UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ENGENHARIA ELETRÔNICA - DAELN
CURSO DE ENGENHARIA ELETRÔNICA

FELIPE DOS SANTOS NEVES,GIULIA HEIDI BORTOLETO

SISTEMA EMBARCADO PARA A MEDIÇÃO DE FORÇA APLICADA E TEMPO DE RESPOSTA DURANTE FRENAGEM

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2018

FELIPE DOS SANTOS NEVES,GIULIA HEIDI BORTOLETO

SISTEMA EMBARCADO PARA A MEDIÇÃO DE FORÇA APLICADA E TEMPO DE RESPOSTA DURANTE FRENAGEM

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia Eletrônica da Universidade Tecnológica Federal do Paraná, como requisito parcial para a obtenção do título de Bacharel.

Orientador: Bertoldo Schneider Junior
Universidade Tecnológica Federal do Paraná

CURITIBA
2018

FELIPE DOS SANTOS NEVES
GIULIA HEIDI BORTOLETO

SISTEMA EMBARCADO PARA A MEDIÇÃO DE FORÇA APLICADA E TEMPO DE RESPOSTA DURANTE FRENAGEM

Este Trabalho de Conclusão de Curso de Graduação foi apresentado como requisito parcial para obtenção do título de Engenheiro Eletrônico, do curso de Engenharia Eletrônica do Departamento Acadêmico de Eletrônica (DAELN) outorgado pela Universidade Tecnológica Federal do Paraná (UTFPR). Os alunos Felipe dos Santos Neves e Giulia Heidi Bortoleto foram arguidos pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Curitiba, 27 de novembro de 2018.

_____
Prof. Dr. Robinson Vida Noronha
Coordenador de Curso
Engenharia Eletrônica

_____
Profª. Drª. Carmen Caroline Rasera
Responsável pelos Trabalhos de Conclusão de Curso
de Engenharia Eletrônica do DAELN

BANCA EXAMINADORA

_____
Prof. Dr. Bertoldo Schneider Junior
Universidade Tecnológica Federal do Paraná
Orientador

_____
Prof. Dr. Gustavo Borba
Universidade Tecnológica Federal do Paraná

_____
Esp. André Almeida Silva
Universidade Tecnológica Federal do Paraná

A Folha de Aprovação assinada encontra-se na Coordenação do Curso

Dedicamos este trabalho às nossas famílias, que sempre apoiaram nossos estudos e sem as quais nada disso seria possível. Também a todos aqueles que de alguma forma estiveram e estão próximos de nós, fazendo nossas vidas valerem cada vez mais a pena.

AGRADECIMENTOS

Agradecemos principalmente a nossas famílias e amigos por todo o carinho e apoio durante o desenvolvimento deste trabalho. Agradecemos também ao Prof. Bertoldo Schneider pela orientação e ao nosso colega Paulo Sérgio Schneider pelo desenvolvimento do cockpit utilizado no projeto. Por fim, um agradecimento especial ao Dr. Andre Almeida pelo aconselhamento e opiniões que continham a visão de um profissional da área onde o projeto será aplicado.

Agradecemos também à UTFPR e ao BIOTA pela disponibilização da estrutura utilizada para o desenvolvimento do nosso trabalho.

Resumo

NEVES, Felipe d. S.; BORTOLETO, Giulia H.. Sistema embarcado para a medição de força aplicada e tempo de resposta durante frenagem. 2018. 133 f. Trabalho de Conclusão de Curso – Curso de Engenharia Eletrônica, Universidade Tecnológica Federal do Paraná. Curitiba, 2018.

Médicos tem dificuldade em determinar o instante em que podem desmamar o paciente de seu processo pós-operatório de cirurgias no joelho por falta de critérios objetivos para a sua avaliação. Este projeto tem como objetivo o desenvolvimento de um sistema embarcado para executar testes de reflexo e força da flexão plantar e músculos posteriores de pacientes com mobilidade alterada, gerando arquivos com os dados e gráficos destes testes, com o objetivo de prover estes profissionais de dados que possam facilitar a avaliação dos seus pacientes. O sistema possui também uma interface amigável com o usuário para fácil utilização por médicos e pacientes.

Palavras-chave: Sistemas Embarcados. Ortopedia. Fisioterapia. Qualidade de Vida. Direção.

ABSTRACT

NEVES, Felipe d. S.; BORTOLETO, Giulia H.. Embedded system for the measurement of applied force and response time during braking. 2018. 133 f. Trabalho de Conclusão de Curso – Curso de Engenharia Eletrônica, Universidade Tecnológica Federal do Paraná. Curitiba, 2018.

Doctors have trouble determining the fitness to drive of patients with knee injury due to the lack of objective criteria for evaluation. This project has the purpose of developing an embedded system to perform plantar flexion and posterior muscles strength as well as refelx tests of patients with impaired mobility, generating files with data and graphs of the results, with the purpose of giving these professionals with that that eases the evaluation of their patients. The system also has a friendly user interface to provide easier interaction for doctors and patients.

Keywords: Embedded Systems. Orthopedics. Physiotherapy. Quality of life. Driving.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
UTFPR	Universidade Tecnológica Federal do Paraná
PET-EE	Programa de Educação Tutorial de Engenharia Eletrônica
SBC	Single Board Computer
LED	Light Emitting Diode
IDE	Integrated Development Environment
SDK	Source Development Kit
GUI	Graphical User Interface
Wi-fi	Wireless Fidelity
ADC	Analog Digital Converter
RAM	Random Access Memory
BIOTA	Laboratório de Biotelemetria Aplicada
HDMI	High-Definition Multimedia Interface
USB	Universal Serial Bus
LCA	Ligamento Cruzado Anterior

SUMÁRIO

1 INTRODUÇÃO

1.1 DELIMITAÇÃO DO TEMA

Este projeto visa resolver um problema existente no campo da ortopedia, que é o desenvolvimento de ferramentas adequadas que possibilitem o estabelecimento de critérios objetivos para a avaliação da capacidade de conduzir um automóvel em pacientes que sofreram lesões e subsequentemente foram submetidos a cirurgia, principalmente na região do joelho. No caso específico deste projeto, o objetivo é a captura de dados de força em um pedal ao longo do tempo para a atividade de frenagem de um automóvel.

1.2 MOTIVAÇÃO

Segundo o código de trânsito brasileiro, no parágrafo II, do capítulo 15 do artigo 252, é proibido dirigir com incapacidade física ou mental temporária que comprometa a segurança do trânsito (PERKONS, 2018). Apesar disso não é citado nenhum critério objetivo com o qual essa capacidade possa ser julgada.

Quando tratasse de pacientes com lesões no joelho, existe uma dificuldade em estabelecer parâmetros mais objetivos sobre o que caracteriza esta incapacidade física, o que faz com que o tema ainda seja tratado de forma branda na maioria dos guias e legislações. Além disso, não há informação baseada em evidência capaz de ajudar os médicos a tomarem decisões com respeito à capacidade física de direção de seus pacientes (FJ BYSZEWSKI AM, 2005).

É importante ressaltar que os médicos devem estar atentos com relação a sua responsabilidade perante tais situações, pois caso posteriormente um paciente que tenha sido liberado para dirigir perante sua orientação venha a se envolver em algum tipo de acidente, a responsabilidade pode cair sobre o próprio médico, cabendo penalidades legais, como suspensão do direito da prática da medicina e até mesmo penas mais graves (BERAN; DEVEREUX, 2017). Entretanto, a legislação e materiais de apoio disponíveis pouco informam de maneira a orientar quais são os critérios a serem considerados nessa tomada de decisão, limitando-se, na maioria dos casos, a recomendar tempo de recuperação de três meses e a necessidade de alertar o paciente sobre eventuais limitações. Assim, o julgamento fica inteiramente ao critério subjetivo do médico.

Atualmente, pessoas com flexão plantar e mobilidade alteradas dos músculos posteriores não têm métodos quantitativos de comprovar que estão aptas para dirigir, sendo feita somente uma avaliação médica geral com um especialista que avalia sensibilidade e evolução pós-cirúrgica geral do paciente. Esses testes não são completos e são altamente suscetíveis a erro. Esses pacientes também não tem registro de evolução, desestimulando o paciente na continuação do tratamento. A quantidade de pessoas que desistem de tratamentos fisioterapêuticos chegam a níveis tão altos como 45% (PRADO, 2010).

Entre as principais motivações para a permanência no tratamento, de acordo com os pacientes, é justamente a percepção da eficiência do tratamento ao qual estão sendo submetidos (CAMPBELL et al., 2001). Neste sentido, é importante viabilizar meios que possibilitem deixar mais claro a estes pacientes que o tratamento esta surtindo os efeitos desejados. Quando deixado apenas a cargo de critérios subjetivos fica mais difícil fazer tal demonstração.

1.3 OBJETIVOS

1.3.1 OBJETIVO GERAL

Espera-se que com esse projeto seja encontrado uma protótipo completamente funcional de um produto para o mercado consumidor, a fim de auxiliar médicos a obterem dados exatos e acessíveis para exames fisioterapêuticos.

1.3.2 OBJETIVO ESPECÍFICO

Partimos para o desenvolvimento deste projeto com a parte mecânica, do desenvolvimento do *cockpit* com o sensor de carga e pedal já prontos, portanto o escopo deste projeto seria o desenvolvimento de hardware para a correta aquisição dos dados da célula de carga e o desenvolvimento do software, que compreende o sistema de comunicação com o mecanismo de aquisição de dados, bem como a interface utilizada para a realização dos exames.

1.4 ESTRUTURA DO TRABALHO

O projeto foi dividido nas seguintes etapas: estudo dos recursos disponíveis do microcontrolador utilizado, estudo da linguagem a ser utilizada, estudo da fisiologia humana, realização de testes em um protótipo, desenvolvimento da interface com o usuário, integração das grandes partes do sistema, testes de validação.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta discussões a respeito do problema a ser solucionado, os critérios objetivos que caracterizem uma pessoa como apta a dirigir e as especificações da interface do equipamento a ser utilizado por médicos para a realização de exames.

2.1 IDENTIFICAÇÃO DO PROBLEMA

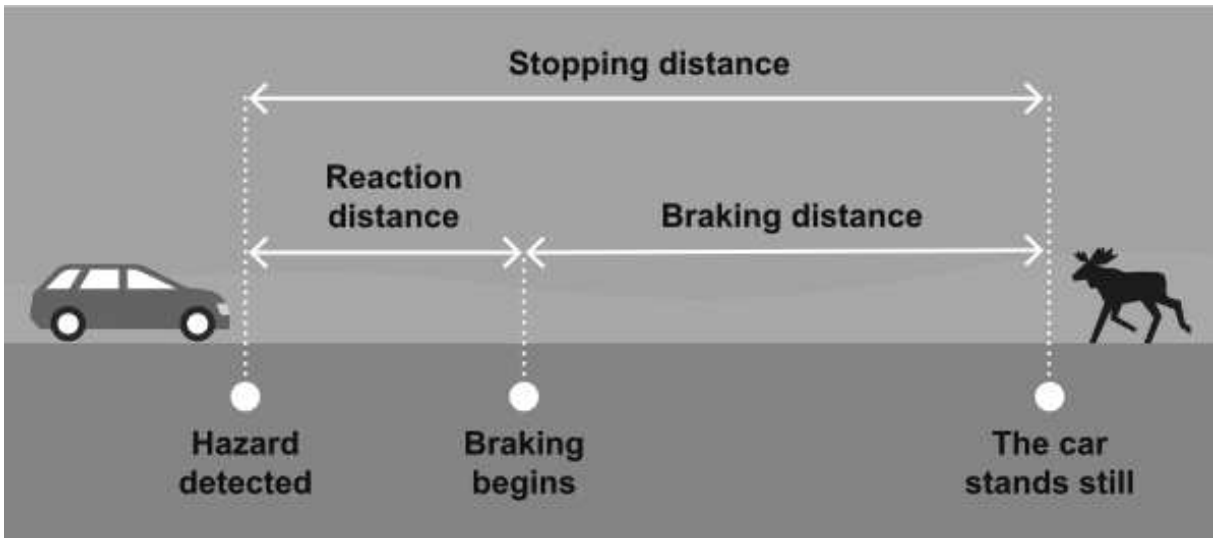
2.1.1 FRENAGEM DE VEÍCULO

Um dos principais aspectos que podem ser utilizados para se determinar a capacidade de um motorista para dirigir um automóvel é a sua capacidade de frear em situações de emergência.

Neste aspecto existem dois principais parâmetros a serem levantados, o tempo de reação e a força aplicada no pedal do freio. Esses parâmetros, por sua vez, dependem das condições da pista, velocidade dos automóveis e distância disponível para frenagem.

Para a avaliação do processo de frenagem diversos fatores devem ser levados em conta, pois o clima, a condição da estrada, do carro e do motorista contribuem de forma a alterar o tempo necessário para parar um carro. Entretanto, esses parâmetros podem ser simplificados de forma a refletir as condições normais de direção de um motorista normal. Assim geralmente temos dois parâmetros a serem considerados para o cálculo da distância de frenagem total, são eles o tempo de reação e o tempo de frenagem, como ilustrado na Figura 1.

Figura 1 – Diagrama de distância frenagem



Fonte: (KORKOT ONLINE, 2018)

A distância de reação é a distância percorrida entre o ponto em que o obstaculo é detectado até o inicio da frenagem. Essa distância de reação é afetada principalmente pela

velocidade inicial do carro e pelo tempo de reação do motorista. Esse tempo de reação é, para motoristas normais, entre 0,5 e 2 segundos. Vários outros fatores podem influenciar esse tempo de reação, como a capacidade de antecipação do motorista, a necessidade da tomada de decisão (frear ou desviar), efeito de substâncias químicas, cansaço, entre outros (KORKOT ONLINE, 2018).

A distância de reação pode ser calculada através da fórmula:

$$d = (v * r)/3.6 \tag{1}$$

Onde *d* é a distância de reação em metros, *v* é a velocidade inicial em km/h e *r* é o tempo de reação em segundos. Para um carro a 50 km/h e um tempo de reação de 1 segundo essa distância é de 13.9 metros.

A distância de frenagem tem uma conta um pouco mais complexa, pois além de considerar a velocidade do carro ainda deve-se levar em conta a desaceleração do automóvel, que pode variar de acordo com a força aplicada no pedal ao longo do tempo, bem como fatores externos como condição da pista e dos pneus e também o peso do veículo. Para cálculos estimados podemos usar a seguinte fórmula:

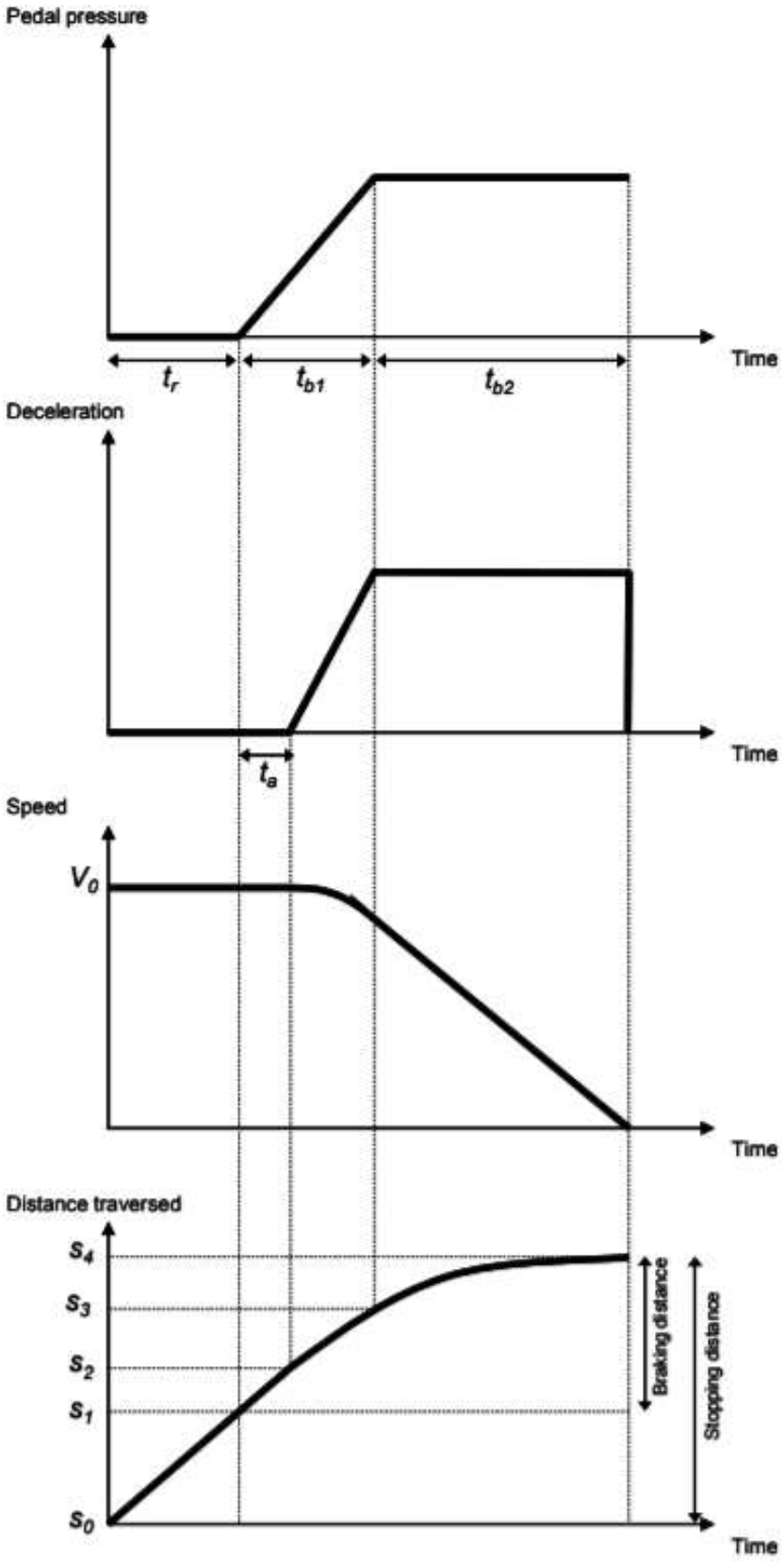
$$d = v^2/(250 * f) \tag{2}$$

Esta fórmula considera um veículo de passeio regular, em pista seca com boas condições e pneus também em boas condições. Nela *d* representa a distância de frenagem em metros, *v* é a velocidade inicial em km/h, *f* é o coeficiente de atrito, que é de aproximadamente 0.8 na pista seca e 0.1 na pista com gelo. Para o mesmo automóvel a 50 km/h em pista seca tem-se uma distância de frenagem de 12.5 metros.

O cálculo da distância de frenagem total é simplesmente a soma da distância de reação e da distância de frenagem. Para o exemplo do carro a 50 km/h essa distância é de 26.4 m, enquanto para um carro a 90 km/h nas mesmas condições essa distância seria de 55m. É claro que esses valores vão variar bastante na prática devido a presença desses diversos fatores externos, mas servem para se ter uma estimativa da ordem de distância necessária.

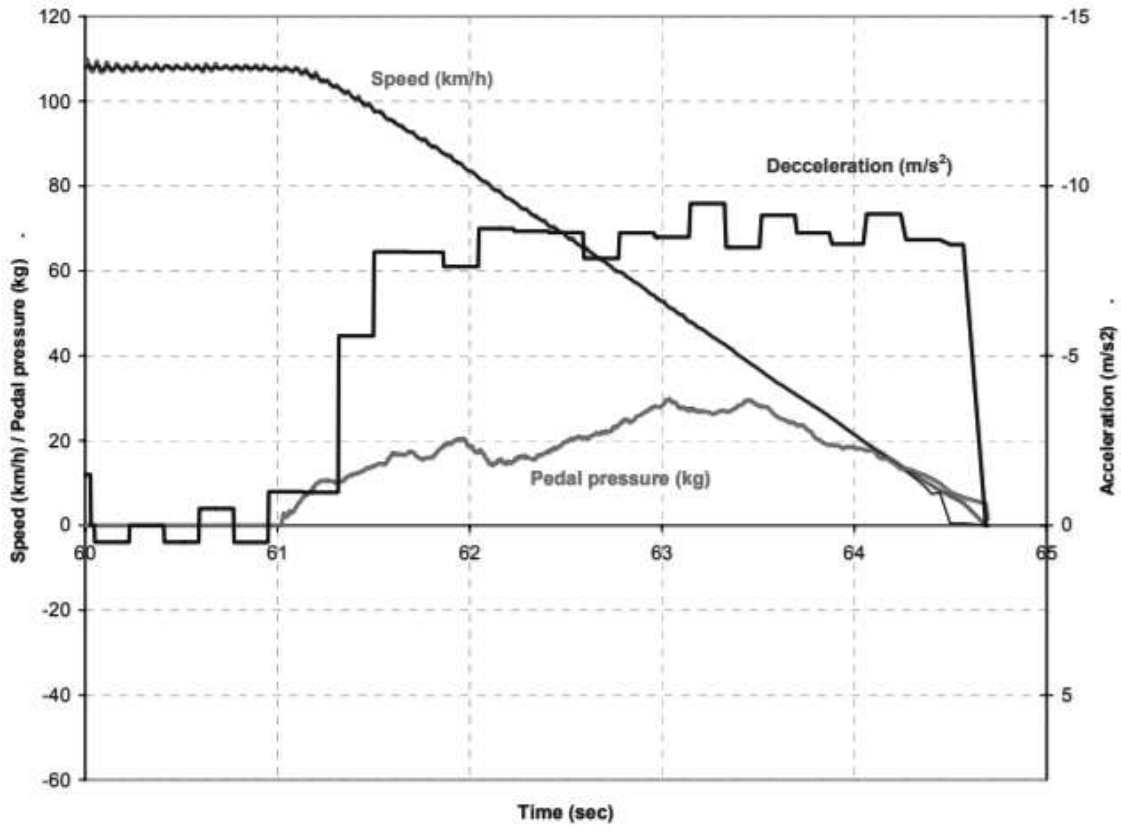
Em GREIBE (2007), um estudo sobre todos esses aspectos é feito e pode-se ver o resultado de um teste que levanta a relação da pressão sobre o pedal de freio, a velocidade de um automóvel e a distância percorrida. Tem-se um diagrama que mostra a relação destes parâmetros com o tempo na Figura 2, e também um exemplo prático de uma tentativa de frenagem, que relaciona todos esses parâmetros em um gráfico na Figura 3, no qual é possível observar dois parâmetros importantes, o tempo de reação, que é de 1 segundo, e a força aplicada ao pedal, que chega a 28 kgf.

Figura 2 – Diagramas de tempos na frenagem



Fonte: (GREIBE, 2007)

Figura 3 – Exemplo de frenagem

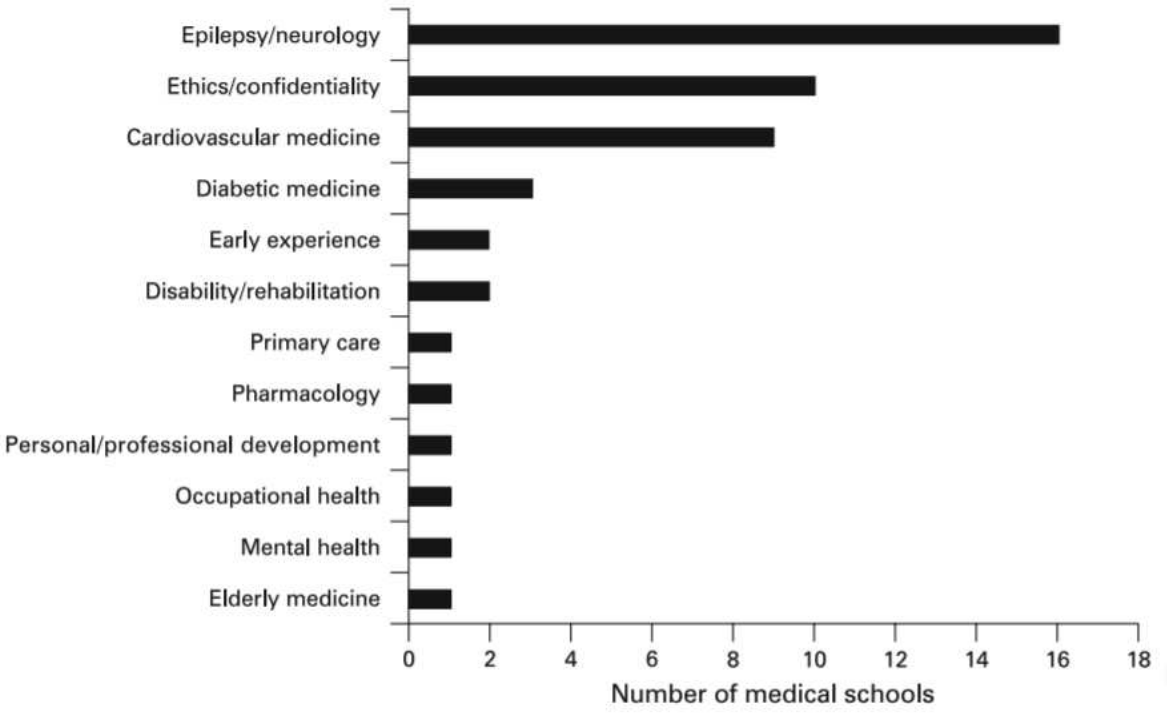


Fonte: (GREIBE, 2007)

2.1.2 RECUPERAÇÃO DE LESÕES

Durante o levantamento bibliográfico para a produção deste trabalho foi constatado grande insuficiência em material relacionado ao problema específico de recuperação cirúrgica. Entretanto, há uma quantidade relativamente boa de material referente à capacidade de direção em idosos, de onde podem ser traçados muito paralelos com o objetivo, visto que pacientes idosos também sofrem de restrições físico-motoras que apresentam características semelhantes a aquelas de pacientes em recuperação cirúrgica. Em ambos os casos o sistema desenvolvido pode ser aplicado no monitoramento da recuperação dos pacientes durante o decorrer de sessões de fisioterapia. Um levantamento feito no Reino Unido também ilustra bem a situação no viés do ensino, onde uma pesquisa nos centros de ensino de medicina do país mostram que a capacidade física de direção de pacientes é ensinada majoritariamente nos campos relacionados à neurologia, sendo que o campo de reabilitação é de certa forma negligenciado, como é possível observar na Figura 4, onde é listada a quantidade de faculdades de medicina onde o assunto é abordado em cada um dos diferentes tópicos, e a área de reabilitação só é citada em 2 das 32 faculdades envolvidas no estudo (HAWLEY; GALBRAITH; DESOUZA, 2008).

Figura 4 – Gráfico dos tópicos considerados relevantes para definir a capacidade de dirigir de pacientes nas faculdades de medicina do Reino Unido



Fonte: (HAWLEY; GALBRAITH; DESOUZA, 2008)

Pacientes que sofreram lesões e foram submetidos a procedimentos cirúrgicos podem ter sua capacidade motora severamente afetada. Neste sentido, lesões na perna e joelhos podem prejudicar a capacidade de direção de um motorista. Nestas situações cabe ao seu médico ortopedista avaliar se o paciente está apto ou não a voltar a dirigir.

Nesse âmbito, um dos principais procedimentos realizados é a cirurgia de reconstrução do ligamento cruzado anterior (LCA). Apenas nos Estado Unidos, chegam a ser realizados mais de cinquenta mil procedimentos por ano (INSALL; SCOTT, 2001).

É estimado que pacientes submetidos a este tipo de procedimento tenham um tempo de recuperação de até 6 semanas para que sejam considerados aptos a voltar a dirigir com segurança, ou seja, que tenham a capacidade de frear em uma situação de emergência (NGUYEN; HAU; BARTLETT, 2000).

A maior dificuldade neste sentido é justamente a indisponibilidade de ferramentas para se analisar objetivamente estes parâmetros, e é nesse sentido que o desenvolvimento do projeto é realizado, buscando oferecer uma ferramenta que possibilita um acompanhamento dos pacientes ao longo do tempo para avaliar a sua recuperação.

2.1.3 USABILIDADE DO SOFTWARE

Ao desenvolver um sistema para ser utilizado neste tipo de aplicação médica ainda é muito importante se considerar sua usabilidade. O sistema será operado por um médico ou fisioterapeuta que podem ter diferentes idades e experiência com computadores. Logo, é necessário desenvolver uma interface intuitiva e que ofereça os recursos necessários para a obtenção correta dos dados.

Existem vários critérios utilizados para tentar definir o que é um bom *software* para a área de saúde, dentre estes, os considerados mais relevantes para o desenvolvimento do nosso projeto estão listados abaixo (QUINTIN; THIZY, 2018):

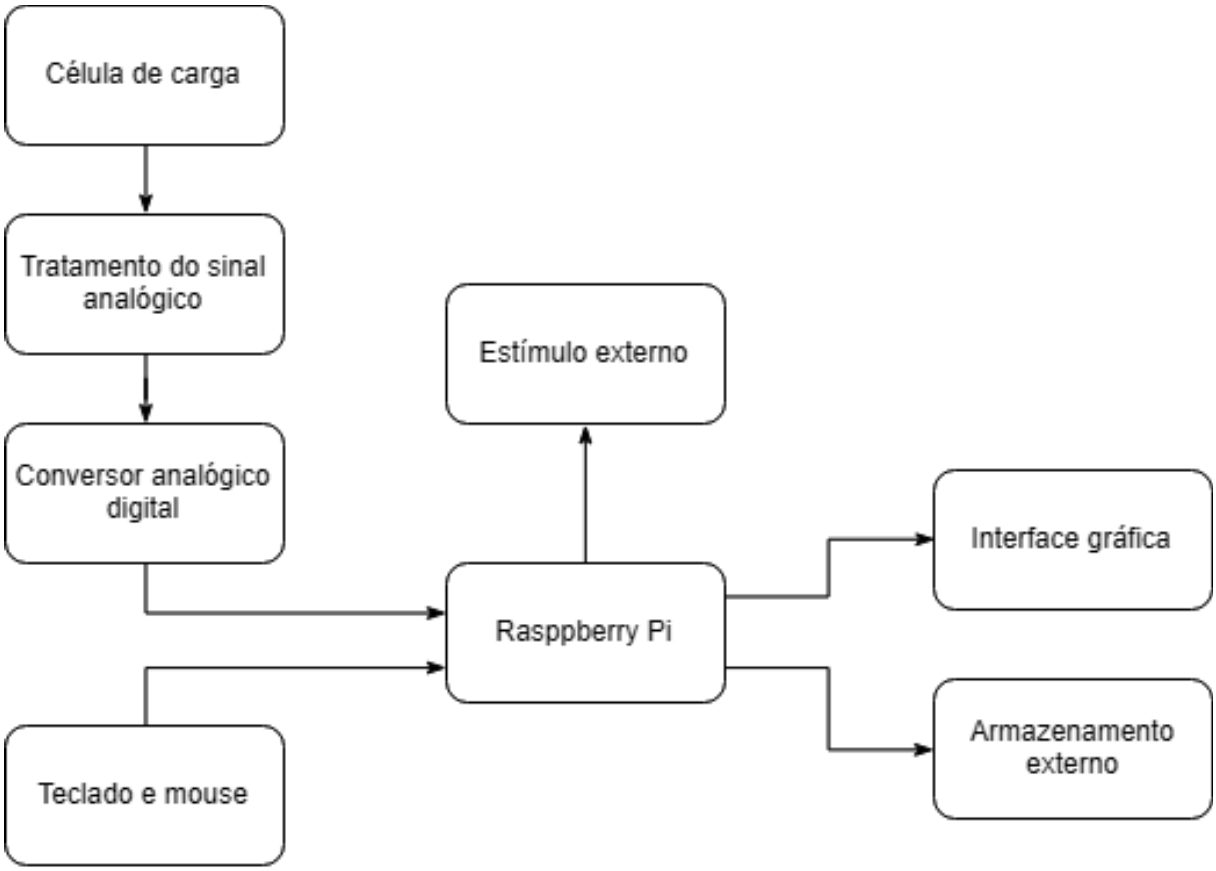
- Validação de dados de entrada não deve bloquear o usuário:
  Softwares médicos muitas vezes requerem a entrada de uma grande quantidade de dados, porém a validação destes dados não pode bloquear o usuário de maneira a impedir de realizar algum procedimento médico, como impedir a realização de um exame por deixar algum campo de identificação vazio.
- Médicos não têm paciência para *software* com grande curva de aprendizagem:
  Desenvolvedores de *software* nem sempre levam em consideração as restrições de tempo impostas sobre os profissionais da área da saúde. Eles precisam completar suas tarefas o mais rápido possível, e muitas vezes o *software* pode atrasar ou complicar demais os seus procedimentos. Os médicos geralmente não têm condições de dispender tempo com isso e a tendência é que esses *software*s eventualmente caiam em desuso caso não ofereçam alguma vantagem clara ao profissional de saúde.
- Requerimentos devem ser desenvolvidos especificamente para a especialidade do médico:
  Desenvolvedores tendem a não considerar o nível de especificidade envolvido em cada prática médica. Um substancial número de programas é desenvolvido com o propósito de servir para todos ou ser de uso geral, porém esse *software* acaba inevitavelmente frustrando o profissional de saúde por não atender às suas necessidades. O levantamento destes requisitos é uma das partes mais importantes, e por muitas vezes uma das mais complicadas no processo de desenvolvimento, pois nem sempre os dados necessários para levantar tais requisitos estão facilmente acessíveis.

