

# Formulário de Login

Aplicando conceitos básicos de programação WEB (front-end)

ETEC UIRAPURU



Revisando alguns conceitos...

# HTML



## HTML

O HTML não é considerado uma linguagem de programação, ele é uma linguagem que define a estruturas e textos de uma página WEB.

O que definimos com ele:

- Cabeçalhos
- Títulos
- Parágrafos

E etc.

# CSS

---

Se com o HTML conseguimos montar a estrutura (esqueleto) das páginas, o CSS nos permite realizar a estilização das páginas.

O CSS é responsável pela parte estética das páginas WEB e também não é uma linguagem de programação.

CSS





# HTML vs CSS



**HTML**



**CSS**



# HANDS-ON

```
import './index.css';  
import { ReactComponent as ArrowIcon } from '../assets/icons/arrow.svg';  
import { ReactComponent as BoltIcon } from '../assets/icons/bolt.svg';  
import { ReactComponent as RightArrowIcon } from '../assets/icons/right-arrow.svg';  
  
import React, { useState, useEffect, useRef } from 'react';  
import { CSSTransition } from 'react-transition-group';
```

```
"eslintConfig": {  
  "extends": [  
    "react-app",  
    "react-app" ]  
}
```

Preferencialmente na área de trabalho

Crie uma pasta chamada “site”

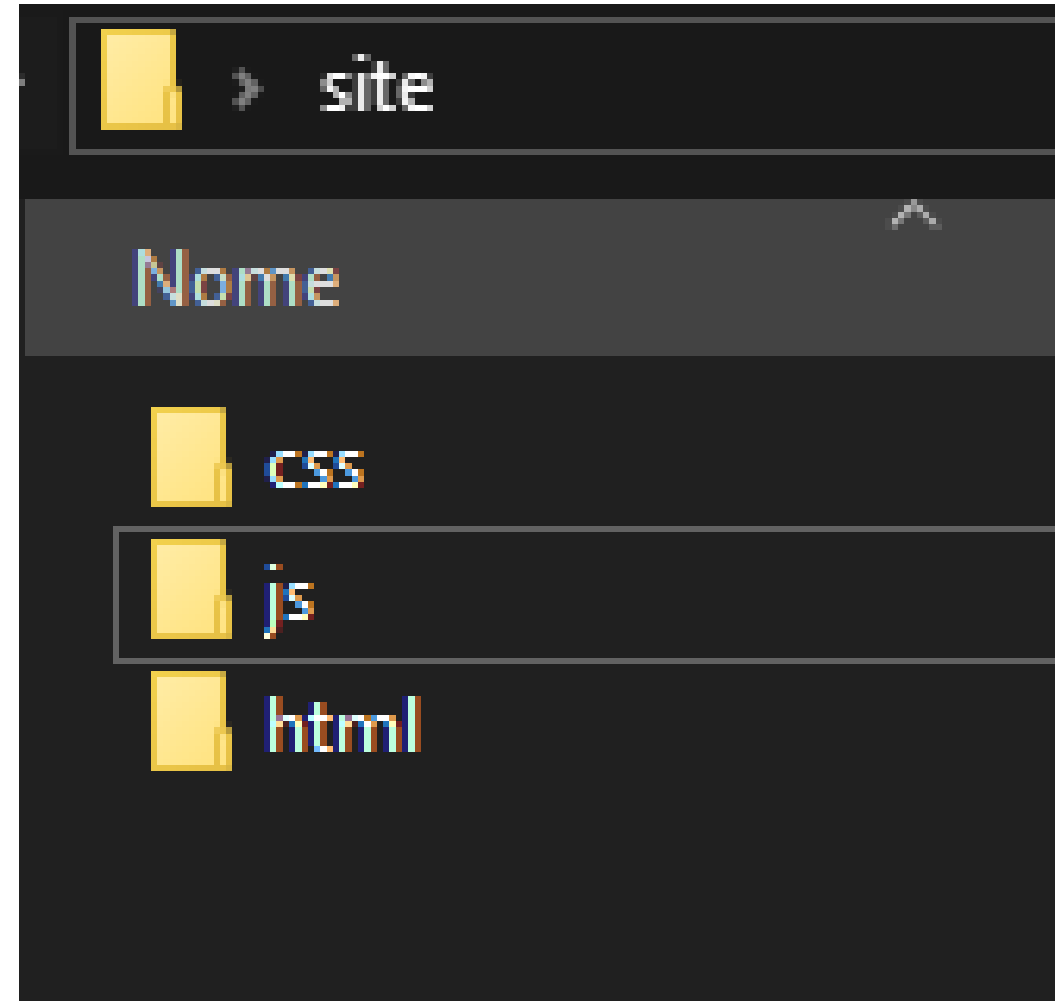


Dentro da pasta “site”, crie três pastas:

