**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 por 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 residencias 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](#_Toc119444800)

[Figura 2 – Cadastro de Usuário em HTML 14](#_Toc119444801)

[Figura 3 - Cadastro de Usuário em HTML (2) 14](#_Toc119444802)

[Figura 4 - Página WebCadastro de Usuário 15](#_Toc119444803)

[Figura 5 – Sintaxe do CSS 16](#_Toc119444804)

[Figura 6 - Código do CSS estilizando o HTML 16](#_Toc119444805)

[Figura 7 – Linha do código em HTML mudada para ser estilizado no CSS 18](#_Toc119444806)

[Figura 8 - Página Web Cadastro de Usuários com CSS 18](#_Toc119444807)

[Figura 9 - Exemplo de utilização básica do JavaScript 20](#_Toc119444808)

[Figura 10 - Operadores Aritméticos 20](#_Toc119444809)

[Figura 11 - Operadores de Comparação 21](#_Toc119444810)

[Figura 12 - Operadores Lógicos 21](#_Toc119444811)

[Figura 13 - Exemplo if e else 21](#_Toc119444812)

[Figura 14 - Exemplo while 22](#_Toc119444813)

[Figura 15 - JS incluído no HTML com a tag script 22](#_Toc119444814)

[Figura 16 - JS importado pelo HTML de um arquivo externo (1) 22](#_Toc119444815)

[Figura 17 - JS importado pelo HTML de um arquivo externo (2) 23](#_Toc119444816)

[Figura 18 - JS incluído em descritores HTML sensíveis a eventos 23](#_Toc119444817)

[Figura 19 - Resultado dos códigos das imagens 15, 16, 17 e 18 após clicar no botão enviar 23](#_Toc119444818)

[Figura 20 - Adcionando JS na página de Cadastro 23](#_Toc119444819)

[Figura 21 - Resultado da Página de Cadastro com JS 24](#_Toc119444820)

[Figura 22 - Página de Cadastro sem JS 24](#_Toc119444821)

[Figura 23 - Declaração de Variável em PHP 26](#_Toc119444822)

[Figura 24 – Cadastro de Usuário com HTML, CSS, JS e PHP (1) 26](#_Toc119444823)

[Figura 25 - Cadastro de Usuário com HTML, CSS, JS e PHP (2) 26](#_Toc119444824)

[Figura 26 - Cadastro de Usuário com HTML, CSS, JS e PHP (3) 27](#_Toc119444825)

[Figura 27 - Figura 26 - Cadastro de Usuário com HTML, CSS, JS e PHP (4) 28](#_Toc119444826)

[Figura 28 - Página Web com HTML, CSS, JS e PHP (1) 28](#_Toc119444827)

[Figura 29 - Página Web com HTML, CSS, JS e PHP (2) 29](#_Toc119444828)

[Figura 30 - Página Web com HTML, CSS, JS e PHP (3) 30](#_Toc119444829)

[Figura 31 - Cadastro de Usuário com HTML, CSS, JS, PHP e Bootstrap (1) 31](#_Toc119444830)

[Figura 32 - Cadastro de Usuário com HTML, CSS, JS, PHP e Bootstrap (2) 32](#_Toc119444831)

[Figura 33 - Como pegar códigos do Bootstrap 32](#_Toc119444832)

[Figura 34 - Página Web com HTML, CSS, JS, PHP e Bootstrap 33](#_Toc119444833)

[Figura 35 - Exemplo Diagrama de Caso de Uso 36](#_Toc119444834)

**Lista de Abreviações**

*Cascading Style Sheet* (CSS)

*European Council for Nuclear Research* (CERN)

*HyperText Markup Language* (HTML)

*Personal Computers* (PCs)

*UCS Transformation Format 8* (UTF-8)

*World Wide Web Consortium* (W3C)

*JavaScript* (JS)

*Hypertext Preprocessor* (PHP)

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

*Object Modeling Technique* (OMT)

*Object-Oriented Software Engineering* –(OOSE)

*Unified Modeling Language* (UML)

*Uniform Resource Locator* (URL)

SUMÁRIO

[1 INTRODUÇÃO 11](#_Toc119753694)

[2 REFERENCIAL TEÓRICO 10](#_Toc119753695)

[2.1 HTML 12](#_Toc119753696)

[2.2 CSS 15](#_Toc119753697)

[2.3 JavaScript 18](#_Toc119753698)

[2.4 PHP 25](#_Toc119753699)

[2.5 Framework 30](#_Toc119753700)

[2.6 Bootstrap 31](#_Toc119753701)

[2.7 Banco de Dados 33](#_Toc119753702)

[2.8 UML 33](#_Toc119753703)

[2.8.1 Análise e Levantamento de Requisitos 34](#_Toc119753704)

[2.8.2 Diagrama de Casos de Uso 35](#_Toc119753705)

[2.8.3 Diagrama de Sequência 37](#_Toc119753706)

[2.8.4 Diagrama de Atividades 37](#_Toc119753707)

[2.8.5 Diagrama de Classe 37](#_Toc119753708)

[2.9 C++ 37](#_Toc119753709)

[2.10 Arduino 37](#_Toc119753710)

[3 METODOLOGIA 37](#_Toc119753711)

[4 DESENVOLVIMENTO 38](#_Toc119753712)

[4.1 Diagrama de Casos de Uso 38](#_Toc119753713)

[4.2 Diagrama de Sequência 38](#_Toc119753714)

[4.3 Diagrama de Atividades 38](#_Toc119753715)

[4.4 Diagrama de Classe 38](#_Toc119753716)

[4.5 Banco de Dados 38](#_Toc119753717)

[4.5.1 DER 39](#_Toc119753718)

[4.5.2 MER 39](#_Toc119753719)

[4.6 Sistema Web 39](#_Toc119753720)

[4.7 Aplicativo Móvel 40](#_Toc119753721)

[5 CONCLUSÃO 40](#_Toc119753722)

[6 REFERÊNCIAS BIBLIOGRÁFICAS 41](#_Toc119753723)

# 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.

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).

