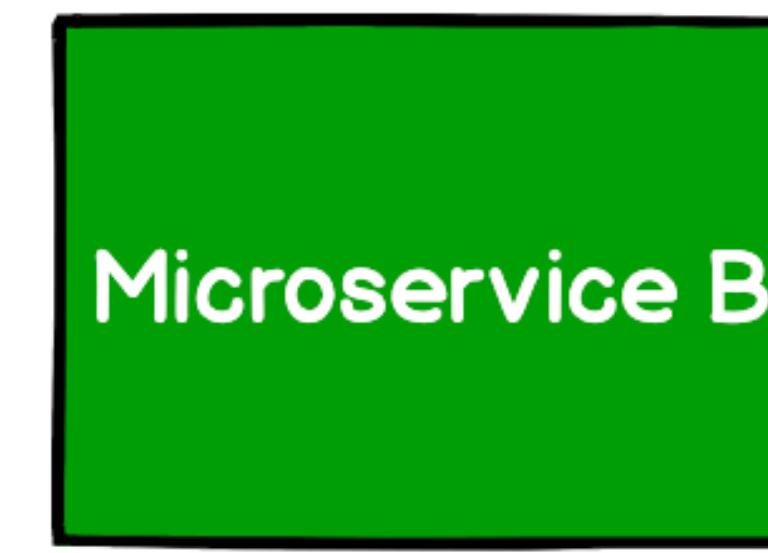


# Arquitetura de Micro front-ends

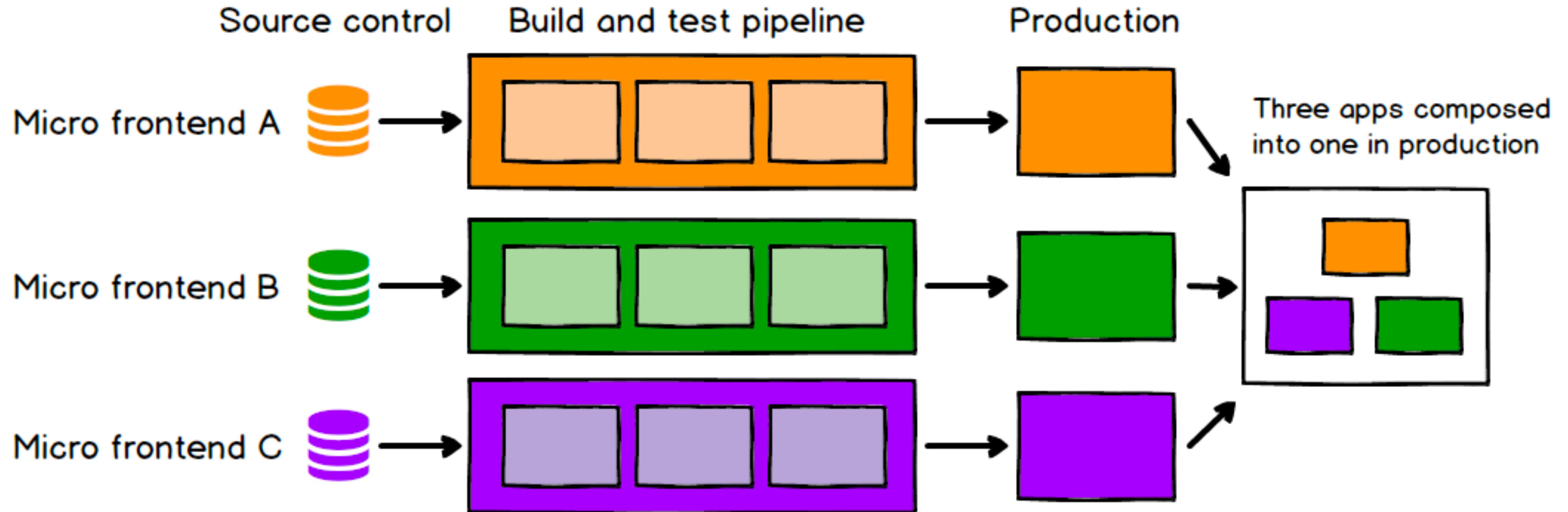
Samuel Martins

# Arquitetura de micro front-ends

- Maior independência entre os módulos;
- Arquitetura mais agnóstica a frameworks;
- Lógica pulverizada em vários projetos;
- Pipeline de build, test e deploys mais rápida;
- Assim como a arquitetura de micro serviços, adiciona uma complexidade a mais no projeto.



