**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA**
**ETEC ZONA LESTE**
**Ensino Médio com Habilitação Profissional de Técnico em Desenvolvimento de Sistemas - AMS**

**Caike Lins Ferreira**

**Gabriel Diniz Silva**

**Giovanna Niz Paes Santana**

**Gustavo Polastrini da Cruz**

**João Vitor de Oliveira dos Santos**

# Search

**São Paulo**
**2022**

**Caike Lins Ferreira**

**Gabriel Diniz Silva**

**Giovanna Niz Paes Santana**

**Gustavo Polastrini da Cruz**

**João Vitor de Oliveira dos Santos**

# Search

Trabalho de Conclusão de Curso apresentado ao Curso do Ensino Médio com Habilitação Profissional de Técnico em Desenvolvimento de Sistemas AMS da Etec Zona Leste, orientado pela Professora Vilma Cardoso dos Santos, como requisito final para obtenção do título de Técnico em Desenvolvimento de Sistemas.

**São Paulo**
**2022**

## DEDICATÓRIA

Dedicamos este projeto aos nossos colegas, que desejam encerrar essa etapa assim como nós.

Dedicamos a Professora Vilma Cardoso, que esteve nos acompanhando e motivando, desde o primeiro ano.

A Jhonata Peres de Andrade Cardoso.

Para Rafael de Alcântara França.

**AGRADECIMENTOS**

Gostaríamos de agradecer a

Professor Rogério Bezerra Costa,

Professor Ediney Ciasi Barreto,

Professora Vilma Cardoso dos Santos,

Professor Jeferson Roberto de Lima,

Professor Wagner de Oliveira Lucca.

## EPÍGRAFE

“Ninguém ignora tudo. Ninguém sabe tudo. Todos nós sabemos alguma coisa. Todos nós ignoramos alguma coisa. Por isso aprendemos sempre.”

PAULO FREIRE

## RESUMO

Neste projeto propomos ajudar e facilitar o lazer dos nossos usuários, para que, através da nossa aplicação, possam encontrar caminhos de assistir séries e filmes de uma forma mais fácil e rápida, afim de que o cliente sinta mais facilidade ao utilizar do seu tempo de lazer, pois assistir filmes e séries reduz seu nível de cortisol, protegendo de riscos cardiovasculares, doenças como o Alzheimer e também diminuindo seu risco de ganho de peso e as consequências que esse excesso pode trazer para sua saúde. Assim, através das tecnologias apresentadas abaixo, tais como: JavaScript, PHP, HTML, MySQL, CSS, XML a aplicação irá receber os dados solicitados pelo utilizador e irá gerir as opções de site para uso do cliente.



**Palavras-chave**: Filmes. Séries. Conteúdos Audiovisuais. Pesquisar.

## ABSTRACT

*In this project we propose to help and facilitate the leisure of our users, so that, through our application, they can find ways to watch series and movies in an easier and faster way, so that the client feels easier to use his free time, counting that watching movies and series reduces his cortisol level, protecting him from cardiovascular risks, diseases like Alzheimer and also decreasing his risk of weight gain and the consequences that this excess can bring to his health. Thus, through the technologies presented below, such as: JavaScript, PHP, HTML, MySQL, CSS, XML, the application will receive the data requested by the user and manage the site's options for the client's use.*



***Keywords**: Movies. Series. Audiovisual Contents. Search.*

# LISTA DE ILUSTRAÇÔES

## LISTA DE ABREVIATURAS E SIGLAS

Banco de Dados (BD)

*Cascading Style Sheets* (CSS)

*Database Management System* (SBDG)

Estados Unidos da América (EUA)

*Extensible HyperText Markup Language* (XHTML)

*Extensible Markup Language* (XML)

*Hypertext Markup Language* (HTML)

*Hypertext Preprocessor* (PHP)

Interface de Programação de Aplicações (API)

JavaScript (JS)

Prompt de Comando (CMD)

*Unified Modeling Language* (UML)

*Uniforme Resource Locator* (URL)

# SUMÁRIO

# 1 INTRODUÇÃO

O primeiro grande serviço de *streaming* chegou ao Brasil em 2011, com ele também veio um mundo de oportunidades. No cenário atual, as plataformas digitais vêm crescendo consideravelmente, assim como o consumo delas. Com tantas opções, acabamos diversas vezes, sem saber o que fazer ou escolher o que consumir.

De acordo com pesquisas, atualmente existem mais de 60 serviços de streaming disponíveis somente no nosso país. Muitos conteúdos como filmes e séries hospedados nesses serviços de streaming, trocam regularmente de plataforma de transmissão.

Dado a isso, onde um indivíduo pode encontrar determinados conteúdos, sem infringir as leis e diretrizes do nosso país? Com base nessa problemática, nosso projeto visa simplificar a pesquisa e a localização de tais conteúdos. Através de uma plataforma online de nossa autoria, será indicado o conteúdo completamente livre de cópias não autorizadas.

Com base em estudos, diagnosticamos e fundamentamos uma problemática. Através desse estudo, solucionamos o problema com o uso de tecnologias de desenvolvimento e programação, com o intuito de alcançar o objetivo de facilitar a localização de filmes, séries, entre outros.

# 2 REFERENCIAL TEÓRICO

Nesta seção, apresentaremos as tecnologias que serão utilizadas em nosso projeto.

## 2.1 HTML

Criado na década de 90, por Tim Berners-Lee, *HyperText Markup Language* (HTML) é uma linguagem de marcação.

De acordo com Eis (2012), o HTML5 é uma nova versão, atualizada em 2010, que tem como principal objetivo trazer facilidade ao manipular os elementos da página. Além da facilidade para manipular o código, também oferece um melhor desempenho para as linguagens, como CSS e JavaScript.

Segundo Ferreira (2013), toda página web possui variados conteúdos, desde imagens á tabelas e listas. A criação dos elementos é definida através de *tags* específicas dentro da estrutura básica do HTML como mostra a Figura 1.

**Figura 1 – Estrutura Básica HTML**

```
<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, inicitial-scale=1.0">
        <title></title>
    </head>
    <body>

    </body>
</html>
```

Fonte: Autoria Própria, 2022

Segundo Duckett (2014), essas são as definições de algumas tags principais.

- Doctype: declara ao navegador qual versão do HTML será usada no arquivo;
- *HTML*: demarca o começo do uso da linguagem. Juntamente com o atributo *lang,* torna-se capaz de definir a linguagem principal do arquivo;
- *Head*: sua função é definir o cabeçalho da página;

- *Meta charset*: faz com que o navegador saiba qual a codificação dos elementos dessa página web;
- *Meta name*: descreve o conteúdo de uma página web, mostrando-o na URL da página e nos mecanismos de pesquisa;
- *Title*: define o título da página, que é exibido na barra de título do navegador;
- *Body*: determina o “corpo” do documento, portanto, tudo dentro dela é apresentado como conteúdo da página;
- *Center*: tudo dentro desta tag será centralizado.
- *P*: essa tag é chamada para definir um parágrafo;
- *Fieldset*: aplica uma borda ao redor do formulário;
- *Form*: cria um formulário;
- *Div*: é usado para agrupamento de conteúdo, geralmente, usado para auxiliar na estilização;
- *Input*: define um campo de entrada de dados, junto com o atributo *type;*
- *Required*: faz com que seja obrigatório o conteúdo ser preenchido;
- *Label*: é utilizada para nomear um atributo.

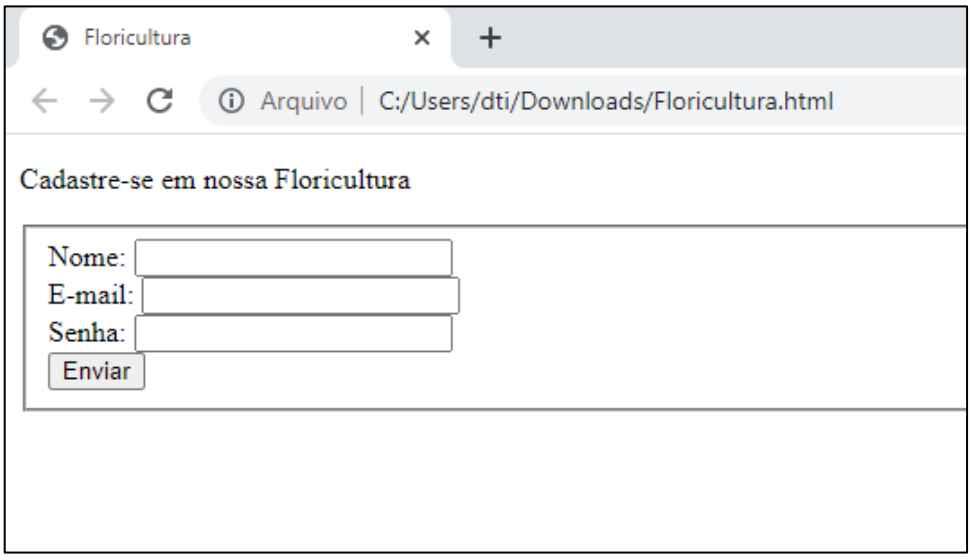
Podemos visualizar estas tags sendo utilizadas na Figura 2, em código e na Figura 3, na web.

**Figura 2 – Cadastro de Usuário HTML**

```html
<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, inicitial-scale=1.0">
        <title>Floricultura</title>
    </head>
    <body>
        <center>
        <p>Cadastre-se em nossa Floricultura</p>
        <fieldset>
            <form action="" method="post">
                <div>
                    <label for="name">Nome:</label>
                    <input type="text" id="nome" required/>
                </div>
                <div>
                    <label for="email">E-mail:</label>
                    <input type="email" id="email" required/>
                </div>
                <div>
                    <label for="senha">Senha:</label>
                    <input type="password" id="senha" required/>
                </div>
                <div>
                    <input type="submit" name= "submit" id="submit" required/>
                </div>
            </form>
        </fieldset>
        </center>
    </body>
</html>
```

Fonte: Autoria própria, 2022

**Figura 3 – Tela de Cadastro HTML**




Fonte: Autoria própria, 2022

## 2.2 CSS

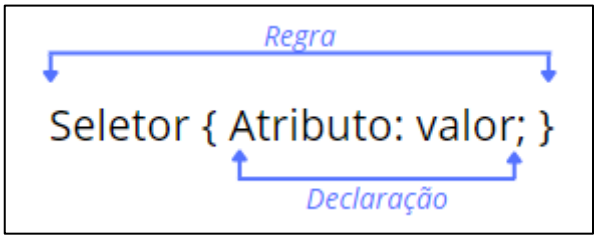
De acordo com Jobstraibizer (2009), o *Cascading Style Sheets* (CSS), serve para construir o layout de seu site ou páginas. É utilizado para definir o aspecto de documentos escritos em uma linguagem de marcação, como HTML e/ou XML.

Regra de estilo é uma pequena porção de código que pode causar efeito de estilização. Segundo Silva (2008), a regra é composta por três partes: seletores, atributos e valores para os atributos.

