UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ (UTFPR)
CURSO DE ENGENHARIA DE COMPUTAÇÃO

MATHEUS VINICIUS BARCARO TURATTI
MURILO MASCARIN GUIMARÃES
THOMAZ HUGO SUZUKI PEREIRA

SMART BARTENDER

OFICINA DE INTEGRAÇÃO 2 – RELATÓRIO FINAL

CURITIBA

2021

MATHEUS VINICIUS BARCARO TURATTI
MURILO MASCARIN GUIMARÃES
THOMAZ HUGO SUZUKI PEREIRA

SMART BARTENDER

Relatório Final da disciplina Oficina de Integração 2, do curso de Engenharia de Computação, apresentado aos professores que ministram a mesma na Universidade Tecnologica Federal do Paraná como requisito parcial para obtenção da aprovação na disciplina.

Orientador: Prof. Dr. César Manuel Vargas Benítez
Prof. Dr. Heitor S. Lopes

CURITIBA

2021

RESUMO

. SMART BARTENDER. 35 f. Oficina de Integração 2 – Relatório Final – Curso de Engenharia de Computação, UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ (UTFPR). Curitiba, 2021.

Relátorio final do projeto smart Bartender, que busca automatização do processo de servir drinks e pagar pelos mesmos

Palavras-chave: Smart Bartender, automatização, drink

ABSTRACT

. SMART BARTENDER. 35 f. Oficina de Integração 2 – Relatório Final – Curso de Engenharia de Computação, UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ (UTFPR). Curitiba, 2021.

Smart Bartender final project report, this project seeks automatization on the pouring drink process and pay for the drink

Keywords: Smart Bartender, automation, drink

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

1.1 VISÃO GERAL

Este relatorio tem como objetivo apresentar o projeto desenvolvido ao longo da disciplina ”Oficina de Integração 2”, durante o primeiro semestre de 2021. O projeto SmartBartender visa automatizar as interações ao servir bebidas, e para isso foi necessário fazer um site para realizar os cadastros necessários e os pedidos dos clientes, para a realização do site utilizamos diversas tecnologias como flask, gunicorn, nginx, Azure, SQLite. utilizamos python para realizar a coleta de informação de todos os sensores do Raspberry Pi e controle do servo motor e controle de bombas.

O objetivo deste documento é apresentar o projeto, suas etapas de desenvolvimento, problemas encontrados ao longo deste processo, assim como tabela de custos do projeto.

Figura 1: Projeto completo

Fonte: Autoria própria

1.2 MOTIVAÇÃO

Fomos motivados pela necessidade de automatizar a distribuição de bebidas em um ambiente de festa. Diminuindo a interação humana, garantindo um padrão elevado de higiene e segurança devido a realidade atual.

1.3 OBJETIVOS

1.3.1 OBJETIVO GERAL

O objetivo desse projeto foi criar um Bartender automatizado, onde o cliente pode fazer seu pedido por um site e retirar a sua bebida sem nenhum contato humano, administrar créditos comprados anteriormente e controle de estoque de bebidas

1.3.2 OBJETIVOS ESPECÍFICOS

O administrador deverá fazer os cadastros necessários, tais como fontes e drinks disponíveis e créditos dos usuários. O cliente deverá realizar o pedido pelo site, onde o site comunica com o raspberry pi, então ele irá atender o pedido do cliente, por meio de sensores e controle de bombas peristálticas

2 FUNDAMENTAÇÃO TEÓRICA

O projeto foi realizado com o uso do Raspberry Pi para todas as tarefas de processamento. Raspberry Pi é um microcontrolador, ou seja, é um hardware integrado, capaz de se conectar com diversos dispositivos de entrada e saída, com capacidade para gerenciar e controlar esses dispositivos. A forma de comunicacão utilizada com o microcontrolador foi através de um site, para uma melhor experiência do cliente.

Para desenvolvimento dos softwares, se tomou como princípio a implementação e a execução do mesmo ser dividida entre o Raspberry Pi e a computação na nuvem. Para isso, utilizou-se de um Sistema Operacional da distribuição Debian para o microcontrolador, conhecido como Raspbian e a plataforma Azure para o serviço de computação em nuvem.

O software executado pelo Raspberry Pi, foi desenvolvido na linguagem Python, e com isso, foi possível realizar a leitura dos dados gerados pelos sensores de proximidade e carga, manipular a atuação das bombas peristálticas e servo motor.

O processamento em nuvem da Azure foi configurado utilizando material de apoio da própria empresa, nele foi contratado uma máquina virtual na modalidade IaaS (Infrastucture as a Service), após a finalização do projeto iremos encerrar o serviço e o mesmo não irá gerar novas cobranças.

Na máquina virtual foi configurado Flask que é utilizado pelo webserviço Gunicorn que atende em múltiplas portas da máquina, o trabalho a ser feito é redirecionado da porta 80, essa porta é controlada pela ferramenta Nginx, todo a configuração foi feita com auxílio de um tutorial (JUELL; CAMISSO, 2020) referente ao sistema operacional escolhido a ser utilizado na máquina virtual.

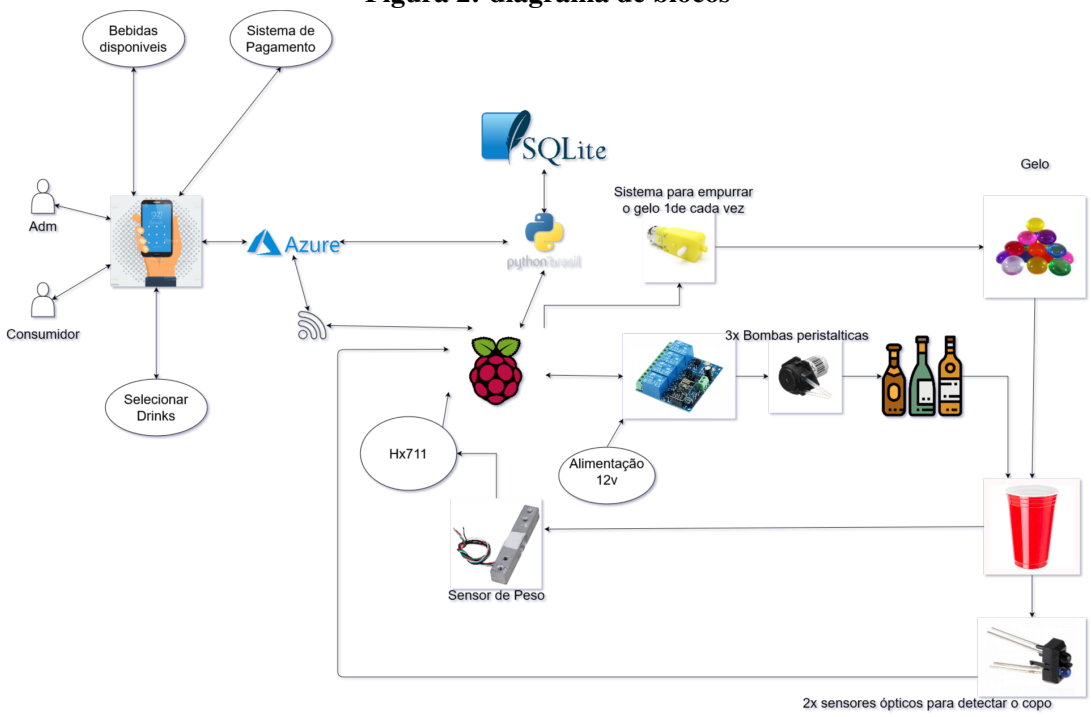
A aplicação configurada no Flask foi desenvolvida em partes com auxílio de um tutorial de criação de blog (SCHAFER, 2019), que em muito se assemelhava nosso caso de uso.

