FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

# PDapp

*A mobile solution for continuous follow-up of Parkinson's disease patients*

Nuno Duarte Ribeiro da Silva Fonseca Oliveira

U.PORTO

FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Mestrado Integrado em Engenharia Informática e Computação

Supervisor: João Paulo Trigueiros da Silva Cunha

February 22, 2021

# PDapp – A mobile solution for continuous follow-up of Parkinson’s disease patients

Nuno Duarte Ribeiro da Silva Fonseca Oliveira

Mestrado Integrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

October 21, 2017

# Abstract

Parkinson's disease is a neurodegenerative disorder. Symptoms develop slowly over the years, and the disease's progression is often different from person to person. These difficulties, paired with limited and infrequent doctor appointments, make the disease's progression hard to track and manage. Parkinson's patients require a better solution to help them manage their disease. The iHandU system is a novel and comfortable wearable device designed to classify the wrist rigidity during Deep Brain Stimulation (DBS) neurosurgery. The iHandU's capabilities have much potential outside of the surgery room and could help patients track their symptoms. PDapp is a mobile app solution to make use of these capabilities and help patients with a continuous follow-up of their disease's progression and treatment management. It strives to connect patients and their doctors outside of the clinical environment and right from the comfort of their homes. It provides patients with a high-quality mobile application for managing their disease with medication information, performing and keeping track of various symptom tests, and keeping their doctors informed of important events related to the disease. On the other hand, the doctors have access to a specialised dashboard, to get an overview of all their patients' history and their most recent events. The PDapp system is composed of a mobile application, a web-based dashboard, and a cloud-based database to persist all the data necessary in real-time. The current system has successfully integrated these components and provides a highly usable and interactive interface for Parkinson's disease patients. The doctor's dashboard hosts a focused and clean overview of their patients' most recent events, history of tests and their results, and prescribed medications. PDapp can significantly impact these patients' lives by easing their access to relevant tools and connecting them to their doctors whenever needed, and all at their arms' reach.

**Keywords**: parkinson disease, mobile application, patient monitoring

# Resumo

A doença de Parkinson é uma doença neuronal degenerativa. Os sintomas desenvolvem-se lentamente ao longo dos anos e a doença progride de uma forma muito variável de pessoa para pessoa. Estas dificuldades tornam a gestão desta doença extremamente difícil, especialmente quando as consultas médicas são pouco frenquentes. Estes doentes necessitam de uma melhor solução para gerir a sua doença. O iHandU é um aparelho inovador que se pode equipar para medir a rigidez do pulso durante cirurgia de estimulação profunda do cérebro (DBS). O iHandU tem potencial para ser usado fora da cirurgia e ajudar doentes a avaliar os seus sintomas. A PDapp é uma aplicação móvel que procura utilizar as funcionalidades do iHandU de forma a ajudar os pacientes com um seguimento contínuo da doença e toda a gestão do seu tratamento. Tem como objetivo criar uma ligação entre os pacientes e os seus médicos, fora do ambiente hospitalar. A PDapp fornece aos seus pacientes ferramentas para gerir a sua doença com controlo da sua medicação, vários testes para os seus sintomas e a possibilidade de informar os seus médicos de acontecimentos importantes relacionados à doença. Os médicos têm acesso a um dashboard especializado onde poderão analisar as informações dos seus doentes, tal como o seu historial e acontecimentos recentes. O sistema é composto de uma aplicação móvel, uma plataforma web para o dashboard e uma base de dados na cloud. A PDapp é capaz de integrar todos estes componentes e fornece uma interface interativa para os doentes gerirem a sua doença. O dashboard dos médicos é focado em apresentar a informação mais essencial de uma forma direta e costumizável. A PDapp pode ter grande impacto nas vidas de pacientes com Parkinson, ajudando a gerir a doença e a liga-los aos seus médicos nos momentos importantes. Tudo no conforto das suas casas, no seu telemóvel.

**Keywords**: doença parkinsons, aplicação móvel, monitorização pacientes

# Acknowledgements

I would like to begin by thanking my family for the invaluable opportunity they provided me with their education and unconditional support throughout my life; I could never be where I am today without it. To my brother, sister, mother, and father, I am forever grateful. To my girlfriend, my unwavering support and partner, your dedication and faith made every hard step while embarking on this journey so much lighter. I can only hope to do the same while you embark on yours.

To my great supervisor Dr João Paulo Cunha, thank you for this opportunity to create a system that will help the lives so many others. Working with such an open and welcoming community like the BRAIN group is a blessing that I will never forget. A special thank you to Duarte Filipe Dias for all the attention and support, so much of this project was possible because of you.

Lastly, I would like to thank the FEUP community, emphasising the teachers that provided me with their knowledge and life experience, for giving me the tools to become a true professional in the coming years.

Nuno Oliveira

*"'Someday' is a disease that will take your dreams*
*to the grave with you."*
Tim Ferriss

# Table of Contents

# List of Figures

# List of Tables

# Abbreviations

| | |
|---|---|
| PD | Parkinson’s Disease |
| app | Application |
| DBS | Deep Brain Stimulation |
| BLE | Bluetooth Low Energy |

# Chapter 1

# Introduction

## 1.1 Context

Parkinson's disease is a neurodegenerative disorder. It affects predominately dopamine-producing ("dopaminergic") neurons in a specific area of the brain called *substantia nigra*. It mainly affects the motor system, which causes unintended or uncontrollable body movements [1].

Its symptoms develop slowly over the years, and the disease's progression varies significantly from patient to patient, making it very hard to predict. Parkinson's is most known for causing various motor difficulties such as:

- Bradykinesia (or slowness of movement), such as difficulty getting up out of a chair
- Dyskinesia, causing involuntary or erratic movements
- Rigidity or stiffness of muscles
- Resting tremor, such as hand tremors while at rest (sitting or walking)

Other non-motor symptoms may also develop throughout the disease, such as:

- Cognitive changes, like attention problems
- Hallucinations and delusions
- Loss of sense of smell or taste
- Sleep disorders
- Nausea

Parkinson's is somewhat manageable in its early stages. The disease's progression is generally divided into five stages, in which stages 1-3 experience a slow increase of the symptoms while still allowing everyday activities to continue. Stages 4-5 are the most debilitating and difficult, 24-hour assistance is needed, and at stage 5, the patient is bedridden.

One of the biggest challenges dealing with Parkinson's is that its progression varies significantly, and symptoms may develop quickly without warning. This erratic nature of the disease makes it hard to prepare and adjust treatment when needed, especially with an average of 2 doctor appointments every year, as claimed in multiple studies [2].

Although Parkinson's has no cure, some treatments help deal with the symptoms and their effect on life quality. The most notable one is *Levodopa* (+ *carbidopa*), a medication that increases dopamine concentrations in the central nervous system. *Levodopa's* most significant issue is its loss of efficiency over time. It also requires constant adjustment to combat side-effects (using *carbidopa).*

A more advanced treatment option is Deep Brain Stimulation (DBS). It is a surgical procedure used to treat several disabling neurological symptoms, in later stages of the disease. DBS uses a surgically implanted, battery-operated medical device called an implantable pulse generator (IPG) to deliver electrical stimulation to specific brain areas that control movement, thus blocking the abnormal nerve signals that cause symptoms [3].

Apart from treatment options, various tools have proven useful over the years in the diagnosing and managing Parkinson's disease. Wearable devices are one of these tools that help health professionals diagnose symptoms quantitively during evaluations. Using small but powerful sensors, with comfortable straps, these devices can be used in various body locations to analyse movements and variations with high precision. Devices like this are used to measure symptoms such as wrist rigidity [4], gait [5], hand tremor [6], and many more.

One vital part of this project is the *iHandU*; a wearable device developed at INESC TEC. This device was developed for measuring wrist rigidity, in real-time, during a DBS surgery. With an accuracy of 83.9%, the iHandU can provide the surgeon with critical information with no delay or the need for subjective scales or clinical agreement [4]. This powerful device can also be easily refitted for measuring other symptoms in other regions of the body, making it a versatile tool for diagnosing PD symptoms.

Another tool that has recently been used with this end is mHealth mobile applications [7]. Alongside many of the wearable devices referred before, a mobile application exists for displaying the data in a portable way. Since smartphones are a part of most peoples' lives, they can become a powerful and accessible tool for managing Parkinson's disease.

## 1.2 Motivation

Parkinson's disease's erratic nature and its progression make it even more challenging to live with than it already is. With very few doctor appointments throughout the year, there is no guarantee that when the disease progresses and the symptoms aggravate, the necessary changes to adjust the treatment accordingly will be done at the right moment. Since managing Parkinson's is all about adjusting medication at the correct times to avoid the infamous "OFF" states, this delay in information is very detrimental to the patient's life.

Another significant issue is identifying when the disease is progressing. Only a health professional will be able to identify the variation in the symptoms accurately, not the patient. As such, either the doctor regularly evaluates the patient, which we already know is not usually the case, or the patient resorts to automated tools that can accurately measure the symptoms' variation. Thankfully, wearable devices have been proven to measure most symptoms with decent accuracy, providing much-needed help and automation in this field. Their biggest issue is that there is no “flagship” option in the market that offers all the most needed features with good quality. There is an opportunity to introduce a well-rounded, high-quality system that can provide constant monitoring and treatment management for each patient and relay it to their respective doctor.

On another domain, the iHandU is a very capable device that focuses on a very niche market. Suppose the iHandU could be brought outside of the surgery room and into the patient’s home and everyday life. In that case, it could provide its accuracy and flexibility at any time or setting. If a system is set up that can use the iHandU for a constant follow-up during the pre-operatory and post-operatory phase of a DBS surgery, health professionals would have even more information on how to act and manage the treatment.

With these three main points, our motivation lies in providing a better quality of life to Parkinson’s patients using the underdeveloped world of mHealth devices, emphasising the iHandU and its capabilities.

## 1.3 Objectives

To tackle the issues stated before, we designed and developed a mobile system that can provide all the tools PD patients and health professionals require using the latest technologies. This system features a mobile application for the patients, a dashboard for the health professionals, the iHandU wearable device for accurate measuring of specific symptoms, and a cloud-based database for managing the necessary data. It eliminates the information barrier between patients and doctors while also providing all the tools needed for managing the disease outside of the clinical environment.

The mobile application provides patients with an intuitive interface for managing their disease. Patients can inform the doctor by reporting a significant event about their symptoms, like an OFF state, accompanied by a self-evaluation so their doctor can be notified and assess the situation accordingly. A wide array of symptom tests can be performed with a wearable device such as the iHandU. Lastly, management features like reminders for medication or regular exercises/tests were partly implemented.

As for the doctors’ domain, the dashboard web application displays all the latest and most important information about their patients and recent events. It uses an overview page that provides a general view of what happened recently and requires the most attention. Dedicated profile views of each patient show their entire history of medication, events, symptoms, and more. Expandable views for analysing multiple data points were implemented for a better user experience.

The interface with the iHandU is essential for providing proper quality measurements of the symptoms and was one of the first challenges to be tackled. A flexible architecture was used to help any future additions of other devices to be implemented easily.

Lastly, a cloud-based database provides access to the necessary data anywhere and with very few scalability issues. Ensuring that such data is available and secured with trusted technologies is extremely important.

## 1.4 Document Structure

This document will detail every step taken since the research, design, development, and testing of the system.

Starting with the current state of the art, with sections dedicated to the state of the mHealth market in the Parkinson’s disease domain, as evaluated by systematic reviews; related work and existing products, gathered from a literature and app store review; and conclusions with details on the crucial aspects identified.

The next chapter is dedicated to the problem and proposed solutions. It will follow a System Requirements Specification (SRS) format as a typical software project that identifies the goals and objectives, project scope, major constraints, product perspective, features, users, and functional requirements.

The framework and coding guidelines used for the development are specified in chapter 4. This chapter should explain the mindset behind most development decisions, allowing a newcomer to quickly understand the system.

Chapters 5-7 are dedicated to implementation details. Starting with the iHandU integration and its challenges and followed every aspect behind the mobile application and the doctor’s dashboard. These three chapters detail all the implementation work and the current version of PDapp.

The final results are presented in chapter 8, detailing the tested use cases and user feedback.

The last chapter hosts the conclusions and future work.

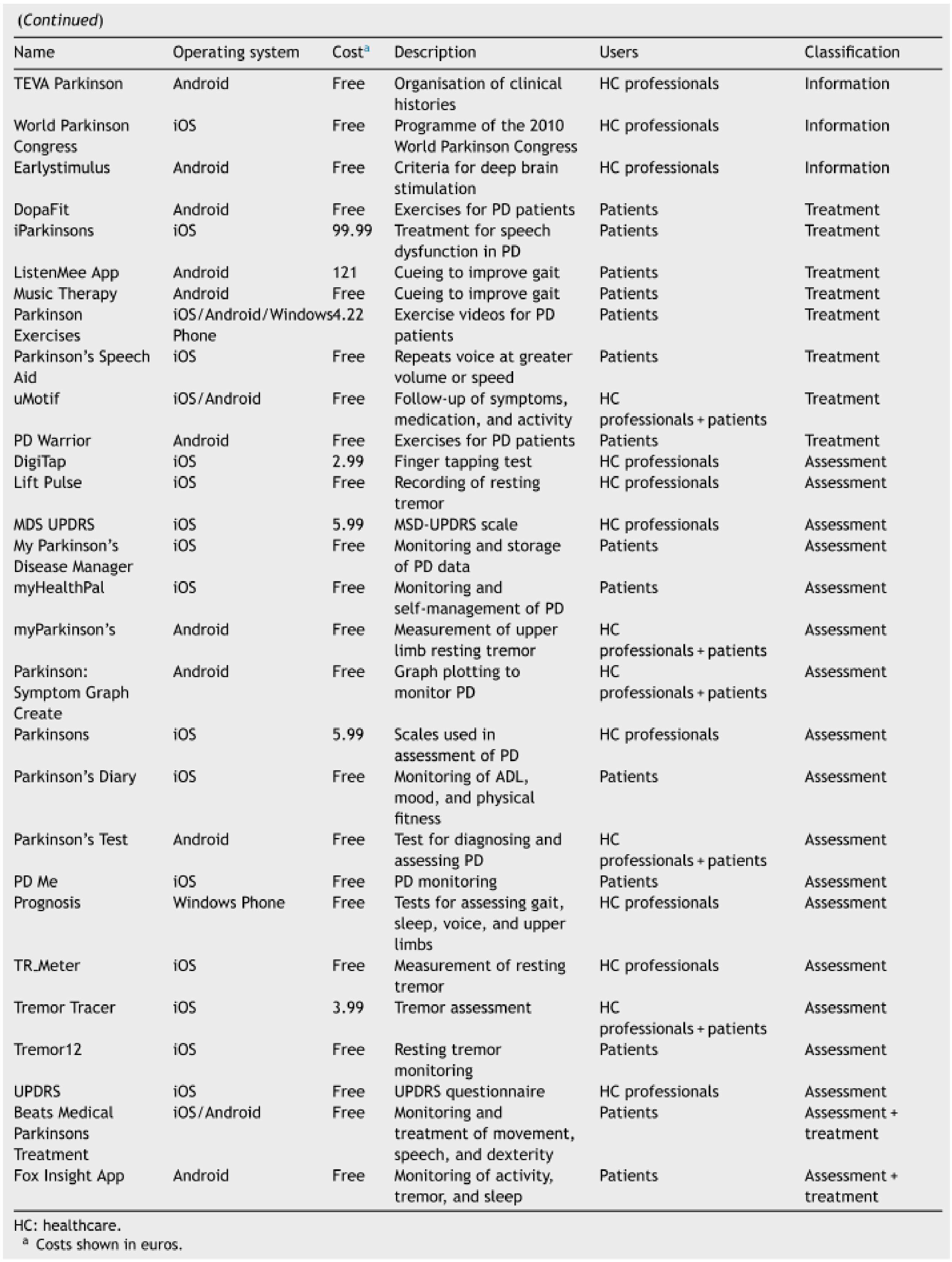
Chapter 2

# State of the art

## 2.1 Analysis of the mHealth market for Parkinson’s disease

The mHealth market for Parkinson’s disease has been developing considerably in the past decade. To evaluate its current state and learn from the most iconic products present, we searched for academic works on Google Scholar that executed a systematic review of the mobile marketplace. Using the keywords “review”, “mobile”, “applications”, “mHealth”, and “parkinson’s disease”, two papers showed potential and touched all the subjects we deemed essential.

The oldest and most considered was a literature and app store review conducted in 2016 by a Spanish researcher team [7]. Their review used multiple databases, retrieving articles from 2011 to 2016, written in English or Spanish. They discovered a total of 125 apps, 56 of which were considered “potentially useful for PD” and 69 as “designed specifically for PD”. Of the Parkinson’s specific ones, 29 were considered dedicated to assessing symptoms, 23 to providing information about the disease, 13 were dedicated to the treatment of the disease, and the last 4 handled both the assessment and treatment of Parkinson’s. The study also showed concerns about the lack of evidence of the application’s usefulness or quality. On the other hand, they acknowledge the mobile devices’ capability at measuring most motor symptoms with considerable accuracy. Finally, they also recognised large companies' initiatives (Apple and Roche) of starting projects in the market.



(*Continued*)

| Name | Operating system | Cost[a] | Description | Users | Classification |
|---|---|---|---|---|---|
| TEVA Parkinson | Android | Free | Organisation of clinical histories | HC professionals | Information |
| World Parkinson Congress | iOS | Free | Programme of the 2010 World Parkinson Congress | HC professionals | Information |
| Earlystimulus | Android | Free | Criteria for deep brain stimulation | HC professionals | Information |
| DopaFit | Android | Free | Exercises for PD patients | Patients | Treatment |
| iParkinsons | iOS | 99.99 | Treatment for speech dysfunction in PD | Patients | Treatment |
| ListenMee App | Android | 121 | Cueing to improve gait | Patients | Treatment |
| Music Therapy | Android | Free | Cueing to improve gait | Patients | Treatment |
| Parkinson Exercises | iOS/Android/Windows Phone | 4.22 | Exercise videos for PD patients | Patients | Treatment |
| Parkinson's Speech Aid | iOS | Free | Repeats voice at greater volume or speed | Patients | Treatment |
| uMotif | iOS/Android | Free | Follow-up of symptoms, medication, and activity | HC professionals + patients | Treatment |
| PD Warrior | Android | Free | Exercises for PD patients | Patients | Treatment |
| DigiTap | iOS | 2.99 | Finger tapping test | HC professionals | Assessment |
| Lift Pulse | iOS | Free | Recording of resting tremor | HC professionals | Assessment |
| MDS UPDRS | iOS | 5.99 | MSD-UPDRS scale | HC professionals | Assessment |
| My Parkinson's Disease Manager | iOS | Free | Monitoring and storage of PD data | Patients | Assessment |
| myHealthPal | iOS | Free | Monitoring and self-management of PD | Patients | Assessment |
| myParkinson's | Android | Free | Measurement of upper limb resting tremor | HC professionals + patients | Assessment |
| Parkinson: Symptom Graph Create | Android | Free | Graph plotting to monitor PD | HC professionals + patients | Assessment |
| Parkinsons | iOS | 5.99 | Scales used in assessment of PD | HC professionals | Assessment |
| Parkinson's Diary | iOS | Free | Monitoring of ADL, mood, and physical fitness | Patients | Assessment |
| Parkinson's Test | Android | Free | Test for diagnosing and assessing PD | HC professionals + patients | Assessment |
| PD Me | iOS | Free | PD monitoring | Patients | Assessment |
| Prognosis | Windows Phone | Free | Tests for assessing gait, sleep, voice, and upper limbs | HC professionals | Assessment |
| TR_Meter | iOS | Free | Measurement of resting tremor | HC professionals | Assessment |
| Tremor Tracer | iOS | 3.99 | Tremor assessment | HC professionals + patients | Assessment |
| Tremor12 | iOS | Free | Resting tremor monitoring | Patients | Assessment |
| UPDRS | iOS | Free | UPDRS questionnaire | HC professionals | Assessment |
| Beats Medical Parkinsons Treatment | iOS/Android | Free | Monitoring and treatment of movement, speech, and dexterity | Patients | Assessment + treatment |
| Fox Insight App | Android | Free | Monitoring of activity, tremor, and sleep | Patients | Assessment + treatment |

HC: healthcare.
[a] Costs shown in euros.

*Figure 1 - Main characteristics of mobile applications for Parkinson's disease.* [8]

This first study revealed that the mHealth market for Parkinson's was growing and quite diverse. The majority of applications were focused on generic aspects, such as spreading information about the disease or testing specific symptoms. On the other hand, the more specialised applications focused on treatment or even offering the entire spectrum of tools, were very few. The market appeared to be in its early stages, lacking enough information to ensure these applications' quality and effectiveness, but it showed promise on the mobile device's capabilities.

A paper from 2019 by another Spanish researchers team conducted a more recent market study [9]. It recognised the lack of studies performed in that field, the most notorious one being the previous study from 2016. It conducted a literature and app store review using the keywords "parkinson's disease" and various descriptors of its symptoms. Similarly to the previous study, most of the market aimed to monitor symptoms of the disease, instead of treatment. Of all the apps analysed, 75% were focused on the patients, 37% were designed for medical professionals, and 9% were designed for clinical evaluations. They also analysed these applications' payment methods, showing that 63% were free and the remaining 37% paid, with a price range of 0.99$ to 9.99$. Unfortunately, the study did not specify how these conclusions were reached. There is no information about how they considered free/premium subscription models and one-time acquisitions instead of monthly subscriptions. The applications' median rating was considered for both platforms, revealing a rating of 3.2 for iOS apps and 4.2 for Android apps. Hybrid apps (both for Android and iOS) achieved a median rating of 4.5 for iOS and 4.1 for Android. The applications were also evaluated according to usability, with a score of 1 to 5. The overall average was 4.2, with the three worst aspects being documentation, visibility of system status and user control/freedom. iOS apps had slightly higher scores than Android (4.6 vs 3.9). A relevant observation was that every major symptom of the disease was handled by at least one application, yet no application covered them all.

In general, most apps had a good usability rating, which could be caused by trimming results during the searching phase, where they selected the top 25% according to rating.

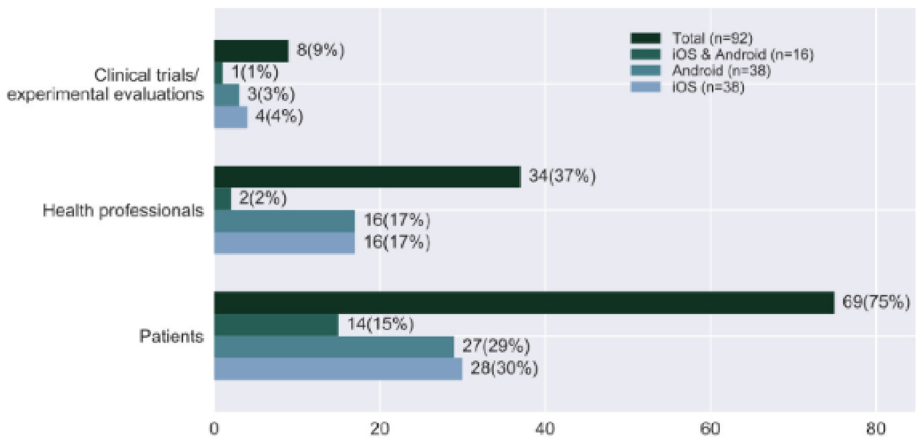


*Figure 2 - App distribution by operating system and target users* [7]

This study provided a much more in-depth analysis of the applications and their quality. The market appeared to favour the more assessment-oriented apps, with a short supply of apps focused on treatment or medical professionals. Documentation and user experience seemed to be the most lacking in terms of usability.

The mHealth market for Parkinson’s disease appears to be well developed and not too populated, with plenty of opportunities to improve. It still lacks well-rounded applications that are more focused on the treatment and professional side. The opportunity to use the mobile phone’s portability and capacity paired with a more specialised mobile device to comply with the health professional’s needs for quality, should not be overlooked.

## 2.2 Existing products and research studies

To better understand which solutions already existed in this space, we conducted a literature review of studies about mobile applications or systems for Parkinson’s disease. We came across multiple studies which represented various aspects of envisioning, designing, deploying, and analysing said systems. A few of the most relevant ones will be discussed and summarised next.

### 2.2.1 PD_Manager

*PD_Manager* [10] is one of the most developed and feature-heavy of the mHealth solutions for Parkinson’s that we encountered. A publication from 2017 [5] profoundly describes the system, and the results gathered. The app attempts to provide all the tools necessary for Parkinson’s patients in a holistic approach. The features designed for the patients use a few lightweight nonobstructive sensors to evaluate all the primary symptoms of the disease with minimum effect on daily activities, specifically a smart wristband and an insole sensor for each foot. The patients also have access to a medication plan, nutrition diet and multiple cognitive/motor self-evaluation tests. Another small but heavily requested feature was for the ability to reproduce text phonetically.

*Figure 3 - PD_Manager logo*

One of the most significant driving forces of this system is the interaction with health professionals to analyse and manage patients and their needs. These medical professionals have access to a unified system for accurate patient monitoring and assessment of symptoms. A major feature is the use of a *decision support system (DSS)* to facilitate patient monitoring and treatment planning, using machine learning. It also offered a patient assessment view for analysing symptom trends. Medication had its view as well, where the intake history and medication times could be followed. There was a notification system for reminding patients or caretakers about medication times or requesting self-evaluations to be performed on-demand. The architecture chosen was a cloud-based infrastructure connected with the three sensors and the smartphone. The application was developed as a hybrid to reduce costs and compatibility issues. Data security was considered, and all communication channels and data transfers were handled using HTTPS, SSL/TLS and OAuth 2.0. Finally, a minor but relevant consideration was the support for various notifications, such as SMS, email and Google Play.

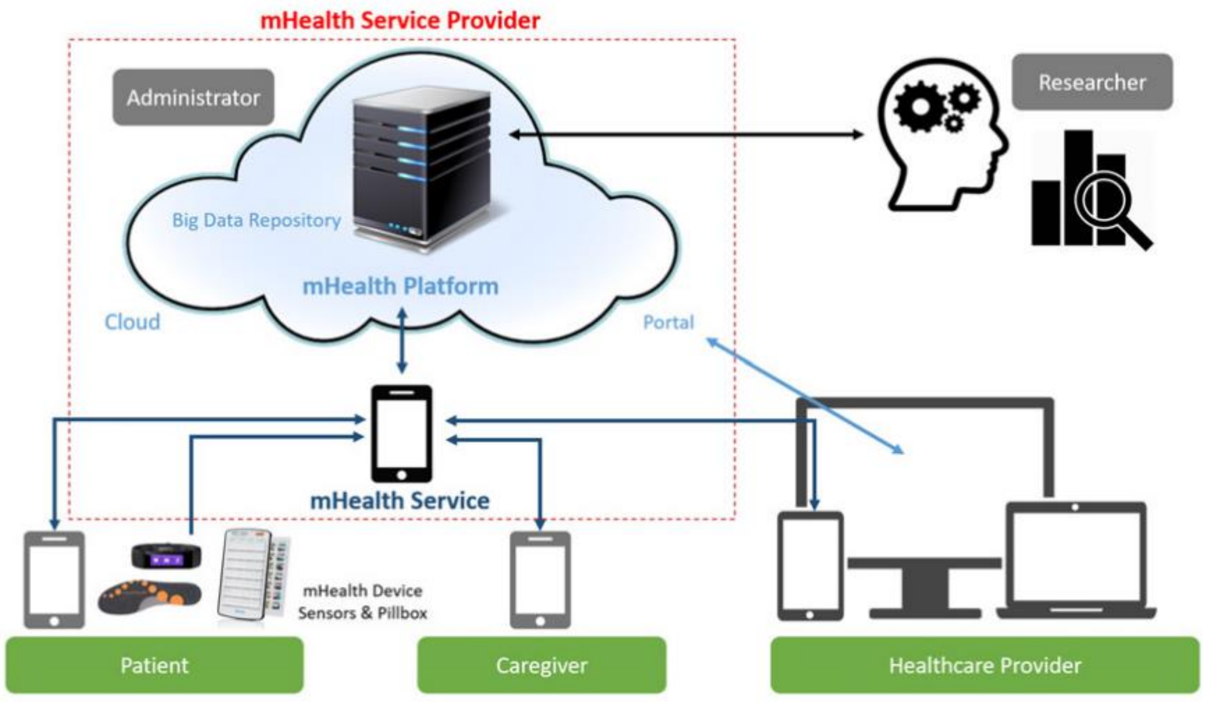


*Figure 4 - Schematic representation of PD_Manager's mHealth platform cloud-based architecture* [5]

PD_Manager's project was completed successfully in March 2018 [10] with intentions of seeking another round of funding to finalise the platform and bring it to market [11]. The current status of PD_Manager is unclear.