- Seletor – define o lugar da marcação que será estilizada;
- Atributo – a característica que deseja configurar;
- Valor – define qual valor o atributo receberá.

A Figura 4 exemplifica uma regra CSS.

**Figura 4 – Regra de Estilo**




Fonte: Autoria Própria, 2022

Conforme Scheidt (2015), o CSS pode ser aplicado a uma página HTML, dentro da tag <head>, atráves do elemento <style>. Como podemos ver na Figura 5, em código, e na Figura 6, em web.

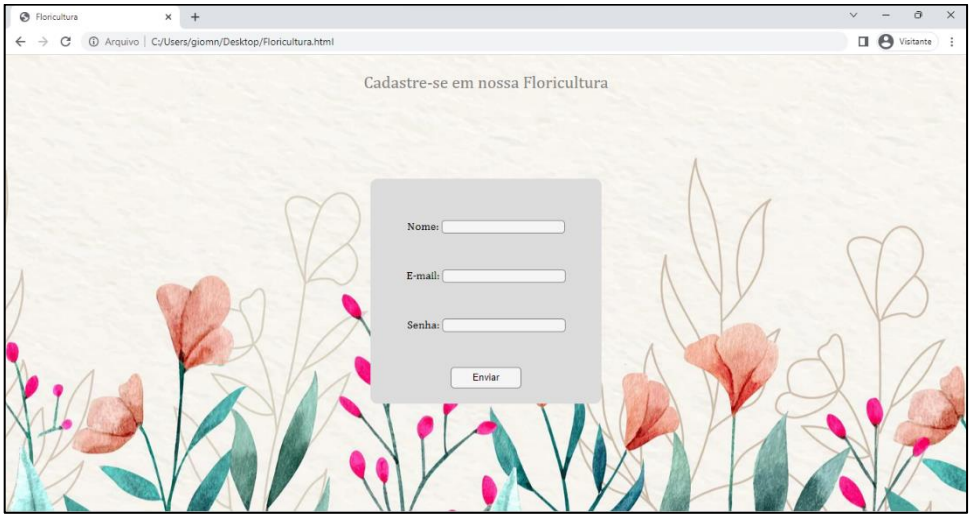
**Figura 5 – Tela de Cadastro com CSS**

```
<!DOCTYPE html>
<html lang="pt-br">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, inicitial-scale=1.0">
        <title>Floricultura</title>
        <style>
            body{
                background-image:url("Flores.png");
                background-position: center;
            }
            p{
                font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;
                font-size: x-large; color: gray; padding-bottom: 100px;

            }
            fieldset{
                background-color: gainsboro; border-radius: 10px; border-color: transparent;
                width: 300px; height: 300px;
            }
            div{
                font-family: Cambria, Cochin, Georgia, Times, 'Times New Roman', serif;
                padding-top: 50px;
            }
            input{
                background-color: whitesmoke; border-radius: 5px; border: 1px solid gray;
            }
            #submit{
                width: 100px; height: 30px; background-color: whitesmoke; border: 1px solid gray;
            }
        </style>
    </head>
```

Fonte: Autoria Própria, 2022

**Figura 6 – Página Estilizada**




Fonte: Autoria própria, 2022

Os elementos utilizados, segundo Duckett (2014), significam:

- *Background*: Com suas variações permite modificar o plano de fundo, como *background-color,* para mudar a cor, *background-image,* para adicionar uma imagem e b*ackground-position,* que define a posição do plano de fundo.
- *Font-size*: Aplica certo tamanho a fonte.
- *Font-family*: capacita a mudança de fonte.
- *Padding*: especifica o tamanho do espaçamento entre os elementos e em que direção ele deve estar, como: *padding-top* e *padding-bottom* que definem, respectivamente, o espaço superior e inferior.
- *Color*: Muda a cor do elemento.
- *Border*: Especifica o tamanho e cor da borda, podendo também, através do *border-radius,* arredondar as bordas.
- *Width* e *height*: Permite mudar a largura e altura.
- Background-color: Muda o fundo para uma determinada cor.
- #: Usado na parte do seletor, serve para convocar um ID específico.

## 2.3 JAVASCRIPT

O *Javascript* (JS) é uma linguagem de programação Web, usada por grande parte dos sites modernos, criada pela *Netscape* em parceria com a *Sun Microsystems* com o intuito de adicionar interatividade as páginas web, conforme Silva (2010).

Segundo Groner (2019), JavaScript pode criar pequenos programas embutidos nos próprios códigos HTML, além disso, também é capaz de gerar números, processar alguns dados, realizar a verificação de formulários, alterar e criar elementos.

O JS é uma linguagem que fica ao lado usuário, ou seja, todo o script é executado no navegador do próprio usuário, diferente de algumas linguagens, como o PHP, que precisam de uma máquina remota que interprete e faça o programa funcionar.

Segundo Morrison (2008), esses são alguns dos comandos usados no JS.

- *Script*: é usado para incluir ou referenciar um roteiro executável;
- *Function()*: são blocos que definem instruções especificas.
- *Var*: cria uma variável;
- *Prompt()*: exibe uma mensagem, opcional, solicitando que algum texto seja inserido;

- *If()*: adiciona uma condição;
- !=: significa diferente, usado para diferenciar condições;
- *Alert*: mostra ao usuário uma mensagem e um botão de confirmação de que visualizou recado;

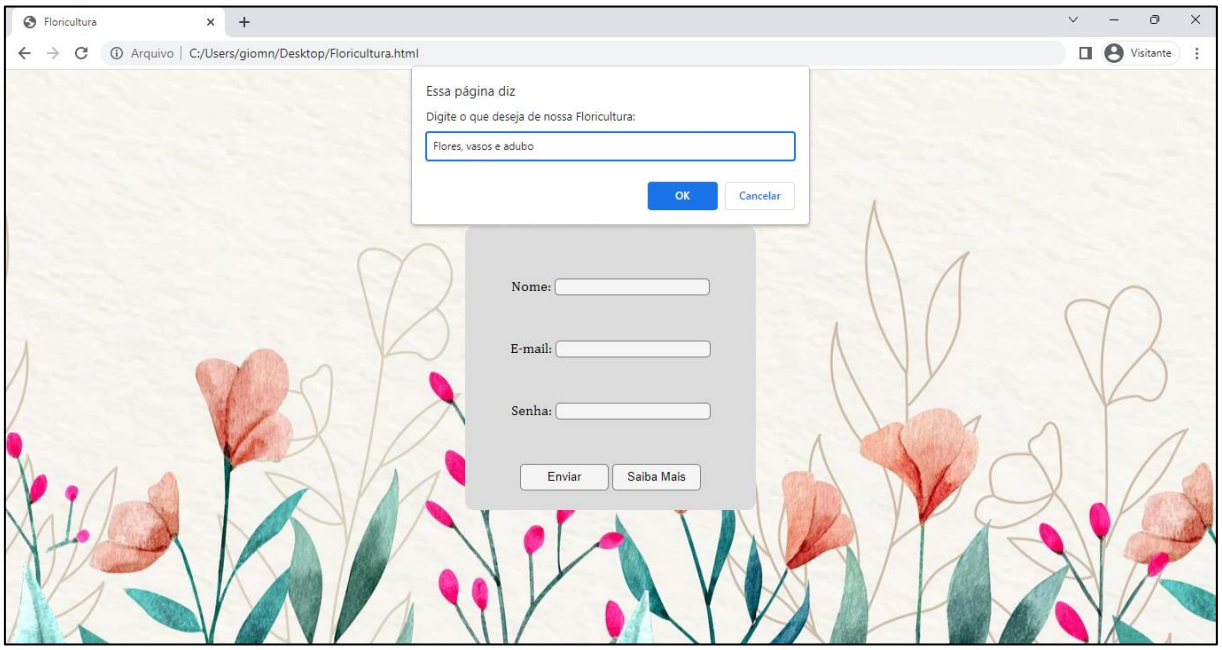
Tais comandos foram implementados nos exemplos apresentados anteriormente e podem ser vistos nas Figura 7, 8 e 9, sendo chamados por um botão presente dentro do formulário.

**Figura 7 – Caixas de Diálogo com JS**

```
36      <body>
37          <center>
38          <p>Cadastre-se em nossa Floricultura</p>
39          <fieldset>
40              <form action="" method="post">
41                  <div>
42                      <label for="name">Nome:</label>
43                      <input type="text" id="nome" required/>
44                  </div>
45                  <div>
46                      <label for="email">E-mail:</label>
47                      <input type="email" id="email" required/>
48                  </div>
49                  <div>
50                      <label for="senha">Senha:</label>
51                      <input type="password" id="senha" required/>
52                  </div>
53                  <div>
54                      <input type="submit" name= "submit" id="submit" required/>
55                      <button onclick="myFunction()">Saiba Mais</button>
56                  </div>
57              </form>
58          </fieldset>
59          <script>
60              function myFunction(){
61
62                  var floric=prompt("Digite o que deseja de nossa Floricultura:");
63
64                  if (floric!=null){
65                      alert("Mostraremos mais conteúdos sobre: "+floric+".");
66                  }
67              }
68          </script>
```

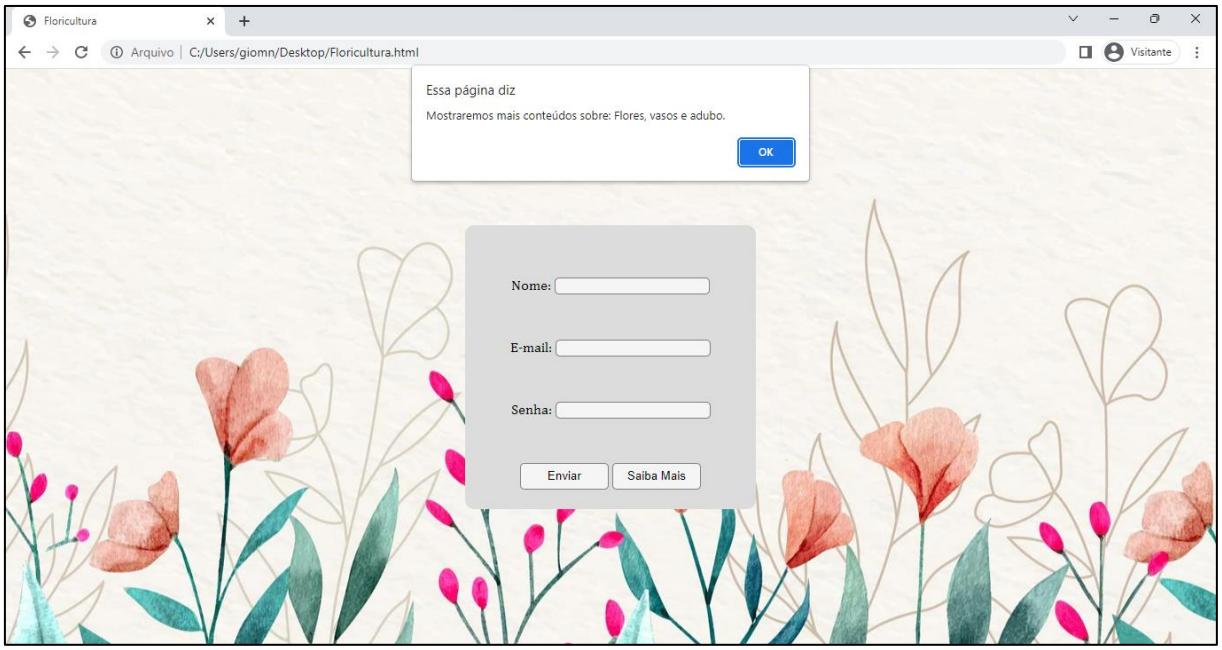
Fonte: Autoria própria, 2022

**Figura 8 – Prompt**




Fonte: Autoria própria, 2022

**Figura 9 – Alert**




Fonte: Autoria própria, 2022

## 2.4 PHP

Segundo Converse e Park (2003), *Personal Home Page Tools*, mais conhecido como PHP, é o produto de um projeto criado em 1994 por Rasmus Lerdof, a fim de criar uma linguagem de programação em binário em linguagem de programação C, para facilitar o acompanhamento de visitas para currículos online.