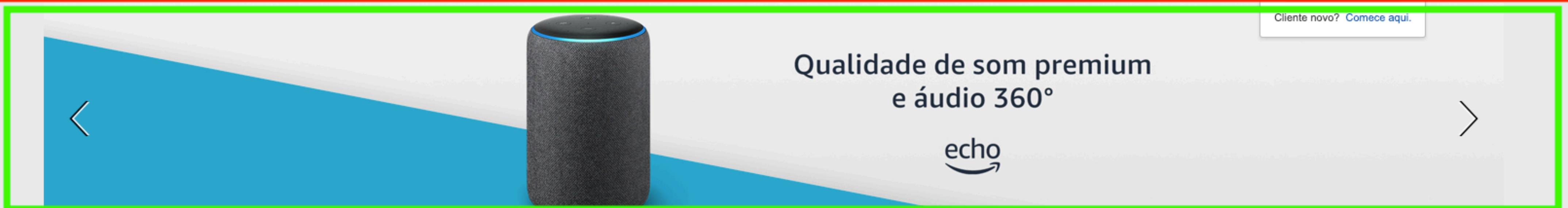
Fonte: <https://martinfowler.com/articles/micro-frontends.html>



Fonte: <https://martinfowler.com/articles/micro-frontends.html>

# Definições

- Baixo ou nenhum acoplamento;
- Alta coesão;
- Não deve assumir responsabilidades de outro micro frontend;
- Não deve interferir ou ser interferido por outro micro frontend;
- Base de código independente;
- Pipeline de build, test e deploys separados e independentes.



### Navegue por loja



Ferramentas

Echo



Games

Jóias

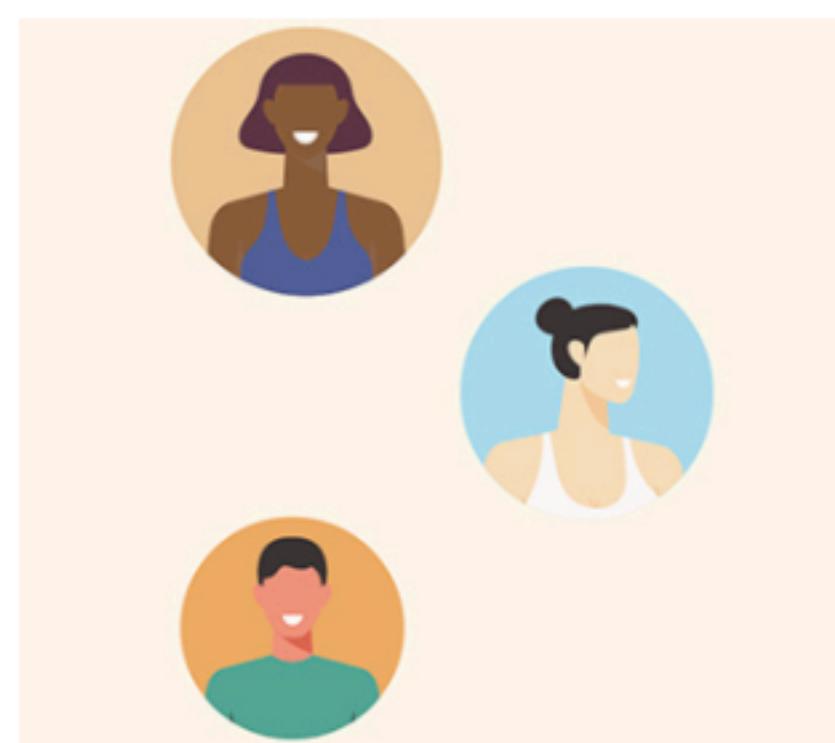
[Veja mais](#)

### Ofertas do Dia



[Veja mais](#)

### Doe e ajude os afetados pela COVID-19



[Veja mais](#)

### Dispositivos Amazon



Echo Dot

Echo Show 5



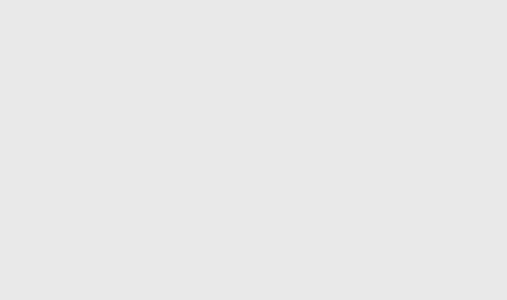
Echo

Echo Show 8

[Confira agora](#)

### Navegue por categoria

[Veja mais](#)



# Benefícios

- Modernização da aplicação sem a necessidade de jogar tudo fora;
- Aplicação agnóstica de novas tecnologias;
- Possibilidade de migração gradativa do código legado;
- Pipeline de build, teste e deploys mais rápida e independente.

# **Arquitetura de Micro front-ends**

## **Formas de implementação**

**Samuel Martins**

# Formas de implementação

- Implementação em tempo de build;
- Integração por meio de funções javascript;
- Integração através de web components;
- Integração via iframes.

