

**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA**  
**ESCOLA TÉCNICA ESTADUAL DA ZONA LESTE**  
**Técnico em Desenvolvimento de Sistemas**

**David Soares Silva**  
**Guilherme Leo de Oliveira**  
**Julia Rodrigues Rocha Franco de Freitas**  
**Maria Eduarda Fiori**

**FOUR HOUSE: Automação Residencial**

**São Paulo**  
**2022**

**David Soares Silva**  
**Guilherme Leo de Oliveira**  
**Julia Rodrigues Rocha Franco de Freitas**  
**Maria Eduarda Fiori**

## **FOUR HOUSE: Automação Residencial**

Trabalho de Conclusão de Curso apresentado ao Curso Técnico em Desenvolvimento de Sistemas da Escola técnica estadual da Zona Leste, orientado pelo Prof. Rogério Bezerra Costa, como requisito parcial para obtenção do título de técnico em Desenvolvimento de Sistemas.

**São Paulo**  
**2022**

## **AGRADECIMENTOS**

Agradecemos a nossa antiga orientadora e professora Tamara, por guiar-nos durante o processo de realização deste projeto, por acreditar em nós e nos corrigir para podermos alcançar o nosso melhor desempenho.

Agradecemos também ao professor Carlos, por nos auxiliar em todas as dúvidas e confiar em nosso projeto, acima de tudo.

Ao professor Luciano, por nos incentivar durante esse processo e se pôr a disposição para nossas dúvidas.

A todos que nos incentivaram durante a realização deste projeto e certamente enriqueceram nosso conhecimento.

*“Quando algo é importante o suficiente, você realiza, mesmo que as chances não estejam a seu favor”*

**(Elon Musk)**

## AUTOMAÇÃO RESIDENCIAL

**Resumo:** A tecnologia tem se tornado cada dia mais presente em todos os lugares, em nossa rotina, em tudo ela se faz necessária, com isso trouxemos um projeto de automação residencial para trazer mais conforto, facilidade e segurança para as tarefas que são realizadas no nosso cotidiano. Este trabalho tem como objetivo implementar nas residências a automação visando mais conforto e funcionalidade no dia-a-dia. Com esse sistema podemos mostrar também, que essa modernização não é algo inalcançável e pode ter um custo-benefício acessível para todos os tipos de público. Nosso protótipo foi feito com Arduino para demonstração dos serviços prestados pela empresa.

**Palavras-chave:** Automação residencial, Arduino.

**Abstract:** The technology has become increasingly present everywhere, in our routine, in everything it is necessary, so we brought a residential automation project to bring more comfort, ease and safety for the tasks that are performed in our daily lives. This work has as objective to implement the automation in the residences aiming more comfort and functionality in the day by day. With this system we can also show that this modernization is not something unreachable, and can have an accessible cost-benefit for all types of public. Our prototype was made with Arduino to demonstrate the services provided by the company.

**Keywords:** Residential automation, Arduino.

## Lista de Figuras

Figura 1 – Estrutura Básica do HTML .....	13
Figura 2 – Cadastro de Usuário em HTML .....	14
Figura 3 - Cadastro de Usuário em HTML (2) .....	15
Figura 4 - Página Web Cadastro de Usuário .....	15
Figura 5 – Sintaxe do CSS .....	16
Figura 6 - Código do CSS estilizando o HTML .....	17
Figura 7 – Linha do código em HTML mudada para ser estilizado no CSS .....	18
Figura 8 - Página Web Cadastro de Usuários com CSS .....	18
Figura 9 - Exemplo de utilização básica do JavaScript .....	20
Figura 10 - Operadores Aritméticos .....	21
Figura 11 - Operadores de Comparação .....	21
Figura 12 - Operadores Lógicos .....	21
Figura 13 - Exemplo if e else .....	21
Figura 14 - Exemplo while .....	22
Figura 15 - JS incluído no HTML com a tag script .....	22
Figura 16 - JS importado pelo HTML de um arquivo externo (1) .....	22
Figura 17 - JS importado pelo HTML de um arquivo externo (2) .....	23
Figura 18 - JS incluído em descritores HTML sensíveis a eventos .....	23
Figura 19 - Resultado dos códigos das imagens 15, 16, 17 e 18 após clicar no botão enviar .....	23
Figura 20 - Adicionando JS na página de Cadastro .....	23
Figura 21 - Resultado da Página de Cadastro com JS .....	24
Figura 22 - Página de Cadastro sem JS .....	24
Figura 23 - Declaração de Variável em PHP .....	26
Figura 24 – Cadastro de Usuário com HTML, CSS, JS e PHP (1) .....	26
Figura 25 - Cadastro de Usuário com HTML, CSS, JS e PHP (2) .....	26
Figura 26 - Cadastro de Usuário com HTML, CSS, JS e PHP (3) .....	27
Figura 27 - Figura 26 - Cadastro de Usuário com HTML, CSS, JS e PHP (4) .....	28
Figura 28 - Página Web com HTML, CSS, JS e PHP (1) .....	28
Figura 29 - Página Web com HTML, CSS, JS e PHP (2) .....	29
Figura 30 - Página Web com HTML, CSS, JS e PHP (3) .....	30
Figura 31 - Cadastro de Usuário com HTML, CSS, JS, PHP e Bootstrap (1) .....	31
Figura 32 - Cadastro de Usuário com HTML, CSS, JS, PHP e Bootstrap (2) .....	32
Figura 33 - Como pegar códigos do Bootstrap .....	32
Figura 34 - Página Web com HTML, CSS, JS, PHP e Bootstrap .....	33
Figura 35 - Exemplo DER .....	34
Figura 36 - Exemplo MER .....	35
Figura 37 - Exemplo Diagrama de Caso de Uso .....	39
Figura 38 - Exemplo Diagrama de Sequência .....	40
Figura 39 - Exemplo Diagrama de Atividade .....	42
Figura 40 - Exemplo Diagrama de Classes .....	44
Figura 41 - Placa Arduino .....	45
Figura 42 - Características do Arduino .....	45
Figura 43 - Protoboard e Jumpers .....	46

Figura 44 - Resistores .....	47
Figura 45 - Servo Motor .....	48
Figura 46 – Exemplo código do Servo Motor .....	48
Figura 47 - Sensor de Luminosidade .....	49
Figura 48 - Exemplo código Sensor de Luminosidade .....	49
Figura 49 - Teclado de Membrana .....	50
Figura 50 - Exemplo código do Teclado de Membrana (1) .....	50
Figura 51 - Exemplo código do Teclado de Membrana (2) .....	51
Figura 52 - Módulo Bluetooth HC-06 .....	51
Figura 53 - Exemplo código do Módulo Bluetooth HC-06 (1) .....	51
Figura 54 - Exemplo código do Módulo Bluetooth HC-06 (2) .....	52
Figura 55 - Diagrama de Caso de Uso fourhouse .....	53
Figura 56 - Diagrama de Atividade Conversar com Chatbot (1) .....	53
Figura 57 - Diagrama de Atividade Conversar com Chatbot (2) .....	54
Figura 58 - Diagrama de Atividade Conversar com chatbot (3) .....	55
Figura 59 - Diagrama de Atividade conversar com Chatbot (4) .....	55
Figura 60 - Diagrama de Atividades Editar conta do Cliente .....	56
Figura 61 - Diagrama de Atividades Excluir Conta do Atendente .....	56
Figura 62 - Diagrama de atividades Fazer Cadastro .....	57
Figura 63 - Diagrama de Atividades Login .....	58
Figura 64 - Diagrama de Atividades Agendamento .....	58
Figura 65 - Diagrama de Atividades fourhouse .....	59
Figura 66 - Diagrama de Sequência Conversar com Chatbot (1) .....	60
Figura 67 - Diagrama de Sequência Conversar com Chatbot (2) .....	61
Figura 68 - Diagrama de Sequência Conversar com Chatbot (3) .....	61
Figura 69 - Diagrama de Sequência Editar Conta do Cliente .....	62
Figura 70 - Diagrama de Sequência Excluir Conta do Atendente .....	63
Figura 71 - Diagrama de Sequência Fazer Cadastro .....	63
Figura 72 - Diagrama de Sequência Login .....	64
Figura 73 - Diagrama de Sequência Agendamento .....	64
Figura 74 - Diagrama de Relacionamento de Entidades fourhouse .....	65
Figura 75 - Tela login fourhouse .....	65
Figura 76 - Tela inicial fourhouse (1) .....	66
Figura 77 - Tela inicial fourhouse (2) .....	67
Figura 78 - Tela inicial fourhouse (3) .....	67
Figura 79 - Tela inicial fourhouse (4) .....	68
Figura 80 - Tela inicial fourhouse (5) .....	68
Figura 81 - Tela de perfil do usuário .....	69
Figura 82 - Tela Serviços contratados .....	69
Figura 83 - Tela de cadastro fourhouse (1) .....	70
Figura 84 - Tela de cadastro fourhouse (2) .....	70
Figura 85 - Tela de erro do login .....	71
Figura 86 - Tela de Login Atendente .....	72
Figura 87 - Tela Dashboard fourhouse .....	72
Figura 88 - Tela Edição de Usuários fourhouse .....	73
Figura 89 - Tela Deletar Usuários fourhouse .....	73

Figura 90 - Tela Deletar Atendente fourhouse .....	74
Figura 91 - Tela de agendamento .....	74
Figura 92 - Rodapé fourhouse .....	75
Figura 93 – Maquete fourhouse (1) .....	76
Figura 94 - Maquete fourhouse (2).....	76
Figura 95 - Lateral maquete .....	76
Figura 96 – Maquete fourhouse de noite (1) .....	77
Figura 97 - Maquete fourhouse de noite (2) .....	77
Figura 98 - Maquete de noite fourhouse (3) .....	78
Figura 99 - Maquete fourhouse de noite (4) .....	78
Figura 100 - Maquete fourhouse de noite (5) .....	79



## **Lista de Abreviações**

*American National Standards Institute (ANSI)*

*Banco de Dados (BG)*

*Cascading Style Sheet (CSS)*

*Diagrama Entidade Relacionamento (DER)*

*European Council for Nuclear Research (CERN)*

*HyperText Markup Language (HTML)*

*Hypertext Preprocessor (PHP)*

*International Standard Organization (ISO)*

*JavaScript (JS)*

*Model Entity Relationship (MER)*

*Object Modeling Technique (OMT)*

*Object-Oriented Software Engineering –(OOSE)*

*Personal Computers (PCs)*

*Personal Home Page Tools / Form Interpreter (PHP/FI)*

*Structured Query Language (SQL)*

*UCS Transformation Format 8 (UTF-8)*

*Unified Modeling Language (UML)*

*Uniform Resource Locator (URL)*

*World Wide Web Consortium (W3C)*

# SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO .....</b>	<b>11</b>
<b>2</b>	<b>REFERENCIAL TEÓRICO .....</b>	<b>13</b>
2.1	HTML.....	13
2.2	CSS .....	16
2.3	JavaScript .....	18
2.4	PHP .....	25
2.5	Framework .....	30
2.6	Bootstrap .....	31
2.7	Banco de Dados .....	33
2.8	UML .....	36
2.8.1	Análise e Levantamento de Requisitos .....	37
2.8.2	Diagrama de Casos de Uso .....	38
2.8.3	Diagrama de Sequência .....	39
2.8.4	Diagrama de Atividades .....	41
2.8.5	Diagrama de Classe.....	43
2.9	Arduino .....	44
<b>3</b>	<b>DESENVOLVIMENTO.....</b>	<b>53</b>
3.1	Diagrama de Casos de Uso .....	53
3.2	Diagrama de Atividades .....	53
3.3	Diagrama de Classe .....	59
3.4	Diagrama de Sequência.....	60
3.5	Diagrama de Relacionamento de Entidades .....	65
3.6	Sistema Web.....	65
3.7	Aplicativo Móvel.....	75
3.8	Maquete.....	75
<b>4</b>	<b>CONCLUSÃO.....</b>	<b>79</b>
<b>5</b>	<b>REFERÊNCIAS BIBLIOGRÁFICAS .....</b>	<b>80</b>

## 1 INTRODUÇÃO

A automação residencial, evoluiu de uma ferramenta de automatização de tarefas para uma junção de várias tecnologias com finalidades de extrema importância para a sociedade. Hoje os sistemas de automação residencial são muito mais profissionais de forma que ficaram mais atraentes ao público, com o aumento da eficiência energética, melhoria da segurança, automação de tarefas diárias, controle remoto, entre outros. Porém, eles possuem um valor alto, fazendo com que o acesso à essa tecnologia na sociedade seja difícil para as classes mais baixas (ORTIZ, 2018).

Durante a construção dos primeiros grandes edifícios nos anos 80, iniciou-se a utilização da Domótica, logo após foi possível notar a necessidade de controlar e interligar as funções prediais aplicando também nas residências. A Automação Residencial é a aplicação de sistemas de controle baseados na automação de processos para as funções encontradas no ambiente residencial, integrando os seus acionamentos, tornando a habitação funcional, inteligente e segura, visando a praticidade, a simplicidade e realizando controle de meio remoto, é possível utilizar dessas tecnologias estando dentro ou fora da residência pela internet (MENDES, 2020).

Pensando em proporcionar conforto, economia e principalmente segurança, a ideia de automatizar uma residência define-se em facilitar diversas ações realizadas diariamente, auxiliando todas as pessoas independente de suas dificuldades, como aquelas que possuem deficiência, que não conseguem se locomover para abrir uma janela. Para que isso ocorra são necessários alguns equipamentos, de preferência de baixo custo, como placas de Arduino, que são interligadas aos dispositivos automatizados da residência. São muitos os dispositivos que podem ser instalados em uma residência, por exemplo luzes, condicionador de ar, portão eletrônico, sensor e alarme (TÓFOLI, 2014).

Este trabalho tem como objetivo principal mostrar que é possível utilizar da Automação Residencial para trazer mais conforto, segurança e racionalização de energia na vida da sociedade de maneira prática e barata, para que não só quem possui mais dinheiro tenha acesso à essa tecnologia.

Como objetivo específico este projeto visa:

- Demonstrar conhecimento em grande parte das matérias do curso e nos conteúdos apresentados e colocá-los em prática.

Para o desenvolvimento deste projeto, será criado um aplicativo web, de uma empresa prestadora de serviços fictícia, que proporcionará ao cliente uma interface para a escolha do serviço desejado. Como forma de demonstrar o conhecimento em maior parte das matérias do componente curricular do curso, será feita uma maquete, que representará uma casa, possuindo todos os serviços prestados pela empresa, será utilizado placas de Arduino e seus componentes para deixar a maquete com funcionalidades da Automação Residencial.

Com o avanço da tecnologia, o mundo moderno vem sofrendo grandes mudanças impactando diretamente na vida de todas as pessoas, as habitações atraíram um forte interesse das comunidades técnicas e científicas, se utilizando da Domótica com o foco no emprego de uma nova ciência para promover conforto, qualidade de vida e bem-estar social. Domótica é a ciência que tem como objetivo fazer a gestão de todos os recursos habitacionais, ela consiste na automação doméstica das habitações (casa, escritório, residência, entre outros), se utilizando da junção de muitas especialidades, como eletricidade, mecânica, telecomunicações e informática, para dar mais qualidade de vida para seus moradores e usuários, gerando conforto, segurança, lazer, comunicação e racionalização de energia (DOMINGUES, 2013).

Portanto, da mesma forma que ocorreu uma revolução na vida das pessoas com o surgimento dos PCs (Personal Computers), é possível que também ocorra uma revolução com o advento da Domótica, fazendo com que a Automação Residencial se torne indispensável aos padrões de qualidade de vida atual (MOURA; CUNHA, 2021).

## 2 REFERENCIAL TEÓRICO

Este capítulo contém o embasamento teórico das tecnologias que serão utilizadas para a elaboração do aplicativo web da empresa FourHouse e da maquete mostrando a Automação Residencial.

### 2.1 HTML

*HyperText Markup Language* (HTML) também chamada de Linguagem de Marcação de Hipertexto foi criada em 1991 por Tim Berners-Lee, no CERN (European Council for Nuclear Research) na suíça, é importante deixar claro que o HTML não é uma linguagem de programação, mas sim, como o próprio nome já diz, é uma linguagem de marcação de hipertexto, sendo a linguagem mais utilizada para o desenvolvimento web, ela permite a criação de documentos estruturados em títulos, parágrafos, listas, links, tabelas, formulários e outros elementos que podem ser necessários no desenvolvimento de uma página web e possibilita a inclusão de imagens e vídeos nesses elementos, formando assim, o conceito de hipertexto, um conjunto de elementos conectados (Flatschart, 2011).

A sintaxe para estruturação das informações na linguagem HTML é realizada por meio de tags, que são delimitadas pelos sinais "<>" e "</>", onde identificam o elemento e seu conteúdo, assim o navegador do usuário consegue entender como ele deve disponibilizar as informações na página da forma que o desenvolvedor escolheu (TORRES, V. M., 2018). A tag p, por exemplo, informa ao navegador que a sua informação é um parágrafo, já a tag h1 informa o navegador que a informação dentro dela é o título principal da página (FERREIRA; EIS, 2011).

A estrutura básica do HTML é montada conforme a figura 1:

Figura 1 – Estrutura Básica do HTML

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <title></title>
6      <link rel="stylesheet" href="style.css">
7  </head>
8  <body>
9
10 </body>
11 </html>
```

Fonte: autoria própria, 2022.

De acordo com Silva, Gonzaga, Rocha e Lucas (2018), esse código funciona da seguinte forma:

- Todo esse código está na linguagem HTML, logo na primeira linha está o doctype, ele não é uma tag do HTML, mas sim uma instrução para o navegador saber em qual versão da linguagem está escrito o código;
- Na segunda linha é a tag HTML, composto por vários elementos que são uns filhos dos outros, nessa tag é indicada qual a língua principal do documento por meio do atributo lang, nesse exemplo a língua principal é “pt-br”, ou seja, o português brasileiro;
- Logo abaixo, na linha três, tem a tag head que indica a “cabeça” do documento (a parte inteligente do código), nela são inseridos todos os metadados (informações sobre a página e seu conteúdo) necessários;
- Como filha da tag head, a metatag meta está ali para informar ao navegador em qual tabela de caracteres a página foi codificada e como ele deve renderizar as informações de texto, nesse exemplo por meio do atributo charset o navegador entenderá que o padrão definido nesse código é o “UTF-8”, a codificação de caracteres mais comum da World Wide Web, pois utiliza uma codificação multibyte;
- Mais uma tag filha da tag head é a tag title, onde o seu conteúdo define o título do documento;
- A última filha da tag head nesse exemplo, é a tag link que mostra relacionamentos entre o código atual e um recurso externo, nesse exemplo o atributo rel indica que essa tag possui relação com um arquivo de folha de estilo (“style sheet”) e o atributo href mostra onde o navegador deve pegar as informações de estilo;
- Saindo da tag head e entrando na tag body, esta tag identifica o corpo da página, todo conteúdo deve ser inserido dentro dessa tag.

Segue as figuras 2 e 3 como exemplo do uso dessa linguagem e a figura 4 para visualizar o resultado do projeto sendo executado em um navegador.

**Figura 2 – Cadastro de Usuário em HTML**

```

1  <!DOCTYPE html>
2  <html lang="pt-br">
3  <head>
4      <meta charset="UTF-8">
5      <link rel="stylesheet" href="style.css">
6      <title>Cadastro</title>
7  </head>
8  <body>
9
10     <h1>Faça o seu Cadastro</h1>
11
12     <form action="" method="post">
13
14         <label for="">Nome:</label>
15         <input type="text" name="nome" id="" placeholder="Ex: João da Silva">
16         <br/> <br/>
17

```

Fonte: autoria própria, 2022.

Figura 3 - Cadastro de Usuário em HTML (2)

```

17
18     <label for="">CPF:</label>
19     <input type="text" name="cpf" id="" placeholder="123.456.789/00">
20     <br/> <br/>
21
22     <label for="">Telefone:</label>
23     <input type="text" name="tel" id="" placeholder="(11)94002-8922">
24     <br/> <br/>
25
26     <label for="">E-mail:</label>
27     <input type="email" name="email" id="" placeholder="jsilva@gmail.com">
28     <br/> <br/>
29
30     <label for="">Senha:</label>
31     <input type="password" name="senha" id="" placeholder="*****">
32     <br/> <br/>
33
34     <input type="submit" value="Enviar">
35
36 </form>
37
38 </body>
39 </html>

```

Fonte: autoria própria, 2022.

Figura 4 - Página Web Cadastro de Usuário



**Faça o seu Cadastro**

Nome:

CPF:

Telefone:

E-mail:

Senha:

Fonte: autoria própria, 2022.

## 2.2 CSS

Folha de Estilo em Cascata, tradução para *Cascading Style Sheet* (CSS), serve para adicionar estilo na informação que está contida no HTML independente de qual seja essa informação (vídeo, imagem, texto etc.) podendo também editar a informação desejada de várias formas, como cores de fundo, características de fonte, tamanho da fonte, margens, entre muitos outros de maneira simples (SOUZA, 2018).

Atualmente o CSS está em sua terceira versão, onde o CSS 1 criou a estrutura básica e o conceito de seletor, enquanto o CSS 2 adicionou novos recursos nos seletores e a capacidade de alinhar elementos com precisão, no CSS 3 não há padrão, não existe diferença perceptível de sua versão, é uma série de módulos diferentes, em diferentes estágios de preparação, onde no futuro não haverá CSS, mas sim diferentes versões de módulos. (SILVA; GONZAGA; ROCHA; LUCAS, 2018).

A sintaxe do CSS é formada por seletor, propriedade e valor, o seletor representa uma estrutura do HTML que será modificada, a propriedade é a característica do seletor que se deseja modificar e o valor representa a quantificação da modificação desta característica (FERREIRA; EIS, 2018).

A figura 5 exemplifica a sintaxe do CSS:

**Figura 5 – Sintaxe do CSS**

```

seletor{
    propriedade1: valor1;
    propriedade2: valor2;
}

```



Fonte: Autoria própria, 2022.

Onde de acordo com Munzlinger (2011) o seletor define o elemento que sofrerá a formatação, a propriedade é a qual será modificada (cor, tamanho, alinhamento, entre outras) e valor é o conteúdo atribuído à propriedade para modificá-la.

Segue as figuras 6 e 7 como exemplo do uso dessa linguagem e a figura 8 para visualizar o resultado do projeto sendo executado em um navegador.

**Figura 6 - Código do CSS estilizando o HTML**

```
1  body{
2      background-image: linear-gradient(to right, darkblue 10%, black 80%);
3      text-align: center;
4  }
5
6  h1{
7      color: white;
8  }
9
10 .formulario{
11     color: white;
12 }
```

Fonte: Autoria própria, 2022.

De acordo com a *World Wide Web Consortium* (W3C), esse código funciona da seguinte forma:

- Na primeira linha do código indica o seletor escolhido para ser modificado, nesse caso foi a tag body do HTML;
- Logo na segunda linha a propriedade selecionada para alterar o código quer dizer que o fundo (background-image) do seletor terá um gradiente de cores (linear-gradient) da esquerda para a direita (to right), com as cores azul escuro e preto;
- Na terceira linha o código quer dizer que todo o conteúdo dentro do seletor escolhido (a tag body) estará alinhado no centro desse seletor;
- Linha quatro o símbolo “}” significa que as alterações finalizaram ali;
- Na linha seis indica que o seletor escolhido para ser modificado dessa vez foi a tag h1 e a única estilização feita nessa tag está logo abaixo, onde diz que todo o conteúdo desse seletor terá a cor branca;
- Na décima linha indica que o seletor escolhido foi a tag com a classe que possui nome “formulario”, nesse caso é possível saber que não foi literalmente uma

tag escolhida, toda vez que o seletor tiver um ponto antes quer dizer que esse seletor é uma classe. Bastante utilizadas no CSS, as classes servem para facilitar a estilização, pois se um seletor for uma tag HTML, quer dizer que todas as tags que estiverem no documento serão estilizadas daquela forma, com as classes é possível estilizar as tags que tiverem apenas o nome daquela respectiva classe, nesse caso foi necessário adicionar a classe “formulario” dentro da tag form para a alteração funcionar como mostra a imagem 7.

**Figura 7 – Linha do código em HTML mudada para ser estilizado no CSS**

