

Nomes: Murilo Costa Bittencourt, Gabriel Vieira Moraes e Carlos Eduardo.

1 Propósito do site

O site tem como objetivo apresentar a linguagem AIML, fala um pouco sobre o porquê de sua criação e quem foi seu criador, e conta com exemplos de onde a linguagem foi utilizada.

2 Index/navbar

Foram montados de maneira responsiva utilizando media queries e java script para alterar as propriedades do CSS das tags, o padrão de telas que foram escolhidas de até 750px após isso foi até 999px e depois disso entrou no padrão:

2.1 JS

O java script foi utilizado para fazer a navbar responsiva e o slides. Cada um desses elementos foram criados classes no java script

Para a classe MobileNavbar foi criado um construtor onde passo por parametro o nome das classes css que utilizarei para adicionar propriedades, logo adiante foi criado método responsável por ao clicar no botão ser executado, dentro desse método buttonClick ele realiza a chamada dos métodos de animação onde no mesmo é responsável por alterar propriedades css.

```
class MobileNavbar{
  constructor(mobileOption, navItems, navlinks){
    this.mobileMenu = document.querySelector(mobileOption);
    this.navItems = document.querySelector(navItems);
    this.navLinks = document.querySelectorAll(navlinks);
    this.activeClass = "active";
    this.buttonIndexRight = document.getElementsByClassName("proximo");
    this.buttonHeaderNone = "Veja exemplos do mundo real do GitHub";
    this.buttonHeader = this;
    this.buttonClick = this.buttonClick.bind(this);
  }
  buttonClick(){
    this.navItems.classList.toggle(this.activeClass);
    this.mobileMenu.classList.toggle(this.activeClass);
    this.animacaoLinks();
    if(this.buttonIndexRight[0].getAttribute("style") == this.buttonHeaderNone)
      this.buttonIndexRight[0].setAttribute("style", this.buttonHeader);
    else
      this.buttonIndexRight[0].setAttribute("style", this.buttonHeaderNone);
  }
  animacaoLinks(){
    this.navLinks.forEach((link, index) => {
      link.style.animation
        ? (link.style.animation = "")
        : (link.style.animation = `navLinkFade 0.5s ease forwards ${index / this.navLinks.length + 0.2}s`);
    });
  }
  addClickEvent(){
    this.mobileMenu.addEventListener("click", this.buttonClick);
  }
  init(){
    if(this.mobileMenu){
```

Já na classe SliderItens no construtor eu declaro a posição do slide inicial e defino o qual o nome da classe que adicionei os itens do meu slider(nome da variável está como id)

```
45 class SliderItens{
46   constructor(id){
47     this.slideIndex = 1;
48     this.slideId = id;
49   }
50
51
52   init(){
53     /*Caso queira retornar algo*/
54     this.viewSlide(1);
55     return this;
56   }
57
58   viewSlide(n){
59     let i;
60     let slides = document.getElementsByClassName(this.slideId);
61     if (n > slides.length) {
62       this.slideIndex = 1;
63     }
64     if(n < 1 ){
65       this.slideIndex = slides.length;
66     }
67     for (i = 0; i < slides.length; i++) {
68       slides[i].style.display = "none";
69     }
70
71     slides[this.slideIndex - 1].style.display = "block";
72   }
73 }
74
```

Para o metodo viewSlide eu chamo as funções abaixo para realizar a troca de imagem que aparece no meu slide ao clicar nos botões.

```
82
83   mobileNavbar.init();
84
85   /*Iniciando obj navbar*/
86   const sliderItens = new SliderItens("slide-item");
87   sliderItens.init();
88
89   function nextSlider(n){
90     sliderItens.viewSlide(sliderItens.slideIndex+=n);
91   }
```

Dificuldades:

A principal foi posicionar os elementos na página de maneira responsiva e que ficasse na posição que fosse mais adequada.

3 Login/Cadastro

As páginas de login e de cadastro foram desenvolvidas de maneira responsiva utilizando as unidades relativas “VH” e “WH”, foi utilizado as tags padrões de formulário HTML e a tag “div” facilitar a utilização do CSS. Foi utilizado o mesmo CSS nas páginas de login e cadastro.

3.1 Login

```
<main id="login">
  <form class="card">
    <div class="card-header">
      <h2 id="h2_login">Login</h2>
    </div>
    <div class="card-content">
      <div class="card-content-area">
        <label for="usuario">Usuário</label>
        <input type="text" id="usuario" autocomplete="off">
      </div>
      <div class="card-content-area">
        <label for="password">Senha</label>
        <input type="password" id="password" autocomplete="off">
      </div>
    </div>
    <div class="card-footer">
      <input type="submit" value="Entrar" class="submit">
      <a href="#" class="criar_cadastro">Não possui cadastrado?</a>
    </div>
  </form>
</main>
```

3.2 Cadastro

```
<main id="login">
  <form class="card">
    <div class="card-header">
      <h2 id="h2_login">Cadastro</h2>
    </div>
    <div class="card-content">
      <div class="card-content-area">
        <label for="usuario">Nome de Usuário</label>
        <input type="text" id="usuario" autocomplete="off">
      </div>
      <div class="card-content-area">
        <label for="email">E-mail</label>
        <input type="email" id="email" autocomplete="off">
      </div>
      <div class="card-content-area">
        <label for="phone">Telefone</label>
        <input type="tel" id="phone" autocomplete="off">
      </div>
      <div class="card-content-area">
        <label for="password">Senha</label>
        <input type="password" id="password" autocomplete="off">
      </div>
      <div class="card-content-area">
        <label for="confirm_password">Confirmar Senha</label>
        <input type="password" id="confirm_password" autocomplete="off">
      </div>
    </div>
    <div class="card-footer">
      <input type="submit" value="Entrar" class="submit">
      <a href="#" class="criar_cadastro">Já possui cadastrado?</a>
    </div>
  </form>
</main>
```

3.3 CSS Login/Cadastro

```
/*css paginas login e cadastro*/
#login {
  display: flex;
  align-items: center;
  justify-content: center;
}

.card {
  padding: 40px;
  border-radius: 15px;
  width: 280px;
  background-color: white;
}

.card-header {
  padding-bottom: 50px;
  opacity: 0.8;
}

#h2_login {
  color: black;
  display: flex;
  justify-content: center;
  font-size: 30px;
}

.card-header::after {
  content: "";
  width: 70px;
  height: 1px;
  background-color: black;
  margin-top: -17px;
  margin-left: -5px;
}

.card-content label {
  color: black;
  font-size: 12px;
  opacity: 0.8;
}
```

```
.card-content-area {
  display: flex;
  flex-direction: column;
  padding: 10px 0;
}

.card-content-area input {
  margin-top: 10px;
  padding: 0 5px;
  background-color: transparent;
  border: none;
  border-bottom: 1px solid black;
  outline: none;
  color: black;
}

.card-footer {
  display: flex;
  flex-direction: column;
}

.card-footer .submit {
  width: 100%;
  height: 40px;
  background-color: black;
  border: none;
  color: white;
  margin: 10px 0;
  cursor: pointer;
}

.criar_cadastro {
  text-align: center;
  font-size: 12px;
  opacity: 0.8;
  color: black;
  text-decoration: none;
}
```

3.4 Dificuldades Login/Cadastro

A dificuldade encontrada pelo grupo foi a questão da utilização de um arquivo css, quando a padronização utiferia na criação de divs novas.

Referência:

- Slide: https://www.w3schools.com/howto/howto_js_slideshow.asp
- Navbar: <https://www.youtube.com/watch?v=bHRXRYTppHM>