



Universidade de Brasília – UnB
Faculdade UnB Gama – FGA
Projeto Integrador Engenharia 2

UMISS - Unidade Móvel de Identificação de Saúde e Socorro

Projeto UMISS
Orientadores: Alex Reis, Luiz Laranjeira, Rhander Viana e
Sebastièn Rondineau

Brasília, DF

2017



Afonso Delgado, Cesar Marques, Dylan Guedes, Felipe Assis, Gustavo Cavalcante, Johnson Andrade, Lucas Castro, Lunara Martins, Mariana Andrade, Nivaldo Lopo, Rafael Amado, Tiago Assunção, Wilton Rodrigues

UMISS - Unidade Móvel de Identificação de Saúde e Socorro

Relatório técnico referente à disciplina de Projeto Integrador 2, reunindo os cursos de Engenharias presentes no Campus Gama, da Universidade de Brasília.

Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Orientador: Alex Reis, Luiz Laranjeira, Rhander Viana e Sébastien Rondineau

Brasília, DF

2017

UMISS - Unidade Móvel de Identificação de Saúde e Socorro

Relatório técnico referente à disciplina de Projeto Integrador 2, reunindo os cursos de Engenharias presentes no Campus Gama, da Universidade de Brasília.

Brasília, DF

2017

Resumo

Pacientes com capacidade motora reduzida, em certo grau, necessitam de observação contínua a fim de evitar acidentes ou outros problemas. Além disso, em alguns casos, a presença de um cuidador é necessária para ajudar na movimentação da cadeira de rodas e na captura de sinais vitais. Tecnologias nesse campo não evoluem rápido o suficiente, não resolvem estes cenários ao mesmo tempo, e, mais ainda, são custosas. Neste trabalho nós apresentamos a UMISS, uma cadeira elétrica que extrai sinais vitais, notifica eventos críticos, e se move sem intervenção de terceiros. Com a UMISS nós esperamos criar uma solução de baixo custo, que permita ao paciente cuidar de si mesmo de maneira segura.

Palavras-chaves: cadeira de rodas. acessível. monitoramento. sensores.

Abstract

Handicapped people, in a certain degree, needs continuous monitoring in order to prevent accidents or other issues. Besides that, in some cases, the presence of a carer is needed to help with the wheelchair, and to track vital signals. Technologies in this field are not evolving fast enough, does not solve these scenarios at the same time, and, even more, are costly. In this work we present UMISS, an electric wheelchair that tracks vital signals, notifies critical events, and moves without third party intervention. With UMISS we expect to create a low cost solution, that allows the patient to securely take care of himself.

Key-words: wheelchair. accessible. monitoring. sensors

Listas de ilustrações

Figura 1 – Sensor para medição de temperatura corporal.	12
Figura 2 – Eletrodos fabricados para medição de umidade e resistência galvânica da pele.	12
Figura 3 – Circuito para captura de umidade da pele por GSR.	12
Figura 4 – Sensor de presença do usuário na cadeira de rodas.	13
Figura 5 – Circuito condicionador e de aquisição de sinais de eletrocardiograma.	14
Figura 6 – Diagrama de classes do Shoelace. Métodos marcados com ** são abstratos.	16
Figura 7 – <i>Design</i> e Arquitetura do servidor <i>Django</i> (TECHNOLOGY, 2013)	18
Figura 8 – Diagrama da arquitetura parcial do UMISS-frontend.	20
Figura 9 – Layout atual da tela principal do módulo Android.	21
Figura 10 – Diagrama de classes do módulo Android.	22
Figura 11 – Foto ilustrativa do motor utilizado no projeto.	23
Figura 12 – Dados da bateria usada no projeto.	25
Figura 13 – Dados da bateria usada no projeto.	25
Figura 14 – Circuito de entrada do transformador.	27
Figura 15 – Circuito de saída do transformador.	27
Figura 16 – Circuito de Chaveamento.	28
Figura 17 – Circuito de chaveamento (à direita) e circuito dobrador de tensão (à esquerda).	29
Figura 18 – Ponte H.	30
Figura 19 – Placa de teste da ponte H.	31
Figura 20 – Circuito do dobrador de tensão.	32
Figura 21 – Exemplo de função usado no controle PWM.	33
Figura 22 – Cadeira de Rodas Jaguaribe 1009.	35
Figura 23 – Desenho em ambiente CAD da Cadeira de Rodas.	36
Figura 24 – Análise Estática da Cadeira de Roda.	37
Figura 25 – Desenho técnico disco de atrito.	39
Figura 26 – Desenho Técnico da placa de suporte do motor e do disco de atrito	40
Figura 27 – Desenho do estado atual da Cadeira de Roda.	40
Figura 28 – Projeto da Cadeira Completa.	43
Figura 29 – <i>Design</i> e Arquitetura do servidor Django	46

Lista de tabelas

Tabela 1 – Valores da Análise Modal.	37
Tabela 2 – Cronograma para a integração dos subsistemas.	43

Lista de abreviaturas e siglas

PSM	Processamento de Sinais e Monitoramento
CeA	Controle e Alimentação
PE	Projeto Estrutural
PC1	Ponto de controle 1
UMISS	Unidade Móvel de Identificação de Saúde e Socorro

Sumário

1	INTRODUÇÃO	10
2	PROCESSAMENTO DE SINAIS E MONITORAMENTO	11
2.1	Aquisição e Condicionamento de Sinais	11
2.2	Middleware	15
2.2.1	Arquitetura	15
2.3	Backend Django	16
2.3.1	Papel do <i>Backend</i>	16
2.3.2	Design e Arquitetura da Solução	17
2.3.3	Projeto Desenvolvido	18
2.4	Módulo JavaScript EmberJS	19
2.5	Módulo Android	19
2.5.1	Funcionalidades	20
2.5.2	Arquitetura	21
3	CONTROLE E ALIMENTAÇÃO	23
3.1	Dimensionamento de Motores e Sistema Energético	23
3.2	Dimensionamento da bateria	24
3.3	Carregador para bateria	26
3.4	Controle PWM e Ponte H	28
3.4.1	Círculo de Chaveamento	28
3.4.2	Ponte H	30
3.4.3	Dobrador de tensão	31
3.4.4	Controlador PWM - Software	31
4	ESTRUTURAS	34
4.1	Estrutura da Cadeira de Rodas	34
4.2	Análises assistidas por Computador (CAD/CAE)	35
4.3	Processo de Fabricação das Peças	38
5	PLANO DE INTEGRAÇÃO	41
5.1	Integração do subsistema Processamento de Sinais e Monitoramento	41
5.2	Integração do subsistema de Estruturas	42
5.3	Cronograma	42
	REFERÊNCIAS	44

ANEXOS

45

ANEXO A – DIAGRAMA DE CLASSES BACKEND 46

1 Introdução

Neste relatório apresentamos o andamento e a progressão do projeto UMISS que ocorreu entre os pontos de controle 1 e 2. Focaremos em questões mais práticas, e levantaremos os resultados obtidos e os esperados.

O objetivo desta etapa é desenvolver os subsistemas separadamente, levando em consideração a segurança e redundância de cada um, bem como a sua qualidade final. Estes são desenvolvidos em ambientes diferentes, porém, com o foco na integração de cada um, que será a próxima fase deste desenvolvimento. Desta maneira, este relatório trata sobre os resultados obtidos ao longo das semanas de implementação do ponto de controle 2.

A organização se dará da seguinte forma: cada capítulo que segue será relativo a um subsistema do projeto. No Capítulo 2 apresentamos o andamento do subsistema de Processamento de Sinais e Monitoramento, que contempla o *middleware*, a aquisição de sinais, o servidor remoto (*backend*), o cliente *web* (*frontend*) e o aplicativo Android. Por fim, no Capítulo 5 traremos o nosso planejamento para a integração final dos diferentes subsistemas, e as considerações finais sobre a segunda parte do projeto.

2 Processamento de Sinais e Monitoramento

2.1 Aquisição e Condicionamento de Sinais

O projeto eletrônico do sistema de processamentos de sinais e monitoramento consiste em quatro módulos de sistemas de captura e condicionamento para os seguintes sinais: temperatura corporal e do ambiente, resistência galvânica da pele (GSR), acontecimento de quedas do usuário da cadeira de rodas e eletrocardiograma.

Para o circuito de monitoramento de sinais de temperatura, foi utilizado um sensor de medição linear de temperatura LM35, fabricado pela *Texas Instruments*. O sensor foi selecionado por sua variação linear de tensão de saída com a variação de temperatura, além do seu baixo consumo e alta precisão de medidas de temperatura com rápida resposta e sua faixa de captura de -55 à 150 graus Celsius, tornando-o viável para monitoramento de temperaturas corporais e ambientais. Tanto o circuito quanto o sistema embarcado foram projetados e dimensionados para tratar de sua variação linear de 10 mV em sua saída a cada 1 grau de variação na temperatura¹.

Foram também realizados testes e algoritmos para medição da temperatura corporal com termistores NTC, porém, os mesmos mostraram-se menos acurados devido ao seu comportamento não linear dado pela equação de *Equação de Steinhart-Hart*² e problemas de arredondamento de valores capturados pelo sistema embarcado. Dessa forma, foi decidido o uso do sensor LM35.

O sensor fabricado para medição da temperatura corporal, apresentado na Figura 1, é aplicado à superfície da pele, no antebraço, possibilitando leitura facilitada e de forma menos invasiva. Uma vez em contato com o corpo do usuário, o sensor apresenta alteração nos valores lineares de temperatura até o momento de estabilidade com a temperatura corporal.

O sistema de monitoramento de sinais de resistência galvânica da pele (GSR), tem como principal função o monitoramento de variações na umidade da pele do usuário, podendo indicar níveis perigosos de estresse ou até mesmo colapsos de hipoglicemias ou por fraqueza por falta de alimentação. O sistema consiste em dois eletrodos fabricados, como visto na Figura 2, para contato com o antebraço do usuário, com uma distância fixa de 5 cm entre eles, dessa forma, mantendo os valores das medições confiáveis para cada usuário com diferentes tipos de pele.

Além disso, o circuito, visto na Figura 3 embarcado conta com um capacitor de

¹ <<http://www.ti.com/lit/ds/symlink/lm35.pdf>>

² <<http://www.sciencedirect.com/science/article/pii/0011747168900570?via%3Dihub>>

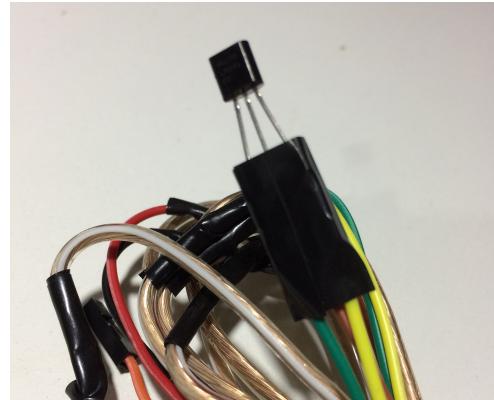


Figura 1 – Sensor para medição de temperatura corporal.