```
12 <form action="" method="post" class="formulario">
```

Fonte: Autoria própria, 2022.

**Figura 8 - Página Web Cadastro de Usuários com CSS**



**Faça o seu Cadastro**

Nome:

CPF:

Telefone:

E-mail:

Senha:

Fonte: Autoria própria, 2022.

## 2.3 JavaScript

Criado em 1995 por Brendan Eich da Netscape, o JavaScript (JS) é uma linguagem de programação interpretada que é utilizada como uma extensão do HTML para o browser Navigator 2.0. Embora ainda seja mantida e estendida pela Netscape, parte da linguagem JavaScript é padrão proposto pela Organização Europeia para Padrões em Comunicações (ECMA), que visa transformá-la em padrão Web. O JS do lado do browser (client-side) tem evoluído e alcançado uma estabilidade razoável como um padrão da Web, é hoje, suportada pelas principais versões de browser que povoam a Web e é a linguagem de programação mais popular do mundo (ROCHA, 1999). Enquanto o HTML é utilizado para organizar as informações e o CSS as estiliza, o JS

é utilizado para implementar elementos dinâmicos na página (GERMANO; ELISEO; SILVEIRA, 2021).

O código em JS é desenvolvido em formato texto, é possível que o texto do código represente instruções organizadas em blocos e comentários, uma instrução em JavaScript consiste em uma série de símbolos, organizados de forma significativa de acordo com as regras da linguagem, que ocupam uma única linha ou terminam em ponto-e-vírgula, dentro de uma instrução pode-se manipular valores de diversos tipos, armazená-los em variáveis e realizar diversas operações com eles. Utilizar ponto-e-vírgula para terminar uma instrução é opcional em JS, pois o interpretador realiza essa tarefa automaticamente, porém é uma boa prática de programação. Com exceção dos caracteres que provocam novas linhas, nenhum outro tipo de caractere que representa espaço em branco interfere no código, o espaço em branco pode e deve ser utilizado para organizar os blocos de código e deixá-los mais legíveis (ROCHA, 1999).

Instruções compostas (sequências de instruções que devem ser tratadas como um grupo) são agrupadas em blocos delimitados por chaves e elas podem ser colocadas em qualquer lugar após a declaração da estrutura que representam, blocos são utilizados no JavaScript para definir funções e estruturas de controle de fluxo, eles são tratados como uma instrução única e podem ser definidos dentro de outros blocos (ROCHA, 1999).

Existem duas formas de incluir comentários em JS, qualquer texto que aparece depois de duas barras (//) o interpretador não o executa até o final da linha, quando o interpretador encontra os caracteres /\* ele ignora tudo o que aparecer pela frente, inclusive caracteres de em outras linhas, até encontrar estes caracteres \*/ (ROCHA, 1999).

Segundo Grillo e Fortes (2008), é possível utilizar variáveis em JS da seguinte forma:

- No JS os números são representados pelo padrão IEEE 754, todos os valores numéricos são declarados pela simples atribuição dos valores a uma variável;
- Já uma variável do tipo booleano pode assumir apenas os valores true e false, os valores deste tipo são usados pela linguagem como resultado de

comparações e podem ser usados pelo usuário para valores de teste ou para atributos que possuam apenas dois estados;

- No caso de uma string, basta colocar uma sequência de caracteres entre aspas simples ou duplas;
- Os Arrays são pares do tipo inteiro-valor para se mapear valores a partir de um índice numérico, em JS os Arrays são objetos com métodos próprios, um objeto do tipo Array serve para se guardar uma coleção de itens em uma única variável, para acessar as variáveis dentro de um array basta usar o nome do array e o índice da variável que se deseja acessar, em JavaScript os arrays podem conter valores de tipos diferentes sem nenhum problema, é possível colocar em um mesmo array inteiros, strings, booleanos e qualquer outro tipo que se desejar;

Segue a imagem 9 para exemplificação dos tópicos acima:

**Figura 9 - Exemplo de utilização básica do JavaScript**

```

38  <script>
39  var a = 1; |
40  //variável com valor inteiro
41
42  var b = 1.5;
43  //variável com valor real
44
45  var c = true;
46  //variável com valor booleano
47
48  var d = 'Feridas se curam com o tempo, não com gaze';
49  //variável com valor de string
50
51  var e = new Array(3);
52  //declarando um Array em JS
53
54  e[0] = "Nenhuma grande descoberta foi feita sem um palpite ousado";
55  e[1] = "Isaac Newton";
56  //inserindo valores do tipo string no primeiro e no segundo item do Array
57
58  e[2] = 0;
59  //inserindo um valor numérico no último item do Array
60
61  </script>

```

Fonte: Autoria própria, 2022.

Ainda segundo Grillo e Fortes (2008), é possível utilizar operadores aritméticos, de comparação e lógicos, também é possível utilizar as estruturas de controle, como if e else, while e outros, segue as imagens 10, 11, 12, 13 e 14 para exemplificação:

**Figura 10 - Operadores Aritméticos**

Operador	Operação	Exemplo
+	Adição	<u>x+y</u>
-	Subtração	x-y
*	Multiplicação	x*y
/	Divisão	x/y
%	Módulo (resto da divisão inteira)	<u>x%y</u>
-	Inversão de sinal	<u>-x</u>
++	Incremento	x++ ou ++x
--	Decremento	x-- ou --x

Fonte: Autoria própria, 2022.

**Figura 11 - Operadores de Comparação**

Operador	Operação	Exemplo
==	Igual a	(x == y)
!=	Diferente de	( <u>x</u> != y)
===	Idêntico a (igual e do mesmo tipo)	(x === y)
!==	Não Idêntico a	( <u>x</u> !== y)
>	Maior que	(x > y)
>=	Maior ou igual a	(x >= y)
<	Menor que	(x < y)
<=	Menor ou igual a	(x <= y)

Fonte: Autoria própria, 2022.

**Figura 12 - Operadores Lógicos**

Operador	Operação	Exemplo
&&	E Lógico	(x && y)
	OU Lógico	(x    y)
!	Negação Lógica	<u>!x</u>

Fonte: Autoria própria, 2022.

**Figura 13 - Exemplo if e else**

```

var a = 1;

if(a < 30){
    alert("O valor dessa variável é menor que 30");
}else{
    alert("O valor dessa variável é maior ou igual a 30");
}

```

Fonte: Autoria própria, 2022.

O if é utilizado quando o desenvolvedor deseja verificar se determinada expressão é verdadeira ou não e executar comandos específicos se a resposta for verdadeira ou falsa, logo se a expressão for avaliada como verdadeira, o primeiro bloco de comandos é executado, ou se ela for avaliada como falsa, o bloco de comandos que segue o else será executado (Nesse caso o bloco executado seria o primeiro bloco);

**Figura 14 - Exemplo while**

```

while(b<5.5){
    b++;
    alert("O valor dessa varável é: "+b);
}

```

Fonte: Autoria própria, 2022.

While é usado quando se deseja que seja executado um bloco de instruções apenas no caso de a expressão da condição for válida, assim, primeiro a expressão é testada e depois o conteúdo do while deve ser executado ou não.

Segundo Rocha (1999) Há três formas de incluir JS em uma página Web, dentro de blocos HTML utilizando a tag script, em um arquivo externo e dentro de descritores HTML sensíveis a eventos (links, botões e componentes de entradas de dados), as três formas podem ser usadas em uma mesma página, como exemplificado abaixo:

**Figura 15 - JS incluído no HTML com a tag script**

```

38      <script>
39          function enviar(){
40              alert("Usuário cadastrado com sucesso");
41          }
42      </script>

```

Fonte: Autoria própria, 2022.

**Figura 16 - JS importado pelo HTML de um arquivo externo (1)**

```
<script src="script.js"></script>
```

Fonte: Autoria própria, 2022.

**Figura 17 - JS importado pelo HTML de um arquivo externo (2)**

```
JS script.js > ...
1 function enviar(){
2     alert("Usuário cadastrado com sucesso");
3 }
```

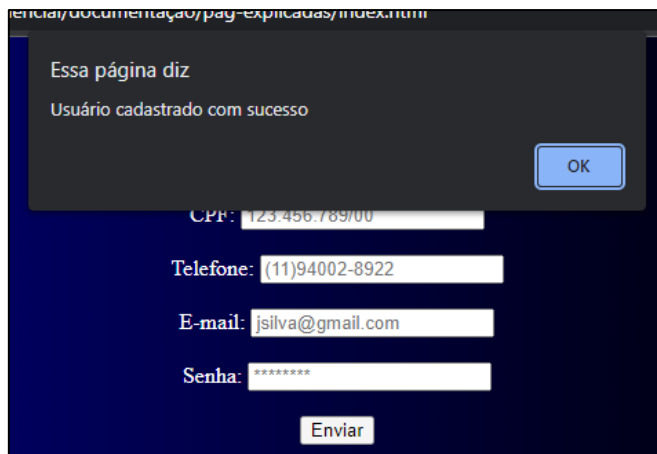
Fonte: Autoria própria, 2022.

**Figura 18 - JS incluído em descritores HTML sensíveis a eventos**

```
<input type="submit" value="Enviar" onclick="alert('Usuário cadastrado com sucesso')">
```

Fonte: Autoria própria, 2022.

**Figura 19 - Resultado dos códigos das imagens 15, 16, 17 e 18 após clicar no botão enviar**



Fonte: Autoria própria, 2022.

Nesses exemplos o atributo HTML onclick foi criado como extensão para dar suporte ao evento de clicar o botão, o código JavaScript que está entre aspas duplas do atributo onclick será interpretado quando o usuário apertar o botão com o mouse. A instrução alert cria uma janela de alerta com a mensagem passada como parâmetro (entre parênteses e aspas) e independente da forma que o JS está incluído no HTML, será exibida a mesma resposta (ROCHA, 1999).

As imagens X, X e X exemplificam como utilizar o JS em uma aplicação web:

**Figura 20 - Adicionando JS na página de Cadastro**

```

<label for="">CPF:</label>
<input type="text" name="cpf" id="cpf" placeholder="123.456.789/00">
<br/> <br/>

<label for="">Telefone:</label>
<input type="text" name="tel" id="tel" placeholder="(11)94002-8922">
<br/> <br/>

<label for="">E-mail:</label>
<input type="email" name="email" id="" placeholder="jsilva@gmail.com">
<br/> <br/>

<label for="">Senha:</label>
<input type="password" name="senha" id="" placeholder="*****">
<br/> <br/>

<input type="submit" value="Enviar">

</form>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js"
  integrity="sha512-894YE6QWD5I59HgZOGReFYm4dnWc1Qt5NtYSaNcOP+u1T9qYdvdihz0PPSiiqn+/+3e7Jo4EaG7TubfWGUrMQ=="
  crossorigin="anonymous" referrerpolicy="no-referrer"></script>
<script src="jquery.mask.js"></script>
<script>
$(document).ready(function () {
  $('#cpf').mask('000.000.000/00');
  $('#tel').mask('(00) 00000-0000');
});
</script>

```

Fonte: Autoria própria, 2022.

**Figura 21 - Resultado da Página de Cadastro com JS**



**Faça o seu Cadastro**

Nome:

CPF:

Telefone:

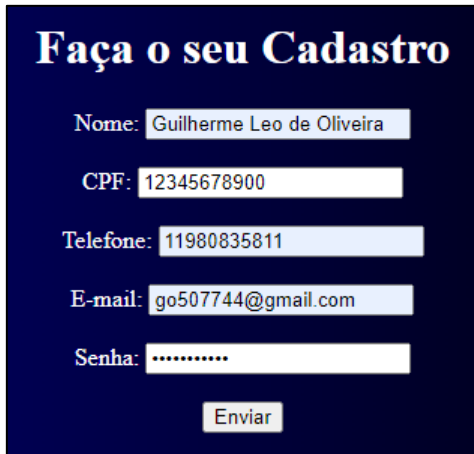
E-mail:

Senha:

Fonte: Autoria própria, 2022.

**Figura 22 - Página de Cadastro sem JS**





**Faça o seu Cadastro**

Nome:

CPF:

Telefone:

E-mail:

Senha:

Fonte: Autoria própria, 2022.

## 2.4 PHP

PHP foi criado em 1994 por Rasmus Lerdof, a primeira versão ficou disponível no início de 1995 e foi conhecida como Personal Home Page Tools. O analisador foi escrito em 1995 e foi chamado de Personal Home Page Tools/Form Interpreter (PHP/FI) versão 2, Rasmus combinou os scripts e adicionou suporte a MySQL. PHP/FI cresceu e as pessoas começaram a contribuir com o seu código. Na metade de 1997 o número de usuários cresceu para mais de 50.000 e nesta época ocorreram mudanças no desenvolvimento do PHP, o analisador foi reescrito por Zeev Suraski e Andi Gutmans e o novo analisador deles formou a base do PHP versão 3 (GONZAGA; BIRCKAN, 2000).

*Hypertext Preprocessor* (PHP) é uma linguagem de programação interpretada e de ampla utilização, que é especialmente interessante para o desenvolvimento web e pode ser incluído no código HTML. A sintaxe da linguagem lembra C, Java e Perl. A linguagem tem como objetivo principal permitir que os desenvolvedores escrevam páginas que serão geradas dinamicamente e rapidamente, mas é possível fazer muito mais com PHP. As páginas em PHP contêm HTML em código mesclado que realiza uma tarefa escolhida pelo desenvolvedor, o código é delimitado pelas tags de início (`<?php`) e fim (`?>`) que permitem que você entre e saia do "modo PHP". O que distingue o PHP de algo como o JS no lado do cliente é que o código é executado no servidor, gerando o HTML que é então enviado para o navegador. O navegador recebe os resultados da execução desse script, mas não sabe qual era o código fonte. Você pode inclusive configurar seu servidor web para processar todos os seus arquivos HTML com o PHP,

e então não há como os usuários diferenciarem o que está sendo rodado no momento de execução (PHP Documentation Group, 2022).

A sintaxe dos operadores aritméticos, de comparação e lógicos e das estruturas de controle em PHP é idêntica à do JavaScript, a forma de tratamento das variáveis também, a única diferença é a forma de declaração de variáveis, enquanto no JS é colocado a palavra var antes da variável, em PHP coloca-se o símbolo \$ (cifrão) antes da variável. A imagem 20 exemplifica isso.

**Figura 23 - Declaração de Variável em PHP**

```
1  <?php
2      $servername;
3  ?>
```

Fonte: Autoria própria, 2022.

Segue o exemplo de como utilizar o PHP em uma aplicação web:

**Figura 24 – Cadastro de Usuário com HTML, CSS, JS e PHP (1)**

```
index.php > html > body
1  <?php
2      $servername = "localhost";
3      $username = "root";
4      $pass = "";
5      $dbname = "monografia";
6
7      $conn = new mysqli($servername, $username, $pass, $dbname);
8
9      if ($conn->connect_error) {
10
11          die("Falha na conexão " . $conn->connect_error);
12
13      }
14
15      $sql = "SELECT * FROM cadastros";
16
17      $result = $conn->query($sql);
18
19  ?>
20
21  <!DOCTYPE html>
22  <html Lang="pt-br">
23  <head>
24      <meta charset="UTF-8">
25      <link rel="stylesheet" href="style.css">
26      <title>Cadastro</title>
27  </head>
```

Fonte: Autoria própria, 2022.

**Figura 25 - Cadastro de Usuário com HTML, CSS, JS e PHP (2)**

```
<body>

<h1>Faça o seu Cadastro</h1>

<form action="entrada.php" method="post" class="formulario">

    <label for="">Nome:</label>
    <input type="text" name="nome" id="" placeholder="Ex: João da Silva">
    <br/> <br/>

    <label for="">CPF:</label>
    <input type="text" name="cpf" id="cpf" placeholder="123.456.789/00">
    <br/> <br/>

    <label for="">Telefone:</label>
    <input type="text" name="tel" id="tel" placeholder="(11)94002-8922">
    <br/> <br/>

    <label for="">E-mail:</label>
    <input type="email" name="email" id="" placeholder="jsilva@gmail.com">
    <br/> <br/>

    <label for="">Senha:</label>
    <input type="password" name="senha" id="" placeholder="*****">
    <br/> <br/>

    <input type="submit" value="Enviar">

</form>
```

Fonte: Autoria própria, 2022.

**Figura 26 - Cadastro de Usuário com HTML, CSS, JS e PHP (3)**

```

<br><br><br>
<h2>Cadastros</h2>

<table border="1">

  <tr class="">
    <td>NOME</td>
    <td>EMAIL</td>
    <td>CPF</td>
  </tr>

  <?php
    $i=0;
    while($row = mysqli_fetch_array($result)) {

  ?>

  <tr>
    <td><?php echo $row["nome"]; ?></td>

    <td><?php echo $row["email"]; ?></td>

    <td><?php echo $row["cpf"]; ?></td>

  </tr>

  <?php
    $i++;
  }
  ?>

```

Fonte: Autoria própria, 2022.

**Figura 27 - Figura 26 - Cadastro de Usuário com HTML, CSS, JS e PHP (4)**

```

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js"
  integrity="sha512-894YE6QND5I59HgZOGReFYm4dnWc1Qt5NtvySaNCOP+u1T9qYdvdihz0PPSiiqn+/+3e7Jo4EaG7TubfWGURMQ=="
  crossorigin="anonymous" referrerpolicy="no-referrer"></script>
<script src="jquery.mask.js"></script>
<script>
  $(document).ready(function () {
    $('#cpf').mask('000.000.000/00');
    $('#tel').mask('(00) 00000-0000');
  });
</script>

</body>
</html>

```

Fonte: Autoria própria, 2022.

**Figura 28 - Página Web com HTML, CSS, JS e PHP (1)**

**Faça o seu Cadastro**

Nome:

CPF:

Telefone:

E-mail:

Senha:

**Cadastros**

Fonte: Autoria própria, 2022.

Figura 29 - Página Web com HTML, CSS, JS e PHP (2)

**Faça o seu Cadastro**

Nome:

CPF:

Telefone:

E-mail:

Senha:

**Cadastros**

Fonte: Autoria própria, 2022.

**Figura 30 - Página Web com HTML, CSS, JS e PHP (3)**

**Faça o seu Cadastro**

Nome:

CPF:

Telefone:

E-mail:

Senha:

**Cadastros**

NOME	EMAIL	CPF
Guilherme Leo de Oliveira	go507744@gmail.com	123.456.789/00

Fonte: Autoria própria, 2022.

## 2.5 Framework

Um Framework é o projeto de um conjunto de objetos que colaboram entre si para execução de um conjunto de responsabilidades, ele reusa análise, projeto e código, reutiliza análise pois descreve os tipos de objetos importantes, reutiliza projeto porque contém algoritmos abstratos e descreve a interface que o programador deve implementar e as restrições a serem satisfeitas pela implementação, reutiliza código porque torna mais fácil desenvolver uma biblioteca de componentes compatíveis e porque a implementação de novos componentes pode herdar grande parte de seu código das super-classes abstratas (MALDONADO; BRAGA; GERMANO; MASIERO, 2002).

Framework é um esqueleto de classes, objetos e relacionamentos agrupados para construir aplicações específicas, essas classes podem fazer parte de uma biblioteca de classes ou podem ser específicas da aplicação. Frameworks possibilitam reutilizar não só componentes isolados, como também toda a arquitetura de um domínio específico (COAD, 1992).

Framework pode ser considerado conceitual e/ou de implementação, o framework conceitual é um esquema abstrato utilizado em um domínio específico, uma representação de alto nível que modela os fatos do mundo real, suas propriedades e seus relacionamentos, já um framework de implementação consiste em um conjunto de sistemas, que contém funcionalidades prontas para serem usadas, sem que os

programadores as tenham de reimplementar para cada aplicação (OLIVEIRA; CARDOSO; BRAGA; CAMPOS, 2018).

## **2.6 Bootstrap**

Criado em 2010 pelo designer do Twitter Mark Otto e pelo desenvolvedor Jacob Thornton, o Bootstrap se tornou um dos mais populares frameworks front-end e projetos de código aberto no mundo. Para resolver um problema interno do Twitter, Otto e Thornton criaram o Bootstrap como solução para as inconsistências de código dentro da equipe, pois antes não existia padrão para estrutura de código usada pela equipe, cada engenheiro utilizava sua própria maneira de programar, isso dificultava a junção dos módulos do projeto (BOOTSTRAP, 2022a).

Para SOUZA (2020), o Bootstrap foi criado incentivar o uso de um padrão na estrutura de código, nomenclatura de classes e dentre outras, pelas equipes de engenharia da empresa. A estratégia deu certo, resultando em menos inconsistências e consequentemente maior rapidez nos projetos. Antes de ser uma estrutura de código aberto, o Bootstrap era conhecido como Twitter Blueprint (Bootstrap) e em agosto de 2011, Bootstrap foi lançado publicamente no Github como um projeto de software livre. O seu código fonte utiliza o Sass, que é um pré-processador de CSS.

Na documentação da versão Alpha do Bootstrap 4 foi informado que houve uma migração do Less para Sass para os arquivos CSS de origem. Em relação a responsividade o Bootstrap é muito eficiente, utilizando media queries de CSS, os projetos web se adaptam facilmente entre as telas de celulares, tablets e desktop. Além disso conta com ótimos recursos, sua documentação é muito rica em conteúdo e com um design bonito, dezenas de componentes HTML e CSS personalizados, e plugins jQuery (BOOTSTRAP, 2015).

**Figura 31 - Cadastro de Usuário com HTML, CSS, JS, PHP e Bootstrap (1)**

```

28 <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
    integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">
29
30 </head>
31 <body>
32
33 <nav class="navbar navbar-expand-lg navbar-light bg-light">
34 <a class="navbar-brand" href="#">Navbar</a>
35 <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarTogglerDemo02"
    aria-controls="navbarTogglerDemo02" aria-expanded="false" aria-label="Alterna navegação">
36 <span class="navbar-toggler-icon"></span>
37 </button>
38
39 <div class="collapse navbar-collapse" id="navbarTogglerDemo02">
40 <ul class="navbar-nav mr-auto mt-2 mt-lg-0">
41 <li class="nav-item active">
42 <a class="nav-link" href="#">Home <span class="sr-only">(Página atual)</span></a>
43 </li>
44 <li class="nav-item">
45 <a class="nav-link" href="#">Link</a>
46 </li>
47 <li class="nav-item">
48 <a class="nav-link disabled" href="#">Desativado</a>
49 </li>
50 </ul>
51 <form class="form-inline my-2 my-lg-0">
52 <input class="form-control mr-sm-2" type="search" placeholder="Pesquisar">
53 <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Pesquisar</button>
54 </form>
55 </div>
56 </nav>

```

Fonte: Autoria própria, 2022.

**Figura 32 - Cadastro de Usuário com HTML, CSS, JS, PHP e Bootstrap (2)**

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <link rel="stylesheet" href="style.css">
  <title>Cadastro</title>

  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
    integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO" crossorigin="anonymous">

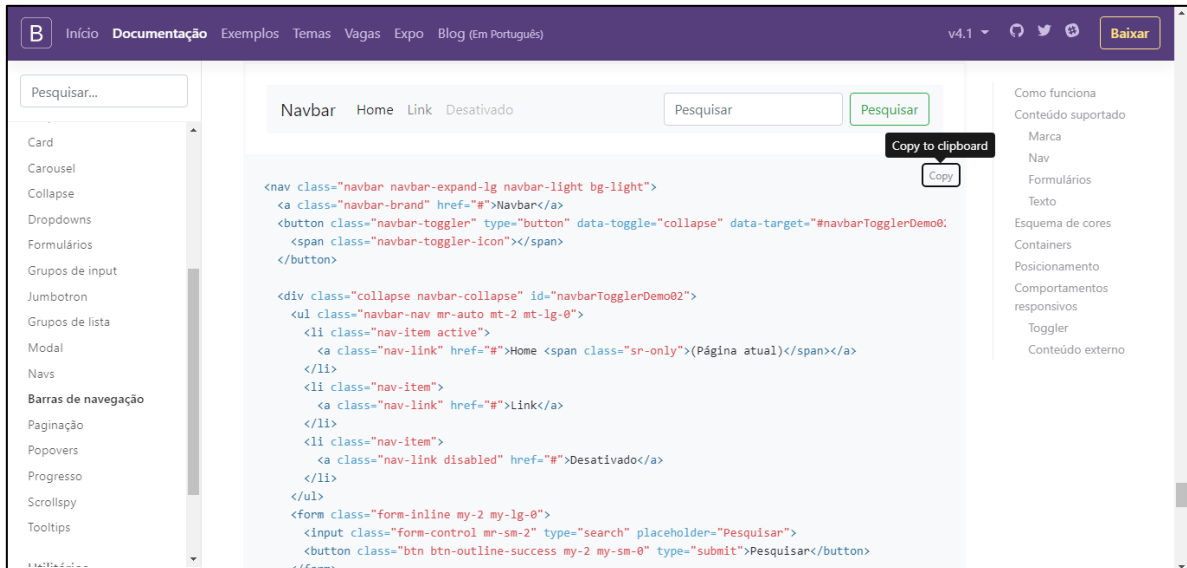
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X
    +965Dz00rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js" integrity="sha384-ZMP7rVo3mIyK
    +2+9J3UJ46jBk0WLaUAdn689aCwoqbBJiSnjAK/18WvCWPiPM49" crossorigin="anonymous"></script>
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js" integrity="sha384-ChfqquxZUCnJSK3
    +MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy" crossorigin="anonymous"></script>
</head>

```

Fonte: Autoria própria, 2022.

**Figura 33 - Como pegar códigos do Bootstrap**





Fonte: <https://getbootstrap.com.br/docs/4.1/components/navbar/>, 2022.

**Figura 34 - Página Web com HTML, CSS, JS, PHP e Bootstrap**

NOME	EMAIL	CPF
Guilherme Leo de Oliveira	go507744@gmail.com	123.456.789/00

Fonte: Autoria própria, 2022.

## 2.7 Banco de Dados

*Databanks*, em português Banco de Dados (BD) é um conjunto fixo de dados que os sistemas usam para uma função específica em um aplicativo específico. Em termos mais simples, um banco de dados é um local onde são armazenadas as informações necessárias ao funcionamento de uma determinada organização, sendo fonte de informações para aplicações que possam surgir para facilitar a produção dessa organização (SILVA; GONZAGA; ROCHA; LUCAS, 2018).

O crescimento do armazenamento virtual explodiu desde os anos 2000, fazendo com que muitos desenvolvedores se preocupassem com a falta de armazenamento. Dados da IBM mostram que em 2015, 4,5 quintilhões de bytes de dados são criados todos os dias a partir de várias fontes de dados. Isso se deve à crescente conexão de negócios como redes sociais, operadoras de telefonia, serviços de streaming à Internet (RABELO; CÂNDIDO, 2017).

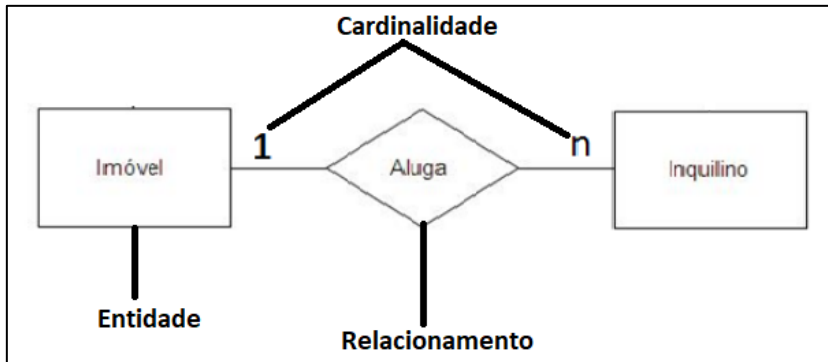
O modelo de banco de dados relacional é baseado no princípio de que os dados são armazenados em tabelas, o modelo relacional representa o banco de dados como uma coleção de relacionamentos. Informalmente, cada relação é semelhante a uma matriz de valores, esse modelo ainda é muito utilizado porque oferece fácil acesso aos dados e permite que os usuários usem uma ampla variedade de abordagens de processamento de dados, além da possibilidade de usar sistemas de gerenciamento de banco de dados que executam comandos na linguagem *Structured Query Language* (SQL) e é responsável por organizar a gestão, processamento e uso de dados, especialmente em relação à segurança (RABELO; CÂNDIDO, 2017).

