Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Projeto Integrador de Engenharia 2

Robô Garçom

Autor: Ana Paula Lopes, Bruna Almeida, Damarcones Porto, Diogo Correia, Gabriel Pinheiro, Hebert Max, Isaac Moura, Isabella Feitosa, João Paulo Porto, Letícia Karla, Letícia Ribeiro, Lucas Pereira, Luiz Alexandre, Normando Perazzo e Samuel Borges

Orientador: Alex Reis, José Felício, Paolo Gessini, Ricardo Matos e Rhander Viana

Brasília, DF

2021

Ana Paula Lopes, Bruna Almeida, Damarcones Porto, Diogo Correia, Gabriel Pinheiro, Hebert Max, Isaac Moura, Isabella Feitosa, João Paulo Porto, Letícia Karla, Letícia Ribeiro, Lucas Pereira, Luiz Alexandre, Normando Perazzo e Samuel Borges

Robô Garçom

Documento submetido aos professores de Projeto Integrador de Engenharia 2.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Alex Reis, José Felício, Paolo Gessini, Ricardo Matos e Rhander Viana

Brasília, DF

2021

Resumo

Com o surgimento da pandemia, cresceu o grau de preocupação com a higiene nos locais de alimentação, onde, em maior parte do tempo do ano de 2020, ocorreu a proibição do consumo de alimentos nos estabelecimentos. Tendo em vista esse contexto, o Grupo 04 da disciplina Projeto Integrador de Engenharia 2, tem como objetivo a criação do projeto de um **Robô Garçom**, que poderá ser utilizado após a pandemia.

O projeto Robô Garçom visa atender restaurantes que tem como objetivo automatizar ao menos parte de seu atendimento e aumentar o grau de higiene em suas refeições, proporcionando conforto e segurança aos clientes.

Tendo em vista essas necessidades, o Robô Garçom dispõe de uma estrutura em que será armazenada a comida durante o transporte evitando que o alimento seja exposto ao ambiente. Ao chegar à mesa, será aberto uma porta que dará acesso ao compartimento de armazenagem onde estará o pedido realizado pelo cliente.

O Robô Garçom funciona de maneira autônoma e se locomove com o auxilio de rodas que estão acopladas em sua estrutura, dando liberdade de movimentação dentro do estabelecimento. Ocorrerá uma troca de informações entre o Robô Garçom e um *tablet* que estará posicionado na mesa, servindo como ponto de referência para a localização da mesa. O *tablet* servirá também ao cliente como meio para realização de pedidos e efetuar pagamentos.

Palavras-chaves: Robô Garçom, automação, estrutura, *software*, integração.

Sumário

1 INTRODUÇÃO . . . 12

2 TERMO DE ABERTURA DO PROJETO . . . 14
2.1 Dados do projeto . . . 14
2.2 Objetivos . . . 14
2.2.1 Objetivo geral . . . 14
2.2.2 Objetivos específicos . . . 14
2.3 Justificativa . . . 15
2.4 Problemática de automação . . . 15

3 PROJETO CONCEITUAL DO PRODUTO . . . 16
3.1 Descrição funcional do produto . . . 16
3.2 Características gerais . . . 17
3.3 Integração de subsistemas . . . 17

4 REQUISITOS DO SISTEMA . . . 19
4.1 Requisitos Gerais . . . 19
4.2 Requisitos de *Software* . . . 20
4.3 Requisitos de Automação . . . 22
4.4 Requisitos de Estrutura . . . 23
4.5 Legislação e Normas . . . 24
4.6 Indicadores . . . 25

5 ESTRUTURA . . . 26
5.1 Dimensionamento do motor do fuso . . . 26
5.2 Dimensionamento do motor da tampa . . . 28
5.3 Dimensionamento da mesa de levantamento . . . 30
5.4 Dimensionamento do garfo da roda . . . 32

6 AUTOMAÇÃO . . . 34
6.1 Funcionamento . . . 34
6.1.1 Sensoriamento e Controle . . . 34
6.1.2 Alimentação . . . 35
6.1.3 Integração de Hardware . . . 36
6.2 Diagramas Elétricos e Eletrônicos . . . 37
6.3 Carregamento e Armazenamento de Energia Elétrica . . . 45
6.3.1 Carregamento Indutivo . . . 45

6.3.2 Base de Carregamento 46
6.3.3 Baterias 47
6.4 Comunicação 48
6.4.1 Dispositivo comunicador 48
6.4.2 Alarme 48
6.5 Locomoção 48
6.5.1 Dispositivo controlador 49
6.5.2 Motores 49
6.5.3 *Drivers* 49
6.5.4 Sensor acelerômetro 50
6.5.5 Sensores infravermelho 50
6.5.6 Sensor ultrassônico 51
6.5.7 Sensor de peso 51
6.5.8 Localizador 52
6.6 Programação dos módulos 54
6.6.1 Código para o controlador 54
6.6.2 Código para o comunicador 55
6.6.3 Código para o comunicador da base de carregamento 56
6.7 Plano de Fabricação 59
6.7.1 PCBs 59
6.7.2 Bobinas 63
6.7.3 Estrutura da base de carregamento 64
6.8 Plano de Montagem 64
6.8.1 Instalação da antena de wi-fi 65
6.8.2 Configuração do controlador 65
6.8.3 Sensor de peso (Calibração) 66
6.8.4 Configuração do comunicador 66
6.8.5 Cabeamento Elétrico 67
6.8.6 Conexões do Cabeamento Elétrico 67
6.9 Testes 68
6.9.1 Localizador 68
6.9.2 Sensor de Peso 68
6.9.3 Baterias 68
6.9.4 Bobinas 69
6.9.5 Sistema elétrico 69
6.9.6 Comunicador 69

7 *SOFTWARE* 71
7.1 Arquitetura do *Software* 71
7.2 Arquitetura da informação 73

7.3 Inovações 77
7.4 Modelagem do sistema 78
7.5 Testes 83
7.6 Manuais 84

8 CRONOGRAMA DO PROJETO 85
8.1 Atividades de Automação 88
8.2 Atividades de Estruturas 89
8.3 Atividades de *Software* 90

REFERÊNCIAS 94

APÊNDICE A – EQUIPE DE TRABALHO 98
A.1 Membros e atribuições 98
A.2 Ferramentas de Gerenciamento 99

APÊNDICE B – ESTRUTURA ANALÍTICA DO PROJETO (EAP) 101

APÊNDICE C – ANÁLISE DE MERCADO 102
C.1 Nome comercial e conceito para o produto 102
C.2 Mercado-alvo 102
C.2.1 Consumidor final 102
C.2.2 Cliente responsável pela comercialização 102
C.3 Concorrência 102
C.3.1 Concorrentes 102
C.3.2 Viabilidade técnica 103

APÊNDICE D – GERENCIAMENTO DE RISCOS 105
D.1 Levantamento de riscos 105
D.2 Plano de contingências 110

APÊNDICE E – ORÇAMENTO DO PROJETO 113
E.1 Custos para Automação 113
E.2 Custos de Estrutura 116
E.2.1 Componentes Estáticos 117
E.2.2 Componentes Móveis 117
E.2.3 Estimativa de Custo 117
E.3 Custos de Software 119
E.4 Orçamento geral 122

APÊNDICE F – TÓPICOS DE AUTOMAÇÃO 123

F.1 Tomada de decisão para automação . . . 123
F.2 Simulações eletro-eletrônicas . . . 126
F.2.1 Motores . . . 126
F.2.2 Sensor de peso . . . 131
F.2.3 Sensores infravermelho e Ultrassônico . . . 133
F.2.4 Bobina . . . 134
F.2.5 Carregamento . . . 135
F.2.6 Modelagem térmica . . . 136

APÊNDICE G – ESTIMATIVA DE CONSUMO ENERGÉTICO . . . 139

APÊNDICE H – ESCOPO DE SOFTWARE . . . 140
H.1 Metodologia . . . 140
H.2 Elicitação e priorização de requisitos de Software . . . 141
H.3 Decisões de *Software* . . . 142
H.3.1 Arquitetura do *Software* . . . 142
H.3.2 Integração com automação . . . 143

APÊNDICE I – CÓDIGOS . . . 144
I.1 Códigos para *Software* . . . 144
I.1.1 Docker . . . 144
I.1.2 Controller de autenticação . . . 145
I.1.3 Banco de dados . . . 146
I.1.4 Johnny 5 . . . 147
I.2 Códigos para a automação . . . 148
I.2.1 Comunicador *wireless* . . . 148
I.2.2 Códigos para o controlador . . . 153
I.2.2.1 Código para calibração da balança . . . 166

APÊNDICE J – DESENHOS TÉCNICOS DA ESTRUTURA . . . 167

APÊNDICE K – CARCAÇA DO CARREGADOR . . . 186

Lista de ilustrações

Figura 1 – Visão geral do funcionamento do produto. 16
Figura 2 – Integração de arquiteturas. 17
Figura 3 – Gráfico de velocidade por tempo desejado durante o levantamento. 27
Figura 4 – Aplicação de simetria na simulação da mesa de levantamento. 30
Figura 5 – Aplicação das condições de apoio na simulação da mesa de levantamento 30
Figura 6 – Aplicação da carga na simulação da mesa de levantamento. 31
Figura 7 – Distribuição de fator de segurança (critério de von-Mises) na mesa de levantamento 31
Figura 8 – Distribuição de fator de segurança (critério de von-Mises) abaixo de 3 na mesa de levantamento 32
Figura 9 – Apoio engastado na simulação do garfo. 32
Figura 10 – Cargas na simulação do garfo. 33
Figura 11 – Distribuição de fator de segurança (critério de von-Mises) no garfo. 33
Figura 12 – Diagrama de caixa do hardware do robô. 36
Figura 13 – Unidade Primária de carregamento. 36
Figura 14 – Unidade Secundária de carregamento. 37
Figura 15 – Diagrama elétrico com Barramentos. 38
Figura 16 – Diagrama elétrico. 39
Figura 17 – Esquemático da placa de carregamento (P2). 40
Figura 18 – Esquemático da placa de comunicação (P3). 41
Figura 19 – Esquemático da placa-mãe (P4) com controlador. 42
Figura 20 – Esquemático da placa de *driver* dos motores de movimentação (P6). 43
Figura 21 – Esquemático da placa de *driver* dos motores de elevação da bandeja (P7). . 44
Figura 22 – Layout das bobinas 45
Figura 23 – Cenário espacial com âncora e *beacons*. 53
Figura 24 – Diagrama de fluxograma para a lógica do Controlador. 57
Figura 25 – Diagrama de fluxograma para a lógica do Comunicador. 58
Figura 26 – Modelo 3D da placa de alimentação. 59
Figura 27 – Dimensões da placa para alimentação. 59
Figura 28 – Layout de alimentação. 60
Figura 29 – Modelo 3D da placa para o comunicador. 60
Figura 30 – Dimensões da placa para o comunicador. 60
Figura 31 – Layout da placa para o comunicador. 61
Figura 32 – Modelo 3D da placa para o controlador. 61
Figura 33 – Dimensões da placa-mãe. 61
Figura 34 – Layout da placa-mãe. 62

Figura 35 – Modelo 3D da placa de controle dos motores das rodas. 62
Figura 36 – Dimensões da placa para movimento. 62
Figura 37 – Layout da placa de controle dos motores das rodas. 63
Figura 38 – Modelo 3D da placa de controle dos motores da fuso. 63
Figura 39 – Dimensões da placa para o fuso. 63
Figura 40 – Layout da placa de controle dos motores do fuso. 64
Figura 41 – Cabo para a antena, plug IPEX UFL à esquerda e SMA à direita. 65
Figura 42 – Região para plugue do cabo, próxima do chip do microcontrolador. 65
Figura 43 – Diagrama de blocos. 72
Figura 44 – Protótipo de baixa fidelidade. 75
Figura 45 – Telas do Protótipo de alta fidelidade. 76
Figura 46 – Protótipo de Alta fidelidade do aplicativo para o cliente. 76
Figura 47 – Protótipo de Alta fidelidade do aplicativo para o administrador do restaurante. 77
Figura 48 – Diagrama de casos de uso do sistema. 79
Figura 49 – Diagrama de Arquitetura Funcional. 79
Figura 50 – Diagrama de Classes. 81
Figura 51 – Diagrama de Atividades. 82
Figura 52 – Diagrama de Componentes. 82
Figura 53 – Diagrama de Pacotes. 83
Figura 54 – Diagrama de blocos. 84
Figura 55 – Organização da hierarquia 99
Figura 56 – Estrutura analítica do projeto. 101
Figura 57 – Estrutura Analítica de Riscos (EAR). Fonte: Autores. 106
Figura 58 – Componentes eletrônicos necessários. 114
Figura 59 – Componentes elétricos necessários. 114
Figura 60 – Fiação necessária para construção. 115
Figura 61 – Estimativa do valor da mão de obra. 115
Figura 62 – Motores de movimentação - para frente. 127
Figura 63 – Motores de movimentação - virando à direita. 127
Figura 64 – Motores de movimentação - virando à esquerda. 127
Figura 65 – Motores de movimentação - dando ré. 128
Figura 66 – Motores de movimentação - parados. 128
Figura 67 – Motor da bandeja - subindo. 129
Figura 68 – Motor da bandeja - descendo. 129
Figura 69 – Motor da bandeja - parado. 130
Figura 70 – Motor da tampa - abrindo. 130
Figura 71 – Motor da tampa - fechando. 131
Figura 72 – Calibração da célula de carga. 132

Figura 73 – Medição da célula de carga. 132
Figura 74 – Infravermelho. 133
Figura 75 – Sensor ultrassônico. 134
Figura 76 – Captura de tela do ambiente de simulação Ansoft Maxwell. 135
Figura 77 – Simulação da Unidade 1. 135
Figura 78 – Simulação da Unidade 2. 136
Figura 79 – Circuito de Simulação do Carregamento por Plug. 136
Figura 80 – Campos de temperatura ao longo do robô em diferentes períodos de tempo.137
Figura 81 – temperatura em diferentes pontos do robô ao longo do tempo. 138
Figura 82 – NFR referente a usabilidade. 141
Figura 83 – Vista em 3D da base de carregamento. 186
Figura 84 – Vista em 2D da base de carregamento. 187
Figura 85 – Detalhamento da bobina. 188

Lista de tabelas

Tabela 1 – Requisitos Gerais - Funcionais 19
Tabela 2 – Requisitos Gerais - Não-Funcionais 19
Tabela 3 – Lista É/Não é 20
Tabela 4 – Legislação e Normas relativas a contato com alimentos 24
Tabela 5 – Normas e leis que tangem robótica. 24
Tabela 6 – Legislação e Normas ISO para produtos de *software*. 25
Tabela 7 – Valores iniciais do dimensionamento do motor de fuso. 26
Tabela 8 – Especificações mínimas do motor dos fusos. 28
Tabela 9 – Valores iniciais do dimensionamento do motor de fuso. 29
Tabela 10 – Especificações mínimas do motor da tampa. 29
Tabela 11 – Parâmetros dimensionais da bobina. 47
Tabela 12 – Especificações da fiação. 67
Tabela 13 – Localizador 68
Tabela 14 – Sensor de Peso 68
Tabela 15 – Teste de baterias 68
Tabela 16 – Teste das Bobinas 69
Tabela 17 – Testes do sistema elétrico 69
Tabela 18 – Teste do Comunicador 70
Tabela 19 – Heurísticas de Nielsen e suas aplicações no *software*. 74
Tabela 20 – Casos de Uso. 78
Tabela 21 – Cronograma geral. 85
Tabela 22 – Cronograma de atividades. 85
Tabela 23 – Cronograma descritivo para atividades especificas da automação 88
Tabela 24 – Cronograma da automação. 89
Tabela 25 – Atividades de Estruturas 89
Tabela 26 – *Sprints* concluídas em relação a cada ponto de controle. 90
Tabela 27 – Atividades de Software 91
Tabela 28 – Membros e atribuições 98
Tabela 29 – Ferramentas de gerenciamento da equipe 100
Tabela 30 – Lista de concorrentes. 103
Tabela 31 – Riscos técnicos 107
Tabela 32 – Riscos de gerenciamento 108
Tabela 33 – Riscos comerciais 108
Tabela 34 – Riscos externos 109
Tabela 35 – Medidas de contingência 110
Tabela 36 – Tabela de Componentes Estimadas 116

Tabela 37 – Custos Preliminares dos Componentes Estáticos. 118
Tabela 38 – Custos Preliminares dos Componentes Móveis. 118
Tabela 39 – Estimativa Total dos Custos 118
Tabela 40 – Média de custo/hora de cada nível de programador. 119
Tabela 41 – Custos de horas. 119
Tabela 42 – Custos de software. 121
Tabela 43 – Orçamento geral. 122
Tabela 44 – Matriz de decisão para o sistema de localização do robô. 124
Tabela 45 – Parâmetros técnicos para a escolha do controlador. 125
Tabela 46 – Matriz de decisão para o controlador do robô. 125
Tabela 47 – Matriz de decisão sobre sistema de carregamento 125
Tabela 48 – Estimativa de consumo energético. 139
Tabela 49 – Objetivos de Negócio 140
Tabela 50 – Objetivos de Produto 140

1 Introdução

Interação: definida como ação recíproca entre usuário e equipamento [Michaelis 1998]. Automação: o equipamento tomar a decisão e definir os melhores caminhos para realizar um serviço ou trabalho, como a navegação autônoma, deve ser realizado tratamento e leitura a cada necessidade [Figueiredo 1999].

Embora filmes desenvolvam um robô como uma máquina capaz de movimentar, decidir, pensar e, em estágios mais avançados, sentir, a robótica traz como uma máquina capaz de realizar percepções do mundo físico por meio de dispositivos controlados por um computador [PEREIRA 2013]. Os robôs podem ser classificados de acordo com sua anatomia, funcionalidade e tipo de controle. Em sua anatomia, podem ser divididos entre aquático, aéreo e terrestre [Cortelletti 2006].

Quanto a funcionalidades, podem ser: pessoal, de serviço, de campo ou industrial. Pessoais interagem com pessoas na forma de entretenimento ou ajuda doméstica. De serviço são responsáveis por realizar tarefas em ambientes internos ou externos que necessitam processar informações e atuar em momentos específicos. O robô de campo trabalha em ambientes pouco conhecidos ou de acesso que apresenta risco para o homem, como robôs de limpezas de acidentes químicos ou de desarme de bombas. Os robôs industriais, então, são aqueles que realizam trabalho repetitivo com precisão, porém baixo custo de planejamento [Pscheidt 2007].

Por fim, tem-se a classificação por tipo de controles: tele-operados, aqueles que são operados por humanos à distância; semi-autônomos que são operados, mas realizam alguns procedimentos sozinhos; e autônomos, aqueles programados para realizar tarefas de forma independente [Lora, Hemerly e Lages 1998].

Atrelado às tecnologias descritas, o ser humano apresenta a cada passo de seu progresso como sociedade uma exigência maior quanto ao atendimento. A solicitação e prestação de serviço exige velocidade, pouco tempo é destinado a cada atividade que precisa ser realizada. Além disso, o processo deve ser preciso e correto, de maneira que não gere mais problemas a serem resolvidos [Costa, Santana e Trigo 2015]. Atualmente, atrair o cliente para uma escolha de produtos e preços é uma tarefa difícil [KOTLER Philip e ARMSTRONG 2003].

Essas foram as diretrizes que guiaram o surgimento do robô garçom aqui descrito. Uma máquina capaz de atrair o cliente e realizar um serviço de forma autônoma - mais higiênica, eficaz, eficiente e veloz. Para explicar o funcionamento e o que compõe tal robô, este relatório técnico apresenta:

- Um termo de abertura, onde são apresentadas as justificativas e objetivos do projeto;

- Um descritivo do produto, de maneira a englobar todas as áreas integradas do projeto e suas decisões iniciais;
- Estipulação do orçamento a ser adotado;
- Gerenciamento de riscos;
- Uma análise de mercado, de maneira a apresentar os concorrentes e o que o produto proposto traz de diferente;
- Os requisitos necessários;
- por fim, cronogramas de projeto por área.

2 Termo de Abertura do Projeto

2.1 Dados do projeto

Nome do Projeto: Robô Garçom

Data de abertura: 30/07/2021

Código: 21-RG

Patrocinador: Universidade de Brasília

Coordenador geral: Damarcones Porto – 15/0122187

Diretora de Qualidade: Letícia Karla – 15/0135939

Líder de Automação: Gabriel Pinheiro – 13/0142069

Líder de *Software*: Bruna Almeida Santos – 17/0100626

Líder de Estrutura: Luiz Alexandre – 15/0041390

Os dados completos de membros, atribuições e posição hierárquica na equipe podem ser encontrados no Apêndice A, assim como as ferramentas utilizadas e a avaliação de cada membro.

2.2 Objetivos

2.2.1 Objetivo geral

O projeto Robô Garçom visa automatizar o atendimento aos clientes de restaurantes com menos contato humano para oferecer maior autonomia e segurança.

2.2.2 Objetivos específicos

Na visão de atendimento, os objetivos são proteger a refeição em restaurantes antes de ser servida e diminuir o número de pessoas que entram em contato direto com o prato. No sentido de projeto, os objetivos são elaborar e projetar um robô capaz de transportar comida entre a cozinha de um restaurante e as diversas mesas em seu interior.

2.3 Justificativa

A quarta revolução industrial, como é chamado o rápido advento de tecnologias inteligentes, afetará todos os setores no futuro. Prevê-se a integração de sistemas ciber-físicos associados a robôs em todas as camadas da sociedade. A tecnologia, mais do que antes, determinará como as pessoas se relacionam, vivem e trabalham, conforme a utilização das máquinas [BBC 2016]. Automatizar o atendimento, ou parte dele, com auxílio de robôs pode garantir à empresa um uso mais eficiente dos seus recursos humanos e financeiros para que seus produtos e serviços sejam mais competitivos.

O Robô garçom pode ser usado como propaganda para os restaurantes, um atrativo para clientes em seu estilo e por demonstrar preocupação em se adequar às novas formas de cuidados sanitários para reduzir a transmissão de vírus e bactérias pelos alimentos. A estrutura do robô garçom garante uma camada a mais de proteção à refeição dos clientes, pois há um compartimento reservado fechado onde fica a comida durante o transporte. Além disso, o número de pessoas em circulação no ambiente seria menor.

A automação do Robô Garçom mostra-se um diferencial de atendimento, por gerar mais conforto ao cliente, que não precisará chamar por um garçom ou esperar longos períodos para receber atenção, e poderá fazer seu pedido com mais calma a qualquer momento. O funcionamento do restaurante também pode ser otimizado ao remanejar os garçons para outros cargos de menos esforço físico e maiores atividades intelectuais.

2.4 Problemática de automação

A problemática do ponto de vista da automação está em propor uma solução que atenda aos requisitos de forma a construir um robô eficiente, que garanta a integridade dos alimentos transportados por ele e segurança aos clientes presentes no restaurante. Esses requisitos podem ser cumpridos utilizando diferentes tipos de sensores.

O Robô Garçom traz em si desafios para a automação, pois apesar de existirem algumas opções disponíveis no mercado, há diferentes formas possíveis de implementação, como por exemplo robôs seguidores de linha, robôs que utilizam inteligência artificial e *machine learning*.

A solução proposta para o Robô Garçom é baseada em rastreamento de posicionamento local [H. S. Hassan and M. Hussein and S. M. and Saad and M. A. Dzahir 2018], buscando a integração de diferentes sensores, como o de obstáculo (sensor reflexivo infravermelho), célula de carga para detecção dos pratos em cima da bandeja com base no peso deles, acelerômetro para inclinação do robô.

3 Projeto Conceitual do Produto

3.1 Descrição funcional do produto

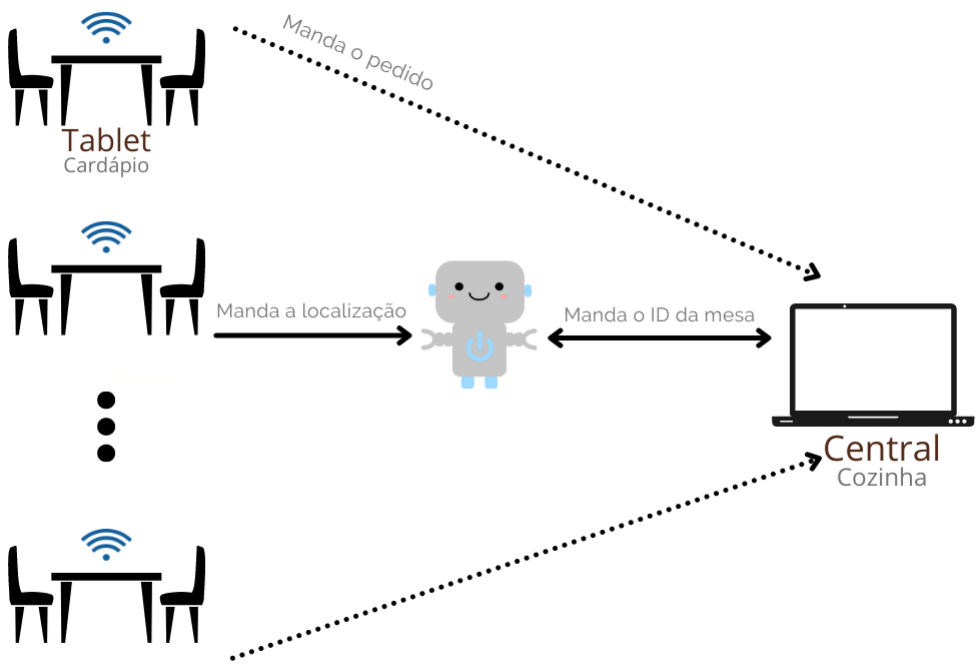
Esta seção descreve as funcionalidades do produto de forma simplificada. No salão do estabelecimento, para cada mesa haverá um *tablet*. Esse dispositivo é responsável por exibir um aplicativo que deverá mostrar o cardápio aos clientes e recolher seus pedidos. Estes serão enviados para o computador central do restaurante. Também será feito o pagamento do pedido pelo aplicativo.

Com os alimentos prontos, o funcionário deverá inseri-los dentro do compartimento do Robô Garçom. A partir daí, será dado o comando de entrega para a mesa do pedido, informação essa dada pelo computador central. O robô receberá o ID de rastreio da localização da mesa destino através do sistema central, localizado na cozinha.

Ao chegar no destino, seu compartimento é aberto e a comida é servida aos clientes. Após a balança interna do compartimento detectar que está vazia, o robô voltará à copa/cozinha. Ele fará quantas voltas forem necessárias para servir todos os pedidos.

A Figura 1 ilustra o funcionamento do sistema a ser implementado.

Figura 1 – Visão geral do funcionamento do produto.



Fonte: Autores.

3.2 Características gerais

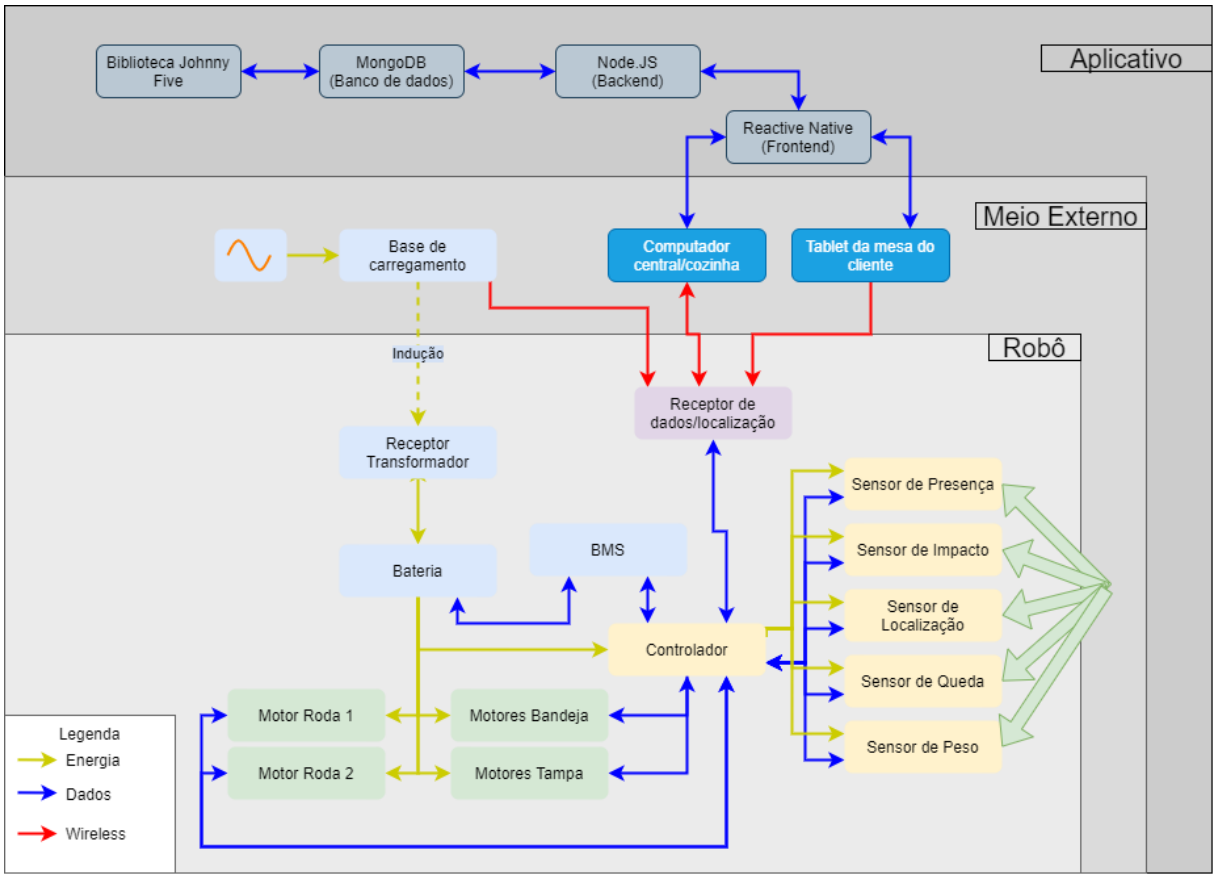
A Estrutura Analítica de Projeto (EAP) é uma subdivisão hierárquica de projeto. Para o projeto Robô Garçom, a EAP foi dividida em *Software*, Estrutura, Automação e Gerência, e está disponível para visualização no Apêndice B. A EAP garante aos gerentes dos projetos a visibilidade das principais entregas, facilitando o controle de tempo e de custo.

O Gerenciamento de Escopo do Projeto, descrito no Guia PMBOK® (*Project Management Body of Knowledge*), define a EAP como as boas práticas utilizadas pelo mercado. Existem quatro estratégias para montar a EAP: por fases, por entregas, por subprojeto e híbrida [PMI - PROJECT MANAGEMENT INSTITUTE 2017]. A estratégia híbrida se mostrou mais adequada às necessidades do presente projeto.

3.3 Integração de subsistemas

A interação e integração dos sistemas de automação entre eles e o meio externo, o que inclui a estrutura do robô e o aplicativo de atendimento, é constituído de maneira a obedecer a Figura 2, na qual apresentam as interações realizadas.

Figura 2 – Integração de arquiteturas.



Fonte: Autores.

São representadas na área de energia, o caminho percorrido e as integrações com outros sistemas, como o controlador e sensores, além dos motores. Estes são controlados por meio do caminho de comando e controle, do qual reúne os dados para o envio para o receptor, e toma as decisões baseadas nas leituras realizadas pelos sensores. Além dos sensores, temos o BMS, do qual o controlador recebe as informações e as envia também para o receptor.

Em termos de funcionamento, o movimento do robô é feito pela recepção dos dados no controlador, do qual computa a sua direção. Baseado neste fator, são enviadas para as rodas 1 e 2 iniciarem e realizarem os movimentos. Ao chegar no seu destino, são acionados os motores da tampa, e posteriormente o dos atuadores. Realizada a entrega, o robô retorna para a base de carregamento, na qual espera para o próximo serviço, e realiza a sua recarga por indução.

Em relação ao *software*, os *tablets* da cozinha e da mesa do cliente, que se comunicam via *wireless* com os receptores de dados e localização, recebem informações através do *frontend*, e as mandam para o *backend*, para que este armazene no banco de dados. As informações sobre localização são acessadas pela biblioteca Johnny Five, que enviará ao robô o cálculo da trajetória até a mesa do pedido.

4 Requisitos do sistema

4.1 Requisitos Gerais

Os requisitos gerais foram levantados a partir da compreensão do projeto e do entendimento do grupo sobre os objetivos a serem atendidos, mostrados em uma lista "É/não é"compilada de demandas quanto ao produto.

Tabela 1 – Requisitos Gerais - Funcionais

ID	Descrição
RF001	Identificar ID da mesa
RF002	Compartimento de carga higiênico
RF003	Levar carga entre dois pontos
RF004	Autonomia da bateria
RF005	Facilidade na recarga de energia com o menor contato humano possível
RF006	Capacidade de aguentar as cargas submetidas aos motores
RF007	Identificação de obstáculos

Fonte: Autores.

Tabela 2 – Requisitos Gerais - Não-Funcionais

ID	Descrição
RNF001	Prover segurança para pagamento
RNF002	Guiar robô até a mesa
RNF003	Prover segurança para pagamento
RNF004	Atender normas de segurança alimentar
RNF005	Cliente não deve tirar o pedido de dentro de um buraco
RNF006	Esteticamente agradável
RNF007	Tempo de viagem menor ou igual ao de um garçom humano
RNF008	Transportar comidas quentes ou geladas
RNF009	Entregar o pedido em altura confortável para clientes sentados à mesa
RNF010	Fácil instalação da base de carregamento
RNF011	Fácil troca de componentes danificados

Fonte: Autores.

Serão detalhados os requisitos específicos para cada área do projeto como definido no Capítulo 2. O desenvolvimento de cada área para realização do projeto será detalhada no Capítulo 3.

Tabela 3 – Lista É/Não é

É	**Não é**
Um equipamento para transporte e armazenamento de comida dentro de um restaurante.	Um equipamento para transporte e armazenamento de comida na rua.
Um equipamento que receberá pedidos por meio da troca de informações com um *tablet* utilizado pelos clientes.	Um equipamento que receberá pedidos dos clientes por comando de voz.
Um equipamento que se posiciona na base de carregamento da bateria.	Um equipamento que necessita da conexão de cabos para carregamento da bateria.

Fonte: Autores.

4.2 Requisitos de *Software*

Os requisitos do sistema de *software* foram levantados através de duas técnicas complementares: *brainstorm* e introspecção. A técnica de *brainstorm* – tempestade de ideias – consiste em uma dinâmica em grupo, onde há troca de ideias, para unir informação e estimular o pensamento criativo para resolução de problemas. Durante a dinâmica, todos os integrantes da equipe expuseram sua visão do sistema e o que seria crucial para o *software*.

Posteriormente, foi aplicada a técnica de introspecção, que se baseia em imaginar que tipo de sistema eu iria querer se fosse utilizar o sistema em questão. Foi pedido que cada integrante, individualmente, se imaginasse como usuário do sistema para o Robô Garçom, e que listasse os requisitos achasse essenciais e viáveis para o *software*.

Através das duas técnicas de elicitação empregadas, foram levantados os seguintes requisitos funcionais e não funcionais do sistema, descritos a seguir.

Funcionais

- O usuário poderá acessar o aplicativo sem ter *login*.
- O usuário poderá visualizar o cardápio do restaurante.
- O usuário poderá visualizar detalhes sobre um item do menu.
- O usuário poderá adicionar um item do menu no carrinho do seu pedido.
- O sistema deverá ser capaz de mostrar a lista de itens que o usuário adicionou ao seu pedido.
- O usuário poderá adicionar observações ao pedido.
- O usuário poderá adicionar ingredientes extras a um item do seu pedido.

- O usuário poderá excluir um item no carrinho do seu pedido.
- O sistema deverá mostrar o valor total do pedido.
- O aplicativo deverá disponibilizar um espaço para o usuário digitar códigos de promoção.
- O sistema deverá mostrar o tempo previsto de espera para o pedido.
- O usuário poderá cancelar o pedido antes do pagamento.
- O aplicativo deverá mostrar as formas de pagamento disponíveis.
- O usuário poderá escolher a forma de pagamento do pedido.
- O sistema deverá processar o pagamento feito pelo usuário.
- O aplicativo deve permitir fechar pedidos seguidos.
- O aplicativo deve enviar o comprovante de pagamento para o usuário.

Não-Funcionais

- O sistema deve ter segurança para os pagamento via cartão.
- O sistema deve ter segurança para os dados dos clientes.
- A API deve ser feita via microsserviços.
- O aplicativo deve se comunicar com o Robô Garçom.
- O aplicativo deve se conectar à internet.
- O aplicativo deve ser aprovado em testes.
- O usuário poderá navegar pelos menus.
- O App deverá ser compatível com normas de acessibilidade e compatibilidade, com solução/assistência/ajuda para debilidades.
- O App deverá estar de acordo com a LGPD e melhores práticas de segurança.
- O sistema deverá ser capaz de receber pagamentos através dos serviços Mercado Pago e Pix.

4.3 Requisitos de Automação

Será necessária a utilização de motores elétricos para as rodas, para a tampa e para os atuadores do robô, responsáveis pelas atividades básicas de movimentação previstas no projeto. Tais motores serão ligados por um sistema de baterias após a transformação de corrente alternada para corrente contínua no carregamento. E a alimentação prevista é por indução, forma que possibilita uma menor interação homem-máquina.

Sensores de rastreamento serão utilizados para que o sistema consiga identificar os pontos de entrega, e outros sensores serão empregados para garantir que o robô se movimente sem esbarrar em obstáculos e cair. Também serão utilizados outros componentes para coleta e processamento de dados.

Quanto ao controlador do robô, é preciso que este seja capaz de armazenar todo o *firmware* do robô, além de serem necessários circuitos de interface entre os sensores e o controlador e circuitos para o acionamento dos motores. É preciso também que o controlador tenha conexão com o WI-FI para comunicação com o computador central, recebendo informações acerca do pedido e da mesa.

Funcionais

- Pontos de energia nas mesas, para carregamento do *tablet*.
- Autonomia da bateria.
- Facilidade na recarga de energia com o menor contato humano possível.
- Capacidade de aguentar as cargas submetidas aos motores (estrutura + pratos de entrega).
- Identificação de obstáculos.
- Sensores devem ter boa imunidade a ruídos.
- *Tablets* e o robô devem ter acesso à WI-FI.

Não-Funcionais

- Fácil instalação da base de carregamento.
- Fácil troca de componentes danificados.

4.4 Requisitos de Estrutura

Os requisitos de Estrutura foram levantados em vista dos requisitos gerais do projeto (Tabela 1 e Tabela 2), sob a ótica da estrutura física que sustentará o Robô Garçom e os subsistemas acoplados a ele. Como partes da estrutura entram em contato com os pedidos de comida preparada, deve haver cuidado especial com a higiene dessas partes. Além disso, como o robô interage com clientes do restaurante, sua apresentação é importante e seu serviço deve apresentar o mínimo de erros possível. Com esses pontos em mente, a equipe levantou os seguintes requisitos específicos:

Funcionais

- Levar carga entre dois pontos.
- Compartimento de carga higiênico.
- Compartimento de carga capaz de levar pedidos típicos de um restaurante.
- Poder passar por espaços onde um ser humano pode passar.
- Não tombar sob funcionamento normal.
- Compartimento de carga higienizável.
- Sustentar a carga assim como os compartimentos internos.

Não-Funcionais

- Deve atender normas de higiene e segurança alimentar.
- Deve transportar copos sem derramar.
- Deve ser esteticamente agradável.
- Cliente não deve ter contato com o compartimento de carga.
- Deve transportar comidas quentes ou geladas.
- Deve entregar o pedido em altura confortável para clientes sentados à mesa.
- Deve desmontar facilmente e sem ferramentas especiais.
- Compartimento de carga deve ser removível.
- Deve manter temperatura dos pedidos transportados.
- Deve haver controle de condensação para não molhar as comidas.

- Não pode cair com impactos externos.
- Deve ser capaz de superar pequenos desníveis no piso.

4.5 Legislação e Normas

As normas e leis que tangem o Robô Garçom, descritas nas Tabela 4, Tabela 5 e Tabela 6, mostram falta de regulamentação e descrição em normas do processo de construção deste tipo de produto, a maioria das leis e normas encontradas referem-se a produtos eletrônicos e instalações elétricas gerais.

Tabela 4 – Legislação e Normas relativas a contato com alimentos

Legislação/Norma	Descrição
Resolução 91/2001	Regulamento Técnico - Critérios Gerais e Classificação de Materiais para Embalagens e Equipamentos em Contato com Alimentos.
Resolução 216/2004	Dispõe sobre Regulamento Técnico de Boas Práticas para Serviços de Alimentação.
Resolução 20/2007	Regulamento Técnico sobre Disposições para Embalagens, Revestimentos, Utensílios, Tampas e Equipamentos Metálicos em Contato com Alimentos.

Fonte: Autores.

O componente legal específico para o produto é o projeto de lei 240/2020 que ainda está em processo de construção, o qual apensou outros três projetos análogos e está em fase de aprovação de parecer das comissões de justiça, trabalho e a de ciência e tecnologia, para em seguida ser votado em plenário.

Tabela 5 – Normas e leis que tangem robótica.

Legislação/Norma	**Descrição**
NBR ISO 14621-1:2013	Sistemas espaciais - Componentes elétricos, eletrônicos e eletromecânicos.
NBR 5410:2004	Instalações elétricas de baixa tensão.
IPC-A-610	Padrões de aceitabilidade mundial de montagem de produtos eletrônicos.
Projeto de lei 240/2020*	Estabelece princípios, direitos e deveres para o uso de inteligência artificial no Brasil, e dá outras providências.

Legenda: Em processo de votação.*

Tabela 6 – Legislação e Normas ISO para produtos de *software*.

Legislação/Norma	**Descrição**
LGPD	Lei Geral de Proteção de Dados, Lei nº 13.709/2018. Legislação brasileira que regula as atividades de tratamento de dados pessoais.
ISO/IEC 25010:2011	Norma ISO para qualidade de produto de *software*, Substituiu a norma ISO/IEC 9126, da qual adicionou às características principais "segurança"e "compatibilidade".
ISO/IEC 25000:2014	Norma ISO com o objetivo de criar uma estrutura para a avaliação da qualidade de produtos de *software*.
ISO 27001	Norma internacional para segurança de informação.

Fonte:Autores.

4.6 Indicadores

- De acordo com o [Flyer 2020], 60% da população adulta brasileira possui um *Smartphone* e, desse grupo, 75% das pessoas fazem ou já fizeram compras por aplicativo.
- De acordo com o IBGE, 75,4% das pessoas que não usam aplicativos alegam não saber usar ou falta de interesse.
- De acordo com a empresa de *delivery* IFood, a facilidade de usar um aplicativo de comida os levou para um pico de mais de 100 milhões de pedidos e as refeições mais pedidas são: Hambúrguer, Esfirra, Carne, Sanduíche, Wrap, Sushi e Sashimi [mensagem 2016].
- De acordo com a empresa de robôs *Service Robots*, os robôs de serviço aumentam as vendas, fazem a empresa se destacar e melhoram o marketing [servicerobots 2020].
- Uma pesquisa realizada pela empresa de plataformas tecnológicas *Schalter* indica que um totem consegue atender até quatro vezes mais clientes que um caixa comum [Mercado & consumo 2018].
- O tamanho do mercado global de sensores foi avaliado em $ 166,69 bilhões em 2019 e está projetado para chegar a $ 345,77 bilhões em 2028, para registrar um CAGR de 8,9% de 2021 a 2028 [Research 2021].
- Segundo estudo de hábitos alimentares, os brasileiros estão entre os que mais gastam com *fastfood*, atrás apenas dos Estados Unidos, Japão e China [EAE business school 2014].

5 Estrutura

O projeto da estrutura do robô estava próximo de completo durante a entrega do relatório anterior. Entretanto, o dimensionamento de algumas partes ainda estavam em aberto, e as seção 5.1 a seção 5.4 finalizam o dimensionamento dessas partes, finalizando assim o projeto técnico.

Além do projeto dos mecanismos e do dimensionamento das partes, também foi feito o projeto de fabricação das peças projetadas, e de montagem do sistema, que podem ser encontrados em documento anexo.

5.1 Dimensionamento do motor do fuso

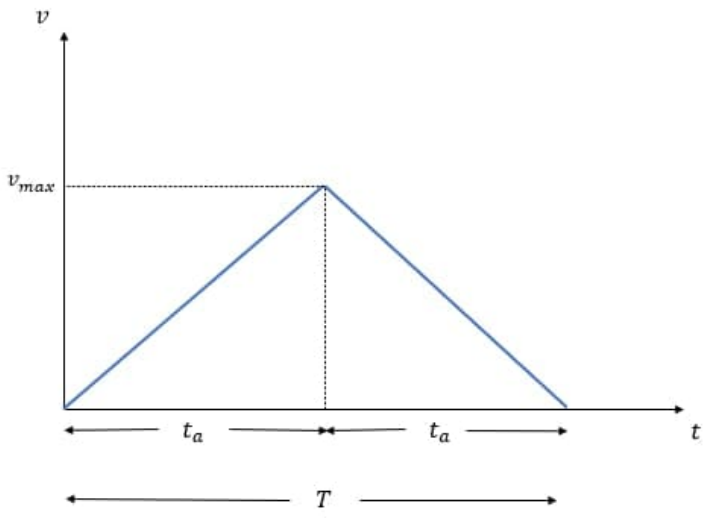
Para o sistema de levantamento com fuso de esferas, é necessário escolher um motor adequado para o uso desejado. No caso do robô garçom, os critérios de escolha foram baseados nas dimensões do curso útil, na carga levantada, no fuso escolhido, e na velocidade desejada. Dessa forma, desejava-se encontrar um par de motores capaz de atender os requisitos dispostos na Tabela 7, com algumas características do sistema necessárias para especificar o motor. A partir desses valores, desejamos determinar a velocidade de rotação N (RPM) e o torque T mínimos para operar o sistema.

Tabela 7 – Valores iniciais do dimensionamento do motor de fuso.

Símbolo	Valor	Descrição
L	500 mm	Curso útil
M	12 kg	Massa transportada
ℓ	5 mm	Passo do fuso
d	12 mm	Diâmetro do fuso
T	6 s	Tempo total de viagem
t_a	3 s	Tempo de aceleração
η	0,8	Fator de eficiência de conversão de movimento
μ	0,003	Coeficiente de atrito

Para determinar a velocidade de rotação mínima do motor, primeiro calculamos a velocidade linear necessária, e então realizamos a conversão de acordo com a Equação 5.2. Como carregamos comidas e bebidas não rigidamente fixados à plataforma de levantamento, acelerações bruscas são indesejadas no sistema; foi escolhido, então, um perfil de aceleração mínima, conforme a Figura 3 e, por consequência, o tempo de aceleração $t_a = 3\,\mathrm{s}$ é metade do tempo de viagem desejado, e podemos calcular a velocidade linear a partir da Equação 5.1.

Figura 3 – Gráfico de velocidade por tempo desejado durante o levantamento.



$$v_{max} = \frac{L}{t_a} \tag{5.1}$$

$$N = 60\frac{v_{max}}{\ell} \tag{5.2}$$

Assim, o motor deve ter velocidade nominal igual ou superior a $N = 2000\,\text{RPM}$. Este valor já havia sido estimado durante o dimensionamento do fuso e, portanto, condiz com a velocidade máxima do nosso fuso.

Para o cálculo de torque necessário, precisamos fazer uma análise um pouco mais cuidadosa. A carga no motor é composta pelo peso da massa levantada M, pelas forças de atrito, pela inércia do sistema, e pelo pré-carregamento das castanhas. No caso do nosso sistema vertical, não é necessário pré-carregamento, e as cargas puramente axiais permitem simplificar o atrito do sistema como apenas o atrito das castanhas. A força axial é simplesmente $P = \frac{Mg}{2}$, e o torque equivalente pode ser calculado de acordo com a Equação 5.3, igual a $T_p = 5{,}8\,\text{N}\,\text{cm}\}$.

$$T_p = \frac{P\ell}{2\pi\eta} \tag{5.3}$$

Quanto à resistência inercial do sistema, ela é a soma da inércia dos fusos, acoplamentos, rolamentos, da massa sendo deslocada, e do próprio motor. O torque relacionado à inércia pode ser calculado como o produto da inércia do sistema pela aceleração angular, isto é, $T_J = J\alpha$. Para calcular os termos da inércia, foram utilizadas as Equações 5.4 a 5.6, onde m é a massa do fuso. O acoplamento, o motor e os rolamentos têm inércia muito menor que o fuso e a massa e foram desconsiderados.

$$J = J_{fuso} + J_{massa} \tag{5.4}$$

$$J_{fuso} = \frac{md^2}{8} \tag{5.5}$$

$$J_{massa} = \frac{1}{2}\frac{M}{4}\left(\frac{\ell}{\pi}\right)^2 \tag{5.6}$$

Assim, o valor total de inércia do sistema é $J = 1{,}7 \times 10^{-5}\,\mathrm{kg\,m^2}$. A aceleração angular α, em $\mathrm{rad/s^2}$, pode ser calculada a partir da Figura 3, de acordo com a Equação 5.7, e é igual a $70\,\mathrm{rad/s^2}$. Assim, o torque relativo à inércia do sistema é $T_J = 0{,}12\,\mathrm{N\,cm}$.

$$\alpha = \frac{v_{max}}{t_a}\frac{2\pi}{\ell} \tag{5.7}$$

O torque relativo ao atrito do sistema é calculado conforme a Equação 5.8. Considera-se que o atrito dos mancais e dos rolamentos lineares é muito menor que o atrito das castanhas, devido à falta ausência de cargas radiais. Assim, o torque relativo ao atrito do sistema é $T_F = 0{,}10\,\mathrm{N\,cm}$

$$T_F = \frac{d}{2}P\mu \tag{5.8}$$

O torque total necessário, então, é $T = 6{,}0\,\mathrm{N\,cm}$. Para operar com margem de segurança, escolheu-se especificar um motor com torque mínimo de $11\,\mathrm{N\,cm}$. Assim, as especificações foram resumidas na Tabela 8 e o motor escolhido de acordo com as especificações mecânicas e de eletrônica foi o modelo *82 800 036 Cruzet 12V 2,5A*, listado na Figura 59.

Tabela 8 – Especificações mínimas do motor dos fusos.

Torque nominal	11 N cm
Velocidade	2000 RPM

5.2 Dimensionamento do motor da tampa

Para o mecanismo de operação da tampa, é necessário escolher um motor adequado para o uso desejado. Os critérios de escolha foram baseados nas dimensões do curso útil, na carga levantada, e na velocidade desejada. Dessa forma, desejava-se encontrar um par de motores capaz de atender os requisitos dispostos na Tabela 9, com algumas características do sistema necessárias para especificar o motor. A partir desses valores, desejamos determinar a velocidade de rotação N (RPM) e o torque T mínimos para operar o sistema. O raio da polia que move a correia pode ser adaptado, desde que seja na ordem de 1 a 3 cm, e será chamado de R, de modo que o motor pode ser especificado em função do raio, escolhido,

e então a polia adaptada ao motor. Assim, podemos também especificar a potência mínima do motor, que é independente do raio da polia.

Tabela 9 – Valores iniciais do dimensionamento do motor de fuso.

Símbolo	Valor	Descrição
L	500 mm	Curso útil
M	0,7 kg	Massa da tampa
T	4 s	Tempo total de viagem
t_a	2 s	Tempo de aceleração

Para o cálculo da velocidade de rotação, foi escolhido o mesmo perfil de aceleração do mecanismo de levantamento, de acordo com a Figura 3. A velocidade máxima linear então é $v_{max} = L/t_a$, e a velocidade do motor é de $N = 60v_{max}/(2\pi R) = (250\,\text{RPM}\,\text{cm})/R$.

O torque nominal deve ser calculado de acordo com o maior carregamento. No nosso mecanismo de levantamento, esse caso é o equivalente a levantar a tampa da posição vertical, onde todo o peso é sustentado pelo motor. Assim, as forças de atrito podem ser desconsideradas, pois não há força normal entre as superfícies. Quanto à inércia do sistema, podemos assumir com segurança que a polia e a correia têm inércia muito menor que a tampa, e considerar apenas a aceleração da tampa. Dessa forma, o torque total do sistema é descrito pela Equação 5.9.

$$T = \frac{M(g+a)}{2}R \tag{5.9}$$

O torque necessário, em função do raio da polia, é $T = (3{,}5\,\text{N})R$.

A potência do motor pode ser calculada a partir da velocidade de rotação e do torque, segundo a Equação 5.10. O raio da polia é eliminado da equação, e a potência é $P = 1\,\text{W}$.

$$P = TN\frac{2\pi}{60} \tag{5.10}$$

As especificações finais do motor, então, podem ser dadas a partir de torque e velocidade mínimos (calculados para o melhor tamanho de polia, 1 cm e 3 cm respectivamente) e a potência mínima. Assim, as especificações são resumidas na Tabela 10. Assim, as especificações foram resumidas na Tabela 10 e o motor escolhido de acordo com as especificações mecânicas e de eletrônica foi o modelo *ES08MAII EMAX 5V 0,2A*, listado na Figura 59.