Segundo Achour, Betz, Dovgal, Lopes, Magnusson, Richter, Seguy e Vrana (1997), em junho de 1995, Rasmus permitiu que o código fosse utilizado pelo público, o que fez com que muitos desenvolvedores fornecessem correções de erros e muitas melhorias para aperfeiçoar o código. Essa grande demanda por melhorias, sugeridas por usuários fez com que Lerdof reescrevesse o código com uma implementação rica em funcionalidades, assim como a novidade da conexão com o banco de dados e uma grande estrutura pra que os usuários pudessem desenvolver aplicações web de uma forma muito mais dinâmica e simples.

Atualmente, o PHP está na sua versão 7, lançada em dezembro de 2015, tendo como grande destaque sua grande rapidez e performance comparadas as últimas versões, o PHP7 está quase 9 vezes mais ágil que as atualizações anteriores.

### Conexão com Banco de dados

Segundo a própria documentação do PHP, este arquivo serve para conectar o banco de dados com a aplicação, dando funcionalidade às funções que precisam utilizar um banco de dados como cadastrar usuários, produtos etc. Como podemos observar nas Figuras 10 e 11.

**Figura 10 – Conexão com Banco de Dados**

```php
<?php
    $hostname = "localhost";
    $bancodedados = "exemplo_tcc";
    $usuario = "root";
    $senha = "";

    $mysqli = new mysqli($hostname, $usuario, $senha, $bancodedados);
    if ($mysqli->connect_errno) {
        echo "falha ao conectar:(" . $mysqli->connect_errno . ")" . $mysqli->connect_errno;
    }
    else
        echo "Conectado ao banco de dados";
?>
```

Fonte: Autoria Própria, 2022

**Figura 11 - Conectado ao banco de dados**


Fonte: Autoria Própria, 2022

Segundo Dall'Oglio (2009), essas são as definições de algumas tags do PHP:

- *Php*: define o começo de arquivos em php.
- *If*: com seu complemento "if(isset($_POST['submit']))", verifica se variável está definida, respondendo com verdadeiro ou falso;
- *Print_r*: tem o objetivo de imprimir ou exibir as informações armazenadas nas variáveis;
- *Include_once*: evita que o mesmo arquivo possa ser validado mais de uma vez na execução de um script.
- $**=: é utilizada para converter o dado recebido para o banco de dados;
- *Result*: retorna os valores capturados ao banco de dados;

- *Echo*: é utilizada para saída de informações, podendo conter letras, números e/ou variáveis;

Podemos ver um exemplo de suas funcionalidades na Figura 12.

**Figura 12 – Formulário em PHP**

```php
<?php

if(isset($_POST['submit']))
{
  /*  print_r('Nome:' . $_POST['nome']);
    print_r('<br>');
    print_r('Email:' . $_POST['email']);
    print_r('<br>');
    print_r('Senha:' . $_POST['senha']);
*/

  include_once('config.php');

 $nome = $_POST['nome'];
 $email = $_POST['email'];
 $senha = $_POST['senha'];

 $result = mysqli_query($conexao, "INSERT INTO usuarios(nome,email,senha) VALUES ('$nome','$email','$senha')");

 echo "<script>alert('" . $_POST['nome']  . " salvo com sucesso !!! ');</script>";

}
?>
```

Fonte: Autoria Própria, 2022

## 2.5 LARAVEL

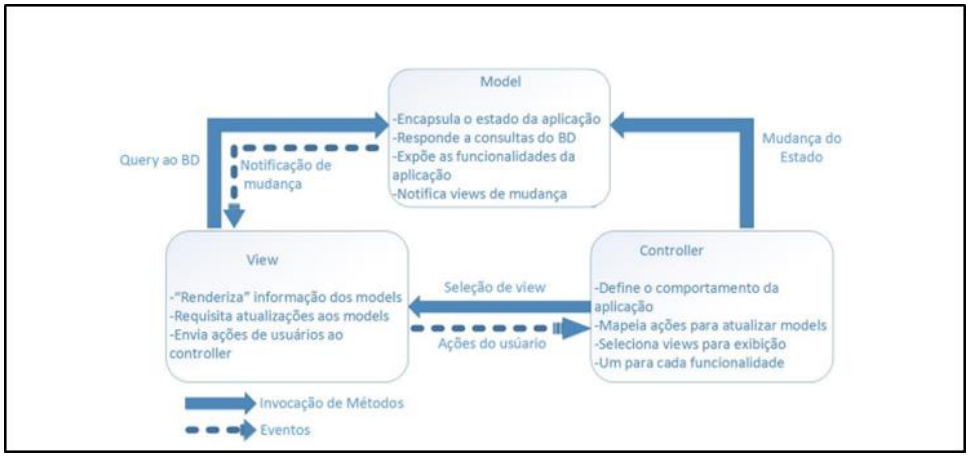
Antes de explicarmos o que é o Laravel em si, é importante saber o que é um *framework*, e para que ele serve. Com o passar do tempo, o desenvolvimento de aplicativos e sites foram ficando cada vez mais complexos e extensos e com a finalidade de ajudar os desenvolvedores surgiu os frameworks. Segundo Gabardo (2017), os frameworks são um conjunto de classes ou funções que ajudam no desenvolvimento de um software neles se encontra um ponto de partida, pois já dão uma estrutura básica, também reduz bastante a reescrita de códigos-fonte e faz com que o desenvolvedor siga um padrão de boas práticas de programação para construir sua aplicação.

Segundo Pelizza, Bertolini e Silveira (2017) o Laravel é um framework PHP utilizado para aplicações *web*. Ele tem uma arquitetura MVC (*Model-View-Controller*), e seu principal objetivo é ajudar no desenvolvimento de aplicações seguras e de alto desempenho de uma forma rápida e simplificada. Para facilitar ainda mais o entendimento sobre o funcionamento do Laravel. O Silva (2019), dividiu os três

conceitos básicos da arquitetura do Laravel Model, *View* e *Controller*, ele descreveu esses conceitos como:

- *Model*: é o componente responsável pela interação com banco de dados e manipulação dos dados, lógicas e as regras;
- *View*: esse componente trabalha com a interação com o usuário, ele exibe as informações de entrada e saída;
- *Controller*: já esse componente executa funcionalidades e requisições que manipula os dados mediante a *model*, e recebe e envia dados para *view*. Esse componente é como se fosse um filtro entre a *model* e a *view*;

**Figura 13 – Exemplo de arquitetura MVC**




Fonte: Pelizza, Bertolini e Silveira, 2017.

Clicando em "*Install* for me *Only*". Clique em "*next*" até terminar a instalação.

Para finalizar, basta apenas criar o seu projeto, onde tal etapa é feita pelo próprio *prompt* de comando (*cmd*): "*composer create-project –prefer-dist laravel/laravel* nome_do_projeto". Veja a Figura 14.

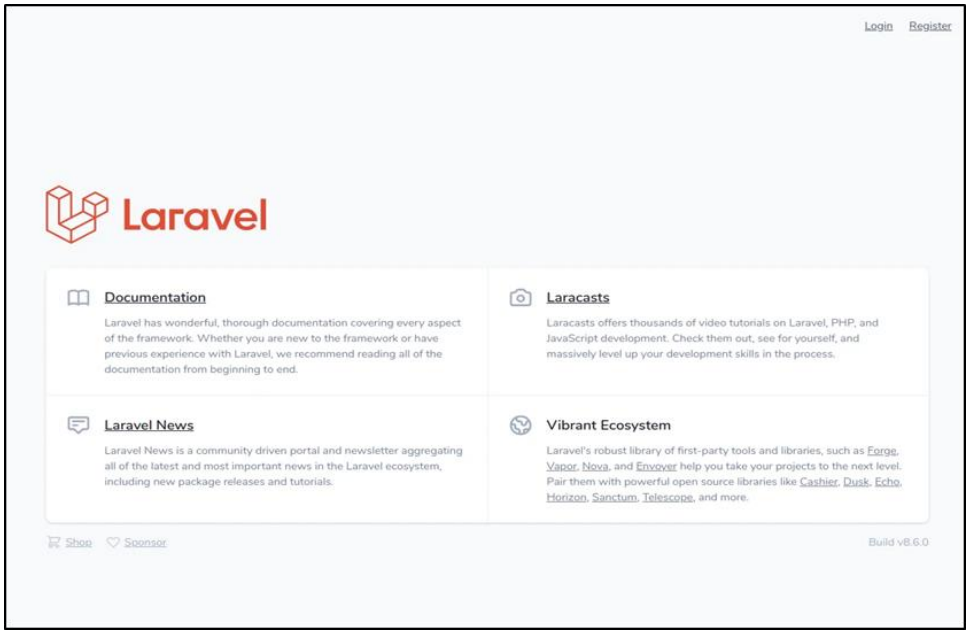
**Figura 14 – Criando pasta do projeto**

```
c:\xampp\htdocs>composer create-project --prefer-dist laravel/laravel nome_do_projeto
```

Fonte: Autoria própria, 2022

Nesta imagem, entramos no diretório do *xampp* “C:*xampp**htdocs*” e enfim, colocado o comando que irá realizar a criação do projeto. Após a criação, se utiliza um outro comando após entrar no diretório da pasta criada: “*php artisan serve*”, que tem a função de gerar um *link* para o seu projeto. Assim, ficando o seu projeto sem nenhuma alteração nas pastas (Figura 15):

**Figura 15 – Tela básica do Laravel**




Fonte: Autoria própria, 2022

## 2.6 BANCO DE DADOS

Bancos de dados armazenam informações em um computador e as relacionam com entidades como nome, e-mail e senha, permitindo consultas de forma simples, direta e mantendo a segurança do usuário. Afinal, é utilizado um SBDG e a empresa proprietária do *software* não é dona do banco de dados dos usuários, segundo Lobo (2008).