### 2.2.2 "Mobile integrated assistance to empower people coping with Parkinson's disease"

A short article [12] from Portugal (2015) proposed a mobile application design to support Parkinson's patients, their carers, and the health professionals involved. It emphasises the importance of involving the patient, carer, and doctor as one. Some performed user studies revealed that Parkinson's patients deemed such a service to be "extremely relevant". Some of the features considered were, the ability to quickly find answers about Parkinson's disease, write notes, have alerts about medication and exercises, and communicate with the doctors through the app.

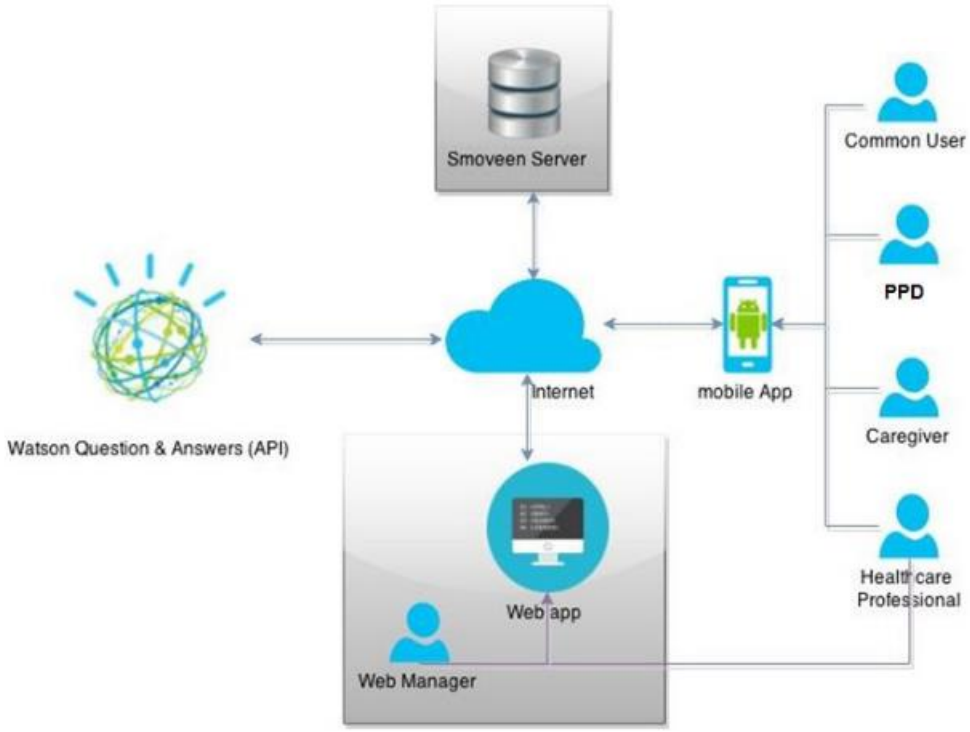


*Figure 5 - Proposed system architecture* [12]

### 2.2.3 Fox Wearable Companion app

The *Fox Wearable Companion (FWC)* app was developed for a 2019 study [13] to assess the feasibility and clinical utility of data obtained using mHealth technology, from Parkinson's patients, in a clinician's dashboard. The patients participating wore a smartwatch and used the FWC app to stream movement data, report symptom severity, and register medication intake, for six months. Clinicians reviewed the patients' data in a dashboard on four occasions throughout the study. A total of 51 patients participated with 76% completing the full 6-month study. One relevant feature available for the patients was the use of symptom reports, which used a scale of "none", "slight", "mild", "moderate", and "severe", corresponding with the MDS-UPDRS scale. Another curious feature was the ability to skip reminders. Throughout the study, many changes were requested by the clinicians, such as the addition of descriptive text and "info" buttons in each display of the dashboard, the availability of various displays for different perspectives of sensor-derived data, the integration of "markers" representing medication intakes on graphs of data acquired from the sensors, to help make connections between the data. Lastly, there was a preference for hourly displays of the graphs.

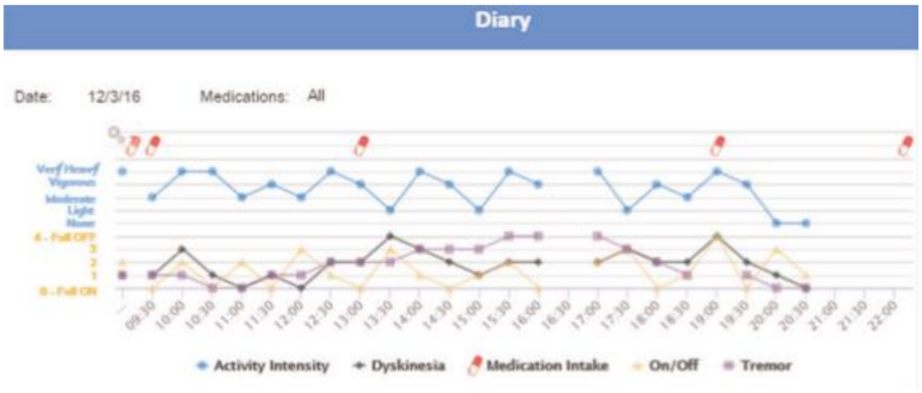


*Figure 6 - Clinician dashboard display of an ON/OFF diary (ePRO)* [13]

Many conclusions were reached from this study: medical compliance and the severity of symptoms were the most beneficial dashboard components. An important observation was that patients skip days, and compliance drops over time. A suggested approach to help compliance was the use of personalised feedback and interaction. In the end, it was stated that clinicians should be involved from the earliest stages of development.

### 2.2.4 STOP

*STOP* was another mobile application developed for a 2018 study [14], with the objective of screening Parkison's symptoms and medication intake. It contained a game for tracking motor symptoms and a medication journal for recording medication intake and adherence. The study conducted a 1-month real-world deployment with 13 patients from two countries. Its two main features were the medication journal and the "Ball game", which utilised the phone's sensors to evaluate symptoms such as tremor, attributing a score in the end.

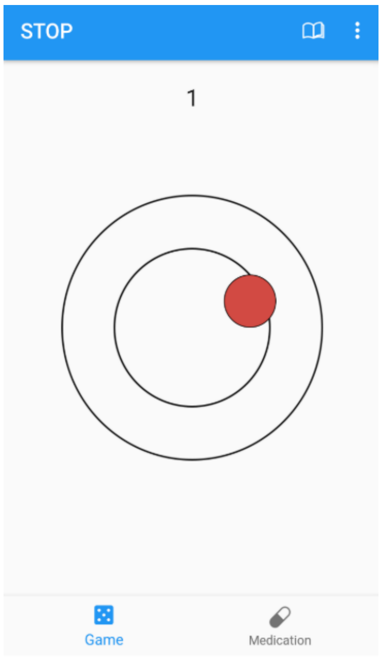
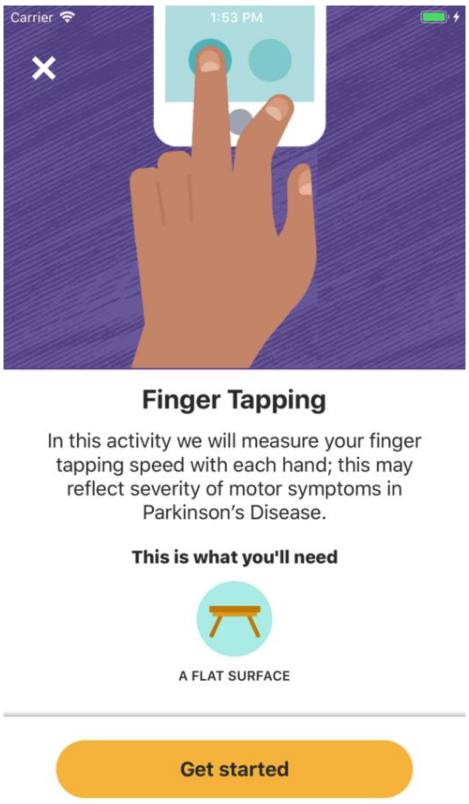


*Figure 7 – STOP's ball game interface* [14]

The authors considered that the mobile phone's main difficulties were due to text input, which revealed patients preferred to use voice input or even a stylus pen. A particular observation was that many patients did not need the medication journal since they already had well-established medication management routines. The patients claimed that the app's primary benefit was a "feeling of control" over their disease, due to following their performance. Concerns for data security were few, and patients claimed that they had no issues providing personal data if it helped medical professionals make better decisions.

### 2.2.5 mPower

*mPower* [15] is probably the most iconic and prominent application in the field. Developed for a clinical observational study [2] in 2016, conducted purely from an iPhone app interface. The study utilised a novel remote approach to enrolment in which participants self-guided through a visually engaging and complete consent process, before deciding to join the study. The app featured four main activities: memory, tapping, voice, and walking. The tasks were supposed to be done before taking medication, immediately after taking medication, and at some other time. A total of 9,520 participants consented to the study and agreed to share their data broadly with the research community. Of the 6,805 participants that completed the enrolment survey, 1,087 (16%) self-identified as having a professional diagnosis of Parkinson's disease. All data acquired is openly available [16]. One of the most significant driving forces of this application was its excellent quality user interface and experience.



Finger Tapping

In this activity we will measure your finger tapping speed with each hand; this may reflect severity of motor symptoms in Parkinson's Disease.

This is what you'll need

*Figure 8 - mPower finger tapping activity* [2]

### 2.2.6 ParkNosis

*ParkNosis* was an application and study [6] from 2016 to provide a mobile application for assessing symptoms using quantitative and qualitative tests. This study provided very insightful and complete research behind each test developed. Its primary objective was to detect Parkinson's symptoms in users. It provided features such as hand tremor tests (using a smartwatch), finger tapping tests, and spiral drawing tests. The scores used a scale that considered data from users' results with and without Parkinson's disease. The instant feedback provided by showing results immediately after testing was considered very valuable by the authors. Phone readings were not considered very accurate. The authors noticed that most other apps did not have involuntary movement detection. Development wise, the use of a remote database was praised for its ability to access data anywhere. Another crucial aspect was using a simple and interactive GUI, with guidelines and tooltips that allow users with little to no technical background to use the application with ease.

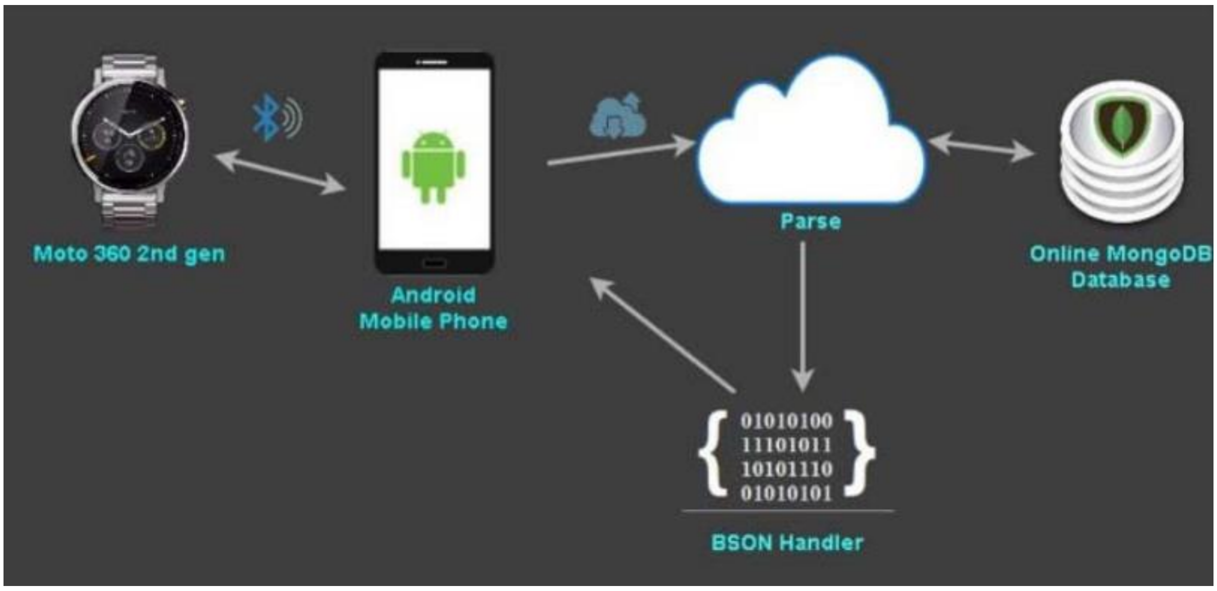


*Figure 9 - ParkNosis system architecture* [6]

### 2.2.7 "Cloud based framework for Parkinson's disease diagnosis and monitoring system for remote healthcare applications"

A couple of studies we discovered took an approach focused on analysing speech disorders and *dysphonia*. One of the studies, conducted in 2017, designed a cloud-based system with a mobile application [17]. This system focused on diagnosing and analysing users as potential Parkinson's patients, in rural areas with little to no access to quality healthcare. The cloud-based system gave patients of low resources areas access to doctors who could check their results and detect if they had Parkinson's disease. The system achieved a 96.6% accuracy at detecting Parkinson's via speech analysis, over the cloud.

### 2.2.8 Apkinson

Another system that focused on speech disorders was *Apkinson*, an android application, detailed in a paper from 2017 [18]. This system took a novel approach by intercepting phone calls and recording the audio for a constant and nonobstructive analysis for speech disorders. If the speech was getting worse, the user was notified with a warning. Although the results did not look as promising as other speech-oriented systems, the innovative and nonobstructive approach has much potential.

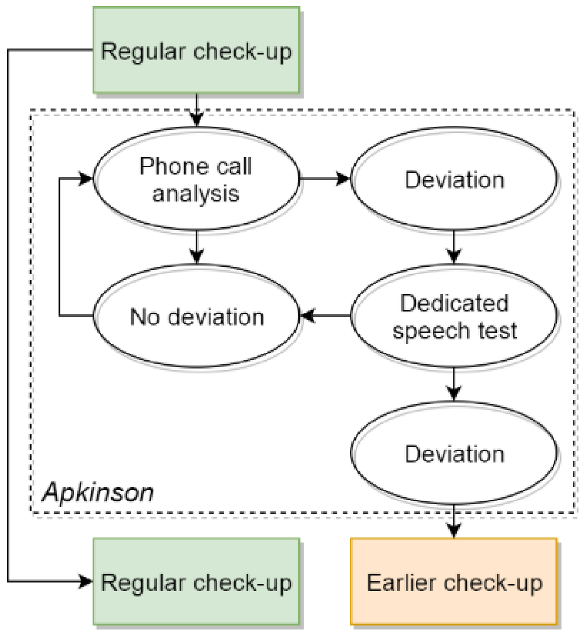


*Figure 10 - Schematic diagram of the remote monitoring* [18]

### 2.2.9 "A Mobile App for the Remote Monitoring and Assistance of Patients with Parkinson's Disease and their Caregivers"

One study from 2018 focused on improving the communication between patient, caregiver and specialists [19]. It conducted a one-year experiment with ten patients and seven caregivers. The system evaluated the patient's condition through specialised surveys sent to both the patient and caregiver. Data was then sent to a repository and later dispatched to three specialists (nurse, neurologist, and psychologist). The experiment showed that compliance reduced for those patients that saw the best results, probably from a lack of any real need to provide new information.

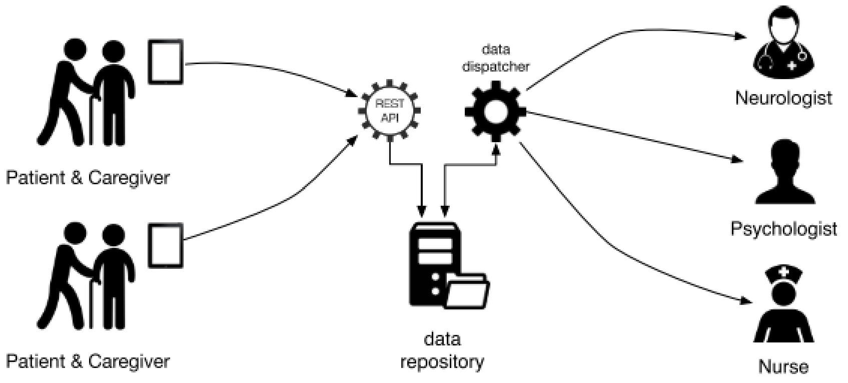


*Figure 11 – System concept diagram* [19]

### 2.2.10 "Design and Usability Evaluation of Interface of Mobile Application for Nutrition Tracking for People with Parkinson's Disease"

A more nutrition-focused study from 2019 appeared as part of the *PD_manager's* system referenced before [20]. This paper detailed the choices behind each interface and the usability tests performed. The tests were conducted to a small group of people, so the results should not be too conclusive, but the detailed thought process throughout the paper is valuable.

## 2.3 Conclusions and important aspects

The mHealth solutions for Parkinson's disease appear to have been developing for quite some time. However, most apps are focused on self-assessment of a few symptoms and very few extra features. Some attempts have been made at making a feature-heavy app that satisfies all the major needs of Parkinson's patients, such as PD_Manager [5]. However, its current status is still unclear, and no readily available solution exists.

After reviewing the studies, it became clear that smartphones are more than capable of performing low-level tests for some symptoms (hand tremor, voice). However, their merit as medical devices is still unconsidered. For this reason, the combination of a mobile app and wearable devices, designed for measuring Parkinson's symptoms accurately, can provide the patients with the quality tools they require and the approval of the health community. Compliance seemed to be a significant issue during many studies, as is usually the case with mHealth solutions. Some authors advised personalised feedback and interactions as a possible solution to the problem. A more event-oriented approach to reminders and interactions, instead of routine-based, could prevent the patient from losing interest due to repetition. It became evident that most applications are centred around the patient and his self-care, while very few connected the patient with medical professionals. This connection provides much more value and directly handles the issue that Parkinson's patients have due to few appointments over the year. Many applications had issues with user experience due to inadequate and uninteresting designs or lack of clear information for the less tech-savvy users. Medication management using journals was a common practice that saw good results and should be considered on future systems. Voice input as an alternative to text was a heavily requested feature, due to the increasing difficulty of using text input in the later stages of the disease. One of the studies [19] stated that providing instant feedback on the evaluation results conveyed a “sense of control” over their disease, said the patients, by following their performance. Being transparent about the results and showing the patient the whole process could help with engagement and provide a more trustworthy product.

Focusing now on using a dashboard for the health professionals, access to various data displays to better understand trends and relations between data points was a heavily requested feature. Following the same thought, providing high customisable graphs, tables, and other displays could help the professionals explore the data freely to achieve their goals. Being able to join different data sets, such as medication intake and symptoms severity, proved to help identify trends and causes that could further manage the disease.

Plenty of valuable information could be extracted from market studies and clinical trials of related applications and systems. With this wide array of viewpoints, we can now develop a system capable of approaching all the major aspects that Parkinson’s patients require to manage their disease. It also became clear that constant interaction with the medical professionals throughout the system's development is crucial to provide the most value from their dashboard. Using this knowledge, we can now design the proposed *PDapp* system.

# Chapter 3

# Problem and proposed solution

There is a lack of a mHealth solution for Parkinson's disease that can offer the necessary tools for keeping track of the disease and managing it. No active system exists that provides the user with ways to measure the most relevant symptoms of their disease, document them for future reference, and inform their respective doctors whenever a significant event occurs while presenting the doctors with all the data gathered, on-demand.

The possibility to create a new system that applies the knowledge acquired over the recent years and makes an invaluable impact on the patients' lives is available and very valuable. Paired with the use of the iHandU's capabilities outside of a DBS surgery, this system can provide a full spectrum service to Parkinson's patients with high quality.

The following topics detail the proposed system and its essential aspects. They follow the general structure of a System Requirements Specification (SRS) document, emphasising goals, objectives, scope, product perspective, functional requirements and system architecture.

## 3.1 Goals and objectives

The goal is to develop a system for the continuous follow-up of PD patients. This system should consist of a mobile application for the patients, a web-based dashboard for the doctors, and a cloud-based database for persisting the data necessary.

*Objectives that were not fully achieved during this thesis are marked with (future work). More details can be found in the final chapter (Conclusions and future work).*

There are two main objectives:

The first is to develop a mobile interface for PD patients to use at home for:

- issuing various events (such as "OFF" states, falls) with an associated test to inform the doctor of the disease's progression
- doing tests regularly with wearable devices (iHandU) to keep a history of results for analysing symptom's progression through time
- checking prescribed medications and routine tests
- getting reminders of medication times and upcoming routine tests *(future work)*

The second objective is to develop a dashboard on a web platform for the doctors. It should provide the doctors with:

- an overview of all their patients and their recent events issued
- each patients' medical profile
- medications prescribed
- events issued history
- test performed history
- medication intake history *(future work)*
- routine tests prescribed *(future work)*

Some secondary objectives include:

- the addition of a clinical mode for doctors performing tests in a clinical setting
- the possibility to integrate other hardware for the analysis of different metrics

## 3.2 Project scope

The scope should consist of a mobile application, a web platform for the doctor's dashboard and a cloud-based database for persisting the data.

The mobile application is expected to integrate with the iHandU for performing tests. All new data acquired should be sent to the system's database in the cloud. Push notifications should be used to notify of significant events (medication times, changes issued by the doctor, and routine tests).

The dashboard should access the database for viewing and editing all relevant data of a patient. Customisable widgets should be used to help interpret the data.

The database should be hosted in the cloud to ensure availability.

## 3.3 Major constraints

The application's usability is crucial, given the target user base of PD patients and their difficulties. The user experience should be tested thoroughly throughout the entire development phase.

The possibility of future integrations with different hardware and use cases requires a flexible and scalable architecture to be defined early.

Data privacy is imperative, given the use of detailed patient data.

## 3.4 Product Perspective

PDapp is in part an enhancement of the iHandU's scope and capabilities. Its objective is to allow the use of the iHandU outside of the clinical environment. As well as provide a continuous follow-up of PD patients and their treatments.

If all the objectives are accomplished, both main and secondary, the system should look as follows:

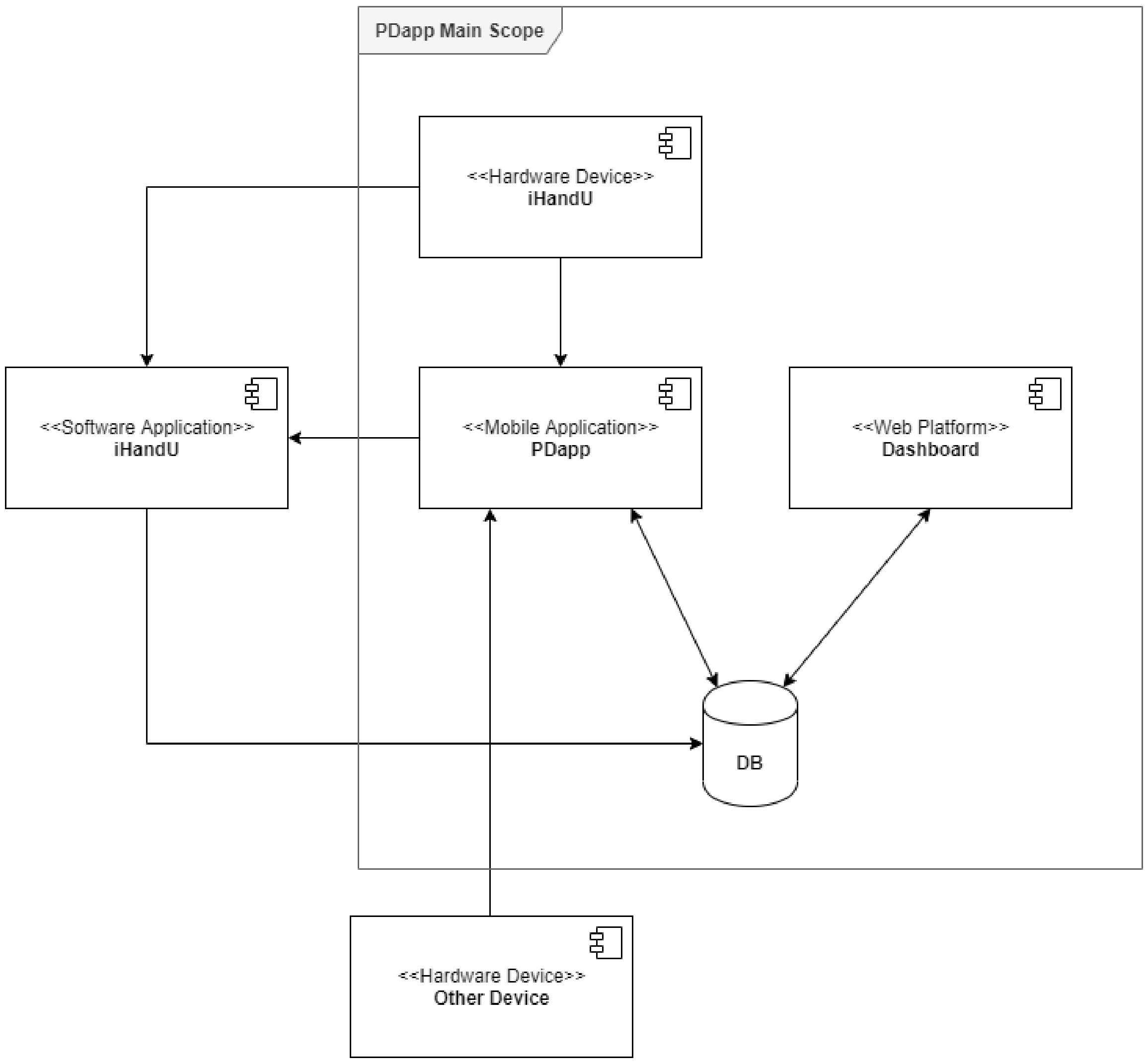
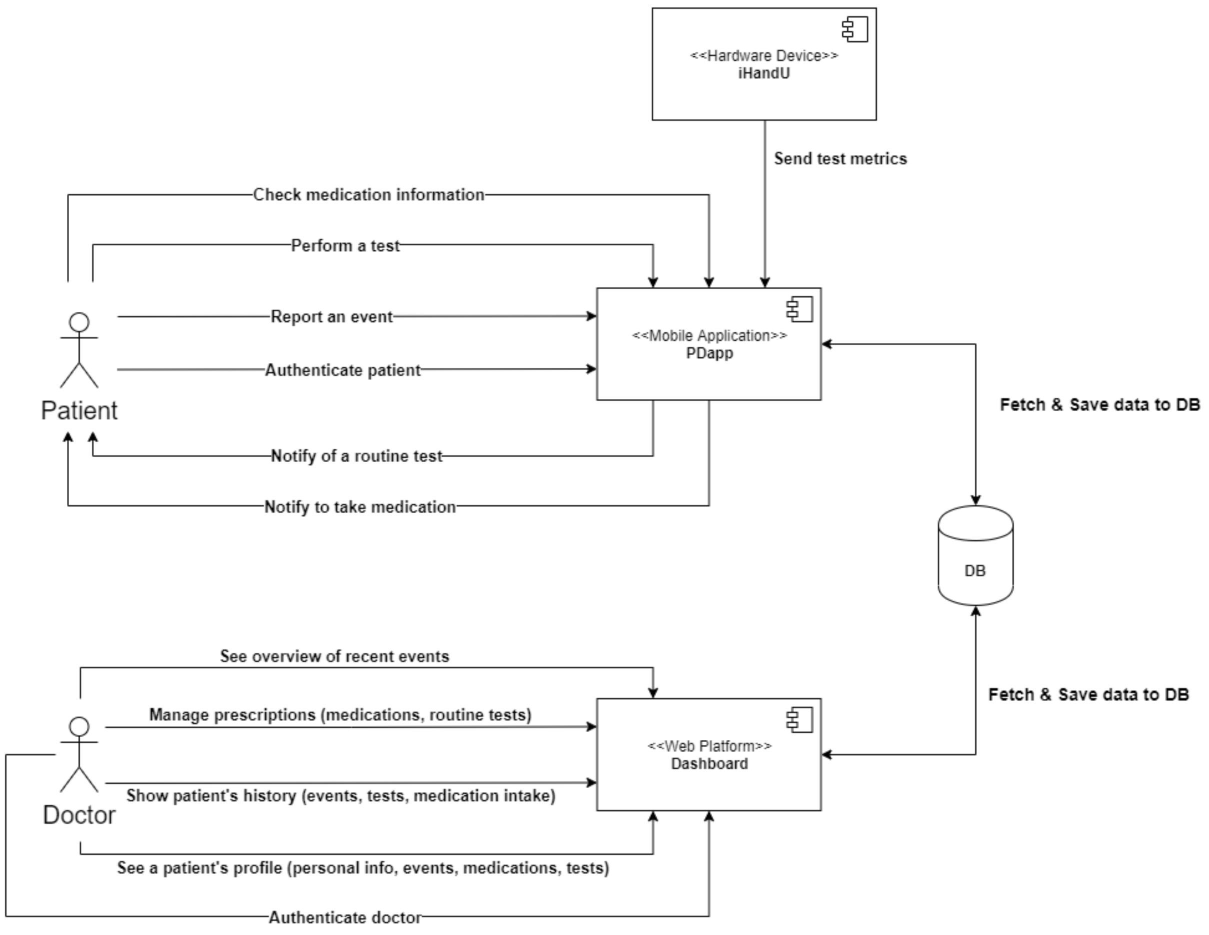


*Figure 12 - System diagram of the overall system*

## 3.5 Product Features

On a high-level, the system should be capable of:

1. authenticating patients and doctors before accessing privileged data
2. accessing and managing the data persisted in the cloud-based database
3. sending patient event reports of relevant occurrences, alongside a self-evaluation (test) performed just before
4. allowing each patient to check their medication information, history of intakes, and be notified of medication times
5. performing tests, with wearable devices, for a constant measuring of their symptoms' progression
6. notifying upcoming routine tests, if prescribed by the doctor
7. allowing a doctor to check an overview of all its patients and their recently issued events
8. show a complete view of each patient's situation, with information about the patient, medication, tests performed, and events issued.
9. manage patients' details, such as prescribed medication and routine tests



<<Hardware Device>>
iHandU
Send test metrics
Check medication information
Perform a test
Report an event
Authenticate patient
Patient
<<Mobile Application>>
PDapp
Fetch & Save data to DB
Notify of a routine test
Notify to take medication
DB
See overview of recent events
Manage prescriptions (medications, routine tests)
Fetch & Save data to DB
Doctor
<<Web Platform>>
Dashboard
Show patient's history (events, tests, medication intake)
See a patient's profile (personal info, events, medications, tests)
Authenticate doctor

*Figure 13 - High-level features diagram*

## 3.6 User classes and characteristics

The following table details the main users of the system at a high-level, in classes.

| **Name** | **Description** |
|---|---|
| *Patient* | The main user of the application. Should be able to access the information specified by the Doctor, such as medication and exercises proposed. |
| *Caretaker* | Virtually the same as the Patient. If the patient is not capable of handling the application himself, it should be the caretaker's responsibility to handle it on his behalf. |
| *Nurse* | A nurse may use the application during evaluations to help the patient perform some exercises correctly. As for the features available to them, they should be same as the patient. |
| *Doctor* | The doctor responsible for a given patient. Can access the dashboard and manage patient data such as medication and exercises. Is also able to check the patient's history and generate reports. |

*Table 1 – Table of high-level user classes of the system*

## 3.7 Functional Requirements

The more specific and technical aspects of the system are detailed in this section.

