

Configuração do Projeto Bra-Elite no Cloudflare

Pontos-chave:

- É provável que você possa hospedar o projeto **Bra-Elite** inteiramente no Cloudflare, utilizando **Cloudflare Pages** para o front-end, **Pages Functions** para o back-end, **Cloudflare Access** para autenticação e, opcionalmente, **Cloudflare D1** para banco de dados.
- A sincronização com o GitHub é automática via Cloudflare Pages, com deploys acionados por pushes no branch configurado.
- Configurações manuais, como criação de projetos no painel do Cloudflare ou ajustes no DNS, podem ser necessárias, e eu posso ajudar com instruções detalhadas.
- A abordagem é otimizada para simplicidade, centralizando a infraestrutura no Cloudflare, com suporte a PWA e segurança robusta.

Configuração do Front-end

Hospede o projeto React/Vite no **Cloudflare Pages**, conectando o repositório GitHub [lrdspc/Bra-Elite](#). Use o comando `npm run build` e o diretório `dist`. Adicione um arquivo `_redirects` para suportar rotas de Single Page Application (SPA).

Configuração do Back-end

Implemente a lógica de back-end com **Pages Functions**, criando um diretório `functions/` no repositório para APIs serverless, como endpoints para formulários ou dados dinâmicos.

Autenticação

Use **Cloudflare Access** para proteger rotas (ex.: `/admin`) com provedores de identidade como Google ou GitHub, integrando com Pages Functions para validação de tokens.

Sincronização com GitHub

O Cloudflare Pages faz deploy automático a cada push no branch `main`. Para maior controle, configure GitHub Actions com Wrangler, se necessário.

Otimização e Segurança

Habilite cache, proteção contra DDoS e regras de firewall (WAF) no Cloudflare. Teste o PWA em HTTPS para garantir funcionalidade offline.

Relatório Detalhado para Configuração do Projeto Bra-Elite no Cloudflare

Este relatório detalha como configurar o projeto **Bra-Elite**, um Progressive Web App (PWA) desenvolvido com React e Vite, no Cloudflare, utilizando apenas ferramentas nativas da plataforma (Pages, Pages Functions, Access e D1). A configuração inclui front-end, back-end, autenticação, banco de dados (opcional) e sincronização com o repositório GitHub [lrdspc/Bra-Elite](#). Cada etapa é baseada na documentação oficial do Cloudflare e otimizada para simplicidade e desempenho. Quando necessário, indico onde você pode precisar de ajuda manual, com exemplos de código e instruções detalhadas.

1. Configuração do Front-end com Cloudflare Pages

Objetivo: Hoster o front-end do **Bra-Elite** no Cloudflare Pages, garantindo suporte ao PWA e sincronização com o GitHub.

Passos:

1. Conectar o Re却tório GitHub:

- Acesse o painel do Cloudflare em [dash.cloudflare.com](#).
- Vá para **Pages > Create a project > Connect to Git**.
- Autorize o Cloudflare a acessar sua conta GitHub.
- Selecione o re却tório [lrdspc/Bra-Elite](#).
- Configure o projeto:
 - **Nome do projeto:** `bra-elite`.
 - **Branch de produção:** `main` (confirme o branch principal do re却tório).
 - **Framework:** Escolha **Vite** ou **React**.
 - **Comando de build:** `npm run build`.
 - **Diretório de saída:** `dist` (padrão do Vite).
- Clique em **Deploy**. Após o build, você receberá uma URL como `bra-`

elite.pages.dev .

2 Configurar Redirecionamentos para SPA:

- No repositório GitHub, crie um arquivo `public/_redirects` com o seguinte conteúdo:

 _redirects
plain

Mostrar o inline

- Faça commit e push:

bash

...



```
git add public/_redirects
git commit -m "Adiciona _redirects para suporte a SPA"
git push origin main
```

- O Cloudflare Pages aplicará o redirecionamento no próximo build, garantindo que rotas como `/sobre` sejam direcionadas ao `index.html`.