Na figura 16, é possível ver informações do banco de dados com as entidades Id, Nome, Email e Senha. O Id é uma chave primária, um campo que não se repete e serve para localizar dados dentro do banco através do SBDG, já que ele é um campo único, segundo Ramakrishnan e Gehrke (2007).

**Figura 16 – Exemplo com SBDG**

| id | nome | email | senha |
|---|---|---|---|
| 1 | Gabriel Diniz Silva | gaabrieldiniz5@gmail.com | asdasddas |
| 2 | Gabriel Diniz Silva | gaabrieldiniz5@gmail.com | asdasddas |

Fonte: Autoria própria, 2022.

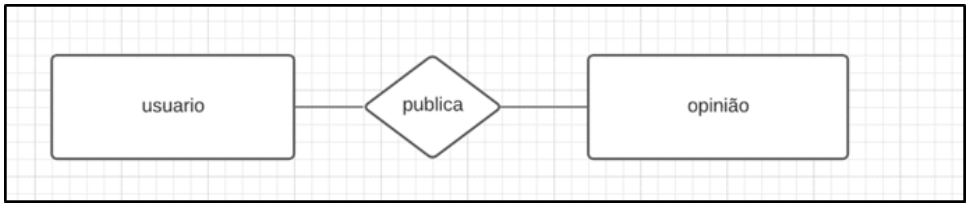
### 2.6.1 DER

Segundo Date (2004), o DER possui as seguintes propriedades e características:

- O DER facilita a visualização das entidades e suas ações dentro do banco de dados;
- Retângulos representam as entidades;
- Elipses representam os atributos;
- Losangos representam relacionamentos;

Linhas ligam atributos a entidade e entidades a relacionamentos como na Figura 17.

**Figura 17 – Exemplo de DER**




Autoria própria – 2022.

### 2.6.2 MER

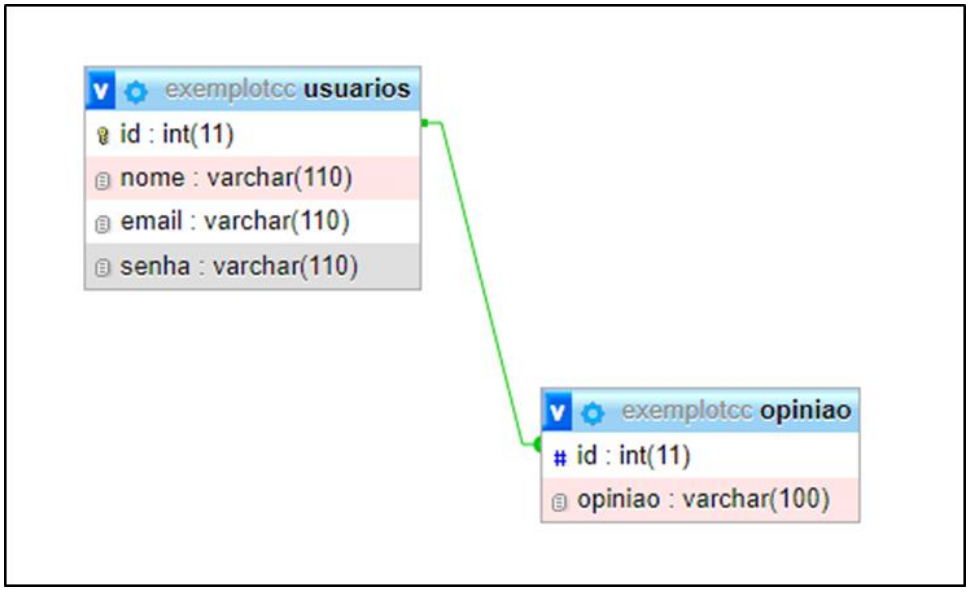
O MER deixa claro as relações das tabelas e suas entidades dentro do banco de dados, segundo Taylor (2015).

Ainda com base em Ramakrishnan e Gehrke (2007), na figura 18 é possível observar a chave primaria (Id) relacionada com a chave estrangeira. A chave estrangeira é o

que relaciona essas duas tabelas, de forma que o valor colocado na chave primaria, seja automaticamente colocado na chave estrangeira da outra tabela.

O valor da chave estrangeira pode se repetir dentro da tabela “opinião”, já que ele é só uma referência para a chave primária da tabela (Figura 18).

**Figura 18 – Exemplo de MER**




Fonte: Autoria própria, 2022

### 2.6.3 TIPOS DE DADOS

Segundo Carvalho (2015), o *Mysql* possui 3 categorias de tipo de dados:

- Char: Guarda até 255 caracteres podendo conter números, letras ou caracteres especiais. O tamanho deve ser definido entre parênteses;
- *Varchar*: Ela tem características char, mas caso seja inserido um registro com mais de 255 caracteres, ele transforma pro tipo *text*;
- *Text*: Guarda uma string com tamanho máximo de 65.535 caracteres;
- INT: Guarda números inteiros suportando de -2147483648 até 2147483648. Pode ter o número máximo de caracteres especificado entre parênteses;
- Tinyint: Guarda números do tipo inteiro e suporta de -128 até 127 caracteres;

- Float: guarda números reais podendo ter o número máximo de caracteres especificado entre parênteses. Tanto o tamanho inteiro quanto o tamanho numérico da coluna;
- *Date*: Guarda datas;

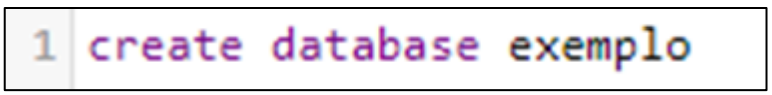
## 2.7 Mysql

Segundo Taylor (2015), *Mysql* é um SBDG que além de rodar em várias plataformas, e ser seguro, ainda é rápido e flexível para empresas que necessitam armazenar dados em um banco.

### 2.7.1 Comandos do Mysql

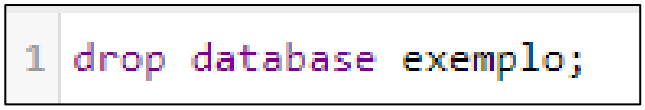
De acordo com Carvalho, (2015) para começar a inserir dados no banco, é preciso criar o próprio utilizando o comando "*create database*", conforme a Figura 19, em seguida, o nome que será direcionado ao banco de dados. Se o usuário desejar excluir essa base de dados, é possível utilizar o comando "*drop database* nome do banco" para que a base seja excluída, conforme Figura 20.

**Figura 19 – Criando o Banco**




Fonte: Autoria própria, 2022

**Figura 20 – Apagando o Banco**



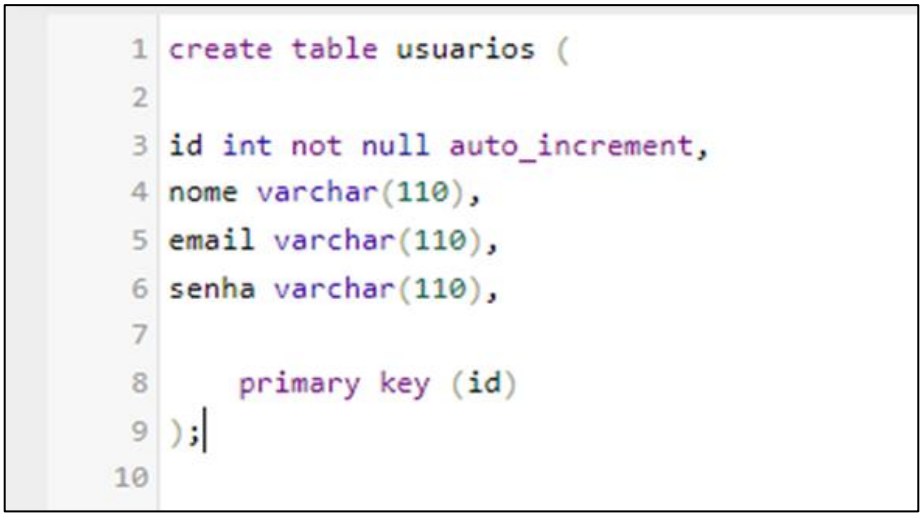

Fonte: Autoria própria, 2022

Como já citado anteriormente, o Id é uma chave primária, ou seja, ela tem um valor que nunca se repete dentro do banco de dados com uma numeração gerada automaticamente (*auto_increment*).

Para definir essa chave é usado abaixo da definição das entidades o comando "*Primary key* (nome do campo que será definido como chave primária)", além de ser uma das entidades dentro do banco de dados, assim como nome e e-mail, segundo Lobo (2008). Veja a figura 21.

O *not null* não permite que seja inserido um registro na tabela sem o campo ser preenchido.

**Figura 21 – Criando Tabela de Usuários**




Fonte: Autoria própria, 2022

Ainda conforme Lobo (2008), o "*insert*" insere novos registros no banco de dados, como é exemplificado na Figura 22.

**Figura 22 – Insert**


Fonte: Autoria própria, 2022

Segundo Carvalho (2015), o “select” tem como função fazer a consulta de dados dentro das tabelas como é possível ver na Figura 23.

**Figura 23 – Select**


Fonte: Autoria própria, 2022

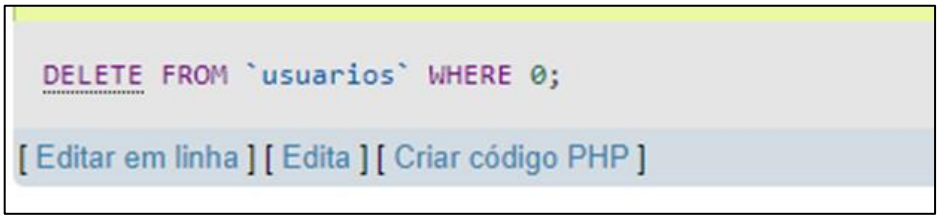
Ainda com base em Taylor (2015), o “*update*” altera linhas da tabela sem que seja preciso apagar ela, como demonstrado na Figura 24.

**Figura 24 – Update**


Autoria própria – 2022

Segundo Lobo (2008), o comando “delete” apaga linhas de uma tabela, como demonstrado na figura 25.

**Figura 25 – Delete**




Fonte: Autoria própria, 2022

## 2.8 UML

Segundo Guedes (2011), UML é uma linguagem que tem como objetivo visualizar, especificar, construir e documentar os artefatos de um sistema complexo de software. Ou seja, vai te fornecer uma visão dos requisitos necessários, identificando as funcionalidades do software e os atores que podem utilizar sem se preocupar com nada além disso.