### 3.7.1 Actors

The main actors of the system are detailed in the following table:

| **Name** | **Description** |
|---|---|
| Patient | The main user of the mobile application. Should have access to all features in the mobile application after being properly authenticated. |
| Doctor | The doctor responsible for a given patient. Can access the dashboard and see the data of the patients under his responsibility. Can also manage patient data such as prescribed medication and exercises. Is also able to generate reports. |
| Database | The cloud-based database for persisting and fetching the required data. |
| Notification Server | A notification server will be responsible for sending all the necessary notifications to the Patient's device |

*Table 2 – Table of the actors of the system*

### 3.7.2 Use cases

All use cases are presented in the following diagrams, divided by the system's two leading platforms (mobile app/dashboard).

The dashboard's homepage will be the overview page. From there, the doctor may check its patients list or the most recent history of events issued. From the patients list, any patient's profile may be viewed with all its relevant data, such as personal information, medication and intake history, events history and all tests issued with the corresponding data registered.

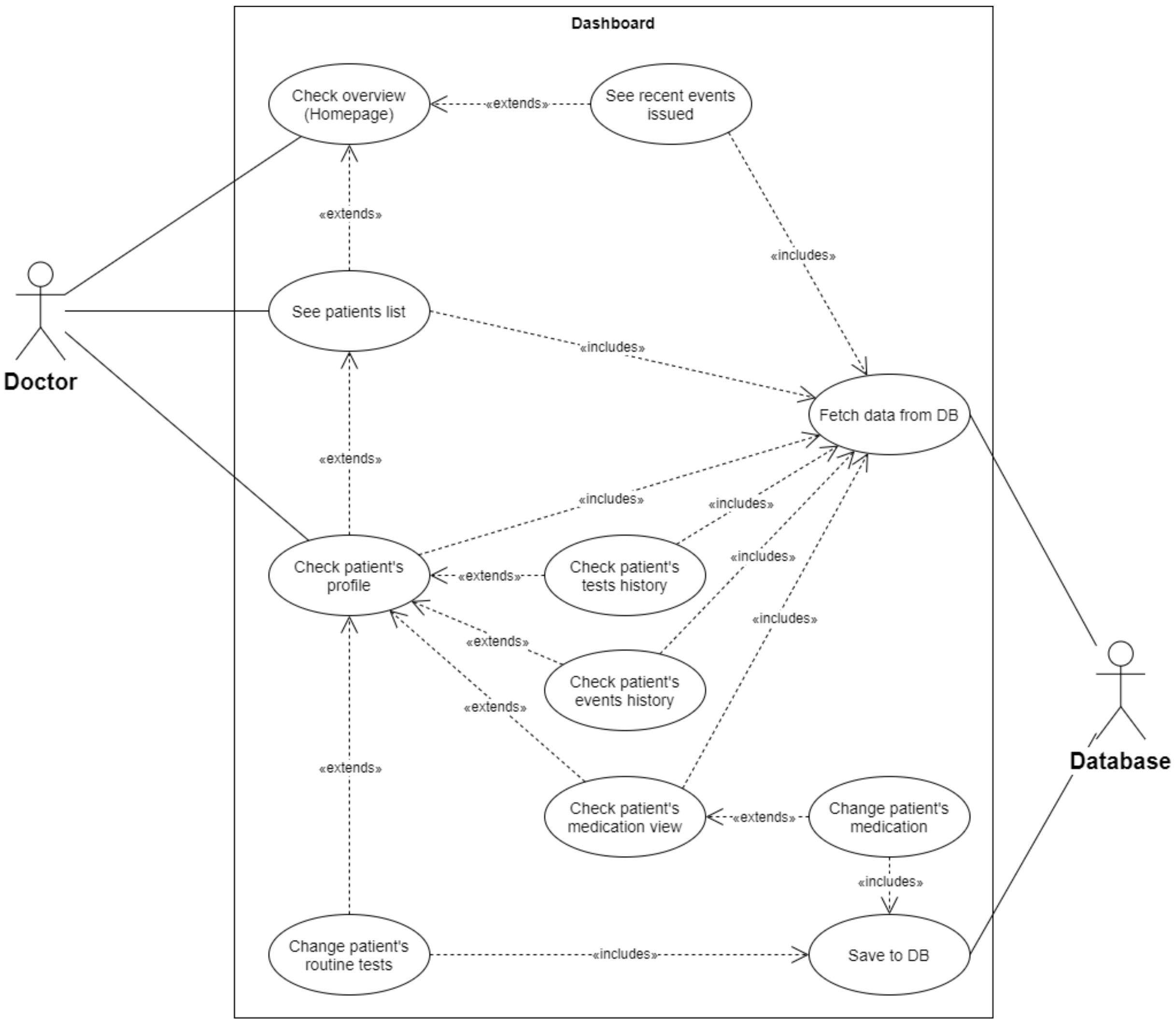


*Figure 14 – Dashboard's usecase diagram.*

Any changes to the medication or routine tests prescribed are done from the patient's medication view or profile, respectively.

The mobile app's main view is where the patient can choose one of the four main features. Report an event where the patient can inform their doctor of something relevant, being "OFF" states, symptoms aggravating, or else. The process of reporting an event usually includes running a test to provide more information to the doctor, which may require connecting to a wearable device like the iHandU. The patient may also see the list of tests available to do and select one and perform it. Every test performed is registered and saved to the database for further analysis by the doctor. Further on, the patient can also check its medication prescribed and see its profile page with all the personal information saved in the system. Additionally, the Notification Server will send notifications whenever medication is to be taken, or a routine test is due.

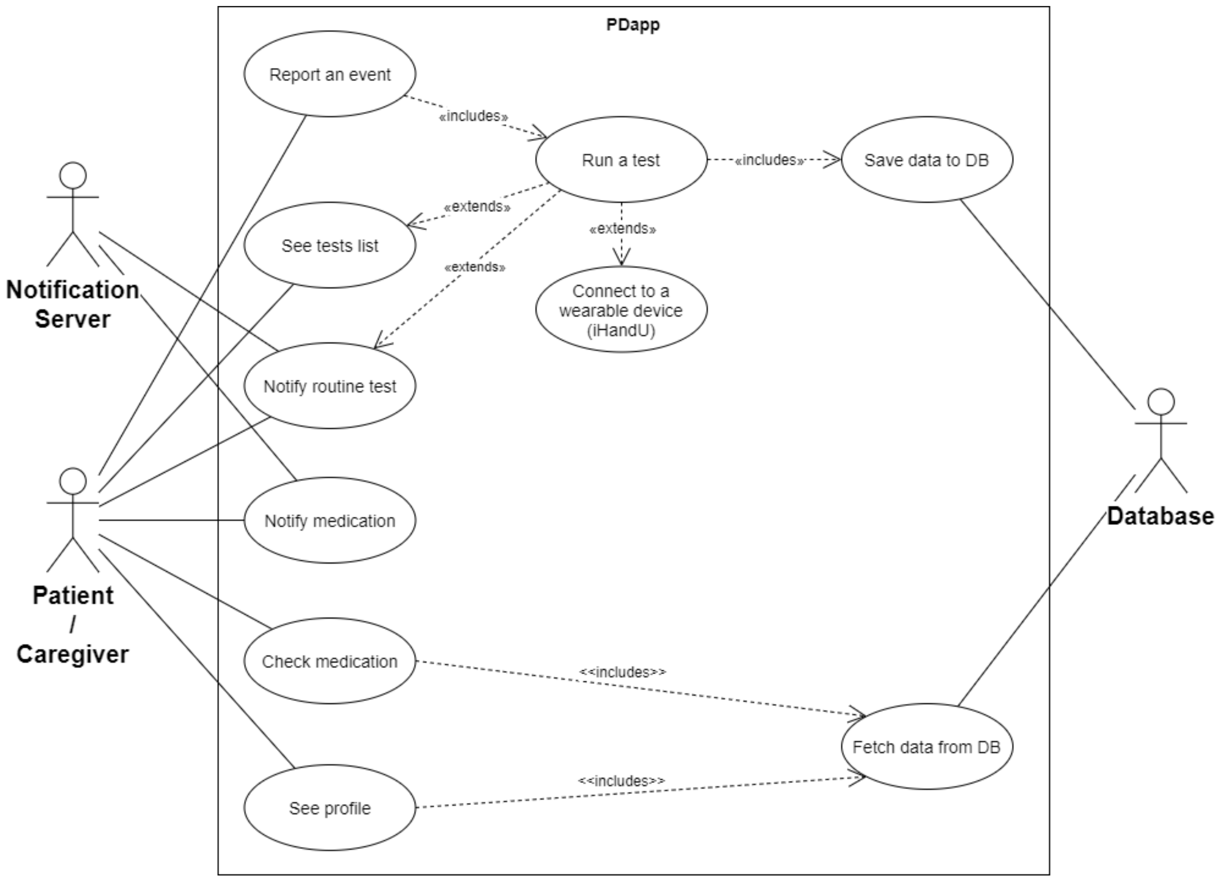


*Figure 15 – Mobile app's usecase diagram*

### 3.7.3 User stories

Based on the use cases defined earlier, the following list of user stories was created. It is separated into three major areas:

- **System / DevOps** – for all requirements related to the project setup and necessary tools for the development
- **PDapp** – for all stories related to the mobile application
- **Dashboard** – for all stories related to the doctors' dashboard

The list is also sorted by its priority in the project's scope. In general, all High priority items are necessary to build an initial prototype for early testing of the system's user experience. All Medium priority items make up the project's main scope and are mostly features that require interaction between the dashboard/mobile app and the database. Finally, the remaining Low priority items are secondary features that will not be developed until the main scope has been fulfilled.

| **System / DevOps** | | |
|---|---|---|
| **ID** | **Title** | **Priority** |
| US001 | Create and deploy database (Firestore) | High |
| US002 | Create base Mobile App | High |
| US003 | Create base website + server | High |
| US004 | Set up Notification Service (Cloud Messaging) | Medium |

*Table 3 - List of user stories related to the system setup and DevOps*

| **PDapp** | | |
|---|---|---|
| **ID** | **Title** | **Priority** |
| US101 | Login view | High |
| US102 | Home view | High |
| US103 | Check medication view | High |
| US104 | Profile view | High |
| US105 | Tests view | High |
| US106 | Perform a test view | High |
| US107 | Report an event view | High |
| US108 | Authentication | Medium |
| US109 | Fetch profile data | Medium |
| US110 | Report an event | Medium |
| US111 | Fetch medication data | Medium |
| US112 | Perform the "Wrist Rigidity" test | Medium |
| US113 | Perform the "Open/Close hands" test | Medium |
| US114 | Perform the "Timed up and go" test | Medium |
| US115 | Perform the "Postural tremor of the hands" test | Medium |
| US116 | Pair with the iHandU device | Medium |
| US117 | Send notification for routine test | Low |
| US118 | Send notification for medication | Low |

*Table 4 - List of user stories related to the mobile app (PDapp)*

| Dashboard | | |
|---|---|---|
| **ID** | **Title** | **Priority** |
| US201 | Login page | High |
| US202 | Dashboard page | High |
| US203 | Patient profile page | High |
| US204 | Medication view | High |
| US205 | Events view | High |
| US206 | Tests view | High |
| US207 | Fetch patients list | Medium |
| US208 | Fetch recent events | Medium |
| US209 | Fetch patient profile data | Medium |
| US210 | Fetch medication history & info | Medium |
| US211 | Fetch event history | Medium |
| US212 | Fetch tests history | Medium |
| US213 | Change patient's medication | Medium |
| US214 | Change patient's routine tests | Medium |

*Table 5 - List of user stories related to the doctor's dashboard*

This list is an initial representation of the work planned and was slightly updated by removing outdated requirements.

### 3.7.4 Conceptual data model

A conceptual data model of the system would look as follows:

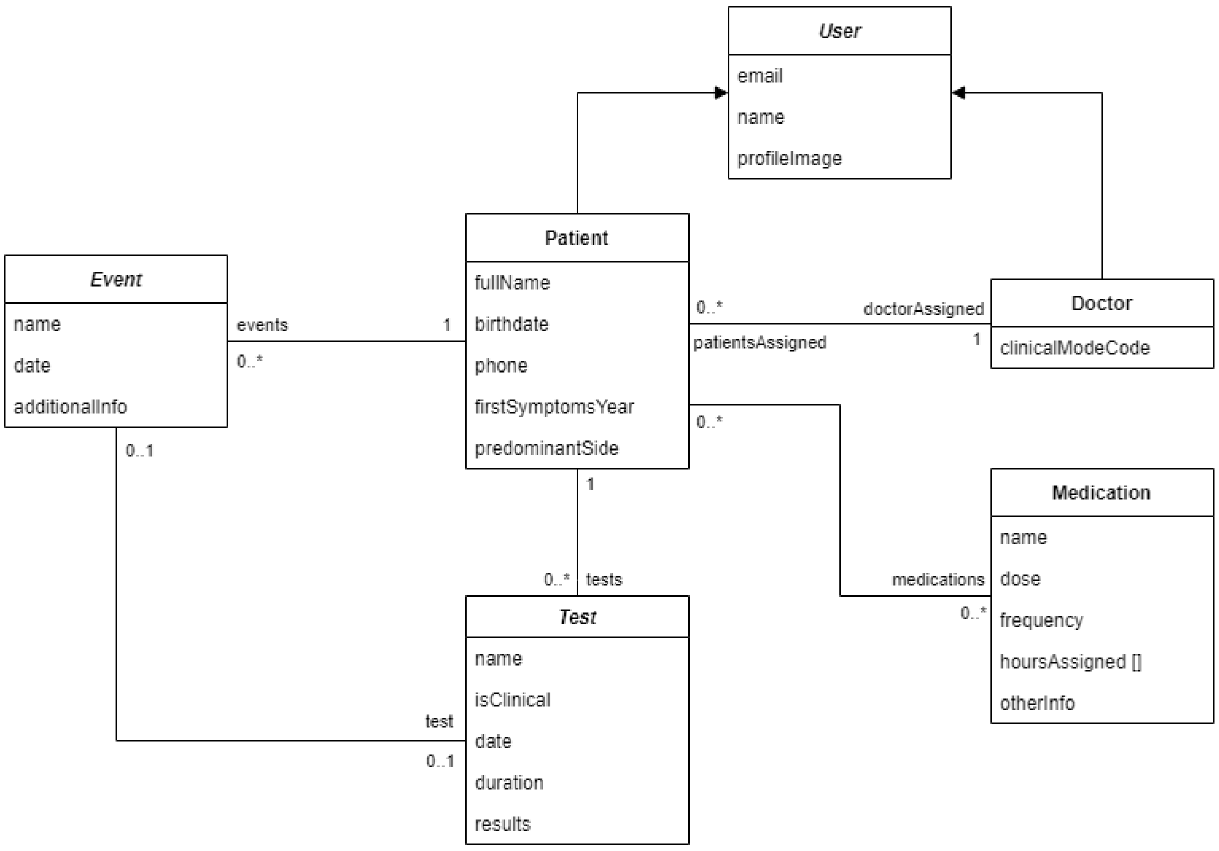


*Figure 16 - Conceptual data model (UML)*

This diagram is a high-level representation of the system data. Specific details like national health IDs or specific parameters of a test's results will likely be defined later, but are unnecessary for the current state of PDapp. This diagram's main takeaway is to plan the relations between the patient, its doctor, tests performed, medications prescribed, and events issued.

### 3.7.5 System architecture

The system's architecture will be scalable and straightforward. It will consist of 4 main components (mobile application, web platform, database and notification server), and any number of secondary components which will be hardware devices such as the *iHandU*.

The system will be developed using the Flutter framework to enable hybrid mobile development to Android and iOS and the dashboard using Flutter Web. Using the same framework for both the applications benefits the entire codebase being written in Dart. The cloud-based database will be built using Google's Cloud Firestore. Likewise, the notification server will use Google's Cloud Messaging.

Communication between the database and the leading applications will be done using the dedicated Cloud Firestore API. Similarly, the communication between the notification server and the applications will be done using the Cloud Messaging API.

As for the hardware devices, any new devices that will be implemented will have an Adapter to adjust their exposed API to the system's generic structure.

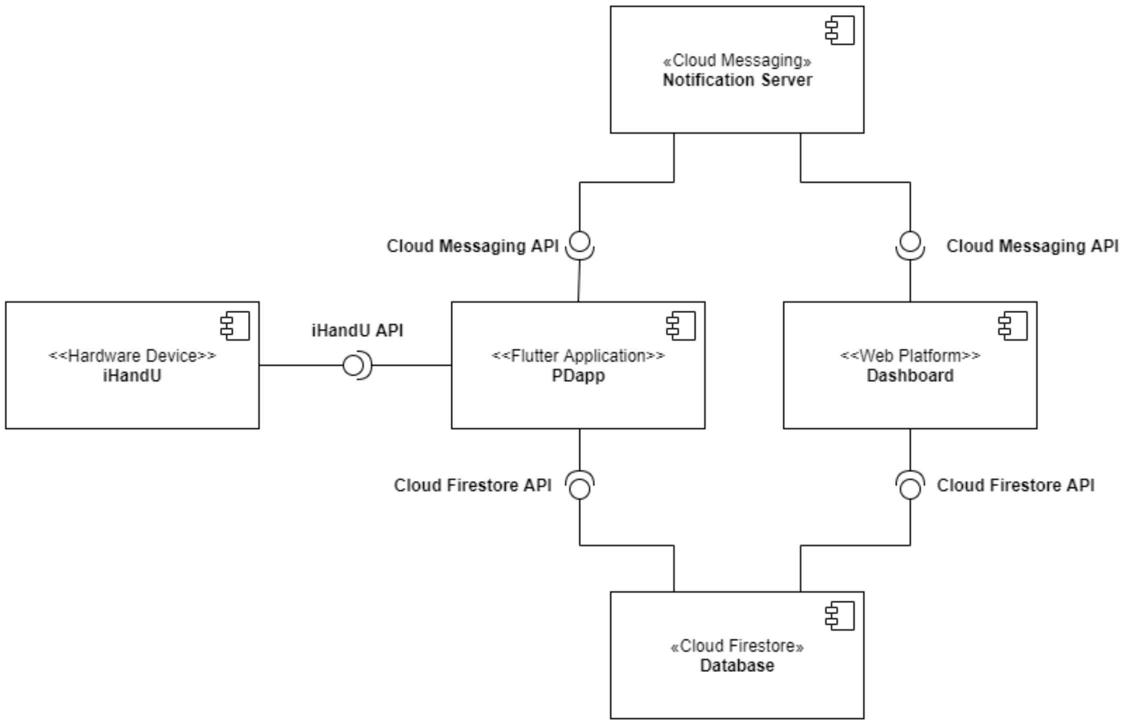


*Figure 17 – Component diagram of the system*

## 3.8 Project plan

This project's development will begin on the 14th of September 2020 and aims to be finished by the 29th of January 2021, the thesis's delivery date. This period spans roughly 20 weeks and will be divided into 10 sprints of two weeks each. The following Gantt diagram displays the allocation of tasks throughout the development process.

Sprint 1 will be dedicated to the setup of all the system's components, such as the mobile application, cloud-based database, web-platform for the dashboard, and notification server.

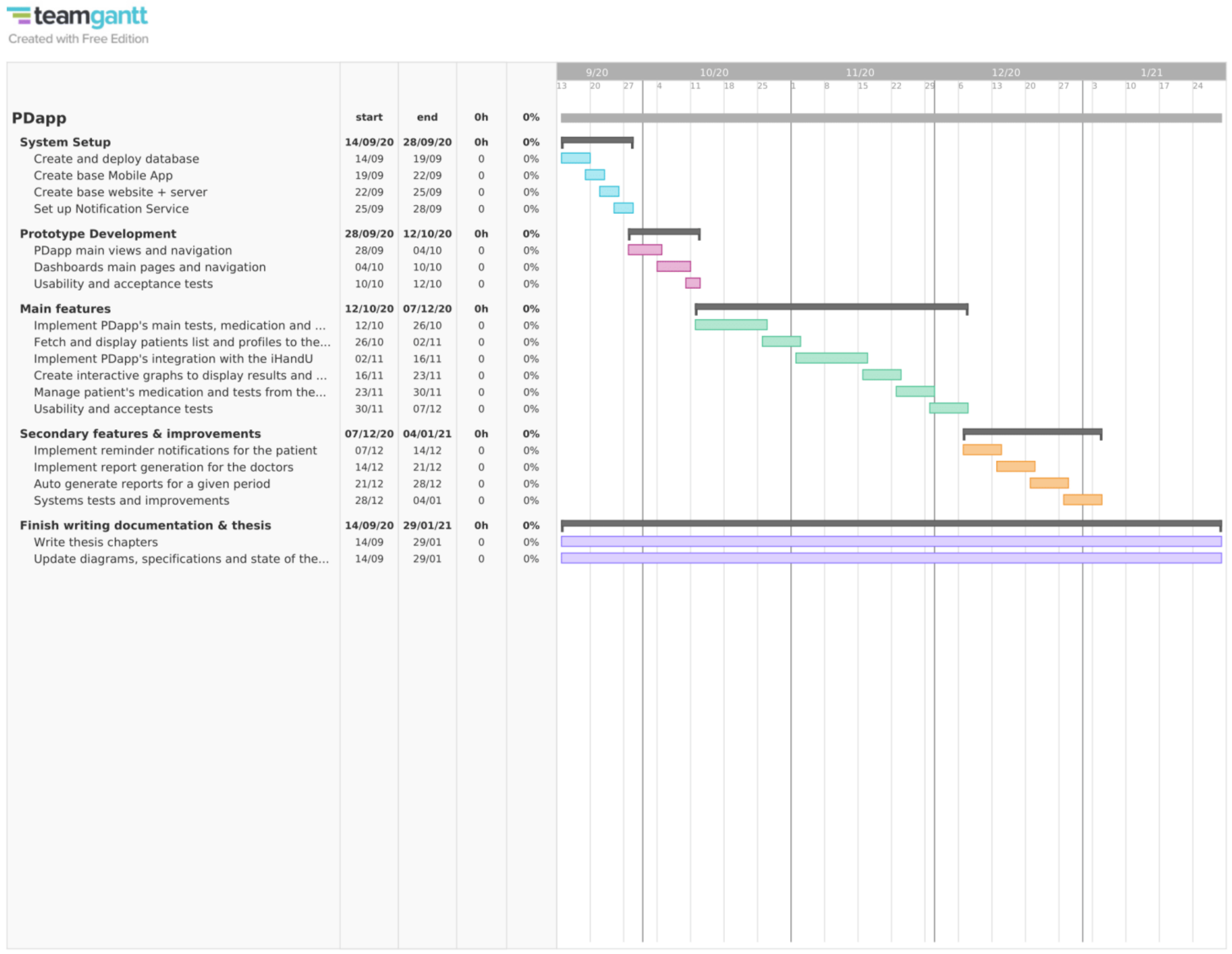
Sprint 2 is when the development begins and focuses only on creating an early prototype with the system's skeleton and navigation, to provide an image and feel as early as possible.

Sprints 3-6 are dedicated to developing the system's main features and functionalities, including the interactions between them and the implementation of the iHandU device with the mobile application.

Sprints 7-8 are the final stretch of the development phase, where necessary improvements and secondary features will be added. Plenty of testing is expected at this phase since the product should be refined and finished by the end.

Sprints 9-10 are dedicated to writing the thesis and final documentation. Although the thesis document and any extra documentation will be written and updated throughout the entire process, the last month is fully dedicated to provide all the finishing touches and handle any fine-tuning required.

Usability and acceptance tests will be performed throughout the entire development phase, specifically at the end of each phase, as specified in the Gantt diagram.



| PDapp | start | end | 0h | 0% |
|---|---|---|---|---|
| **System Setup** | **14/09/20** | **28/09/20** | **0h** | **0%** |
| Create and deploy database | 14/09 | 19/09 | 0 | 0% |
| Create base Mobile App | 19/09 | 22/09 | 0 | 0% |
| Create base website + server | 22/09 | 25/09 | 0 | 0% |
| Set up Notification Service | 25/09 | 28/09 | 0 | 0% |
| **Prototype Development** | **28/09/20** | **12/10/20** | **0h** | **0%** |
| PDapp main views and navigation | 28/09 | 04/10 | 0 | 0% |
| Dashboards main pages and navigation | 04/10 | 10/10 | 0 | 0% |
| Usability and acceptance tests | 10/10 | 12/10 | 0 | 0% |
| **Main features** | **12/10/20** | **07/12/20** | **0h** | **0%** |
| Implement PDapp's main tests, medication and ... | 12/10 | 26/10 | 0 | 0% |
| Fetch and display patients list and profiles to the... | 26/10 | 02/11 | 0 | 0% |
| Implement PDapp's integration with the iHandU | 02/11 | 16/11 | 0 | 0% |
| Create interactive graphs to display results and ... | 16/11 | 23/11 | 0 | 0% |
| Manage patient's medication and tests from the... | 23/11 | 30/11 | 0 | 0% |
| Usability and acceptance tests | 30/11 | 07/12 | 0 | 0% |
| **Secondary features & improvements** | **07/12/20** | **04/01/21** | **0h** | **0%** |
| Implement reminder notifications for the patient | 07/12 | 14/12 | 0 | 0% |
| Implement report generation for the doctors | 14/12 | 21/12 | 0 | 0% |
| Auto generate reports for a given period | 21/12 | 28/12 | 0 | 0% |
| Systems tests and improvements | 28/12 | 04/01 | 0 | 0% |
| **Finish writing documentation & thesis** | **14/09/20** | **29/01/21** | **0h** | **0%** |
| Write thesis chapters | 14/09 | 29/01 | 0 | 0% |
| Update diagrams, specifications and state of the... | 14/09 | 29/01 | 0 | 0% |

*Figure 18 - Gantt diagram of the project's schedule*

# Chapter 4

# Framework & Guidelines

In this section, the framework, code structure, and overall guidelines applied during the PDapp system development are defined. It should give a clear overview of what to expect. As well as make the development philosophy behind each decision clear.

## 4.1 Flutter

This project was entirely developed using the Flutter framework [21] by Google. This rising hybrid development framework enabled the mobile application to be available for both Android and iOS. It also allowed for both components (mobile and web) to be developed under a single code base, making the overall code easier to understand and structure uniformly.

