Manual de Instalação

Front-End

Documentações das tecnologias

- Next.js https://nextjs.org/
- React.js https://pt-br.reactjs.org/
- Apollo Client https://www.apollographql.com/docs/react/
- MUI https://mui.com/

Ambiente de desenvolvimento

Requisitos:

Sistema operacional MacOS, Windows (incluindo WSL) ou Linux;

Git;

Node.js (v16 é recomendável);

Instruções de instalação

1. Clone o repositório do front-end para sua máquina:

```
git clone https://github.com/EduardoGiacomini/mafcust-frontend.git
```

2. Configure as variáveis de ambiente a partir do exemplo `.env.example`:

```
cp .env.example .env
```

3. Instale as dependências:

```
npm install
```

Back-End

Documentações das tecnologias

- Apollo Server: https://www.apollographql.com/docs/apollo-server/
- Prisma: https://www.prisma.io/docs/
- Typescript: https://www.typescriptlang.org/docs/

Ambiente de desenvolvimento

Requisitos:

Sistema operacional MacOS, Windows (incluindo WSL) ou Linux;

Git;

Node.js (v16 é recomendável);

Docker;

Docker Compose;

Instruções de instalação:

1. Clone o repositório do back-end para a sua máquina:

```
git clone https://github.com/EduardoGiacomini/mafcust-backend.git
```

2. Configure as variáveis de ambiente a partir do exemplo `.env.example`:

```
cp .env.example .env
```

3. Inicie o banco de dados usando docker:

```
docker-compose up -d
```

4. Inicie o front-end em ambiente de desenvolvimento:

```
npm run dev
```

Ambiente de Produção:

Infraestrutura Vercel + Github [Recomendado]

1. Certifique-se de empurrar as alterações da aplicação Next.js para o GitHub
2. Para criar uma conta gratuita na Vercel vá para https://vercel.com/signup e escolha "***Continue with GitHub***" e siga as etapas do processo.
3. Importe o seu repositório no link https://vercel.com/import/git.
4. É necessário a instalação do "***Vercel for GitHub***". Conceda permissão para instalação no repositório respectivo.
5. É possível se utilizar as opções padrões para:
    - Project Name
    - Root Directory
    - Build Command
    - Output Directory
    - Development Command
6. O build da aplicação irá iniciar e deve finalizar com uma URL de testes dentro de poucos minutos.

Servidor Pessoal

A aplicação Next.js pode ser instalar em qualquer servidor que dê suporte a Node.js v16 ou posterior.

Para instruções específicas deste modelo de instalação siga os passos em https://nextjs.org/learn/basics/deploying-nextjs-app/other-hosting-options

OBS: Você pode usar um banco de dados Postgres local em vez de utilizar o docker. Não se esqueça de verificar se o serviço esteja executando em sua máquina e de atualizar as credenciais de `DATABASE_URL` no arquivo `.env`.

4. Instale as dependências:

```
npm install
```

5. Atualize as migrations e crie os tipos para as entidades do prisma:

```
npm run prisma
```

6. Inicie o back-end com o nodemon (hot reload habilitado):

```
npm run dev
```

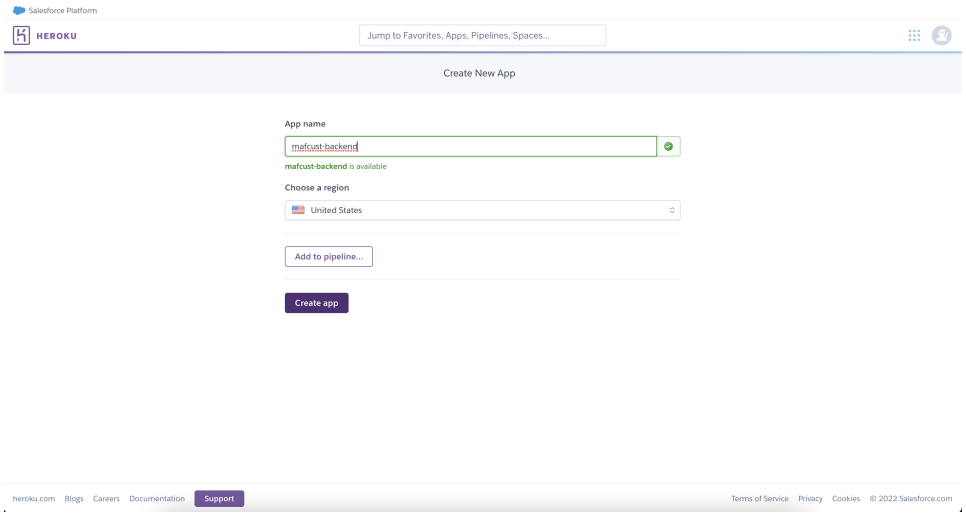
Após seguir todos os passos você será capaz de acessar o backend na URL: http://localhost:4000.