### 2.8.1 Termos e conceitos

Alguns termos e conceitos da UML, segundo Booch (2006):

- Classe: as classes são blocos de construção mais importantes de qualquer sistema orientado a objetos;
- Nome: cada classe deve ter um nome que se diferencie de outras classes;
- Atributos: um atributo é uma propriedade nomeada de uma classe que descreve um intervalo de valores que as instancias da propriedade podem apresentar;
- Operações: uma operação é a implementação de um serviço que pode ser solicitado por algum objeto da classe para modificar o comportamento;
- Objetos: qualquer coisa é um objeto, segundo Bezerra (2007), os objetos realizam tarefas por meio da requisição de serviços a outros objetos e cada um deles pertence a uma classe que agrupa os objetos similares.

### 2.8.2 Requisitos

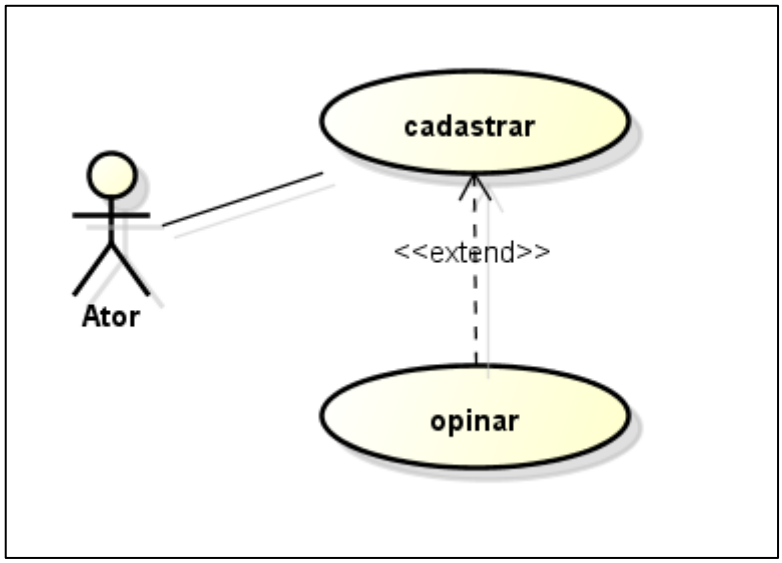
Existem dois tipos de requisitos, os funcionais e não funcionais. Os funcionais correspondem ao que o cliente deseja que o sistema realize, segundo Guedes (2011). Os não funcionais são mais amplos: usabilidade, confiabilidade, desempenho, interface e segurança. Ainda de acordo com Bezerra (2007), alguns requisitos não funcionais identificam regras de negócios que são políticas, normas e condições estabelecidas pela empresa. Essas normas e condições são seguidas na execução de uma funcionalidade.

### 2.8.3 Diagramas

A UML possui muitos diagramas, segundo Guedes (2011). Aqui estão exemplo de 3 diagramas e sobre relacionamento entre os mesmos:

- Diagrama de caso de uso: usado na fase de levantamento de requisitos do sistema e acaba servindo de base para outros diagramas ao decorrer do projeto;
- *Extend*: juntamente com o include são dependências, enquanto o extend define a relação não obrigatória, include estabelece uma relação obrigatória entre os casos de uso como é demonstrado na Figura 26.

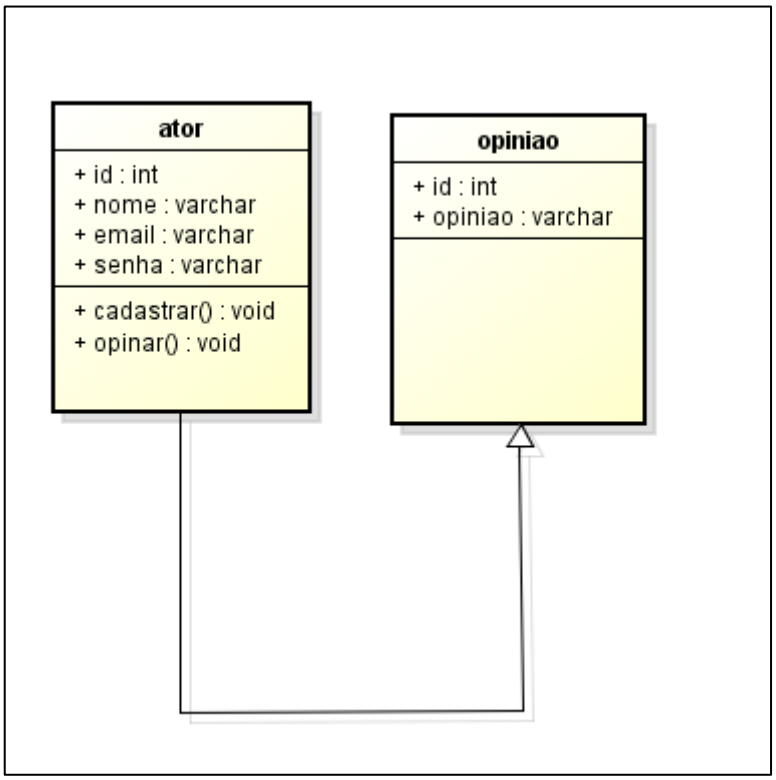
**Figura 26 – Diagrama de Caso de Uso**




Fonte: Autoria Própria, 2022

- Diagrama de classes: define a estrutura de classes utilizadas pelo sistema, determinando os atributos e métodos que cada classe tem, como exemplificado na Figura 27.

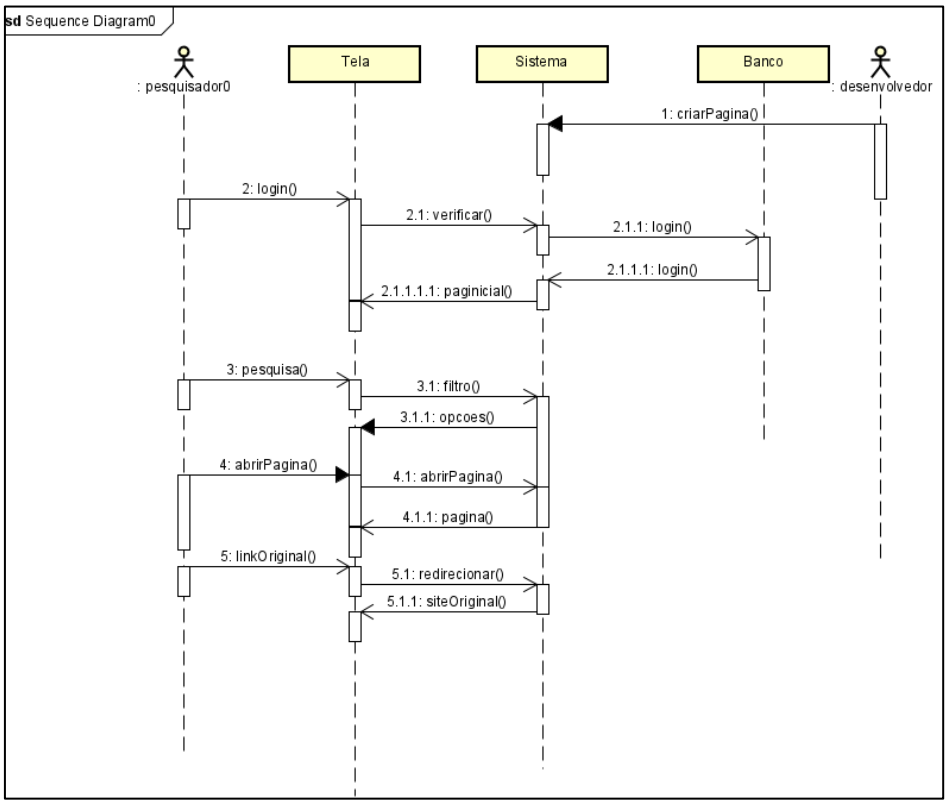
**Figura 27 – Diagrama de Classe**




Fonte: Autoria Própria, 2022

- Diagrama de objetos: é praticamente um complemento para o diagrama de classes. O diagrama fornece valores armazenados por objetos de um diagrama de classes em um determinado momento da execução de um processo do software;
- Diagrama de Sequência: deixa clara a sequência que o sistema segue, como exemplificado na figura 28.

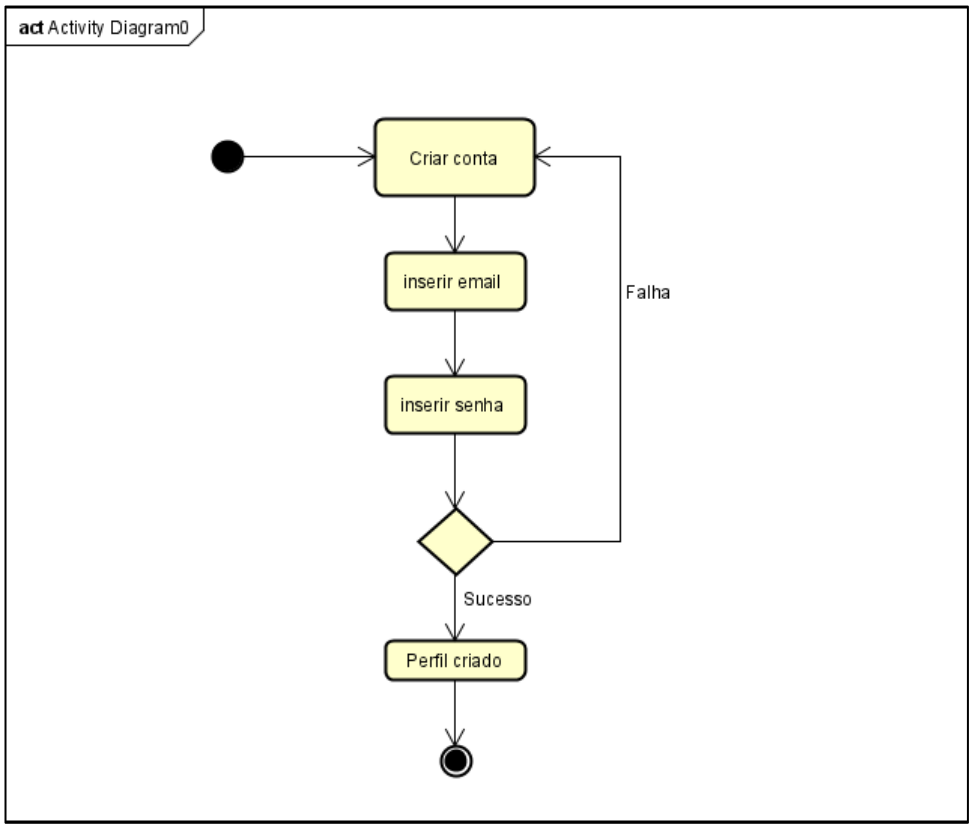
**Figura 28 – Diagrama de Sequência**




Fonte: Autoria Própria, 2022

- Diagrama de Atividade: descreve uma atividade especifica da aplicação, como por exemplo o login que é demonstrado na Figura 29.