3 METODOLOGIA

3.1 VISÃO GERAL DO SISTEMA

A Figura 5 apresenta o diagrama em blocos com a visão geral do sistema desenvolvido. As principais funcionalidades são a aquisição de dados de forma simplificada, o escopo do projeto é a aquisição de dados de tensão analógica, provenientes de uma célula de carga, exibição destes dados em uma interface gráfica e a geração de um relatório ao final da sessão de exames, a ser salvo em um dispositivo de armazenamento externo, além de gerar um estímulo externo para informar o paciente do início da captura dos dados.

Figura 5 – Visão geral do sistema desenvolvido

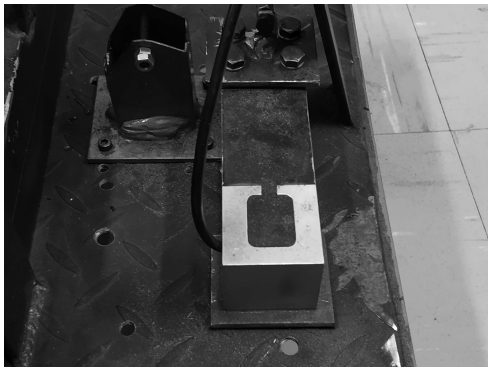


Fonte: Autoria própria

O *cockpit* com as células de carga instaladas, mostrado na Figura 6, já estava à disposição da equipe. Assim, conforme mencionado anteriormente, o escopo deste trabalho envolve principalmente a aquisição dos dados da célula de carga e a elaboração do software. É importante destacar que o software é utilizado pelo médico administrados das sessões de exame. Desta forma, é um requisito para o software ser de fácil utilização, responsivo e intuitivo, exigindo o mínimo de interação por parte do médico para a realização dos exames, ao mesmo

tempo que oferece ferramentas suficientes para a entrada de dados e configuração adequada do exame.

(a) Vista lateral do *cockpit*



(b) Célula de carga instalada no *cockpit*

Figura 6 – Fotos do *cockpit*

3.2 EXECUÇÃO

O desenvolvimento deste projeto passou por duas fases distintas. Inicialmente o grupo PET-EE juntamente com a orientação do Prof. Bertoldo já estava fazendo o desenvolvimento de um sistema para a aquisição e exibição dos dados. Esse sistema era composto de um Arduíno, responsável pela coleta dos dados, conectado via USB a um computador, que então executaria uma aplicação responsável por exibir os dados e gerar relatórios. Quando nossa equipe assumiu o projeto nenhum desses itens estava funcional e então inicialmente trabalhou-se no sentido de continuar a plataforma já existente.

Porém, durante o desenvolvimento, e após já termos uma versão funcional do projeto foi considerado interessante explorar a possibilidade de embarcar de forma integral essa solução, de forma a eliminar a necessidade do uso de um computador.

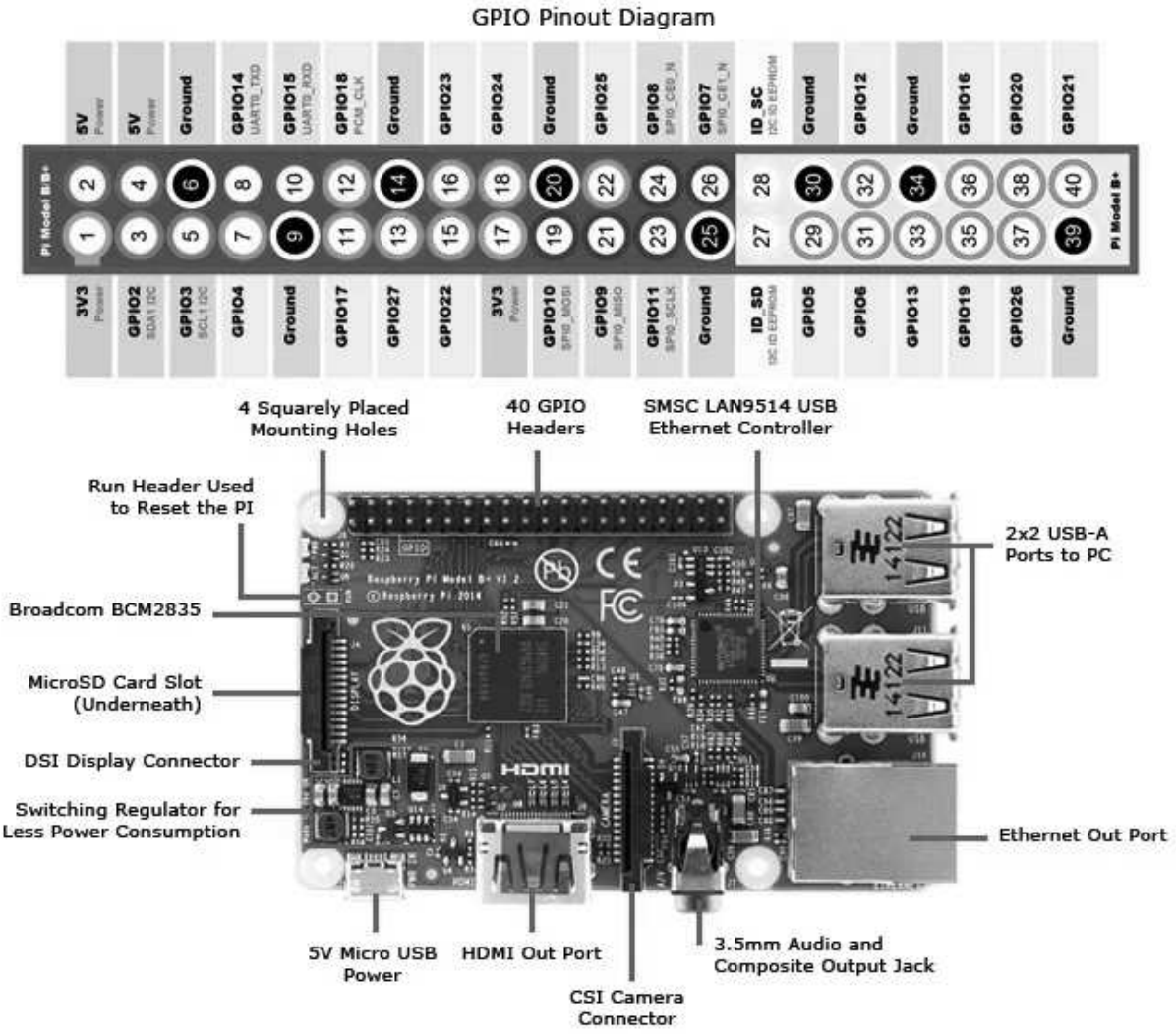
3.2.1 HARDWARE

A escolha do *hardware* muitas vezes se torna um dos aspectos mais complexos na tomada de decisão para o desenvolvimento de um projeto, isso se deve ao fato de muitas vezes representar um significativo investimento inicial e também porque eventuais mudanças de escopo durante o processo de desenvolvimento podem não ser possíveis ou terem um impacto muito severo dependendo do *hardware* inicialmente escolhido, muitas vezes se tornando necessária a aquisição ou troca completa do *hardware* atualmente em uso.

3.2.1.1 SBC

A proposta inicial do hardware era composta por um microcontrolador *Arduino UNO*, que é fácil de programar, barato e amplamente disponível no mercado nacional. Porém, esta é

Figura 7 – Placa Raspberry Pi model 3B



Fonte: Aspencore (2018)

composta por microcontrolador e eletrônica de apoio simples e não seria capaz de embarcar um sistema completo compreendendo a aquisição e exibição dos dados.

Foi decidido então por utilizar um SBC (Single Board Computer). Um SBC é um computador completo embarcado em uma única placa, diferente de computadores tradicionais, que requerem diversas partes e módulos separados (memória, processador, placa de vídeo) para para operar, os SBC's estão prontos para operar por si só, pois já contem todos os componentes necessários em sua placa.

A principal vantagem ao se escolher esta plataforma de desenvolvimento, ao contrário do Arduíno por exemplo, é a sua capacidade de integração, pois podemos ter todos os módulos do sistema operando em uma única placa, pois esses computadores têm uma capacidade de processamento muito elevada quando comparamos com os microcontroladores. A abundância de recursos também possibilita uma versatilidade muito grande ao projeto, pois podemos facilmente adicionar recursos como *Wi-Fi* ou *Bluetooth*, uma vez que o mesmo já possuí o

hardware necessário na placa. Em contrapartida, os SBC's costumam operar baseados em sistemas operacionais robustos, como o Linux, o que pode trazer problemas para a operação em tempo real ou para a aquisição de dados em alta frequência, mas esse não é o nosso caso.

A escolha de um SBC para o projeto foi relativamente simples, tomando como base os fatores preço e disponibilidade no Brasil a única opção viável é o Raspberry Pi, no seu modelo 3B, mostrado na Figura 7, por ser a placa mais fácil de encontrar para venda no Brasil, e contar com um preço altamente competitivo, cerca de 150 a 200 reais, não ficando muito mais caro que o próprio Arduino UNO.

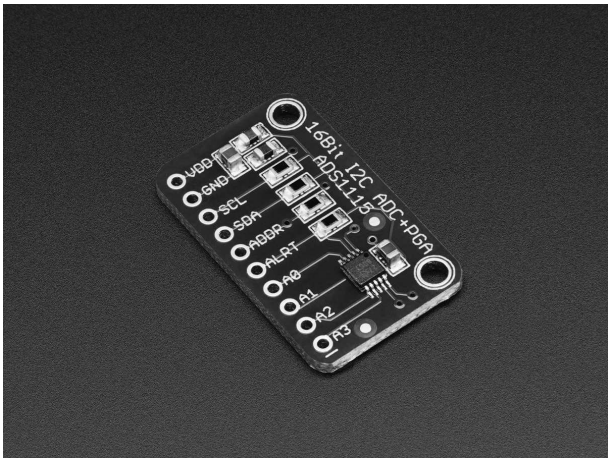
Em seu modelo 3B, a placa do Raspberry conta com um processador quad-core de 64 bits operando a 1.2GHz e 1GB de RAM, chip de Wi-fi, entrada Ethernet, 4 portas USB 2.0, saída HDMI e 40 pinos de entrada, além de outros recursos que não utilizamos no projeto, como bluetooth, saída de áudio e entrada para câmera (RASPBERRY. . . , 2018).

3.2.1.2 AQUISIÇÃO DE DADOS

O sistema opera baseado nos dados coletados de um sensor de força de pisada que mede forças entre 0 kgf e 50 kgf, gerando tensões na faixa de 0.25V a 0.275V. Para capturar esses dados com o Raspberry Pi é necessário um conversor analógico digital externo, pois o Raspberry possuí várias entradas saídas digitais, mas nenhuma analógica. Em contrapartida o Raspberry Pi oferece conectividade SPI e I2C, como vemos na Figura 7, que pode ser usada para conectar a um ADC externo.

O ADS1115, modelo de ADC adotado no projeto, possui 16 bits de resolução e ganho de configurável de até 16 vezes. Ele é fabricado pela Texas instruments, mas também possuí um módulo, apresentado na Figura 8, que é comercializado pela Adafruit, e já vem pronto para ser conectado ao Raspberry ou outro controlador através da interface I2C.

Figura 8 – Módulo ADS1115 da Adafruit

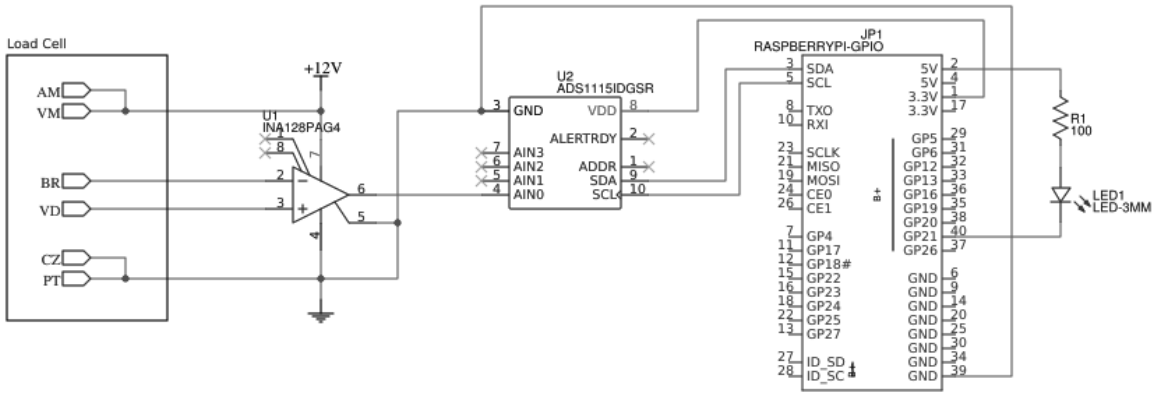


Fonte: Adafruit Industries (2018)

De forma a conectar o sinal da célula de carga ao ADC também foi adicionado um

estágio com um amplificador operacional de instrumentação, o INA128P, funcionando como um *buffer*, com ganho unitário, com o propósito de isolar os circuitos assim como eliminar ruído de modo comum proveniente da saída da célula de carga. O circuito é apresentado na Figura 9.

Figura 9 – Esquemático do circuito para aquisição dos sinais da célula de carga



Fonte: Autoria própria

3.2.1.3 CÉLULA DE CARGA

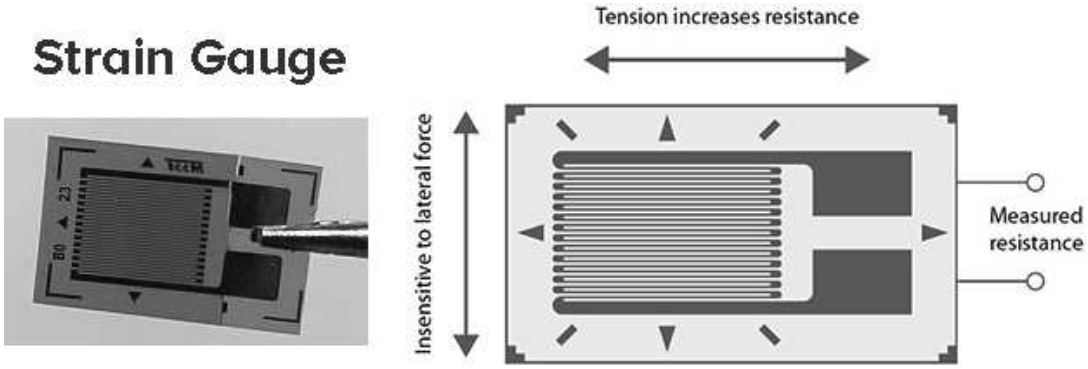
Uma célula de carga é um transdutor que converte força em tensão elétrica. Seu principio de funcionamento se baseia na variação da resistência ôhmica de um sensor denominado *strain gauge*, mostrado na Figura 10. Geralmente são conectados 4 destes na forma de uma ponte de Wheatstone, como na Figura 11, alimentada por uma tensão, usualmente no valor de 12V. O variação da resistência entre eles causa um desequilíbrio nesta ponte de Wheatstone que gera a tensão que desejamos medir Straightpoint (2018). A célula de carga adotada no projeto foi o modelo GL-50 da Alfa Instrumentos, as especificações da mesma são disponibilizadas em Alfa Instrumentos (2018).

O ADC é utilizado para coletar os dados de uma célula de carga, esta é a célula que está acoplada a um pedal no *cockpit* na qual serão realizados os testes.

Com o levantamento dos dados de tensão da célula foi obtido o gráfico da Figura 12, do qual podemos levantar uma equação linear utilizada para converter os dados de tensão para força, que serão efetivamente registrados pelo sistema, dado pela equação (3). Na equação, V representa a tensão gerada na saída da célula de carga em Volts, enquanto F é a força total aplicada sobre a mesma em quilograma-força.

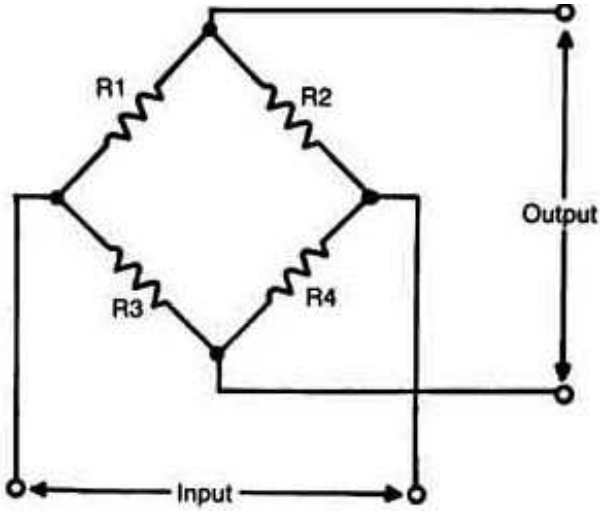
$$F = 0{,}00051096 * V + 0{,}25124 \tag{3}$$

Figura 10 – Strain gauge



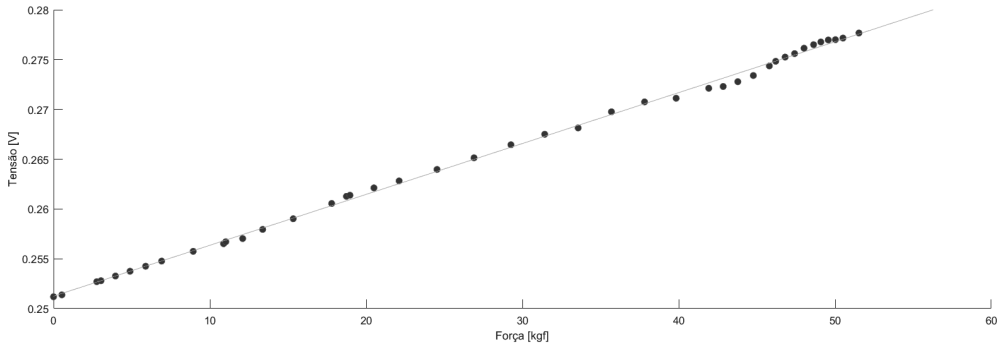
Fonte: (STRAIGHTPOINT, 2018)

Figura 11 – Ponte de Wheatsone



Fonte: (STRAIGHTPOINT, 2018)

Figura 12 – Curva de resposta da célula de carga

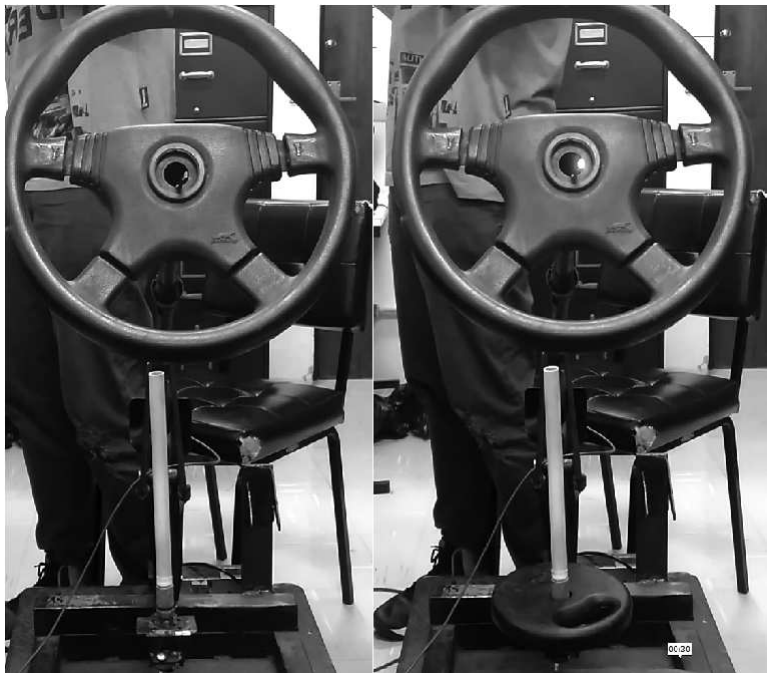


Fonte: Autoria própria

3.2.1.4 ESTÍMULO VISUAL

Conforme mostrado anteriormente na Figura 9, foi adicionado um LED (diodo emissor de luz), com o propósito de servir como estímulo visual Figura 13. O propósito deste estímulo é permitir a sincronização do tempo da captura dos dados, de forma a possibilitar a medição do tempo de reação do paciente.

Figura 13 – Estímulo visual gerado pelo LED antes do teste iniciar (esq.) e após o início do teste (dir.)



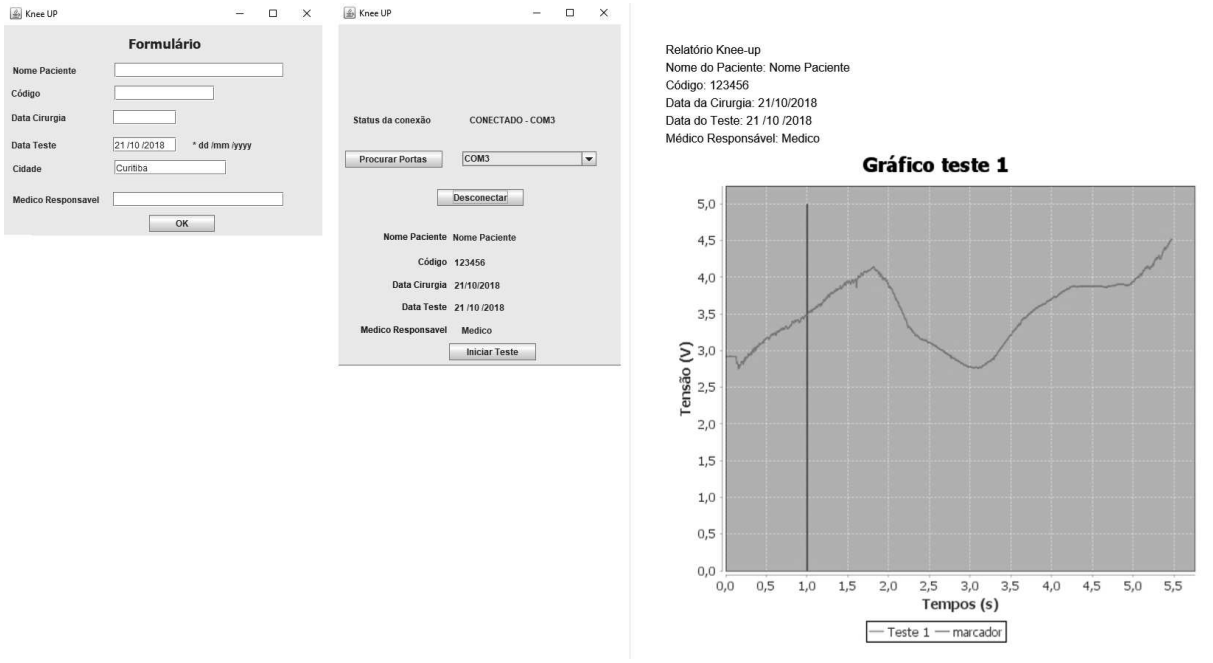
Fonte: Autoria própria

3.2.2 SOFTWARE

Quando iniciamos o desenvolvimento do projeto já existia uma versão inicia incompleta do projeto, cedida pelo professor Bertoldo, que utilizava um *Arduino* e uma aplicação em *Java* no computador. A interface desta versão é mostrada na Figura 14. Ela permitia um cadastro simples do paciente, aquisição dos dados com falhas e um relatório em .pdf, porém tinha muitos problemas de funcionamento e replicabilidade, como por exemplo só reconhecia o *Arduino* em certos computadores e a escolha da porta tinha que ser realizada pelo usuário. Inicialmente foi complementada essa versão com uma UI mais simples, correção dos erros e melhor relatório, como mostrado na Figura 15, porém chegou-se a alguns impedimentos, como geração de gráfico somente com atraso e desempenho/biblioteca gráfica limitados.

O software desenvolvido posteriormente foi baseado nesse projeto inicial, porém reimplementado em linguagem *Python* para se adequar as ferramentas e bibliotecas disponíveis a serem utilizadas no ambiente Linux adotado no Raspberry Pi. A linguagem Python é

Figura 14 – Tela principal, tela de confirmação e relatorio final (dir. para esq.) da primeira versão do Software



Fonte: Autoria própria

Figura 15 – Tela principal, tela de confirmação e relatorio final (dir. para esq.) da segunda versão do Software

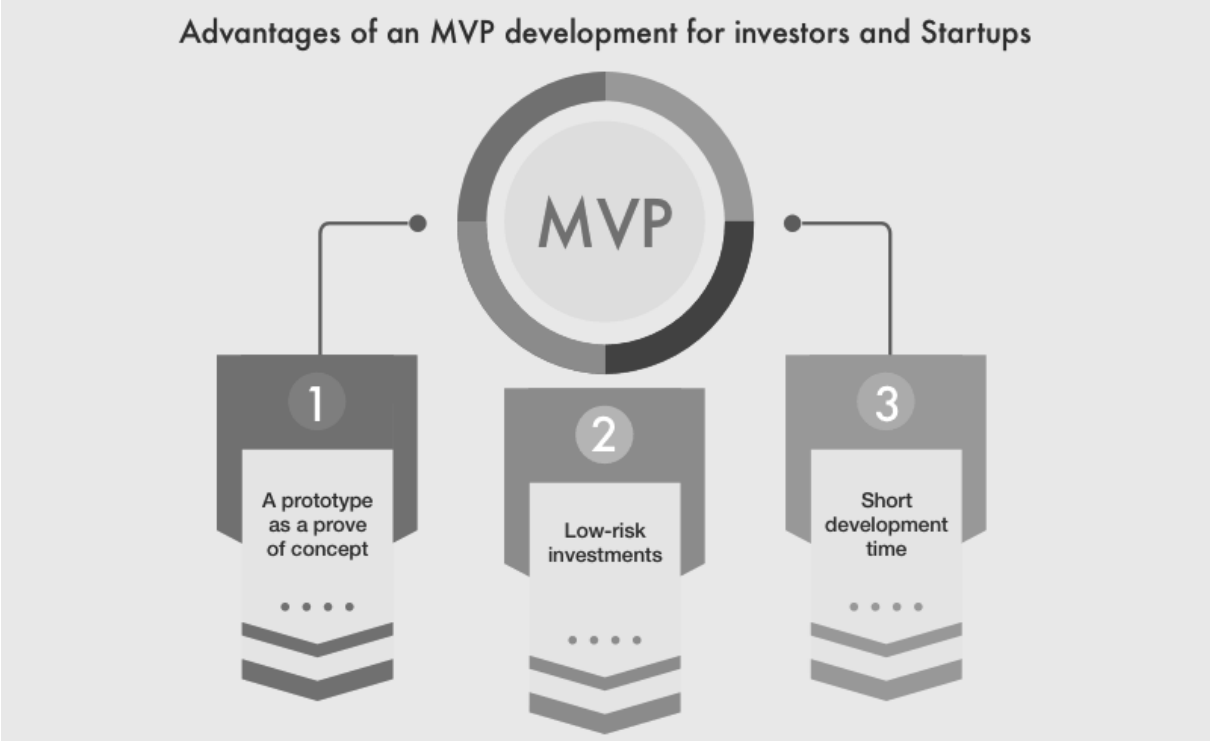
Fonte: Autoria própria

considerada mais lenta que outras como C, C++ e Java, porém a escolha dela foi devido às facilidades que ela oferecia, como as bibliotecas gráficas pré existentes, assim como por ser uma linguagem mais fácil de se trabalhar de forma geral. Como estávamos trabalhando com um SBC de desempenho adequado para o projeto (Raspberry Pi), foi possível adotar a linguagem Python.

O Python é uma das linguagens mais utilizadas pelas *startups*devido à sua flexibilidade, facilidade e disponibilidade de integração com outras plataformas. Notavelmente a Google é conhecida por disponibilizar suas plataformas em Python. Por isso, é muito fácil e rápido desenvolver aplicações completamente funcionais com Python (VALUECODERS, 2018).

Na Figura 16 fica bem ilustrada essa ideia, no conceito de MVP (Minimum Viable Product), o produto mínimo viável, cujas vantagens são dadas pelo desenvolvimento de um protótipo como prova de conceito com investimento de baixo risco e curto tempo de desenvolvimento. Esses são elementos que se encaixam justamente na proposta de desenvolvimento do nosso trabalho.

Figura 16 – MVP Python



Fonte: ValueCoders (2018)

3.2.2.1 LINUX EMBARCADO

Onde antigamente sistemas embarcados ficavam restritos a aplicações sem o uso de sistemas operacionais, ou com sistemas operacionais extremamente leves, com o rápido desenvolvimento de processadores cada vez mais rápidos e baratos fica cada vez mais fácil embarcar um sistema operacional completo e robusto como o Linux.