Ambiente de Produção:

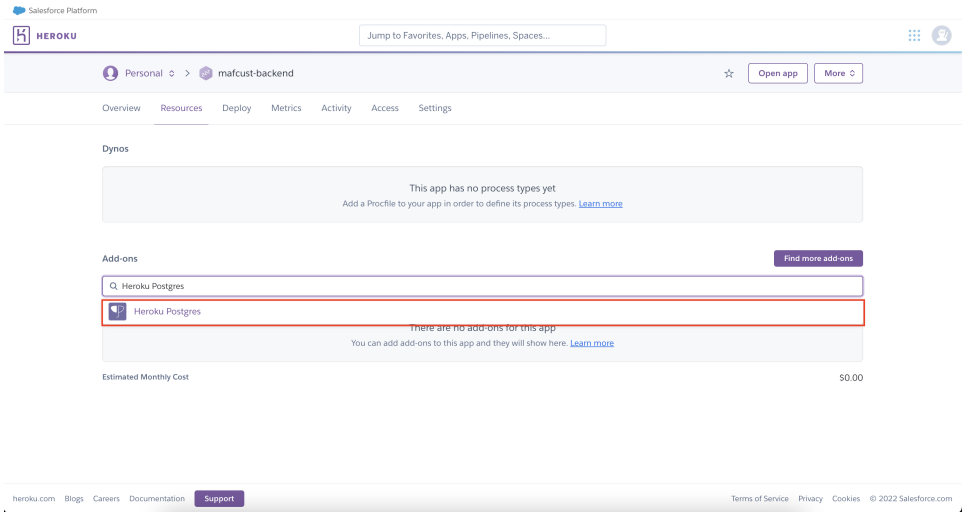
Infraestrutura Heroku [Recomendo]

1. Autentique-se ou crie uma conta no site https://www.heroku.com/.
2. Em sua dashboard, crie uma nova aplicação em `New` → `Create new app`.

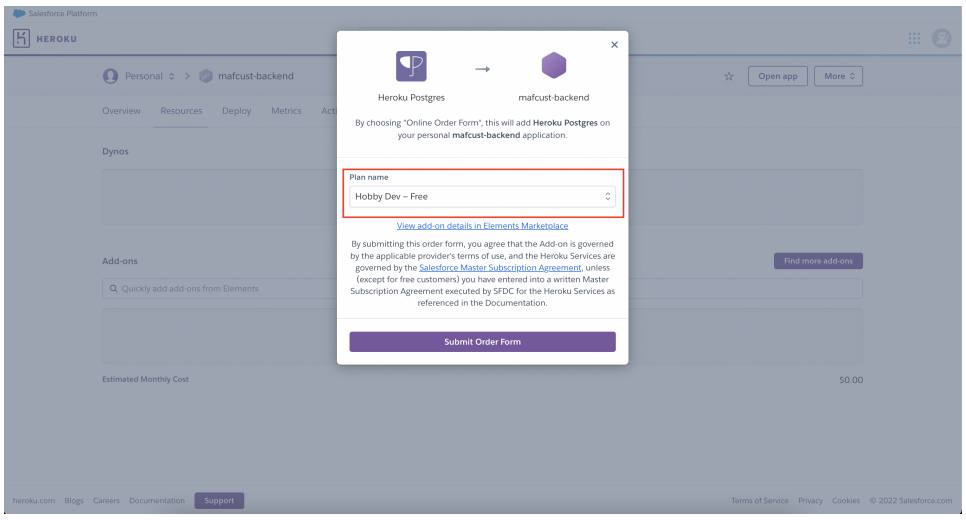
Dê um nome para sua aplicação e selecione a região de publicação (Estados Unidos é recomendável) e selecione `Create app`.



