Visão Geral da Documentação do Projeto Web

Este documento fornece uma visão geral abrangente do projeto web, abordando seus principais recursos, estrutura, componentes, tecnologias utilizadas e detalhes técnicos de partes específicas. O projeto demonstra modularidade, responsividade e interatividade, incluindo funcionalidades como um player de música com visualizador, validação de formulários e menu de navegação responsivo.

Principais Componentes e Funcionalidades:

- **Player de Música com Visualizador:** Implementado com a Web Audio API, oferece reprodução de áudio com visualização gráfica do espectro sonoro.
- **Validação de Formulários:** Garante a integridade dos dados através da validação no lado do cliente, utilizando JavaScript e jQuery.
- **Menu de Navegação Responsivo:** Adapta-se a diferentes tamanhos de tela, proporcionando uma experiência de usuário consistente em dispositivos móveis e desktop.
- **Estilização CSS:** Organizada em arquivos separados para estilos globais, player de música e página principal, promovendo a manutenção e reutilização do código.

Tecnologias Chave:

- **HTML5:** Fornece a estrutura semântica para a página web.
- **CSS3:** Permite a estilização e o design responsivo da interface.
- **JavaScript:** Adiciona interatividade e manipulação dinâmica do DOM.
- **jQuery:** Facilita a manipulação do DOM e a exibição de alertas.
- **Web Audio API:** Possibilita a reprodução e análise de áudio para o visualizador de música.

Arquitetura do Visualizador de Música:

- **Modular:** Dividido em módulos independentes (Player, Framer, Tracker, Scene, Controls) para maior organização e facilidade de manutenção.
- **Player:** Gerencia o carregamento, reprodução e manipulação do áudio, fornecendo dados de frequência para a visualização.
- **Framer:** Desenha os ticks e a borda do visualizador com base nos dados de frequência do áudio.
- **Tracker:** Permite ao usuário navegar pela música através de um arco interativo.
- **Scene:** Configura o canvas e controla o loop de renderização.
- **Controls:** Lida com os controles de reprodução (play, pause, mute, etc.) e a exibição do tempo decorrido.

Estrutura de Pastas e Arquivos:

A organização dos arquivos em diretórios lógicos facilita a localização e a manutenção do código.

```
.
├── css/
│   ├── estilo1.css         # Estilos CSS globais (formulários, elementos básicos)
│   ├── musica.css          # Estilos específicos do player de música
│   └── principal.css       # Estilos da página principal (navegação, categorias)
│
├── img/
│   ├── prin.jpg           # Imagem de background da página principal
│   └── samambaia.jpg      # Imagem de background das categorias
│
├── js/
│   ├── cadastro_js.js     # Lógica de validação de formulários de cadastro e login
│   ├── musica.js          # Código JavaScript do player de música e visualizador
│   └── principal_js.js    # JavaScript para o menu de navegação responsivo
│
├── index.html             # Arquivo HTML principal
│
└── lofi songs for slow days_160k.mp3 # Arquivo de áudio de exemplo
```

Visualização da Estrutura de Pastas:

```
. (raiz do projeto)
├── css/ (Estilos CSS)
│   ├── estilo1.css
│   ├── musica.css
│   └── principal.css
├── img/ (Imagens)
│   ├── prin.jpg
│   └── samambaia.jpg
├── js/ (JavaScript)
│   ├── cadastro_js.js
│   ├── musica.js
│   └── principal_js.js
├── index.html (Página principal)
└── lofi songs for slow days_160k.mp3 (Arquivo de música)
```

Breve Descrição dos Arquivos:

- **`index.html`:** Arquivo principal que estrutura o conteúdo da página web.
- **`css/estilo1.css`:** Estilos CSS globais aplicados a elementos básicos e formulários.
- **`css/musica.css`:** Estilos CSS específicos para o player de música e o visualizador.
- **`css/principal.css`:** Estilos CSS para a página principal, incluindo navegação e categorias.
- **`img/prin.jpg` e `img/samambaia.jpg`:** Imagens utilizadas como backgrounds.
- **`js/cadastro_js.js`:** Funções JavaScript para validação de formulários de cadastro e login.
- **`js/musica.js`:** Código JavaScript responsável pela funcionalidade do player de música e do visualizador.
- **`js/principal_js.js`:** Script JavaScript para o menu de navegação responsivo.
- **`lofi songs for slow days_160k.mp3`:** Arquivo de música utilizado como exemplo para o player.

Esta documentação detalhada visa fornecer uma compreensão completa do projeto, permitindo que desenvolvedores, designers e outros interessados naveguem e contribuam de forma eficiente.

Documentação Técnica: Validação de Cadastro

Este documento descreve a função `validar_login` e `validar_cadastro` implementada no arquivo `cadastro_js.js`. Estas funções são responsáveis por validar os campos de usuário e senha em um formulário de login e cadastro, respectivamente.

Função `validar_login`

A função `validar_login` verifica se os campos de usuário e senha estão preenchidos. Se algum dos campos estiver vazio, exibe uma mensagem de alerta utilizando um modal do Bootstrap.

Código-fonte

```
function validar_login() {

if (usuario.value === "" && senha.value === ""){
        mensagem.innerHTML = "Preencha os campos faltantes.";
        $('#alerta').modal('show');
}

    if (usuario.value === ""){
        mensagem.innerHTML = "Preencha o campo usuario.";
        $('#alerta').modal('show');
}

if (senha.value === ""){
        mensagem.innerHTML = "Preencha o campo senha.";
        $('#alerta').modal('show');
}
```

Detalhes da Função

Parâmetro	Tipo	Descrição
Nenhum		A função não recebe nenhum parâmetro diretamente. Ela acessa os valores dos campos `usuario` e `senha` (presumivelmente elementos HTML) diretamente.

Variável	Descrição
`usuario`	Elemento HTML do tipo input referente ao campo de usuário. A função acessa sua propriedade `value`.
`senha`	Elemento HTML do tipo input referente ao campo de senha. A função acessa sua propriedade `value`.
`mensagem`	Elemento HTML onde a mensagem de erro será exibida. A função modifica sua propriedade `innerHTML`.

Fluxo de Execução

1. Verifica se ambos os campos `usuario` e `senha` estão vazios. Se sim, exibe a mensagem "Preencha os campos faltantes." no modal de alerta.
2. Verifica se o campo `usuario` está vazio. Se sim, exibe a mensagem "Preencha o campo usuario." no modal de alerta.
3. Verifica se o campo `senha` está vazio. Se sim, exibe a mensagem "Preencha o campo senha." no modal de alerta.

Dependências

- **jQuery:** Utilizada para exibir o modal de alerta (`$('#alerta').modal('show')`).
- **Bootstrap:** Necessário para o funcionamento do modal de alerta.