**Figura 29 – Diagrama de Atividade**




Fonte: Autoria Própria, 2022

### 2.8.4 Atores

Qualquer elemento externo que interage com o sistema é considerado um ator, segundo Bezerra (2007), estas são algumas categorias de atores:

- Cargos;
- Organizações;
- Outros sistemas de software;
- Equipamentos com qual o sistema deve se comunicar.

# 3 DESENVOLVIMENTO

Nesse capítulo, mostraremos o desenvolvimento do nosso projeto através de requisitos, diagramas etc.

## 3.1 Requisitos

Abaixo estão os requisitos que englobam nosso site:

**Requisitos Funcionais:**

RF1 – O sistema deve permitir que um usuário faça cadastro usando seu nome de usuário, e-mail e senha;

RF2 – Deve permitir que um usuário faça login usando somente seu e-mail e senha;

RF3 – O usuário deve pesquisar o que deseja assistir.

**Requisitos Não Funcionais:**

RNF1 – Mostrar uma curiosidade relacionada ao tema pesquisado;

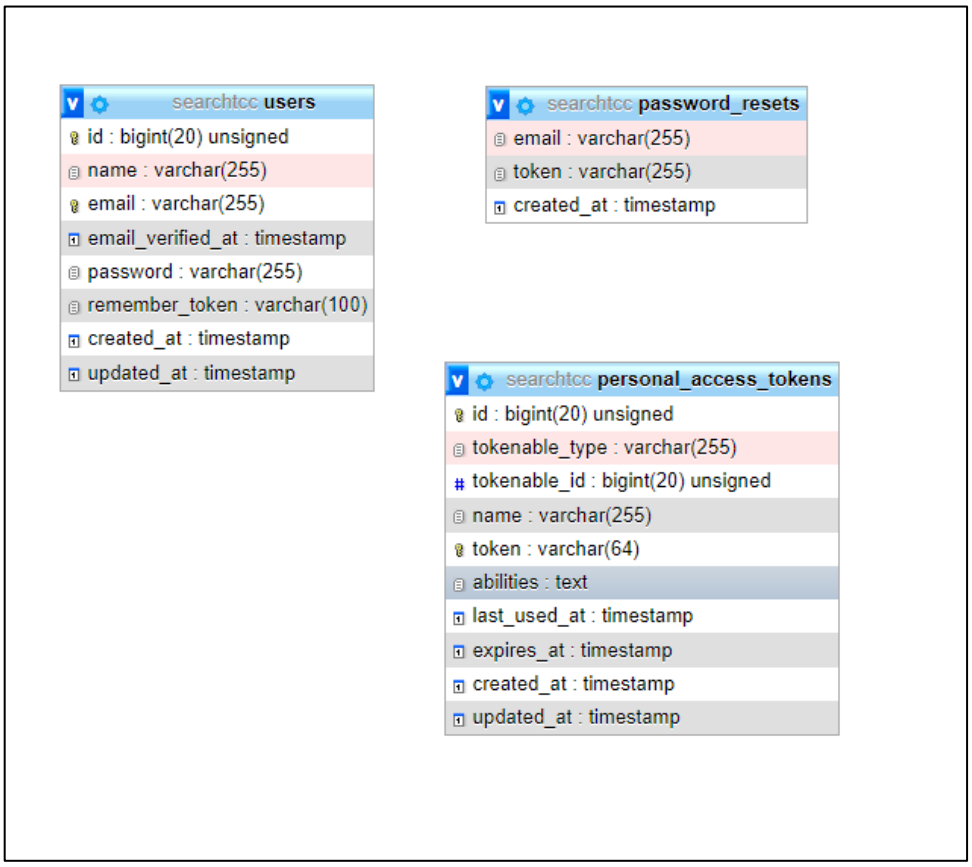
RNF2 - Apresentar elenco do conteúdo;

RNF3 - Mostrar sinopse

## 3.2 MER

Este é o MER do projeto Search. Ele representa o cadastro e login de usuários presentes na aplicação web. Veja a Figura 30.

**Figura 30 – MER**

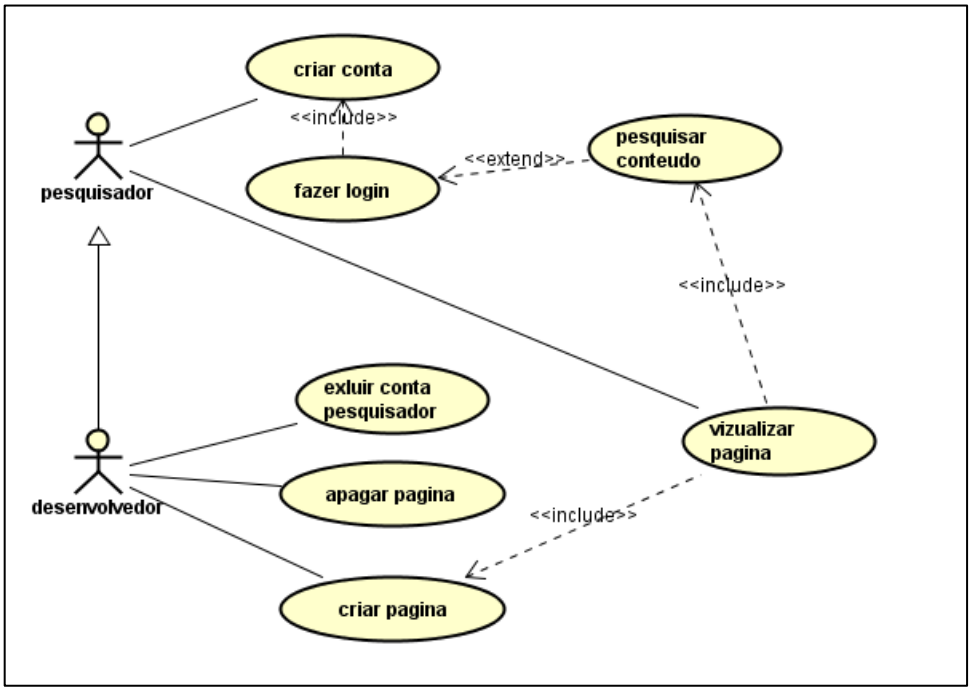



Fonte: Autoria Própria, 2022

## 3.3 Diagramas

O diagrama de caso de uso demonstra a interação do cliente com a aplicação. Ele apresenta as possibilidades e decisões que o usuário (pesquisador) pode tomar. Além disso, o diagrama mostra a interação do desenvolvedor com a aplicação web. Veja a Figura 31.

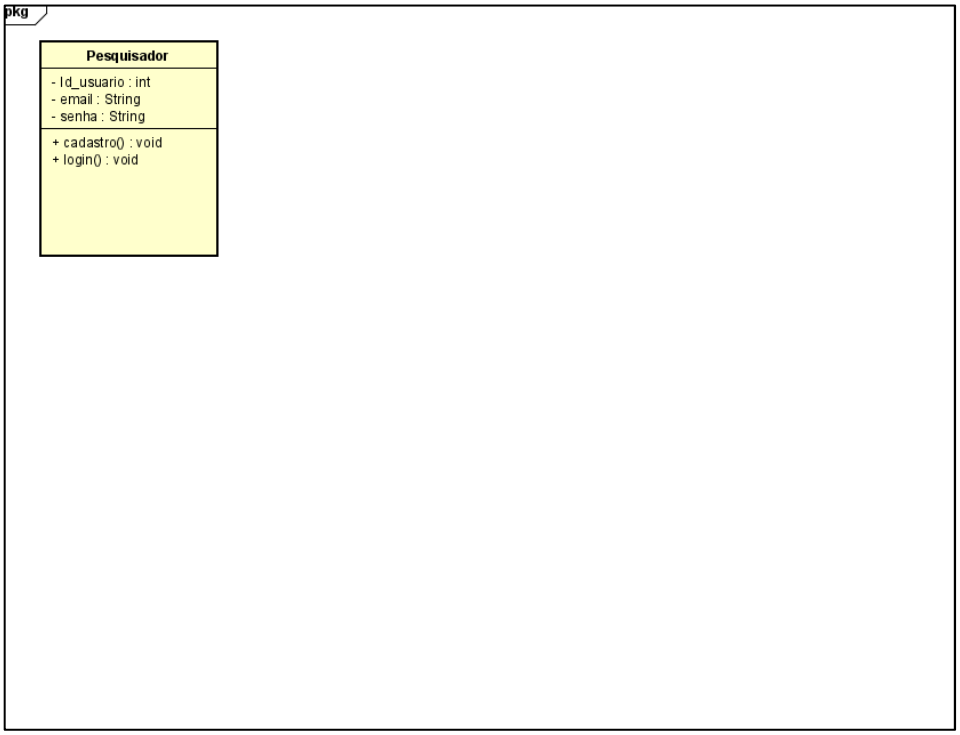
**Figura 31 – Diagrama de caso de uso**




Fonte: Autoria Própria, 2022

O diagrama de classe apresenta a classe utilizada na estrutura de banco de dados. Especificamente no cadastro e login do usuário. Veja a Figura 32.

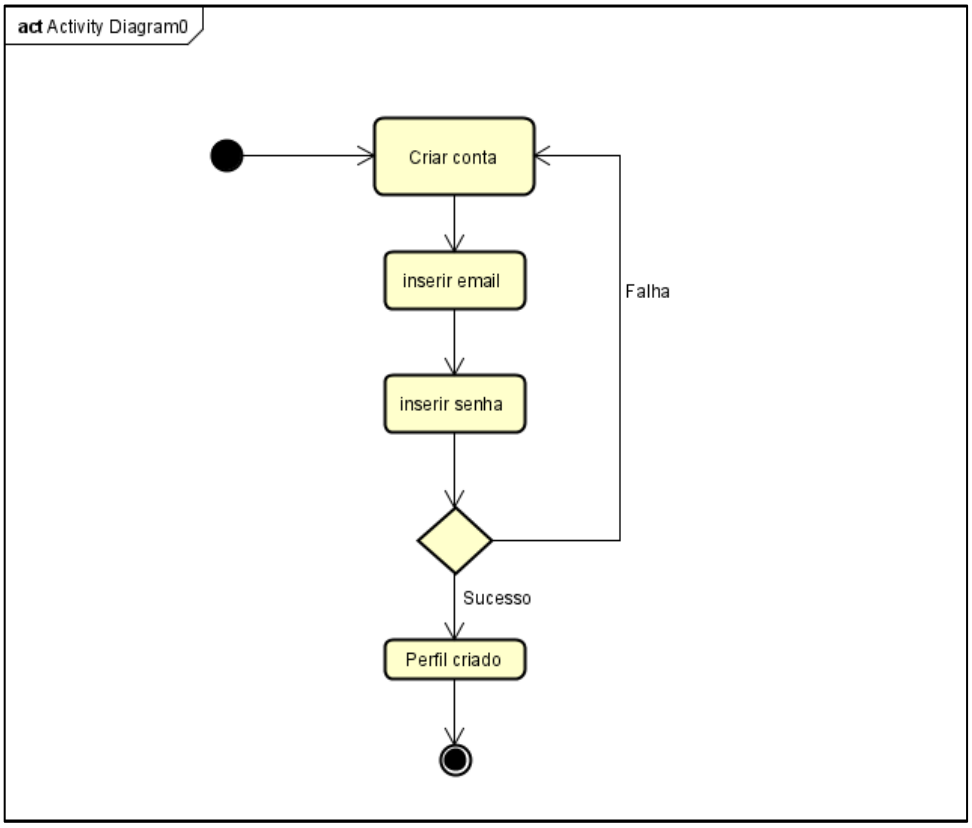
**Figura 32 – Diagrama de classe**




Fonte: Autoria Própria, 2022

O diagrama de atividade na Figura 33, descreve a atividade de cadastro da nossa aplicação.

