .json, .tsv and .csv file formats.


Download data from Pedro Maia

Devices

Device	ECG	HR	TP	ACC
1fb51b090b00870f	✓	☐	☐	☐
5c0991c616e388eb	✓	✓	✓	✓
c06186cf70ce9107	✓	☐	☐	☐
f513b132f871ddde	✓	☐	☐	☐

DESELECT ALL SELECT ALL

From: 0 h 0 m 0 s 0 ms To: 0 h 0 m 2 s 0 ms

File format

✓ json ☐ tsv ☐ csv

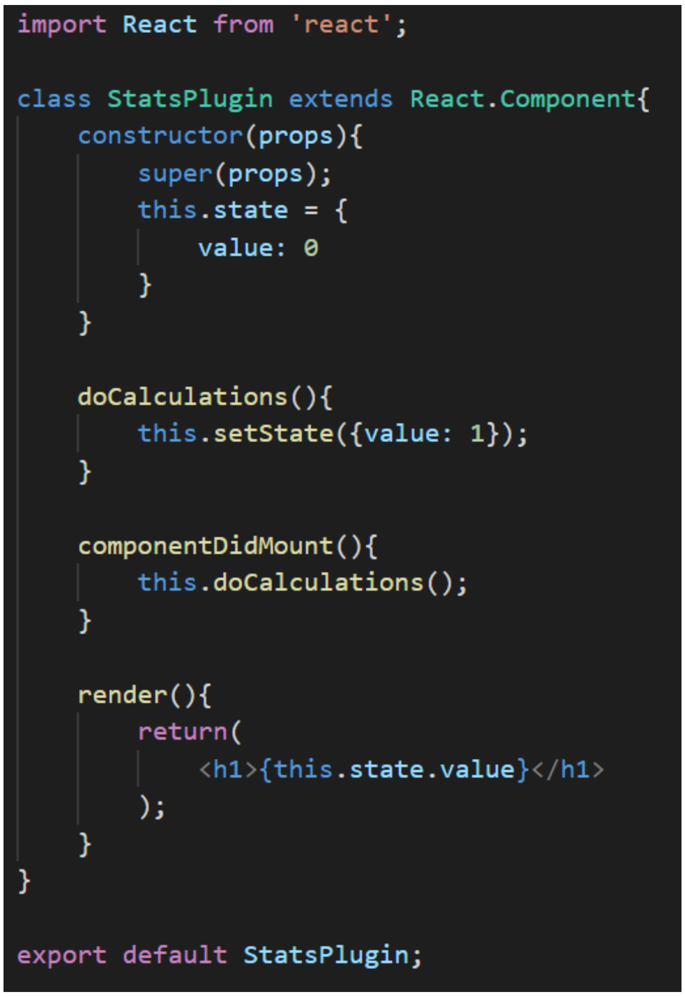
CANCEL DOWNLOAD

Figure 22 – Example of downloading a .json file from the timeline interface

During a live capture of a session, where the information is being received directly from the message broker via a web socket, none of the above features are available as the data is constantly changing and it would not make sense to navigate to non-existent events or focus on a particular time interval. These features can only be accessed after the session is over.

Plugin support

The system has a plugin system built in that allows external developed code to interface with the system in a simple and straightforward way. The plugin itself is a React.js Class Component that must follow the typical structure (figure 23) without the need for specific functions or predefined interfaces.

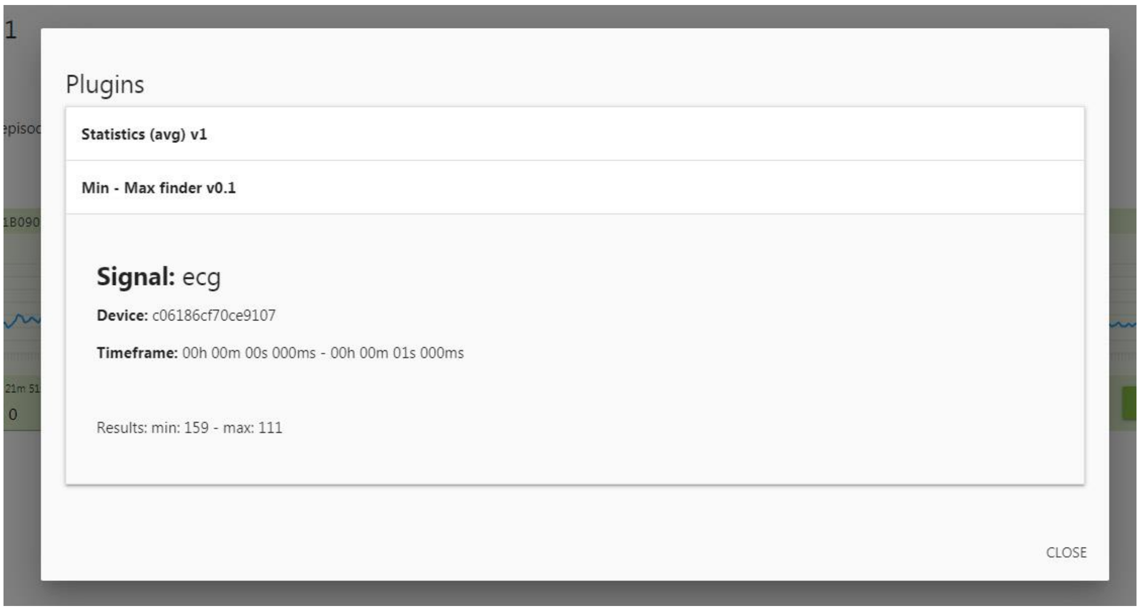


```
import React from 'react';

class StatsPlugin extends React.Component{
    constructor(props){
        super(props);
        this.state = {
            value: 0
        }
    }

    doCalculations(){
        this.setState({value: 1});
    }

    componentDidMount(){
        this.doCalculations();
    }

    render(){
        return(
            <h1>{this.state.value}</h1>
        );
    }
}

export default StatsPlugin;
```