O modelo conceitual serve para preparar o banco de dados independente do Sistema de Gerenciamento de Banco de Dados (SGBD), definindo como os dados são organizados no banco de dados sem priorizar sua implementação. Nesta etapa, os conceitos de *Model Entity Relationship* (MER - Modelo Entidade Relacionamento) são aplicados aos requisitos funcionais do sistema e servem de referência para o desenho da estrutura de banco de dados que suporta a operação. Para melhor compreensão e visualização pelos projetistas de banco de dados, o MER é representado por um Diagrama Entidade Relacionamento (DER) (RIBEIRO; TIOSSO; PETRUCOLI, 2019).

Bazzi (2013) o DER da Figura 35 da seguinte forma:

- **Entidade:** Corresponde a tudo aquilo que se deseja guardar dados, podendo ser concreto ou abstrato, e que é composta pelas características ou atributos que deverão ser armazenados no banco de dados;
- **Relacionamento:** Corresponde à representação que indica qual é a relação entre uma entidade e outra;
- **Cardinalidade:** Corresponde ao grau de relação entre duas entidades.

Figura 35 - Exemplo DER



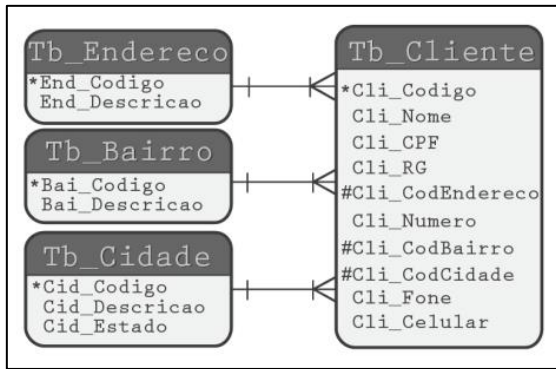
Fonte: Adaptada de <https://www.devmedia.com.br/mer-e-der-modelagem-de-bancos-de-dados/14332>, 2022.

Nesse exemplo é possível entender que “muitos inquilinos alugam um imóvel”, ou também, “um imóvel é alugado por vários inquilinos”, essa é a mensagem que o DER da figura 35 quer passar.

Bazzi (2013) explica que o MER contém os seguintes atributos:

- **Chave Primária:** corresponde ao atributo da entidade (coluna da tabela) que representa de forma única um registro;
- **Chave Estrangeira:** é aquela utilizada para realizar o relacionamento entre entidades;
- **Dados Alfanuméricos:** (VARCHAR, NVARCHAR ou CHAR) são os mais comumente utilizados, pois além de permitirem o armazenamento de dados referentes a caracteres alfanuméricos, incluindo números, acentuação e pontuação;
- **Dados Numéricos:** (NUMBER, FLOAT, INTEGER) para armazenamento de dados numéricos;
- **Dados tipo Data:** (DATE, TIME, TIMESTAMP) para tratar de dados referentes a datas.

**Figura 36 - Exemplo MER**



Fonte: Bazzi (2013).

No exemplo da Figura 36 é possível identificar as tabelas cliente, endereço, bairro e cidade, cada uma dessas tabelas possui seus próprios atributos e todas possuem também necessariamente uma única chave primária para cada, é possível identificar as chaves estrangeiras na tabela cliente, onde são elas que fazem a relação da tabela cliente com as outras tabelas.

SQL é atualmente a principal linguagem de processamento de banco de dados relacional, consegue ser uma ferramenta simples, mas muito poderosa, a linguagem SQL representa o padrão mundial de processamento de banco de dados reconhecido pelo *American National Standards Institute* (ANSI - Instituto Nacional Americano de Padrões) e a *International Standard Organization* (ISO - Organização Internacional de Padronização) como uma linguagem de consulta universal (SOUZA; OLIVEIRA, 2019).

O MySQL é um sistema de banco de dados que foi originalmente desenvolvido para uso em pequenas e médias empresas, que era uma realidade porque então seu tamanho era menor do que hoje, superando os anteriores com exceção. É um gerenciador de banco de dados bastante conhecido, pois é utilizado por grandes empresas atuantes no mercado como HP, Bradesco, NASA, SONY, seu crescimento é importante pelo patamar que atingiu no mercado. Sobre o uso do MySQL (SOUZA; OLIVEIRA, 2019).

## 2.8 UML

A Linguagem de Modelagem Unificada ou *Unified Modeling Language* (UML) é uma linguagem visual usada para modelagem de softwares baseados no paradigma de orientação a objetos, ela pode ser aplicada a todos os domínios de aplicação. Sendo a linguagem-padrão de modelagem adotada internacionalmente pela indústria de

engenharia de software, é necessário que fique claro que a UML não é uma linguagem de programação, mas sim uma linguagem de modelagem, cujo o objetivo é ajudar os engenheiros de software a definirem as características do sistema, requisitos, comportamento, estrutura lógica e dinâmica dos processos e até mesmo necessidades físicas em relação ao equipamento sobre o qual o sistema deverá ser implantado (SILVA; GONZAGA; ROCHA; LUCAS, 2018).

A UML surgiu da união dos métodos de Booch, de Jacobson (*Object Modeling Technique* - OMT) e o de Rumbaugh (*Object-Oriented Software Engineering* - OOSE). Até a década de 90, os métodos de modelagem orientada a objetos mais populares entre os profissionais da área de desenvolvimento de software eram esses. A união desses métodos contou com o amplo apoio da Rational Software, que a incentivou e financiou (GUEDES, 2011).

### **2.8.1 Análise e Levantamento de Requisitos**

Para Rosa, Lucca, Lemos, Bernardi e Medina (2017), o desenvolvimento de um software possui três fases: definição, desenvolvimento e manutenção. O Levantamento de Requisitos está logo na primeira fase, tendo responsabilidade por entender o funcionamento do software, como será a experiência do usuário final e como o sistema irá influenciar nos negócios do cliente. O entendimento correto dos requisitos consiste na etapa mais crítica do desenvolvimento de um software, possuindo influência direta na qualidade final do produto. Conforme apontado pelos autores, o levantamento de requisitos viabiliza a definição das restrições e características que o sistema deverá apresentar. No entanto, a sua incorreta realização ainda se apresenta como grande responsável pelo fracasso de projetos, por isso a grande relevância de uma correta execução dele, tal problema pode estar relacionado com a dificuldade destes profissionais em realizar corretamente esta etapa do desenvolvimento, o que demonstra uma falta dos conhecimentos necessários para essa tarefa.

Para Pompilho (1995) uma das razões para o baixo grau de satisfação dos usuários estão na fase de Levantamento de Requisitos do projeto, pois não é utilizada uma técnica adequada para extrair os requisitos do sistema, outra razão é quando o analista não descreve os requisitos claramente, sem ambiguidades, conciso e consistente com todos os aspectos significativos do sistema proposto.

De acordo com Mendonça (2014) as técnicas de levantamento de requisitos possuem um conceito próprio e podem ser utilizadas em conjunto pelo analista, elas são:

- Entrevistas;
- Questionários;
- Brainstorming;
- Joint Application Design (JAD);
- Prototipagem.

Cada uma dessas técnicas possui suas próprias características e existe um contexto específico para usá-las, por isso o analista deve entender bem como o sistema funciona para conseguir aplicar a técnica que mais o ajudará para conseguir informações importantes para o desenvolvimento do software.

### **2.8.2 Diagrama de Casos de Uso**

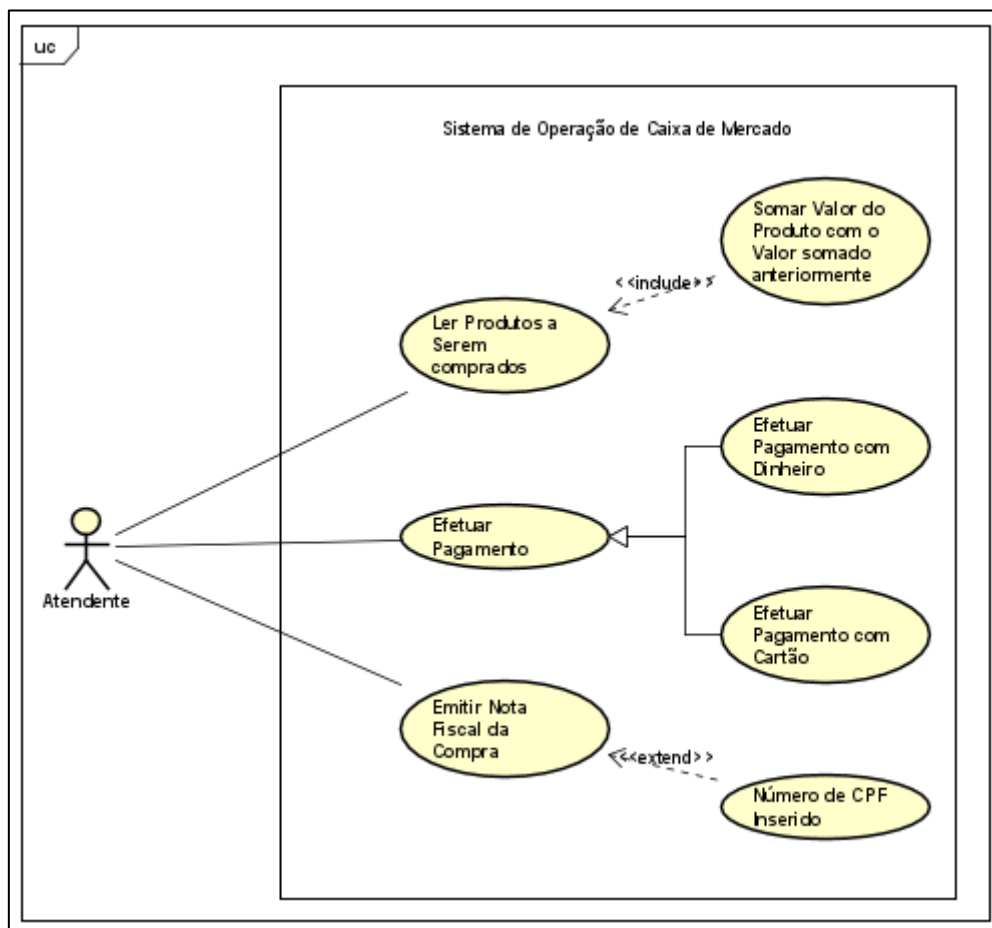
O diagrama de casos de uso é o diagrama mais geral e informal da UML, apresentando uma linguagem simples e de fácil compreensão para que os usuários possam ter uma ideia geral de como o sistema irá se comportar, esse diagrama procura identificar os atores (usuários, outros sistemas ou até mesmo algum hardware especial) que utilizarão de alguma forma o software e as funcionalidades que o sistema disponibilizará aos atores, chamadas de casos de uso (GUEDES, 2011). O Diagrama de Caso de Uso é um instrumento eficiente para determinação e documentação dos serviços que serão desempenhados pelo sistema e um meio para comunicação com os clientes no processo de definição dos requisitos do sistema (JUNIOR, 2020).

No relacionamento entre ator e caso de uso expressa sempre terá comunicação, porque o ator é uma entidade externa, então ele não poderia ter qualquer relacionamento estrutural com o sistema computacional, a notação UML para este tipo de relacionamento é um segmento de reta ligando ator e caso de uso (ROSA; LUCCA; LEMOS; BERNARDI; MEDINA, 2017).

As relações entre casos de uso sempre serão estruturais, sendo elas inclusão, extensão e generalização. Isto porque casos de uso são aplicações completas do sistema e não faz sentido em fazer-se comunicar dois “usos do sistema” (JUNIOR, 2020).

Em relacionamentos de inclusão, quando mais de um caso de uso inclui uma sequência comum de interações, essa sequência comum pode ser descrita em outro caso de uso, a partir disso vários casos de uso do sistema podem incluir o comportamento desse caso de uso comum. No caso de um relacionamento de extensão, ele inclui a especificação de uma condição de extensão, esta condição habilita a extensão, ou seja, indica quando aplicar o relacionamento. Já para os relacionamentos de generalização, um caso de uso mãe representa uma sequência geral de comportamentos, os casos de uso filhos especializam o caso de uso mãe, inserindo etapas adicionais ou detalhando etapas (ROSA; LUCCA; LEMOS; BERNARDI; MEDINA, 2017). Segue o exemplo na imagem 35.

**Figura 37 - Exemplo Diagrama de Caso de Uso**



Fonte: Autoria própria, 2022.

### 2.8.3 Diagrama de Sequência

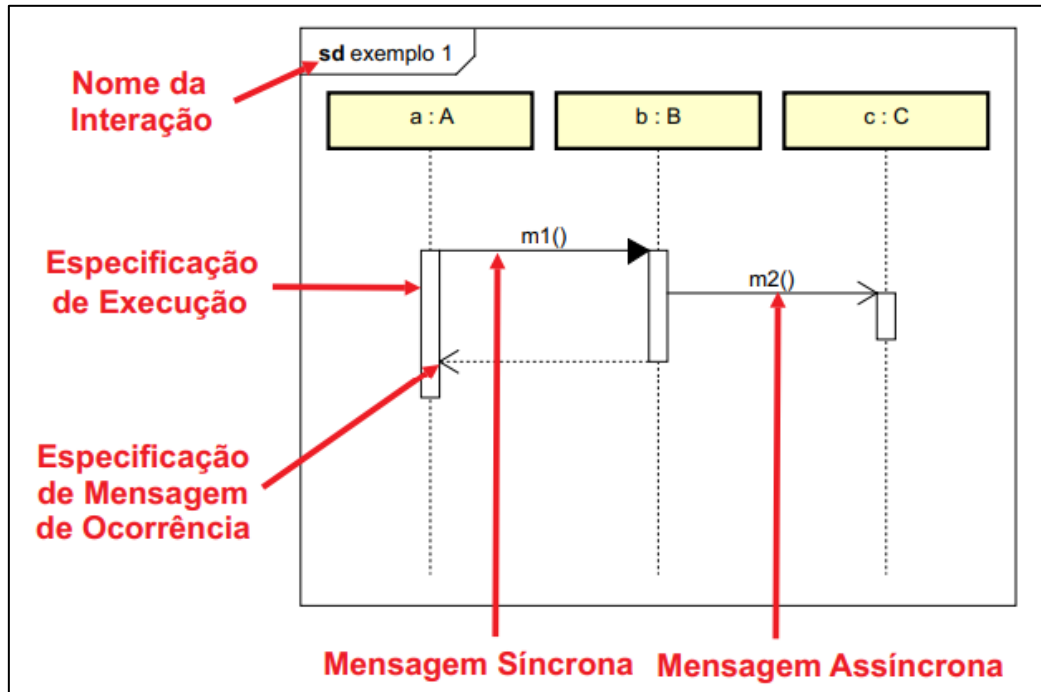
Um Diagrama de Sequência representa mensagens passadas entre objetos no programa em desenvolvimento em ordem de tempo iterativa, com o tempo, essa

distribuição faz com que o diagrama de sequência crie um cenário separado para cada alvo e ator na interação, identificando cada um de seus papéis com setas entre as linhas de vida. O princípio de funcionamento do diagrama de temporização é bidimensional, a informação vertical no diagrama refere-se ao eixo do tempo, e a horizontal, ou diagonal em um determinado momento, refere-se ao objeto de troca de informações, representando a troca de mensagens. objetos, identificando corretamente seus nomes, parâmetros e condições para tais interações. Há também mensagens de retorno, identificadas por uma linha tracejada horizontal (JÚNIOR, 2012).

Os elementos das interações e suas notações podem ser divididas em dois tipos: interação básica e fragmento combinado. A figura 36 apresenta as interações básicas de um Diagrama de Sequência, as linhas de vida representam os participantes da interação que se comunicam por meio de mensagens. Essas mensagens podem ser síncronas ou assíncronas e podem corresponder a chamadas de operação, sinalização ou mensagens de retorno. Se a assinatura da mensagem for uma operação, a operação será realizada de acordo com os parâmetros correspondentes aos seus parâmetros. As mensagens de retorno representam retornos de chamada de operações síncronas. Se a assinatura da mensagem for um sinalizador, a mensagem representará Envio assíncrono de instâncias de sinal. Uma especificação de execução é uma unidade de comportamento ou ação em uma linha de vida que representa quando um objeto está vivo, ou seja, quando ele executa alguma ação. As mensagens enviadas e recebidas são marcadas com a especificação de ocorrência da mensagem (DA ROCHA, 2021).

**Figura 38 - Exemplo Diagrama de Sequência**





Fonte: DA ROCHA (2021).

#### 2.8.4 Diagrama de Atividades

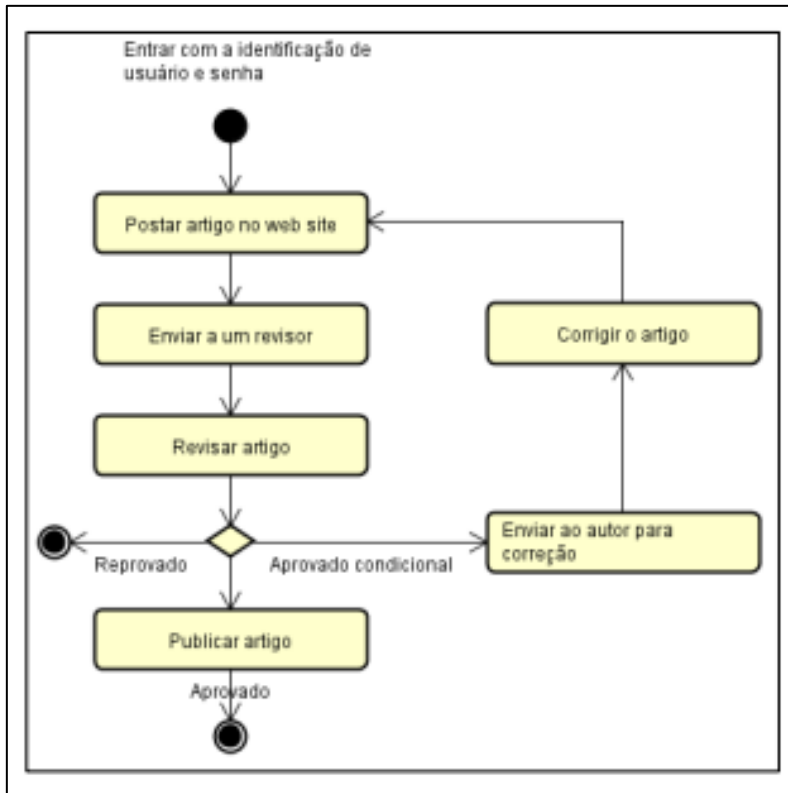
Os Diagramas de Atividades são considerados um caso especial dos antigos diagramas de estado, agora chamados de diagramas de máquina de estado. A partir da UML 2.0, eles são considerados independentes dos diagramas de máquina de estado. Um Diagrama de Atividades tem como foco descrever os passos a serem percorridos para a conclusão de uma determinada atividade, que pode ser expressa por um método, algoritmo ou até mesmo um processo completo com certa complexidade (GUEDES, 2011).

Os diagramas de atividades se concentram em representar o fluxo de controle das atividades. Atividades são comportamentos que simulam execução não atômica, enquanto ações representam processamento atômico. Atividades (Activity) coordenar ações (Action). Essa coordenação é capturada como um grafo, onde os nós (ActivityNode) representam ações e são conectados por arestas (ActivityEdge). O fluxo de dados é representado por nós de objeto (ObjectNode) e bordas de fluxo de dados (ObjectFlow). O controle e os dados fluem ao longo da borda e são manipulados por nós, roteados para outros nós ou para armazenamento temporário. Mais especificamente, os nós de ação processam o controle e os dados recebidos através das arestas do gráfico e fornecem controle e dados para outras ações; os nós de

controle direcionam o controle e os dados através do gráfico; os nós de objeto armazenam temporariamente os dados até serem movidos pelo gráfico. O termo token é usado para se referir a controles e valores de dados que fluem por meio de uma atividade (FERREIRA; MARTINS, 2010).

Segundo Silva, Gonzaga, Rocha e Lucas (2018), o diagrama começa com um círculo preenchido, que é o identificador do início do fluxo, que é o nó fonte. Então a marca retangular é a ação que contém seu nome, que neste caso é enviar um artigo para um site. As ações devem começar com verbos no infinitivo e as ações formam uma ação. Essas funções representam ações do usuário e reações do sistema. Essas funções conectam um segmento de linha com uma seta apontando para a próxima função. Esses segmentos são fluxos de controle que conectam funções. Quando as funções enviam para o revisor e verificam o artigo, existe um nó de decisão, o nó de decisão é um losango, é utilizado em situações em que o usuário tem escolhas diferentes ou reações do sistema a determinadas condições, pode-se dizer que o nó de decisão é a representação das instruções IF e ELSE condicionais usadas em linguagens de programação. Pela Figura 37, identifica-se que o nó de decisão pode levar a três situações diferentes, quando após a leitura de um artigo, o fluxo é rejeitado para um padrão circular preenchido com um envelope oco, quando o artigo é aceito, é publicado e então o fluxo chega ao fim e quando o artigo é parcialmente aceito e precisa de correções para publicação, ele passa por todas as etapas novamente até que a função de validação determine seu destino.

**Figura 39 - Exemplo Diagrama de Atividade**



Fonte: SILVA; GONZAGA; ROCHA; LUCAS (2018).

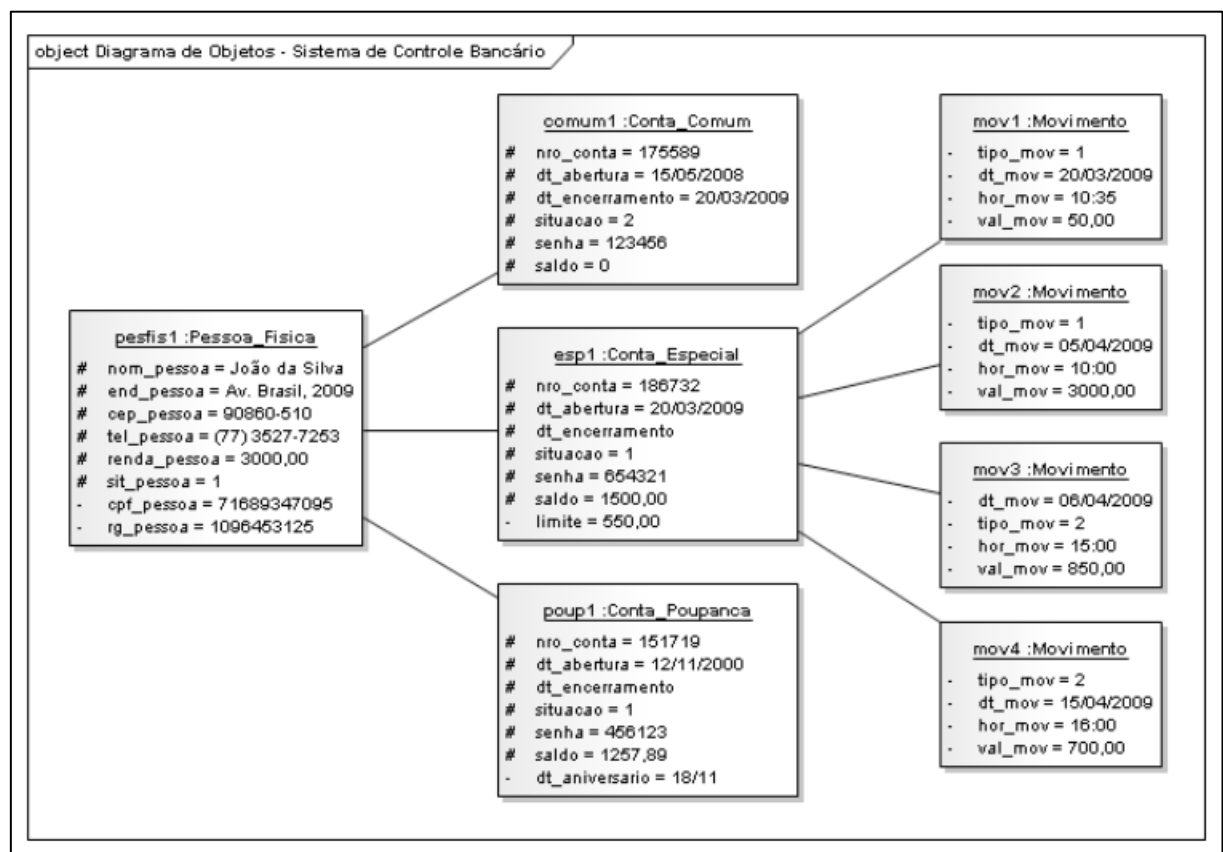
### 2.8.5 Diagrama de Classe

O Diagrama de Classes serve como base para a criação de diversos outros modelos UML, considerando essa dependência, se os modelos de classe forem criados com defeitos ou inconsistências, isso afetará outros diagramas e o próprio código, por isso, para garantir que os diagramas de classe apresentem uma qualidade de alto nível é preciso utilizar técnicas que inspecionem esses diagramas com o intuito de encontrar possíveis erros, assim existem técnicas de inspeção que visam verificar os defeitos e inconsistências contidos nos diagramas de classe, bem como ajudar a reduzir custos, já que os possíveis erros que passariam para os próximos passos seriam identificados e corrigidos nos estágios iniciais do desenvolvimento (SIQUEIRA; PAULON; GUEDES, 2019).