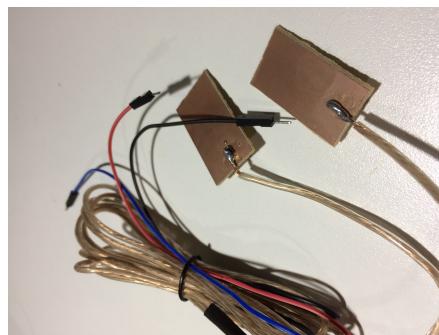


Figura 2 – Eletrodos fabricados para medição de umidade e resistência galvânica da pele.

acoplamento para o contato com a pele humana e um resistor de alta impedância para a medição da variação da resistência da pele por divisor de tensão. O sistema foi simplificado para que pudesse ser implementado junto aos cabos dos eletrodos, de forma a utilizar o mínimo de espaço possível na cadeira de rodas.



Figura 3 – Circuito para captura de umidade da pele por GSR.

O algoritmo para o monitoramento da resistência galvânica e umidade conta com comparadores para diferentes status do usuário, podendo alertar quando o usuário não está realizando a medição, quando apresenta características normais para sua pele ou quando

o usuário encontra-se em situação de risco. O sistema embarcado para o monitoramento e integração com o *middleware* foi programado em linguagem Python.

O sistema de alerta de quedas é responsável por enviar alertas para o servidor caso o usuário encontre-se fora da cadeira, indicando que o mesmo caiu da cadeira e necessita de auxílio urgente. Para esse sistema, apresentado na Figura 4 foi desenvolvido um circuito com um sistema de medição de distância e presença utilizando um módulo com sensor piezoelétrico. O sistema é responsável pelo monitoramento de variações de tensão no sensor, possibilitando o processamento dessa variação, alertando a presença ou não do usuário na cadeira de rodas. O sensor piezoelétrico é inserido no assento da cadeira para que o mesmo esteja pressionado durante todo o momento que o usuário esteja sentado na cadeira, e dessa forma, o algoritmo é capaz de alertar caso a presença do usuário na cadeira não seja identificada.

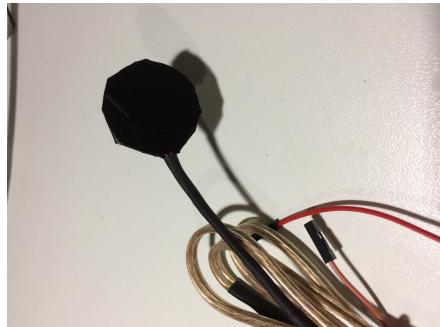


Figura 4 – Sensor de presença do usuário na cadeira de rodas.

O sistema de monitoramento dos sinais de eletrocardiografia, apresentado na ?? foi desenvolvido para captura dos sinais de pulso cardíaco a partir de eletrodos conectados ao antebraço do usuário, para que as medidas e a captura do sinal seja feita da forma menos invasiva possível. Para isso, o circuito deve ser capaz de capturar o sinal de ECG com precisão e assim, foi decidido pelo uso de um amplificador de instrumentação INA 128p, fabricado pela *Texas Instruments*³, principalmente por suas características de alta rejeição de CMRR, além de ganho de fácil regulagem. Além do amplificador de instrumentação, foram projetados filtros passa-baixa para rejeição de frequências acima de 100Hz que possam interferir no sinal de eletrocardiograma, que para monitoramento, variam entre baixas frequências de 0.5Hz e 40Hz⁴.

O projeto de captura dos sinais de eletrocardiograma utilizando o amplificador INA128 é dado a partir do ganho necessário para a visualização e processamento do sinal, com isso, o ganho do amplificador é dado pela equação:

³ <<http://www.ti.com/lit/ds/symlink/ina129.pdf>>

⁴ Yazicioglu RF, van Hoof C, Puers R. Biopotential Readout Circuits for Portable Acquisition Systems, 2009, Springer Science, ISBN: 978-1-4020-9092-9)

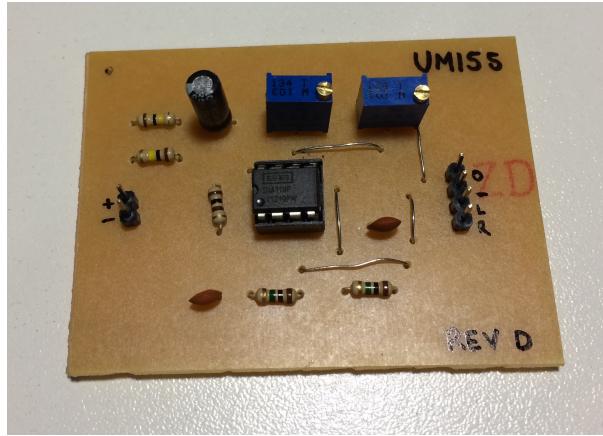


Figura 5 – Circuito condicionador e de aquisição de sinais de eletrocardiograma.

$$Av = 1 + \frac{50000}{R_G} \quad (2.1)$$

Sabe-se que a amplitude de sinais de ECG é baixa, na faixa de 0.1 mV a 5 mV . Com isso, foi implementado um resistor $R_G = 100\omega$, resultando em um ganho $Av = 501$. Dessa forma, o sinal terá uma amplitude de cerca de 2.5 V .

Após testes, os primeiros sinais obtidos foram satisfatórios para monitoramento de frequência cardíaca. Dessa forma, após o condicionamento, os sinais são enviados para um microcontrolador Arduino, que realiza a amostragem e envia os dados de forma Serial para o processador principal presente na Raspberry Pi, onde os dados são capturados pela porta USB, e dispostos em um *array*, alterado a cada período de tempo. Para teste de visualização destes dados, os mesmos foram plotados no sistema da Raspberry Pi utilizando a biblioteca *matplotlib*⁵.

Após a captura e condicionamento de todos os sinais analógicos, os mesmos são direcionados para um módulo de conversão Analógico-Digital (A/D). O conversor selecionado foi o ADS1115⁶, devido a sua alta resolução de 16 bits, capaz de converter valores analógicos em valores digitais de 0 a 65535, e por sua taxa de amostragem de 800Hz, suficientes para a captura dos sinais a serem monitorados para o projeto. O conversor é conectado com a Raspberry Pi via protocolo I2C, possibilitando o uso de menos portas para o envio de até 4 sinais simultâneos.

Uma vez com os sinais devidamente convertidos para digital, o sistema embarcado e de *middleware* é responsável por monitorar os sinais, realizando verificações de seus valores a partir de calibrações realizadas para cada um dos módulos.

⁵ <<https://matplotlib.org/>>

⁶ <<https://cdn-shop.adafruit.com/datasheets/ads1115.pdf>>

2.2 Middleware

O *middleware* do subsistema, uma Raspberry, tinha como resultados esperados uma aplicação que pudesse, ao rodar no embarcado, receber sinais e enviá-los de maneira correta ao servidor.

Os resultados esperados foram atingidos. Foi desenvolvido a aplicação Shoelace⁷, que serve como abstração para a aquisição dos dados do conversor A/D e que envia os resultados para um servidor remoto.

Definimos que uma regra de negócio deveria ser que toda Raspberry tivesse um *token* e uma senha incluída, e esses dados são então utilizados nas requisições para os servidores. Isso foi feito através da geração de dados aleatórios (*token* e senha), que são obtidos sempre que a Raspberry é ligada, por estarem no *bash_rc*. Assim, todas as adições de sinais feitas por uma dada Raspberry já serão relacionados com o respectivo paciente, que poderá ter seus dados visualizados por parentes cadastrados.

A comunicação entre a Raspberry e o conversor é feita através do pacote em Python **Adafruit_ADS**, capaz de ler de até quatro canais ao mesmo tempo. Contudo, uma ressalva: os valores enviados pelo sensor de temperatura não são recebidos de uma maneira apresentável, por não estarem normalizados. Utilizamos então a equação de Steinhart-hart:

$$1/t = A + B * \ln(R) + C[\ln(R)]^3$$

Onde A, B e C são coeficientes, R é a resistência, e T a temperatura que desejamos apresentar. Utilizamos os seguintes valores para os coeficientes:

$$A = 0.001129148; B = 0.000234125; C = 0.0000000876741$$

Para o cálculo do \ln utilizamos a função *log1p* do Python, e ressaltamos que tivemos diversos problemas de precisão, pois o Python arredonda os resultados das operações de maneira grosseira em diversas situações. Por fim, ajustamos outros parâmetros (como os utilizados na conversão da resistência) utilizando outros resultados como base, de maneira experimental.

2.2.1 Arquitetura

A arquitetura do Shoelace foi feita pensando-se principalmente na **extensão**. Desenvolvemos então essa aplicação de modo que, caso novos sensores dos mais diversos tipos precisem ser utilizados, a adaptação no código da Raspberry é simples. Uma classe abstrata **Sensor** força a implementação do limiar referente ao sensor, e trás consigo funções que serão úteis, como o cálculo da diferença percentual entre o valor enviado para o servidor e o valor que está sendo analisado. A implementação de um novo sensor deve

⁷ <https://github.com/cadeiracuidadora/shoelace>

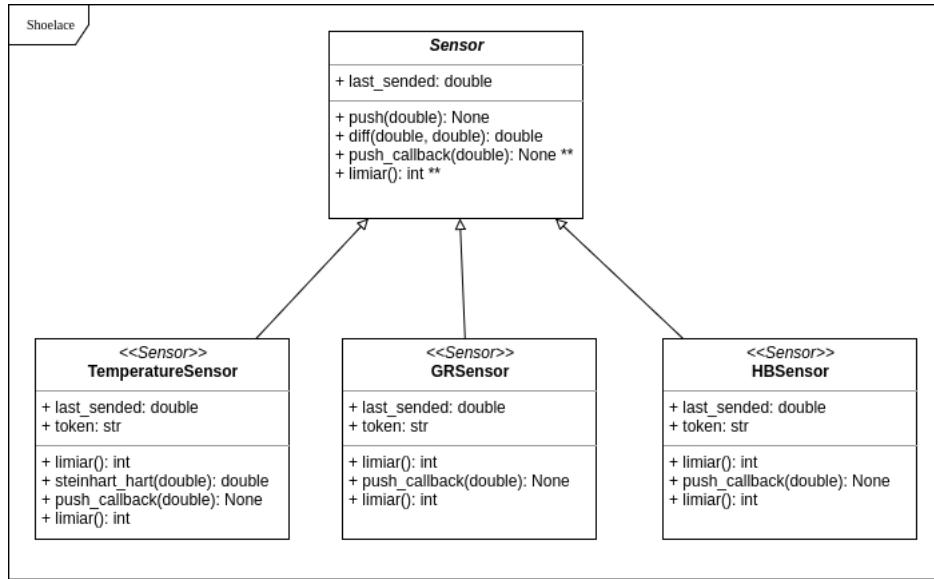


Figura 6 – Diagrama de classes do Shoelace. Métodos marcados com ** são abstratos.

então sobrescrever somente dois métodos: (i) o método *limiar*, que diz qual a diferença mínima entre o último valor enviado e um novo valor para que seja enviado (útil na diminuição de sobrecarga do servidor, dificultando cenários de *backpressure*); e (ii) o método *push_callback*, que dá para o sensor a liberdade de decidir o que fazer quando o limiar definido for atingido. É nesse *callback* que fazemos a requisição no servidor remoto.