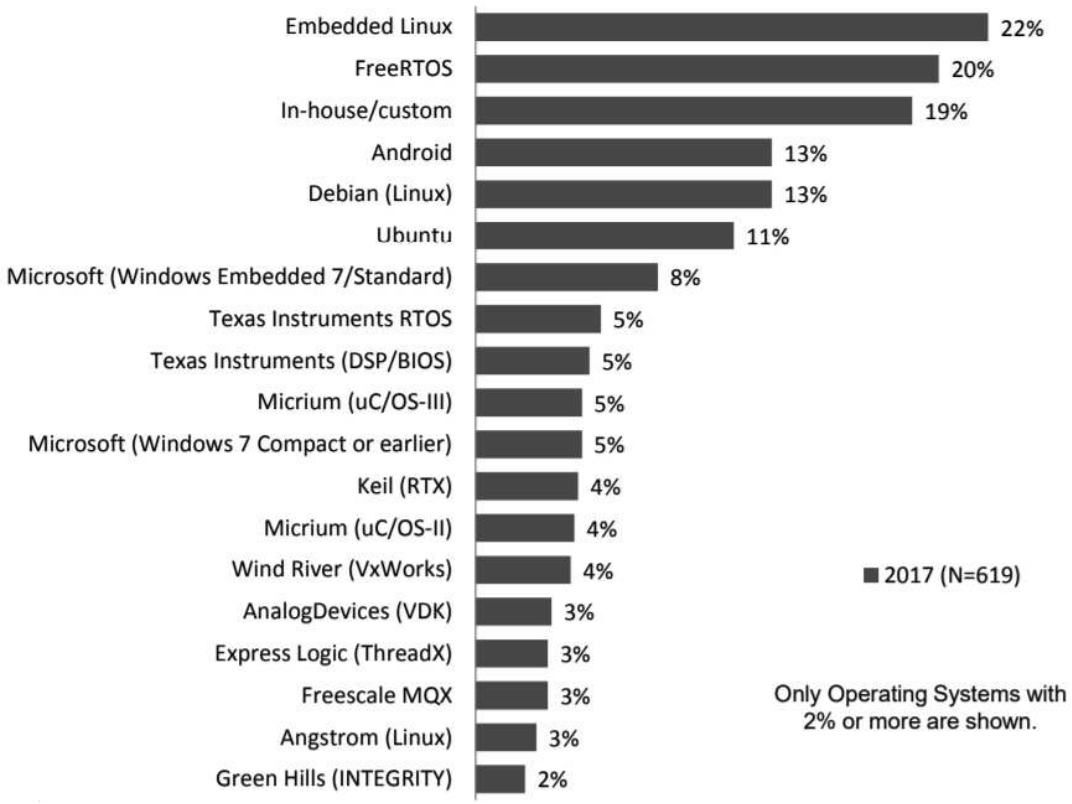
A utilização do Linux embarcado em um projeto traz muitas vantagens, pois o sistema por si só já oferece diversos recursos e a possibilidade de acoplar diversas bibliotecas e aplicações pré-existentes, assim como integrar todos esses recursos de maneira fácil. Não se faz necessário reimplementar bibliotecas para cada recurso ou periférico específico, mas apenas integrar os módulos existentes.

Todas essas vantagens vem acopladas a um alto custo de processamento demandado ao executar todos esses recursos em paralelo, porém plataformas como o Raspberry Pi são mais do que capazes de atender ao processamento demandado. Como contrapartida porém

ainda temos um alto consumo de energia, que pode chegar a um máximo de 2.5A, alimentado com 5V, no caso do Raspberry Pi, isso em situação de uso completo de todos os seus recursos (portas e processamento). Porém esse consumo ainda é muito baixo ao ser comparado com o consumo de um computador desktop tradicional.

Um estudo de mercado realizado pela Aspencore Maxfield (2017) mostra também que o Linux já é o sistema operacional mais utilizado em sistemas embarcados, assim como a própria Raspberry Pi aponta como uma das principais placas utilizadas no desenvolvimento de aplicações embarcadas, conforme vemos nas figuras Figura 17 e Figura 18.

Figura 17 – Fatia de mercado do Linux embarcado

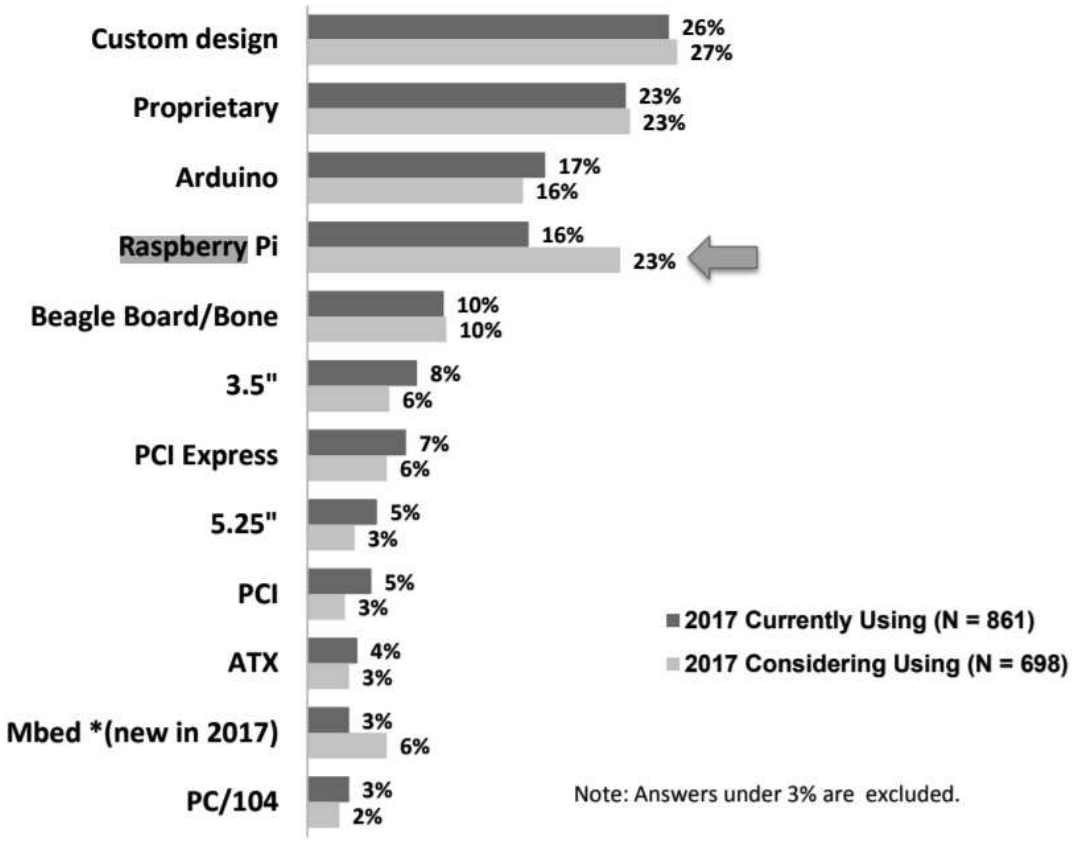


Fonte: Maxfield (2017)

3.2.2.2 CONTROLE DE VERSIONAMENTO

Projetos de software normalmente são desenvolvidos com o auxílio de ferramentas de versionamento. Essas ferramentas fornecem a opção de *merge* que, através do uso de conceitos incrementais, mesclam arquivos de diferentes fontes em um arquivo base. Isso permite que múltiplos desenvolvedores trabalhem nos mesmos arquivos simultaneamente. Além disso, esse tipo de ferramenta também realiza o armazenamento de todas as alterações passadas. Logo, facilita a resolução de problemas para casos em que uma alteração nova possa fazer com que o código pare de funcionar, por exemplo.

Figura 18 – Intenção de uso de diferentes plataformas



Fonte: Maxfield (2017)

Existem muitas plataformas para serem utilizadas, a mais popular delas sendo o GitHub. Neste projeto a ferramenta escolhida foi o Git, através da plataforma Gitlab por oferecer a possibilidade de criar um repositório privado gratuitamente. O código do projeto então fica todo disponível no repositório https://gitlab.com/feupos/ExaminationManager.

3.2.2.3 AMBIENTES DE DESENVOLVIMENTO

Uma das principais partes do projeto é o desenvolvimento de uma interface gráfica (GUI), pois o objetivo do projeto é que ele seja utilizado por um médico fisioterapeuta durante sessões de exames. Portanto, é crucial que o *software* seja acompanhado de uma interface de fácil utilização e que seja responsiva aos comandos. Para o desenvolvimento dessa interface buscamos uma plataforma que pudesse atender a dois requisitos: ser programável em Python e poder ser executada em ambiente Linux.

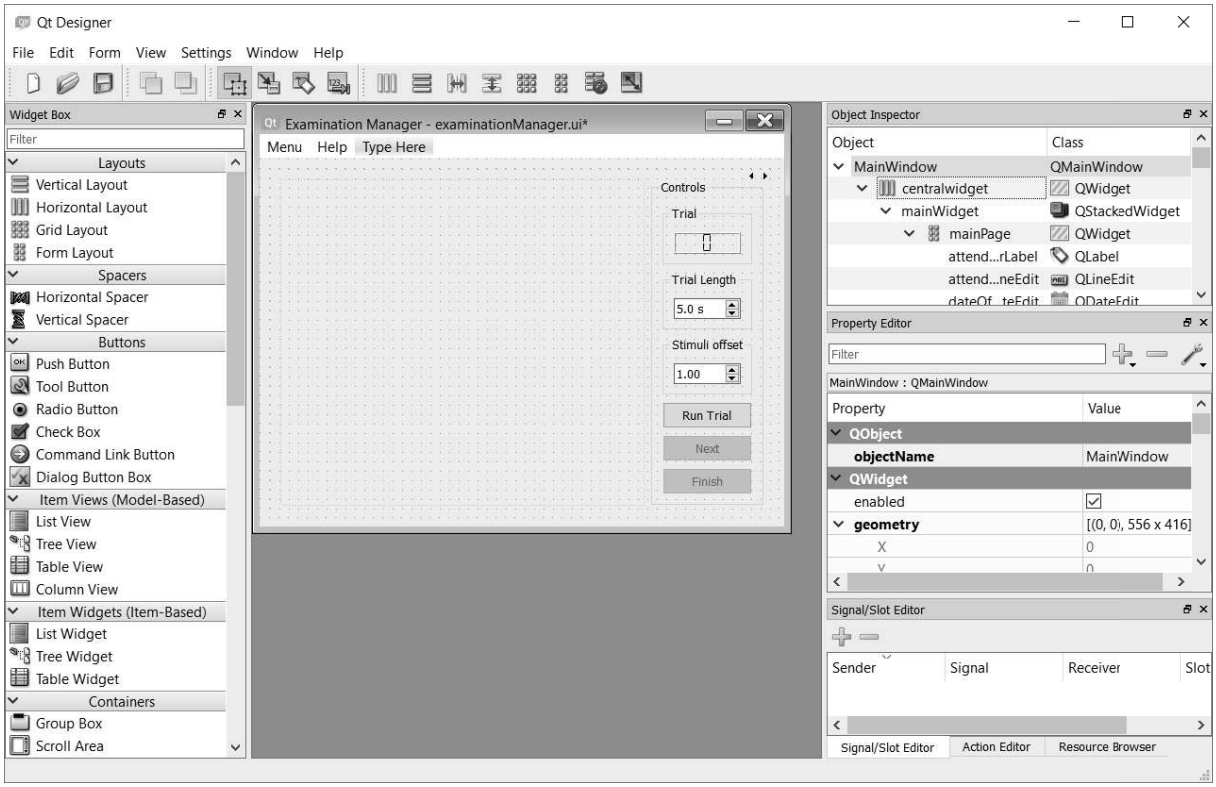
Para isso, foi escolhida a *source development kit* (SDK) Qt 5, que é uma SDK multi plataforma, que assim como o Python, pode ser executada tanto em Windows, como Linux ou até mesmo dispositivos móveis. O Qt também já é amplamente utilizado na indústria atualmente.

Uma das principais vantagens de se trabalhar com tecnologias multi-plataforma é

que isso nos possibilitou testar o código em sua maior parte apenas nos computadores de desenvolvimento, que utilizavam Windows, antes de passar o código para a placa do Raspberry Pi, isso foi importante pois a equipe possuía apenas uma placa, que não estava sempre disponível para todos, portanto era importante poder realizar testes em outra plataforma sem utilizar a placa.

Para o desenvolvimento desta GUI em Qt foi utilizado o software Qt Designer, onde é possível fazer o desenvolvimento de toda a parte gráfica de forma interativa, com recursos de *drag-and-drop* (selecionar os itens desejados e posiciona-los na interface gráfica). É uma ferramenta bastante robusta e fácil de utilizar. Após finalizada a interface é possível gerar código automaticamente em C++, ou no nosso caso utilizar ferramentas da biblioteca em Python para converter para Python. Na Figura 19 é mostrada uma das telas da interface de desenvolvimento.

Figura 19 – Interface de desenvolvimento em Qt



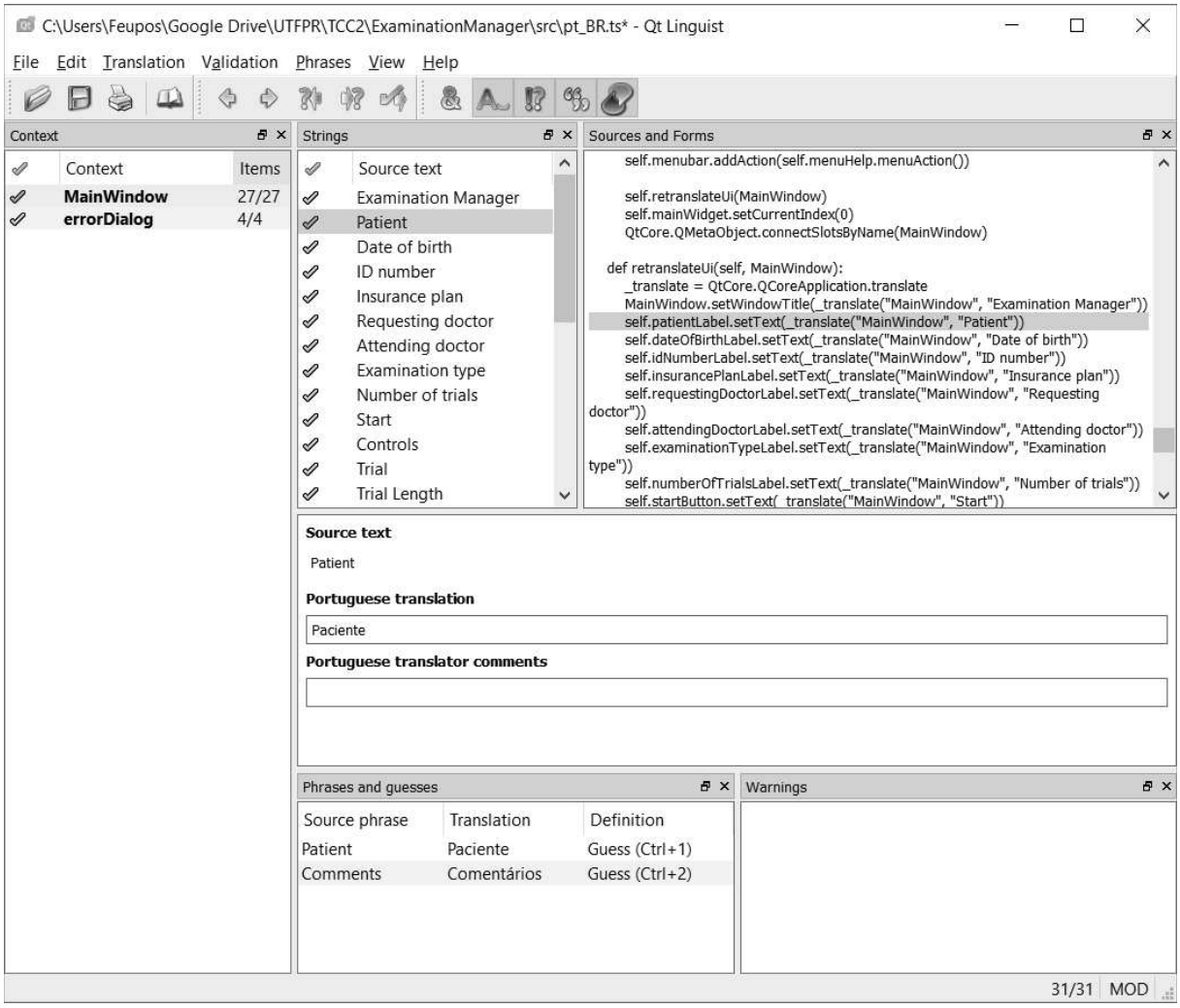
Fonte: Autoria própria

Com o código gerado podemos então customizar e controlar a interface de maneira mais completa, editando campos interativamente e anexando funções aos botões.

Também foi utilizada a ferramenta Qt Linguist, que serve para auxiliar a tradução do aplicativo, permitindo que o mesmo tenha todas as suas *strings* traduzidas de acordo com o arquivo de tradução selecionado. Essa ferramenta foi utilizada de forma a fazer todo o desenvolvimento do aplicativo em inglês, visando sua internacionalização, porém também havendo a possibilidade de selecionar o idioma português, facilitando sua aplicação na situação

de usuários que não falem inglês. Isso também torna futuramente a adição de novos idiomas simples e rápida. O programa, mostrado na Figura 20, consegue detectar todas as *strings* inseridas no formato *_translate("escopo","texto")* e então podemos configurar manualmente a tradução desejada.

Figura 20 – Interface do Qt Linguist



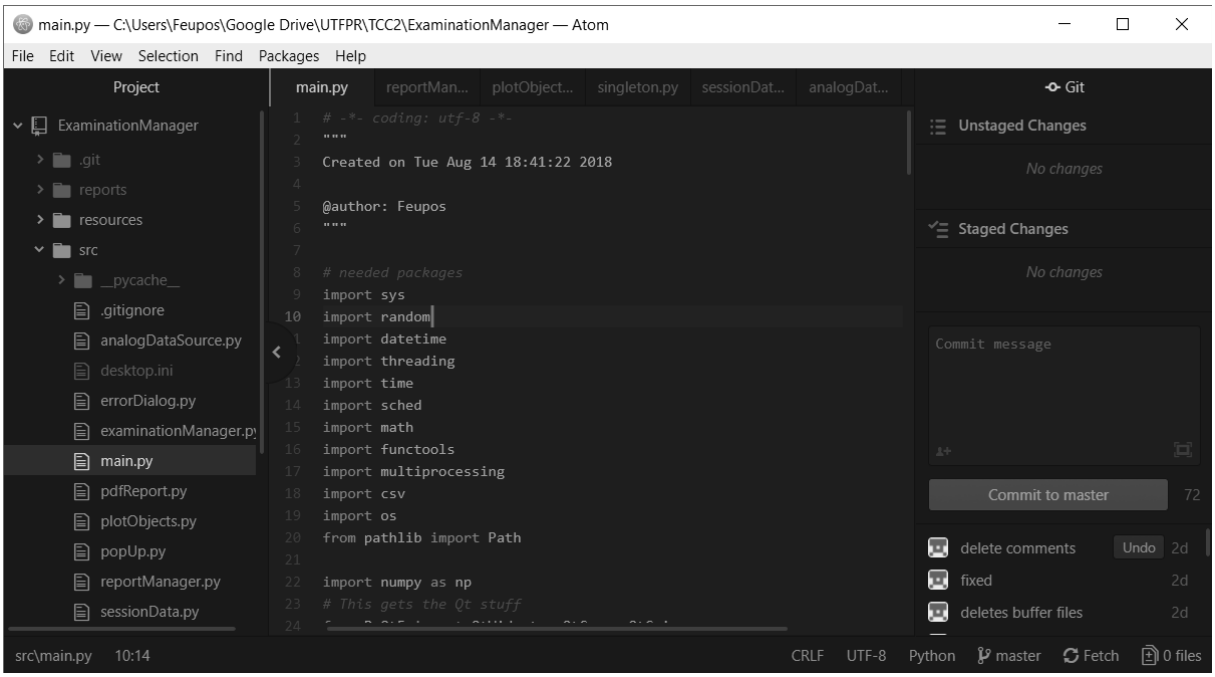
Fonte: Autoria própria

No que se refere ao desenvolvimento do software em si, que foi feito inteiramente em Python, foi utilizado o software Atom, uma IDE (*Integrated Development Environment*) simples porém cheia de recursos que facilitam o desenvolvimento, como a integração com a ferramenta de versionamento Git, permitindo executar os comandos do Git diretamente da IDE, que é mostrada na Figura 21.

3.3 INTEGRAÇÃO

Um dos aspectos mais cruciais do projeto em si foi a parte da integração do sistema mecânico do *cockpit* com o sistema embarcado. Como ambas as partes foram desenvolvidas

Figura 21 – IDE Atom



Fonte: Autoria própria

em paralelo por equipes diferentes, não houve testes com o sensor do *cockpit* ao longo do desenvolvimento, sendo utilizado um potênciometro simples para simular as medições. O desafio se colocou no fato da célula de carga não corresponder exatamente às medições esperadas de acordo com os dados fornecidos, mostrados na Figura 12.

Dentre os fatores para estas divergências estão o fato da fonte utilizada ser diferente, levando a uma pequena variação na tensão de alimentação, assim como os mecanismos instalados na célula de carga apresentarem características possivelmente dos dispositivos utilizados durante o levantamento dos dados na sua calibração inicial, como o fato do tamanho do pedal ter sido ajustado para ficar melhor posicionado e a mola reajustada para permitir que o pedal atingisse a amplitude de movimento para suportar a força incidente de 50 kgf.

3.3.1 CALIBRAÇÃO

Como relatado na seção Seção 3.3, divergências na instalação do sistema, principalmente na parte mecânica, poderiam levar a valores incorretos nas medições, de forma que o resultado do exame poderia levar a interpretações incorretas, como por exemplo, no caso de o valor inicial apontado ser superior a zero, dar a impressão de que o paciente está fazendo uma força que de fato não existe.

Poderíamos calibrar manualmente o sensor, tirando medições e ajustando diretamente no software os coeficientes de conversão, porém ainda haveria o potencial de ocorrerem divergência nos dados em campo, onde não poderíamos fazer nenhum tipo de alteração deste nível. Então foi optado por adotar dois mecanismos adicionais, um de calibração automática e

outro de calibração manual.

O mecanismo de calibração automática atua no durante a inicialização do sistema. Neste momento o software adquire 250 amostras de dados e utiliza a média como um ponto de ajuste do zero. Ou seja, nesse ponto é considerado como se não houvesse força nenhuma atuando sobre o pedal, de forma que qualquer valor medido representa um *offset* indesejado do sinal. Caso o sensor tenha sido calibrado de forma incorreta, como no caso da presença de alguma força sobre o pedal no momento da inicialização, também é possível calibrar manualmente o mesmo através de um botão no menu de configuração.

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Utilizando as ferramentas apresentadas na seção anterior foi possível desenvolver um sistema embarcado, completamente funcional e pronto para ser utilizado na sua aplicação desejada, que é a avaliação de pacientes em recuperação de lesões ou cirurgias que possam afetar sua capacidade de dirigir.

Neste ponto pudemos capturar e exibir os sinais adquiridos do sensor com alta fidelidade, isso se deu devido ao uso um ADC de 16 bits e ganho de 16 vezes, que eliminou completamente a necessidade da utilização de qualquer outro circuito externo para o tratamento do sinal antes da sua captura, eliminando outras possíveis fontes de ruído. A resolução de 16 bits também ajuda nesse sentido pois a resolução do sinal é de 1,28e-16 V, ou seja, podemos capturar o sinal de pequena amplitude proveniente da célula de carga, conforme podemos ver nas imagens abaixo obtidas em sessões de teste, com o pedal levado do repouso ao freio máximo.

4.1 INTERFACE

Esta seção tem o objetivo de explicar o funcionamento do software assim como mostrar as suas diversas telas. É importante destacar que os dados exibidos nas telas da Figura 23 e Figura 24 são simulados e tem apenas a finalidade de ilustrar o comportamento do software. Apenas dados simulados foram utilizados pois o desenvolvimento deste trabalho ocorreu em paralelo com o desenvolvimento da parte mecânica, sendo que no momento da publicação deste a mesma não estava pronta e disponível para a realização da captura de dados in vivo.

O software desenvolvido é composto basicamente de 3 telas, uma tela principal, mostrada na Figura 22, onde o usuário, no caso o médico, teria a opção de inserir todos os dados do paciente e da sessão de exames. Os dados inseridos nesta etapa determinam o número de repetições do exame, e também são utilizados na geração do relatório na última etapa. Essa tela é bem simples e serve apenas como uma interface para a inserção dos dados.

Após a inserção dos dados na tela anterior, o usuário é direcionado para outra tela, mostrada na Figura 23, onde inicialmente é mostrado um gráfico em tempo real que mostra o valor da força aplicada no pedal. Este momento inicial serve para conferir se o pedal está funcionando corretamente e realizar eventuais testes antes de iniciar o exame em si. Nesta etapa também foi colocado um painel à direita onde podem ser configurados tanto a duração do exame como o tempo para o disparo do estímulo.

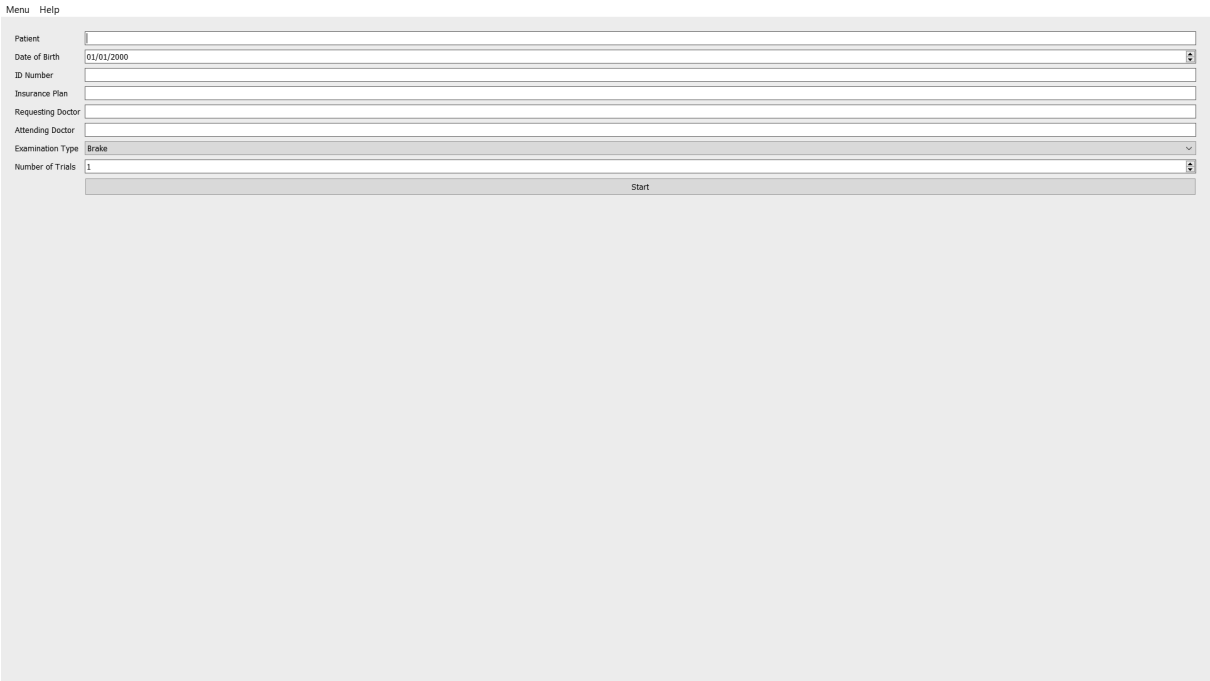
Após finalizados todos os exames o usuário é redirecionado para a tela final, mostrada na Figura 24, onde é possível rever novamente os resultados de todos os exames realizados na sessão atual. Então, o usuário tem a opção de adicionar comentários adicionais a serem colocados no relatório que é gerado automaticamente após esta etapa. O usuário tem a opção de escolher o nome do arquivo, que é salvo automaticamente em qualquer dispositivo de

armazenamento USB que esteja conectado ao dispositivo. Também é salva uma cópia integral de todos os dados em formato *.csv*.

De forma a evitar que sejam perdidos os dados por falhas no dispositivo USB, nomes inválidos, ou até mesmo caso seja esquecido de conectar o dispositivo de armazenamento USB, o programa emite um alerta nestes casos dando a opção de tentar salvar novamente. Essa tela de erro pode ser vista na Figura 25.

O código está disponível integralmente no Apêndice B.

Figura 22 – Tela principal



Fonte: Autoria própria

4.2 CUSTOS

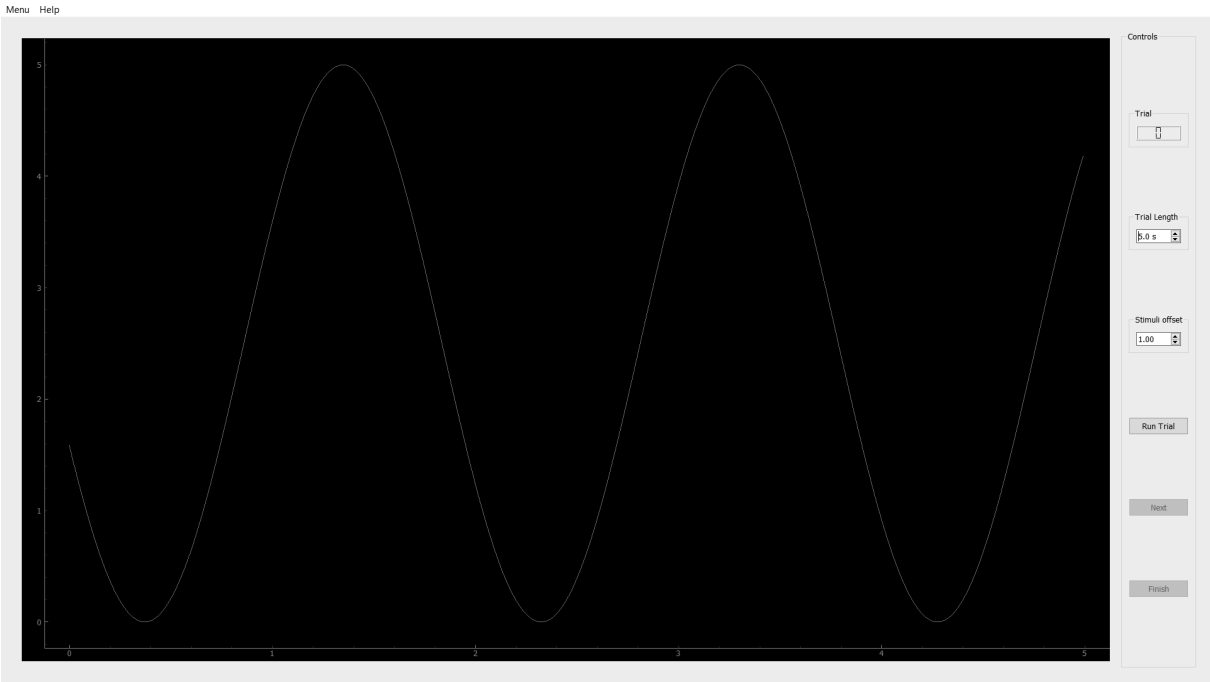
Como se tratava de uma aplicação médica o custo nunca foi um dos fatores a pesar tanto durante os estágios de planejamento do projeto, apesar disso foi possível desenvolver um sistema barato e eficiente.

4.2.1 COCKPIT

O *cockpit* utilizado para a aquisição dos dados de força do pedal foi desenvolvido em um projeto de mestrado desenvolvido no BIOTA, também sob a orientação do Prof. Bertoldo. Desta forma o levantamento de custo deste *cockpit* não entrou no escopo do nosso projeto, e como muitos dos materiais foram reciclados não é possível fazer tal levantamento.