Figure 23 – Example of the structure of a plugin component

These components upon creation have access to the timeline state where all the signal information is stored and can be used by the plugin for further calculations and analysis. Because of the nature of javascript, these components can be as simple or as complex as the developer sees fit, being possible to import external libraries and extra logic otherwise not part of the original system.

The component is rendered within a viewport in the plugin menu, and components are completely independent of each other (figure 24).


Plugins

Statistics (avg) v1

Min - Max finder v0.1

Signal: ecg

Device: c06186cf70ce9107

Timeframe: 00h 00m 00s 000ms - 00h 00m 01s 000ms

Results: min: 159 - max: 111

Figure 24 – Example of a plugin that finds statistical information of a given signal interval

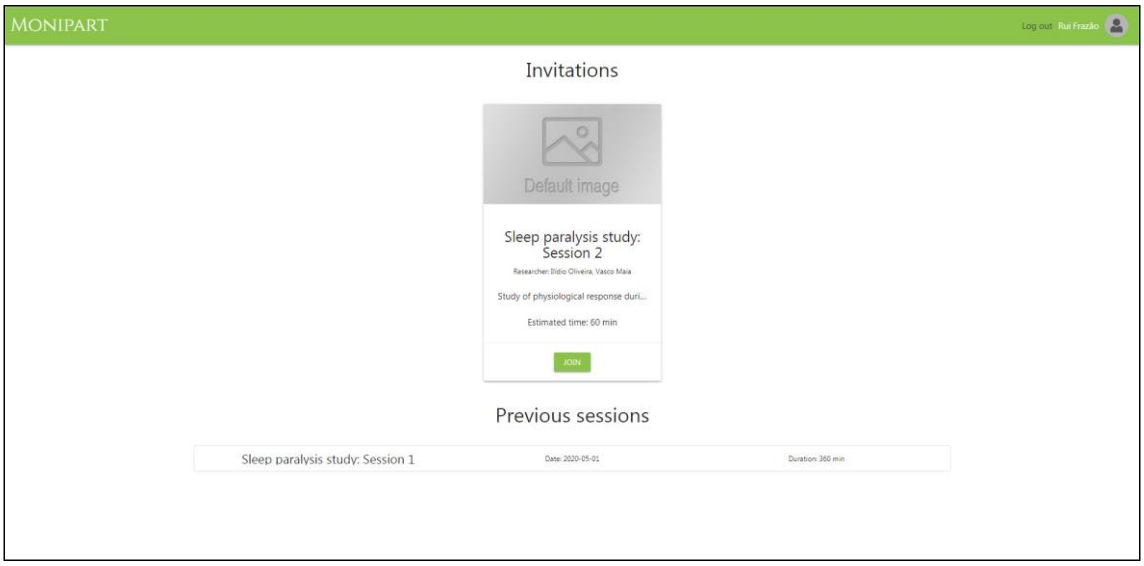
In order for the system to load plugins, there is a .json configuration file with a listing of installed plugins with each entry containing a plugin name and the relative path to the file to load.

When installing a plugin, the plugin configuration file must be changed to reflect the introduction of the new code by adding a new entry specifying the plugin name and the entrypoint or the component file to import and create the plugin component out of.

Currently the application has two developed plugins that work as a proof of concept for this functionality. The two plugins calculate the average of a signal segment and find the maximum and minimum values in a segment.

5.2.2 Participant area

The participant is the type of user that less interacts with the system and as such, the number of functionalities available for the user to interact with are rather simple in nature.



Sleep paralysis study: Session 2

Researcher: Ilídio Oliveira, Vasco Maia

Study of physiological response duri...

Estimated time: 60 min

Sleep paralysis study: Session 1 Date: 2020-05-01 Duration: 360 min

Figure 25 – Landing page of the users of type "participant"

This is the area where the participant is redirected after logging in (figure 25). From here they can join an ongoing session which are displayed as invites. These invitations will only be available during the duration of the session, meaning that after the session is over, the participant will no longer be able to join.

If a participant wishes to take part in a session, they must first agree to an agreement the researcher responsible for creating the sessions uploaded during its creation. After doing so, and only after, there'll be the necessary codes to pair all the devices for capturing the sensor information.

This step can be skipped entirely and done exclusively on the already developed Android application. However, the addition of having to accept in order to join a session is an advantage over the existing system. Moreover, this functionality can be expanded further to support the answering of questionnaires both before and after a session is over, eliminating the need of yet another external way of managing the information given by the participants and the data collected during the experiments.