Initially, building an Android native application was highly considered. The iHandU's software was developed in Android, and staying with an Android codebase would help re-use the existing code. Nevertheless, Flutter could help with two features that Android could not.

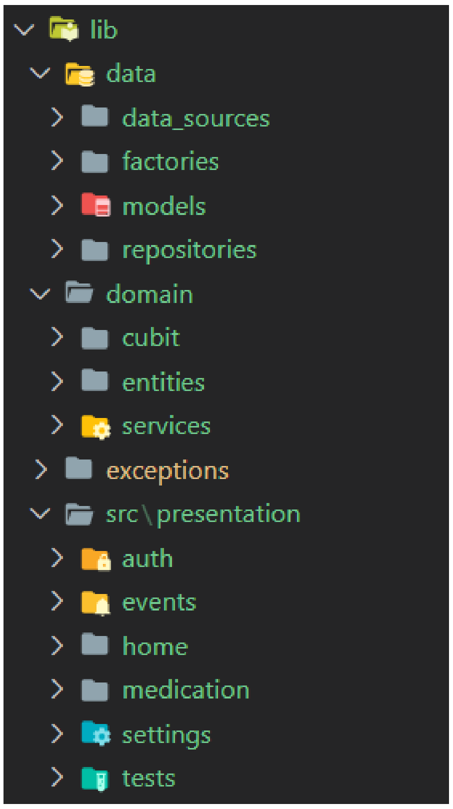
Firstly, hybrid development allows the mobile app to be available in both Android and iOS platforms. Not only does this spare the costs of developing a nearly identical application for iOS later on, but it also allows new changes to be easily propagated to both systems. This way, PDapp's mobile app can be treated as the same, independently of the platform it is used in.

The second distinguishing feature is the recent addition of Flutter Web [22]. This feature allows the development of web applications using the Flutter framework. This way, even the dashboard can be developed using the same framework and codebase. Keeping the entire system under the same codebase contributes to a better understanding of the system. The onboarding process for any new developers is also much easier than learning three frameworks and languages.

Other hybrid platforms, such as React Native [23] and Ionic [24] were also considered. However, Flutter is renowned for its native-like performance, which is one of the biggest concerns surrounding hybrid development frameworks. Paired with the support for web applications provided by Flutter Web, Flutter was chosen as PDapp's framework.

## 4.2 Clean architecture

Both modules (mobile and web) were developed as separate packages, as specified in the system's architecture (section 3.7.5). The code structure used in each of these packages (*PDapp* and *Dashboard*) follows various aspects proposed by Robert C. Martin (*"Uncle Bob"*) in his Clean Architecture [25]. *Uncle Bob*'s proposed architecture allows a clear separation of concerns that makes development easy, decoupled, and simple to understand from the outside. This project's adaptation can be seen in the figure below.



```
lib
  data
    data_sources
    factories
    models
    repositories
  domain
    cubit
    entities
    services
  exceptions
  src\presentation
    auth
    events
    home
    medication
    settings
    tests
```

*Figure 19 – PDapp's package code structure*

Essentially, the code is separated into three main layers:

- Presentation – all code concerning the UI - layouts and widgets (Flutter's UI components)
- Domain – business logic and concepts – the logic behind use cases and user interactions, and core concepts such a *Test* or a *Patient* defined as entities
- Data – data fetching and handling – database interactions and converting raw data into domain objects (entities)

The *presentation* folder is in a separate *src* folder to help the documentation generator ignore the UI code (a limitation of the *dartdoc* package [26]). Otherwise, the documentation is heavily inflated with unnecessary code. In a general scenario, it should be right next to the *data* and *domain* folders.

This clear separation of concerns helps decouple the various areas of the application and allows changes to be made with little to no rippling effects. It also helps any new developers understand where each component is defined and keeps complexity to a minimum.

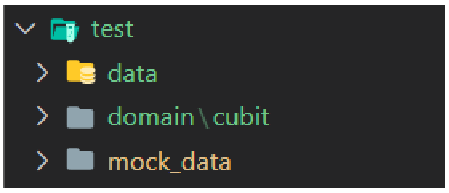
## 4.3 Test-driven development

As explained by the notorious Martin Fowler: “Test-Driven Development (TDD) is a technique for building software that guides software development by writing tests” [27]. It follows a few steps that require tests to be implemented before the production code. This approach helps developers keep the code tested and well structured.

TDD is a highly respected development approach in the developer community and is supported by respected personalities such as Martin Fowler and Robert C. Martin (*Uncle Bob*, as referenced before).

Most of the development followed the guidelines of TDD. Unit tests were created for all areas of the code except the UI. Widget testing (UI tests in Flutter) seemed to provide little value compared to the time concerns of an already ambitious project and were neglected due to that reason.

All tests can be found in each package's *test* folder and follow the same structure as defined previously in the project’s architecture (figure 20).



```
test
  data
  domain\cubit
  mock_data
```

*Figure 20 – Test’s code structure*

The *presentation* folder is not present since widget tests (UI tests) were not done, as mentioned before.

Chapter 5

# Mobile App

The mobile companion app is the defining part of the overall system. As mentioned previously, it was developed with the Flutter framework and has support for Android and iOS. Due to time constraints, the iOS app could not be tested, but with access to a macOS virtual machine, it should be simple to do so in the future.

In this chapter, details are presented for each functionality and the implementation choices behind them. Information such as the UI, UX considerations, services integrated, and the interaction between components will be explained in the following sections.

## 5.1 Authentication

Authentication (feature 1 from section 3.5) is always an important aspect when dealing with a closed system like PDapp. Patients cannot create accounts independently, as they should all be assigned to a doctor that should introduce them to the service and create their account on request.

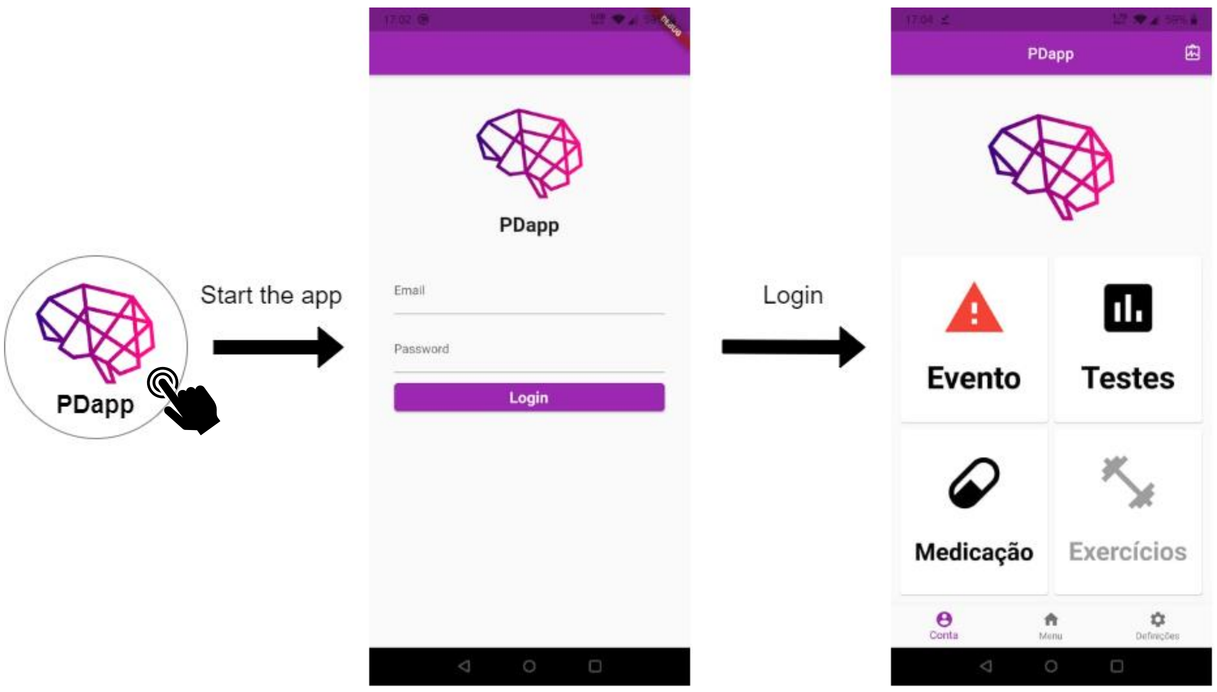


*Figure 21 - Authentication flow*

## 5.1.1 Auth Page

Upon opening the app, the Auth Page will appear. Only an email/password pair is accepted for now as a valid login. Upon submitting the credentials, the app will interact with the *AuthService*, which currently uses Firebase Auth to handle authentication for the entire system. If there is an error with the credentials an error label will be shown, and the patient may try to login again. If everything goes well, the app is redirected to the homepage.

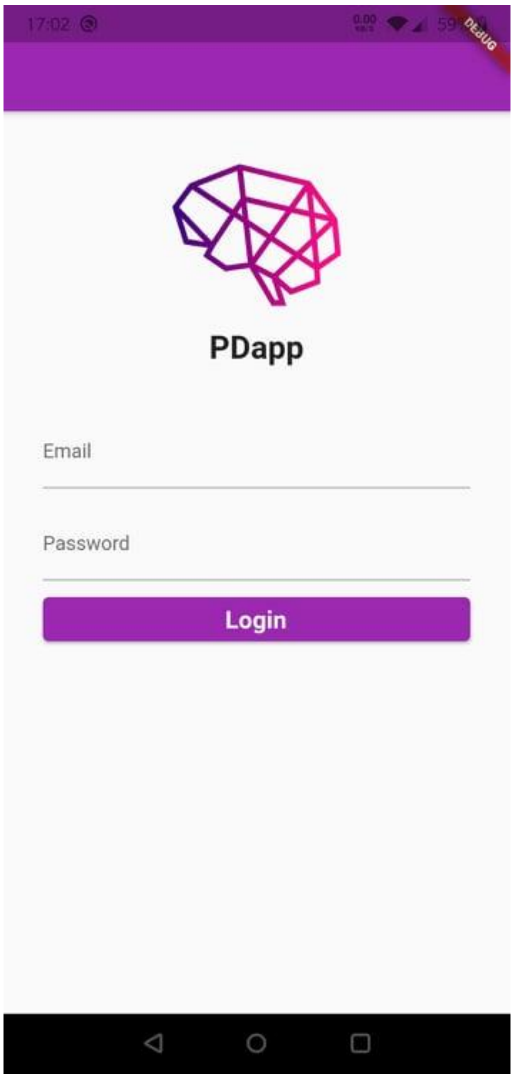


Figure 22 - Authentication page

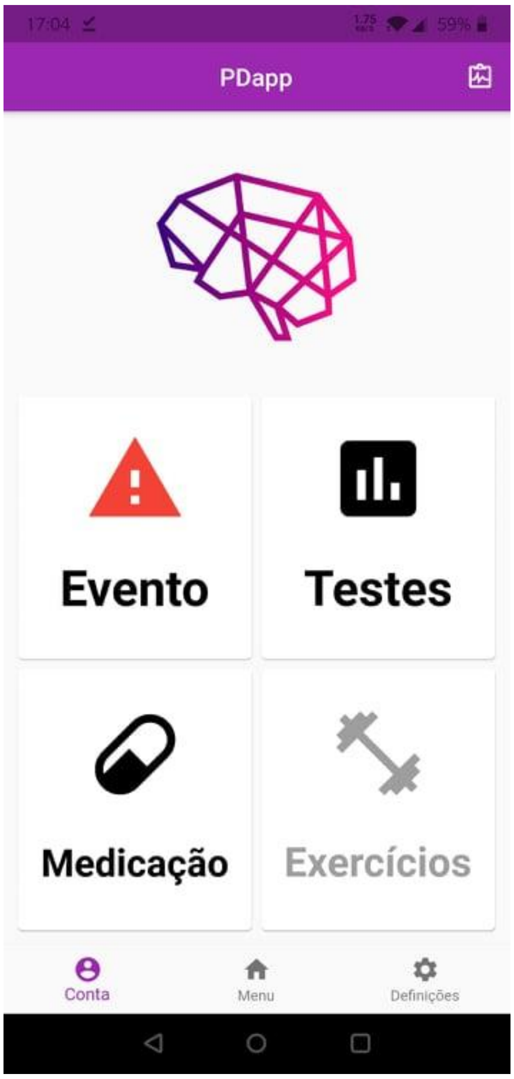
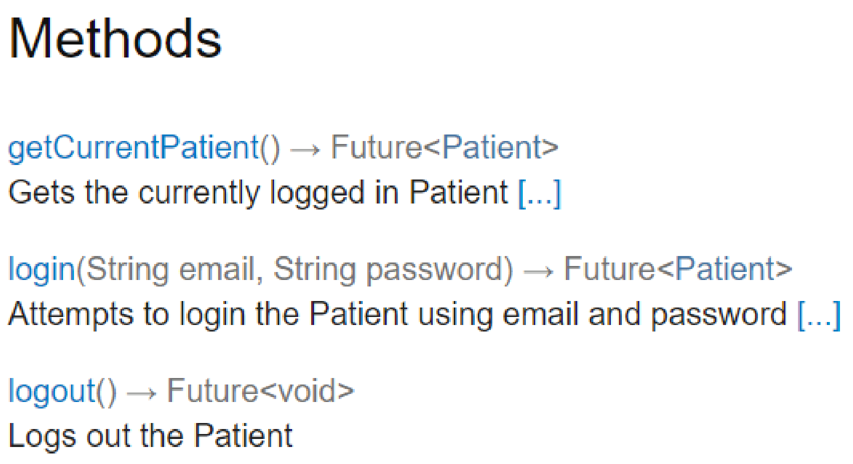


Figure 23 - Homepage

## 5.1.2 Auth Repository

The *AuthRepository* is responsible for handling the authentication functions using *AuthService* and retrieving Patient data when necessary. The documentation for its interface is described in its docs (figure 13).


Methods

getCurrentPatient() → Future<Patient>
Gets the currently logged in Patient [...]

login(String email, String password) → Future<Patient>
Attempts to login the Patient using email and password [...]

logout() → Future<void>
Logs out the Patient

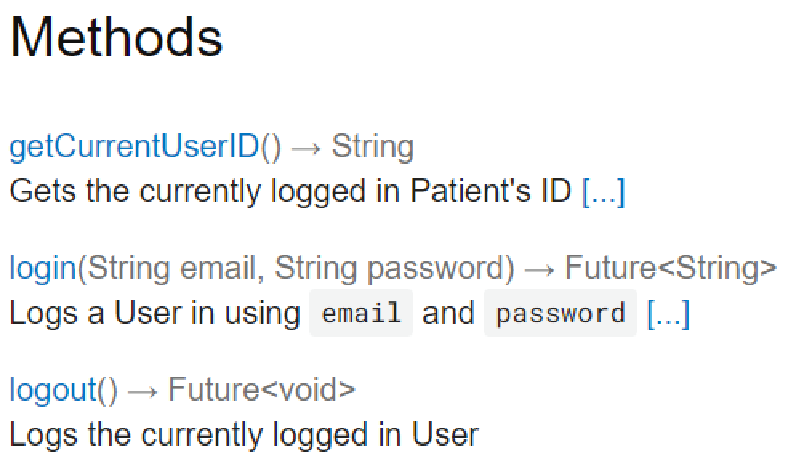
*Figure 24 - AuthRepository documentation*

It generally takes care of logging a user in or out with the *AuthService* and uses the provided PatientID to fetch the Patient data from the *RemoteDataSource* (Cloud Firestore).

### 5.1.3 Auth Service & Firebase Auth

The integration with Firebase Auth was done via the *firebase_auth* package for Flutter. An interface *AuthService* was defined to decouple the implementation and allow for future changes to the authentication process. Which is already expected to happen since using a public cloud-based solution like Firebase is very inadequate for handling patient personal data.

The *AuthService* is a simple interface with the following methods, as described in the documentation.


Methods

getCurrentUserID() → String
Gets the currently logged in Patient's ID [...]

login(String email, String password) → Future<String>
Logs a User in using `email` and `password` [...]

logout() → Future<void>
Logs the currently logged in User

*Figure 25 - AuthService documentation*

The current implementation of this service uses the FirebaseAuth instance (provided by the *firebase_auth* package) to login with an email/password, logout and retrieve the ID of the currently authenticated user (null if unauthenticated).

## 5.2 Events

Reporting events (feature 3 from section 3.5) is a valuable feature of the system, due to the unique interaction it provides between patients and doctors. In PDapp, events are created by the Patient upon request, and after completing a brief test, the results are submitted and displayed in the doctor's dashboard for further analysis. A typical scenario of reporting an "Off" state can be seen in the diagram below:

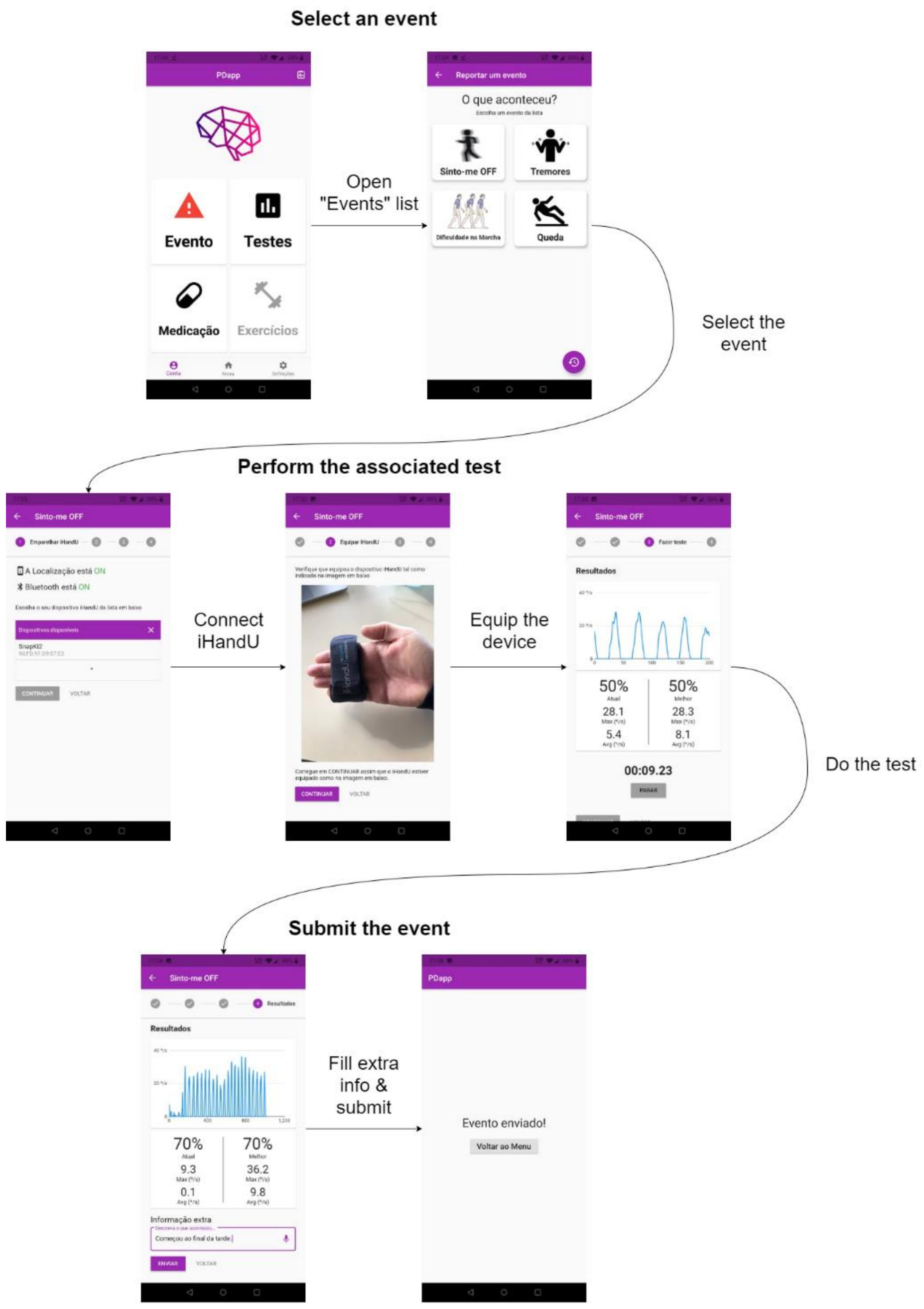


*Figure 26 – Report an event flow for an Off-state event*

Reporting an event follows three important steps:

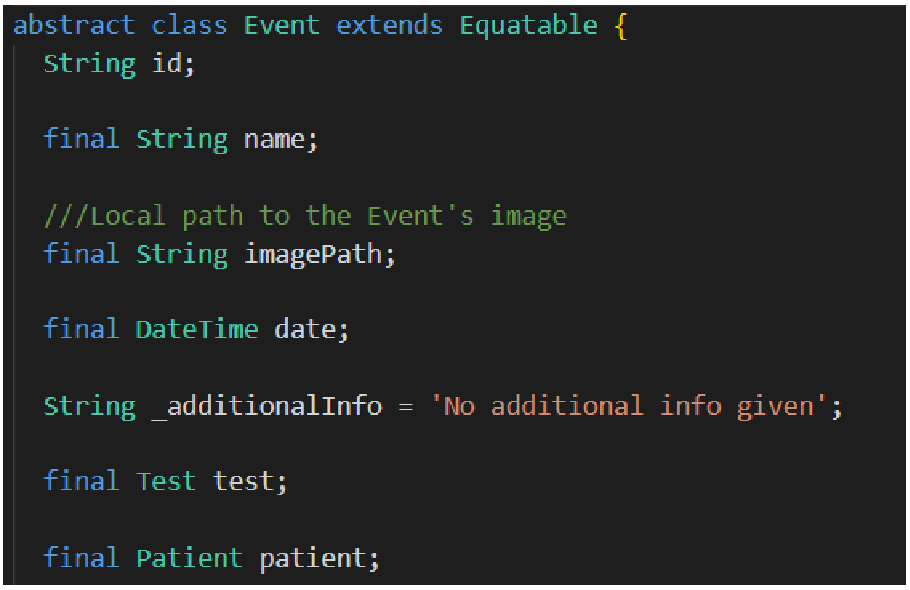
- *Selecting the event* – from the list of available events
- *Performing the associated test* – each event has an associated test which can be performed; each test follows four steps: setup of the device, equipping the device, running the test, and submitting the final results
- *Submitting the event* – after checking the results of the test, the patient may add additional information that can be valuable to the doctor; once ready, the event can be submitted

The entire process and implementation of the steps are detailed in section 5.2.2.

### 5.2.1 Data structure

An Event is an important entity in the system. Therefore, it follows an abstract model both in the mobile app as in the dashboard. Apart from the evident id, name, image, and date properties, each event must be assigned:

- a specific *Test*
- the *Patient* that owns the event
- any additional information that the patient deems necessary to submit.



```
abstract class Event extends Equatable {
  String id;

  final String name;

  ///Local path to the Event's image
  final String imagePath;

  final DateTime date;

  String _additionalInfo = 'No additional info given';

  final Test test;

  final Patient patient;
```

*Figure 27 - Event class declaration*

### 5.2.2 Reporting an event

Reporting an event starts at the homepage by selecting the “Events” card, which opens the events’ menu that lists all the currently implemented events. The patient may now choose the event that better describes the current situation and perform the associated test before submitting.

The following table holds the currently implemented events in the app. The associated tests are also indicated.

| **Event** | **Description** | **Associated Test** |
|---|---|---|
| “Off” State | For when the patient is feeling an “Off” state and wishes to inform the doctor | Open/Close hands |
| Tremors | If the patient is feeling a significant increase in resting tremors | Resting tremor |
| Difficulty walking | Gait issues and walking difficulties. | Timed Up and Go |
| Fall | If the patient suffered a fall recently. | Timed Up and Go |

*Table 6 – Implemented events table*

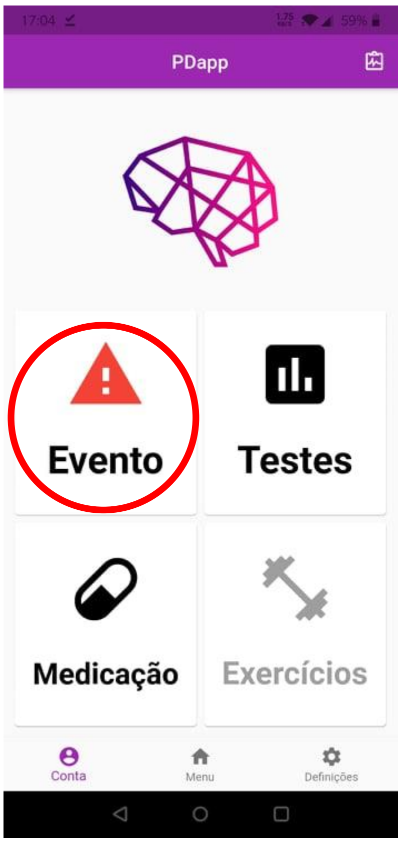


*Figure 28 - Homepage's "Events" option*

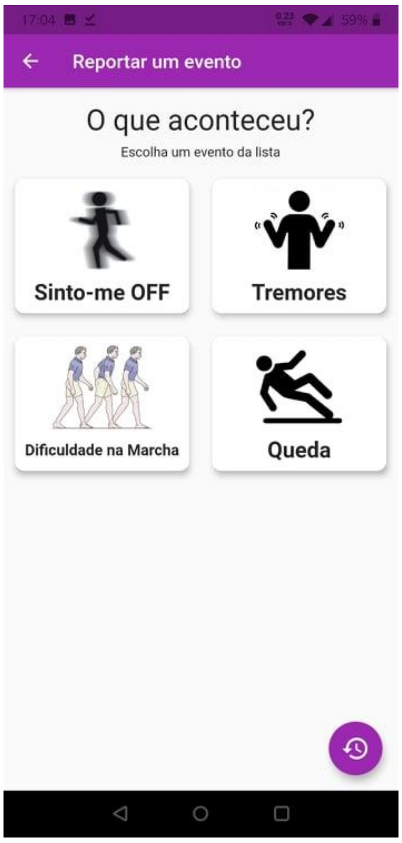


*Figure 29 - Events list*

Once an event has been chosen, the app redirects to the associated test's step-by-step guide. This guide contains four steps:

A. *Setup of the device* – searching and connecting to the wearable device
B. *Equip the device correctly* – step with instructions of how to properly equip the device to the specific tests
C. *Perform the test* – instructions on how to perform the test correctly followed by the test's timer and live results
D. *Final results* – view for displaying the final results and submitting

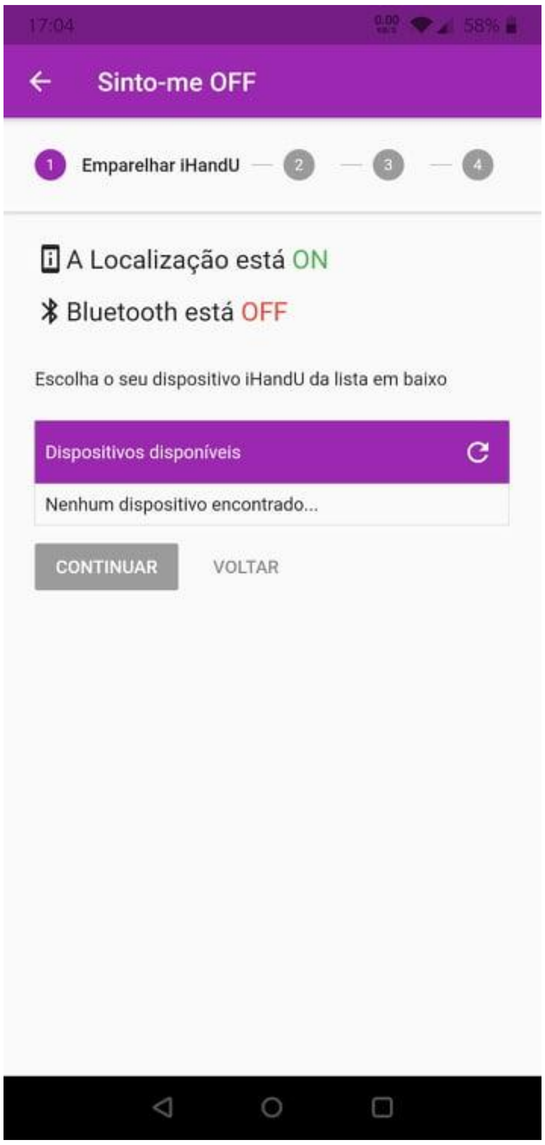
During the following steps, the *BLE Service* and *iHandU Service* will be used in the background to interact with the iHandU device via Bluetooth. These specific interactions will be later explained in detail in section 5.5 dedicated to the iHandU integration.

### A. Setup the device

This step is dedicated to discovering and connecting to the test's device (currently only the iHandU).

A list of the requirements (and their status) is displayed at the top. In the case of the iHandU, which requires a BLE connection, the device's Bluetooth must be turned on, and the Location permissions must be granted.