A criação de um modelo de classe é resultado de um processo de abstração a partir do qual são identificados os objetos relacionados ao contexto (entidades e conceitos) que compõem o sistema modelado, o objetivo é descrever as características comuns de propriedades (atributos) e comportamentos (ações), essa descrição geral é chamada de classe, as classes descrevem os objetos juntamente com seus atributos e suas funções comuns (métodos) e cumprem dois propósitos: possibilitam a

compreensão do mundo real na parte essencial para o sistema de informação que está sendo desenvolvido e fornecem uma base prática para a implementação do software. O diagrama de classes também indica seu uso na fase de análise após a coleta e constrói um modelo conceitual das informações necessárias ao software, no modelo conceitual, o engenheiro se preocupa apenas em fornecer as informações necessárias que o software deve ter em termos de classes, seus atributos e os relacionamentos entre as classes, nesta etapa não é necessário modelar as funções que o software pode conter, como os métodos que as classes contêm, pois os métodos já fazem parte do software, somente na fase de projeto é retirado o modelo conceitual do diagrama de classes e produzido o modelo de domínio, já pensando na solução de software (E SOUZA, 2018).

**Figura 40 - Exemplo Diagrama de Classes**



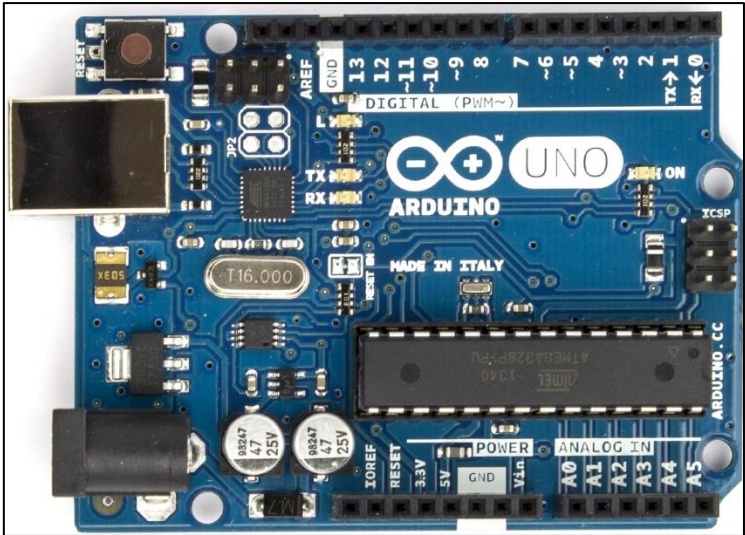
Fonte: SIQUEIRA; PAULON; GUEDES, (2019)

## 2.9 Arduino

O arduíno é uma placa controladora de prototipagem eletrônica com entrada (input) e saída (output), e livre para programação baseada em linguagem C/C++. Para a utilização da placa, é possível fazer projetos que envolvam chaves, LEDs, Servo

motores, sensores, etc. Existem vários tipos de arduino, são divididos em modelos de placas, entre elas, Arduino Uno R3, Uno SMD, Arduino Mega 2560, Nano, Micro, Mini, Arduino DUE, Arduino ADK, além das versões paralelas desenvolvidas por diversos fabricantes ao redor do mundo. A placa arduino uno consiste em 14 pinos digitais, que podem ser usados como Pulse Width Modulation (PWM) para conectar os componentes para fazer qualquer projeto, 6 Inputs analógicos, botão reset para reiniciar a placa, conector USB, ICSP header, Atmega328p que é o microcontrolador, Oscilador de 16MHz e o conector de alimentação externa (ALVES, 2014).

Figura 41 - Placa Arduino



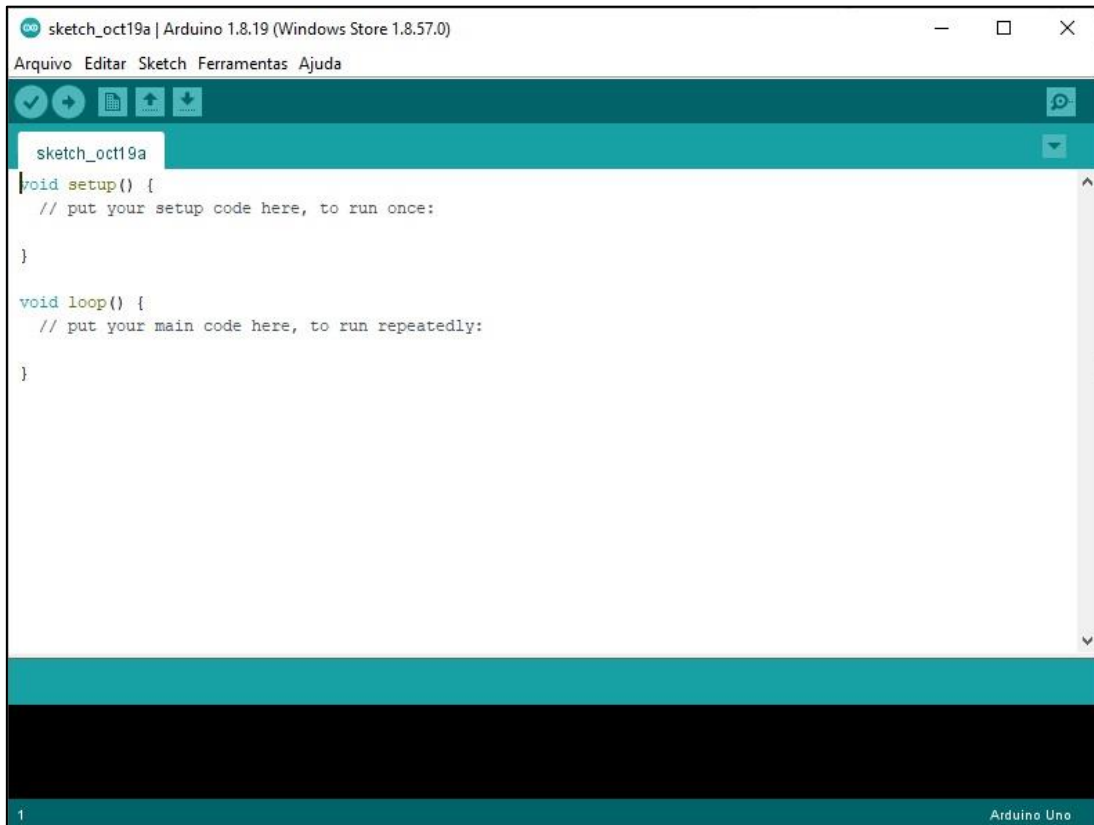
Fonte: <https://blog.fazedores.com/arduino-ino-ino-original-clone-compativel-ou-mera-falsificacao/>, 2022.

Figura 42 - Características do Arduino

Componentes	Detalhes
Microcontrolador	ATMega328
Tensão de operação	5V
Tensão de entrada (recomendado)	7-12V
Pinos digitais de I/O	14 (6 com saída PWM)
Pinos de entrada analógica	6
Corrente por pino de I/O	40 mA
Corrente pelo pino de 3,3V	50 mA
Memória Flash	32 KB (ATmega328)
SRAM	1 KB (ATmega328)
Velocidade do Clock	16 MHz

Fonte: Autoria própria, 2022.

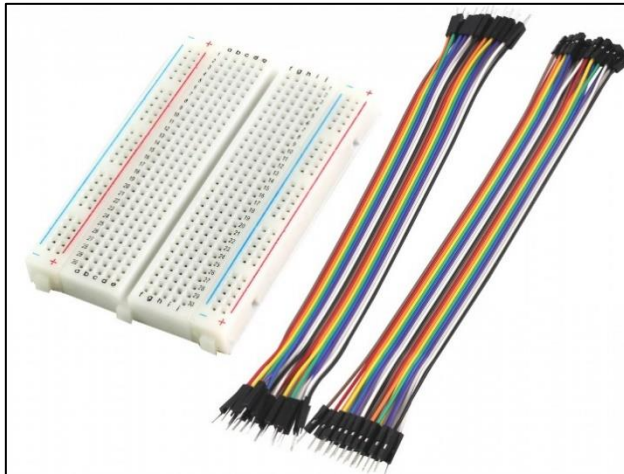
O Arduino IDE é um ambiente usado para fazer os códigos para os projetos desenvolvidos para a placa arduino uno. Esta IDE foi desenvolvida em JAVA (Linguagem de programação orientada a objeto), A linguagem utilizada é C/C++ baseada em wiring (Alves, 2014).



FONTE: Autoria própria, 2022.

A placa protoboard é utilizada em experimentos para circuitos elétricos. É construída por uma base plástica que possui furos e conexões internas, que servem para a alimentação elétrica de todos os componentes que estão conectados na trilha vertical da placa, e para essa conexão, são utilizados cabos jumpers (CONFORTO; CAVEDINI; MIRANDA; CAETANO, 2018).

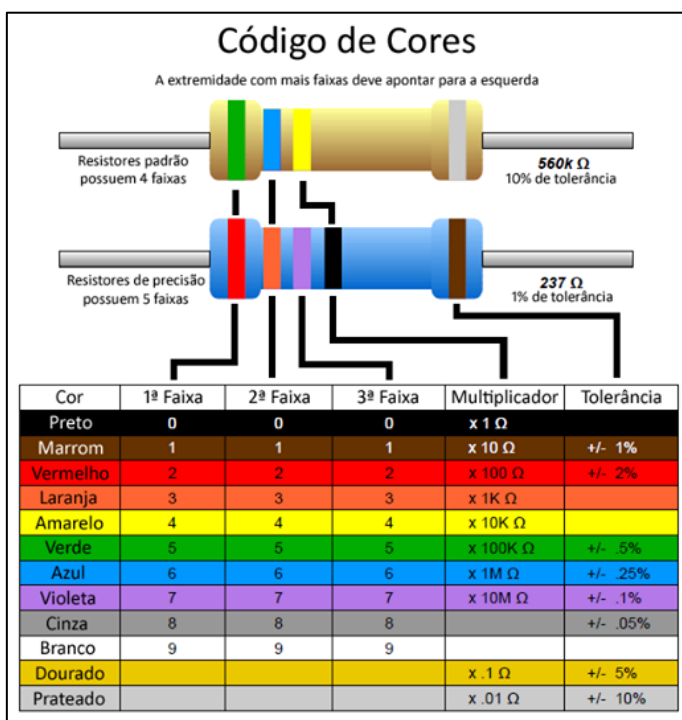
**Figura 43 - Protoboard e Jumpers**



Fonte: [https://www.usinainfo.com.br/1027100-thickbox\\_default/kit-prototipagem-basico-com-protoboard-400-pontos-jumpers-41-pecas.jpg](https://www.usinainfo.com.br/1027100-thickbox_default/kit-prototipagem-basico-com-protoboard-400-pontos-jumpers-41-pecas.jpg), 2022.

O resistor é um componente projetado para provocar resistência a uma corrente elétrica, causando uma queda na voltagem em seus terminais. A maior parte dos resistores é feita de materiais de alta resistência elétrica, e esses são conhecidos como dielétricos. Os que apresentam uma resistência elétrica constante são conhecidos como resistores ôhmicos. Eles podem ser diferentes de acordo com o material com que foram produzidos, além disso, existem resistores que mudam de resistência quando submetidos a diferentes agentes externos. Alguns deles são sensíveis a variações de temperatura e conhecidos como termoresistores. Além deles, há os que respondem a variações na luminosidade, conhecidos como fotoresistores (MCROBERTS, 2011).

**Figura 44 - Resistores**



Fonte: <http://eletronsdadepressao.blogspot.com/2015/01/codigo-de-cores-de-resistores.html>, 2022.

O Servomotor é um atuador eletromecânico, que apresenta movimento proporcional a um comando, como dispositivos de malha fechada, ou seja, recebem um sinal de controle eletrônico que verifica a posição atual para controlar o seu movimento indo para a posição desejada. O eixo dos servos motores possui a liberdade de apenas cerca de 180º graus (360º em alguns modelos) mas são precisos quanto à sua posição e é excelente para controle de posição angular (-90 a 90) possuindo um grau de liberdade de 180 através de Pulse Width Modulation (PWM), é utilizado em robótica e modelismo por serem pequenos e compactos (PIRES, 2020).

**Figura 45 - Servo Motor**



Fonte: <https://blog.fazedores.com/wp-content/uploads/2018/11/DESTAQUE.png>, 2022.

**Figura 46 – Exemplo código do Servo Motor**

```

sketch_dec05a | Arduino 1.8.18
Arquivo Editar Sketch Ferramentas Ajuda

sketch_dec05a$
#include <Servo.h> //biblioteca

Servo servomotor; // Cria um objeto servo

// Variável para armazenar a posição do servol
int pos = 0;

void setup()
{
  // Agrega o objeto servomotor ao pino digital 11
  servo.attach(11);
}

void loop()
{
  for(pos = 0; pos < 180; pos += 1) // Vai de 0 grau a 180 em passo de 1 grau
  {
    // Chama o servo para ir para a posição da variável "pos"
    servol.write(pos);
    delay(50); // Agurade 50ms para para o servomotor atingir a posição
  }
  for(pos = 180; pos>=1; pos--=1) // Vai de 180 graus ate 0
  {
    // Chama o servo para ir para a posição da variável "pos"
    servol.write(pos);
    delay(10); // Aguarda 10ms para o servomotor atingir a posição
  }
}

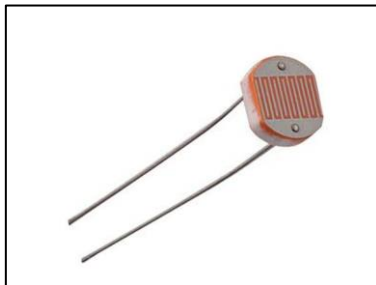
```

Fonte: Autoria própria, 2022.



O LDR (Resistor dependente da luz), é um componente semicondutor, possui apenas dois terminais e não tem polaridade definida. Quando as partículas de luz (fótons) incidem sobre a superfície do sensor, os elétrons que estão no material semicondutor são liberados, dessa forma a condutividade do LDR aumenta e a sua resistência diminui. Em estado normal o material do LDR possui uma alta resistência, por isso quanto maior a incidência de luz sobre o sensor LDR, menor será a sua resistência, ou seja, ao escurecer a resistência do LDR é máxima, e se o ambiente estiver claro a sua resistência será muito menor (JÚNIOR; JUNIOR, 2013).

**Figura 47 - Sensor de Luminosidade**



Fonte: <https://www.piscaled.com.br/resistor-dependente-de-luz-sensor-de-luminosidade-ldr-5mm>, 2022.

**Figura 48 - Exemplo código Sensor de Luminosidade**

```
int limiteDisparo = 600;
// Ligue o LED ao pino digital 9
int led = 9;

// O fotoresistor (LDR) é conectado ao pino analógico 0
int sensor = A0;

// Armazena o valor de leitura analógica
int sensorValue = 0;

void setup() {

    // Define o LED como uma saída
    pinMode(led, OUTPUT);

    // Define o fotoresistor como uma entrada
    pinMode(sensor, INPUT);

    // Inicia a comunicação serial com uma taxa de transmissão de 9600 baud rate
    Serial.begin(9600);
}

void loop(){

    // Lê o valor atual do fotoresistor
    sensorValue = analogRead(sensor);

    if (sensorValue < limiteDisparo) {
        digitalWrite(led, HIGH);
    }
    else {
        digitalWrite(led, LOW);
    }

    delay(130);
}
```

Fonte: Autoria própria, 2022.

O Teclado Matricial de Membrana 4X4 foi desenvolvido com a finalidade de facilitar a entrada de dados em projetos com plataformas microcontroladas. Este teclado possui 16 teclas, onde 10 teclas são numerais, 4 literais e 2 de caracteres. As 16 teclas estão dispostas em 4 linhas por 4 colunas e o teclado possui um conector de 8 pinos para ligação. Esses teclados são chamados de teclados matriciais, normalmente usados em aparelhos telefônicos (SILVA; VALIM, 2011).

**Figura 49 - Teclado de Membrana**



Fonte: <https://www.filipeflop.com/produto/teclado-matricial-de-membrana-16-teclas/>, 2022.

**Figura 50 - Exemplo código do Teclado de Membrana (1)**

```
#include <Keypad.h> // BIBLIOTECA PARA O FUNCIONAMENTO DO TECLADO DE MEMBRANA
#include <Servo.h> // BIBLIOTECA PARA O FUNCIONAMENTO DO SERVO

Servo servo_Motor; //OBJETO DO TIPO SERVO
char* password = "123"; //SENHA CORRETA PARA DESTRANCAR A FECHADURA
int position = 0; //VARIÁVEL PARA LEITURA DE POSIÇÃO DA TECLA PRESSIONADA
const byte ROWS = 4; //NUMERO DE LINHAS DO TECLADO
const byte COLS = 4; //NUMERO DE COLUNAS DO TECLADO
char keys[ROWS][COLS] = { //DECLARAÇÃO DOS NUMEROS, LETRAS E CARACTERES DO TECLADO
  {'1','2','3','A'},
  {'4','5','6','B'},
  {'7','8','9','C'},
  {'*','0','#','D'}
};

byte rowPins[ROWS] = { 8, 7, 6, 9 }; // PINOS DE CONEXAO DAS LINHAS DO TECLADO
byte colPins[COLS] = { 5, 4, 3, 2 }; //PINOS DE CONEXAO DAS COLUNAS DO TECLADO
Keypad keypad = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );

const int ledVermelho = 12; //PINO EM QUE ESTÁ CONECTADO O LED VERMELHO
const int ledVerde = 13; //PINO EM QUE ESTÁ CONECTADO O LED VERDE

void setup(){
  pinMode(ledVermelho, OUTPUT); //DECLARA O PINO COMO SAÍDA
  pinMode(ledVerde, OUTPUT); //DECLARA O PINO COMO SAÍDA

  servo_Motor.attach(11); //PINO DE CONTROLE DO SERVO MOTOR
  setLocked(true); //ESTADO INICIAL DA FECHADURA (TRANCADA)
}

void loop(){
  char key = keypad.getKey(); //LEITURA DAS TECLAS PRESSIONADAS
  if (key == '*' || key == '#') { //SE A TECLA PRESSIONADA POR IGUAL A CARACTERE "*" OU "#", FAZ
    position = 0; //POSIÇÃO DE LEITURA DA TECLA PRESSIONADA INICIA EM 0
    setLocked(true); //FECHADURA TRANCADA
  }
  if (key == password[position]){ //SE A TECLA PRESSIONADA CORRESPONDER A SEQUÊNCIA DA SENHA, FAZ
    position++; //PULA PARA A PRÓXIMA POSIÇÃO
  }
  if (position == 3){
    setLocked(false); //FECHADURA DESTRANCADA
  }
  delay(100); //INTERVALO DE 100 MILISSEGUNDOS
}
```

Fonte: Autoria própria, 2022.

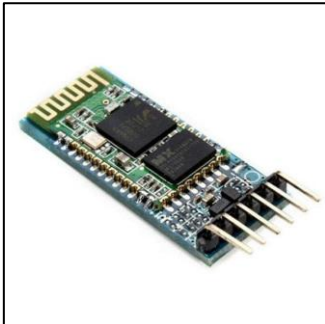
**Figura 51 - Exemplo código do Teclado de Membrana (2)**

```

}
void setLocked(int locked){ //TRATANDO O ESTADO DA FECHADURA
if (locked){ //SE FECHADURA TRANCADA, FAZ
    digitalWrite(ledVermelho, HIGH); // LED VERMELHO ACENDE
    digitalWrite(ledVerde, LOW); // LED VERDE APAGA
    servo_Motor.write(0); //POSIÇÃO DO SERVO FICA EM 0° (FECHADURA TRANCADA)
}
else{
    digitalWrite(ledVerde, HIGH); // LED VERDE ACENDE
    digitalWrite(ledVermelho, LOW); // LED VERMELHO APAGA
    servo_Motor.write(82); // SERVO GIRA A 82° (FECHADURA DESTRANCADA)
}
}
}

```

O HC-06 permite a utilização de comunicação sem fio bluetooth, uma tecnologia de radiofrequência a qual possibilita a comunicação entre componentes, com baixo consumo de energia e de forma descomplicada em sua interligação. O modulo HC-06 opera somente em estado slave sendo que apenas é possível que outros aparelhos se conectem a ele, mas ele não pode por si só, se conectar a outros (CUNHA; ROCHA; SOUZA; OLIVEIRA; AMADO; MARQUES, 2017).

**Figura 52 - Módulo Bluetooth HC-06**

Fonte: <https://awstatic5754.kxcdn.com/1hC70AoWkXqruv3JWBDH8oNzRtQ=/0x0/loja/casadoled.com.br/prodimg/21/0241513001599089305.jpg>, 2022.

**Figura 53 - Exemplo código do Módulo Bluetooth HC-06 (1)**

```
#include <SoftwareSerial.h> //INCLUSÃO DE BIBLIOTECA

const int pinoRX = 2; //PINO DIGITAL 2 (RX)
const int pinoTX = 3; //PINO DIGITAL 3 (TX)
const int pinoLed = 12; //PINO DIGITAL UTILIZADO PELO LED
int dadoBluetooth = 0; //VARIÁVEL QUE ARMAZENA O VALOR ENVIADO PELO BLUETOOTH
boolean loopLED = false; //VARIÁVEL BOOLEANA QUE FAZ O CONTROLE DE ATIVAÇÃO DO LOOP DO LED
SoftwareSerial bluetooth(pinoRX, pinoTX);

void setup(){
  Serial.begin(9600); //INICIALIZA A SERIAL
  bluetooth.begin(9600); //INICIALIZA A SERIAL DO BLUETOOTH
  bluetooth.print("$"); //IMPRIME O CARACTERE
  bluetooth.print("$"); //IMPRIME O CARACTERE
  bluetooth.print("$"); //IMPRIME O CARACTERE
  delay(100); //INTERVALO DE 100 MILISSEGUNDOS
  pinMode(pinoLed, OUTPUT); //DEFINE O PINO COMO SAÍDA
}

void loop(){
  if(bluetooth.available()){ //SE O BLUETOOTH ESTIVER HABILITADO, FAZ
    dadoBluetooth = bluetooth.read(); //VARIÁVEL RECEBE O VALOR ENVIADO PELO BLUETOOTH

    if(dadoBluetooth == '1'){ //SE O VALOR RECEBIDO FOR IGUAL A 1, FAZ
      Serial.println("LED LIGADO"); //IMPRIME O TEXTO NA SERIAL
      digitalWrite(pinoLed, HIGH); //LIGA O LED
    }
    if(dadoBluetooth == '0'){ //SE O VALOR RECEBIDO FOR IGUAL A 0, FAZ
      Serial.println("LED DESLIGADO"); //IMPRIME O TEXTO NA SERIAL
      digitalWrite(pinoLed, LOW); //DESLIGA O LED
    }
    if(dadoBluetooth == 'b'){ //SE O VALOR RECEBIDO FOR IGUAL A b, FAZ
      Serial.println("LOOP DO LED ATIVADO"); //IMPRIME O TEXTO NA SERIAL
      loopLED = true; //VARIÁVEL RECEBE verdadeiro
    }else{ //SENÃO, FAZ
      loopLED = false; //VARIÁVEL RECEBE falso
    }
  }

  //MÉTODO RESPONSÁVEL PELO LOOP (LIGA / DESLIGA) DO LED
  if(loopLED){
    digitalWrite(pinoLed, HIGH); //LIGA O LED
  }
}
```

Fonte: Autoria própria, 2022.

**Figura 54 - Exemplo código do Módulo Bluetooth HC-06 (2)**

```
Serial.println("LOOP (LED LIGADO)"); //IMPRIME O TEXTO NA SERIAL
delay(500); //INTERVALO DE 500 MILISSEGUNDOS
digitalWrite(pinoLed, LOW); //DESLIGA O LED
Serial.println("LOOP (LED DESLIGADO)"); //IMPRIME O TEXTO NA SERIAL
delay(500); //INTERVALO DE 500 MILISSEGUNDOS
}
}
```

Fonte: Autoria própria, 2022.

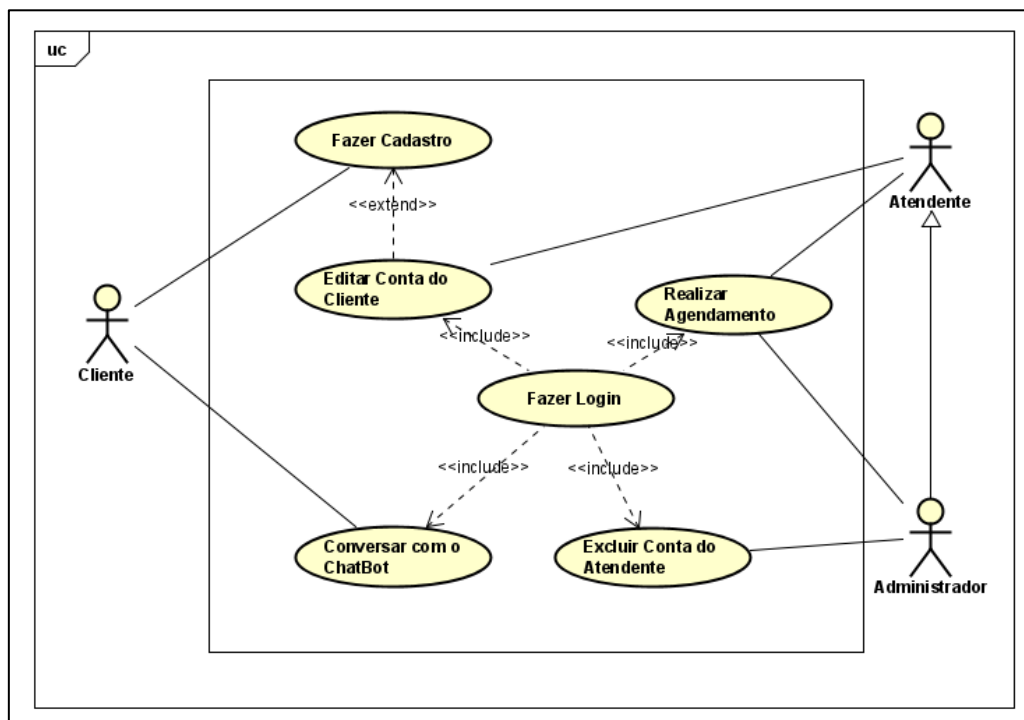
### 3 DESENVOLVIMENTO

Neste capítulo será abordado o desenvolvimento do aplicativo web FourHouse, através dos diagramas desenvolvidos no estudo de UML, a criação do Banco de Dados e todas as tecnologias aplicadas na construção do aplicativo web e da maquete.