---

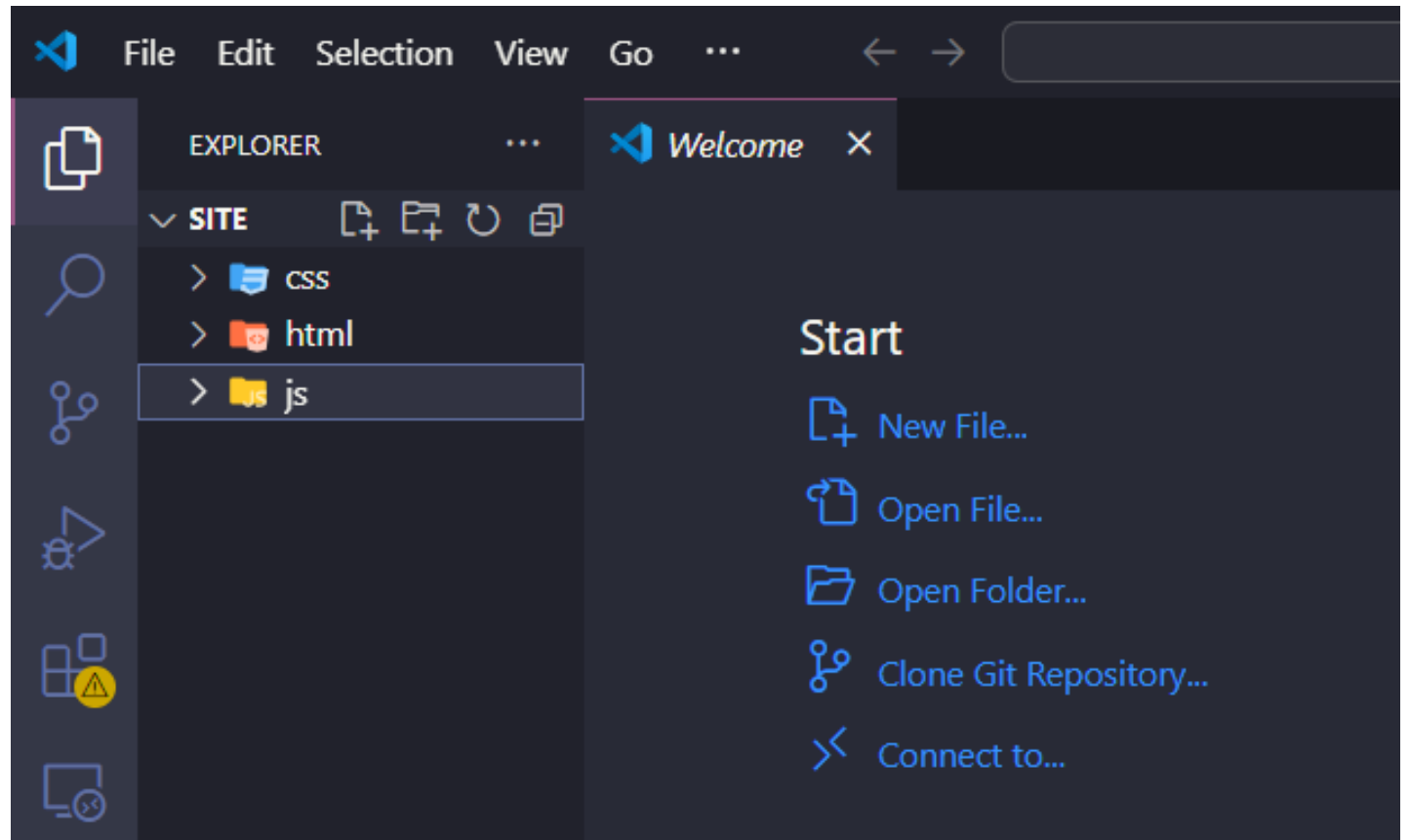
- css
- js
- html

**Em seguida, abra a pasta “site” no Visual Studio Code**





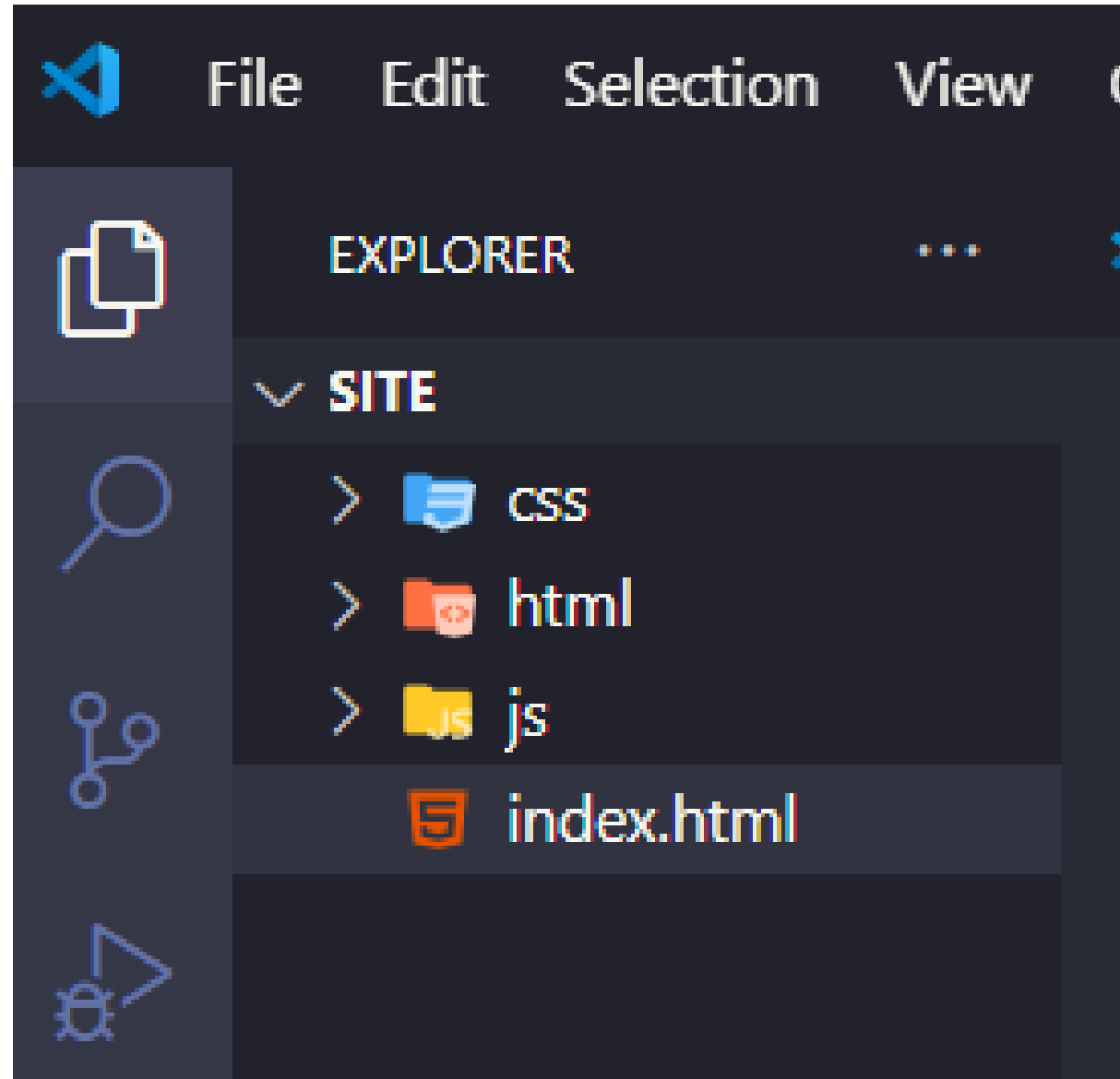
# VS Code



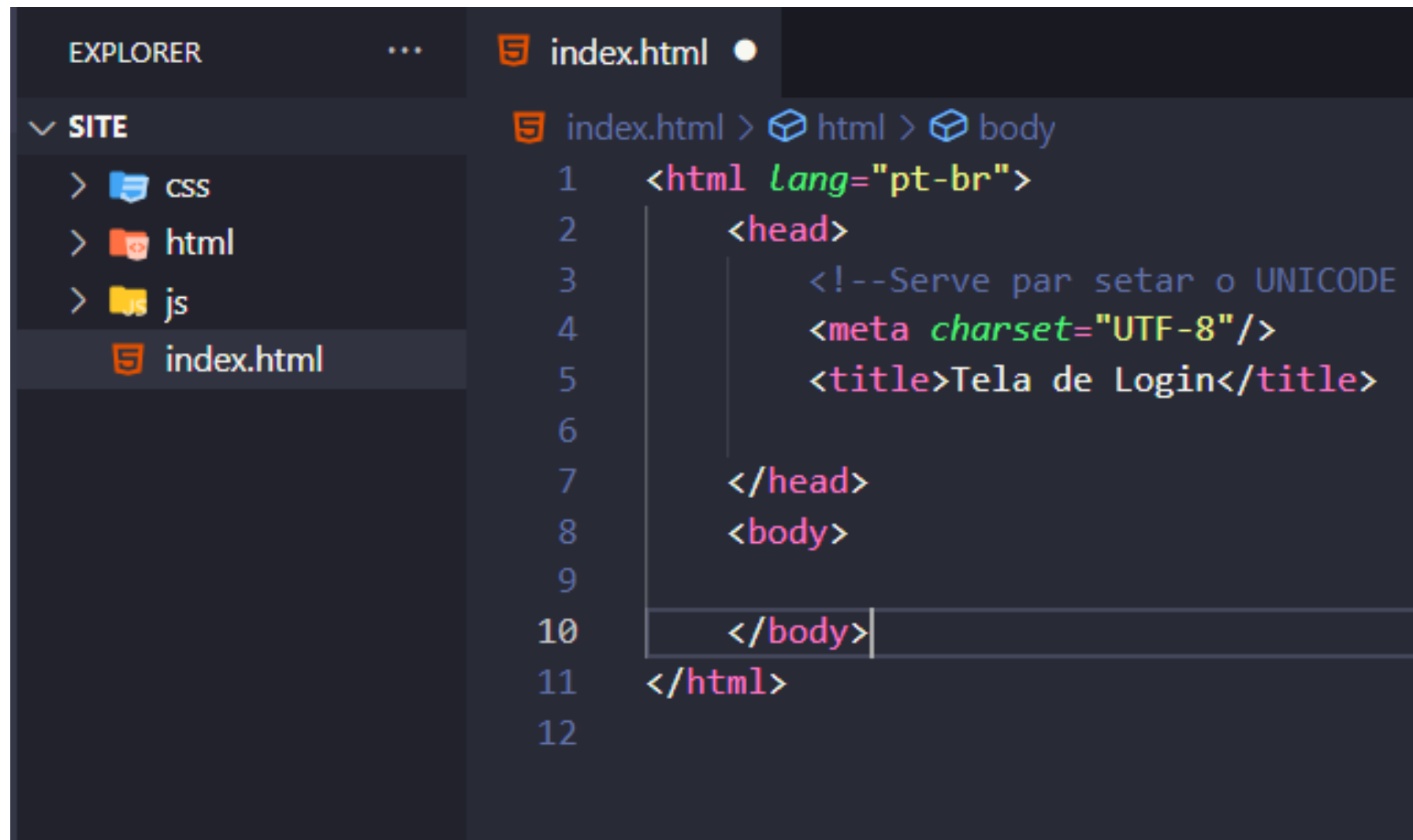
# Agora, crie o arquivo index.html

---

Nesse arquivo, é onde  
vamos desenvolver a  
nossa tela de LOGIN.



INDEX.HTML,  
vamos codar!'



The image shows a screenshot of the Visual Studio Code editor interface. On the left, the Explorer sidebar is open, showing a project structure under the name 'SITE'. It contains three folders: 'css', 'html', and 'js', each with a corresponding icon. Below these folders is a file named 'index.html', which is highlighted with a dark background. On the right, the Editor view is open, displaying the 'index.html' file. The breadcrumb navigation at the top of the editor shows the path 'index.html > html > body'. The code in the editor is as follows:

```
1  <html lang="pt-br">
2      <head>
3          <!--Serve par setar o UNICODE
4              <meta charset="UTF-8"/>
5              <title>Tela de Login</title>
6      </head>
7      <body>
8
9
10     </body>
11 </html>
12
```

# Vamos criar os containers da página WEB

Os containers vão nos apoiar a distribuir o conteúdo da página, com eles podemos definir as margens da tela e consequentemente a proporção de cada componente.

```
8  <body>
9    <div class="container-principal">
10      <div class="formulario">
11
12
13
14      </div>
15
16    </div>
17  </body>
18 </html>
```



# Criando o formulário...

- **minLength=“”** DEFINE MINIMO DE TEXTO A SER INSERIDO
- **maxLength=“”** DEFINE MAXIMO DE TEXTO A SER INSERIDO
- **required=“true”** DEFINE QUE O CAMPO É OBRIGATÓRIO
- **placeholder=“”** DEFINE UMA “LABEL” SOBRE O CAMPO

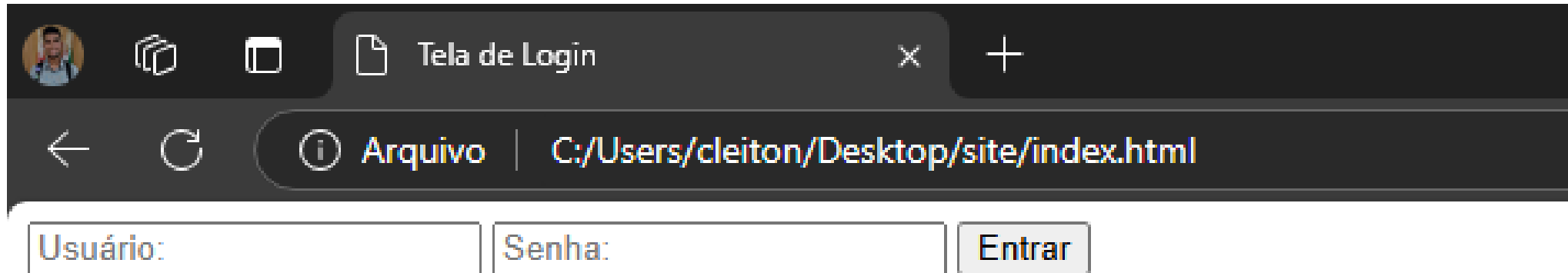
```
<body>
  <!--Essa div, representa onde todo o conteúdo do site vai ficar-->
  <div class="container-principal">
    <!-- aqui é o espaço onde vamos criar o formulário-->
    <div class="formulario">
      <form>
        <!-- adicionando campos do formulário-->
        <input type="text" placeholder="Usuário:" id="usuario" required="true"
          minlength="3" maxlength="15">

        <input type="password" placeholder="Senha:" id="senha" required="true"
          minlength="3" maxlength="8">

        <!--adicionando botão que irá enviar as informações do site -->
        <input type="submit" value="Entrar">
      </form>
    </div>
  </div>
</body>
```

# Abra o arquivo index.html no navegador

Deverá estar similar ao da imagem abaixo



Usuário:

Senha:

Entrar




Preencha este campo.

`required="true"`

Quando adicionamos  
obrigatoriedade em um campo no  
HTML ele precisa ser preenchido

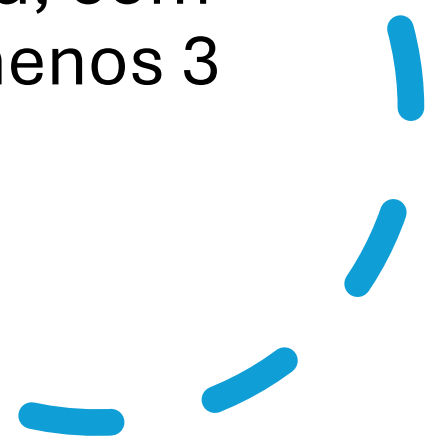
---

Entrar


 Aumente este texto para 3 caracteres ou mais. No momento, você está usando 2 caracteres).

minLength=3

- Utilizamos o parâmetro acima, com isso temos que digitar pelo menos 3 caracteres no campo







O site está sem  
funcionalidade...



# Javascript

---

- Para isso, utilizaremos a linguagem de programação Javascript. Linguagens de programação nos permitem criar funcionalidades lógicas, seja para site, aplicativo ou sistemas locais.

A large, bold, black 'JS' logo is centered on a bright yellow square background. The letters are thick and stylized, with the 'J' having a curved bottom and the 'S' being a simple, rounded 'S'.