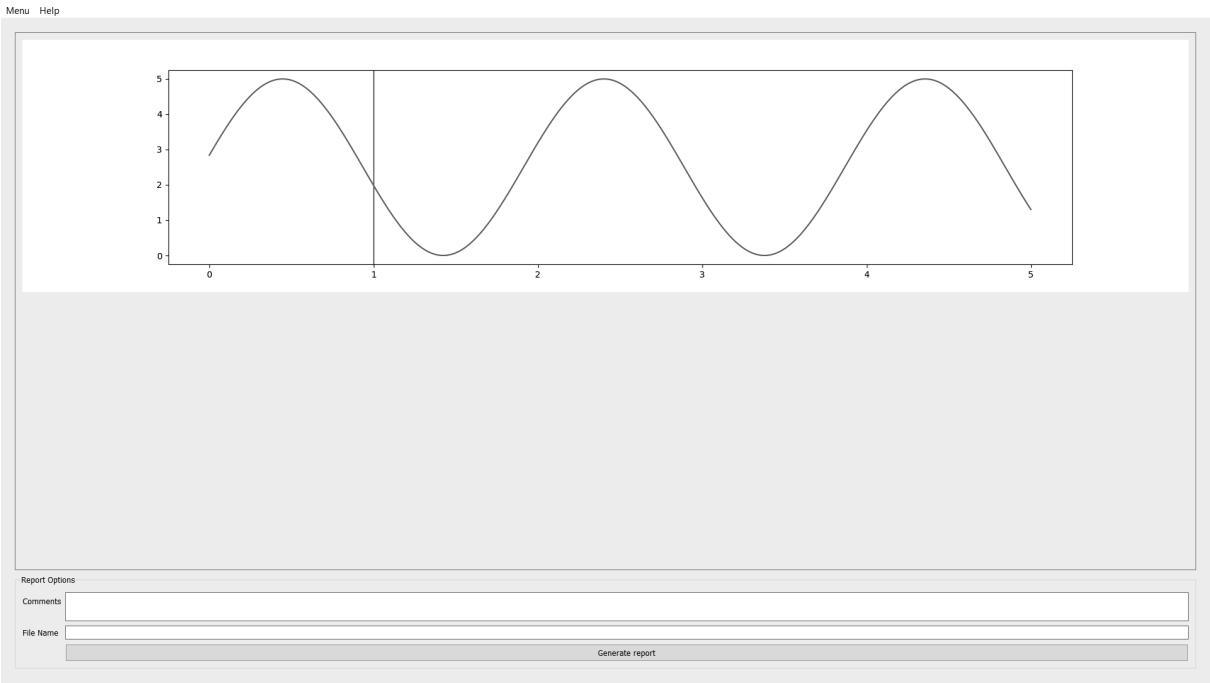
Apesar disso foram buscadas alternativas comerciais similares para fazer uma estimativa aproximada. Um dispositivo voltado ao mercado de jogos, mas que poderia facilmente ser

Figura 23 – Tela de exame



Fonte: Autoria própria

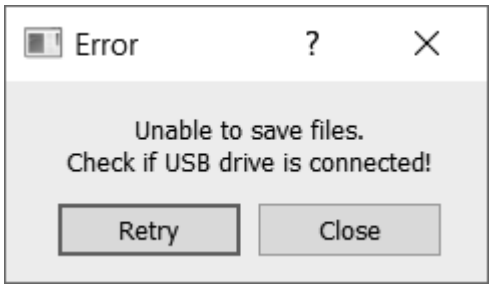
Figura 24 – Tela de relatório



Fonte: Autoria própria

adaptado ao projeto e é mostrado na Figura 26,tem o valor de R$2599,90 . Adicionalmente haveria necessidade da célula de carga, similar ao modelo GL-50 utilizado no protótipo. Os fabricantes não fornecem o preço diretemente na internet, mas pesquisas em websites de

Figura 25 – Mensagem de erro



Fonte: Autoria própria

comércio entre usuários resultaram em preços na faixa de R$290,00. Como a interface mecânica e a célula de carga foi inteiramente de desenvolvimento próprio pelo projeto não temos como levantar seu custo.

Figura 26 – Modelo comercial de *cockpit*

Fonte: Autoria própria

4.2.2 SISTEMA EMBARCADO

Com relação à parte referente ao sistema embarcado em si foram utilizados dois itens, a placado do Raspberry Pi 3B e o módulo conversor analógico digital ADS1115. É bom ressaltar a facilidade de se encontrar a venda destes itens no Brasil, pois esse foi um importante fator na decisão sobre a utilização dos mesmos. Os valores foram todos levantados de acordo com os preços da loja FilipeFlop, e estão listados na Tabela 1.

Tabela 1 – Custo do hardware embarcado

Item	Valor
Raspberry Pi Model 3B+	R$279,90
Case Raspberry Pi	R$42,90
Fonte DC USB 5V/3A	R$37,90
Cartão de Memória 16GB	R$64,90
Módulo ADS1115	R$39,90
Total	R$465,50

Fonte: Adaptado de FilipeFlop Componentes Eletrônicos (2018)

4.2.3 ACESSÓRIOS ADICIONAIS

Adicionalmente para a operação do sistema é necessário o uso de teclado, mouse e um monitor. Como estes itens não fazem parte do sistema desenvolvido, foram contabilizados de forma separada, pois nesse caso muitas vezes podem ser utilizados equipamentos já disponíveis no local, pois o sistema é compatível com qualquer teclado e mouse USB e também qualquer monitor HDMI. Para o desenvolvimento do projeto não foi comprado nenhum destes itens, sendo utilizados os materiais disponíveis nos laboratórios da UTFPR.

Para fins de análise financeira do projeto foi feito um levantamento geral do custo para estes itens, mostrado na Tabela 2.

Tabela 2 – Custo de acessórios adicionais

Item	Valor
Monitor LCD com entrada HDMI	R$500,00
Teclado USB	R$50,00
Monitor USB	R&50,00
Total	R$600,00

Fonte: Autoria própria

4.2.4 CUSTO TOTAL

Considerando todos os custos do *cockpit*, hardware embarcado e dos acessórios adicionais temos um custo total de R$3955,40, conforme apresentado na Tabela 3.

Tabela 3 – Custo total do projeto

Item	Valor
cockpit	R$2889,90
Hardware USB	R$465,50
Acessórios	R$600,00
Total	R$3955,40

Fonte: Autoria própria

5 CONCLUSÃO

5.1 CONSIDERAÇÕES FINAIS

O projeto deu inicio e finalizou o desenvolvimento de um protótipo totalmente funcional do sistema embarcado para a medição de força aplicada e tempo de resposta durante frenagem. Todos os requisitos iniciais foram atendidos. Todos os requisitos iniciais, estabelecidos na Seção 3.1, foram perfeitamente atendidos, assim como novos recursos foram implementados. O código também está integralmente disponível na plataforma GitLab, no enderço https://gitlab.com/feupos/ExaminationManager.

Como o objetivo deste projeto é ser utilizado para futuras pesquisas de alunos de mestrado do BIOTA, durante todo o processo de desenvolvimento, assim como na validação de sua versão final, contamos com o acompanhamento de um médico ortopedista, que eventualmente fará uso deste mesmo projeto, com o propósito de validar o projeto do ponto de vista do médico que irá operar o sistema, bem como fazer a análise dos dados.

A proposta inicial do sistema desenvolvido em java foi deixada de lado por termos considerado que a proposta do sistema embarcado proporcionaria uma experiência melhor ao usuário, assim como eliminou fatores que podiam trazer problemas à sua implementação em campo, como a necessidade do médico levar um notebook ao local do exame, ou ter a necessidade de uma estação de trabalho dedicada ao sistema. Apesar do valor do sistema embarcado ser mais caro por si só, seu custo compensa por não haver a necessidade de nenhum computador adicional envolvido. Também é muito melhor a proposta do software embarcado pois evita uma eventual incompatabilidade e travamentos de software, assim como proporciona maior controle sobre o sistema si do ponto de vista de desenvolvimento. Desta maneira basta ligar o equipamento na tomada que o mesmo já é inicializado e o *software* começa a funcionar, sem nenhuma interação do usuário. Todo o desenvolvimento também foi nesse sentido, de reduzir a interação necessária, ao mesmo tempo que proporciona a flexibilidade e capacidade de responder de forma adequada quanto a interação se faz necessária.

A escolha da programação em Python atendeu às necessidades do desenvolvimento, assim como permitiu o desenvolvimento ágil do projeto, levando à possibilidade de se adicionar vários recursos sem demandar um tempo adicional muito grande. Tudo isso resultou numa versão final do *software* estável e sem *bugs* que pudessem comprometer o seu uso. A estabilidade do sistema sempre foi uma das maiores preocupações durante o desenvolvimento, dado pelo fato que o sistema virá a ser operado por pessoas que podem não ter o conhecimento técnico necessário para resolver eventuais problemas em campo.

5.1.1 TRABALHOS FUTUROS

Durante o desenvolvimento do sistema sua aplicação foi dada de forma bastante limitada, sendo aplicado para testes apenas em pessoas saudáveis, no caso os membros do desenvolvimento do projeto, então já num primeiro momento este mesmo protótipo virá a ser utilizado em outras atividades de pesquisa pela UTFPR, agora englobando a pesquisa clínica, em testes com pacientes que passaram por cirurgias no joelho, que devido a isso possuem mobilidade mais limitada. Desta forma o sistema será utilizado para acompanhar o progresso do pós-cirúrgico destes pacientes.

Já referente ao aspecto técnico também há uma grande abertura para desenvolvimento futuro. A escolha de se optar por uma arquitetura Linux provê muita flexibilidade e capacidade de melhoria para o sistema. Do ponto de vista de software isso nos permite a acessar uma abundância de recursos para atender a todas estar melhorias, enquanto a capacidade de processamento da plataforma Raspberry Pi atende com folga a todos os requisitos do projeto, sobrando muito espeço para a adição de novos recursos e funcionalidades.

O projeto foi todo desenvolvido com a visão de um terminal local e isolado, porém como estamos trabalhando com a plataforma Raspberry Pi 3B, que possui conectividade Ethernet, Wi-Fi existe a possibilidade de integração do mesmo com a internet, possibilitando por exemplo a operação remota do sistema, ou a comunicação do mesmo com o banco de dados do hospital onde for aplicado, podendo gerenciar automaticamente dados de médico e paciente, assim como anexar os dados dos exames ao prontuário eletrônico do paciente, removendo a necessidade do aplicador do exame de coletar os dados localmente.

Outro grande campo de trabalho é a possibilidade de expandir o sistema para permitir a realização de outros exames. O sistema foi desenvolvido focado para sua aplicação apenas nesse tipo de medição de força de frenagem, mas existe a possibilidade de conectar o mesmo a diversos outros tipos de sensores para aplicação no ambiente hospitalar. A modularidade dos sistema desenvolvido permitiria que facilmente fosse adicionado o suporte a novos tipos de exames.

Referências

ADAFRUIT INDUSTRIES. **ADS1115 16-Bit ADC - 4 Channel with Programmable Gain Amplifier**. 2018. Disponível em: <https://www.adafruit.com/product/1085>. Acesso em: 09 de Setembro de 2018. Citado na página 24.

ALFA INSTRUMENTOS. **Célula de Carga - GL**. 2018. Disponível em: <http://www.alfainstrumentos.com.br/produto/gl/>. Acesso em: 26 de Outubro de 2018. Citado na página 25.

ASPENCORE. **Raspberry Pi Pinout Diagram**. Jameco Electronics, 2018. Disponível em: <https://www.jameco.com/Jameco/workshop/circuitnotes/raspberry-pi-circuit-note.html>. Acesso em: 9 de Setembro de 2018. Citado na página 23.

BERAN, R. G.; DEVEREUX, J. A. Medical certificates attesting fitness to drive. **Internal Medicine Journal**, v. 47, n. 6, p. 637–639, 2017. Citado na página 13.

CAMPBELL, R. et al. Why don't patients do their exercises? understanding non-compliance with physiotherapy in patients with osteoarthritis of the knee. **Journal of Epidemiology & Community Health**, BMJ Publishing Group Ltd, v. 55, n. 2, p. 132–138, 2001. ISSN 0143-005X. Disponível em: <https://jech.bmj.com/content/55/2/132>. Citado na página 14.

FILIPEFLOP COMPONENTES ELETRôNICOS. **Loja Virtual**. 2018. Disponível em: <https://www.filipeflop.com/>. Acesso em: 9 de Setembro de 2018. Citado na página 40.

FJ BYSZEWSKI AM, M. S. M.-S.-H. M. M. In-office evaluation of medical fitness to drive: practical approaches for assessing older people. **Can Fam Physician**, 2005. Citado na página 13.

GREIBE, P. **Braking distance, friction and behaviour**. 2007. Disponível em: <http://www.trafitec.dk/sites/default/files/publications/brakingdistance-frictionanddriverbehaviour.pdf>. Acesso em: 16 de Outubro de 2018. Citado 3 vezes nas páginas 16, 17 e 18.

HAWLEY, C. A.; GALBRAITH, N. D.; DESOUZA, V. A. Medical education on fitness to drive: a survey of all uk medical schools. **Postgraduate Medical Journal**, v. 84, n. 998, p. 635–638, Jan 2008. Citado 2 vezes nas páginas 18 e 19.

INSALL, J. N.; SCOTT, W. N. **Surgery of the knee**. [S.l.]: Churchill Livingstone, 2001. Citado na página 19.

KORKOT ONLINE. **Stopping distance = reaction distance + braking distance**. 2018. Disponível em: <https://korkortonline.se/en/theory/reaction-braking-stopping/>. Acesso em: 24 de Outubro de 2018. Citado 2 vezes nas páginas 15 e 16.

MAXFIELD, C. **2017 Embedded Markets Study - EE Times**. Aspencore, 2017. Disponível em: <https://m.eet.com/media/1246048/2017-embedded-market-study.pdf>. Citado 2 vezes nas páginas 30 e 31.

NGUYEN, T.; HAU, R.; BARTLETT, J. Driving reaction time before and after anterior cruciate ligament reconstruction. **Knee surgery, sports traumatology, arthroscopy : official journal of the ESSKA**, v. 8, p. 226–30, 02 2000. Citado na página 19.

PERKONS. **Código Brasileiro de Trânsito**. [S.l.], 2018. Disponível em: <http://www.ctbdigital.com.br/>. Acesso em: 22 de Outubro de 2018. Citado na página 13.

PRADO, L. R. Índices e motivos da desistÊncia do tratamento fisioterapÊutico de pacientes encaminhados ao serviÇo de fisioterapia do centro municipal de reabilitaÇÃo-poa/rs. **Salão de Iniciação Científica - Livro de resumos**, 2010. Citado na página 13.

QUINTIN, A.; THIZY, D. **Developing Successful Healthcare Software: 10 Critical Lessons**. 2018. Disponível em: <http://www.trafitec.dk/sites/default/files/publications/brakingdistance-frictionanddriverbehaviour.pdf>. Acesso em: 24 de Outubro de 2018. Citado na página 20.

RASPBERRY Pi 3 Model B. 2018. Disponível em: <https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>. Citado na página 24.

STRAIGHTPOINT. **O que é uma célula de carga?** 2018. Disponível em: <https://www.straightpoint.com/pt/what-is-a-load-cell.html>. Acesso em: 26 de Outubro de 2018. Citado 2 vezes nas páginas 25 e 26.

VALUECODERS. **10 Ways Python Development Can Benefit Your Business**. 2018. Disponível em: <https://www.valuecoders.com/blog/technology-and-apps/how-python-development-can-benefit-your-business/>. Acesso em: 22 de Outubro de 2018. Citado 2 vezes nas páginas 28 e 29.

Apêndices

APÊNDICE A – CÓDIGO FONTE - JAVA

```
1 package knee_up_v2;
2
3 import java.awt.event.ActionEvent;
4 import java.awt.event.ActionListener;
5
6 import javax.swing.ImageIcon;
7 import javax.swing.JButton;
8 import javax.swing.JFrame;
9 import javax.swing.JPanel;
10 import javax.swing.border.EmptyBorder;
11 import javax.swing.JMenu;
12 import javax.swing.JMenuBar;
13 import javax.swing.JMenuItem;
14 import java.awt.Dimension;
15 import java.awt.Toolkit;
16 import java.awt.Color;
17
18 public class Principal {
19
20     private static Cadastro cadastro;
21     private static Paciente paciente;
22     private static IniciarTeste iniciarteste;
23     private static RealizarTeste realizarteste;
24     private static ConectarSerial conectarserial;
25     private static JFrame frame;
26     private static JPanel contentPane;
27     private static int Step;
28     private static boolean Loop;
29     private static JMenuBar menuBar;
30     private static JMenuItem Conectar;
31     private static JMenu coneMenu;
32
33     public static void main(String[] args) throws InterruptedException {
34         // TODO Auto-generated method stub
35         frame = setframe();
36         Step = 1;
37         Loop =true;
38         while(Loop) {
39             switch(Step){
40
41             case 1:
42                 conectarserial= new ConectarSerial();
43                 conectarserial.setFrame(frame);
```

```
44                 conectarserial.SerialList(coneMenu);
45                 coneMenu.addSeparator();
46                 coneMenu.add(Conectar);
47 
48                 Thread verifica_serial = new Thread() {     //thread que
    verifica se a janela de cadastro foi completa
49                     public void run() {
50                             while (conectarserial.getStep() == Step) {
51                                 try {
52                                     Thread.sleep(100);
53                                 } catch (InterruptedException e) {
54                                     e.printStackTrace();
55                                 }
56                                 //System.out.println("Working now");
57                             }
58                         }
59                 };
60 
61                 verifica_serial.start();
62                 verifica_serial.join();
63 
64                 Step = conectarserial.MostrarJanelaCadastro();
65                 verifica_serial.interrupt();
66                 break;
67             case 2:
68                 cadastro = new Cadastro();
69                 cadastro.setFrame(frame);
70                 cadastro.setJPanel(contentPane);
71                 cadastro.setConectado(!conectarserial.getDesconectado());
72                  System.out.println(conectarserial.getDesconectado());
73                 Step = cadastro.MostrarJanelaCadastro();
74 
75                 Thread verifica_cadastro = new Thread() {     //thread que
    verifica se a janela de cadastro foi completa
76                     public void run() {
77                             while (!cadastro.getDadosInseridos()) {
78                                 try {
79                                     Thread.sleep(100);
80                                 } catch (InterruptedException e) {
81                                     e.printStackTrace();
82                                 }
83                                 //System.out.println("Working now");
84                             }
85                         }
86                 };
87 
88                 verifica_cadastro.start();
```

```
89                  verifica_cadastro.join();
90
91                  //Carrega dados inseridos na janela Cadastro no objeto da
    classe "Paciente"
92                  //Dados digitados como Nome do Paciente, Data do Teste,
    Data Cirurgia, etc..
93                  Step = cadastro.getStep();
94
95                  paciente = new Paciente();
96                  paciente.setNomePaciente(cadastro.getNomePaciente());
97                  paciente.setCodigo(cadastro.getCodigo());
98                  paciente.setDatadeNascimento(cadastro.getDatadeNascimento()
    );
99                  paciente.setDataTeste(cadastro.getDataTeste());
100                 paciente.setMedicoResponsavel(cadastro.getMedicoResponsavel
    ());
101                 paciente.setMedicoRequisitante(cadastro.
    getMedicoRequisitante());
102                 paciente.setConvenio(cadastro.getConvenio());
103                 paciente.setPerna(cadastro.getPerna());
104                 paciente.setNumTeste(cadastro.getNumTestes());
105                 verifica_cadastro.interrupt();
106                 break;
107
108             case 3:
109
110                 iniciarteste = new IniciarTeste();
111                 iniciarteste.setFrame(frame);
112                 iniciarteste.setJPanel(contentPane);
113                 iniciarteste.setNumTestes(paciente.getNumTeste());
114                 iniciarteste.setNomePaciente(paciente.getNomePaciente());
115                 iniciarteste.setNomeMedico(paciente.getMedicoResponsavel())
    ;
116                 iniciarteste.setDataTeste(paciente.getDataTeste());
117                 iniciarteste.setDatadeNascimento(paciente.
    getDatadeNascimento());
118                 iniciarteste.setCodigo(paciente.getCodigo());
119                 iniciarteste.setConvenio(paciente.getConvenio());
120                 iniciarteste.setMedicoRequisitante(paciente.
    getmedicoRequisitante());
121                 iniciarteste.setPerna(paciente.getPerna());
122                 iniciarteste.SetDesconectado(conectarserial.getDesconectado
    ());
123                 iniciarteste.setBotaoIniciarTeste(false);
124
125             //Chama Janela da classe Iniciar Teste
126                 iniciarteste.MostrarJanelaIniciarTeste();
```

```
127                 Thread verifica_inicio = new Thread() {    //thread que
    verifica se teste foi completo
128                     public void run() {
129                             while (!iniciarteste.getClicouBotaoIniciarTeste
    () && iniciarteste.getVoltar()==Step) {
130
131                                 try {
132                                     Thread.sleep(100);
133                                 } catch (InterruptedException e) {
134                                     e.printStackTrace();
135                                     System.out.println("Thread was
    interrupted, Failed to complete operation");
136                                 }
137                             }
138                         }
139                 };
140
141                 verifica_inicio.start();
142                 verifica_inicio.join();
143                 Step = iniciarteste.getVoltar();
144                 verifica_inicio.interrupt();
145                 break;
146             case 4:
147                 realizarteste = new RealizarTeste();
148
149                 realizarteste.setFrame(frame);
150                 realizarteste.setJPanel(contentPane);
151                 realizarteste.setPorta(conectarserial.getPorta());
152                 realizarteste.setNomePaciente(paciente.getNomePaciente());
153                 realizarteste.setNomeMedico(paciente.getMedicoResponsavel()
    );
154                 realizarteste.setDataTeste(paciente.getDataTeste());
155                 realizarteste.setDatadeNascimento(paciente.
    getDatadeNascimento());
156                 realizarteste.setCodigo(paciente.getCodigo());
157                 realizarteste.setConvenio(paciente.getConvenio());
158                 realizarteste.setMedicoRequisitante(paciente.
    getmedicoRequisitante());
159                 realizarteste.setPerna(paciente.getPerna());
160                 realizarteste.setNumTestes(paciente.getNumTeste());
161             //Chama Janela da classe Iniciar Teste
162                 realizarteste.MostrarJanelaIniciarTeste();
163                 Thread verifica_teste = new Thread() {    //thread que
    verifica se teste foi completo
164                     public void run() {
165                             while (realizarteste.getVoltar()==Step) {
166
```

```
167                                 try {
168                                     Thread.sleep(100);
169                                 } catch (InterruptedException e) {
170                                     e.printStackTrace();
171                                     System.out.println("Thread was
    interrupted, Failed to complete operation");
172                                 }
173                             }
174                         }
175                 };
176
177                 verifica_teste.start();
178                 verifica_teste.join();
179                  Step = realizarteste.getVoltar();
180                 verifica_teste.interrupt();
181                 break;
182             case 5:
183                 frame.dispose();
184                 Loop=false;
185                 break;
186
187             }
188
189             System.out.println("fim");
190         }
191
192     }
193     public static JFrame setframe() {
194           Dimension dim = Toolkit.getDefaultToolkit().getScreenSize();
195
196         frame = new JFrame();
197         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
198         frame.setBounds(dim.width/2-941/2, dim.height/2-849/2, 941, 849);
199         contentPane = new JPanel();
200         contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
201         contentPane.setLayout(null);
202         contentPane.setBackground(new Color(226,226,226));
203
204         frame.setVisible(true);
205         // Cria uma barra de menu para o JFrame
206         menuBar = new JMenuBar();
207                 // Define e adiciona dois menus drop down na barra de menus
208         JMenu fileMenu = new JMenu("Arquivo");
209         JMenu editMenu = new JMenu("Editar");
210         menuBar.add(fileMenu);
211         menuBar.add(editMenu);
212         JMenuItem newAction = new JMenuItem("Novo Paciente");
```

```
213         JMenuItem Pdf = new JMenuItem("Pdf");
214         fileMenu.add(newAction);
215         fileMenu.addSeparator();
216 
217         editMenu.add(Pdf);
218 
219         coneMenu = new JMenu("Conexo");
220         fileMenu.add(coneMenu);
221         Conectar= new JMenuItem ("Conectar");
222         Thread verifica_JMenu = new Thread() {    //thread que verifica se
    a janela de cadastro foi completa
223             public void run() {
224                     while (true) {
225                         try {
226                             Thread.sleep(100);
227                         } catch (InterruptedException e) {
228                             e.printStackTrace();
229                         }
230 
231                         Conectar.addActionListener(new ActionListener() {
232                             public void actionPerformed(ActionEvent e) {
233 
234                                 conectarserial.SerialList(coneMenu);
235                             }
236                         });
237                     }
238                 }
239         };
240 
241         verifica_JMenu.start();
242         frame.setJMenuBar(menuBar);
243         return frame;
244 
245     }
246 
247 }
```

./dados/src/Principal.java

```
1 package knee_up_v2;
2 import java.io.BufferedReader;
3 import java.io.IOException;
4 import java.io.InputStream;
5 import java.io.InputStreamReader;
6 import java.io.OutputStream;
7 import java.io.PrintWriter;
8 import javax.swing.JFrame;
9 import javax.swing.JOptionPane;
```

```
10 import javax.swing.JPanel;
11 import org.jfree.chart.ChartFactory;
12 import org.jfree.chart.ChartFrame;
13 import org.jfree.chart.ChartPanel;
14 import org.jfree.chart.JFreeChart;
15 import org.jfree.chart.plot.PlotOrientation;
16 import org.jfree.data.xy.XYSeries;
17 import org.jfree.data.xy.XYSeriesCollection;
18 import org.jfree.chart.axis.ValueAxis;
19 import org.jfree.chart.plot.XYPlot;
20 import com.fazecast.jSerialComm.SerialPort;
21
22 public class PortaSerial {
23
24     SerialPort chosenPort;
25     static OutputStream out;
26     boolean portaconectada = false;
27
28
29     private String TituloGrafico;
30     private JFreeChart chart;
31     private ChartFrame frame;
32     private ChartPanel myChartPanel;
33     private XYSeriesCollection dataset= new XYSeriesCollection();
34
35     PortaSerial(String portaselecionada) {
36         chosenPort = SerialPort.getCommPort(portaselecionada);
37         chosenPort.setComPortTimeouts(SerialPort.TIMEOUT_READ_BLOCKING,
       1000, 1000);
38         if (chosenPort.openPort()) {
39             chosenPort.setBaudRate(38400);
40             portaconectada = true;
41             System.out.println("abriu porta");
42         }
43         else System.out.println("n abriu porta");
44     }
45
46     public boolean testarArduino() throws IOException // escreve mensagem
       especfica no arduino e aguarda retorno
47                                                       // correto
48     {
49         String str = new String("knee_up"); // STRING PARA TESTAR O ARDUINO
       - NO MUDAR
50         System.out.println(chosenPort.getBaudRate());
51
52         Thread thread = new Thread() // cria thread de escrita pois   mais
       fcil  usar os tempos de espera
```

```
53          {
54              @Override
55              public void run() {
56                  try {
57                      Thread.sleep(100);
58                  } catch (Exception e) {
59                  }
60                  PrintWriter output = new PrintWriter(chosenPort.
    getOutputStream());
61                  for (int i = 0; i < 10; i++) { // mostrou-se necessrio um
    loop com algumas iteraes para que a
62                                                  // mensagem fosse enviada
    corretamente
63                      output.print(str);
64                      output.flush();
65                      try {
66                          Thread.sleep(50);
67                      } catch (Exception e) {
68                      }
69                  }
70                  try {
71                      Thread.sleep(100);
72                  } catch (Exception e) {
73                  }
74                  output.close();
75              }
76
77          };
78          thread.start();
79          boolean returnvalue = false;
80
81          try {
82              thread.join(); // espera thread de escrita terminar
83          } catch (InterruptedException e) {
84              // TODO Auto-generated catch block
85              e.printStackTrace();
86          }
87
88          System.out.println("escreveu");
89          InputStream in = chosenPort.getInputStream();
90        try {
91           System.out.println("tentou ");
92           byte[] inputData = new byte[1024];
93           int received = in.read(inputData,1, in.available());
94              System.out.println("recebeu " + received+" recebeu " +
    inputData);
95
```

```
96              if (received != 0) {
97                  returnvalue = true; // CONFIRMA ARDUINO PROGRAMADO
    CORRETAMENTE
98              }
99              else returnvalue = false;
100
101             in.close();
102         }
103       catch (Exception e) {
104
105          System.out.println("entrou no catch ");
106       }
107         chosenPort.closePort();
108         portaconectada = false;
109         return returnvalue;
110     }
111
112     public Grafico realizarTeste(int numTeste, String NomePaciente, String
    DataTeste) throws IOException {
113
114         String str = new String("iniciarteste"); // STRING PARA TESTAR O
    ARDUINO - NO MUDAR
115         Grafico graf = new Grafico();
116
117         if (chosenPort.openPort()) {
118             portaconectada = true;
119         }
120
121         Thread thread = new Thread() { // cria thread de escrita pois
    mais fcil usar os tempos de espera
122
123             @Override
124             public void run() {
125                 try {
126                     Thread.sleep(100);
127                 } catch (Exception e) {
128                 }
129                 PrintWriter output = new PrintWriter(chosenPort.
    getOutputStream());
130                 for (int i = 0; i < 10; i++) { // mostrou-se necessrio um
    loop com algumas iteraes para que a
131                                                 // mensagem fosse enviada
    corretamente
132                     output.print(str);
133                     output.flush();
134                     try {
135                         Thread.sleep(50);
```

```
136                     } catch (Exception e) {
137                     }
138                 }
139                 try {
140                     Thread.sleep(100);
141                 } catch (Exception e) {
142                 }
143                 System.out.println("enviou " + str);
144                 output.close();
145             }
146 
147         };
148         thread.start();
149         try {
150             thread.join(); // espera thread de escrita terminar
151         } catch (InterruptedException e) {
152             // TODO Auto-generated catch block
153             e.printStackTrace();
154         }
155 
156         InputStream in = chosenPort.getInputStream();
157         int received = in.read();
158         System.out.println("recebeu " + received);
159 
160         while (received != 77) {
161 
162         }
163         System.out.println("leitura confirmada");
164         in.close();
165 
166         Txt txt1 = new Txt();
167         txt1.openFile(numTeste, NomePaciente, DataTeste);
168 
169         XYSeries xySeries = new XYSeries("Teste ");
170         XYSeries marker = new XYSeries("marcador");
171         chart = ChartFactory.createXYLineChart(TituloGrafico, "Tempos (s)",
        "Tenso (V)", dataset,
172                 PlotOrientation.VERTICAL, true, false, false);
173 
174         XYPlot xyPlot = (XYPlot) chart.getPlot();
175         myChartPanel = new ChartPanel(chart, true); //criei o painel de
        grafico colocando meu grafico previamente gerado
176         myChartPanel.setSize(500,300); //setei o tamanho do grafico
        conforme o painel que usarei
177         myChartPanel.setVisible(true);
178 
179         ValueAxis domain = xyPlot.getDomainAxis();
```

```
180         domain.setRange(0.0,5.5);
181         domain.setVerticalTickLabels(true);
182         ValueAxis range =  xyPlot.getRangeAxis();
183         range.setRange(0.0, 3.0);
184 
185         TituloGrafico = "Knee-Up - ";
186         frame = new ChartFrame(TituloGrafico, chart);
187         frame.setVisible(true);
188         frame.setSize(800, 650);
189 
190         //criar thread para escutar porta serial:
191         Thread threadLeitura = new Thread() {
192             @Override public void run() {
193                 BufferedReader br = new BufferedReader(new
    InputStreamReader(chosenPort.getInputStream())); //BufferedReader   mais
     r pido  que scanner
194                 int i = 0;
195                 int tensaoadc[] = new int[1000];
196                 int tempomillis[] = new int[1000];
197                 System.out.println("iniciou leitura");
198                 while(i<1000) {
199                     try{
200                     tensaoadc[i] = Integer.parseInt(br.readLine());
201                     tempomillis[i] = Integer.parseInt(br.readLine());
202                     xySeries.add(tempomillis[i]/1000.0, (tensaoadc[i]*5.0)
    /1023.0);
203                     marker.add(1,i/200);
204                     if (dataset!=null) {
205                         dataset.removeAllSeries();
206                     }
207                     dataset.addSeries(xySeries);
208                     dataset.addSeries(marker);
209                     i++;
210 
211                     } catch(Exception e) {
212                         System.out.println("erro");
213                         JOptionPane.showMessageDialog(null,"Erro na  medi
     - reinicie  o programa","Knee UP",JOptionPane.ERROR_MESSAGE);
214                     }
215 
216                 }
217 
218                  System.out.println("terminou  leitura");
219 
220                 try {
221                     br.close();
222                 } catch (IOException e) {
```

```
223                     System.out.println("erro ao fechar o BufferedReader");
224                 }
225
226                 for(i = 0; i<1000 ;i++){
227                     txt1.addTempo(tempomillis[i]/1000.0);
228                     txt1.addTensao((tensaoadc[i]*5.0)/1023.0);
229                 }
230                  System.out.println("terminou grava em arquivo txt");
231             }
232         };
233         threadLeitura.start();
234         try {
235             threadLeitura.join();
236         } catch (InterruptedException e) {
237             System.out.println("erro na leitura do arduino");
238         }
239
240         txt1.closeFile();
241
242         graf.GeraGrafico(txt1.getNomeArquivo(), numTeste);
243         frame.removeAll();
244         frame.revalidate();
245         frame.repaint();
246         frame.dispose();
247         System.out.println("terminou gera de grfico ");
248         return graf;
249     }
250
251     public void FecharCom() {
252         portaconectada = false;
253         chosenPort.closePort();
254     }
255
256     public boolean getPortaconectada() {
257         return portaconectada;
258     }
259
260     ChartPanel getChartPanel()
261     {
262         return myChartPanel;
263     }
264
265 }
```

