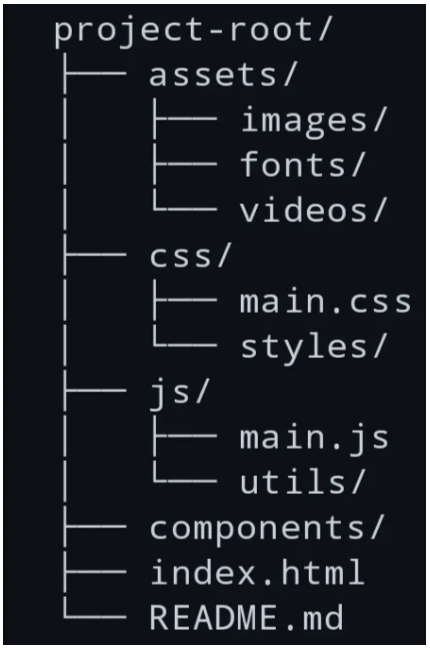
Estrutura de pastas

Status	Review de Leitura 📖
Text	docs: https://github.com/kellen-xavier/qa-readme

Organizar a estrutura de pastas em um projeto de landing page é essencial para manter o código limpo, facilitar a manutenção e permitir que a equipe de desenvolvimento compreenda rapidamente a estrutura do projeto.



---

1. **Estrutura de Pastas**

project-nome/
├── assets/
│ ├── images/
│ ├── fonts/
│ └── videos/
├── css/
│ ├── main.css
│ └── styles/
├── js/
│ ├── main.js

```
│   └── utils/
├── components/
├── index.html
└──
```

README.md

Explicação para Cada Pasta 📂

1. assets/

Função: Armazena todos os arquivos de mídia, como imagens, vídeos e fontes, que são usados na landing page.

Por que usar? Separar os ativos do projeto em uma pasta específica evita poluir as pastas principais com arquivos de mídia e facilita a localização de imagens e outros recursos. Isso é especialmente útil se, no futuro, for preciso trocar uma imagem ou fonte, pois todos os recursos visuais estarão centralizados.

1. css/

Função: Contém os arquivos CSS que definem o estilo da landing page. Dentro desta pasta, o arquivo main.css pode ser o ponto central de estilo, enquanto a subpasta styles/ organiza estilos adicionais.

Por que usar? Manter os estilos separados permite melhor organização, especialmente em projetos que usam CSS modularizado (como SCSS ou pré-processadores). A pasta styles/ pode dividir o CSS em seções específicas, como buttons.css, header.css, etc., facilitando futuras alterações no estilo.

1. js/

Função: Armazena scripts JavaScript que controlam a interatividade e funcionalidade da landing page. O main.js serve como ponto de entrada, e a subpasta utils/ pode conter funções auxiliares.

Por que usar? Uma boa organização de JavaScript permite fácil manutenção e escalabilidade. Ao segmentar scripts e funções reutilizáveis em subpastas, é possível reutilizar ou atualizar funcionalidades sem impactar o projeto inteiro.

1. components/

Função: Contém os componentes reutilizáveis da página, como headers, footers, cards, ou modais, que podem ser armazenados como HTML, CSS e JS modulares.

Por que usar? O uso de componentes facilita a manutenção e a reutilização. Caso haja uma alteração no layout ou estilo de um componente, só é preciso alterá-lo em um lugar. Esse conceito vem do design modular, que é uma prática recomendada para simplificar projetos e otimizar o código.

1. index.html

Função: É o ponto inicial da landing page e carrega todos os estilos e scripts necessários.

Por que usar? Esse arquivo serve como ponto de entrada da página, onde todos os links para CSS, JavaScript e outros recursos são referenciados.

1. README.md

Função: Contém uma descrição do projeto, instruções de instalação, estrutura das pastas e qualquer outra informação útil para quem trabalha no projeto.

Por que usar? Ter um README é uma boa prática, pois ajuda novos membros da equipe ou colaboradores a entenderem rapidamente como o projeto está estruturado, o que facilita a colaboração.

⚠ Por que seguir essa estrutura? ⚠

Facilidade de Navegação e Manutenção: Uma estrutura organizada de pastas torna o projeto intuitivo para quem quer que precise acessá-lo, facilitando a manutenção e futuras expansões.

Escalabilidade: Mesmo que o projeto inicial seja pequeno, uma estrutura de pasta organizada permite que ele cresça sem se tornar confuso ou difícil de manter.

Separação de Responsabilidades: Cada pasta tem um propósito específico, evitando que arquivos com funções diferentes fiquem misturados e dificultem o entendimento do projeto.