# Implementação em tempo de build - projetos como pacotes npm

```
{  
  "name": "@my-project/main",  
  "version": "1.0.0",  
  "description": "My amazing application",  
  "dependencies": {  
    "@my-project/frontend-1": "1.0.0",  
    "@my-project/frontend-2": "1.0.0",  
    "@my-project/frontend-3": "1.0.0",  
  }  
}
```

# Implementação em tempo de build - projetos como pacotes npm

- ✓ Baixo ou nenhum acoplamento;
- ✓ Alta coesão;
- ✓ Não deve assumir responsabilidades de outro micro frontend;
- ✓ Não deve interferir ou ser interferido por outro micro frontend;
- ✓ Base de código independente;
- ✓ Pipeline de build, test e deploys separados independentes;

# Integração por funções javascript

```
<!-- These scripts don't render anything immediately -->
<!-- Instead they attach entry-point functions to `window` -->
<script src="https://browse.example.com/bundle.js"></script>
<script src="https://order.example.com/bundle.js"></script>
<script src="https://profile.example.com/bundle.js"></script>

<div id="micro-frontend-root"></div>

<script type="text/javascript">
  // These global functions are attached to window by the above scripts
  const microFrontendsByRoute = {
    '/': window.renderBrowseRestaurants,
    '/order-food': window.renderOrderFood,
    '/user-profile': window.renderUserProfile,
  };

  const renderFunction = microFrontendsByRoute[window.location.pathname];

  // Having determined the entry-point function, we now call it,
  // giving it the ID of the element where it should render itself
  renderFunction('micro-frontend-root');
</script>
```

Fonte: <https://martinfowler.com/articles/micro-frontends.html>

# Integração por funções javascript

- ✓ Baixo ou nenhum acoplamento;
- ✓ Alta coesão;
- ✓ Não deve assumir responsabilidades de outro micro frontend;
- ✓ Não deve interferir ou ser interferido por outro micro frontend;
- ✓ Base de código independente;
- ✓ Pipeline de build, test e deploys separados independentes;

# Integração através de web components

```
// /about page  
  
<div id="container">  
  <about-micro-frontend></about-micro-frontend>  
</div>  
  
// /products page  
  
<div id="container">  
  <products-micro-frontend></products-micro-frontend>  
</div>
```

# Integração através de web components (1)

```
{  
  "name": "@my-project/web-components",  
  "version": "1.0.0",  
  "description": "My amazing application",  
  "dependencies": {  
    "@my-project/about-micro-frontend": "1.0.0",  
    "@my-project/products-micro-frontend": "1.0.0"  
  }  
}
```

# Integração através de web components (2)

```
<script src="https://about.project.com/bundle.js"></script>  
  
<script src="https://products.project.com/bundle.js"></script>  
  
// /about page  
  
<div id="container">  
  
    <about-micro-frontend></about-micro-frontend>  
  
</div>  
  
// /products page  
  
<div id="container">  
  
    <products-micro-frontend></products-micro-frontend>  
  
</div>
```