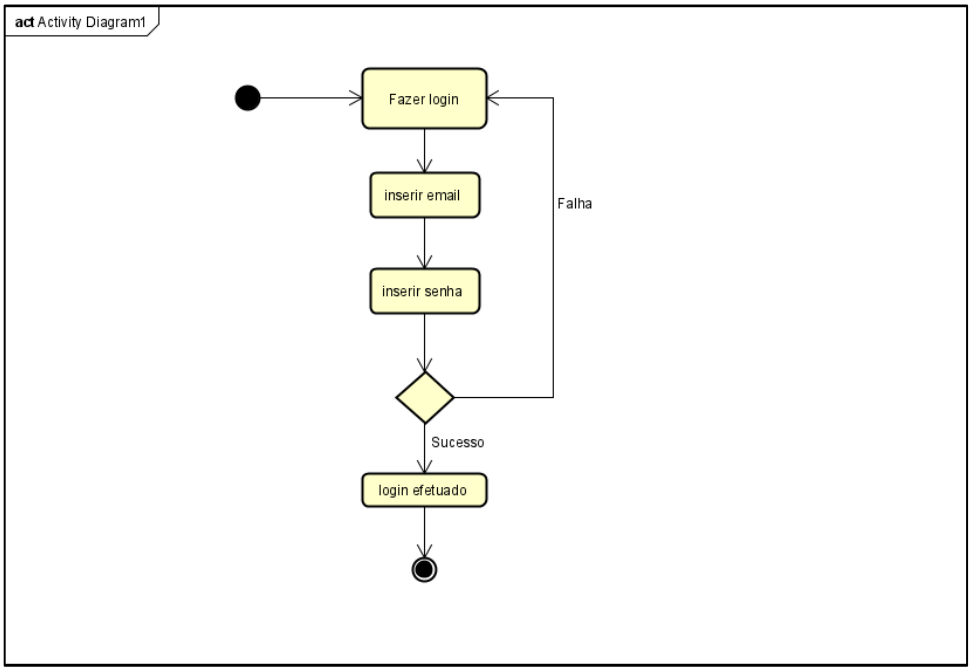
**Figura 33 – Diagrama de atividade**




Fonte: Autoria Própria, 2022

O diagrama de atividade na Figura 34, descreve a atividade de login do usuário, na aplicação.

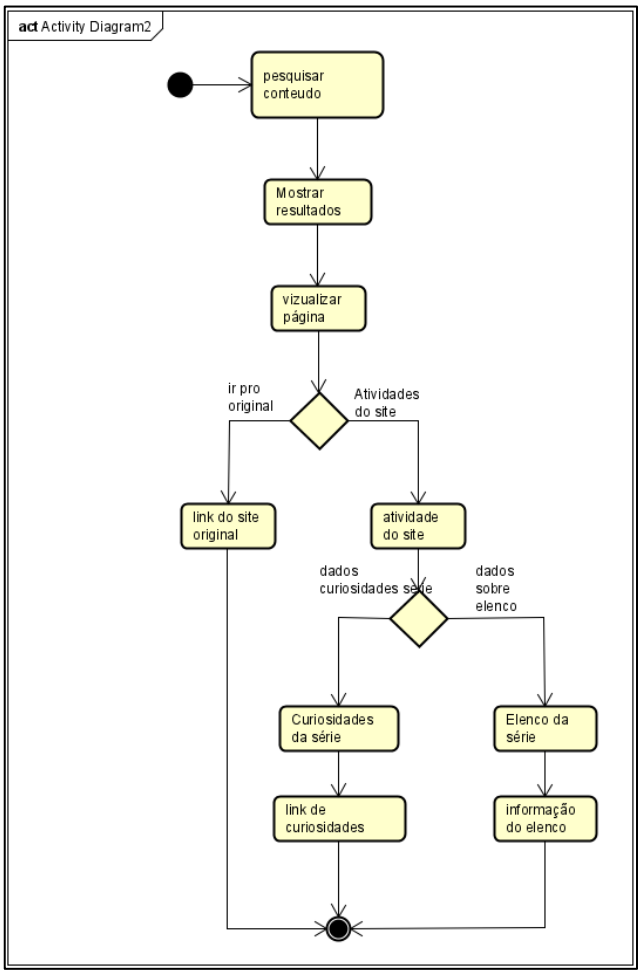
**Figura 34 – Diagrama de atividade**




Fonte: Autoria Própria, 2022

O diagrama de atividades referente a imagem 35, descreve a funcionalidade da barra de pesquisa na aplicação web.

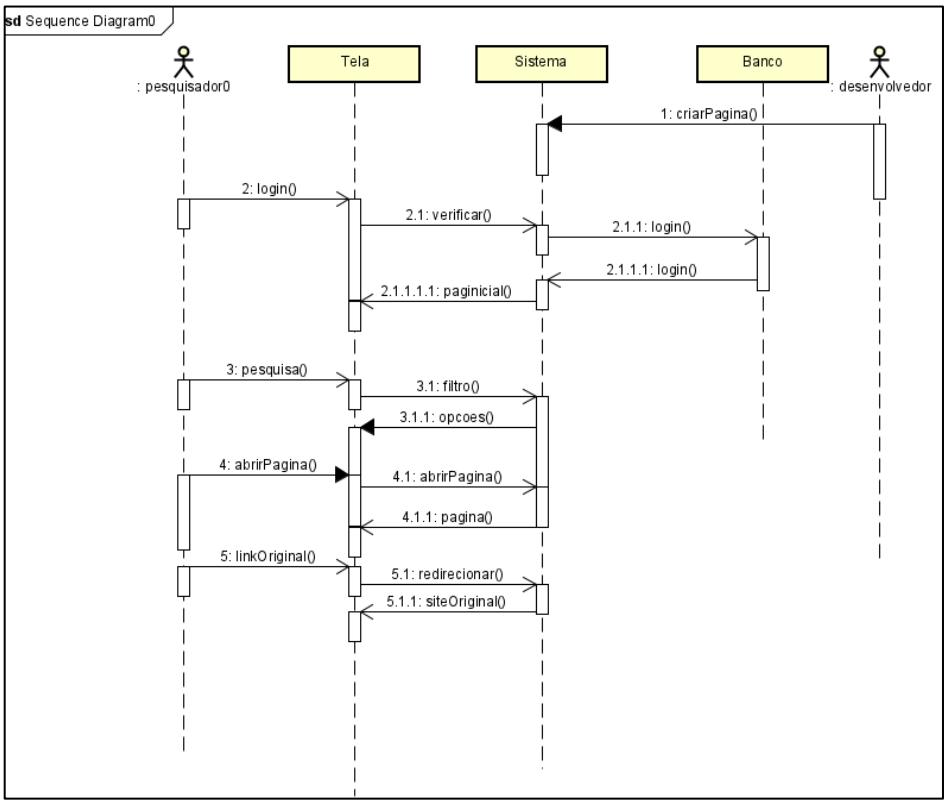
**Figura 35 – Diagrama de atividade**




Fonte: Autoria Própria, 2022

O diagrama de sequência apresentado na Figura 36, mostra de forma sequencial, todas as funcionalidades presentes na aplicação web.

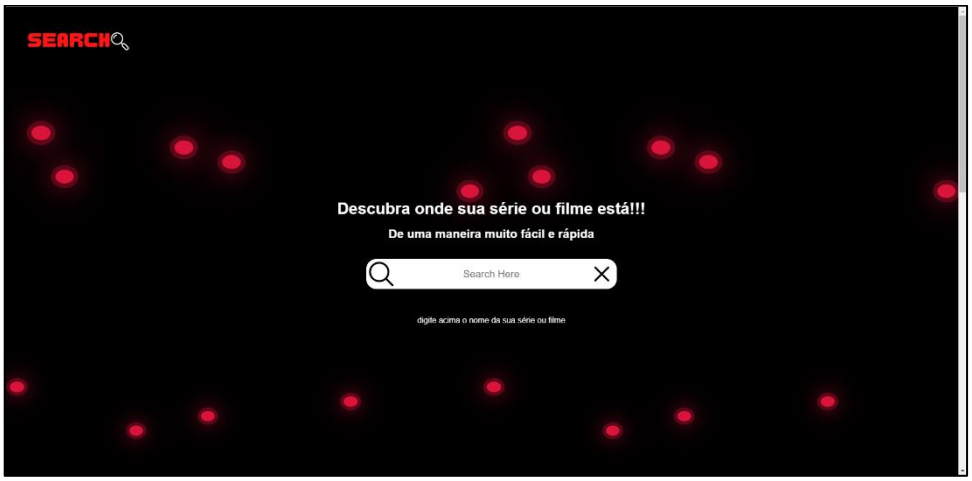
**Figura 36 – Diagrama de sequência**




Fonte: Autoria Própria, 2022

A figura 37 mostra a tela inicial e principal da nossa aplicação web. Ela contém a logo da aplicação e a barra de pesquisa, onde o usuário digitará o conteúdo desejado.

**Figura 37 – Tela inicial e barra de pesquisa**




Fonte: Autoria Própria, 2022

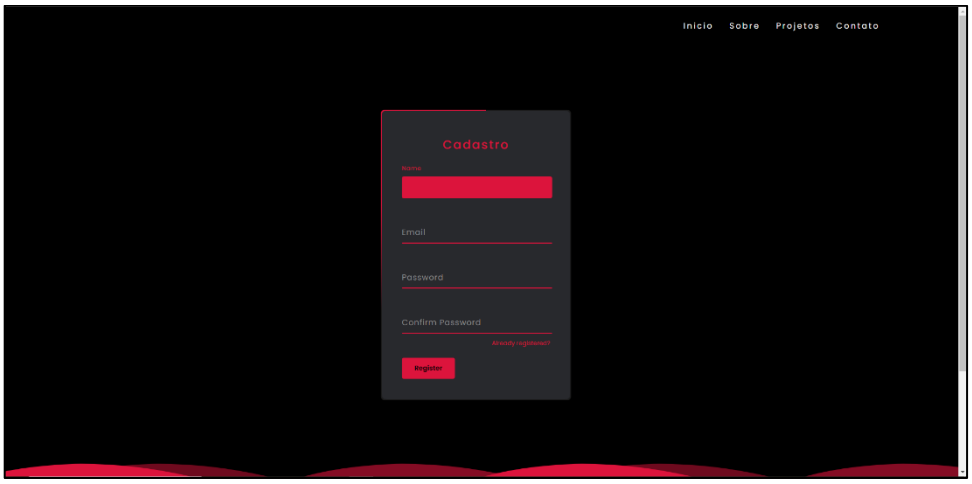
A figura 38 apresenta ainda a tela inicial da aplicação. Contém informações sobre o site e desenvolvedores.