A Figura 6 apresenta a arquitetura geral do Shoelace. É ilustrado a implementação dos três sensores que utilizamos, mas caso novos sensores precisem ser usados, basta criar uma classe que herde de **Sensor**, e implementar os métodos necessários (*limiar* e *push_callback*).

2.3 Backend Django

Esta sessão tratará sobre o servidor do sistema, também conhecido como *backend* ou *API Django*. Na primeira subsessão, é tratada uma breve descrição sobre as funções e características deste. A segunda apresenta o *design* e a arquitetura de desenvolvimento. Por fim, trataremos as instâncias do projeto desenvolvido.

2.3.1 Papel do Backend

O *backend* do sistema, responsável por fazer o controle das requisições, atuando como intermediador à cadeira e aos seus respectivos monitores foi desenvolvida utilizando o *Django Rest Framework*, que é uma referência no mundo de desenvolvimento de API's para *Python*.

O *backend* manipula todos os dados enviados pela cadeira, utilizando a *Raspberry* para executar o processamento desses dados e enviar notificações para os *smartphones* cadastrados no sistema. Além disso, serve dados para uma interface web de controle do monitor.

A manipulação dos dados é feita de forma segura, tratando a autenticação dos usuários que lidam com o sistema e redirecionando os dados de acordo com a sua autenticação. Para a execução dos passos, o servidor conta com um poderoso sistema de autenticação dos seus usuários, tanto para os pacientes que utilizam a cadeira, quanto para os monitores que manipulam as interfaces para cliente, como mobile e interface web. Os pacientes são cadastrados no sistema e enviam os sinais corporais para o servidor. O servidor, além de enviar uma notificação para o monitor respectivo, armazena este sinal em sua base de dados e retorna ao monitor quando solicitado. Os dados enviados ao monitor são apenas os sinais respectivos ao paciente que está sendo monitorado.

A ligação entre o paciente e o monitor pode ser feita no momento do cadastro ou em atualizações futuras. O paciente contém um identificador único em sua cadeira, que pode ser utilizado pelo monitor para supervisioná-lo. No momento que o monitor insere os dados no sistema, automaticamente será ligado ao paciente e terá acesso aos dados de sinais enviados pelo paciente.

2.3.2 Design e Arquitetura da Solução

O *django* utiliza de uma arquitetura própria para padronização e aplicação de boas práticas em desenvolvimento de *API's*. Esta arquitetura está dividida em *tiers*, que são responsáveis por funções específicas e cada uma tem seu papel definido para a manipulação de dados no sistema. A seguir está uma ilustração do funcionamento da arquitetura do servidor.

Como pode ser observado, existem três *Tiers* principais, que são utilizadas para a manipulação dos dados. Elas são:

- *Presentation Tier*;
- *Business Tier*;
- *Data Tier*.

A primeira, *Presentation Tier* é responsável por ser a interface com as demais aplicações que irão utilizar o servidor. Essa *tier* é responsável por gerir as rotas do sistema, direcionando as urls solicitadas a funções específicas. Após mapeado nas funções, estas irão verificar as permissões de cada requisição, bem como cada usuário que está solicitando uma determinada ação.

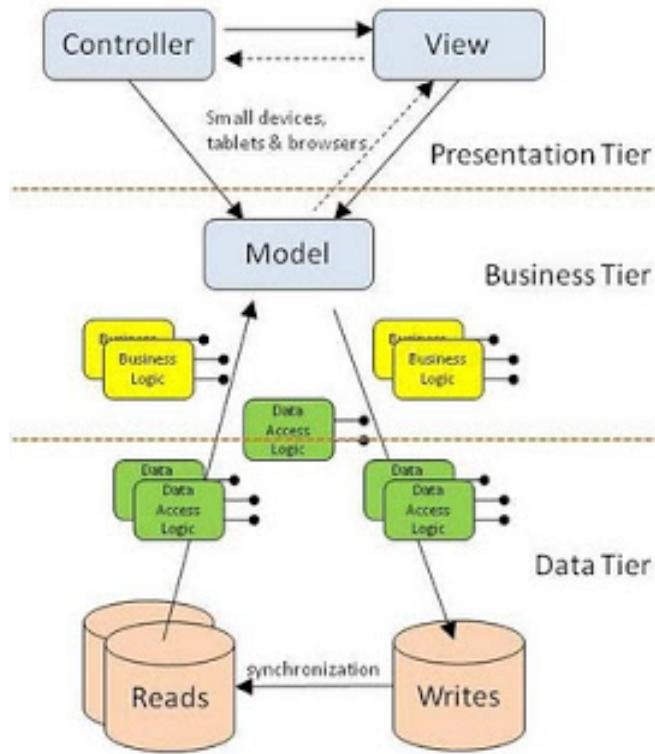


Figura 7 – Design e Arquitetura do servidor *Django* ([TECHNOLOGY, 2013](#))

Além disso, nesta *tier*, existe uma transformação importante para o consumo de dados por parte dos clientes. Esta é conhecida como serialização dos dados, tanto para quem envia, quanto para quem solicita. Sua responsabilidade é aplicar um modelo de dados formalizado na comunidade, conhecido como *json*. Dessa forma, todo tipo de dado recebido, assim como os enviados são transformados a partir dos objetos *Python*. Esta fase é aplicada logo após a aplicação das rotas para as respectivas ações.

As funções mapeadas na *Tier* passada utiliza de modelos predefinidos na Business *Tier*. Aqui são definidos os domínios da aplicação, bem como quais são as classes e negócio da aplicação. Nesta fase são definidos os tipos de usuários do sistema, as classes de sinais corporais do paciente e toda parte de negócios da aplicação.

A última *tier*, chamada de *Data Tier* é responsável por lidar com a persistência dos dados de todos os usuários e sinais que são manipulados no sistema. Esta *tier* modela o bando de dados assim como foi definido na aplicação de negócios e aplica inserções neste a medida que novas requisições são feitas.

2.3.3 Projeto Desenvolvido

O projeto UMISS-BackEnd, desenvolvido exatamente na arquitetura apresentada na sessão 2.3.2, possui entidades que deram formato à Business *Tier* do projeto, onde são

definidas as propriedades do sistema, bem como ele atuará.

No projeto, foi desenvolvido o diagrama de classes, que auxilia na atividade de compreender o domínio do projeto. Este está no Anexo A.

É possível visualizar que existem várias entidades de usuários. Isto foi desenvolvido pois existem dois tipos de usuários diferentes no sistema. Eles são o Paciente e o Monitor. Estes possuem atributos e comportamentos diferentes, gerando assim a necessidade de implementação de dois domínios diferentes. Da mesma maneira, existem três tipos de sinais corporais que são enviados pelo paciente. Esses sinais possuem características diferentes, bem como atributos. Dessa maneira, existe uma entidade generalista chamada *BodySignal* com características comuns de todas. E foram criados as demais especializações com suas características específicas. Elas são: Sinais de temperatura corporal, Sinais de batimento cardíaco e Sinais de resistência galvânica.

Por fim, existem ligações dos usuários com os sinais corporais, sendo que os pacientes são donos destes. Assim como existem pacientes que possuem monitores ligados, gerando um relacionamento cíclico entre as entidades de usuário.

2.4 Módulo JavaScript EmberJS

O *frontend* do subsistema, módulo responsável pela visualização dos dados, tinha como resultados esperados uma aplicação que pudesse prover a visualização dos sinais referentes a um determinado paciente, baseado nas informações recebidas do servidor de *backend*. Devido à necessidade de uma certa dinamicidade na apresentação desses dados, foi escolhido o *Framework* JavaScript EmberJS⁸. Desta forma, foi então desenvolvida a aplicação UMISS-frontend⁹, que apresenta de forma gráfica e textual o histórico dos sinais monitorados pelo projeto UMISS.

Baseado no *token* fornecido em cada cadeira, o usuário monitor pode fazer o cadastro na aplicação e assim fazer o monitoramento do paciente vinculado à cadeira em questão. O servidor de *backend* é responsável pela filtragem dos dados recebidos, garantindo então que apenas os dados do paciente vinculado ao token fornecido possam ser visualizados pelo usuário monitor.

2.5 Módulo Android

O módulo Android do sistema é responsável pela visualização dos dados referentes a um determinado paciente e também pela notificação do usuário monitor via *push notification*.

⁸ <<https://emberjs.com/>>

⁹ <<https://github.com/cadeiracuidadora/UMISS-frontend>>

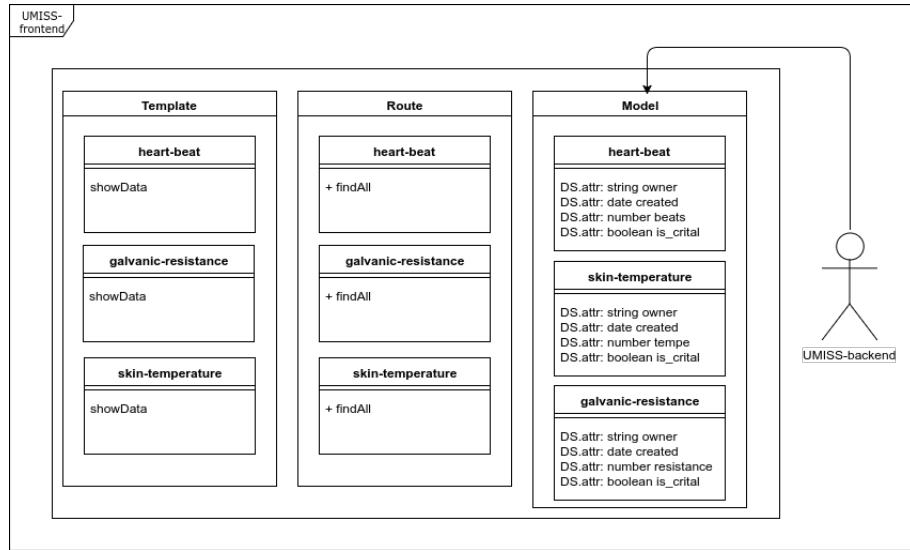


Figura 8 – Diagrama da arquitetura parcial do UMISS-frontend.

Este módulo atua como cliente, ou seja, os dados referentes a um paciente estão armazenados no *backend* e a aplicação Android somente consome esses dados. O único valor que é armazenado na aplicação Android é um *token* de autenticação enviado do servidor quando um usuário monitor efetua o *login*.

2.5.1 Funcionalidades

- *Login*:

É possível fazer login com um usuário do tipo monitor na aplicação, ao fazê-lo, é recebido um *token* de autenticação do servidor que é armazenado no Android. Essa etapa se difere um pouco do *frontend* pois no Android ocorre um passo adicional de enviar um identificador para o *backend* ser capaz de enviar as notificações para o Android.

- Cadastro:

Caso o usuário do tipo monitor ainda não tenha feito um cadastro no *frontend*, é possível fazer o cadastro direto da aplicação Android, basta informar os dados de *login*, senha e o identificador da cadeira.

- Notificações:

Caso o *backend* receba um dado crítico do *shoelace*, é enviado ao aplicativo Android uma notificação do tipo *push notification* para que o usuário seja alertado que há algo errado acontecendo com o paciente.