HTML, CSS e  
JavaScript



**HTML**



**CSS**

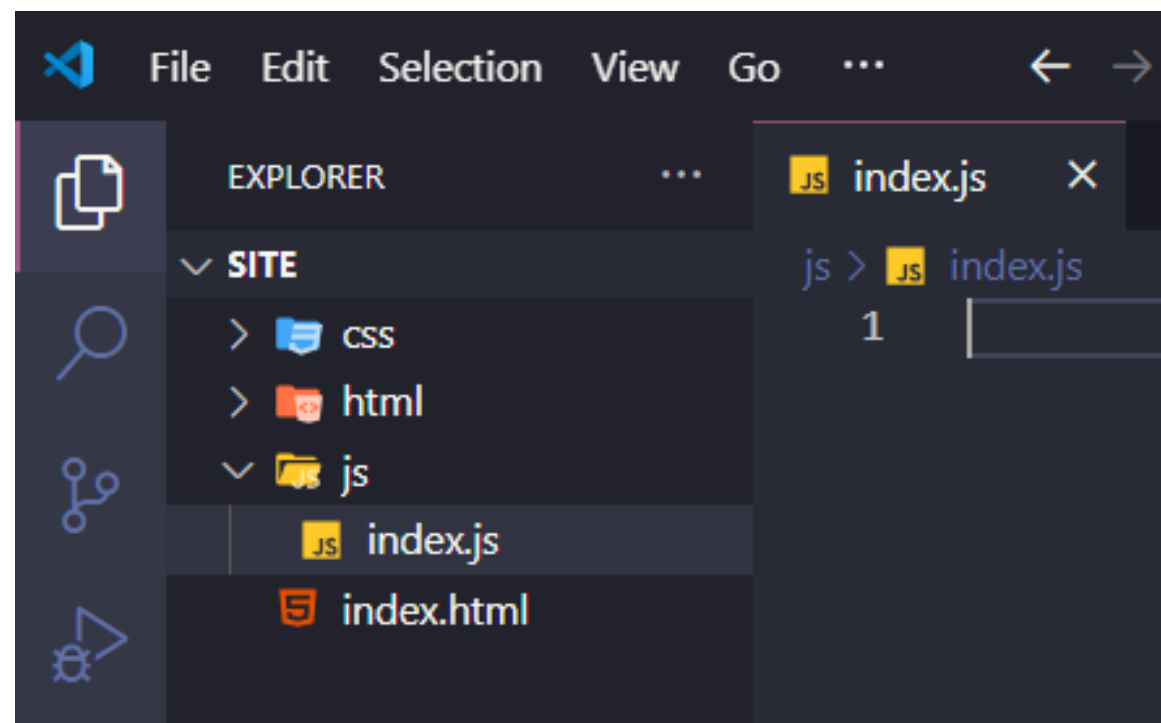


**Javascript**



Dentro da pasta “js” crie o arquivo  
“index.js”

# index.js





## Depois de criar o arquivo Javascript

- Acesse o arquivo index.html e vincule a página WEB ao index.js, isso significa que toda programação daquela página, por enquanto, virá daquele script.

```
index.html > html > head > script
1 <html lang="pt-br">
2 <head>
3     <!--Serve par setar o UNICODE do site para o usado em PT-BR-->
4     <meta charset="UTF-8"/>
5     <title>Tela de Login</title>
6
7     <!--vinculando tela com o script que vamos criar as funcionalidades-->
8     <script src="./js/index.js">*/script>
9
10 </head>
```

Pegando o valor que foi digitado no formulário HTML e armazenando dentro da variável da linguagem de programação através da função “logar()”

JS index.js

js > JS index.js > ...

```
1  function logar() {  
2      var usuario = document.getElementById('usuario').value;  
3      var senha = document.getElementById('senha').value;  
4  
5  }  
6  
7
```

# logar()

- É uma função em Javascript, funções servem para que possamos definir uma tarefa específica no código.
- Nesse caso, nossa função vai ser responsável por validar o usuário e a senha digitada, a função será chamada toda vez que o botão “entrar” for clicado no formulário HTML



OLHA O PONTO E VIRGULA!

;



# Vinculando a função JS com o botão HTML.

onclick="logar();return false"

```
<!--adicionando botão que irá enviar as informações do site -->  
<input type="submit" value="Entrar" onclick="logar();return false;">  
</form>
```

Ao clicar chama a função `logar()`, depois impede o site de ficar voltando pro formulário (vamos abrir a tela de login se tiver tudo ok, senão colocamos `return false`, ele não avança pra próxima tela)

index.js

js > index.js > ...

```
1  function logar() {  
2      var usuario = document.getElementById('usuario').value;  
3      var senha = document.getElementById('senha').value;  
4  
5  }  
6
```

Vamos continuar desenvolvendo a lógica da função...

Por enquanto ela só armazena os dados digitados no formulário nas variáveis JS, no próximo slide vamos realizar uma validação.

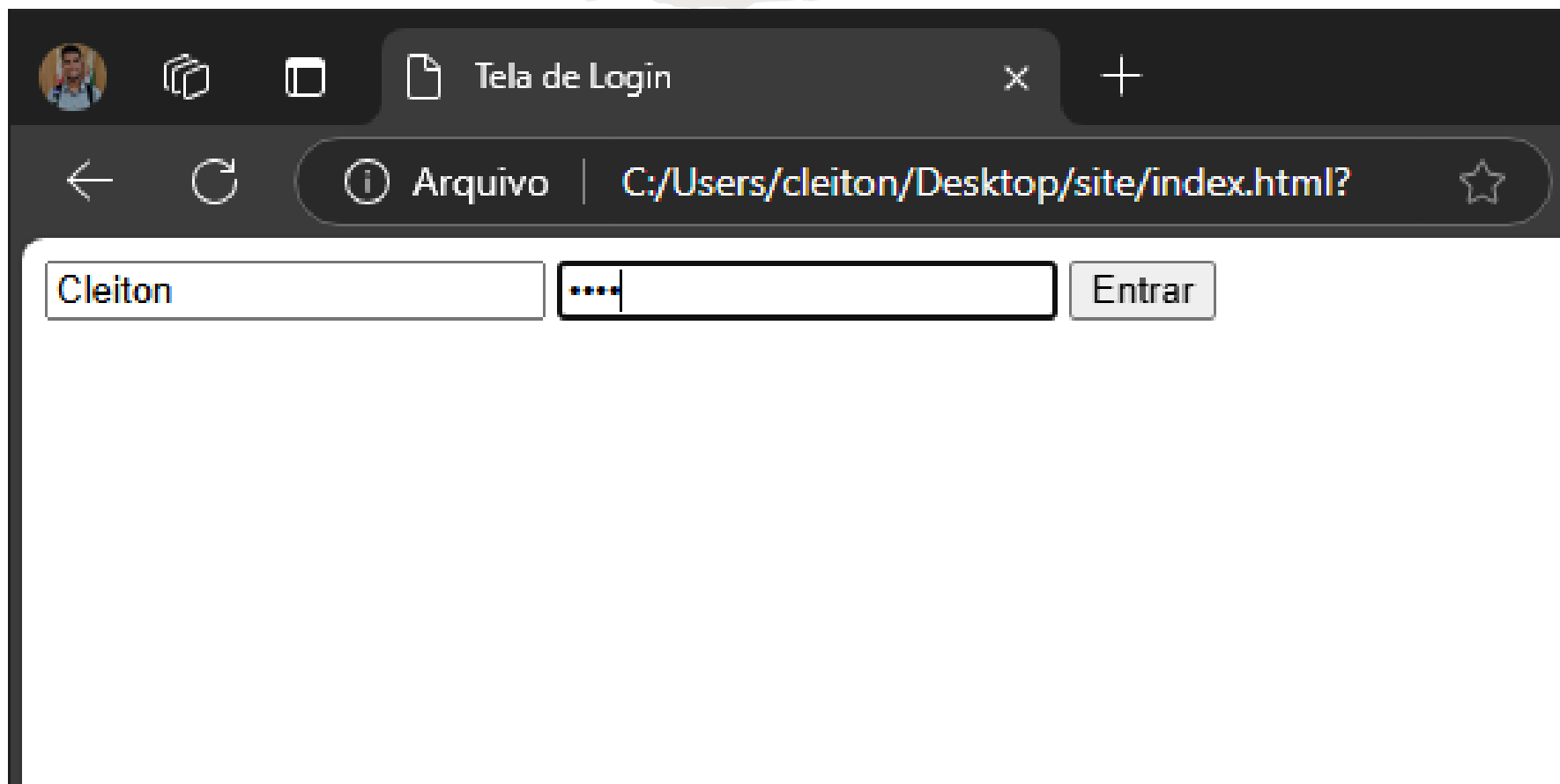
# Validando login

Se o usuário e a senha for igual a 'admin' vamos retornar 'OK'.

```
JS index.js X
js > JS index.js > ...
1  function logar() {
2      var usuario = document.getElementById('usuario').value;
3      var senha = document.getElementById('senha').value;
4
5      if (usuario == 'admin' && senha == 'admin') {
6          alert('LOGIN OK - SEJA BEM VINDO!');
7      } else {
8          alert('Usuário ou Senha incorretos!');
9      }
10 }
11
12
```

Digitei “Cleiton” e a senha “1234”, o que deve acontecer?

Para testar, abra o index.html no navegador.



Deve aparecer a mensagem de erro!

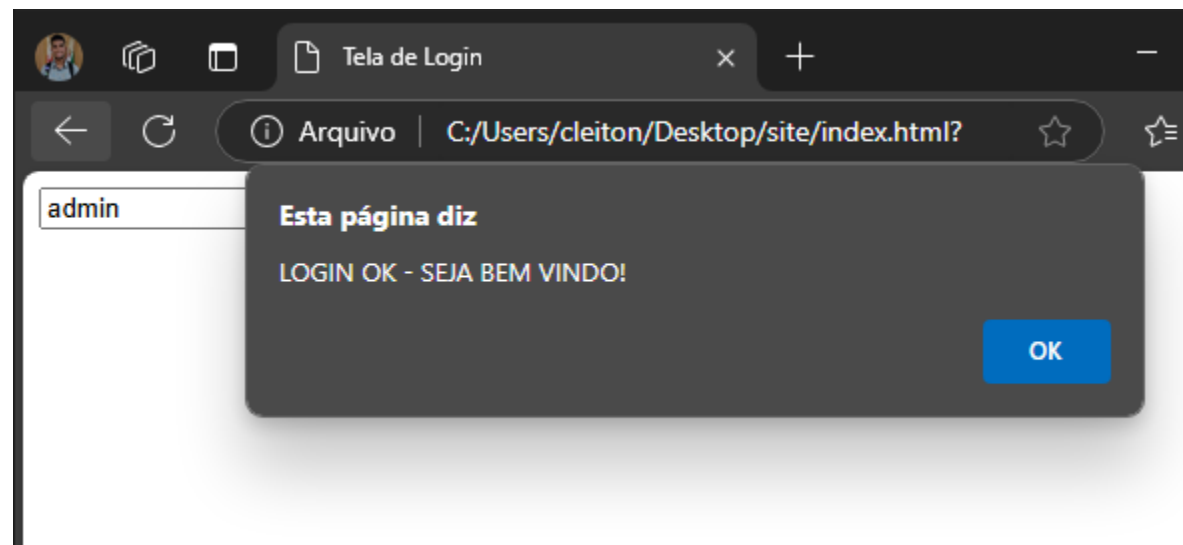
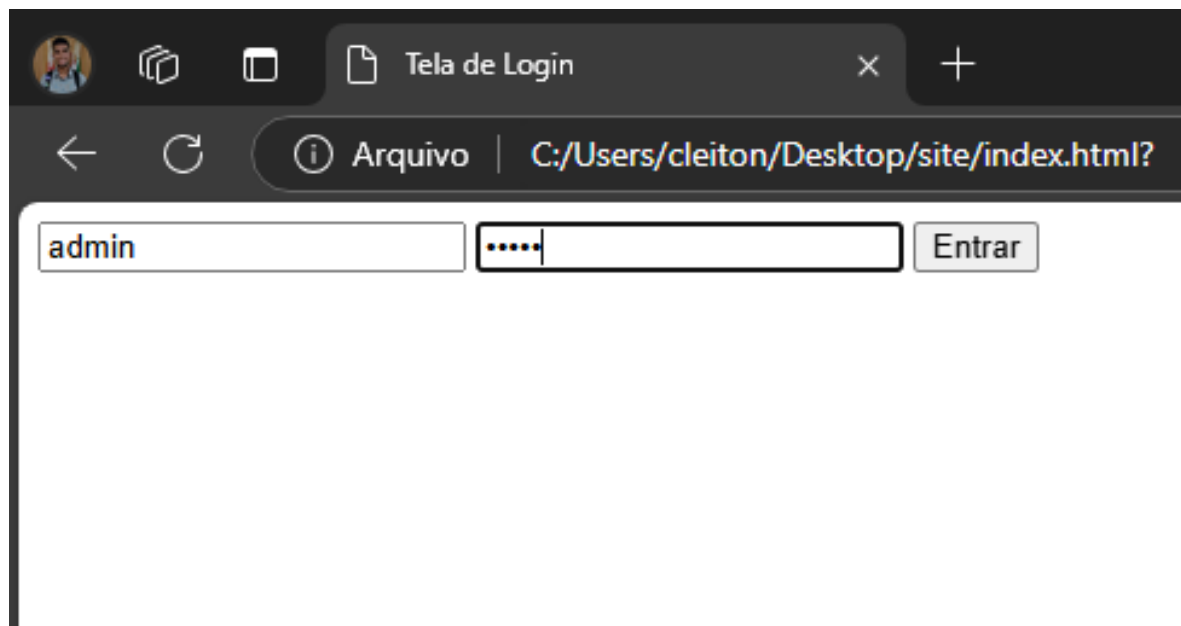
**Esta página diz**