Tabela 10 – Especificações mínimas do motor da tampa.

Potência nominal	1 W
Torque nominal	3,5 N cm
Velocidade	85 RPM

5.3 Dimensionamento da mesa de levantamento

A mesa de levantamento é fixada às castanhas dos fusos de esferas e aos rolamentos das guias lineares. Ela deve ser capaz de sustentar a carga de projeto de 10 kg. Para determinar a espessura adequada, foi realizada uma simulação com método de elementos finitos e aplicado o critério de von-Mises para um fator de segurança desejado de 3.

O modelo foi projetado com dupla simetria para simplificar a simulação, de acordo com a Figura 4. Os furos de parafuso foram considerados engastados, por estarem presos a uma peça fixa, conforme a Figura 5, e uma carga de 2,5 kg (1/4 do peso total, devido à simetria) foi aplicada no centro da placa, conforme mostrado na Figura 6. Na prática, a carga está mais distribuída que um ponto central, mas como a distribuição não é conhecida, foi escolhido o pior caso (concentração distante dos pontos de apoio).

Figura 4 – Aplicação de simetria na simulação da mesa de levantamento.

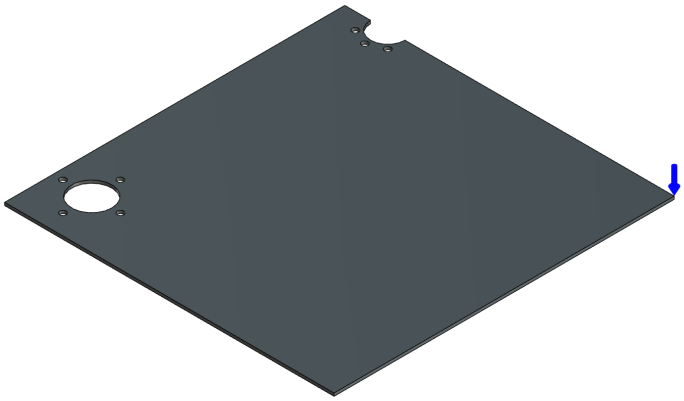


Figura 5 – Aplicação das condições de apoio na simulação da mesa de levantamento

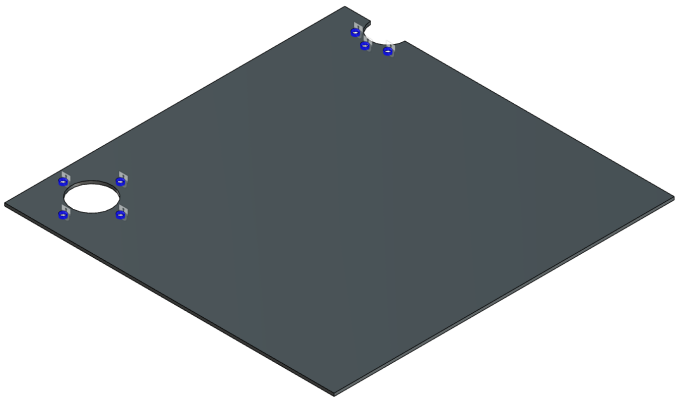
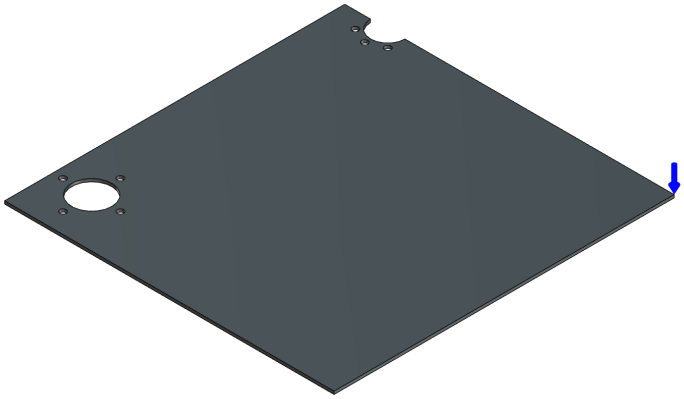


Figura 6 – Aplicação da carga na simulação da mesa de levantamento.



A Figura 7 mostra os resultados preliminares para uma mesa de alumínio 6061 com 2 mm de espessura, e parece indicar subdimensionamento do projeto. Entretanto, análise mais cuidadosa da simulação permite mostrar que as tensões máximas são geradas por singularidades numéricas. De fato, a Figura 8 mostra que os elementos finitos que não atendem nosso critério de segurança são muito pequenos, e de fato existem apenas em razão de singularidades numéricas. Portanto, podemos dizer que a placa atende os requisitos de projeto.

Figura 7 – Distribuição de fator de segurança (critério de von-Mises) na mesa de levantamento

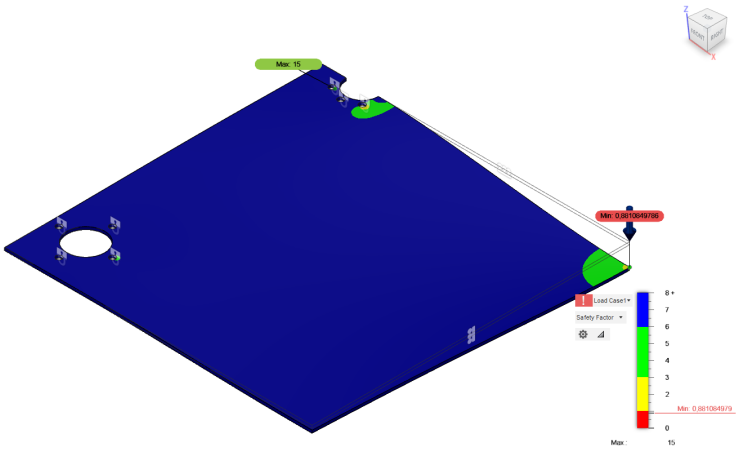
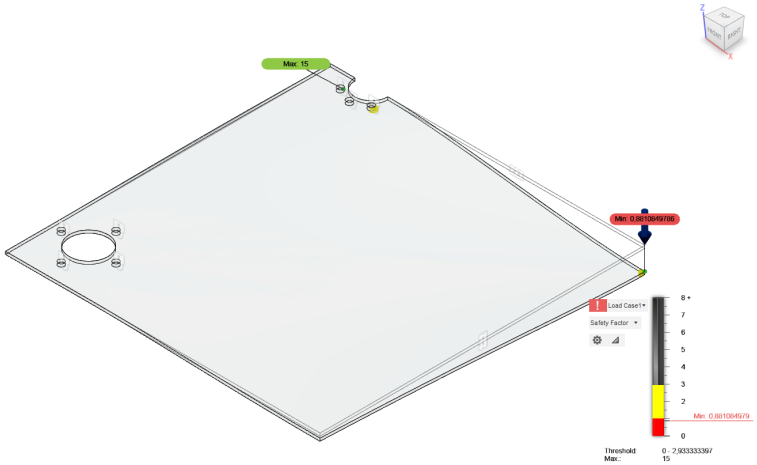


Figura 8 – Distribuição de fator de segurança (critério de von-Mises) abaixo de 3 na mesa de levantamento

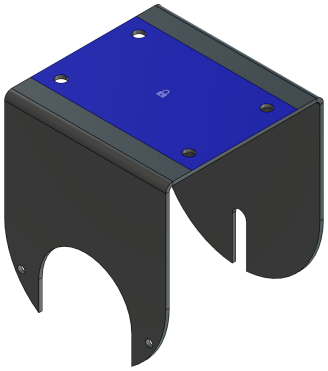


5.4 Dimensionamento do garfo da roda

O garfo da roda motorizada é fixo à placa de suporte acima dele e apoiado nos eixos da roda motorizada. Ele é projetado para suportar o peso do robô, distribuído entre as 6 rodas. Entretanto, como a distribuição de peso não é uniforme, seu dimensionamento é feito para o peso dividido em apenas 3 rodas, que é o pior caso possível. Dessa forma, ele deve ser capaz de sustentar um peso de 70 N. Foi utilizado o critério de von-Mises com fator de segurança desejado de 3.

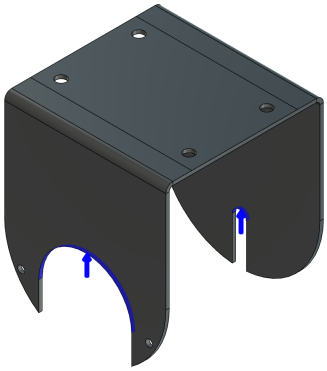
Para a simulação, foi aplicada uma condição de apoio fixo à superfície em contato com o suporte, conforme a Figura 9. às superfícies em contato com a roda, foi aplicada uma força vertical, conforme Figura 10.

Figura 9 – Apoio engastado na simulação do garfo.



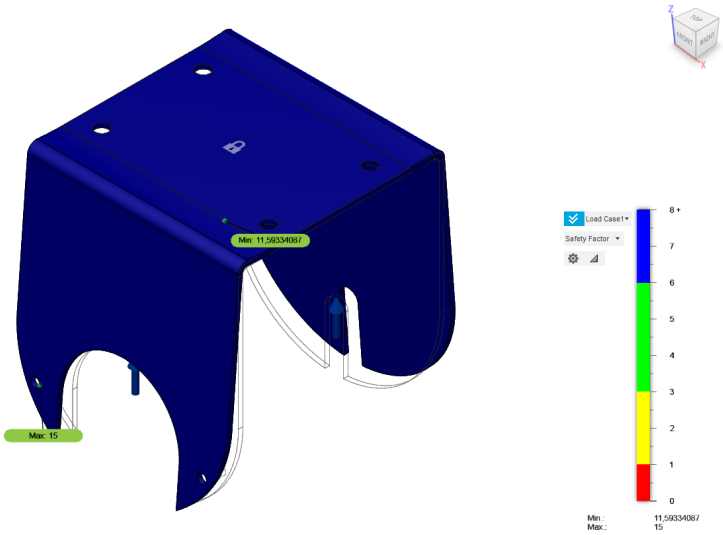
A Figura 11 mostra os resultados preliminares para garfo feito de aço AISI 1006 com 2 mm de espessura, próximo às dimensões do garfo do patinete original da roda, e parece indicar superdimensionamento do projeto. Na verdade, mesmo as tensões mais altas parecem ser resultado de singularidades numéricas. Entretanto, foi escolhido manter essa espessura,

Figura 10 – Cargas na simulação do garfo.



pois efeitos de flambagem poderiam ser amplificados nas paredes laterais para espessuras menores.

Figura 11 – Distribuição de fator de segurança (critério de von-Mises) no garfo.



6 Automação

Nessa divisão do projeto, é visada a automatização do sistema, em seu sentido mais amplo. Através dos subsistemas desenvolvidos busca-se possibilitar o funcionamento autônomo do robô. Isso confere um caráter importante na concepção do produto, que é a dispensa, ao máximo o possível, da necessidade de utilização de recursos humanos no desempenho dos serviços braçais o que necessitem deslocamento. A maior finalidade de ganhos é na facilitação do trabalho desempenhado e na melhora da qualidade do serviço.

6.1 Funcionamento

Para a concretização dessas premissas, devem ser seguidos os requisitos do produto, e nessa parte foram desenvolvidas as partes eletroeletrônicas do produto. Em síntese, a programação de controle de movimentos do robô, integração entre componentes, sensores e atuadores do produto, alocação de motores e baterias e projeto de carregador elétrico. Tais que foram anteriormente estimados e desenhados na Figura 56 da EAP, situada no Apêndice B.

Para os trabalhos, o desenvolvimento da automação do produto foi abordado em áreas:

6.1.1 Sensoriamento e Controle

Recepção de dados do destino do pedido

Fará a recepção dos dados enviados pelo computador central, como informações acerca dos pedidos, nome do ID do *tablet* da mesa, entre outros. Para tal procedimento, é utilizado um transmissor cujo protocolo de comunicação é wi-fi.

Sensoriamento

Pode-se ressaltar as seguintes funcionalidades.

- Foram implementados para detecção de presenças na frente do robô, sensores de infravermelho e ultrassônicos.
- Para a verificação se a estrutura não virou/caiu no chão, ou mesmo se se chocou com um obstáculo (impacto), é empregado um acelerômetro.
- Para a verificação do compartimento da comida, é feita a medição do peso deste. Para tal é implementada uma célula de carga com conversor.

- O rastreamento das mesas e lugares será feito com um rastreador de posicionamento local com wi-fi.

Controle do robô

Neste processo é realizado o controle dos atuadores referentes a movimentação do robô desde o local de espera e recarga, indo em seguida para o local em que pegará o pedido, seguindo até após a realização da entrega na mesa em que o cliente está esperando, para que o robô retorne para o local de espera e volte a recarregar.

Além dos atuadores de movimentação do robô em si, também é realizado o acionamento dos atuadores referentes à movimentação da bandeja e da tampa.

É fundamental o uso de microcontroladores embarcados para a integração e controle de todos os demais componentes eletro-eletrônicos.

6.1.2 Alimentação

Como já descrito na EAP, no Apêndice B, a elétrica do robô está subdividida em três sistemas: Carregamento, Baterias e Movimentação. Estes três sistemas são constituídos por:

Sistema de Carregamento

Composto por quatro divisões básicas: carregamento por indução; carregamento por plug; base de carregamento; e bobinas indutoras. O escopo deste sistema tem por limites a coleta da energia na rede elétrica do estabelecimento, até o armazenamento dela nas baterias.

Sistema de Baterias

Após a entrada e transformação da tensão para 12V, a energia é armazenada em duas baterias ligadas em paralelo. Tanto a entrada de energia, quanto o seu armazenamento e despacho, será realizado pelo *Battery Management System* (BMS), no qual é responsável pelo controle de carga e por informar caso exista algum erro detectado pelo BMS, o qual realizará o desligamento automático do robô. O Sistema de Baterias ainda conta com a chave liga/desliga do robô, na qual alimenta todos os outros componentes.

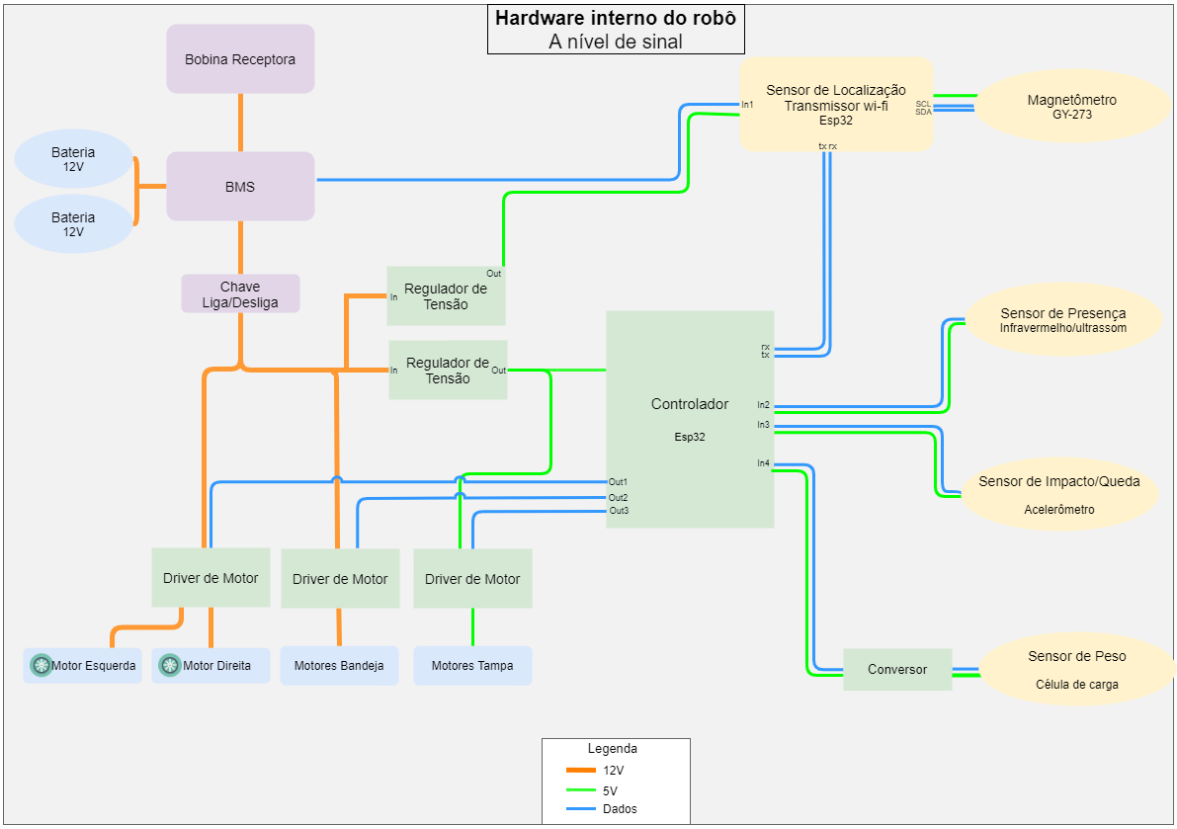
Sistema de Locomoção

O Sistema de locomoção é responsável por toda a alimentação e controle dos motores de movimentação, da tampa e dos atuadores.

6.1.3 Integração de Hardware

A Figura 12 mostra os componentes do hardware interno do robô a uma perspectiva de mais alto nível.

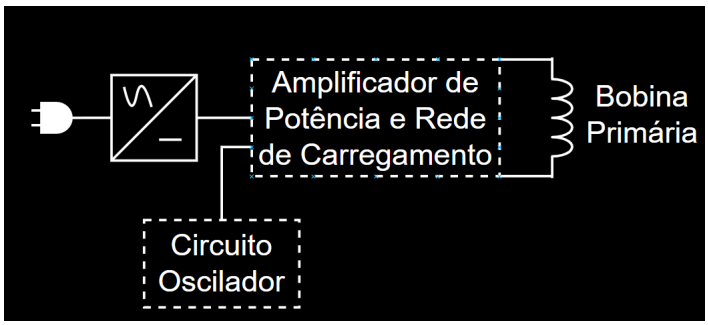
Figura 12 – Diagrama de caixa do hardware do robô.



Fonte: Autores.

Para o Robô Garçom o carregamento é realidade por meio indutivo. Este carregamento está dividido em duas unidades. A primeira é definida como a base de carregamento do robô e fica posicionada na parede do estabelecimento, ela é responsável por conectar-se à rede elétrica do restaurante e excitar a bobina primária, como demonstrado no diagrama de caixas da Figura 13.

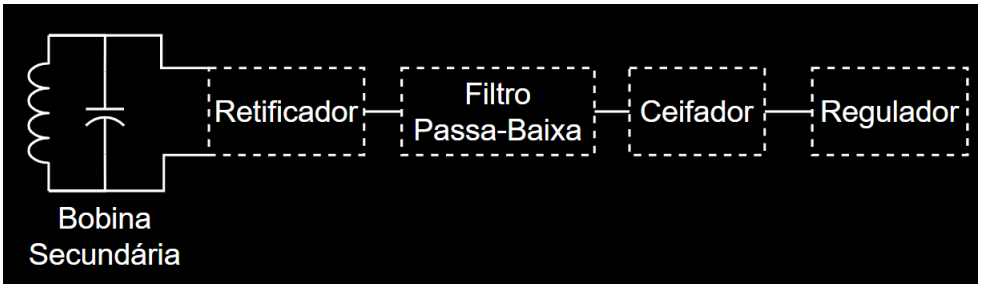
Figura 13 – Unidade Primária de carregamento.



Fonte: Autores.

A unidade secundária está localizada dentro do robô, e sua função é captar a corrente por meio do campo magnético formado entre as bobinas primária e secundária. A unidade secundária apresenta um retificador, filtro passa-baixa, ceifador e regulador, que estão representados na Figura 14.

Figura 14 – Unidade Secundária de carregamento.



Fonte: Autores.

As descrições mais detalhadas dos componentes e módulos seguem adiante.

6.2 Diagramas Elétricos e Eletrônicos

A estrutura elétrica do robô garçom é representada nas Figura 15 e Figura 16, onde a primeira expressa o básico da parte elétrica com os barramentos, enquanto a segunda representa todas as áreas e conexões realizadas.

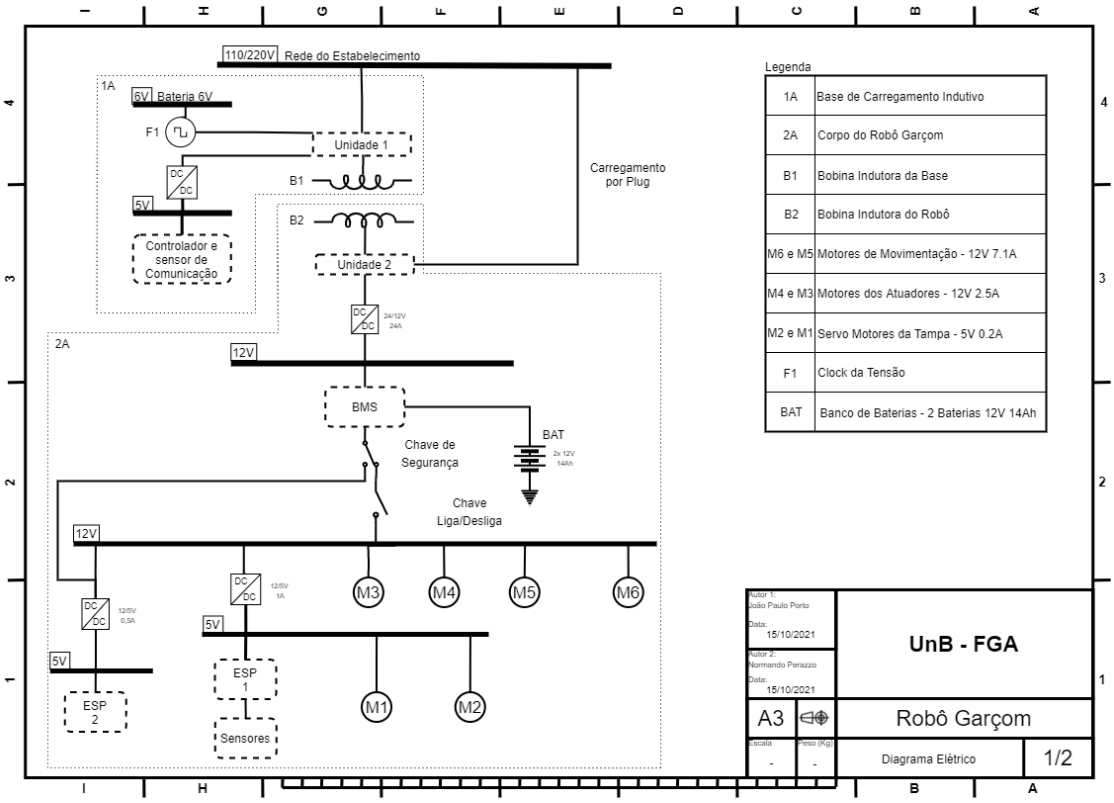
O robô apresenta duas formas de carregamento. Um carregamento indutivo por proximidade, realizado entre a base e o robô, o segundo é um cabo presente no robô que pode ser conectado diretamente na rede do estabelecimento. Tanto a base do robô, quanto o seu carregamento por *plug*, podem ser feitos em uma rede de 110 ou 220V.

A Base do robô apresenta a sua bobina de indução por frequência. O circuito da base utiliza de uma pilha de 6V para a indução, por meio de um *clock*. No caso do carregamento por indução, dentro do robô temos a bobina receptora e a unidade 2, responsáveis por coletar a energia do campo magnético e realizar as transformações iniciais para 12V. No caso do carregamento por *plug*, é realizada também a transformação para 12V.

As duas formas de carregamento funcionam para o carregamento de duas baterias de 12V e 14Ah. As baterias são gerenciadas por um BMS o qual realiza os testes das células das baterias e o controle de erros por meio de um alarme. Que por conseguinte, manda a informação do erro direto para o comunicador que, por fim, envia ao computador central. Para o barramento de 12V existem quatro motores, dois de movimento e dois para o atuador. Já para o de 5V existem mais dois motores, que são servo motores.

A Figura 16 apresenta de maneira mais detalhada o sistema elétrico que integra o robô garçom. São representados neste diagrama as conexões, os pontos de ligação, as placas e os componentes integrados.

Figura 15 – Diagrama elétrico com Barramentos.

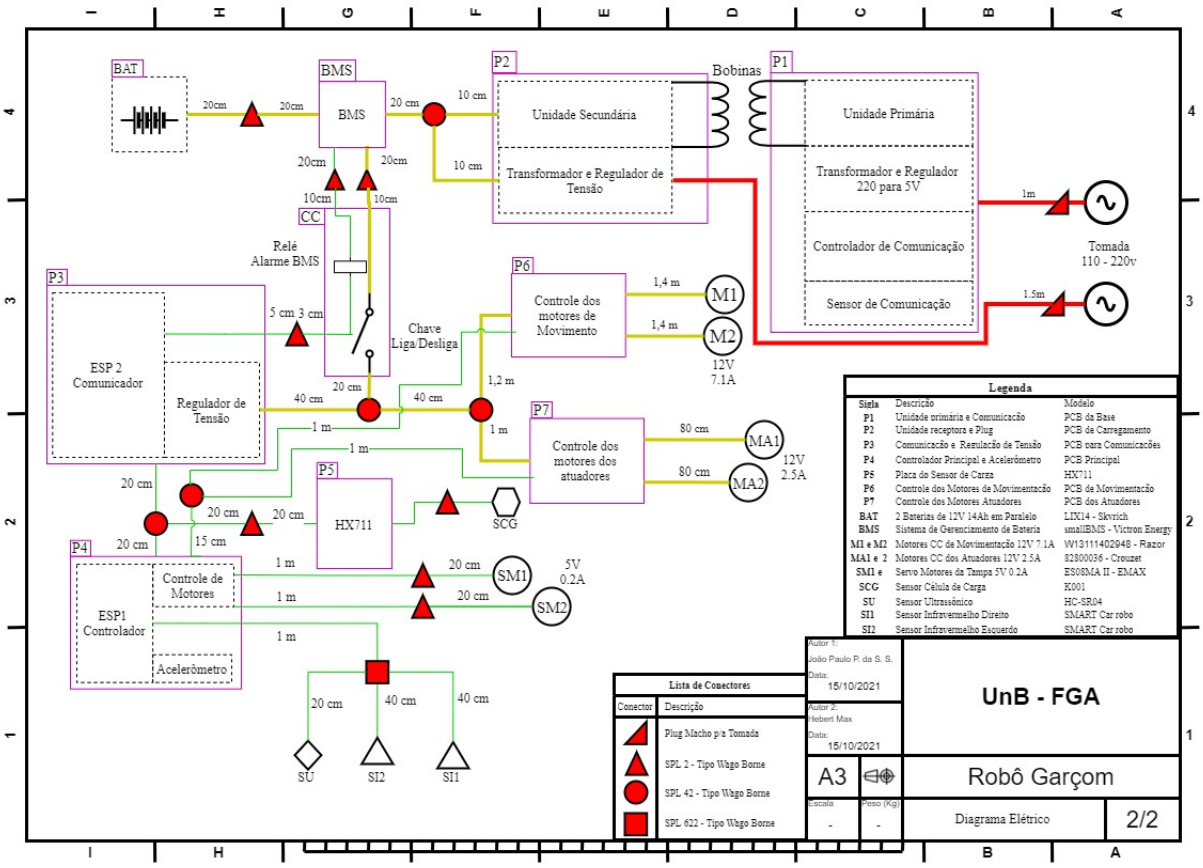


Fonte: Autores.

Neste diagrama são representadas as conexões realizadas entre cada uma das sete placas, enumeradas de P1 a P7, além do BMS, banco de baterias e o chaveamento, composto pelo alarme e pelo botão liga e desliga do robô. São representadas as três tensões no diagrama diferenciadas pelas cores vermelha (110/220V), amarela (12V) e verde (5V).

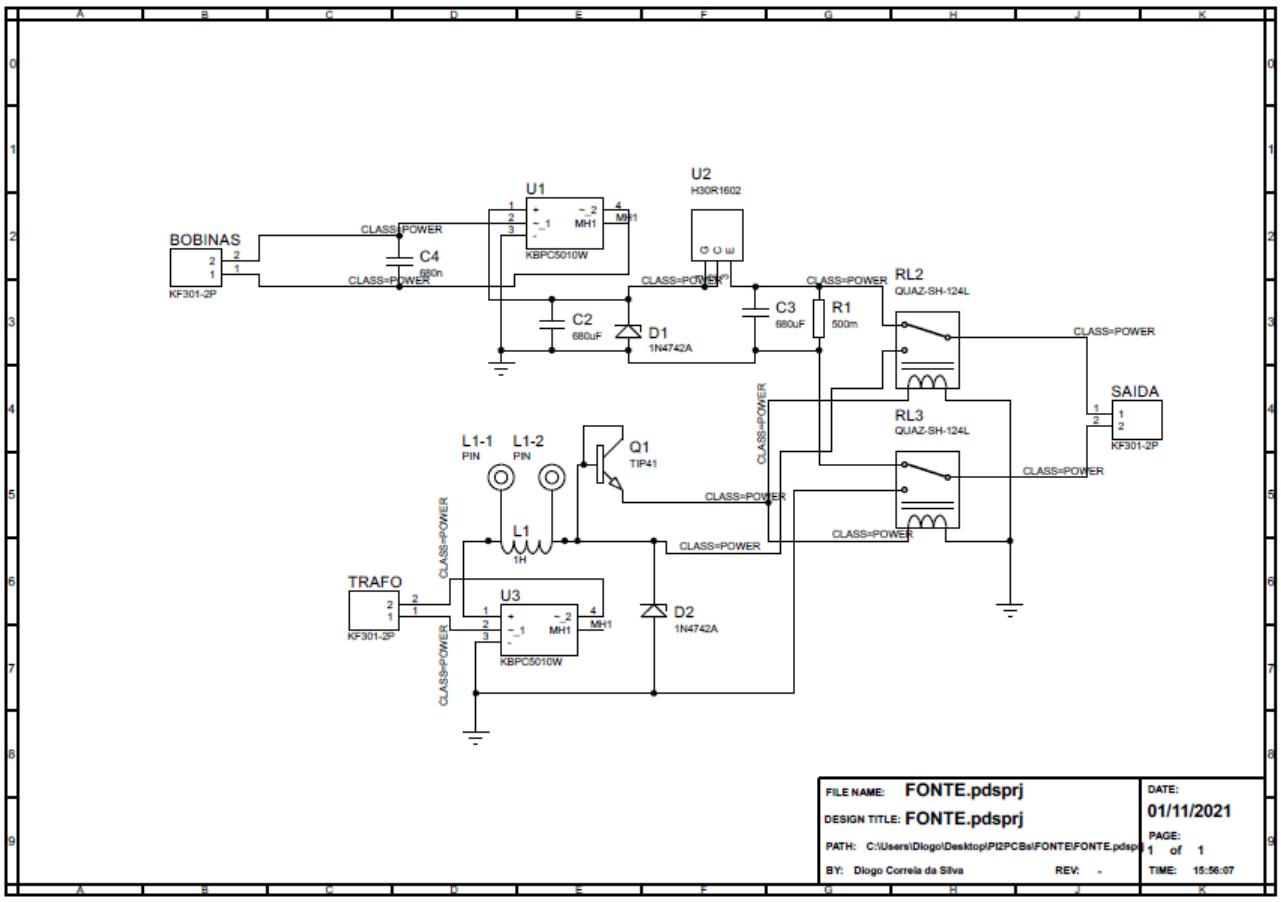
Para o cabeamento foram aplicados conectores entre todas as placas, para fácil remoção e troca. Foram aplicados três tipos de conectores para emenda e derivação dos cabos. Todos os cabos estão especificados com o seu comprimento, diâmetro e quantidade de vias na Figura 16.

Figura 16 – Diagrama elétrico.



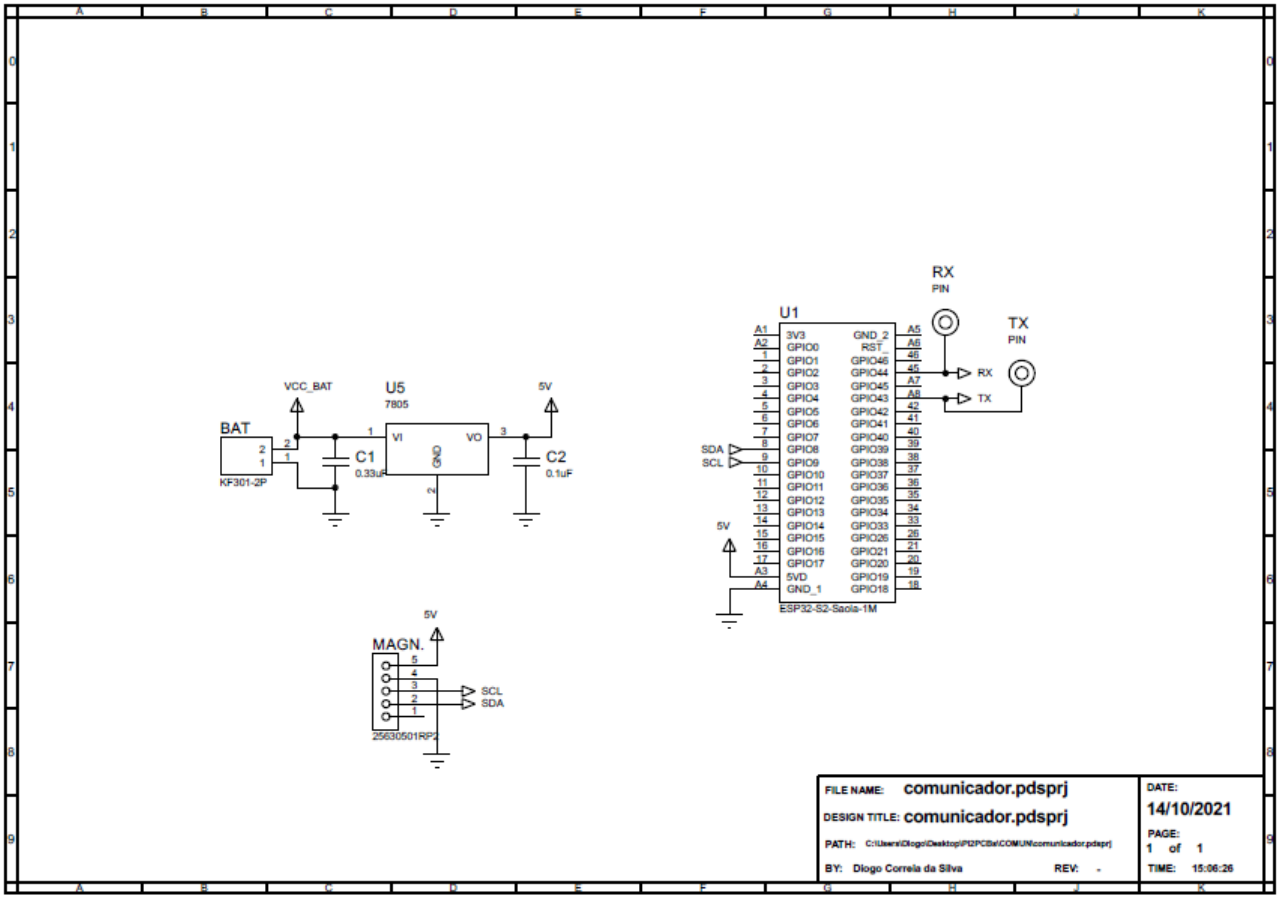
Fonte: Autores.

Figura 17 – Esquemático da placa de carregamento (P2).



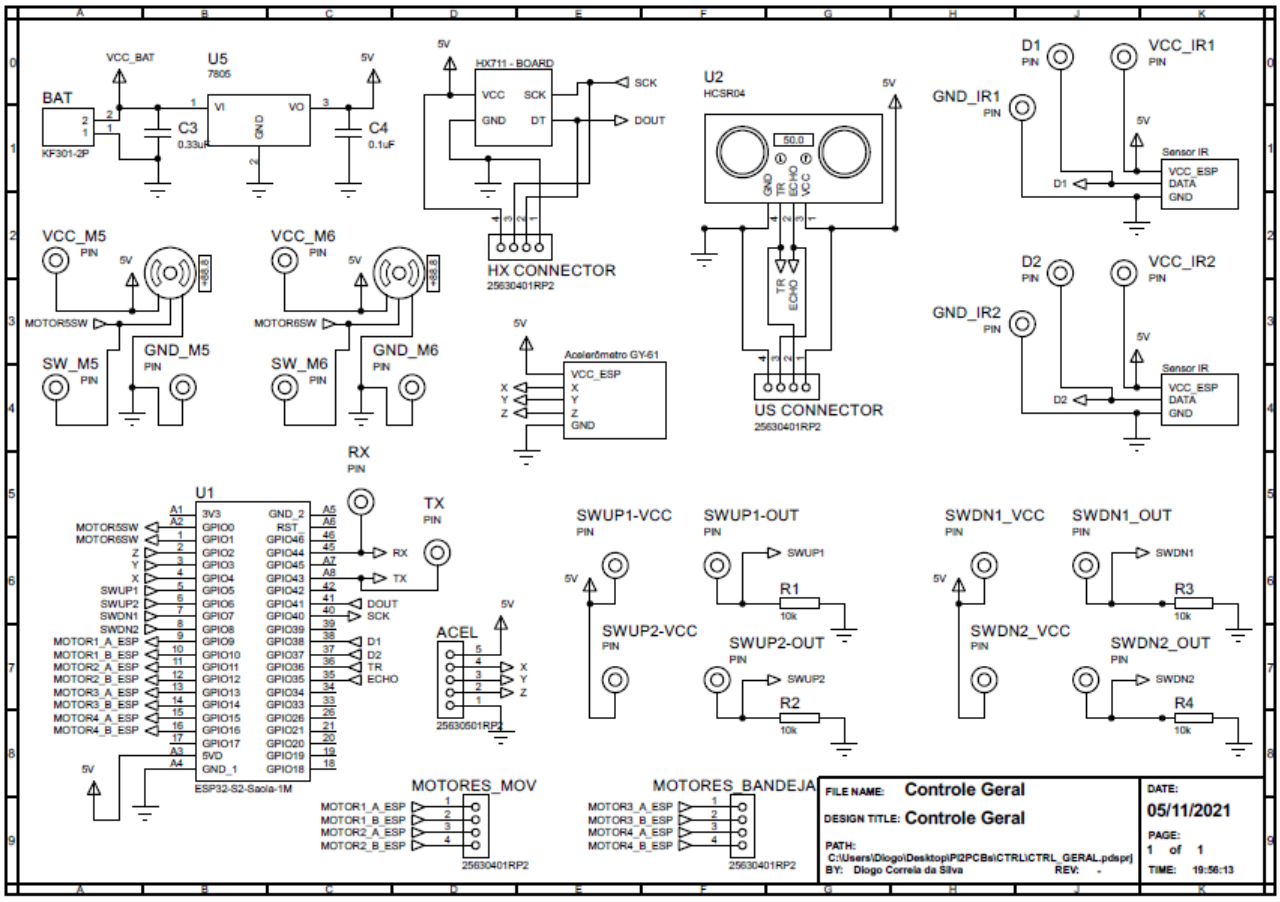
Fonte: Autores.

Figura 18 – Esquemático da placa de comunicação (P3).



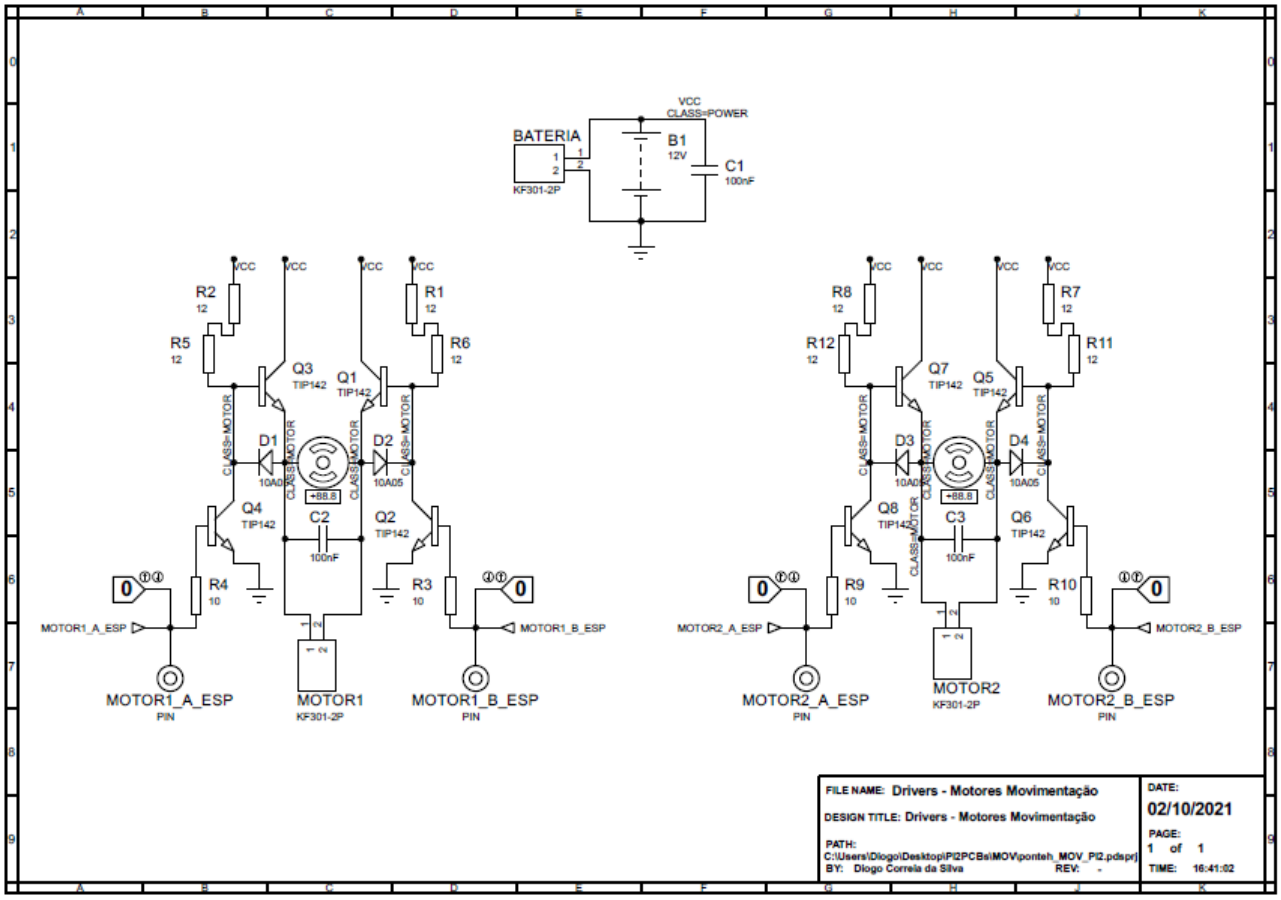
Fonte: Autores.

Figura 19 – Esquemático da placa-mãe (P4) com controlador.



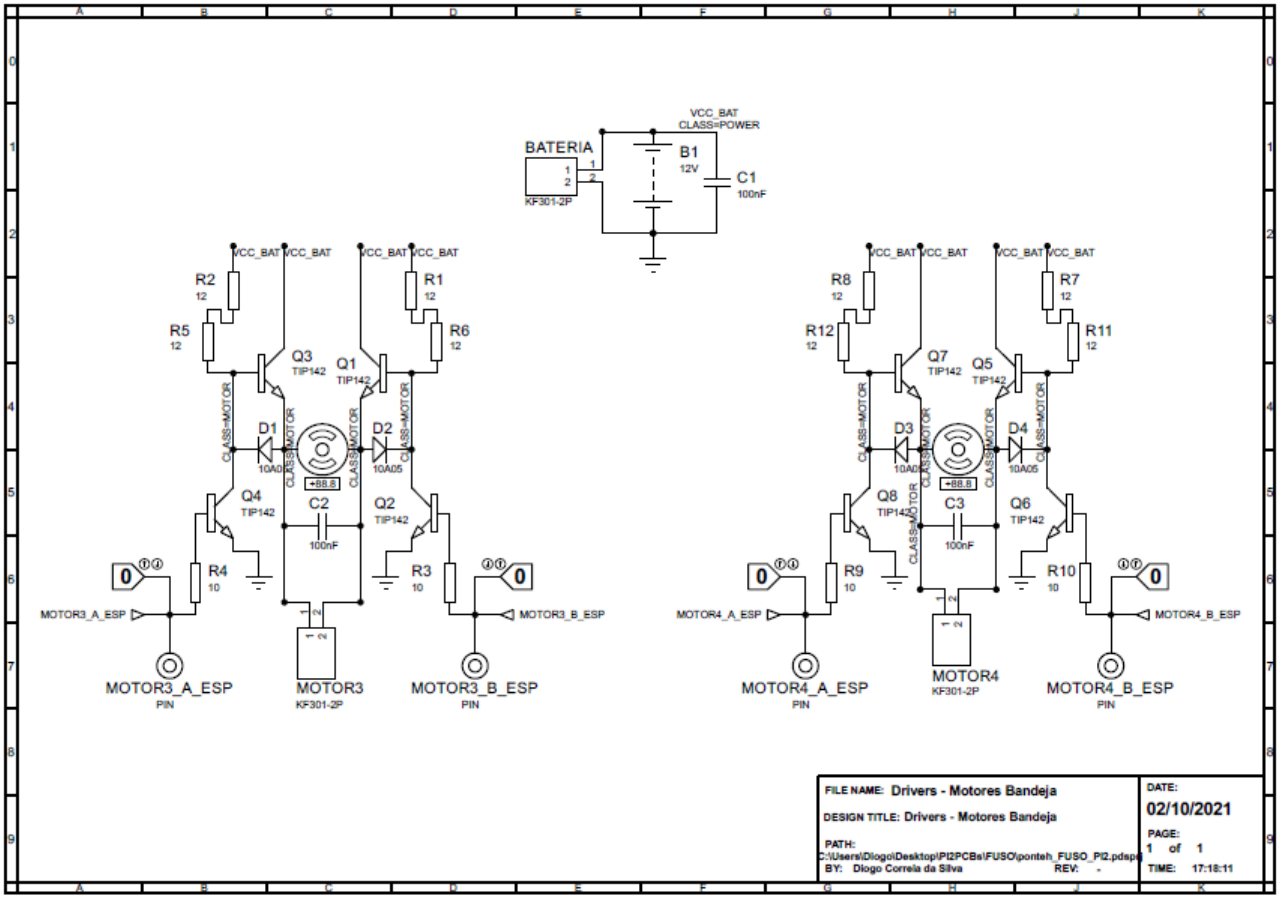
Fonte: Autores.

Figura 20 – Esquemático da placa de *driver* dos motores de movimentação (P6).



Fonte: Autores.

Figura 21 – Esquemático da placa de *driver* dos motores de elevação da bandeja (P7).



Fonte: Autores.

6.3 Carregamento e Armazenamento de Energia Elétrica

O carregamento do Robô Garçom é realizado por duas formas, uma delas por *plug*, na qual existe um cabo no robô que conecta-se diretamente na rede do estabelecimento. A segunda forma não necessita de contato humano, e é realizada pela base de carregamento por meio de indução. O carregamento indutivo foi empregado no robô garçom visando a autossuficiência no procedimento de recarga enquanto o robô estiver ocioso próximo à base de carregamento. O sistema de carregamento alimentará as duas baterias que servem de fonte para os demais componentes do robô. Caso haja necessidade de um carregamento mais rápido ou de emergência, realizado por meio de um *plug* macho, que se conecta na tomada do estabelecimento. Esta, pode ser de 110 ou 220V.

6.3.1 Carregamento Indutivo

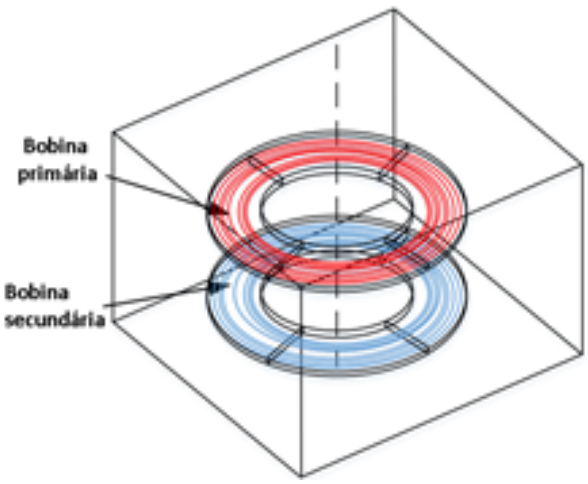
O carregamento indutivo foi escolhido devido a autonomia de operação do robô. Este tipo de carregamento possui algumas derivações, e para a melhor eficiência de carregamento, é estudado e dimensionado o carregamento do tipo acoplamento indutivo ressonante, utilizado em projetos semelhantes, tendo os benefícios de pouco aquecimento e maior taxa de transferência de energia [DALTIO 2019].

O dimensionamento das bobinas partiu da escolha da geometria que foi limitada pelo layout do robô. As bobinas serão posicionadas na lateral da carenagem e limitado a até 40cm de diâmetro para o tamanho, e os parâmetros de tensão e corrente adotados foram de 12V e 12A respectivamente. Para estas condições foi escolhido o layout planar em forma de anel como no esquemático da Figura 22.

As bobinas foram dimensionadas de maneira a utilizar a Equação 6.1.

$$D = d \cdot 2\sqrt{2} \tag{6.1}$$

Figura 22 – Layout das bobinas



O diâmetro D é determinado em função da distância máxima d entre as bobinas, definido como 5cm. Para o cálculo da indutância de um indutor de geometria semelhante, e de frequência de operação de até 30MHz, foi utilizado a Equação 6.3 [Hershenson et al. 1999].

$$L = \frac{\left(\frac{\text{Diâmetro}_{\text{ext}}}{2}\right)^2 \cdot A^2}{30 \cdot A - 11 \cdot \text{Diâmetro}_{\text{int}}} \tag{6.2}$$

$$A = \frac{\text{Diâmetro}_{\text{int}} + N\,(w + s)}{2} \tag{6.3}$$

em que N é o número de espiras, w e s o diâmetro e espaçamento do fio, respectivamente. Com o cálculo destes e demais parâmetros, como a obtenção do número de voltas. O Fator de qualidade aplicado é definido como Q, onde este mede o desempenho da transmissão de energia. Quanto maior o fator Q, maior a eficiência da transmissão. Este parâmetro relaciona reatância e a resistência, como mostrado na equação a seguir, sendo f a frequência de operação, L a indutância e R a resistência, adotada como $R = 0,09\Omega$ [DALTIO 2019].

$$Q = \frac{2\pi f L}{R} \tag{6.4}$$

Como resultado da Equação 6.4 o fator de qualidade apresentou o resultado de $1,85$. No estudo das bobinas, o conceito de coeficiente de acoplamento K, em que K varia de 0 a 1, indica o nível de acoplamento das bobinas, sendo $k = 1$ totalmente acoplado (acoplamento ideal), e $k = 0$, desacoplado. O coeficiente de acoplamento compõe a definição da indutância mútua, que é o fluxo de ligação das duas bobinas. Neste estudo, por as bobinas primária e secundária possuírem as mesmas dimensões, e considerando um acoplamento perfeito, a indutância mútua será equivalente às indutâncias de L_1 e L_2. Caso estes valores mudem, a indutância mútua M pode ser obtida na Equação 6.5 [MOTTA 2016].

$$M = K\sqrt{L_1 L_2} \tag{6.5}$$

Na literatura foi evidenciado a importância de medições experimentais para a validação do dimensionamento, tornando este um limite metodológico no projeto das bobinas de indução para o carregamento do Robô Garçom.

6.3.2 Base de Carregamento

O sistema de carregamento indutivo é composto pela base de carregamento que inclui a bobina e circuito primário, que pode ser visualizado no subseção F.2.5, e a base receptora fixada à lateral do robô, que contém a bobina e o circuito secundário. O principal

requisito do carregamento é alimentar as baterias, fornecendo uma corrente de saída de 24A e tensão de saída 12V, com proximidade de até 100mm entre as bobinas.

O dimensionamento das bobinas foi realizado a partir do estudo de frequência nos circuitos detalhados na subseção F.2.4, cujo foi definido a indutância de 100µH, e suas dimensões constam na Tabela 11. A partir dessa definição, foi fixado o valor da indutância e o tamanho máximo definido pela estrutura do robô, como diâmetro do fio, espaço entre as espiras, número de voltas e o diâmetro interno e externo. Esses parâmetros foram definidos a partir das fórmulas para bobina plana helicoidal e a simulação no ambiente *Ansys Maxwell*, como mostrado na subseção F.2.4. A bobina primária e secundária possuem as mesmas dimensões, assim não havendo diferenciação entre ambas no quadro a seguir.

Tabela 11 – Parâmetros dimensionais da bobina.