./dados/src/PortaSerial.java

```
1 package knee_up_v2;
2
```

```
3 import java.awt.HeadlessException;
4 import java.io.IOException;
5
6 import javax.swing.ImageIcon;
7 import javax.swing.JDialog;
8 import javax.swing.JFrame;
9 import javax.swing.JMenu;
10 import javax.swing.JMenuItem;
11 import javax.swing.JOptionPane;
12 import javax.swing.JPanel;
13 import javax.swing.AbstractAction;
14 import java.awt.event.*;
15
16 import com.fazecast.jSerialComm.SerialPort;
17
18 public class ConectarSerial {
19     private int Step;
20     private String [] portas = new String[10];
21     private boolean desconectado = true;
22     private PortaSerial portaserial;
23     private ImageIcon check= new ImageIcon("images/check.png");
24     private ImageIcon cross= new ImageIcon("images/cross.png");
25     private ImageIcon icon= new ImageIcon("images/loading.gif");
26
27     private JFrame window;
28     private JDialog dialog;
29
30     public int MostrarJanelaCadastro () {
31         Step=2;
32         return Step;
33     }
34
35     public void SerialList(JMenu Menubar) {
36
37         Menubar.removeAll();
38
39         SerialPort[] portNames = SerialPort.getCommPorts();
40             for (int i = 0 ; i < portNames.length ; i ++ ) {
41                 Menubar.add(new JMenuItem (portNames[i].getSystemPortName()
    ));
42             }
43
44
45         ConectPort(Menubar,portNames);
46
47
48     }
```

```
49     public void ConectPort(JMenu Menubar,SerialPort[] portNames ) {
50
51         //loading = JOptionPane.showMessageDialog(null, "Carregando", "Knee
       -up", JOptionPane.INFORMATION_MESSAGE, icon);
52         dialog = new JDialog(window, false); // Sets its owner but makes it
        non-modal
53         JOptionPane optionPane = new JOptionPane("Carregando", JOptionPane.
       INFORMATION_MESSAGE, JOptionPane.DEFAULT_OPTION, null, new Object[]{},
       null);// Same arguments as in JOptionPane.showMessageDialog(...)
54         optionPane.setIcon(icon);
55
56         dialog.getContentPane().add(optionPane); // Adds the JOptionPane to
        the dialog
57
58         dialog.pack(); // Packs the dialog so that the JOptionPane can be
       seen
59         dialog.setVisible(true); // Shows the dialog
60         dialog.setLocation(300,200);
61         desconectado=true;
62         if(portNames.length==0) {
63             JOptionPane.showMessageDialog(null," necessrio conectar o
       equipamento!","Knee UP",JOptionPane.INFORMATION_MESSAGE);
64         }
65         for (int i = 0 ; i < portNames.length ; i ++ ) {
66             if (desconectado==true)
67             { // CONECTA A PORTA
68                 //Coloca a porta serial seleciona em Srting[0];
69                     portas[0]= Menubar.getItem(i).getText();
70                     //Conecta a porta serial definida por portas[0] com
       Baud-Rate de 19200
71                     portaserial = new PortaSerial(portas[0]);
72                     if(portaserial.getPortaconectada())
73                     {
74                         try {
75                             if(portaserial.testarArduino()) {
76                                 Menubar.getItem(i).setIcon(check);
77                                 dialog.setVisible(false);
78                                 desconectado = false;
79                             }
80                             else
81                             {
82                                 portaserial.FecharCom();
83                                 portaserial = null;
84                                 Menubar.getItem(i).setIcon(cross);
85                             }
86                         } catch (HeadlessException | IOException e) {
87                             // TODO Auto-generated catch block
```

```
88                              e.printStackTrace();
89                              System.out.println("DEU RUIM");
90                          }
91                      }
92                      else {
93                          portaserial.FecharCom();
94                          portaserial = null;
95                          Menubar.getItem(i).setIcon(cross);
96                          }
97
98              }
99
100
101             /*else
102             {   //DESCONECTA A PORTA
103                 portaserial.FecharCom();
104                 desconectado = true;
105                 System.out.println("aqui no else");
106             }*/
107         }
108         if(desconectado) {
109             dialog.setVisible(false);
110             JOptionPane.showMessageDialog(null,"  necessrio conectar o
    equipamento!","Knee UP",JOptionPane.INFORMATION_MESSAGE);
111
112         }
113
114
115     }
116
117     public void setFrame(JFrame labelframe)
118     {
119         window = labelframe;
120
121     }
122
123
124
125     public int getStep() {
126
127         return Step;
128     }
129     public boolean getDesconectado() {
130         return this.desconectado;
131     }
132     public PortaSerial getPorta() {
133         return this.portaserial;
```

```
134     }
135
136 }
```

./dados/src/ConectarSerial.java

```
 1 package knee_up_v2;
 2
 3 public class Paciente {
 4
 5     /**
 6      * @param args
 7      */
 8
 9
10     private String nomePaciente;
11     private String codigo;
12     private String MedicoResponsavel;
13     private String dataTeste;
14     private String datadeNascimento;
15     private String medicoRequisitante;
16     private String convenio;
17     private String perna;
18     private int NumTeste;
19
20     public String getmedicoRequisitante() {
21         return medicoRequisitante;
22     }
23
24     public String getPerna() {
25         return perna;
26     }
27
28     public int getNumTeste() {
29         return NumTeste;
30     }
31
32
33     public String getNomePaciente() {
34         return nomePaciente;
35     }
36
37     public String getConvenio() {
38         return convenio;
39     }
40     public String getCodigo() {
41         return codigo;
42     }
```

```
43
44
45      public String getMedicoResponsavel() {
46          return MedicoResponsavel;
47      }
48
49
50      public String getDataTeste() {
51          return dataTeste;
52      }
53
54
55      public String getDatadeNascimento() {
56          return datadeNascimento;
57      }
58
59
60      public void setNomePaciente(String nomePaciente) {
61          this.nomePaciente = nomePaciente;
62      }
63
64
65      public void setCodigo(String codigo) {
66          this.codigo = codigo;
67      }
68
69      public void setConvenio(String convenio) {
70          this.convenio = convenio;
71      }
72
73      public void setMedicoResponsavel(String medicoResponsavel) {
74          this.MedicoResponsavel = medicoResponsavel;
75      }
76
77
78      public void setDataTeste(String dataTeste) {
79          this.dataTeste = dataTeste;
80      }
81
82
83      public void setDatadeNascimento(String datadeNascimento) {
84          this.datadeNascimento = datadeNascimento;
85      }
86
87
88      public void setMedicoRequisitante (String stringMedicoRequisitante) {
89          this.medicoRequisitante = stringMedicoRequisitante;
```

```
90      }
91
92
93      public void setPerna(String perna) {
94          this.perna = perna;
95      }
96      public void setNumTeste(int labelNumTeste) {
97          this.NumTeste = labelNumTeste;
98      }
99
100
101     public Paciente (){}
102
103
104
105 }
```

./dados/src/Paciente.java

```
1  package knee_up_v2;
2  import java.awt.Color;
3
4  import java.awt.event.ItemEvent;
5  import java.awt.Font;
6  import java.awt.event.ActionEvent;
7  import java.awt.event.ActionListener;
8  import java.sql.Date;
9  import java.text.SimpleDateFormat;
10 import javax.swing.JOptionPane;
11 import javax.swing.BorderFactory;
12 import javax.swing.ImageIcon;
13 import javax.swing.JButton;
14 import javax.swing.JFrame;
15 import javax.swing.JLabel;
16 import javax.swing.JPanel;
17 import javax.swing.JTextField;
18 import javax.swing.SwingConstants;
19
20
21
22 import javax.swing.JComboBox;
23 import javax.swing.JToggleButton;
24 import java.awt.event.ItemListener;
25
26
27 public class Cadastro {
28
29      private JFrame window;
```

```
30      private JPanel contentPane;
31      private JTextField MedicoResponsavel;
32      private JTextField DataTeste;
33      private JTextField DatadeNascimento;
34      private JTextField Codigo;
35      private JTextField NomePaciente;
36      private boolean DadosInseridos;
37      private boolean Confirmar;
38      private boolean Conectado;
39      private JTextField MedicoRequisitante;
40
41      private String[] ConvenioStrings = {"--", "Funda X", "Convenio Y", "
        Dog", "Z", "W" };
42      private JComboBox<String> ConvenioList;
43      private String Convenio;
44
45      private String[] NumTestesArray = {"1","2","3","4" };
46      private JComboBox<String> NumTestesList;
47      private int  NumTestes;
48      private JToggleButton Perna;
49      private int Step;
50      private JButton BotaoOK;
51
52      public void FecharJanelaConectar( int step) throws InterruptedException
53      {
54          contentPane.removeAll();
55
56          DadosInseridos = true;
57          window.repaint();
58          Step=step;
59
60      }
61
62      public int MostrarJanelaCadastro () {
63          window.setTitle("Knee UP");
64
65          Cadastro();
66          window.invalidate();
67          window.validate();
68          window.repaint();
69          return Step;
70      }
71
72      /**
73       * Create the frame.
74       */
75
```

```
76     public void ChecarDados() throws InterruptedException{
77
78         NomePaciente.setEditable(false);
79         Codigo.setEditable(false);
80         Codigo.setDisabledTextColor(new Color(52, 59, 152 ));
81         DatadeNascimento.setEditable(false);
82         DataTeste.setEditable(false);
83         MedicoRequisitante.setEditable(false);
84         MedicoResponsavel.setEditable(false);
85         ConvenioList.setEnabled(false);
86         NumTestesList.setEnabled(false);
87         Perna.setEnabled(false);
88          ImageIcon ConfirmarDadosIcon = new ImageIcon("images/confirmar-
    dados.png");
89         BotaoOK.setIcon(ConfirmarDadosIcon);
90         BotaoOK.setBounds(100, 718, ConfirmarDadosIcon.getIconWidth(),
    ConfirmarDadosIcon.getIconHeight());
91         window.revalidate();
92         window.repaint();
93         Confirmar=true;
94
95          System.out.println(DadosInseridos);
96     }
97
98     public void Cadastro() {
99         Step =2;
100        //DadosInseridos = false;
101        if (window !=null && !DadosInseridos) {
102                System.out.println(DadosInseridos);
103
104                ImageIcon bloco = new ImageIcon("images/bloco-cinza.png");
105                ImageIcon bloco_data = new ImageIcon("images/bloco-cinza-com
    -data.png");
106        ImageIcon logo = new ImageIcon("images/logo-knee-up.png");
107        JLabel logo_knee = new JLabel(logo);
108        logo_knee.setHorizontalAlignment(SwingConstants.CENTER);
109        logo_knee.setBounds(15, 17, logo.getIconWidth(), logo.getIconHeight
    ());
110        contentPane.add(logo_knee);
111
112        ImageIcon image_5 = new ImageIcon("images/formulario2.png");
113        JLabel label_5 = new JLabel(image_5);
114        label_5.setHorizontalAlignment(SwingConstants.CENTER);
115        label_5.setBounds(477, 85, image_5.getIconWidth(), image_5.
    getIconHeight());
116        contentPane.add(label_5);
117
```

```
118         ImageIcon image = new ImageIcon("images/nome-do-paciente2.png");
119         JLabel label = new JLabel(image);
120         label.setBounds(107, 148, image.getIconWidth(), image.getIconHeight
        ());
121         contentPane.add(label);
122 
123         ImageIcon image_1 = new ImageIcon("images/codigo2.png");
124         JLabel label_1 = new JLabel(image_1);
125         label_1.setBounds(209, 210, image_1.getIconWidth(), image_1.
        getIconHeight());
126         contentPane.add(label_1);
127 
128         ImageIcon image_2 = new ImageIcon("images/data-de-nascimento2.png")
        ;
129         JLabel label_2 = new JLabel(image_2);
130         label_2.setBounds(92, 272, image_2.getIconWidth(), image_2.
        getIconHeight());
131         contentPane.add(label_2);
132 
133         ImageIcon image_3 = new ImageIcon("images/data-do-teste2.png");
134         JLabel label_3 = new JLabel(image_3);
135         label_3.setBounds(152, 331, image_3.getIconWidth(), image_3.
        getIconHeight());
136         contentPane.add(label_3);
137 
138         ImageIcon image_6 = new ImageIcon("images/convenio.png");
139         JLabel label_6 = new JLabel(image_6);
140         label_6.setBounds(182, 451, image_6.getIconWidth(), image_6.
        getIconHeight());
141         contentPane.add(label_6);
142 
143         ImageIcon image_7 = new ImageIcon("images/perna2.png");
144         JLabel label_7 = new JLabel(image_7);
145         label_7.setBounds(218, 514, image_7.getIconWidth(), image_7.
        getIconHeight());
146         contentPane.add(label_7);
147 
148         ImageIcon image_4 = new ImageIcon("images/medico-responsavel2.png")
        ;
149         JLabel label_4 = new JLabel(image_4);
150         label_4.setBounds(95, 633, image_4.getIconWidth(), image_4.
        getIconHeight());
151         contentPane.add(label_4);
152 
153 
154 
155         ImageIcon image_m = new ImageIcon("images/medico-requisitante2.png"
```

```
    );
156         JLabel lblmedicoRequisitante = new JLabel(image_m);
157         lblmedicoRequisitante.setBounds(95, 388, image_m.getIconWidth(),
    image_m.getIconHeight());
158         contentPane.add(lblmedicoRequisitante);
159 
160         ImageIcon image_8 = new ImageIcon("images/repeticoes2.png");
161         JLabel label_8 = new JLabel(image_8);
162         label_8.setBounds(175, 572, image_8.getIconWidth(), image_8.
    getIconHeight());
163         contentPane.add(label_8);
164         //*****************************
165         //Variaveis que o usuario digita
166         /*Pega a data do sistema e joga em DataTeste*/
167         System.currentTimeMillis();
168         Date date = new Date(System.currentTimeMillis());
169         String dataString = new SimpleDateFormat("dd/MM/yyyy").format(date)
    ;
170         String data[] = dataString.split("\\/");
171 
172         JLabel bloco_0 = new JLabel(bloco);
173         bloco_0.setBounds(285, 135, bloco.getIconWidth(), bloco.
    getIconHeight());
174         contentPane.add(bloco_0);
175         NomePaciente = new JTextField();
176         NomePaciente.setColumns(10);
177         NomePaciente.setOpaque( false );
178         NomePaciente.setBorder((BorderFactory.createLineBorder(Color.white,
     0)));
179         NomePaciente.setBounds(295, 132, bloco.getIconWidth(), bloco.
    getIconHeight());
180         contentPane.add(NomePaciente);
181 
182         JLabel bloco_1 = new JLabel(bloco);
183         bloco_1.setBounds(285, 192, bloco.getIconWidth(), bloco.
    getIconHeight());
184         contentPane.add(bloco_1);
185         Codigo = new JTextField();
186         Codigo.setColumns(10);
187         Codigo.setOpaque( false );
188         Codigo.setBorder((BorderFactory.createLineBorder(Color.white, 0)));
189         Codigo.setBounds(295, 192, bloco.getIconWidth(), bloco.
    getIconHeight());
190         contentPane.add(Codigo);
191 
192         JLabel bloco_2 = new JLabel(bloco);
193         bloco_2.setBounds(285, 253, bloco.getIconWidth(), bloco.
```

```
    getIconHeight());
194         contentPane.add(bloco_2);
195         DatadeNascimento = new JTextField( data[0] + "/" + data[1] + "/" +
    data[2]);
196         DatadeNascimento.setColumns(10);
197         DatadeNascimento.setOpaque( false );
198         DatadeNascimento.setBorder((BorderFactory.createLineBorder(Color.
    white, 0)));
199         DatadeNascimento.setBounds(295, 253,bloco.getIconWidth(), bloco.
    getIconHeight());
200         contentPane.add(DatadeNascimento);
201
202         JLabel bloco_3 = new JLabel(bloco);
203         bloco_3.setBounds(285, 313, bloco.getIconWidth(), bloco.
    getIconHeight());
204         contentPane.add(bloco_3);
205         DataTeste = new JTextField( data[0] + "/" + data[1] + "/" + data
    [2]);
206         DataTeste.setBounds(295, 313, bloco.getIconWidth(), bloco.
    getIconHeight());
207         DataTeste.setOpaque( false );
208         DataTeste.setBorder((BorderFactory.createLineBorder(Color.white, 0)
    ));
209         contentPane.add(DataTeste);
210         DataTeste.setColumns(10);
211
212         JLabel bloco_4 = new JLabel(bloco);
213         bloco_4.setBounds(285, 373, bloco.getIconWidth(), bloco.
    getIconHeight());
214         contentPane.add(bloco_4);
215         MedicoRequisitante = new JTextField();
216         MedicoRequisitante.setColumns(10);
217         MedicoRequisitante.setOpaque( false );
218         MedicoRequisitante.setBorder((BorderFactory.createLineBorder(Color.
    white, 0)));
219         MedicoRequisitante.setBounds(295, 373, bloco.getIconWidth(), bloco.
    getIconHeight());
220         contentPane.add(MedicoRequisitante);
221
222
223         JLabel bloco_5 = new JLabel(bloco);
224         bloco_5.setBounds(285, 433, bloco.getIconWidth(), bloco.
    getIconHeight());
225         contentPane.add(bloco_5);
226         ConvenioList = new JComboBox<String>(ConvenioStrings);
227         ConvenioList.setSelectedIndex(0);
228         ConvenioList.setOpaque( false );
```

```
229         ConvenioList.setBackground(Color.white);
230         ConvenioList.setBorder((BorderFactory.createLineBorder(Color.white,
        0)));
231         ConvenioList.setBounds(295,433, bloco.getIconWidth(), bloco.
    getIconHeight());
232         contentPane.add(ConvenioList);
233 
234         ConvenioList.addActionListener(new ActionListener() {
235             public void actionPerformed(ActionEvent arg0) {
236                 if (arg0.getSource() == ConvenioList) {
237                     Convenio = (String)ConvenioList.getSelectedItem();
238                 }
239 
240             }
241         });
242 
243         JLabel bloco_6 = new JLabel(bloco);
244         bloco_6.setBounds(285, 493, bloco.getIconWidth(), bloco.
    getIconHeight());
245         contentPane.add(bloco_6);
246         Perna = new JToggleButton("Direita");
247         Perna.setBounds(295, 493, bloco.getIconWidth(), bloco.getIconHeight
    ());
248         Perna.setOpaque(false);
249         Perna.setContentAreaFilled(false);
250         Perna.setBorderPainted(false);
251         contentPane.add(Perna);
252         Perna.addItemListener(new ItemListener() {
253                 public void itemStateChanged(ItemEvent ev) {
254                    if(ev.getStateChange()==ItemEvent.SELECTED){
255                     Perna.setText("Esquerda");
256                    } else if(ev.getStateChange()==ItemEvent.DESELECTED){
257                     Perna.setText("Direita");
258                    }
259                 }
260             });
261 
262         JLabel bloco_7 = new JLabel(bloco);
263         bloco_7.setBounds(285, 554, bloco.getIconWidth(), bloco.
    getIconHeight());
264         contentPane.add(bloco_7);
265         NumTestesList = new JComboBox<String>(NumTestesArray);
266         NumTestesList.setSelectedIndex(2); // 3 repeticoes eh o padrao
267         NumTestesList.setBounds(295, 554, bloco.getIconWidth(), bloco.
    getIconHeight());
268         NumTestesList.setOpaque( false );
269         NumTestesList.setBorder((BorderFactory.createLineBorder(Color.white
```

```
    , 0)));
270         contentPane.add(NumTestesList);
271         NumTestes = 3;
272         NumTestesList.addActionListener(new ActionListener() {
273             public void actionPerformed(ActionEvent arg0) {
274                 if (arg0.getSource() == NumTestesList) {
275                     NumTestes = Integer.parseInt((String)NumTestesList.
    getSelectedItem());
276                         System.out.println(NumTestes);
277                 }
278 
279             }
280         });
281 
282         JLabel bloco_8 = new JLabel(bloco);
283         bloco_8.setBounds(285, 614, bloco.getIconWidth(), bloco.
    getIconHeight());
284         contentPane.add(bloco_8);
285         MedicoResponsavel = new JTextField();
286         MedicoResponsavel.setColumns(10);
287         MedicoResponsavel.setOpaque( false );
288         MedicoResponsavel.setBorder((BorderFactory.createLineBorder(Color.
    white, 0)));
289         MedicoResponsavel.setBounds(295, 614, bloco.getIconWidth(), bloco.
    getIconHeight());
290         contentPane.add(MedicoResponsavel);
291 //Variaveis que o usuario digita
292 //*****************************
293         ImageIcon OK = new ImageIcon("images/botao-ok.png");
294 
295         BotaoOK = new JButton("OK", OK);
296         BotaoOK.setOpaque(false);
297         BotaoOK.setContentAreaFilled(false);
298         BotaoOK.setBorderPainted(false);
299         BotaoOK.setBounds(260, 718, OK.getIconWidth(), OK.getIconHeight());
300         contentPane.add(BotaoOK);
301         BotaoOK.addActionListener(new ActionListener() {
302             public void actionPerformed(ActionEvent arg0) {
303                 //Fecha Janela "Conectar"
304                 if (!Confirmar) {
305                         if(NomePaciente.getText().equals("")) {
306                         JOptionPane.showMessageDialog(null,"  necessrio
    definir o nome do paciente!","Knee UP",JOptionPane.INFORMATION_MESSAGE);
307                     }
308                     else if(Codigo.getText().equals("")) {
309                         JOptionPane.showMessageDialog(null,"  necessrio
    definir o  c digo !","Knee UP",JOptionPane.INFORMATION_MESSAGE);
```

```
310                     }
311
312
313                     else if(Codigo.getText().matches("[0-9]+")==false ) {
314                         JOptionPane.showMessageDialog(null," necessrio
    definir o c digo SOMENTE n merico!","Knee UP",JOptionPane.
    INFORMATION_MESSAGE);
315                     }
316                     else if(DataTeste.getText().equals("") ||DataTeste.
    getText().matches("([0-9]{2})/([0-9]{2})/([0-9]{4})")==false ) {
317                         JOptionPane.showMessageDialog(null," necessrio
    definir a data do teste!","Knee UP",JOptionPane.INFORMATION_MESSAGE);
318                     }
319
320                     else if(MedicoRequisitante.getText().equals("")) {
321                         JOptionPane.showMessageDialog(null," necessrio
    definir o Medico Requisitante do teste!","Knee UP",JOptionPane.
    INFORMATION_MESSAGE);
322                     }
323
324                     else if(MedicoResponsavel.getText().equals("")) {
325                         JOptionPane.showMessageDialog(null," necessrio
    definir um Medico Responsavel!","Knee UP",JOptionPane.
    INFORMATION_MESSAGE);
326                     }
327
328                     else{
329
330                         try {ChecarDados();
331
332                         ImageIcon Voltar = new ImageIcon("images/voltar.png
    ");
333                         JButton btnVoltar = new JButton("Voltar",Voltar);
334                         btnVoltar.setBounds(400, 718, Voltar.getIconWidth()
    , Voltar.getIconHeight());
335                         btnVoltar.setOpaque(false);
336                         btnVoltar.setContentAreaFilled(false);
337                         btnVoltar.setBorderPainted(false);
338                         contentPane.add(btnVoltar);
339                         btnVoltar.addActionListener(new ActionListener() {
340                             public void actionPerformed(ActionEvent arg0) {
341
342                                 try {
343                                     FecharJanelaConectar(2);
344                                 } catch (InterruptedException e) {
345                                     // TODO Auto-generated catch block
346                                     e.printStackTrace();
```

```
347                                     }
348                                 }
349                             });
350                             }
351                             catch (InterruptedException e) {e.printStackTrace()
    ;}
352                         }
353                     }
354                     else {
355                         try {
356                             FecharJanelaConectar(4);}
357                         catch (InterruptedException e) {e.printStackTrace();}
358                     }
359                     }
360             });
361
362             window.add(contentPane);
363 }
364     }
365
366     public String getMedicoRequisitante() {
367
368         String medicoRequisitante = MedicoRequisitante.getText();
369         medicoRequisitante= medicoRequisitante.substring(0, 1).toUpperCase
    () + medicoRequisitante.substring(1);
370         return medicoRequisitante;
371     }
372
373
374     public String getMedicoResponsavel() {
375
376         String medicoResponsavel = MedicoResponsavel.getText().trim();
377         medicoResponsavel= medicoResponsavel.substring(0, 1).toUpperCase()
    + medicoResponsavel.substring(1);
378         return medicoResponsavel;
379     }
380
381     public String getDataTeste() {
382
383         String dataTeste = DataTeste.getText();
384         return dataTeste;
385
386     }
387
388     public String  getDatadeNascimento() {
389
390         String datadeNascimento = DatadeNascimento.getText();
```

```
391         return datadeNascimento;
392 
393     }
394 
395     public String getCodigo() {
396 
397         String codigo = Codigo.getText();
398         return codigo;
399 
400     }
401     public String getConvenio() {
402 
403         String convenio = Convenio;
404         return convenio;
405 
406     }
407     public String getPerna() {
408 
409         String perna = Perna.getText();
410         return perna;
411 
412     }
413 
414     public String getNomePaciente() {
415 
416         String nomepaciente = NomePaciente.getText();
417         nomepaciente= nomepaciente.substring(0, 1).toUpperCase() +
    nomepaciente.substring(1);
418         return nomepaciente;
419 
420     }
421 
422     public boolean getDadosInseridos()
423     {
424         return this.DadosInseridos;
425     }
426 
427     public int getNumTestes()
428     {
429         return this.NumTestes;
430     }
431     public int getStep() {
432 
433         return Step;
434     }
435 
436     public void setConectado (boolean LabelConectado) {
```

```
437
438         Conectado= LabelConectado;
439     }
440     public void setFrame(JFrame labelframe)
441     {
442         window = labelframe;
443
444     }
445
446     public void setJPanel(JPanel labelPanel)
447     {
448         contentPane = labelPanel;
449
450     }
451
452 }
```