**Usuário ou Senha incorretos!**

OK

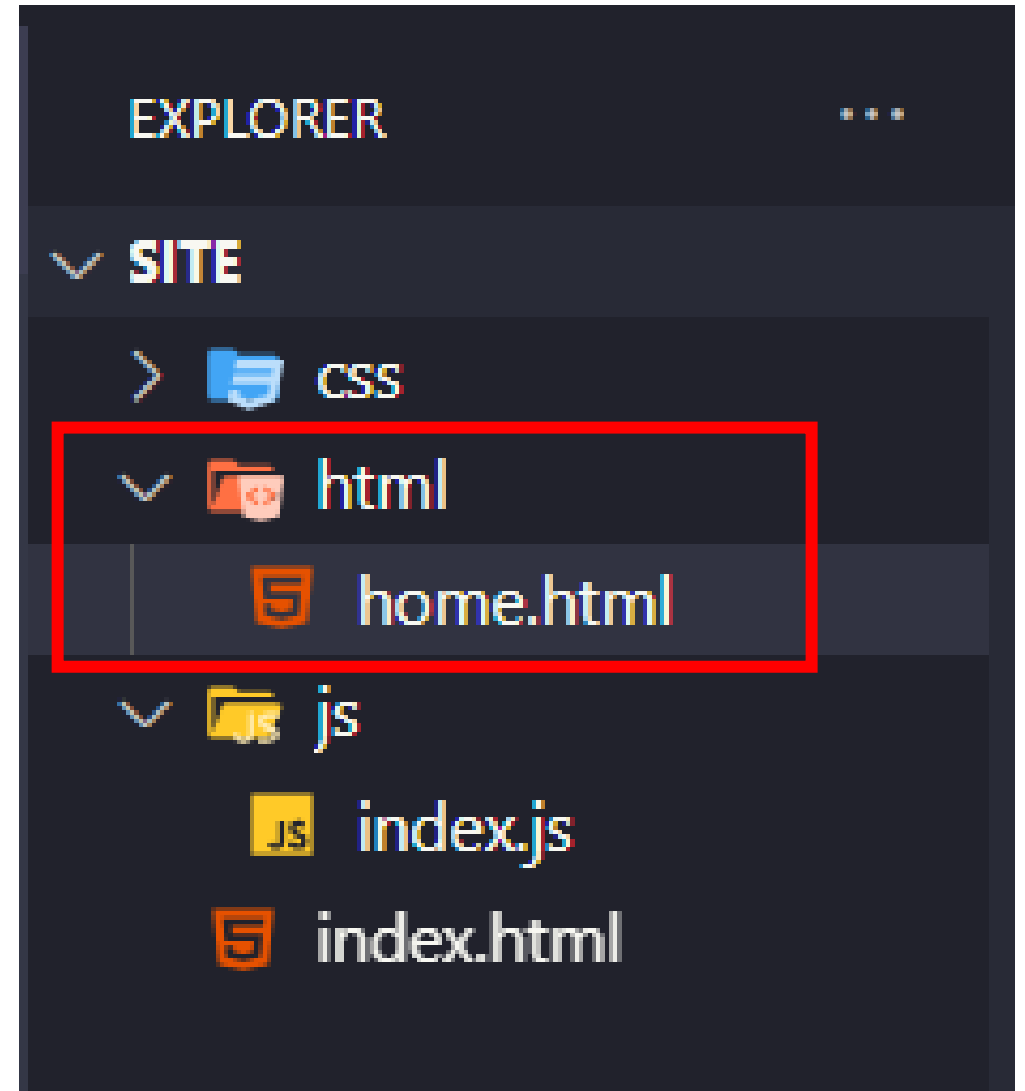
# ‘admin’

Vamos testar a senha e o usuário que definimos



# Agora vamos criar a página inicial do site

- Após realizar o login, o usuário será redirecionado para a “**home.html**”
- Dentro da pasta “**html**” crie o arquivo “**home.html**”



Adicionando  
o conteúdo  
na tela nova

home.html X

html > home.html > html

```
1  <html>
2    <head>
3      <meta charset="UTF-8"/>
4      <title>Tela de Login</title>
5    </head>
6    <body>
7      <h1>Olá, seja bem vindo!</h1>
8      <br/>
9      <hr/>
10     <h2>Página inicial do nosso site.</h2>
11   </body>
12 </html>
```





Essa será a  
página inicial

---

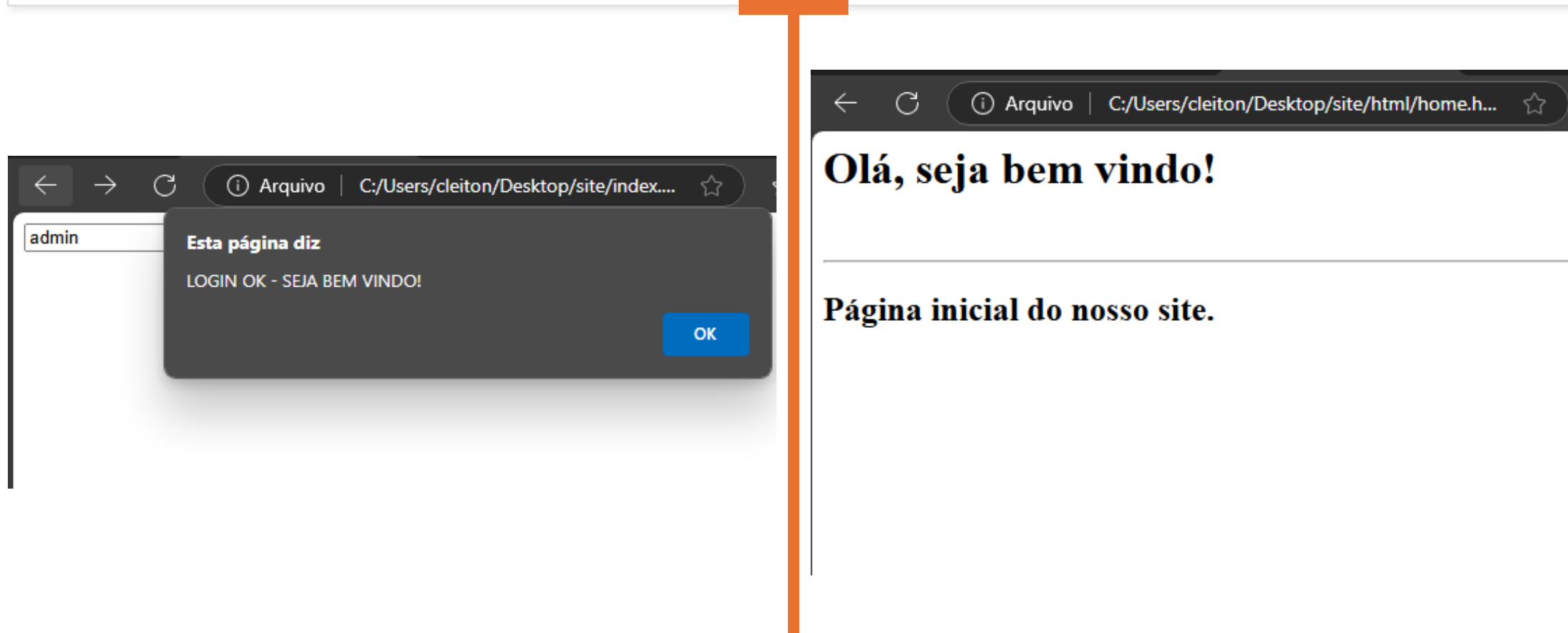


Se o login estiver correto, vamos direcionar o usuário para a página que acabamos de criar

```
JS index.js X
js > JS index.js > ...
1  function logar() {
2      var usuario = document.getElementById('usuario').value;
3      var senha = document.getElementById('senha').value;
4
5      if (usuario == 'admin' && senha == 'admin') {
6          alert('LOGIN OK - SEJA BEM VINDO!');
7          location.href = "../html/home.html";
8      } else {
9          alert('Usuário ou Senha incorretos!');
10     }
11 }
12
```

# Ao realizar o login corretamente (admin)

Após realizar o login com usuário e senha 'admin', o usuário será redirecionado para a tela inicial

