1. Fluxo de Trabalho (Workflow)

Descrição:

Adotaremos um modelo de **branch por pessoa**. Cada desenvolvedor terá seu próprio branch de trabalho. A nomenclatura do branch será `dev-<nome-do-desenvolvedor>`. Após concluir suas tarefas, o desenvolvedor abrirá um Pull Request (PR) para a branch `main`, onde o código será revisado antes de ser mergeado.

Fluxograma:

O fluxo segue os seguintes passos:

1. Cada desenvolvedor cria seu branch a partir de `main`.
2. O trabalho é realizado no branch pessoal com commits regulares.
3. Após finalizar, o branch é enviado para o repositório remoto.
4. O desenvolvedor abre um PR para a branch `main`.
5. O código é revisado e mergeado na `main`.

Fluxograma:

```
graph TD
    A[main] -->|Criar Branch| B[dev-<nome-do-desenvolvedor>]
    B --> C[Desenvolvimento]
    C -->|Commit/Push| B
    B -->|Abrir Pull Request| D[PR para main]
    D -->|Revisar| E[Merge para main]
    E --> A
```

2. Tutorial para Fluxo de Trabalho no Git

Comandos:

Aqui está a lista completa de comandos para o fluxo de trabalho no modelo **branch por pessoa**:

```
# Clonar o repositório
git clone <url-do-repositorio>

# Criar um novo branch pessoal
git checkout -b dev-<nome-do-desenvolvedor>
```

```
# Adicionar mudanças
git add .

# Criar um commit
git commit -m "Descrição da mudança"

# Enviar para o repositório remoto
git push origin dev-<nome-do-desenvolvedor>

# Após finalizar o trabalho, abrir Pull Request no GitHub (GUI)

# Fazer merge do PR no GitHub

# Voltar para a branch main após o merge
git checkout main

# Atualizar a branch main localmente
git pull origin main
```

3. Ferramentas, Bibliotecas e Configuração

Ferramentas Utilizadas

Descrição: Estas ferramentas são utilizadas para o desenvolvimento, gerenciamento de banco de dados e execução do servidor.

1. **Node.js**
   - Gerenciamento de pacotes e execução do servidor.
   - Link para download.
2. **Prisma**
   - ORM para banco de dados que simplifica a interação com o banco de dados.

Comandos para instalação:
bash
Copiar código
```
npm install prisma --save-dev
npx prisma init
```

   -
3. **SQLite**
   - Banco de dados local integrado ao Prisma para armazenamento eficiente e simples.
4. **Express.js**
   - Framework para construção de APIs robustas.

Comando para instalação:
bash
Copiar código
`npm install express`

-

Configuração

Certifique-se de ter o `Node.js` e o `npm` instalados antes de continuar. Siga as etapas abaixo para configurar o projeto:

Instale as dependências do projeto:

```
npm install
```

1. Configure o Prisma com o banco de dados SQLite:

Inicie o Prisma:

```
npx prisma init
```

- Configure o arquivo `schema.prisma` conforme necessário.

Execute as migrações:

```
npx prisma migrate dev
```

Inicie o servidor para desenvolvimento:

```
npm run dev
```