3 METODOLOGIA

Para a realização do projeto, utilizamos Azure para hospedar o site do projeto,Python para fazer a lógica realizada pelo Raspberry e o controle de seus sensores e saídas lógicas, para o banco de dados utilizamos o SQLite

3.1 VISÃO GERAL

Figura 2: diagrama de blocos

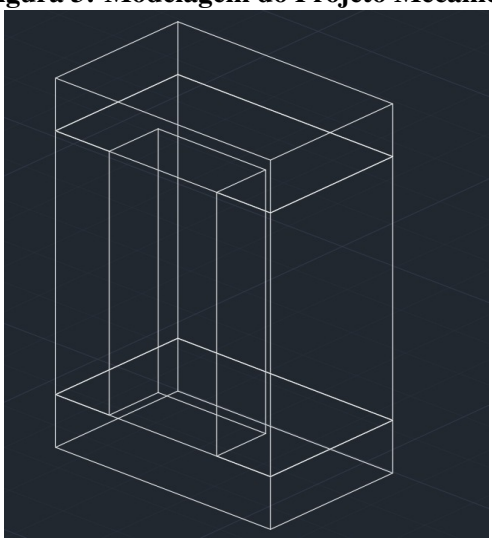


Fonte: Autoria própria

3.2 PROJETO MECÂNICO

O projeto Mecânico foi feito em MDF, onde pedimos uma estrutura personalizada que comportasse todo o hardware, a estrutura foi baseada na seguinte modelagem:

Figura 3: Modelagem do Projeto Mecanico



Fonte: Autoria própria

E o resultado da estrutura foi a seguinte:

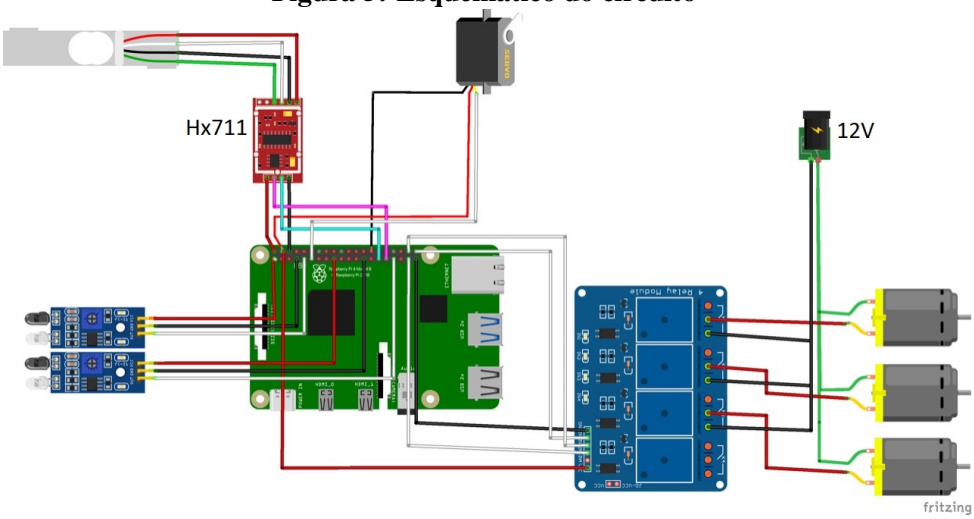
Figura 4: Projeto Mecanico

Fonte: Autoria própria

3.3 PROJETO DE HARDWARE

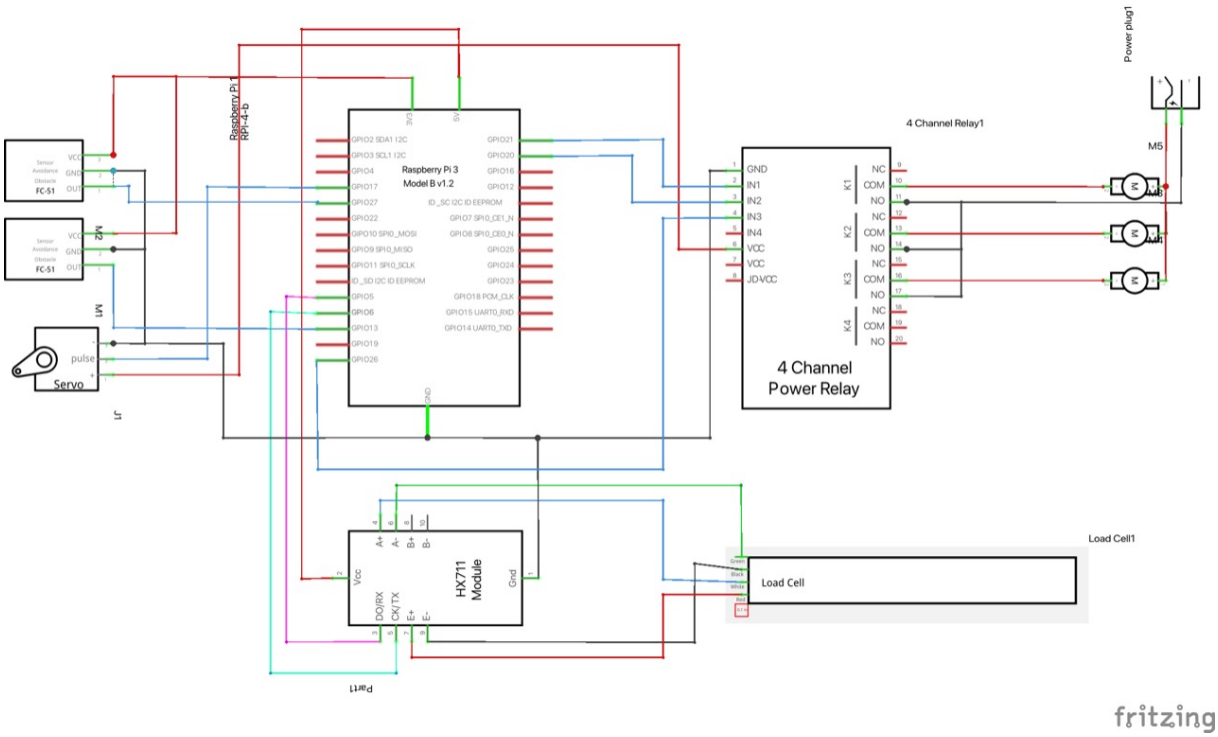
As seções abaixo apresentam o hardware do projeto. Foi utilizado um Raspberry Pi, uma célula de carga conectada a um HX711, três bombas peristálticas, dois sensores infravermelho, um servo motor, fonte de alimentação de 12v e um relé de 4 canais.