Below the invitations panel, there is a listing of previous sessions in which the participant took part. By entering the page of any of these sessions, only general information is given to the participant. No information regarding the collected or their performance during the sessions is provided.

5.2.3 Cross-cutting functions

Login and registration

The login system developed for this application completely made from the ground up and is at its core a system of unique tokens that grant access to the rest of the system.

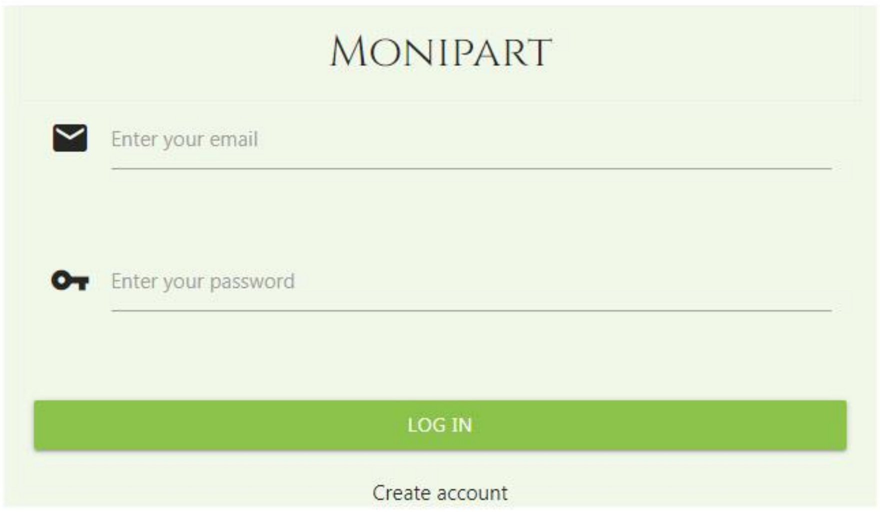


Figure 26 – Login prompt

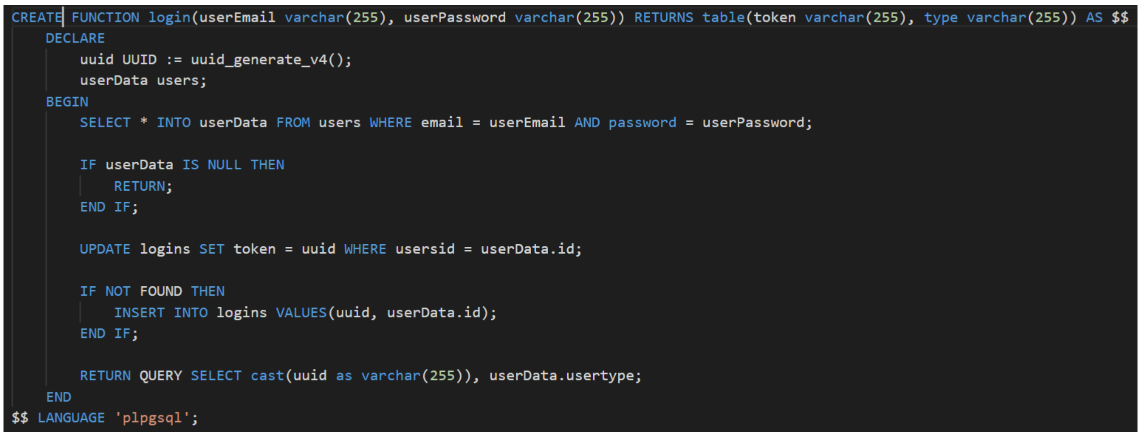
When a user inserts their credentials in the login screen (figure 26), those are sent to the backend, which will assess their validity and, in case they are valid, it will return a unique token that will be stored in the browser's local storage and sent with every API call to the backend.

This unique token or UUID is a 128-bit number that for practical reasons is unique but, while the probability that a UUID will be duplicated is not zero, it is close enough to zero to be negligible.

The UUID is represented as a string of 32 hexadecimal digits, displayed in five groups separated by hyphens, in the form 8-4-4-4-12 for a total of 36 characters. For example:

123e4567-e89b-12d3-a456-426614174000
xxxxxxxx-xxxx-Mxxx-Nxxx-xxxxxxxxxxxx

The process of determining if a user is logged in and the attribution of a UUID is done entirely in the database via a function (figure 27) the backend calls, thus removing the backend from any responsibility regarding the token creation and, since it is treated as a transaction, if there is some error during this process, the database rolls back to the previous state before the error, reducing the possibility of an unauthorized login.



```
CREATE FUNCTION login(userEmail varchar(255), userPassword varchar(255)) RETURNS table(token varchar(255), type varchar(255)) AS $$
    DECLARE
        uuid UUID := uuid_generate_v4();
        userData users;
    BEGIN
        SELECT * INTO userData FROM users WHERE email = userEmail AND password = userPassword;

        IF userData IS NULL THEN
            RETURN;
        END IF;

        UPDATE logins SET token = uuid WHERE usersid = userData.id;

        IF NOT FOUND THEN
            INSERT INTO logins VALUES(uuid, userData.id);
        END IF;

        RETURN QUERY SELECT cast(uuid as varchar(255)), userData.usertype;
    END
$$ LANGUAGE 'plpgsql';
```

Figure 27 – Login token handling function

Once the user has been successfully logged in, they are redirected to their respective home page according to their type of user, either “participant” or “researcher”. From then on, all API calls in order to retrieve information is accompanied by their given UUID.