Parâmetros dimensionai	**Valor**
Diâmetro interno	10 mm
Diâmetro externo	296,4 mm
Indutância	100 μH
Número de voltas	19
Espaço entre as voltas	0,81 mm
Diâmetro do fio utilizado	3,665 mm
Comprimento do fio	18000 mm

6.3.3 Baterias

O Robô apresenta duas baterias do modelo LIX14 da Skyrich. Cada uma possui uma tensão de 12V e carga de 14Ah. A escolha pela bateria foi primeiro pela sua utilização em larga escala em motocicletas, e pelo seu peso (700g) e tamanho (10,5cm × 8,5cm × 14,5cm) baixos. A bateria possui 240A de arranque a frio (CCA) [SKYRICH 2021].

As baterias serão interligadas em paralelo e seguirão para o BMS, no qual realizará o controle de carga e descarga das baterias, que está setado para 80% e 10% e sua amperagem hora sobe para 24Ah.

Para otimizar a utilização das baterias, de maneira a protegê-las e aumentar a eficiência, será utilizado um *SmallBMS* da Victron Energy. O BMS será interligado ao controlador e também a chave liga/desliga, no qual em caso de seu alarme ser disparado, realizará a parada completa do robô e informará ao computador central a causa da parada, podendo ser estas por alta (> 50°C) ou baixa (< 5°C) temperatura, ou em caso de curto circuito [VICTRON 2021].

6.4 Comunicação

Esse módulo é responsável por toda a comunicação pelo robô no ambiente de serviço, incluindo o roteador do estabelecimento (Central), o Robô Garçom, sua base de carregamento e os *tablets* das mesas de clientes, sempre via wi-fi. Segue a descrição.

6.4.1 Dispositivo comunicador

O Robô Garçom se comunicará com o computador central (*backend*), responsável pela recepção e processamento dos pedidos, através da rede WI-FI (padrão IEEE 802.11) do sistema. Com tal, é estabelecida uma conexão cliente-servidor entre o(s) robô(s) e a Central, respectivamente, via *socket*.

O componente responsável por essa transmissão é um outro microcontrolador Esp32, conectado via UART ao controlador principal. Ele possui comunicação wi-fi nativa [Espressif 2021]. Este também é responsável pela localização espacial da mesa do *tablet*. Ele é conectado com uma antena wi-fi através de um cabo *pigtail* e sua saída UTF. Também nele é conectado um sensor magnetômetro (GY-273) via protocolo I2C.

Detalhes sobre o funcionamento e algoritmo do comunicador são dados na subseção 6.6.2. Este dispositivo também é responsável pela localização do robô, que é descrita na subseção 6.5.8.

Também há um módulo comunicador dentro da base de carregamento. Ele é necessário para que o robô também possa encontrá-la. Ele também é uma Esp32.

6.4.2 Alarme

Para assegurar a integridade dos componentes eletro-eletrônicos e ciência do correto estado de funcionamento do robô pelo proprietário, há um sistema de alarme que detecta irregularidades no fornecimento das baterias ou mesmo cargas muito baixas, que integra o BMS com o comunicador. Ao detectar algum problema do tipo o BMS manda um sinal de pré-alarme (até 30 segundos antes de desligar o fornecimento de energia [VICTRON 2021]) para um pino (18) de *input* do comunicador (GPIO18 da ESP32). Ao receber o sinal, o comunicador faz chamada de uma interrupção interna que envia uma mensagem ao servidor central avisando que há “Problema com a bateria”, após algum tempo os circuitos do robô são desligados.

6.5 Locomoção

Estão dispostos nesta sessão todos os componentes responsáveis por todas as movimentações realizadas pelo robô. Essas movimentações são: Locomoção sobre o seu eixo,

movimentação vertical da bandeja de alimentos, e abertura ta tampa superior.

6.5.1 Dispositivo controlador

O microcontrolador ESP32 é responsável por receber os sinais de todos os sensores para a tomada de decisões sobre os movimentos dos motores e localizar a mesa do cliente. Serão utilizadas duas destas, sendo uma exclusivamente para realizar a localização das mesas e a outra contendo todos os sensores (exceto de localização) e atuadores presentes no robô.

6.5.2 Motores

Para a movimentação do robô, são utilizados dois motores *Razor W13111402948* [RAZOR 2021]. Estes motores foram escolhidos dada a sua versatilidade em ser o motor acoplado à roda. Estes motores são de corrente contínua com 12V e $7,1$A. Cada motor apresenta um peso de $9,5$Kg.

Os motores responsáveis por movimentar a bandejá, foram denominados motores atuadores. estes são da CROUZET modelo 82800036 [CROUZET 2021]. Estes motores apresentam uma tensão de 12V e corrente de $2,5$A, com peso de $0,8$g.

Para a tampa, a qual realiza a abertura superior para a retirada dos alimentos, foi utilizado dois servos motores da EMAX por modelo ES08MA II [EMAX 2021].

Os cálculos para o dimensionamento foram utilizados os cálculos fornecidos pela área de estruturas, e assim escolhidos os motores de acordo com os pré-requisitos.

6.5.3 *Drivers*

Para o controle dos motores de movimentação do robô e de movimentação das bandejas, foram criadas duas placas de *driver*, cujos esquemáticos se encontram nas figuras 20 e 21, ambas contendo duas pontes H responsáveis pelo direcionamento dos motores quanto ao seu sentido de rotação e para a correta distribuição de energia para os mesmos. Nelas foram utilizados os transistores bipolares de junção TIP142 da Motorola, que suportam até 10A de corrente continua, suportando picos de 15A (ONSEMI, 2021), juntamente com diodos 10A05 que permitem a passagem de 10A (BYTES, 2021). Encontram-se no seção F.2 algumas capturas de tela relativas às simulações dessas placas.

A ativação desses motores se dá através de quatro pinos conectados na ESP U1 da **??**, os quais correspondem ao sinal de direção dos motores. Além de definir a direção que os motores estarão girando, esses pinos também são responsáveis pelo sinal de *Pulse Width Modulation* (PWM), o qual define a velocidade de rotação dos mesmos.

O PWM define a velocidade do motor através do *Duty Cicle*, o qual representa a porcentagem da potência de giro do motor, sendo o *duty* de 100% igual ao motor girando com toda sua potência, *duty* de 50% é metade da potência, 0% é igual a motor parado.

Para a alimentação e conexão dessas placas de *driver* com a placa BMS e com os motores, foram adicionados 3 bornes, cada um com 2 conexões. Dois bornes são para conexão com os dois motores de cada placa, enquanto o terceiro borne foi adicionado para a conexão com o BMS.

Quanto ao motor relativo à abertura da tampa, não foi preciso criar uma placa de controle para ele em específico,conectando o pino de controle do mesmo diretamente na ESP U1.

6.5.4 Sensor acelerômetro

Para a detecção de impactos e de queda do robô, é utilizado módulo sensor acelerômetro GY-61, o qual possui três saídas analógicas correspondentes à aceleração do robô nos eixos X, Y e Z do mesmo. A faixa de aceleração que esse módulo consegue captar é de +/- $3g$, sendo $g = 9,81$ m/s^2 (aceleração gravitacional), e ele varia a tensão de suas saídas em 300 mV/g [INDIAMART 2021]. Para isso, são usadas três entradas analógicas da ESP e realizadas as conversões dos sinais analógicos recebidos para digital através da função ADC de 10 bits dessas entradas e esse valor é então interpretado pelo microcontrolador. A detecção de queda se dá pela variação da aceleração no eixo Z, o qual deve manter aceleração para baixo e caso o valor recebido do acelerômetro referente a esse eixo mostre variação, um sinal de alerta é enviado ao computador central informando que o robô está caído. Da mesma forma, a detecção de impacto se dá através da aceleração nos eixos X e Y. Caso a aceleração nesses eixos se dê de maneira abrupta, se considera que houve um impacto e um sinal é enviado ao computador central para informar tal situação.

A implantação no controlador é efetivada pela função acelerômetro(); que realiza a leitura dos dados enviados pelo sensor acelerômetro, indicando se o robô sofreu alguma queda. Caso aconteça, o controlador automaticamente envia uma mensagem para o módulo comunicador apontando qual a direção da queda.

6.5.5 Sensores infravermelho

Para a detecção de obstáculos a frente do robô são utilizados dois módulos sensores infravermelho, cada um possui um receptor e um emissor IR, os quais enviam e recebem luzes infravermelhas a luz enviada é refletida para o sensor quando um obstáculo se coloca a frente dessa luz levando o pino de saída do sensor para o nível lógico baixo [Mattede 2021], detecta obstáculos de 2 a 80 cm de distância do robô.

Os sensores infravermelho indicam quando há algum obstáculo à frente do robô,

quando isso ocorre a saída dos sensores vão para nível lógico baixo. A implementação é realizada por uma função dentro do código do controlador chamada de obstaculo();, os sensores são empregados no robô para garantir a automação.

6.5.6 Sensor ultrassônico

O sensor ultrassônico é utilizado para detectar a distância a qual o obstáculo se encontra do robô, o sensor ultrassônico emite uma onda sonora de alta frequência e mede o intervalo de tempo em que o impulso é emitido e o eco dessa onda que é recebido, os pulsos de onda são emitidos em ciclos e quando o eco é coletado e convertido em um sinal elétrico [SENSE 2014].

O sensor ultrassônico mede a distância entre o robô e o obstáculo, é implementado como uma função no código do controlador, a função é a getdistance(); que mede a distância e converte ela para centímetros.

Para determinar se há algum obstáculo na frente do robô e se esse está obstruindo a movimentação do robô a função obstáculo(); é utilizada, dentro dela é verificado se um dos dois infravermelhos foram acionados, caso isso aconteça, a função getdistance é chamada para verificar qual a distância e caso a distância seja menor do que 60 centímetros a tomada de decisão sobre como desviar do obstáculo é realizada.

6.5.7 Sensor de peso

Para o sensor de peso é utilizada a célula de carga, a qual realiza uma leitura da força aplicada sobre ela através da mudança do valor de resistência da ponte de *Wheatstone* em seu interior.

Para tal leitura, ela verifica a deformação que ocorreu no metal em que está fixada e, caso uma das resistências da ponte mude, altera-se a tensão em um dos terminais dessa ponte [STRAUB 2019]. Essa variação da tensão é muito baixa, portanto o conversor HX711 é usado para amplificar esse sinal.

A ESP32, ao ler esse valor, o converte para o que seria o equivalente a massa. A célula de carga é empregada para detectar se há ou não comida na bandeja.

Essa função é usada para garantir que o pedido já foi entregue por completo ao cliente, dentro do código do controlador a função balança(); pega a média dos últimos 5 valores lidos pela célula de carga para determinar o valor atual que consta de carga em cima da estrutura da balança, o pedido é considerado entregue quando o valor retornado pela leitura for zero.

6.5.8 Localizador

Essa função é realizada pelo comunicador descrito na subseção 6.4.1. É possível localizar dispositivos que emitem ondas eletromagnéticas identificadas.

Para o caso do robô serão localizados os sinais emitidos pela rede wi-fi dos dispositivos de interesse - nesse caso podendo ser o robô, sua base de carregamento, o roteador central e o *tablet* da mesa de entrega - sendo obtidos seus SSID e a partir desses parâmetros, são estimadas.

Vale ressaltar que os sinais medidos são sempre submetidos a um filtro média móvel com N amostras:

$$x_k = \frac{1}{N}\sum_{n=1}^{N} x_{n-N} \tag{6.6}$$

Obtenção das distâncias

Através da potência de sinal recebido (RSSI) de cada rede de interesse medida, pode ser estimada a distância pelo seguinte cálculo na equação 6.7 [Cunha 2017]:

$$P_{r(k)} = -\left(10\eta\log_{10}\left(\frac{d_k}{1\text{m}}\right) + P_0\right) \tag{6.7}$$

Onde têm-se os parâmetros:

- $P_{r(k)}$ é a potencia do sinal recebida do emissor
- P_0 é a potência de sinal medida na distância de 1m.
- η é a constante de perda de percurso do sinal
- d_k é a distância do emissor, em metros.
- Esses parâmetros - P_0 e η - podem ser estimados e ajustados com algumas medidas de amostras.

Portanto a distância do dispositivo alvo pode ser estimada por:

$$d_k = 10^{-\frac{P_{r(k)}+P_0}{10\eta\cdot 1\text{dBm}}}\cdot 1\text{m} \tag{6.8}$$

Por calibração, foram atribuídas às constantes como $\eta = 1,9$ e P_0.

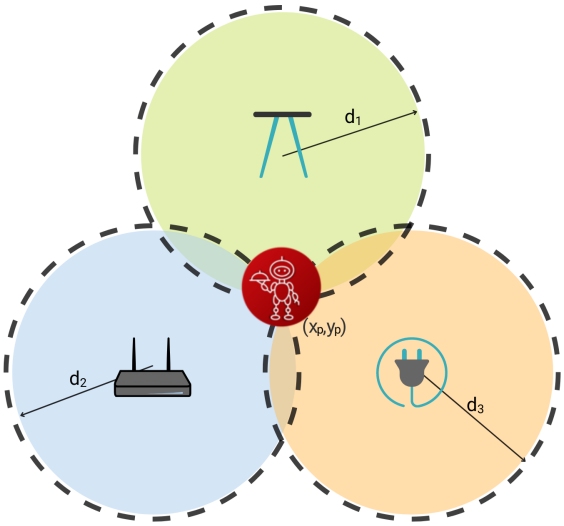
Trilateração

Este procedimento mapeia as coordenadas de cada dispositivo.

Os que serão localizados, chamados *beacons*, devem ter suas coordenadas previamente definidas. No caso são o roteador Central, a base de carregamento e o *tablet* da mesa alvo.

O localizador é chamado de âncora e nesse caso é o Robô Garçom. Ele a partir desses dados pode modelar o sistema mostrado na Figura 23 :

Figura 23 – Cenário espacial com âncora e *beacons*.



Fonte: Autores.

Sabendo as coordenadas (x_k, y_k) e distâncias d_k, onde $k = 1,2,3$, dos *beacons* temos:

$$\begin{cases} (x_1 - x_p)^2 + (y_1 - y_p)^2 = d_1 \\ (x_2 - x_p)^2 + (y_2 - y_p)^2 = d_2 \\ (x_3 - x_p)^2 + (y_3 - y_p)^2 = d_3 \end{cases}$$

Para isolar (x_p, y_p) é utilizada a Equação 6.8 para estimar as coordenadas espaciais de cada ponto da âncora [Reck 2016].

$$x_p = \frac{(d_2^2 - d_3^2 + y_3^2 - y_2^2 + x_3^2 - x_2^2)(y_2 - y_1) - (d_1^2 - d_2^2 + y_2^2 - y_1^2 + x_2^2 - x_1^2)(y_3 - y_2)}{2(x_3 - x_2)(y_2 - y_1) - 2(x_2 - x_1)(y_3 - y_2)}$$

$$y_p = \frac{(d_2^2 - d_3^2 + y_3^2 - y_2^2 + x_3^2 - x_2^2)(x_2 - x_1) - (d_1^2 - d_2^2 + y_2^2 - y_1^2 + x_2^2 - x_1^2)(x_3 - x_2)}{2(y_3 - y_2)(x_2 - x_1) - 2(y_2 - y_1)(x_3 - x_2)}$$

Assim, sabendo sua localização, o robô garçom poderá seguir para o ponto de localização escolhido como destino.

Escolha da direção

Para poder seguir para o caminho apropriado, é necessário saber o ângulo de sua orientação no cenário. Para isso é utilizado um magnetômetro (GY-273) para tal. O componente funciona como um sensor bússola, medindo sua orientação (absoluta θ_0) em relação

ao eixo magnético global. Antes do robô se mover é medida sua direção θ_i. Em seguida ele calcula a direção a se virar por:

$$\theta_d = \arctan\left(\frac{y_d - y_p}{x_d - x_p}\right) + \theta_0 \tag{6.9}$$

Onde (x_d, y_d) é coordenada do dispositivo destino. A partir de $\theta_d - \theta_i$ ele decide se envia a mensagem 'Direita', 'Esquerda' ou 'Atrás' para o Controlador, a fim de deslocá-lo a direção pretendida. Depois de sucessivos pequenos giros, quando for medido

$$\theta_d \simeq \theta_i$$

a direção é considerada atingida e o Comunicador manda a mensagem 'Frente' ao Controlador. Após um determinado tempo (cerca de 1 s) e deslocado um pequeno trecho do caminho, é repetido todo o ciclo de localização para deslocamento, a partir da obtenção de novas distâncias. Tal algoritmo é implementado na subseção 6.6.2.

6.6 Programação dos módulos

São descritos os algoritmos que regem os microcontroladores envolvidos no projeto.

6.6.1 Código para o controlador

O ESP32 conta com dois núcleos físicos XTENSA LX6 (dual-core), cada um deles roda a 240MHz, isso permite a programação multicore(RTOS), viabiliza que dois ou mais processos sejam executados simultaneamente [Neves 2020], para que isso ocorra são criadas tarefas que podem ser atribuídas ao núcleo zero ou um e definidas prioridades na execução dessas tarefas. Para o Robô garçom foram criadas duas tarefas uma para a movimentação do robô e outra para a leitura dos sensores de obstáculos e queda, algumas vantagens desse tipo de implementação são o aumento na velocidade e a diminuição no consumo de energia.

A tarefa para movimentação do robô é baseada nos comandos recebidos por comunicação UART provenientes do comunicador que podem ser vistos na Figura 24. Após receber o comando das direções que deve seguir para chegar até a mesa, parar ou voltar para a base o robô inicia as duas *tasks* responsáveis pela sua movimentação no salão do restaurante, a *task zero* é encarregada de acionar os motores das rodas , motor da bandeja e motores da tampa, e que decisões tomar quando cada um dos comandos do comunicador com chegar e a *task one* fica encarregada de ler os sensores de obstáculo e queda, quando um obstáculo é detectado pelo sensor de infravermelho o sensor ultrassônico mede a distância a que está esse obstáculo e em seguida define o melhor caminho a ser feito pelo robô para desviar do mesmo, se o obstáculo estiver a mais de 60 centímetros de distância o robô continua a se movimentar normalmente caso contrário ele verifica qual dos dois sensores infravermelhos que captou o obstáculo caso tenha sido o infravermelho da esquerda o robô irá

desviar pela direita e vice-versa, caso os dois sensores estejam captando o obstáculo o robô para por alguns segundos e em seguida continua sua movimentação. No caso do sensor acelerômetro detectar alguma queda do robô uma mensagem é enviada para para o módulo comunicador e o sistema do robô é desligado.

Ao chegar a mesa para entregar o pedido o motor da bandeja e o da tampa são acionados, após essa etapa ser concluída a célula de carga começa a ser lida constantemente até que o valor retornado na média das últimas cinco medidas por ela seja igual a zero, em seguida um atraso de 3 segundos é implementado após os término do atraso os motores da bandeja e tampa são acionados para o fechamento do compartimento , depois o robô retorna para a base onde fica esperando receber um novo pedido de entrega.

A Figura 24 mostra as decisões deste módulo.

6.6.2 Código para o comunicador

Ao ligar, são configuradas a conexão UART com o controlador e a comunicação wi-fi. Em seguida ele tenta estabelecer com conexão com o roteador do estabelecimento e, ao êxito, estabelece conexão de cliente com o servidor central. Também define uma interrupção acionada com o sinal de "alarme"do pino para o BMS (caso este ocorra ele manda a mensagem "Erro na bateria"ao servidor central). Finalmente ele define uma *callback* para recepção de pedidos.

Tal qual, inicialmente o Robô Garçom ficará ocioso em sua base até que o servidor central o enviará o identificador de rede (SSID) da mesa (*tablet*) para o qual deverá entregar o pedido. Nessa etapa ele está carregado com o pedido.

A partir daí ele:

1. Detecta o sinal do *tablet* referido.
2. A partir do RSSI ele calcula a distancia de 3 dispositivos - o roteador, a base e o *tablet* da mesa, como descrito na 6.5.8.
3. Então ele calcula a coordenada de sua posição por trilateração.
4. A partir daí é escolhida a direção pela qual o robô deve virar (movimento das rodas) para ficar alinhado ao lugar correto. O método também é descrito na 6.5.8.
5. É dado o comando ao controlador para qual direção ele deve acionar a locomoção.

E então ele se dirige até a mesa. Quando entregue todo o pedido - o compartimento interno vazio - ele volta a sua base (sempre enviando mensagens de *feedback* à central), esperando um próximo SSID de mesa para repetir o ciclo de entregas. Apesar de que ele poderá receber comandos de parada ou volta da Central a qualquer momento de seu percurso.

A Figura 25 mostra as decisões deste módulo.

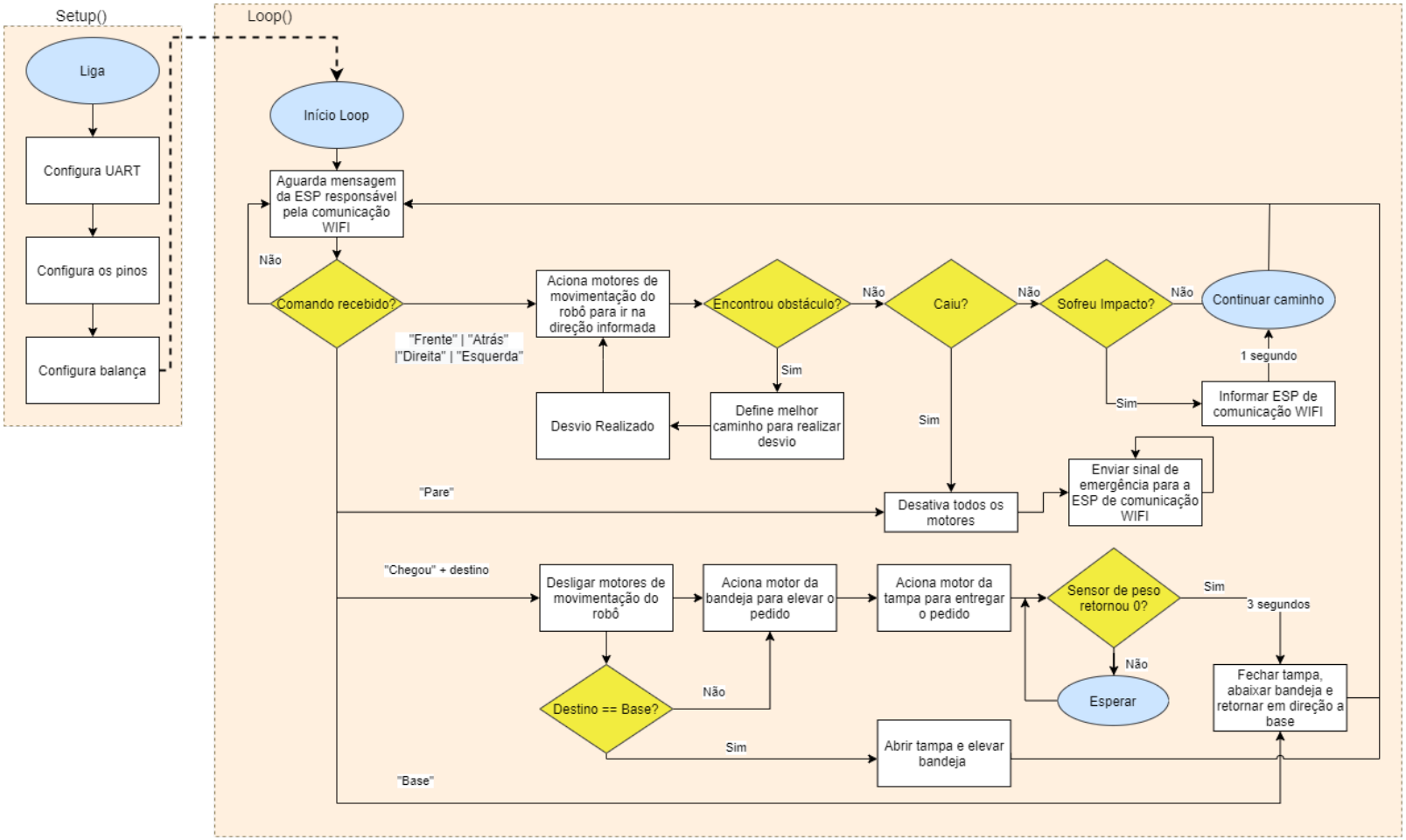
O código implementado[1] encontra-se no seção I.2

6.6.3 Código para o comunicador da base de carregamento

A base de carregamento precisa que seu comunicador emita seu SSID para que o robô possa rastreá-la. O código implementado encontra-se no seção I.2.

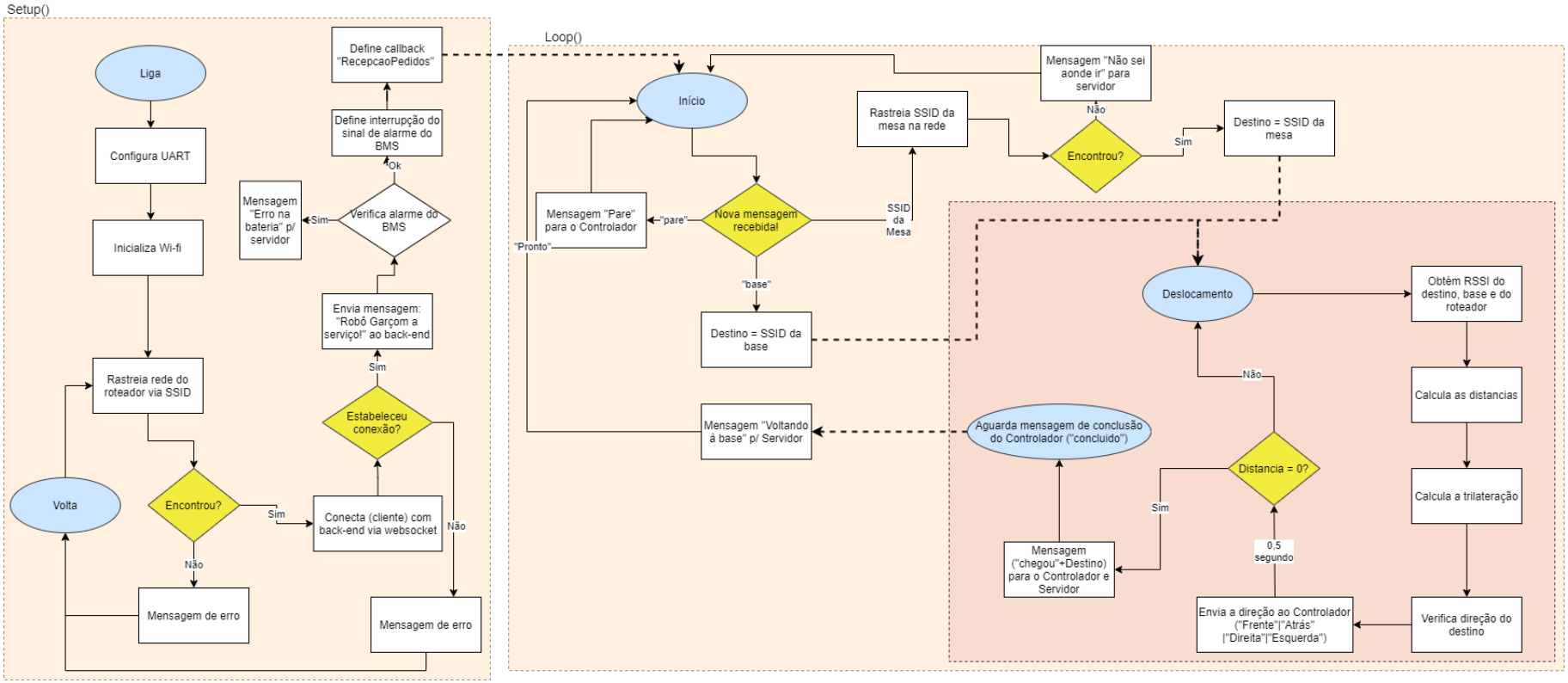
[1] As funções de medir a direção do robô (bússola) e de enviar a direção para o Controlador via UART não puderam ser feitas a tempo.

Figura 24 – Diagrama de fluxograma para a lógica do Controlador.



Fonte: Autores.

Figura 25 – Diagrama de fluxograma para a lógica do Comunicador.



Fonte: Autores.

6.7 Plano de Fabricação

São descritos os processos necessários para a montagem e configuração fundamental do robô.

6.7.1 PCBs

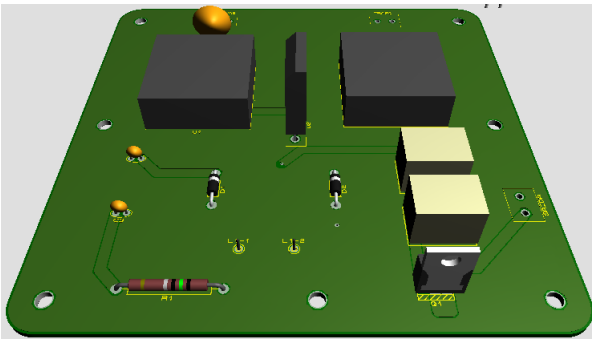
A partir dos esquemáticos das figuras **??** e **??** foram desenhados layout para essas placas.

Os arquivos encontram-se no repositório [Gabriel, Letícia e Diogo 2021].

- Alimentação

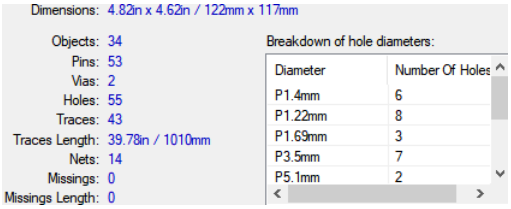
Nas Figura 26, Figura 27 e Figura 28 é mostrada a placa com o componentes de alimentação elétrica do robô, incluindo Unidade secundaria (placa P2 da Figura 17).

Figura 26 – Modelo 3D da placa de alimentação.



Fonte: Autores.

Figura 27 – Dimensões da placa para alimentação.



Dimensions: 4.82in x 4.62in / 122mm x 117mm

Objects:	34
Pins:	53
Vias:	2
Holes:	55
Traces:	43
Traces Length:	39.78in / 1010mm
Nets:	14
Missings:	0
Missings Length:	0

Breakdown of hole diameters:

Diameter	Number Of Holes
P1.4mm	6
P1.22mm	8
P1.69mm	3
P3.5mm	7
P5.1mm	2

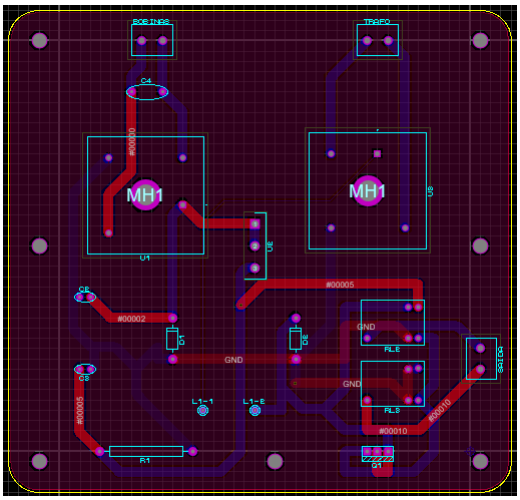
Fonte: Autores.

- Comunicador

Nas Figura 29, Figura 30 e Figura 31 é mostrada a placa com integração do comunicador (placa P3 da Figura 18).

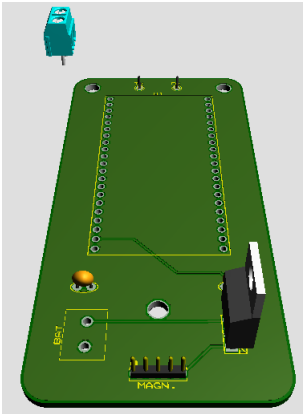
- Placa-mãe

Figura 28 – Layout de alimentação.



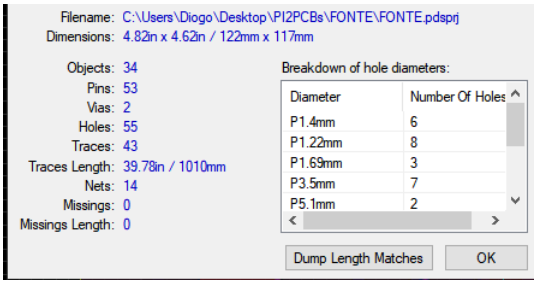
Fonte: Autores.

Figura 29 – Modelo 3D da placa para o comunicador.



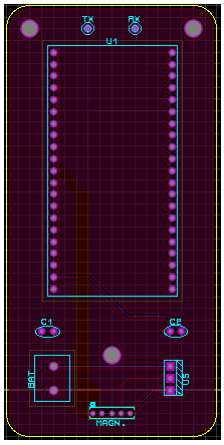
Fonte: Autores.

Figura 30 – Dimensões da placa para o comunicador.



Fonte: Autores.

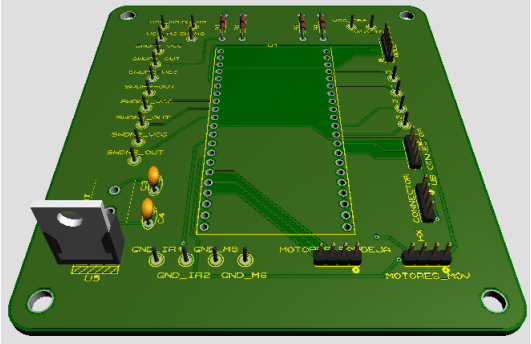
Figura 31 – Layout da placa para o comunicador.



Fonte: Autores.

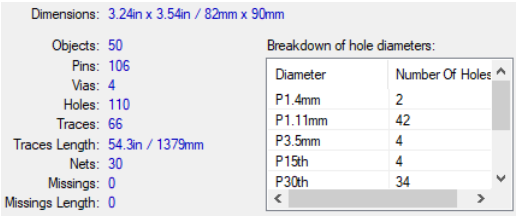
Nas Figura 32, Figura 33 e Figura 34 é mostrada a placa que integra todos os componentes principais, incluindo o controlador (placa P4 da Figura 19).

Figura 32 – Modelo 3D da placa para o controlador.



Fonte: Autores.

Figura 33 – Dimensões da placa-mãe.



Dimensions: 3.24in x 3.54in / 82mm x 90mm

Objects:	50
Pins:	106
Vias:	4
Holes:	110
Traces:	66
Traces Length:	54.3in / 1379mm
Nets:	30
Missings:	0
Missings Length:	0

Breakdown of hole diameters:

Diameter	Number Of Holes
P1.4mm	2
P1.11mm	42
P3.5mm	4
P15th	4
P30th	34

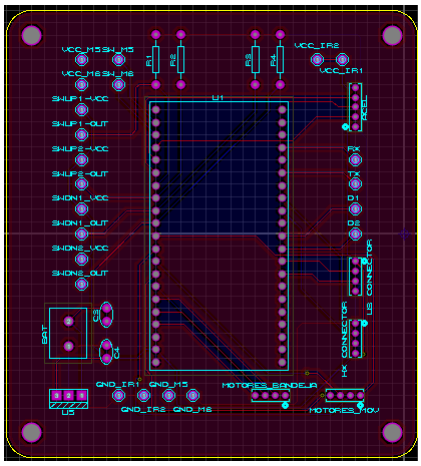
Fonte: Autores.

- *Driver* dos motores de locomoção

Nas figuras 35, 36 e 37 é mostrada a placa do *driver* para acionamento das rodas (placa P6 da 20), as quais foram dimensionadas para permitir o alimentação dos motores com corrente de 10A contínuos, com picos de 15A.

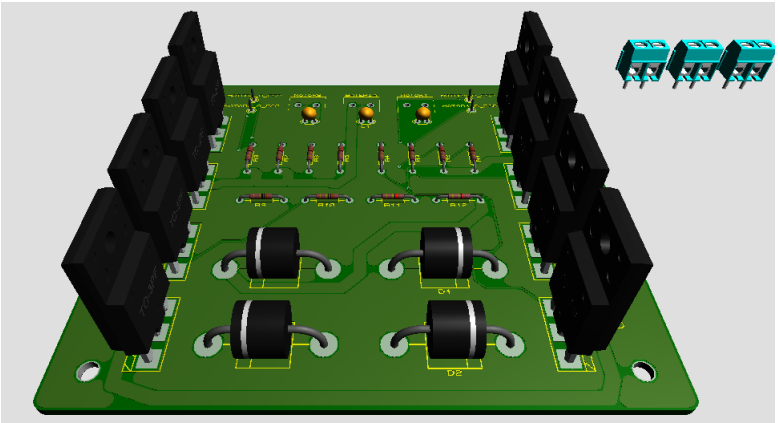
- Driver dos motores da bandeja

Figura 34 – Layout da placa-mãe.



Fonte: Autores.

Figura 35 – Modelo 3D da placa de controle dos motores das rodas.



Fonte: Autores.

Figura 36 – Dimensões da placa para movimento.

Dimensions: 4.42in x 3.83in / 112mm x 97mm

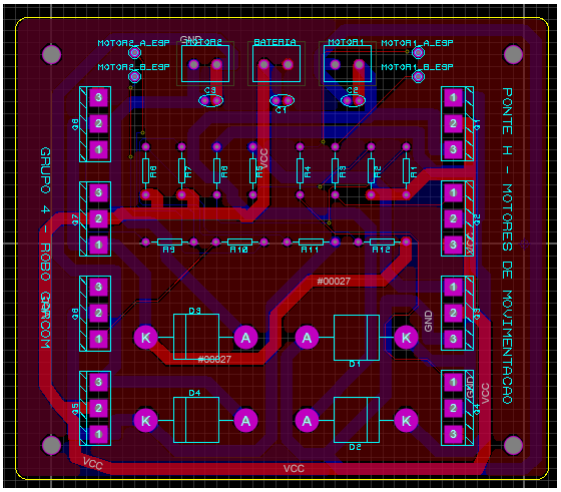
Objects: 51
Pins: 76
Vias: 7
Holes: 83
Traces: 85
Traces Length: 90.59in / 2301mm
Nets: 24
Missings: 0
Missings Length: 0

Breakdown of hole diameters:

Diameter	Number Of Holes
P1.4mm	6
P3.5mm	4
P15th	7
P30th	34
P60th	8

Fonte: Autores.

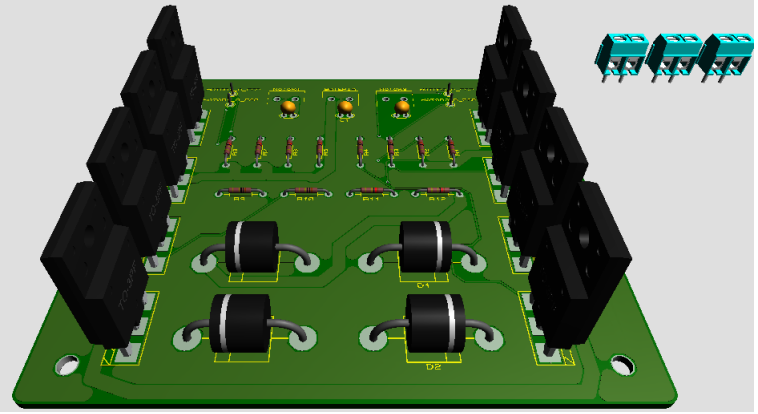
Figura 37 – Layout da placa de controle dos motores das rodas.



Fonte: Autores.

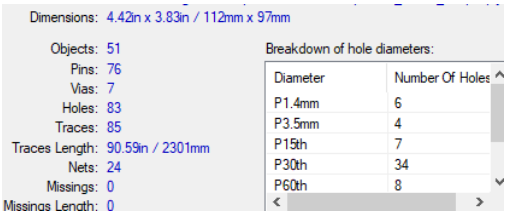
Nas Figura 38, Figura 39 e Figura 40 é mostrada a placa do *driver* para acionamento da bandeja (placa P7 da Figura 21), sendo que o circuito é o mesmo da placa de *driver* dos motores das rodas..

Figura 38 – Modelo 3D da placa de controle dos motores da fuso.



Fonte: Autores.

Figura 39 – Dimensões da placa para o fuso.



Dimensions: 4.42in x 3.83in / 112mm x 97mm

Objects:	51
Pins:	76
Vias:	7
Holes:	83
Traces:	85
Traces Length:	90.59in / 2301mm
Nets:	24
Missings:	0
Missings Length:	0

Breakdown of hole diameters:

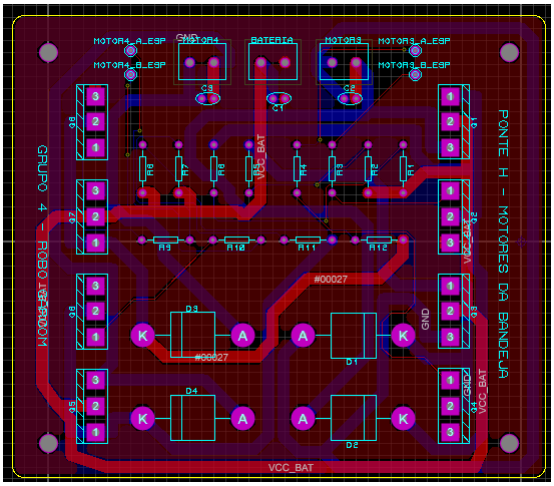
Diameter	Number Of Holes
P1.4mm	6
P3.5mm	4
P15th	7
P30th	34
P60th	8

Fonte: Autores.

6.7.2 Bobinas

Para alcançar os parâmetros desejados, a bobina deve ser enrolada em uma máquina específica de confecção das bobinas para respeitar o espaçamento de projeto. Este compo-

Figura 40 – Layout da placa de controle dos motores do fuso.



Fonte: Autores.

nente deve ter a fabricação terceirizada para o projeto.

6.7.3 Estrutura da base de carregamento

A case de carregamento do robô é a estrutura que comporta a unidade 1, bobina primária de carregamento, comunicador e o cabo de força para conexão com a rede do estabelecimento.

Assim, esta é a região onde o robô irá aguardar os pedidos antes de enviá-los para as mesas. A base é uma caixa com dimensões de 65 cm de altura por 60 cm de largura e 20 cm de profundidade. Ela deve ser parafusada na parede, de maneira a manter a bobina existente no robô, isso proporciona a melhor eficiência no carregamento possível.

Sua estrutura é composta por quatro placas de ACM, a placa de circuito denominada unidade 1, a bobina primária, e um cabo para conexão na rede do estabelecimento. A base apresenta três tipos de parafusos diferentes. O responsável pela fixação na parede é parafuso fenda + philips maquina de cabeça escareada, de aço inoxidável $2 \times 1,5 \times 25$. O outro modelo empregado é para as fixações das placas e da bobina, sendo um parafuso philips máquina cabeça escareada, aço inoxidável $2 \times 0,8 \times 4$. Por fim os parafusos para a unidade 1, que são quatro parafusos, fenda máquina cabeça panela, aço inoxidável $1 \times 0,8 \times 3,5$. O modelo tridimensional da base pode ser visualizado pelas Figura 83, Figura 84 e Figura 85.

As vistas cotadas encontram-se no final do apêndice Apêndice J.

6.8 Plano de Montagem

Esses procedimentos são necessários para o funcionamento do produto.

6.8.1 Instalação da antena de wi-fi

A uso da antena, embora não obrigatório, melhora bastante a recepção de sinal, e possibilita um raio de alcance maior.

- Ela deve ser ligada à Esp32 do comunicador do robô. É o item 1 especificado lista da Figura 60. Para conecta-la através de um cabo, também chamado "*pigtail*", tal qual é o item 2 da lista .

Figura 41 – Cabo para a antena, plug IPEX UFL à esquerda e SMA à direita.

- A antena é encaixado num furo na carenagem, e no outro lado desse furo deve ser conectado a ela a ponta SMA do cabo. A outra ponta, UFL, deve ser conectada na região da Esp32 mostrada na Figura 42.

Figura 42 – Região para plugue do cabo, próxima do chip do microcontrolador.

6.8.2 Configuração do controlador

O código do controlador pode ser implementado utilizando a IDE do Arduino.

- Abra a IDE vá em Arquivo > Preferências , coloque em URLs adicionais de Gerenciadores de Placas: http://arduino.esp8266.com/stable/package_esp8266com_index.json;
- Em seguida acesse Ferramentas > Placa > Gerenciador de Placas , pesquise por ESP e clique em instalar;

- Após a conclusão da instalação copie o código do controlador em um novo sketch , vá em Ferramentas > Placas > ESP32 Arduino e selecione a placa 'ESP32 Wrover Module'.
- Conecte a ESP32 pelo USB ao computador, vá em Ferramentas > Porta > COM "X", selecione a porta em que a ESP32 estiver conectada ao seu dispositivo;
- Depois de realizar essas configurações clique em carregar no canto superior esquerdo o código será compilado e implementado na ESP;
- Após a conclusão desse processo a ESP estará pronta para uso.

6.8.3 Sensor de peso (Calibração)

Para realizar a calibração do sensor de peso utilize o código subseção I.2.2.1, após implementar o código utilize o monitor serial para verificar a leitura da balança em seguida realize as seguintes etapas:

- Coloque um peso com o valor conhecido em cima da estrutura da balança(espere até que o valor no monitor serial se estabilize).
- Calcule a média dos 5 últimos valores apresentados no monitor serial após a estabilização(o valor apresentado pode ser negativo).
- Divida o valor obtido na média pelo peso de teste que foi colocado inicialmente na balança(o peso deve ser convertido para quilogramas antes de realizar a divisão).
- Insira o valor resultante da divisão em na parte do código que estiver escrito "scale.set_scale();" (coloque o valor inteiro sem ponto ou vírgula).
- Implemente novamente o código com o valor de escala atualizado.
- Coloque novamente na estrutura o peso utilizado para a calibração.
- Verifique o valor obtido no monitor serial ele deverá ser igual ao peso de teste.
- Caso exista alguma divergência entre o valor de referência e o lido que ultrapasse poucas casas decimais, repita o processo.

6.8.4 Configuração do comunicador

Importante! O pino para o alarme de BMS, além de ser conectado na saída de alarme deste, deve conectado com um resistor no pino de 3,3V!

A configuração da IDE é semelhante à da subseção 6.8.2, porém o modelo de placa a ser selecionado é 'DOIT ESP32 DEVKIT V1'.

Antes da compilação e implementação de seu código 'módulo_comunicador.ino', devem ser feitos as seguintes alterações:

- Instalar a biblioteca 'ArduinoWebsockets.h' presente no repositório.
- Nos campos 'LOGIN_CENTRAL' e 'SSID_CENTRAL' devem ser declarados com o SSID do roteador Central do estabelecimento, o campo 'password' com a chave de acesso.
- Nos campos 'WEBSOCKETS_SERVER_HOST' e 'WEBSOCKETS_SERVER_PORT' devem ser declarados com o endereço de IP do servidor/computador central do estabelecimento e sua porta de conexão, respectivamente.
- Ainda no mesmo código, as coordenadas de localização (em metros) de cada dispositivo, bem como a base de carregamento, roteador central e *tablets* das mesas devem ser declaradas.

A placa para a base é a mesma da subseção 6.8.2 e seu código é 'base.ino'.

6.8.5 Cabeamento Elétrico

O dimensionamento da bitola da fiação foi traçado a partir de referências da norma ABNT NBR 5410, além de tabelas de fabricantes, que se baseiam na corrente máxima do circuito envolvido. As áreas das seções definidas para os condutores conectados aos diferentes circuitos/componentes são apresentados na Tabela 12. Os comprimentos necessários para cada condutor estão presentes na Figura 16.

Tabela 12 – Especificações da fiação.

Circuito	Corrente (A)	Seção (mm²)
Base de carregamento	24	6
BMS/Bateria	24	6
Motores de movimentação	7,1	0,75
Motores atuadores	2,5	0,5
Demais conexões	Menos de 1	0,5

6.8.6 Conexões do Cabeamento Elétrico

Para o Robô Garçom foram utilizados quatro tipos de conectores. Esses conectores tem a função de auxiliar e facilitar no desacoplamento dos componentes, para eventuais trocas ou reparos.

Para as conexões com a rede elétrica do estabelecimento, são utilizados dois Plugs Machos de três pinos. As outras conexões são feitas por conectores tipo Wago Borne modelos SPL: 2, 42 e 622.

6.9 Testes

Segue descrito o plano de testes para os elementos abordados nesse capítulo.

6.9.1 Localizador

Tabela 13 – Localizador

Objetivo do teste	Verificar se o robô vai até o destino pretendido
Técnica	Enviar os nomes de todos os dispositivos de onde ele deve ir: a base e os *tablets* das mesas. O teste é via mensagens pelo servidor central. O nome é enviado e deve-se esperar ele concluir o percurso e voltar para enviar o próximo destino.
Critério de Finalização	Ele chega corretamente em todos pontos testados.
Considerações Especiais	-

6.9.2 Sensor de Peso

Tabela 14 – Sensor de Peso

Objetivo do teste	Avaliar o estado da célula de carga
Técnica	Conectar um multímetro (ligado em Ohms) e testar a resistência de entrada e saída verificando se os valores estão de acordo com o datasheet e em seguida conectar o multímetro aos fios de saída da célula de carga, alimentar ela com 5V e colocar uma carga de teste especificada no datasheet na célula de carga, no multímetro será lido um valor de tensão.
Critério de finalização	A medida de tensão lida no multímetro deverá ser similar a descrita no datasheet quando há carga e quando não há.
Considerações Especiais	Para valores de tensão lidos como zero quando há carga de teste na célula de carga e a fonte de alimentação funcionando corretamente a célula de carga necessita ser trocada.

6.9.3 Baterias

Tabela 15 – Teste de baterias

Objetivo do teste	Avaliar o estado das baterias

Técnica	Medição da tensão em circuito aberto após o tempo necessário para carregamento pleno.
Critério de finalização	Tensões abaixo de 12,2 V indicam o comprometimento da bateria
Considerações Especiais	Pela possibilidade do registro de tensões de flutuação ou da contribuição da temperatura, tensões acima de 13V podem ser registradas. Um novo teste pode ser realizado posteriormente.

6.9.4 Bobinas

Tabela 16 – Teste das Bobinas

Objetivo do teste	Avaliar a bobina pelo parâmetro de indutância
Técnica	Utilizando um medidor LCR, usar a escala de indutância da ordem de grandeza de $100\mu H$. Ligar um resistor em série a um dos terminais do indutor e realizar a medição ligando o fio negativo do medidor na ponta do resistor e o fio positivo no outro terminal do indutor.
Critério de finalização	Verificar se a indutância mostrada no visor está próxima dos 100μH
Considerações Especiais	Ambas bobinas possuem a mesma dimensão. Realizar o teste separado para ambas. Se os valores estiverem semelhantes, garante a mesma indutância mútua de projeto.

6.9.5 Sistema elétrico

Tabela 17 – Testes do sistema elétrico

Objetivo do teste	Avaliar as conexões.
Técnica	Medição de corrente e voltagem pelo circuito utilizando os métodos descritos na norma ABNT NBR IEC 61189-3:2012.
Critério de finalização	Seguir as tensões e correntes descritas no diagrama elétrico para as análises.
Considerações Especiais	Aceitar variação de corrente na unidade 2 das bobinas.

6.9.6 Comunicador

Tabela 18 – Teste do Comunicador

Objetivo do teste	Verificar se os dispositivos se conectam corretamente com a rede
Técnica	Ligar o robô no alcance da rede wi-fi e conferir se o servidor central recebe a mensagem “Robô Garçom a serviço”.
Critério de finalização	A mensagem é recebida em até 30 s.
Considerações Especiais	Raio de alcance máximo de aproximadamente 30 m (sem paredes no caminho).

7 *Software*

Visando automatizar o sistema de atendimento ao cliente, é proposto um *software* para gerenciamento de pedidos do restaurante em que o Robô Garçom irá operar. O sistema funcionará da seguinte forma: o *software* enviará os pedidos feitos pelos clientes para a cozinha, gerando um ID único para cada. Assim que o pedido é preparado, a cozinha, através de um *tablet*, denominado computador central, exclusivo para esse serviço, escolhe qual robô será o responsável para realizar a entrega e digita o ID do pedido. Posteriormente, o robô, que receberá o ID, será abastecido e estará pronto para entregar na mesa. Os IDs contém as informações daquilo que é constituído o pedido e em qual mesa ele deve ser entregue, sendo esse último o de máxima importância para o robô, pois será utilizado como identificador para localizar a mesa a qual pertence o pedido.

Para iniciar a definição do escopo do projeto de *software*, a equipe definiu a metodologia que seria usada para a organização do núcleo, e fez um levantamento dos objetivos do sistema, que estão detalhados no Apêndice H. Posteriormente, foram utilizadas técnicas de elicitação e priorização para definir os requisitos funcionais e não funcionais do sistema. O processo de elicitação e priorização estão especificados na seção H.2.

Com o objetivo de organizar de maneira mais eficiente e clara as informações relacionadas ao núcleo de *software*, foi criada uma organização no GitHub para hospedar os repositórios para o projeto Robô Garçom. Dentro da organização, existem três repositórios dedicados aos artefatos de *software*, são eles: documentação, *Frontend* e *Backend* do sistema. Para a parte de documentação, também foi criada uma WIKI, para facilitar a visualização de todos os documentos relacionados ao *Software* do projeto.