#### 3.1 Diagrama de Casos de Uso

O Diagrama de Caso de Uso demonstra a interação dos atores cliente, atendente e administrador com a aplicação, ilustrando a acessibilidade e as ações que o usuário poderá realizar através dos casos de uso e suas relações.

**Figura 55 - Diagrama de Caso de Uso fourhouse**

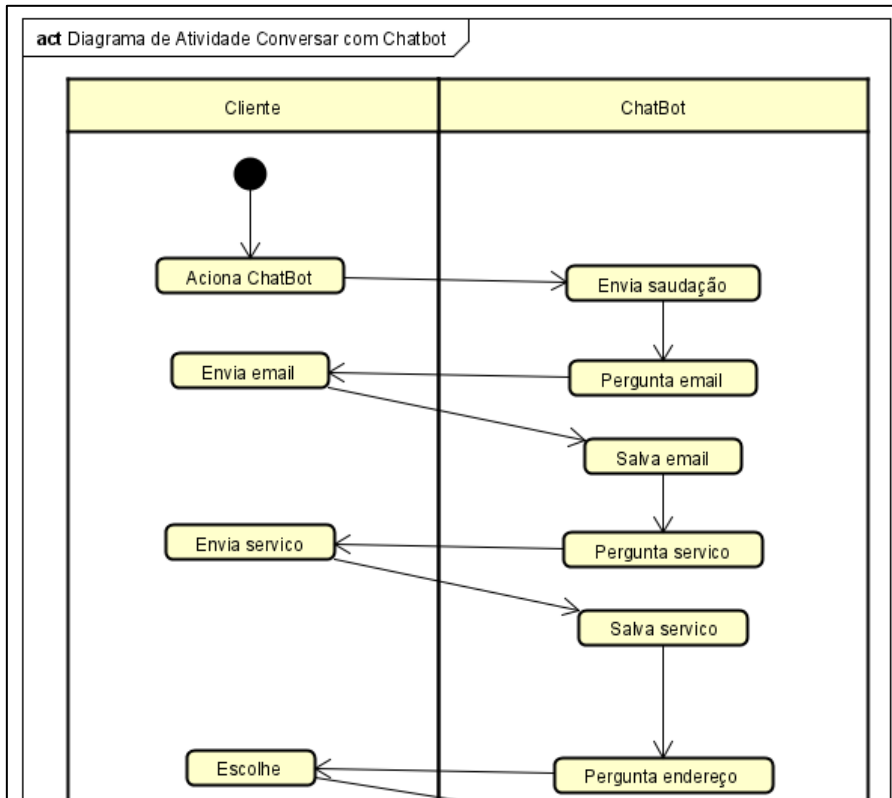


Fonte: Autoria própria, 2022.

#### 3.2 Diagrama de Atividades

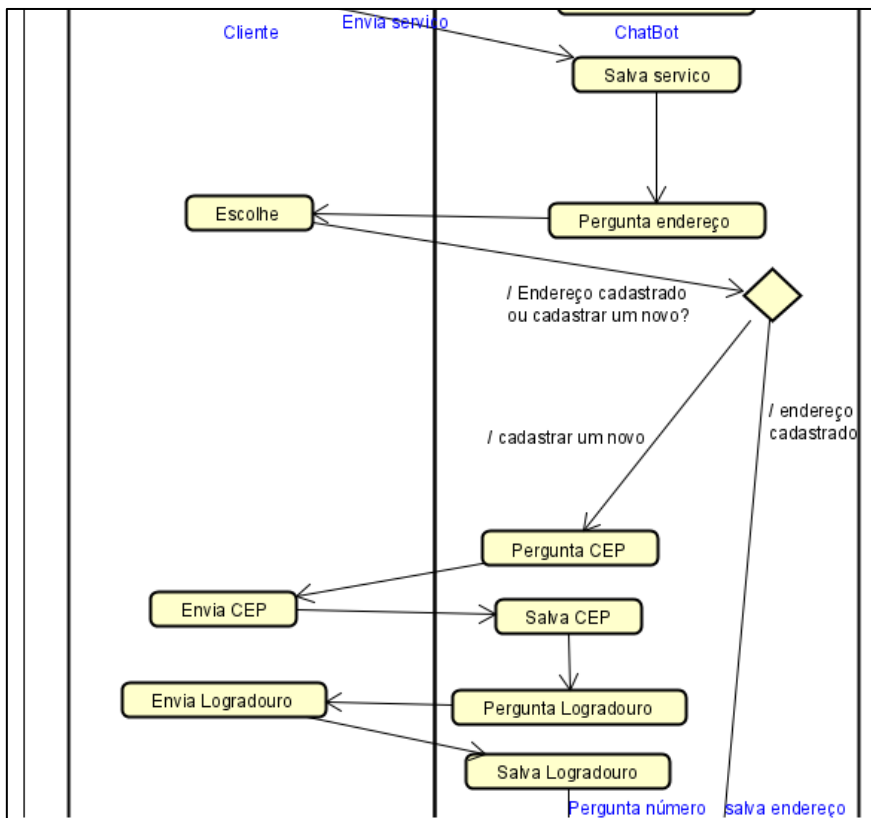
Os diagramas de atividades foram divididos de acordo com a quantidade de casos de uso, para cada caso de uso, existe um diagrama de atividades.

**Figura 56 - Diagrama de Atividade Conversar com Chatbot (1)**



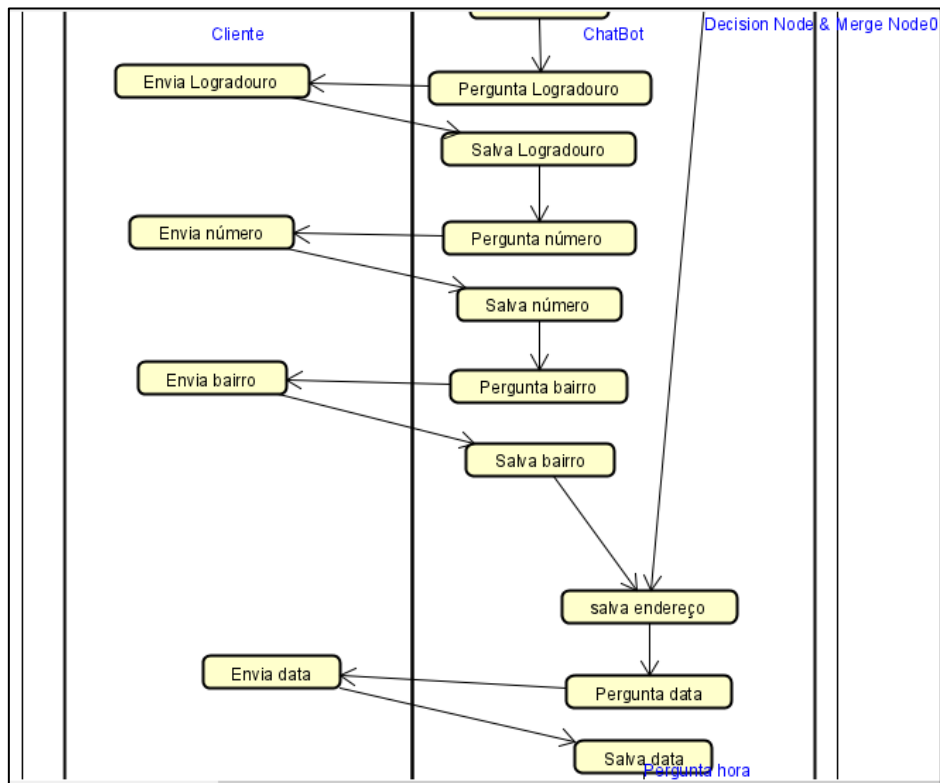
Fonte: Autoria própria, 2022.

**Figura 57 - Diagrama de Atividade Conversar com Chatbot (2)**



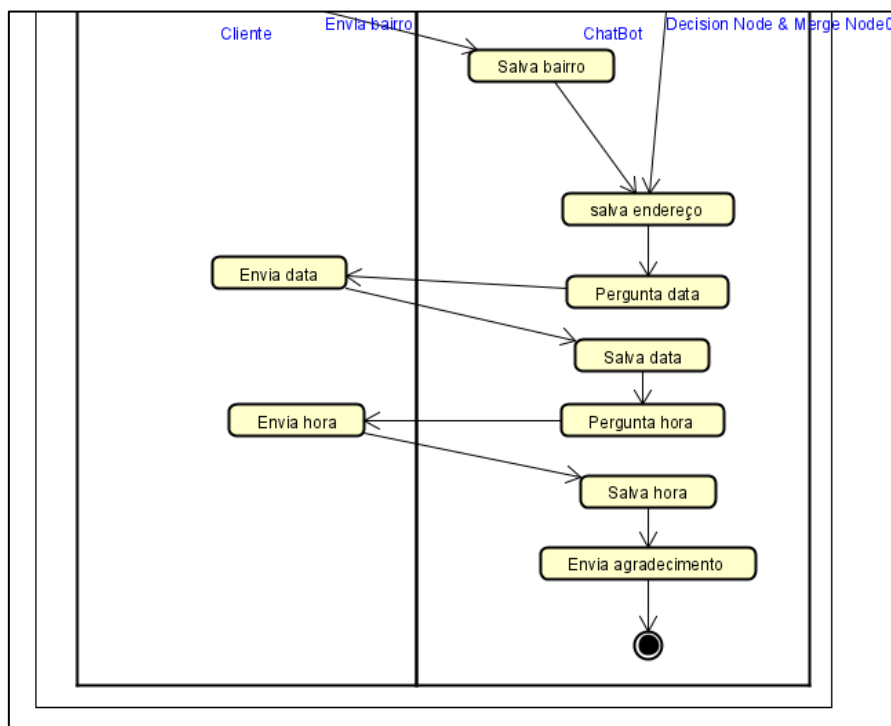
Fonte: Autoria própria, 2022.

**Figura 58 - Diagrama de Atividade Conversar com chatbot (3)**



Fonte: Autoria própria, 2022.

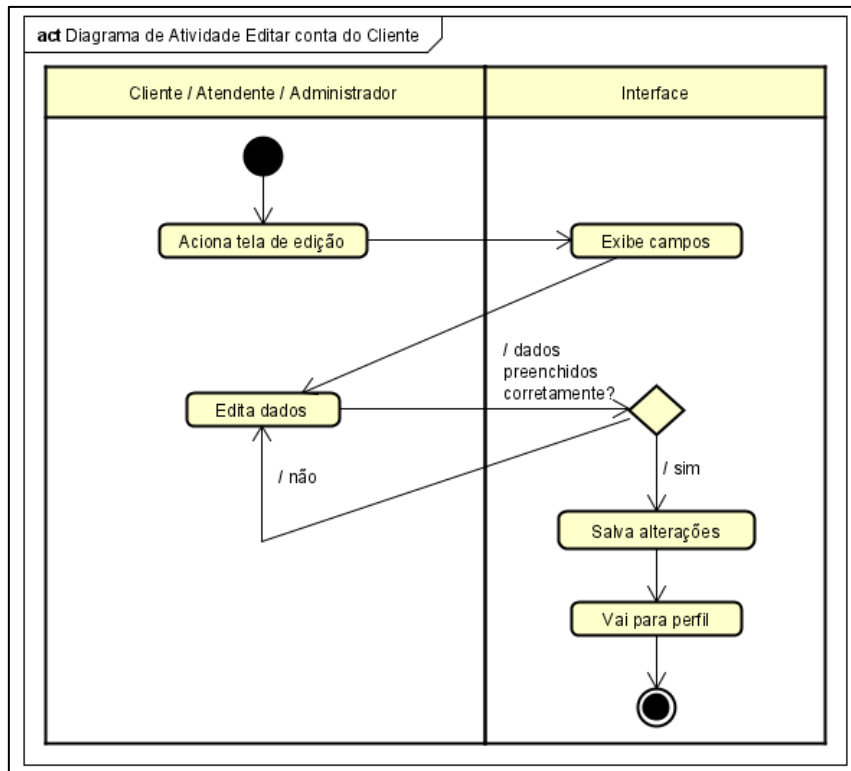
**Figura 59 - Diagrama de Atividade conversar com Chatbot (4)**



Fonte: Autoria própria, 2022.

As Figuras 56, 57, 58 e 59 apresentam um único diagrama de atividades, nele é ilustrado como deve ser a interação do cliente com o Chatbot do site.

**Figura 60 - Diagrama de Atividades Editar conta do Cliente**

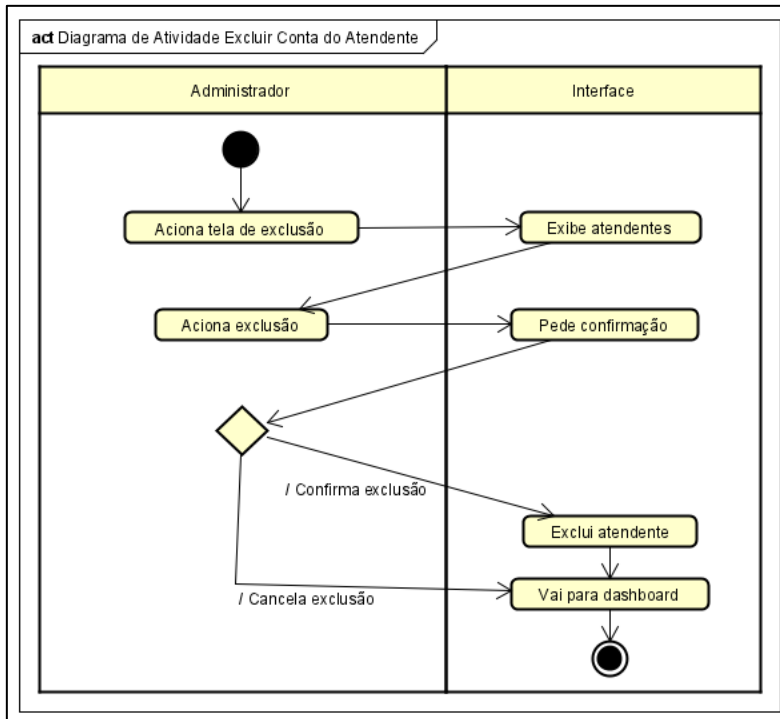


Fonte: Autoria própria, 2022.

Na Figura 60 é apresentado o diagrama de atividades do caso de uso Editar conta Cliente, onde demonstra como deve acontecer a interação dos atores Cliente, Atendente ou Administrador com a funcionalidade do site de editar a conta dos clientes.

**Figura 61 - Diagrama de Atividades Excluir Conta do Atendente**

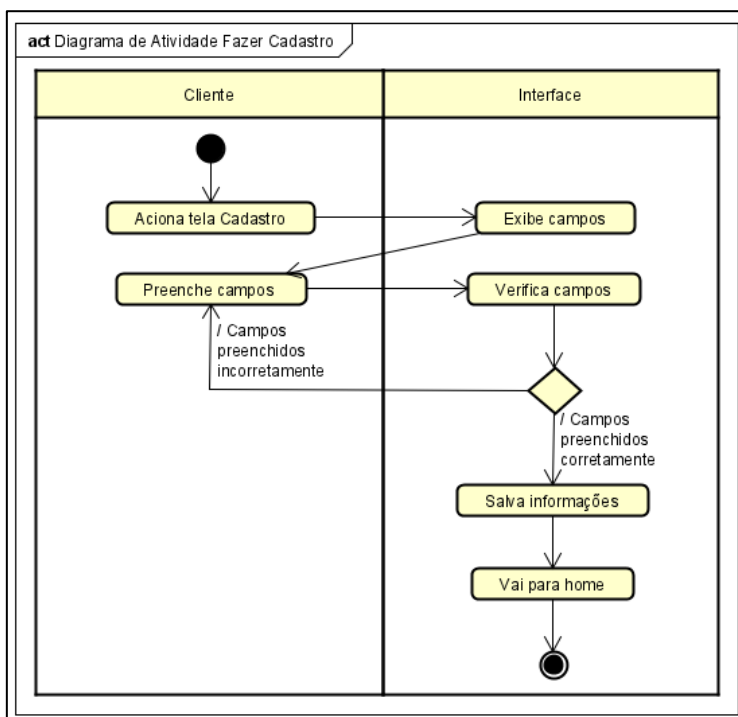




Fonte: Autoria própria, 2022.

Na Figura 61 está o diagrama de atividades do caso de uso Excluir Conta do Atendente, nele é ilustrado os passos que devem ser seguidos quando o Administrador for excluir a conta de um Atendente.

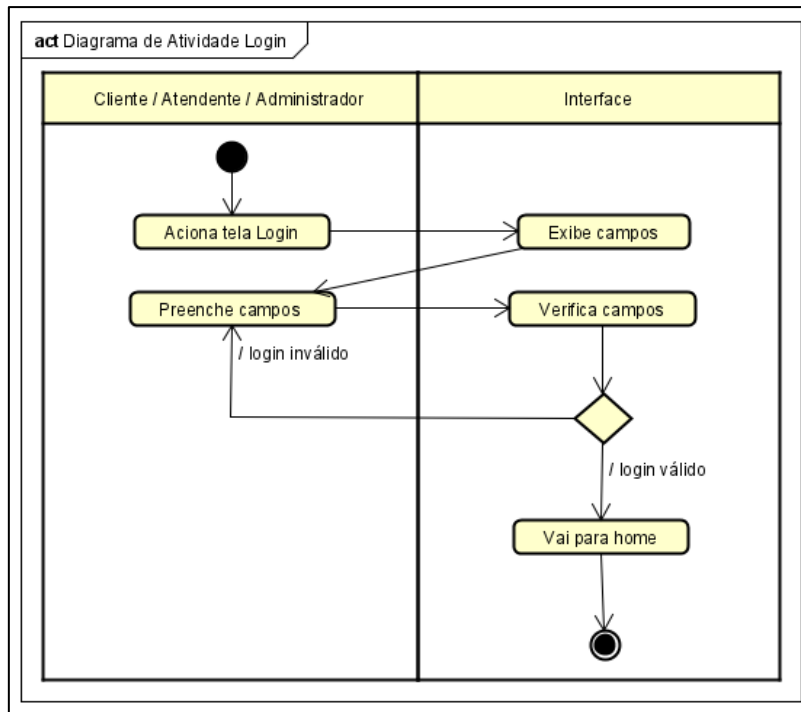
**Figura 62 - Diagrama de atividades Fazer Cadastro**



Fonte: Autoria própria, 2022.

No Diagrama de Atividades da Figura 62, está ilustrando o que deve ocorrer no caso de uso Fazer Cadastro, os passos que o Cliente e o sistema devem seguir quando o cliente quiser realizar um cadastro.

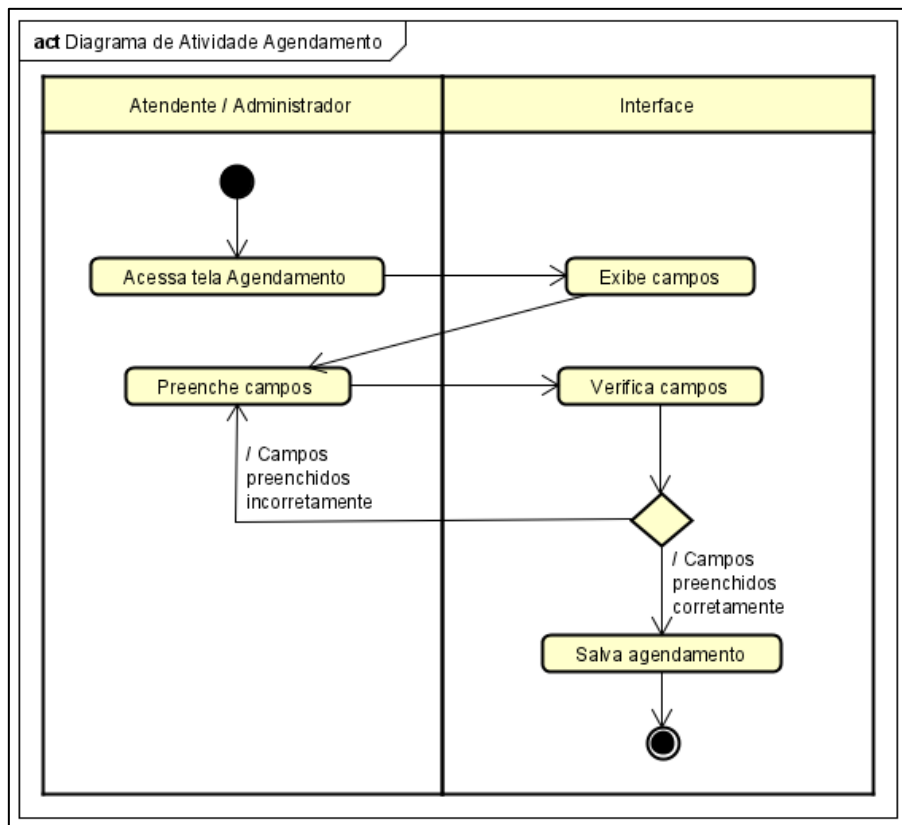
**Figura 63 - Diagrama de Atividades Login**



Fonte: Autoria própria, 2022.

O Diagrama de Atividades da Figura 63 ilustra os passos que devem ser seguidos quando o Cliente, Atendente ou Administrador forem fazer login no site.

**Figura 64 - Diagrama de Atividades Agendamento**

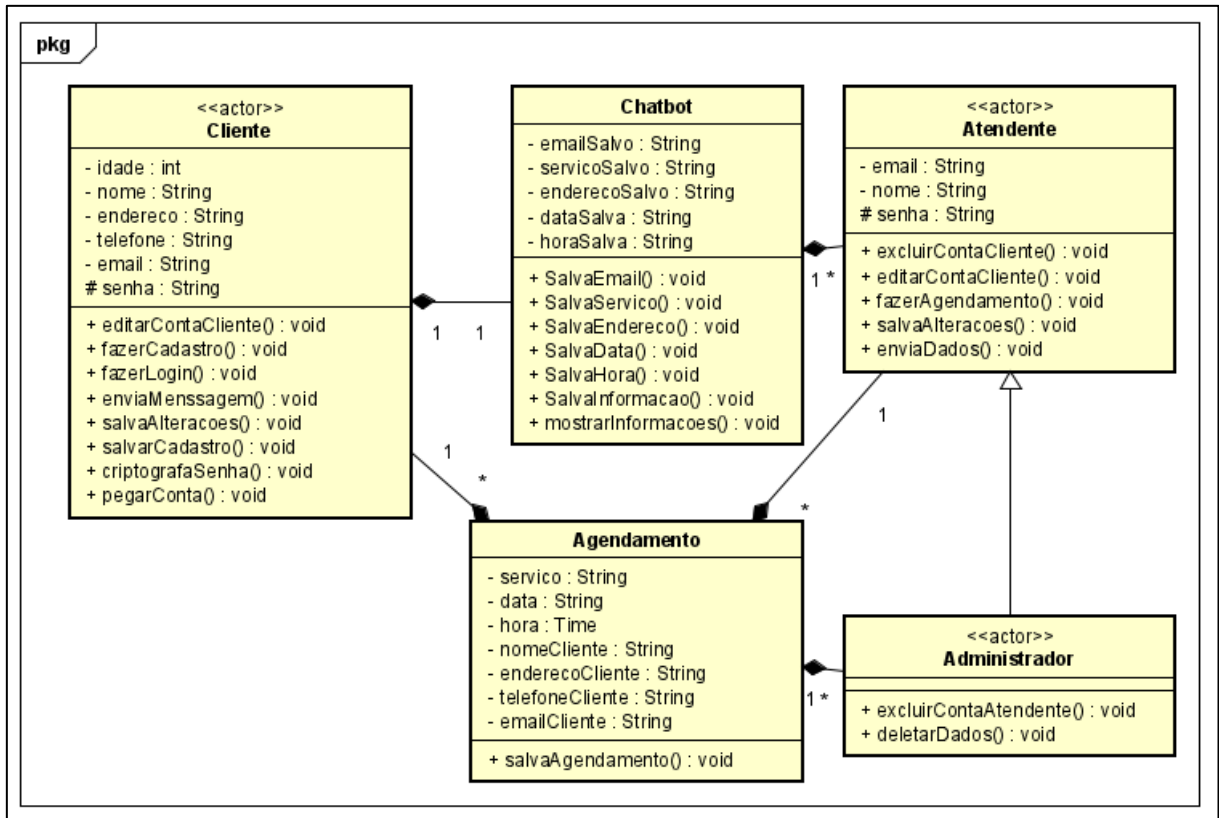


Fonte: Autoria própria, 2022.

O Diagrama de Atividades da Figura 64 ilustra os passos que devem ser seguidos quando o Atendente ou Administrador forem realizar um agendamento no site.