The backend has a middleware (figure 28) that verifies if a user is currently logged in and if not, return an error message and refuses to serve the desired information.

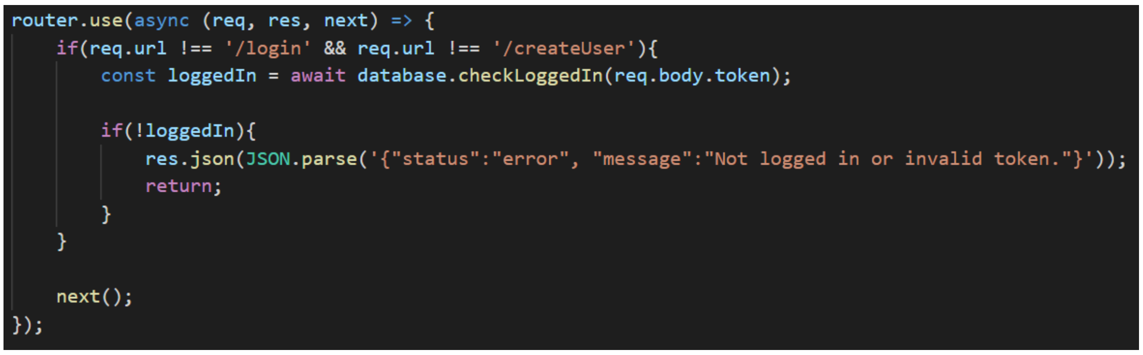

```
router.use(async (req, res, next) => {
    if(req.url !== '/login' && req.url !== '/createUser'){
        const loggedIn = await database.checkLoggedIn(req.body.token);

        if(!loggedIn){
            res.json(JSON.parse('{"status":"error", "message":"Not logged in or invalid token."}'));
            return;
        }
    }

    next();
});
```

Figure 28 – Login UUID verification middleware

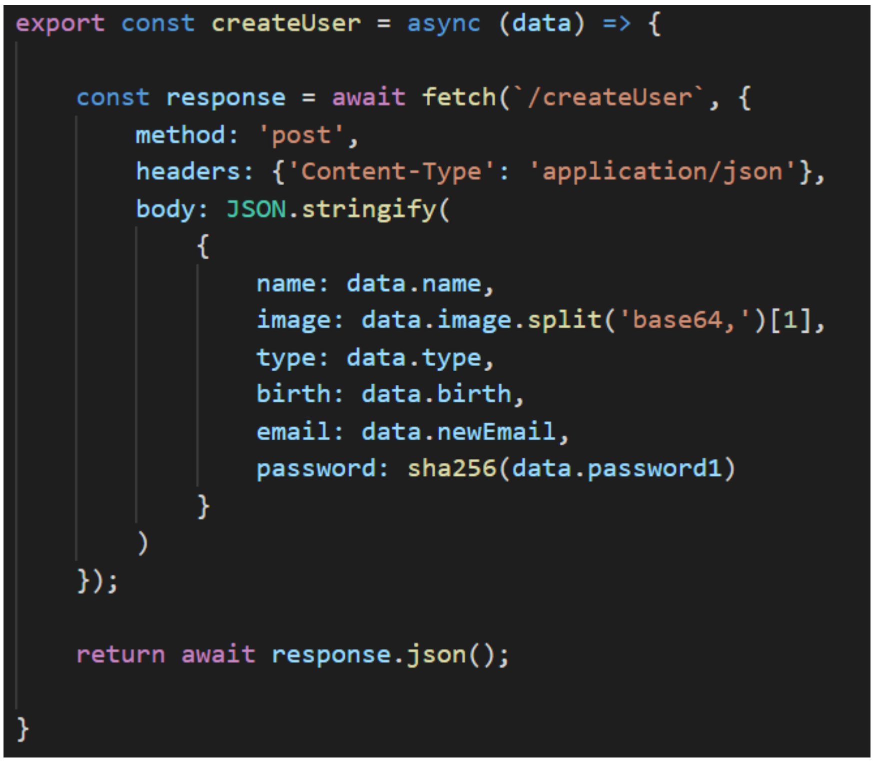
The only exceptions where a valid UUID is not required is in the process of logging in itself and in the process of creating a new user account.

Regarding the registration of new user accounts, a few details are asked to be provided, such as a user image, which, if left unchanged, will be a default image, user email, password, name, date of birth and user type. There are checks in place in order to ensure the user provides all the required information and that the credentials provided are not already in user.

Personal information

The user’s personal information is stored in a database, arising the need to deal with the information in such a way that information such as passwords are never known to anyone except the user themselves.