Função `validar_cadastro`

A função `validar_cadastro` (código incompleto no snippet fornecido) provavelmente contém a lógica para validar os campos de um formulário de cadastro. Uma implementação completa verificaria outros campos como email, confirmação de senha, termos de uso, etc.

Código-fonte

```
function validar_cadastro() {
```

Próximos passos

- Implementar a lógica de validação para os campos de cadastro.
- Adicionar validação de formato para o campo de email.
- Adicionar verificação de igualdade entre os campos de senha e confirmação de senha.
- Adicionar verificação de termos de uso.

Documentação Técnica do Projeto

Estilo CSS: `estilo1.css`

Este documento descreve o arquivo CSS `estilo1.css`, responsável pela estilização da interface do usuário.

Visão Geral

O arquivo CSS define estilos globais para a página, incluindo fontes, cores, tamanhos e layouts. Ele também inclui estilos específicos para elementos como `input`, `select`, `div` e links (`a`).

Importações

O arquivo CSS importa a fonte `Merriweather` do Google Fonts:

```
@import url('https://fonts.googleapis.com/css2?family=Merriweather:ital,wght@1,300&family=Noto+Sans+JP:wght@100&display=swap');
```

Estilos Globais

Os seguintes estilos globais são aplicados:

```
* {
    margin: 0px;
    padding: 0px;
}

body{
    font-family: 'Merriweather';
    background-attachment: fixed;
    background-size: cover;
    background-repeat: no-repeat;
}
```

Estes estilos resetam as margens e paddings padrão dos elementos e definem a fonte e o background do corpo da página.

Estilização de Input e Select

Os elementos `input` e `select` são estilizados com as seguintes propriedades:

```
input{
    display: block;
    padding: 3px;
    margin-top: 5px;
    width: 350px;
    height: 35px;
    text-align: center;
    font-size: 20px;
    border: none;
    background-color: rgba(222,184,135,0.5);
    color: white;
    border-radius: 10px;
}

select{
    display: block;
    padding: 3px;
    margin-top: 5px;
    width: 350px;
    height: 35px;
    text-align: center;
    font-size: 20px;
    border: none;
    background-color: rgba(222,184,135,0.5);
    color: white;
    border-radius: 10px;
}

input::placeholder {
  color: rgba(220,220,220,0.9);
}
```

Estes estilos definem a aparência dos campos de entrada de texto e seleção, incluindo tamanho, cor de fundo, cor do texto e borda. O `input::placeholder` estiliza o texto de dica dentro do campo de entrada.

Estilização de Div

O elemento `div` principal que envolve o conteúdo é estilizado da seguinte forma:

```
div {
    height: 335px;
    width: 500px;
    margin: 200px auto 150px auto;
    background-color: rgba(0, 0, 0,0.3);
    border: 3px solid black;
}
```

Este estilo define a altura, largura, margem, cor de fundo e borda do contêiner principal.

Estilização de Links

Os links (`a`) são estilizados com as seguintes propriedades:

```
.a{
  font-size: 20px;
  color: white;
  text-align: center;
  display: block;
  text-decoration: none;
  transition: 0.3s;
}

.a:hover{
   text-decoration: underline;
   transition: 0.3s;
   color: rgba(60, 179, 113,1);
}
```

Estes estilos definem a aparência dos links, incluindo tamanho da fonte, cor, alinhamento e efeito de hover.

Estilização de Botões

Os botões (elementos com a classe `botao`) são estilizados com as seguintes propriedades:

```
.botao{
    background-color: rgba(222,184,135,0.3);
    cursor: pointer;
    transition: 0.5s;
    padding-left: 10px;
}

.botao:hover{
    background-color: rgba(222,184,135,0.7);
    transition: 0.5s;
}
```

Estes estilos definem a aparência dos botões, incluindo cor de fundo, cursor e efeito de hover.

Classe `esconder`

A classe `esconder` é usada para ocultar elementos:

```
.esconder{
    background-color: rgba(220,220,220,0.01);
    color: rgba(220,220,220,0.1);
    display: none;
    font-size: 1px;
    height: 1px;
    width: 1px;
    margin: 0px;
    padding: 1px 1px;
}
```

Esta classe define `display: none` para ocultar o elemento, além de definir tamanho e cor para garantir que não ocupe espaço visível.

Documentação Técnica: Player de Música Minimalista

Introdução

Este documento descreve a estrutura e o funcionamento do player de música minimalista, detalhando os principais componentes e suas interações.

Estrutura de Arquivos

O projeto é composto por um único arquivo CSS, `musica.css`, que contém toda a estilização do player.

`musica.css`

Este arquivo define toda a aparência do player de música, incluindo o layout, cores, ícones e interações.