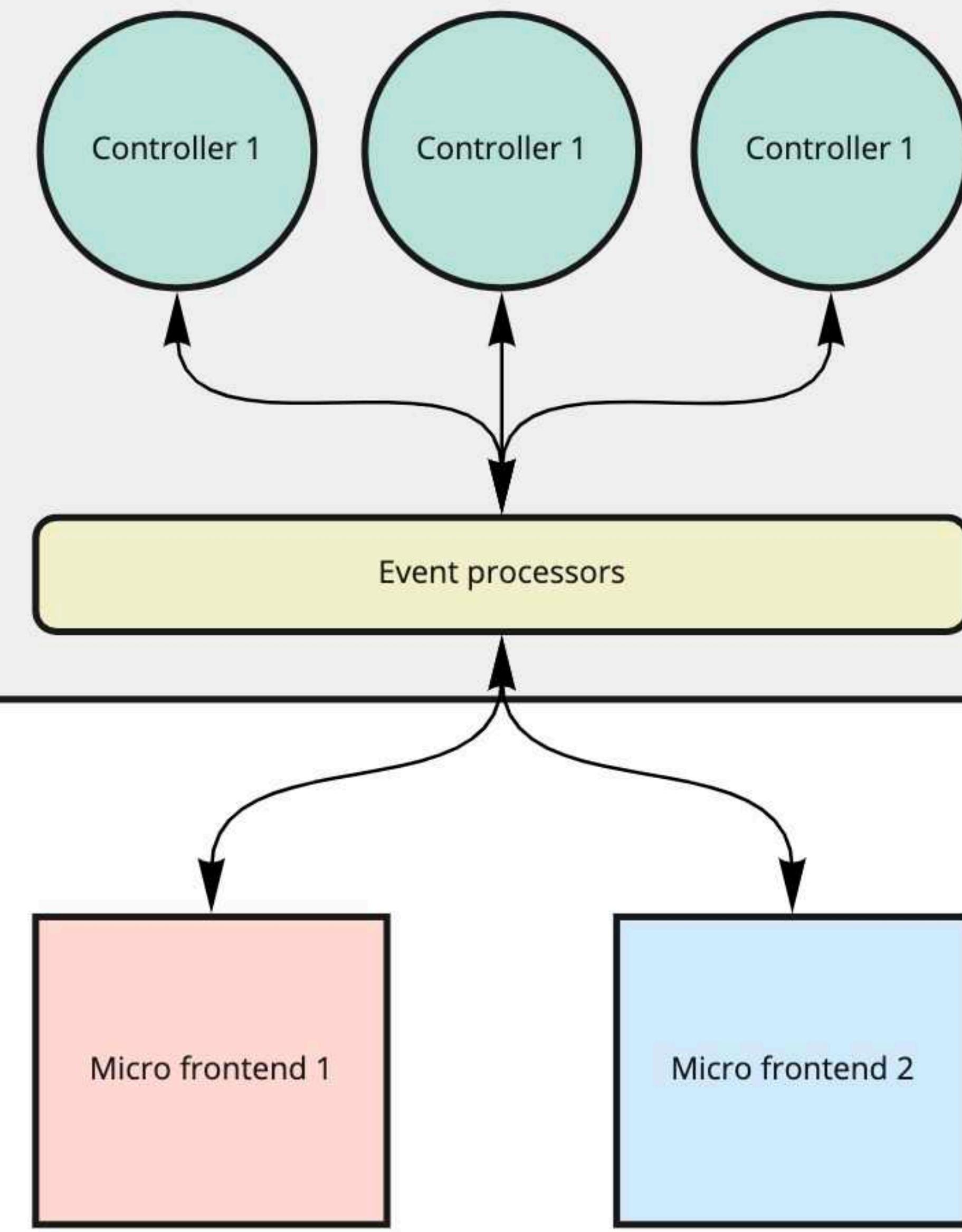
# Integração através de web components

- ✓ Baixo ou nenhum acoplamento;
- ✓ Alta coesão;
- ✓ Não deve assumir responsabilidades de outro micro frontend;
- ✓ Não deve interferir ou ser interferido por outro micro frontend;
- ✓ Base de código independente;
- ✓ Pipeline de build, test e deploys separados independentes.

# Integração via iframes

```
// /about page  
  
<div id="container">  
  
    <iframe src="https://about.project.com"></iframe>  
  
</div>  
  
  
  
  
// /products page  
  
<div id="container">  
  
    <iframe src="https://products.project.com"></iframe>  
  
</div>
```

Aplicação "Satélite"



**Novo  
fire tv stick 4K**  
Streaming em 4K,  
controles de TV e Alexa



prime video



NETFLIX

TELE  
CINE**Navegue por loja**

Ferramentas



Dispositivos Echo



Games



Jóias

[Veja mais](#)**Conheça os dispositivos**  
Amazon

Echo Dot



Kindle 10ª geração



Fire TV Stick Lite



Echo

[Veja todos](#)**Micro frontend**R\$79<sup>90</sup>

Kit 2 Calça Legging Laço Amarrar BOYOU Moda Fitness

[Veja todas ofertas](#)**Assine o Amazon Prime**

Frete GRÁTIS, filmes, séries, músicas, eBooks e jogos por R\$ 9,90/mês.

1º mês de teste GRÁTIS

**Cresça seu negócio vendendo para milhões de clientes na Amazon.**[Registre-se](#)

Patrocinado

# Integração via iframes - proxies

- Window post message proxy: <https://github.com/microsoft/window-post-message-proxy>;
- Iframe Message Proxy: <https://github.com/takenet/iframe-message-proxy>.