In order to preserve the privacy of sensitive information, only SHA256 hashes (figure 29) are sent to the backend and in turn stored in the database of information like passwords.



```
export const createUser = async (data) => {

    const response = await fetch(`/createUser`, {
        method: 'post',
        headers: {'Content-Type': 'application/json'},
        body: JSON.stringify(
            {
                name: data.name,
                image: data.image.split('base64,')[1],
                type: data.type,
                birth: data.birth,
                email: data.newEmail,
                password: sha256(data.password1)
            }
        )
    });

    return await response.json();

}
```

Figure 29 – Create user function where sensitive information is hashed with SHA256

SHA256 is implemented in some widely used security applications and protocols, including TLS and SSL, PGP, SSH, S/MIME, and IPsec, even though with the passing of time it has become less secure [18] through the use of pseudo-collision attacks and expanded dictionaries that can be used for brute force attacks. Still, the onus for a secure password should lie mostly on the user by, among other things, using a strong password.

Still, at this point in time it is still the standard used in internet communications and still remains mostly secure, such is the reason why this algorithm is employed in this project.

Furthermore, the images uploaded by the user are sent and received from the backend via base64 encoding and are stored in the backend file system, storing only in the database the path to the file. This measure allows the database to not be unnecessarily big and reduce the percentage of allocated space to the user information, increasing the amount of study and session information to be stored.

Ultimately a storage solution could be devised separate from the backend altogether, however given the specificity of this application and the low file storage demands, it doesn't justify such an architecture.

5.3 Services API

The backend API is divided in two logical parts, the first has to do with the interaction and manipulation of the information system, the other has to do with the insertion and retrieval of signal information from the system.

All the calls supported by the system (table 5) need to be accompanied by at least a valid login token and whatever necessary information for each specific call.

HTTP METHOD	***PATH***	***Action***
POST	/login	Logs a user in and returns a security token
POST	/logout	Revokes a user security token
POST	/searchResearchers/:value	Return a list of researchers matching the search term
POST	/searchParticipants/:value	Return a list of participants matching the search term
POST	/createUser	Creates a new user in the system
PUT	/updateUser	Updates user information
POST	/image/:id	Returns an image if it exists or a default image
POST	/agreement/:id	Returns an agreement .pdf file if exist or a default
POST	/study/:id	Return study information (name, description…)
POST	/session/:id	Return session information (name, time…)
POST	/createStudy	Creates a new study in the system
DELETE	/deleteStudy/:id	Deletes a study from the system
POST	/lockStudy/:id	Locks the study (can only be deleted)
POST	/createSession	Creates a session in the system
POST	/importLocalSession	Imports a study from one or more .zip files
POST	/getSessionData/:id	Return session information (duration, devices…)
POST	/getEcgData	Returns signal data in a time interval for a device
POST	/getHeartrateData	Returns signal data in a time interval for a device
POST	/getTemperatureData	Returns signal data in a time interval for a device
POST	/getAccelerometerData	Returns signal data in a time interval for a device
POST	/downloadSession	Returns session data in a given file format
POST	/endSession/:id	Ends session

Table 5 – API call methods supported by the system

All messages are exchanged in json format with file being sent in base64 encoding. This applies to images, pdf files and zip files for the upload of new sessions.

If a call is not successful, a message will always be returned detailing the problem.

5.4 Physiological data repository

The physiological data repository, given the high capture rate of the sensors and the average duration of the experiments, needs to handle a large amount of data in a time-oriented fashion. As an example, in a typical setup consisting of five sensors, each capturing at 500 Hz and an average experiment duration of 25 minutes, there can be upwards of 4 million rows of data per signal type, totalling 24 million rows per experiment on average.

The use of TimescaleDB over a traditional sql database provides several advantages in this regard namely: higher ingest rates, better query performance and time-oriented features.

By far the most useful feature used in this project is time bucketing, allowing to get a segment of signal quickly by averaging values in a set interval period. This feature is widely used when retrieving ECG, heartrate, temperature or accelerometer information to display in the timeline.

```
let query_getData = `
        SELECT string_agg(ecg::text, ', ') AS vals
        FROM (
            SELECT time_bucket('__bucket__ millisecond', time) AS times, avg(ecg) AS ecg
            FROM ecgs
            WHERE time > to_timestamp($1) AND time <= to_timestamp($2)
            AND sessionsid = $3
            AND sessionsuserid = $4
            AND sessionsdeviceid = '__device__'
            GROUP BY times
            ORDER BY times ASC
        ) AS sel1
    `;
```

The query above, is used to retrieve ECG data from the time-series database in a quick fashion and return it ready to be displayed in the timeline. Other queries are similar to this with some small variations to accommodate each specific case.

The bucket parameter, or the time between each point in time, is defined by the user and can greatly affect the performance of the query, however, it is a trade of between speed and information resolution.

5.5 System prototypes

The process of creating a system geared towards a final user is more often than not an iterative process that changes how the final product looks and behaves based on an increasingly better understanding of a system and what its main requirements are.

Pertaining to the web app or the frontend, it underwent some significant changes from the first low fidelity prototypes to the final product. These changes came about mostly due to the necessity of having to separate various elements into different pages altogether in order to abide by the principle of the separation of concerns.

This case is most evident in the researcher's landing page (figure 30), where the researcher would have been able to create new studies and access to current and previous studies would have been bundled together.