- Gráficos:

Os dados referentes à temperatura, batimentos cardíacos e resistência galvânica são exibidos em forma de gráfico, toda vez que o usuário abrir a aplicação uma nova requisição será feita ao servidor e os gráficos serão criados.

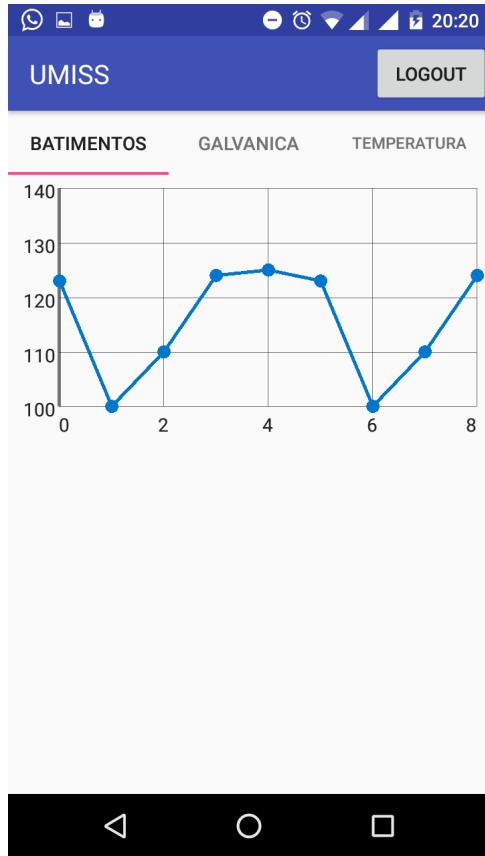


Figura 9 – Layout atual da tela principal do módulo Android.

2.5.2 Arquitetura

A arquitetura do módulo Android foi elaborada da seguinte maneira, foram criados dois pacotes, um chamado *com.umiss* que contém todas as *fragments* e *activities* da aplicação e um pacote chamado *network* que contém as classes responsáveis pelas requisições e notificações.

A *MainActivity* é a classe principal do sistema, ao iniciar, é executado o método *isLoggedIn* que verifica se existe um *token* de autenticação válido, caso exista, será instanciado as *fragments* *HeartBeatsFragment*, *GalvanicFragment* e *TemperatureFragment* que são responsáveis pela criação dos gráficos de batimentos cardíacos, temperatura e resistência galvânica respectivamente. Caso contrário, será instanciado a *activity* de *Login*, a partir da *activity* de *Login* é possível abrir a *activity* de *Cadastro*.

O pacote *network* contém a classe *UMISSRest* que é responsável pelos métodos http *POST*, *GET* e *PUT*. Todas as classes que realizam alguma requisição com o *ba-*

kend utilizam os métodos estáticos da classe *UMISSRest*. As notificações são feitas pelo serviço Firebase Cloud Messaging através das classes *MyFirebaseMessagingService* que é responsável por criar uma notificação e pela classe *MyFirebaseInstanceIdService* que cria um identificador que é enviado ao *backend*.

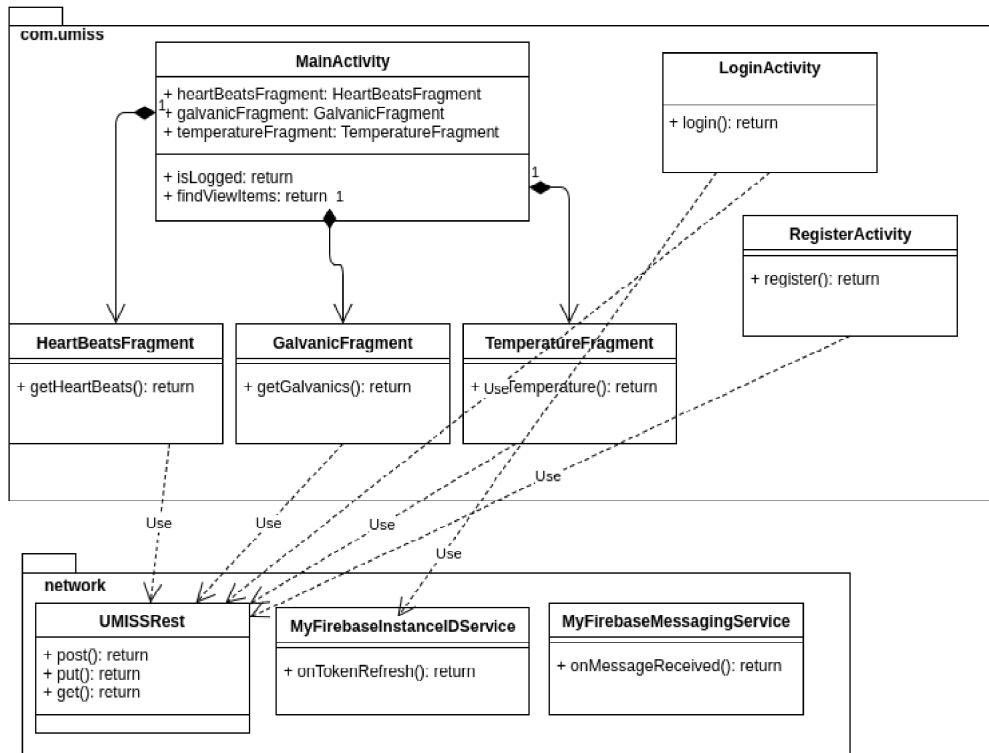


Figura 10 – Diagrama de classes do módulo Android.

3 Controle e Alimentação

3.1 Dimensionamento de Motores e Sistema Energético

Neste projeto estão sendo utilizados dois motores de corrente contínua¹. O dimensionamento do motor foi realizado de acordo com o equacionamento e os dados obtidos do fornecedor, também disponíveis no *datasheet*.



Figura 11 – Foto ilustrativa do motor utilizado no projeto.

O modelo do motor utilizado é o MR 210 VER 240, com tensão de 12V, fabricado pela Motron, e como a redução utilizada (sistema de transmissão) é de 6 para 1, a partir dos dados do motor é possível calcular a carga nominal do conjunto cadeira e usuário.

$$\frac{n_1}{n_2} = \frac{d_1}{d_2} = \frac{T_1}{T_2}$$

$$T_2 = 6 * T_1$$

Torque nominal de um motor = 24 Nm

Aplicando a relação de transmissão, o torque na roda movida é:

$$T_2 = 6 * T_1 = 6 * 24 = 144Nm$$

A força aplicada na roda movida, com raio de 0,3m, é dado por:

¹ O *datasheet* do motor utilizado pode ser encontrado em: <<http://www.motron.com.br/?pg=mr210>>.

$$\frac{n_1}{n_2} = \frac{T_2}{R} = \frac{144}{0.3} = 480N$$

Então a força aplicada nas duas rodas é igual a 960 N.

A força necessária para mover a cadeira em uma superfície horizontal plano é dada pela seguinte equação:

$$F = F_a + F_r$$

Onde F_a é a força de aceleração e F_r é a força de resistência causada pelo atrito estático.

$$F_a = M * a$$

$$F_R = M * g * \mu$$

A Rotação nominal do motor é de 215 RPM, usando a relação de transmissão a rotação da roda movida é reduzida seis vezes, então $n_2 = 35,84$ RPM. A partir do valor de n_2 e do raio da roda é possível encontrar sua velocidade linear.

$$V_2 = 35.84 * \frac{2 * \pi}{60} * 0.3 = 1.1m/s$$

A partir do tempo de 4 segundos definido para se alcançar a velocidade máxima é possível calcular a aceleração da cadeira.

$$a = \frac{1.1}{4} = 0.28m/s^2$$

A partir da equação para força movida é possível encontrar a massa que os motores poderão mover.

$$960 = M * a + M * g * \mu = 0.28M + 2.94M$$

$$M = \frac{960}{3.22} = 298Kg$$

3.2 Dimensionamento da bateria

O dimensionamento da bateria foi realizado de acordo com o equacionamento e os dados obtidos a partir das figuras a seguir.



Figura 12 – Dados da bateria usada no projeto.

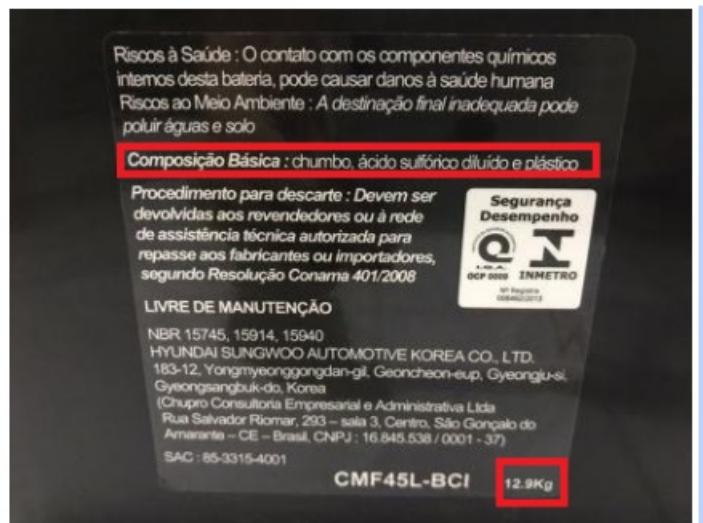


Figura 13 – Dados da bateria usada no projeto.

Foi definido que para fins de teste e apresentação o tempo de funcionamento da cadeira será de 15 minutos, com essa informação e a corrente nominal do sistema, de 24A, é possível estimar a bateria necessária.

$$M = 24 * 0.25 = 6Ah$$

O grupo adquiriu uma bateria automotiva de 45 Ah para ser utilizada nos testes, o que é mais de sete vezes maior do que a bateria dimensionada.

3.3 Carregador para bateria

Um carregador constitui-se de uma fonte que estabelece uma corrente em sentido contrário na célula, pilha ou bateria que se deseja carregar. O conceito em volta de se recarregar uma bateria é bastante simples: ao passar pela substância fornecedora de energia uma corrente no sentido contrário àquela que ela fornece normalmente, a reação se inverte e a substância "absorve" a energia liberada, voltando à sua condição inicial (BAHIA, 2016).

A recarga da bateria pode ser feita de forma rápida ou lenta. A carga rápida ou lenta nada mais é do que o nível de corrente aplicado numa bateria para atingir o seu nível de carga num determinado tempo. A carga rápida consiste em aplicar níveis de correntes elevados para reduzir o tempo de recarga. A elevação da corrente provoca superaquecimento da bateria e em consequência reduz significativamente a sua vida útil. Este tipo de recarga rápida não é recomendado. O sistema recomendado para a recarga de baterias é o de carga lenta, pois o de carga rápida provoca muitos danos à bateria e por isso, não deve ser utilizado. Para isso, recomenda-se utilizar como corrente de recarga mínima de 5% e máxima de 10% da capacidade nominal em Ah da bateria (SOARES; DAMASCENO; BACHAL, 2012).