Estrutura Geral

O CSS define estilos para o `body` (background), o container principal `.player`, e suas áreas internas.

```
canvas {
    display: block;
}

body {
    background:
url(
Pj4+NjY19fX2JiYl/f39ra2uRkZGZmZlpaWmXl5dvb29xcXGTk5NnZ2c8TV1mAAAAG3RSTlNAQEBAQEBAQEBAQEBAQEBAQEBAQEAvEOwtAAAFVklEQVR4XpWWB67c
2BUFb3g557T/hRo9/WUMZHlgr4Bg8Z4qQgQJlHI4A8SzFVrapvmTF9O7dmYRFZ60YiBhJRCgh1FYhiLAmdvX0CzTOpNE77ME0Zty/nWWzchDtiqrmQDeuv3powQ5ta2eN0FY0
InkqDD73lT9c9lEzwUNqgFHs9VQce3TVClFCQrSTfOiYkVJQBmpbq2L6iZavPnAPcoU0dSw0SUTqz/GtrGuXfbyyBniKykOWQWGqwwMA7QiYAxi+IlPdqo+hYHnUt5ZPfnsHJ
yNiDtnpJyayNBkF6cWoYGAMY92U2hXHF/C1M8uP/ZtYdiuj26UdAdQQSXQErwSOMzt/XWRWAz5GuSBIkwG1H3FabJ2OsUOUhGC6tK4EMtJO0ttC6IBD3kM0ve0tJwMdSfjZo+
EEISaeTr9P3wYrGjXqyC1krcKdhMpxEnt5JetoulscpyzhXN5FRpuPHvbeQaKxFAEB6EN+cYN6xD7RYGpXpNndMmZgM5Dcs3YSNFDHUo2LGfZuukSWyUYirJAdYbF3MfqEKmj
M+I2EfhA94iG3L7uKrR+GdWD73ydlIB+6hgref1QTlmgmbM3/LeX5GI1Ux1RWpgxpLuZ2+I+IjzZ8wqE4nilvQdkUdfhzI5QDWy+kw5Wgg2pGpeEVeCCA7b85BO3F9DzxB3cd
qvBzWcmzbyMiqhzuYqtHRVG2y4x+KOlnyqla8AoWWpuBoYRxzXrfKuILl6SfiWCbjxoZJUaCBj1CjH7GIaDbc9kqBY3W-
Rgjda1iqQcOJu2WW+76pZC9QG7M00dffe9hNnseupFL53r8F7YHSwJWUKP2q+k7RdsxyOB11n0xtOvnW4irMMFNV4H0uqwS5ExsmP9AxbDTc9JwgneAT5vTiUSm1E7BSflSt3
bfa1tv8Di3R8n3Af7MNWzs49hmauE2wP+ttrq+AsWpFG2awvsuOqbipWHgtuvuaAE+A1Z/7gC9hesnr+7wqCwG8c5yAg3AL1fm8T9AZtp/bbJGwl1pNrE7RuOX7PeMRUERVaP
pEs+yqeoSmuOlokqw49pgomjLeh7icHNlG19yjs6XXOMedYm5xH2YxpV2tc0Ro2jJfxC50ApuxGob7lMsxfTbeUv07TyYxpeLucEH1gNd4IKH2LAg5TdVhlCafZvpskfncCfx
8pOhJzd76bJWeYFnFciwcYfubRc12Ip/ppIhA1/mSZ/RxjFDrJC5xifFjJpY2Xl5zXdguFqYyTR1zSp1Y9p+tktDYYSNflcxI0iyO4TPBdlRcpeqjK/piF5bklq77VSEaA+z8
qmJTFzIWiitbnzR794USKBUaT0NTEsVjZqLaFVqJoPN9ODG70IPbfBHKK+/q/AWR0tJzYHRULOa4MP+W/HfGadZUbfw177G7j/OGbIs8TahLyynl4X4RinF793Oz+BU0saXtU
HrVBFT/DnA3ctNPoGbs4hRIjTok8i+algT1lTHi4SxFvONKNrgQFAq2/gFnWMXgwffgYMJpiKYkmW3tTg3ZQ9Jq+f8XN+A5eeUKHWvJWJ2sgJ1Sop+wwhqFVijqWaJhwtD8MN
lSBeWNNWTa5Z5kPZw5+LbVT99wqTdx29lMUH4OIG/D86ruKEauBjvH5xy6um/Sfj7ei6UUVk4AIl3MyD4MSSTOFgSwsH/QJWaQ5as7ZcmgBZkzjjU1UrQ74ci1gWBCSGHtuV1
H2mhSnO3Wp/3fEV5a+4wz//6qy8JxjZsmxxy5+4w9CDNJY09T072iKG0EnOS0arEYgXqYnXcYHwjTtUNAcMelOd4xpkoqiTYICWFq0JSiPfPDQdnt+4/wuqcXY47QILbgAAAA
BJRU5ErkJggg==);
}

.player {
    width: 740px;
    height: 740px;
    margin-left: -370px;
    margin-top: -370px;
    display: block;
    position: absolute;
    left: 50%;
    top: 50%;
}
```

Área de Playback

A área de playback (`.playarea`) contém os controles principais: play/pause, anterior e próxima música.

```
.player .playarea {
    position: absolute;
    top: 50%;
    left: 50%;
    height: 126px;
    width: 320px;
    margin-top: -63px;
    margin-left: -160px;
}

.player .playarea div {
    display: inline-block;
}
```