Once all the requirements are met, the search for devices will begin, and the patient may select their device from the list. After successfully connecting to the iHandU device, a green tick should appear, and the patient may continue to the next step.


17:04 58%

Sinto-me OFF

1 Emparelhar iHandU — 2 — 3 — 4

A Localização está ON

Bluetooth está OFF

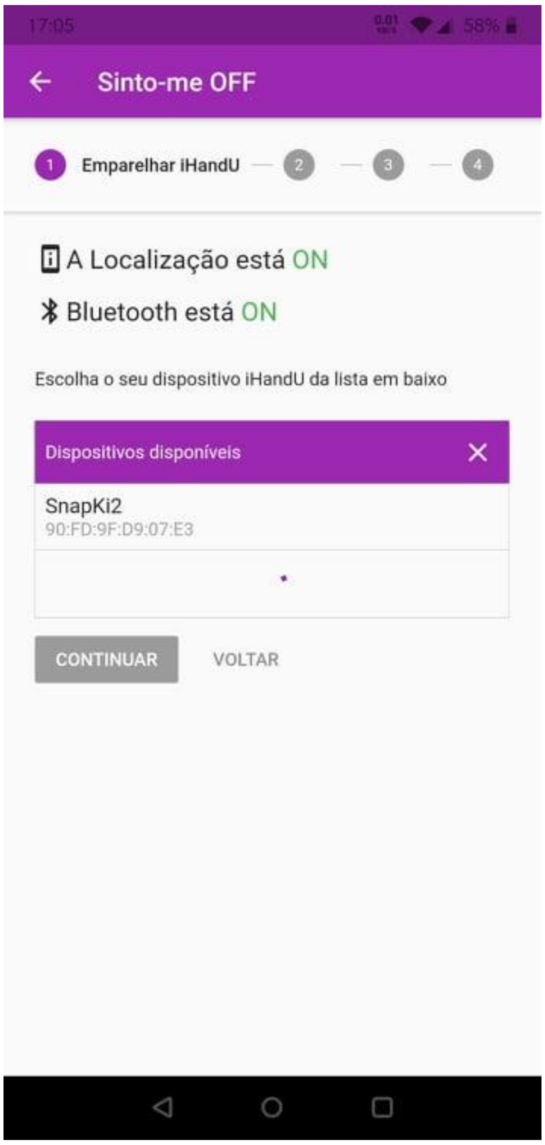
Escolha o seu dispositivo iHandU da lista em baixo

Dispositivos disponíveis

Nenhum dispositivo encontrado...

CONTINUAR VOLTAR

*Figure 30 - Missing permissions view*


17:05 58%

Sinto-me OFF

1 Emparelhar iHandU — 2 — 3 — 4

A Localização está ON

Bluetooth está ON

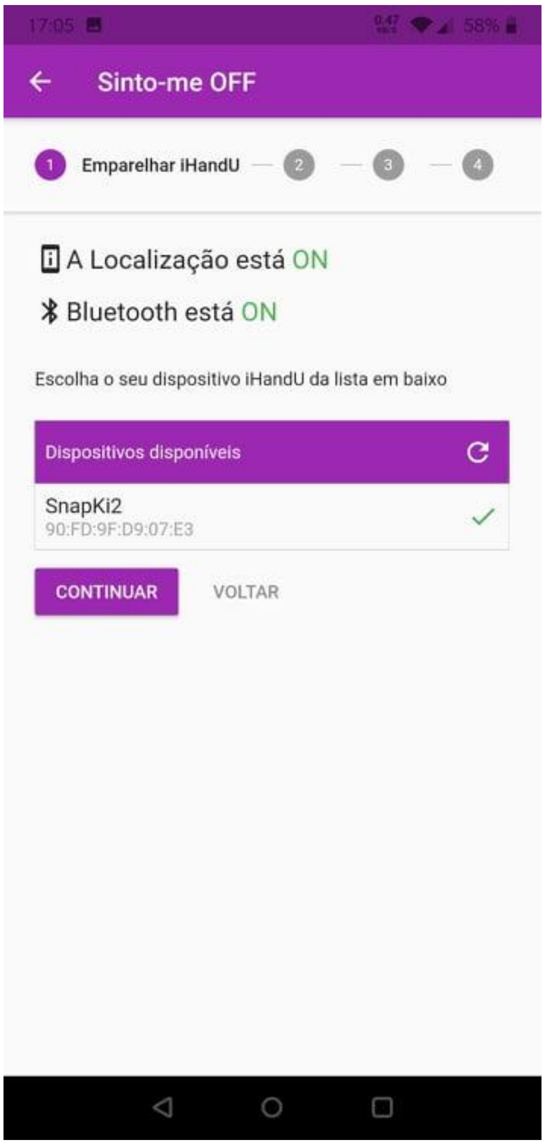
Escolha o seu dispositivo iHandU da lista em baixo

Dispositivos disponíveis

SnapKi2
90:FD:9F:D9:07:E3

CONTINUAR VOLTAR

*Figure 31 - Searching for devices view*



A Localização está ON

Bluetooth está ON

Escolha o seu dispositivo iHandU da lista em baixo

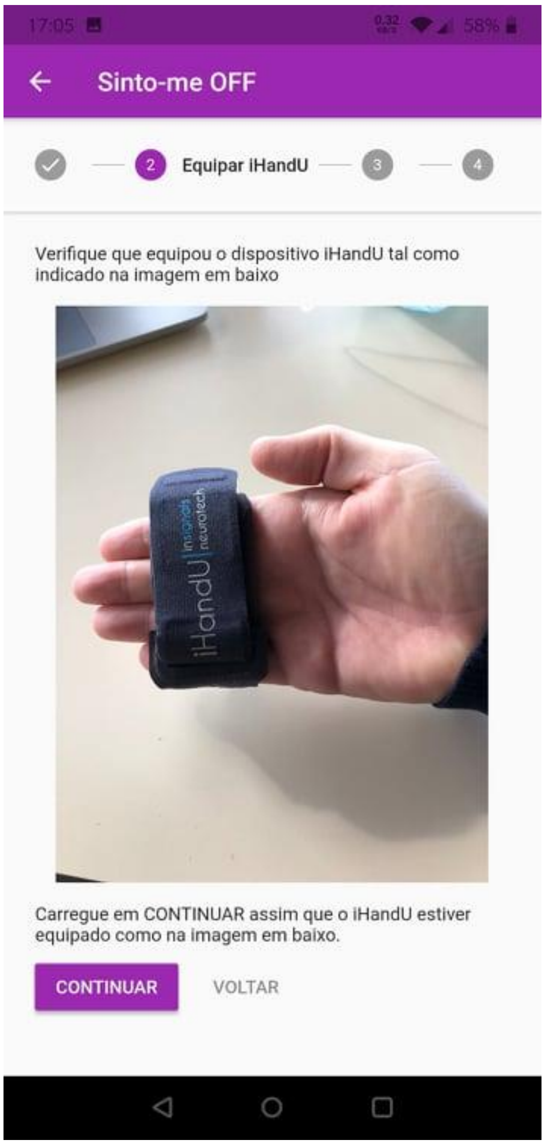
| Dispositivos disponíveis |
|---|
| SnapKi2 90:FD:9F:D9:07:E3 ✓ |

*Figure 32 - Device connected view*

## B. Equip the device correctly

On the Equip step, instructions are shown on how to equip the device for the test correctly. Text descriptions, images, and videos may be used to describe the equipping process.

Once the patient is ready to advance, it may continue to the next step.



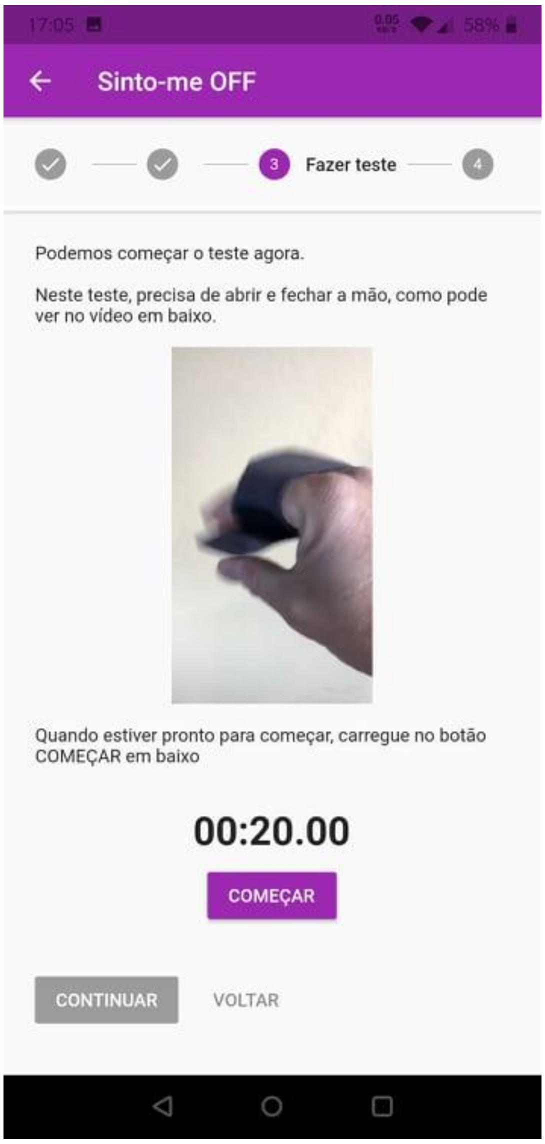
Verifique que equipou o dispositivo iHandU tal como indicado na imagem em baixo

Carregue em CONTINUAR assim que o iHandU estiver equipado como na imagem em baixo.

*Figure 33 - Equip screen for an "Off State" event*

## C. Perform the test

This step is where the actual test is performed. It begins with detailed instructions about what the test requires. Like with the Equip step, it too uses text, images and video if necessary.



Podemos começar o teste agora.

Neste teste, precisa de abrir e fechar a mão, como pode ver no vídeo em baixo.

Quando estiver pronto para começar, carregue no botão COMEÇAR em baixo

00:20.00

*Figure 34 - Test instructions view*

Some tests have a set timer that automatically completes the test, while others require the patient to stop the timer. This division depends on if the test allows for user interaction or not. In the case of a "Stretch arms" test, the patient must keep both arms stretched for the duration of the exercise, and therefore cannot stop the timer by himself.

Once the patient is ready and starts the timer, the test begins, and a graph of the real-time results is shown. Below the graph, some calculated statistics are updated throughout the test so any spectators can evaluate the exercise accordingly (like in a clinical environment). If any mistake happens throughout the test, it can be cancelled and restarted using the controls.

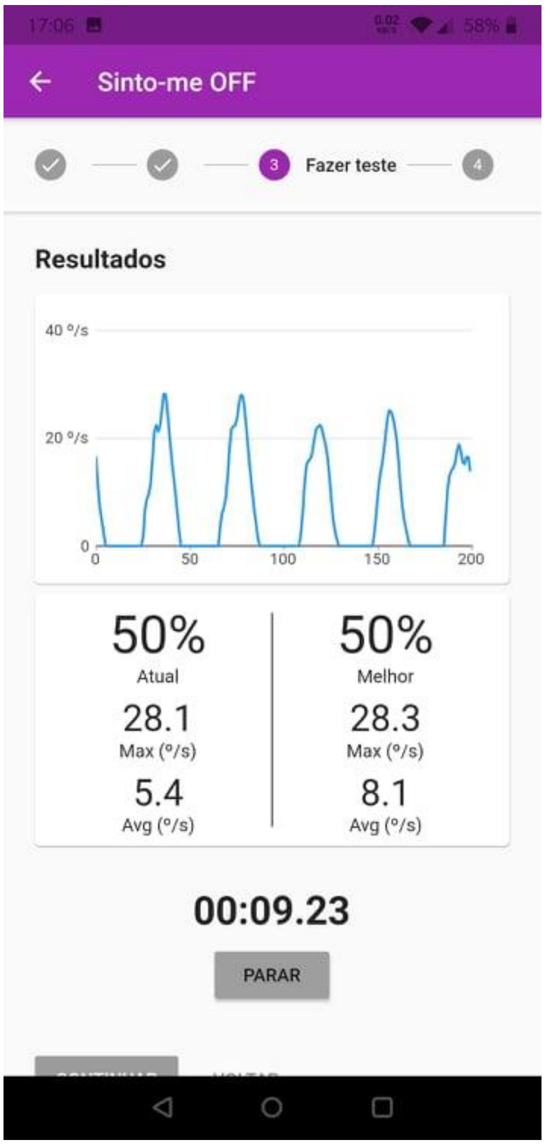
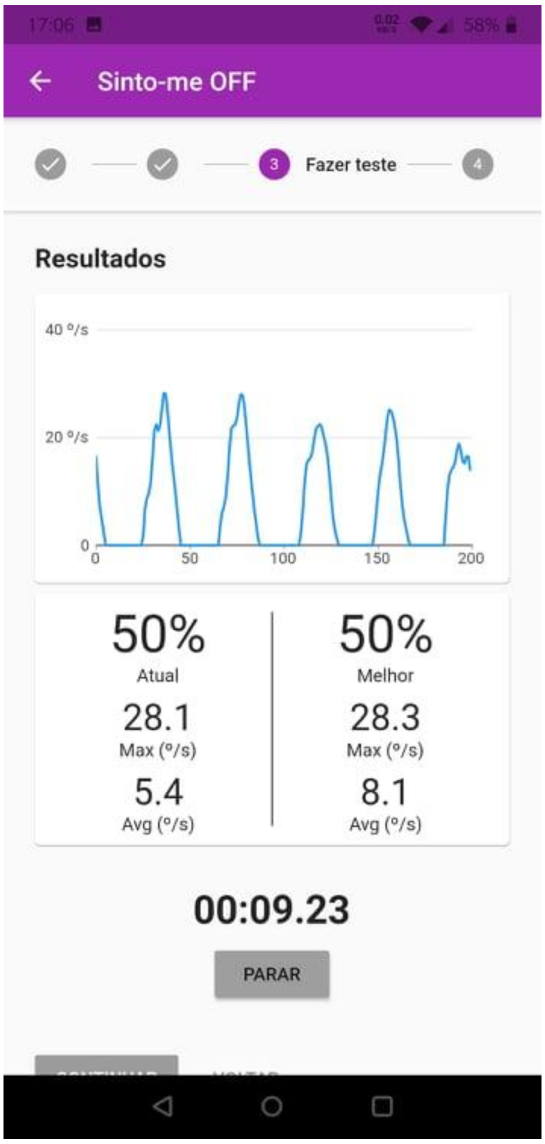


*Figure 35 - Test running view*

After the timer finishes or is stopped, the test is completed. The patient may now continue to the final results.



50%
Atual
28.1
Max (°/s)
5.4
Avg (°/s)

50%
Melhor
28.3
Max (°/s)
8.1
Avg (°/s)

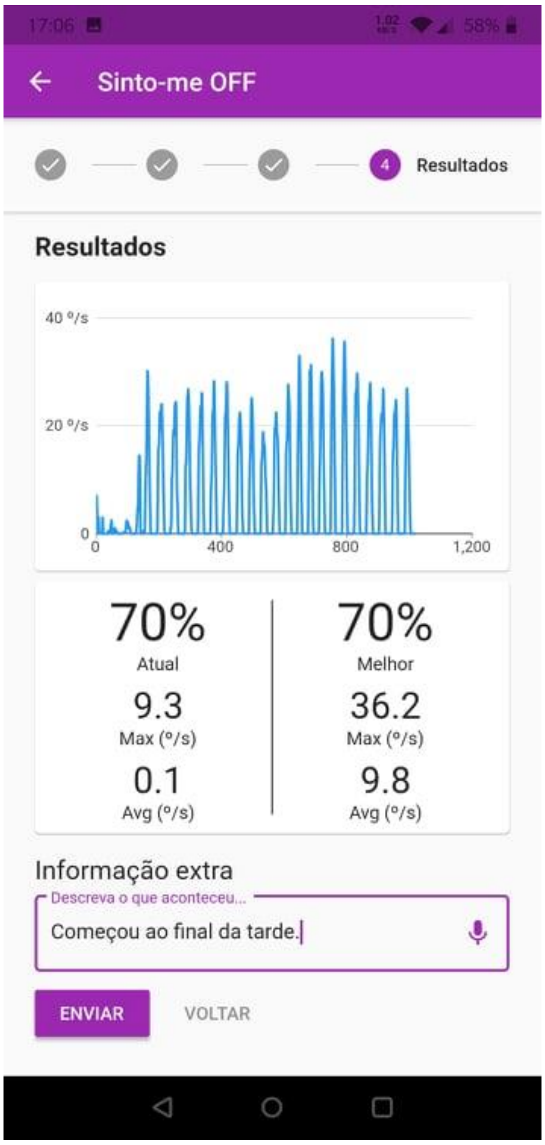
00:09.23

*Figure 36 - Test finished view*

## D. Final results

In this final step, the results graph and statistics are revealed and can be reviewed. Any additional information that the patient deems necessary can be added to the event via the text box in the end.

After reviewing everything, the patient may submit the event through the "Submit" button. If successful, a simple screen will be shown asserting the event was sent, and a button can be used to go back to the homepage.

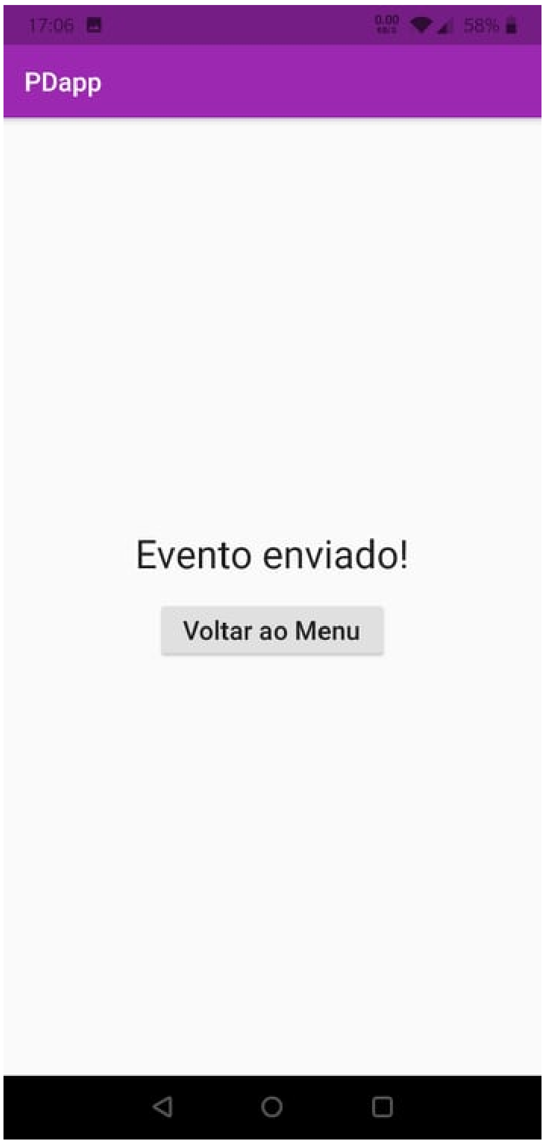


70% Atual | 70% Melhor
9.3 Max (°/s) | 36.2 Max (°/s)
0.1 Avg (°/s) | 9.8 Avg (°/s)

Informação extra

Começou ao final da tarde.

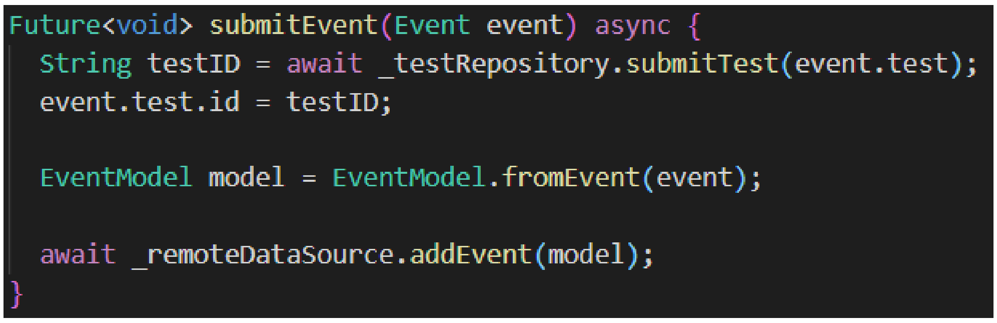
*Figure 37 – Event's final results view*



Evento enviado!

*Figure 38 – Event submitted view*

The app uses the *EventRepository* and the *RemoteDataSource* to upload the new event to the database. The *EventRepository* first submits the test associated (using the *TestRepository*) and afterwards submits the event itself. This order is due to the event requiring the associated test's ID in its data, which is only generated after submitting the new test.

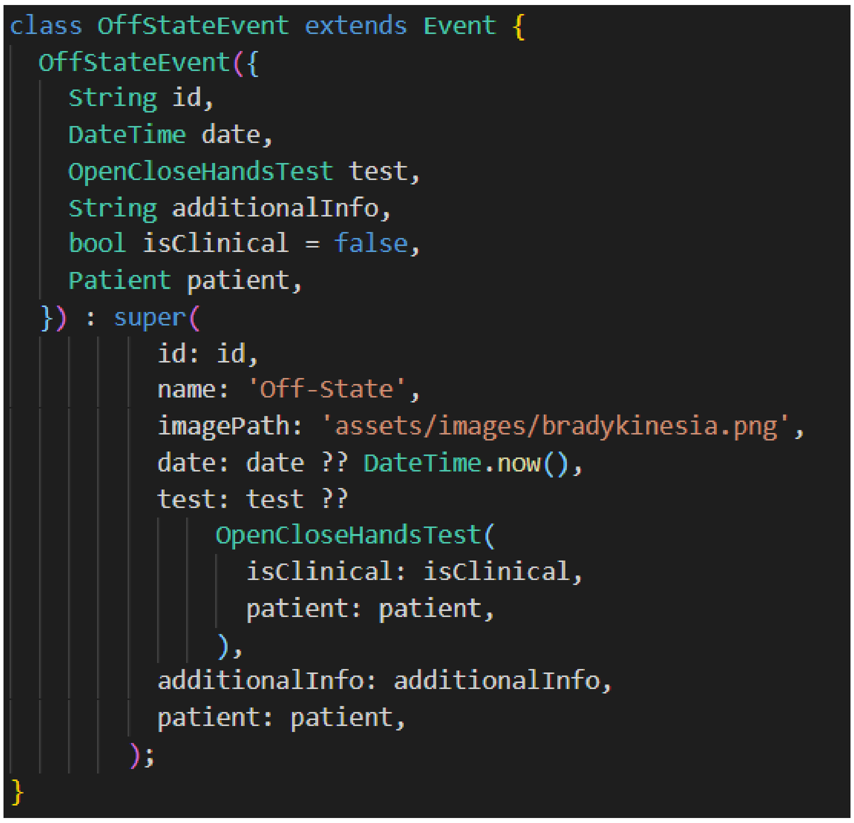


```dart
Future<void> submitEvent(Event event) async {
  String testID = await _testRepository.submitTest(event.test);
  event.test.id = testID;

  EventModel model = EventModel.fromEvent(event);

  await _remoteDataSource.addEvent(model);
}
```

*Figure 39 - EventRepository's submitEvent implementation*

### 5.2.3 Adding new events

Keeping the process of adding new types of events simple is paramount. Since events follow the abstract structure specified on section 5.2.1, adding a new event type only requires a subclass of the *Event* class and the specific associations for each property. Here is an example with the *OffStateEvent* class:



```dart
class OffStateEvent extends Event {
  OffStateEvent({
    String id,
    DateTime date,
    OpenCloseHandsTest test,
    String additionalInfo,
    bool isClinical = false,
    Patient patient,
  }) : super(
            id: id,
            name: 'Off-State',
            imagePath: 'assets/images/bradykinesia.png',
            date: date ?? DateTime.now(),
            test: test ??
                OpenCloseHandsTest(
                  isClinical: isClinical,
                  patient: patient,
                ),
            additionalInfo: additionalInfo,
            patient: patient,
          );
}
```

*Figure 40 - OffStateEvent class definition*

Creating the *OffStateEvent* class required a specific name, image, and test. The remaining properties (id, date, additional information, and patient) are generic for every event type, and not relevant. Since an event is treated equally in the entire system, as long as new *Event* subclass is specified this way, and the UI is updated to include a new option for the new event, new types can be introduced easily.

## 5.3 Tests

The tests implementation (feature 5 from section 3.5) is very similar to the events'. In terms of implementation details, events are a test with extra information. Despite the similarities, performing a test follows a very different use case. Generally, individual tests (not assigned to an event) should be performed regularly due to the doctor's prescription or in a clinical setting to help the doctor analyse the current situation. A typical scenario for performing a test is detailed below.

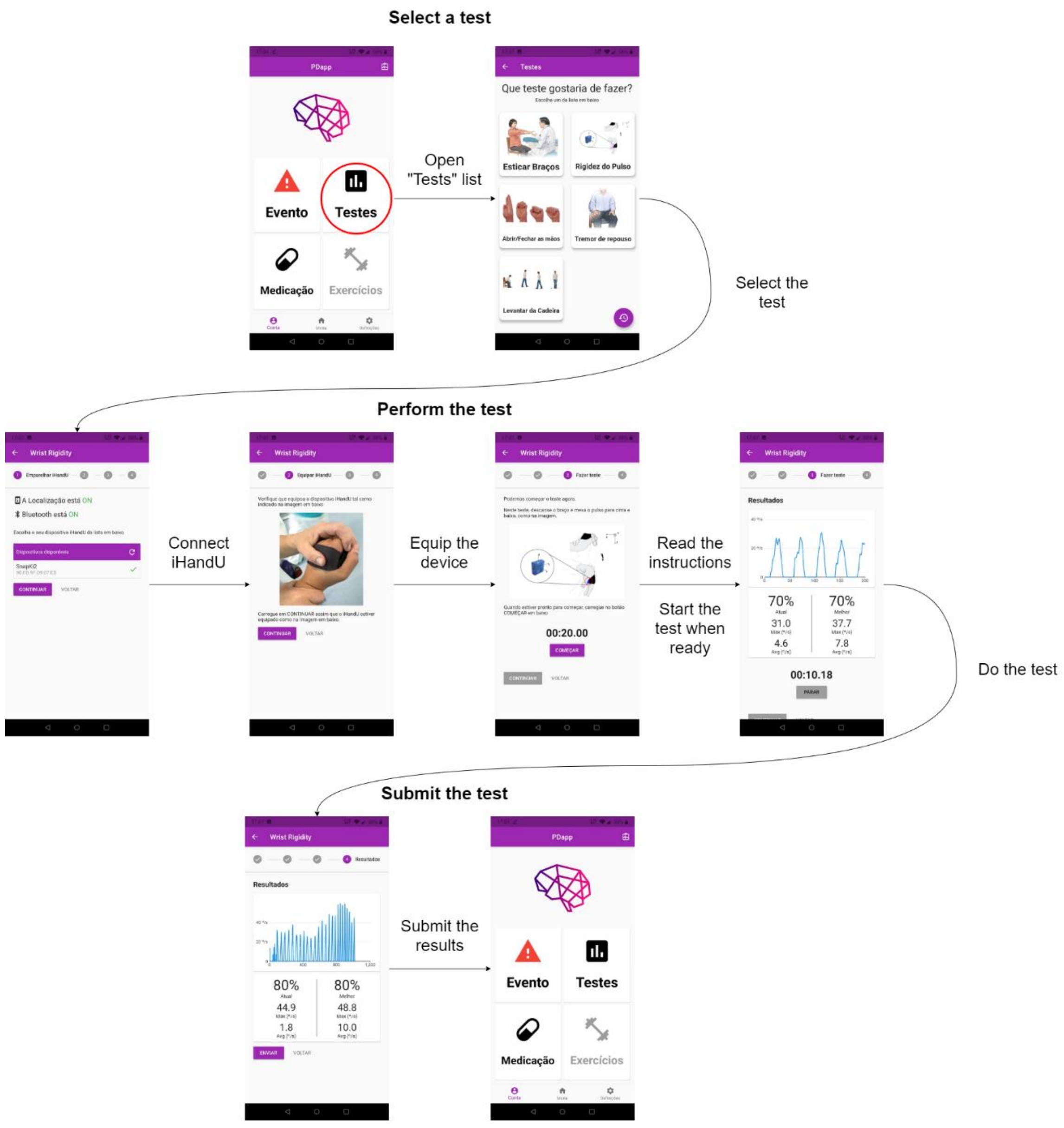


*Figure 41 – Test flow diagram for a Wrist Rigidity test*

Performing a test follows three important steps:

- *Selecting the test* – from the list of available tests; in the future, this could be automatically from a notification about a routine test
- *Performing the test* – each test follows four steps: setup of the device, equipping the device, doing the test, and reviewing the final results
- *Submitting the test* – after checking the results of the test, the event can be submitted

The implementation details of each step are defined in section 5.3.2.