./dados/src/Cadastro.java

```
 1 package knee_up_v2;
 2
 3 import javax.swing.JPanel;
 4 import javax.swing.JFrame;
 5 import javax.swing.JLabel;
 6 import javax.swing.JTextArea;
 7 import javax.swing.JOptionPane;
 8 import java.awt.Font;
 9 import javax.swing.SwingConstants;
10 import java.awt.BorderLayout;
11 import java.awt.Color;
12 import javax.swing.JButton;
13 import java.awt.event.ActionListener;
14 import java.awt.event.ActionEvent;
15
16 public class IniciarTeste extends JFrame  {
17
18     private JFrame frame;
19     private JPanel contentPane;
20     private boolean desconectado ;
21     private JLabel StatusConexo;
22     private JLabel LabelNomePaciente;
23     private JLabel LabelCodigo;
24     private JLabel LabelDatadeNascimento ;
25     private JLabel LabelDataTeste ;
26     private JLabel LabelMedico ;
27     private JLabel LabelConvenio ;
28     private JLabel LabelMedicoRequisitante ;
29     private String NomePaciente;
```

```
30      private String MedicoRequisitante;
31      private String Codigo;
32      private String DatadeNascimento ;
33      private String DataTeste ;
34      private String NomeMedico;
35      private String Convenio;
36      private String Perna;
37      private JTextArea Laudo;
38      private int NumTestes;
39      private boolean BotaoIniciarTeste;
40      private int VoltarHit;
41
42
43
44      public int MostrarJanelaIniciarTeste()  {
45
46          IniciarTeste();
47          LabelNomePaciente.setText(NomePaciente);
48          LabelCodigo.setText(Codigo);    //Seta codigo
49          LabelDatadeNascimento.setText(DatadeNascimento);    //Seta data da
        cirurgia
50          LabelDataTeste.setText(DataTeste);    //Seta data do teste
51          LabelMedico.setText(NomeMedico);    //Seta nome do medico
52          LabelConvenio.setText(Convenio);    //Seta convenio
53          frame.invalidate();
54          frame.validate();
55          frame.repaint();
56
57          return VoltarHit;
58      }
59
60      public void FecharJanelaIniciarTeste () throws InterruptedException
61      {    contentPane.removeAll();
62       frame.repaint();    }
63
64      /**
65       * Create the frame.
66       */
67      public void IniciarTeste() {
68          contentPane.removeAll();
69           frame.repaint();
70      if (frame!=null) {
71          VoltarHit =3;
72          frame.setBounds(100, 100, 400, 600);
73          JLabel label = new JLabel("Nome Paciente");
74          label.setHorizontalAlignment(SwingConstants.RIGHT);
75          label.setForeground(new Color(0, 0, 128));
```

```
76          label.setFont(new Font("Arial", Font.BOLD, 12));
77          label.setBackground(new Color(46, 139, 87));
78          label.setBounds(20, 280, 128, 14);
79          contentPane.add(label);
80
81          JLabel label_1 = new JLabel("C\u00F3digo");
82          label_1.setHorizontalAlignment(SwingConstants.RIGHT);
83          label_1.setForeground(new Color(0, 0, 128));
84          label_1.setFont(new Font("Arial", Font.BOLD, 12));
85          label_1.setBackground(new Color(46, 139, 87));
86          label_1.setBounds(20, 310, 128, 23);
87          contentPane.add(label_1);
88
89          JLabel label_2 = new JLabel("Data de Nasci.");
90          label_2.setHorizontalAlignment(SwingConstants.RIGHT);
91          label_2.setForeground(new Color(0, 0, 128));
92          label_2.setFont(new Font("Arial", Font.BOLD, 12));
93          label_2.setBackground(new Color(46, 139, 87));
94          label_2.setBounds(20, 345, 128, 14);
95          contentPane.add(label_2);
96
97          JLabel label_3 = new JLabel("Data Teste");
98          label_3.setHorizontalAlignment(SwingConstants.RIGHT);
99          label_3.setForeground(new Color(0, 0, 128));
100         label_3.setFont(new Font("Arial", Font.BOLD, 12));
101         label_3.setBackground(new Color(46, 139, 87));
102         label_3.setBounds(20, 375, 128, 14);
103         contentPane.add(label_3);
104
105         JLabel label_5 = new JLabel("Convenio");
106         label_5.setHorizontalAlignment(SwingConstants.RIGHT);
107         label_5.setForeground(new Color(0, 0, 128));
108         label_5.setFont(new Font("Arial", Font.BOLD, 12));
109         label_5.setBackground(new Color(46, 139, 87));
110         label_5.setBounds(10, 406, 138, 14);
111         contentPane.add(label_5);
112
113         JLabel label_4 = new JLabel("Medico Responsavel");
114         label_4.setHorizontalAlignment(SwingConstants.RIGHT);
115         label_4.setForeground(new Color(0, 0, 128));
116         label_4.setFont(new Font("Arial", Font.BOLD, 12));
117         label_4.setBackground(new Color(46, 139, 87));
118         label_4.setBounds(10, 450, 138, 14);
119         contentPane.add(label_4);
120
121
122         JButton btnIniciarTeste = new JButton("Iniciar Teste");
```

```
123
124
125         JButton btnVoltar = new JButton("Voltar");
126
127         btnVoltar.setBounds(201, 500, 118, 23);
128         contentPane.add(btnVoltar);
129         btnVoltar.addActionListener(new ActionListener() {
130             public void actionPerformed(ActionEvent arg0) {
131
132                 try {
133                     VoltarHit=2;
134                     FecharJanelaIniciarTeste();
135                 } catch (InterruptedException e) {
136                     // TODO Auto-generated catch block
137                     e.printStackTrace();
138                 }
139             }
140         });
141
142         LabelNomePaciente = new JLabel("New label");
143         LabelNomePaciente.setBounds(156, 277, 236, 20);
144         contentPane.add(LabelNomePaciente);
145
146
147         LabelCodigo = new JLabel("New label");
148         LabelCodigo.setBounds(158, 311, 136, 20);
149         contentPane.add(LabelCodigo);
150
151         LabelDatadeNascimento = new JLabel("New label");
152         LabelDatadeNascimento.setBounds(158, 342, 120, 20);
153         contentPane.add(LabelDatadeNascimento);
154
155         LabelDataTeste = new JLabel("New label");
156         LabelDataTeste.setBounds(158, 372, 120, 20);
157         contentPane.add(LabelDataTeste);
158
159         LabelMedico = new JLabel("New label");
160         LabelMedico.setBounds(158, 450, 224, 14);
161         contentPane.add(LabelMedico);
162
163         LabelConvenio = new JLabel("New label");
164         LabelConvenio.setBounds(158, 406, 224, 14);
165         contentPane.add(LabelConvenio);
166
167         JLabel lblStatusConeco = new JLabel("Status da conexo");
168         lblStatusConeco.setBounds(20, 122, 150, 14);
169         contentPane.add(lblStatusConeco);
```

```
170
171         StatusConexo = new JLabel("DESCONECTADO");
172         StatusConexo.setBounds(179, 122, 138, 14);
173         contentPane.add(StatusConexo);
174
175         btnIniciarTeste.setBounds(81, 500, 118, 23);
176         contentPane.add(btnIniciarTeste);
177         if (!desconectado) {
178             StatusConexo.setText("CONECTADO");
179             btnIniciarTeste.setEnabled(true);
180         }
181         else{
182             StatusConexo.setText("DESCONECTADO");
183             btnIniciarTeste.setEnabled(false);
184         }
185
186         btnIniciarTeste.addActionListener(new ActionListener() {
187             public void actionPerformed(ActionEvent arg0) {
188                 //BOTO INICIAR TESTE
189
190                 if(desconectado) {
191                     JOptionPane.showMessageDialog(null,
192                             "Para iniciar o teste, conecte-se antes a uma
    porta serial!",
193                             "Knee UP",
194                             JOptionPane.INFORMATION_MESSAGE);
195                 }
196                 else {
197                     System.out.println("MUDOU O BOTAO ");
198
199                     try {
200                         BotaoIniciarTeste= true;
201                         VoltarHit=4;
202                         FecharJanelaIniciarTeste();
203                         //realizarTeste();
204                     } catch (InterruptedException e) {
205                         // TODO Auto-generated catch block
206                         e.printStackTrace();
207                     }
208
209                 }
210             }
211         });
212
213         frame.add(contentPane, BorderLayout.CENTER);
214     }
215     }
```

```
216
217
218     public  void setNomePaciente(String labelNomePaciente) {
219         NomePaciente = labelNomePaciente;
220
221     }
222
223     public  void setCodigo(String labelCodigo) {
224         Codigo = labelCodigo;
225     }
226
227     public  void setDatadeNascimento(String labelDatadeNascimento) {
228         DatadeNascimento = labelDatadeNascimento;
229     }
230
231     public  void setDataTeste(String labelDataTeste) {
232         DataTeste = labelDataTeste;
233     }
234
235     public  void setNomeMedico(String labelNomeMedico) {
236         NomeMedico = labelNomeMedico;
237     }
238
239     public  void setConvenio(String labelConvenio) {
240         Convenio = labelConvenio;
241     }
242     public  void setMedicoRequisitante(String labelMedicoRequisitante) {
243         MedicoRequisitante = labelMedicoRequisitante;
244     }
245     public  void setPerna(String labelPerna) {
246         Perna = labelPerna;
247     }
248
249     public  boolean getClicouBotaoIniciarTeste() {
250         return this.BotaoIniciarTeste ;
251     }
252
253     public void setFrame(JFrame labelframe)
254     {
255         frame = labelframe;
256
257     }
258
259     public void setJPanel(JPanel labelPanel)
260     {
261         contentPane = labelPanel;
262
```

```
263     }
264     public void setNumTestes(int Num)
265     {
266         NumTestes = Num;
267 
268     }
269     public void setBotaoIniciarTeste(boolean botao) {
270         BotaoIniciarTeste = botao;
271     }
272 
273     public int getVoltar() {
274         return VoltarHit;
275     }
276 
277     public void SetDesconectado(boolean status) {
278         desconectado =status;
279     }
280 
281 }//FIM DA CLASSE
```

./dados/src/IniciarTeste.java

```
 1 package knee_up_v2;
 2 
 3 import java.awt.BorderLayout;
 4 import java.awt.Color;
 5 import java.awt.Font;
 6 import java.awt.event.ActionEvent;
 7 import java.awt.event.ActionListener;
 8 import java.io.IOException;
 9 import java.text.DateFormat;
10 import java.text.SimpleDateFormat;
11 import java.util.Calendar;
12 import java.util.Date;
13 
14 import javax.swing.ImageIcon;
15 import javax.swing.JButton;
16 import javax.swing.JFrame;
17 import javax.swing.JLabel;
18 import javax.swing.JMenuItem;
19 import javax.swing.JOptionPane;
20 import javax.swing.JPanel;
21 import javax.swing.JTextArea;
22 
23 import org.jfree.chart.ChartFrame;
24 import org.jfree.chart.JFreeChart;
25 
26 import com.itextpdf.text.DocumentException;
```

```
27
28 public class RealizarTeste {
29
30     private JFrame frame;
31     private JPanel contentPane;
32     private PortaSerial portaserial;
33
34
35     private String NomePaciente;
36     private String MedicoRequisitante;
37     private String Codigo;
38     private String DatadeNascimento ;
39     private String DataTeste ;
40     private String NomeMedico;
41     private String Convenio;
42     private String Perna;
43     private JTextArea Laudo;
44
45     private int NumTestes;
46     private int Step;
47
48     public int MostrarJanelaIniciarTeste()  {
49             try {
50                     realizarTeste();
51                 }
52             catch(IOException e) {
53                 System.out.println("IOException");}
54         frame.invalidate();
55         frame.validate();
56         frame.repaint();
57
58         return Step;
59     }
60     public void FecharJanelaIniciarTeste () throws InterruptedException
61
62     {
63
64     contentPane.removeAll();
65      frame.repaint();    }
66
67     public void realizarTeste() throws IOException
68     {
69         Step=4;
70         JOptionPane.showMessageDialog(null,"Ser Iniciada uma s rie de "+
       NumTestes +" Testes","Knee UP",JOptionPane.INFORMATION_MESSAGE);
71         Grafico[] graf = new Grafico[NumTestes];
72
```

```
73
74          for (int i = 0 ; i < NumTestes ; i ++ ) {
75              graf[i]= Teste(i+1);
76          }
77
78
79          frame.setBounds(100, 100, 1080, 900);
80          contentPane.removeAll();
81          for (int i = 0 ; i < NumTestes ; i ++ ) {
82              if(((i+1)%2)==0) {
83                  graf[i].getChartPanel().setBounds(540,20+320*((i)/2)
    ,500,300);
84              }
85              else {
86                  graf[i].getChartPanel().setBounds(20,20+320*((i)/2)
    ,500,300);
87              }
88              if(graf[i].getFrame()!=null) {
89              graf[i].closeWindow();
90              }
91              contentPane.add(graf[i].getChartPanel());
92          }
93
94
95          contentPane.revalidate(); // revalidei meu painel, para que ele se
    atualize
96          contentPane.repaint();
97
98          JLabel label_Laudo = new JLabel("Comentrios:");
99          label_Laudo.setForeground(new Color(0, 0, 128));
100         label_Laudo.setFont(new Font("Arial", Font.BOLD, 12));
101         label_Laudo.setBackground(new Color(46, 139, 87));
102         label_Laudo.setBounds(20, 650, 138, 14);
103         contentPane.add(label_Laudo);
104
105
106         Laudo = new JTextArea();
107         Laudo.setColumns(10);
108         Laudo.setBounds(20, 670 , 1000, 50);
109         Laudo.setText("Os seguintes aspectos foram observados: ");
110         contentPane.add(Laudo);
111
112
113         ImageIcon iconpdf = new ImageIcon("images/botao-salvar-pdf.png");
114         JButton button = new JButton("Salvar PDF ",iconpdf);
115         button.setBounds(20, 730,  iconpdf.getIconWidth(), iconpdf.
    getIconHeight());
```

```
116         button.setOpaque(false);
117         button.setContentAreaFilled(false);
118         button.setBorderPainted(false);
119         contentPane.add(button);
120         button.addActionListener(new ActionListener() {
121             public void actionPerformed(ActionEvent arg0) {
122 
123         Pdf relatorio = new Pdf();
124         try {
125 
126             // Recebe String com horrio atual no formato Hora Minuto
    Segundo
127             DateFormat df = new SimpleDateFormat("HHmmss");
128             Date today = Calendar.getInstance().getTime();
129             String now = df.format(today);
130 
131             String nomeArquivo = String.format(NomePaciente.replaceAll("\\s
    ","") + "-" + DataTeste.replaceAll("\\s","").replaceAll("/","") + now);
132             relatorio.createPdf(nomeArquivo + ".pdf");
133             System.out.println("pdf criado");
134             relatorio.setDadosPaciente(NomePaciente, Codigo,
    DatadeNascimento, DataTeste, NomeMedico, Convenio,
135                     Laudo.getText(), MedicoRequisitante, Perna);
136             System.out.println("pdf escrito");
137 
138             relatorio.setGraficos(graf);
139             System.out.println("pdf graficos");
140             relatorio.closePdf();
141             System.out.println("pdf fechado");
142         } catch (DocumentException e) {
143             System.out.println("erro no pdf");
144         }catch (IOException e)
145         {System.out.println("IOException");}
146 
147         JOptionPane.showMessageDialog(null,"Arquivo PDF criado!","Knee UP",
    JOptionPane.INFORMATION_MESSAGE);
148 
149         ///Set = 1;
150             }
151         });
152         ImageIcon iconfechar = new ImageIcon("images/botao-fechar.png");
153         JButton button_F = new JButton("Fechar",iconfechar);
154         button_F.setBounds(20+iconpdf.getIconWidth(), 730, iconfechar.
    getIconWidth(), iconfechar.getIconHeight());
155         button_F.setOpaque(false);
156         button_F.setContentAreaFilled(false);
157         button_F.setBorderPainted(false);
```

```
158         contentPane.add(button_F);
159         button_F.addActionListener(new ActionListener() {
160             public void actionPerformed(ActionEvent arg0) {
161                 try {
162                 FecharJanelaIniciarTeste();
163                 Step=5;}
164                 catch (InterruptedException e) {
165                     // TODO Auto-generated catch block
166                     e.printStackTrace();
167                 }
168 
169             }});
170         ImageIcon iconnovo = new ImageIcon("images/botao-novo-paciente.png"
    );
171         JButton BUtton_Voltar = new JButton("Novo Paciente",iconnovo);
172         BUtton_Voltar.setBounds(20+iconpdf.getIconWidth()*2, 730,iconnovo.
    getIconWidth(), iconnovo.getIconHeight());
173         BUtton_Voltar.setOpaque(false);
174         BUtton_Voltar.setContentAreaFilled(false);
175         BUtton_Voltar.setBorderPainted(false);
176         contentPane.add(BUtton_Voltar);
177         BUtton_Voltar.addActionListener(new ActionListener() {
178             public void actionPerformed(ActionEvent arg0) {
179                 try {
180                 FecharJanelaIniciarTeste();
181                 Step=2;}
182                 catch (InterruptedException e) {
183                     // TODO Auto-generated catch block
184                     e.printStackTrace();
185                 }
186 
187             }});
188         frame.add(contentPane,BorderLayout.CENTER);
189     }
190 
191     public Grafico Teste(int number)  throws IOException
192     {
193         JOptionPane.showMessageDialog(null,"TESTE "+number+" - Para iniciar
     pressione OK","Knee UP",JOptionPane.INFORMATION_MESSAGE);
194 
195         Grafico graf = portaserial.realizarTeste(number,NomePaciente,
    DataTeste);
196         return graf;
197     }
198 
199     public void setFrame(JFrame labelframe)
200     {
```

```
201         frame = labelframe;
202 
203     }
204 
205     public void setJPanel(JPanel labelPanel)
206     {
207         contentPane = labelPanel;
208 
209     }
210     public  void setNomePaciente(String labelNomePaciente) {
211         NomePaciente = labelNomePaciente;
212 
213     }
214 
215     public  void setCodigo(String labelCodigo) {
216         Codigo = labelCodigo;
217     }
218 
219     public  void setDatadeNascimento(String labelDatadeNascimento) {
220         DatadeNascimento = labelDatadeNascimento;
221     }
222 
223     public  void setDataTeste(String labelDataTeste) {
224         DataTeste = labelDataTeste;
225     }
226 
227     public  void setNomeMedico(String labelNomeMedico) {
228         NomeMedico = labelNomeMedico;
229     }
230 
231     public  void setConvenio(String labelConvenio) {
232         Convenio = labelConvenio;
233     }
234     public  void setMedicoRequisitante(String labelMedicoRequisitante) {
235         MedicoRequisitante = labelMedicoRequisitante;
236     }
237     public  void setPerna(String labelPerna) {
238         Perna = labelPerna;
239     }
240     public int getVoltar() {
241         return Step;
242     }
243 
244     public void setNumTestes(int Num)
245     {
246         NumTestes = Num;
247 
```

```
248     }
249     public  void setPorta(PortaSerial labelPorta) {
250         portaserial= labelPorta;
251     }
252 }
```

./dados/src/RealizarTeste.java

```
 1 package knee_up_v2;
 2
 3 import org.jfree.chart.ChartFactory;
 4 import org.jfree.chart.JFreeChart;
 5 import org.jfree.chart.plot.PlotOrientation;
 6 import org.jfree.data.xy.XYSeries;
 7 import org.jfree.data.xy.XYSeriesCollection;
 8 import org.jfree.chart.ChartFrame;
 9 import org.jfree.chart.ChartPanel;
10
11 public class Grafico {
12
13     private String TituloGrafico;
14     private Txt arquivo;
15     private JFreeChart chart;
16     private ChartFrame frame;
17     private ChartPanel myChartPanel;
18
19     public void GeraGrafico(String nomeArquivo, int numeroTeste) {
20
21         arquivo = new Txt();
22         arquivo.setNomeArquivo(nomeArquivo);
23         float[] dataTensao = arquivo.readTensao();
24         float[] dataTempo = arquivo.readTempo();
25
26
27         XYSeriesCollection dataset = new XYSeriesCollection();
28         XYSeries xySeries = new XYSeries("Teste " + numeroTeste);
29         XYSeries marker = new XYSeries("marcador");
30         float contador = 0;
31
32         for (contador = 0; contador != 1000; contador++) {
33             xySeries.add(dataTempo[(int) contador], dataTensao[(int)
    contador]);
34             marker.add(1,contador/200); // 1 segundo   quando acende o LED
    no arduino
35         }
36
37
38         dataset.addSeries(xySeries);
```

```
39          dataset.addSeries(marker);
40
41          TituloGrafico = "Knee-Up - " + numeroTeste;
42
43          chart = ChartFactory.createXYLineChart(TituloGrafico, "Tempos (s)",
        "Tenso (V)", dataset,
44                  PlotOrientation.VERTICAL, true, false, false);
45          myChartPanel = new ChartPanel(chart, true); //criei o painel de
       grafico colocando meu grafico previamente gerado
46          myChartPanel.setSize(500,300); //setei o tamanho do grafico
       conforme o painel que usarei
47          myChartPanel.setVisible(true);
48
49          frame = new ChartFrame(TituloGrafico, chart);
50          frame.setVisible(true);
51          frame.setSize(800, 650);
52      }
53
54      void closeWindow() {
55          frame.dispose();
56      }
57      JFreeChart getChart() {
58          return chart;
59
60      }
61      ChartFrame getFrame() {
62          return frame;
63
64      }
65      ChartPanel getChartPanel()
66      {
67          return myChartPanel;
68      }
69      void setChartPanel(ChartPanel panel)
70      {
71          this.myChartPanel =panel;
72      }
73 }
```

./dados/src/Grafico.java

```
1 package knee_up_v2;
2
3 import java.io.File;
4 import java.io.FileNotFoundException;
5 import java.text.DateFormat;
6 import java.text.SimpleDateFormat;
7 import java.util.Calendar;
```

```
 8 import java.util.Date;
 9 import java.util.Formatter;
10 import java.util.Locale;
11 import java.util.Scanner;
12
13 public class Txt {
14
15     private Formatter x;
16     private Scanner sc;
17     private String nomeArquivo;
18
19     public void openFile(int numFile,String NomePaciente, String DataTeste)
       {
20         try {
21             NomePaciente = NomePaciente.replaceAll("\\s",""); //Remove
       espaos das strings
22             DataTeste = DataTeste.replaceAll("\\s","");
23             DataTeste = DataTeste.replaceAll("/",""); //Remove barras das
       datas
24
25             // Recebe String com horrio atual no formato Hora Minuto
       Segundo
26             DateFormat df = new SimpleDateFormat("HHmmss");
27             Date today = Calendar.getInstance().getTime();
28             String now = df.format(today);
29
30             nomeArquivo = String.format("teste"+ numFile + "-"+
       NomePaciente + "-" + DataTeste + now + ".csv");
31             x = new Formatter(nomeArquivo);
32             System.out.println("abriu arquibo");
33         }
34         catch(Exception e) {
35             System.out.println("Erro ao abrir/criar o arquivo txt");
36         }
37     }
38
39     public void addTensao(double data) {
40         String s = String.format(Locale.US, "%.3f", data);
41         x.format(s + "%n");
42     }
43
44     public void addTempo(double data) {
45         String s = String.format(Locale.US, "%.3f", data);
46         x.format(s + ",");
47     }
48
49     public void closeFile() {
```

```
50          x.close();
51      }
52
53      public String getNomeArquivo() {
54          return nomeArquivo;
55      }
56      public void setNomeArquivo(String nome) {
57          nomeArquivo = nome;
58      }
59
60      float[] readTensao() {
61
62          float[] ret = new float[1000];
63          int i = 0;
64          String str;
65
66          try {
67              sc = new Scanner(new File(nomeArquivo));
68              System.out.println("Abriu arquivo " + nomeArquivo + " para
    leitura de tenso");
69          } catch (FileNotFoundException e) {
70              System.out.println("No abriu arquivo");
71          }
72
73          while(sc.hasNext()) {
74              str = sc.next();
75              str = str.substring(str.indexOf(",")+1,str.length());
76              ret[i] = Float.parseFloat(str);
77              i++;
78          }
79          return ret;
80
81      }
82
83      float[] readTempo() {
84
85          float[] ret = new float[1000];
86          int i = 0;
87          String str;
88
89          try {
90              sc = new Scanner(new File(nomeArquivo));
91              System.out.println("Abriu arquivo " + nomeArquivo + " para
    leitura de tempo");
92          } catch (FileNotFoundException e) {
93              System.out.println("No abriu arquivo");
94          }
```

```
95
96          while(sc.hasNext()) {
97              str = sc.next();
98              str = str.substring(0,str.indexOf(","));
99              ret[i] = Float.parseFloat(str);
100             i++;
101         }
102         return ret;
103
104     }
105
106 }
```

./dados/src/Txt.java

```
1  package knee_up_v2;
2
3  import java.awt.image.BufferedImage;
4  import java.io.FileNotFoundException;
5  import java.io.FileOutputStream;
6  import java.io.IOException;
7
8  import org.jfree.chart.JFreeChart;
9  import com.itextpdf.text.Document;
10 import com.itextpdf.text.DocumentException;
11 import com.itextpdf.text.Image;
12 import com.itextpdf.text.Paragraph;
13 import com.itextpdf.text.pdf.PdfWriter;
14 import com.itextpdf.text.Phrase;
15 import com.itextpdf.text.Font;
16 public class Pdf {
17
18     Document document;
19     PdfWriter writer;
20     Font vboldFont = new Font(Font.FontFamily.HELVETICA, 12, Font.BOLD);
21     Font normalFont = new Font(Font.FontFamily.HELVETICA, 12);
22
23     public void createPdf(String filename) throws FileNotFoundException,
       DocumentException {
24         document = new Document();
25         writer = PdfWriter.getInstance(document, new FileOutputStream(
       filename));
26         document.open();
27     }
28
29     public void closePdf() {
30         document.close();
31         writer.close();
```

```
32      }
33
34      public void setDadosPaciente(String NomePaciente, String Codigo, String
        DatadeNascimento, String DataTeste,
35              String NomeMedico, String Convenio, String Comentario, String
        MedicoRequisitante, String Perna) throws DocumentException {
36
37          Paragraph titulo = new Paragraph("Relatrio Knee-up",vboldFont);
38          Paragraph assinatura0 = new Paragraph("
        ----------------------------------------");
39          Paragraph assinatura2 = new Paragraph("Mdico Responsvel ",
        normalFont);
40          Paragraph assinatura1 = new Paragraph(NomeMedico, vboldFont);
41
42          titulo.setAlignment(Paragraph.ALIGN_CENTER);
43          assinatura0.setAlignment(Paragraph.ALIGN_CENTER);
44          assinatura1.setAlignment(Paragraph.ALIGN_CENTER);
45          assinatura2.setAlignment(Paragraph.ALIGN_CENTER);
46          document.add(new Phrase( "Nome do Paciente: ", vboldFont));
47          document.add(new Phrase(NomePaciente+"\n",normalFont));
48          document.add(new Phrase("Data de Nascimento: ", vboldFont));
49          document.add(new Phrase(DatadeNascimento+"\n",normalFont));
50          document.add(new Phrase("Teste: Knee-up - Perna "+Perna+"\n",
        vboldFont));
51          document.add(new Phrase("Data do Teste: ", vboldFont));
52          document.add(new Phrase(DataTeste+"\n",normalFont));
53          document.add(new Phrase("Cdigo: ", vboldFont));
54          document.add(new Phrase(Codigo+"\n",normalFont));
55          document.add(new Phrase("Convenio: " , vboldFont));
56          document.add(new Phrase(Convenio+"\n",normalFont));
57          document.add(new Phrase("Medico Requisitante: ", vboldFont));
58          document.add(new Phrase(MedicoRequisitante+"\n",normalFont));
59          document.add(new Paragraph(titulo));
60          document.add(new Paragraph(" "));
61          document.add(new Paragraph(" "));
62          document.add(new Paragraph(Comentario));
63          document.add(new Paragraph(" "));
64          document.add(new Paragraph(" "));
65          document.add(new Paragraph(" "));
66          document.add(new Paragraph(" "));
67          document.add(new Paragraph(" "));
68          document.add(new Paragraph(" "));
69          document.add(new Paragraph(" "));
70          document.add(assinatura0);
71          document.add(assinatura1);
72          document.add(assinatura2);
73
```

```
74
75          document.newPage();
76      }
77 public void setGraficos(Grafico[] chart1) throws IOException,
       DocumentException {
78
79          Paragraph titulo  = new Paragraph("Relatrio Knee-up -  Grficos ",
       vboldFont);
80          titulo.setAlignment(Paragraph.ALIGN_CENTER);
81          document.add(new Paragraph(titulo));
82          BufferedImage[] bufferedImage1 = new BufferedImage[chart1.length];
83
84          for (int i = 0 ; i < chart1.length ; i ++ ) {
85              bufferedImage1[i] = chart1[i].getChart().createBufferedImage
       (400, 300);
86              Image image1 = Image.getInstance(writer, bufferedImage1[i], 1.0
       f);
87              image1.setAlignment(Image.MIDDLE);
88              document.add(image1);
89          }
90
91      }
92 }
```

./dados/src/Pdf.java

```
1 #define DATA_TRANSFER_PERIOD 5
2 #define DATA_TRANSFER_WINDOW 5
3 #define PORTA_INDICADOR 8
4
5 String texto;
6 int analogPin = 0;
7 int val = 0;
8 bool led = 0;
9
10 void acendeLed(){
11   if(!led){
12      digitalWrite(LED_BUILTIN, HIGH);
13      digitalWrite(PORTA_INDICADOR, HIGH);
14      led = 1;
15   }
16 }
17
18 void apagaLed(){
19   if(led){
20      digitalWrite(LED_BUILTIN, LOW);
21      digitalWrite(PORTA_INDICADOR, LOW);
22      led = 0;
```

```
23   }
24 }
25
26 void serialFlush(){
27   while(Serial.available() > 0) {
28     char t = Serial.read();
29   }
30 }
31
32 void setup() {
33         Serial.begin(38400);
34         Serial.setTimeout(5);
35         pinMode(LED_BUILTIN,OUTPUT);
36         pinMode(PORTA_INDICADOR,OUTPUT);
37         apagaLed();
38 }
39
40 void loop() {
41
42       //teste de verifica   o do arduino
43        while(1){
44         texto = Serial.readString();
45         if (texto.length() > 0) {
46             break;
47         }
48        }
49        if(texto.equals("knee_up")){   //teste de verifica   o do arduino
50         Serial.write(5);
51        }
52
53
54        if(texto.equals("iniciarteste")){   //inicio de uma bateria de
     testes
55         apagaLed();
56         Serial.write(77);
57         int i = 0;
58         int tempo_inicio = millis();
59         int tempo;
60         while(i<((DATA_TRANSFER_WINDOW*1000)/DATA_TRANSFER_PERIOD)){
61            val = analogRead(analogPin);
62            tempo = millis() - tempo_inicio;
63            Serial.println(val);
64            Serial.println(tempo);
65            delay(DATA_TRANSFER_PERIOD);
66            i++;
67            if(tempo>=1000) {
68             acendeLed();
```

```
69                  }
70              }
71              apagaLed();
72              delay(500);
73              serialFlush();
74          }
75 }
```

./dados/src/arduino_knee_up.ino

APÊNDICE B – CÓDIGO FONTE FINAL

B.1 CÓDIGO FONTE EM PYTHON

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Tue Aug 14 18:41:22 2018
4
5 @author: Feupos
6 """
7
8 # needed packages
9 import sys
10 import random
11 import datetime
12 import threading
13 import time
14 import sched
15 import math
16 import functools
17 import multiprocessing
18 import csv
19 import os
20 from pathlib import Path
21
22 import numpy as np
23 # This gets the Qt stuff
24 from PyQt5 import QtWidgets, QtCore, QtGui
25 from PyQt5.QtWidgets import QApplication, QMainWindow
26 app = QApplication(sys.argv)
27
28
29
30 stub_gpio = False
31 try:
32     import RPi.GPIO as IO
33 except:
34     stub_gpio = True
35     print("Unable to import RPi.GPIO library")
36
37 # This is our window from QtCreator
38 import examinationManager
39 from popUp import errorPopUp
40
41
```

```
42 # Additional files
43 from singleton import singleton
44 from sessionData import sessionData
45 from pdfReport import pdfReport
46 from analogDataSource import analogDataSource
47 from plotObjects import realTimePlotWidget, MyStaticMplCanvas
48 from reportManager import saveReport
49
50 LED_STIMULI_PIN = 40
51
52
53 # functions to control led stimuli
54
55 if(stub_gpio == False):
56     IO.setmode(IO.BOARD)
57     IO.setup(LED_STIMULI_PIN,IO.OUT)
58
59 def ledStimuliON():
60     if(stub_gpio == False):
61         IO.output(LED_STIMULI_PIN,0)
62     else:
63         print("LED ON")
64
65 def ledStimuliOFF():
66     if(stub_gpio == False):
67         IO.output(LED_STIMULI_PIN,1)
68     else:
69         print("LED OFF")
70
71 class mainWindow(QMainWindow, examinationManager.Ui_MainWindow):
72  # access variables inside of the UI's file
73     def __init__(self):
74
75         super(self.__class__, self).__init__()
76         self.setupUi(self) # gets defined in the UI file
77         self.startButton.clicked.connect(self.startClick)
78         self.runTrialButton.clicked.connect(self.runTrialClick)
79         self.nextTrialButton.clicked.connect(self.nextTrialClick)
80         self.finishButton.clicked.connect(self.finishClick)
81         self.reportButton.clicked.connect(self.reportClick)
82         self.examinationTypeComboBox.addItem("Brake")
83         self.realTimePlot = realTimePlotWidget(self, app)
84         self.graphLayout.addWidget(self.realTimePlot)
85         self.showMaximized()
86         self.showFullScreen()
87         self.show()
88
```