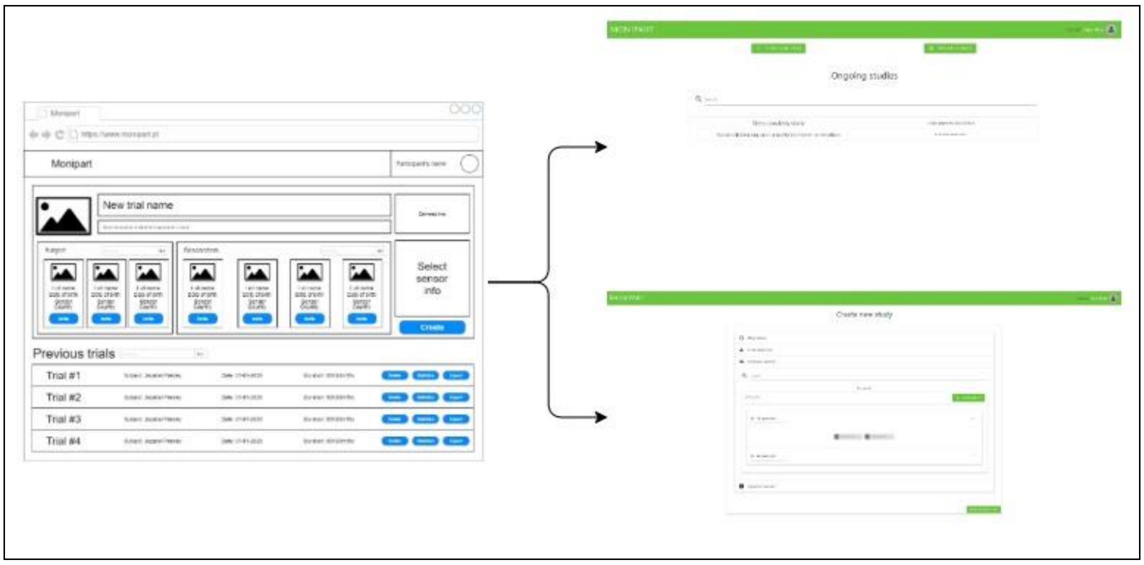


Figure 30 – Evolution of the researcher's landing page, illustrating the separation of concerns by separating the content in multiple pages

This separation of functionalities across multiple pages not only makes evident the logical separation between the different functionalities of the application but also allows for the streamlining of the interface.

Regarding the backend, it also has suffered multiple changes from its prototyping stages to the final build, namely with introduction of web sockets to establish a connection to the frontend and deliver information from the message broker. This came about because of the need to communicate with the database as the information was coming in from the broker and some incompatibilities found between the RabbitMQ web client library and Webpack used by React.js. Thusly having the message broker communicate with the backend instead of the frontend directly, solved both these problems and reduced overhead compared to the original devised architecture.

All in all, the current system is the result of continued iterations that were brought about either by having to work around incompatibility issues or by finding a better way of making the system friendly and intuitive and ultimately useful to the final user.

5.6 Functional testing

Taking into consideration the present epidemiological situation regarding the COVID-19 pandemic and the restrictions put in place since the start of 2020, the second half of this project has been impacted and some aspect had to be changed to conform to the circumstances. These restrictions impacted the development of this project regarding mostly the testing of the application with other people.

Under normal circumstances, the "functional testing" portion of the project would involve several people from the scientific area this application is supposed to be used testing the application and giving feedback as to whether or not the application meets the requirements and expectations of the users.

Since it was not possible to conduct tests with participants to assess the applications conformity with the specified requirements, this responsibility has fallen solely on the programmer of the system.

The strategy adopted at first to deal with this task was, for the most part, manually testing the features as they were being implemented. This method proved adequate to this particular project given the fact that this process was usually not very time consuming and gave the opportunity to promptly fix issues as they were detected.

However, as the application grew larger, the need to test every possible feature of the application soon became unfeasible to do manually and so automated unit tests were developed to guarantee the application behaves as expected.

The developed tests cover a variety of workflows expected of the final user including edge cases as well as forbidden actions, exhausting all possible avenues for the system to fail. In any case, the workflow adopted for this process is detailed below:

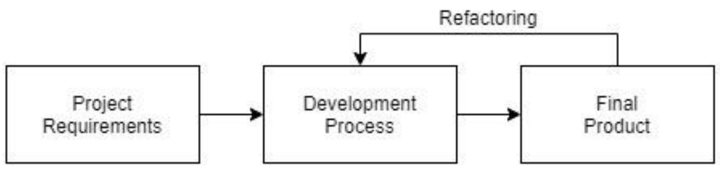


Figure 31 – Testing pipeline to ensure the final product conforms to the project requirements

5.7 Usability assessment

The usability tests were conducted in order to evaluate how the target users interact with the system and how effective it is. With these tests the following information can be noted:

- How many tasks the users were able to complete;
- How long it took to complete each task;
- Retrieve some feedback during the test;
- The level of satisfaction with the product.

These metrics allow the identification of areas of the software where some problems might exist, functionalities that should be refactored, added or discarded.

Test procedure

Due to the current epidemiological situation, only a small number of participants were able to be gathered for the testing of the system and all the tests where done via Skype. Nevertheless, the procedure remained mostly unchanged.