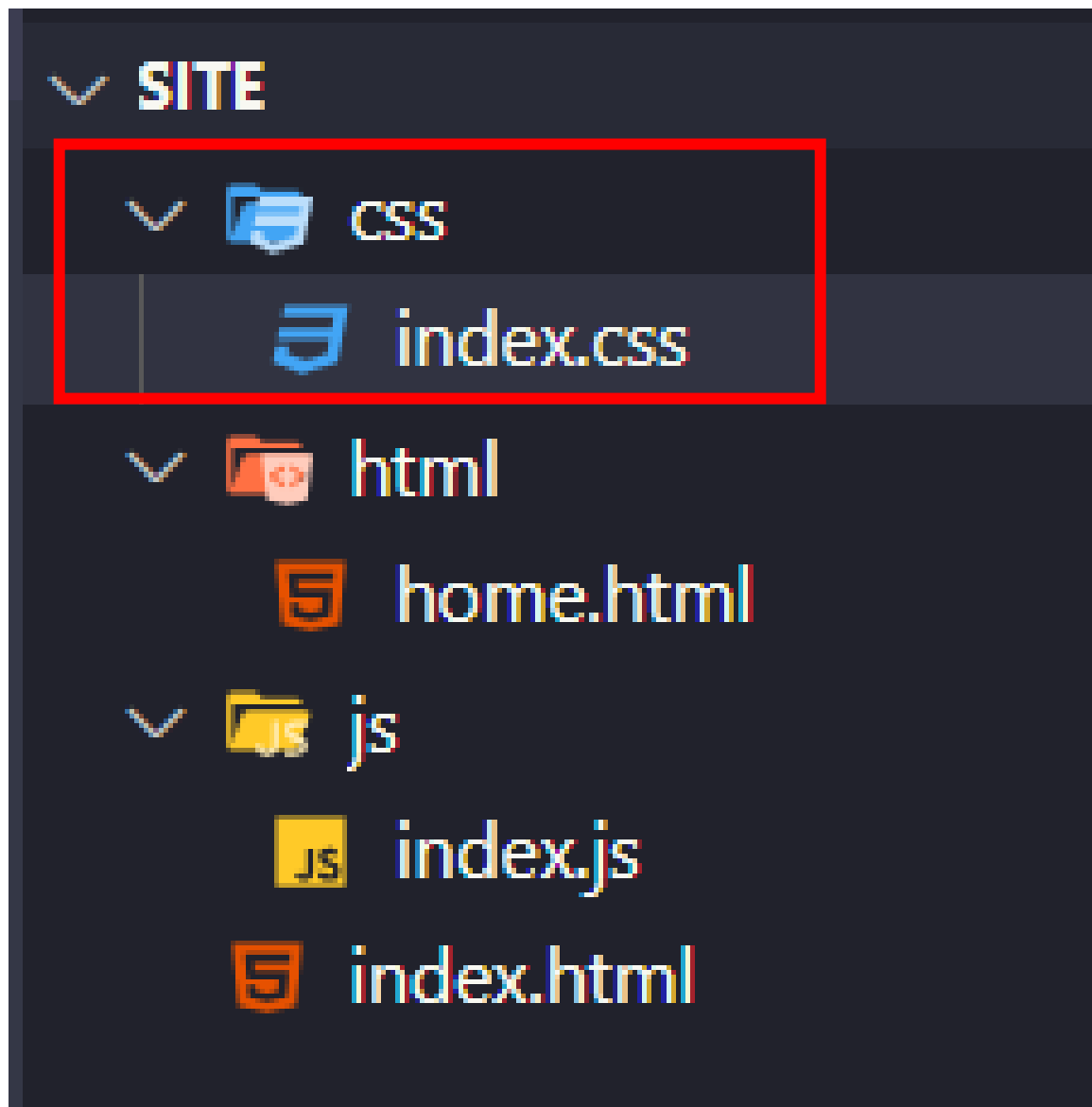


Agora vamos  
estilizar nossa  
tela de login

**CSS**



Crie o  
arquivo  
index.css

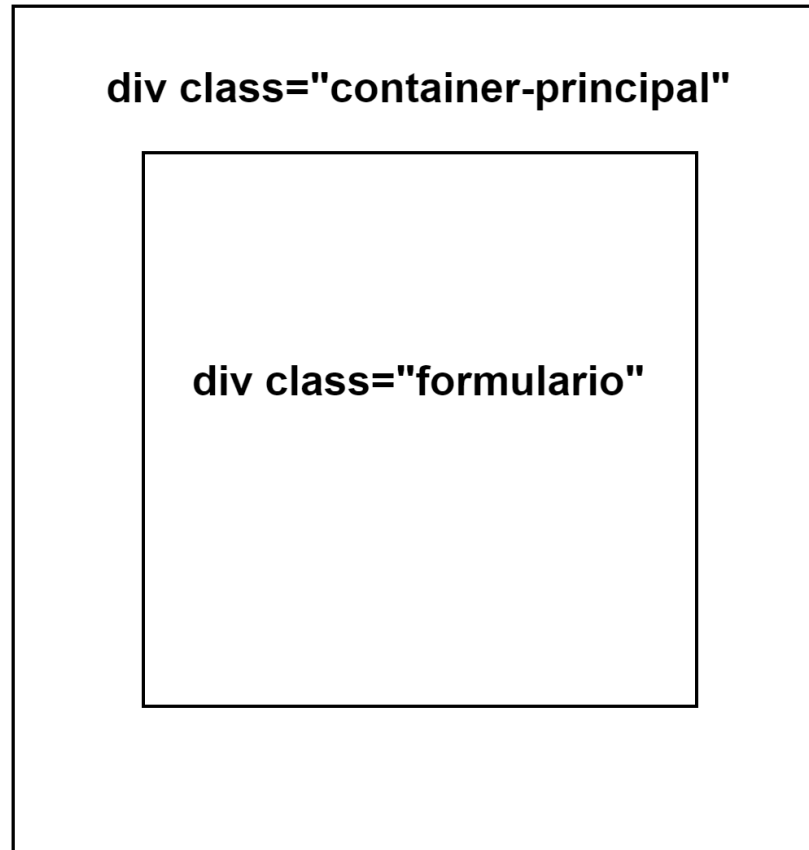


# Vinculando a tela de login, com o CSS

index.html > html > head

```
1  <html lang="pt-br">
2    <head>
3      <!--Serve par setar o UNICODE do site para o usado em PT-BR-->
4      <meta charset="UTF-8"/>
5      <title>Tela de Login</title>
6      <title>Seja bem-vindo</title>
7
8      <!--vinculando a tela, com o CSS para estilizar a página-->
9      <link rel="stylesheet" type="text/css" href="._css/index.css">
10
11      <!--vinculando tela com o script que vamos criar as funcionalidades-->
12      <script src="._js/index.js"></script>
13
14    </head>
```

# No HTML, definimos essas divs (containers)



Agora no CSS,  
vamos  
organizar  
nossa página.





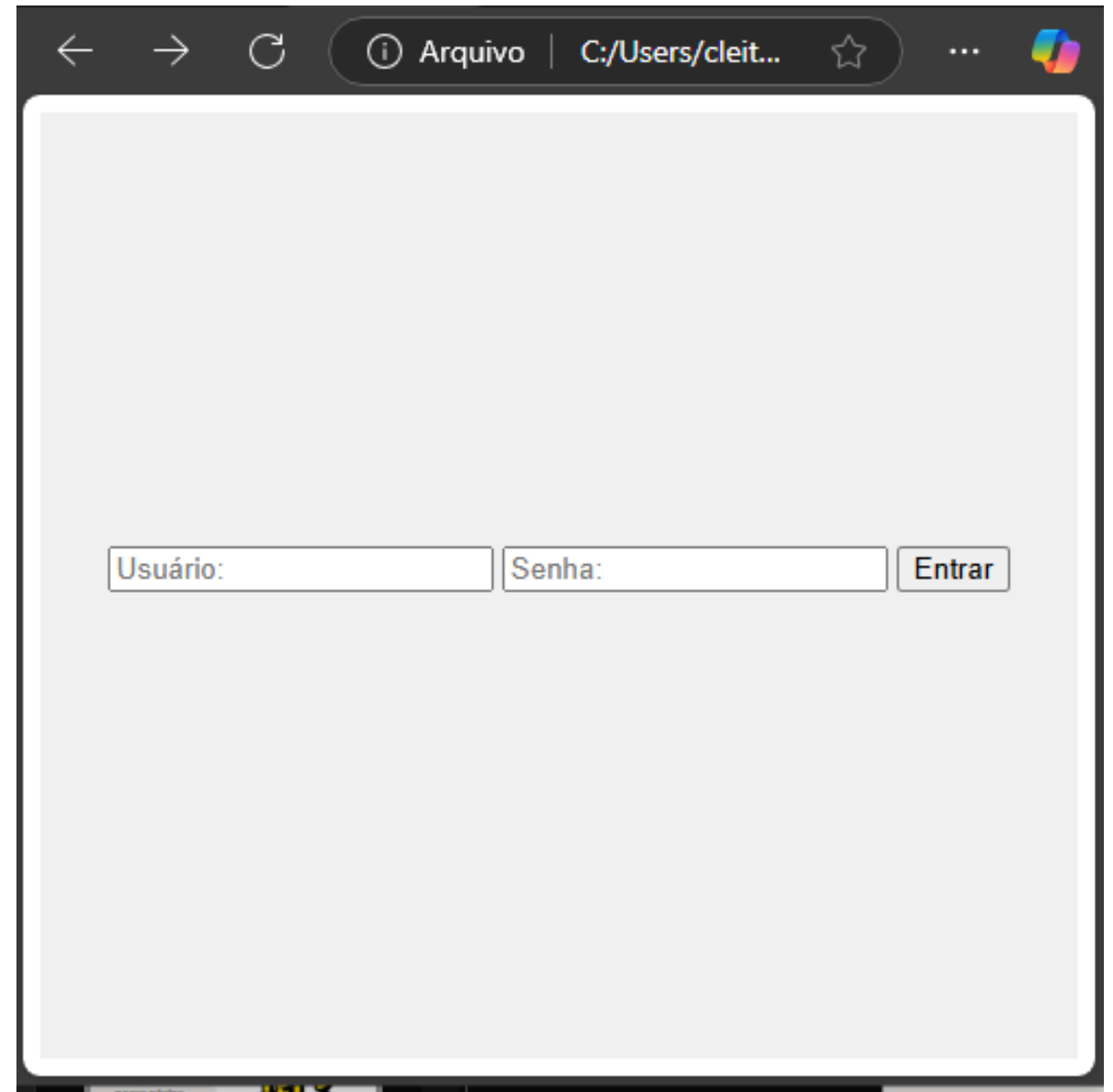
index.css

css > index.css > .container-principal

```
1  /* Definindo o estilo para centralizar o formulário */
2  .container-principal {
3      display: flex;                /* Usando Flexbox */
4      justify-content: center;      /* Centraliza horizontalmente */
5      align-items: center;          /* Centraliza verticalmente */
6      height: 100%;                /* Altura total da tela */
7      background-color: #f0f0f0;    /* Cor de fundo suave */
8  }
```

Vamos usar flexbox para deixar o login responsivo e centralizado

Centralizamos a  
tela, por conta  
que mexemos no  
container  
principal



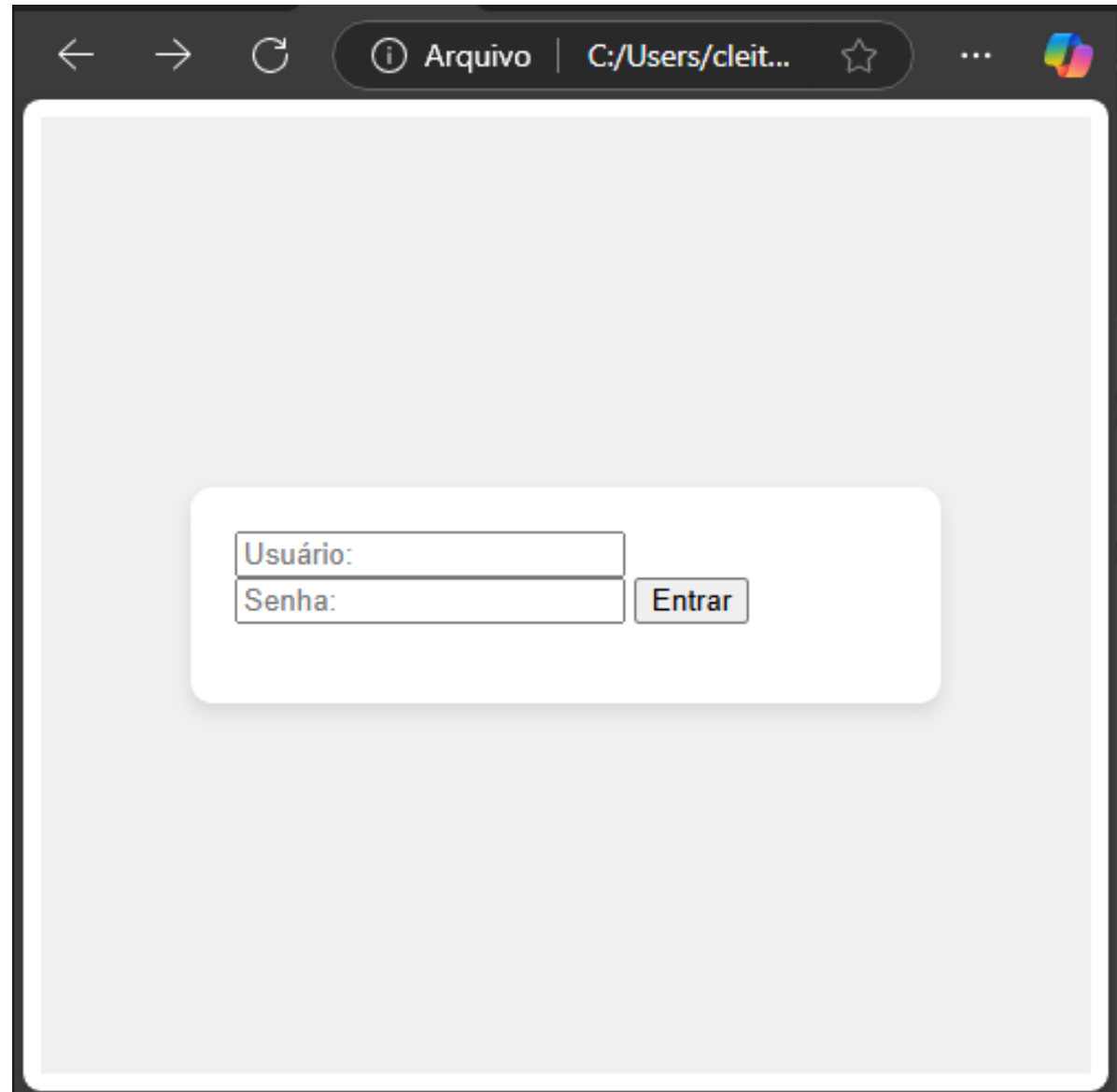
## Agora, vamos mexer no espaço do formulário

```

9
10 .formulario {
11     background-color: #fff; /* Cor de fundo branca para o formulário */
12     padding: 20px; /* Espaçamento interno */
13     border-radius: 10px; /* Bordas arredondadas */
14     box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1); /* Sombra suave */
15     width: 300px; /* Largura do formulário */
16 }