```
89      def startClick(self):
90          #global currentSessionData
91          sessionData().updateData(self)
92          self.mainWidget.setCurrentIndex(1)
93          self.runTrialButton.setEnabled(True)
94          self.trialLengthSpinBox.setEnabled(True)
95          self.nextTrialButton.setEnabled(False)
96          self.finishButton.setEnabled(False)
97          self.trialCounter.display(0)
98          self.realTimePlot.start()
99
100     def runTrialClick(self):
101         self.runTrialButton.setEnabled(False)
102         self.nextTrialButton.setEnabled(False)
103         self.finishButton.setEnabled(False)
104         self.trialCounter.display(1 + int(self.trialCounter.value()))
105         QtCore.QTimer.singleShot(self.stimuliOffsetSpinBox.value()*1000,
    ledStimuliON)
106         self.realTimePlot.startTrial()
107
108     def nextTrialClick(self):
109         ledStimuliOFF()
110         self.runTrialButton.setEnabled(True)
111         self.trialLengthSpinBox.setEnabled(True)
112         self.nextTrialButton.setEnabled(False)
113         self.realTimePlot.start()
114
115     def finishClick(self):
116         ledStimuliOFF()
117         self.finishButton.setEnabled(False)
118         self.mainWidget.setCurrentIndex(2)
119         #currentSessionData.generatePDFReport()
120         for data in sessionData().trialData:
121             reportPlot = MyStaticMplCanvas()
122             reportPlot.compute(data["datasetx"],data["datasety"],data["
    stimuli"])
123             self.graphReportLayout.addWidget(reportPlot)
124
125     def reportClick(self):
126         self.setCursor(QtGui.QCursor(QtCore.Qt.WaitCursor))
127         try:
128             fileSaved = False
129             error = 0
130             if (len(self.fileNameLineEdit.text()) > 0):
131                 fileName = self.fileNameLineEdit.text()
132             else:
133                 fileName = "report"
```

```
134             sessionData().addComment(self.commentTextEdit.toPlainText())
135             fileSaved = saveReport(fileName)
136 
137         except Exception as e:
138             print(e)
139             self.setEnabled(False)
140             error = errorPopUp("Unable to save files.\n"+
141                                "Check if USB drive is connected!")
142             self.setEnabled(True)
143 
144         finally:
145             self.setCursor(QtGui.QCursor(QtCore.Qt.ArrowCursor))
146             if (fileSaved == True or error != 0):
147                 self.mainWidget.setCurrentIndex(0)
148                 for i in reversed(range(self.graphReportLayout.count())):
149                     self.graphReportLayout.itemAt(i).widget().deleteLater()
150 
151 #app = QApplication(sys.argv)
152 #form = mainWindow()
153 
154 
155 def main():
156     try:
157         print("Starting GUI")
158         window = mainWindow()
159         sys.exit(app.exec_())
160     except:
161         pass
162     finally:
163         pass
164 
165 if __name__ == "__main__":
166     main()
```

./dados/src/main.py

```
1  try:
2      import Adafruit_ADS1x15
3  except:
4      print("Unable to import ADS1115 library")
5  import math
6  import threading
7  import time
8  import numpy as np
9  from threading import Lock
10 
11 
12 dataLock = threading.Lock()
```

```
13 #from singleton import singleton
14
15 # Create an ADS1115 ADC (16-bit) instance.
16 # Choose a gain of 1 for reading voltages from 0 to 4.09V.
17 # Or pick a different gain to change the range of voltages that are read:
18 #  - 2/3 = +/-6.144V
19 #  -   1 = +/-4.096V
20 #  -   2 = +/-2.048V
21 #  -   4 = +/-1.024V
22 #  -   8 = +/-0.512V
23 #  -  16 = +/-0.256V
24 # See table 3 in the ADS1015/ADS1115 datasheet for more info on gain.
25 #ADC_TO_VOLTAGE = 0.000125
26 ADC_TO_VOLTAGE_GAIN_16 = 0.0000078125
27 ADC_TO_VOLTAGE_GAIN_8 = 0.000015625
28 #VOLT_TO_KGF = 0.018
29 VOLT_TO_KGF = 2000
30 ZERO_ADJUST = 0.25175
31 GAIN = 8
32 SAMPLE_RATE = 128
33
34
35 class analogDataSource(object):
36     def __init__(self):
37         print("Starting ADS1115 instance")
38         #self.gain = g;
39         #self.sample_rate = sr;
40         self.running = False
41         self.finished = False
42         self.datasety = []
43         self.datacount = 0
44         self.timelimit = 0
45         self.gain = GAIN
46         self.sample_rate = SAMPLE_RATE
47         try:
48             self.adc = Adafruit_ADS1x15.ADS1115()
49             #self.adc.start_adc_difference(0, gain=GAIN, data_rate=
    SAMPLE_RATE)
50             self.adc.start_adc(0, gain=GAIN, data_rate=SAMPLE_RATE)
51             self.STUBBED_MODE = False
52         except:
53             print("Running in stubbed mode")
54             self.STUBBED_MODE = True
55         self.running = True
56         self.thread = threading.Thread(target=self.updateData, args=())
57         self.thread.daemon = True                                # Daemonize
    thread
```

```
58          self.thread.start()                                     # Start the
    execution
59
60
61      def start(self, t = math.inf):
62          self.running = True
63          self.timelimit = t
64
65      def reset(self, t = math.inf):
66          self.datasety = []
67
68      def stop(self):
69          self.running = False
70
71      def getData(self):
72          if(self.STUBBED_MODE == False):
73              sample = (self.adc.get_last_result()*ADC_TO_VOLTAGE_GAIN_8-
    ZERO_ADJUST)*VOLT_TO_KGF
74              #sample = self.adc.read_adc_difference(0, self.gain)*0.000125
75          else:
76              sample = 2.5*(1+np.sin(2*np.pi*0.5*time.time()))
77          return sample
78
79      def updateData(self):
80          lostSamples = 0
81          t = time.time()
82          told = t
83          while(self.finished == False):
84              while(self.running == True):
85                  t = time.time()
86                  if (t-told > 1./self.sample_rate):
87                      #dataLock.acquire(True)
88                      self.datasety.append(self.getData())
89                      #dataLock.release()
90                      try:
91                          if(t-told > 2*1./self.sample_rate):
92                              raise Exception("Unable to sample data within
    time")
93                      except:
94                          lostSamples = lostSamples + 1
95                          print("{0} lost samples".format(lostSamples))
96                      told = t
97              def __exit__(self):
98                  self.running = False
99                  self.finished = True
```

./dados/src/analogDataSource.py

```
1 # -*- coding: utf-8 -*-
2
3 # Form implementation generated from reading ui file 'ui\errordialog.ui'
4 #
5 # Created by: PyQt5 UI code generator 5.9.2
6 #
7 # WARNING! All changes made in this file will be lost!
8
9 from PyQt5 import QtCore, QtGui, QtWidgets
10
11 class Ui_errorDialog(object):
12     def setupUi(self, errorDialog):
13         errorDialog.setObjectName("errorDialog")
14         errorDialog.resize(240, 100)
15         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Fixed,
    QtWidgets.QSizePolicy.Fixed)
16         sizePolicy.setHorizontalStretch(0)
17         sizePolicy.setVerticalStretch(0)
18         sizePolicy.setHeightForWidth(errorDialog.sizePolicy().
    hasHeightForWidth())
19         errorDialog.setSizePolicy(sizePolicy)
20         icon = QtGui.QIcon()
21         icon.addPixmap(QtGui.QPixmap("../resources/error.png"), QtGui.QIcon
    .Normal, QtGui.QIcon.Off)
22         errorDialog.setWindowIcon(icon)
23         self.textField = QtWidgets.QLabel(errorDialog)
24         self.textField.setGeometry(QtCore.QRect(11, 11, 221, 41))
25         self.textField.setScaledContents(True)
26         self.textField.setAlignment(QtCore.Qt.AlignCenter)
27         self.textField.setWordWrap(True)
28         self.textField.setIndent(-1)
29         self.textField.setObjectName("textField")
30         self.buttonBox = QtWidgets.QDialogButtonBox(errorDialog)
31         self.buttonBox.setEnabled(True)
32         self.buttonBox.setGeometry(QtCore.QRect(25, 60, 193, 28))
33         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Fixed,
    QtWidgets.QSizePolicy.Fixed)
34         sizePolicy.setHorizontalStretch(0)
35         sizePolicy.setVerticalStretch(0)
36         sizePolicy.setHeightForWidth(self.buttonBox.sizePolicy().
    hasHeightForWidth())
37         self.buttonBox.setSizePolicy(sizePolicy)
38         self.buttonBox.setOrientation(QtCore.Qt.Horizontal)
39         self.buttonBox.setStandardButtons(QtWidgets.QDialogButtonBox.Close|
    QtWidgets.QDialogButtonBox.Retry)
40         self.buttonBox.setCenterButtons(True)
41         self.buttonBox.setObjectName("buttonBox")
```

```
42
43          self.retranslateUi(errorDialog)
44          self.buttonBox.accepted.connect(errorDialog.accept)
45          self.buttonBox.rejected.connect(errorDialog.reject)
46          QtCore.QMetaObject.connectSlotsByName(errorDialog)
47
48      def retranslateUi(self, errorDialog):
49          _translate = QtCore.QCoreApplication.translate
50          errorDialog.setWindowTitle(_translate("errorDialog", "Error"))
51          self.textField.setText(_translate("errorDialog", "ERROR!\n"
52 "lol"))
53
54
55 if __name__ == "__main__":
56      import sys
57      app = QtWidgets.QApplication(sys.argv)
58      errorDialog = QtWidgets.QDialog()
59      ui = Ui_errorDialog()
60      ui.setupUi(errorDialog)
61      errorDialog.show()
62      sys.exit(app.exec_())
```

./dados/src/errorDialog.py

```
1 # -*- coding: utf-8 -*-
2
3 # Form implementation generated from reading ui file 'examinationManager.ui
      '
4 #
5 # Created by: PyQt5 UI code generator 5.9.2
6 #
7 # WARNING! All changes made in this file will be lost!
8
9 from PyQt5 import QtCore, QtGui, QtWidgets
10
11 class Ui_MainWindow(object):
12      def setupUi(self, MainWindow):
13          MainWindow.setObjectName("MainWindow")
14          MainWindow.setWindowModality(QtCore.Qt.NonModal)
15          MainWindow.resize(640, 480)
16          sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Expanding,
       QtWidgets.QSizePolicy.Expanding)
17          sizePolicy.setHorizontalStretch(0)
18          sizePolicy.setVerticalStretch(0)
19          sizePolicy.setHeightForWidth(MainWindow.sizePolicy().
     hasHeightForWidth())
20          MainWindow.setSizePolicy(sizePolicy)
21          MainWindow.setAutoFillBackground(True)
```

```
22          MainWindow.setLocale(QtCore.QLocale(QtCore.QLocale.English, QtCore.
    QLocale.UnitedStates))
23          self.centralwidget = QtWidgets.QWidget(MainWindow)
24          self.centralwidget.setEnabled(True)
25          self.centralwidget.setObjectName("centralwidget")
26          self.horizontalLayout = QtWidgets.QHBoxLayout(self.centralwidget)
27          self.horizontalLayout.setObjectName("horizontalLayout")
28          self.mainWidget = QtWidgets.QStackedWidget(self.centralwidget)
29          self.mainWidget.setFrameShape(QtWidgets.QFrame.NoFrame)
30          self.mainWidget.setObjectName("mainWidget")
31          self.mainPage = QtWidgets.QWidget()
32          self.mainPage.setObjectName("mainPage")
33          self.patientData = QtWidgets.QFormLayout(self.mainPage)
34          self.patientData.setObjectName("patientData")
35          self.patientLabel = QtWidgets.QLabel(self.mainPage)
36          self.patientLabel.setObjectName("patientLabel")
37          self.patientData.setWidget(0, QtWidgets.QFormLayout.LabelRole, self
    .patientLabel)
38          self.patientLineEdit = QtWidgets.QLineEdit(self.mainPage)
39          self.patientLineEdit.setObjectName("patientLineEdit")
40          self.patientData.setWidget(0, QtWidgets.QFormLayout.FieldRole, self
    .patientLineEdit)
41          self.dateOfBirthLabel = QtWidgets.QLabel(self.mainPage)
42          self.dateOfBirthLabel.setObjectName("dateOfBirthLabel")
43          self.patientData.setWidget(1, QtWidgets.QFormLayout.LabelRole, self
    .dateOfBirthLabel)
44          self.dateOfBirthDateEdit = QtWidgets.QDateEdit(self.mainPage)
45          self.dateOfBirthDateEdit.setObjectName("dateOfBirthDateEdit")
46          self.patientData.setWidget(1, QtWidgets.QFormLayout.FieldRole, self
    .dateOfBirthDateEdit)
47          self.idNumberLabel = QtWidgets.QLabel(self.mainPage)
48          self.idNumberLabel.setObjectName("idNumberLabel")
49          self.patientData.setWidget(2, QtWidgets.QFormLayout.LabelRole, self
    .idNumberLabel)
50          self.idNumberLineEdit = QtWidgets.QLineEdit(self.mainPage)
51          self.idNumberLineEdit.setObjectName("idNumberLineEdit")
52          self.patientData.setWidget(2, QtWidgets.QFormLayout.FieldRole, self
    .idNumberLineEdit)
53          self.insurancePlanLabel = QtWidgets.QLabel(self.mainPage)
54          self.insurancePlanLabel.setObjectName("insurancePlanLabel")
55          self.patientData.setWidget(3, QtWidgets.QFormLayout.LabelRole, self
    .insurancePlanLabel)
56          self.insurancePlanLineEdit = QtWidgets.QLineEdit(self.mainPage)
57          self.insurancePlanLineEdit.setObjectName("insurancePlanLineEdit")
58          self.patientData.setWidget(3, QtWidgets.QFormLayout.FieldRole, self
    .insurancePlanLineEdit)
59          self.requestingDoctorLabel = QtWidgets.QLabel(self.mainPage)
```

```
60          self.requestingDoctorLabel.setObjectName("requestingDoctorLabel")
61          self.patientData.setWidget(4, QtWidgets.QFormLayout.LabelRole, self
    .requestingDoctorLabel)
62          self.requestingDoctorLineEdit = QtWidgets.QLineEdit(self.mainPage)
63          self.requestingDoctorLineEdit.setObjectName("
    requestingDoctorLineEdit")
64          self.patientData.setWidget(4, QtWidgets.QFormLayout.FieldRole, self
    .requestingDoctorLineEdit)
65          self.attendingDoctorLabel = QtWidgets.QLabel(self.mainPage)
66          self.attendingDoctorLabel.setObjectName("attendingDoctorLabel")
67          self.patientData.setWidget(5, QtWidgets.QFormLayout.LabelRole, self
    .attendingDoctorLabel)
68          self.attendingDoctorLineEdit = QtWidgets.QLineEdit(self.mainPage)
69          self.attendingDoctorLineEdit.setObjectName("attendingDoctorLineEdit
    ")
70          self.patientData.setWidget(5, QtWidgets.QFormLayout.FieldRole, self
    .attendingDoctorLineEdit)
71          self.examinationTypeLabel = QtWidgets.QLabel(self.mainPage)
72          self.examinationTypeLabel.setObjectName("examinationTypeLabel")
73          self.patientData.setWidget(6, QtWidgets.QFormLayout.LabelRole, self
    .examinationTypeLabel)
74          self.examinationTypeComboBox = QtWidgets.QComboBox(self.mainPage)
75          self.examinationTypeComboBox.setEditable(False)
76          self.examinationTypeComboBox.setObjectName("examinationTypeComboBox
    ")
77          self.patientData.setWidget(6, QtWidgets.QFormLayout.FieldRole, self
    .examinationTypeComboBox)
78          self.numberOfTrialsLabel = QtWidgets.QLabel(self.mainPage)
79          self.numberOfTrialsLabel.setObjectName("numberOfTrialsLabel")
80          self.patientData.setWidget(7, QtWidgets.QFormLayout.LabelRole, self
    .numberOfTrialsLabel)
81          self.numberOfTrialsSpinBox = QtWidgets.QSpinBox(self.mainPage)
82          self.numberOfTrialsSpinBox.setMinimum(1)
83          self.numberOfTrialsSpinBox.setMaximum(5)
84          self.numberOfTrialsSpinBox.setProperty("value", 1)
85          self.numberOfTrialsSpinBox.setObjectName("numberOfTrialsSpinBox")
86          self.patientData.setWidget(7, QtWidgets.QFormLayout.FieldRole, self
    .numberOfTrialsSpinBox)
87          self.startButton = QtWidgets.QPushButton(self.mainPage)
88          self.startButton.setStyleSheet("")
89          self.startButton.setObjectName("startButton")
90          self.patientData.setWidget(8, QtWidgets.QFormLayout.FieldRole, self
    .startButton)
91          self.mainWidget.addWidget(self.mainPage)
92          self.graphPage = QtWidgets.QWidget()
93          self.graphPage.setObjectName("graphPage")
94          self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.graphPage)
```

```
95          self.horizontalLayout_2.setObjectName("horizontalLayout_2")
96          self.verticalWidget = QtWidgets.QWidget(self.graphPage)
97          sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Expanding,
     QtWidgets.QSizePolicy.Expanding)
98          sizePolicy.setHorizontalStretch(0)
99          sizePolicy.setVerticalStretch(0)
100         sizePolicy.setHeightForWidth(self.verticalWidget.sizePolicy().
    hasHeightForWidth())
101         self.verticalWidget.setSizePolicy(sizePolicy)
102         self.verticalWidget.setObjectName("verticalWidget")
103         self.graphLayout = QtWidgets.QVBoxLayout(self.verticalWidget)
104         self.graphLayout.setObjectName("graphLayout")
105         self.horizontalLayout_2.addWidget(self.verticalWidget)
106         self.verticalGroupBox_2 = QtWidgets.QGroupBox(self.graphPage)
107         self.verticalGroupBox_2.setEnabled(True)
108         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Fixed,
    QtWidgets.QSizePolicy.Preferred)
109         sizePolicy.setHorizontalStretch(0)
110         sizePolicy.setVerticalStretch(0)
111         sizePolicy.setHeightForWidth(self.verticalGroupBox_2.sizePolicy().
    hasHeightForWidth())
112         self.verticalGroupBox_2.setSizePolicy(sizePolicy)
113         self.verticalGroupBox_2.setObjectName("verticalGroupBox_2")
114         self.controlsLayout = QtWidgets.QVBoxLayout(self.verticalGroupBox_2
    )
115         self.controlsLayout.setSizeConstraint(QtWidgets.QLayout.
    SetDefaultConstraint)
116         self.controlsLayout.setObjectName("controlsLayout")
117         self.horizontalGroupBox = QtWidgets.QGroupBox(self.
    verticalGroupBox_2)
118         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred,
     QtWidgets.QSizePolicy.Fixed)
119         sizePolicy.setHorizontalStretch(0)
120         sizePolicy.setVerticalStretch(0)
121         sizePolicy.setHeightForWidth(self.horizontalGroupBox.sizePolicy().
    hasHeightForWidth())
122         self.horizontalGroupBox.setSizePolicy(sizePolicy)
123         self.horizontalGroupBox.setObjectName("horizontalGroupBox")
124         self.horizontalLayout_3 = QtWidgets.QHBoxLayout(self.
    horizontalGroupBox)
125         self.horizontalLayout_3.setObjectName("horizontalLayout_3")
126         self.trialCounter = QtWidgets.QLCDNumber(self.horizontalGroupBox)
127         self.trialCounter.setSmallDecimalPoint(False)
128         self.trialCounter.setDigitCount(1)
129         self.trialCounter.setSegmentStyle(QtWidgets.QLCDNumber.Flat)
130         self.trialCounter.setObjectName("trialCounter")
131         self.horizontalLayout_3.addWidget(self.trialCounter)
```

```
132         self.controlsLayout.addWidget(self.horizontalGroupBox)
133         self.verticalGroupBox = QtWidgets.QGroupBox(self.verticalGroupBox_2
    )
134         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred,
     QtWidgets.QSizePolicy.Fixed)
135         sizePolicy.setHorizontalStretch(0)
136         sizePolicy.setVerticalStretch(0)
137         sizePolicy.setHeightForWidth(self.verticalGroupBox.sizePolicy().
    hasHeightForWidth())
138         self.verticalGroupBox.setSizePolicy(sizePolicy)
139         self.verticalGroupBox.setFlat(False)
140         self.verticalGroupBox.setCheckable(False)
141         self.verticalGroupBox.setObjectName("verticalGroupBox")
142         self.verticalLayout = QtWidgets.QVBoxLayout(self.verticalGroupBox)
143         self.verticalLayout.setSizeConstraint(QtWidgets.QLayout.
    SetDefaultConstraint)
144         self.verticalLayout.setObjectName("verticalLayout")
145         self.trialLengthSpinBox = QtWidgets.QDoubleSpinBox(self.
    verticalGroupBox)
146         self.trialLengthSpinBox.setPrefix("")
147         self.trialLengthSpinBox.setDecimals(1)
148         self.trialLengthSpinBox.setSingleStep(0.1)
149         self.trialLengthSpinBox.setProperty("value", 5.0)
150         self.trialLengthSpinBox.setObjectName("trialLengthSpinBox")
151         self.verticalLayout.addWidget(self.trialLengthSpinBox)
152         self.controlsLayout.addWidget(self.verticalGroupBox)
153         self.groupBox = QtWidgets.QGroupBox(self.verticalGroupBox_2)
154         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred,
     QtWidgets.QSizePolicy.Fixed)
155         sizePolicy.setHorizontalStretch(0)
156         sizePolicy.setVerticalStretch(0)
157         sizePolicy.setHeightForWidth(self.groupBox.sizePolicy().
    hasHeightForWidth())
158         self.groupBox.setSizePolicy(sizePolicy)
159         self.groupBox.setObjectName("groupBox")
160         self.verticalLayout_2 = QtWidgets.QVBoxLayout(self.groupBox)
161         self.verticalLayout_2.setObjectName("verticalLayout_2")
162         self.stimuliOffsetSpinBox = QtWidgets.QDoubleSpinBox(self.groupBox)
163         self.stimuliOffsetSpinBox.setSingleStep(0.1)
164         self.stimuliOffsetSpinBox.setProperty("value", 1.0)
165         self.stimuliOffsetSpinBox.setObjectName("stimuliOffsetSpinBox")
166         self.verticalLayout_2.addWidget(self.stimuliOffsetSpinBox)
167         self.controlsLayout.addWidget(self.groupBox)
168         self.runTrialButton = QtWidgets.QPushButton(self.verticalGroupBox_2
    )
169         self.runTrialButton.setObjectName("runTrialButton")
170         self.controlsLayout.addWidget(self.runTrialButton)
```

```
171         self.nextTrialButton = QtWidgets.QPushButton(self.
    verticalGroupBox_2)
172         self.nextTrialButton.setEnabled(False)
173         self.nextTrialButton.setObjectName("nextTrialButton")
174         self.controlsLayout.addWidget(self.nextTrialButton)
175         self.finishButton = QtWidgets.QPushButton(self.verticalGroupBox_2)
176         self.finishButton.setEnabled(False)
177         self.finishButton.setObjectName("finishButton")
178         self.controlsLayout.addWidget(self.finishButton)
179         self.horizontalLayout_2.addWidget(self.verticalGroupBox_2)
180         self.mainWidget.addWidget(self.graphPage)
181         self.reportPage = QtWidgets.QWidget()
182         self.reportPage.setObjectName("reportPage")
183         self.verticalLayout_4 = QtWidgets.QVBoxLayout(self.reportPage)
184         self.verticalLayout_4.setObjectName("verticalLayout_4")
185         self.scrollArea = QtWidgets.QScrollArea(self.reportPage)
186         self.scrollArea.setWidgetResizable(True)
187         self.scrollArea.setObjectName("scrollArea")
188         self.scrollAreaWidgetContents = QtWidgets.QWidget()
189         self.scrollAreaWidgetContents.setGeometry(QtCore.QRect(0, 0, 98,
    22))
190         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred,
     QtWidgets.QSizePolicy.Fixed)
191         sizePolicy.setHorizontalStretch(0)
192         sizePolicy.setVerticalStretch(0)
193         sizePolicy.setHeightForWidth(self.scrollAreaWidgetContents.
    sizePolicy().hasHeightForWidth())
194         self.scrollAreaWidgetContents.setSizePolicy(sizePolicy)
195         self.scrollAreaWidgetContents.setObjectName("
    scrollAreaWidgetContents")
196         self.graphReportLayout = QtWidgets.QVBoxLayout(self.
    scrollAreaWidgetContents)
197         self.graphReportLayout.setObjectName("graphReportLayout")
198         self.scrollArea.setWidget(self.scrollAreaWidgetContents)
199         self.verticalLayout_4.addWidget(self.scrollArea)
200         self.groupBox_2 = QtWidgets.QGroupBox(self.reportPage)
201         sizePolicy = QtWidgets.QSizePolicy(QtWidgets.QSizePolicy.Preferred,
     QtWidgets.QSizePolicy.Fixed)
202         sizePolicy.setHorizontalStretch(0)
203         sizePolicy.setVerticalStretch(0)
204         sizePolicy.setHeightForWidth(self.groupBox_2.sizePolicy().
    hasHeightForWidth())
205         self.groupBox_2.setSizePolicy(sizePolicy)
206         self.groupBox_2.setMaximumSize(QtCore.QSize(16777215, 150))
207         self.groupBox_2.setObjectName("groupBox_2")
208         self.formLayout = QtWidgets.QFormLayout(self.groupBox_2)
209         self.formLayout.setObjectName("formLayout")
```

```
210         self.label_2 = QtWidgets.QLabel(self.groupBox_2)
211         self.label_2.setObjectName("label_2")
212         self.formLayout.setWidget(1, QtWidgets.QFormLayout.LabelRole, self.
    label_2)
213         self.commentTextEdit = QtWidgets.QPlainTextEdit(self.groupBox_2)
214         self.commentTextEdit.setObjectName("commentTextEdit")
215         self.formLayout.setWidget(1, QtWidgets.QFormLayout.FieldRole, self.
    commentTextEdit)
216         self.reportButton = QtWidgets.QPushButton(self.groupBox_2)
217         self.reportButton.setObjectName("reportButton")
218         self.formLayout.setWidget(3, QtWidgets.QFormLayout.FieldRole, self.
    reportButton)
219         self.fileNameLineEdit = QtWidgets.QLineEdit(self.groupBox_2)
220         self.fileNameLineEdit.setObjectName("fileNameLineEdit")
221         self.formLayout.setWidget(2, QtWidgets.QFormLayout.FieldRole, self.
    fileNameLineEdit)
222         self.label = QtWidgets.QLabel(self.groupBox_2)
223         self.label.setObjectName("label")
224         self.formLayout.setWidget(2, QtWidgets.QFormLayout.LabelRole, self.
    label)
225         self.verticalLayout_4.addWidget(self.groupBox_2)
226         self.mainWidget.addWidget(self.reportPage)
227         self.horizontalLayout.addWidget(self.mainWidget)
228         MainWindow.setCentralWidget(self.centralwidget)
229         self.menubar = QtWidgets.QMenuBar(MainWindow)
230         self.menubar.setGeometry(QtCore.QRect(0, 0, 640, 26))
231         self.menubar.setObjectName("menubar")
232         self.menuMenu = QtWidgets.QMenu(self.menubar)
233         self.menuMenu.setObjectName("menuMenu")
234         self.menuHelp = QtWidgets.QMenu(self.menubar)
235         self.menuHelp.setObjectName("menuHelp")
236         MainWindow.setMenuBar(self.menubar)
237         self.actionAbout = QtWidgets.QAction(MainWindow)
238         self.actionAbout.setObjectName("actionAbout")
239         self.actionConfiguration = QtWidgets.QAction(MainWindow)
240         self.actionConfiguration.setObjectName("actionConfiguration")
241         self.menuMenu.addAction(self.actionConfiguration)
242         self.menuHelp.addAction(self.actionAbout)
243         self.menubar.addAction(self.menuMenu.menuAction())
244         self.menubar.addAction(self.menuHelp.menuAction())
245
246         self.retranslateUi(MainWindow)
247         self.mainWidget.setCurrentIndex(0)
248         QtCore.QMetaObject.connectSlotsByName(MainWindow)
249
250     def retranslateUi(self, MainWindow):
251         _translate = QtCore.QCoreApplication.translate
```

```
252         MainWindow.setWindowTitle(_translate("MainWindow", "Examination
    Manager"))
253         self.patientLabel.setText(_translate("MainWindow", "Patient"))
254         self.dateOfBirthLabel.setText(_translate("MainWindow", "Date of
    Birth"))
255         self.idNumberLabel.setText(_translate("MainWindow", "ID Number"))
256         self.insurancePlanLabel.setText(_translate("MainWindow", "Insurance
     Plan"))
257         self.requestingDoctorLabel.setText(_translate("MainWindow", "
    Requesting Doctor"))
258         self.attendingDoctorLabel.setText(_translate("MainWindow", "
    Attending Doctor"))
259         self.examinationTypeLabel.setText(_translate("MainWindow", "
    Examination Type"))
260         self.numberOfTrialsLabel.setText(_translate("MainWindow", "Number
    of Trials"))
261         self.startButton.setText(_translate("MainWindow", "Start"))
262         self.verticalGroupBox_2.setTitle(_translate("MainWindow", "Controls
    "))
263         self.horizontalGroupBox.setTitle(_translate("MainWindow", "Trial"))
264         self.verticalGroupBox.setTitle(_translate("MainWindow", "Trial
    Length"))
265         self.trialLengthSpinBox.setSuffix(_translate("MainWindow", " s"))
266         self.groupBox.setTitle(_translate("MainWindow", "Stimuli offset"))
267         self.runTrialButton.setText(_translate("MainWindow", "Run Trial"))
268         self.nextTrialButton.setText(_translate("MainWindow", "Next"))
269         self.finishButton.setText(_translate("MainWindow", "Finish"))
270         self.groupBox_2.setTitle(_translate("MainWindow", "Report Options")
    )
271         self.label_2.setText(_translate("MainWindow", "Comments"))
272         self.reportButton.setText(_translate("MainWindow", "Generate report
    "))
273         self.label.setText(_translate("MainWindow", "File Name"))
274         self.menuMenu.setTitle(_translate("MainWindow", "Menu"))
275         self.menuHelp.setTitle(_translate("MainWindow", "Help"))
276         self.actionAbout.setText(_translate("MainWindow", "About"))
277         self.actionConfiguration.setText(_translate("MainWindow", "
    Configuration"))
278
279
280 if __name__ == "__main__":
281     import sys
282     app = QtWidgets.QApplication(sys.argv)
283     MainWindow = QtWidgets.QMainWindow()
284     ui = Ui_MainWindow()
285     ui.setupUi(MainWindow)
286     MainWindow.show()
```

```
287     sys.exit(app.exec_())
```

./dados/src/examinationManager.py

```
1 import os
2 import matplotlib.pyplot as plt
3 from fpdf import FPDF
4
5 class PDF(FPDF):
6     def header(self):
7         pass
8         # Logo
9         #self.image('buffer1.png', 10, 8, 33)
10        # Arial bold 15
11        #self.set_font('Arial', 'B', 15)
12        # Move to the right
13        #self.cell(80)
14        # Title
15        #self.cell(30, 10, 'Title', 1, 0, 'C')
16        # Line break
17        #self.ln(20)
18
19    # Page footer
20    def footer(self):
21        # Position at 1.5 cm from bottom
22        self.set_y(-15)
23        # Arial italic 8
24        self.set_font('Arial', 'I', 8)
25        # Page number
26        self.cell(0, 10, 'Page ' + str(self.page_no()) + '/{nb}', 0, 0, 'C'
      )
27
28 # Instantiation of inherited class
29
30 def pdfReport(fileName, data):
31     pdf = PDF()
32     pdf.alias_nb_pages()
33     pdf.add_page()
34     pdf.set_font('Times', '', 12)
35
36     for element in data.textData:
37         ptext = "{0}: {1}".format(element["label"],element["data"])
38         pdf.cell(0, 10, ptext, 0, 1)
39
40     n = 0
41     for element in data.trialData:
42         n = n+1
43         fig, ax = plt.subplots()
```