3 Verificar o PWA:

- Acesse `bra-elite.pages.dev` no navegador.
- Abra o Chrome DevTools > **Application** > **Service Workers** e confirme que o service worker (gerado pelo `vite-plugin-pwa`) está ativo.
- Teste offline: desligue a internet e verifique se o PWA carrega corretamente.
- Confirme que o `manifest.json` em `public/` está configurado com ícones, nome e descrição corretos.

4 Sincronização com GitHub:

- Cada push no branch `main` aciona um build automático no Cloudflare Pages.
Verifique os logs em **Pages > Deployments** para depurar erros.

Ajuda Manual:

- Se o build falhar, verifique se `npm run build` funciona localmente e se todas as dependências estão no `package.json`. Peça ajuda para analisar os logs de build.
- Se o service worker não funcionar, revise o `vite.config.ts` ou peça exemplos de configuração do `vite-plugin-pwa`.

Referência: [Cloudflare Pages - Conectar ao Git](#).

2. Configuração do Back-end com Pages Functions

Objetivo: Implementar uma API serverless para o Bra-Elite usando Pages Functions, integrada ao projeto Pages.

Passos:

1. Criar o Diretório `functions/`:

- No repositório GitHub, crie um diretório `functions/` na raiz do projeto.
- Dentro de `functions/`, crie um arquivo `api/hello.js` com o seguinte código:



`api/hello.js`
javascript

Mostrar o inline

- Faça commit e push:

bash

...



```
git add functions/api/hello.js  
git commit -m "Adiciona endpoint de teste no Pages Functions"  
git push origin main
```

- O Cloudflare Pages detectará o diretório `functions/` e deployará a API automaticamente.

2 Testar o Endpoint:

- Acesse `https://bra-elite.pages.dev/api/hello` no navegador ou use:

```
bash
```

... ⌂

```
curl https://bra-elite.pages.dev/api/hello
```

- A resposta deve ser `{"message": "Olá do Bra Elite"}`.

3 Adicionar Endpoints Avançados (Opcional):

- Para funcionalidades como formulários, crie arquivos adicionais em `functions/api/` (ex.: `form.js`):

```
javascript
```

... ⌂

```
export async function onRequestPost(context) {  
  const formData = await context.request.json();  
  // Processar dados do formulário (ex.: salvar em D1)  
  return new Response(JSON.stringify({ success: true }), {
```

```
    headers: { 'Content-Type': 'application/json' }
  });
}
```

- No React, faça chamadas ao endpoint:

```
javascript
```

```
...
```

```
const handleSubmit = async (data) => {
  const response = await fetch('https://bra-elite.pages.dev/api/form', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify(data)
  });
  const result = await response.json();
  console.log(result);
};
```

Ajuda Manual:

- Para lógica complexa, peça exemplos de endpoints ou integração com bibliotecas como Hono.
- Para depurar erros, habilite logs em **Workers > Logs** e peça ajuda para analisar.

Referência: [Cloudflare Pages Functions - Introdução](#).

3. Configuração de Autenticação com Cloudflare Access

Objetivo: Proteger rotas ou APIs do Bra-Elite com autenticação via Cloudflare Access.

Passos:

1. Configurar Cloudflare Access:

- No painel, vá para **Zero Trust > Access > Applications > Add an Application**.
- Escolha **Self-hosted**.
- Configure:
 - **Nome da aplicação:** Bra Elite Admin.
 - **Domínio:** bra-elite.pages.dev/admin/* (ou seu domínio customizado).

- **Provedor de identidade:** Escolha Google, GitHub ou e-mail OTP (disponível no plano gratuito).
- **Políticas:** Defina quem pode acessar (ex.: e-mails específicos como `seuemail@exemplo.com`).
- Salve a aplicação.