```

*.formulario*



A screenshot of a web browser window. The address bar shows the path "C:/Users/cleit..." and the word "Arquivo" is visible. The main content area is light gray and contains a white rounded rectangle with a shadow. Inside this rectangle is a login form with two input fields: "Usuário:" and "Senha:". To the right of the "Senha:" field is a button labeled "Entrar".

Usuário:	
Senha:	

Entrar

Mexemos nos espaços, agora vamos estilizar os componentes que ficam nesses espaços!

Centralizando o título do formulário

```
17
18 form h3 {
19     text-align: center;           /* Alinha o título ao centro */
20     margin-bottom: 20px;         /* Espaçamento abaixo do título */
21 }
22
```

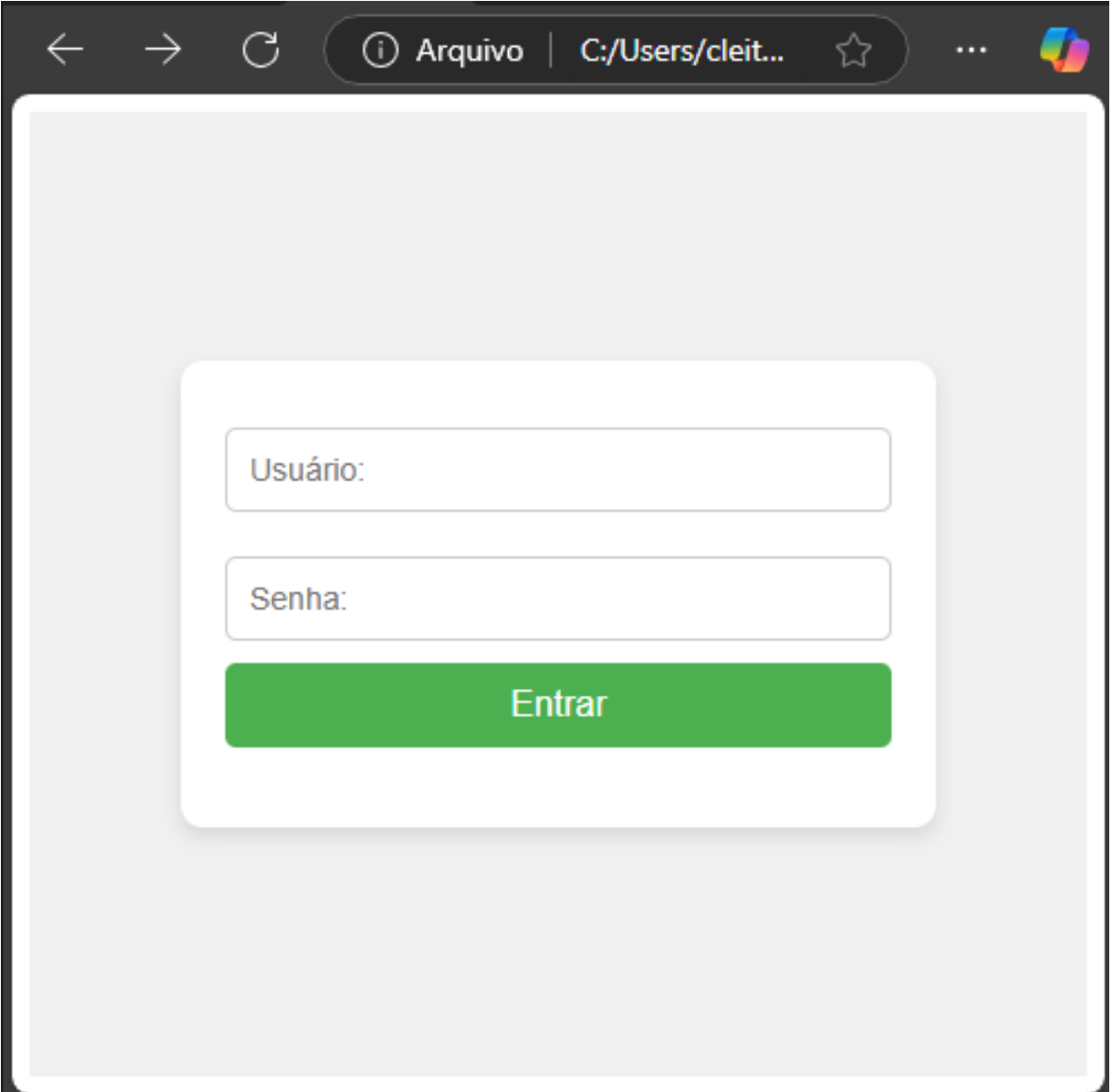
```
23 input[type="text"], input[type="password"] {
24     width: 100%;           /* Largura total do campo */
25     padding: 10px;        /* Espaçamento interno */
26     margin: 10px 0;       /* Espaçamento entre os campos */
27     border: 1px solid #ccc; /* Borda suave */
28     border-radius: 5px;    /* Bordas arredondadas */
29     font-size: 14px;       /* Tamanho da fonte */
30 }
31
```

Agora vamos mexer  
nos campos (login,  
senha)

## Restou, o componente “botão” (submit)

```
32  ✓ input[type="submit"] {  
33      width: 100%;           /* Largura total do botão */  
34      padding: 10px;        /* Espaçamento interno */  
35      background-color: #4CAF50; /* Cor de fundo verde */  
36      color: white;          /* Cor do texto */  
37      border: none;          /* Remove a borda padrão */  
38      border-radius: 5px;    /* Bordas arredondadas */  
39      font-size: 16px;       /* Tamanho da fonte */  
40      cursor: pointer;       /* Cursor de mãozinha */  
41  }  
42
```

Verificando a  
nossa tela de  
login



A screenshot of a web browser window displaying a login form. The browser's address bar shows the path "C:/Users/cleit..." and the word "Arquivo". The login form is centered on a light gray background and consists of a white rounded rectangle containing two input fields and a button. The first input field is labeled "Usuário:" and the second is labeled "Senha:". Below these fields is a green button with the text "Entrar".

← → ↻ ⓘ Arquivo | C:/Users/cleit... ☆ ...

Usuário:

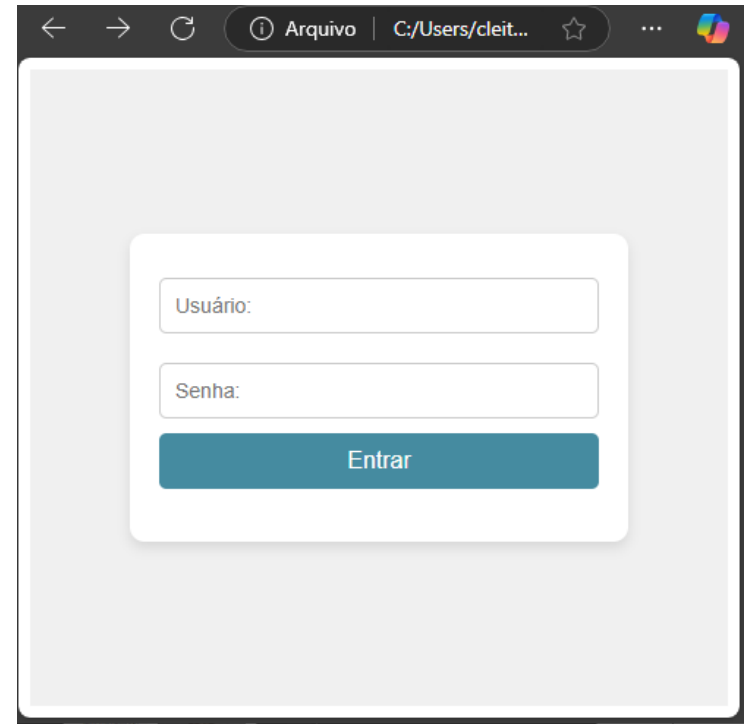
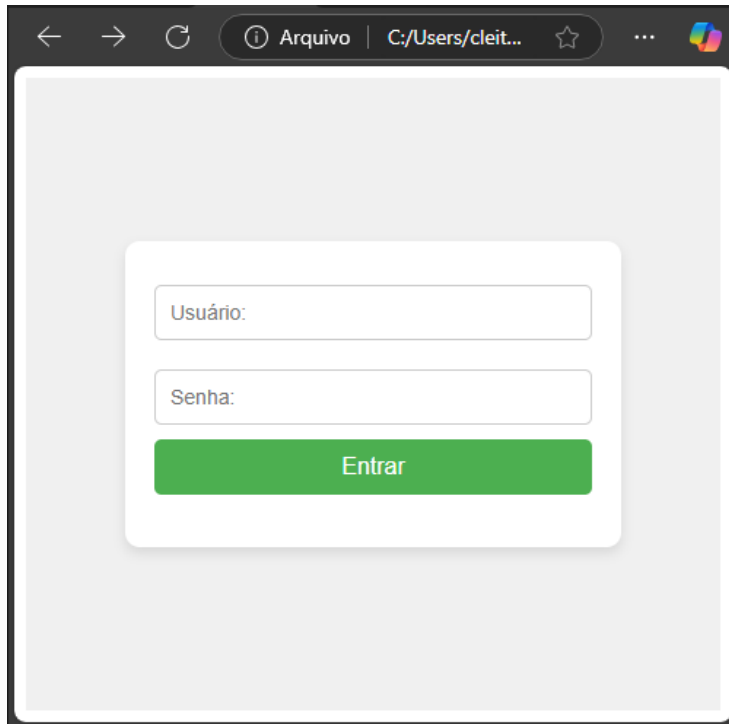
Senha:

Entrar



Muito Toooooop!  
Sem framework!