# Integração via iframes

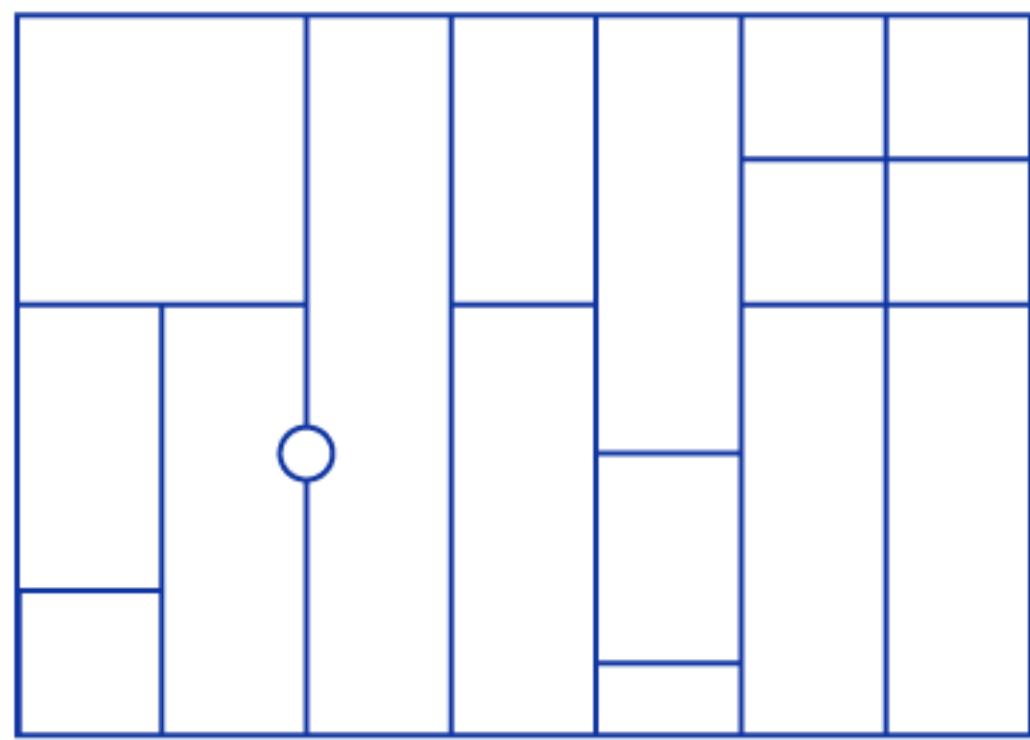
- Baixo ou nenhum acoplamento;
- Alta coesão;
- Não deve assumir responsabilidades de outro micro frontend;
- Não deve interferir ou ser interferido por outro micro frontend;
- Base de código independente;
- Pipeline de build, test e deploys separados independentes;

# Arquitetura de aplicações em Monorepos

Samuel Martins

# Monorepos

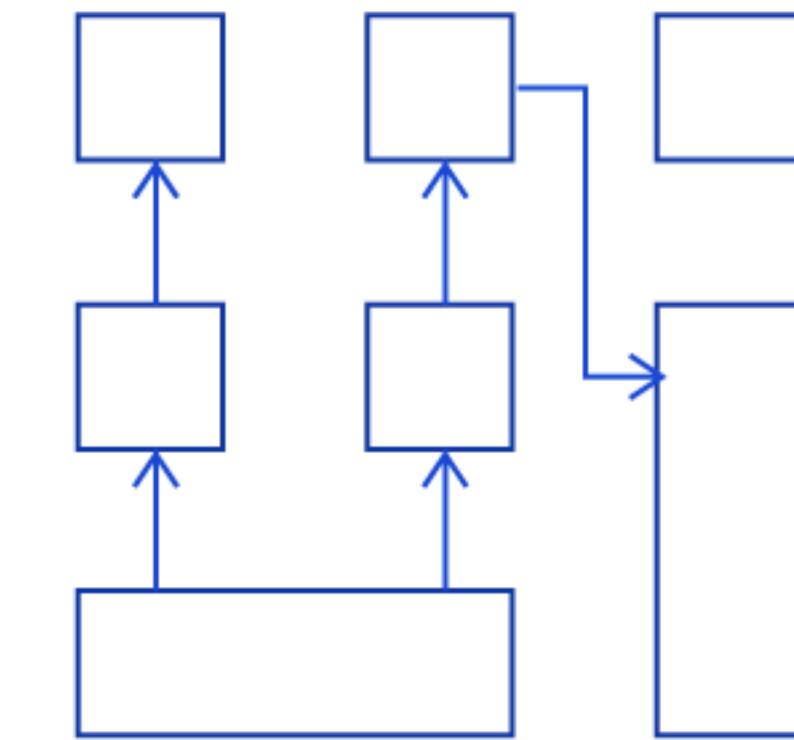
- Conceito arquitetural que propõe um repositório para múltiplos pacotes;
- Cada pacote representa um projeto (seja um micro frontend ou um módulo);
- Isolamento de código com possibilidade de dependência entre pacotes;
- Controle de versão tanto do repositório quanto dos pacotes;
- Refatoração fácil de configurações globais e compartilhadas entre os pacotes.