Figura 5: Esquemático do circuito



Fonte: Autoria própria

Figura 6: Diagrama elétrico do circuito



Fonte: Autoria própria

3.3.1 SENSORES INFRAVERMELHO

Foram utilizados 2 módulos sensores infravermelho, dispostos próximos à área de despejo dos drinks, no intuito de capturar as informações de posição do copo e garantir que a máquina só sirva o drink caso o copo esteja corretamente inserido, evitando o derramamento e possíveis prejuízos.

3.3.2 CÉLULA DE CARGA

A célula de carga foi posicionada na base da área de inserção do copo, possibilitando a captura das informações de peso do mesmo, permitindo calcular a quantidade de bebida despejada no copo, de forma a assegurar que o copo não transborde com a quantidade de líquido inserida. Foi necessário a utilização da célula de carga conectada ao HX711, que amplifica o sinal do sensor possiblitando sua leitura pelo Raspberry Pi.

3.3.3 SERVO MOTOR

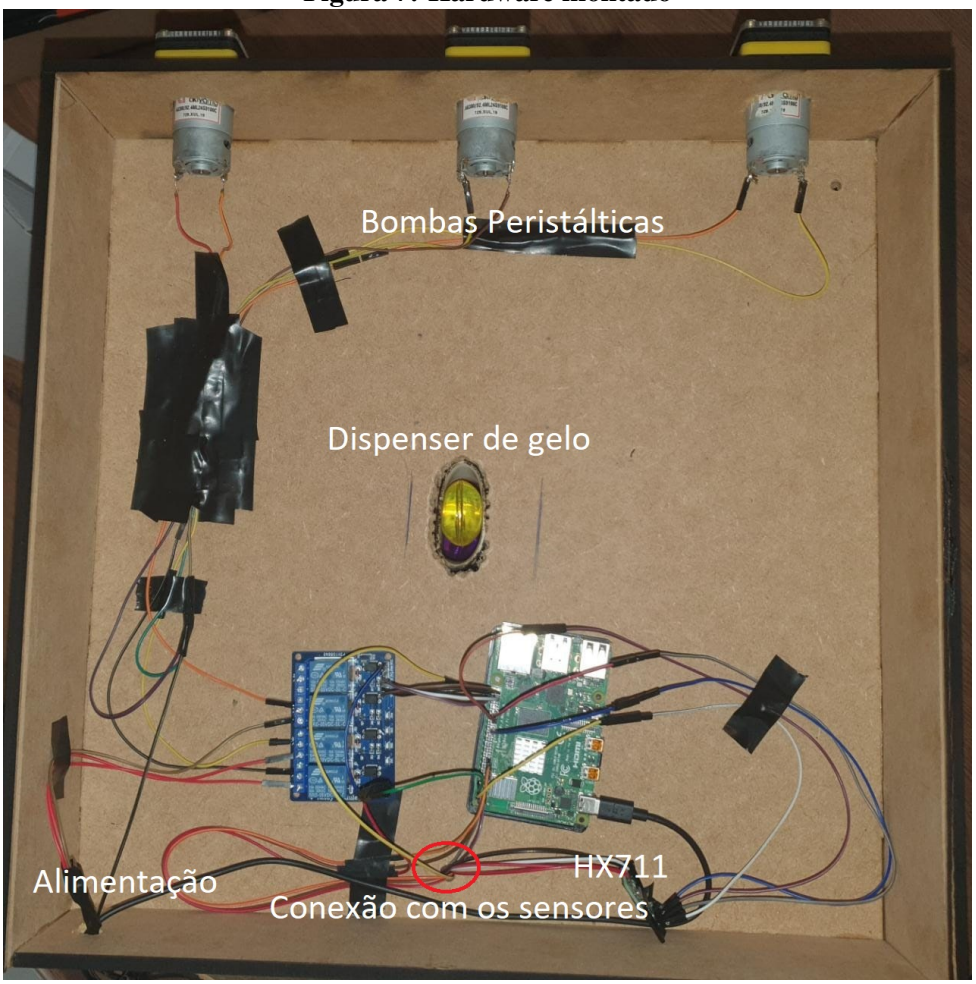
O servo motor foi inserido junto à estrutura mecânica do dispenser de gelo. Sendo responsável por garantir que será servido apenas uma pedra de gelo por drink.

3.3.4 BOMBAS PERISTÁLTICAS E RELÉ

As bombas peristálticas foram instaladas utilizando-se tubos de silicone atóxico para garantir a segurança do cliente no consumo dos drinks preparados pelo Smart Bartender. As bombas foram conectadas ao relé, possibilitando o controle do tempo de ativação individual.

3.3.5 INTEGRAÇÃO MECÂNICA E DO HARDWARE

Figura 7: Hardware montado



Fonte: Autoria própria

3.4 PROJETO DE SOFTWARE

Para a realização do software utilizamos as seguintes ferramentas e serviços: Azure, serviço de DNS da google,NGINX, web framework Gunicorn, Flask, SQLite, Sqlalchemy, Flask-WTForm, Jinja2, Bcrypt, PIL

3.4.1 AZURE

Formada por uma série de ferramentas diferentes, o Azure é um serviço que permite a empresas e desenvolvedores adquirirem as capacidades de processamento e armazenamento dos datacenters da Microsoft para aplicação em seus negócios como alternativa ao modelo convencional.
Utilizamos a azure para manter o hardware funcionando e atender os requisitos funcionais na parte do web service.