### 5.3.1 Data structure

Since tests should also be open for extension, they too follow an abstract model specified below. There are multiple properties to be explained here:

- *equipImagePath* – every test needs an image (or video in future work) to describe the proper way to equip the device
- *movementMediaPath* – the movement required for the test should also be deeply described with some sort of media, to ensure the patient understands it completely.
- *isClinical* – since the addition of a *Clinical Mode,* a finished test is now flagged if done in a clinical setting
- *duration* – a test may have a pre-set duration (always 20 seconds) or unknown until the test is finished, at which point it will be set to the time it took to perform
- *results* – the results of a test are updated while a test is running and stored once the test finishes
- *equipDescription* – a text description is always presented to help describe how to equip the device
- *testDescription* – like the *equipDescription*, the test also requires a text description on how to perform the movements
- *calculate* – the algorithm used to calculate the test results is defined in this variable. It follows the defined type *Algorithm* defined before the class, which currently requires a list of bytes, an offset for reading these bytes, and returns an updated version of the results.

Following this abstract model allows for easy extension of new test types and updating current ones (due to improved algorithms and descriptions).

```
///Definition of the Algorithm function type
typedef Algorithm = TestResults Function(List<int> bytes, int offset);

abstract class Test extends Equatable {
  String id;

  ///Local path to the test's image
  final String imagePath;

  ///Local path to the test's correct way of equipping the device image
  final String equipImagePath;

  ///Local path to the test's correct movement image or video
  final String movementMediaPath;

  final String name;

  ///boolean describing if the Test was performed in [ClinicalMode] or [Patien
tMode]
  final bool isClinical;

  final DateTime date;

  ///Default Duration of the Test
  ///OR
  ///final duration of an already performed Test
  ///
  ///If null, test has no set duration and should update its final duration at
 the end
  Duration duration;

  TestResults results;

  ///Written description of how to equip the device
  final String equipDescription;

  ///Written description of how to perform the Test
  final String testDescription;

  final Patient patient;

  ///Assigned algorithm for calculating results from the device data
  final Algorithm calculate;
```

*Figure 42 - Test class definition*

### 5.3.2 Perform a test

To perform a test, the patient may access the “Tests” menu from the homepage and select the appropriate test from the given list. Once notifications are implemented, it is expected for scheduled tests to throw a push notification that redirects the patient directly to the correct test.

| **Test** | **Description** | **Algorithm implemented** |
|---|---|---|
| Stretch Arms | Analyse tremors by holding your arms stretched in front of you. | No |
| Wrist Rigidity | Analyse wrist rigidity with the iHandU. | Yes |
| Open/Close hands | Open and close your hands as quickly as possible to observe hesitations and velocity variations. | Yes |
| Resting Tremor | Calmly sit and rest your arms on the chair arms. This analyses passive tremors on all limbs. | No |
| Timed Up and Go | Start sitting, get up, walk 6 meters in front, walk back, and sit again—this analyses gait and difficulties rising from the chair. | No |

*Table 7 - Implemented tests table*

Once selected, the very similar step-by-step guide explained in chapter 5.2.2 begins.

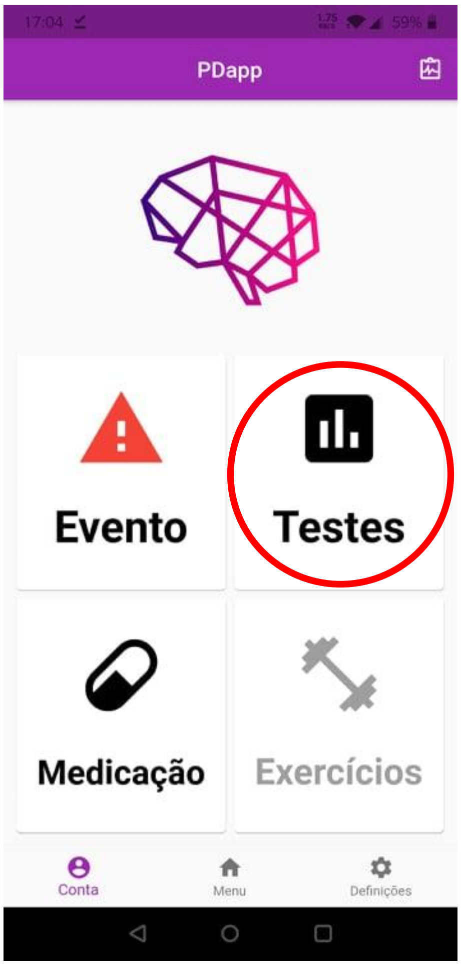


*Figure 44 – Homepage' “Tests” option*

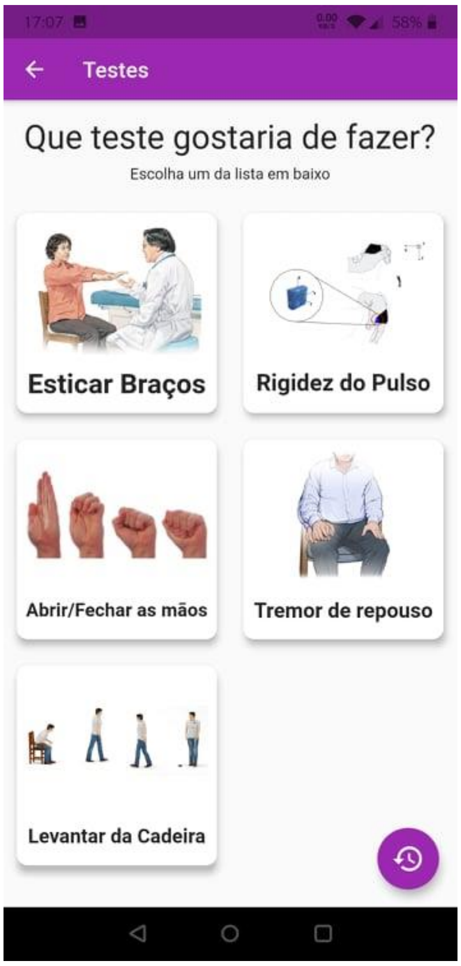
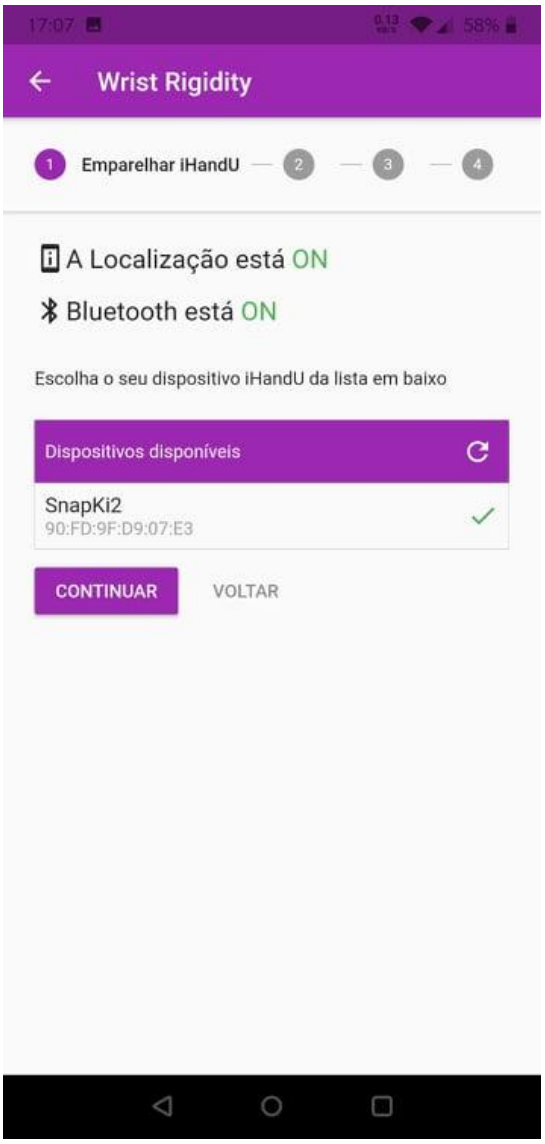


*Figure 43 – Tests list*

## A. Setup the device

This step is dedicated to discovering and connecting to the test's device (currently only the iHandU). Once all requirements are met, the search for devices begins, and the iHandU device can be selected and connected.

Once connected, the patient can move on to the next step.



A Localização está ON

Bluetooth está ON

Escolha o seu dispositivo iHandU da lista em baixo

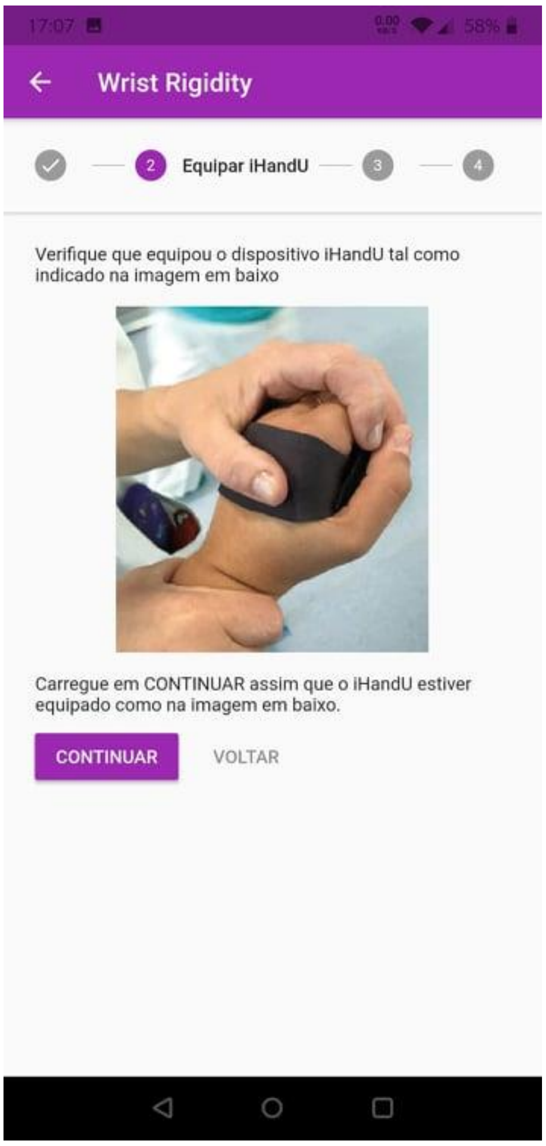
SnapKi2
90:FD:9F:D9:07:E3

*Figure 45 - Setup view*

## B. Equip the device correctly

On the Equip step, instructions are shown on how to equip the device for the test correctly. Text descriptions, images, and videos may be used to describe the equipping process better.

Once the patient is ready to advance, it may continue to the next step.



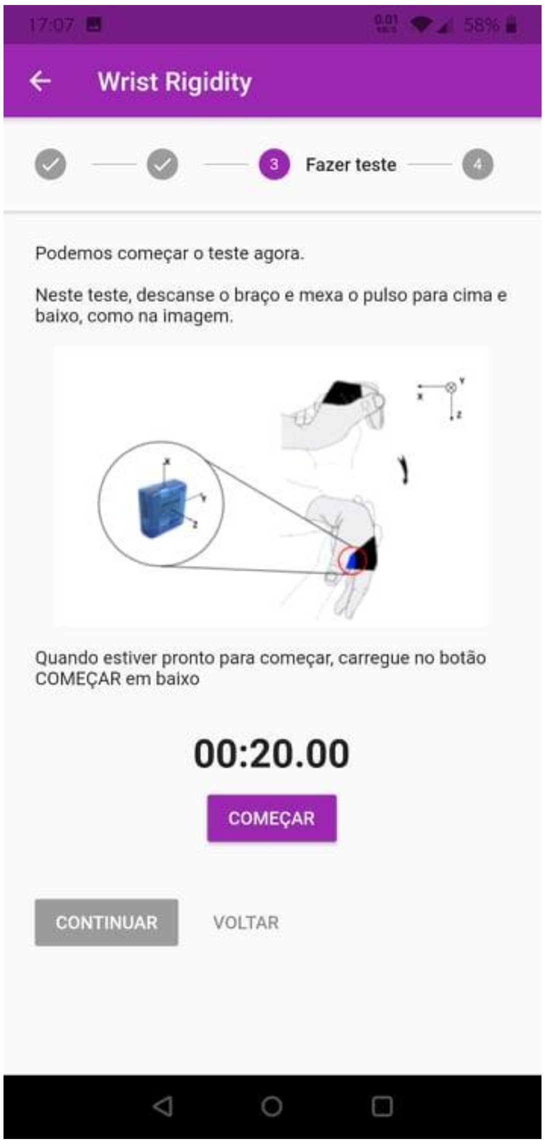
Verifique que equipou o dispositivo iHandU tal como indicado na imagem em baixo

Carregue em CONTINUAR assim que o iHandU estiver equipado como na imagem em baixo.

*Figure 46 - Equip instructions view*

## C. Perform the test

This step is where the actual test is performed. It begins with detailed instructions about what the test requires. Like with the Equip step, it too uses text, images, and video if necessary.



Podemos começar o teste agora.

Neste teste, descanse o braço e mexa o pulso para cima e baixo, como na imagem.

Quando estiver pronto para começar, carregue no botão COMEÇAR em baixo

00:20.00

*Figure 47 - Test instructions view*

If the test has a pre-set duration, the timer will countdown the duration necessary while the patient performs the test. In tests with no defined duration, the patient will have to stop the test once finished.

After the timer finishes or is stopped, the test is completed. The patient may now continue to the final results.

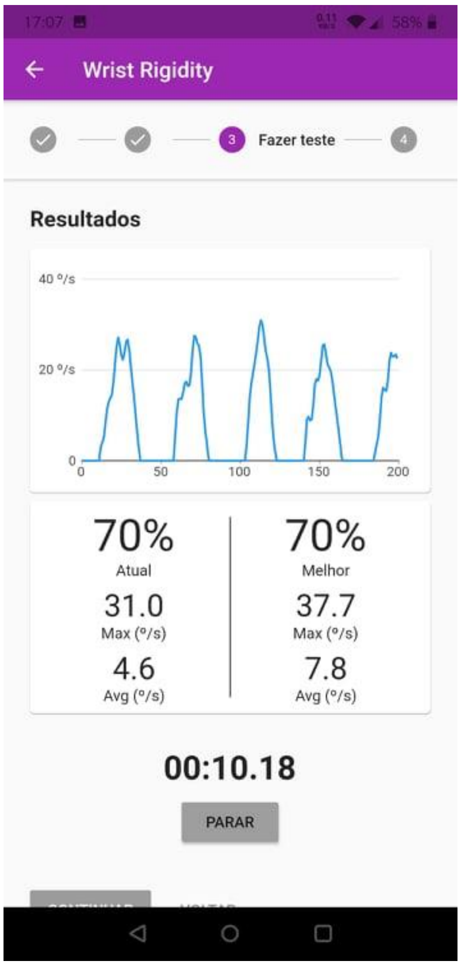


*Figure 49 - Test running view*

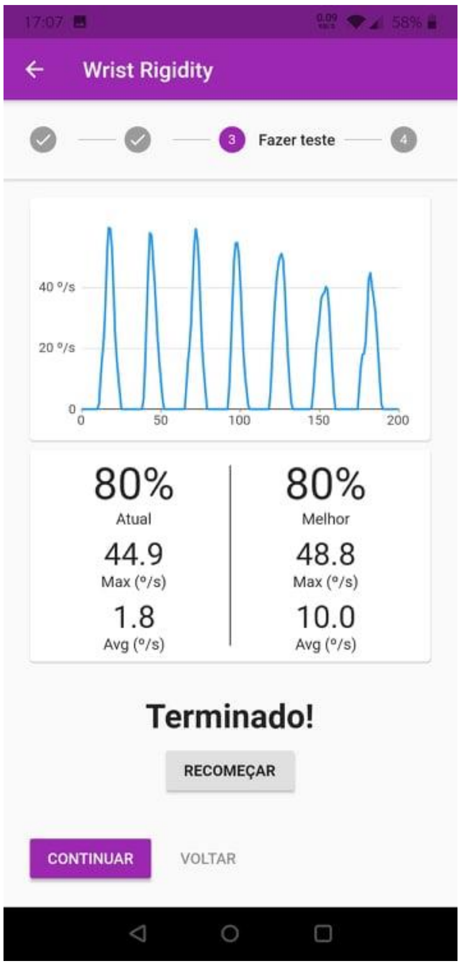
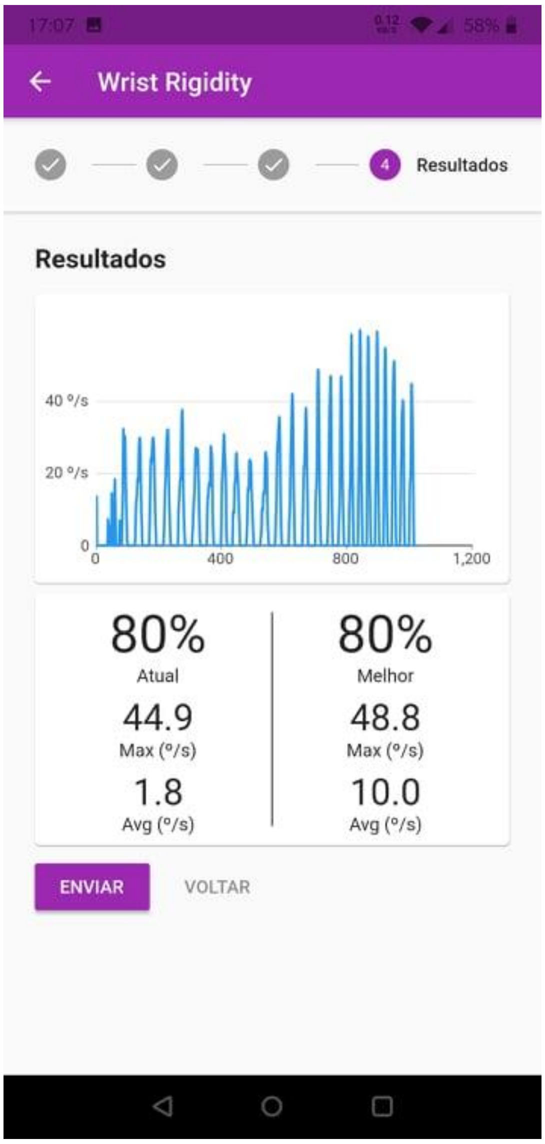


*Figure 48 - Test finished view*

## D. Final results

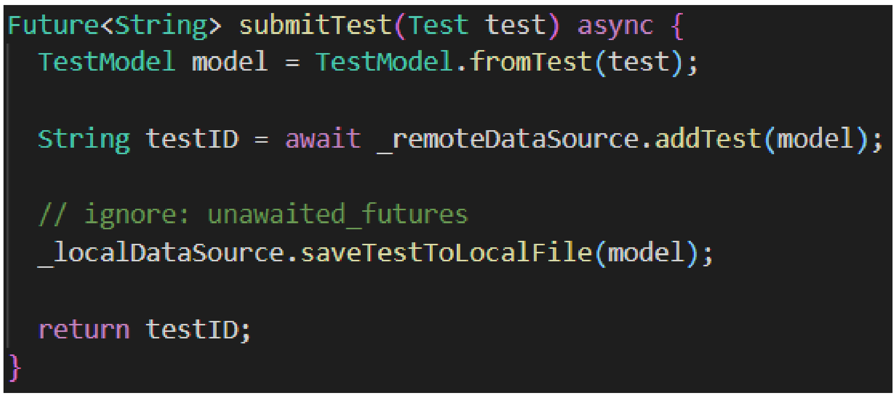
In this final step, the results graph and statistics are revealed and can be reviewed. Unlike an event, no additional information is necessary. After reviewing everything, the patient may submit the test through the “Submit” button and is redirected back to the homepage.



| | |
|---|---|
| 80% | 80% |
| Atual | Melhor |
| 44.9 | 48.8 |
| Max (°/s) | Max (°/s) |
| 1.8 | 10.0 |
| Avg (°/s) | Avg (°/s) |

*Figure 50 - Test final results view*

After triggering the submission, the app uses the *TestRepository* to submit the test. Before submitting to the database (using the *RemoteDataSource*) it creates a local *JSON* file with the results, for testing purposes. The results sent to the database are simplified to avoid submitting large arrays of bytes unnecessarily. If the real results are ever needed, they are stored in the app's dedicated local storage.

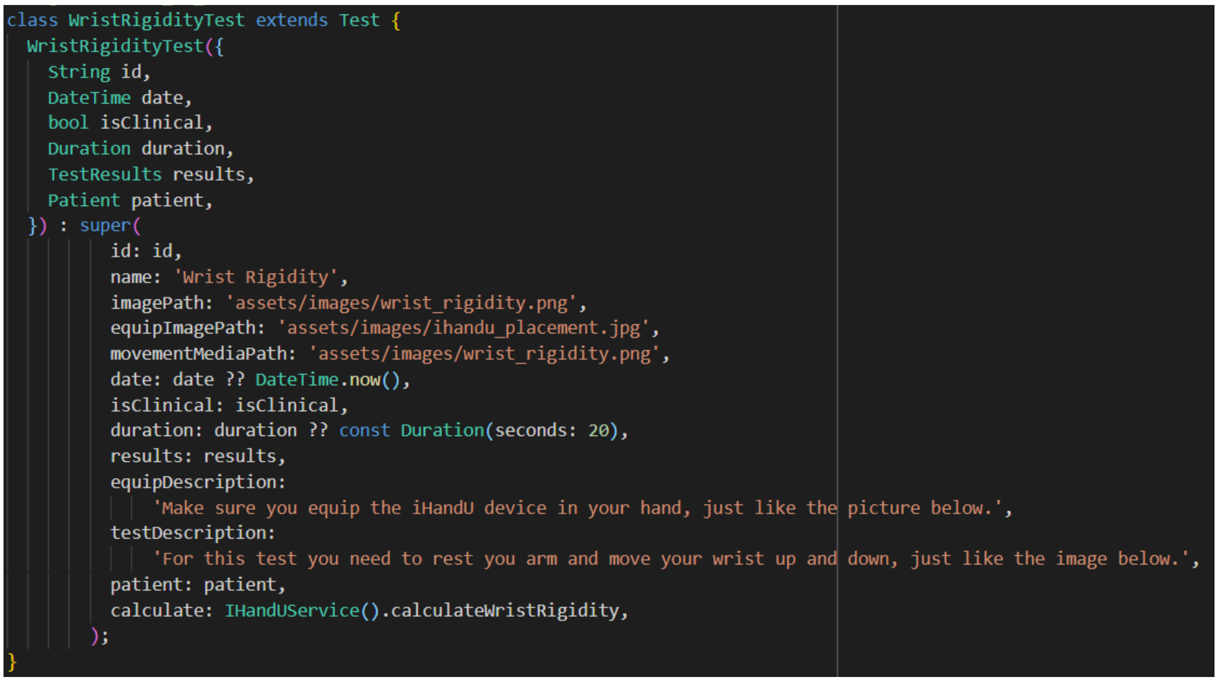


```
Future<String> submitTest(Test test) async {
  TestModel model = TestModel.fromTest(test);

  String testID = await _remoteDataSource.addTest(model);

  // ignore: unawaited_futures
  _localDataSource.saveTestToLocalFile(model);

  return testID;
}
```

*Figure 51 - TestRepository's submitTest implementation*

### 5.3.3 Adding new tests

Similarly to the events, tests also need to be easily extensible. With the abstract model defined in chapter 6.3.1, new tests can be created through hierarchy and changes are safe to be made without any rippling effects. Let us follow the example of the *WristRigidityTest*:



```
class WristRigidityTest extends Test {
  WristRigidityTest({
    String id,
    DateTime date,
    bool isClinical,
    Duration duration,
    TestResults results,
    Patient patient,
  }) : super(
          id: id,
          name: 'Wrist Rigidity',
          imagePath: 'assets/images/wrist_rigidity.png',
          equipImagePath: 'assets/images/ihandu_placement.jpg',
          movementMediaPath: 'assets/images/wrist_rigidity.png',
          date: date ?? DateTime.now(),
          isClinical: isClinical,
          duration: duration ?? const Duration(seconds: 20),
          results: results,
          equipDescription:
              'Make sure you equip the iHandU device in your hand, just like the picture below.',
          testDescription:
              'For this test you need to rest you arm and move your wrist up and down, just like the image below.',
          patient: patient,
          calculate: IHandUService().calculateWristRigidity,
        );
}
```

*Figure 52 - WristRigidityTest class declaration*

The defining properties of a test type: images and descriptions for the equip and test steps, duration, and algorithm are all defined here. If an improved algorithm is later developed, a simple change will update the implementation with no risk. The same goes for updating any descriptions or media used.

## 5.4 Medications

PDapp helps the patient keep up with their prescribed medication and any changes to their doctors' prescriptions (feature 4 from section 3.5). Once notifications are fully implemented, this feature will also keep track of the patient's intake history and help the doctors analyse this information.

As with the previous features, here is a simple diagram of the "checking a medication" use case.

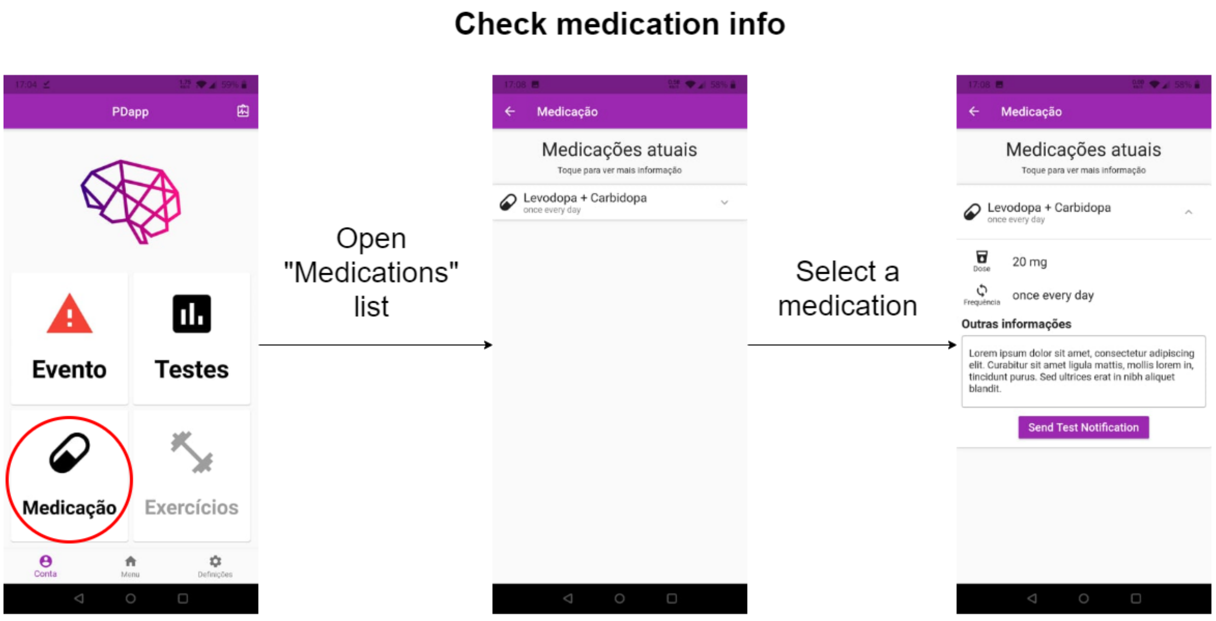


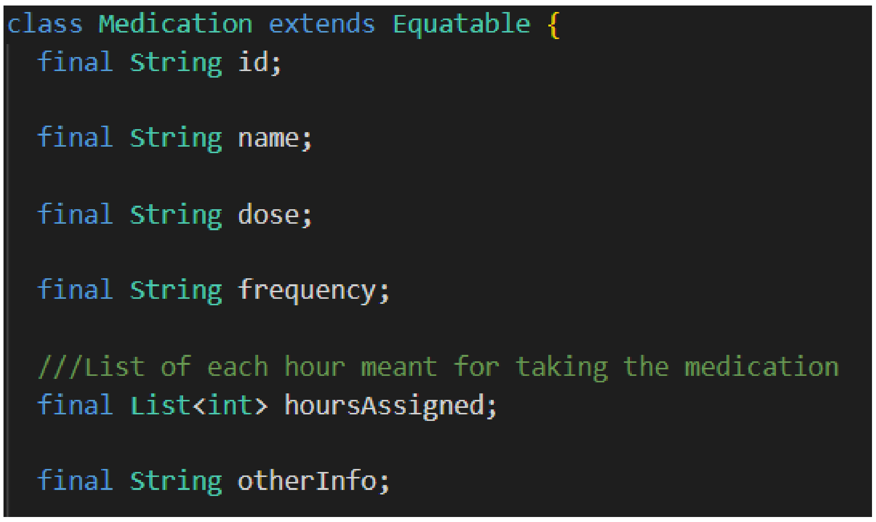
*Figure 53 – Checking a medication flow diagram*

Checking the prescribed medication is a simple process involving:

- *Opening the "Medications" list* – by simply tapping the "Medications" option in the homepage
- *Selecting one of the medications prescribed* – by tapping one of the panels shown
- *Analysing the detailed information of the medication* – once expanded, the medication panel holds all the information regarding the chosen medication

### 5.4.1 Data structure

The following class represents medications:



```
class Medication extends Equatable {
  final String id;

  final String name;

  final String dose;

  final String frequency;

  ///List of each hour meant for taking the medication
  final List<int> hoursAssigned;

  final String otherInfo;
```

*Figure 54 - Medication class definition*

Some of the most relevant properties:

- *dose* – the dosage of the medication, for each intake
- *frequency* – description of the frequency of each intake (ex. “two times a day”)
- *hoursAssigned* – if there are specific times of the day the medication should be taken, they are specified here (ex. “8:00 and 20:00”)
- *otherInfo* – any additional information the doctor wants to provide can be written here

## 5.4.2 Checking medications

As with the other main features, checking medications starts at the homepage by selecting the "Medications" option.