7.1 Arquitetura do *Software*

O *software* proposto para o projeto será uma aplicação estruturada no **modelo arquitetural MVC** (*Model-View-Controller*), e permitirá que os clientes, por meio de um *tablet* disponível em cada mesa, visualizem o cardápio, façam seus pedidos e realizem o pagamento, através das plataformas Mercado Pago® ou PIX.

A arquitetura MVC permite dividir as funcionalidades do sistema em camadas, o que facilita a resolução de um problema maior [wagon 2020]. As camadas consistem em:

- Modelo *(Model)*: Sua responsabilidade é gerenciar e controlar a forma como os dados se comportam por meio das funções, lógica e regras de negócios estabelecidas. É a camada que se comunica com o banco de dados, recebe as informações do Controlador, válida se ela está correta ou não e envia a resposta mais adequada.

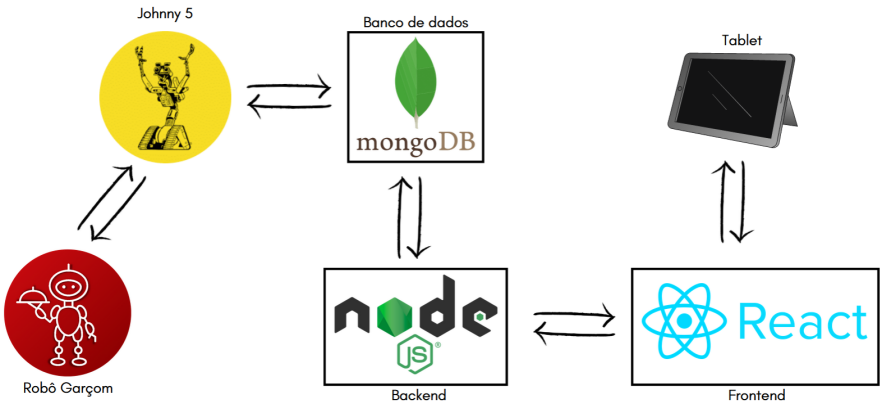
- Visualização *(View)*: Responsável por tudo que o usuário final visualiza, toda a interface e recursos ligados a aparência como mensagens, botões ou telas [NET 2011]. Transmite questionamentos ao *controller* e entrega as respostas obtidas ao usuário.

- Controlador *(Controller)*: Responsável por controlar todo o fluxo de informação que passa pelo sistema. É a camada que executa uma regra de negócio e repassa a informação para a visualização.

A arquitetura MVC mostrou-se a mais adequada ao projeto, por permitir que o *software* seja desenvolvido e entregue mais rapidamente, além de deixar o código mais limpo e de fácil entendimento.

O diagrama de blocos na Figura 43 representa a arquitetura do sistema, integrando o **Robô Garçom** a todo o sistema de automação do restaurante.

Figura 43 – Diagrama de blocos.



Fonte: Autores.

Linguagem

A aplicação será desenvolvida em **JavaScript(JS)**, uma linguagem leve, interpretada e baseada em objetos com funções de primeira classe, ou seja, os objetos são criados dinamicamente, destruídos, transmitidos para uma função, retornados como um valor e têm todos os direitos que outras variáveis da linguagem de programação possuem. JS é conhecido como a linguagem de *script* para páginas Web, mas também é utilizada em vários outros ambientes sem *browser*, como por exemplo o *Node.js*.

De acordo com o [Overflow 2021], o JS foi eleito pelo oitavo ano seguido a linguagem de programação mais usada no mundo. Sua popularidade é uma consequência por ela ser uma linguagem muito flexível e multifacetada.

Uma das vantagens de se utilizar o JS é a existência de um *Framework* voltado ao controle de robôs, denominado Johnny Five, para integração direta com os controladores

utilizados no equipamento, e que é escrito em JavaScript, o que facilita a produção do software.

Frontend e *Backend*

A biblioteca ***React*** será utilizada para o desenvolvimento da interface do usuário (*frontend*). Trata-se de de uma biblioteca de código aberto escrita em JavaScript, e será utilizada no desenvolvimento do sistema, em conjunto com o módulo ***React Native***, que possibilita o desenvolvimento de aplicativos para Android e iOS.

Já para o *backend* será utilizado o ***Node.JS***. *Node* é uma plataforma de aplicação, na qual os programas são escritos em JS e compilados, otimizados e interpretados pela Máquina Virtual V8, que é a mesma usada pelo Google para executar JavaScript no navegador Chrome. O resultado deste processo é entregue como código de máquina *server-side*, tornando o Node muito mais eficiente na sua execução e consumo de recursos [Escudelario 2018]. O Node.JS será utilizado em conjunto com o *Framework Express.JS*.

Banco de dados

O banco de dados utilizado para a aplicação será o **MongoDB**, um banco do tipo *NoSQL*. Por ser um banco de dados do tipo *NoSQl*, o MongoDB é orientado a documentos, ou seja, armazena todos os dados em arquivos do tipo JSON, ao contrário do modelo de dados relacional que armazena os dados em linhas e colunas. O MongoDB é ágil e escalável horizontalmente e também permite consultas diretas com JavaScript.

Todas as justificativas para as decisões tomadas com relação à arquitetura do software para o Robô Garçom são descritas na seção H.3.

7.2 Arquitetura da informação

Segundo definição do *Information Architecture Institute*, a Arquitetura da Informação (AI) é "a prática de decidir como organizar as partes de alguma coisa de modo a torná-la compreensível" [Institute 2020]. Em outras palavras, a AI ajuda a equipe de software a determinar como se dará a organização do sistema, de maneira que este seja compreensível e que torne eficaz a interação entre o usuário e a interface com a qual ele está interagindo.

A aplicação das heurísticas de Nielsen e a prototipagem foram os métodos escolhidos para construir a arquitetura da informação da aplicação, sendo esta última executada em duas fases: protótipo de baixa fidelidade e de alta fidelidade.

Heurísticas de Nielsen

Visando a criação de um layout funcional para o sistema, foram utilizadas como base algumas das dez heurísticas criadas por Jakob Nielsen, cientista da computação, que ajudam a projetar uma boa interface e por consequência uma ótima experiência de uso. As heurísticas selecionadas e suas aplicações na interface do sistema estão descritas na Tabela 19.

Tabela 19 – Heurísticas de Nielsen e suas aplicações no *software*.

Nº	**Heurística**	**Aplicação**
1	Visibilidade do Status do sistema	O sistema manterá o usuário constantemente informado sobre suas ações, como confirmação de adição e exclusão de item, e confirmação de pagamento.
3	Controle e liberdade para o usuário	Esta heurística será aplicada através dos botões de voltar e cancelar, que dão opção ao usuário de desfazer algumas de suas ações.
4	Consistência e padronização	A interface segue um padrão de design, com cores e fontes iguais em todas as telas, e distribuição de elementos semelhantes entre as telas.
5	Prevenção de erros	Esta heurística será aplicada exclusivamente na tela de confirmação de exclusão de item.
6	Reconhecimento em vez de recordação	Aplicada ao cardápio, implementando-o de maneira em que o usuário não necessite de grande esforço para identificar os itens do cardápio.
8	Estética e design minimalista	O sistema terá uma interface simples, sem muitas informações, facilitando a compreensão do usuário quanto às funcionalidades disponíveis.

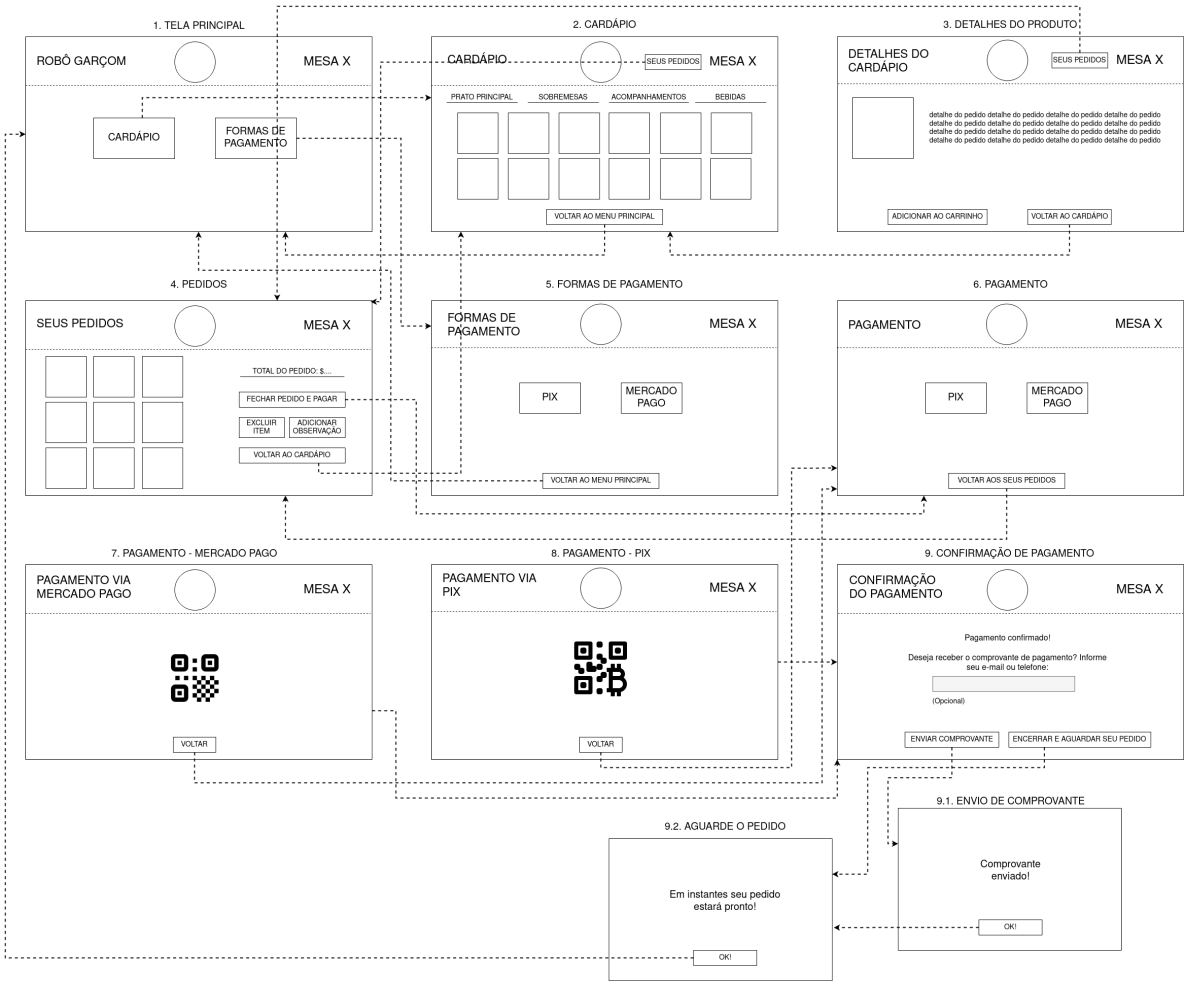
Protótipo de baixa fidelidade

O objetivo da criação do protótipo de baixa fidelidade é apresentar telas e elementos de uma forma não tão próxima com o produto final. Essa prática permite a discussão sobre detalhes mais superficiais no momento das tomadas de decisão a respeito do design do software e a identificação de possíveis problemas de usabilidade e requisitos.

O design das telas do sistema foi discutido em reunião com a equipe de *software*, em que todos deram seus pontos de vista acerca dos pontos iniciais da interface do produto em termos visuais e estéticos, e os aspectos básicos da experiência do usuário. Durante a reunião, alguns desenhos foram feitos com base nas sugestões dos membros, e podem ser visualizados na WIKI do projeto.

Após a equipe de *software* entrar em consenso quanto à aparência das telas, foi desenvolvida uma segunda versão do protótipo de baixa-fidelidade (Figura 44), na plataforma *draw.io*, adicionando mais telas para complementar a navegabilidade do aplicativo. Além das telas, também foi desenhado o fluxo das funcionalidades do aplicativo.

Figura 44 – Protótipo de baixa fidelidade.



Fonte: Autores.

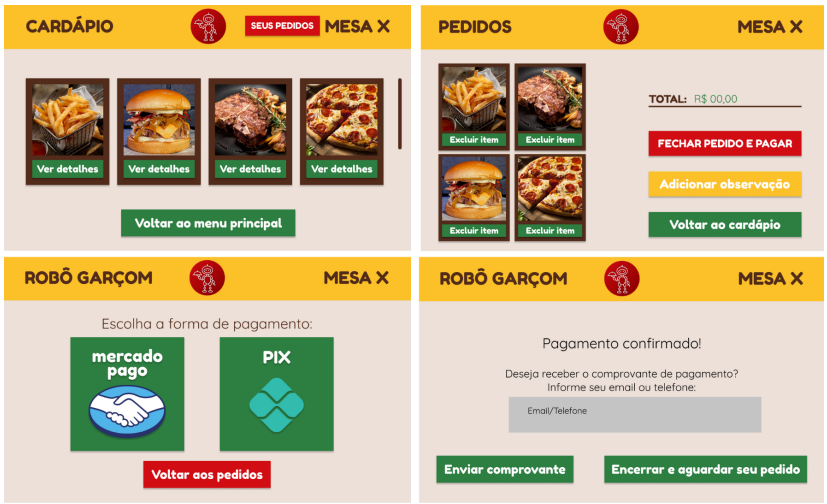
Protótipo de alta fidelidade

A prototipação de alta fidelidade apresenta as telas do sistema de forma mais próxima com o produto final. Esse processo permite o debate sobre detalhes aprofundados no momento das tomadas de decisão a respeito do design do *software* e a identificação de possíveis problemas de usabilidade, com base nas interações dos potenciais usuários do sistema.

O protótipo de alta fidelidade foi desenvolvido na plataforma **Figma**, por questões de facilidade na utilização da ferramenta e pela diversidade de funcionalidades disponíveis. O protótipo tem como objetivo abranger, além da interface do produto em termos visuais e estéticos, os aspectos da experiência do usuário em termos de interações, fluxo e comportamento. A Figura 45 mostra algumas telas do protótipo de alta fidelidade. Para acessar o

protótipo de alta fidelidade na plataforma Figma basta clicar na Figura 46.

Figura 45 – Telas do Protótipo de alta fidelidade.



Fonte: Autores.

Figura 46 – Protótipo de Alta fidelidade do aplicativo para o cliente.

Fonte: Autores.

Além do protótipo de alta-fidelidade do aplicativo que estará disponível nos *tablets* nas mesas para acesso dos clientes, também foi desenvolvido um protótipo para o administrador do restaurante. O aplicativo para o administrador, que será instalado no *tablet* central na cozinha, terá as opções de gerenciar o cardápio, cadastrar mesa e iniciar o atendimento do restaurante. Para acessar o protótipo de alta para o administrador, clique na Figura 47.

Figura 47 – Protótipo de Alta fidelidade do aplicativo para o administrador do restaurante.

Fonte: Autores.

7.3 Inovações

Com o intuito de apresentar um projeto inovador ao mercado consumidor, o Robô Garçom será equipado com um algoritmo de determinação de trajeto para otimizar cada entrega de pedido, gerando um aumento de autonomia e reduzindo o tempo de transporte do pedido da cozinha até a mesa do cliente, evitando um desperdício de energia e que o cliente espere por um período de tempo a mais que o previsto.

Utilizando a biblioteca **Johnny Five**, dedicada à programação de robôs e automação de componentes eletrônicos, o Robô Garçom terá seus componentes integrados entre si para gerar uma virtualização do ambiente do restaurante, possibilitando que o Robô se locomova sem a intervenção de terceiros.

Algoritmo de melhor caminho

O **Algoritmo de Dijkstra** foi escolhido como o algoritmo de melhor caminho. Concebido pelo cientista da computação holandês Edsger Dijkstra em 1956, este algoritmo soluciona o problema do caminho mais curto num grafo dirigido ou não dirigido com arestas de peso não negativo, em tempo computacional

$$O(E + V \log V)$$

onde V é o número de vértices e E é o número de arestas [Frana e Misa 2010].

Framework Johnny Five

É um *Framework* de programação de código aberto baseado em protocolo Firmata, o qual faz a comunicação com microcontroladores a partir de um *software* em um computador/*tablet*/*smartphone*.

Os *softwares* desenvolvidos utilizando Johnny-Five podem realizar a conexão com Arduino, Electric Imp, Beagle Bone, Intel Galileo & Edison, Linino One, Pinoccio, pcDuino3, Raspberry Pi, Particle / Spark Core & Photon, Tessel 2 e TI Launchpad.

A Johnny Five é escrita em JavaScript, e será implementada na camada *Controller* do *backend* do *software*.

7.4 Modelagem do sistema

A modelagem é uma das principais atividades que levam à implementação de um bom *software* [DevMedia 2011]. Construímos modelos para comunicar a estrutura e o comportamento desejados do sistema, visualizar e controlar a arquitetura do mesmo e compreender melhor o sistema que estamos elaborando.

A modelagem de *software* utiliza vários modelos para projetar um sistema. Um modelo é uma simplificação da realidade, criado para facilitar o entendimento de sistemas complexos. As subseções a seguir apresentam os diagramas desenvolvidos para facilitar o entendimento do sistema do Robô Garçom.

Visão de casos de uso

A exploração dos casos de uso da aplicação visa descrever em detalhes como será o uso de cada atividade no sistema. Os casos de uso identificados para a aplicação do Robô Garçom são descritos na Tabela 20.

Tabela 20 – Casos de Uso.

ID UC	Descrição
UC01	Manter Usuário do Restaurante
UC02	Manter Cardápio do Restaurante
UC03	Visualizar o Cardápio
UC04	Visualizar Item do Cardápio
UC05	Manejar Itens no Carrinho
UC06	Adicionar observações aos Itens do Pedido
UC07	Incrementar Itens do Pedido
UC08	Visualizar Itens do Pedido
UC09	Realizar pedido
UC10	Cancelar Pedido
UC11	Processar Pagamento
UC12	Visualizar pedidos pagos
UC13	Visualizar tempo restante de espera

O diagrama geral de casos de uso (Figura 48) apresenta a lógica de funcionamento do sistema.

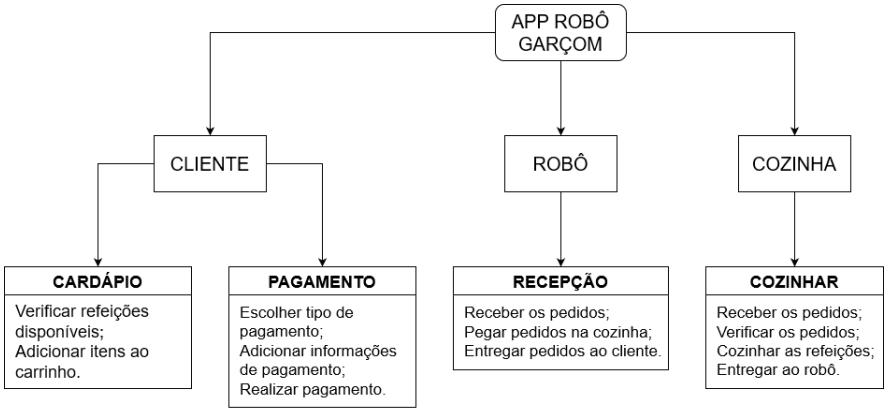
Figura 48 – Diagrama de casos de uso do sistema.

Fonte: Autores.

Diagrama de Arquitetura Funcional

Os diagramas de arquitetura de *software* são a forma de comunicar o planejamento da construção de um sistema de software ou como um sistema existente funciona. A arquitetura do sistema de *software* para o **Robô Garçom** está exemplificado na Figura 49.

Figura 49 – Diagrama de Arquitetura Funcional.



Fonte: Autores.

Diagrama de Classes

O diagrama de classes ilustra graficamente como será a estrutura do *software*, em nível micro ou macro e, como cada um dos componentes da sua estrutura estarão interligados. Ele tem como principal objetivo elucidar o ponto de vista estrutural do *Software*.

Considerando as possibilidades de aplicação do diagrama (Figura 50), este foi utilizado para diferentes contextos na produção do aplicativo do Robô Garçom, destacando:

- Design do Projeto de *Software*: Foi definido um modelo a ser seguido na fase inicial do projeto, quando ele ainda não existia, especificado antes do *software* ser desenvolvido efetivamente.
- Engenharia Reversa: Foi refletido a estrutura de código que criada no PC1, a partir do código fonte realizou-se a leitura automática das classes e suas relações para depois produzir-se o diagrama para compreensão da estrutura do *software* executável.
- Esboço de ideias: Foram desenhados modelos estruturais para alinhar as ideias.

Diagrama de Atividades

O Diagrama de Atividades é um diagrama comportamental, que especifica o comportamento do *software*. Através dele é dada a modelagem de partes do comportamento do sistema, ilustrando graficamente como será o funcionamento do *software*, definindo a execução do aplicativo e como será a atuação do sistema na realidade de negócio do projeto. O diagrama de atividades (Figura 51) demonstra o fluxo da realização de um pedido, principal atividade disponibilizada pelo sistema.

Diagrama de Componentes

Os diagramas de componentes são usados para visualizar a organização dos componentes do sistema e os relacionamentos de dependência entre eles. Fornece uma visão de alto nível dos componentes dentro de um sistema.

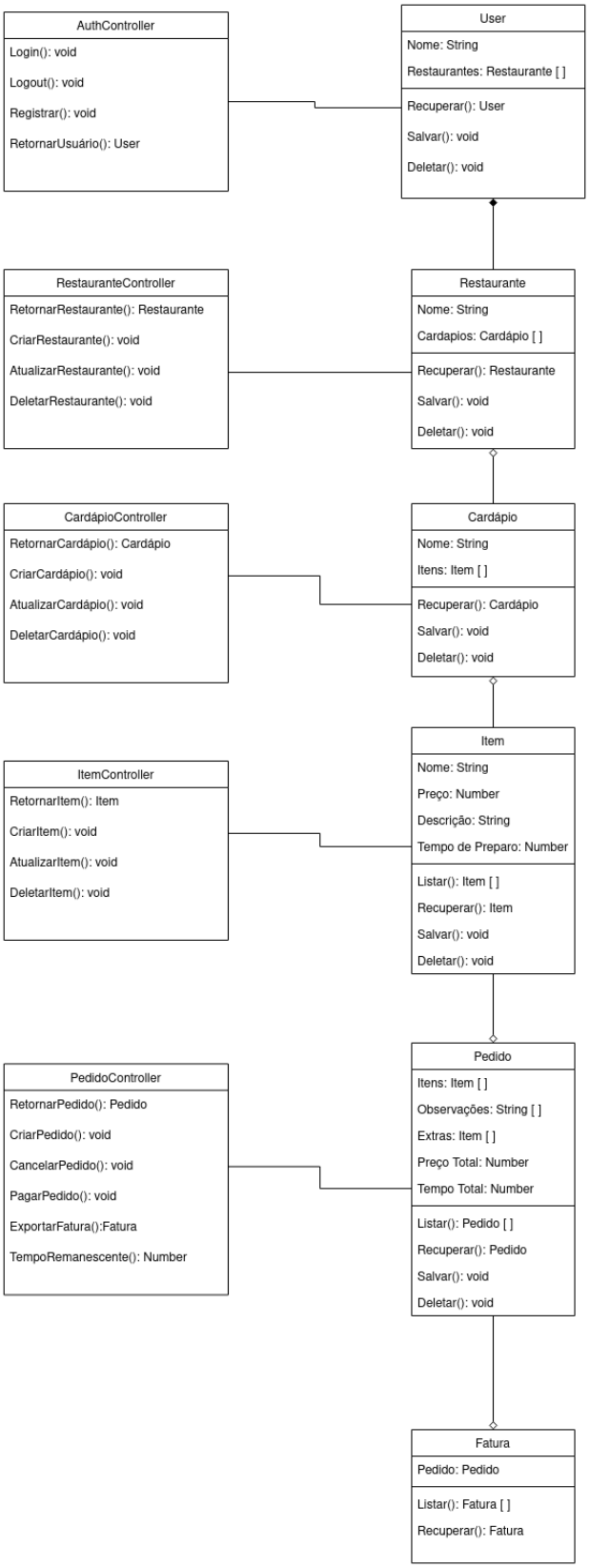
Os componentes podem ser um componente de *software*, como um banco de dados ou interface de usuário; ou um componente de *hardware*, como um circuito, microchip ou dispositivo; ou uma unidade de negócios, como fornecedor, folha de pagamento ou envio.

O diagrama de componentes (Figura 52) ilustra como se dará a organização de todo o sistema e suas dependências, os componentes podem ser lógicos ou físicos.

Diagrama de Pacotes

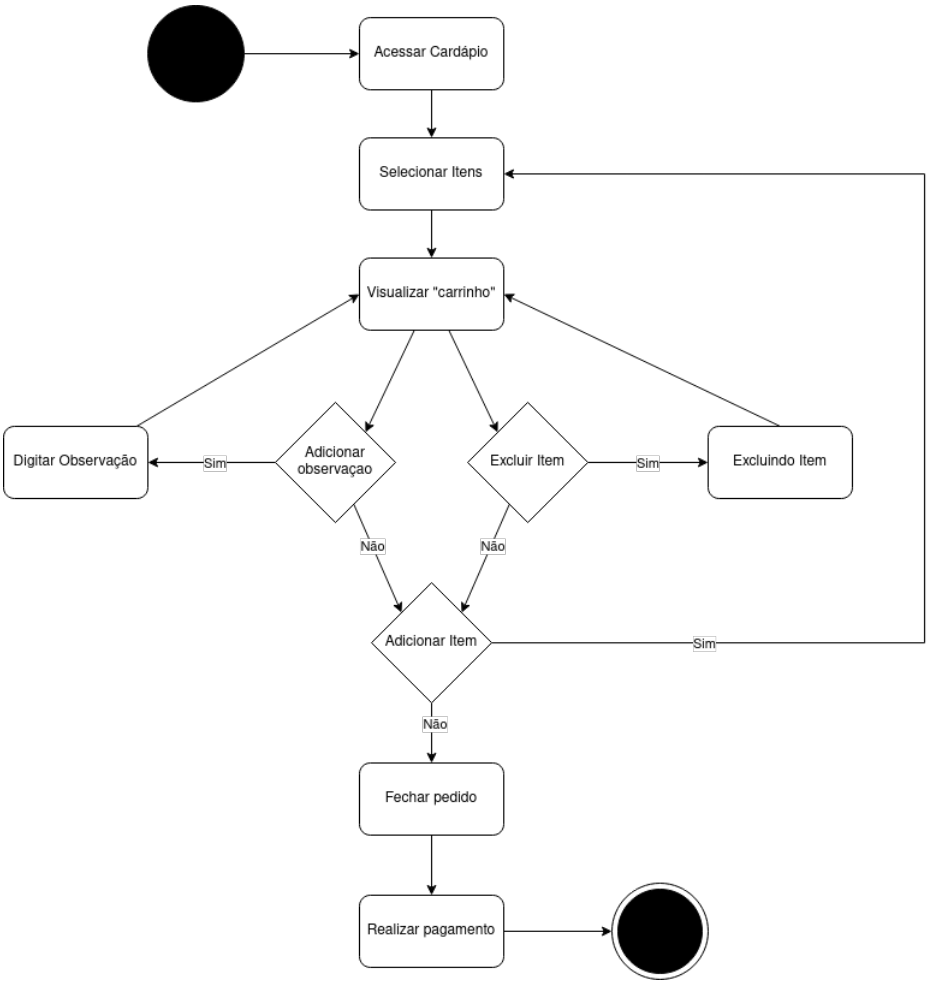
Os Diagramas de Pacotes têm como conceito agrupadores lógicos de pedaços dos sistemas com interação entre os pacotes, ou seja, pacotes podem depender de outros paco-

Figura 50 – Diagrama de Classes.



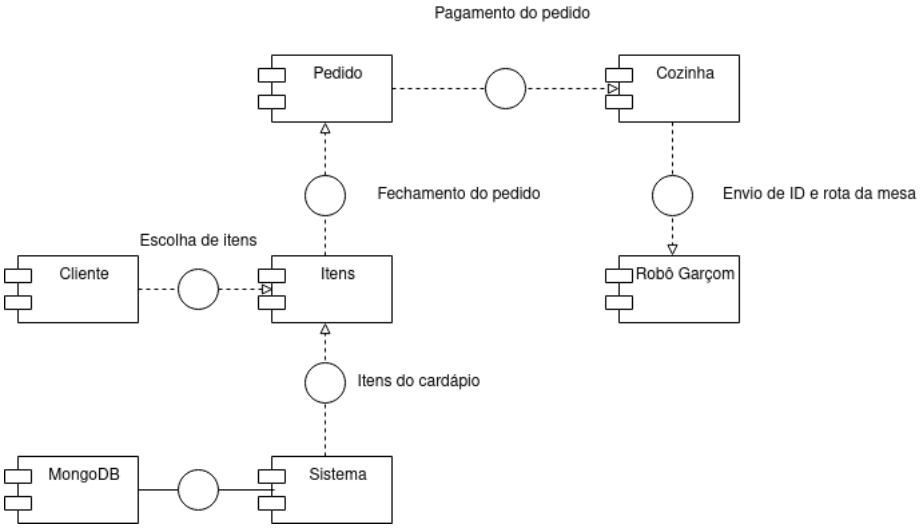
Fonte: Autores.

Figura 51 – Diagrama de Atividades.



Fonte: Autores.

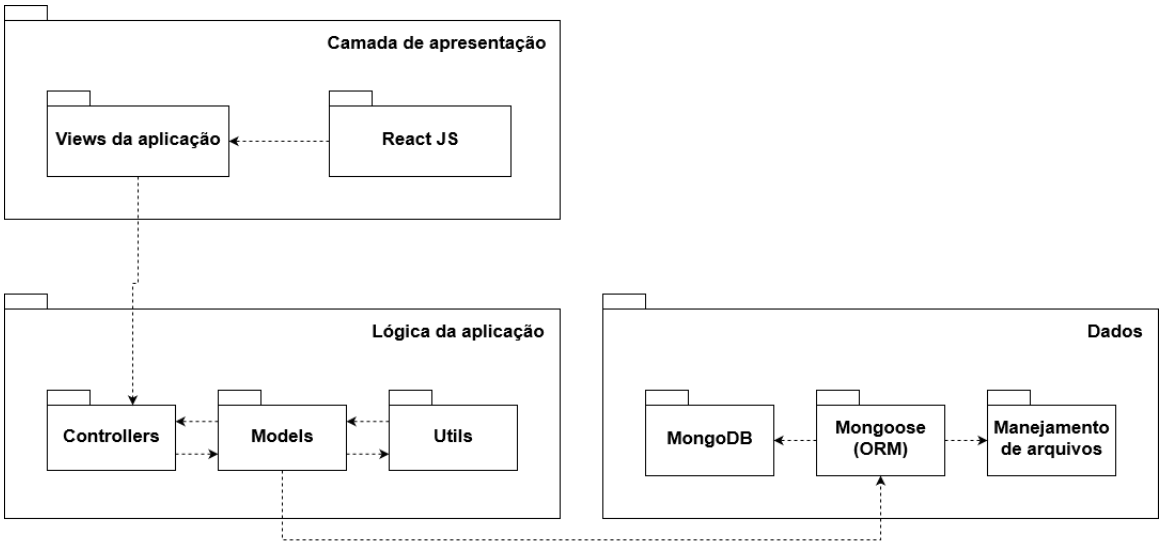
Figura 52 – Diagrama de Componentes.



Fonte: Autores.

tes. Esses elementos agrupados podem ser diagramas, classes, outros pacotes, entre outros. O diagrama de pacotes do sistema Robô Garçom (Figura 53) mostra a organização e disposição dos elementos e camadas do sistema.

Figura 53 – Diagrama de Pacotes.



Fonte: Autores.

A partir da modelagem do sistema, foi possível dar início ao desenvolvimento do *software*, partindo para a criação de um **Docker**, estruturação do banco de dados do sistema, desenvolvimento dos CRUDs (*Create, Rename, Update, Delete*) para o *backend*, e criação dos códigos para integração dos componentes do Robô com o computador central. Os códigos desenvolvidos para o sistema Robô Garçom e para a integração entre estão contidos no Apêndice I.

7.5 Testes

A equipe optou por aplicar testes unitários para validar parte do código do *software* do Robô Garçom. Teste unitário, ou de unidade, é um método onde testa-se unidades menores de um *software*, de modo isolado, para ver se todas funcionam adequadamente. O *backend* foi escolhido para passar por esses testes.

Os *endpoints* do *backend* foram testados, como se fossem uma espécie de *"black-box"*, para assegurar que o sistema está respondendo como esperado. *"Black-box"* refere-se ao teste de caixa preta, cujo objetivo é validar as entradas e saídas do sistema, aumentando a qualidade e, consequentemente, blindando o *software* de falhas.

O relatório dos testes, apresentado na Figura 54 aponta um grau de aceitação por volta de 90% em todo o código escrito para o *backend* até o momento, indicando uma robustez do sistema como um todo, minimizando as chances de ocorrerem falhas que impeçam o devido funcionamento da aplicação.

Figura 54 – Diagrama de blocos.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Line #s
All files	83.16	68.75	85.33	78.28	
src	85.71	75	60	81.81	
server.js	85.71	75	60	81.81	16-17
src/controllers	69.51	70	83.33	62.12	
...Controller.js	92.3	62.5	100	90.47	10,37
...Controller.js	67.21	75	83.33	60	...1,79-89,97-102
...Controller.js	40.47	75	60	26.47	...42,50-57,65-72
...Controller.js	91.42	75	91.66	88.88	6-9
src/middleware	79.31	57.69	85.71	73.91	
...middleware.js	93.75	70	100	92.3	17
...middleware.js	94.44	66.66	100	92.3	18
...middleware.js	40	37.5	50	25	4-20
...middleware.js	88.88	50	100	87.5	10
src/models	98.38	80	100	97.95	
Item.js	100	100	100	100	
Menu.js	100	75	100	100	1
Order.js	100	100	100	100	
Restaurant.js	100	75	100	100	1
User.js	94.44	100	100	93.75	35
src/routers	100	75	100	100	
auth.router.js	100	75	100	100	1
index.js	100	75	100	100	1
menu.router.js	100	75	100	100	1
order.router.js	100	75	100	100	1
...ant.router.js	100	75	100	100	1
src/services	90.9	100	100	90	
jwt.js	90.9	100	100	90	12

Fonte: Autores.

7.6 Manuais

A equipe desenvolveu três manuais para auxiliar os usuários na instalação e no entendimento da navegabilidade do aplicativo Robô Garçom.

O manual de uso irá ajudar os usuários a entender o funcionamento e a navegabilidade entre telas do aplicativo, através de dois vídeos realizados com os protótipos de alta-fidelidade do cliente e do administrador.

O objetivo do manual de usuário é orientar o usuário na instalação do Aplicativo Robô Garçom, apresentando as instruções de forma prática e objetiva. As informações e funções contidas neste manual estão sujeitas a alterações sem aviso prévio.

De forma semelhante ao manual de usuário, o manual de instalação também tem a finalidade de orientar o usuário no processo de instalação, mas dessa vez do ambiente de desenvolvimento do *software*. Este manual torna-se necessário pois não foi gerado um arquivo executável para instalação do aplicativo Robô Garçom.

Todos esses manuais estão disponíveis na WIKI do projeto, na aba "Manuais".

8 Cronograma do projeto

A Tabela 21 descreve o início e o fim previsto do projeto, bem como o início e o fim realizado. Em sequência, a Tabela 22 traz as atividades gerais da equipe, como reuniões e preenchimento de seções do relatório.

Tabela 21 – Cronograma geral.

Geral	Previsto	Realizado
Início do projeto	30/07/2021	30/07/2021
Fim do projeto	28/10/2021	05/11/2021

Tabela 22 – Cronograma de atividades.

ID	Atividade	Início Realizado	Fim Previsto	Fim Realizado	Responsáveis
1	Reunião - Decisão sobre forma de gestão	30/07/21	30/07/21	30/07/21	Geral
2	Criação do projeto no overleaf	28/07/21	28/07/21	28/07/21	Coordenador geral
3	Criação do repositório do projeto	28/07/21	28/07/21	28/07/21	Coordenador geral
4	Reunião líderes - Alinhamento de ideias	13/08/21	13/08/21	13/08/21	Líderes
5	Cronograma do projeto	14/08/21	19/08/21	16/08/21	Líder de *software*
6	Requisitos do projeto	15/08/21	20/08/21	23/08/21	Líderes
7	Termo de abertura do projeto (TAP)	15/08/21	16/08/21	15/08/21	Coordenador geral
8	Estrutura Analítica de Projeto (EAP)	14/08/21	17/08/21	16/08/21	Líderes
9	RH: Equipe de trabalho	30/07/21	30/07/21	30/07/21	Geral
10	Resumo relatório	16/08/21	20/08/21	16/08/21	Coordenador geral

11	Introdução relatório	23/08/21	22/08/21	30/08/21	Geral
12	Nome Comercial e conceito para o produto	16/08/21	20/08/21	20/08/21	Coordenador geral
13	Elaborar requisitos por núcleo	17/08/21	23/08/21	28/08/21	Líderes
14	Mercado alvo	16/08/21	20/08/21	20/08/21	Coordenador geral
15	Legislação/Normas	17/08/21	23/08/21	25/08/21	Líderes
16	Justificativa	18/08/21	23/08/21	30/08/21	Líder de estrutura
17	Indicadores	17/08/21	23/08/21	28/08/21	Líder de *software*
18	Concorrência	18/08/21	23/08/21	23/08/21	Líder de automação
19	Descrição de estrutura	23/08/21	28/08/21	30/08/21	Equipe de estrutura
20	Descrição de automação e componentes	16/08/21	28/08/21	29/08/21	Equipe de automação
21	Descrição de *Software*	18/08/21	23/08/21	29/08/21	Equipe de *software*
22	Levantamento de custos	21/08/21	23/08/21	27/08/21	Líderes
23	Levantamento de riscos e plano de contingência	18/08/21	23/08/21	29/08/21	Líderes
24	Tabela de atividades por núcleo	16/08/21	23/08/21	30/08/21	Líderes
25	Planilha de auto avaliação PC 1	23/08/21	28/08/21	30/08/21	Geral
26	Apresentação PC1	03/09/21	03/09/21	03/09/21	Líderes
27	Reunião sobre *feedback* 01	08/09/21	08/09/21	08/09/21	Geral
28	Descrição de Estrutura e CAD	09/09/21	17/09/21	19/09/21	Equipe de estrutura
29	Descrição de automação e Diagramas	09/09/21	17/09/21	18/09/21	Equipe de automação
30	Diagramas Arquiteturais de *Software*	09/09/21	17/09/21	18/09/21	Equipe de *software*
31	Apresentação PC2	20/09/21	22/09/21	22/09/21	Líderes
32	Reunião sobre *feedback* 02	27/09/21	27/09/21	27/09/21	Geral
33	Finalização do capítulo de estrutura	27/09/21	22/10/21	24/10/21	Equipe de estrutura
34	Finalização do capítulo de automação	27/09/21	22/10/21	24/10/21	Equipe de automação

35	Finalização do capítulo de *software*	27/09/21	22/10/21	24/10/21	Equipe de *software*
36	Diagrama de integração	15/10/21	22/10/21	24/10/21	Líderes
37	Manual de uso: estrutura	10/10/21	22/10/2021	24/10/21	Ana Paula
38	Manual de uso: automação	10/10/21	22/10/2021	24/10/21	Ana Paula
39	Manual de uso: *software*	10/10/21	22/10/2021	23/10/21	Ana Paula
40	Gravação dos vídeos para apresentação PC3	25/10/21	27/10/21	27/10/21	Líderes
41	Revisão do capítulo de automação para entrega final	30/10/21	05/11/21	05/11/21	Equipe de automação
42	Revisão do capítulo de estrutura para entrega final	30/10/21	05/11/21	05/11/21	Equipe de estrutura
43	Revisão do capítulo de *software* para entrega final	30/10/21	05/11/21	05/11/21	Equipe de *software*
44	Revisão Final do relatório	30/10/21	05/11/21	05/11/21	Líderes
45	Vídeo propaganda	25/10/21	03/11/21	30/10/21	Líderes

8.1 Atividades de Automação

A área de automação, da qual conta com as subáreas de energia e eletrônica dividiram as suas atividades a partir dos sistemas e sensores. A Tabela 23 apresenta o descritivo das atividades a serem desenvolvidas na área de automação.

Tabela 23 – Cronograma descritivo para atividades especificas da automação

En.1.	Sistema de Carregamento	El.1.	Pesquisas
En.1.1.	Carregamento por indução	El.2.	Determinação de Sensores
En.1.2.	Bobinas	El.2.1.	Sensor de Impacto
En.1.3.	Base	El.2.2.	Sensor de Queda
En.2.	Sistema de Baterias	El.2.3.	Sensor de Obstáculo
En. 2.1.	Transformação CA/CC	El.2.4.	Sensor de Peso
En.2.2.	Bateria	El.3.	Circuitos de Inferface
En.2.3.	BMS	El.3.1.	Módulo Conversor Hx711
En.3.	Sistema de Movimentação	El.4.	Atuadores
En.3.1	Motor atuador	El.4.1.	Circuitos de Driver p/ Motores
En.3.2.	Motor Tampa	El.5.	Programação micro controladores
En.3.3.	Motor Rodas	El.6.	Documentação
En.4.	Pesquisas	El.7.	Simulação
En.5.	Documentação		
En.6.	Simulação		
En.7.	Matriz de decisão		

Definidas as atividades, foi realizado a distribuição delas de acordo com o desenvolvimento do projeto e das interações entre as áreas, de maneira em que todo o projeto tenha um andamento conjunto. Este andamento está apresentado na Tabela 24.

Tabela 24 – Cronograma da automação.

Tarefas	Data de Início	Duração em dias	Data Término
En.1.	04/08/2021	30	03/09/2021
En.1.1.	04/08/2021	10	14/08/2021
En.1.2.	14/08/2021	10	24/08/2021
En.1.3.	24/08/2021	10	03/09/2021
En.2.	03/09/2021	50	23/10/2021
En. 2.1.	03/09/2022	10	13/09/2021
En.2.2.	13/09/2021	20	03/10/2021
En.2.3.	13/09/2021	40	23/10/2021
En.3.	03/09/2021	20	23/09/2021
En.3.1	03/09/2021	10	19/09/2021
En.3.2.	03/09/2021	10	13/09/2021
En.3.3.	03/09/2021	20	23/09/2021
En.4.	04/08/2021	86	29/10/2021
En.5.	14/08/2021	76	29/10/2021
En.6.	05/10/2021	24	29/10/2021
En.7.	14/08/2021	30	13/09/2021
El.1.	04/08/2021	86	29/10/2021
El.2.	04/08/2021	20	24/08/2021
El.2.1.	04/08/2021	20	24/08/2021
El.2.2.	04/08/2021	20	24/08/2021
El.2.3.	04/08/2021	20	24/08/2021
El.2.4.	04/08/2021	20	24/08/2021
El.3.	24/08/2021	30	23/09/2021
El.3.1.	24/08/2021	30	23/09/2021
El.4.	24/08/2021	30	23/09/2021
El.4.1.	24/08/2021	30	23/09/2021
El.5.	03/09/2021	30	03/10/2021
El.6.	14/08/2021	76	29/10/2021
El.7.	05/10/2021	24	29/10/2021

8.2 Atividades de Estruturas

Tabela 25 – Atividades de Estruturas

Atividade	Início	Fim	Responsáveis
Dimensionamento do robô	12/08/2021	26/08/2021	Luiz
Requisitos	06/08/2021	26/08/2021	Todos
Regra de Negócios	06/08/2021	26/08/2021	Isabella
Orçamentário	06/08/2021	26/08/2021	Isabella
Riscos	12/08/2021	26/08/2021	Isaac
Arquitetura da tampa	12/08/2021	26/08/2021	Isabella

Arquitetura do sistema de levantamento automático	12/08/2021	26/08/2021	Isaac
Arquitetura do Sistema de Locomoção	12/08/2021	26/08/2021	Lucas
Análise das Velocidade e Acelerações	12/08/2021	26/08/2021	Ana
Especificações finais	03/09/2021	10/09/2021	
CADs para Simulações	08/09/2021	11/09/2021	
Simulações	11/09/2021	13/09/2021	
CADs de subsistemas	08/09/2021	14/09/2021	
Desenhos técnicos e revisões dos CADs	11/09/2021	16/09/2021	
CAD completo	13/09/2021	15/09/2021	
Relatório PC2	06/09/2021	17/09/2021	
Revisão do relatório	16/09/2021	19/09/2021	
Apresentação	17/09/2021	21/09/2021	

8.3 Atividades de *Software*

As atividades do núcleo de *Software* são organizadas por meio de *issues* disponibilizadas no repositório de documentação da equipe. As tarefas são divididas em *Sprints*, que duram em média 1 semana. Cada integrante tem o prazo do início ao fim de cada *Sprint* para finalizar a atividade que está em sua responsabilidade. A Tabela 26 mostra como se deu a divisão de *Sprints* entre os pontos de controle do projeto, e a duração de cada uma. A distribuição das tarefas de software ao longo das *Sprints* está apresentado na Tabela 27.

Tabela 26 – *Sprints* concluídas em relação a cada ponto de controle.

	Sprint	**Início**	**Término**
Ponto de controle 1	1	06/08/2021	11/08/2021
	2	11/08/2021	18/08/2021
	3	18/08/2021	29/08/2021
Ponto de controle 2	4	30/08/2021	05/09/2021
	5	06/09/2021	11/09/2021
	6	12/09/2021	17/09/2021
Ponto de controle 3	7	08/10/2021	05/11/2021

Tabela 27 – Atividades de Software

Atividade	**Início**	**Fim**	**Responsáveis**
Visão de produto - Visão geral	06/08/21	11/08/21	Todos
Objetivo de negócio - Visão geral	06/08/21	11/08/21	Todos
Objetivo do produto - Visão geral	06/08/21	11/08/21	Todos
O que é? O que faz?	06/08/21	11/08/21	Todos
Desenhos - Protótipo baixa fidelidade	06/08/21	11/08/21	Todos
Visão de produto	12/08/21	17/08/21	Samuel
Objetivos de produto e negócio	12/08/21	17/08/21	Letícia
Requisitos funcionais e não funcionais	12/08/21	17/08/21	Bruna
Rich Pictures	12/08/21	17/08/21	Todos
MoScoW	12/08/21	17/08/21	Damarcones
First things First	12/08/21	17/08/21	Samuel
NFR	12/08/21	17/08/21	Damarcones
Especificação suplementar	12/08/21	17/08/21	Letícia
Protótipo de baixa fidelidade	12/08/21	17/08/21	Todos
Cronograma de atividades	12/08/21	17/08/21	Bruna
Organizar a WIKI	12/08/21	17/08/21	Bruna
Protótipo de alta fidelidade	18/08/21	23/08/21	Todos
Casos de uso	18/08/2021	23/08/2021	Samuel
Diagramas de casos de uso	18/08/2021	23/08/2021	Samuel
Histórias de usuário	18/08/21	23/08/21	Letícia
Épicos	18/08/21	23/08/21	Letícia
Plano gerenc. riscos/contingência	18/08/21	23/08/21	Bruna
Estimativa de custos	18/08/21	23/08/21	Bruna
Metodologia	18/08/21	23/08/21	Bruna
Legislação/normas	18/08/21	23/08/21	Samuel
Indicadores	18/08/21	23/08/21	Letícia
Preencher planilha de atividades	18/08/21	23/08/21	Bruna
Requisitos de software	18/08/21	23/08/21	Bruna
Requisitos gerais	18/08/21	23/08/21	Damarcones
Descrição do aplicativo	18/08/21	23/08/21	Damarcones
Proposta de inovação	18/08/21	23/08/21	Todos
Arquitetura da informação (protótipos)	18/08/21	23/08/21	Damarcones
Repositório (Relatório)	18/08/21	23/08/21	Damarcones
Identidade visual	18/08/21	23/08/21	Bruna
Backlog	18/08/21	23/08/21	Letícia

Diagrama de atividades	30/08/21	05/09/21	Bruna, Damarcones
Algoritmo de melhor caminho	30/08/21	05/09/21	Bruna, Damarcones
Diagrama de classes	30/08/21	05/09/21	Samuel
Diagrama de arquitetura	30/08/21	05/09/21	Letícia
Heurísticas de Nielsen	06/09/21	11/09/21	Bruna
Criar o Docker	06/09/21	11/09/21	Samuel
Desenvolvimento do backend (CRUDs)	06/09/21	11/09/21	Samuel
Levantamento de componentes a serem integrados	06/09/21	11/09/21	Damarcones
Descrição do banco de dados no relatório	06/09/21	11/09/21	Damarcones
Descrição da arquitetura no relatório	06/09/21	11/09/21	Bruna
Criar repositórios para o back e front	06/09/21	11/09/21	Damarcones
Descrição do front e back no relatório	06/09/21	11/09/21	Bruna
Reestruturação da parte de software	06/09/21	11/09/21	Bruna
Descrição da linguagem e Johnny Five	06/09/21	11/09/21	Letícia
Diagrama de pacotes	06/09/21	11/09/21	Samuel
Estruturação do banco de dados	06/09/21	11/09/21	Samuel
Códigos para integração	12/09/21	17/09/21	Bruna, Damarcones
Fazer linkagem dos códigos no relatório	12/09/21	17/09/21	Bruna, Damarcones
Diagrama de componentes	12/09/21	17/09/21	Damarcones
Descrição dos diagramas no relatório	12/09/21	17/09/21	Letícia, Bruna, Damarcones
Diagrama de entidade-relacionamento	12/09/21	17/09/21	Letícia
Apêndice de decisões de software	12/09/21	17/09/21	Bruna, Damarcones
Organização da WIKI para o PC2	12/09/21	17/09/21	Bruna
Revisão do cap. de software para o PC3	08/10/21	23/10/21	Bruna
Organização da WIKI entrega final	08/10/21	23/10/21	Bruna
Refatorar tópico do MVC	29/10/21	05/11/21	Bruna
Refatorar tópico J5	29/10/21		Damarcones
Refatorar tabela de custos	29/10/21	05/11/21	Damarcones
Código J5 p/ integração	08/10/21	05/11/21	Damarcones
Manual de uso	08/10/21	23/10/21	Letícia

Manual do usuário	08/10/21	23/10/21	Letícia
Manual de instalação	08/10/21	23/10/21	Letícia
Finalizar o backend	08/10/21	31/10/21	Samuel
Fazer testes no backend	08/10/21	01/11/21	Samuel
Desenvolver frontend	29/10/21	05/11/21	Samuel

Referências

Mattede. *Sensor infravermelho – O que é e como funciona!* 2021. <https://www.mundodaeletrica.com.br/sensor-infravermelho-o-que-e-como-funciona/>. *Online*, acesso em 15 de Setembro de 2021. Citado na página 50.

STRAUB. *BALANÇA ARDUINO COM CÉLULA DE CARGA E HX711*. 2019. <https://www.usinainfo.com.br/blog/balanca-arduino-com-celula-de-peso-e-hx711-tutorial-calibrando-e-verificando-peso/>. *Online*, acesso em 14 de Setembro de 2021. Citado na página 51.

BBC. *O que é a 4ª revolução industrial - e como ela deve afetar nossas vidas*. 2016. <https://www.bbc.com/portuguese/geral-37658309>. *Online*, acesso em 14 de Setembro de 2021. Citado na página 15.

BELLABOT. 2021. <https://www.pudurobotics.com/ads/bellabot/br?gclid=CjwKCAjwmeiIBhA6EiwA-uaeFZebHWfA7mQd27BZM_NsdsqItX7VkpM6LJOhgyDg_qMLm0BE_qO8OxoCNnQQAvD_BwE>. *Online*, acesso em 20 de agosto de 2021. Citado na página 103.

CORTELLETTI, D. *Introdução à programação de microcontroladores Microchip PIC*. [S.l.]: Dossiê Técnico, SENAI-RS, Centro Tecnológico de Mecatrônica, 2006. Citado na página 12.

COSTA, A. d. S. C.; SANTANA, L. C. d.; TRIGO, A. C. Qualidade do atendimento ao cliente: um grande diferencial competitivo para as organizações. *Revista de Iniciação Científica–RIC Cairu*, v. 2, n. 2, p. 155–172, 2015. Citado na página 12.

CROUZET. *MICRO MOTOR DC C/ CX RED*. 2021. <https://www.google.com/url?q=https://br.mouser.com/ProductDetail/Crouzet/82800036?qs%3DBSygsA9le0Ptfxre%252FXJqqA%253D%253D&sa=D&source=editors&ust=1636159865342000&usg=AOvVaw357-EIb-t1tBTbyKHze0gn>. *Online*, acesso em 18 de setembro de 2021. Citado na página 49.

CSJBOT. 2021. <https://portuguese.alibaba.com/product-detail/csjbot-catering-hotel-serving-food-waiter-human-restaurant-service-delivery-humanoid-intelligent-r
html?spm=a2700.7724857.topad_classic.d_title.5afa101etVxchO>. *Online*, acesso em 20 de agosto de 2021. Citado na página 103.