**Monorepo**



**Single-repo Monolith**



**Multi-repo**



Fonte: <https://www.toptal.com/front-end/guide-to-monorepos>

# Monorepos - ferramentas

- Lerna: <https://lerna.js.org/>;
- Yarn Workspaces: <https://classic.yarnpkg.com/en/docs/workspaces/>.

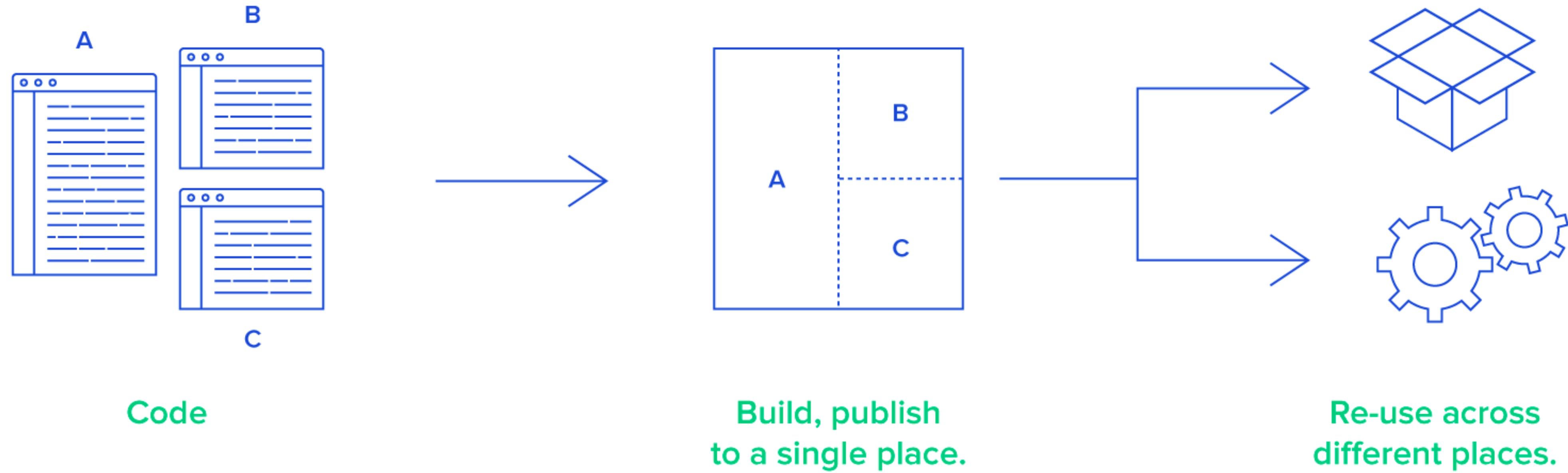


# Lerna

A tool for managing JavaScript projects with multiple packages.

 **Star** 23,435

[Follow Lerna on Twitter](#)



Fonte: <https://www.toptal.com/front-end/guide-to-monorepos>

# Lerna

- **lerna init:** cria um novo repositório com as configurações iniciais;
- **lerna publish:** cria uma release dos pacotes atualizados tanto no git quanto no NPM;
- **lerna run [script]:** roda o script especificado em todos os pacotes que contém esse script configurado (ex.: lerna run start).