Botões Play/Pause

Os botões play e pause utilizam imagens SVG codificadas em Base64 para os ícones. O botão `pause` é inicialmente escondido (`display: none;`).

```
.player .playarea .play {
    cursor: pointer;
    opacity: 0.85;
    vertical-align: middle;
    margin: 0 26px;
    border: 3px solid #FE4365;
    border-radius: 120px;
    width: 120px;
    height: 120px;
    background:
url(
0PSI1NXB4IiB2aWV3Qm94PSIwIDAgNDUgNTUiIHZlcnNpb249IjEuMSIgeG1sbnM9Imh0dHA6Ly93d3cudzMub3JnLzIwMDAvc3ZnIiB4bWxuczp4bGluaz0iaHR0cDovL3d3
dy53My5vcmcvMTk5OS94bGluayIgeG1sbnM6c2tldGNoPSJodHRwOi8vd3d3LmJvaGVtaWFuY29kaW5nLmNvbS9za2V0Y2gvbnMiPiAgICAgICAgPHRpdGxlPnBsYXktPC90a
XRsZT4gICAgPGRlc2M+Q3JlYXRlZCB3aXRoIFNrZXRjaC48L2Rlc2M+ICAgIDxkZWZzPiAgICAgICAgPGZpbHRlciB4PSItNTAlIiB5PSItNTAlIiB3aWR0aD0iMjAwJSIgaG
VpZ2h0PSIyMDAlIiBmaWx0ZXJVbml0cz0ib2JqZWN0Qm91bmRpbmdCb3giIGlkPSJmaWx0ZXItMSI+ICAgICAgICAgICAgPGZlT2Zmc2V0IGR4PSIwIiBkeT0iMCIgaW49IlN
vdXJjZUFscGhhIiByZXN1bHQ9InNoYWRvd09mZnNldE91dGVyMSI+PC9mZU9mZnNldD4gICAgICAgICAgICA8ZmVHYXVzc2lhbkJsdXIgc3RkRGV2aWF0aW9uPSIxLjUiIGlu
PSJzaGFkb3dPZmZzZXRPdXRlcjEiIHJlc3VsdD0ic2hhZG93Qmx1ck91dGVyMSI+PC9mZUdhdXNzaWFuQmx1cj4gICAgICAgICAgICA8ZmVDb2xvck1hdHJpeCB2YWx1ZXM9I
jAgMCAwIDAgMC45NDI0NzI1NjIgICAwIDAgMCAwIDAuOTYxNzk0NTYyICAgMCAwIDAgMCAwLjk4MjE5NjAwMyAgMCAwIDAgMC4zNSAwIiBpbj0ic2hhZG93Qmx1ck91dGVyMS
IgdHlwZT0ibWF0cml4IiByZXN1bHQ9InNoYWRvd01hdHJpeE91dGVyMSI+PC9mZUNvbG9yTWF0cml4PiAgICAgICAgICAgIDxmZU1lcmdlPiAgICAgICAgICAgICAgICA8ZmV
NZXJnZU5vZGUgaW49InNoYWRvd01hdHJpeE91dGVyMSI+PC9mZU1lcmdlTm9kZT4gICAgICAgICAgICAgICAgPGZlTWVyZ2VOb2RlIGluPSJTb3VyY2VHcmFwaGljIj48L2Zl
TWVyZ2VOb2RlPiAgICAgICAgICAgIDwvZmVNZXJnZT4gICAgICAgIDwvZmlsdGVyPiAgICA8L2RlZnM+ICAgIDxnIGlkPSJQYWdlLTEiIHN0cm9rZT0ibm9uZSIgc3Ryb2tlL
XdpZHRoPSIxIiBmaWxsPSJub25lIiBmaWxsLXJ1bGU9ImV2ZW5vZGQiIHNrZXRjaDp0eXBlPSJNU1BhZ2UiPiAgICAgICAgPGcgaWQ9IlBsYXllci12ZXJzaW9uLTEiIHNrZX
RjaDp0eXBlPSJNU0FydGJvYXJkR3JvdXAiIHRyYW5zZm9ybT0idHJhbnNsYXRlKC00ODIuMDAwMDAwLCAtMjk2LjAwMDAwMCkiIGZpbHRlcj0idXJsKCNmaWx0ZXItMSkiIGZ
pbGw9IiNGRTQzNjUiPiAgICAgICAgICAgIDxwYXRoIGQ9Ik00ODUsMzE0LjM1MDE1NCBMNDg1LDM0OCBMNTI1LDMyNC4wMzI2MDkgTDQ4NSwyOTkgTDQ4NSwzMTQuMzUwMTU0
IFoiIGlkPSJwbGF5LSIgc2tldGNoOnR5cGU9Ik1TU2hhcGVHcm91cCI+PC9wYXRoPiAgICAgICAgPC9nPiAgICA8L2c+PC9zdmc+) 42px 34px no-repeat;
    background-size: 45px 55px;
}

.player .playarea .pause {
    display: none;
    cursor: pointer;
    opacity: 0.85;
    vertical-align: middle;
    margin: 0 26px;
    border: 3px solid #FE4365;
    border-radius: 120px;
    width: 120px;
    height: 120px;
    background:
url(
0PSI1MXB4IiB2aWV3Qm94PSIwIDAgMzQgNTEiIHZlcnNpb249IjEuMSIgeG1sbnM9Imh0dHA6Ly93d3cudzMub3JnLzIwMDAvc3ZnIiB4bWxuczp4bGluaz0iaHR0cDovL3d3
dy53My5vcmcvMTk5OS94bGluayIgeG1sbnM6c2tldGNoPSJodHRwOi8vd3d3LmJvaGVtaWFuY29kaW5nLmNvbS9za2V0Y2gvbnMiPiAgICAgICAgPHRpdGxlPkxpbmU8L3Rpd
GxlPiAgICA8ZGVzYz5DcmVhdGVkIHdpdGggU2tldGNoLjwvZGVzYz4gICAgPGRlZnM+PC9kZWZzPiAgICA8ZyBpZD0iUGFnZS0xIiBzdHJva2U9Im5vbmUiIHN0cm9rZS13aW
R0aD0iMSIgZmlsbD0ibm9uZSIgZmlsbC1ydWxlPSJldmVub2RkIiBza2V0Y2g6dHlwZT0iTVNQYWdlIj4gICAgICAgIDxnIGlkPSJQbGF5ZXItdmVyc2lvbi0xIiBza2V0Y2g
6dHlwZT0iTVNBcnRib2FyZEdyb3VwIiB0cmFuc2Zvcm09InRyYW5zbGF0ZSgtNDgyLjAwMDAwMCwgLTI5OC4wMDAwMDApIiBzdHJva2U9IiNGRTQzNjUiIHN0cm9rZS1vcGFj
aXR5PSIwLjg1IiBmaWxsPSIjRkU0MzY1IiBzdHJva2UtbGluZWNhcD0ic3F1YXJlIiBmaWxsLW9wYWNpdHk9IjAuODUiPiAgICAgICAgICAgIDxwYXRoIGQ9Ik00ODMsMzAxL
jI2ODUxOSBMNDgzLDM0NS43MzE0ODEgTDQ4MywzNDggTDQ4OC4wNzI0NjQsMzQ4IEw0ODguMDcyNDY0LDM0NS43MzE0ODEgTDQ4OC4wNzI0NjQsMzAxLjI2ODUxOSBMNDg4Lj
A3MjQ2NCwyOTkgTDQ4MywyOTkgTDQ4MywzMDEuMjY4NTE5IFogTTUwOS45Mjc1MzYsMzAxLjI2ODUxOSBMNTA5LjkyNzUzNiwzNDUuNzMxNDgxIEw1MDkuOTI3NTM2LDM0OCB
MNTE1LDM0OCBMNTE1LDM0NS43MzE0ODEgTDUxNSwzMDEuMjY4NTE5IEw1MTUsMjk5IEw1MDkuOTI3NTM2LDI5OSBMNTA5LjkyNzUzNiwzMDEuMjY4NTE5IFoiIGlkPSJMaW5l
IiBza2V0Y2g6dHlwZT0iTVNTaGFwZUdyb3VwIj48L3BhdGg+ICAgICAgICA8L2c+ICAgIDwvZz48L3N2Zz4=) 43px 35px no-repeat;
    background-size: 35px 51px;
}
```

Botões Próxima/Anterior

```
.player .playarea .prevSong,
.player .playarea .nextSong {
    vertical-align: middle;
    background-size: 66px 43px;
    width: 66px;
    height: 43px;
}
```

Efeitos de Hover

```
.player .playarea .prevSong:hover,
.player .playarea .nextSong:hover,
.player .playarea .pause:hover,
.player .playarea .play:hover,
.player .soundControl:hover {
    opacity: 0.7;
}
```

Informações da Música

A seção `.song` exibe o artista e o nome da música.

```
.player .song {
    font-family: Roboto, sans-serif;
    color: #FE4365;
    position: absolute;
    top: 225px;
    left: 0;
    width: 100%;
    text-align: center;
}

.player .song .artist {
    font-size: 22px;
    margin-bottom: 5px;
}

.player .song .name {
    font-size: 18px;
}
```