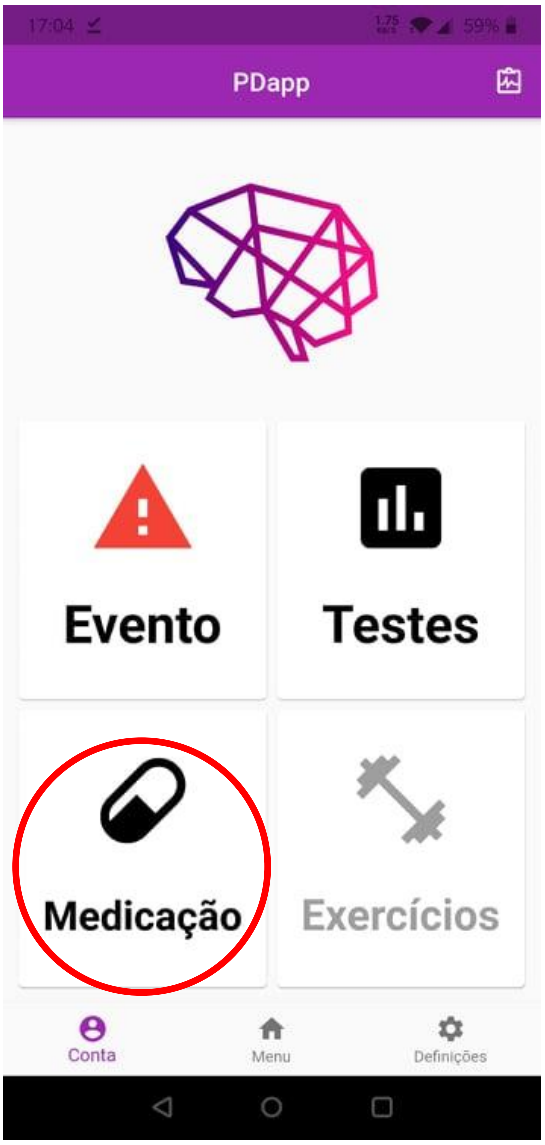


*Figure 55 - Homepage's "Medications" option*

The medications view hosts a list of every medication prescribed to the patient. Each medication is represented as an expansion panel and may be selected to show every specific information.

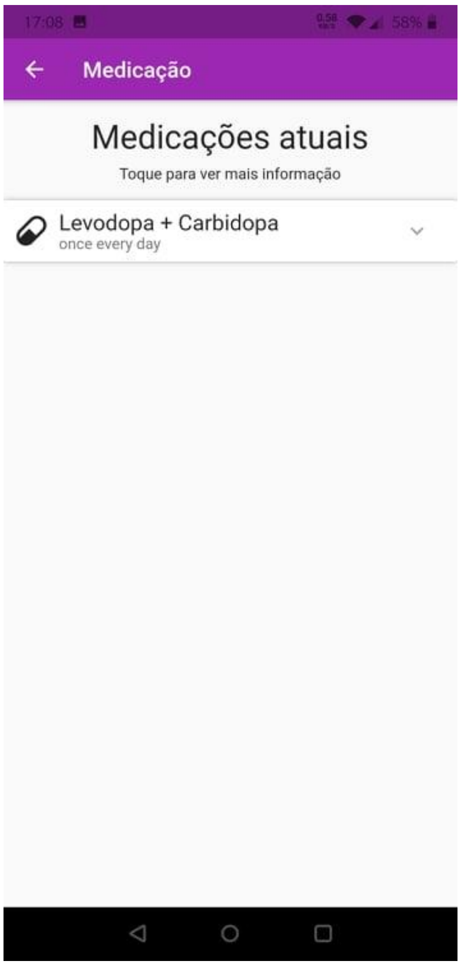
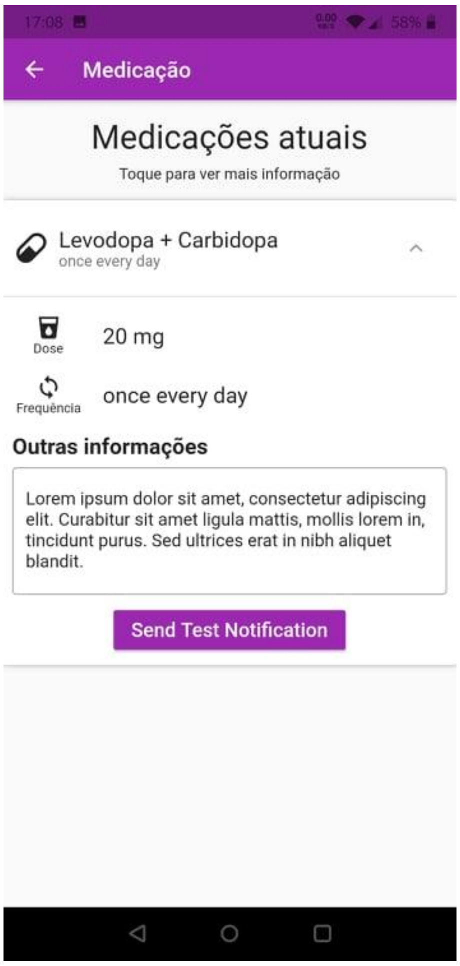


*Figure 57 – Medications list*

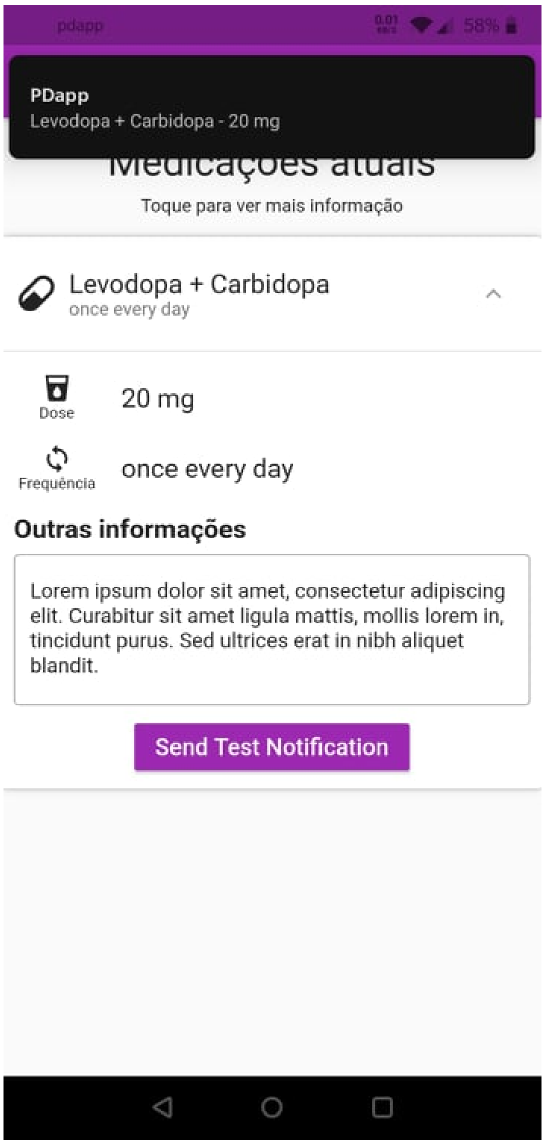


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur sit amet ligula mattis, mollis lorem in, tincidunt purus. Sed ultrices erat in nibh aliquet blandit.

*Figure 56 – Expanded medication view*

### 5.4.3 Notifications

Notifications could not be fully implemented due to time constraints and issues with integrating Cloud Messaging (Google's notifications service). However, a simple demo integration with local notifications allows for testing the behaviour of receiving notifications due to medications.



PDapp
Levodopa + Carbidopa - 20 mg

Medicações atuais
Toque para ver mais informação

Levodopa + Carbidopa
once every day

Dose 20 mg
Frequência once every day

Outras informações

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur sit amet ligula mattis, mollis lorem in, tincidunt purus. Sed ultrices erat in nibh aliquet blandit.

*Figure 58 - Medication notification*

## 5.5 iHandU Integration

Integration with the iHandU device was the core concern of the project and had to use multiple technologies to be done successfully. The following sections define how the connection via the recent *Bluetooth Low Energy* (BLE) technology was made with the device and how an *iHandU Service* was defined for handling the data received accordingly.

### 5.5.1 BLE Integration

In order to use the phone's Bluetooth capabilities, third-party packaged are required in Flutter. Due to the recent creation of BLE, integration with multiple packages was attempted, and in the end, the *FlutterBlue* package [28] was used.

For handling the bulk of the BLE connection cycle, a *BLEService* was created to interface the necessary actions needed to connect and read an iHandU device, such as:

- Check permissions
- Request permissions
- Search for devices
- Connect to a device
- Read data from a device

This interface allows packages to be switched later on with no need to switch dependencies all over the code, significantly reducing development time.

#### A. Permissions

The new BLE technology requires two conditions to be met in order to function on Android. The phone's Bluetooth must be enabled, and Location permissions must be granted to the app. Given these conditions, third-party packages were used to find these conditions' status and activate them on request.

For the Bluetooth functionality, the package *system_shortcuts* [29] was used to check the Bluetooth status and turn it on demand, to ease the user's need for interaction. For the Location permissions, the package *permission_handler* [30] checked the permissions granted to the device and requested access to the Location permission only when the app was being used.

With these two permissions handled, the BLE functionality may now be used in Android.

## B. Searching for devices

The *FlutterBlue* package allows searching for any BLE device nearby that follows specific parameters. This feature is very useful for only detecting the iHandU devices from any other BLE device nearby. It can be done by searching for BLE devices that broadcast a service with the same ID as programmed in the iHandU devices.

A timeout can also be specified for the search, should there be any need for it.

## C. Connecting to a device

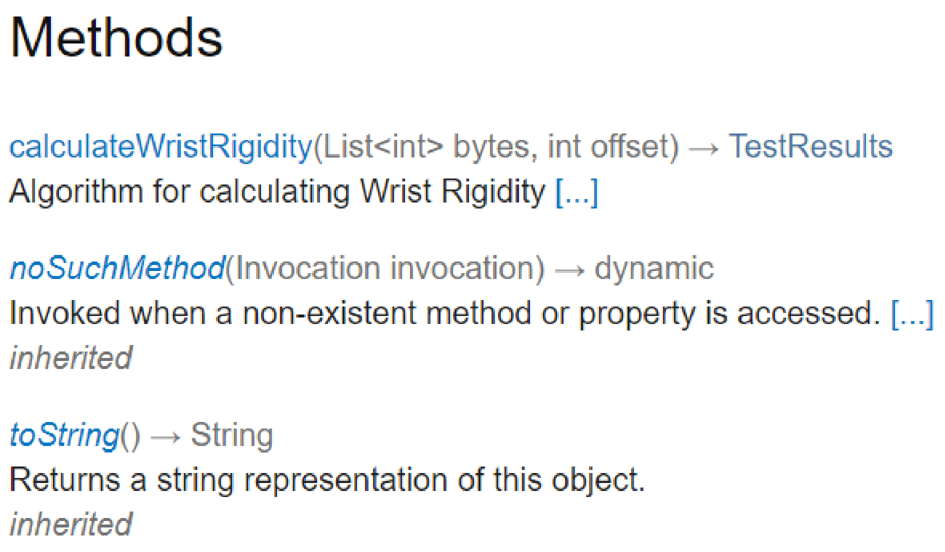
After finding a device, connecting to it requires the device's ID. As long as the connection is established (within the given timeout), it is connected until cancelled.

## D. Reading data from a device

After connecting to a device, each iHandU device broadcasts data through various *characteristics* [31] of the iHandU service. Depending on the data required, the BLE service creates a stream from the given *characteristic* and broadcasts the given bytes whenever they are emitted. This allows for asynchronous and real-time handling of any data required.

### 5.5.2 iHandU Service

The *IHandUService* was created to host all the iHandU system variables (service and characteristic IDs) and implement all the needed algorithms to handle and transform the data received. Any given test requires a function to handle a list of bytes and return processed data. This function should be one of the algorithm functions hosted in this service.



Methods

calculateWristRigidity(List<int> bytes, int offset) → TestResults
Algorithm for calculating Wrist Rigidity [...]

*noSuchMethod*(Invocation invocation) → dynamic
Invoked when a non-existent method or property is accessed. [...]
*inherited*

*toString*() → String
Returns a string representation of this object.
*inherited*

*Figure 59 - iHandU Service method definitions*

For example, for the wrist rigidity test, the *IHandUService* has a *calculateWristRigidity* function that handles all the raw data and returns the results. When a *WristRigidityTest* object is created, it is given this function as an attribute and uses it throughout the app when needed. This highly abstract architecture allows any new tests and algorithms to be implemented quickly and switched whenever needed.

# Chapter 6

# Dashboard

The doctor’s dashboard was developed with Flutter Web, and the design was to remain as simple and straightforward as possible. After multiple iterations and input from the doctor, the dashboard's current state is defined in the following sections.

## 6.1 Authentication

Since the dashboard is limited to verified doctors, authentication is essential. It uses Firebase Auth to handle the authentication process.

### 6.1.1 Auth Page

A simple authentication page was created with inputs for the credentials. It has all the necessary error handling if any error occurs during the login attempt.

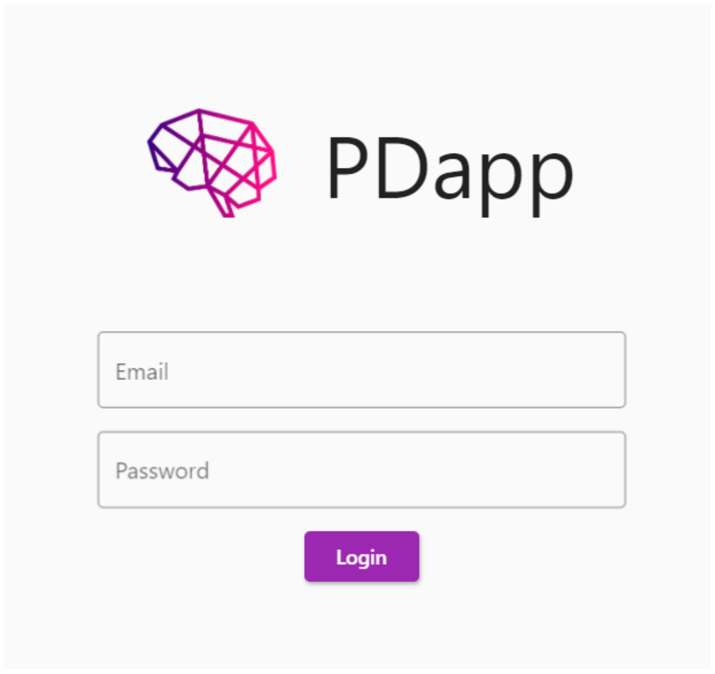
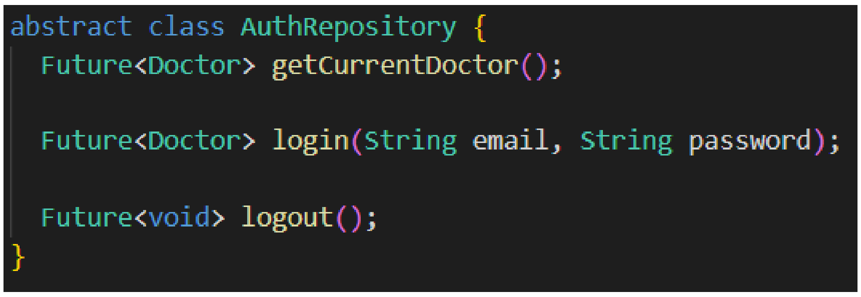


*Figure 60 - Authentication page*

### 6.1.2 Firebase Auth

After submitting the credentials, the *AuthRepository* handles the request and uses the Firebase Auth instance to verify the account. After a successful login, it uses the *DoctorRepository* to fetch the doctor data with the doctor's ID.

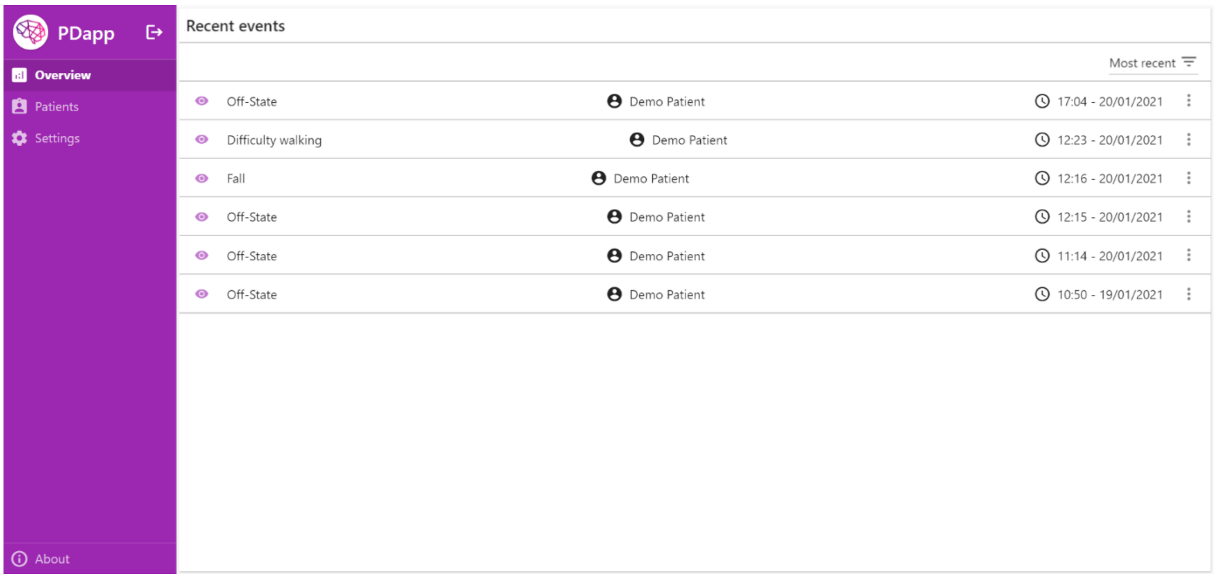


```
abstract class AuthRepository {
  Future<Doctor> getCurrentDoctor();

  Future<Doctor> login(String email, String password);

  Future<void> logout();
}
```

Figure 61 - AuthRepository definition

## 6.2 Overview

After authenticating, the user is redirected to the "Overview" page (feature 7 from section 3.5). This page hosts the most important statistics and alerts for the doctor. After interacting with the doctor, we concluded that showing a list of all recent events by every assigned patient was the only relevant feature, at the moment. As the app progresses, this overview page will likely change to meet new requirements.

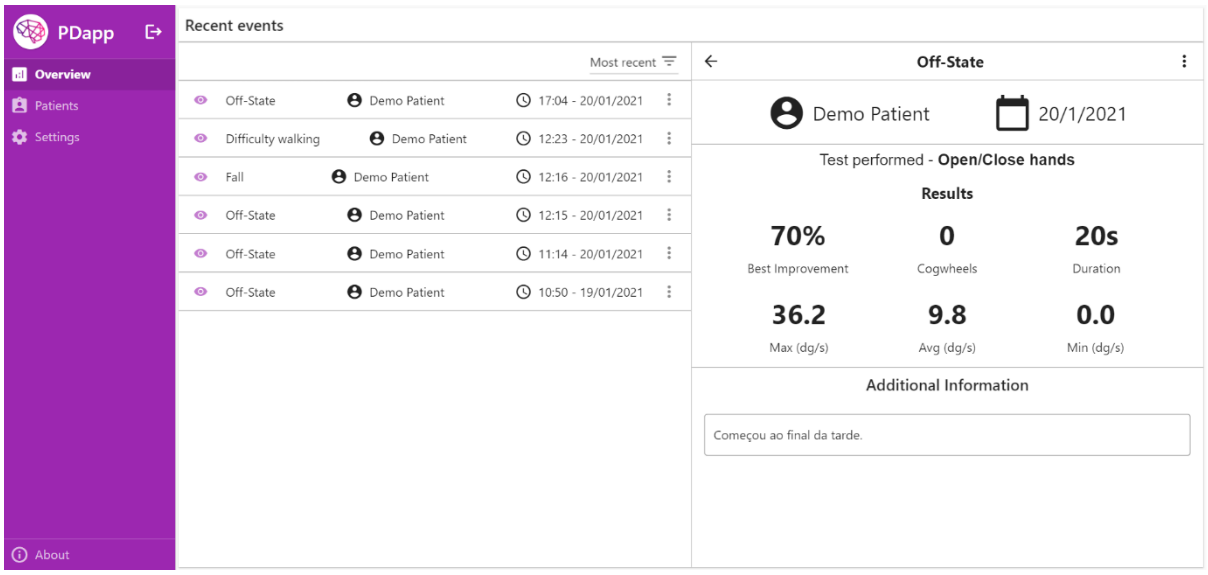


| | | |
|---|---|---|
| Off-State | Demo Patient | 17:04 - 20/01/2021 |
| Difficulty walking | Demo Patient | 12:23 - 20/01/2021 |
| Fall | Demo Patient | 12:16 - 20/01/2021 |
| Off-State | Demo Patient | 12:15 - 20/01/2021 |
| Off-State | Demo Patient | 11:14 - 20/01/2021 |
| Off-State | Demo Patient | 10:50 - 19/01/2021 |

Figure 62 - Overview page

### 6.2.1 Recent Events

The recent events widget is a list of the most recently issued events (last 30 days currently) by every patient assigned to the authenticated doctor. It follows an “email inbox” style for checking each event quickly while keeping the overall list available for searching.



| | | |
|---|---|---|
| Off-State | Demo Patient | 17:04 - 20/01/2021 |
| Difficulty walking | Demo Patient | 12:23 - 20/01/2021 |
| Fall | Demo Patient | 12:16 - 20/01/2021 |
| Off-State | Demo Patient | 12:15 - 20/01/2021 |
| Off-State | Demo Patient | 11:14 - 20/01/2021 |
| Off-State | Demo Patient | 10:50 - 19/01/2021 |

Off-State

Demo Patient 20/1/2021

Test performed - **Open/Close hands**

**Results**

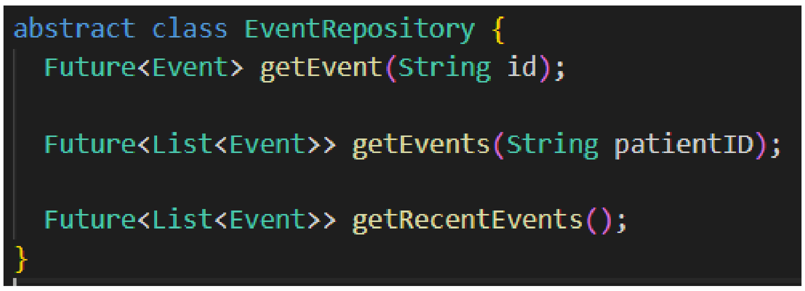
| | | |
|---|---|---|
| 70% | 0 | 20s |
| Best Improvement | Cogwheels | Duration |
| 36.2 | 9.8 | 0.0 |
| Max (dg/s) | Avg (dg/s) | Min (dg/s) |

Additional Information

Começou ao final da tarde.

*Figure 63 - Recent events widget with an event selected*

The app fetches every recent event by the doctor’s patients upon loading the page. It uses the *EventRepository* to access Cloud Firestore, fetch every event from the patients, and then sorts for the most recent ones.



```
abstract class EventRepository {
  Future<Event> getEvent(String id);

  Future<List<Event>> getEvents(String patientID);

  Future<List<Event>> getRecentEvents();
}
```

*Figure 64 - EventRepository's definition*

Selecting an event from the list opens up a detailed view containing every detail. Information such as the date, results, duration, and any additional information the patient submitted alongside. The side-by-side view allows the doctor to shift between events quickly. The related patient’s profile can be accessed by clicking the patient’s name.

## 6.3 Patients List

In order to search and access every patient assigned to the doctor, a dedicated page was created. It contains a grid composed of every patient and a search bar. The search functionality is currently not implemented yet, but the grid can still access every patient's profile.

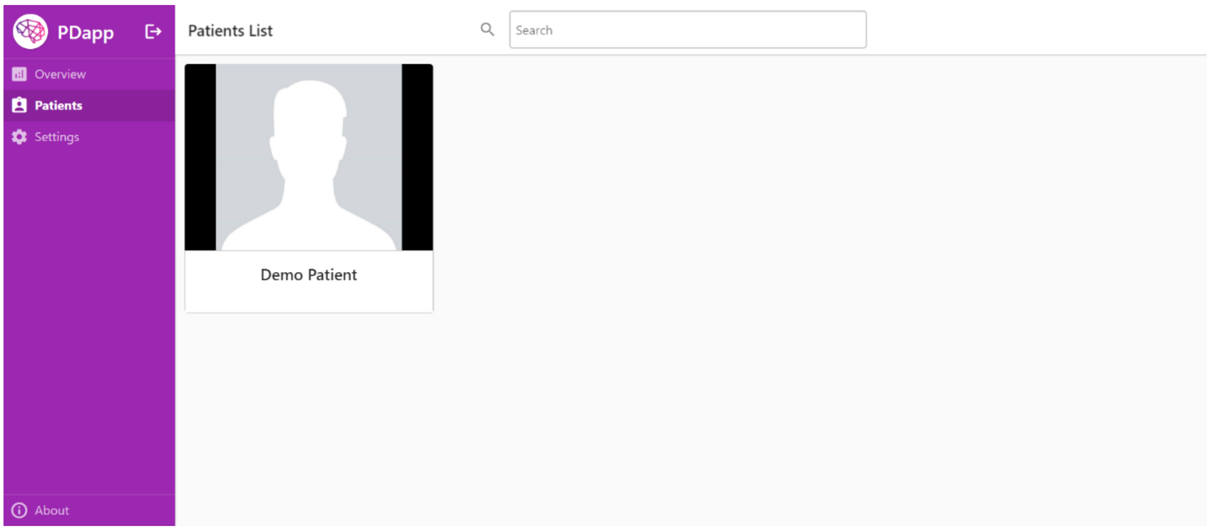
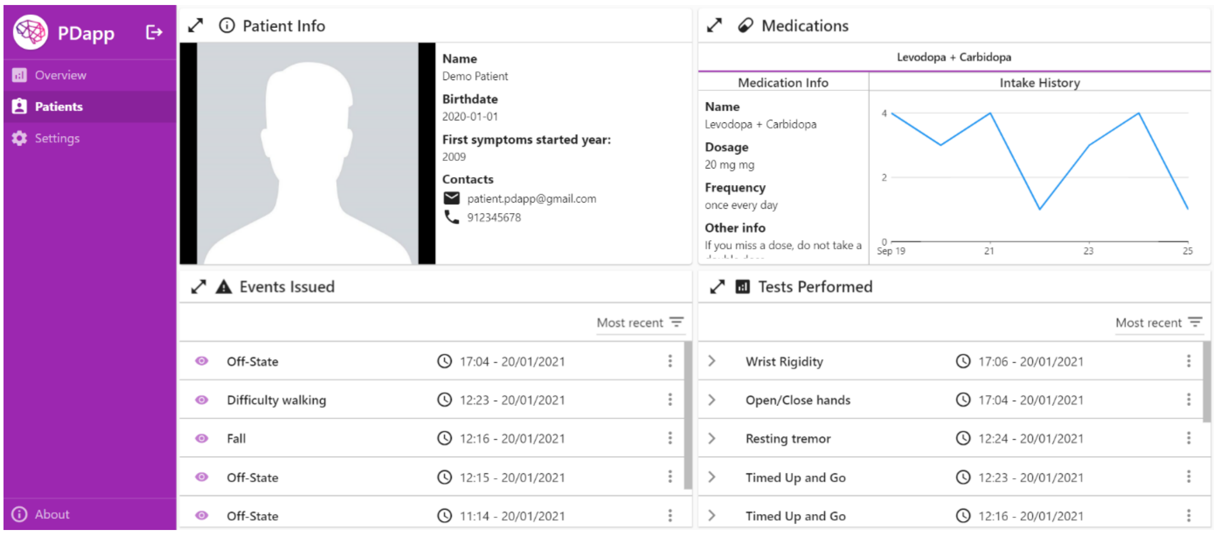


*Figure 65 - Patients list page*

## 6.4 Patient's Profile

The patient's profile is where every single patient data can be accessed (feature 8 from section 3.5). It features personal information, prescribed medications and intake history, every event issued, and tests performed. This page is where the doctor can analyse the progression of the patient through recent history.