3.4.2 DNS GOOGLE

O DNS do Google, criado em 2009, visa melhorar a qualidade da conexão dos usuários. Uma de suas características é redirecionar o acesso para o data center mais próximo, o que colabora para o carregamento das páginas e reduz a taxa de erros. Utilizamos esse serviço para mapear o nome do site para o endereço ip

3.4.3 NGNIX

NGINX é um servidor web que também funciona como proxy de email, proxy reverso, e balanceador de carga. A estrutura do software é assíncrona e orientada a eventos; possibilitando o processamento de muitas solicitações ao mesmo tempo. Esse serviço foi usado como reverse proxyng para mapear os diferentes chamados do Port 80(internet) ligando com Workers em vários ports diferentes e esses workers criam forks de processamento pelo Gunicorn, que é responsável pela criação de processos em que os usuário poderão interagir, fazendo assim que uma máquina possa atender vários pedidos.

3.4.4 SQLITE

Como banco de dados, utilizamos o SQLite, que é um banco de dados relacional, para armazenar todos os nossos dados e registros, escolhemos esse banco de dados por ser leve e

estar na própria máquina. Para manipular tabelas e registros, utilizamos o SQLalchemy que serve de ORM (Object-Relational Mapping), para facilitar a escrita no banco de dados

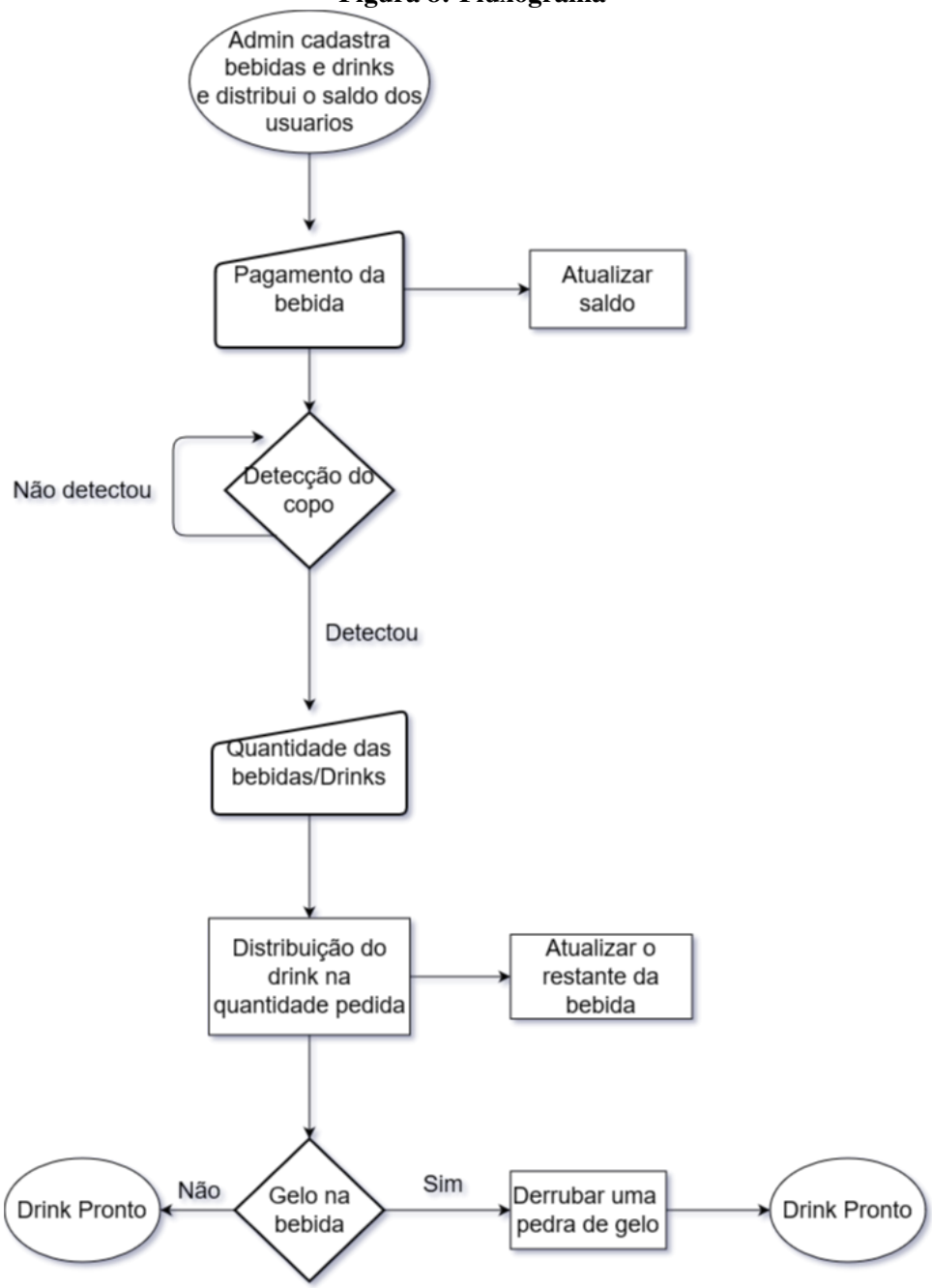
3.4.5 FLASK-WTFORM

Foi utilizado o Flask-WTForm para dar validações as estruturas do site, como os botões, drop-box, check-box, entre outros mecanismos que se utilizam de input do usário. utilizamos também o PIL para realizar a modelagem de imagens do site e Bcrypt para proteger a senha do usuário.