Controle de Som

O controle de som (`.soundControl`) permite ativar/desativar o som.

```
.player .soundControl {
    cursor: pointer;
    width: 31px;
    height: 27px;
    position: absolute;
    bottom: 240px;
    left: 50%;
    margin-left: -16px;
    background:
url(
0PSIyN3B4IiB2aWV3Qm94PSIwIDAgMzEgMjciIHZlcnNpb249IjEuMSIgeG1sbnM9Imh0dHA6Ly93d3cudzMub3JnLzIwMDAvc3ZnIiB4bWxuczp4bGluaz0iaHR0cDovL3d3
dy53My5vcmcvMTk5OS94bGluayIgeG1sbnM6c2tldGNoPSJodHRwOi8vd3d3LmJvaGVtaWFuY29kaW5nLmNvbS9za2V0Y2gvbnMiPiAgICAgICAgPHRpdGxlPmhpZ2gtdm9sd
W1lPC90aXRsZT4gICAgPGRlc2M+Q3JlYXRlZCB3aXRoIFNrZXRjaC48L2Rlc2M+ICAgIDxkZWZzPiAgICAgICAgPGZpbHRlciB4PSItNTAlIiB5PSItNTAlIiB3aWR0aD0iMj
AwJSIgaGVpZ2h0PSIyMDAlIiBmaWx0ZXJVbml0cz0ib2JqZWN0Qm91bmRpbmdCb3giIGlkPSJmaWx0ZXItMSI+ICAgICAgICAgICAgPGZlT2Zmc2V0IGR4PSIwIiBkeT0iMCI
gaW49IlNvdXJjZUFscGhhIiByZXN1bHQ9InNoYWRvd09mZnNldE91dGVyMSI+PC9mZU9mZnNldD4gICAgICAgICAgICA8ZmVHYXVzc2lhbkJsdXIgc3RkRGV2aWF0aW9uPSIx
IiBpbj0ic2hhZG93T2Zmc2V0T3V0ZXIxIiByZXN1bHQ9InNoYWRvd0JsdXJPdXRlcjEiPjwvZmVHYXVzc2lhbkJsdXI+ICAgICAgICAgICAgPGZlQ29sb3JNYXRyaXggdmFsd
WVzPSIwIDAgMCAwIDAuOTQxMTc2NDcxICAgMCAwIDAgMCAwLjk2MDc4NDMxNCAgIDAgMCAwIDAgMC45ODAzOTIxNTcgIDAgMCAwIDAuMzUgMCIgaW49InNoYWRvd0JsdXJPdX
RlcjEiIHR5cGU9Im1hdHJpeCIgcmVzdWx0PSJzaGFkb3dNYXRyaXhPdXRlcjEiPjwvZmVDb2xvck1hdHJpeD4gICAgICAgICAgICA8ZmVNZXJnZT4gICAgICAgICAgICAgICA
gPGZlTWVyZ2VOb2RlIGluPSJzaGFkb3dNYXRyaXhPdXRlcjEiPjwvZmVNZXJnZU5vZGU+ICAgICAgICAgICAgICAgIDxmZU1lcmdlTm9kZSBpbj0iU291cmNlR3JhcGhpYyI+
PC9mZU1lcmdlTm9kZT4gICAgICAgICAgICA8L2ZlTWVyZ2U+ICAgICAgICA8L2ZpbHRlcj4gICAgPC9kZWZzPiAgICA8ZyBpZD0iUGFnZS0xIiBzdHJva2U9Im5vbmUiIHN0c
m9rZS13aWR0aD0iMSIgZmlsbD0ibm9uZSIgZmlsbC1ydWxlPSJldmVub2RkIiBza2V0Y2g6dHlwZT0iTVNQYWdlIj4gICAgICAgIDxnIGlkPSJQbGF5ZXItdmVyc2lvbi0xIi
Bza2V0Y2g6dHlwZT0iTVNBcnRib2FyZEdyb3VwIiB0cmFuc2Zvcm09InRyYW5zbGF0ZSgtNDgzLjAwMDAwMCwgLTQyNS4wMDAwMDApIiBmaWxsLW9wYWNpdHk9IjAuOCIgZml
sdGVyPSJ1cmwoI2ZpbHRlci0xKSIgZmlsbD0iI0ZFNDM2NSI+ICAgICAgICAgICAgPHBhdGggZD0iTTUwOC4wMDY2MzQsNDI3IEM1MTAuNTE0ODQ4LDQzMC4xNzU0ODQgNTEy
LDQzNC4xODQ0NjMgNTEyLDQzOC41NDI0NyBDNTEyLDQ0Mi44NjE3MSA1MTAuNTQxMTUzLDQ0Ni44MzgxMDQgNTA4LjA3MzMyNiw0NTAgTDUwNi45NDMyNjksNDQ5LjE1MzQwN
SBDNTA5LjI1OTIwNiw0NDYuMTgwMjQ0IDUxMC42NDk5OTksNDQyLjUxMTY5MSA1MTAuNjQ5OTk5LDQzOC41NDI0NyBDNTEwLjY0OTk5OSw0MzQuNTI5NDcyIDUwOS4yMjgzNT
gsNDMwLjgyMzgwOSA1MDYuODY2MzAyLDQyNy44MzM0MTQgTDUwOC4wMDY2MzQsNDI3IEw1MDguMDA2NjM0LDQyNyBaIE01MDMuNjcwMjEyLDQ0Ni43MDEzNTggQzUwNS40OTc1
zMjgsNDQ0LjQzNjg2IDUwNi41OTk5OTksNDQxLjYwOTMzMSA1MDYuNTk5OTk5LDQzOC41NDI0NjkgQzUwNi41OTk5OTksNDM1LjQxNDc0IDUwNS40NTMxMjQsNDMyLjUzNTkz
NyA1MDMuNTYwNTcsNDMwLjI0OTQxOSBMNTAyLjU4MjY1OCw0MzAuOTY0MTI5IEM1MDQuMjQ4MDEsNDMzLjA4NDYzOCA1MDUuMjUwMDAxLDQzNS43MDU2MyA1MDUuMjUwMDAxL
DQzOC41NDI0NjkgQzUwNS4yNTAwMDEsNDQxLjMxMzU1NCA1MDQuMjkzOTIyLDQ0My44Nzg2ODEgNTAyLjY5NzM4Nyw0NDUuOTcyNTU1IEw1MDMuNjcwMjEyLDQ0Ni43MDEzNT
ggTDUwMy42NzAyMTIsNDQ2LjcwMTM1OCBaIE00OTkuNjk0ODIsNDQzLjcyMzE0OSBDNTAxLjEwNTE0MSw0NDIuMjkzMTQ3IDUwMS45NzMwNzcsNDQwLjM0NDU3NSA1MDEuOTc
zMDc3LDQzOC4xOTcwMDcgQzUwMS45NzMwNzcsNDM2LjE2NjYzMiA1MDEuMTk3MjgyLDQzNC4zMTQxMjYgNDk5LjkyMDgwNSw0MzIuOTA5NTUxIEw0OTguOTQyMDkxLDQzMy42
MjQ4NDcgQzQ5OS45NzUwOTUsNDM0Ljg3MjYxNSA1MDAuNTk0MjMxLDQzNi40NjM4MzUgNTAwLjU5NDIzMSw0MzguMTk3MDA3IEM1MDAuNTk0MjMxLDQ0MC4wNDE3OTQgNDk5L
jg5Mjc4NCw0NDEuNzI1NzU2IDQ5OC43Mzc0MjUsNDQzLjAwNTkwNSBMNDk5LjY5NDgyLDQ0My43MjMxNDkgTDQ5OS42OTQ4Miw0NDMuNzIzMTQ5IFogTTQ4NSw0MzUuMjExMj
I1IEw0ODUsNDQxLjg3MzcxNSBMNDg5LjcyNSw0NDIuNDk3NDk0IEw0ODkuNzI1LDQzNC41MDI1MDYgTDQ4NSw0MzUuMjExMjI1IEw0ODUsNDM1LjIxMTIyNSBaIE00ODkuNzI
1LDQ0Mi41Mzk5NjQgTDQ5Ni40NzUsNDQ3LjIwMzcwOCBMNDk2LjQ3NSw0MjkuODgxMjMyIEw0ODkuNzI1LDQzNC41NDQ5NzYgTDQ4OS43MjUsNDQyLjUzOTk2NCBaIiBpZD0i
aGlnaC12b2x1bWUiIHNrZXRjaDp0eXBlPSJNU1NoYXBlR3JvdXAiPjwvcGF0aD4gICAgICAgIDwvZz4gICAgPC9nPjwvc3ZnPg==);
    background-size: 31px 27px;
    text-align: center;
}

.player .soundControl.disable {
    opacity: 0.4;
}
```

