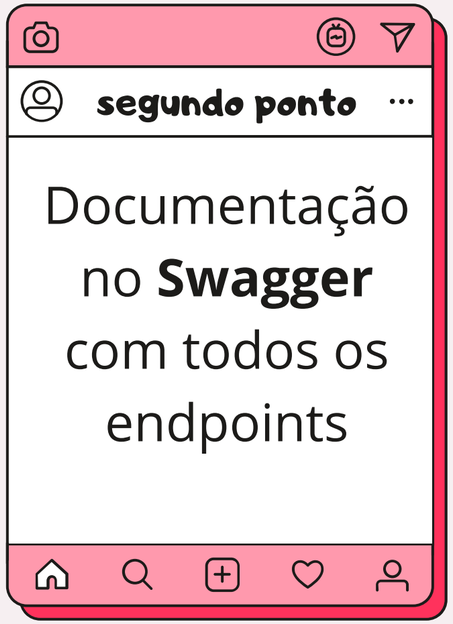
Projeto Individual Livre

Demorou, mas chegou!
Bora lá gatinhas?!

SEMANA 17 & 18 - On16 - Backend - Prof May

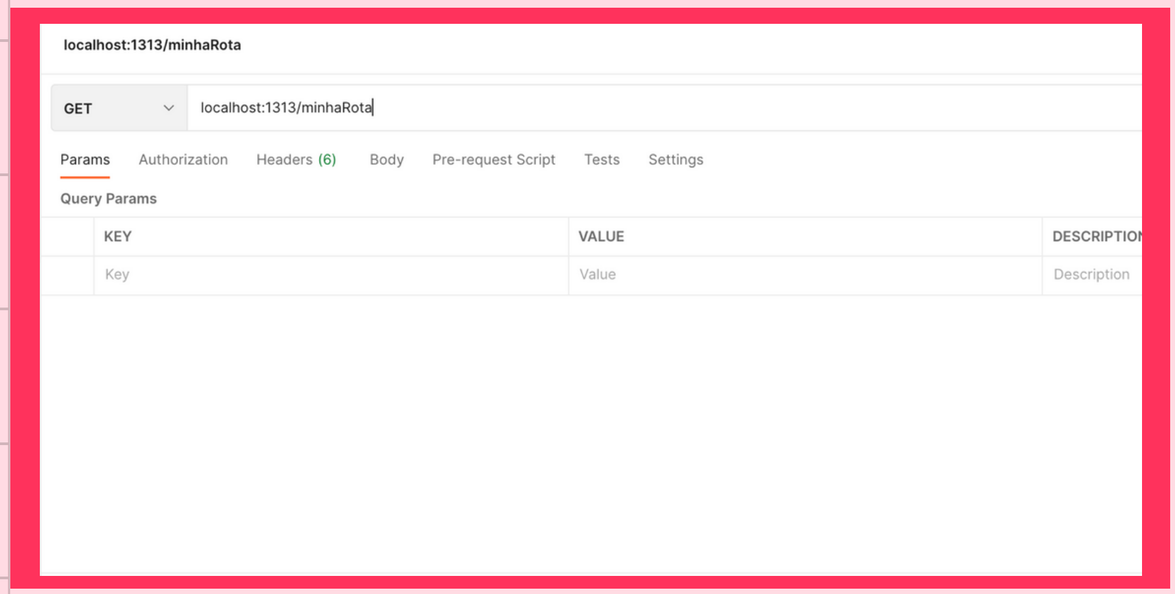
O projeto é livre, mas temos entregas que serão obrigatórias, ta gatinhas?



primeiro ponto

Consulta da API pelo postman

Essa parte já estamos familiarizadas, certo?
Criamos a nossa API e daí realizamos as consultas por rotas através do postman



segundo ponto

Documentação no **Swagger** com **todos os endpoints**

O que é **SWAGGER**?
O Swagger é um framework composto por diversas ferramentas que, independente da linguagem, auxilia a descrição, consumo e visualização de serviços de uma API REST.

SHORT WORDS: O swagger serve pra documentar a nossa API Rest

Agora vamos aprender como colocar ele no nosso projetinho, bora lá?

Primeira parte

Gatinhas, sigam esse passo a passo que é sucesso:

1- Abro meu projeto
2- Abro o terminal na raiz do projeto
3- Executo os seguintes comandos:
$ npm i swagger-autogen swagger-ui-express
$ touch swagger.js (isso fará com que um arquivo swagger seja criado no nosso projeto)
$ mkdir swagger/ (isso fará com que uma pasta swagger seja criada no nosso projeto)
4- Depois da criação da pasta, vamos no arquivo *swagger.js* e adicionamos esse pedaço de código:

```
const swaggerAutogen = require('swagger-autogen')();
const outputFile = './swagger/swagger_output.json';
const endpointsFiles = ['./src/app.js'];
swaggerAutogen(outputFile, endpointsFiles);
```

- Ao passarmos o app.js, todos os arquivos de rotas que adicionarmos no arquivo principal já serão interpretados no momento da criação do JSON da documentação.