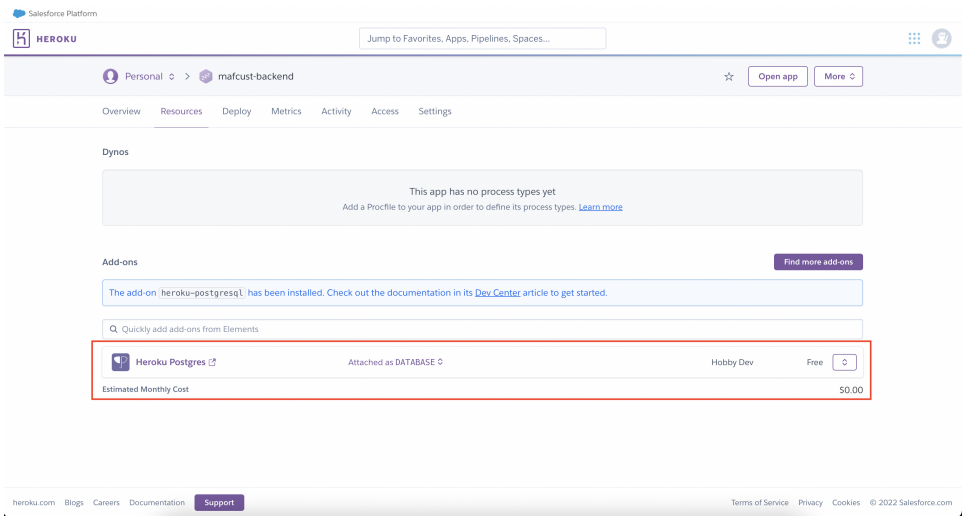
3. Crie o banco de dados como o recurso de sua aplicação. Na aba `Resources` , pesquise por `Heroku Postgres` e selecione a primeira opção.



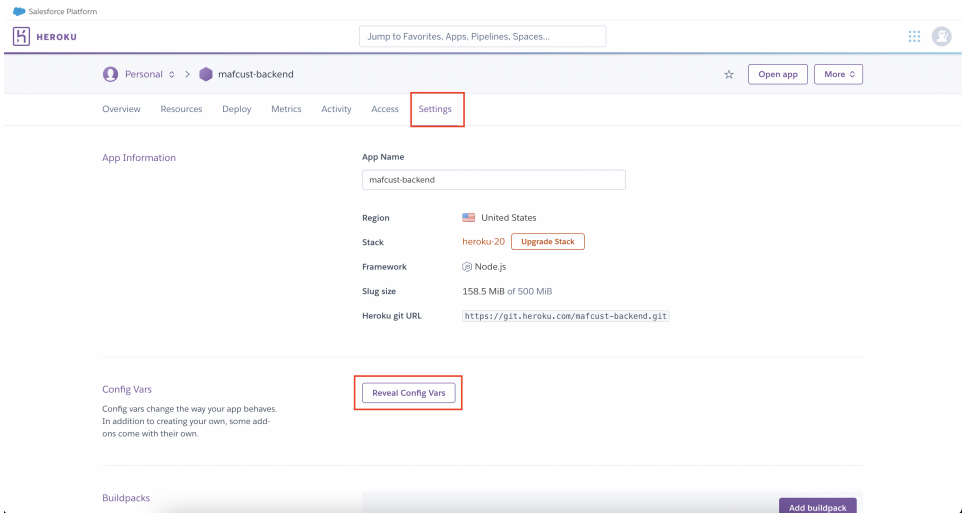
Certifique-se de utilizar o plano `Hobby Dev - Free` e selecione `Submit Order Form` .