**Name**
Demo Patient
**Birthdate**
2020-01-01
**First symptoms started year:**
2009
**Contacts**
patient.pdapp@gmail.com
912345678

**Name**
Levodopa + Carbidopa
**Dosage**
20 mg mg
**Frequency**
once every day
**Other info**
If you miss a dose, do not take a

| Events Issued | |
|---|---|
| Off-State | 17:04 - 20/01/2021 |
| Difficulty walking | 12:23 - 20/01/2021 |
| Fall | 12:16 - 20/01/2021 |
| Off-State | 12:15 - 20/01/2021 |
| Off-State | 11:14 - 20/01/2021 |

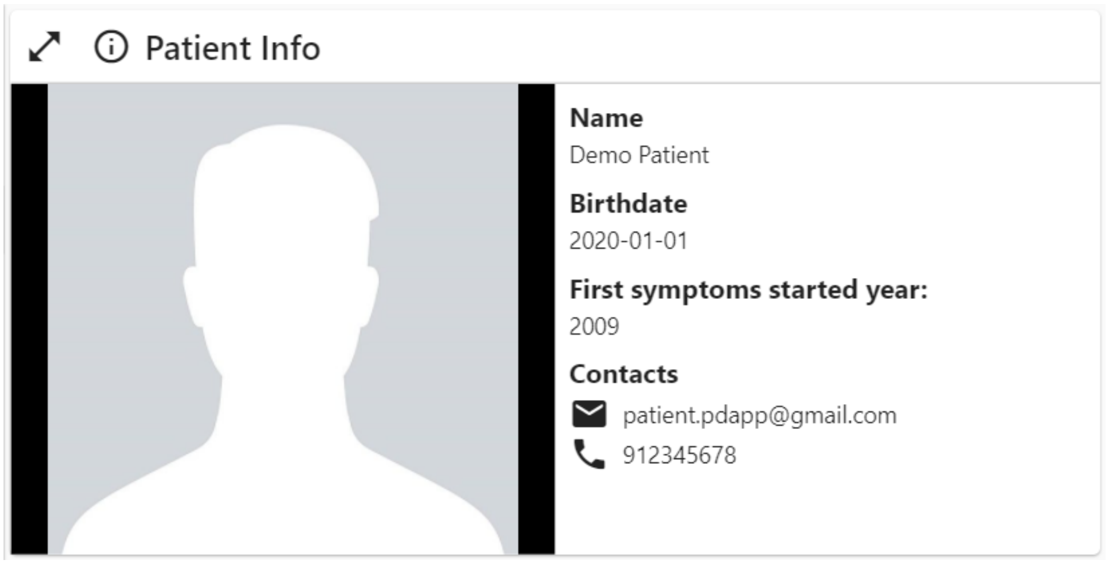
| Tests Performed | |
|---|---|
| Wrist Rigidity | 17:06 - 20/01/2021 |
| Open/Close hands | 17:04 - 20/01/2021 |
| Resting tremor | 12:24 - 20/01/2021 |
| Timed Up and Go | 12:23 - 20/01/2021 |
| Timed Up and Go | 12:16 - 20/01/2021 |

*Figure 66 - Patient profile*

Every panel on this page can be expanded to fit the whole visible area. This feature helps to focus on one widget and search through the events and tests lists easily, similarly to the recent events widget from the overview page.

### 6.4.1 Patient Info

The Patient Info panel is a simple display of the personal information saved in the system. It provides contact information and Parkinson's relevant data, such as the year the first symptoms started.



Patient Info

**Name**
Demo Patient

**Birthdate**
2020-01-01

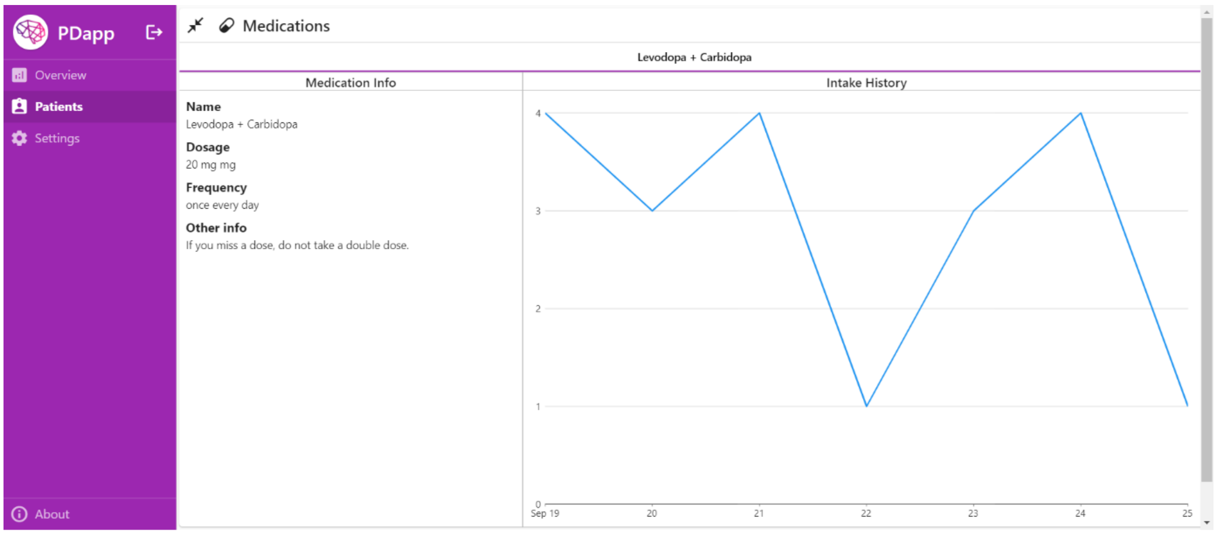
**First symptoms started year:**
2009

**Contacts**
patient.pdapp@gmail.com
912345678

*Figure 67 - Patient information panel*

### 6.4.2 Medications prescribed

The prescribed medications panel allows the doctor to manage current medications by makes any changes necessary, removing and adding new ones, and checking the intake history. Due to time constraints, full functionality was not possible to implement. Currently, this panel fetches the current medications and displays the information. The management of medications and intake history will be developed in future work.

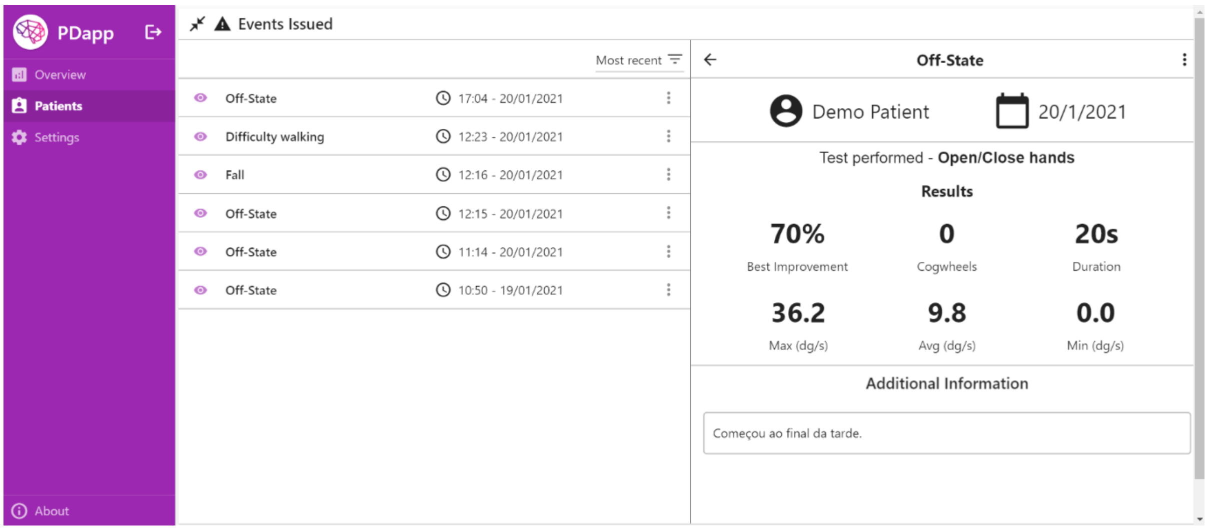


**Name**
Levodopa + Carbidopa
**Dosage**
20 mg mg
**Frequency**
once every day
**Other info**
If you miss a dose, do not take a double dose.

*Figure 68 - Medications panel expanded*

### 6.4.3 Events emitted

Every event issued by the patient is presented as an ordered list in this panel. It can be expanded to use the same side-by-side layout as the previous recent events widget. In its current form, it is possible to check the results and additional information about each event and assess the current situation that way.



| | |
|---|---|
| Off-State | 17:04 - 20/01/2021 |
| Difficulty walking | 12:23 - 20/01/2021 |
| Fall | 12:16 - 20/01/2021 |
| Off-State | 12:15 - 20/01/2021 |
| Off-State | 11:14 - 20/01/2021 |
| Off-State | 10:50 - 19/01/2021 |

**Off-State**

Demo Patient 20/1/2021

Test performed - **Open/Close hands**

**Results**

| 70% | 0 | 20s |
|---|---|---|
| Best Improvement | Cogwheels | Duration |
| 36.2 | 9.8 | 0.0 |
| Max (dg/s) | Avg (dg/s) | Min (dg/s) |

Additional Information

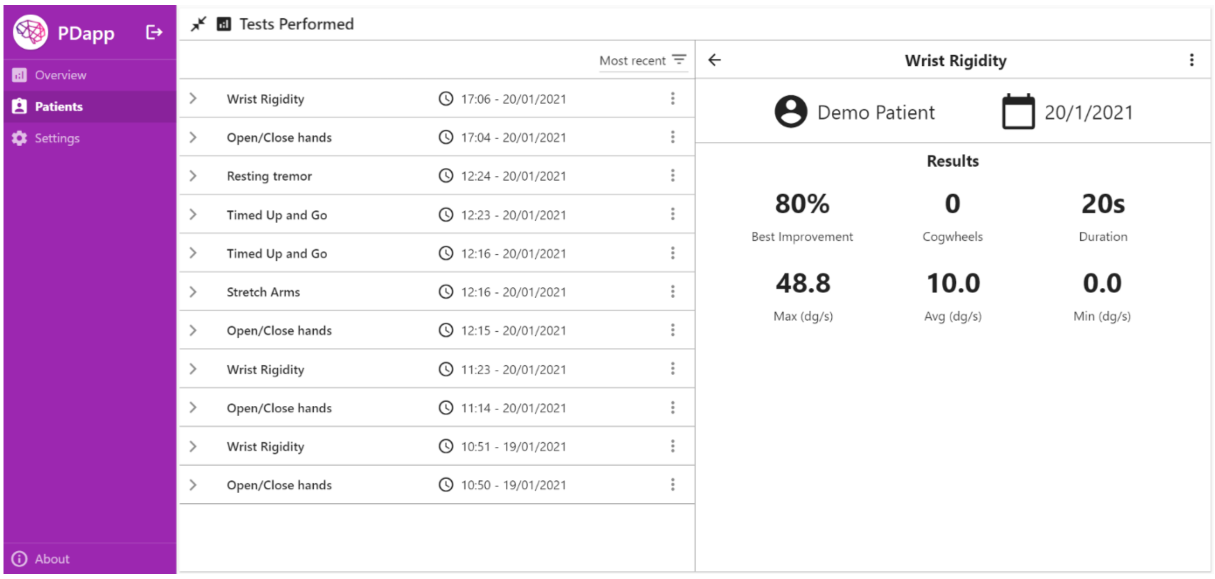
Começou ao final da tarde.

*Figure 69 - Events issued expanded panel*

As future work, the ability to compare multiple events side-by-side would be beneficial. A calendar view to better analyse the events in their proper context would also be a good addition.

### 6.4.4 Tests performed

The tests panel is very similar to the events issued one. This panel also features a list of every test performed by the patient and its results. This list is the best way to understand the patient's progression through time by analysing the prescribed regular tests. Currently, it also features every test associated with an event, which could be changed if requirements are altered.



| | |
|---|---|
| Wrist Rigidity | 17:06 - 20/01/2021 |
| Open/Close hands | 17:04 - 20/01/2021 |
| Resting tremor | 12:24 - 20/01/2021 |
| Timed Up and Go | 12:23 - 20/01/2021 |
| Timed Up and Go | 12:16 - 20/01/2021 |
| Stretch Arms | 12:16 - 20/01/2021 |
| Open/Close hands | 12:15 - 20/01/2021 |
| Wrist Rigidity | 11:23 - 20/01/2021 |
| Open/Close hands | 11:14 - 20/01/2021 |
| Wrist Rigidity | 10:51 - 19/01/2021 |
| Open/Close hands | 10:50 - 19/01/2021 |

| | | |
|---|---|---|
| 80% | 0 | 20s |
| Best Improvement | Cogwheels | Duration |
| 48.8 | 10.0 | 0.0 |
| Max (dg/s) | Avg (dg/s) | Min (dg/s) |

*Figure 70 - Tests performed panel*

Having time-focused views would help analyse the patient's progression and is being highly considered future work.

# Chapter 7

# Results

Being the first iteration of the entire PDapp system, and accompanying the Covid-19 pandemic, gathering various review points was difficult. There were only two meetings with a doctor to show the current status and both were met with positive remarks and few adjustments. In the end, a working prototype was developed and could be tested in a clinical setting as a proof-of-concept to gather feedback from the patient's and doctor's point of view. A photograph of a live demo can be seen in figure 71.

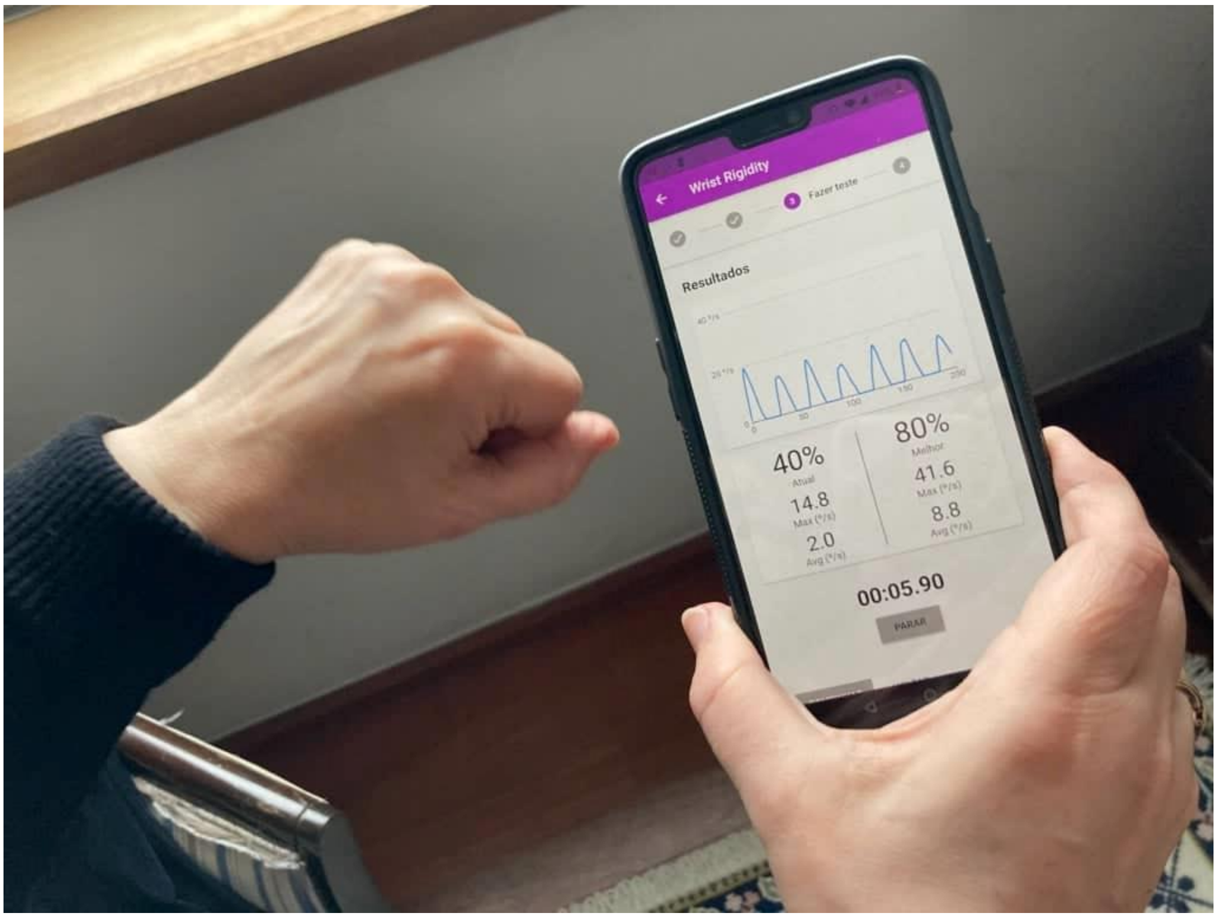


*Figure 71 - PDapp being used to measure the wrist rigidity*

The following sections will describe the use cases tested by the doctor and the results from a user evaluation form created for this purpose.

## 7.1 Tested use cases

Each available feature in the demo is listed below. In addition to the demo application, a detailed step-by-step user guide was provided to cover every detail of each use case and possible issues.

Every feature has a detailed live demonstration video published in the following youtube playlist: http://bit.ly/pdapp-playlist

### A. Reporting an Event

The mobile demo app is capable of selecting any event from the list, searching and connecting with the iHandU device, and performing the associated test. Once submitted, the dashboard was immediately updated, and the results could be checked both in the overview and the patient's profile.

### B. Performing a test

Performing any of the implemented tests is possible through the *Tests* feature. Like with the events, the user can select a test from the list, connect with the iHandU device, perform the test, and submit the results. The results are directly uploaded and available in the dashboard from the patient's profile.

### C. Checking medication

Checking the currently prescribed medications is implemented, as well as testing a simple notification for demo purposes. Managing medications (add, remove, update) could not be developed in time and has been added as future work.

### D. Checking patient history

The doctor can see past events and test performed from the patient's profile. The currently implemented widgets allow for quick and easy shifting between events/tests and their results.

## 7.2 User evaluation

To evaluate user experience, a form was created using Google Forms to get feedback from the testing users. Most answers consisted of 1-5 (very good to very bad) choices, with an occasional open answer to gather feedback. The form contained the following questions:

### A. Mobile App General Questions

- Did you find PDapp too complicated? (hard to navigate)
- How would you rate the app's design and interfaces

### B. Report an Event

- Which event did you report? (if any)
- What did you think of the variety of events available?
- Did the connection with the iHandU go well?
- Were the test instructions useful?
- How was the experience and quality of the test?
- What did you make of the final results?

### C. Perform a Test

- Which test did you perform? (if any)
- What did you think of the variety of tests available?
- Did the connection with the iHandU go well?
- Were the test instructions useful?
- How was the experience and quality of the test?
- What did you make of the final results?

### D. Check medications

- How was the variety of information available?

### E. Dashboard General Questions

- Did you find PDapp too complicated? (hard to navigate)
- How would you rate the app's design and interfaces

### F. Recent Events

- Did you like the Recent Events widget?
- What did you think of the information available about each event?

## G. Patient’s Profile

- How was the patient’s profile?
- Was there enough information about the patient?
- What did you make of the medication’s view?
- What did you make of the issued events’ view?
- What did you make of the tests performed view?

## H. Final suggestions

- Any final suggestions or observations about PDapp?

Due to the issues mentioned before (covid-19 and overall time), only one result has been gathered so far. This answer can be seen in the annexed pdf file. More results are expected to come soon as the demo is handed to multiple doctors, and they have the chance to try the system for themselves.

# Conclusions and future work

PDapp is a rather ambitious project that can provide Parkinson’s disease patients with a powerful tool to manage every aspect of their disease. Throughout this thesis development, a very solid proof-of-concept has been developed that has pleased doctors and gave the powerful iHandU device a way to perform outside of the clinical environment. The foundations of this system are built, and its flexible architecture allows for easy extension and scalability. In its current form, only a few improvements are required to get PDapp in patients' hands to start enjoying its capabilities.

From this project’s scope, a few things are now in line as future work. The medications feature lacks proper management utilities to be useful, such as the addition, removal and updating of medications. The use of Google’s Firebase [32] services allowed for faster development, but carry too many data privacy concerns to be useful in production. A dedicated database solution should be designed and implemented before PDapp is ready to acquire real patient data. The notifications system is very valuable for the patient and should be implemented next to ensure medications and routine tests get all the attention they require. The dashboard could also be improved with more customisable views for interpreting the data by comparing results, tracking histories, and more. A final consideration could be to further develop algorithms to support adding more tests since they are the only bottleneck for expanding this feature.

PDapp is one step further in improving the quality of life of Parkinson’s patients. The constant difficulties of tracking the disease’s progression, adjusting medication doses, contacting doctors in a timely fashion, will soon be handled by this innovative system and patients will be able to enjoy their lives with less concern.

# References

[1] "Parkinson's Foundation." https://www.parkinson.org/understanding-parkinsons/what-is-parkinsons (accessed May 02, 2020).

[2] B. M. Bot *et al.*, "The mPower study, Parkinson disease mobile data collected using ResearchKit," *Sci. Data*, vol. 3, pp. 1–9, 2016, doi: 10.1038/sdata.2016.11.

[3] "Deep Brain Stimulation for Movement Disorders Information Page | National Institute of Neurological Disorders and Stroke." https://www.ninds.nih.gov/Disorders/All-Disorders/Deep-Brain-Stimulation-Movement-Disorders-Information-Page (accessed Jun. 15, 2020).

[4] E. M. Lopes, M. D. C. Vilas-Boas, D. Dias, M. J. Rosas, R. Vaz, and J. P. S. Cunha, "IHandU: A novel quantitative wrist rigidity evaluation device for deep brain stimulation surgery," *Sensors (Switzerland)*, vol. 20, no. 2, pp. 1–21, 2020, doi: 10.3390/s20020331.

[5] K. M. Tsiouris *et al.*, "PD-Manager: An mHealth platform for Parkinson's disease patient management," *Healthc. Technol. Lett.*, vol. 4, no. 3, pp. 102–108, 2017, doi: 10.1049/htl.2017.0007.

[6] A. Sahyoun, K. Chehab, O. Al-Madani, F. Aloul, and A. Sagahyroon, "ParkNosis: Diagnosing Parkinson's disease using mobile phones," *2016 IEEE 18th Int. Conf. e-Health Networking, Appl. Serv. Heal. 2016*, 2016, doi: 10.1109/HealthCom.2016.7749491.

[7] S. Estévez-Martín, M. E. Cambronero, Y. García-Ruiz, and L. Llana, "Mobile applications for people with Parkinson's disease: A systematic search in app stores and content review," *J. Univers. Comput. Sci.*, vol. 25, no. 7, pp. 740–763, 2019, doi: 10.3217/jucs-025-07-0740.

[8] M. Linares-del Rey, L. Vela-Desojo, and R. Cano-de la Cuerda, "Mobile phone applications in Parkinson's disease: A systematic review," *Neurologia*, vol. 34, no. 1, pp. 38–54, 2019, doi: 10.1016/j.nrl.2017.03.006.

[9] S. Estévez-Martín, M. E. Cambronero, Y. García-Ruiz, and L. Llana, "Mobile applications for people with Parkinson's disease: A systematic search in app stores and content review," *J. Univers. Comput. Sci.*, vol. 25, no. 7, pp. 740–763, 2019, Accessed: Jun. 07, 2020. [Online]. Available: http://www.jucs.org/jucs_25_7/mobile_applications_for_people/jucs_25_07_0740_0763_martin.pdf.

[10] "mHealth platform for Parkinson's disease management." http://www.parkinson-manager.eu/ (accessed Jun. 29, 2020).

[11] "Mobile health system helps to better manage Parkinson's disease | Shaping Europe's digital future." https://ec.europa.eu/digital-single-market/en/news/mobile-health-system-helps-better-manage-parkinsons-disease?utm_content=buffer04568&utm_medium=social&utm_source=twitter.com&utm_campaign=buffer (accessed Jan. 25, 2021).

[12] C. Pereira, P. Macedo, and R. N. Madeira, "Mobile integrated assistance to empower people coping with Parkinson's disease," *ASSETS 2015 - Proc. 17th Int. ACM SIGACCESS Conf. Comput. Access.*, pp. 409–410, 2015, doi: 10.1145/2700648.2811394.

[13] J. J. Elm *et al.*, "Feasibility and utility of a clinician dashboard from wearable and mobile application Parkinson's disease data," *npj Digit. Med.*, vol. 2, no. 1, 2019, doi: 10.1038/s41746-019-0169-y.

[14] E. Kuosmanen *et al.*, "Mobile-based Monitoring of Parkinson's Disease," in *ACM International Conference Proceeding Series*, Nov. 2018, pp. 441–448, doi: 10.1145/3282894.3289737.

[15] "mPower 2.0." https://parkinsonmpower.org/your-story (accessed Jun. 29, 2020).

[16] "mPower Public Researcher Portal - syn4993293." https://www.synapse.org/#!Synapse:syn4993293 (accessed Jun. 29, 2020).

[17] K. A. Al Mamun, M. Alhussein, K. Sailunaz, and M. S. Islam, "Cloud based framework for

Parkinson’s disease diagnosis and monitoring system for remote healthcare applications,” *Futur. Gener. Comput. Syst.*, vol. 66, pp. 36–47, 2017, doi: 10.1016/j.future.2015.11.010.

[18] P. Klumpp, T. Janu, T. Arias-Vergara, J. C. V. Correa, J. R. Orozco-Arroyave, and E. Nöth, “Apkinson - A mobile monitoring solution for Parkinson’s disease,” *Proc. Annu. Conf. Int. Speech Commun. Assoc. INTERSPEECH*, vol. 2017-Augus, no. September, pp. 1839–1843, 2017, doi: 10.21437/Interspeech.2017-416.

[19] S. Bernardini, C. Cianfrocca, M. Maioni, M. Pennacchini, D. Tartaglini, and L. Vollero, “A Mobile App for the Remote Monitoring and Assistance of Patients with Parkinson’s Disease and their Caregivers,” *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS*, vol. 2018-July, pp. 2909–2912, 2018, doi: 10.1109/EMBC.2018.8512989.

[20] B. Blažica, P. Novak, F. Novak, and B. K. Seljak, “Design and Usability Evaluation of Interface of Mobile Application for Nutrition Tracking for People with Parkinson’s Disease,” *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11582 LNCS, pp. 200–208, 2019, doi: 10.1007/978-3-030-22219-2_15.

[21] “Flutter - Beautiful native apps in record time.” https://flutter.dev/ (accessed Jan. 26, 2021).

[22] “Web support for Flutter - Flutter.” https://flutter.dev/web (accessed Jan. 27, 2021).

[23] “React Native · A framework for building native apps using React.” https://reactnative.dev/ (accessed Jan. 27, 2021).

[24] “Ionic - Cross-Platform Mobile App Development.” https://ionicframework.com/ (accessed Jan. 27, 2021).

[25] “Clean Coder Blog.” https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html (accessed Jan. 27, 2021).

[26] “TestDrivenDevelopment.” https://martinfowler.com/bliki/TestDrivenDevelopment.html (accessed Jan. 27, 2021).

[27] “flutter_blue | Flutter Package.” https://pub.dev/packages/flutter_blue (accessed Jan. 26, 2021).

[28] “system_shortcuts | Flutter Package.” https://pub.dev/packages/system_shortcuts (accessed Jan. 26, 2021).

[29] “permission_handler | Flutter Package.” https://pub.dev/packages/permission_handler (accessed Jan. 26, 2021).

[30] “A Developer’s Guide To Bluetooth | Bluetooth® Technology Website.” https://www.bluetooth.com/blog/a-developers-guide-to-bluetooth/#characteristics (accessed Jan. 26, 2021).

[31] “Firebase.” https://firebase.google.com/ (accessed Jan. 26, 2021).