5- Iremos lá no nosso package.json e faremos a seguinte alteraçao:

```
"scripts": {
 "start": "nodemon index.js",
 "swagger-autogen": "node swagger.js",
}
```

6- Em seguida digitaremos o seguinte comando la no terminal:
npm run swagger-autogen

Note que foi criado um arquivo chamado *swagger_output.json* dentro da nossa **pasta swagger.**

Parabéns, vc ja tem sua documentação!!!!

Segunda parte

Que tal deixarmos nossa documentaçao mais visual?

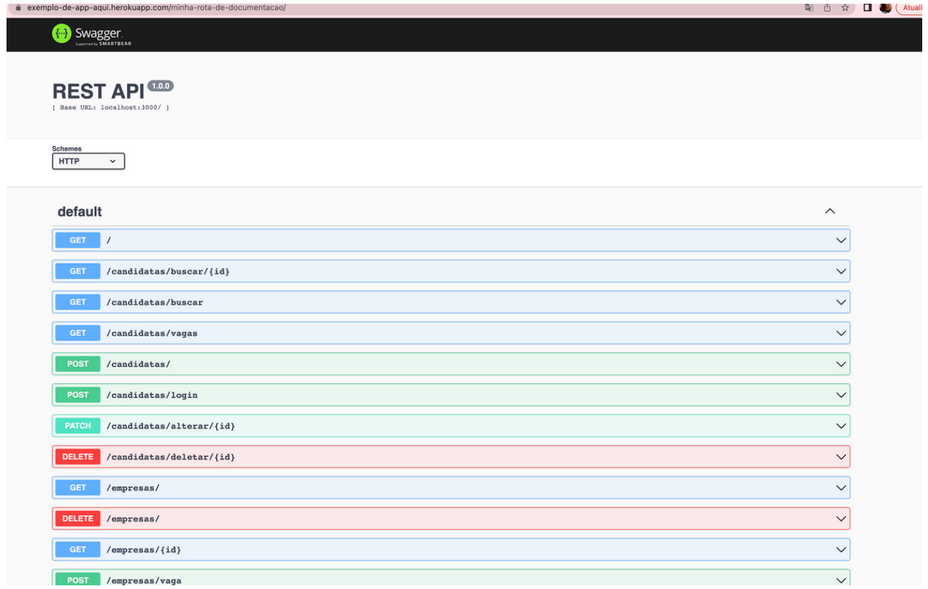
7- Vamos lá no nosso app.js e adicionaremos o seguinte código:

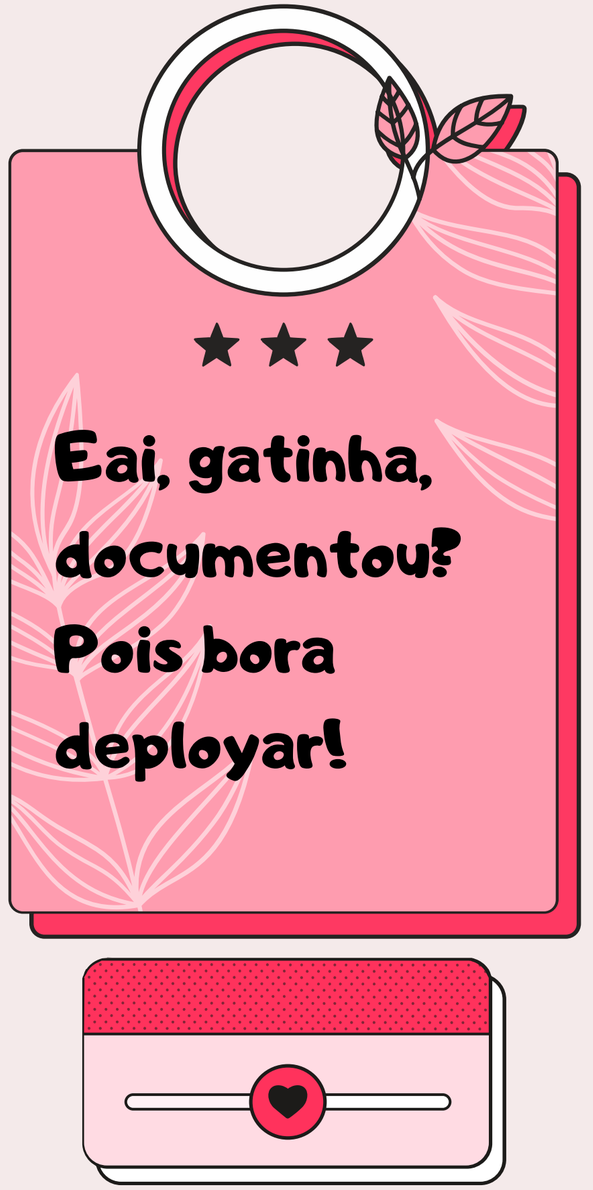
```
const swaggerUi = require('swagger-ui-express');
const swaggerFile = require('../swagger/swagger_output.json');

app.use('/minha-rota-de-documentacao', swaggerUi.serve, swaggerUi.setup(swaggerFile));
```