### 3.3 Diagrama de Classe

Figura 65 - Diagrama de Atividades fourhouse



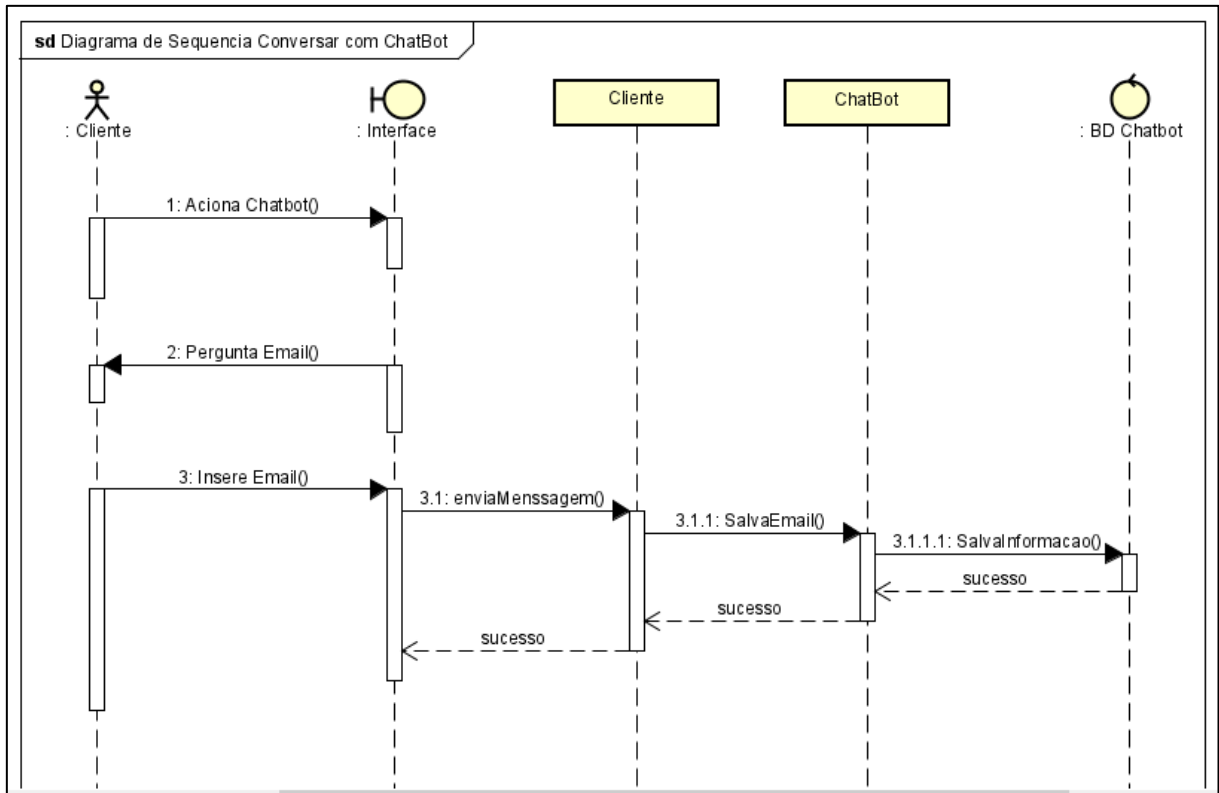
Fonte: Autoria própria, 2022.

O Diagrama de Classes da Figura 65 todas as classes que são as utilizadas na aplicação e no banco, exceto Chatbot, que é utilizada apenas na aplicação por ser classe de um sistema terceirizado.

### 3.4 Diagrama de Sequência

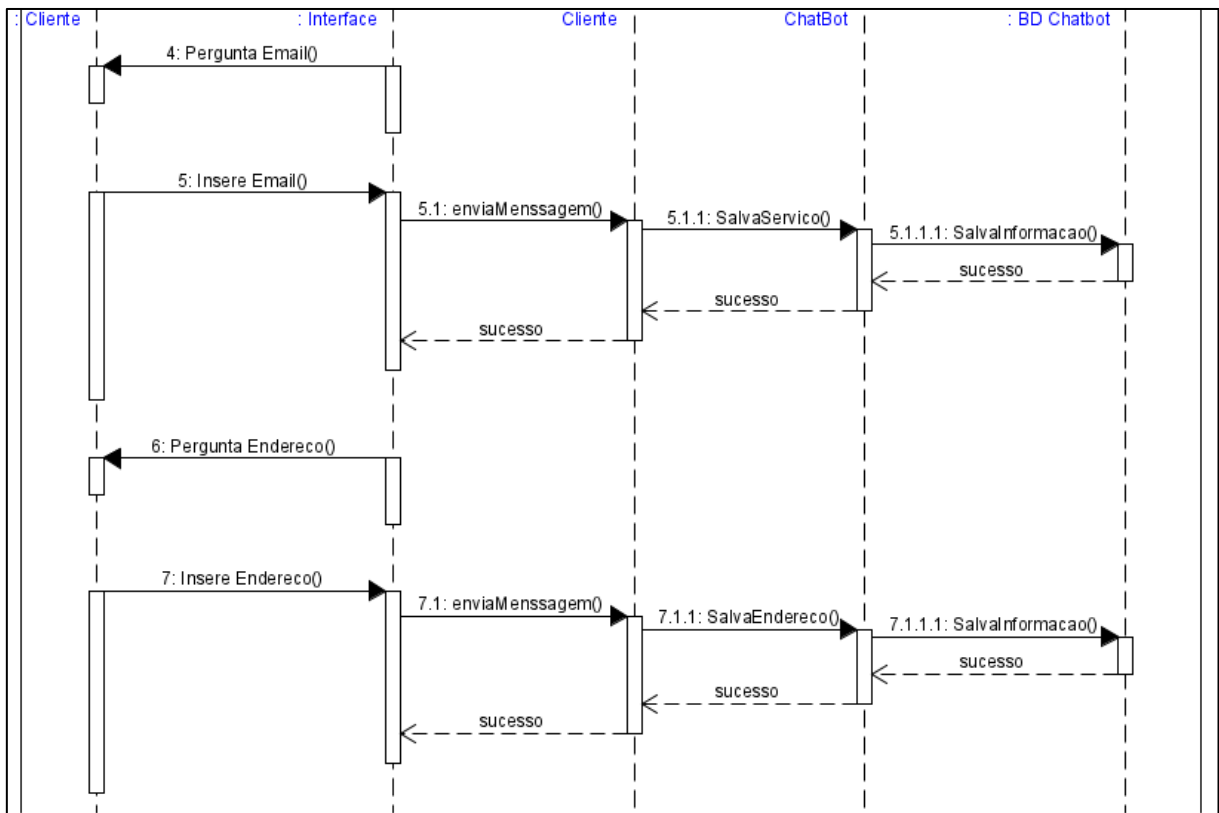
Os diagramas de sequência foram divididos de acordo com a quantidade de casos de uso, para cada caso de uso, existe um diagrama de sequência.

**Figura 66 - Diagrama de Sequência Conversar com Chatbot (1)**

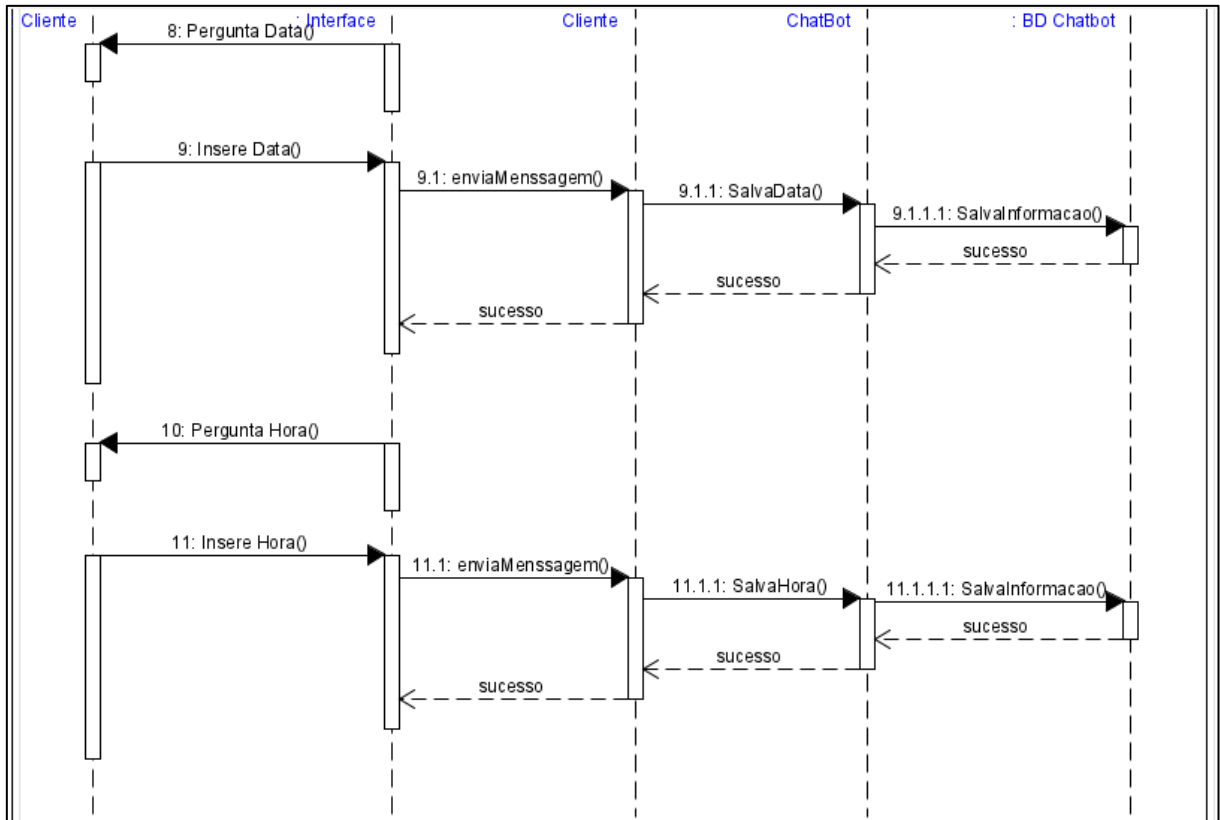


Fonte: Autoria própria, 2022.

**Figura 67 - Diagrama de Sequência Conversar com Chatbot (2)**



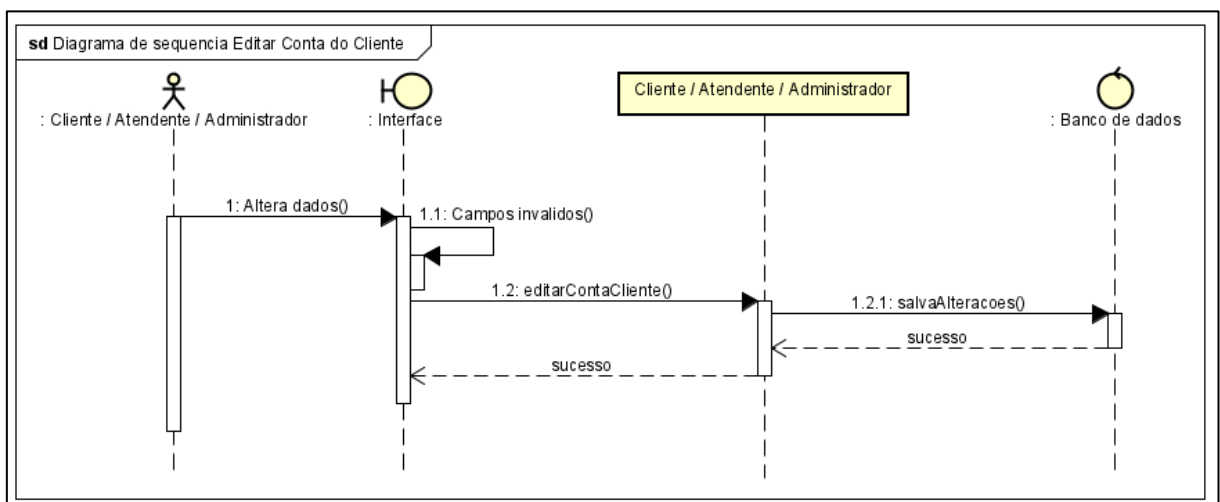
**Figura 68 - Diagrama de Sequência Conversar com Chatbot (3)**



Fonte: Autoria própria, 2022.

As Figuras 66, 67 e 68 apresentam um único diagrama de sequência, nele é ilustrado como o sistema deve se comportar quando o Chatbot do site for acionado.

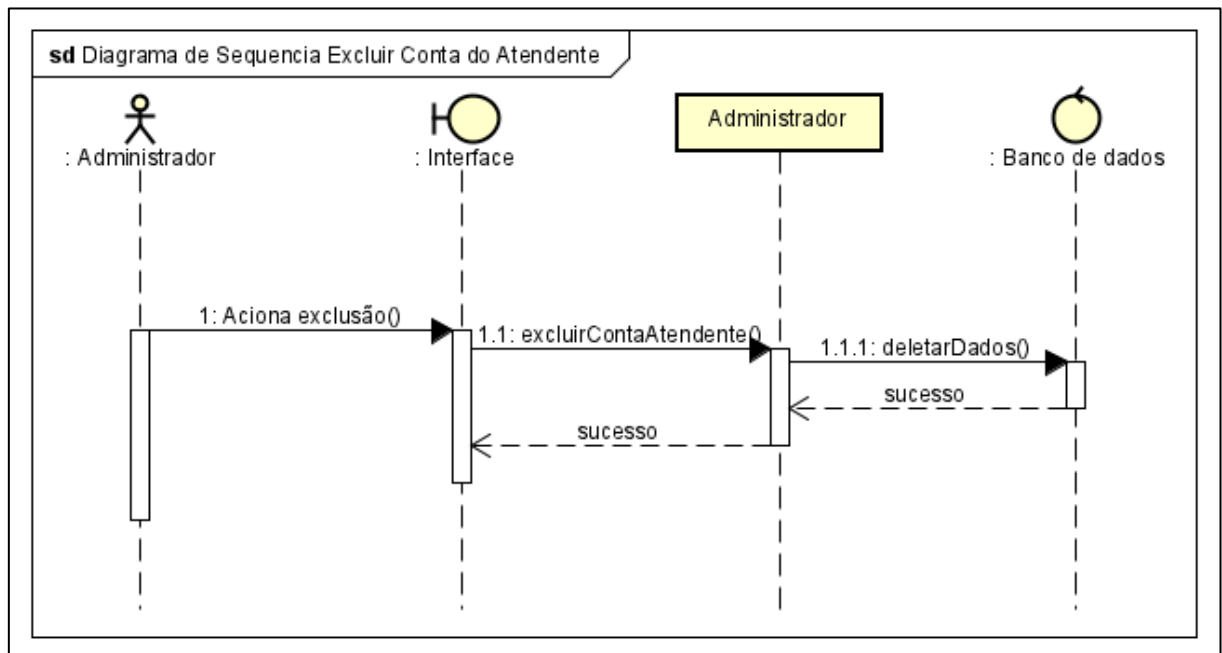
**Figura 69 - Diagrama de Sequência Editar Conta do Cliente**



Fonte: Autoria própria, 2022.

Na Figura 69 está representado o diagrama de sequência Editar Conta do Cliente, onde ilustra cada passo do sistema quando um dos atores for utilizar essa função.

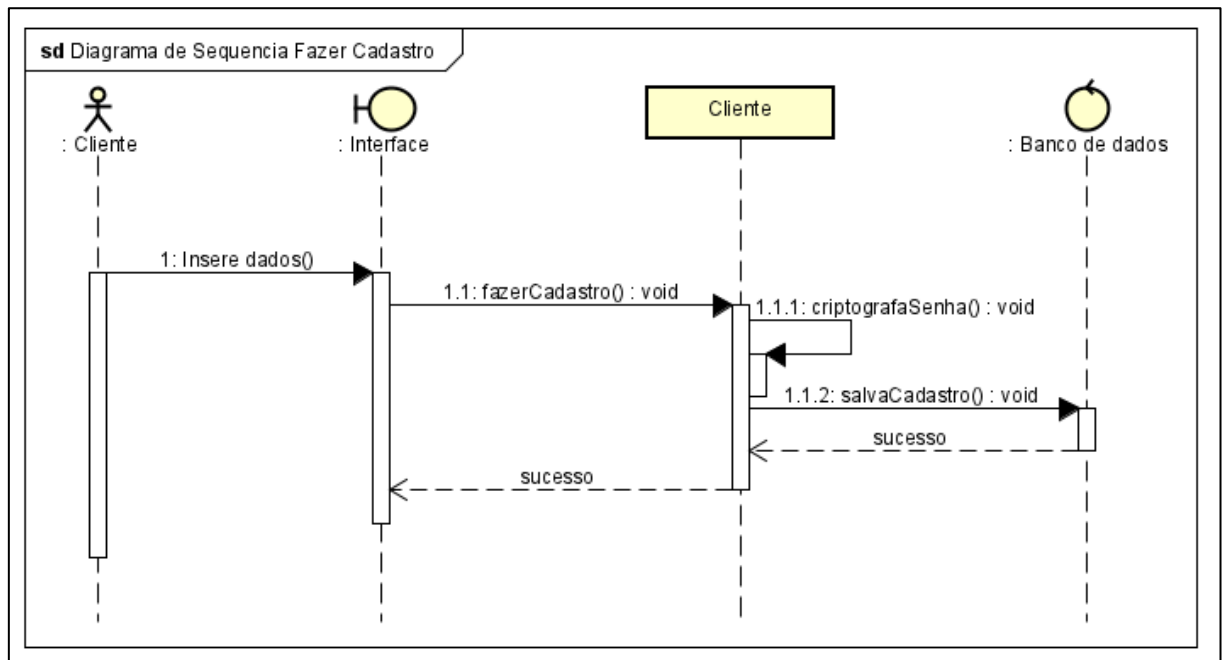
**Figura 70 - Diagrama de Sequência Excluir Conta do Atendente**



Fonte: Autoria própria, 2022.

Na Figura 70 está o Diagrama de Sequência da exclusão da conta do Atendente, ele ilustra cada passo que o sistema deve seguir quando essa ação for acionada pelo Administrador.

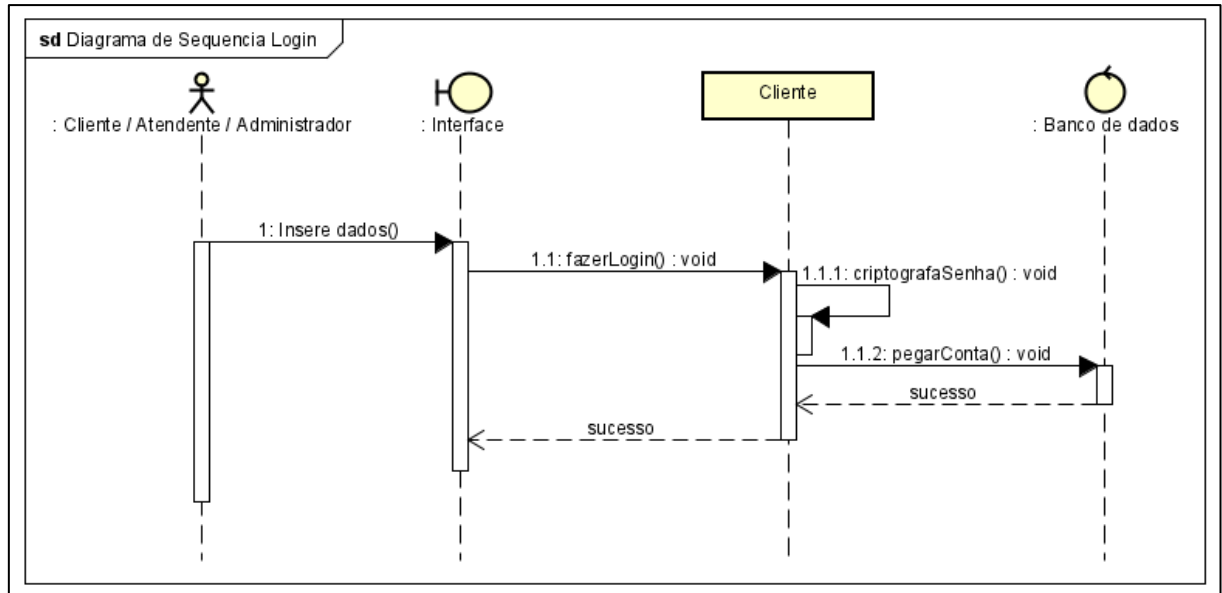
**Figura 71 - Diagrama de Sequência Fazer Cadastro**



Fonte: Autoria própria, 2022.

Na Figura 71 está o diagrama de sequência de quando o Cliente for fazer o cadastro no site, nele estão contidos todos os passos que o sistema deve seguir quando essa ação for chamada.

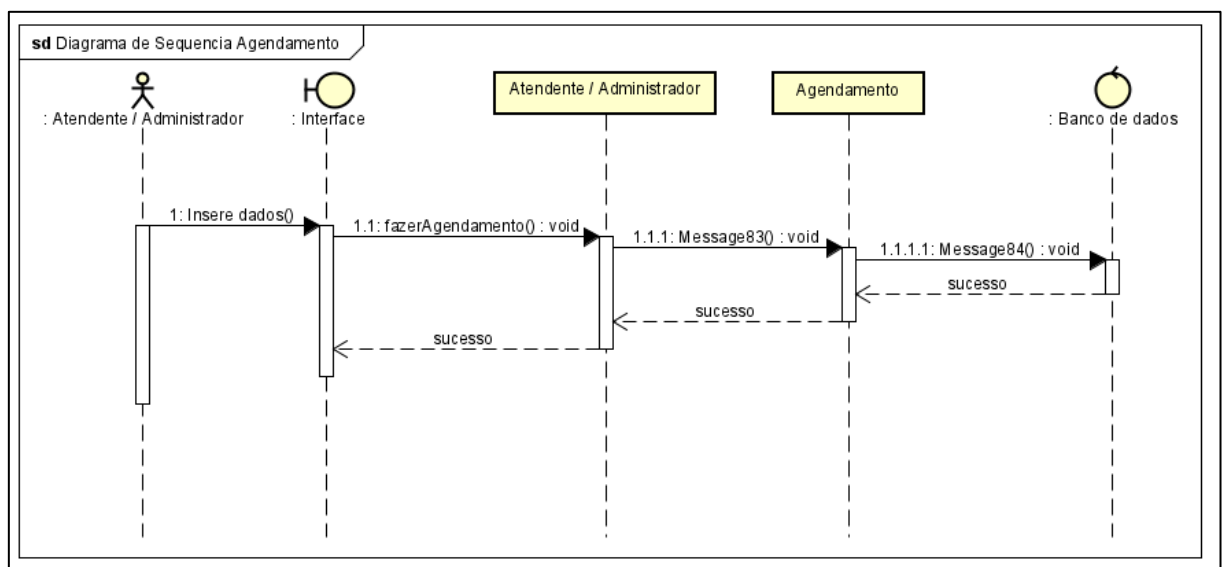
**Figura 72 - Diagrama de Sequência Login**



Fonte: Autoria própria, 2022.

Na Figura 72 o diagrama de sequência que tem nela é o de fazer login, quando um dos atores (Cliente, Atendente ou Administrador) forem fazer login no site, esses são os passos que o sistema deve executar.

**Figura 73 - Diagrama de Sequência Agendamento**



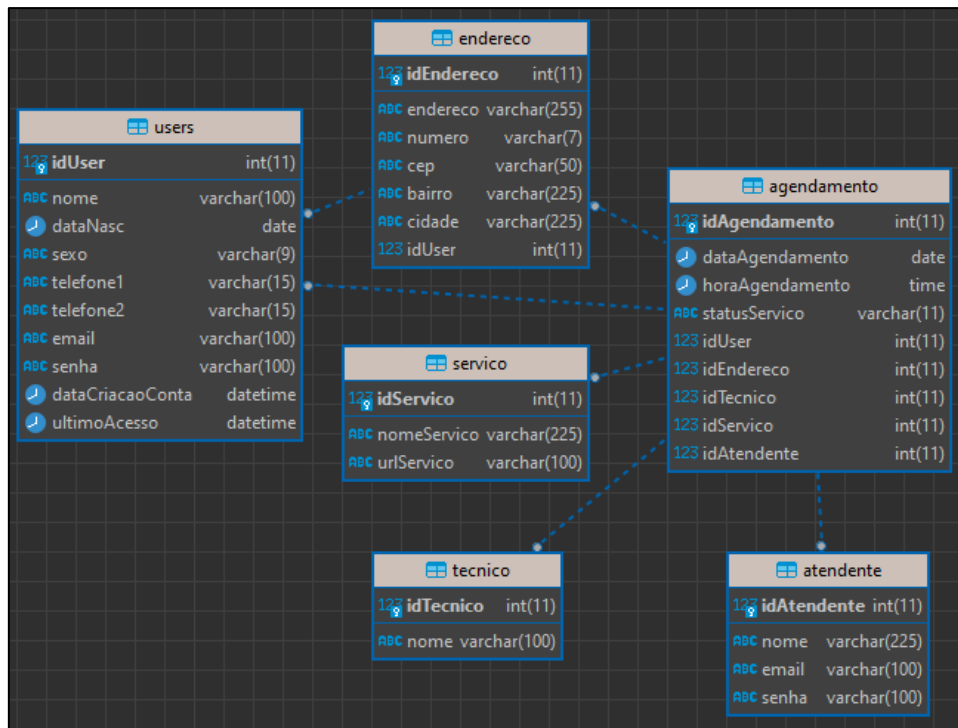
Fonte: autoria própria, 2022.



O diagrama de sequência da Figura 73 demonstra todos os passos que o sistema segue quando a ação de agendamento for acionada.