3.4.6 DIAGRAMAS

O projeto se resume a cadastros por parte de um usuário administrador e consumo por parte do cliente.A seguir um fluxograma de como a aplicação funciona pode ser observado:

Figura 8: Fluxograma

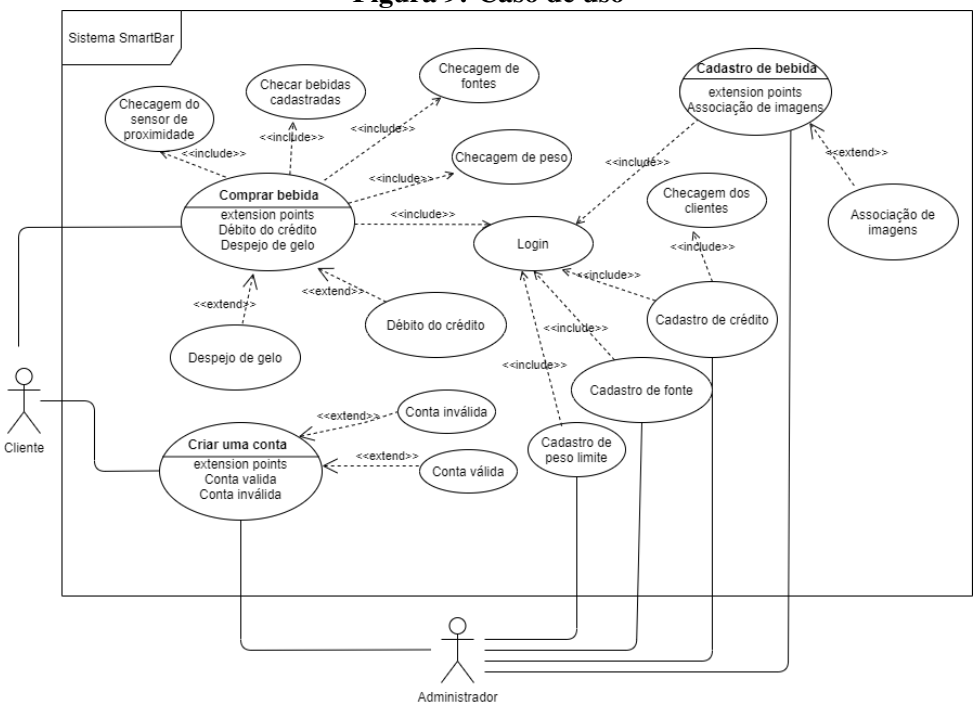


Fonte: Autoria própria

O diagrama de casos de uso relata que essencialmente são apenas dois os Atores que interagem com o sistema, sendo que o Administrador possui diversas funções que, conforme poderemos averiguar nos diagramas de sequência, garantem a disponibilidade do sistema.

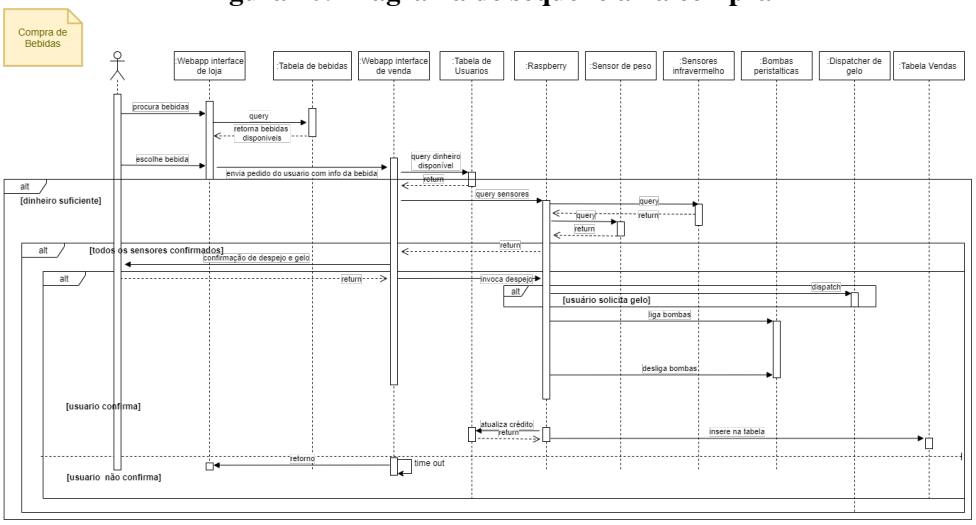
O primeiro diagrama de sequência é o de interações que ocorrem durante a compra de uma bebida, temos então a principal interação do Cliente com o sistema.

Figura 9: Caso de uso



Fonte: Autoria própria

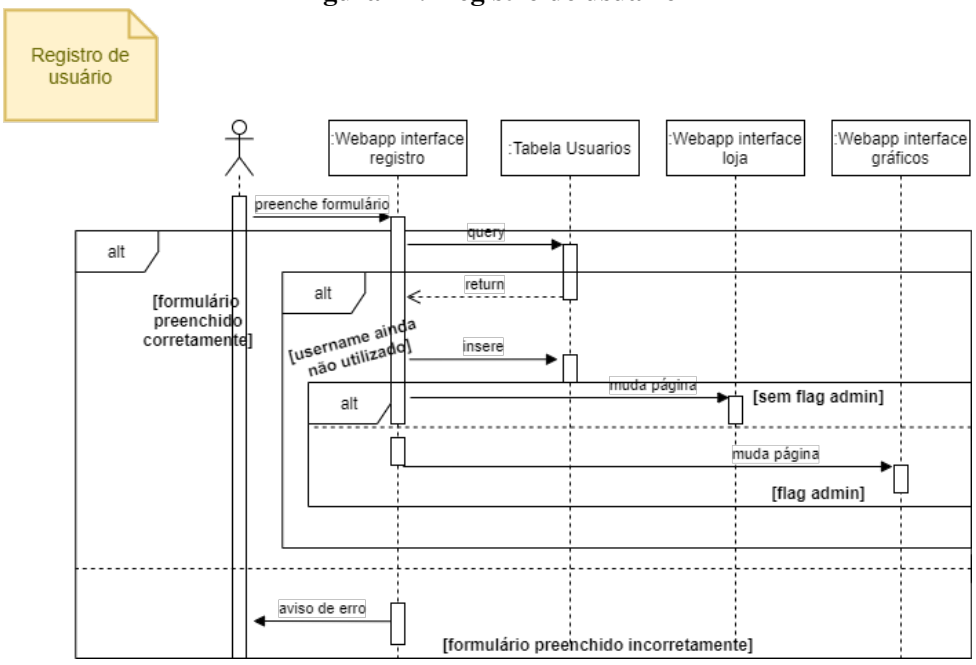
Figura 10: Diagrama de sequência na compra