Tempo da Música

A seção `.time` exibe o tempo decorrido da música.

```
.player .time {
    text-align: center;
    font-family: Roboto, sans-serif;
    color: #FE4365;
    position: absolute;
    left: 50%;
    margin-left: -22px;
    font-size: 20px;
    bottom: 190px;
}
```

Tabelas de Métodos (Exemplo - JavaScript)

Embora o código fornecido seja apenas CSS, a funcionalidade do player (play, pause, etc.) normalmente seria implementada em JavaScript. Para fins de ilustração, vamos supor que existam os seguintes métodos JavaScript para controlar o player:

Método	Descrição	Parâmetros	Retorno
`playPause()`	Alterna entre os estados de play e pause.	Nenhum	`void`
`nextSong()`	Avança para a próxima música na playlist.	Nenhum	`void`
`prevSong()`	Retrocede para a música anterior na playlist.	Nenhum	`void`
`toggleSound()`	Alterna entre os estados de som ligado e desligado (mute/unmute).	Nenhum	`void`
`updateTime(currentTime)`	Atualiza o tempo exibido na interface.	`currentTime` (number) - Tempo atual em segundos	`void`

Considerações Finais

Esta documentação fornece uma visão geral da estrutura e estilização do player de música minimalista. A implementação da lógica de controle (play, pause, etc.) seria realizada em JavaScript, interagindo com os elementos HTML estilizados por este CSS.

Documentação Técnica: Visualizador de Música

Este documento descreve a arquitetura e o funcionamento interno do visualizador de música. Ele abrange os principais componentes, seus métodos e como eles interagem para criar a visualização.

Arquitetura Geral

O visualizador de música é construído usando JavaScript e a API Canvas. Ele consiste em vários módulos principais:

- **Player:** Responsável por carregar, decodificar e reproduzir a música. Também gerencia o contexto de áudio e o analisador.
- **Framer:** Responsável por desenhar os ticks (barras) que representam o espectro de frequência da música.
- **Tracker:** Responsável por desenhar e controlar o seletor circular que permite navegar pela música.
- **Scene:** Responsável por configurar o canvas, gerenciar o tamanho da tela e coordenar o desenho dos outros módulos.
- **Controls:** Responsável por gerenciar os controles de reprodução (play, pause, next, prev, mute).

Módulo `Player`

O módulo `Player` é o coração do visualizador de música. Ele utiliza a API Web Audio para processar o áudio e fornecer dados de frequência para o módulo `Framer`.

Código-fonte relevante

```
var Player = {

    buffer: null,

    duration: 0,

    tracks: [
        {
            artist: "Lo-fi",
            song: "NOGYMX - bikes at the pier",
            url: "lofi songs for slow days_160k.mp3"
        }
    ],

    init: function () {
        window.AudioContext = window.AudioContext || window.webkitAudioContext;
        this.context = new AudioContext();
        this.context.suspend && this.context.suspend();
        this.firstLaunch = true;
        try {
            this.javascriptNode = this.context.createScriptProcessor(2048, 1, 1);
            this.javascriptNode.connect(this.context.destination);
            this.analyser = this.context.createAnalyser();
            this.analyser.connect(this.javascriptNode);
            this.analyser.smoothingTimeConstant = 0.6;
            this.analyser.fftSize = 2048;
            this.source = this.context.createBufferSource();
            this.destination = this.context.destination;
            this.loadTrack(0);

            this.gainNode = this.context.createGain();
            this.source.connect(this.gainNode);
            this.gainNode.connect(this.analyser);
            this.gainNode.connect(this.destination);

            this.initHandlers();
        } catch (e) {
            Framer.setLoadingPercent(1);
        }
        Framer.setLoadingPercent(1);
        Scene.init();
    },

    loadTrack: function (index) {
        var that = this;
        var request = new XMLHttpRequest();
        var track = this.tracks[index];
        document.querySelector('.song .artist').textContent = track.artist;
        document.querySelector('.song .name').textContent = track.song;
        this.currentSongIndex = index;
```

```
        request.open('GET', track.url, true);
        request.responseType = 'arraybuffer';

        request.onload = function() {
            that.context.decodeAudioData(request.response, function(buffer) {
                that.source.buffer = buffer;
            });
        };

        request.send();
    },

    play: function () {
        this.context.resume && this.context.resume();
        if (this.firstLaunch) {
            this.source.start();
            this.firstLaunch = false;
        }
    },

    pause: function () {
        this.context.suspend();
    },

    mute: function () {
        this.gainNode.gain.value = 0;
    },

    unmute: function () {
        this.gainNode.gain.value = 1;
    },

    initHandlers: function () {
        var that = this;

        this.javascriptNode.onaudioprocess = function() {
            Framer.frequencyData = new Uint8Array(that.analyser.frequencyBinCount);
            that.analyser.getByteFrequencyData(Framer.frequencyData);
        };
    }
};
```

Métodos

Método	Descrição	Parâmetros	Retorno
`init()`	Inicializa o contexto de áudio, o analisador, carrega a primeira faixa e conecta os nós de áudio.	Nenhum	Nenhum
`loadTrack(index)`	Carrega uma faixa de áudio a partir de sua URL, decodifica-a e define o buffer da fonte.	`index`: Índice da faixa a ser carregada.	Nenhum
`play()`	Inicia a reprodução da faixa.	Nenhum	Nenhum
`pause()`	Pausa a reprodução da faixa.	Nenhum	Nenhum
`mute()`	Silencia o áudio.	Nenhum	Nenhum
`unmute()`	Remove o silêncio do áudio.	Nenhum	Nenhum
`initHandlers()`	Inicializa o manipulador de eventos `onaudioprocess` para obter os dados de frequência do analisador e passá-los para o módulo `Framer`.	Nenhum	Nenhum

Módulo `Framer`

O módulo `Framer` é responsável por desenhar os ticks que representam o espectro de frequência da música. Ele recebe os dados de frequência do módulo `Player` e os utiliza para calcular o tamanho e a posição de cada tick.