² Integrar com Pages Functions:

- Instale o plugin Cloudflare Access:

```
bash
```

```
...
```



```
npm install @cloudflare/pages-plugin-cloudflare-access
```

- Crie um arquivo `functions/_middleware.js`:

```
javascript
```

```
...
```



```
import { cloudflareAccessPlugin } from "@cloudflare/pages-plugin-cloudflare-access"
export const onRequest = cloudflareAccessPlugin({
  domain: "https://bra-elite.cloudflareaccess.com",
  aud: "your-audience-id" // Obtenha no painel Zero Trust
});
```

- Faça commit e push:

```
bash
```

```
...
```



```
git add functions/_middleware.js
git commit -m "Adiciona middleware de autenticação"
git push origin main
```

³ Proteger Rotas no Front-end:

- No React, configure redirecionamentos para rotas protegidas:

javascript

...



```
const checkAuth = async () => {
  const response = await fetch('https://bra-elite.pages.dev/admin', { credentials
    if (response.status === 401) {
      window.location.href = 'https://bra-elite.pages.dev/admin';
    }
  };
};
```

Ajuda Manual:

- Para obter o `aud` do Cloudflare Access, peça ajuda para localizá-lo no painel Zero Trust.
- Se a autenticação falhar, teste em um navegador anônimo e peça ajuda para depurar.

Referência: [Cloudflare Access - Plugin para Pages](#).

4. Configuração de Banco de Dados com Cloudflare D1 (Opcional)

Objetivo: Adicionar um banco de dados SQL serverless para armazenar dados do Bra-Elite.

Passos:

1 Criar um Banco D1:

- No painel, vá para **D1 > Create Database**.
- Nomeie como `bra-elite-db`.
- Crie tabelas com SQL:

sql

...



```
CREATE TABLE users (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    email TEXT NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

2 Vincular o Banco ao Projeto:

- Em **Workers & Pages > bra-elite > Settings > Functions > D1 database bindings**, adicione um binding chamado `DB` vinculado a `bra-elite-db`.

3 Acessar o Banco no Pages Functions:

- Crie um endpoint em `functions/api/users.js`:

```
javascript
```

...



```
export async function onRequest(context) {  
    const { results } = await context.env.DB.prepare("SELECT * FROM users").all();  
    return new Response(JSON.stringify(results), { headers: { 'Content-Type': 'appl  
    }  
}
```

Ajuda Manual:

- Para escrever queries SQL ou integrar com o front-end, peça exemplos específicos.
- Para gerenciar o banco via Wrangler, use `wrangler d1 execute bra-elite-db --file=../schema.sql`.

Referência: [Cloudflare D1 - Introdução](#).

5. Configuração de Domínio Customizado (Opcional)

Objetivo: Configurar um domínio próprio para o **Bra-Elite** com SSL.

Passos:

1 Adicionar Domínio ao Cloudflare:

- Vá para **Add a Site** e insira seu domínio (ex.: `braelite.com`).
- Configure os registros DNS:
 - **CNAME**: `braelite.com` → `bra-elite.pages.dev`.
 - **CNAME**: `www.braelite.com` → `braelite.com`.
- Atualize os nameservers no provedor do domínio (ex.: Registro.br).

2 Habilitar SSL:

- Em **SSL/TLS > Overview**, selecione **Full**.
- Ative **HSTS** em **Edge Certificates**.

Ajuda Manual:

- Se o DNS não propagar em 24 horas, peça ajuda para verificar os registros.

- Para configurar SSL, peça orientações sobre opções específicas.

Referência: [Cloudflare - Configuração de Domínio](#).

6. Otimizações e Segurança

Objetivo: Melhorar desempenho e segurança do Bra-Elite.

Passos:

1. Configurar Cache:

- Em **Caching > Configuration**, habilite cache para arquivos estáticos (JS, CSS, imagens).
- Crie uma **Page Rule** para `bra-elite.pages.dev/*` com "Cache Everything".

2 Habilitar Segurança:

- Em **Security > WAF**, ative regras gerenciadas contra bots e ataques.
- Habilite **DDoS Protection** em **Security > DDoS**.

3 Monitorar Desempenho:

- Use **Analytics** para monitorar tráfego.
- Habilite logs em **Workers > Logs** para depurar.

Ajuda Manual:

- Para ajustes de cache ou regras de segurança, peça orientações detalhadas.

Referência: [Cloudflare - Otimização de Cache](#).

7. Sincronização com GitHub

Objetivo: Garantir que o projeto seja atualizado automaticamente com pushes no GitHub.

Passos:

1. Sincronização Automática:

- O Cloudflare Pages faz deploy automático a cada push no branch `main`.
- Para Pages Functions, o diretório `functions/` é incluído no deploy.

2 GitHub Actions (Opcional):

- Crie um arquivo `.github/workflows/deploy.yml`:

yaml

...