Fonte: Autoria própria

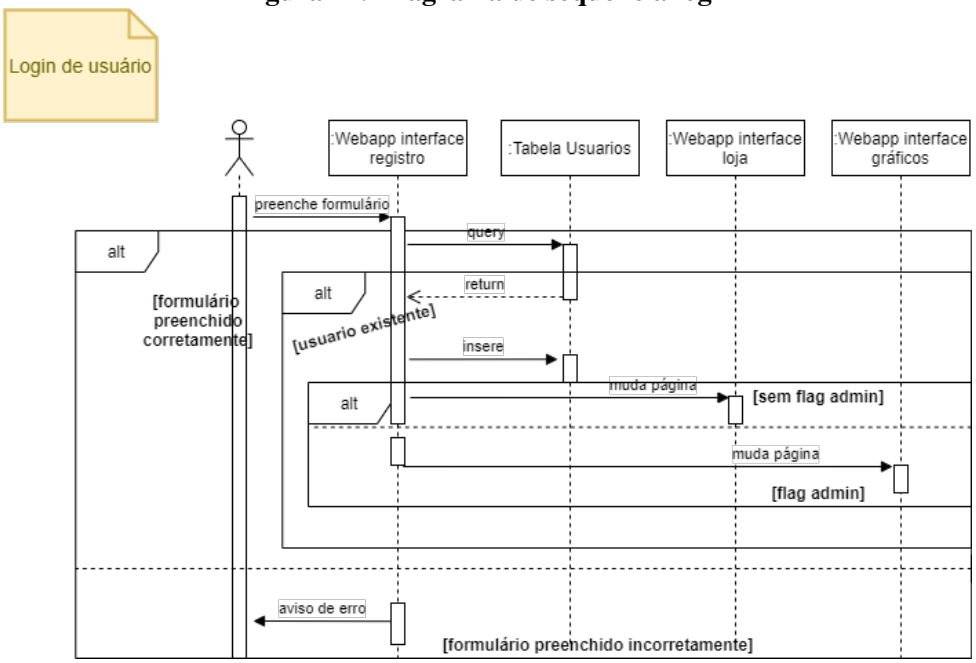
As outras funcionalidades que Cliente tem acesso, são Registrar uma conta e Logar na mesma.

Figura 11: Registro de usuário



Fonte: Autoria própria

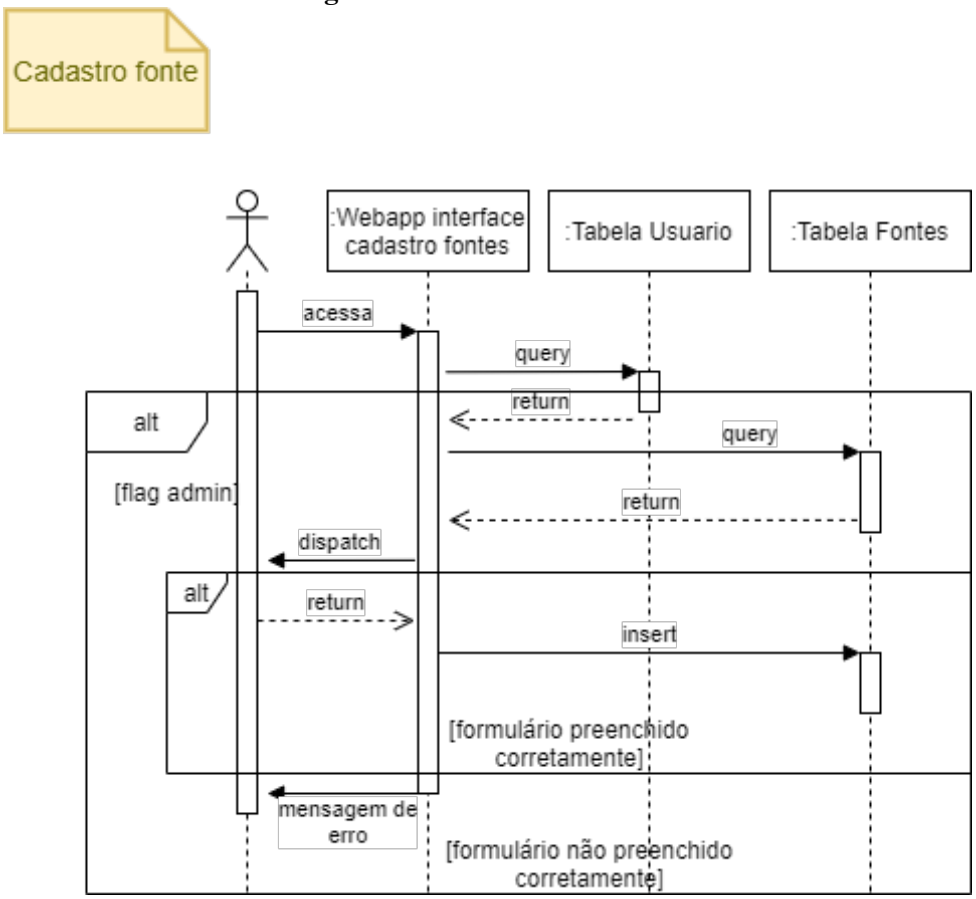
Figura 12: Diagrama de sequencia login



Fonte: Autoria própria

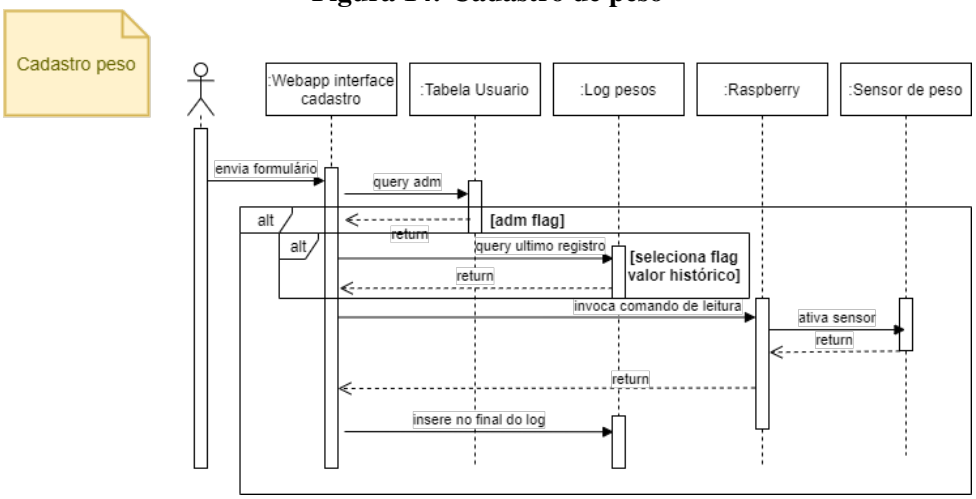
Já as funções de Administrador se resumem a cadastros.