CUNHA, R. de L. *"ESTIMAÇÃO DA DISTÂNCIA DE PONTOS MÓVEIS BASEADA NA POTÊNCIA DE SINAIS DE ROTEADORES DE REDES SEM FIO"*. 2017. Citado na página 52.

DALTIO, M. L. Sistema de transmissÃo de energia sem fio com ajuste automÁtico da frequÊncia de ressonÂncia para maximizaÇÃo da potÊncia transferida. 2019. Citado 2 vezes nas páginas 45 e 46.

DEVMEDIA. *Modelagem de software com UML*. 2011. <https://www.devmedia.com.br/modelagem-de-software-com-uml/20140>. *Online*, acesso em 11 de setembro de 2021. Citado na página 78.

EAE business school. *Fastfood consuption*. 2014. <https://brasil.elpais.com/brasil/2016/01/21/economia/1453403379_213071.html>. *Online*, acesso em 20 de Agosto de 2021. Citado na página 25.

EMAX. *SERVO MOTO SG90*. 2021. <https://www.filipeflop.com/produto/micro-servo-emax-es08ma-ii-12g-metalico/>. *Online*, acesso em 18 de setembro de 2021. Citado na página 49.

ESCUDELARIO, B. de F. *Vale a pena aprender NodeJS?* 2018. <https://imasters.com.br/back-end/vale-pena-aprender-nodejs>. *Online*, acesso em 07 de setembro de 2021. Citado na página 73.

ESPRESSIF. *ESP32*. 2021. <https://www.espressif.com/en/products/socs/esp32>. *Online*, acesso em 18 de setembro de 2021. Citado na página 48.

FIGUEIREDO, M. Navegação autônoma de robôs. Escola de Informática da SBC - Regional Sul., 1999. Citado na página 12.

FLYER, A. *The State of App Marketing LATAM 2020*. 2020. <https://www.appsflyer.com/br/resources/others/state-of-app-marketing-latam/>. *Online*, acesso em 30 de agosto de 2021. Citado na página 25.

FRANA, P. L.; MISA, T. J. An interview with edsger w. dijkstra. *Communications of the ACM*, ACM New York, NY, USA, v. 53, n. 8, p. 41–47, 2010. Citado na página 77.

GABRIEL; LETíCIA; DIOGO. *Robo$_Firmware$*.2021.<>. *Online*, acesso em 07 de setembro de 2021. Citado 2 vezes nas páginas 59 e 148.

H. S. Hassan and M. Hussein and S. M. and Saad and M. A. Dzahir. An overview of local posicioning system: tecnologies, techniques and applications. 2018. Citado 2 vezes nas páginas 15 e 123.

HERSHENSON, M. del M. et al. Optimization of inductor circuits via geometric programming. 1999. Citado na página 46.

INDIAMART. *Pcb + Brass GY-61 ADXL335 3-Axis Accelerometer Module*. 2021. <https://www.indiamart.com/proddetail/gy-61-adxl335-3-axis-accelerometer-module-3489131733.html>. *Online*, acesso em 18 de setembro. 2021. Citado na página 50.

INSTITUTE, I. A. *What is Information Architecture?* 2020. <https://www.iainstitute.org/what-is-ia>. *Online*, acesso em 27 de agosto de 2021. Citado na página 73.

KEENON. 2020. <https://www.keenonrobot.com/EN/Product/pro2.html>. *Online*, acesso em 20 de agosto de 2021. Citado na página 103.

KOTLER PHILIP E ARMSTRONG, G. *Princípios de Marketing*. 9. ed. [S.l.]: PrenticeHall, 2003. v. 2. Citado na página 12.

LORA, F.; HEMERLY, E. M.; LAGES, W. F. Sistema para navegação e guiagem de robôs móveis autônomos. *Controle & Automação*, v. 9, n. 3, p. 107–118, 1998. Citado na página 12.

MENSAGEM meio . *Pesquisa do iFood revela hábitos de consumo no delivery*. 2016. <https://www.meioemensagem.com.br/home/marketing/2016/07/01/pesquisa-do-ifood-revela-habitos-de-co html>. *Online*, acesso em 30 de agosto de 2021. Citado na página 25.

Mercado & consumo. *Autoatendimento cresce no foodservice brasileiro*. 2018. <https://mercadoeconsum com.br/2018/06/18/autoatendimento-cresce-no-foodservice-brasileiro/>. *Online*, acesso em 20 de Agosto de 2021. Citado na página 25.

MICHAELIS. *Moderno dicionário da língua portuguesa*. 1998. <https://michaelis.uol.com.br/busca?id=BVqoW>. *Online*, acesso em 29 de agosto de 2021. Citado na página 12.

MOTTA, T. D. S. TransmissÃo de energia sem-fio: Método de acoplamento indutivo ressonante. 2016. Citado na página 46.

NET, O. D. *O que é Model-view-controller (MVC)?* 2011. <https://www.oficinadanet.com.br/artigo/desenvolvimento/o_que_e_model-view-controller_mvc>. *Online*, acesso em 07 de setembro de 2021. Citado na página 72.

NEVES, F. *"ESP32 – Lidando com Multiprocessamento – Parte I"*. 2020. <https://www.embarcados.com.br/esp32-lidando-com-multiprocessamento-parte-i/>. Citado na página 54.

ORIHIME-D. 2021. <https://orylab.com/en/#product>. *Online*, acesso em 20 de agosto de 2021. Citado na página 103.

OVERFLOW, S. ***Most Popular Technologies**: programming, scripting, and markup languages. Programming, Scripting, and Markup Languages*. 2021. <https://insights.stackoverflow.com/survey/2020#most-popular-technologies>. *Online*, acesso em 07 de setembro de 2021. Citado na página 72.

PEREIRA, e. a. A concepção de robótica dos alunos de nível médio a partir da representação de protótipos relacionados ao conceito de titulação. In: *Encontro Nacional de Pesquisa em Educação em Ciências*. [S.l.: s.n.], 2013. Citado na página 12.

PMI - PROJECT MANAGEMENT INSTITUTE. *GUIA PMBOK®: Um Guia para o Conjunto de Conhecimentos em Gerenciamento de Projetos*. 2017. <Sextaediç~ao,Pennsylvania:PMI>. *Sexta edição, Pennsylvania: 2017*. Citado 2 vezes nas páginas 17 e 105.

PSCHEIDT, É. R. Robô autônomo–modelo chão de fábrica. *UNICENP/NCET, Engenharia da Computação, Curitiba*, 2007. Citado na página 12.

RAZOR. *Power Core 90/E90 (V2+) Rer Wheel w/ Motor Green*. 2021. <https://razor.com/products/replacement-parts/shop-by-product/electric-scooter-parts/power-core-90-e90-v2-rear-wheel-w-mot>. *Online*, acesso em 18 de setembro de 2021. Citado na página 49.

RECK, M. S. *"Beacons BLE – BLUETOOTH LOW ENERGY – Design e análise de um sistema de localização indoor"*. 2016. Citado na página 53.

RESEARCH, A. M. *Sensor Market*. 2021. <https://www.alliedmarketresearch.com/sensor-market>. *Online*, acesso em 20 de Agosto de 2021. Citado na página 25.

SENSE. *Sensores ultrassônicos*. 2014. <https://www.sense.com.br/arquivos/produtos/arq1/Sensores_Ultrass%C3%B4nicos_Sense_Folheto_Rev_%20J.pdf>. *Online*, acesso em 15 de Setembro de 2021. Citado na página 51.

SERVI. 2021. <https://www.bearrobotics.ai/O>. *Online*, acesso em 20 de agosto de 2021. Citado na página 103.

SERVICEROBOTS. *Amy Waitress*. 2020. <https://www.servicerobots.com/amy-waitress/>. *Online*, acesso em 20 de agosto de 2021. Citado 2 vezes nas páginas 25 e 103.

SIMõES, M. Oficina seguidor de linha - o guia definitivo do mochileiro maker. *UNIVERSIDADE FEDERAL DE MINAS GERAIS*, 2017. Citado na página 123.

SKYRICH. *Catálogo Skyrich*. 2021. <http://www.skyrich.com.ar/wp-content/uploads/2020/11/Catalogo_Skyrich_Haijiu.pdf>. *Online*, acesso em 18 de setembro de 2021. Citado na página 47.

VICTRON. [S.l.], 2021. Citado na página 48.

VICTRON. *smallBMS with pre-alarm*. 2021. <https://www.victronenergy.pt/upload/documents/Datasheet-smallBMS-with-pre-alarm-EN.pdf>. *Online*, acesso em 18 de setembro de 2021. Citado na página 47.

WAGON le. *O que é padrão MVC? Entenda arquitetura de softwares!* 2020. <https://www.lewagon.com/pt-BR/blog/o-que-e-padrao-mvc>. *Online*, acesso em 07 de setembro de 2021. Citado na página 71.

APÊNDICE A – Equipe de Trabalho

A.1 Membros e atribuições

A divisão de atividades foi feita conforme as atribuições de cada membro da equipe apresentada na Tabela 28.

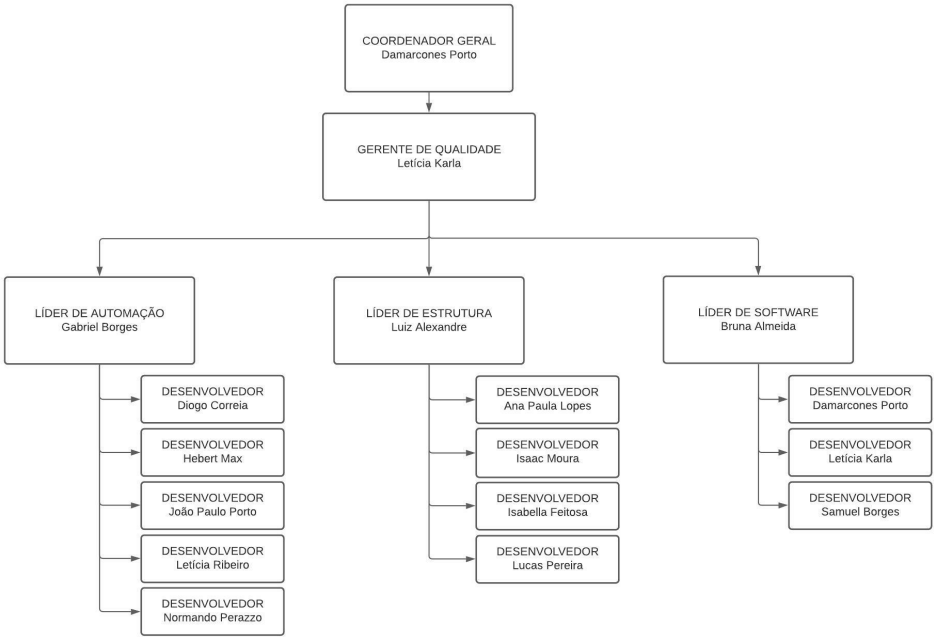
Tabela 28 – Membros e atribuições

Nome	**Matrícula**	**Curso**	**Atribuições**
Ana Paula Lopes	15/0005342	Eng. Aeroespacial	Desenvolvedor (estrutura)
Bruna Almeida	17/0100626	Eng. de Software	Líder de Software
Damarcones Porto	15/0122187	Eng. de Software	Coordenador Geral Desenvolvedor (software)
Diogo Correia	15/0058641	Eng. Eletrônica	Desenvolvedor (automação)
Gabriel Borges	13/014206	Eng. Eletrônica	Líder de Automação
Hebert Max	19/0046180	Eng. de Energia	Desenvolvedor (automação)
Isaac Moura	17/0059171	Eng. Aeroespacial	Desenvolvedor (estrutura)
Isabella Feitosa	16/0152437	Eng. Aeroespacial	Desenvolvedor (estrutura)
João Paulo Porto	11/0062167	Eng. de Energia	Desenvolvedor (automação)
Letícia Karla	15/0135939	Eng. de Software	Gerente de Qualidade Desenvolvedor (software)
Letícia Ribeiro	16/0131995	Eng. Eletrônica	Desenvolvedor (automação)
Lucas Pereira	17/0016951	Eng. Automotiva	Desenvolvedor (estrutura)
Luiz Alexandre	15/0041390	Eng. Aeroespacial	Líder de Estrutura
Normando Perazzo	16/0016258	Eng. de Energia	Desenvolvedor (automação)
Samuel Borges	16/0054389	Eng. de Software	Desenvolvedor (software)

Fonte: Autores

Em termos hierárquicos, em ordem decrescente, pode-se ressaltar o Coordenador Geral, a Gerente de Qualidade, os Líderes de área (Automação, Estrutura e Software) e os Desenvolvedores das respectivas áreas de trabalho. Essa organização é evidenciada na Figura 55.

Figura 55 – Organização da hierarquia



Fonte: Autores

A.2 Ferramentas de Gerenciamento

Para facilitar questões relacionadas à comunicação, integração de documentos e acompanhamento de produtividade de equipe, levantamos alguns Indicadores-chave de Desempenho, KPIs.

- Deadline: Data limite para entrega de um projeto. O deadline serve para alinhar a necessidade da urgência da entrega de uma demanda especifica para que seja possível alcançar objetivo do projeto.
- Produtividade: A produtividade do nosso projeto diz respeito à capacidade de produzir mais, com a utilização de menos recursos possíveis. Foram definidos objetivos para a capacidade de produção individual e coletiva, agilidade, e otimização de processos.
- Gestão de Escopo: Nós definimos um conjunto de processos essenciais para garantir que o Projeto do Robô Garçom inclui todo o trabalho necessário, para ter sucesso. Os processos que fazem parte desse conjunto são:
  - Planejar o Gerenciamento de Escopo.
  - Coletar os Requisitos.
  - Definir o Escopo.
  - Criar a EAP.
  - Validar o Escopo.

– Controlar o Escopo.

E para auxiliar no desenvolvimento dessas KPIs algumas ferramentas foram selecionadas para uso comum de toda a equipe, descritas na Tabela 29, que descreve quais são tais ferramentas e seus objetivos de uso.

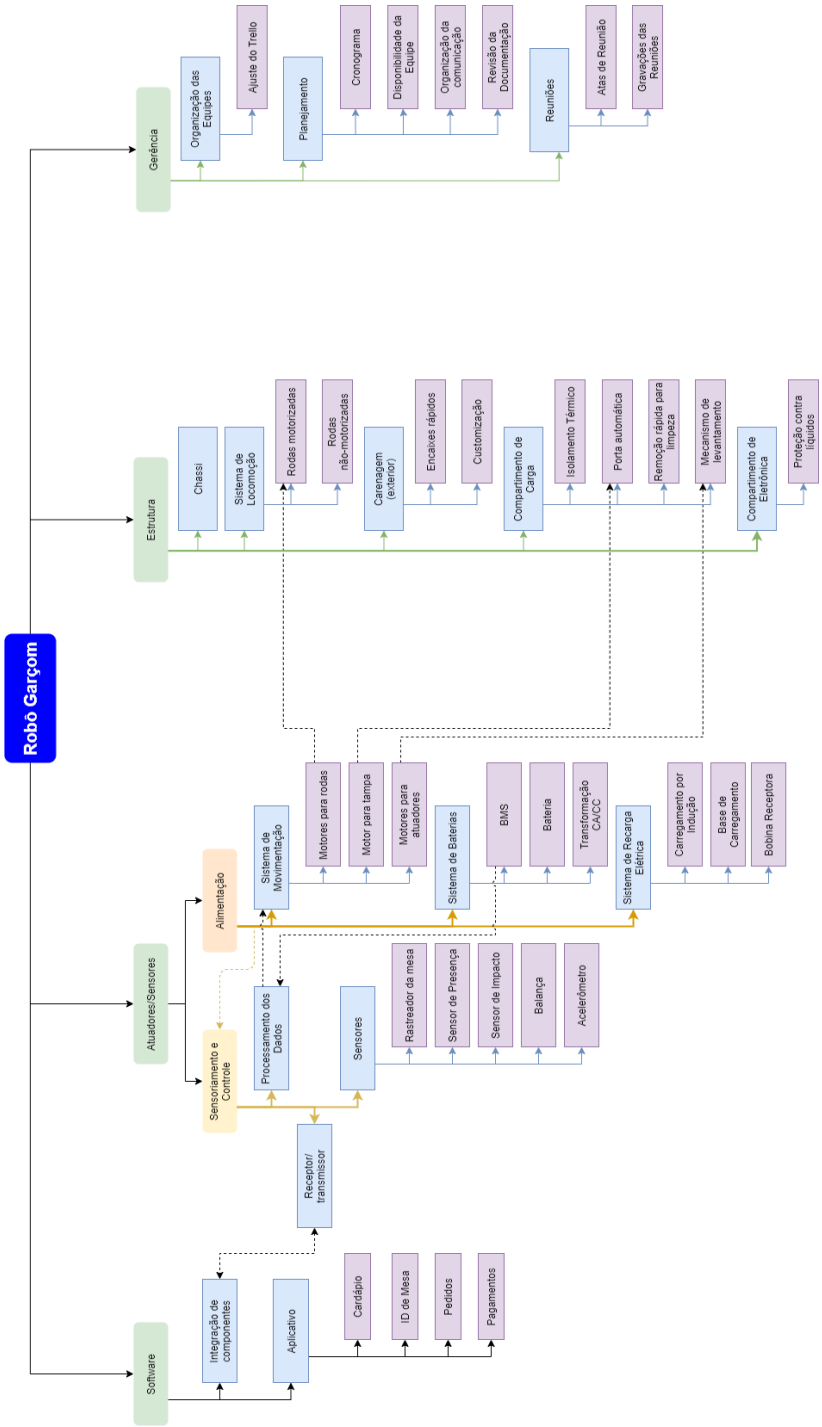
Tabela 29 – Ferramentas de gerenciamento da equipe

Ferramenta	**Objetivo de uso**
Telegram	Principal canal de comunicação rápida entre integrantes da equipe, além de ser canal informativo dos professores com a turma.
Trello	Gerenciamento de tarefas de equipe, com foco nos agendamentos de reuniões e atividades para os líderes do grupo.
Google Drive	Ponto de armazenamento colaborativo de artefatos de todos os núcleos, gerando um acesso mais rápido.
GitHub	Plataforma usada para armazenar o versionamento dos códigos dos núcleos de automação e *Software*, e para armazenar a documentação de *Software*.
Microsoft Teams	Canal para realização das reuniões gerais da equipe, bem como das específicas de cada núcleo.
Overleaf	Editor LaTeX colaborativo do relatório final para a disciplina.
Corrello	Para realizar o burndown da equipe e de cada integrante. Dessa forma, a gerência consegui verificar quem estava entregando e sendo produtivo.

Fonte: Autores

APÊNDICE B – Estrutura analítica do projeto (EAP)

Figura 56 – Estrutura analítica do projeto.



Fonte: Autores.

APÊNDICE C – Análise de mercado

C.1 Nome comercial e conceito para o produto

O nome comercial do produto será Robô Garçom LTDA., tendo como nome fantasia **Robô Garçom**.

C.2 Mercado-alvo

C.2.1 Consumidor final

O produto será voltado para restaurantes que tem o interesse de automatizar os processos de atendimento ao cliente. Além da automatização do atendimento, o Robô Garçom fornecerá uma barreira protetora aos pratos solicitados pelo cliente, já que estará equipado com um compartimento fechado onde estará o pedido, de forma a evitar exposição ao ambiente durante o trajeto da cozinha à mesa.

O Robô Garçom necessita de um espaçamento mínimo de 75 cm para total locomoção, para que sejam evitados possíveis acidentes por transitar em locais estreitos. O Robô Garçom poderá ser utilizado por estabelecimentos que possuam pisos planos e sem desnível no salão onde serão servidas as refeições.

C.2.2 Cliente responsável pela comercialização

A comercialização será de inteira responsabilidade da equipe de desenvolvimento, não sendo necessária a intervenção de empresas distribuidoras. Todas as unidades do produto serão negociadas diretamente com estabelecimentos interessados que atendam aos requisitos do produto.

C.3 Concorrência

C.3.1 Concorrentes

O cenário comercial atual mostra que a concorrência é composta de empresas internacionais que desenvolvem robôs de serviço. Os produtos concorrentes estão melhor detalhados na Tabela 30.

Tabela 30 – Lista de concorrentes.

Nome	Preço	Informações
Csjbot	U$7000	Bateria: 20 Ah. Duração: 12 h. Peso: 56 kg. Altura: 1,5 m. Tempo de carregamento: 10 h (estação de carregamento). Sensores: infravermelho, liDAR e ultrassônico [Csjbot 2021].
Keenon	Não informado	Altura: 1,2 m. Peso 67 kg. Tempo de carregamento: 4 h. Sensores: liDAR, visão de mecânica, percepção de profuntidade, infravermelho, sensor de toque, reconhecimento de voz e inteligência artificial [Keenon 2020].
Amy waitress	Não informado	Altura: 1,5 m. Interação por fala, navegação sem trilhas e desviando de obstáculos. Faz pedidos direto no robô, pagamento online [servicerobots 2020].
Bellabot	Não informado	Altura: 1,29 m. Peso: 57 kg. Tempo de carregamento: 4,5 h. Duração da bateria: 12 a 24 h. Câmera de profundidade RGBD para percepção tridimensional, módulo de voz por inteligência artificial, posicionamento a laser [Bellabot 2021].
Servi	Não informado	Altura:1,04 m. Peso: 34 kg. Duração da bateria: 8 a 12 h. Câmera, sensores a laser, mapeamento do ambiente de trabalho [Servi 2021].
OriHime-D	Não informado	Altura: 1,2 m. Câmera, microfone, alto-falante, operado pela internet. Move para frente, trás e gira. Parte superior com 14 juntas motoras capazes de realizar diversos movimentos. Software V-Sido para operar o robô [OriHime-D 2021].

Encontram-se alguns robôs no mercado como o Csjbot com custo explicitado, Keenon que oferece diferentes modelos de robôs, Amy waitress, Bellabot e Servi em cada um deles utiliza diferentes modelos de solução tecnológica.

C.3.2 Viabilidade técnica

A automação, do ponto de vista de construção do Robô Garçom, é viável apesar do cronograma apertado com entrega final prevista para o dia 29 de outubro de 2021.

Tomando por base o valor do [Csjbot 2021] e considerando que os outros modelos tenham valores semelhantes, o Robô Garçom desenvolvido neste projeto tem um custo de desenvolvimento mais baixo do que os concorrentes, pois utiliza soluções mais simples e sensores mais básicos com nível adequado de funcionamento. O maior desafio para a im-

plementação do Robô Garçom é desenvolver o produto por completo em apenas 3 meses, não há tempo hábil para a elaboração de testes massivos com componente, estrutura ou funcionamento.

O Robô Garçom está na frente dos seus concorrentes por apresentar uma relação custo-benefício melhor, já que é um produto nacional com valor final de venda mais baixo justamente por ser um robô mais simples, enquanto desempenha as mesmas funções de forma satisfatória.

Outra vantagem acerca do projeto Robô Garçom é a implementação de um software de gerenciamento de pedidos do restaurante, que integra o robô ao sistema de pedidos e à cozinha. Junto a isso, tem-se também um algoritmo para maximizar seu desempenho durante a locomoção.

APÊNDICE D – Gerenciamento de riscos

D.1 Levantamento de riscos

O gerenciamento dos riscos do projeto tem por objetivo diminuir a probabilidade e/ou o impacto de riscos negativos, a fim de otimizar as chances de sucesso do projeto [PMI - PROJECT MANAGEMENT INSTITUTE 2017].

Baseando-se na metodologia de gerenciamento dos riscos de projeto apresentada no livro *Project Management Body of Knowledge* (PMBOK), os riscos levantados pelo grupo serão classificados quanto à prioridade, que é definida a partir do produto entre os pesos de impacto do risco ao projeto e probabilidade de ocorrência.

$$Prioridade = Impacto \times Probabilidade$$

O impacto do risco ao projeto é categorizado em cinco níveis:

- Muito baixo: De pouca relevância para o projeto - Peso 1
- Baixo: Expressa pouco impacto sobre o projeto - Peso 2
- Médio: Impacto considerável - Peso 3
- Alto: Grande impacto no projeto - Peso 4
- Muito alto: Impede o prosseguimento do projeto - Peso 5

Da mesma forma, a categorização da probabilidade do risco se dá em cinco níveis de porcentagem:

- Muito baixa: intervalo de 0% à 20% - Peso 1
- Baixa: intervalo de 21% à 40% - Peso 2
- Média: intervalo de 41% à 60% - Peso 3
- Alta: intervalo de 61% à 80% - Peso 4
- Muito alta: intervalo de 81% à 100% - Peso 5

Quanto à Estrutura Analítica de Riscos (EAR), será utilizado um modelo personalizado que se baseia nos objetivos do projeto. A Figura 57 demonstra a EAR planejada para o projeto Robô Garçom.

Figura 57 – Estrutura Analítica de Riscos (EAR). Fonte: Autores.

EAR NÍVEL 0	EAR NÍVEL 1	EAR NÍVEL 2
0. TODAS AS FONTES DE RISCO DO PROJETO	**1.** RISCO TÉCNICO	**1.1** Definição do escopo
		1.2 Definição dos requisitos
		1.3 Estimativas, premissas, e restrições
		1.4 Processos técnicos
		1.5 Tecnologia
	2. RISCO DE GERENCIAMENTO	**2.1** Gerenciamento de projetos
		2.2 Gerenciamento de operações
		2.3 Organização
		2.4 Recursos
		2.5 Comunicação
	3. RISCO COMERCIAL	**3.1** Termos e condições do contrato
		3.2 Fornecedores e prestadores de serviço
		3.3 Estabilidade do cliente
		3.4 Parcerias e joint ventures
	4. RISCO EXTERNO	**4.1** Legislação
		4.2 Local/instalações
		4.3 Meio ambiente/clima
		4.4 Concorrência
		4.5 Regulamentação

Por fim, as tabelas a seguir apresentam os riscos levantados, classificados a partir da categorização da EAR e da prioridade.

Tabela 31 – Riscos técnicos

ID	Descrição do risco	Consequência	I	P	I×P
RT01	Trabalho fora do escopo	Gasto de tempo e energia.	3	2	6
RT02	Dificuldade da equipe com as tecnologias utilizadas	Atraso nas entregas, dificuldade de interação entre a equipe e baixa qualidade do produto final.	4	2	8
RT03	Escolha de tecnologias inadequados ao projeto	Não entrega do produto ou entrega inferior.	5	2	10
RT04	Foco em atividades pouco relevantes para o projeto	Tempo e energia desperdiçados.	2	2	4
RT05	Falta de perícia na integração software/eletrônica	Não entrega do produto ou entrega inferior.	5	3	15
RT06	Algoritmo de procura não ser adequado à necessidade do Robô Garçom	Solução não otimizada, aumentando a probabilidade de erros ou defeitos.	5	4	20
RT07	Falha na integração dos sensores e atuadores após a montagem	Não funcionamento completo do produto	5	1	5
RT08	Falha na comunicação Hardware/Software	Não funcionamento completo do produto	4	1	4
RT09	Erro no dimensionamento da alimentação do produto	Alimentação não adequada do sistema	3	1	3
RT10	Falha no percurso Cozinha/mesa	Não entregará o alimento o cliente	4	2	8
RT11	Falha na comunicação entre robô e a mesa	O robô não chegará até a mesa	5	2	10
RT12	Falha no deslocamento e desvio de obstáculos	Acidentes ou dano ao produto e/ou alimento	3	2	6
RT13	Falta de carga durante a operação do robô	Robô descarregado durante a operação	3	2	6
RT14	Robô não ser suficientemente estável	Não entrega do produto ou entrega inferior	5	3	15
RT15	Estrutura não ser capaz de sustentar todas as cargas	O robô não será capaz de entregar alguns ou todos os pedidos	5	1	4
RT16	Excesso de vibrações durante operação	Entrega inferior do produto	4	4	16
RT17	Dificuldade de fabricação das peças projetadas	Aumento de custos e atraso nas entregas	3	2	6

Tabela 32 – Riscos de gerenciamento

ID	Descrição do risco	Consequência	I	P	I×P
RG01	Desistência de um membro	Sobrecarga de trabalho para o restante da equipe.	4	2	8
RG02	Falta de comunicação entre membros	Atraso nas entregas e retrabalho.	3	3	9
RG03	Mal gerenciamento da equipe	Desmotivação da equipe e produto final de baixa qualidade.	5	2	10
RG04	Conflitos internos	Desmotivação da equipe e desistência de membros.	4	3	12
RG05	Cronograma inviável	Sobrecarga da equipe e não cumprimento de prazos.	4	2	8
RG06	Sobrecarga de membros	Desmotivação da equipe e entregas de baixa qualidade.	4	3	12
RG07	Atraso nas entregas	Impedimento da continuidade do projeto.	5	3	15
RG08	Produtividade abaixo do esperado	Atraso na entrega e sobrecarga de outros membros.	5	2	10

Tabela 33 – Riscos comerciais

ID	Descrição do risco	Consequência	I	P	I×P
RC01	Fornecedores entregarem insumos de baixa qualidade	Atraso na produção.	5	3	15
RC02	Atraso na entrega de insumos	Impedimento da continuidade do projeto.	5	4	20
RC03	Troca de prestadores de serviço	Atraso na produção.	3	3	9

RC04	Aumento de custo da matéria-prima	Aumento do valor do produto final.	3	3	9
RC05	Quebra de contrato	Multa e perda de um possível cliente para o produto.	5	4	20
RC06	Inadimplência de clientes	Falta de recebimento pelas vendas realizadas.	3	1	3

Tabela 34 – Riscos externos

ID	Descrição do risco	Consequência	I	P	I×P
RE01	Restaurante não ter infraestrutura adequada	Quebra de contrato por inviabilidade de uso do produto.	5	2	10
RE02	Produto concorrente com um custo-benefício maior	Diminuição do número de possíveis clientes.	4	3	12
RE03	Produto não ser aprovado em testes de qualidade	Reestruturação de todo o projeto.	5	2	10
RE04	Produto não atender alguma norma técnica ou legislativa	Reestruturação de todo o projeto.	5	2	10

D.2 Plano de contingências

Nessa seção, são apresentadas medidas preventivas, para evitar que os riscos apresentados anteriormente ocorram efetivamente, e medidas corretivas, com o objetivo de amenizar o impacto da complicação no projeto.

Tabela 35 – Medidas de contingência

ID	Medidas Preventivas	Medidas Corretivas
RT01	Ter um entendimento maior sobre o projeto como um todo e manter comunicação constante com os membros de outros núcleos	Revisar o escopo e refatorá-lo, se necessário.
RT02	e Fazer um levantamento em equipe sobre as tecnologias conhecidas e criar um quadro de conhecimentos	Fazer uma nova escolha de tecnologias de acordo com o perfil da equipe.
RT03	Fazer um levantamento em equipe sobre as tecnologias que sejam mais adequadas às necessidades do projeto	Fazer uma nova escolha de tecnologias de acordo com a necessidade do produto.
RT04	Priorizar as atividades essenciais para o projeto	Repensar os objetivos do projeto e redefinir as tarefas.
RT05	Constante comunicação entre os núcleos de software e elêtronica, para alinhamento de ideias.	Aumentar o nível de conhecimento acerca das técnicas de integração e melhorar a interação entre os núcleos.
RT06	Fazer um levantamento da melhor tecnologia que satisfaça esse requisito.	Pesquisar novos algoritmos para determinar qual será mais adequado ao projeto.
RT07	Estudo e simulação dos sistemas por partes antes da construção completa	Montagem dos componentes por partes para identificação e correção do problema.
RT08	Estudo de compatibilidade dos softwares embarcados no hardware com a programação utilizada no projeto	Readequação da programação para integrar os componentes utilizados
RT09	Subdimensionamento supondo casos críticos, acrescentando margens de erro	Novo dimensionamento de componentes chaves na alimentação

RT10	Estudo aprimorado dos componentes responsáveis por identificar o trajeto	studo de novos componentes para guiar o robô e/ou outro método
RT11	Acréscimo de redundâncias na comunicação robô/mesa	Aprimoramento e/ou pesquisa de novos meios de comunicação robô/mesa
RT12	Estudo de métodos de desvio de obstáculos utilizados em produtos semelhantes	Incluir restrições na operação do robô no manual do produto e sugestão das condições ideais de trabalho
RT13	Subdimensionamento da alimentação	Inclusão de fonte de alimentação de emergência, como um plug conector.
RT14	Simulações numéricas para garantir a estabilidade do robô em condições de operação	Reprojetar a estrutura e/ou diminuir as velocidades e acelerações de operação
RT15	Simulações numéricas e projeto com fator de segurança adequado	Reprojetar a estrutura
RT16	Simulações numéricas e projeto de suspensão de acordo com a necessidade	Reprojetar a suspensão
RG01	Motivação constante da equipe	Redistribuição de tarefas
RG02	Reuniões constantes para estimular o entrosamento da equipe.	Procurar o problema que está causando a falta de comunicação.
RG03	Buscar feedbacks constantes sobre o desempenho da equipe de trabalho	Mudança de metodologia de gerenciamento.
RG04	Manter diálogo constante e amigável.	O gerente deve buscar resolver o conflito entre os membros.
RG05	Planejamento com base nas entregas do projeto, tentando sempre manter um período de revisão antes de cada entrega.	Priorização e redistribuição de tarefas.
RG06	Distribuição proporção de tarefas entre os membros.	Redistribuição de tarefas.
RG07	Determinar um prazo de acordo com o nível de dificuldade da tarefa.	Aumentar o prazo de entrega da tarefa ou designar mais responsáveis para concluí-la.
RG08	Conferimento constante da qualidade das tarefas realizadas.	Redefinição de prazos e/ou redistribuição das tarefas

RC01	Escolher fornecedores com boa reputação.	Buscar novos fornecedores.
RC02	Considerar no prazo uma margem para atrasos.	Dar prosseguimento em outras etapas do projeto que não necessitam do insumo em atraso.
RC03	Ter em mãos uma lista com outros prestadores de serviço, em caso de inviabilidade do fornecedor principal.	Contatar novos prestadores de serviço o mais rápido possível.
RC04	Possuir um fundo de garantia no orçamento para eventual aumento de preços e desenvolver um estoque de insumos	Fazer uma nova pesquisa de preços para encontrar o melhor custo-benefício.
RC05	Confirmar se a necessidade de ambas as partes estão sendo atendidas, antes da assinatura do contrato.	Buscar um acordo entre as partes envolvidas.
RC06	Criação de um mecanismo de gestão de crédito dos clientes.	Estabelecimento e aplicação de regras de cobrança.
RE01	Deixar claro no escopo os pré-requisitos para a utilização do Robô Garçom	Sugerir modificações no estabelecimento.
RE02	Garantir que o produto tenha um custo-benefício justo.	Não há medida corretiva.
RE03	Realização de testes contínuos	Refatoração de projeto e aplicação de testes mais rigorosos.
RE04	Realizar ampla pesquisa de normas e legislações que se aplicam ao projeto.	Buscar correções o mais breve possível.

APÊNDICE E – Orçamento do projeto

Seguem dispostos os custos os levantados para execução do projeto. Componentes com o preço muito pequeno (como alguns eletro-eletrônicos) foram omitidos, pois não foi julgado relevante para o orçamento.

É importante ressaltar que o levantamento inicial realizado pelas áreas de automação e estruturas, foram focadas no produto, de maneira a não implementar os valores concretos de mão-de-obra e serviços. Por outro ângulo, a área de software realizou um levantamento focado mais na mão-de-obra e serviços.

E.1 Custos para Automação

Foram identificados todos os componentes eletroeletrônicos que compõe a área de automação, assim como os servições necessários para a reprodução.

Os componentes eletrônicos são apresentados na Figura 58. O descritivo da valoração dos sistemas de energia do Robô, está baseado em pesquisas referentes a equipamentos já existentes, para o aproveitamento máximo de tecnologias, e o foco no desenvolvimento da estrutura e sistemas. Para tanto, estão descritos os valores para cada componente na Figura 59. Além dos valores relacionados aos componentes, foram contabilizados os custos dos itens referentes à fiação e conexão elétrica e eletrônica do robô, descritos na Figura 60. Como resultado total da automação, os custos de mão de obra para manutenção, fabricação e montagem dos sistemas está descrito na Figura 61.

Os custos totais da automação estão estimados em R$ 6.553,58. É importante levar em consideração que os componentes selecionados são quase todos vendidos no Brasil, no entanto, não foram coletados os valores quanto à transporte e variações de valores.

Figura 58 – Componentes eletrônicos necessários.

Componentes eletrônicos							
Item		**Componente**			Quantidade	Preço unitário (R$)	Preço (R$)
1	Módulo	Comunicador	Modelo	devkit v4	1	62	62
	Componente	Esp32 wroom-32u	Referência	https://produto.mercadolivre.com.br/MLB-1761224156-esp32-wroom-32u-			
2	Módulo	Detecção de peso	Modelo	YZC-133	1	25,98	25,98
	Componente	Célula de carga + Conversor HX711	Referência	https://produto.mercadolivre.com.br/MLB-1813403024-kit-modulo-hx711-a			
3	Módulo	Detecção de obstáculos	Modelo	HC-SR04	1	12,4	12,4
	Componente	Sensor ultrassônico	Referência	https://www.filipeflop.com/produto/sensor-de-distancia-ultrassonico-hc-sr0			
4	Módulo	Detecção de obstáculos	Modelo	-	2	9	18
	Componente	Sensor infravermelho	Referência	https://www.baudaeletronica.com.br/sensor-de-obstaculo-infravermelho-ir.			
5	Módulo	Magnetômetro (bússola eletrônica)	Modelo	GY-273	1	30	30
	Componente	HMC5883L	Referência	https://www.eletrogate.com/modulo-bussola-magnetica-gy-273			
6	Módulo	Circuitos de interface e acionamento*			7	75	525
	Componente	Componentes e PCB dos circuitos desenvolvidos pela equipe					
						Total (R$)	673,38

Figura 59 – Componentes elétricos necessários.

Componentes Elétricos							
Item		**Componente**			Quantidade	Preço unitário (R$)	Preço (R$)
1	Módulo	Motor de Movimentação	Modelo	W13111402948	2	415,6	831,2
	Componente	Razor, 12V 7,1A	Referência	https://razor.com/products/replacement-parts/shop-by-product/electric-sco			
2	Módulo	Motor Atuador	Modelo	82 800 036	2	644,08	1288,16
	Componente	Cruzet 12V 2,5A	Referência	https://br.mouser.com/ProductDetail/Crouzet/82800036?qs=BSygsA9le0P			
3	Módulo	Motor Tampa	Modelo	ES08MAII	2	46,9	93,8
	Componente	EMAX 5V 0,2A	Referência	https://www.filipeflop.com/produto/micro-servo-emax-es08ma-ii-12g-metal			
4	Módulo	Bateria	Modelo	Lix14	2	489	978
	Componente	Skyrich 12V 14Ha	Referência	https://produto.mercadolivre.com.br/MLB-1582491807-bateria-litio-skyrich-			
5	Módulo	BMS	Modelo	smallBMS	1	273,78	273,78
	Componente	Victron Energi BMS	Referência	https://invertersrus.com/product/victron-smallbms/			
6	Módulo	Pilha	Modelo	4LT44	1	10,87	10,87
	Componente	Pilha Alcalina GP Super 6V 476A	Referência	https://produto.mercadolivre.com.br/MLB-1348532366-01-pilha-4lr44-6v-a			
7	Módulo	Regulador DC/DC	Modelo	XWST 24v-12v	1	350,77	350,77
	Componente	Regulador de Tensão 24/12	Referência	https://pt.aliexpress.com/item/4000738099099.html?src=google&aff_fcid=			
8	Módulo	Regulador AC/DC	Modelo	MiniConversor Fermarc	1	12,75	12,75
	Componente	Regulador de Tensão 110/220-5V	Referência	https://www.fermarc.com/mini-fonte-conversor-acdc-110v-120v-220v-para			
9	Módulo	Regulador DC/DC	Modelo	OEM	2	131,96	263,92
	Componente	Regulador de Tensão 12/5V	Referência	https://www.google.com/shopping/product/9712269321841264418?q=12V			
10	Módulo	Gerador de Clock	Modelo	CJMCU-5351 Si5351	1	44	44
	Componente	Gerador de Clock 6V	Referência	https://www.google.com/shopping/product/1?q=Fonte+gerador+de+clock&			
						Total (R$)	4147,25

Figura 60 – Fiação necessária para construção.

Fiação							
Item		**Componente**			Quantidade	Preço unitário (R$)	Preço (R$)
1	Módulo	Antena wi-fi	Modelo	Wireless 2,4GHz	1	16	16
	Componente	Antena receptora wireless 2,4GHz	Referência	https://produto.mercadolivre.com.br/MLB-2028179105-antena-receptor-wi-			
2	Módulo	Cabo para antena	Modelo	Pigtail	1	20	20
	Componente	Cabo ipex UFL fêmea para SMA fêmea de 50cm	Referência	https://pt.aliexpress.com/item/1005001480408807.html?spm=a2g0o.produ			
3	Módulo	Plug Macho	Modelo	Tripolar macho Porcelana	2	26,89	53,78
	Componente	3 pinos BR 30a	Referência	https://produto.mercadolivre.com.br/MLB-1147761675-plugue-eletrico-trip			
4	Módulo	Cabo de Força	Modelo	9211	1	24,75	24,75
	Componente	Cabo pp com 3 vias, 2,5m 2,5mm preto	Referência	https://www.frigelar.com.br/cabo-flexivel-pp-3-vias-bitola-25mm-megatron-			
5	Módulo	Conector 2x2	Modelo	SPL-2	7	8,55	59,85
	Componente	Conector Tipo Wago alavanca	Referência	https://produto.mercadolivre.com.br/MLB-1423410051-1-conector-emenda			
6	Módulo	Conector 2x4	Modelo	SPL-42	4	77,99	311,96
	Componente	Conector Tipo Wago alavanca	Referência	https://produto.mercadolivre.com.br/MLB-1929593693-5-x-conector-tipo-w			
7	Módulo	Conector 2x6	Modelo	SPL-622	1	71,61	71,61
	Componente	Conector Tipo Wago alavanca	Referência	https://alexnld.com/product/3pcs-spl-62-two-groups-of-parallel-one-in-and-			
8	Módulo	Cobre para Bobinas	Modelo	AWG 9	1	1000	1000
	Componente	Fio de Cobre esmaltado 3.665mm	Referência	https://produto.mercadolivre.com.br/MLB-1587526786-fio-de-cobre-esmal			
						Total (R$)	1557,95

Figura 61 – Estimativa do valor da mão de obra.

Mão de Obra							
Item		**Componente**			Quantidade	Preço unitário (R$)	Preço (R$)
1	Nome	Engenheiro Eletrônico			6	266,67	1600,02
	Função	Validação do projeto e construção do robô	Referência	https://www.vagas.com.br/cargo/engenheiro-eletronico			
						Total (R$)	1600,02

E.2 Custos de Estrutura

A Tabela 36 apresenta uma lista preliminar dos componentes que serão necessários para a concepção da estrutura do robô.

Tabela 36 – Tabela de Componentes Estimadas

Estruturas	**Componente**	**Dimensões (mm x mm x mm)**	**Referência**
Chassi	Tubos Metalon Quadrados	(15 x 15 x 0,9)	https://bityli.com/svnHBT
Chassi	Cantoneira	(44,45 x 44,45 x 4,76)	-
Sistema de Locomoção	Rodas Motorizadas	-	https://bityli.com/tc8sO7
Sistema de Locomoção	Rodas Não-Motorizadas	-	https://bityli.com/oS7yGl
Sistema de Locomoção	Chapa de Aço para o Garfo	(292 x 95 x 4)	https://bityli.com/6wNdKa
Sistema de Locomoção	Chapa Aço para o suporte	(63 x 95 x 2)	https://bityli.com/Lq8fvM
Carenagem Externa	Placa de ACM	(950 x 500 x 3)	https://bityli.com/PqGNy9
Carenagem Externa	Bordas de ACM	(950 x 60 x 1)	https://bityli.com/PqGNy9
Compartimento de Carga	Chapa de ACM para suporte da bandeja	(460 x 475 x 2)	https://bityli.com/NDrldg
Compartimento de Carga	Chapa de ACM para parede da “Casca”	(503 x 2000 x 3)	https://bityli.com/PqGNy9
Compartimento de Carga	Chapa de ACM para base da “Casca”	(481 x 481 x 3)	https://bityli.com/PqGNy9
Mecanismo de Levantamento	Tubo de Alumínio para as Guias do Fuso	(Diâmetro: 16 Comprimento: 482)	https://bityli.com/I0bRQu
Porta Automática	Tampa Flexível de Alumínio	(515 x 490 x 1)	https://bityli.com/P1cPlZ

Parafusos	Parafusos Estruturais	(Diâmetro: 2 Comprimento de rosqueamento: 10 Passo: 4)	https://bityli.com/IGbEvo
Parafusos	Parafusos Não-Estruturais	(Diâmetro: 3,3)	
Parafusos	Parafusos Não-Estruturais	(Diâmetro: 2,5)	https://bityli.com/NZRTv2
Parafusos	Parafusos Não-Estruturais	(Diâmetro: 2,65)	
Parafusos	Parafusos Mancal	(Diâmetro: 10 Comprimento: 45)	https://bityli.com/L2oaOr
Parafusos	Parafusos Suporte de Guia	(Diâmetro: 5 Comprimento: 24)	https://bityli.com/poVF6M
Parafusos	Parafusos Rodas	(Diâmetro: Comprimento:)	

E.2.1 Componentes Estáticos

Para a carenagem externa da estrutura e o compartimento de carga, o custo inicial foi estimado considerando o ACM como material para concepção, que é basicamente um revestimento composto por duas chapas de alumínio com um núcleo termoplástico de polietileno de baixa densidade. O material usado como parâmetro para a estimativa de custo do chassi foi o aço. A estimativa de custos total foi compilada na Tabela 37, da qual apresenta a quantidade pela letra "Q".

E.2.2 Componentes Móveis

O custo dos componentes móveis foi estimado na Tabela 38. Foram encontrados de acordo com produtos existentes no mercado. O custo da tampa flexível e do trilho foram aproximados de acordo com mecanismos semelhantes.

E.2.3 Estimativa de Custo

Além da adição de uma estimativa geral ao custo total referente à fabricação do projeto, envolvendo a soldagem dos tubos, montagem das peças, cortes e dobragens das placas,

Tabela 37 – Custos Preliminares dos Componentes Estáticos.

Estruturas	Componente	Q	Preço Unitário (R$)	Preço (R $)
Chassi	Tubos Metalon Quadrados	2	54,00	108,00
Chassi	Cantoneira	4	3,50	14,00
Caranagem Externa	Placa de ACM	4	54,51	218,04
Compartimento de Carga	Chapa de ACM para parede da "Casca"	1	115,44	115,44
Compartimento de Carga	Chapa de ACM para base da "Casca"	1	26,55	26,55
Compartimento de Carga	Cantoneira	4	3,50	14,00
Parafusos	Parafusos Estruturais	68	1,50	102,00
Parafusos	Parafusos Mancal	8	3,98	31,84
Parafusos	Parafusos Suporte de Guia	16	2,40	38,40
			Total (R$)	668,27

Tabela 38 – Custos Preliminares dos Componentes Móveis.

Estruturas	Componente	Q	Preço Unitário (R$)	Preço (R $)
Sistema de Locomoção	Rodas Motorizadas	2	150,87	301,74
Sistema de Locomoção	Rodas Não-Motorizadas	4	68,24	272,96
Sistema de Locomoção	Chapa de Aço para o Garfo	2	25	50
Sistema de Locomoção	Chapa Aço para o suporte	6	3	18
Mecanismo de Levantamento	Tubo de Alumínio para as Guias do fuso	4	90,46	361,84
Porta Automática	Tampa Flexível de Alumínio	1	52,25	52,25
Porta Automática	Trilho	2	11	22
			Total (R $)	1078,79

foram considerados acréscimos percentuais em cima dos valores estimados a fim de obter diferentes suposições para o orçamento do projeto. Para o cenário otimista o acréscimo percentual foi de 50% do valor prévio, já para os cenários realista e pessimista foram adicionados 100% e 150%, respectivamente.

Tabela 39 – Estimativa Total dos Custos

Núcleo	Prévia	Realista	Otimista	Pessimista
Componentes Estáticos	668,27	1002,41	1336,54	1670,68
Componentes Móveis	1078,79	1618.185	2157,58	2696.975
Fabricação	1500,00	2250,00	3000,00	3750,00
TOTAL (R $)	3247,06	4870,60	6494,12	8117,65

E.3 Custos de Software

O orçamento dos custos de Software é calculado a partir do produto entre a quantidade de horas gastas para desenvolver cada funcionalidade do sistema - que são embasadas nos requisitos funcionais e não-funcionais levantados para o projeto -, e o custo de mão-de-obra dos desenvolvedores, que são divididos em: júnior, pleno e sênior.

Também é incluído na estimativa de custos o valor da integração entre software e a parte eletrônica do projeto, que consiste na programação de todos os componentes e a análise de dados obtidos através dos sensores.

O custo de mão-de-obra dos desenvolvedores foi baseado em um levantamento de salários de programadores, que está disponível na plataforma talent.com. Segundo o levantamento, o custo/hora de cada nível de programador é (Tabela 40):

Tabela 40 – Média de custo/hora de cada nível de programador.

Junior	Pleno	Senior
R$ 15,33	R$ 25,78	R$ 37,69

A partir de uma análise de dificuldade no desenvolvimento de funcionalidades que irão satisfazer cada um dos requisitos, é definido a quantidade de horas que cada requisito irá demandar, que está representado na Tabela 41.

Tabela 41 – Custos de horas.

ID	Requisitos	Horas
1	O usuário poderá acessar o aplicativo sem ter login.	10
2	O usuário poderá visualizar o cardápio do restaurante.	15
3	O usuário poderá visualizar detalhes sobre um item do menu.	14
4	O usuário poderá adicionar um item do menu no carrinho do seu pedido.	16
5	O sistema deverá ser capaz de mostrar a lista de itens que o usuário adicionou ao seu pedido.	18
6	O usuário poderá adicionar observações ao pedido.	12
7	O usuário poderá adicionar ingredientes extras a um item do seu pedido.	13
8	O usuário poderá excluir um item no carrinho do seu pedido.	10
9	O sistema deverá mostrar o valor total do pedido.	20
10	O aplicativo deverá disponibilizar um espaço para o usuário digitar códigos de promoção.	16
11	O sistema deverá mostrar o tempo previsto de espera para o pedido.	16
12	O usuário poderá cancelar o pedido antes do pagamento.	10

13	O aplicativo deverá mostrar as formas de pagamento disponíveis.	10
14	O usuário poderá escolher a forma de pagamento do pedido.	18
15	O sistema deverá processar o pagamento feito pelo usuário.	30
16	O aplicativo deve permitir fechar pedidos seguidos.	12
17	O aplicativo deve enviar o comprovante de pagamento para o usuário.	20
18	Integração de *software* e eletrônica.	30
19	Implementação da biblioteca *Johnny 5* no backend	30

A Tabela 42 descreve o orçamento detalhado para o software, onde é descrito a quantidade de programadores de cada nível são necessários para desenvolver cada funcionalidade, e o valor da demanda por nível.

Tabela 42 – Custos de software.

ID	Horas	QTD Junior	QTD Pleno	QTD Senior	Valor Junior	Valor Pleno	Valor Senior	Total
1	10	1	2	0	R$ 153,30	R$ 515,60	R$ 0,00	R$ 668,90
2	15	2	1	2	R$ 459,90	R$ 386,70	R$ 1.130,70	R$ 1.977,30
3	14	1	0	1	R$ 214,62	R$ 0,00	R$ 527,66	R$ 742,28
4	16	0	2	0	R$ 0,00	R$ 824,96	R$ 0,00	R$ 824,96
5	18	0	1	1	R$ 0,00	R$ 464,04	R$ 678,42	R$ 1.142,46
6	12	1	1	1	R$ 183,96	R$ 309,36	R$ 452,28	R$ 945,60
7	13	0	2	1	R$ 0,00	R$ 670,28	R$ 489,97	R$ 1.160,25
8	10	2	1	0	R$ 306,60	R$ 257,80	R$ 0,00	R$ 564,40
9	20	2	0	0	R$ 613,20	R$ 0,00	R$ 0,00	R$ 613,20
10	16	0	1	3	R$ 0,00	R$ 412,48	R$ 1.809,12	R$ 2.221,60
11	16	0	1	3	R$ 0,00	R$ 412,48	R$ 1.809,12	R$ 2.221,60
12	10	1	1	0	R$ 153,30	R$ 257,80	R$ 0,00	R$ 411,10
13	10	2	1	0	R$ 306,60	R$ 257,80	R$ 0,00	R$ 564,40
14	18	2	2	3	R$ 551,88	R$ 928,08	R$ 2.035,26	R$ 3.515,22
15	30	2	2	3	R$ 919,80	R$ 1.546,80	R$ 3.392,10	R$ 5.858,70
16	12	1	1	1	R$ 183,96	R$ 309,36	R$ 452,28	R$ 945,60
17	20	0	1	3	R$ 0,00	R$ 515,60	R$ 2.261,40	R$ 2.777,00
18	30	0	1	3	R$ 0,00	R$ 773,40	R$ 3.392,10	R$ 4.165,50
19	30	0	1	2	R$ 0,00	R$ 773,40	R$ 2.261,40	R$ 3.034,80
Total	320	.	.	.	R$ 4.047,12	R$ 9.616,66	R$ 20.691,81	R$ 34.354,87

O total de horas estimadas para o desenvolvimento do software é de 320. Considerando um mês com 168 horas de trabalho (21 dias úteis e 8 horas de trabalho ao dia), o prazo para o desenvolvimento da aplicação é de aproximadamente 1 mês e 19 dias.