# 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.

## 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 – Estrutura Básica do HTML

Texto

Descrição gerada automaticamente

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 charsert 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 – Cadastro de Usuário em HTML

Texto

Descrição gerada automaticamente

Fonte: autoria própria, 2022.

Figura - Cadastro de Usuário em HTML (2)

Texto

Descrição gerada automaticamente

Fonte: autoria própria, 2022.

Figura - Página WebCadastro de Usuário

Interface gráfica do usuário

Descrição gerada automaticamente com confiança baixa

Fonte: autoria própria, 2022.

## 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 – Sintaxe do CSS

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

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 - Código do CSS estilizando o HTML

Tela preta com letras brancas

Descrição gerada automaticamente

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



Fonte: Autoria própria, 2022.

Figura - Página Web Cadastro de Usuários com CSS

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

## 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 Européia 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 - Exemplo de utilização básica do JavaScript

Texto

Descrição gerada automaticamente

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 - Operadores Aritméticos

Tabela

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

Figura - Operadores de Comparação

Tabela

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

Figura - Operadores Lógicos

Tabela

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

Figura - Exemplo if e else

Texto

Descrição gerada automaticamente

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 - Exemplo while

Uma imagem contendo Texto

Descrição gerada automaticamente

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 - JS incluído no HTML com a tag script

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

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

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

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

Texto

Descrição gerada automaticamente com confiança baixa

Fonte: Autoria própria, 2022.

Figura - JS incluído em descritores HTML sensíveis a eventos



Fonte: Autoria própria, 2022.

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

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

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 - Adcionando JS na página de Cadastro

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

Figura - Resultado da Página de Cadastro com JS

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

Figura - Página de Cadastro sem JS

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

## 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, e é fácil de aprender. O objetivo principal da linguagem é permitir a desenvolvedores escreverem páginas que serão geradas dinâmica 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 PHP 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 (<https://www.php.net/manual/pt_BR/intro-whatis.php>, documentação oficial do PHP).

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 - Declaração de Variável em PHP

Texto

Descrição gerada automaticamente com confiança média

Fonte: Autoria própria, 2022.

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

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

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

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

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

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

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

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

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

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

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

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

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

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

Interface gráfica do usuário

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

## 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).

## Bootstrap

Criado em 2010 pelo designer do Twitter Mark Otto e pelo desenvolvedor Jaocb 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 jQuerys (BOOTSTRAP, 2015).

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

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

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

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

Figura - Como pegar códigos do Bootstrap

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

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

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

Tela de computador com texto preto sobre fundo branco

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

## Banco de Dados

## 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 métods 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).

### 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.

### 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).

Vide exemplo na imagem 35.

Figura - Exemplo Diagrama de Caso de Uso

Diagrama

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

### **Diagrama de Sequência**

### **Diagrama de Atividades**

### **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).

## C++

## Arduino

# METODOLOGIA

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).

# 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.

## Diagrama de Casos de Uso

## Diagrama de Sequência

## Diagrama de Atividades

## Diagrama de Classe

## Banco de Dados

### **DER**

### **MER**

## Sistema Web

A primeira parte que o Usuário tem contato ao acessar o sistema é a tela de acesso que dispõe de quatro funcionalidades que são: acesso ao sistema como usuário, acesso ao sistema *Dashboard* como administrador ou atendente, criação de conta de usuário e recuperação de senha. Essas funcionalidades são integradas com o banco de dados mySQL.

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.

Na tela de login o usuário tem as seguintes opções, entrar em uma conta já existente, criar uma nova conta ou alterar a senha.

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.

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.

Para redefinir a senha, será feito um questionário a respeito do cadastro realizado para conferir que se refere ao titular da conta, como por exemplo, data de nascimento cadastrada, endereço cadastrado. Após essa confirmação, o usuário irá entrar em uma tela solicitando a criação da nova senha.

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.

O dashboard do atendente, tem acesso a todos os usuários cadastrados, exibindo os seus dados, possibilitando editar o seu perfil.

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.

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.

## Aplicativo Móvel

Nosso 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.

# CONCLUSÃO

# REFERÊNCIAS BIBLIOGRÁFICAS

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

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

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

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

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

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

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.

JUNIOR, Prof. Edwar Saliba. **Diagrama de Caso de Uso. Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro**, 2020.

MALDONADO, José Carlos; BRAGA, Rosana Teresinha Vaccare; GERMANO, Fernão Stella Rodrigues; MASIERO, Paulo Cesar Masiero. **Padrões e Frameworks de Software**, 2002.

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.

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.

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, 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, 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.