```
44          ax.plot(element["datasetx"], element["datasety"])
45          ax.axvline(x=element["stimuli"], color = "r")
46          plt.savefig("buffer.png")
47          pdf.image("buffer.png")
48          plt.close()
49          os.remove("buffer.png")
50
51      ptext = "Comments:\n"+data.comment
52      pdf.cell(0, 10, ptext, 0, 1)
53
54      pdf.output(fileName, 'F')
```

./dados/src/pdfReport.py

```
1 import matplotlib
2 import pyqtgraph as pg
3 import numpy as np
4 from numpy import arange, sin, pi
5 #from scipy import signal
6 from matplotlib.backends.backend_qt5agg import FigureCanvasQTAgg as
      FigureCanvas
7 from matplotlib.figure import Figure
8 from PyQt5 import QtWidgets, QtCore, QtGui
9 from analogDataSource import analogDataSource
10 from sessionData import sessionData
11
12 #self.mainWindow = self.mainWindow
13
14 GAIN = 16
15 SAMPLE_RATE = 128
16
17 class realTimePlotWidget(pg.PlotWidget):
18     def __init__(self, window, aplication, t = 5):
19         pg.PlotWidget.__init__(self)
20         self.mainWindow = window
21         self.app = aplication
22         self.plotTimer = QtCore.QTimer(self)
23         self.plotTimer.timeout.connect(self.updatePlot)
24         self.dataSource = analogDataSource()
25         self.dataSource.start()
26         self.running = False
27         self.runTrial = False
28         self.timeLimit = t
29         self.datasety = []
30         self.datasetx = []
31         #self.setMenuEnabled(enableMenu=False)
32         self.curve = self.plot(self.datasetx, self.datasety, pen=(255,0,0))
33         self.setYRange(0,5) #modify this
```

```
34          self.plotTimer.start(100)
35          #self.b , self.a = signal.butter(8,30/(SAMPLE_RATE/2),btype = "
    lowpass")
36
37      def start(self):
38          self.datasety = []
39          self.datasetx = []
40          self.dataSource.running = True
41          self.running = True
42
43      def stop(self):
44          self.running = False
45
46      def startTrial(self):
47          self.runTrial = True
48          self.start()
49
50
51      def updatePlot(self):
52          if (self.running == True):
53              self.timeLimit = self.mainWindow.trialLengthSpinBox.value()
54              #dataLock.acquire(True)
55              self.datasety = self.datasety + self.dataSource.datasety
56              self.dataSource.datasety = []
57
58              if(len(self.datasety) >= SAMPLE_RATE*self.timeLimit+1):
59                  del self.datasety[0:len(self.datasety) - int(SAMPLE_RATE*
    self.timeLimit)]
60                  if(self.runTrial):
61                      self.stop()
62                      self.dataSource.stop()
63                      self.dataSource.reset()
64                      #ledStimuliOFF()
65                      self.runTrial = False
66                      self.datasetx = np.linspace(0,len(self.datasety)/
    SAMPLE_RATE, num = len(self.datasety))
67                      self.savePlotData()
68
69                      if(self.mainWindow.trialCounter.value() < self.
    mainWindow.numberOfTrialsSpinBox.value()):
70                          self.mainWindow.nextTrialButton.setEnabled(True)
71                      else:
72                          self.mainWindow.finishButton.setEnabled(True)
73
74              self.datasetx = np.linspace(0,len(self.datasety)/SAMPLE_RATE,
    num = len(self.datasety) ,endpoint = False)
75              if (len(self.datasety) > 100):
```

```
76                  #self.datafilt = signal.filtfilt(self.b, self.a, self.
    datasety)
77                  self.curve.setData(self.datasetx, self.datasety)
78                  self.setDownsampling(ds=True,auto=True,mode='subsample')
79              #self.setDownsampling(mode='peak')
80              self.setXRange(0,self.timeLimit)
81
82              #dataLock.release()
83              self.app.processEvents()
84
85      def savePlotData(self):
86          sessionData().addTrial(self.datasetx,self.datasety,self.mainWindow.
    stimuliOffsetSpinBox.value())
87
88      def __exit__(self):
89          self.dataSource.stop()
90
91  class MyMplCanvas(FigureCanvas):
92      """Ultimately, this is a QWidget (as well as a FigureCanvasAgg, etc.).
    """
93      def __init__(self, parent=None, width=5, height=4, dpi=100):
94          fig = Figure(figsize=(width, height), dpi=dpi)
95          self.axes = fig.add_subplot(111)
96
97          self.compute_initial_figure()
98
99          FigureCanvas.__init__(self, fig)
100         self.setParent(parent)
101
102         FigureCanvas.setSizePolicy(self,
103                                    QtWidgets.QSizePolicy.Expanding,
104                                    QtWidgets.QSizePolicy.Expanding)
105         FigureCanvas.updateGeometry(self)
106
107     def compute_initial_figure(self):
108         pass
109
110 class MyStaticMplCanvas(MyMplCanvas):
111     """Simple canvas with a sine plot."""
112     def compute(self, x = [], y = [], stimuli = 0):
113         self.datasetx = x
114         self.datasety = y
115         self.axes.plot(self.datasetx, self.datasety)
116         if(stimuli):
117             self.axes.axvline(x=stimuli, color = "r")
```

./dados/src/plotObjects.py

```
1 import sys
2
3 from errorDialog import Ui_errorDialog
4
5 from PyQt5 import QtCore, QtGui, QtWidgets
6
7 returnVal = -1
8
9 def setReturnVal(value):
10     global returnVal
11     returnVal = value
12
13 def errorPopUp(errorMsg):
14     dialog = QtWidgets.QDialog()
15     dialog.ui = Ui_errorDialog()
16     dialog.ui.setupUi(dialog)
17     dialog.ui.textField.setText(errorMsg)
18     dialog.ui.buttonBox.accepted.connect(lambda: setReturnVal(0))
19     dialog.ui.buttonBox.rejected.connect(lambda: setReturnVal(1))
20     dialog.exec_()
21     dialog.show()
22     return returnVal
23
24 if __name__ == "__main__":
25     errorPopUp("test")
```

./dados/src/popUp.py

```
1 from pathlib import Path
2 import os
3 import sys
4
5 from sessionData import sessionData
6 from pdfReport import pdfReport
7
8 def saveReport(fileName = "report"):
9     fileSaved = False
10     n = 0
11     if(sys.platform.startswith('linux') ):
12         p = Path('/media')
13         if(p.stat().st_size == 0):
14             raise FileNotFoundError("Empty path")
15         for child in p.iterdir():
16             with open(str(child)+"/"+fileName+".csv", 'w', newline='') as
    csvfile:
17                 for data in sessionData().trialData:
18                     n = n+1
19                     csvfile.write("Trial {0} - stimuli at {1} seconds\n".
```

```
    format(n,data['stimuli']))
20                      for x,y in zip(data['datasetx'], data['datasety']):
21                          csvfile.write(str(x)+";"+str(y)+"\n")
22                  csvfile.close()
23                  print("File saved to "+str(child)+"/"+fileName+".csv")
24
25          for child in p.iterdir():
26              pdfReport(str(child)+"/"+fileName+".pdf",sessionData())
27              print("File saved to "+str(child)+"/"+fileName+".pdf")
28          fileSaved = True
29      elif(sys.platform.startswith('win') ):
30          p = Path.cwd()
31          with open(str(p)+"\\reports\\"+fileName+".csv", 'w', newline='') as
     csvfile:
32              for data in sessionData().trialData:
33                  n = n+1
34                  csvfile.write("Trial {0} - stimuli at {1} seconds\n".format
    (n,data['stimuli']))
35                  for x,y in zip(data['datasetx'], data['datasety']):
36                      csvfile.write(str(x)+";"+str(y)+"\n")
37              csvfile.close()
38          pdfReport(str(p)+"\\reports\\"+fileName+".pdf",sessionData())
39          fileSaved = True
40
41      if(fileSaved == False):
42          raise NameError("Unable to save files")
43
44      return fileSaved
```

./dados/src/reportManager.py

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Wed Aug 15 19:39:21 2018
4
5 @author: Feupos
6 """
7 # This class is a singleton to be manipulated during a sessionData
8
9 # This gets the Qt stuff
10 import PyQt5
11 from PyQt5.QtWidgets import *
12
13 # This is our window from QtCreator
14 import examinationManager
15 from singleton import singleton
16
17 @singleton
```

```
18 class sessionData(object):
19     def __init__(self):
20         self.trialData = []
21         pass
22 
23     def updateData(self, form):
24         print("Storing current session data")
25         self.trialData = []
26         self.textData = []
27         self.comment = ""
28         self.textData.append({"label":form.patientLabel.text(),"data":form.
       patientLineEdit.text()})
29         self.textData.append({"label":form.dateOfBirthLabel.text(),"data":
       form.dateOfBirthDateEdit.text()})
30         self.textData.append({"label":form.idNumberLabel.text(),"data":form
       .idNumberLineEdit.text()})
31         self.textData.append({"label":form.insurancePlanLabel.text(),"data"
       :form.insurancePlanLineEdit.text()})
32         self.textData.append({"label":form.requestingDoctorLabel.text(),"
       data":form.requestingDoctorLineEdit.text()})
33         self.textData.append({"label":form.attendingDoctorLabel.text(),"
       data":form.attendingDoctorLineEdit.text()})
34         self.textData.append({"label":form.examinationTypeLabel.text(),"
       data":form.examinationTypeComboBox.currentText()})
35         self.textData.append({"label":form.requestingDoctorLabel.text(),"
       data":form.requestingDoctorLineEdit.text()})
36 
37 
38     def addTrial(self, x = [], y = [], s = 0):
39         print("Storing current trial data")
40         #self.trialData.append((datasetx,datasety, stimuli))
41         self.trialData.append({"datasetx":x, "datasety":y, "stimuli":s})
42 
43     def addComment(self, s):
44         self.comment = s
```

./dados/src/sessionData.py

B.2 CÓDIGO FONTE DA INTERFACE GRÁFICA

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <ui version="4.0">
3  <class>errorDialog</class>
4  <widget class="QDialog" name="errorDialog">
5   <property name="geometry">
6    <rect>
7     <x>0</x>
8     <y>0</y>
```

```
 9      <width>240</width>
10      <height>100</height>
11     </rect>
12    </property>
13    <property name="sizePolicy">
14     <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
15      <horstretch>0</horstretch>
16      <verstretch>0</verstretch>
17     </sizepolicy>
18    </property>
19    <property name="windowTitle">
20     <string>Error</string>
21    </property>
22    <property name="windowIcon">
23     <iconset>
24      <normaloff>../resources/error.png</normaloff>../resources/error.png</
       iconset>
25    </property>
26    <widget class="QLabel" name="textField">
27     <property name="geometry">
28      <rect>
29       <x>11</x>
30       <y>11</y>
31       <width>221</width>
32       <height>41</height>
33      </rect>
34     </property>
35     <property name="text">
36      <string>ERROR!
37 lol</string>
38     </property>
39     <property name="scaledContents">
40      <bool>true</bool>
41     </property>
42     <property name="alignment">
43      <set>Qt::AlignCenter</set>
44     </property>
45     <property name="wordWrap">
46      <bool>true</bool>
47     </property>
48     <property name="indent">
49      <number>-1</number>
50     </property>
51    </widget>
52    <widget class="QDialogButtonBox" name="buttonBox">
53     <property name="enabled">
54      <bool>true</bool>
```

```
55    </property>
56    <property name="geometry">
57     <rect>
58      <x>25</x>
59      <y>60</y>
60      <width>193</width>
61      <height>28</height>
62     </rect>
63    </property>
64    <property name="sizePolicy">
65     <sizepolicy hsizetype="Fixed" vsizetype="Fixed">
66      <horstretch>0</horstretch>
67      <verstretch>0</verstretch>
68     </sizepolicy>
69    </property>
70    <property name="orientation">
71     <enum>Qt::Horizontal</enum>
72    </property>
73    <property name="standardButtons">
74     <set>QDialogButtonBox::Close|QDialogButtonBox::Retry</set>
75    </property>
76    <property name="centerButtons">
77     <bool>true</bool>
78    </property>
79   </widget>
80  </widget>
81  <resources/>
82  <connections>
83   <connection>
84    <sender>buttonBox</sender>
85    <signal>accepted()</signal>
86    <receiver>errorDialog</receiver>
87    <slot>accept()</slot>
88    <hints>
89     <hint type="sourcelabel">
90      <x>248</x>
91      <y>254</y>
92     </hint>
93     <hint type="destinationlabel">
94      <x>157</x>
95      <y>274</y>
96     </hint>
97    </hints>
98   </connection>
99   <connection>
100   <sender>buttonBox</sender>
101   <signal>rejected()</signal>
```

```
102     <receiver>errorDialog</receiver>
103     <slot>reject()</slot>
104     <hints>
105      <hint type="sourcelabel">
106       <x>316</x>
107       <y>260</y>
108      </hint>
109      <hint type="destinationlabel">
110       <x>286</x>
111       <y>274</y>
112      </hint>
113     </hints>
114    </connection>
115   </connections>
116 </ui>
```

./dados/ui/errordialog.ui

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <ui version="4.0">
3  <class>MainWindow</class>
4  <widget class="QMainWindow" name="MainWindow">
5   <property name="windowModality">
6    <enum>Qt::NonModal</enum>
7   </property>
8   <property name="geometry">
9    <rect>
10     <x>0</x>
11     <y>0</y>
12     <width>640</width>
13     <height>480</height>
14    </rect>
15   </property>
16   <property name="sizePolicy">
17    <sizepolicy hsizetype="Expanding" vsizetype="Expanding">
18     <horstretch>0</horstretch>
19     <verstretch>0</verstretch>
20    </sizepolicy>
21   </property>
22   <property name="windowTitle">
23    <string>Examination Manager</string>
24   </property>
25   <property name="autoFillBackground">
26    <bool>true</bool>
27   </property>
28   <property name="locale">
29    <locale language="English" country="UnitedStates"/>
30   </property>
```

```
31  <widget class="QWidget" name="centralwidget">
32   <property name="enabled">
33    <bool>true</bool>
34   </property>
35   <layout class="QHBoxLayout" name="horizontalLayout">
36    <item>
37     <widget class="QStackedWidget" name="mainWidget">
38      <property name="frameShape">
39       <enum>QFrame::NoFrame</enum>
40      </property>
41      <property name="currentIndex">
42       <number>0</number>
43      </property>
44      <widget class="QWidget" name="mainPage">
45       <layout class="QFormLayout" name="patientData">
46        <item row="0" column="0">
47         <widget class="QLabel" name="patientLabel">
48          <property name="text">
49           <string>Patient</string>
50          </property>
51         </widget>
52        </item>
53        <item row="0" column="1">
54         <widget class="QLineEdit" name="patientLineEdit"/>
55        </item>
56        <item row="1" column="0">
57         <widget class="QLabel" name="dateOfBirthLabel">
58          <property name="text">
59           <string>Date of Birth</string>
60          </property>
61         </widget>
62        </item>
63        <item row="1" column="1">
64         <widget class="QDateEdit" name="dateOfBirthDateEdit"/>
65        </item>
66        <item row="2" column="0">
67         <widget class="QLabel" name="idNumberLabel">
68          <property name="text">
69           <string>ID Number</string>
70          </property>
71         </widget>
72        </item>
73        <item row="2" column="1">
74         <widget class="QLineEdit" name="idNumberLineEdit"/>
75        </item>
76        <item row="3" column="0">
77         <widget class="QLabel" name="insurancePlanLabel">
```

```
78          <property name="text">
79           <string>Insurance Plan</string>
80          </property>
81         </widget>
82        </item>
83        <item row="3" column="1">
84         <widget class="QLineEdit" name="insurancePlanLineEdit"/>
85        </item>
86        <item row="4" column="0">
87         <widget class="QLabel" name="requestingDoctorLabel">
88          <property name="text">
89           <string>Requesting Doctor</string>
90          </property>
91         </widget>
92        </item>
93        <item row="4" column="1">
94         <widget class="QLineEdit" name="requestingDoctorLineEdit"/>
95        </item>
96        <item row="5" column="0">
97         <widget class="QLabel" name="attendingDoctorLabel">
98          <property name="text">
99           <string>Attending Doctor</string>
100         </property>
101        </widget>
102       </item>
103       <item row="5" column="1">
104        <widget class="QLineEdit" name="attendingDoctorLineEdit"/>
105       </item>
106       <item row="6" column="0">
107        <widget class="QLabel" name="examinationTypeLabel">
108         <property name="text">
109          <string>Examination Type</string>
110         </property>
111        </widget>
112       </item>
113       <item row="6" column="1">
114        <widget class="QComboBox" name="examinationTypeComboBox">
115         <property name="editable">
116          <bool>false</bool>
117         </property>
118        </widget>
119       </item>
120       <item row="7" column="0">
121        <widget class="QLabel" name="numberOfTrialsLabel">
122         <property name="text">
123          <string>Number of Trials</string>
124         </property>
```

```
125          </widget>
126         </item>
127         <item row="7" column="1">
128          <widget class="QSpinBox" name="numberOfTrialsSpinBox">
129           <property name="minimum">
130            <number>1</number>
131           </property>
132           <property name="maximum">
133            <number>5</number>
134           </property>
135           <property name="value">
136            <number>1</number>
137           </property>
138          </widget>
139         </item>
140         <item row="8" column="1">
141          <widget class="QPushButton" name="startButton">
142           <property name="styleSheet">
143            <string notr="true"/>
144           </property>
145           <property name="text">
146            <string>Start</string>
147           </property>
148          </widget>
149         </item>
150        </layout>
151       </widget>
152       <widget class="QWidget" name="graphPage">
153        <layout class="QHBoxLayout" name="horizontalLayout_2">
154         <item>
155          <widget class="QWidget" name="verticalWidget" native="true">
156           <property name="sizePolicy">
157            <sizepolicy hsizetype="Expanding" vsizetype="Expanding">
158             <horstretch>0</horstretch>
159             <verstretch>0</verstretch>
160            </sizepolicy>
161           </property>
162           <layout class="QVBoxLayout" name="graphLayout"/>
163          </widget>
164         </item>
165         <item>
166          <widget class="QGroupBox" name="verticalGroupBox_2">
167           <property name="enabled">
168            <bool>true</bool>
169           </property>
170           <property name="sizePolicy">
171            <sizepolicy hsizetype="Fixed" vsizetype="Preferred">
```

```
172             <horstretch>0</horstretch>
173             <verstretch>0</verstretch>
174            </sizepolicy>
175           </property>
176           <property name="title">
177            <string>Controls</string>
178           </property>
179           <layout class="QVBoxLayout" name="controlsLayout">
180            <property name="sizeConstraint">
181             <enum>QLayout::SetDefaultConstraint</enum>
182            </property>
183            <item>
184             <widget class="QGroupBox" name="horizontalGroupBox">
185              <property name="sizePolicy">
186               <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
187                <horstretch>0</horstretch>
188                <verstretch>0</verstretch>
189               </sizepolicy>
190              </property>
191              <property name="title">
192               <string>Trial</string>
193              </property>
194              <layout class="QHBoxLayout" name="horizontalLayout_3">
195               <item>
196                <widget class="QLCDNumber" name="trialCounter">
197                 <property name="smallDecimalPoint">
198                  <bool>false</bool>
199                 </property>
200                 <property name="digitCount">
201                  <number>1</number>
202                 </property>
203                 <property name="segmentStyle">
204                  <enum>QLCDNumber::Flat</enum>
205                 </property>
206                </widget>
207               </item>
208              </layout>
209             </widget>
210            </item>
211            <item>
212             <widget class="QGroupBox" name="verticalGroupBox">
213              <property name="sizePolicy">
214               <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
215                <horstretch>0</horstretch>
216                <verstretch>0</verstretch>
217               </sizepolicy>
218              </property>
```

```
219              <property name="title">
220               <string>Trial Length</string>
221              </property>
222              <property name="flat">
223               <bool>false</bool>
224              </property>
225              <property name="checkable">
226               <bool>false</bool>
227              </property>
228              <layout class="QVBoxLayout" name="verticalLayout">
229               <property name="sizeConstraint">
230                <enum>QLayout::SetDefaultConstraint</enum>
231               </property>
232               <item>
233                <widget class="QDoubleSpinBox" name="trialLengthSpinBox">
234                 <property name="prefix">
235                  <string/>
236                 </property>
237                 <property name="suffix">
238                  <string> s</string>
239                 </property>
240                 <property name="decimals">
241                  <number>1</number>
242                 </property>
243                 <property name="singleStep">
244                  <double>0.100000000000000</double>
245                 </property>
246                 <property name="value">
247                  <double>5.000000000000000</double>
248                 </property>
249                </widget>
250               </item>
251              </layout>
252             </widget>
253            </item>
254            <item>
255             <widget class="QGroupBox" name="groupBox">
256              <property name="sizePolicy">
257               <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
258                <horstretch>0</horstretch>
259                <verstretch>0</verstretch>
260               </sizepolicy>
261              </property>
262              <property name="title">
263               <string>Stimuli offset</string>
264              </property>
265              <layout class="QVBoxLayout" name="verticalLayout_2">
```

```
266              <item>
267               <widget class="QDoubleSpinBox" name="stimuliOffsetSpinBox">
268                <property name="singleStep">
269                 <double>0.100000000000000</double>
270                </property>
271                <property name="value">
272                 <double>1.000000000000000</double>
273                </property>
274               </widget>
275              </item>
276             </layout>
277            </widget>
278           </item>
279           <item>
280            <widget class="QPushButton" name="runTrialButton">
281             <property name="text">
282              <string>Run Trial</string>
283             </property>
284            </widget>
285           </item>
286           <item>
287            <widget class="QPushButton" name="nextTrialButton">
288             <property name="enabled">
289              <bool>false</bool>
290             </property>
291             <property name="text">
292              <string>Next</string>
293             </property>
294            </widget>
295           </item>
296           <item>
297            <widget class="QPushButton" name="finishButton">
298             <property name="enabled">
299              <bool>false</bool>
300             </property>
301             <property name="text">
302              <string>Finish</string>
303             </property>
304            </widget>
305           </item>
306          </layout>
307         </widget>
308        </item>
309       </layout>
310      </widget>
311      <widget class="QWidget" name="reportPage">
312       <layout class="QVBoxLayout" name="verticalLayout_4">
```

```
313         <item>
314          <widget class="QScrollArea" name="scrollArea">
315           <property name="widgetResizable">
316            <bool>true</bool>
317           </property>
318           <widget class="QWidget" name="scrollAreaWidgetContents">
319            <property name="geometry">
320             <rect>
321              <x>0</x>
322              <y>0</y>
323              <width>98</width>
324              <height>22</height>
325             </rect>
326            </property>
327            <property name="sizePolicy">
328             <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
329              <horstretch>0</horstretch>
330              <verstretch>0</verstretch>
331             </sizepolicy>
332            </property>
333            <layout class="QVBoxLayout" name="graphReportLayout"/>
334           </widget>
335          </widget>
336         </item>
337         <item>
338          <widget class="QGroupBox" name="groupBox_2">
339           <property name="sizePolicy">
340            <sizepolicy hsizetype="Preferred" vsizetype="Fixed">
341             <horstretch>0</horstretch>
342             <verstretch>0</verstretch>
343            </sizepolicy>
344           </property>
345           <property name="maximumSize">
346            <size>
347             <width>16777215</width>
348             <height>150</height>
349            </size>
350           </property>
351           <property name="title">
352            <string>Report Options</string>
353           </property>
354           <layout class="QFormLayout" name="formLayout">
355            <item row="1" column="0">
356             <widget class="QLabel" name="label_2">
357              <property name="text">
358               <string>Comments</string>
359              </property>
```

```
360             </widget>
361            </item>
362            <item row="1" column="1">
363             <widget class="QPlainTextEdit" name="commentTextEdit"/>
364            </item>
365            <item row="3" column="1">
366             <widget class="QPushButton" name="reportButton">
367              <property name="text">
368               <string>Generate report</string>
369              </property>
370             </widget>
371            </item>
372            <item row="2" column="1">
373             <widget class="QLineEdit" name="fileNameLineEdit"/>
374            </item>
375            <item row="2" column="0">
376             <widget class="QLabel" name="label">
377              <property name="text">
378               <string>File Name</string>
379              </property>
380             </widget>
381            </item>
382           </layout>
383          </widget>
384         </item>
385        </layout>
386       </widget>
387      </widget>
388     </item>
389    </layout>
390   </widget>
391   <widget class="QMenuBar" name="menubar">
392    <property name="geometry">
393     <rect>
394      <x>0</x>
395      <y>0</y>
396      <width>640</width>
397      <height>26</height>
398     </rect>
399    </property>
400    <widget class="QMenu" name="menuMenu">
401     <property name="title">
402      <string>Menu</string>
403     </property>
404     <addaction name="actionConfiguration"/>
405    </widget>
406    <widget class="QMenu" name="menuHelp">
```

```
407      <property name="title">
408       <string>Help</string>
409      </property>
410      <addaction name="actionAbout"/>
411     </widget>
412     <addaction name="menuMenu"/>
413     <addaction name="menuHelp"/>
414    </widget>
415    <action name="actionAbout">
416     <property name="text">
417      <string>About</string>
418     </property>
419    </action>
420    <action name="actionConfiguration">
421     <property name="text">
422      <string>Configuration</string>
423     </property>
424    </action>
425   </widget>
426   <resources/>
427   <connections/>
428  </ui>
```

./dados/ui/examinationManager.ui

B.3 CÓDIGO FONTE DAS TRADUÇÕES

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <!DOCTYPE TS>
3  <TS version="2.1" language="pt_BR">
4  <context>
5      <name>MainWindow</name>
6      <message>
7          <location filename="examinationManager.py" line="254"/>
8          <source>Examination Manager</source>
9          <translation>Gerenciador de Exames</translation>
10     </message>
11     <message>
12         <location filename="examinationManager.py" line="255"/>
13         <source>Patient</source>
14         <translation>Paciente</translation>
15     </message>
16     <message>
17         <location filename="examinationManager.py" line="256"/>
18         <source>Date of birth</source>
19         <translation>Data de nascimento</translation>
20     </message>
21     <message>
```

```
22          <location filename="examinationManager.py" line="257"/>
23          <source>ID number</source>
24          <translation>N mero de registro</translation>
25      </message>
26      <message>
27          <location filename="examinationManager.py" line="258"/>
28          <source>Insurance plan</source>
29          <translation>Plano de sa de</translation>
30      </message>
31      <message>
32          <location filename="examinationManager.py" line="259"/>
33          <source>Requesting doctor</source>
34          <translation>M dico requisitante</translation>
35      </message>
36      <message>
37          <location filename="examinationManager.py" line="260"/>
38          <source>Attending doctor</source>
39          <translation>M dico aplicador</translation>
40      </message>
41      <message>
42          <location filename="examinationManager.py" line="261"/>
43          <source>Examination type</source>
44          <translation>Tipo de exame</translation>
45      </message>
46      <message>
47          <location filename="examinationManager.py" line="262"/>
48          <source>Number of trials</source>
49          <translation>N mero de sess es</translation>
50      </message>
51      <message>
52          <location filename="examinationManager.py" line="263"/>
53          <source>Start</source>
54          <translation>Iniciar</translation>
55      </message>
56      <message>
57          <location filename="examinationManager.py" line="264"/>
58          <source>Controls</source>
59          <translation>Controles</translation>
60      </message>
61      <message>
62          <location filename="examinationManager.py" line="265"/>
63          <source>Trial</source>
64          <translation>Sess o</translation>
65      </message>
66      <message>
67          <location filename="examinationManager.py" line="266"/>
68          <source>Trial Length</source>
```

```
69          <translation>Dura  o</translation>
70      </message>
71      <message>
72          <location filename="examinationManager.py" line="267"/>
73          <source> s</source>
74          <translation> s</translation>
75      </message>
76      <message>
77          <location filename="examinationManager.py" line="268"/>
78          <source>Stimuli offset</source>
79          <translation>Est mulo</translation>
80      </message>
81      <message>
82          <location filename="examinationManager.py" line="269"/>
83          <source>Run Trial</source>
84          <translation>Iniciar</translation>
85      </message>
86      <message>
87          <location filename="examinationManager.py" line="270"/>
88          <source>Next</source>
89          <translation>Pr ximo</translation>
90      </message>
91      <message>
92          <location filename="examinationManager.py" line="271"/>
93          <source>Finish</source>
94          <translation>Finalizar</translation>
95      </message>
96      <message>
97          <location filename="examinationManager.py" line="272"/>
98          <source>Report Options</source>
99          <translation>Op  es de relat rio</translation>
100     </message>
101     <message>
102         <location filename="pdfReport.py" line="56"/>
103         <source>Comments</source>
104         <translation>Coment rios</translation>
105     </message>
106     <message>
107         <location filename="examinationManager.py" line="274"/>
108         <source>Generate report</source>
109         <translation>Gerar realt rio</translation>
110     </message>
111     <message>
112         <location filename="examinationManager.py" line="275"/>
113         <source>File Name</source>
114         <translation>Nome do arquivo</translation>
115     </message>
```

```
116     <message>
117         <location filename="examinationManager.py" line="276"/>
118         <source>Menu</source>
119         <translation>Menu</translation>
120     </message>
121     <message>
122         <location filename="examinationManager.py" line="278"/>
123         <source>Help</source>
124         <translation type="obsolete">Ajuda</translation>
125     </message>
126     <message>
127         <location filename="examinationManager.py" line="278"/>
128         <source>About</source>
129         <translation>Sobre</translation>
130     </message>
131     <message>
132         <location filename="examinationManager.py" line="277"/>
133         <source>Configuration</source>
134         <translation>Configura  o</translation>
135     </message>
136     <message>
137         <location filename="examinationManager.py" line="279"/>
138         <source>Shutdown</source>
139         <translation>Desligar</translation>
140     </message>
141     <message>
142         <location filename="examinationManager.py" line="280"/>
143         <source>Recalibrate</source>
144         <translation>Recalibrar</translation>
145     </message>
146 </context>
147 <context>
148     <name>errorDialog</name>
149     <message>
150         <location filename="main.py" line="142"/>
151         <source>Unable to save files.
152 </source>
153         <translation>N o foi poss vel salvar os arquivos.
154 </translation>
155     </message>
156     <message>
157         <location filename="main.py" line="142"/>
158         <source>Check if USB drive is connected!</source>
159         <translation>Confira se o pen drive est  conectado!</translation>
160     </message>
161     <message>
162         <location filename="errorDialog.py" line="50"/>
```

```
163         <source>Error</source>
164         <translation>Erro</translation>
165     </message>
166     <message>
167         <location filename="errorDialog.py" line="51"/>
168         <source>ERROR!
169 lol</source>
170         <translation>ERRO!
171 lol</translation>
172     </message>
173 </context>
174 </TS>
```

./dados/src/pt_BR.ts