Figura 13: Cadastro de Fontes



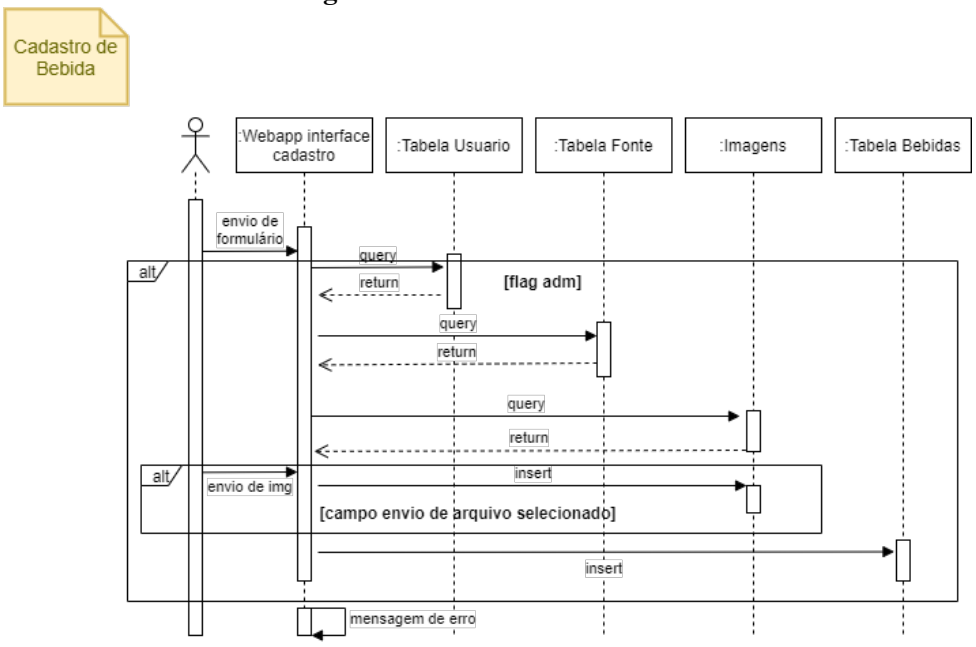
Fonte: Autoria própria

Figura 14: Cadastro de peso



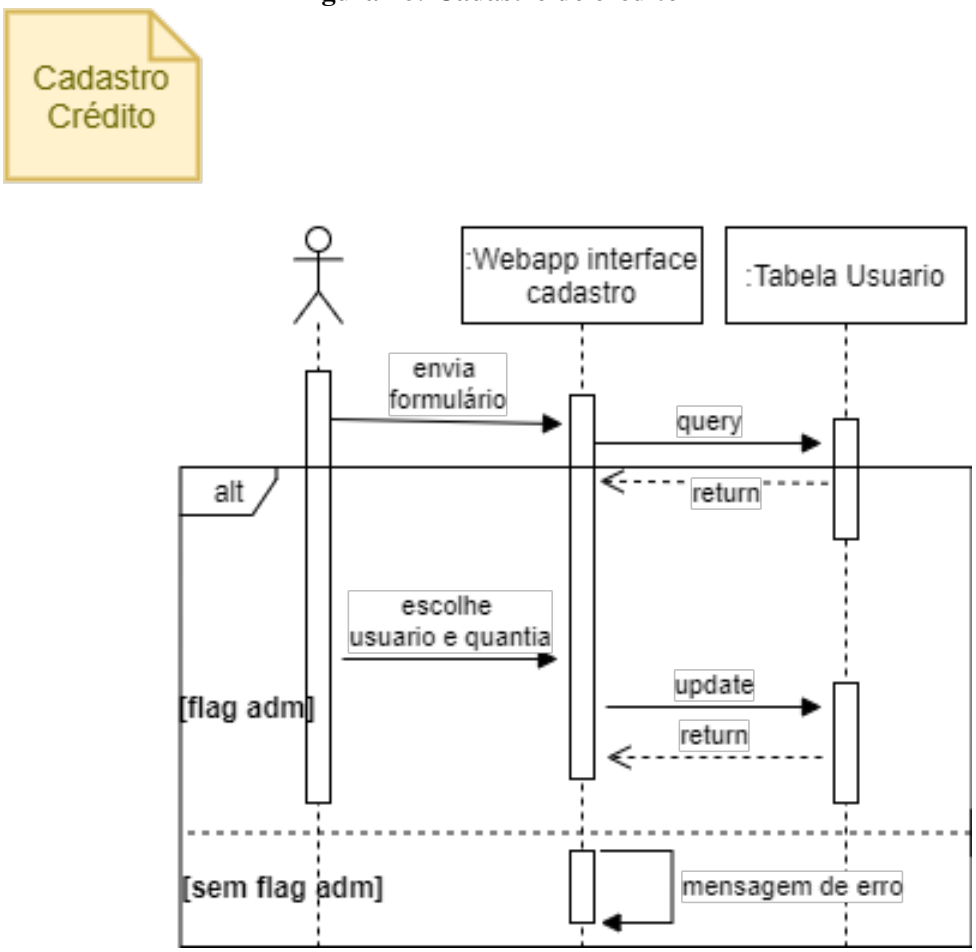
Fonte: Autoria própria

Figura 15: Cadastro de bebida



Fonte: Autoria própria

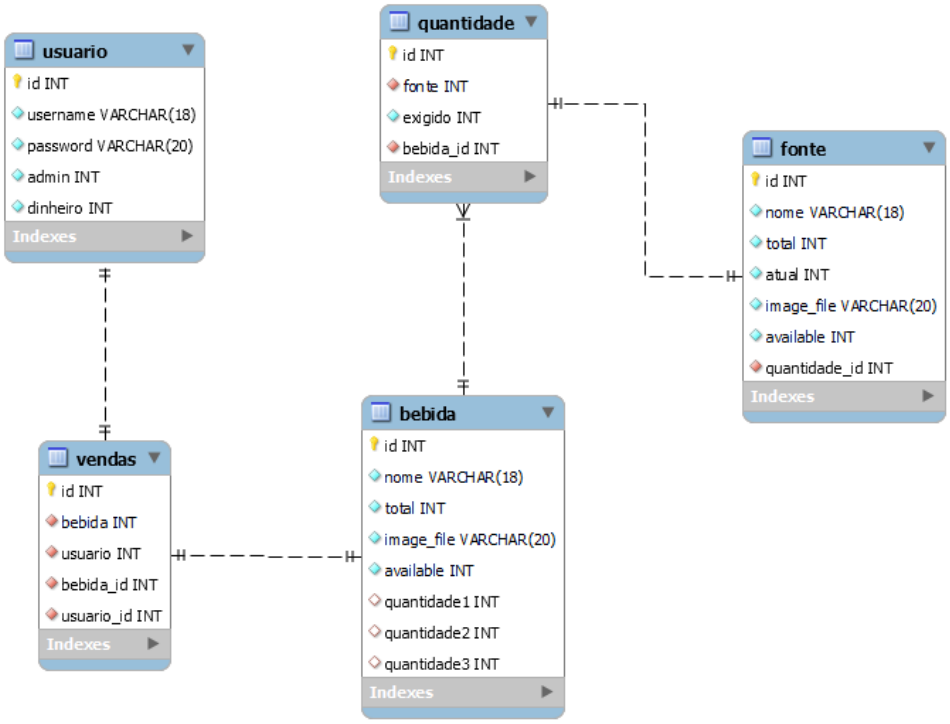
Figura 16: Cadastro de crédito



Fonte: Autoria própria

E por último o diagrama de Entidade-Relacionamento do banco de dados utilizado

Figura 17: Banco de dados



Fonte: Autoria própria

3.5 INTEGRAÇÃO

Esta seção é constituída pela etapa de integração dos módulos do projeto. Estando incluído a comunicação entre as chamadas do site e execução no Raspberry Pi localmente através dos softwares descritos na seção a cima. Quando o cliente fizer o pedido e o copo estiver posicionado corretamente, o Raspberry Pi irá acionar os relés, para as bombas peristálticas começarem a bombear as bebidas, de acordo com o tempo necessário para preencher o copo, após isso o servo motor é acionado despejando um gelo no copo. Caso o copo seja retirado durante o processo de despejo da bebida as bombas irão parar no mesmo momento.

4 EXPERIMENTOS E RESULTADOS

Os testes mais difíceis de serem realizados, foram o de peso e o despejo do gelo, a detecção do peso foi um problema no começo pois, não havia muito material sobre a célula de carga que havíamos comprado, apresentava muita variação do peso real do objeto que estava posicionado em cima e criar uma base que todo o peso do objeto estivesse concentrado na célula. Então optamos por conprar outro modelo de célula de carga e os resultados obtidos foram bem melhores

Referente ao despejo de gelo, a maior dificuldade foi criar um método que fizesse o gelo cair somente uma unidade. Resolvemos isso criando uma estrutura de cano pvc que alinhasse os gelos, fazendo assim o servo motor empurrar unitariamente os gelos.

Nosso outro grande e maior problema encontrado foi relaizar a comunicação de modo geral, do site com o Raspberry

5 CRONOGRAMA E CUSTOS DO PROJETO

5.1 CRONOGRAMA

Figura 18: Cronograma

	A	B	C	D	E	F	G	H
1	Feito no Tempo Correto		A ser feito	Deadline				
2	Em progresso	Atrasado	Iniciado Antes	Feito antes				
3	Atividade/entregas	Responsável	Ajudante	Status	Início	Fim	Total estimado (horas)	Real (horas)
4	Plano de Projeto detalhado	Todos			01/07	07/07	7	5
5	Site/blog de acompanhamento	Todos			07/07	14/07	19	10
6	Projeto e montagem da estrutura mecânica	Todos			07/07	21/07	41	11