E.4 Orçamento geral

A estimativa de custo geral do projeto está descrito na Tabela 43.

Tabela 43 – Orçamento geral.

Geral	**Previsto (R$)**
Estrutura	1.542,36
Automação	6.553,58
Custo de software	34.354,87
Total:	42.450,81

APÊNDICE F – Tópicos de automação

F.1 Tomada de decisão para automação

Sensoriamento e controle

A movimentação do robô até a mesa poderá ser realizada através de triangulação via Wi-fi (LPS [H. S. Hassan and M. Hussein and S. M. and Saad and M. A. Dzahir 2018] ou também equivalente RTLS - Real Time Locating System) ou através da linha a ser seguida por tal (Robô Seguidor de Linha [Simões 2017]). Com isso, para a tomada de decisão a respeito de qual sistema deve ser implementado para a movimentação do robô seria utilizado neste projeto, foi utilizada a matriz de decisão em que são atribuídos a cada variável pesos, e eles são multiplicados pela nota dada a cada opção.

Os pesos atribuídos a cada variável foram escolhidos em relação à sua relevância dentro do projeto e podem variar de -3 a 3. Se tiver influência negativa, o peso recebe um valor negativo, no caso da nota distribuída a cada opção de construção do robô, foi considerado um valor de 1 até 5, sendo 1 a nota mais baixa que poderia receber e 5 a nota mais alta. Para a decisão final de qual implementação seria utilizada, a opção que tivesse no somatório final o valor mais alto seria o escolhido.

Para o monitoramento do robô durante a movimentação serão necessários sensores, aos quais foi atribuído o peso 3 devido a sua importância. Quanto à precisão de localização da mesa em relação ao robô, sua importância tem peso 3. A dificuldade de implementação recebeu o peso negativo de -2 pois com o curto prazo para a construção do robô, uma implementação trabalhosa poderia atrapalhar a execução do projeto.

Ao preço também foi atribuído o peso negativo de -2, pois um custo elevado agregado ao robô vai tornar a sua confecção não viável. A autonomia de movimentação recebeu peso 3 devido ao quanto essa variável agrega ao projeto e, por fim, a dificuldade de implementação do sistema no restaurante recebeu o peso -2 por ser algo que prejudica a viabilidade do projeto. A seguir, pode-se ver a tabela com as variáveis, seus pesos e as notas atribuídas a cada sistema da decisão.

Como pode ser visto na matriz de decisão da tabela Tabela 44, a diferença entre o robô seguidor de linha é de 5 pontos de vantagem para o RTLS, por isso ele foi escolhido para ser implementado no robô garçom.

Em relação à decisão do controlador a ser usado, foi a quantidade de portas de entrada/saída do dele pode ser considerada como um fator a se observar, visto que serão utilizados diversos sensores e atuadores e é necessário que o controlador consiga realizar a

Tabela 44 – Matriz de decisão para o sistema de localização do robô.

Variáveis	Peso	Seguidor de Linha	RTLS (Real Time Locating System)
Sensores (Necessidade de utilização / Conhecimento acerca deles)	3	5	5
Precisão de localização (Milímetros)	3	4	4
Dificuldade de implementação (Bibliografia existente)	-2	2	4
Preço (Reais)	-2	3	5
Autonomia do sistema quanto a movimentação (liberdade de movimentação / possibilidade de esquivar de obstáculos)	3	2	5
Dificuldade para implementação no restaurante (Instalação do sistema no restaurante)	-2	4	2
Total		15	20

devida comunicação com cada um deles. Para esta variável foi atribuído o peso 3, por sua alta relevância no projeto.

A memória do controlador deve ser capaz de armazenar todo o firmware do robô, portanto, quanto maior a memória, melhor para o projeto. Devido a isto recebeu o peso 3. A velocidade do controlador, ou seja, seu clock, definirá o quão rápido o robô responderá aos sinais recebidos pelos sensores ou pela comunicação Wi-Fi, portanto recebe o peso 2.

A dificuldade de implementação recebeu um peso negativo -2 pois, como foi dito anteriormente, uma implementação trabalhosa poderia atrapalhar a execução do projeto. O preço do controlador recebeu o peso negativo de -2 e à quantidade de interfaces do controlador foi atribuído o peso 2.

Na Tabela 45 está o comparativo entre os controladores escolhidos para o projeto: Raspberry Pi Zero W[1] e Esp32 Wroom 32-E[2].

Com isso, pôde-se criar a tabela referente a matriz de decisão do controlador, que pode ser vista na Tabela 46.

Como resultado, a diferença entre o Raspberry Pi Zero W e o Esp32 Wroom 32-E é de 6 pontos de vantagem para o Esp32, por isso ele foi escolhido para ser o controlador do robô garçom.

Alimentação

Para as decisões de alimentação do produto, houve a necessidade do método matriz de decisão para a escolha do tipo de carregamento, bateria e BMS. Por falta de informações nesta fase inicial do projeto, não foram avaliadas as baterias e BMS, de maneira que o dimen-

[1] Disponível em: <https://www.raspberrypi.org/products/raspberry-pi-zero/>
[2] Disponível em: <https://www.espressif.com/en/products/modules/esp32>

Tabela 45 – Parâmetros técnicos para a escolha do controlador.

	Raspberry Pi Zero W	Esp32 Wroom 32-E
CPU	Single core	Dual core
Memória	512 Mbit	16 Mbit (flash), 520 kbit (SRAM)
Pinos	40	38
Clock	1 GHz	240 MHz
Bluetooth	Bluetooth Classic 4.1 e Bluetooth LE	v4.2 BR/EDR and Bluetooth Low Energy (BLE)
Wifi	802.11n de 2,4 GHz	802.11n de 2,4 GHz
Interfaces	UART,SPI, I2C,GPIO,ADC,SD card	SD card, UART, SPI, SDIO, I2C, LED PWM, Motor PWM, I2S, IR, pulse counter, GPIO, capacitive touch sensor, ADC, DAC
Preço(Reais)	R$ 160,00	R$ 65,00

Tabela 46 – Matriz de decisão para o controlador do robô.

	Peso	Raspberry Pi Zero W	Esp32 Wroom 32-E
Portas (Comporta todos os sensores / atuadores do projeto)	3	5	5
Memória (Mb)	3	5	2
Velocidade (Mhz)	2	5	2
Dificuldade de implementação (Bibliografia existente)	-2	3	2
Preço (Reais)	-2	3	1
Interfaces (Quantidade)	2	2	4
Total		32	38

sionamento dos motores e do peso do robô, além da definição de todos os componentes é necessário para a escolha. Desta forma, foi realizada a matriz de escolha de carregamento, apresentada na Tabela 47.

Tabela 47 – Matriz de decisão sobre sistema de carregamento

Carregamento	Velocidade	Eficiência	Autonomia no funcionamento (x2)	Ocupação de espaço	Notas
Plug	3	3	2	3	54
Contato	2	2	4	2	32
Indução	1	1	6	1	6

Foram analisados três formas de carregamento: plug-in, indutivo ou contato. Dos analisados o que obteve o menor resultado foi o carregamento por indução, no entanto, devido ao requisito geral do projeto definir que o carregamento deve ser realizado sem o contato humano, este foi o escolhido.

O carregamento por contato foi excluído devido a necessidade de se colocar câmeras no corpo do robô para a identificação e posicionamento correto do mesmo, e dado a movimentação e posicionamento do robô ser realizado por triangulação, este posicionamento seria de grande dificuldade.

Apesar da baixa nota do carregamento por indução, ele foi escolhido devido a autonomia do robô, que poderá operar sozinho e com margens de erro ao se encaixar na base, ao contrário do carregamento por contato que necessita de um grande refinamento para o encaixe preciso na base. Já o contato por plug foi descartado como carregamento principal devido a necessidade de uma pessoa ficar responsável pelo carregamento do robô.

Embora o carregamento por plug-in tenha sido excluído, estuda-se neste momento a possibilidade de o adicionar como uma segunda alternativa, em caso de descarregamento total.

F.2 Simulações eletro-eletrônicas

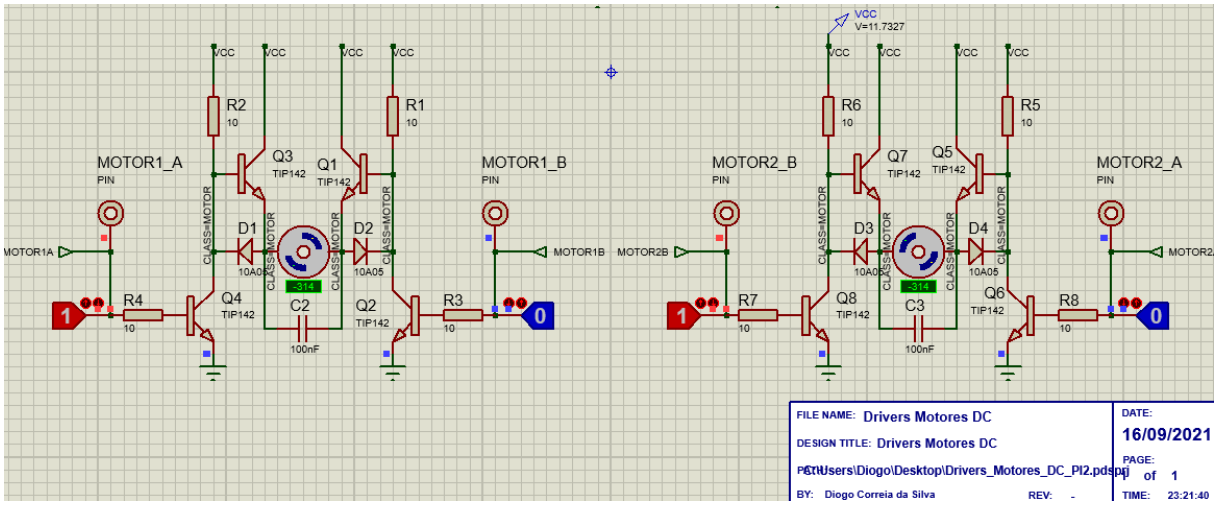
Os códigos utilizados para as simulações apresentadas aqui, podem ser vistos na seção I.2.2.

F.2.1 Motores

Para realizar a simulação dos circuitos responsáveis pela atuação dos motores DC do robô, foi utilizado o software Proteus, onde foram colocadas entradas lógicas nas conexões do microcontrolador com a placa. Dessa forma, quando a entrada está em 1 simboliza que a ESP32 enviou um sinal de PWM para o pino correspondente do motor, e quando está 0 subentende-se que a ESP32 desligou o sinal de PWM naquele pino.

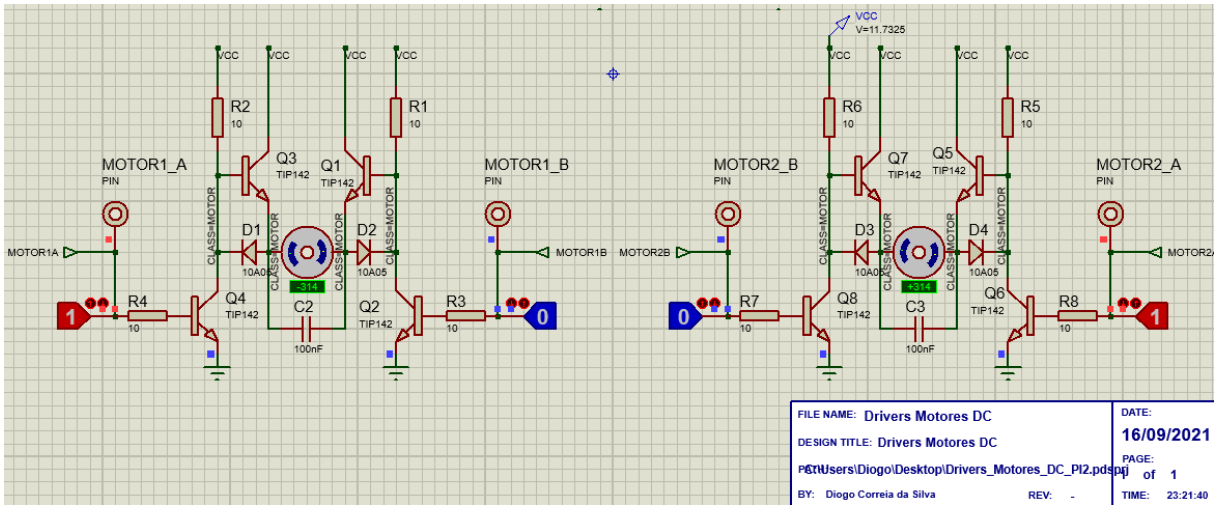
O número encontrado abaixo do motor representa a quantidade de rotações do motor por minuto (RPM), e possui um símbolo atrelado (+ ou -), sendo que o símbolo + antes do número representa que o motor está girando no sentido horário, enquanto o símbolo - antes do número representa que o motor está girando no sentido anti-horário. A seguir pode-se ver algumas capturas de tela referente a essas simulações, especificando a direção para onde o robô está indo ou se a bandeja está subindo ou descendo.

Figura 62 – Motores de movimentação - para frente.



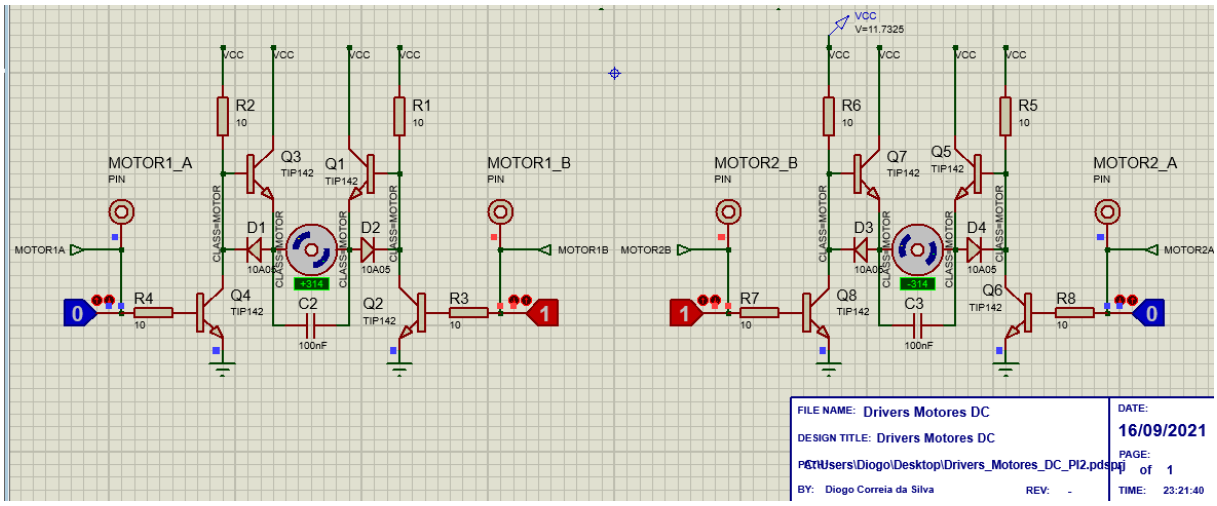
Fonte: Autores.

Figura 63 – Motores de movimentação - virando à direita.



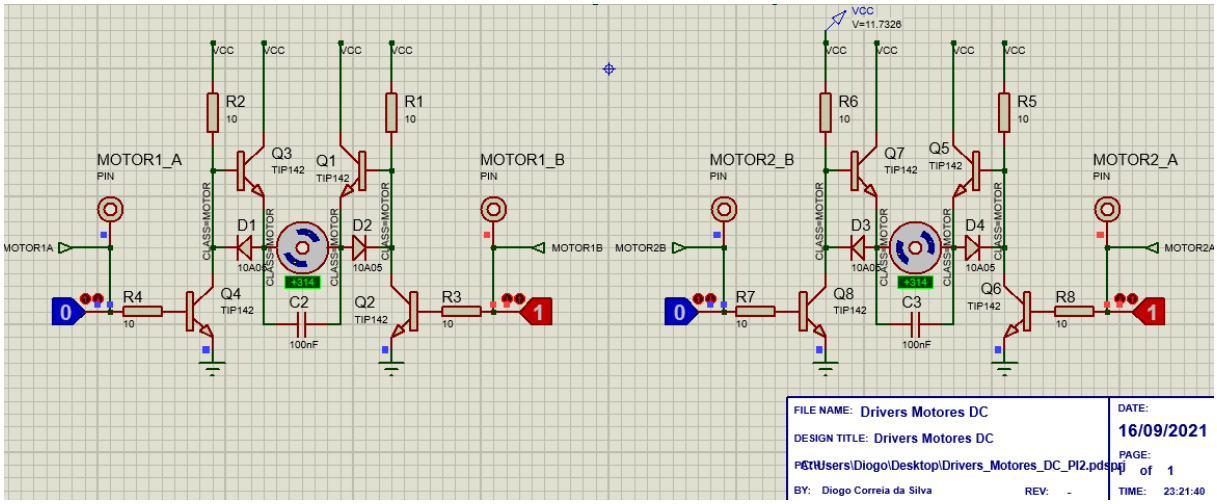
Fonte: Autores.

Figura 64 – Motores de movimentação - virando à esquerda.



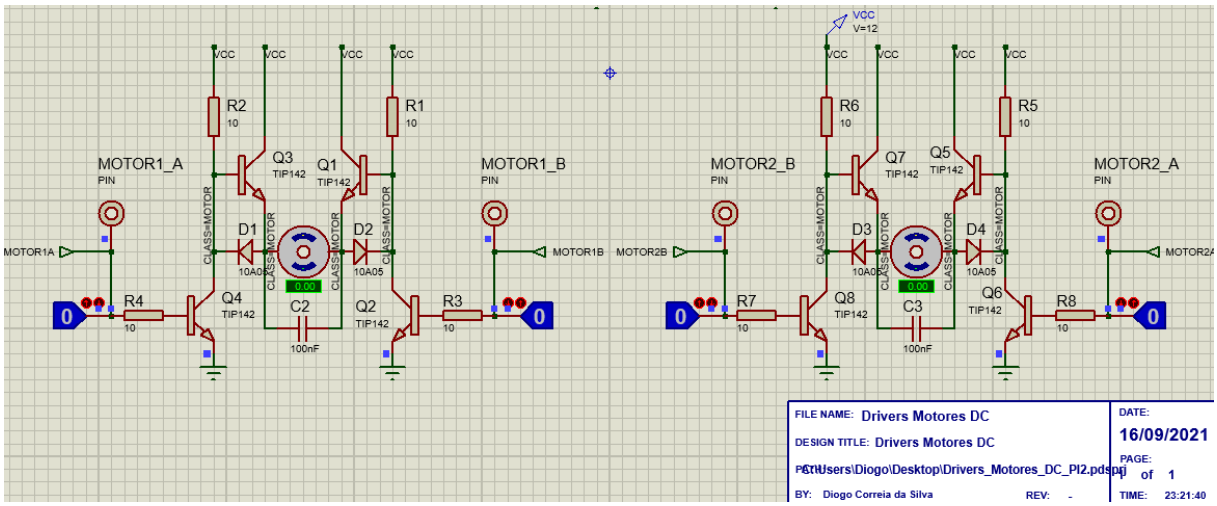
Fonte: Autores.

Figura 65 – Motores de movimentação - dando ré.



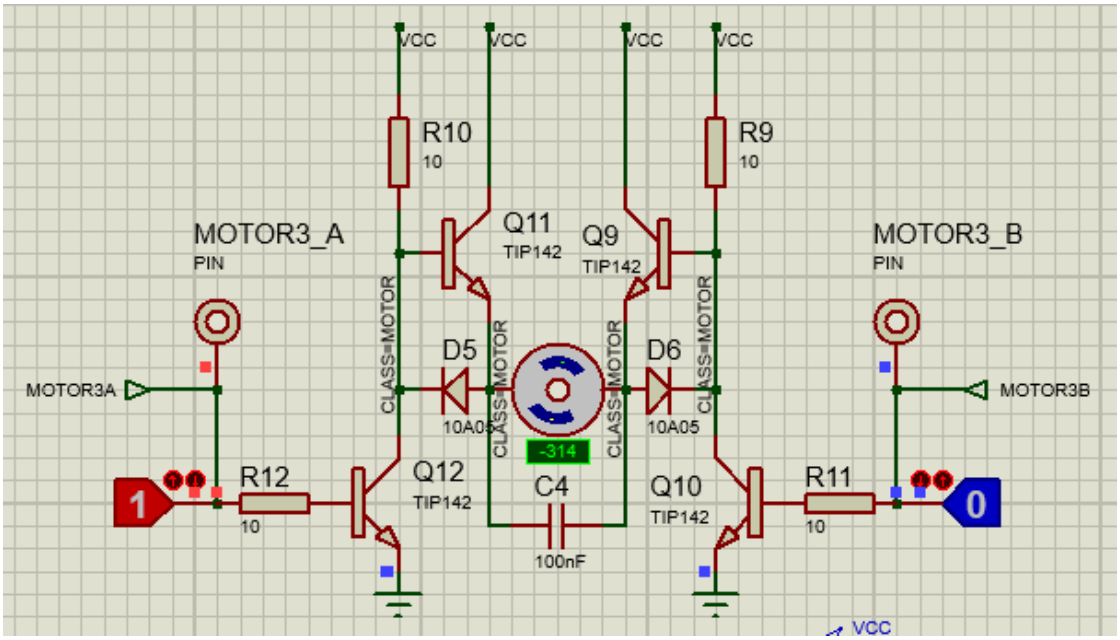
Fonte: Autores.

Figura 66 – Motores de movimentação - parados.



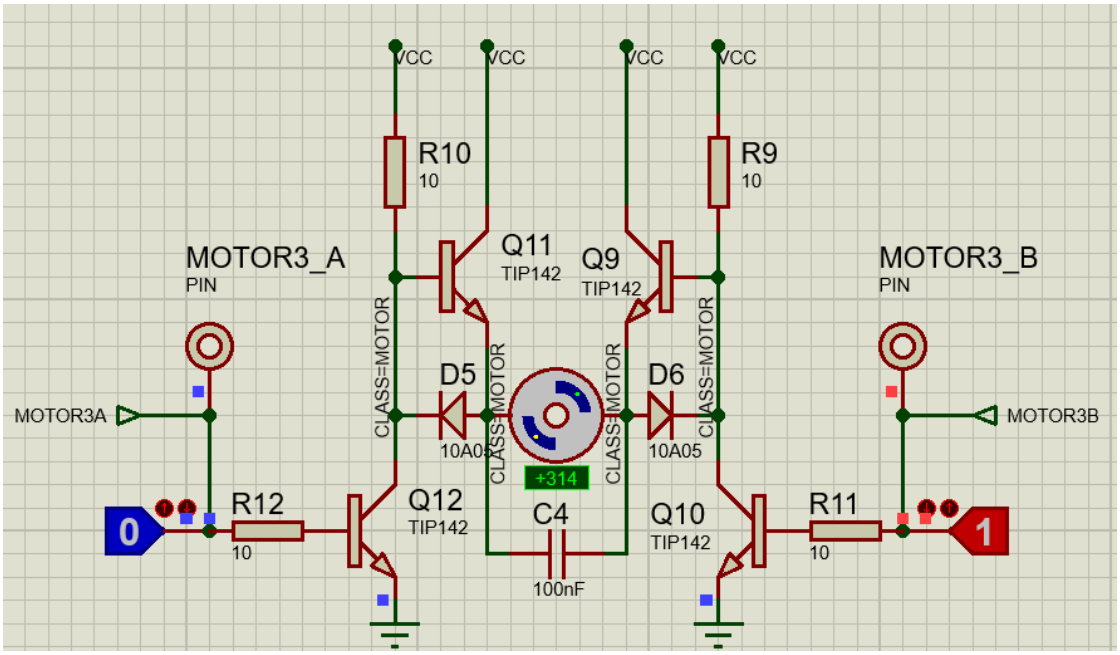
Fonte: Autores.

Figura 67 – Motor da bandeja - subindo.



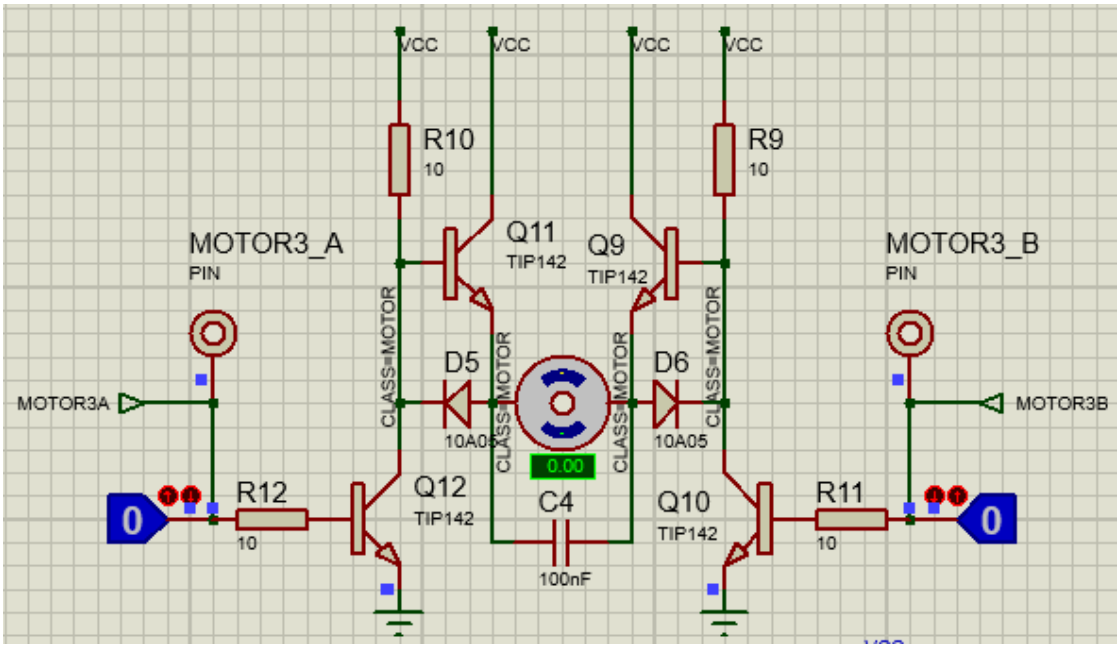
Fonte: Autores.

Figura 68 – Motor da bandeja - descendo.



Fonte: Autores.

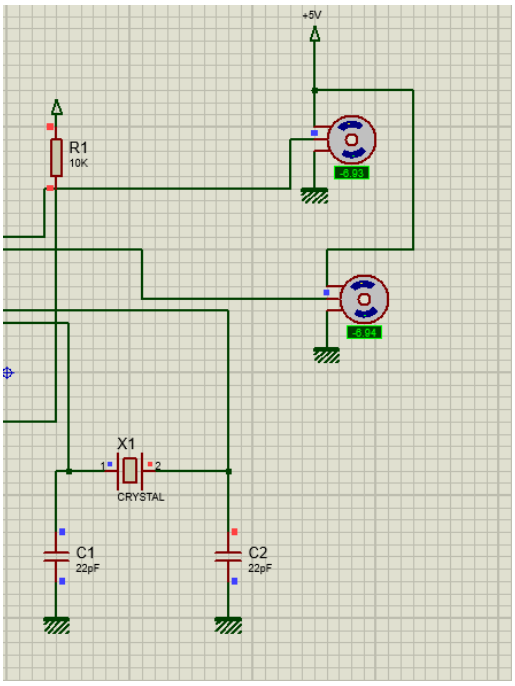
Figura 69 – Motor da bandeja - parado.



Fonte: Autores.

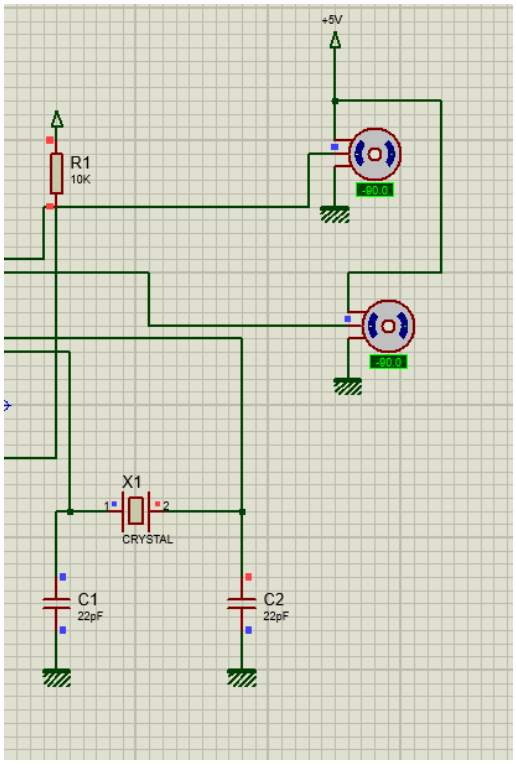
A simulação dos motores da tampa foi realizada no software *Proteus* como pode ser visto na Figura 70 e na Figura 71 , os motores realizam o movimento de abertura e fechamento da tampa , eles se movimentam em conjunto (os dois na mesma direção ao mesmo tempo).

Figura 70 – Motor da tampa - abrindo.



Fonte: Autores.

Figura 71 – Motor da tampa - fechando.

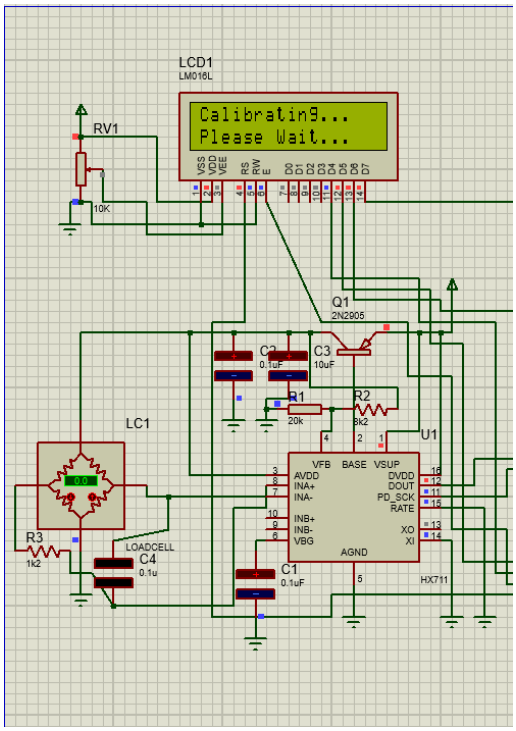


Fonte: Autores.

F.2.2 Sensor de peso

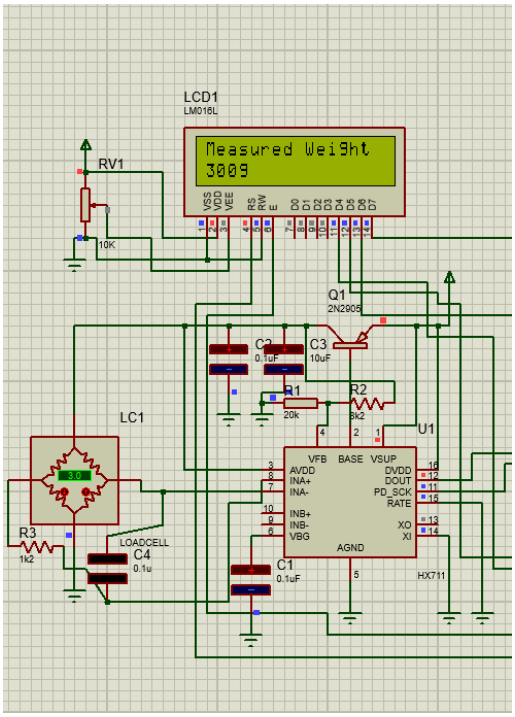
A célula de carga também foi simulada no software *Proteus* na Figura 72 é possível ver a simulação do código de calibração , para a melhor visualização nas simulações foi inserido o display LCD , a célula de carga leva alguns instantes para calibrar a balança, a Figura 73 mostra a simulação da célula de carga em funcionamento, considerando um peso de 300g.

Figura 72 – Calibração da célula de carga.



Fonte: Autores.

Figura 73 – Medição da célula de carga.

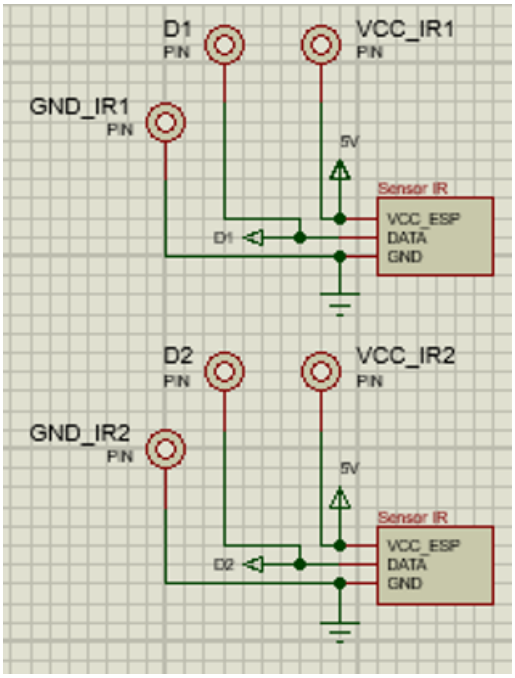


Fonte: Autores.

F.2.3 Sensores infravermelho e Ultrassônico

Para a simulação dos sensores infravermelho e ultrassônico foi utilizado o software *Proteus*. A Figura 74 mostra a simulação dos dois sensores infravermelhos indicando que os sensores foram ativados.

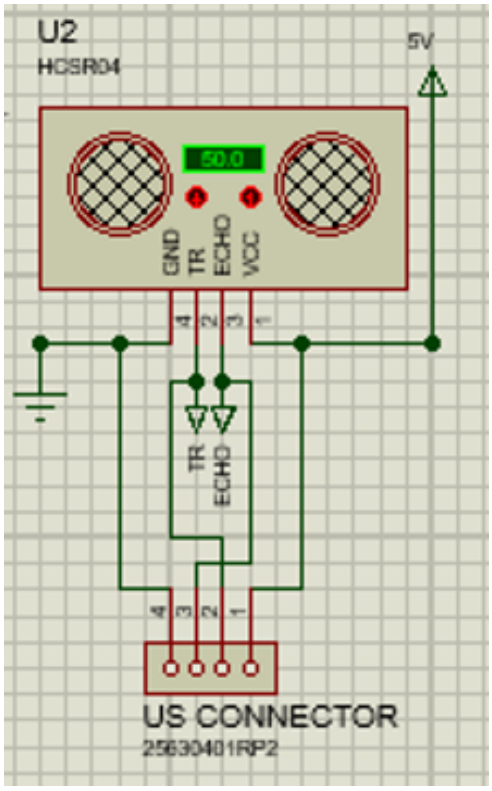
Figura 74 – Infravermelho.



Fonte: Autores.

Para a simulação do sensor ultrassônico apresentada na Figura 75, foi simulado que um obstáculo estaria a uma distância de 50 cm do robô.

Figura 75 – Sensor ultrassônico.



Fonte: Autores.

F.2.4 Bobina

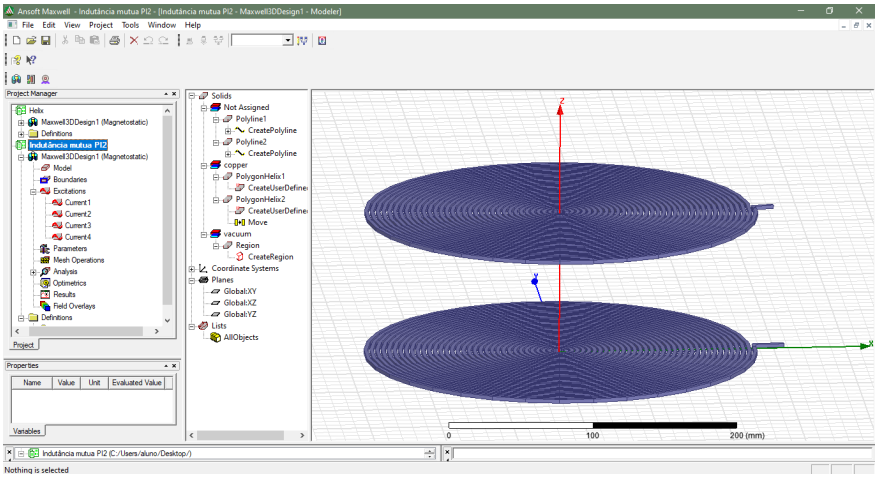
A simulação das bobinas foi realizada no *Ansoft Maxwell*, mostrada na Figura 76, onde pôde verificar a indutância mútua desejada. Por se tratar de bobinas semelhantes, a indutância mútua é igual a indutância da bobina separada, 100µH. Este estudo foi feito a partir de um modelo simples, que não considera os protetores térmicos e carcaça que envolverá a bobina. Outro fator no estudo da indutância mútua, como pode-se observar na fórmula abaixo, é o fator de acoplamento magnético K que varia de 0 a 1, sendo 0, as bobinas desacopladas e 1, totalmente acoplado. Em condições ideais, as bobinas não são totalmente acopladas, logo o K foi definido como $0,9$, portanto a indutância mútua M será de 90µH.

$$M = k\sqrt{L_1 L_2} \tag{F.1}$$

Sendo: M = indutância mútua (µH) K = Fator de acoplamento magnético L = Indutância individual de cada bobina (µH)

No estudo dos indutores, não foi possível calcular o erro entre a bobina simulada e real devido a não construção do protótipo, causando incertezas geradas pela bobina real vide as técnicas de fabricação dos enrolamentos e perdas por um acoplamento fora das condições desejadas.

Figura 76 – Captura de tela do ambiente de simulação Ansoft Maxwell.



Fonte: Autores.

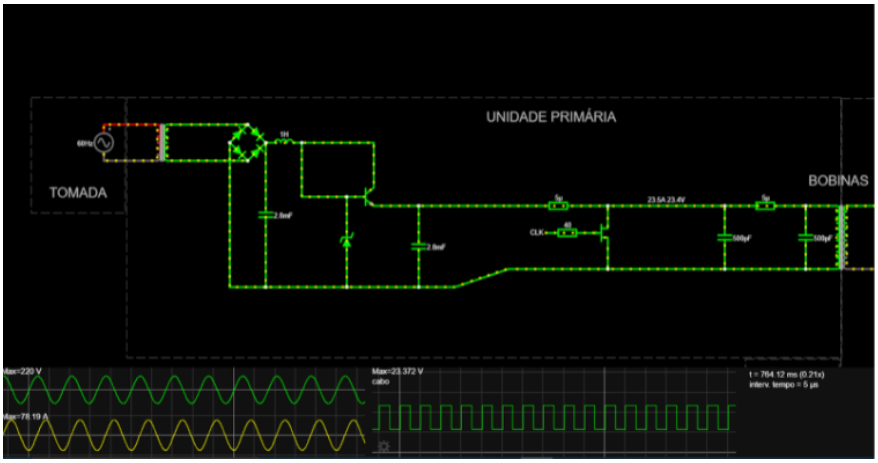
F.2.5 Carregamento

Para a simulação dos circuitos de carregamento indutivo e por plug, foi utilizado o software *Falstad*.

Foi realizada a simulação dos carregamentos por indução e por plug, em que o carregamento por indução foi dividido nas suas duas unidades para verificação de cada etapa.

A unidade 1 é composta de um transformador, circuito oscilador, amplificador de potência e a bobina primária. Na simulação foram observadas as oscilações de tensão entre 23,4 e 11,4V permanecendo com uma corrente de 23A. Essa oscilação é realizada pela fonte de corrente contínua em *clock* abastecida com uma pilha de 6V. A Figura 77 apresenta a simulação.

Figura 77 – Simulação da Unidade 1.

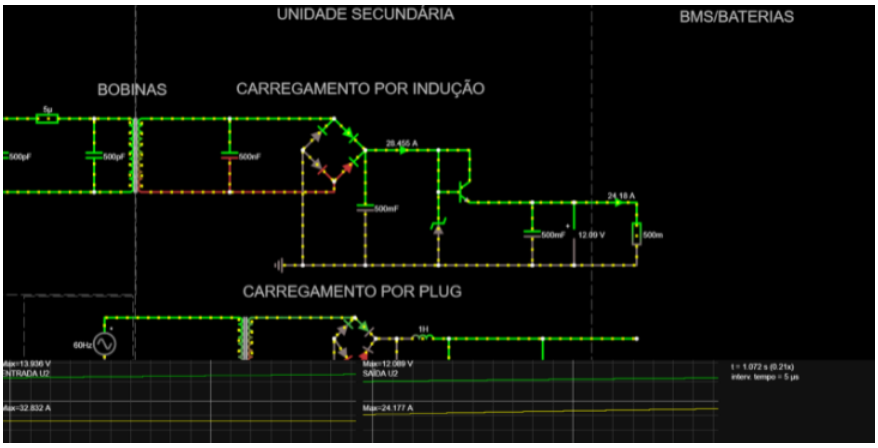


Fonte: Autores.

A unidade 2 apresenta um retificador, filtro passa-baixa e ceifador. Como resultado da simulação, o circuito apresentou uma voltagem de 12V porém com a corrente variando

no tempo entre 11 e 24A. A sua simulação é apresentada na Figura 78.

Figura 78 – Simulação da Unidade 2.

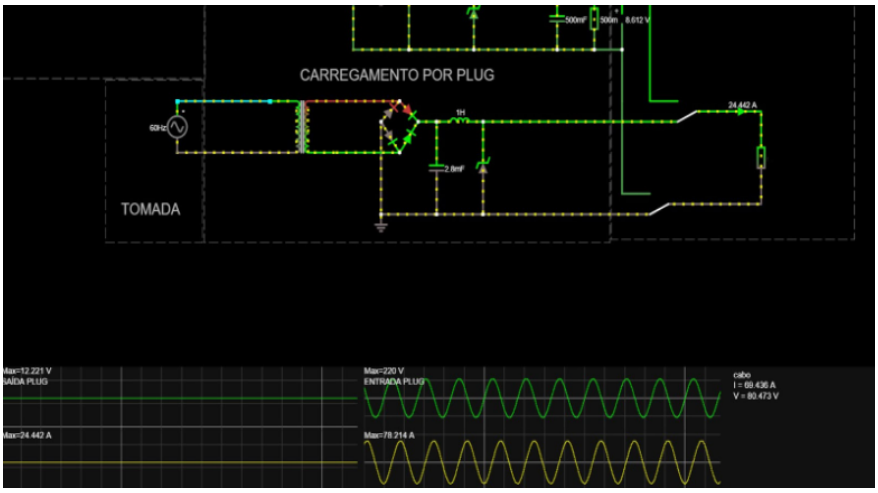


Fonte: Autores.

A simulação realizada para o carregamento por plug teve entrada de 110 ou 220V. É composto por um transformador e um regulador de tensão.

A Figura 79 apresenta a simulação do circuito, e as visualizações na entrada e saída do carregamento. O resultado da simulação apresentou na saída para o BMS e baterias o valor de 12,221V e 24,442A.

Figura 79 – Circuito de Simulação do Carregamento por Plug.



Fonte: Autores.

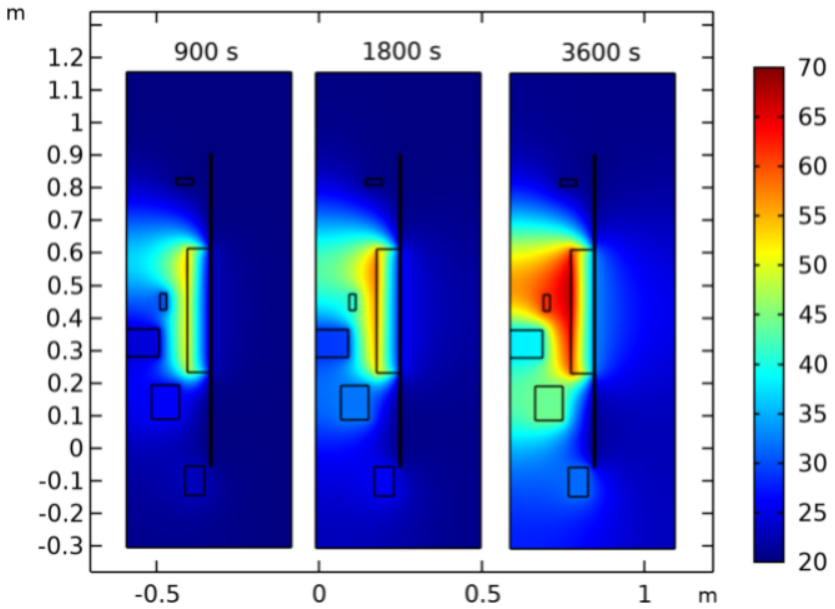
F.2.6 Modelagem térmica

O interior do robô Garçom possui componentes e equipamentos que produzem calor. Dado este fato, foi realizada uma análise térmica que levou em consideração uma condição de operação hostil e inadequada, justamente para analisar os limites de exposição dos componentes. Assim, os parâmetros utilizados foram de:

- Geometria construída a partir da definição final do posicionamento dos componentes;
- Período aplicado de 1 hora sem interrupções à máxima potência de operação dos componentes e carregamento indutivo pelo período de 5 minutos;
- Robô sem dissipação de calor externa por convecção forçada.

Como resultado dessa análise, foi obtida a Figura 80, que apresenta o comportamento do campo de temperaturas no interior do robô. A temperatura máxima encontrada foi de 67 °C. Este valor é identificado na superfície da bobina de carregamento indutivo após uma hora de funcionamento. Enquanto isso, as temperaturas ao longo do restante do robô ficam majoritariamente entre 30 e 40 °C.

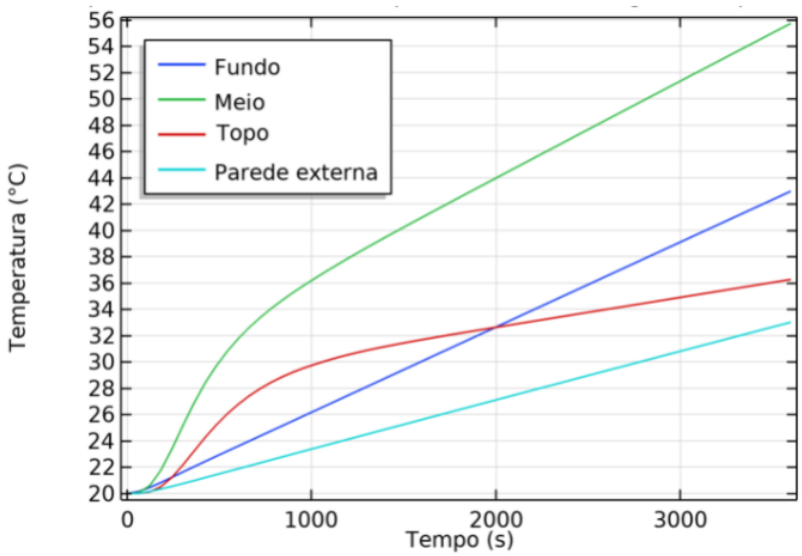
Figura 80 – Campos de temperatura ao longo do robô em diferentes períodos de tempo.



Fonte: Autores.

Os resultados para cada região de análise podem ser melhor observados no gráfico da Figura 81. O fundo, o topo, o centro e a parede externa foram especialmente selecionados para comparação. Foi considerado para a análise que o robô está parado sem um fluxo de ar que possa resfriá-lo.

Figura 81 – temperatura em diferentes pontos do robô ao longo do tempo.



Fonte: Autores.

Dos equipamentos utilizados, o que apresenta menor temperatura limite para operação são as baterias, que suportam temperaturas de até 45 °C. Na simulação realizada, a bateria chegou a 44 °C após uma hora de funcionamento, valor muito próximo ao limite, mas não preocupante por se tratar do pior cenário proposto. Já para os períodos de tempo de 15 e de 30 minutos, foram atingidas as temperaturas de 25 e 30 °C, sendo tais valores de temperatura satisfatórios.

Assim, a partir do modelo térmico construído, não se faz necessária a instalação de *coolers* no sistema, o que ocuparia mais espaço e consumiria mais energia no sistema. Também não se faz necessária aplicação de aberturas adicionais na estrutura, já que as aberturas definidas são suficientes para a dissipação do calor gerado pelos componentes.

APÊNDICE G – Estimativa de Consumo Energético

Ao considerar o funcionamento do robô durante os horários de serviços, foi pré-estabelecida uma estimativa de consumo, de maneira a considerar os componentes principais já descritos. Para esta estimativa estipulou-se três horários de utilização diferentes: quatro horas por dia; oito horas por dia; e doze horas por dia. Em continuidade, extrapolou-se os consumos para a utilização destes horários para o mês.

É apresentada uma estimativa, dados os cenários apresentados. É importante ressaltar que estes dados são apenas estimações iniciais feitas no Ponto de Controle 1, a análise com informações completas será feita futuramente.

Tabela 48 – Estimativa de consumo energético.

Sistema	**Componente**	**Consumo** (W)	**Consumo** ($W \times h$)			**Consumo** ($W \times h \times$mês)		
			4h/dia	**8h/dia**	**12h/dia**	**4h/dia mês**	**8h/dia mês**	**12h/dia mês**
Sensores	Sensor Ultrasônico	12	24	36	360	720	1080	
	Sensor Reflexivo							
	Sensor Infravermelho							
	Célula de Carga							
	Acelerômetro							
Movimentação	Motor roda 1	48	192	384	576	5760	11520	17280
	Motor roda 2	48	192	384	576	5760	11520	17280
	Motor tampa	5,5	22	44	66	660	1320	1980
	Motor Atuador	5,5	22	44	66	660	1320	1980
Total	**10**	**110**	**440**	**880**	**1320**	**13200**	**26400**	**39600**

APÊNDICE H – Escopo de Software

Para definir o escopo do *software*, os integrantes da equipe de *software* levantaram os objetivos de negócio e de produto cruciais para o sistema. Também foi desenvolvido um Rich Picture para melhor visualização da equipe sobre o que seria desenvolvido. Dessa forma, foi possível delimitar o escopo do aplicativo.

Tabela 49 – Objetivos de Negócio

ID Objetivo	Descrição
ON01	Prover segurança para pagamento
ON03	Guiar robô até a mesa
ON08	Prover segurança para pagamento
ON10	Atender normas de segurança alimentar

Tabela 50 – Objetivos de Produto

ID Objetivo	Descrição
ON01	Disponibilizar o menu do restaurante para que o cliente faça seu pedido.
ON06	Dar tempo ao cliente decidir sobre o pedido sem gerar um desconforto (ansiedade).
ON08	Diminuir os custos com mão-de-obra física.
ON09	Reduzir a propagação de erros.
ON11	Possibilitar a realização de pedidos diretamente da mesa.

H.1 Metodologia

Quanto à metodologia, a equipe de software optou por unir aspectos das metodologias ágeis Scrum, KANBAN e *SMART*, com o objetivo de utilizar uma metodologia que melhor se adaptasse ao nosso contexto.

A metodologia Scrum está sendo empregada através do *Product Backlog*, que reúne todos os requisitos do sistema, As *Sprints*, que são os ciclos de tarefas que duram em média de 5 a 7 dias, e a *Sprint Planning*, onde a equipe se reúne para discutir sobre a última *Sprint* e planejar a próxima. Já o KANBAN é utilizado na organização do quadro de *Sprints* do núcleo de software, que é dividido nas colunas de: *To do*, *Doing* e *Done*. Já a metologia *SMART* é utilizada na criação das tarefas, em que é importante frisar o objetivo e o que deve ser feito na atividade, além do prazo em que deve ser entregue.

H.2 Elicitação e priorização de requisitos de Software

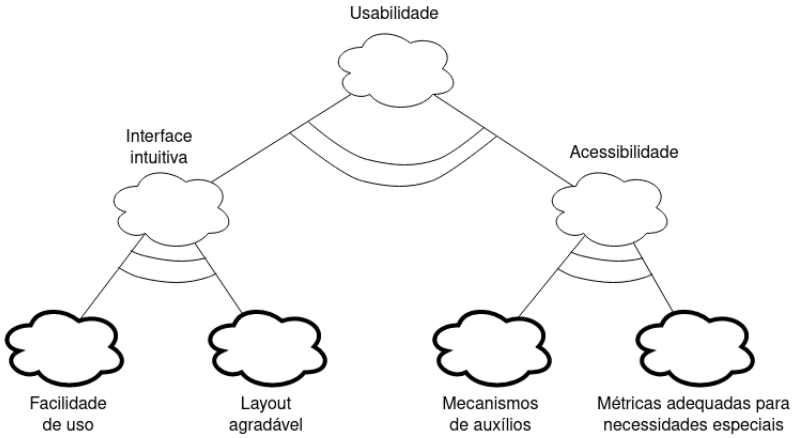
Os requisitos de software citados na Seção 4.2 foram elicitados através de duas técnicas. Os requisitos principais do sistema foram levantados pela equipe de software durante um *Brainstorming*. A técnica de *brainstorming* - tempestade de ideias - consiste em uma dinâmica em grupo, onde há troca de ideias, para unir informação e estimular o pensamento criativo para resolução de problemas. Durante a dinâmica, todos os integrantes da equipe expuseram sua visão do sistema e o que seria crucial para o sistema.