Tendo em vista a bateria escolhida para o projeto, que possui 45Ah de capacidade nominal, o carregador desenvolvido para a mesma deve fornecer corrente de recarga de aproximadamente 4,5A por volta de 10h. Como já explicitado, esse tempo se faz necessário para que a vida útil da bateria não seja comprometida. Dessa forma, de acordo com a disponibilidade de componentes encontrados no mercado, para a confecção do carregador, foi possível desenvolvê-lo de tal forma a fornecer 5A de corrente de recarga, diminuindo então, o tempo de recarga para 9h. Assim sendo, o carregador proposto foi desenvolvido seguindo as orientações descritas nos circuitos ilustrados nas figuras 14 e 15, utilizando os seguintes materiais:

- 1 transformador (entrada 110/220V, saída 15V 5^a);
- 1 ponte retificadora 10A;
- fios vermelho e preto;
- 2 garras modelo “jacaré” (1 vermelho e 1 preto);
- 1 cabo de força;
- 1 fusível 10A;
- 1 porta fusível;
- 1 LED;

- 1 resistor de $1K\Omega$;
- botão de liga desliga e caixa de madeira.

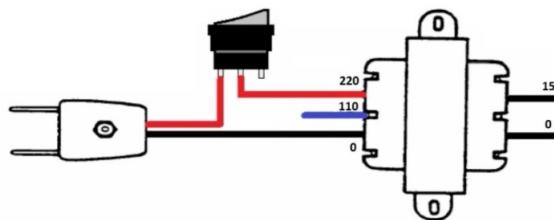


Figura 14 – Circuito de entrada do transformador.

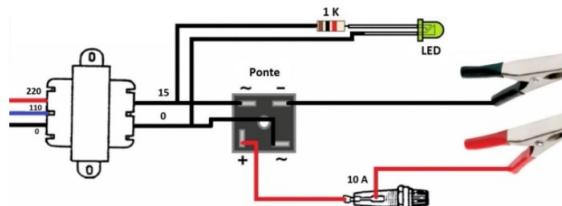


Figura 15 – Circuito de saída do transformador.

Durante o carregamento da bateria, com o auxílio de um multímetro, pode-se observar que a tensão em seus terminais irá aumentar gradativamente, quando esse aumento gradativo cessar significará que a bateria estará totalmente carregada. Para determinar esse valor em que a tensão irá parar de aumentar, deve-se aplicar a seguinte razão:

$$V_{CC} = V_{CA} * \sqrt{2} - V_D$$

Onde:

- V_{CC} – Tensão máxima na bateria
- V_{CA} – Tensão alternada que sai do transformador
- V_D – Perda existente na ponte retificadora

O transformador utilizado tem saída de aproximadamente 15 V RMS, equivalente a 21 V de pico, dessa forma a tensão se torna excessiva para a carga da bateria:

$$V_{CC} = 14.8 * \sqrt{2} - 1.4$$

$$V_{CC} = 19.6V$$

Para solucionar esse problema, um regulador de tensão será acoplado após a ponte retificadora, garantindo que a saída do carregador seja sempre cerca de 15 V. Portanto, após aproximadamente 9h, e constatado por volta de 15 V nos terminais da bateria em processo de carregamento, a bateria pode ser considerada totalmente carregada.

3.4 Controle PWM e Ponte H

Para o controle do motor através de um *joystick* foi projetado e implementado um *driver* que consiste basicamente de quatro partes para seu funcionamento, sendo elas: Circuito de chaveamento, ponte H, dobrador de tensão e o código de controle.

3.4.1 Circuito de Chaveamento

A Figura 16 ilustra o circuito de chaveamento, o qual é responsável por chavear as entradas do PWM com a ponte H e indicar ao circuito para qual sentido o motor deve rodar, seja ela no seu sentido direto ou inverso. O circuito projetado funciona da seguinte maneira:

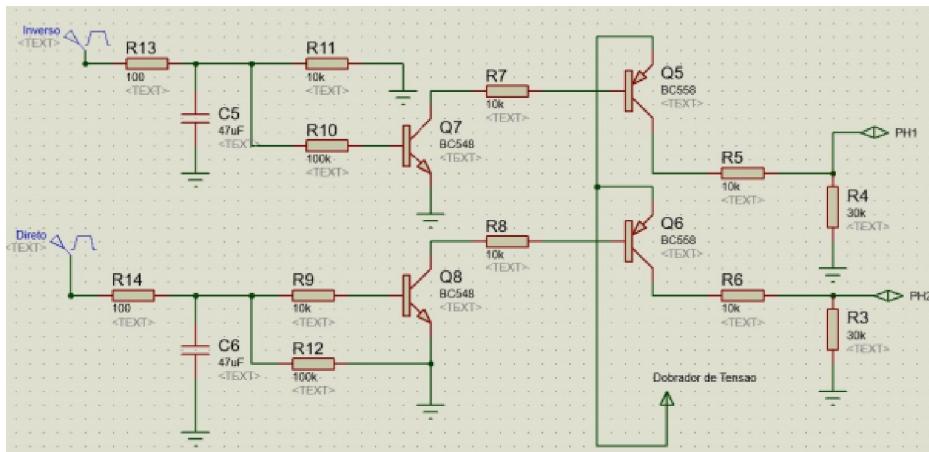


Figura 16 – Circuito de Chaveamento.

- Para que ocorra passagem de corrente nos transistores Q5 e Q6 é necessário que os transistores Q7 e Q8 estejam em saturação e, portanto, quando em corte, nenhum dos transistores é capaz de transmitir corrente;
- Os capacitores C6 e C5 e os resistores R13 e R14 funcionam como filtros passa-baixa que são responsáveis por obter a tensão analógica para o PWM;

- Com a tensão dos PWMs obtida ocorrerá a saturação de Q7 e Q8 ou não, depende dos valores que forem entrados pelo PWM. Suponha-se que a tensão do PWM Direto entre com 5V e o PWM Inverso com 0V, apenas o transistor Q8 estará em saturação, deixando que exista corrente apenas entre o coletor e emissor de Q8, sendo Q7 em corte e sem corrente.
- Os transistores Q5 e Q6 recebem em seu emissor uma tensão dobrada da fonte de tensão, a necessidade de uma tensão dobrada vai ser explicada mais adiante. Quando Q7 e Q8 estão em saturação ocorre então uma saturação também nos transistores Q5 e Q6, existindo então uma passagem de corrente entre o emissor-coletor de Q5 e Q6. Pegando a mesma suposição feita no item acima, Q5 permanece em corte e portanto sem corrente para a saída PH2, gerando assim uma tensão de 0V, no passo que Q6 em saturação permite a passagem de corrente entre emissor-coletor e PH1 tem sua saída determinada pelo divisor de tensão R6 e R3.
- As saídas PH1 e PH2 são as saídas que são ligadas a ponte H para que o motor possa girar no sentido desejado.

O teste do circuito de chaveamento foi feito em conjunto do circuito de dobrador de tensão, explicado mais à frente, os circuitos foram montados em uma placa *protoboard*. Posteriormente, para se instalar na cadeira, os circuitos serão feitos em placas de circuito impresso (PCB). A imagem abaixo ilustra o circuito de teste montado.

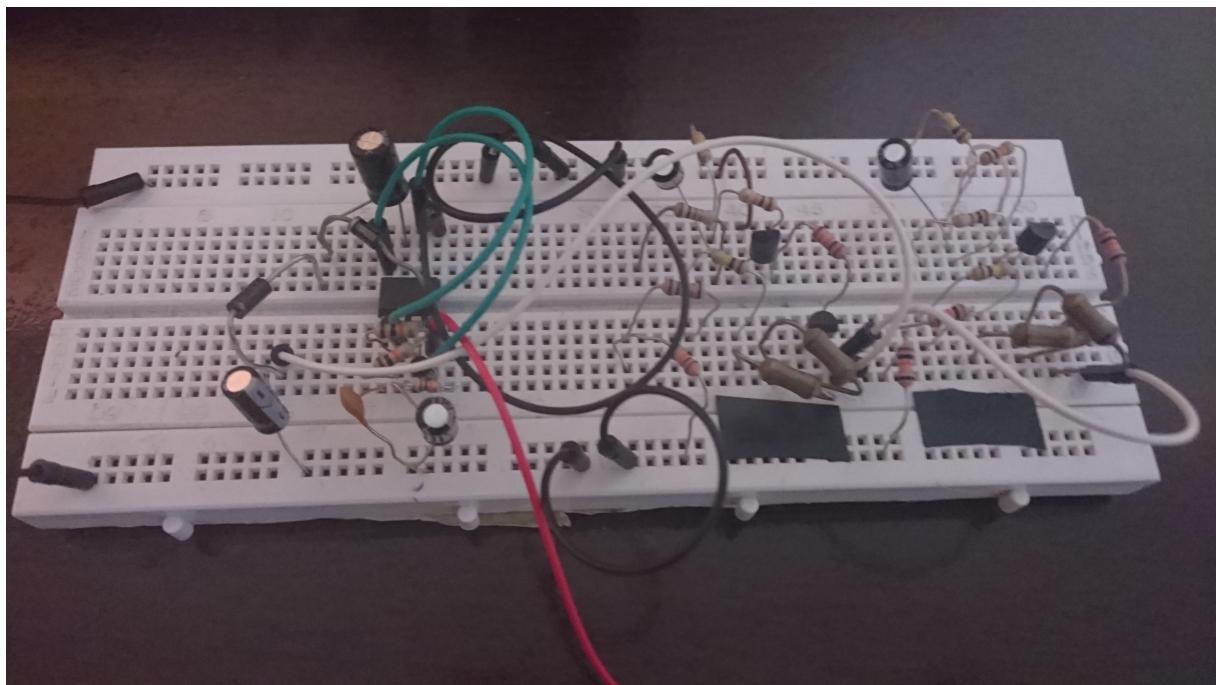


Figura 17 – Circuito de chaveamento (à direita) e circuito dobrador de tensão (à esquerda).

3.4.2 Ponte H

A ponte H é responsável por fazer os motores girarem para os sentidos desejados, ou seja, sentido direto ou sentido inverso. A Figura 18 ilustra a ponte H realizada.

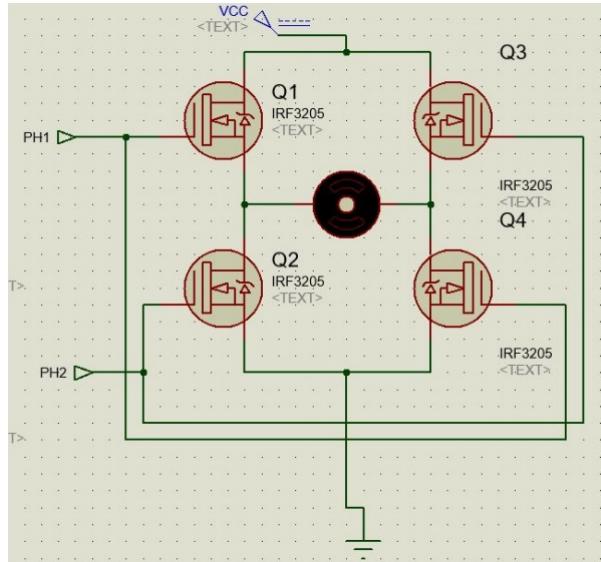


Figura 18 – Ponte H.