**Figura 38 – Informações tela principal**


Fonte: Autoria Própria, 2022

A figura abaixo (Figura 39), demonstra a tela de cadastro presente em nossa aplicação.

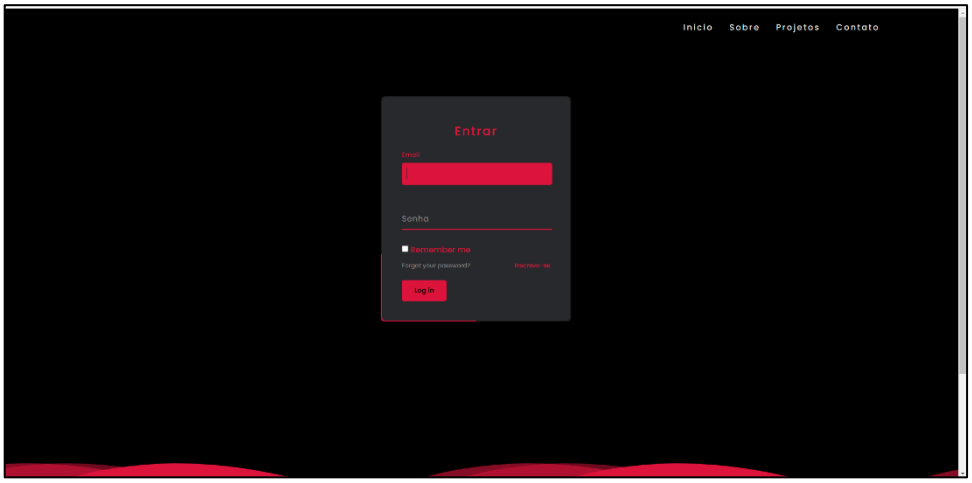
**Figura 39 – Tela de cadastro**




Fonte: Autoria Própria, 2022

A figura 40 apresenta a tela sucessora à tela de cadastro, a tela de login que o usuário utilizará.

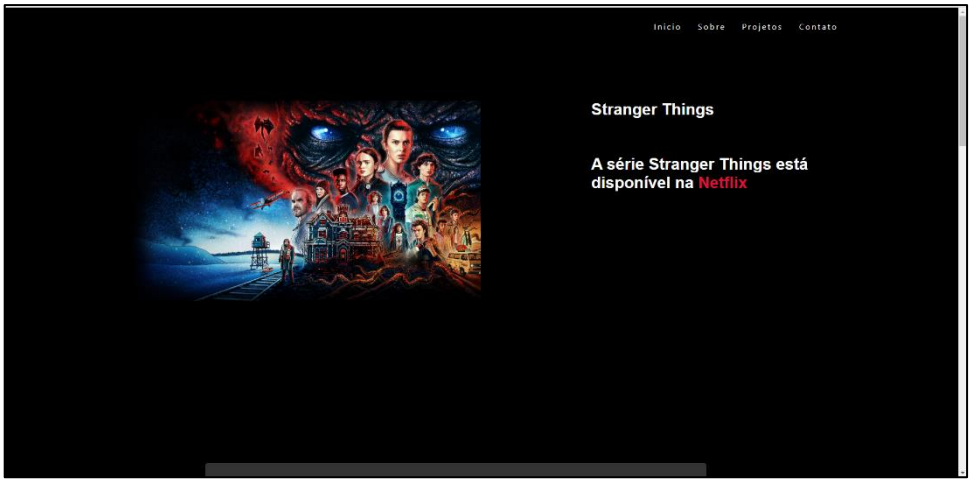
**Figura 40 – Tela de login**




Fonte: Autoria Própria, 2022

A tela de indicação (Figuras 41 e 42), é apresentada após a pesquisa do conteúdo. Como dito, ela indica onde o conteúdo pode ser encontrado e acessado.

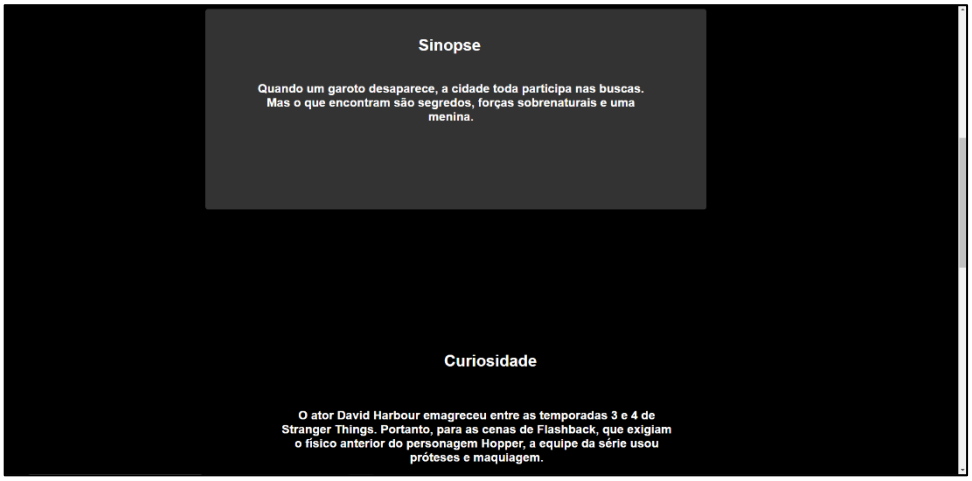
**Figura 41 – Tela de indicação**




Fonte: Autoria Própria, 2022

Ainda na tela de indicação, a aplicação apresenta a sinopse e curiosidades a respeito do conteúdo pesquisado no site.

**Figura 42 – Tela de indicação**




Fonte: Autoria Própria, 2022

# 4 CONCLUSÃO

Os recursos tecnológicos são vitais atualmente. Esses recursos unidos a estudos e pesquisas, possibilitaram um desenvolvimento embasado e concreto do projeto. Através do projeto Search, um indivíduo pode pesquisar e localizar um conteúdo audiovisual desejado, sem infringir as leis e diretrizes do nosso país. Isso foi possível, graças às tecnologias de programação utilizadas no projeto, como HTML, CSS, Javascript, PHP e MySql.

Além disso, os diagramas auxiliaram no desenvolvimento e compreendimento da lógica, sequência, modelagem de dados e estrutura da aplicação web. Dentre as tecnologias citadas, o Javascript foi essencial para o cumprimento do objetivo principal do projeto. Alunos e companheiros de classe sugeriram implementações e recursos que podemos usar como futuras atualizações para a nossa aplicação

## REFERÊNCIAS

ACHOUR, Mehdi; BETZ, Friedhelm; DOVGAL, Antony; LOPES, Nuno; MAGNUSSON, Hannes; RICHTER, Georg; SEGUY, Damien; VRANA, Jakub. **Manual do php**. 1997. Disponível em: https://www.php.net/manual/pt_BR/history.php.books.php. Acesso em: 30 ago. 2022.

BEZERRA, Eduardo. **Princípios de Análise e Projeto de Sistemas com UML**. 2. ed. São Paulo: Elsevier, 2006.

BOOCH, Grady. **Uml - Guia do Usuário**. Rio de Janeiro: Gen Ltc, 2006.

CARVALHO, Vinícius. **MySQL: Comece com o principal banco de dados open source do mercado**. São Paulo: Casa do Código, 2015.

CONVERSE, Tim; PARK, Joyce. **PHP**: a bíblia. Rio de Janeiro: Elsevier, 2003. Tradução da 2. ed. original de Edson Furmankiewicz.

DALL`OGLIO, Pablo. **PHP**: programando co orientação a objetos. 2. ed. São Paulo: Novatec Editora, 2009

DATE, C.J. **Introdução a Sistemas de Bancos de Dados**. Rio de Janeiro: Elsevier, 2004.

DUCKETT, Jon. **HTML & CSS**: projete e construa websites. Rio de Janeiro: Alta Books, 2014.

EIS, Diego. **HTML e CSS3**: com farinha e pimenta. São Paulo: Tableless, 2012.

FERREIRA, Silvio. **Guia Prático de HTML5**. São Paulo: Universo dos Livros, 2013.

FLATSCHART, Fábio. **HTML 5**: embarque imediato. Rio de Janeiro: Brasport, 2011.

GRONER, Loiane. **Estrutura de dados e algoritmos com JavaScript**: escreva um código JavaScript complexo e eficaz usando a mais recente ECMASript. São Paulo: Novatec Editora, 2019.

GUEDES, Gilleanes T. A. **Uml - uma Abordagem Prática**. Santa Terezinha: Novatec, 2011.

JOBSTRAIBIZER, Flávia. **Criação de Sites com o CSS**: desenvolva páginas web mais leves e dinâmicas em menos tempo. São Paulo: Digerati Books, 2009.

LOBO, Edson Junio Rodrigues. **Curso Prático de Mysql**. São Paulo: Digerati Books, 2008.

PELIZZA, Angelica Caetane; BERTOLINI, Cristiano; SILVEIRA, Sidnei Renato. **Um estudo sobre Técnicas de Teste de Software no Framework Laravel**. 2017. 20 f. TCC (Doutorado) - Curso de Departamento de Tecnologia da Informação, Ufsm, Rio Grande do Sul, 2017. Cap. 4.

RAMAKRISHNAN, Raghu; GEHRKE, Johannes. **Sistemas de Gerenciamento de Bancos de Dados**. 3. ed. Porto Alegre: Amgh, 2007.

SCHEIDT, Felippe Alex**. Fundamentos de CSS**: criando design para sistemas web. Foz do Iguaçu: [s.n.], 2015.

SILVA, Maurício Samy. **JavaScript Guia do Programador**. São Paulo: Novatec, 2010.

SILVA, Maurício Samy. **Construindo Sites com CSS e (X)HTML**: sites controlados por folhas em estilo cascata. São Paulo: Novatec Editora Ltda., 2008.

SILVA, Erikson Dutra de Miranda. **Desenvolvimento de uma ferramenta de gestão para construção civil utilizando o Framework Laravel MVC.** 2019. 93 f. Monografia (Especialização) - Curso de Curso de Bacharelado em Ciências da Computação, Universidade Federal de Goiás, Catalão, 2019. Cap. 2.

TAYLOR, Allen G. **SQL para leigos**. Rio de Janeiro: Alta Books, 2015.