Posteriormente, foi aplicada a técnica de introspecção, que se baseia em imaginar que tipo de sistema eu iria querer se fosse utilizar o sistema em questão. Foi pedido que cada integrante, individualmente, se imaginasse como usuário do sistema para o robô garçom, e que listasse os requisitos funcionais e não-funcionais que achasse essencial e viável para o software.

Após a elicitação dos requisitos, foram utilizadas as técnicas NFR *Framework*, MoSCoW, *First Things First* e Especificação Suplementar, para priorizar os requisitos levantados.

O NFR *Framework* é uma abordagem orientada a processos, que procura tornar explícitas as metas a serem atingidas para que se obtenha sucesso na implementação de um requisito não funcional. É representado pelo SIG (*Softgoal interdependency graph*), que descreve como os *softgoals* são decompostos. Um exemplo prático é o NFR de usabilidade (Figura 82), nele é representado as necessidades que o sistema deve cumprir para chegar ao nível de satisfação do público alvo, que neste caso são os clientes do restaurante.

Figura 82 – NFR referente a usabilidade.



Fonte: Autores.

A técnica *MoSCoW* foi utilizada para ordenar os requisitos funcionais e não funcionais, em escala de: Deve-se implementar esse requisito (*Must have*); deveria ter este requisito (*Should have*); poderia ter este requisito (*Could have*); e não terá este requisito no momento, mas pode ser implementado mais tarde (*Won't Have*).

Assim como o *MoSCoW*, o *First Things First* ordena e prioriza os requisitos do sis-

tema, levando em consideração uma escala quantitativa, quanto à benefícios, penalidades, custos e riscos

Por fim, as Especificações Suplementares capturam os requisitos do sistema relacionados à: requisitos legais e de regulamentação e padrões de aplicativo; atributos de qualidade do sistema, incluindo requisitos de usabilidade, confiabilidade, desempenho e suportabilidade; e outros requisitos, como aqueles para os sistemas e ambientes operacionais, compatibilidade com outro software e restrições de design.

Todos as técnicas de elicitação e priorização estão documentadas e podem ser acessadas através da WIKI do projeto.

H.3 Decisões de *Software*

H.3.1 Arquitetura do *Software*

As escolhas quanto à arquitetura do software se deram levando em consideração a facilidade de integração entre os componentes. Um exemplo disso é a escolha da linguagem **JavaScript**, que é a mesma utilizada no *Framework* Johnny Five, que fará integração entre o robô e o computador central da cozinha.

A escolha do **MVC** (*Model-View-Controller*) como a arquitetura do sistema se deu pelo fato dessa arquitetura proporcionar uma separação clara entre as camadas de visualização e regras de negócios, o que torna mais fácil a manutenção do sistema.

A definição das ferramentas utilizadas para o desenvolvimento do Back e Front-end, se deu por conta da familiaridade da equipe com as plataformas *NodeJS* e *React*, ambos utilizados para a execução de códigos JavaScript.

Em específico, o ***NodeJS***, que está sendo utilizado através do pacote *Express.js*, foi escolhido por ser eficiente na implementação de aplicações em tempo real, que precisam transferir mensagens de um lado para o outro de forma rápida. As mensagens no caso do sistema Robô Garçom, serão os pedidos. Além disso, o ***NodeJS*** é uma API com NoSQL por trás, o que facilita a integração com o banco de dados MongoDB.

Já o ***React***, que será utilizado em conjunto com o módulo *React Native*, foi definido para o desenvolvimento da interface do sistema por sua flexibilidade - facilidade de integração com outras bibliotecas e *Frameworks* - e pelo uso de componentes reutilizáveis.

Para gerenciar e armazenar todas as informações utilizadas pelo robô e o restaurante, o bando de dados **MongoDB** se mostrou a alternativa mais adequada, já que é escalável horizontalmente, o que permite a ampliação de todo o seus sistema caso seja necessário. Suporta também um alto índice de consultas e envios de dados em JavaScript, pois trata-se de um banco do tipo NoSQL.

H.3.2 Integração com automação

A **Johnny Five** é uma biblioteca dedicada a programação de robôs desenvolvida em JavaScript, que visa agilizar o funcionamento de componentes eletrônicos com o objetivo de dar vida ao robô. A escolha dessa biblioteca se deu pelo fato de, por ser escrito em JavaScript, facilita o uso por toda a equipe desenvolvedora e possui uma ampla documentação mantida por comunidades *open-source* e por sua criadora, Bocoup.

Após a realização de uma pesquisa entre diversos algoritmos de melhor caminho - como o do caixeiro viajante e de Johnson -, o **Algoritmo de *Dijkstra*** foi escolhido como aquele que será implementado no projeto. Apesar de se limitar a grafos com pesos apenas positivos, este algoritmo é mais rápido na sua execução em comparação à outros algoritmos existentes, por esse motivo torna-se adequado para a proposta do sistema, que necessita de transmissão rápida de respostas.

Realizando a inserção das mesas no sistema central e suas respectivas localidades, o **Algoritmo de *Dijkstra*** será executado toda vez que o robô receber o comando de entregar pedido. No momento de enviar as informações do pedido constará a qual mesa ele pertence. Essa informação que será aplicada no **Algoritmo de *Dijkstra*** e retorna toda informação para ser enviada pelo **Johnny Five**.

O computador central é o responsável por realizar o cálculo e enviar o melhor percurso ao robô, dessa maneira ocorre uma economia energética sobre a bateria do robô.

O primeiro componente eletrônico a ser integrado no robô é a **ESP32**, mais especificamente seu componente interno que permite a conexão com o *Wi-fi*, principal ator para implementar a localização do robô dentro do restaurante e também receber os pedidos e a rota de entrega enviados pelo computador central.

APÊNDICE I – Códigos

I.1 Códigos para *Software*

Os códigos a seguir foram implementados pelo núcleo de *software* para dar início ao desenvolvimento do sistema **Robô Garçom**.

I.1.1 Docker

A programação teve seu primeiro passo com a criação do **Docker**, para padronizar os ambientes de desenvolvimento, evitando o conflito entre as diversas versões, por exemplo do **NodeJs**.

```
version: "3.5"

services:
  backend:
    build:
      context: ./
    volumes:
      - .:/usr/src/app
    ports:
      - 3000:3000
    depends_on:
      - mongo

  mongo:
    image: mongo:latest
    restart: always
    ports:
      - "27017:27017"
    environment:
      MONGO_INITDB_ROOT_USERNAME: root
      MONGO_INITDB_ROOT_PASSWORD: password
      MONGO_INITDB_DATABASE: database

  mongo-express:
    image: mongo-express
    restart: always
    ports:
      - 8081:8081
    depends_on:
      - mongo
```

```
31      environment:
32        ME_CONFIG_MONGODB_SERVER: mongo
33        ME_CONFIG_MONGODB_ADMINUSERNAME: root
34        ME_CONFIG_MONGODB_ADMINPASSWORD: password
35        ME_CONFIG_MONGODB_PORT: 27017
```

Código - Estruturação do Docker.

I.1.2 Controller de autenticação

O primeiro passo para a utilização da aplicação é realizar o login de administrador, essa ação é verificada pela classe *AuthController*.

```
1  import User from "../models/User";
2  import { generateAccessToken } from "../services/jwt";
3
4  class AuthController {
5    register = async (req, res) => {
6      const { username, email, password } = req.body;
7
8      try {
9        await User.create({ username, email, password });
10       res.json({
11         status: "success",
12         data: null,
13         message: "Registration Successful",
14       });
15     } catch (error) {
16       res
17         .status(401)
18         .json({ status: "error", data: null, message: error.message });
19     }
20   };
21
22   login = async (req, res) => {
23     const { username, password } = req.body;
24     const user = await User.findOne({ username });
25     if (user) {
26       const match = await user.comparePassword(password);
27       if (match) {
28         const token = generateAccessToken(user.id);
29         return res.json({
30           status: "success",
31           data: { token },
32           message: "Authentication Successful",
33         });
34       }
35     }
```

```
36      return res.status(400).json({
37        status: "error",
38        data: null,
39        message: "Invalid Credentials",
40      });
41    };
42
43    authUser = (req, res) => {
44      return res.json({
45        status: "success",
46        data: { user: req.user },
47        message: null,
48      });
49    };
50  }
51
52  export default new AuthController();
```

Código - Controller de autenticação.

I.1.3 Banco de dados

Todos as informações geradas no *Back-end* são salvas no **MongoDB**, o banco utilizado pela aplicação. O banco entra em operação no momento em que o Docker é executado, desse modo é possível realizar testes de utilização na fase de desenvolvimento.

```
1  import app from "./server";
2  import mongoose from "mongoose";
3
4  const dbUrl = "mongodb://root:password@mongo:27017";
5
6  mongoose
7    .connect(dbUrl, {
8      useNewUrlParser: true,
9    })
10   .then(() => {
11     app.server();
12   })
13   .catch((err) => {
14     console.error("App starting error:", err.stack);
15     process.exit(1);
16   });
17
18 export default app;
```

Código - Estruturação do banco de dados MongoDB.

Outros códigos relacionados ao desenvolvimento do back-end do projeto podem ser visualizados no repositório da equipe.

I.1.4 Johnny 5

```
var VirtualSerialPort = require('udp-serial').SerialPort;
var firmata = require('firmata');
var five = require("johnny-five");
var mapa = require('./dijkstra')

// var sp = new VirtualSerialPort({
//   host: '192.168.1.103'
// });

var sp = new VirtualSerialPort({
  host: 'localhost',
  type: 'udp4',
  port: 41234
});

// console.log('IO Ready');

var io = new firmata.Board(sp);
io.once('ready', function(){
    console.log('IO Ready');
    io.isReady = true;

    var board = new five.Board({io: io, repl: true});

    // teste para conectivade da placa
    board.on('ready', function(){
        console.log('five ready');
        var led = new five.Led(13);
        led.blink();
    });
});

var wifi = new firmata.Board(sp);

// var wifi = new five.Sensor({
//   pin: "A0",
//   freq: 250,
//   threshold: 5
// });


wifi.once("ready", function() {

  wifi.on("change", function() {
    mapa();
  });
```

```
});
```

Código - Para utilização da Johnny-Five.

I.2 Códigos para a automação

Serão mostrados a seguir os códigos utilizados para teste e implementação dos componentes do robô e validação de suas funcionalidades. Eles foram escritos em C++ (.ino) para os microcontroladores ESP32. Eles estão finalizados e estão disponíveis no repositório [Gabriel, Letícia e Diogo 2021].

I.2.1 Comunicador *wireless*

Implementa a busca de SSIDs, medição de RSSI, conexão de cliente (para o servidor do computador central) via *websocket*, recebimento e envio de dados e calculo de distancias dos dispositivos, trilateração (x_p, y_p), medição da direção do robô (θ_d) e comando para o controlador.

O código foi compilado e implementado via Arduino IDE[1].

```
/*Autor: Gabriel Borges Pinheiro
 Código não completo.
Falta: - obtenção da direção do robô (via magnetômetro)
       - Comando de movimentação das rodas para o Controlador
*/
//Necessário preencher essas definições com os dados do estabelecimento!
#define SSID_CENTRAL  "SSID_do_roteador" // Nome da rede wi-fi
#define PASSWORD  "batatinha1,2,3" // Senha da rede  wi-fi
#define SSID_BASE "SSID_da_base" // ID da base de carregamento
#define SSID_MESA1 "SSID_desta_mesa" // ID da base de carregamento
#define SERVER_HOST "192.168.1.103" // IP local do servidor (central) websocket
#define SERVER_PORT 8091 // Porta de conexão do servidor (deve ser a mesma do back-end!)
#define LOCALIZACAO_CENTRAL {-4,-1} //Coordenadas (x,y) do roteador da rede em metros
#define LOCALIZACAO_BASE  {0.0,0.0} //Coordenadas (x,y) da base de carregamento em metros
#define LOCALIZACAO_MESA1 {-1,-3.5} //Coordenadas (x,y) da mesa 01 em metros
//Pronto! Os proximos campos nao precisam ser modificados

//ATENÇÃO! O pino para bms, além de ser conectado na saída de alarme deste, deve conectado com um
#define pinoBMS 18 // Pino utilizado para sinal de 'alarme' do BMS (parte elétrica)
#define media_movel 10 //Largura do filtro de media móvel

#include <ArduinoWebsockets.h>

typedef struct{
```

[1] Disponível em: <https://www.arduino.cc/en/software>

```
25    float x = 0;
26    float y = 0;
27  } coordenada;
28
29  typedef struct {
30    char ssid[];
31    int network=0;
32    int32_t rssi=0;
33    float distancia;
34    coordenada localizacao;
35  } Rede;
36
37  Rede central,base,mesa1;
38
39  // Utilizamos o namespace de websocket para podermos utilizar a classe WebsocketsClient
40  using namespace websockets;
41  WebsocketsClient client;// Objeto websocket client
42
43  //Fucoes utilizadas
44  void atribuiCoordenada(coordenada* localizacao, float vetor[]);
45  String destino; //SSID para onde ir
46  int32_t getNetworkNumber(const String target_ssid);//Obtem o numero da rede wi-fi do SSID associad
47  int32_t getRSSI(const int network);//Obtem o valor de potencia do sinal (RSSI) da rede wi-fi do nu
48  void callBack(void);//Rotina para funcionamento do robô como cliente
49  float calculaDistancia(int32_t RSSI);//Algoritmo para calcular a distancia de um dispositivo emiss
50  coordenada trilatera(Rede* central,Rede* base,Rede* mesa);
51  //float trilatera(float d1,d2,d3){}
52
53  void setup(){
54    float loc_central[] = LOCALIZACAO_CENTRAL, loc_base[] = LOCALIZACAO_BASE, loc_mesa1[] = LOCALIZA
55
56    strcpy(central.ssid, SSID_CENTRAL);
57    strcpy(base.ssid, SSID_BASE);
58    strcpy(mesa1.ssid, SSID_MESA1);
59    atribuiCoordenada(&central.localizacao, loc_central);
60    atribuiCoordenada(&mesa1.localizacao, loc_mesa1);
61    atribuiCoordenada(&base.localizacao, loc_base);
62
63    // Inicia a comunicacao UART e wi-fi
64    Serial.begin(115200);
65    WiFi.begin();
66
67    // Conectamos o wifi com o roteador
68    WiFi.begin(SSID_CENTRAL, PASSWORD);
69    while(WiFi.status() != WL_CONNECTED){
70        delay(1000);
71    }
72
73    // Tentamos conectar com o websockets server
```

```
74    bool connected = client.connect(SERVER_HOST, SERVER_PORT, "/");
75    // Se foi possível conectar
76    if(connected){
77        // Enviamos uma msg para o servidor
78        client.send("Robo garçom a servico");
79    }   // Se não foi possível conectar
80    else{
81        return;
82    }
83    //Obtem numero de rede desses dispositivos
84    central.network = getNetworkNumber(SSID_CENTRAL);
85    base.network = getNetworkNumber(SSID_BASE);
86    //Faz a leitura do pino_alarme do BMS e envia mensagem p/ servidor central
87    pinMode(pinoBMS,OUTPUT);
88    if(!digitalRead(pinoBMS)){
89      client.send("Erro na bateria do robo!");
90      delay(1000);
91    }
92    //Interrupção para chamar setup() e ler o pino do BMS
93    attachInterrupt(pinoBMS, setup, LOW);
94    //Chama rotina de cliente para realizar tarefas rotineiras do robo
95    callBack();
96  }
97
98  void loop(){
99    //  De tempo em tempo, o websocket client checa por novas mensagens recebidas e repetir rotina c
100   if(client.available())
101     client.poll();
102   delay(500);
103 }
104
105 //Corpo das funcoes
106 void atribuiCoordenada(coordenada* localizacao, float vetor[]){
107   localizacao->x = vetor[0];
108   localizacao->y = vetor[1];
109   return;
110 }
111 int32_t getNetworkNumber(const String target_ssid){
112   byte available_networks = WiFi.scanNetworks(); //escaneia o numero de redes
113   int32_t rssi = 0;
114
115   for (int32_t network = 0; network < available_networks; network++) {
116     if (WiFi.SSID(network).compareTo(target_ssid) == 0) {return network;} //compara se alguma das
117   }
118   return 0;
119 }
120 //
121 int32_t getRSSI(const int network) {
122   int acum=0,rssi;
```

```
123   for (int sample=0; sample<media_movel; sample++){
124     if(rssi = WiFi.RSSI(network)){acum += rssi;}
125   }
126   return  acum/media_movel; ///retorna o ssid da rede
127 }
128 void callBack(void){ //char* ssID_mesa){
129   // Iniciamos o callback onde as mesagens serão recebidas
130   client.onMessage([&](WebsocketsMessage message){
131   if(message.data().equalsIgnoreCase("Pare")){
132     Serial.print("Pare");
133   }
134   else if(message.data().equalsIgnoreCase("base")){
135     Rede outra;
136     int32_t rssi_outra;
137 //    int network_outra=0;
138
139     client.send("Voltando a base");
140     destino = String(message.data());
141     client.send("Indo a");
142     client.send(destino);
143     delay(500);
144     //Rotina de calculos até chegar ao destino
145     do{
146       central.rssi = getRSSI(central.network);
147       base.rssi = getRSSI(base.network);
148       outra.rssi = getRSSI(outra.network);
149       //Calculo da distancia dos dispositivos
150       base.distancia = calculaDistancia(base.rssi);
151       central.distancia = calculaDistancia(central.rssi);
152       outra.distancia = calculaDistancia(outra.rssi);
153       //Trilatarecao para descobrir a coordenada do robo
154       coordenada posicao = trilatera(&central,&base,&outra);
155 //    FuncaoMedeDirecao(); (FALTA)
156 //    FuncaoComandoDirecaoControlador(); (FALTA)
157       delay(500);
158     }while(outra.distancia != 0); //Enquanto a distancia do alvo existir a rotina é repetida
159   }
160   else{
161     destino = String(message.data());
162     mesa1.network = getNetworkNumber(destino);
163     if(!mesa1.network){
164      client.send("Robo diz: Nao sei para onde ir");
165       delay(500);
166       return;
167     }
168     client.send("Indo a");
169     client.send(destino);
170     delay(500);
171     //Rotina de calculos até chegar no destino
```

```
172     do{
173       central.rssi = getRSSI(central.network);
174       base.rssi = getRSSI(base.network);
175       mesa1.rssi = getRSSI(mesa1.network);
176       //Calculo da distancia dos dispositivos
177       base.distancia = calculaDistancia(base.rssi);
178       central.distancia = calculaDistancia(central.rssi);
179       mesa1.distancia = calculaDistancia(mesa1.rssi);
180       //Trilatarecao para descobrir a coordenada do robo
181       coordenada posicao = trilatera(&central,&base,&mesa1);
182 //    FuncaoMedeDirecao(); (FALTA)
183 //    FuncaoComandoDirecaoControlador(); (FALTA)
184       delay(500);
185     }while(mesa1.distancia != 0); //Enquanto a distancia do alvo existir a rotina é repetida
186   }
187   });
188   return;
189 }
190 //
191 float calculaDistancia(int32_t RSSI){
192   float distancia;
193   const int32_t Pr = RSSI, P0 = -40; //dBm
194   const int n = 12;
195   //Essa formula eh abordada no relatorio
196   distancia = 10^(-(P0+Pr)/(n));
197
198   return distancia;
199 }
200 //
201 coordenada trilatera(Rede* central,Rede* base,Rede* mesa){
202   float d1,x1,y1,d2,x2,y2,d3,x3,y3, xp=0, yp=0;
203   coordenada p;
204
205   x1 = central->localizacao.x; y1 = central->localizacao.y;
206   d1 = central->distancia;
207   x2 = base->localizacao.x; y2 = base->localizacao.y;
208   d2 = base->distancia;
209   x3 = mesa->localizacao.x; y3 = mesa->localizacao.y;
210   d3 = mesa->distancia;
211   //Essa formula eh abordada no relatorio
212   xp = ((powf(d2,2)-powf(d3,2)+powf(y3,2)-powf(y2,2)+powf(x3,2)-powf(x2,2))*(y2-y1)-(powf(d1,2)-po
213   yp = ((powf(d2,2)-powf(d3,2)+powf(y3,2)-powf(y2,2)+powf(x3,2)-powf(x2,2))*(x2-x1)-(powf(d1,2)-po
214   p.x = xp; p.y = yp;
215   return p;
216 }
```

Código para o comunicador *wireless*.

I.2.2 Códigos para o controlador

Aqui situa-se o código final do controlador, integrando todos os sensores e a comunicação serial entre as duas ESPs.

Código para o controlador

```
#include <HX711.h>
#include <Ultrasonic.h>
#include <Servo.h>


HardwareSerial Receiver(2); // Define a  porta serial do receptor


#define Receiver_Txd_pin 27 // recepção e transmissão serial
#define Receiver_Rxd_pin 14


#define SERVO1 9 // Porta Digital 9 PWM
#define SERVO2 10 // Porta Digital 10 PWM


Servo s1; // Variável Servo
Servo s2; // Variável Servo

int posicao; // Posição Servo


#define INTERVALO_LEITURA 250 //(ms)
// conexão pinos acelerometro
#define pinAcelX A0
#define pinAcelY A1
#define pinAcelZ A2
//conexão dos pinos para o sensor ultrasonico
#define PIN_TRIGGER   4
#define PIN_ECHO      5

// Module connection pins (Digital Pins)
#define CLK 15
#define DIO 13
#define pinDT  2
#define pinSCK  3
#define pesoMin 0.00
#define pesoMax 30.0

//variável responsável pela distância
unsigned int distancia = 0;
```

```
42 // conexão para o sensor infravermelho
43 const int pinoSensor1 = 7; //PINO  UTILIZADO PELO SENSOR INFRAVERMELHO 1
44 const int pinoSensor2 = 8; //PINO  UTILIZADO PELO SENSOR INFRAVERMELHO 2
45 //Inicializa o sensor nos pinos definidos acima
46 Ultrasonic ultrasonic(PIN_TRIGGER, PIN_ECHO);
47
48 #define escala 0.0f
49 // INSTANCIANDO OBJETOS
50 HX711 scale;
51
52 float medida=0;
53 char cmd1 = frente;
54 char cmd2 = atras;
55 char cmd3 = direita;
56 char cmd4 = esquerda;
57 char cmd5 = pare;
58 char cmd6 = chegou;
59 char cmd7 = base;
60
61 void setup() {
62
63
64
65 Serial.begin(115200);
66     configurarPinos();
67         scale.begin(pinDT, pinSCK); // CONFIGURANDO OS PINOS DA BALANÇA
68     scale.set_scale(escala); // ENVIANDO O VALOR DA ESCALA CALIBRADO
69     delay(2000);
70     scale.tare(); // ZERANDO A BALANÇA PARA DESCONSIDERAR A MASSA DA ESTRUTURA
71         Receiver.begin(115200, SERIAL_8N1, Receiver_Txd_pin, Receiver_Rxd_pin);
72         s1.attach(SERVO1);
73         s2.attach(SERVO2);
74         s1.write(0); // Inicia motor da tampa na  posição zero
75         s2.write(0); // Inicia motor da tampa na  posição zero
76
77
78 //cria uma tarefa que será executada na função coreTaskZero,
79   xTaskCreatePinnedToCore(
80                     coreTaskZero,   /* função que implementa a tarefa */
81                     "coreTaskZero", /* nome da tarefa */
82                     10000,      /* número de palavras a serem alocadas para uso com a pilha da tar
83                     NULL,       /* parâmetro de entrada para a tarefa (pode ser NULL) */
84                     2,          /* prioridade da tarefa (0 a N) */
85                     NULL,       /* referência para a tarefa (pode ser NULL) */
86                     taskCoreZero);         /* Núcleo que executará a tarefa */
87
88   delay(500); //tempo para a tarefa iniciar
89
90
```

```
91  //cria uma tarefa que será executada na função coreTaskOne
92    xTaskCreatePinnedToCore(
93                        coreTaskOne,   /* função que implementa a tarefa */
94                        "coreTaskOne", /* nome da tarefa */
95                        10000,      /* número de palavras a serem alocadas para uso com a pilha da tar
96                        NULL,       /* parâmetro de entrada para a tarefa (pode ser NULL) */
97                        2,          /* prioridade da tarefa (0 a N) */
98                        NULL,       /* referência para a tarefa (pode ser NULL) */
99                        taskCoreOne);         /* Núcleo que executará a tarefa */
100
101     delay(500); //tempo para a tarefa iniciar
102
103
104 }
105
106
107
108
109
110 void coreTaskZero( void * pvParameters ){
111 scale.power_up();//inicializa balança
112 while(true){
113 int valX = analogRead(pinAcelX);
114 int valY = analogRead(pinAcelY);
115 int valZ = analogRead(pinAcelZ);
116
117
118         while (Receiver.available()){
119
120         char rxdchar= receive.read();
121
122         if(rxdchar == cmd1||rxdchar == cmd2||rxdchar == cmd3||rxdchar == cmd4){
123                 platectrl();
124                 fechartampa();
125                 movectrl();
126                 obstaculo();
127                 delay(INTERVALO_LEITURA);
128                 acelerometro();
129         }
130         else if (rxdchar == cmd5){// parar
131
132                 movectrl();
133         }
134         else if(rxdchar == cmd6){ // chegou
135                 if(destino == base){// chegou na base
136                         abrirtampa();
137                         platectrl();
138                         acelerometro();
139
```

```
140                 }
141                 else{// chegou na mesa
142                         platectrl();
143                         abrirtampa();
144                         balanca();
145
146                         while(medida != 0 ){
147
148                                 balanca();
149
150                         }
151                         delay(3000);
152                         fechartampa();
153                         platectrl();
154
155
156                 }
157
158     }
159         else {// ir para a base
160                 movectrl();
161                 platectrl();
162                 fechartampa();
163
164
165         }
166
167 }
168 }
169 }
170
171 void coreTaskOne( void * pvParameters ){
172
173 while(true){
174 obstaculo(){ // pinoSensor1 = infrared esquerda , pinoSensor2 = infrared direita
175         if (digitalRead(pinoSensor1) == LOW || digitalRead(pinoSensor2) == LOW){
176                 getdistance();
177                 if (distancia < 60 && digitalRead(pinoSensor1) == LOW  && digitalRead(pinoSensor2)
178                         percentual = 0;
179                         pwmmov = (percentual/100)*255;
180                         analogWrite(motor1a, 0);
181                         analogWrite(motor1b, 0);
182                         analogWrite(motor2a, 0);
183                         analogWrite(motor2b, 0);
184                 }
185                 else if(distancia <60 && digitalRead(pinoSensor1) == LOW){ // obstaculo mais a  es
186                         percentual = 40;
187                         pwmmov = (percentual/100)*255;
188                         analogWrite(motor1a, pwmmov);// vira a direita
```

```
189                         analogWrite(motor1b, 0);
190                         analogWrite(motor2a, pwmmov);
191                         analogWrite(motor2b, 0);
192                         delay(100);
193                         analogWrite(motor1a, pwmmov);//segue em frente
194                         analogWrite(motor1b, 0);
195                         analogWrite(motor2a, 0);
196                         analogWrite(motor2b, pwmmov);
197                         delay(100);
198                         analogWrite(motor1a, 0); // vira a esquerda
199                         analogWrite(motor1b, pwmmov);
200                         analogWrite(motor2a, 0);
201                         analogWrite(motor2b, pwmmov);
202                         delay(100);
203                         analogWrite(motor1a, pwmmov);//segue em frente
204                         analogWrite(motor1b, 0);
205                         analogWrite(motor2a, 0);
206                         analogWrite(motor2b, pwmmov);
207
208                 }
209                 else  (distancia <60 && digitalRead(pinoSensor2) == LOW){ // obstaculo mais a dire
210                         percentual = 40;
211                         pwmmov = (percentual/100)*255;// vira a esquerda
212                         analogWrite(motor1a, 0);
213                         analogWrite(motor1b, pwmmov);
214                         analogWrite(motor2a, 0);
215                         analogWrite(motor2b, pwmmov);
216                         delay(100);
217                         analogWrite(motor1a, pwmmov);//segue em frente
218                         analogWrite(motor1b, 0);
219                         analogWrite(motor2a, 0);
220                         analogWrite(motor2b, pwmmov);
221                         delay(100);
222                         analogWrite(motor1a, pwmmov);// vira a direita
223                         analogWrite(motor1b, 0);
224                         analogWrite(motor2a, pwmmov);
225                         analogWrite(motor2b, 0);
226                         delay(100);
227                         analogWrite(motor1a, pwmmov);//segue em frente
228                         analogWrite(motor1b, 0);
229                         analogWrite(motor2a, 0);
230                         analogWrite(motor2b, pwmmov);
231
232                 }
233         }
234         else{
235                 delay(10);
236
237         }
```

```
238
239
240 }
241 void acelerometro(){
242
243 if (valZ < 300) {
244      Receiver.print("Capotado\n");
245   } else if ( (valZ < 360) && (valY < 270)) {
246      Receiver.print("Caiu para Esquerda\n");
247   } else if ( (valZ < 360) && (valY > 380)) {
248      Receiver.print("Caiu para Direita\n");
249   } else  if ( (valZ < 360) && (valX < 270)) {
250      Receiver.print("Caiu para Frente\n");
251   } else if ( (valZ < 360) && (valX > 380)) {
252      Receiver.print("Caiu para Tras\n");
253   } else {
254      Receiver.print("Normal\n");
255   }
256
257
258 }
259  //pega distancia do obstaculo
260
261 int getDistance()
262 {
263       //faz a leitura do sensor (em cm)
264       int distanciaCM;
265       long microsec = ultrasonic.timing();
266       distanciaCM = ultrasonic.convert(microsec, Ultrasonic::CM);
267       return distanciaCM;
268 }
269
270
271
272 }
273 }
274
275 // movimentação do robô -- rodas
276
277 void movectrl(rxdchar){
278   switch(rxdchar):{
279     case cmd1: //frente
280     percentual = 70;
281     pwmmov = (percentual/100)*255;
282     analogWrite(motor1a, pwmmov);
283     analogWrite(motor1b, 0);
284     analogWrite(motor2a, 0);
285     analogWrite(motor2b, pwmmov);
286     break;
```

```
287      case cmd3: //virar a direita
288      percentual = 70;
289      pwmmov = (percentual/100)*255;
290      analogWrite(motor1a, pwmmov);
291      analogWrite(motor1b, 0);
292      analogWrite(motor2a, pwmmov);
293      analogWrite(motor2b, 0);
294      break;
295      case cmd4: //virar a esquerda
296      percentual = 70;
297      pwmmov = (percentual/100)*255;
298      analogWrite(motor1a, 0);
299      analogWrite(motor1b, pwmmov);
300      analogWrite(motor2a, 0);
301      analogWrite(motor2b, pwmmov);
302      break;
303      case cmd2: //para trás
304      percentual = 70;
305      pwmmov = (percentual/100)*255;
306      analogWrite(motor1a, 0);
307      analogWrite(motor1b, pwmmov);
308      analogWrite(motor2a, pwmmov);
309      analogWrite(motor2b, 0);
310      break;
311      default: //parado
312      percentual = 0;
313      pwmmov = (percentual/100)*255;
314      analogWrite(motor1a, 0);
315      analogWrite(motor1b, 0);
316      analogWrite(motor2a, 0);
317      analogWrite(motor2b, 0);
318      break;
319    }
320  }
321
322
323
324  void abrirtampa()
325  {
326    for(posicao = 0; posicao < 180; posicao++)
327    {
328      s1.write(posicao);
329          s2.write(posicao);
330    delay(15);
331    }
332
333  }
334
335  void fechartampa()
```

```
336 {
337   for(posicao = 180; posicao >= 0; posicao--)
338   {
339     s1.write(posicao);
340         s2.write(posicao);
341   delay(15);
342   }
343
344 }
345
346 //configura pinos infrared
347 void configurarPinos()
348 {
349 pinMode(pinoSensor1, INPUT);
350 pinMode(pinoSensor2, INPUT);
351  }
352
353  // pega valor medido pela balança
354 void balanca()
355 {
356 medida = scale.get_units(5); // SALVANDO NA VARIAVEL O VALOR DA MÉDIA DE 5 MEDIDAS
357  }
358
359
360
361 // movimentação bandeja
362 void platectrl(int dir2){
363   switch(dir2):{
364     case 1: //subindo
365     analogWrite(motor3a, 1);
366     analogWrite(motor3b, 0);
367     break;
368     case 2: //descendo
369     analogWrite(motor3a, 0);
370     analogWrite(motor3b, 1);
371     break;
372     default: //parado
373     analogWrite(motor3a, 0);
374     analogWrite(motor3b, 0);
375     break;
376   }
377 }
378
379
380
381 void loop() {}
```

Códigos para as simulações

Neste local é exposto algumas funções que foram usadas para simulação do controlador. Alguns foram simulados já em implementação no *software* Proteus, outros na plataforma Tinkercad[2].

Funções para motores DC

```
// movimentação do robô

void movectrl(int dir1){
  switch(dir):{
    case 1: //frente
    percentual = 70;
    pwmmov = (percentual/100)*255;
    analogWrite(motor1a, pwmmov);
    analogWrite(motor1b, 0);
    analogWrite(motor2a, 0);
    analogWrite(motor2b, pwmmov);
    break;
    case 2: //virar a direita
    percentual = 70;
    pwmmov = (percentual/100)*255;
    analogWrite(motor1a, pwmmov);
    analogWrite(motor1b, 0);
    analogWrite(motor2a, pwmmov);
    analogWrite(motor2b, 0);
    break;
    case 3: //virar a direita
    percentual = 70;
    pwmmov = (percentual/100)*255;
    analogWrite(motor1a, 0);
    analogWrite(motor1b, pwmmov);
    analogWrite(motor2a, 0);
    analogWrite(motor2b, pwmmov);
    break;
    case 4: //para trás
    percentual = 70;
    pwmmov = (percentual/100)*255;
    analogWrite(motor1a, 0);
    analogWrite(motor1b, pwmmov);
    analogWrite(motor2a, pwmmov);
    analogWrite(motor2b, 0);
    break;
    default: //parado
    percentual = 0;
```

[2] Disponível em: <https://www.tinkercad.com/>

```
39      pwmmov = (percentual/100)*255;
40      analogWrite(motor1a, 0);
41      analogWrite(motor1b, 0);
42      analogWrite(motor2a, 0);
43      analogWrite(motor2b, 0);
44      break;
45    }
46  }
47
48  // movimentação bandeja
49  void platectrl(int dir2){
50    switch(dir2):{
51      case 1: //subindo
52      analogWrite(motor3a, 1);
53      analogWrite(motor3b, 0);
54      break;
55      case 2: //descendo
56      analogWrite(motor3a, 0);
57      analogWrite(motor3b, 1);
58      break;
59      default: //parado
60      analogWrite(motor3a, 0);
61      analogWrite(motor3b, 0);
62      break;
63    }
64  }
```

Código para teste do acionamento dos motores DC.

Teste para detecção de queda

O seguinte código foi usado para teste do acelerômetro GY-61.

```
1   #define pinAcelX A0
2   #define pinAcelY A1
3   #define pinAcelZ A2
4
5   void setup() {
6
7             Serial.begin(9600);
8   }
9
10  void loop() {
11
12    int valX = analogRead(pinAcelX);
13    int valY = analogRead(pinAcelY);
14    int valZ = analogRead(pinAcelZ);
15
16    /*
```

```
17    Serial.print("X: ");
18    Serial.print(valX);
19    Serial.print("  Y: ");
20    Serial.print(valY);
21    Serial.print("  Z: \n");
22    Serial.println(valZ);
23    */
24
25    if (valZ < 300) {
26       Serial.println("Capotado\n");
27    } else if ( (valZ < 360) && (valY < 270)) {
28       Serial.println("Caiu para Esquerda\n");
29    } else if ( (valZ < 360) && (valY > 380)) {
30       Serial.println("Caiu para Direita\n");
31    } else  if ( (valZ < 360) && (valX < 270)) {
32       Serial.println("Caiu para Frente\n");
33    } else if ( (valZ < 360) && (valX > 380)) {
34       Serial.println("Caiu para Tras\n");
35    } else {
36       Serial.println("Normal\n");
37    }
38 }
```

Código para testar detecção de queda.

Teste do sensor de peso

Para teste do sensor de peso a partir do peso lido, foi usado o seguinte código:

```
1  // INCLUSÃO DE BIBLIOTECAS
2  #include <HX711.h>
3  #include <LiquidCrystal.h>
4  LiquidCrystal lcd(7, 6, 5, 4, 3, 2);
5
6  // DEFINIÇÕES DE PINOS
7  #define pinDT  2
8  #define pinSCK  3
9
10 // DEFINIÇÕES
11 #define pesoMin 0.00
12 #define pesoMax 30.0
13
14 #define escala 0.0f
15
16 // INSTANCIANDO OBJETOS
17 HX711 scale;
18
19 // DECLARAÇÃO DE VARIÁVEIS
20 float medida=0;
```

```
21
22 void setup() {
23          Serial.begin(9600);
24          scale.begin(pinDT, pinSCK); // CONFIGURANDO OS PINOS DA BALANÇA
25          scale.set_scale(escala); // ENVIANDO O VALOR DA ESCALA CALIBRADO
26          delay(2000);
27          scale.tare(); // ZERANDO A BALANÇA PARA DESCONSIDERAR A MASSA DA ESTRUTURA
28 }
29
30 void loop() {
31
32
33                  medida = scale.get_units(5); // SALVANDO NA VARIAVEL O VALOR DA MÉDIA DE 5 MEI
34            lcd.print("Measured Weight");
35            lcd.setCursor(0,1);
36            lcd.print(medida);
37  }
```

Código para o sensor de peso.

Teste para detecção de distância via sensor ultrassônico

Para teste do sensor ultrassônico juntamente com uma lógica inicial para tomada de decisões acerca da distância lida, foi usado o seguinte código:

```
1  // codigo do sensor ultrassonico
2  #include <Ultrasonic.h>
3
4  #define INTERVALO_LEITURA 250 //(ms)
5
6  //conexão dos pinos para o sensor ultrasonico
7  #define PIN_TRIGGER   4
8  #define PIN_ECHO      5
9
10 // Module connection pins (Digital Pins)
11 #define CLK 14
12 #define DIO 13
13
14 //variável responsável pela distância
15 unsigned int distancia = 0;
16 // conexão para o sensor infravermelho
17 const int pinoSensor1 = 7; //PINO  UTILIZADO PELO SENSOR INFRAVERMELHO 1
18 const int pinoSensor2 = 8; //PINO  UTILIZADO PELO SENSOR INFRAVERMELHO 2
19
20 //Inicializa o sensor nos pinos definidos acima
21 Ultrasonic ultrasonic(PIN_TRIGGER, PIN_ECHO);
22
23 void setup()
```

```
24 {
25          Serial.begin(9600);
26          configurarPinos();
27 }
28
29 void loop()
30 {
31          verificarDistancia();
32          delay(INTERVALO_LEITURA);
33 }
34
35 void configurarPinos()
36 {
37 pinMode(pinoSensor1, INPUT);
38 pinMode(pinoSensor2, INPUT);
39  }
40
41 /*
42 verificar a distância entre o objeto e o robô se a distância  for alta continua movimentando,
43 se a distância for média diminui o pwm dos motores e se a distancia for baixa para na mesma hora
44   */
45 void verificarDistancia()
46 {
47            //pega  a distância atual lida pelo sensor
48           distancia = getDistance();// distancia em cm
49
50         //caso a distância lida seja menor ou igual a 15 considerada uma distância de perigo
51          // se a distância for de 15 cm parar os motores
52          if( digitalRead(pinoSensor) == LOW && distancia <= 15 )
53   {
54            //Colocar código do motor aqui
55       }
56          //caso a distância seja maior que 15 e menor ou igual a 60,
57          //uma distância de atenção diminuir o pwm dos motores
58          else if(distancia <=60)
59   {
60          // Colocar código do motor aqui
61
62   }
63
64          //para distâncias maiores que 60 ,uma distância sem perigo pwm dos motores continua alto
65
66   else
67   {
68           // Colocar código do motor aqui
69   }
70 }
71
72 //  Função para a leitura da distância medida pelo sensor ultrassônico
```

```
int getDistance()
{
            //faz a leitura do sensor (em cm)
            int distanciaCM;
            long microsec = ultrasonic.timing();
            distanciaCM = ultrasonic.convert(microsec, Ultrasonic::CM);
            return distanciaCM;
}
```

Código para teste do ultrassom.

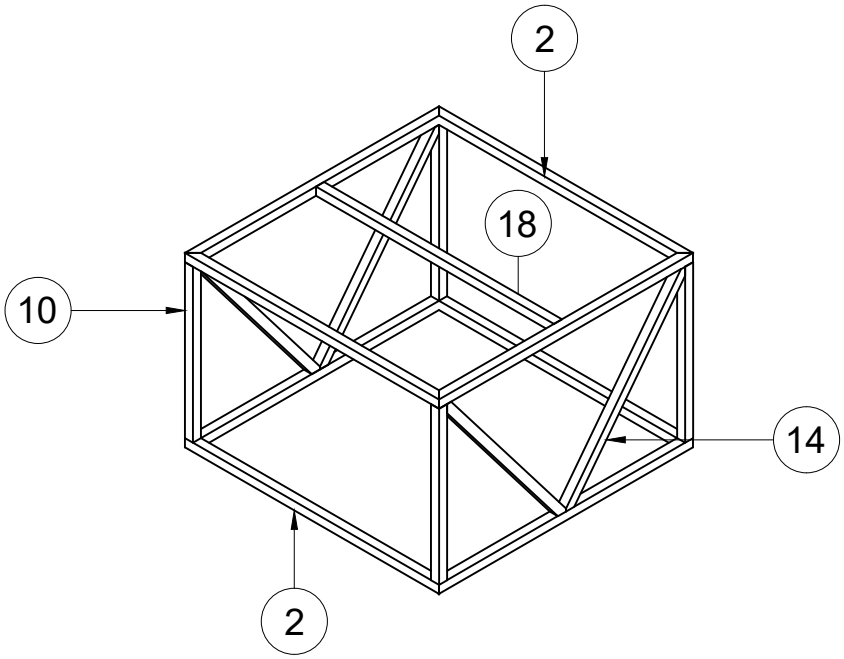
I.2.2.1 Código para calibração da balança

O código abaixo deve ser utilizado para a calibração da balança.

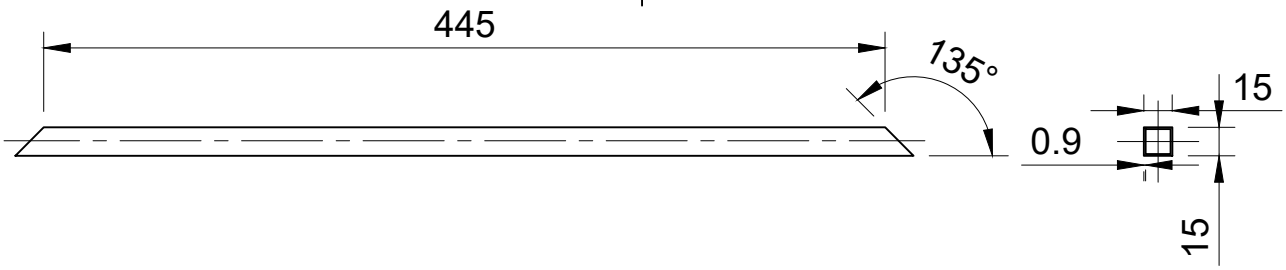
```
#include <HX711.h>


#define pinDT  2
#define pinSCK  3
#define pinBotao 4


HX711 scale;

float medida = 0;

void setup() {
  Serial.begin(57600);

  scale.begin(pinDT, pinSCK);
  scale.set_scale(); // valor da escala

  delay(2000);
  scale.tare(); // zera a balança

  Serial.println("Balança Zerada");
}

void loop() {

  medida = scale.get_units(5); // pega as 5 últimas medidas
  Serial.println(medida, 3); // envia pelo monitor serial o valor lido

  delay(1000);

}
```

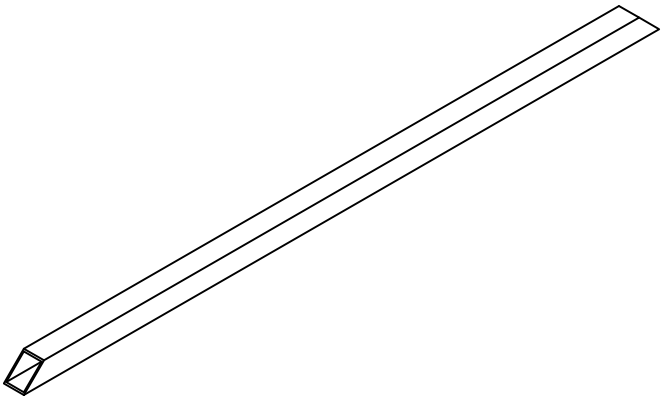
APÊNDICE J – Desenhos técnicos da estrutura

Lista de Componentes			
Num.	Qtd.	Nome	Especificação
2	8	Tipo 1	Tubo metalon quadrado 15mm x 0,9mm de aço 1006 (tubo 1 de 4)
10	4	Tipo 3	Tubo metalon quadrado 15mm x 0,9mm de aço 1006 (tubo 3 de 4)
14	4	Tipo 4	Tubo metalon quadrado 15mm x 0,9mm de aço 1006 (tubo 4 de 4)
18	1	Tipo 2	Tubo metalon quadrado 15mm x 0,9mm de aço 1006 (tubo 2 de 4)



Grupo	Projeto	Autor				Diedro
4	Robô garçom	Lucas Pereira da Silva 17/9/2021				
		Revisor Lucas Pereira da S. 24/10 2021	Unidade	Escala 1:10	Tamanho A4	Rev. 2
		Título Estrutura Tubular		Data de Emissão 24/10/2021		Página 1

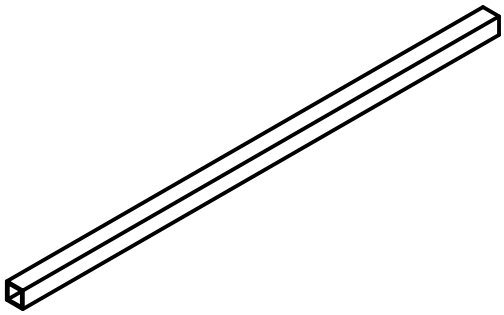


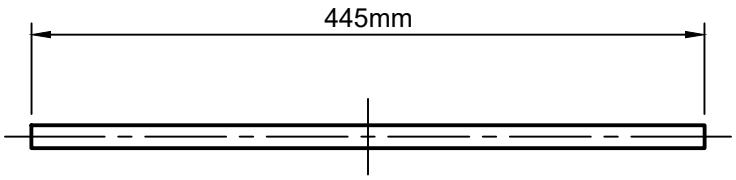


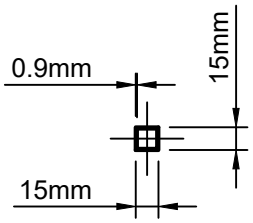
Grupo	Projeto	Autor	Material	Massa	Diedro
04	Robô garçom	Lucas Pereira da Silva 24/10/2021	Aço 1006	177g	

Revisor	Unidade	Escala	Tamanho	Rev.
Isaac Alencar 24/10/2021	mm	1:4	A4	1

Título	Data de Emissão	Página
Tubo 1/4	24/10/2021	2



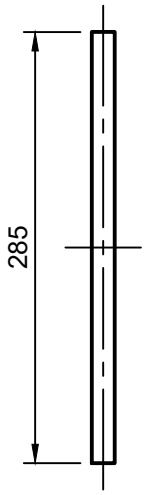


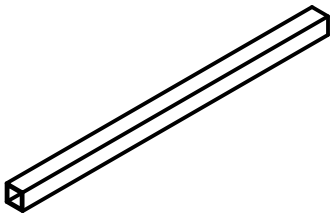


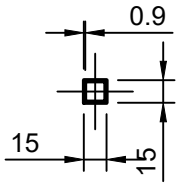
Grupo	Projeto	Autor	Material	Massa	Diedro
04	Robô garçom	Lucas Pereira da Silva 18/9/2021	Aço 1006	177g	

Revisor	Unidade	Escala	Tamanho	Rev.
Isaac Alencar 24/10/2021/	mm	1:5	A4	1

Título	Data de Emissão	Página
Tubo 2/4	24/10/2021	3



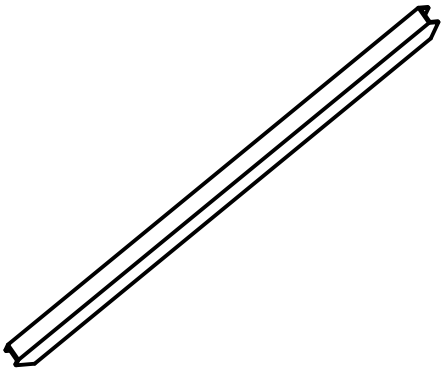


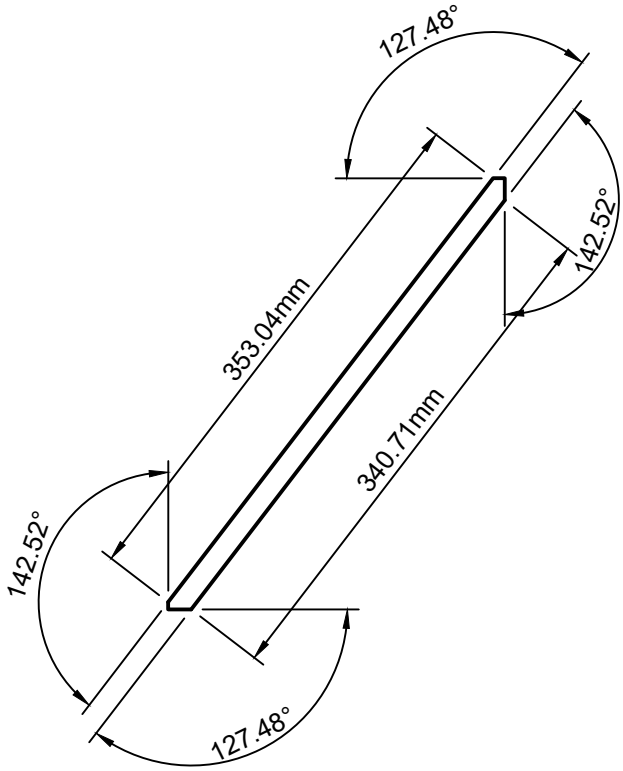


Grupo	Projeto	Autor	Material	Massa	Diedro
04	Robô garçom	Lucas Pereira da Silva 18/9/2021	Aço 1006	177g	

Revisor	Unidade	Escala	Tamanho	Rev.
Isaac Alencar 24/10/2021	mm	1:5	A4	1

Título	Data de Emissão	Página
Estrutura Tubular	24/10/2021	4

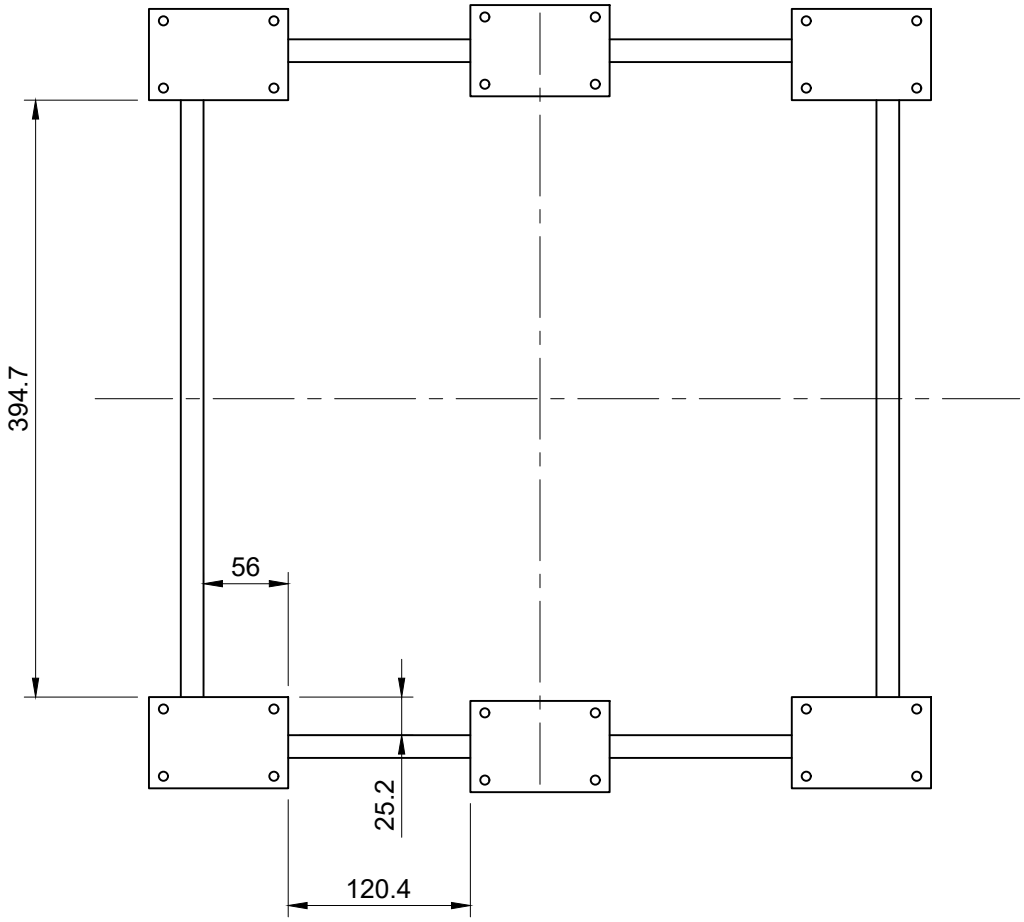




Grupo	Projeto	Autor	Material	Massa	Diedro
04	Robô garçom	Lucas Pereira da Silva 9/18/2021	Aço 1006	177g	