The test is divided into three parts:

1. The users where asked a few personal questions at the beginning in order to identify what type of user they were;
2. The users were asked to perform a certain number of tasks within the system while being monitored the time they spent and how well they performed the task;
3. The users where asked a final questionnaire in order to assess their satisfaction with the system and given the chance to provide some feedback and suggestions.

In the first part the users where asked a few questions that are surmised in the following table:

Participant	***Age***	***Gender***	***Have used other similar software***	***Technology literacy (1-10)***
P1	20	F	Yes	9
P2	20	M	No	9
P3	22	F	No	7

Table 6 – Characterization of the users involved in the usability testing

After the first part, the participants were briefed of the purpose and main functionalities of the software they were about to use. After that the second part of the test took place.

In this part of the test, participants were asked to complete fifteen different tasks (table 7) that are an eclectic representation of the whole system.

Task ID	*Description*	*Estimated time*
T1	Create a user of type “researcher”	1 min
T2	Create a study with the invited researcher “Vasco Maia” and an individual participant named “Rui Madeira”	1 min
T3	Create a study with a group named “grupo 1” with two participants “Rui Madeira” and “Maria Tavares”	1 min
T4	In the first created study, create a session from a Dropbox file	30 sec
T5	In the first study, create a session that receives signal from all types of sensors	30 sec
T6	In the first created session, visualize the ECG, heartrate, temperature and accelerometer information for all the devices	1 min
T7	Choose a visualization interval between “4m10s” and “4m12s”	10 sec
T8	Jump to the event “fimrepouso”	10 sec
T9	Zoom in a segment of the signal and return to the initial view	30 sec
T10	Download that segment for all the devices in .json format	1 min
T11	Do a search in the past studies list	1 min
T12	See the details of a study	30 sec
T13	Logout and login with the following credentials “rui.madeira@ua.pt” “teste”	1 min
T14	Change the user’s birthday	1 min
T15	Open a session	30 sec

Table 7 – Listing of the tasks asked of the participants during the second part of the usability tests

The performance of the participants did not vary much between them and did fall within the expected time to complete each task. However, where most of the participants experienced some difficulty was during the session creation.

In the third and last part of the test, after all the tasks had been completed, the participants were asked to answer a System Usability Scale (SUS) questionnaire. The SUS questionnaire for the participants consisted of ten questions that the users had to give a rating ranging from 1 to 5 (from totally disagree to totally agree).

This questionnaire was manually translated to Portuguese however, it was created by John Brooke and it is used to evaluate websites, software and applications [19].

The answers given to the questionnaire are presented below in the following table:

Question	***P1***	***P2***	***P3***
Q1	5	4	4
Q2	1	1	1
Q3	5	5	4
Q4	1	1	1
Q5	5	5	4
Q6	1	1	2
Q7	5	4	5
Q8	1	1	1
Q9	5	4	5
Q10	1	1	1

Table 8 – Answers given by the participants to the SUS questionnaire

The above values can be used to calculate the System Usability Score. This value can range from 0 to 100 and is representative of how usable the system is.

This score can then be mapped to a scale that shows how the system fared.

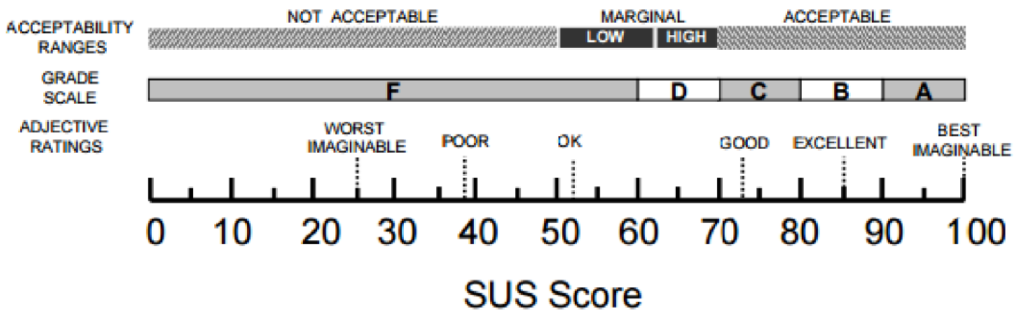


Figure 32 – SUS Score rating scale

The system got a score of 94.17, giving it a rating of "A". This high value may be heavily influenced by the extremely small population that was part of this study.

This is corroborated by reviewing the feedback given by the participants after experiencing the system, and the conclusion that can be taken from the answers is that most users found the application easy to use, not overly complex but needed some degree of learning beforehand in order to use it efficiently.

The most problematic aspect of the application is the creation of new studies where the participants spent a disproportionate amount of time in relation to the other tasks. Also the feedback received also pointed some flaws of the system, aspects such as returning to a previous page should be made clearer at certain times.

6 Conclusion

6.1 Work performed

Regarding the main goal of this project which was to create an information system to support and expand the usability and functionalities of an already existing mobile-based solution for capturing physiology data using sensors during experiments with participants, a solution has been developed that corresponds to the requirements set forth.

Considering the objectives set forth in the first chapter, this project was able to fully address all the requirements. This project expanded upon the existing system and added the ability to manipulate and perform statistical analysis of the data collected from experiments, both conducted after the deployment of the application as well as retroactively by importing old sessions into the system and making them able to be manipulated.