# Aplicações server-side rendering

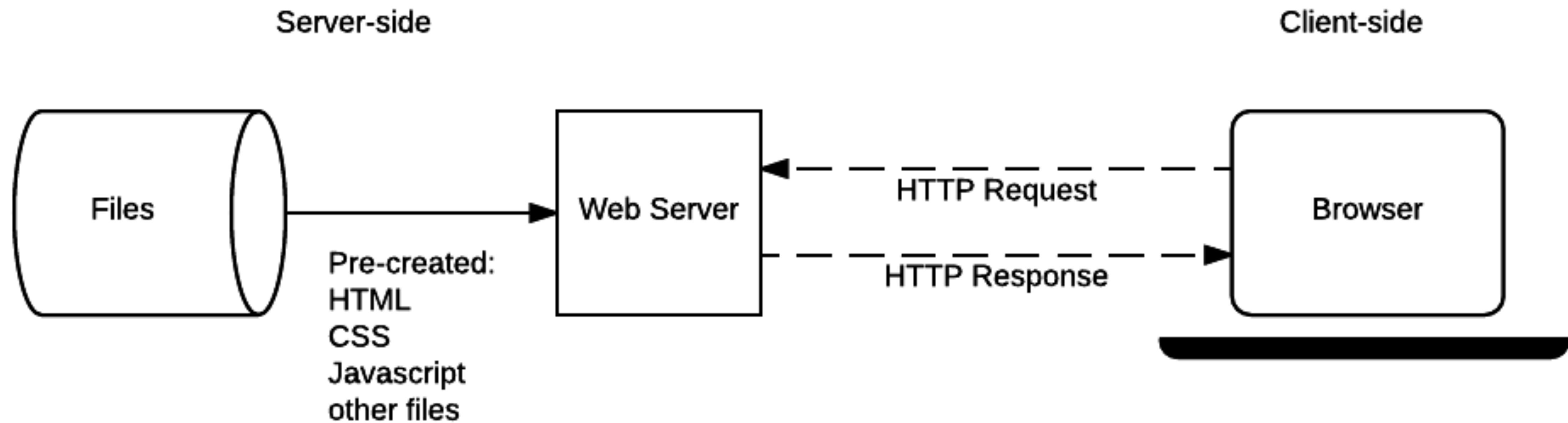
Samuel Martins

# Contexto

- Até antes de 2010, todas as aplicações web dinâmicas eram server-side rendered:
  - PHP;
  - ASP.NET MVC;
  - Java Server Pages (JSP).

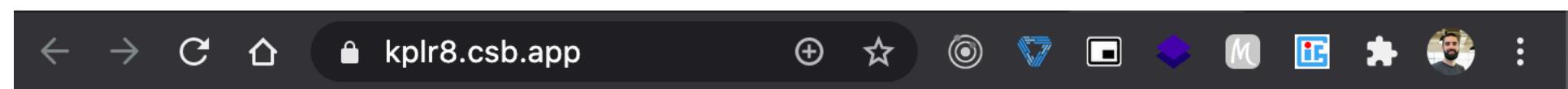
# Aplicações server-side rendered

- Em cada uma das requisições, o servidor é o responsável por devolver a página completa ao usuário, pois ela já foi pré-processada;
- Regras de negócio na mesma base de código da camada de apresentação;
- Roteamento de URLs direto no servidor.



# Aplicações server-side rendered - Vantagens

- Search Engine Optimization (SEO): páginas já possuem o HTML semântico nos resultados de busca

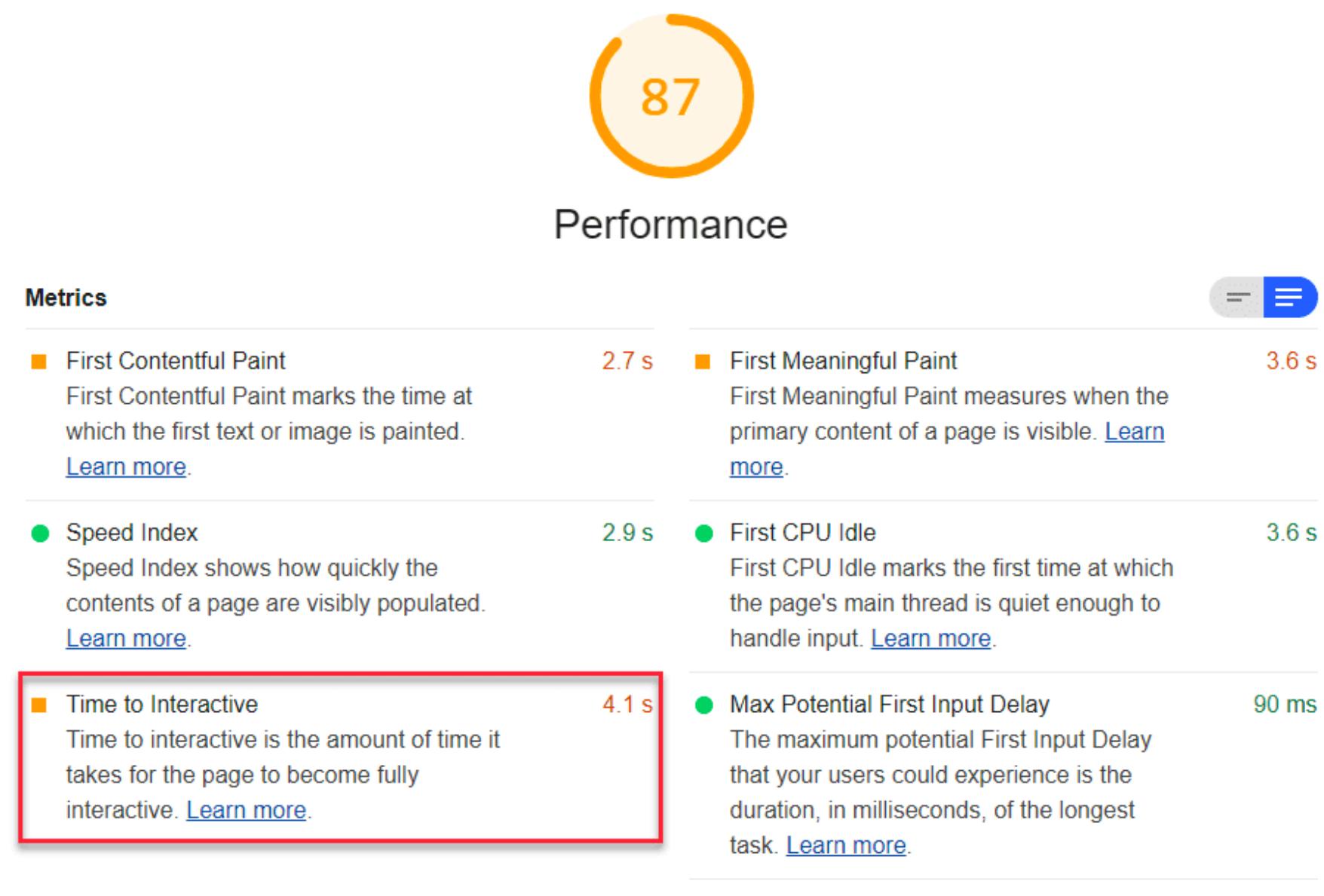


```
24 <body>
25   <app-root></app-root>
26   <script crossorigin type="text/javascript"
src="https://codesandbox.io/static/js/vendors~app-codemirror~editor~monaco-
editor-sandbox.a82ab2b82.chunk.js"></script>
27   <script crossorigin type="text/javascript"
src="https://codesandbox.io/static/js/common-sandbox.26857ffcc.chunk.js"></script>
28   <script crossorigin type="text/javascript"
src="https://codesandbox.io/static/js/vendors~sandbox.20bb6cd4b.chunk.js"></script>
29   <script crossorigin type="text/javascript"
src="https://codesandbox.io/static/js/default~app~embed-sandbox.b75412372.chunk.js">
</script>
30   <script crossorigin type="text/javascript"
src="https://codesandbox.io/static/js/sandbox.fcf79e2d2.js"></script>
31   <script>
```