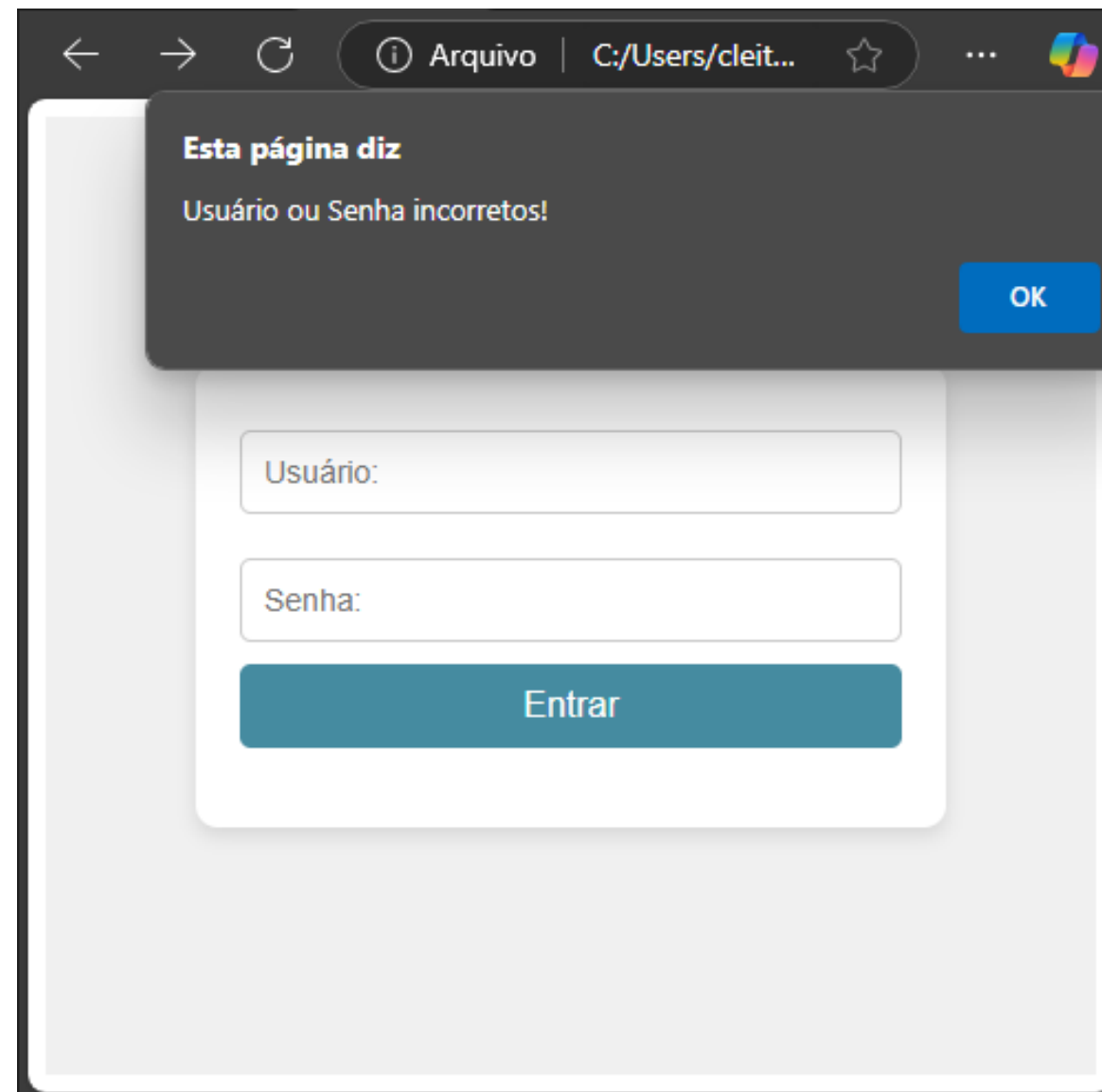
Mudar de cor ao  
passar o mouse

• -

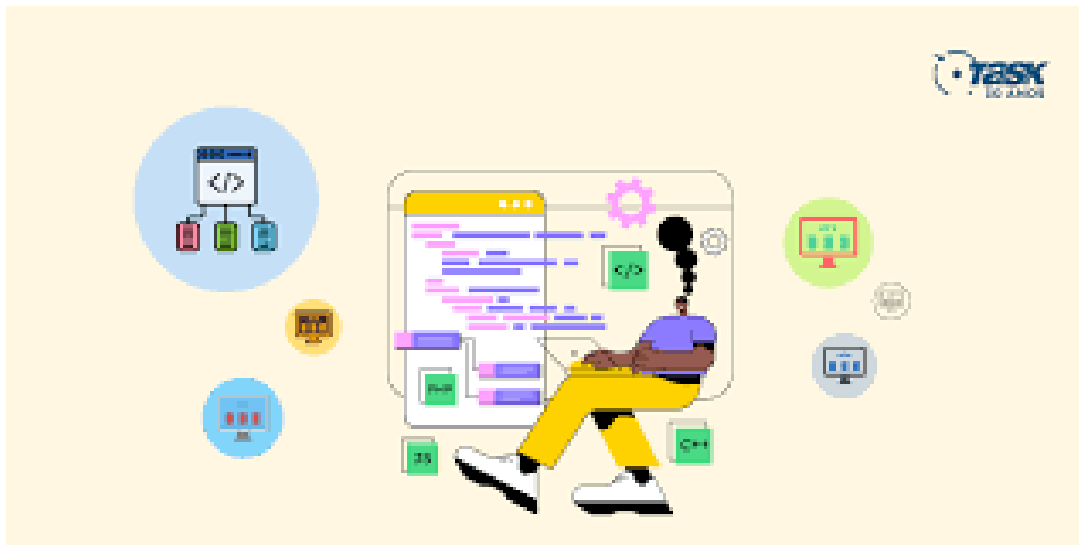
# hover

```
41
42 input[type="submit"]:hover {
43     background-color: #458ba0;    /* Muda a cor do botão ao passar o mouse */
44 }
45
```

Ao clicar no botão,  
vamos estilizar a  
mensagem enviada  
pelo Javascript

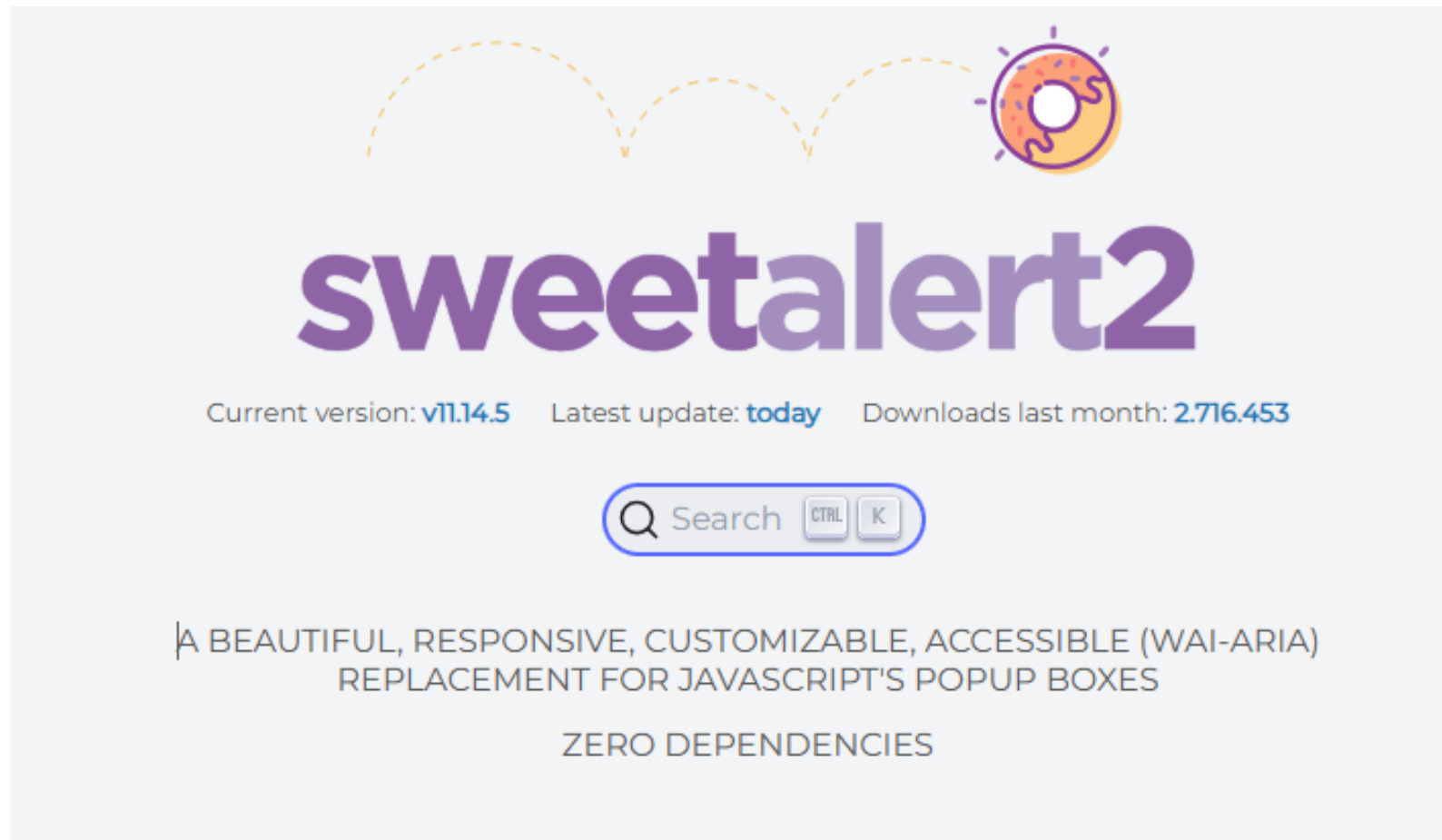


# Framework



- São ferramentas que nos auxiliam no desenvolvimento otimizando o tempo, ao invés de construir tudo do zero, os frameworks nos entregam um molde do que queremos desenvolver.
- Nesse caso, vamos desenvolver um modal para a tela de login usando Javascript com o SweetAlert

Para isso vamos usar o framework sweetalert2



# Vinculando a nossa tela, ao framework

```
index.html > html > head
1  <html lang="pt-br">
2    <head>
3      <!--Serve para setar o UNICODE do site para o usado em PT-BR-->
4      <meta charset="UTF-8"/>
5      <title>Tela de Login</title>
6      <title>Seja bem-vindo</title>
7
8      <!--vinculando a tela, com o CSS para estilizar a página-->
9      <link rel="stylesheet" type="text/css" href="./css/index.css">
10
11     <!--vinculando tela com o script que vamos criar as funcionalidades-->
12     <script src="./js/index.js"></script>
13
14     <!-- Para usar o SweetAlert2 CDN -->
15     <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
16
17  </head>
```

**<script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>**

```
index.html > html > head
1  <html lang="pt-br">
2    <head>
3      <!--Serve para setar o UNICODE do site para o usado em PT-BR-->
4      <meta charset="UTF-8"/>
5      <title>Tela de Login</title>
6      <title>Seja bem-vindo</title>
7
8      <!--vinculando a tela, com o CSS para estilizar a página-->
9      <link rel="stylesheet" type="text/css" href="./css/index.css">
10
11     <!--vinculando tela com o script que vamos criar as funcionalidades-->
12     <script src="./js/index.js"></script>
13
14     <!-- Para usar o SweetAlert2 CDN -->
15     <script src="https://cdn.jsdelivr.net/npm/sweetalert2@11"></script>
16
17  </head>
```



js > JS index.js > logar

```
1  function logar(event) {  
2  
3      event.preventDefault();  
4  
5      var usuario = document.getElementById('usuario').value;  
6      var senha = document.getElementById('senha').value;  
7  
8      if (usuario == 'admin' && senha == 'admin') {  
9  
10  
11          location.href = "../html/home.html";  
12      } else {  
13  
14  
15      }  
16 }
```