The achieved result is an application that is cloud-enabled and tailored to handle time-oriented data supported by a friendly and powerful interface geared towards the researchers, allowing them to conduct some level of analysis of the collected information during the experiments without the need for third party software or having to sort through the information manually as it was the case previously.

6.2 Future work

Although the system works as expected and meets the criteria specified at the start of the project, there are certain aspects that can benefit from some more time dedicated to them, namely:

- Improving the login system by using a different login authentication system such as OAuth and integration with the University's credential authentication system;
- Implement more sophisticated algorithms for analysing the collected data;
- Allow the data to be plotted differently, including more variety of graphs / charts and correlation between signals.

7 References

[1] M. Dempster Wiley and H. Donncha, *Research Methods in Psychology For Dummies*. John Wiley & Sons, 2015.

[2] "Institute of Electronics and Informatics Engineering of Aveiro - IEETA." [Online]. Available: http://wiki.ieeta.pt/wiki/index.php/Main_Page

[3] G. M. Harari, N. D. Lane, R. Wang, B. S. Crosier, A. T. Campbell, and S. D. Gosling, "Using Smartphones to Collect Behavioral Data in Psychological Science: Opportunities, Practical Considerations, and Challenges," *Perspect. Psychol. Sci.*, vol. 11, no. 6, pp. 838–854, 2016.

[4] W. Goddard and S. Melville, *Handbook of Psychophysiology*. Cambridge University Press, 2016.

[5] Z. Geng, X. Zhang, Z. Fan, X. Lv, Y. Su, and H. Chen, "Recent progress in optical biosensors based on smartphone platforms," *Sensors (Switzerland)*, vol. 17, no. 11, 2017.

[6] S. Walker, "12 lead ECG placement – a simple guide," 2019. [Online]. Available: https://www.adinstruments.com/blog/correctly-place-electrodes-12-lead-ecg. [Accessed: 05-Nov-2019]

[7] S. Debener, R. Emkes, M. De Vos, and M. Bleichner, "Unobtrusive ambulatory EEG using a smartphone and flexible printed electrodes around the ear," *Sci. Rep.*, vol. 5, 2015.

[8] C. L. Lisetti and F. Nasoz, "Using noninvasive wearable computers to recognize human emotions from physiological signals," *EURASIP J. Appl. Signal Processing*, vol. 2004, no. 11, pp. 1672–1687, 2004.

[9] N. Ramanathan *et al.*, "Ohmage: An open mobile system for activity and experience sampling," *2012 6th Int. Conf. Pervasive Comput. Technol. Healthc.*

Work. PervasiveHealth 2012, pp. 203–204, 2012.

[10] “BioPac website,” 2019. [Online]. Available: https://www.biopac.com/products/. [Accessed: 14-Nov-2019]

[11] “Vital Jacket Website,” 2019. [Online]. Available: https://www.vitaljacket.com/en/. [Accessed: 14-Nov-2019]

[12] T. A. L. de Bastos, “Vitals Recorder : sistema móvel para apoiar a realização de estudos de psicofisiologia,” Universidade de Aveiro, 2018 [Online]. Available: http://hdl.handle.net/10773/25979

[13] M. Matic, S. Ivanovic, M. Antic, and I. Papp, “Health monitoring and auto-scaling rabbitMQ queues within the smart home system,” in *IEEE International Conference on Consumer Electronics - Berlin, ICCE-Berlin*, 2019, vol. 2019-Septe, pp. 380–384.

[14] G. I. Radchenko, A. B. A. Alaasam, and A. N. Tchernykh, “Comparative analysis of virtualization methods in Big Data processing,” *Supercomput. Front. Innov.*, vol. 6, no. 1, pp. 48–79, 2019.

[15] J. Li, T. Isobe, K. Shibutani, Sony China Research Laboratory, and Sony Corporation, “Converting Meet-in-the-Middle Preimage Attack into Pseudo Collision Attack: Application to SHA-2.” 2012 [Online]. Available: http://fse2012.inria.fr/SLIDES/67.pdf

[16] W. Albert and T. Tullis, “Measuring the User Experience: Collecting, Analyzing, and Presenting Usability Metrics,” *Newnes*, 2013.

Annex 1 – SUS questionnaire

System Usability Scale

	Discordo totalmente				Concordo totalmente
1. Acho que gostaria de utilizar este produto com frequência	1	2	3	4	5
2. Considerei o produto mais complexo do que necessário	1	2	3	4	5
3. Achei o produto fácil de utilizar	1	2	3	4	5
4. Acho que necessitaria de ajuda de um técnico para conseguir utilizar este produto	1	2	3	4	5
5. Considerei que as várias funcionalidades deste produto estavam bem integradas	1	2	3	4	5
6. Achei que este produto tinha muitas inconsistências	1	2	3	4	5
7. Suponho que a maioria das pessoas aprenderia a utilizar rapidamente este produto	1	2	3	4	5
8. Considerei o produto muito complicado de utilizar	1	2	3	4	5
9. Senti-me muito confiante a utilizar este produto	1	2	3	4	5
10. Tive que aprender muito antes de conseguir lidar com este produto	1	2	3	4	5