### 3.5 Diagrama de Relacionamento de Entidades

Figura 74 - Diagrama de Relacionamento de Entidades fourhouse



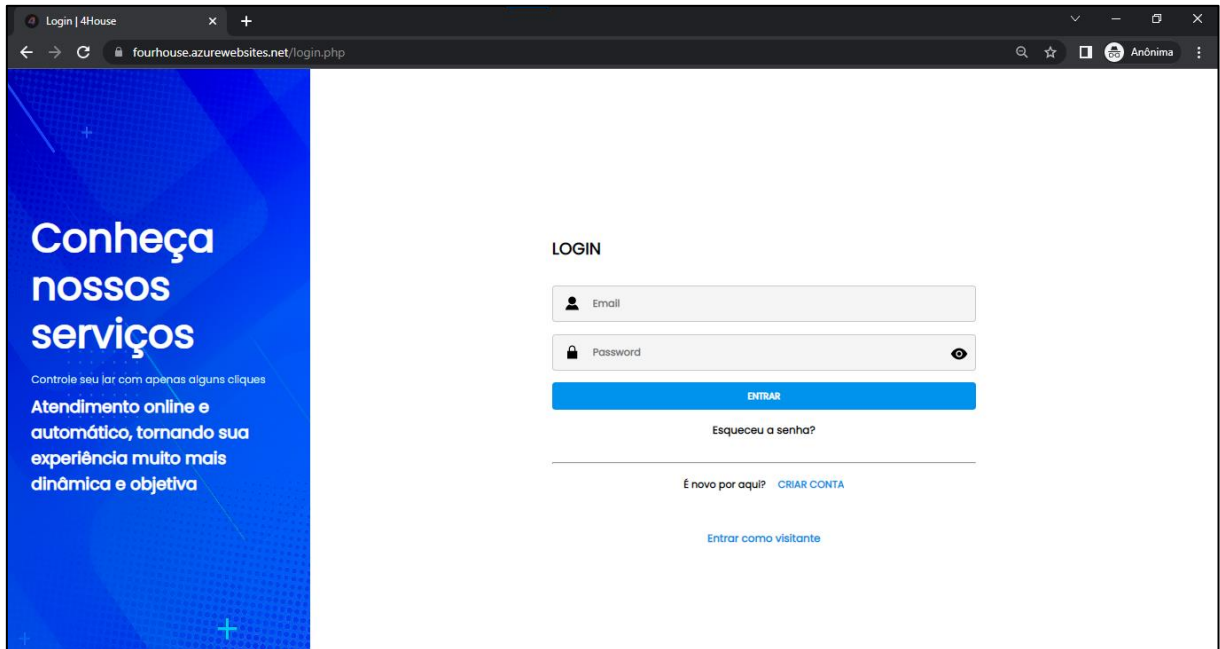
Fonte: Autoria própria, 2022.

### 3.6 Sistema Web

Para hospedagem do site, foi utilizado a plataforma da Microsoft Azure, foi utilizado também o Chatbot da empresa Crisp.

A primeira parte que o Usuário tem contato ao acessar o sistema, como mostrado da Figura 75, é a tela de acesso que dispõe de quatro funcionalidades que são: acesso ao sistema como usuário ou como visitante, criação de conta de usuário e recuperação de senha. Essas funcionalidades são integradas com o banco de dados MySQL.

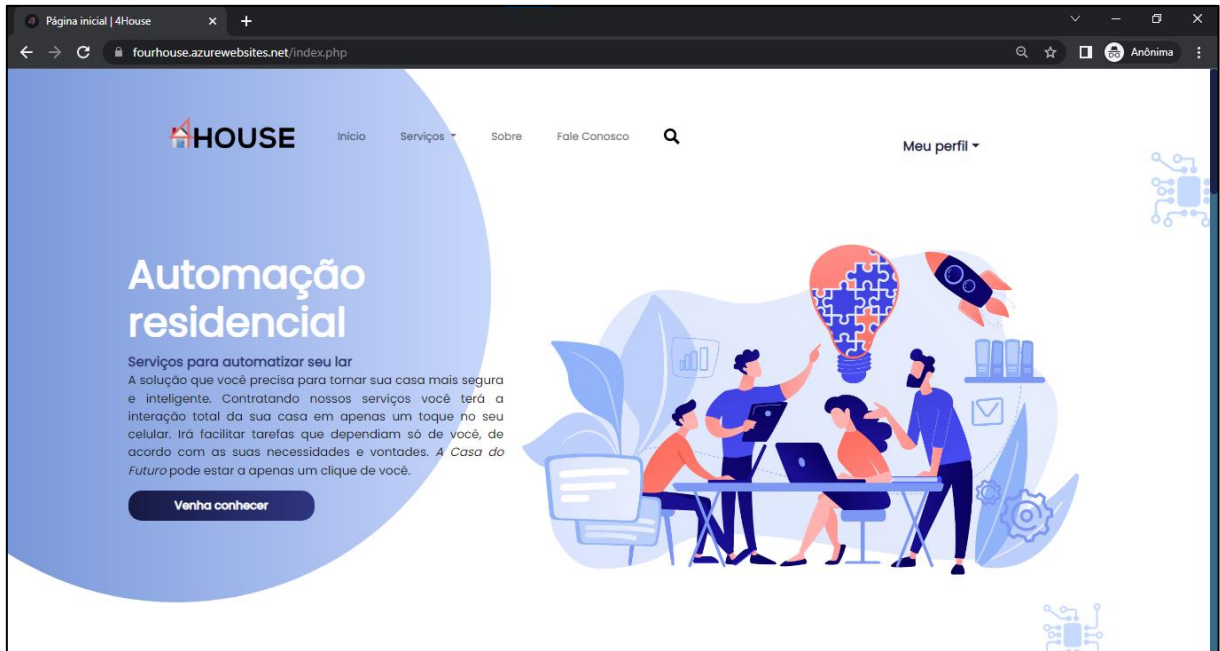
Figura 75 - Tela login fourhouse



Fonte: Autoria própria, 2022.

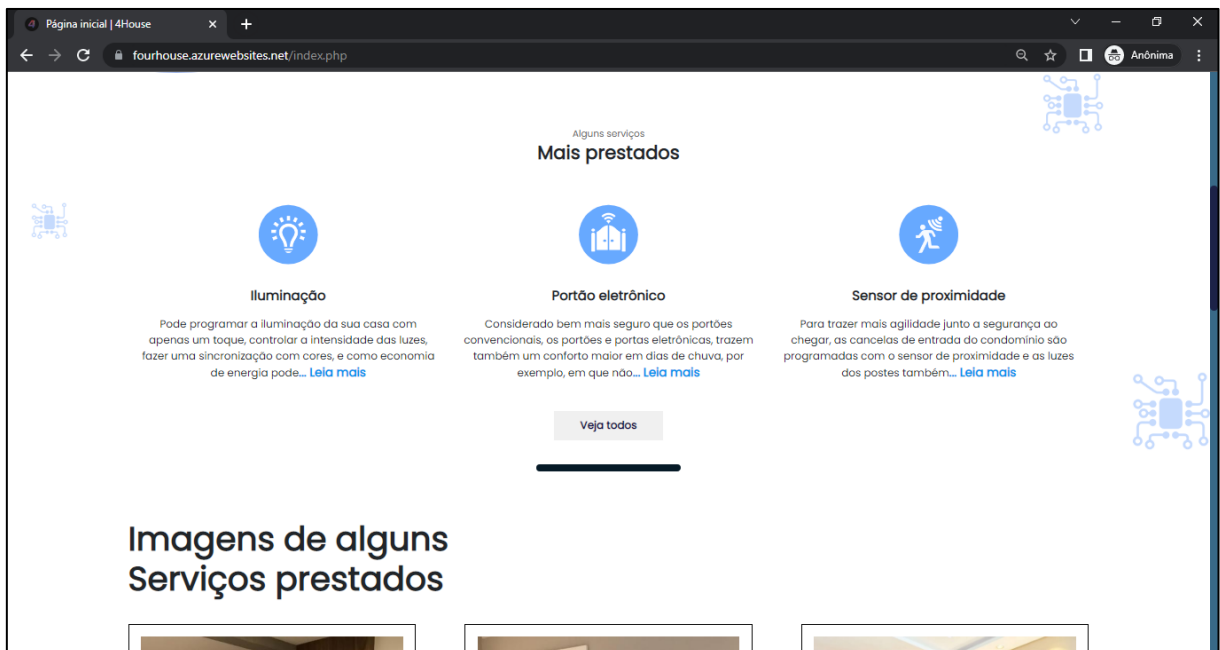
Na tela inicial do site o usuário irá se deparar com as seções que disponibilizam a navegação por todas as páginas presentes no site. No topo da tela o usuário irá visualizar a Logo no lado esquerdo tendo a opção de clicar e ser redirecionado ao início do sistema, continuando a navbar o usuário irá se deparar com opções que direcionam a páginas do sistema, sendo elas: Serviços, Sobre, Fale Conosco. À direita temos a opção Meu perfil, que ao clicar sem estar logado no sistema, irá te retornar a opção Login, caso esteja logado você terá acesso ao seu perfil com seu nome clicável cadastrado no banco de dados MySQL.

**Figura 76 - Tela inicial fourhouse (1)**



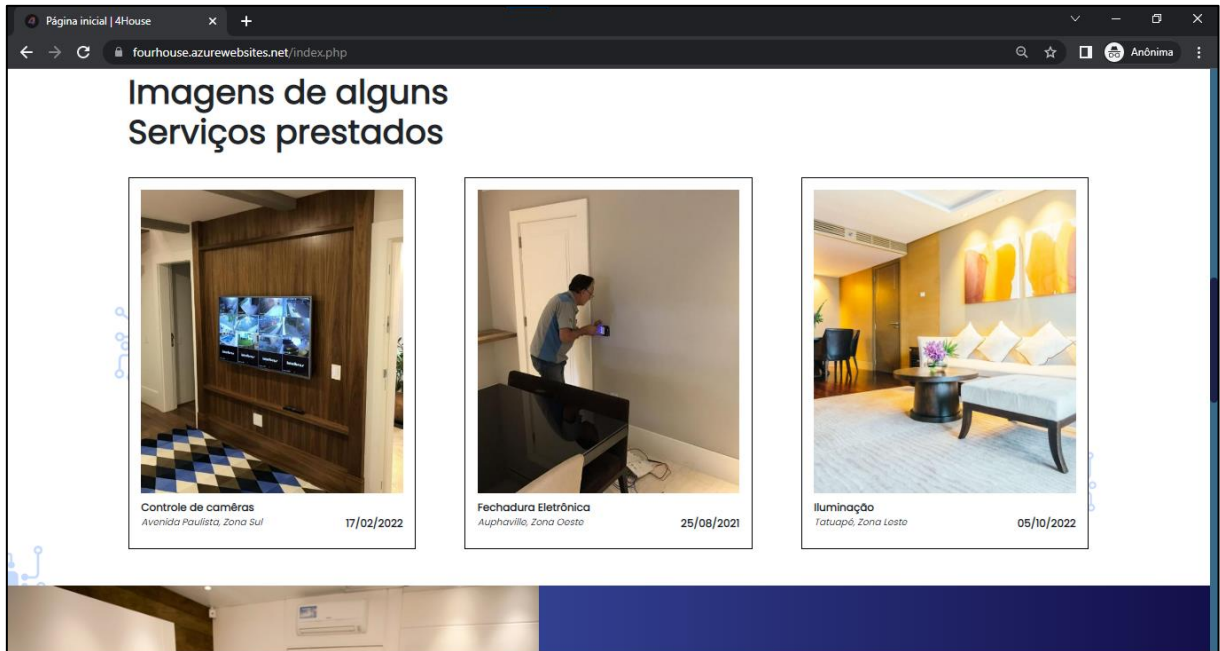
Fonte: Autoria própria, 2022.

Figura 77 - Tela inicial fourhouse (2)



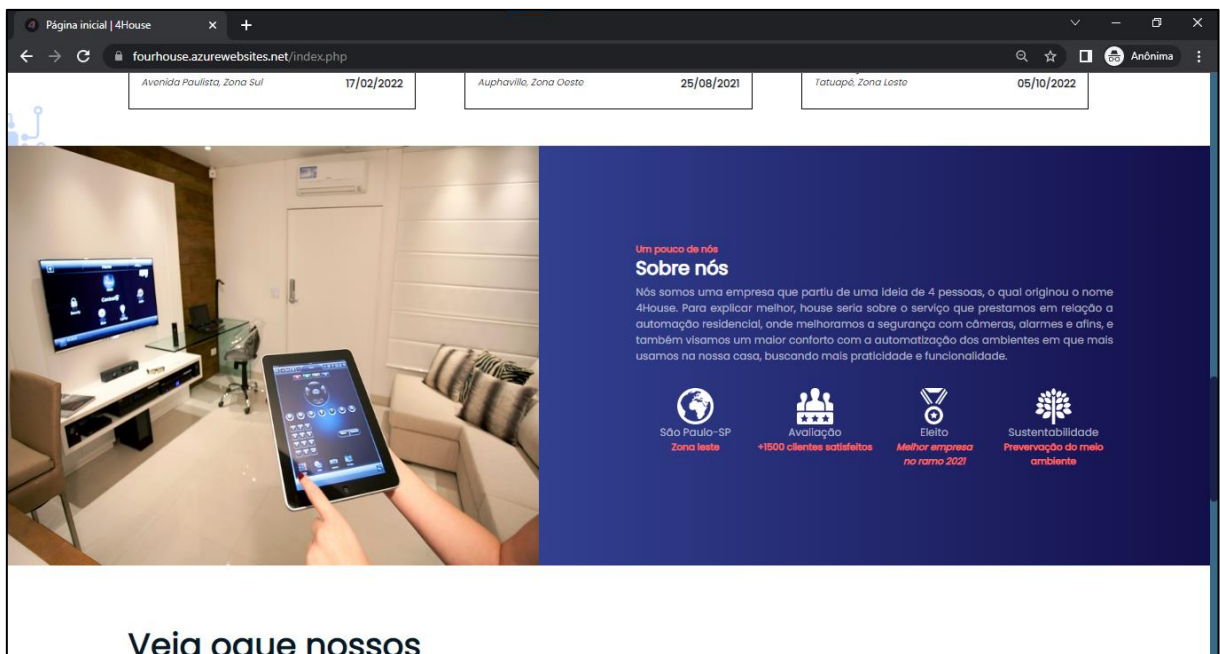
Fonte: Autoria própria, 2022.

Figura 78 - Tela inicial fourhouse (3)



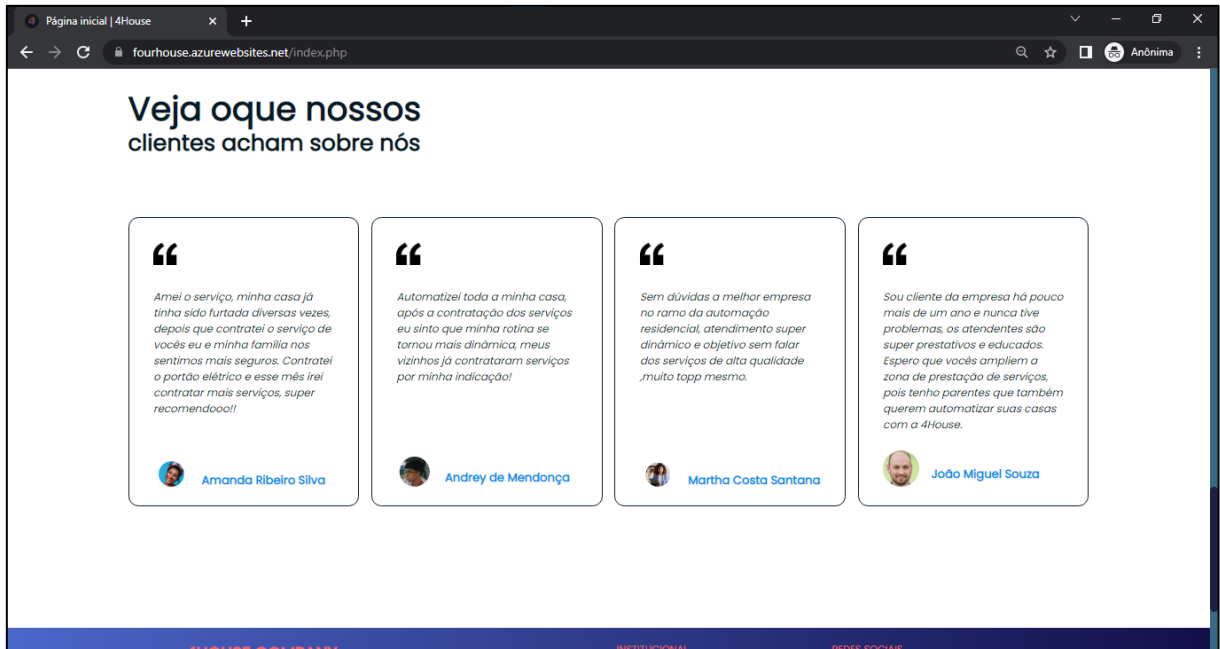
Fonte: Autoria própria, 2022.

**Figura 79 - Tela inicial fourhouse (4)**



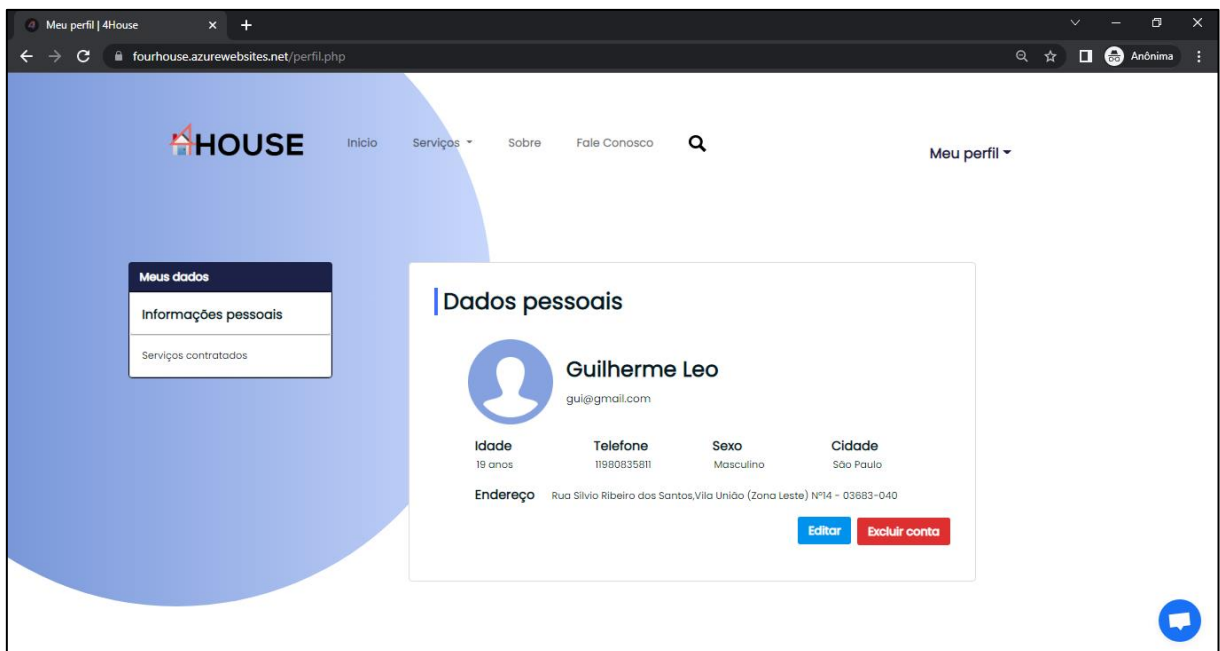
Fonte: Autoria própria, 2022.

**Figura 80 - Tela inicial fourhouse (5)**



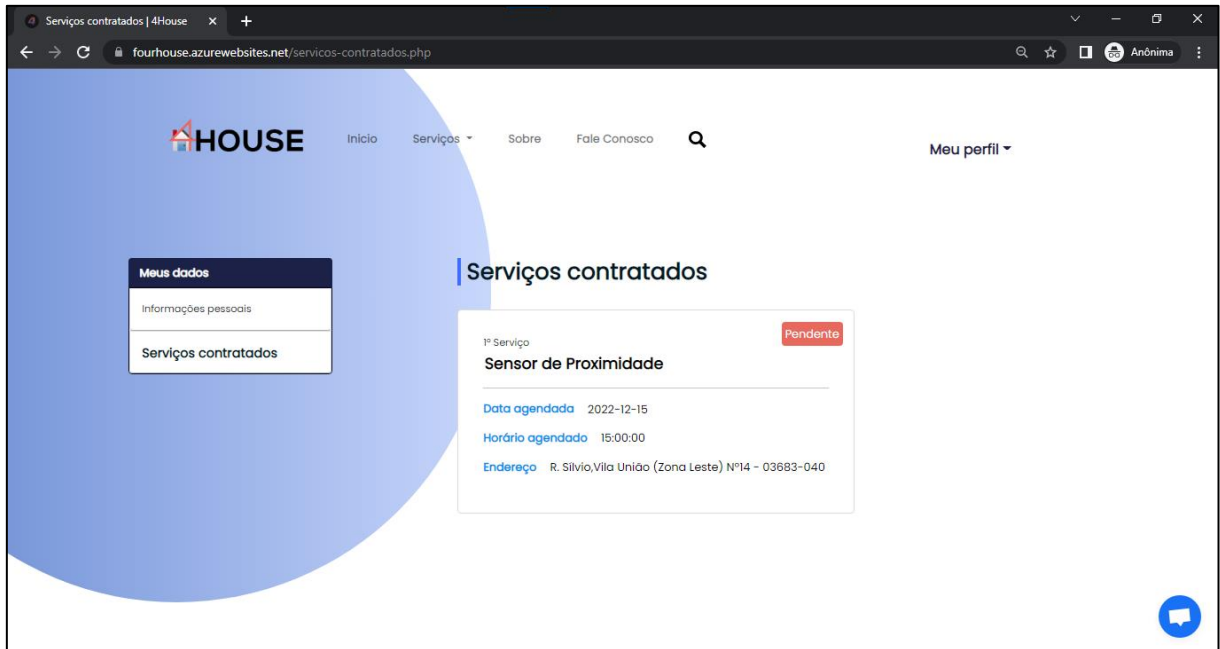
Fonte: Autoria própria, 2022.

**Figura 81 - Tela de perfil do usuário**



Fonte: Autoria própria, 2022.

**Figura 82 - Tela Serviços contratados**



Fonte: Autoria própria, 2022.

Para criar uma nova conta, o usuário se depara com a solicitação de preenchimento de dados, como, nome e sobrenome, data de nascimento, sexo, telefone, endereço, e-mail e senha; ao finalizar o cadastro o usuário é redirecionado a tela de login.

**Figura 83 - Tela de cadastro fourhouse (1)**

Fonte: Autoria própria, 2022.

**Figura 84 - Tela de cadastro fourhouse (2)**

Sexo\*  
Masculino

Telefone\*  
Telefone 1  
Telefone 2 (opcional)

Endereço\*  
CEP ex.0000-000  
Rua  
Rua  
Bairro  
Bairro  
Cidade  
Cidade  
Número da casa  
Número da casa

Digite um email\*  
Email

Insira uma senha\*  
Senha

Confirme sua senha\*  
Repita a senha

**CRIAR CONTA**

Já tem uma conta? [REALIZAR LOGIN](#)

Fonte: Autoria própria, 2022.

Para entrar em uma conta já existente, o usuário insere o seu e-mail e senha utilizados no cadastro. Caso o e-mail seja inválido, aparecerá uma mensagem de erro, ou que o usuário não existe.

**Figura 85 - Tela de erro do login**

**Conheça nossos serviços**

Controle seu lar com apenas alguns cliques

**Atendimento online e automático, tornando sua experiência muito mais dinâmica e objetiva**

**LOGIN**

Usuário não encontrado!

Email

Password

**ENTRAR**

[Esqueceu a senha?](#)

[É novo por aqui? CRIAR CONTA](#)

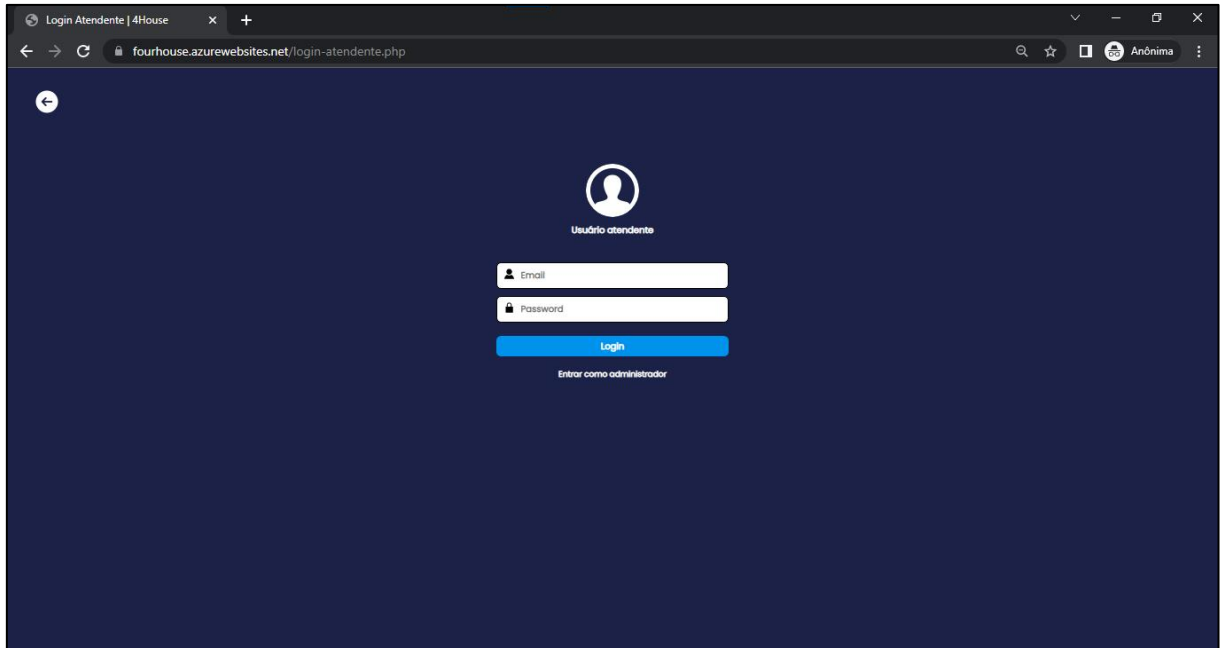
[Entrar como visitante](#)

Fonte: Autoria própria, 2022.

Para entrar no dashboard, que seria o acesso de administrador e de atendente, nesse caso apenas pessoas autorizadas terão acesso a esse canal, diretamente pela URL,

também entrará com e-mail e senha, porém apenas quem tem acesso ao banco de dados consegue realizar o cadastro, tanto de administrador como de atendente.