```
name: Deploy to Cloudflare Pages
on:
  push:
    branches:
      - main
jobs:
  deploy:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
```

```

- name: Install Wrangler
  run: npm install -g wrangler
- name: Publish
  run: wrangler pages publish dist
env:
  CLOUDFLARE_API_TOKEN: ${{ secrets.CLOUDFLARE_API_TOKEN }}

```

- Adicione o token da API do Cloudflare em **Settings > Secrets > Actions** no GitHub.

Ajuda Manual:

- Para configurar o token da API, peça ajuda para gerá-lo no painel do Cloudflare.

Referência: [Cloudflare Pages - Integração com Git](#).

Tabela de Configurações

	Ferramenta	Ação Principal	Ajuda Manual Necessária
Componente	Cloudflare		
Front-end	Cloudflare Pages	Hospedar React/Vite, configurar <code>_redirects</code>	Depurar falhas de build, service worker
Back-end	Pages Functions	Criar APIs em <code>functions/</code>	Configurar endpoints complexos
Autenticação	Cloudflare Access	Proteger rotas com provedores de identidade	Configurar políticas, integrar com front-end
Banco de Dados	Cloudflare D1	Criar banco SQL serverless	Escrever queries SQL, vincular ao projeto
Sincronização GitHub	Cloudflare Pages	Deploy automático por push	Configurar GitHub Actions, token da API
Domínio	Cloudflare DNS	Configurar CNAME, habilitar SSL	Verificar propagação de DNS, configurar SSL

Segurança	WAF, DDoS Protection	Ativar regras de segurança, cache	Ajustar regras de cache ou WAF
-----------	----------------------	-----------------------------------	--------------------------------

Possíveis Problemas e Soluções

- **Build Falha no Cloudflare Pages:**

- Verifique se `npm run build` funciona localmente.
- Confirme a versão do Node.js no `package.json` (ex.: `"engines": { "node": "18" }`).
- Consulte os logs em **Pages > Deployments**.

- **Rotas 404 no PWA:**

- Confirme que o arquivo `_redirects` está em `public/` e contém `/* /index.html 200`.

- **Service Worker Não Funciona:**

- Teste em HTTPS (`bra-elite.pages.dev`).
- Verifique o `vite.config.ts` para configurações do `vite-plugin-pwa`.

- **Autenticação Falha:**

- Confirme as políticas do Cloudflare Access em **Zero Trust**.

- Teste em um navegador anônimo.

Ajuda Manual: Para qualquer problema, peça ajuda especificando a etapa e o erro encontrado.

Conclusão

Esta configuração permite hospedar o **Bra-Elite** inteiramente no Cloudflare, com front-end no Pages, back-end no Pages Functions, autenticação via Access e banco de dados no D1 (se necessário). A sincronização com o GitHub é automática, e otimizações de cache e segurança garantem desempenho e proteção. Siga os passos acima e peça ajuda para configurações manuais ou depuração.

Key Citations:

- [Cloudflare Pages - Conectar ao Git](#)
- [Cloudflare Pages - Redirecionamentos](#)
- [Cloudflare Pages Functions - Introdução](#)
- [Cloudflare Access - Plugin para Pages](#)
- [Cloudflare D1 - Introdução](#)
- [Cloudflare - Configuração de Domínio](#)
- [Cloudflare - Otimização de Cache](#)