Código-fonte relevante

```
var Framer = {

    countTicks: 360,

    frequencyData: [],

    tickSize: 10,

    PI: 360,

    index: 0,

    loadingAngle: 0,

    init: function (scene) {
        this.canvas = document.querySelector('canvas');
        this.scene = scene;
        this.context = scene.context;
        this.configure();
    },

    configure: function () {
        this.maxTickSize = this.tickSize * 9 * this.scene.scaleCoef;
```

```
        this.countTicks = 360 * Scene.scaleCoef;
    },

    draw: function () {
        this.drawTicks();
        this.drawEdging();
    },

    drawTicks: function () {
        this.context.save();
        this.context.beginPath();
        this.context.lineWidth = 1;
        this.ticks = this.getTicks(this.countTicks, this.tickSize, [0, 90]);
        for (var i = 0, len = this.ticks.length; i < len; ++i) {
            var tick = this.ticks[i];
            this.drawTick(tick.x1, tick.y1, tick.x2, tick.y2);
        }
        this.context.restore();
    },

    drawTick: function (x1, y1, x2, y2) {
        var dx1 = parseInt(this.scene.cx + x1);
        var dy1 = parseInt(this.scene.cy + y1);

        var dx2 = parseInt(this.scene.cx + x2);
        var dy2 = parseInt(this.scene.cy + y2);

        var gradient = this.context.createLinearGradient(dx1, dy1, dx2, dy2);
        gradient.addColorStop(0, '#FE4365');
        gradient.addColorStop(0.6, '#FE4365');
        gradient.addColorStop(1, '#FF1493');
        this.context.beginPath();
        this.context.strokeStyle = gradient;
        this.context.lineWidth = 2;
        this.context.moveTo(this.scene.cx + x1, this.scene.cx + y1);
        this.context.lineTo(this.scene.cx + x2, this.scene.cx + y2);
        this.context.stroke();
    },

    getTicks: function (count, size, animationParams) {
        size = 15;
        var ticks = this.getTickPoitns(count);
        var x1, y1, x2, y2, m = [], tick, k;
        var lesser = 160;
        var allScales = [];
        for (var i = 0, len = ticks.length; i < len; ++i) {
            var coef = 1 - i / (len * 2.5);
            var delta = ((this.frequencyData[i] || 0) - lesser * coef) * this.scene.scaleCoef;
            if (delta < 0) {
                delta = 0;
            }
            tick = ticks[i];
            if (animationParams[0] <= tick.angle && tick.angle <=  animationParams[1]) {
                k = this.scene.radius / (this.scene.radius - this.getSize(tick.angle, animationParams[0], animationParams[1]) -
delta);
            } else {
                k = this.scene.radius / (this.scene.radius - (size + delta));
            }
            x1 = tick.x * (this.scene.radius - size);
            y1 = tick.y * (this.scene.radius - size);
            x2 = x1 * k;
            y2 = y1 * k;
            m.push({ x1: x1, y1: y1, x2: x2, y2: y2 });
            if (i < 20) {
                var scale = delta / 45;
                scale = scale < 1 ? 1 : scale;
                allScales.push(scale);
            }
        }
        var sum = allScales.reduce(function(pv, cv) { return pv + cv; }, 0) / allScales.length;
        this.canvas.style.transform = 'scale('+sum+')';
        return m;
    },

    getSize: function (angle, l, r) {
        var m = (r - l) / 2;
        var x = (angle - l);
        var h;

        if (x == m) {
            return this.maxTickSize;
        }
        var d = Math.abs(m - x);
        var v = 70 * Math.sqrt(1 / d);
        if (v > this.maxTickSize) {
            h = this.maxTickSize - d;
```

```
        } else {
            h = Math.max(this.tickSize, v);
        }

        if (this.index > this.count) {
            this.index = 0;
        }

        return h;
    },

    getTickPoitns: function (count) {
        var coords = [], step = this.PI / count;
        for (var deg = 0; deg < this.PI; deg += step) {
            var rad = deg * Math.PI / (this.PI / 2);
            coords.push({ x: Math.cos(rad), y: -Math.sin(rad), angle: deg });
        }
        return coords;
    }
};
```

Métodos

| Método | Descrição | Parâmetros

Documentação Técnica: Estilos CSS Principais

Este documento descreve os estilos CSS principais (`principal.css`) utilizados no projeto, detalhando sua estrutura, classes e funcionalidades.

Estrutura Geral

O arquivo `principal.css` define o estilo visual da aplicação, incluindo:

- Importação de fontes do Google Fonts.
- Estilos globais para elementos HTML.
- Estilos para a barra de navegação principal (superior e inferior).
- Estilos para o menu responsivo.
- Estilos para a seção de categorias.

Importação de Fontes

O arquivo CSS importa a fonte `Merriweather` e `Noto Sans JP` do Google Fonts:

```
@import url('https://fonts.googleapis.com/css2?family=Merriweather:ital,wght@1,300&family=Noto+Sans+JP:wght@100&display=swap');
```

Estilos Globais

Estilos básicos são aplicados para resetar margens e preenchimentos:

```
* {
    margin: 0px;
    padding: 0px;
}
```

Barra de Pesquisa

Estilos para a barra de pesquisa (classe `.barra`):

```
.barra {
    width: 400px;
    height: 30px;
    border:solid 1px;
    border-radius:15px;
    padding: 5px;
}

input#bb::placeholder {
  padding-left: 3px;
  padding-bottom: 1.5px;
}
```

Responsividade da Barra de Pesquisa

A largura da barra de pesquisa é ajustada para diferentes tamanhos de tela usando media queries:

```
@media (max-width: 700px) {
    .barra {
        width: 300px;
        height: 30px;
        border:solid 1px;
        border-radius:15px;
        padding: 5px;
    }
}

@media (max-width: 500px) {
    .barra {
        width: 200px;
        height: 30px;
        border:solid 1px;
        border-radius:15px;
        padding: 5px;
    }
}

@media (max-width: 400px) {
    .barra {
        width: 100px;
        height: 30px;
        border:solid 1px;
        border-radius:15px;
        padding: 5px;
    }
}
```

Navegação Principal

Estilos para a barra de navegação principal (ID `principal`):

```
nav#principal {
    display: flex;
    justify-content: space-around;
    align-items: center;
    font-family: system-ui, -apple-system, Helvetica, Arial, Sans-serif;
    background: #23232e;
    height: 8vh;
}
```

Lista de Navegação

Estilos para a lista de itens de navegação (classe `.nav-list`):

```
.nav-list {
    list-style: none;
    display: flex;
}

.nav-list li {
    margin-top: 20px;
    letter-spacing: 3px;
    margin-left: 32px;
}
```

Menu Mobile

Estilos para o menu mobile (classe `.mobile-menu`):

```
.mobile-menu {
    display: none;
    cursor: pointer;
}

.mobile-menu div {
    width: 32px;
    height: 2px;
    background: #fff;
    margin: 8px;
}

@media (max-width: 999px) {
    .mobile-menu {
        display: block;
    }
}
```