7	Projeto eletrônico e testes de hardware	Todos			14/07	28/07	10	30
8	Projeto do software com UML	Todos			14/07	04/08	79	85
9	Integração mecânica + hadware + software	Todos			28/07	11/08	32	53
10	Demonstração do funcionamento do protótipo	Todos			11/08	18/08	7	
11	Relatório Técnico	Todos			28/07	18/08	21	15
12	Vídeo de 15-20 minutos com a apresentação e demonstração do projeto	Todos			18/08	25/08	7	
13	Avaliação por uma banca de professores	Todos			01/09	01/09	1	
14							224	209

Fonte: Autoria própria

Link para o cronograma completo:
https://docs.google.com/spreadsheets/d/1oinIaGECioSk7ImVLZIM7tvNnt9PvSmKvX52TTydd1g/edit?usp

5.2 CUSTOS

Tabela de custos de todos os componentes do projeto

Item	**Preço**
MDF	R$ 160,00
Raspberry PI	R$ 370,13
Célula de carga	R$ 23,00
3 * Bomba Peristáltica	R$ 236,70
Fonte 12V	R$15,80
2 * Sensor de Proximidade	R$ 85,36
Hx711	R$ 23,24
Tubo de silicone	R$ 22,00
Shield Relé 4 canais	R$ 44,90
Servo motor	R$ 18,90

Total: R$ 1000,03

6 CONCLUSÕES

O desenvolvimento e a execução do projeto descrito nesse relatório levou a uma série ´ de observações e conclusões para o grupo. Em relação ao projeto, o formato de desenvolvimento individual com o processo de integração ao longo desse desenvolvimento se mostrou bastante efetiva, demonstrando que a organização prévia do desenvolvimento e a criação do cronograma em etapas serviu como uma base de estudo, alem de demonstrar a importáncia e a efetividade do trabalho em equipe para desenvolvimentos de projetos.apesar de alguns atrasos em entregas pontuais, na entrega final o resultado esperado foi obtido.

além da integração entre os conhecimentos dos estudantes, que foram agregados ao longo do projeto, como a integração hardware e software, o desenvolvimento de software em cima de um microcontrolador, utilização de sensores e atuadores, além dos demais conhecimentos envolvendo a pesquisa técnica e a aplicação de ideias para encontrar soluções

REFERÊNCIAS

JUELL, K.; CAMISSO, J. **How To Serve Flask Applications with Gunicorn and Nginx on Ubuntu 20.04**. 2020. Disponível em: <https://www.digitalocean.com/community/tutorials/how-to-serve-flask-applications-with-gunicorn-and-nginx-on-ubuntu-20-04>. Acesso em: 18 de agosto de 2021.

SCHAFER, C. **Flask Tutorials**. 2019. Disponível em: <https://www.youtube.com/playlist?list=PL-osiE80TeTs4UjLw5MM6OjgkjFeUxCYH>. Acesso em: 18 de agosto de 2021.