**Figura 86 - Tela de Login Atendente**

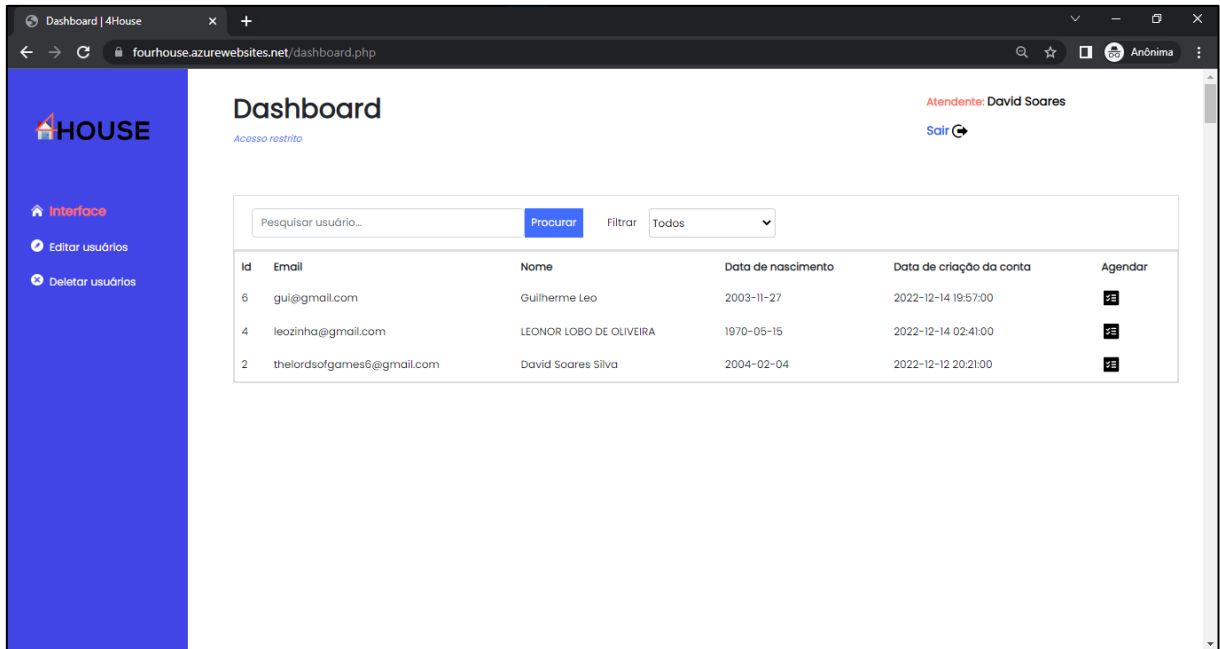


Fonte: Autoria própria, 2022.

O dashboard do atendente, tem acesso a todos os usuários cadastrados, do lado esquerdo são exibidas as seções para editar os usuários, deletar os usuários e agendar serviços. Ao exibir todos os usuários, são divididos em colunas de ID do usuário, que seria o código de identificação de cada usuário no banco de dados, o e-mail que foi usado no cadastro, o nome, a data de criação da conta, a opção de editar o cadastro e a opção de agendar um serviço.

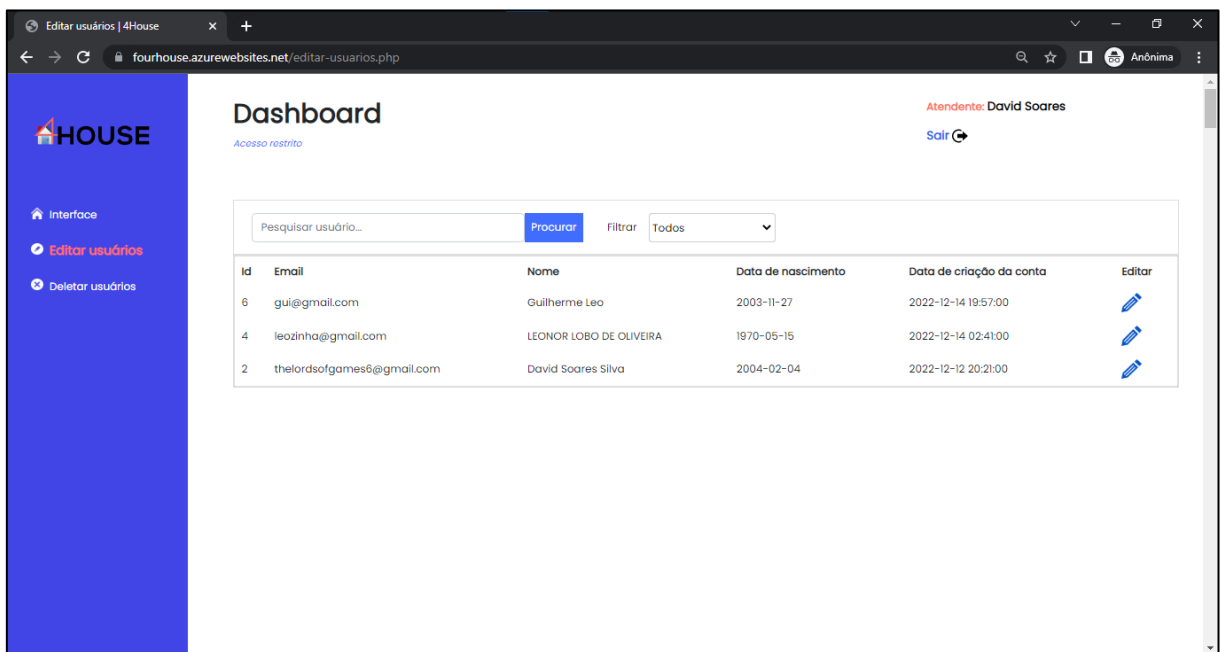
**Figura 87 - Tela Dashboard fourhouse**





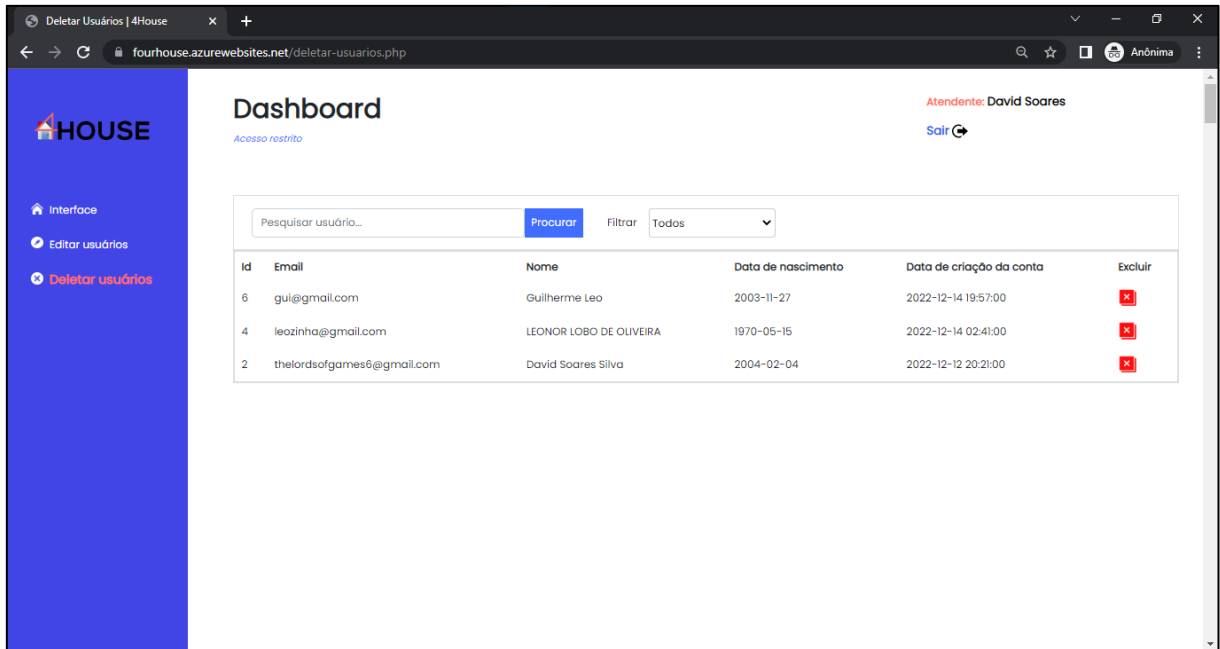
Fonte: Autoria própria, 2022.

**Figura 88 - Tela Edição de Usuários fourhouse**



Fonte: Autoria própria, 2022.

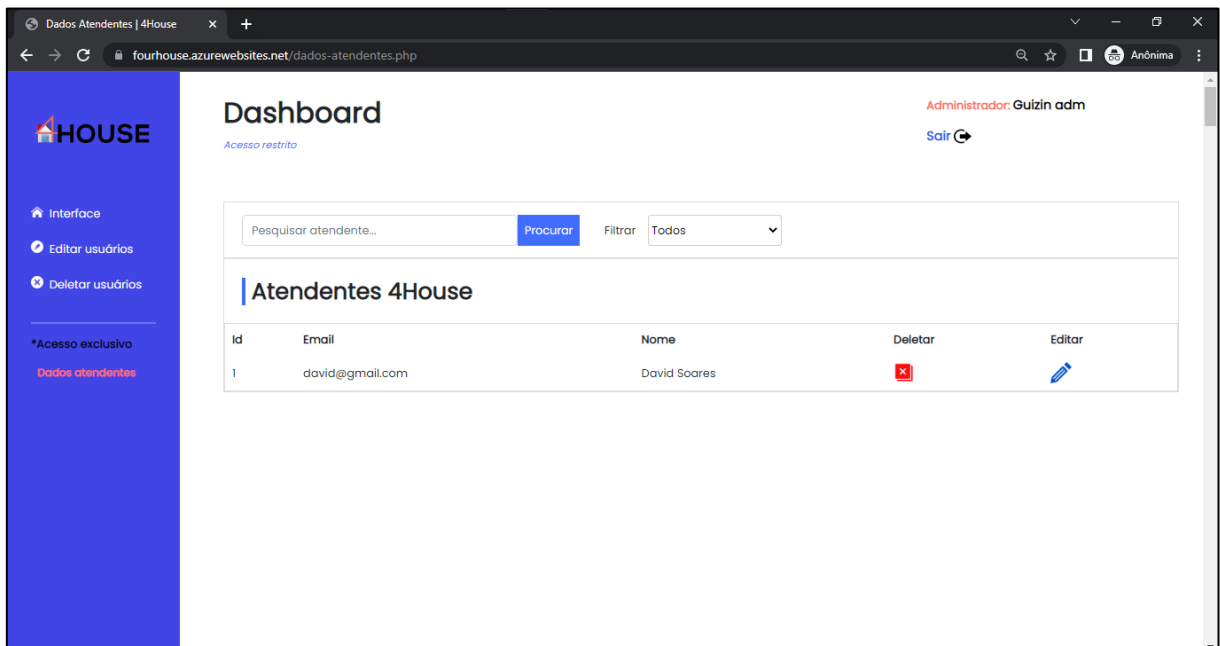
**Figura 89 - Tela Deletar Usuários fourhouse**



Fonte: Autoria própria, 2022.

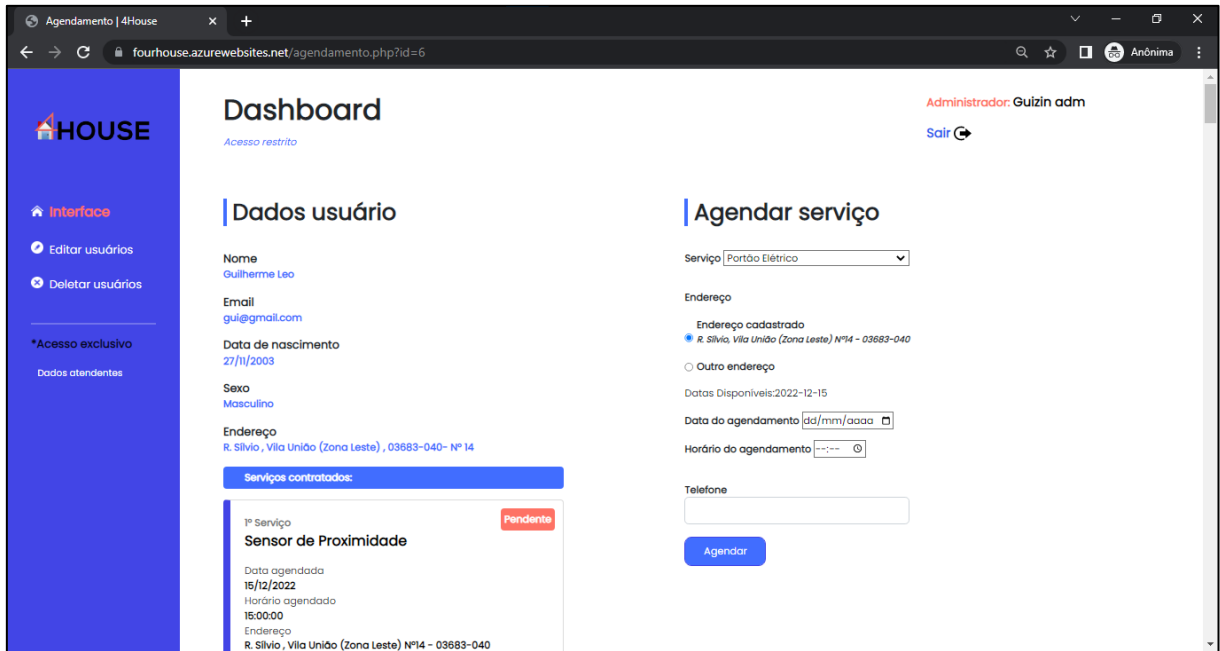
Já o dashboard do administrador, possui todas as funções que o atendente tem com o usuário e mais as mesmas autorizações podem ser usadas para o atendente, por exemplo, o administrador tem autorização de deletar o atendente.

**Figura 90 - Tela Deletar Atendente fourhouse**



Fonte: Autoria própria, 2022.

**Figura 91 - Tela de agendamento**



Fonte: Autoria própria, 2022.

No final de cada tela da aplicação tem um rodapé contendo as principais informações da empresa, como mostra a Figura X.

**Figura 92 - Rodapé fourhouse**



Fonte: Autoria própria, 2022.

### 3.7 Aplicativo Móvel

Este sistema também possui versão *mobile*, com todas as mesmas funcionalidades da aplicação web porém de forma responsiva, ou seja, se adaptando ao *layout* da tela do usuário.

### 3.8 Maquete

Para demonstrar alguns dos serviços, foi feita uma maquete de um condomínio com os serviços de ligar as luzes dos cômodos das casas e levantar os portões das garagens pelo celular, também um sistema de segurança da entrada do condomínio por meio da digitação de uma senha e de iluminação do condomínio por meio da falta de luz do ambiente. Para movimentar a cancela e os portões das garagens foi utilizado

Servo Motor, para digitar a senha para a entrada no condomínio foi utilizado o teclado de membrana, para detectar a luminosidade foi utilizado o sensor LDR, para iluminar os ambientes foi utilizado Leds de auto brilho e um Led RGB e para fazer conexão do celular à casa foi utilizado o Módulo Bluetooth HC-06. Segue as Figuras para demonstração

**Figura 93 – Maquete fourhouse (1)**



Fonte: Autoria própria, 2022.

**Figura 94 - Maquete fourhouse (2)**



Fonte: Autoria própria, 2022.

**Figura 95 - Lateral maquete**



Fonte: Autoria própria, 2022.

**Figura 96 – Maquete fourhouse de noite (1)**



Fonte: Autoria própria, 2022.

**Figura 97 - Maquete fourhouse de noite (2)**



Fonte: Autoria própria, 2022.

**Figura 98 - Maquete de noite fourhouse (3)**



Fonte: Autoria própria, 2022.

**Figura 99 - Maquete fourhouse de noite (4)**





Fonte: Autoria própria, 2022.

**Figura 100 - Maquete fourhouse de noite (5)**



Fonte: Autoria própria, 2022.

## **4 CONCLUSÃO**

Com base nos estudos obtidos durante o desenvolvimento desse projeto, foi possível demonstrar como a automação residencial traz mais conforto e segurança para a rotina dos usuários.

Ainda também tem como comprovar que é uma tecnologia acessível para a população, pois a placa Arduino e seus componentes possuem um valor mais baixo.

Com relação à racionalização de energia, é possível que observar que essa tecnologia serve de grande ajuda para o meio ambiente e o futuro da sociedade mundial.

## 5 REFERÊNCIAS BIBLIOGRÁFICAS

ALVES, Ígor Felipe Rodrigues. **AUTOMAÇÃO RESIDENCIAL COM ARDUINO E ANDROID: Casa Inteligente**, 2014.

BAZZI, Claudio Leones. **Introdução a Banco de Dados**, 2013.

BOOTSTRAP. **Documentação: componentes**. Componentes. 2020b.

CARRIL, MARLY. **HTML – Passo A Passo**, 2012.

COAD, Peter. **Object-Oriented Patterns**, 1992.

CONFORTO, D.; CAVEDINI, P.; MIRANDA, R.; CAETANO, S. **Pensamento computacional na educação básica: interface tecnológica na construção de competências do século XXI**. Revista Brasileira de Ensino de Ciências e Matemática, v. 1, n. 1, 10 ago. 2018.

COSTA, Carlos J. **“Desenvolvimento para Web”**, 2007.

CUNHA, Romualdo Teixeira; ROCHA, Adriano de Oliveira; SOUZA, Diones Silva; OLIVEIRA, Leandro Machado; AMADO, José Alberto Diaz; MARQUES, João Erivando Soares. **IMPLEMENTAÇÃO DE UM DRIVE PARA CONTROLE DE MOTOR CC UTILIZANDO BLUETOOTH E REDES NEURAIAS**, 2017.

DA ROCHA, Mauricio Rêgo Mota. **Geração de testes a partir de máquinas de estados finitos estendidas extraídas de diagramas de sequência UML**, 2021.

DE MELLO, Rafael Maiani. **TÉCNICA PARA INSPEÇÃO DE DIAGRAMAS DE ATIVIDADES**, 2011.

DOMINGUES, Ricardo Gil. **A DOMÓTICA COMO TENDÊNCIA NA HABITAÇÃO: Aplicação em Habitações de Interesse Social com Suporte aos Idosos e Incapacitados**, 2013.

E SOUZA, Glaycow Silveira Silva. **DESENVOLVIMENTO DE UMA FERRAMENTA PARA ANÁLISE E EXTRAÇÃO DE MÉTRICAS DE QUALIDADE DE DIAGRAMAS DE CLASSE UML BASEADO NO QUALITY MODEL OBJECT ORIENTED DESIGN**, 2018.

FERREIRA, Elcio; EIS, Diego. **HTML5: Curso W3C Escritório Brasil**. São Paulo: W3C Escritório Brasil, 2011.



- FERREIRA, Jeferson; MARTINS, Eliane. **Fluxo de Exceções Intraprocedimentais a partir do Diagrama de Atividades da UML 2.0**, 2010.
- FLATSCHAR, Fábio. **HTML5: embarque imediato**, 2011.
- GERMANO, Renan Soares; ELISEO, Maria Amelia; SILVEIRA, Ismar Frango. **Introdução à Acessibilidade na Web: do Conceito à Prática**, 2021.
- GONZAGA, Flávio S.; BIRCKAN, Guilherme. **Curso de PHP e MySQL**, 2000.
- GRILLO, Filipe Del Nero; FORTES, Renata Pontin de Mattos. **Aprendendo JavaScript**, 2008.
- GUEDES, Gilleanes T. A.. **UML 2 uma abordagem prática**, 2011.
- JÚNIOR, José Jair Alves Mendes; JUNIOR, Sérgio Luiz Stevan. **LDR E SENSORES DE LUZ AMBIENTE: FUNCIONAMENTO E APLICAÇÕES**, 2013.
- JUNIOR, Prof. Edwar Saliba. **Diagrama de Caso de Uso. Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro**, 2020.
- JÚNIOR, Sebastião Estefanio Pinto Rabelo. **Verificação de Conformidade entre Diagramas de Sequência UML e Código Java**, 2012.
- MALDONADO, José Carlos; BRAGA, Rosana Teresinha Vaccare; GERMANO, Fernão Stella Rodrigues; MASIERO, Paulo Cesar Masiero. **Padrões e Frameworks de Software**, 2002.
- MCROBERTS, Michael. **Arduino Básico**, 2011.
- MENDES, Aleticiana Generoso. **AUTOMAÇÃO RESIDENCIAL**, 2020.
- MOURA, Gabrielle Fernanda de Arruda. CUNHA, Nycollas Peixoto Nogueira da. **AUTOMAÇÃO RESIDENCIAL**, 2021.
- MUNZLINGER, Elizabete. **CSS – Cascading Style Sheets Sintaxe**, 2011.
- OLIVEIRA, Rháleff Nascimento Rodrigues de; CARDOSO, Rodrigo Pennella; BRAGA, Juliana Cristina; CAMPOS, Rafaela Vilela da Rocha. **Frameworks para Desenvolvimento de Jogos Educacionais: uma revisão e comparação de pesquisas recentes**, 2018.
- ORTIZ, Luiz Henrique Oliveira. **SISTEMA DE AUTOMAÇÃO RESIDENCIAL COM ÊNFASE EM SEGURANÇA E ECONOMIA ENERGÉTICA**, 2018.

PEIXOTO, Anderson Gomes. **O USO DE METODOLOGIAS ATIVAS COMO FERRAMENTA DE POTENCIALIZAÇÃO DA APRENDIZAGEM DE DIAGRAMAS DE CASO DE USO**, 2016.

PHP Documentation Group. **O que é o PHP?**, 2022.

PIRES, Fábio de Oliveira. **SISTEMA DE CONTROLE PARA FECHADURAS DE PORTAS COM O ARDUINO**, 2020

POMPILHO, S. **Análise Essencial Guia Prático de Análise de Sistemas**. Rio de Janeiro: Ed. Ciência Moderna Ltda, 1995.

MENDONÇA, Ricardo Augusto Ribeiro de. **Levantamento de requisitos no desenvolvimento ágil de Software**, 2014.

RABELO, Daniel Ferreira; CÂNDIDO, Marco Vinicius Isecke. **ANÁLISE DE DESEMPENHO DE BANCO DE DADOS NOSQL EM UM SISTEMA QUE UTILIZA UM BANCO DE DADOS RELACIONAL E NÃO RELACIONAL PARA ARMAZENAMENTO DE DADOS**, 2017.

RIBEIRO, M. C. D.; TIOSSO, F.; PETRUCELLI, E. E. **LIMITAÇÕES DE UM SISTEMA DE GERENCIAMENTO DE BANCO DE DADOS EM MEIO A UM MODELO TRANSACIONAL**. Revista Interface Tecnológica, [S. l.], v. 16, n. 1, p. 102–113, 2019. Disponível em: <https://revista.fatectq.edu.br/interfacetecnologica/article/view/564>.

ROCHA, Helder Lima Santos. **Desenvolvendo Web Sites Interativos com JavaScript**, 1999.

RODRIGUES, Samuel da Costa. **ESTUDO E IMPLEMENTAÇÃO DE INTERFACES WEB EM HTML5**, 2014.

ROSA, Luis Henrique Carvalho; LUCCA, Luísa Perin; LEMOS, Eduardo Luis; BERNARDI, Giliane; MEDINA, Roseclea Duarte. **Jogos para Ensino de Levantamento de Requisitos de Software: uma Revisão Sistemática de Literatura**, 2017.

SILVA, Bruno Rodrigues; VALIM, Paulo Roberto Oliveira. **Kit Modular de Desenvolvimento Baseado em Microcontrolador Pic**, 2011.

SILVA, Luis Augusto Lopes; GONZAGA, Luis Eduardo Lima; ROCHA, Paloma Rangel; LUCAS, Sarah de Souza Ribeiro. **SCRATCH OUT: Gerenciador de Tarefas**, 2018.

SILVA, Maurício Samy. **HTML5: A linguagem de marcação que revolucionou a web**, 2019.

SIQUEIRA, Déborados Santos e; PAULON, Matheus Montanha; GUEDES, Gilleanes Thorwald Araujo. **Técnicas de Inspeção para Diagramas de Classes UML: Uma Revisão Sistemática**. In: **ESCOLA REGIONAL DE ENGENHARIA DE SOFTWARE (ERES)**, 3. , 2019, Rio do Sul. Anais [...]. Porto Alegre: Sociedade Brasileira de Computação, 2019 . p. 41-48.

SOUZA, E. C.; OLIVEIRA, M. R. de. **COMPARATIVO ENTRE OS BANCOS DE DADOS MYSQL E MONGODB: quando o MongoDB é indicado para o desenvolvimento de uma aplicação**. Revista Interface Tecnológica, [S. l.], v. 16, n. 2, p. 38–48, 2019. DOI: 10.31510/infa.v16i2.664. Disponível em: <https://revista.fatectq.edu.br/interfacetecnologica/article/view/664>.

SOUZA, Felipe Walflan de. **DESENVOLVIMENTO DE ARQUITETURAS CSS APLICADO EM JOGOS WEB**, 2018.

SOUZA, L. P.; ESPÍRITO SANTO, F. do. **COMPARATIVO ENTRE FRAMEWORKS DE CSS BOOTSTRAP E BULMA PARA DESENVOLVIMENTO DE PROJETOS WEB**. Revista Interface Tecnológica, [S. l.], v. 17, n. 1, p. 140–152, 2020. DOI: 10.31510/infa.v17i1.785, 2022.

TÓFOLI, Ricardo José. **CASA INTELIGENTE – SISTEMA DE AUTOMAÇÃO RESIDENCIAL**, 2014.

TORRES, V. M. **HTML e seus Componentes**. Revista Ada Lovelace, [S. l.], v. 2, p. 99–101, 2018.