Revisor	Unidade	Escala	Tamanho	Rev.
Isaac Alencar 24/10/2021	mm	1:5	A4	1

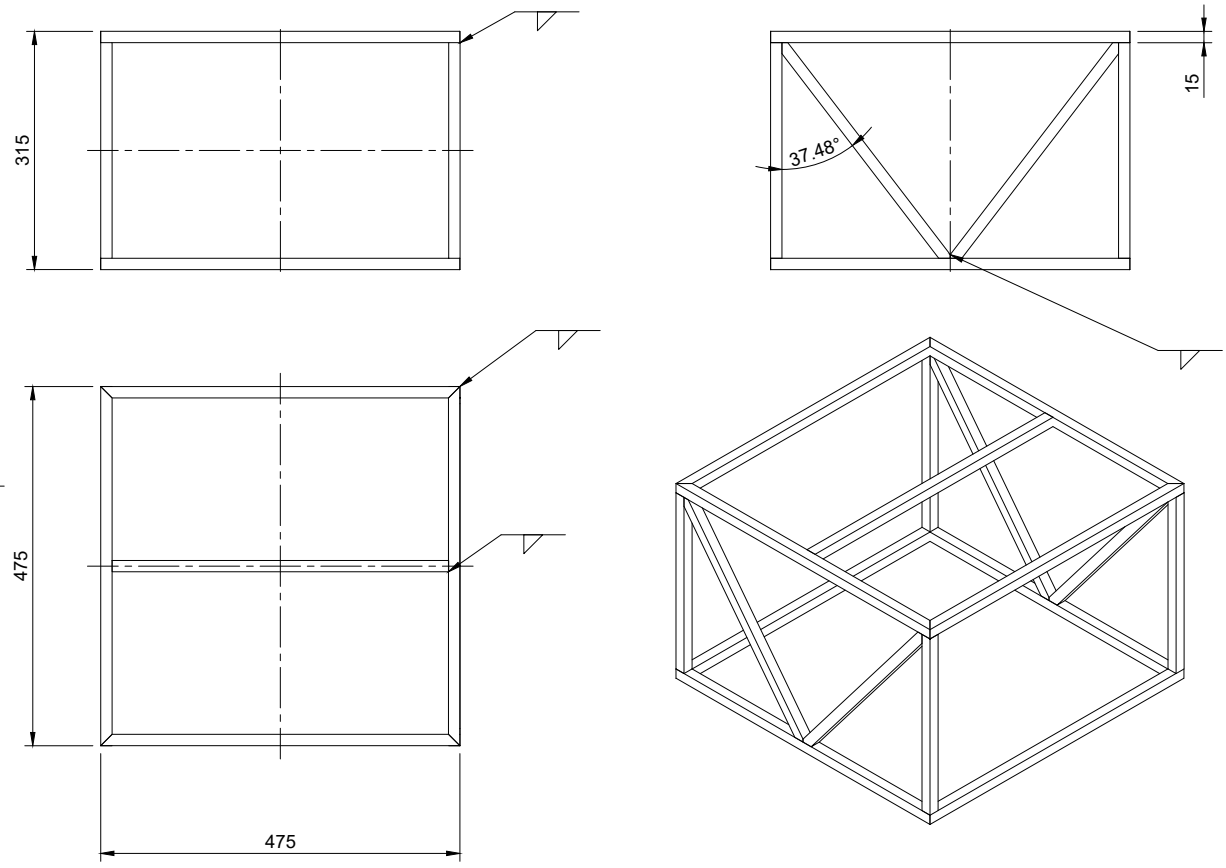
Título	Data de Emissão	Página
Estrutura Tubular	24/10/2021	5



Grupo	Projeto	Autor	Material	Massa	Diedro
04	Robô garçom	Lucas Pereira da Silva 10/24/2021	Aço 1006	5,2 kg	

Revisor	Unidade	Escala	Tamanho	Rev.
Isaac Alencar 24/10/2021	mm	1:5	A4	1

Título	Data de Emissão	Página
Posiçao suportes de rodas	10/24/2021	6

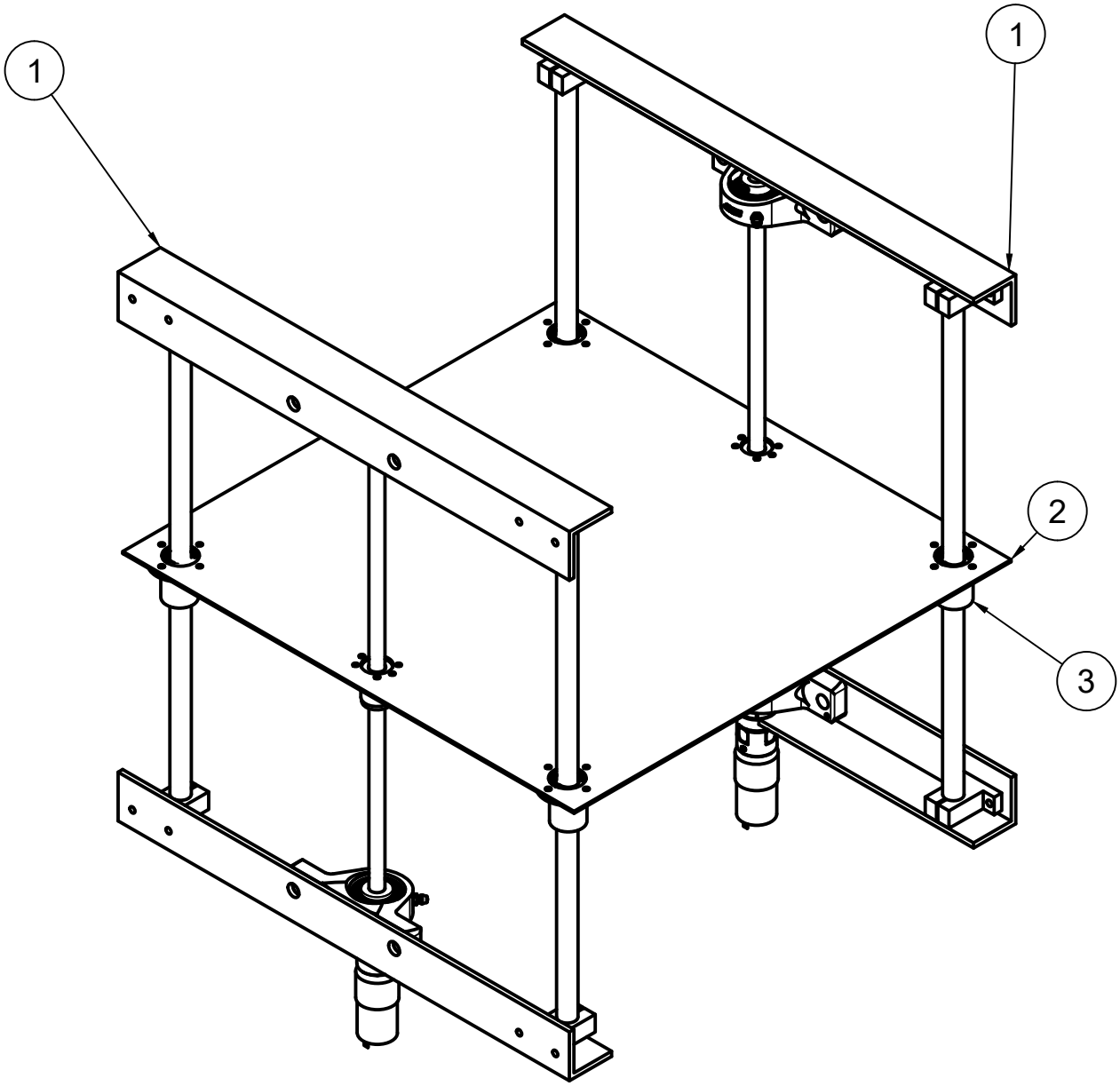


Grupo	Projeto	Autor	Material	Massa	Diedro
04	Robô garçom	Lucas Pereira da Silva 24/10/2021	Aço 1006	5,2kg	

Revisor	Unidade	Escala	Tamanho	Rev.
Isaac Alencar 24/10/2021	mm	1:5	A3	1

Título	Data de Emissão	Página
Estrutura Tubular	24/10/2021	7

Lista de Componentes			
Item	Qtd.	Nome	Descrição
1	1	Suporte da Bandeja	Placa sobre a qual a bandeja é apoiada
2	4	Rolamento linear	Rolamento linear para a guia
4	2	Sistema de fuso	Sistema de apoio do fuso e das guias para movimento linear

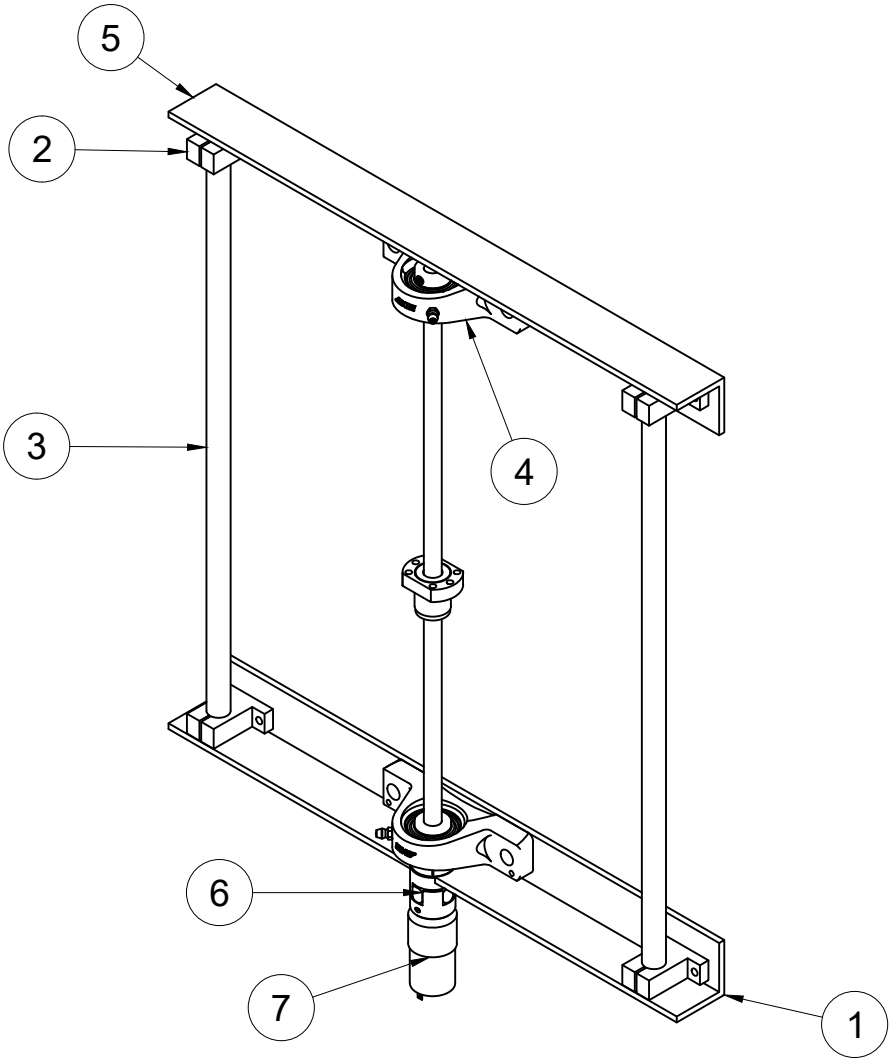


Grupo	Projeto	Autor	Material	Massa	Diedro
04	Robô garçom	Isaac Alencar 9/19/2021		15,8kg	

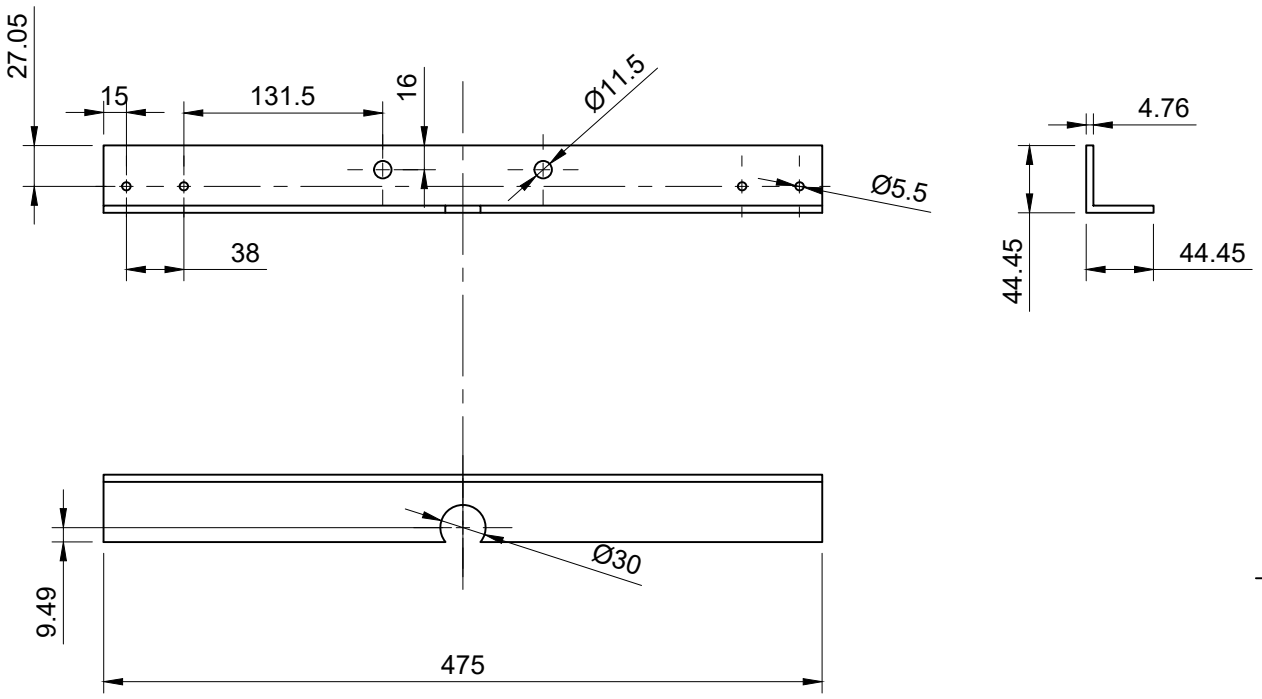
Revisor	Unidade	Escala	Tamanho	Rev.
Isaac Alencar 24/10/2021	mm	1:5	A4	2

Título	Data de Emissão	Página
Sistema de Levantamento	24/10/2021	8

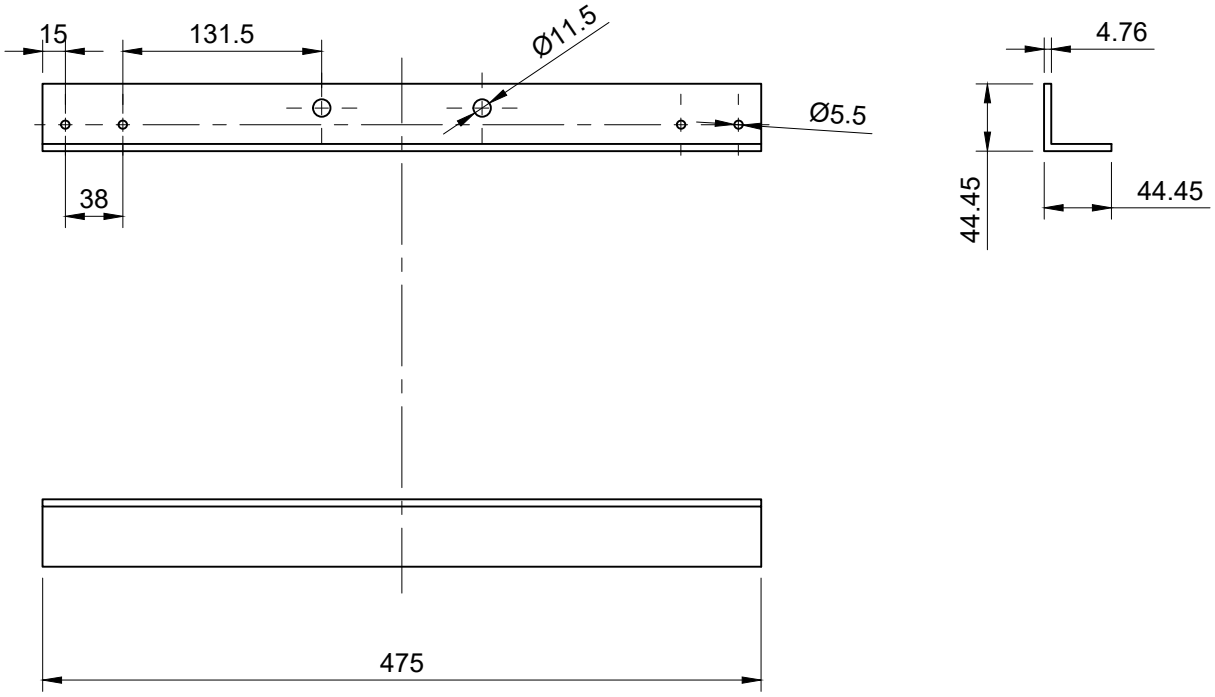
Lista de Componentes			
Item	Qtd.	Nome	Descrição
1	1	Suporte inferior	Cantoneira para fixação das guias e do fuso
2	4	Fixador 16 mm	Fixador das guias lineares
3	2	Eixo retificado 16 mm	Guia linear do sistema
4	2	Mancal 12 mm	Mancal das extremidades do fuso
5	1	Suporte superior	Cantoneira para fixação das guias e do fuso
6	1	Acoplamento	Acoplamento entre motor e fuso
7	1	Motor	Motor que move o fuso
8	1	Fuso 12 mm	Fuso de esferas de 12 mm de diâmetro



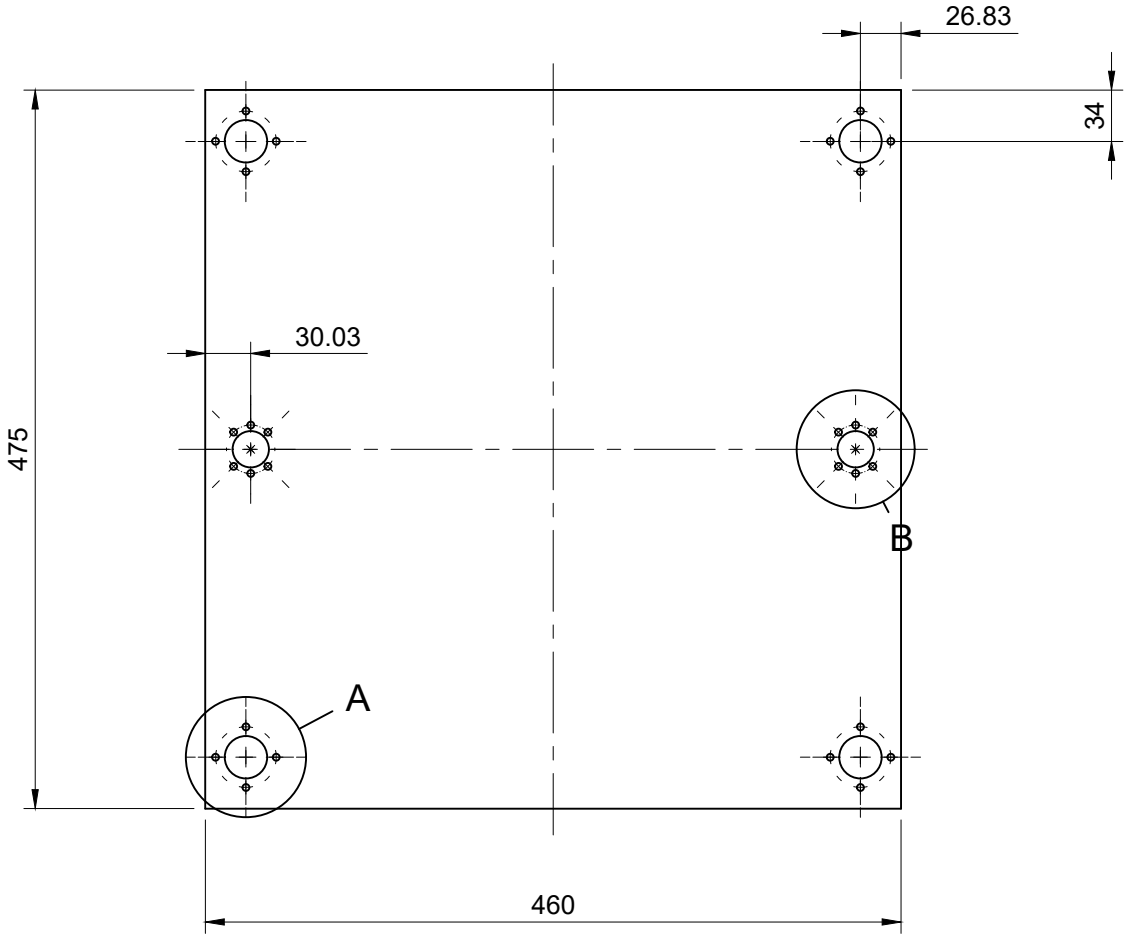
Grupo	Projeto	Autor		Material		Massa	Diedro
04	Robô garçom	Isaac Alencar	9/19/2021	Diversos		7,2kg	
		Revisor		Unidade	Escala	Tamanho	Rev.
		Isaac Alencar	24/10/2021	mm	1:5	A4	2
		Título			Data de Emissão		Página
		Sistema de Fuso			24/10/2021		



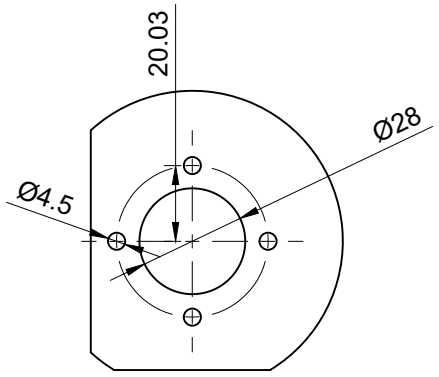
Grupo	Projeto	Autor		Material		Massa	Diedro
04	Robô garçom	Isaac Alencar	9/19/2021	Aço 1006		1,4kg	
		Revisor		Unidade	Escala	Tamanho	Rev.
		Isaac Alencar	24/10/2021	mm	1:5	A4	2
		Título			Data de Emissão		Página
		Suporte inferior			24/10/2021		10



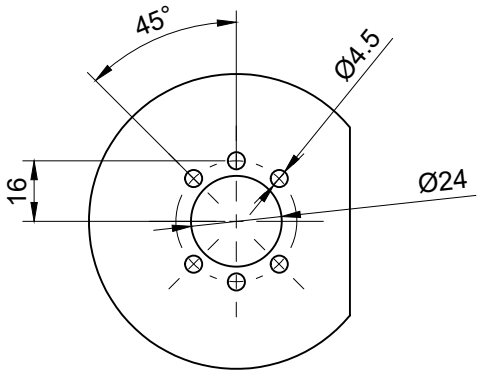
Grupo	Projeto	Autor		Material		Massa	Diedro
4	Robô garçom	Isaac Alencar	9/19/2021	Aço 1006		1,4kg	
		Revisor		Unidade	Escala	Tamanho	Rev.
		Isaac Alencar	24/10/2021	mm	1:5	A4	2
		Título			Data de Emissão		Página
		Suporte superior			24/10/2021		11



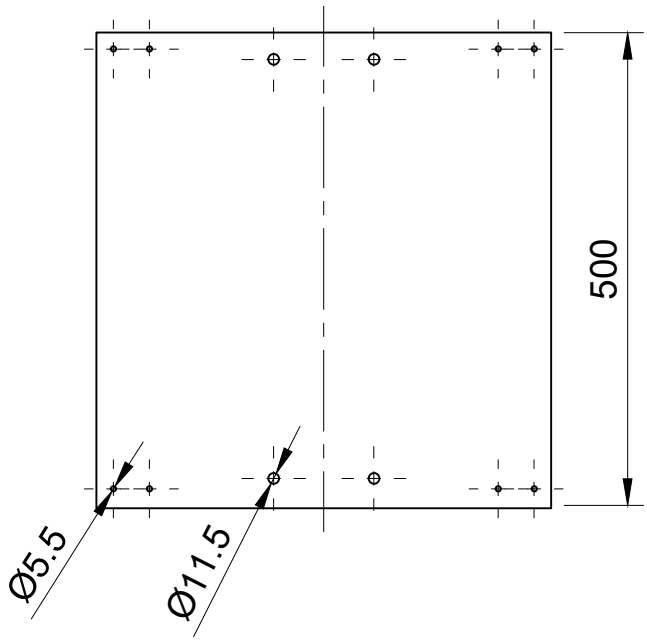
A (1:2)

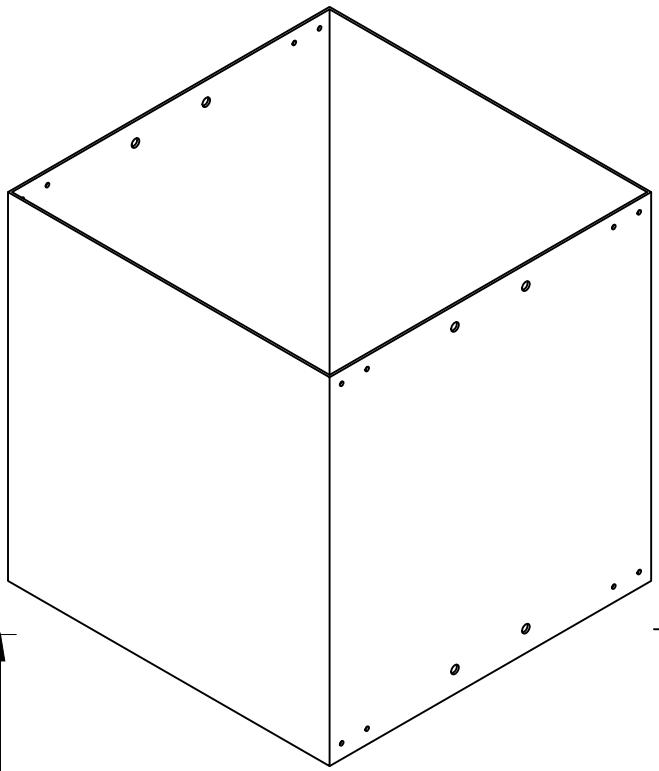


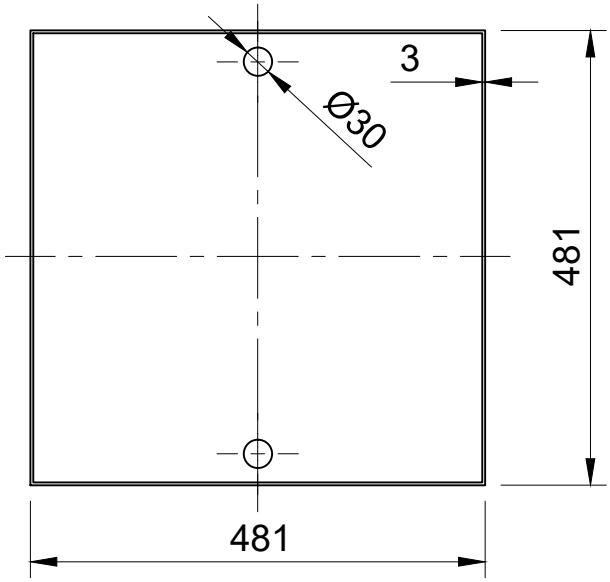
B (1:2)



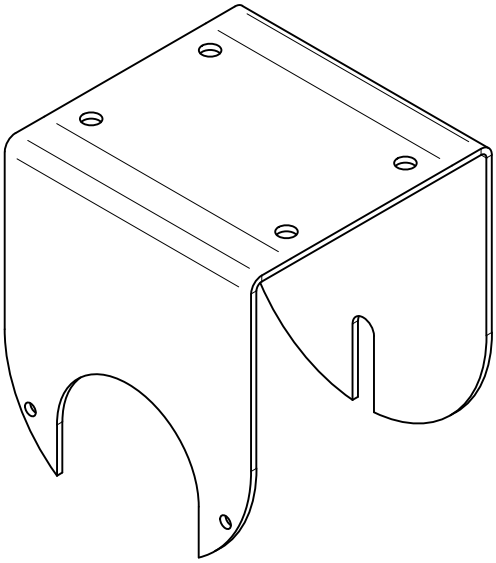
Grupo	Projeto	Autor		Material		Massa	Diedro
4	Robô Garçom	Luiz Georg	19/09/2021	Alumínio 6061 2mm		0.7 kg	
		Revisor		Unidade	Escala	Tamanho	Rev.
		Isaac Alencar	24/10/2021	mm	1:5	A4	2
		Título				Data de Emissão	Página
		Mesa de levantamento				24/10/2021	12



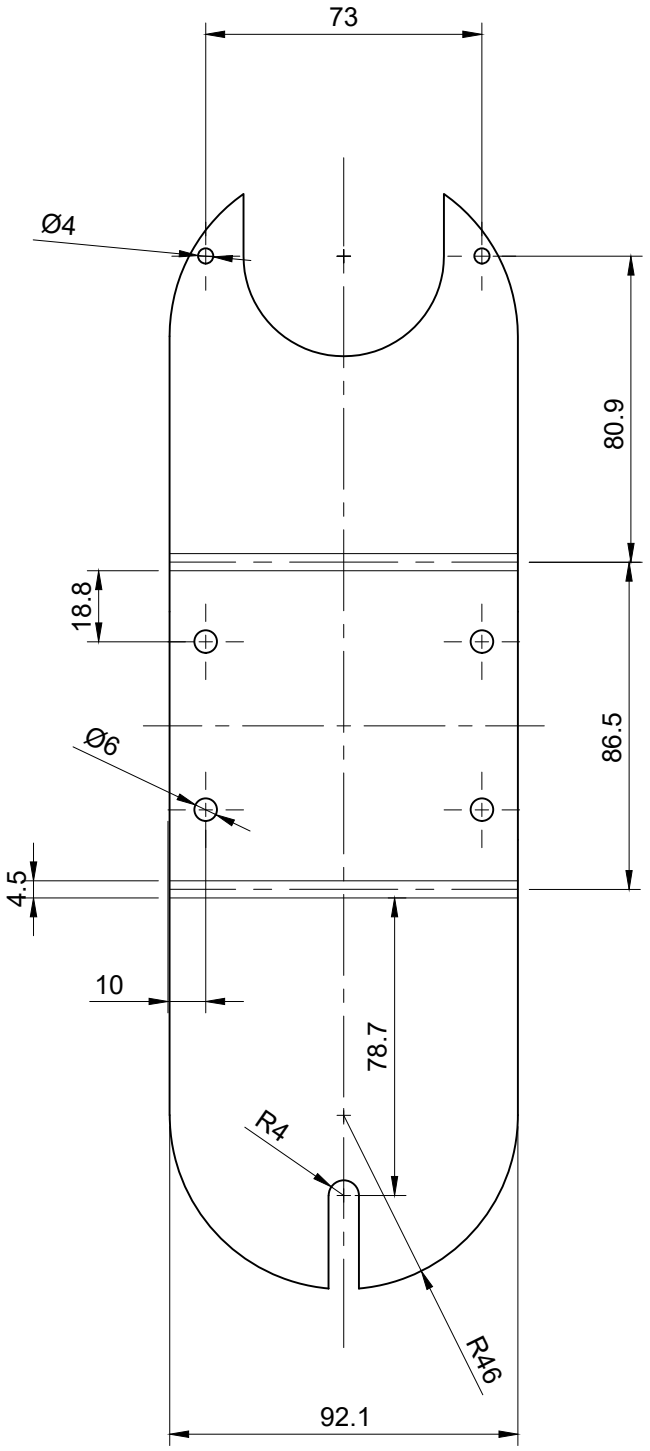




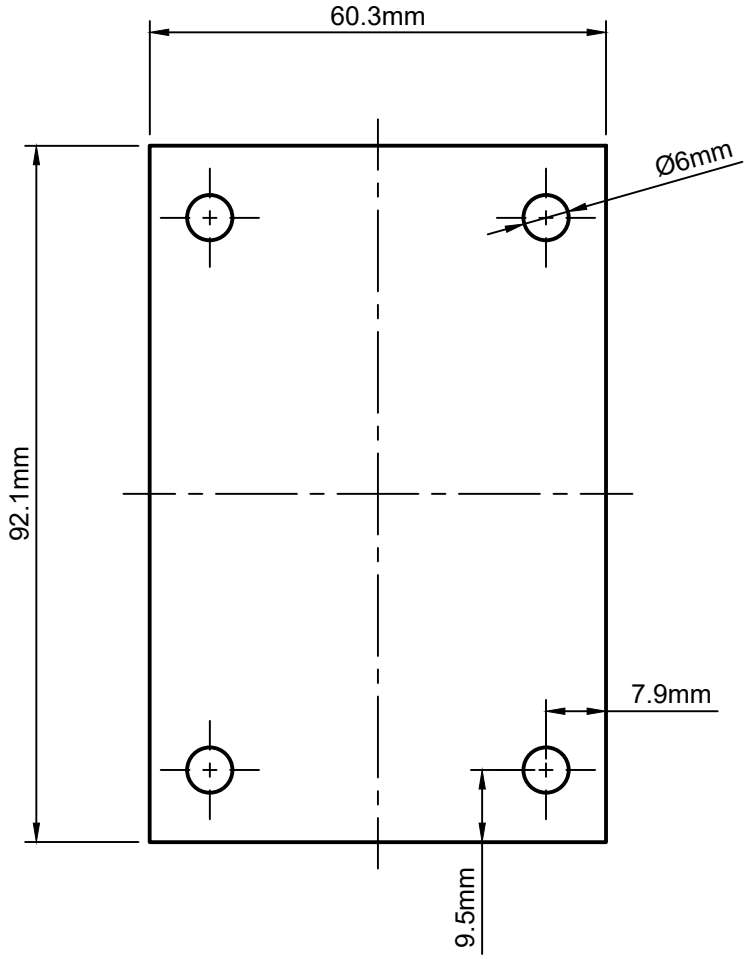
Grupo	Projeto	Autor		Material		Massa	Diedro
04	Robô garçom	Isaac Alencar	10/24/2021	ACM		4.6kg	
		Revisor		Unidade	Escala	Tamanho	Rev.
		Isaac Alencar	24/10/2021	mm	1:8	A4	1
		Título			Data de Emissão		Página
		Casca compartimento			24/10/2021		13



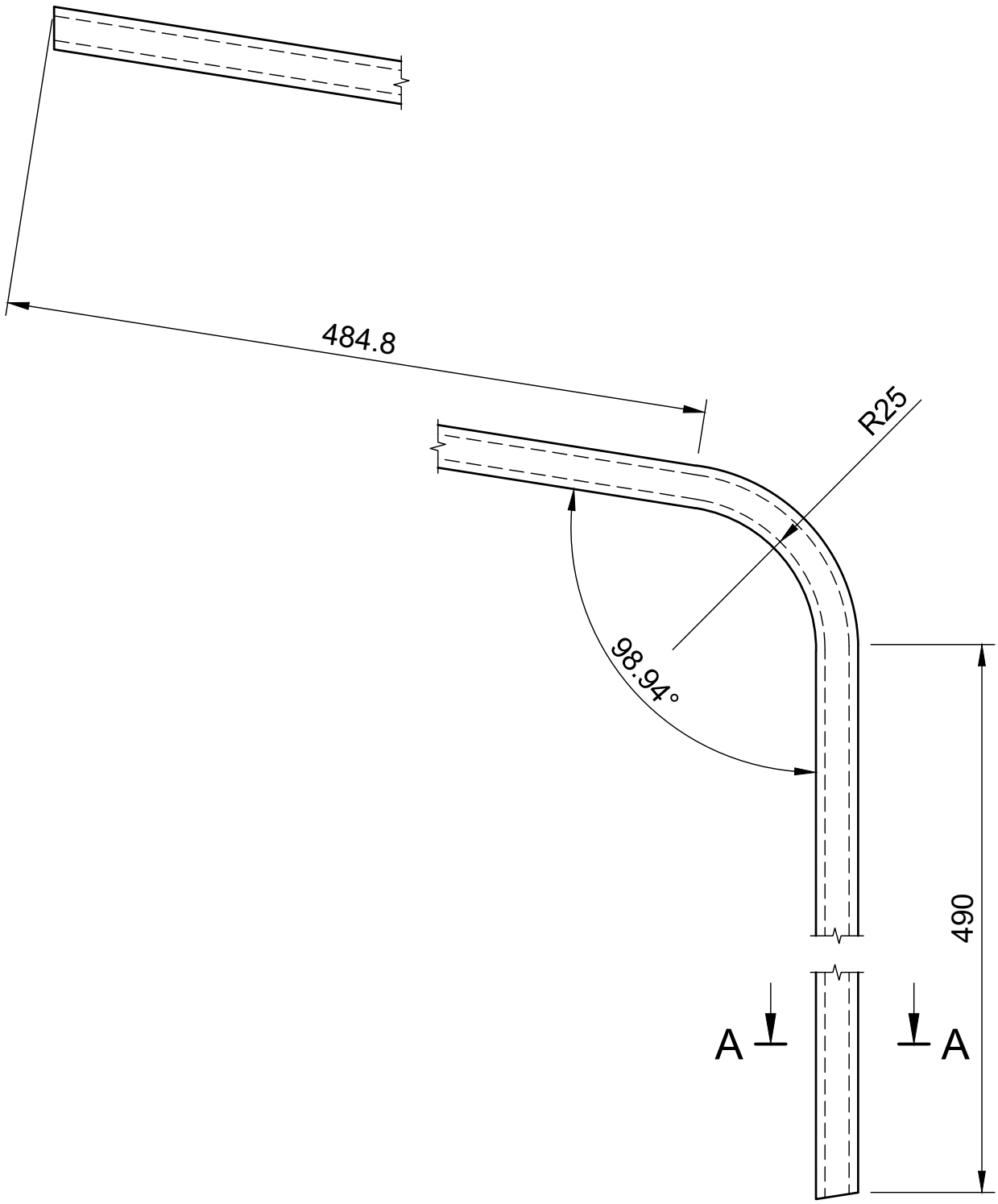
Grupo	Projeto	Autor		Material		Massa	Diedro
4	Robô garçom	Isaac Alencar	9/19/2021	Aço 1006 2mm		0,3kg	
		Revisor		Unidade	Escala	Tamanho	Rev.
		Lucas Pereira da S.	24/10/2021	mm	1:2	A4	2
		Título			Data de Emissão		Página
		garfo powercore e90			24/10/2021		14



Grupo	Projeto	Autor		Material		Massa	Diedro
4	Robô Garçom	Isaac Alencar	9/19/2021	Aço 1006 2mm		300 g	
		Revisor		Unidade	Escala	Tamanho	Rev.
		Luiz Georg	24/10/2021	mm	1:2	A4	2
		Título			Data de Emissão		Página
		Garfo p/ rodas motorizadas			24/10/2021		14

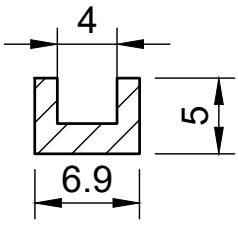


Grupo	Projeto	Autor		Material		Massa	Diedro
4	Robô Garçom	Luiz Georg	19/09/2021	Aço 1006 2mm		86 g	
		Revisor		Unidade	Escala	Tamanho	Rev.
		Isaac M. Alencar	24/10/2021	mm	1:1	A4	2
		Título			Data de Emissão		Página
		Suporte Rodas			24/10/2021		16

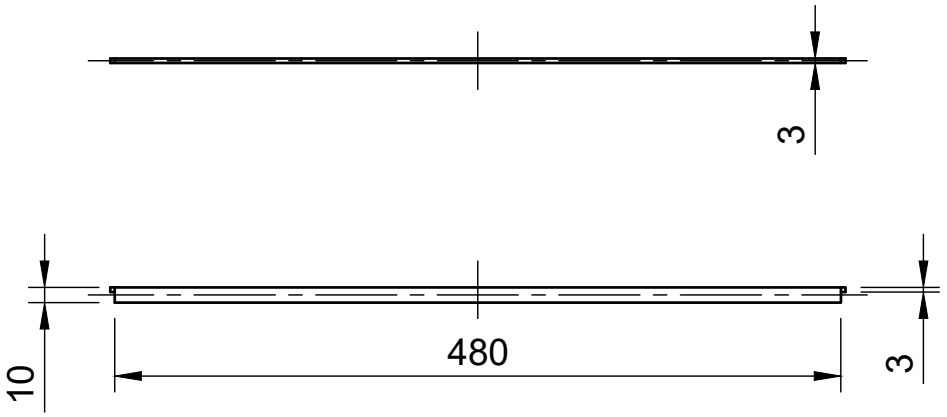


Obs: o segundo trilho
é uma versão
espelhada dessa

A-A (2:1)



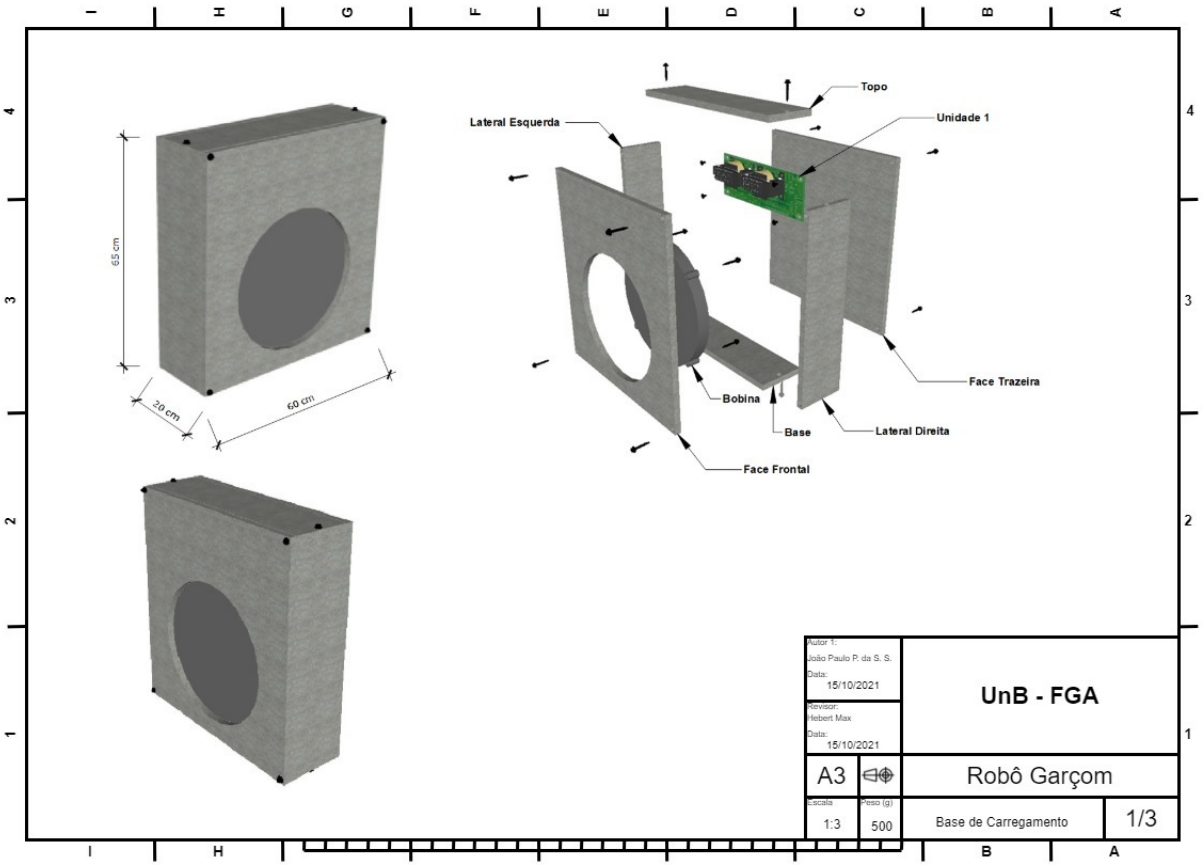
Grupo	Projeto	Autor		Material		Massa	Diedro
04	Robô garçom	Isaac Alencar	24/10/2021	Aço 1006		180g	
		Revisor		Unidade	Escala	Tamanho	Rev.
		Luiz Georg	24/10/2021	mm	1:8	A4	1
		Título			Data de Emissão		Página
		Trilho 1			24/10/2021		17



Grupo	Projeto	Autor		Material		Massa	Diedro
04	Robô garçom	Isaac Alencar	24/10/2021	Aço 1006		113g	
		Revisor		Unidade	Escala	Tamanho	Rev.
		Isaac Alencar	24/10/2021	mm	1:5	A4	1
		Título			Data de Emissão		Página
		Filamento_tampa			24/10/2021		18

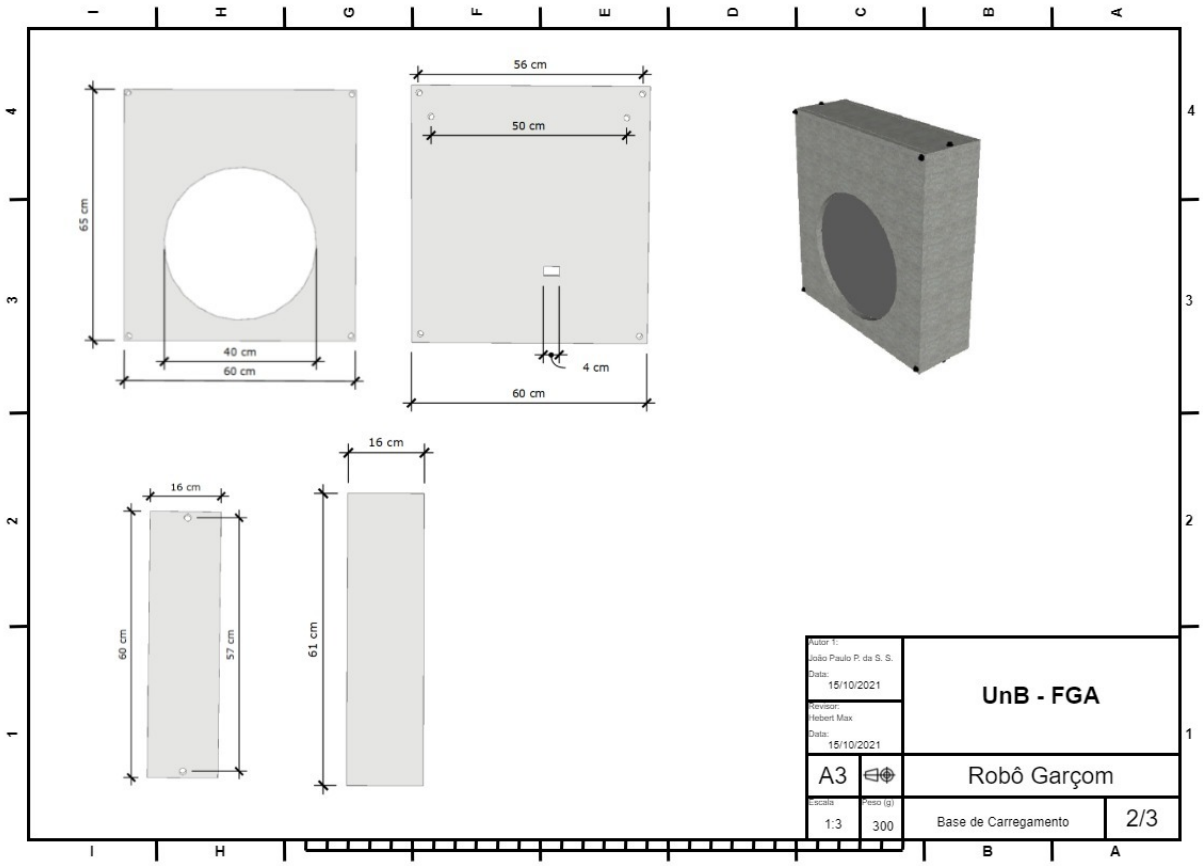
APÊNDICE K – Carcaça do carregador

Figura 83 – Vista em 3D da base de carregamento.



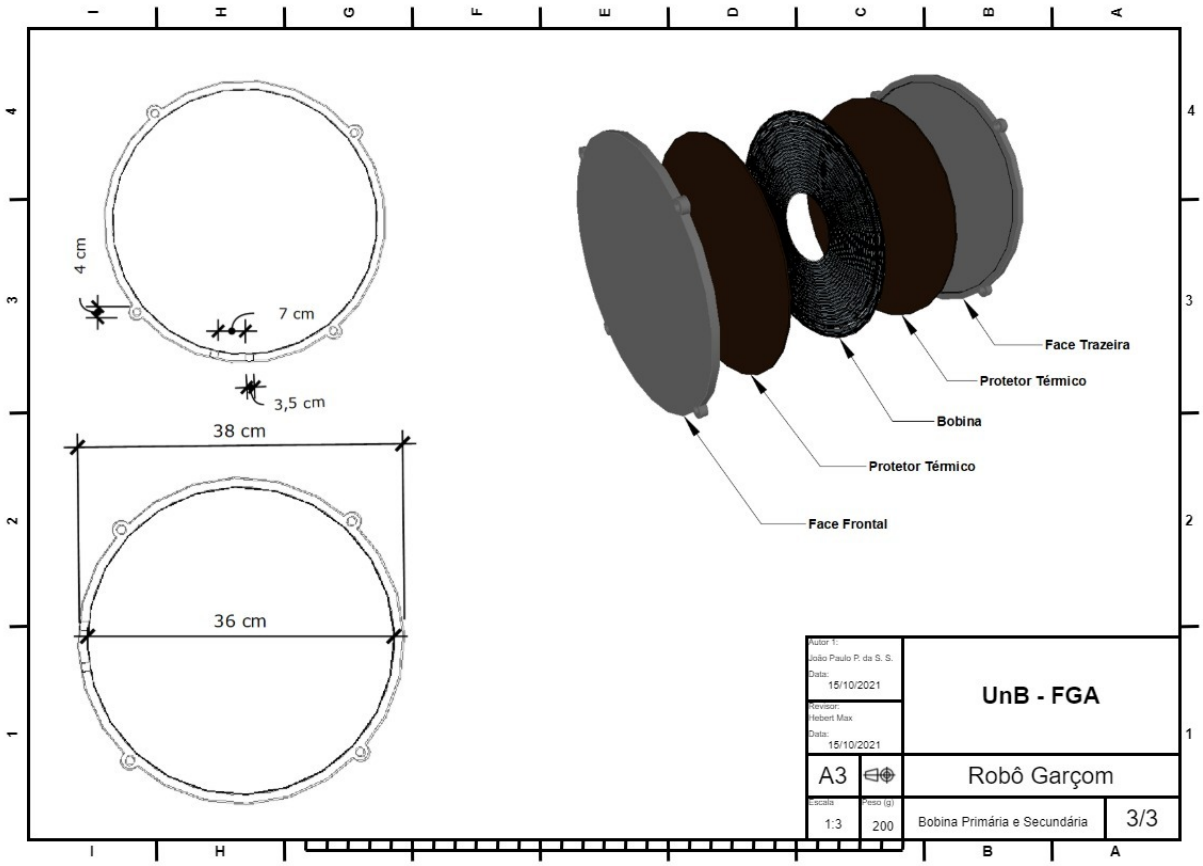
Fonte: Autores.

Figura 84 – Vista em 2D da base de carregamento.



Fonte: Autores.

Figura 85 – Detalhamento da bobina.



Fonte: Autores.