A ponte H projetada funciona da seguinte maneira:

- Os transistores Q1 e Q3 recebem no seu terminal DRAIN a tensão da bateria de 12V e no seu terminal GATE a tensão advinda do chaveamento da parte 1. Para que exista corrente entre os terminais DRAIN e SOURCE os transistores devem estar em saturação e, portanto, a tensão que chega ao GATE desses transistores para que possa estar em saturação e entregar ao motor 12V exigidos pelo mesmo em seu terminal SOURCE, deve ser uma tensão de 12V + 4.5V, pois 12V é a tensão entregue pelo terminal SOURCE e 4.5V é a tensão de limiar do transistor IRF3205 em uso, ou seja, para o correto funcionamento dos transistores Q1 e Q3 é necessário que uma tensão de 16.5V seja entregue por PH1 e PH2.
- O dobrador de tensão serve para dar essa tensão a mais que a tensão entregue pela bateria, pois 12 V não seriam o suficiente para que os motores girassem em sua plenitude visto que o terminal GATE de Q1 e Q3 necessitam de 16.5 V.
- Os transistores Q2 e Q4 são os transistores responsáveis por ligar a ponte H ao terra do circuito, possibilitando a passagem de corrente pelos motores.
- Os transistores Q1 e Q4 são curto-circuitados e Q2 e Q3 também. Estas ligações são as responsáveis por dizer o sentido da corrente e portanto o sentido que o motor vai girar. Quando Q1 e Q4 recebem 16.5V em seu GATE e Q2 e Q3 0V em seu

GATE, aqueles entram em saturação enquanto estes entram em corte fazendo com que a corrente percorra o sentido esquerda-direita. Já quando o contrário ocorre e Q2 e Q3 estão em saturação e Q1 e Q4 em corte, a corrente percorre o sentido direita-esquerda.

- A condição dos 4 transistores estarem em saturação deve ser evitada ou melhor, não deve ocorrer, uma vez que os transistores podem superaquecer enfrentando problemas para o circuito, portanto esta condição foi pensada e efetuada nos controles de PWM.

A placa de testes da ponte H pode ser vista na imagem abaixo. A montagem da mesma foi feita em uma placa de fenolite furada, de 10x10 cm, com os componentes soldados e com as conexões realizadas utilizando fios de cobre com $6.5\ mm^2$ de seção nominal, que são suficientes para correntes de até 80 A de acordo com as legislações atuais. A placa final à ser instalada na cadeira será feita com placas de circuito impresso, atentando ao fato de que suas trilhas devem suportar correntes de até 50 A.

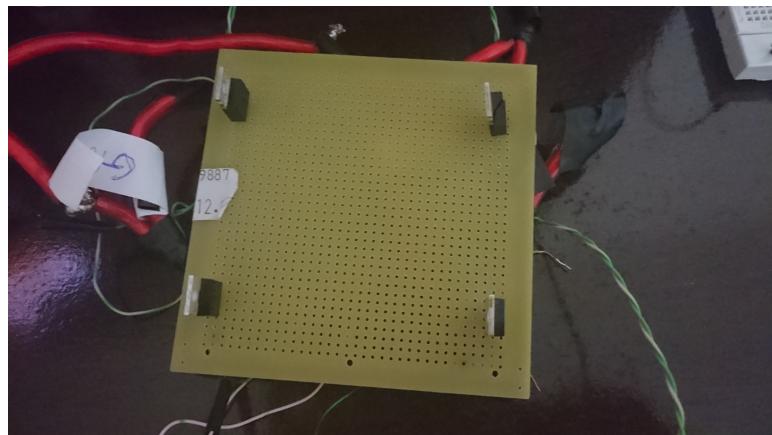


Figura 19 – Placa de teste da ponte H.

3.4.3 Dobrador de tensão

O circuito dobrador de tensão, como dito anteriormente, é necessário para realizar o chaveamento do circuito. O dobrador foi realizado utilizando o CI 555 em sua configuração astável. A Figura 20 ilustra o circuito do dobrador de tensão.

3.4.4 Controlador PWM - Software

Para se controlar os motores através do *joystick*, foi utilizado um microcontrolador Arduino Nano.

O *joystick* é constituído de 2 potenciômetros que têm seus valores analógicos medidos pelo Arduíno. Em seu ponto inicial, os valores digitais do eixo x e y do *joystick* são,

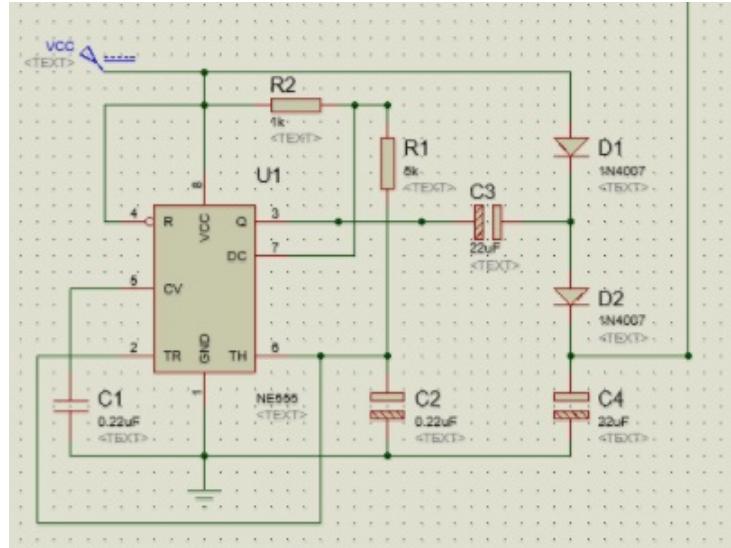


Figura 20 – Circuito do dobrador de tensão.

respectivamente, 511 e 491. Na direção positiva, os valores de y e x diminuem até chegarem à 0 em seu limite positivo, já na direção negativa, os valores aumentam até chegarem à 1024, em seu limite negativo. A Figura 21 mostra o *joystick* e o Arduíno utilizados

O sinal de PWM é gerado utilizando os timers do Arduíno. Cada motor possui 2 pulsos PWM (um para controlar a corrente no sentido direto e outro para o inverso), assim sendo, são necessários 4 timers para o controle total da cadeira. No código, os timers utilizados foram os TCCR1A, TCCR1B, TCCR2A, TCCR2B. O dutycycle dos PWMs pode ser modificados de acordo com o valor inserido em sua função, com 255 sendo 100% de DutyCycle e 0 sendo 0%.

Foram criadas funções para os movimentos da cadeira (parado, frente, trás, curva à direita e curva à esquerda). Essas funções são chamadas quando executadas pelo usuário no *joystick*, através de lógica condicional. Também é modificado o DutyCycle do PWM de acordo com a intensidade do comando no *joystick*. Um exemplo de função pode ser visto na figura acima.

Há também condições para garantir que a cadeira não dê uma aceleração súbita, evitando assim riscos de quedas por parte do usuário. Essas funções garantem que, ao acionar os motores com força acima de 75% saindo do repouso, os sinais de PWM aumentem seus dutycycles pausadamente, indo de 0% a 50%, de 50% a 70% e de 70% até o valor selecionado originalmente, com intervalos entre os acréscimos.

```
void turnRight(float dutyRatio)
{
    //Condicao para evitar arranque da cadeira
    if(dutyRatio > 0.75)
    {
        if(flag_smoothX == 0)
        {
            leftPWM(0.5, 0);
            rightPWM(0.02, 0);
            delay(100);           //espera 100 ms
            leftPWM(0.7, 0);
            rightPWM(0.02, 0);
            delay(100);           //espera 100 ms
            leftPWM(dutyRatio, 0);
            rightPWM(0.02, 0);
        }
        else
        {
            leftPWM(dutyRatio, 0);
            rightPWM(0.02, 0);
        }
    }
    else
    {
        leftPWM(dutyRatio, 0);           //PWM com o duty cicle para controle da tensao no motor esquerdo
        rightPWM(0.02, 0);             //PWM com o duty cicle para controle da tensao no motor direito
    }

    Serial.write("\n turning right");
}
```

Figura 21 – Exemplo de função usado no controle PWM.

4 Estruturas

Como explicado no último relatório, o subsistema estrutural ficou encarregado de fazer a escolha de como obter uma cadeira de rodas e como fazer a transmissão do torque do motor para a roda. O Primeiramente, foi definido como seria feita a aquisição da cadeira de roda, se seria manufaturada, comprada ou emprestada. Pelo curto período de desenvolvimento de todo o projeto, e pelo fato dos componentes do grupo não terem experiência com equipamentos de solda e modelagem de gabaritos para a manufatura da estrutura, foi descartada a hipótese de construção da cadeira. A segunda opção foi a estudada a possibilidade de comprar uma cadeira de roda, porém não houve possibilidade de arrecadar fundos para esse investimento e pelo tempo não houve a captação de investidores para o projeto. Por fim, com o auxílio do Prof. Rhander Viana foi obtida uma cadeira de rodas do Hospital Universitário de Brasília (HUB) onde foi emprestada uma cadeira de roda Jaguaribe 1009. Ao fim do projeto, a cadeira de roda será devolvida ao hospital.

O segundo ponto principal destinado a estrutura, foi a escolha de um tipo de transmissão para realizar a movimentação da cadeira. Em conjunto com o subsistema de alimentação a forma escolhida para transmitir o torque do motor para a cadeira foi a utilização de um disco de atrito que fica em contato com a roda. Foi escolhida esta maneira para transmissão pelo fato de manufaturar uma caixa de redução para o motor seria algo que não teria tempo hábil, ou seria algo fora dos padrões orçamentários do projeto.

4.1 Estrutura da Cadeira de Rodas

Como explicado anteriormente, foi escolhido a aquisição de uma cadeira junto ao Hospital Universitário de Brasília, onde foi realizado o empréstimo da cadeira até o fim do projeto. A cadeira é a Cadeira de Rodas Jaguaribe 1009, como mostra a Figura 22 abaixo.

Esta cadeira é composta de uma estrutura metálica de aço carbono 1009, que possui características de ser mais leve e suporta normalmente 90kg. O assento e o encosto traseiro são constituídos de courvim, um tipo de tecido muito utilizado para este fim.

A cadeira adquirida pelo grupo tinha algumas avarias, como rodas empenadas, alguns aros das rodas faltando, tecido de courvim do assento e do encosto rasgados e câmara de ar dos pneus rompidas. Foi necessário fazer a desempenagem das rodas, troca da câmara de ar dos pneus, e compra dos aros para ter uma cadeira em perfeito estado.

Como não foi necessário a manufatura da cadeira, apenas realizamos a modelagem



Figura 22 – Cadeira de Rodas Jaguaribe 1009.

da cadeira de rodas em ambiente CAD para ter uma primeira impressão das primeiras modificações que seriam necessárias. Como o escopo do projeto abrange uma cadeira de rodas elétrica que faz monitoramentos de saúde do usuário, a cadeira teria que possuir alguns detalhes na estrutura. A cadeira modelada em ambiente CAD é apresentada na Figura 23.

Na Figura 23 é possível perceber a inserção da disco de atrito no projeto, para realização do papel da transmissão, utilização de uma placa para suportar o disco de atrito e o motor, além da gaveta para suportar o peso e tornar de fácil remoção para recarga a bateria.