# Vamos adicionar o event como parametro

Em seguida, forçamos o envio do formulário (event) a acessar nossa função

# Desenvolvendo o modal com o Swal

Cada item dentro do “fire” é uma coisa que irá aparecer dentro do modal

```
js > JS index.js > logar
1  function logar(event) {
2      // Impede o envio tradicional do formulário
3      event.preventDefault();
4
5      var usuario = document.getElementById('usuario').value;
6      var senha = document.getElementById('senha').value;
7
8      if (usuario == 'admin' && senha == 'admin') {
9          Swal.fire({
10             title: 'Login realizado!',
11             text: 'Bem-vindo, ' + usuario + '!',
12             icon: 'success',
13             confirmButtonText: 'OK'
14         })
15     }
16
17 }
```



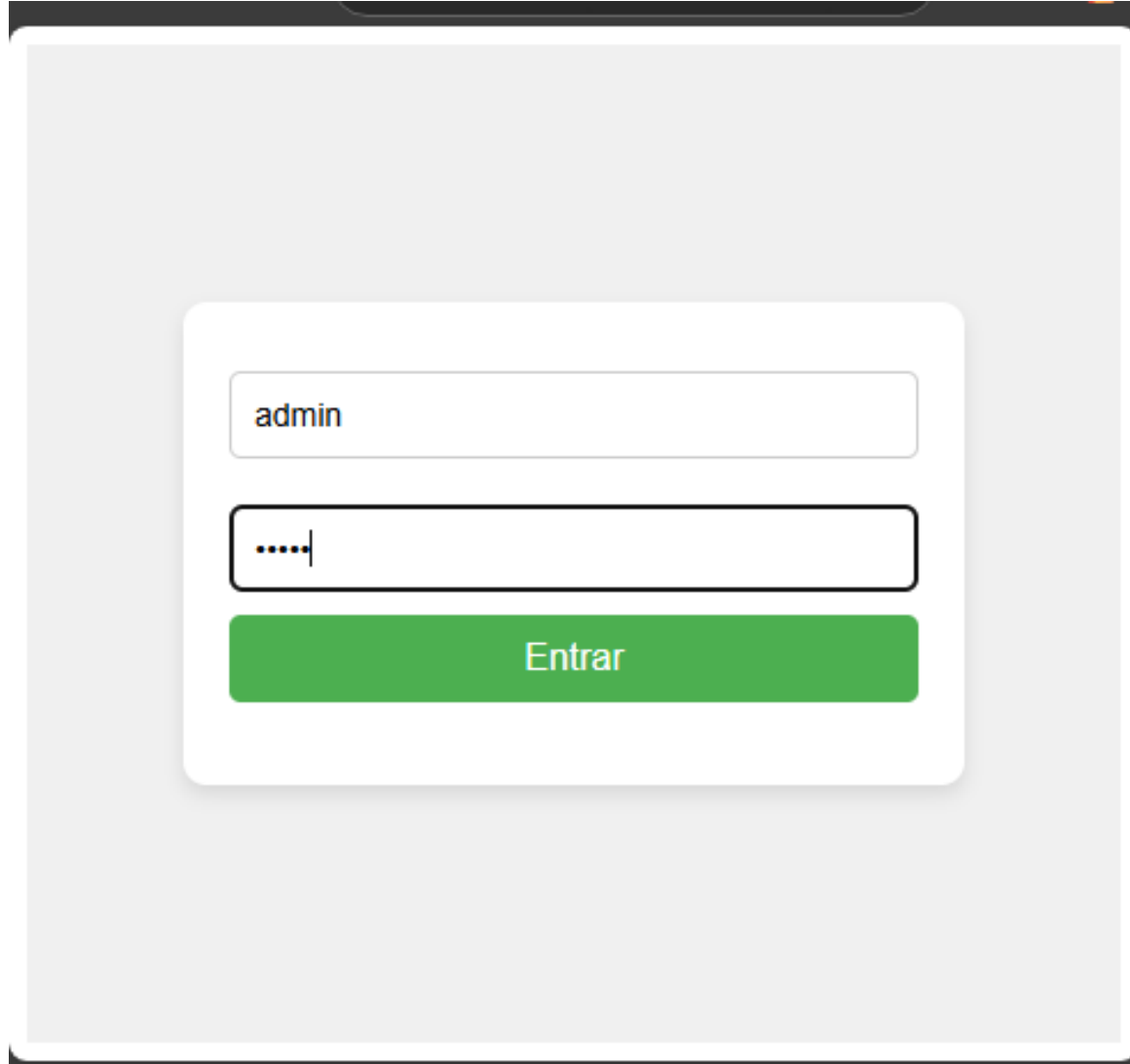
# Após o modal, vamos redirecionar para a tela HOME.html

```
if (usuario == 'admin' && senha == 'admin') {  
    Swal.fire({  
        title: 'Login realizado!',  
        text: 'Bem-vindo, ' + usuario + '!',  
        icon: 'success',  
        confirmButtonText: 'OK'  
    }).then(() => {  
        // Após o alerta ser fechado, adicionamos o delay para o redirecionamento  
        setTimeout(() => {  
            // Redireciona para home.html após 2 segundos (2000ms)  
            location.href = "./html/home.html";  
        }, 100); // Delay  
    });  
} else {  
}
```

```
<body>
  <!--Essa div, representa onde todo o conteúdo do site vai ficar-->
  <div class="container-principal">
    <!-- aqui é o espaço onde vamos criar o formulário-->
    <div class="formulario">
      <form id="form-login" onsubmit="logar(event)">
        <!-- adicionando campos do formulário -->
        <input type="text" placeholder="Usuário:" id="usuário">
        <input type="password" placeholder="Senha:" id="senha">
        <!--adicionando botão que irá enviar as informações-->
        <input type="submit" value="Entrar" onclick="logar(event)">
      </form>
    </div>
  </div>
</body>
```

Para testar,  
acesse o  
index.html

No formulário, vamos  
adicionar o:  
**onsubmit="logar(event)"**



A login form is displayed on a light gray background. The form consists of a white rounded rectangle containing three elements: a text input field with the username "admin", a password input field with masked characters ".....", and a green button labeled "Entrar".

Testando,  
inserindo login  
corretamente...



**Login realizado!**

Bem-vindo, admin!

OK

Com o login  
correto,  
olha o Swal  
em ação



Clicando em  
“OK”, será  
direcionado  
a tela Home



Porém, precisamos desenvolver o modal, caso a senha ou o login estejam incorretos.

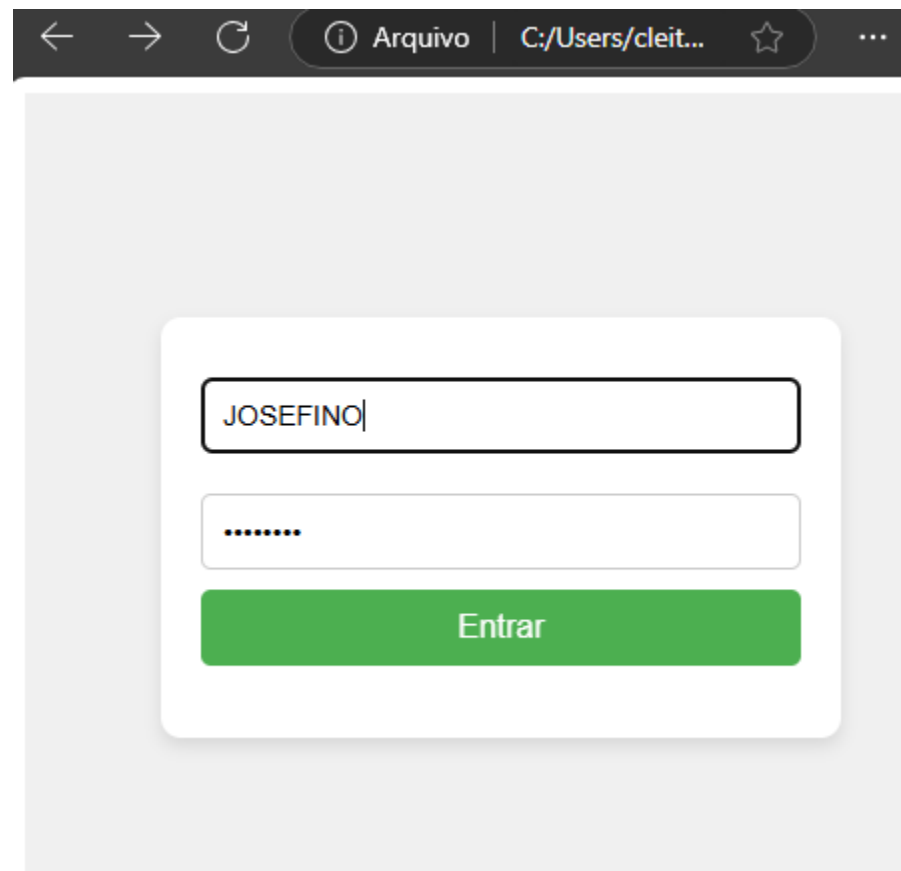
No **ELSE**

```
21     } else {  
22         Swal.fire({  
23             title: 'Erro!',  
24             text: 'Usuário ou senha incorretos.',  
25             icon: 'error',  
26             confirmButtonText: 'Tentar novamente'  
27         });  
28     }  
29 }  
30 }  
31
```

# IF/ELSE

```
if (usuario == 'admin' && senha == 'admin') {  
  Swal.fire({  
    title: 'Login realizado!',  
    text: 'Bem-vindo, ' + usuario + '!',  
    icon: 'success',  
    confirmButtonText: 'OK'  
  }).then(() => {  
    // Após o alerta ser fechado, adicionamos o delay  
    setTimeout(() => {  
      // Redireciona para home.html após 2 segundos  
      location.href = "./html/home.html";  
    }, 1000); // Delay  
  });  
} else {  
  Swal.fire({  
    title: 'Erro!',  
    text: 'Usuário ou senha incorretos.',  
    icon: 'error',  
    confirmButtonText: 'Tentar novamente'  
  });  
}
```

# Testando o “else”



A screenshot of a web browser window. The address bar shows the path `C:/Users/cleit...`. The main content area displays a login form with a white background and rounded corners. The form contains two input fields: the first is labeled "Nome" and contains the text "JOSEFINO"; the second is labeled "Senha" and contains a masked password ".....". Below the input fields is a green button labeled "Entrar". A green message box at the top of the form displays the text "Bem-vindo ao sistema".

## Modal de erro



**Erro!**

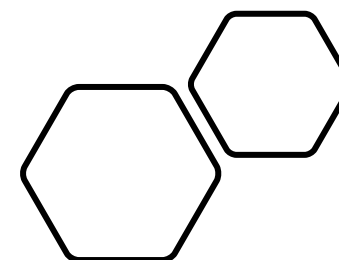
Usuário ou senha incorretos.

Tentar novamente

**EM CASO DE DÚVIDA:**



**LEIA OS SLIDES NOVAMENTE**





# ETEC UIRAPURU

- Cleiton S Dias
- Fábio Claret
- Thiago Pascotto
- Sueli Muniz