Menu Mobile Ativo

Estilos aplicados quando o menu mobile está ativo (classe `.mobile-menu.active`):

```
.mobile-menu.active .line1 {
  transform: rotate(-45deg) translate(-8px, 8px);
}

.mobile-menu.active .line2 {
  opacity: 0;
}

.mobile-menu.active .line3 {
  transform: rotate(45deg) translate(-5px, -7px);
}
```

Responsividade do Menu

O menu se transforma em um menu hambúrguer em telas menores:

```
@media (max-width: 999px) {
    .nav-list {
        z-index: 1;
        position: absolute;
        top: 8vh;
        right: 0px;
        width: 25vw;
        height: 92vh;
        background: #23232e;
        flex-direction: column;
        align-items: center;
        justify-content: space-around;
        transform: translateX(100%);
        transition: transform 0.3s ease-in;
    }

    .nav-list li{
        margin-top: 0px;
        margin-left: 0px;
        opacity: 0;
    }
}

.nav-list.active {
    transform: translateX(0);
}

@keyframes navLinkFade {
   from {
        opacity: 0;
   }
   to {
        opacity: 1;
        transform: translateX(0);
   }
}
```

Seção de Categorias

Estilos para a seção de categorias (classe `.categorias`):

```css
.categorias{
    background-color: rgba(255, 255, 255, 0.6);
    width: 100%;
    height: 60px;
    font-size: 50px;
    font-family: Impact, fantasy;
    color: black;
    border-radius: 15px;
}

.catimg{
    height: 200px;
    width: 200px;
    padding: 15px;
    transition: 0.3s;
    background-color: rgba(255, 255, 255,0.8);
    border-radius: 100%;
}

.catimg:hover{
     padding: 0px;
     transition: 0.3s;
     height: 250px;
     width: 250px;
     filter: drop-shadow(8px 8px 10px gray);
     background-color: rgba(119, 136, 153,0.9);
}

.nome_cat{
    font-size: 20px;
    text-decoration: none;
    color: white;
    transition: 0.3s;
}

.nome_cat:hover{
    color: rgba(255, 215, 0, 1);
    font-size: 20px;
    text-decoration: none;
    transition: 0.3s;
}

div#fundocat{
    z-index: -1;
    background: url("img/samambaia.jpg") repeat center center;
    background-attachment: fixed;
}
```

Documentação Técnica: Mobile Navbar

Este documento descreve a classe `MobileNavbar` e sua utilização para criar um menu de navegação responsivo.

Visão Geral

A classe `MobileNavbar` facilita a criação de um menu de navegação que se adapta a telas menores, como dispositivos móveis. Ela utiliza JavaScript para adicionar interatividade, como abrir e fechar o menu ao clicar em um ícone específico e animar os links de navegação.

Código-fonte

```
class MobileNavbar {
  constructor(mobileMenu, navList, navLinks) {
    this.mobileMenu = document.querySelector(mobileMenu);
    this.navList = document.querySelector(navList);
    this.navLinks = document.querySelectorAll(navLinks);
    this.activeClass = "active";

    this.handleClick = this.handleClick.bind(this);
  }

  animateLinks() {
    this.navLinks.forEach((link, index) => {
      link.style.animation
        ? (link.style.animation = "")
        : (link.style.animation = `navLinkFade 0.5s ease forwards ${
            index / 7 + 0.3 }s`);
    });
  }

  handleClick() {
    this.navList.classList.toggle(this.activeClass);
    this.mobileMenu.classList.toggle(this.activeClass);
    this.animateLinks();
  }

  addClickEvent() {
    this.mobileMenu.addEventListener("click", this.handleClick);
  }

  init() {
    if (this.mobileMenu) {
      this.addClickEvent();
    }
    return this;
  }
}

const mobileNavbar = new MobileNavbar(
  ".mobile-menu",
  ".nav-list",
  ".nav-list li",
);
mobileNavbar.init();
```

Construtor

O construtor da classe `MobileNavbar` recebe três parâmetros:

- `mobileMenu`: Seletor CSS do elemento que atuará como o botão do menu mobile (ex: ".mobile-menu").
- `navList`: Seletor CSS da lista de navegação (ex: ".nav-list").
- `navLinks`: Seletor CSS dos links de navegação dentro da lista (ex: ".nav-list li").

Métodos

A classe `MobileNavbar` possui os seguintes métodos:

Método	Descrição
`constructor`	Inicializa a classe, definindo os seletores e o nome da classe ativa.
`animateLinks`	Adiciona ou remove a animação dos links de navegação quando o menu é aberto ou fechado.
`handleClick`	Alterna a classe ativa nos elementos do menu mobile e da lista de navegação e chama o método `animateLinks`.
`addClickEvent`	Adiciona um ouvinte de evento de clique ao elemento do menu mobile, que chama o método `handleClick`.
`init`	Inicializa a funcionalidade do menu mobile, adicionando o ouvinte de evento de clique, se o elemento do menu mobile existir.

Utilização

Para utilizar a classe `MobileNavbar`, siga os passos:

1. Inclua o código JavaScript da classe no seu projeto.
2. Crie os elementos HTML necessários para o menu mobile, a lista de navegação e os links de navegação.
3. Instancie a classe `MobileNavbar`, passando os seletores CSS corretos para os elementos HTML.
4. Chame o método `init()` para inicializar a funcionalidade do menu mobile.

Exemplo:

```html
<div class="mobile-menu">
  <div class="line1"></div>
  <div class="line2"></div>
  <div class="line3"></div>
</div>

<nav>
  <ul class="nav-list">
    <li><a href="#">Home</a></li>
    <li><a href="#">Sobre</a></li>
    <li><a href="#">Serviços</a></li>
    <li><a href="#">Contato</a></li>
  </ul>
</nav>

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

CSS (Exemplo Básico):

```css
.mobile-menu {
  display: none; /* Esconde por padrão em telas maiores */
  cursor: pointer;
}

.nav-list {
  display: flex;
  list-style: none;
}

@media (max-width: 768px) {
  .mobile-menu {
    display: block; /* Mostra em telas menores */
  }

  .nav-list {
    position: absolute;
    top: 8vh;
    right: 0;
    width: 50vw;
    height: 92vh;
    background: #23232e;
    flex-direction: column;
    align-items: center;
    justify-content: space-around;
    transform: translateX(100%);
    transition: transform 0.3s ease-in;
  }

  .nav-list.active {
    transform: translateX(0);
  }
}

@keyframes navLinkFade {
  from {
    opacity: 0;
    transform: translateX(50px);
  }
  to {
    opacity: 1;
    transform: translateX(0);
  }
}
```

Observações:

- A animação dos links de navegação depende da existência de uma animação CSS chamada `navLinkFade`.
- O CSS fornecido é um exemplo básico e pode ser adaptado para atender às necessidades específicas do seu projeto.
- Certifique-se de ajustar os seletores CSS no construtor da classe `MobileNavbar` para corresponder aos elementos HTML do seu projeto.