4.2 Análises assistidas por Computador (CAD/CAE)

Antes de manufaturar, houve um estudo minucioso em relação à como a estrutura se comportaria em relação as cargas que são inseridas no sistema. Como a cadeira já é planejada para suportar 90kg, e por ser um produto de grande escala já é feito as análises estáticas, modal e de fadiga da mesma na própria indústria para validar o projeto. Porém, como será adicionado parâmetros que não são de fábrica, serão feitas análises tanto estática quanto modal na cadeira para ter o projeto possuir confiabilidade. A análise estática é utilizada para identificar tanto as deformações como as tensões que a estrutura sofre quando excitada uma força. No projeto, foi utilizado o Software Solid Works para fazer a análise, os parâmetros de estudo foram a Tensão de Von Mises e Deformação de Von Mises. Assim para a estrutura ser aceita, é necessário que a mesma tenha valores de tensão e deformação que não cause dano à estrutura, neste caso não atinja a tensão de escoamento do material. A cadeira de roda como citado anteriormente é constituída de aço carbono



Figura 23 – Desenho em ambiente CAD da Cadeira de Rodas.

1009, onde é composto por 0,09% de carbono, sendo assim um aço muito dúctil com baixa tensão de escoamento, no caso 170MPa. Porém, como o Solid Works possui uma lista de materiais reduzida, onde a tensão de escoamento mais próxima da usada na cadeira é de 206MPa, com o aço 1010, que é muito parecido com o 1009, sendo assim, para a análise estática será usado este material. A Figura 24 demonstra os dados obtidos com a análise.

Na análise foi omitido diversos pontos, para que a análise fosse mais rápida, porém todas as estruturas que foram removidas foram levada em consideração, para isso estes componentes foram substituídos por forças.

Como pode ser visto na Figura 24, a tensão máxima que ocorre na estrutura é de 221MPa, ou seja, a estrutura escoou, no caso passou do regime elástico para o plástico. Porém, como pode ser visto também onde há a maior tensão de escoamento são em pontos muito próximos ou quase nos apoios, ou seja, a teoria de mecânica dos sólidos nos trás que uma carga no apoio não causa deformação, sendo assim, será descartado os valores nos apoios, assim sobrando valores entre 92 e 129MPa, ou seja, validando o projeto.

Para a Análise Modal, foi ensaiado também no Solid Works, esta análise tem como objetivo entregar ao usuário as frequências naturais que o produto tem, ou seja, as frequências que a cadeira não pode ser excitada. A análise modal funciona da seguinte forma,

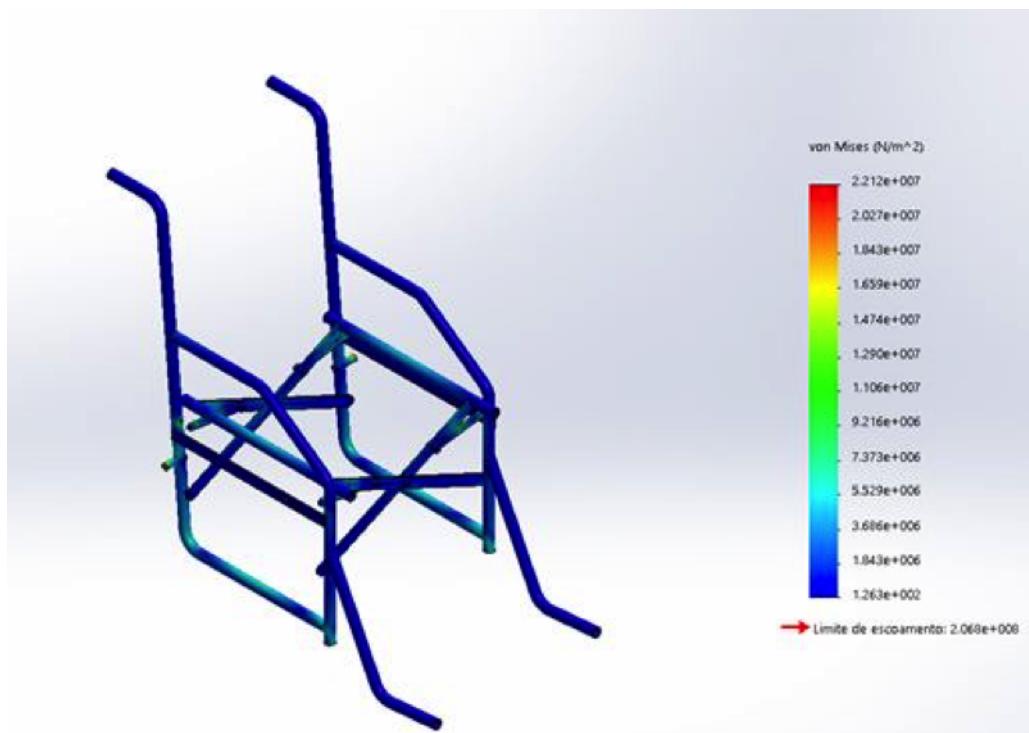


Figura 24 – Análise Estática da Cadeira de Roda.

a peça é deixada sem apoios, e assim sofre uma excitação, de acordo com a frequência de resposta da peça temos os valores da frequência natural. Como possuímos 2 motores, que excitam a cadeira a 3 Hz aproximadamente, é necessário que a cadeira tenha frequências naturais maiores ou menores que este valor, caso contrário o projeto pode entrar em colapso. Com as análises feitas no Software, obtivemos os seguintes valores:

Tabela 1 – Valores da Análise Modal.

Número da frequência	Rad/s	Hertz
1	0	0
2	0	0
3	0	0
4	0.0028431	0.00045249
5	0.0063894	0.0010169
6	0.010349	0.0016472
7	0.1351	0.021501
8	0.17398	0.02769
9	311.96	49.65
10	342.36	54.488

Os modos de frequência, que estão situados na primeira coluna condizem com os modos em que o produto foi excitado, os valores de 1 até 6 deram muito próximo de 0 pelo fato destes modos serem de translação e rotação. A partir do 7 já existem frequências consideráveis, assim sendo estas que serão consideradas. Como a frequência de trabalho é

de 3Hz, temos que as frequências estão longe de alcançar este ponto, a menos que exista algum problema de desalinhamento ou desbalanceamento.

4.3 Processo de Fabricação das Peças

Para a manufatura do disco de atrito, foi necessário a utilização de técnicas de processos de fabricação e conformação mecânica. Para a realização desta etapa foi necessário a compra de um tarugo de alumínio que seria capaz de produzir dois discos, um para cada roda. Todo o processamento do produto foi realizado no Galpão da Universidade de Brasília Campus Gama, com o auxílio dos Técnicos de Máquinas do local. Para a usinagem do tarugo foi necessário utilização do torno para realizar os rebaixos necessários, foi utilizado também uma técnica de conformação mecânica que tornava a superfície do disco áspera, fazendo com que a mesma obtivesse o coeficiente de atrito necessário para realização do movimento da cadeira. Após esta conformação, foi realizado o corte do disco e a usinagem dos furos necessários para a utilização do disco. Nesta etapa foi furado na furadeira de bancada com uma broca com diâmetro praticamente igual ao do eixo do motor, para haver uma conexão com interferência. Além da fabricação de um furo para a passagem de um parafuso mosca, além da manufatura das roscas para entrada do parafuso, que tem como função o melhor travamento entre o eixo do motor e o disco de atrito. Na Figura 25 temos o Desenho técnico do disco de atrito.

Após o término do disco, foi manufaturada a placa que faz o suporte e fixação entre o eixo do motor, disco de atrito e a cadeira de roda. Para conseguir uma geometria que encaixasse de maneira ótima, foi necessário fazer um gabarito de um material que fosse de fácil usinagem, no caso o material escolhido foi o MDF. Com o auxílio do motor, foi feito os furos na furadeira de bancada de acordo com os furos do motor, onde eram feitos testes imediatamente após o furo para ter certeza do tamanho do furo e de sua posição. Depois de encontrar um padrão para a peça, foi usinada uma chapa de aço onde a mesma foi furada e esmerilhada para chegar à geometria padrão. Os passos foram replicados para os dois motores. A Figura 26 demonstra o desenho técnico da placa de suporte.

Para não ser necessário mexer na estrutura da cadeira, foi decidido em amassar o freio da cadeira de roda, de maneira que ele sirva de apoio, onde fizemos furos para acoplar a placa que suporta o eixo. Desta forma, a cadeira irá poder trabalhar tanto no manual quanto no automático, pelo fato do freio da cadeira movimentar a placa e o disco de atrito. Esta foi uma das inúmeras preocupações, tendo em vista que teria que existir um contato significativo entre o pneu e o disco de atrito, por isso foram feitos vários testes em relação à posição da placa, para ter a certeza de que a cadeira irá funcionar das duas formas.

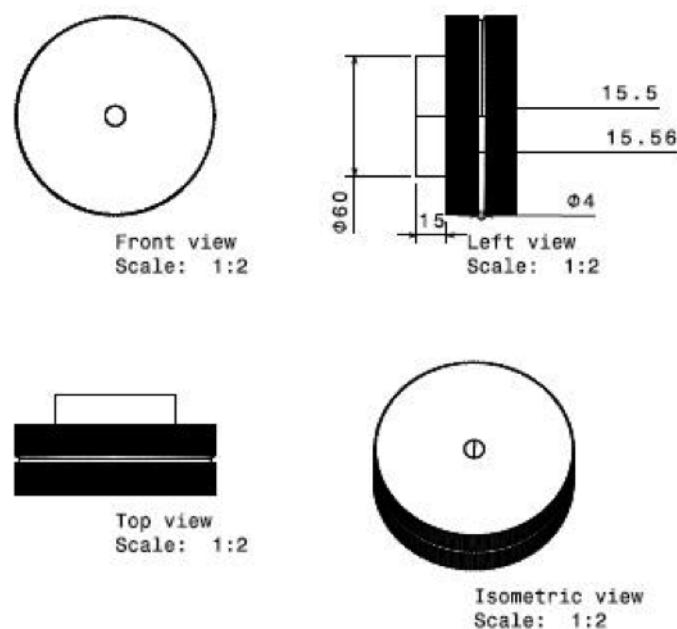


Figura 25 – Desenho técnico disco de atrito.

Assim, a cadeira neste momento está disposta da seguinte forma, como mostra a Figura 27.