# Aplicações server-side rendered - Vantagens

- Performance e Time To Interactive (TTI)

<https://developers.google.com/web/tools/lighthouse/audits/time-to-interactive>



# Pontos de atenção

- Algumas arquiteturas de server-side rendering estão em desuso;
- Separação das responsabilidades pode ser mais difícil;
- Dificuldade de componentização e reaproveitamento de códigos de interface;

**FRONT-END + SSR**

~~NEXT~~.JS



NE~~X~~T.JS

# SSR + REACT

- Aplicação é pré-renderizadas no servidor, somente pela primeira vez;
- Navegações subsequentes utilizam o “efeito SPA”;
- Melhor desempenho em mecanismos de busca;
  - Requisições feitas nas URL retornam a página HTML com conteúdo completo pré-renderizado;
- Possibilidade de usar todos os conceitos de componentização presentes no React;
- Sistema de rotas baseado na estrutura de pastas.

# SSR + REACT

```
pages/  
    ├── about.js  
    ├── contact.js  
    └── index.js
```

# SSR + REACT

```
pages/  
  └── post/  
      ├── [postId].js  
      └── [id]  
          └── [commentId].js
```

<https://my-website.com/post/123>

<https://my-website.com/post/123/456>

# Possibilidade de criar funções serverless

```
pages/api/  
    └── posts.js
```

<https://my-domain.com/api/posts>

# Possibilidade de criar funções serverless

- Arquivos são funções *serverless*, estilo *AWS Lambda* ou *Azure Functions*;
- Possibilidade de executar funções de servidor (envio de emails, requisição em APIs com chaves privadas...);

# DEMONSTRAÇÃO

Next.js: <https://codesandbox.io/s/cold-water-9tb3k>

Nuxt.js: <https://codesandbox.io/s/nuxtjs-example-g2wsu>



**Gatsby**

# Gatsby

- JAMStack: Javascript, Api e Markup;
- Utilização de SSR para criação de SSG: **Static Site Generators**;
- Utiliza o **React** para criação das páginas e componentes;
- Principais linhas de utilização:
  - Criação de landing pages;
  - Criação de blogs pessoais.

# Gatsby

- **Comunidade:** bem madura, com boa utilização principalmente em projetos pessoais de baixo custo;
- **Flexibilidade:** personalização total dos recursos disponíveis. Grande showcase de [starter projects](#) para iniciar sua aplicação;
- **Hospedagem simplificada:** no caso de blogs pessoais, as postagens são armazenadas juntamente com o código, evitando a contratação de um espaço em banco;
- **GraphQL:** utilização da linguagem de query para listagem dos posts em arquivos.

# DEMONSTRAÇÃO

CodeSandbox: <https://codesandbox.io/s/gatsby-example-meu18>



PUC Minas  
Virtual