Será possível visualizar o recurso recém-criado semelhante à imagem seguinte:



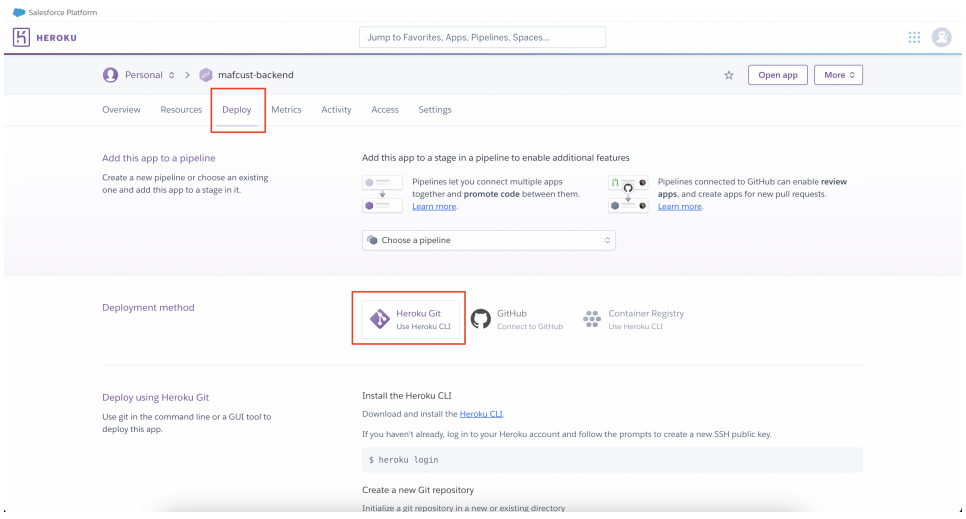
4. Antes de publicar a aplicação, é preciso configurar as variáveis de ambiente. Vá para a aba Settings e selecione a opção `Reveal Config Vars` .



A variável `DATABASE_URL` já estará com valor configurado, adicione as seguintes *(não se esqueça de mudar os valores)*:

```
SECRET=<JWT secret used to generate tokens>
FIRST_USER_PASSWORD=<secret needed to create first user>
```

5. Publique a aplicação utilizando **Heroku CLI**. Vá para a aba `Deploy` e certifique-se de selecionar a opção `Heroku CLI`.



Seguindo os passos do tutorial `Deploy using Heroku Git`. Faça download e instale o Heroku CLI.

Em seu terminal, autentique-se no Heroku CLI:

```
heroku login
```

Acesse o diretório da aplicação:

```
cd your/path/to/mafcust-backend
```

Crie o git remote para o Heroku:

```
heroku git:remote -a mafcust-backend
```

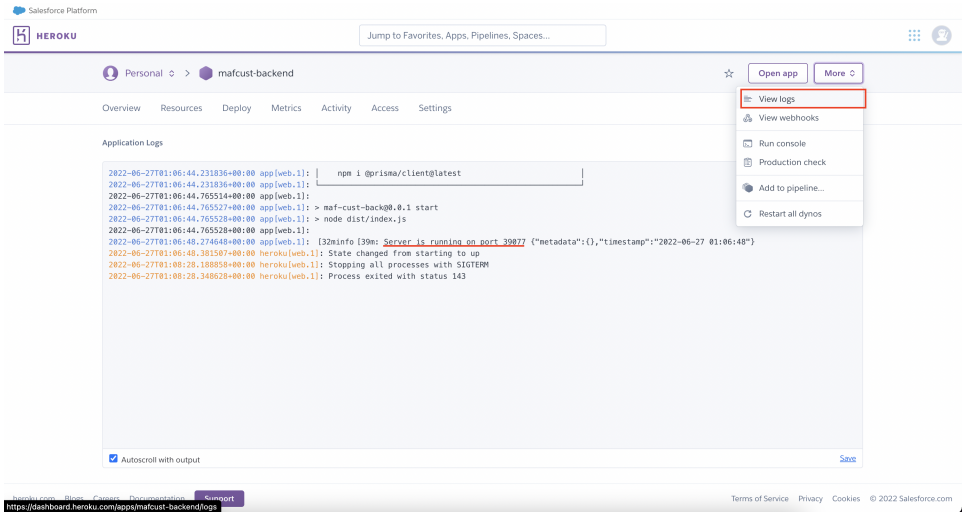
Com esse comando, um novo remote será adicionado ao seu projeto git local, você pode observar com:

```
git remote -v
```

Finalmente, execute o seguinte comando para atualizar o head do **Heroku Git** remoto e publique sua aplicação:

```
git push heroku master
```

Verifique os logs da aplicação para ver se foi publicada corretamente. Acesse as opções em `More` → `View logs` e procure por `Server is running on port XXXXX` .



Parabéns 🎉, sua aplicação foi publicada no Heroku! Acesse a URL selecionando o botão `Open app` no canto superior direito.