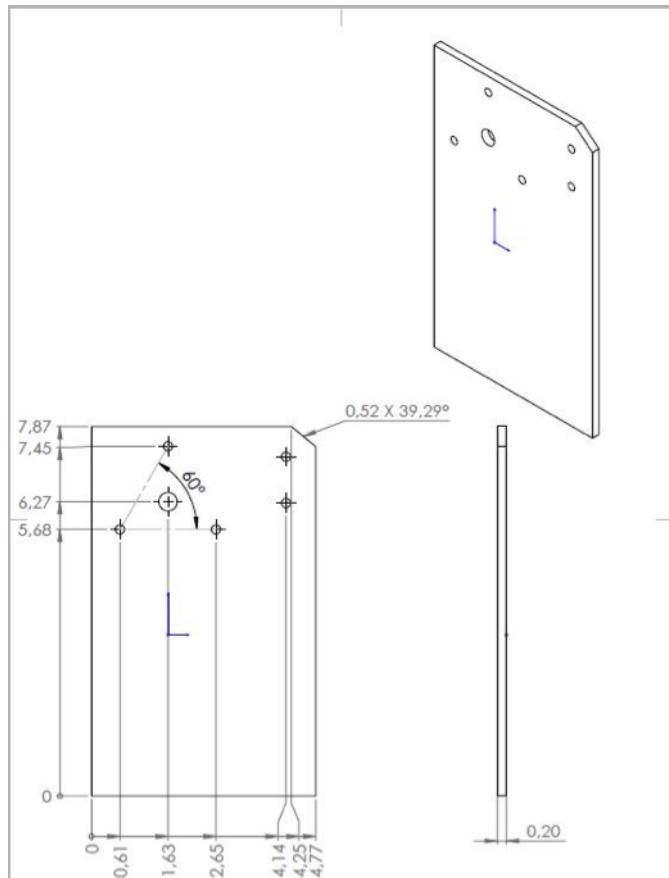


Figura 26 – Desenho Técnico da placa de suporte do motor e do disco de atrito



Figura 27 – Desenho do estado atual da Cadeira de Roda.

5 Plano de Integração

Neste capítulo apresentamos nosso planejamento para a integração dos diferentes subsistemas, que deve ocorrer na terceira etapa do projeto. Adaptamos os pontos levantados no guia do PMBOK, adequando o plano ao contexto da disciplina, e focando na integração de subsistemas de um projeto de engenharia, e não da área de gestão de pessoas.

A integração dos subsistemas ocorrerá no tempo das aulas, contudo, caso necessário, em outros horários o grupo se reunirá de forma a conseguir a integração completa. Ainda, ressaltamos que o subsistema de Processamento de Sinais e Monitoramento não é fortemente acoplado aos outros subsistemas, de modo que a integração entre ele e o restante do projeto será fácil e não carecerá de um grande esforço. Os outros dois subsistemas, Projeto Estrutural e Controle e Alimentação, por outro lado, são bem acoplados, e um maior cuidado será tomado a respeito desses dois subsistemas.

5.1 Integração do subsistema Processamento de Sinais e Monitoramento

Como mencionado, o subsistema de Processamento de Sinais e Monitoramento não deverá apresentar grandes problemas na integração, principalmente por não ser acoplado aos outros subsistemas. Os únicos componentes físicos desse subsistema são os componentes eletrônicos relacionados a aquisição de sinais (sensores, amplificadores, filtros e conversores) e o sistema embarcado (Raspberry Pi). Os outros componentes (servidor remoto, cliente *frontend* e *mobile*) estão na nuvem, e não causam impacto na integração com os outros componentes físicos.

Para o terceiro ponto de controle também faremos o desenvolvimento dos componentes que tornem o UMISS mais tolerante a falha, através de circuitos que adicionem redundância ao projeto. Essa tolerância ocorrerá através da adição do módulo GPRS/3G, que possibilitará uma alternância no envio de dados caso a *internet* da residência do paciente esteja indisponível, ou ainda em casos onde a energia da residência ou fatores externos afetem a conexão. Outro esforço será empregado na adição dos componentes que detectam a presença do paciente na cadeira, que será importante em diversos cenários. Por exemplo, caso o paciente caia da cadeira, os sensores de temperatura podem capturar valores anômalos, que devem ser ignorados.

É esperado que os componentes físicos desse subsistema sejam alocados em um compartimento de fácil manutenção, para que seja facilmente manuseado durante os di-

versos testes. Além disso, esse compartimento deve disponibilizar saídas para os fios, que serão então conectados a outros subsistemas, ou disponibilizados para serem utilizados pelo paciente. Assim, a integração deve ocorrer da seguinte forma:

1. Alocação dos componentes eletrônicos de modo seguro, mas que ocupe o menor espaço possível, pois os outros subsistemas carecem de bastante espaço;
2. Acoplamento do compartimento na parte inferior da cadeira (nos braços), parafusando-o (incluindo o *case* da Raspberry Pi);
3. Extensão e disponibilização dos cabos e dos sensores, para que sejam facilmente utilizados pelos pacientes;
4. Extensão da bateria conectada a Raspberry Pi, e conexão entre ela e a bateria do subsistema de Controle e Alimentação.

5.2 Integração do subsistema de Estruturas

Com as etapas mencionadas anteriormente completas, o subsistema de estruturas tem por sua única preocupação futura de desenvolver componentes físicos que agreguem todos os equipamentos dos demais subsistemas. Sempre se preocupando com a facilidade de manuseio durante os testes que serão feitos. Com isso, a integração será feita da seguinte maneira e terá como objetivo ficar similar com a Figura 28, que foi criada no ambiente CAD:

1. Alocação dos equipamentos eletrônicos e do Raspberry Pi do subsistema de processamento de sinais e monitoramento com suas dimensões e integrá-los a estrutura da cadeira de maneira ergonômica, se preocupando com a disponibilização dos cabos e sensores, de forma que o usuário tenha seus sinais vitais adquiridos da maneira mais confortável possível.
2. Outra preocupação é a interface da estrutura da cadeira e todo o sistema de alimentação e controle dos motores, na qual irá ser feita um sistema de gaveta abaixo da cadeira que irá suportar a bateria de forma que ela seja de fácil acesso e removível para o seu carregamento de energia.

5.3 Cronograma



Figura 28 – Projeto da Cadeira Completa.

Tabela 2 – Cronograma para a integração dos subsistemas.

Atividade	Responsável	Deadline
Desenvolvimento do sensor de presença	Afonso	01/06
Criar compartimento com sensores e embarcado	Afonso, Dylan, Gustavo, Tiago e Wilton	08/06
Acoplamento do compartimento na parte inferior da cadeira	Dylan e Afonso	08/06
Acoplamento do módulo GPRS/3G aos outros componentes		
Extensão e disponibilização dos cabos	Gustavo, Tiago e Wilton	08/06
Extensão da bateria da Raspberry Pi	Dylan e Afonso	08/06
Teste do subsistema de Processamento e Monitoramento	Afonso, Dylan, Gustavo, Tiago e Wilton	08/06
Manufatura do sistema de gaveta	Nivaldo, Rafael e Lucas Matheus	02/06
Acoplamento da gaveta com a bateria e eletrônicos na cadeira	Nivaldo, Rafael e Lucas Matheus	09/06
Criação do compartimento dos eletrônicos do processamento de sinais e monitoramento	Nivaldo, Rafael e Lucas Matheus	16/06
Integração da atividade acima com a estrutura da cadeira	Nivaldo, Rafael e Lucas Matheus	23/06

Referências

BAHIA, M. L. Desenvolvimento de um conversor ca-cc para carga de bateria chumbo-ácido. 2016. Citado na página [26](#).

SOARES, C.; DAMASCENO, R.; BACHAL, T. V. *Sistema de controle de energia para veículos de recreação*. Dissertação (B.S. thesis) — Universidade Tecnológica Federal do Paraná, 2012. Citado na página [26](#).

TECHNOLOGY, C. *Arquitetura do servidor Django*. [S.l.], 2013. Disponível em: <<http://criticaltechnology.blogspot.com.br/2011/09/mvc-in-three-tier-architecture.html>>. Citado 2 vezes nas páginas [5](#) e [18](#).

Anexos

ANEXO A – Diagrama de Classes Backend

Diagrama de classes do backend na figura 29, desenvolvido no projeto UMISS.

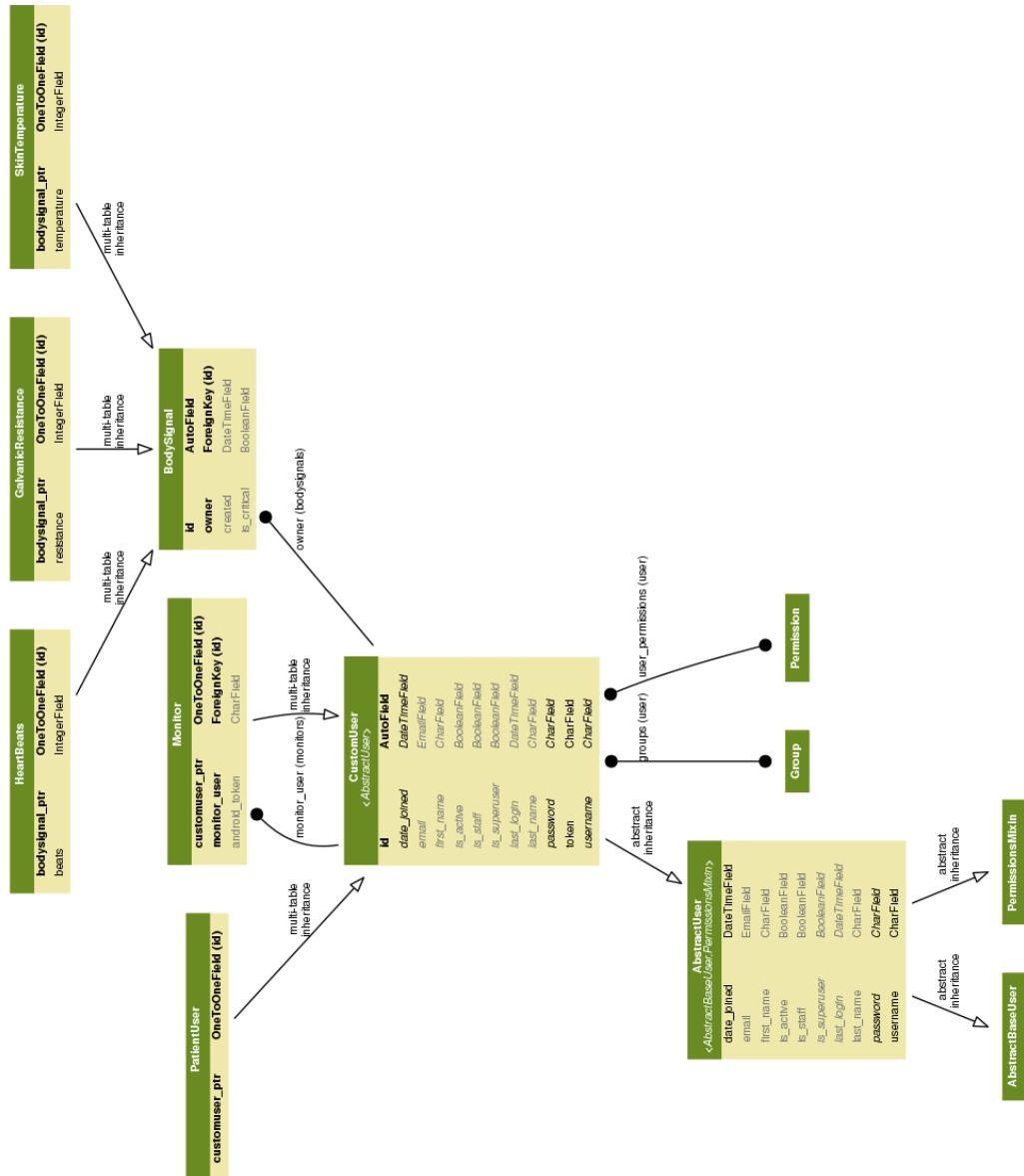


Figura 29 – Design e Arquitetura do servidor Django