8- Em seguida, inicializaremos nosso projeto:
$ npm start

9- Feito isso, acessaremos a nossa rota
localhost:8080/minha-rota-de-documentacao





O que é **HEROKU**?
É uma Plataforma como um Serviço (PaaS) que permite hospedagem, configuração, testagem e publicação de projetos virtuais na nuvem. Entre outras funções, ele facilita o trabalho dos desenvolvedores na configuração da infraestrutura para o deploy, ou seja, a implantação das aplicações.

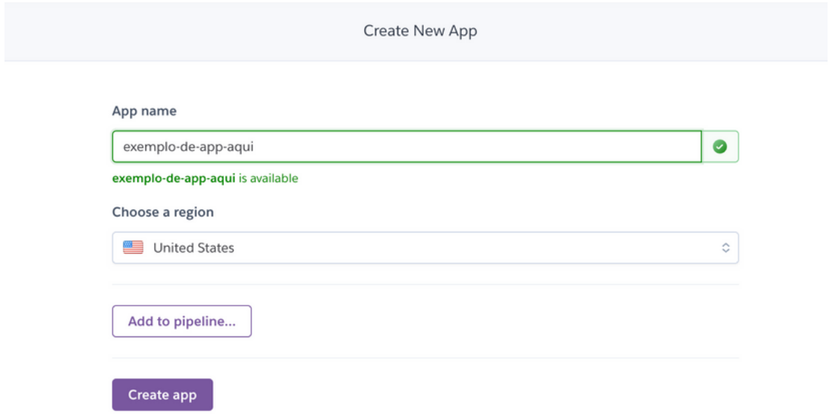
SHORT WORDS: Um computador(servidor) que vai ficar com sua aplicaçao aberta e disponibilizada para que outros acessem.

Agora vamos aprender como colocar ele no nosso projetinho, bora lá?

Gatinhas, sigam esse passo a passo que é sucesso:

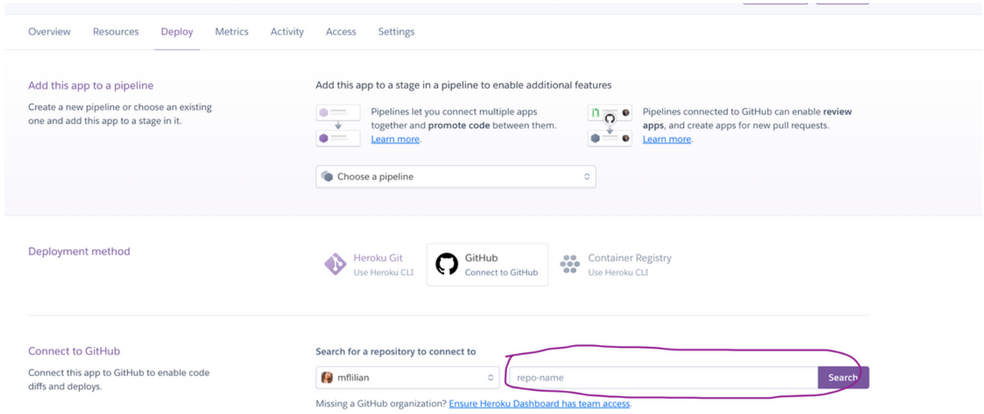
1- Iremos conectar nosso repositório com o heroku, para isso precisaremos criar uma conta no heroku:
https://www.heroku.com
e crie o seu primeiro app clicando no botão "new".

2-Coloque o nome do seu projeto e escolha a região em que o seu projeto ficará, pode escolher os EUA ou Europa, não faz diferença nesse caso.



3- Na area de deploy conecte-se à sua conta no Github como método de deployment.

4- Em seguida, encontre o github do projeto que você deseja dar deploy, e entao escolha qual branch deseja dar deploy.



Agora bora la no projeto de novoooo, ta bom?!

Após criar sua conta no Heroku e conectar seu repositório com o Heroku, algumas mudanças devem ser feitas no seu projeto.

Primeiramente, precisamos criar um arquivo chamado **Procfile**.

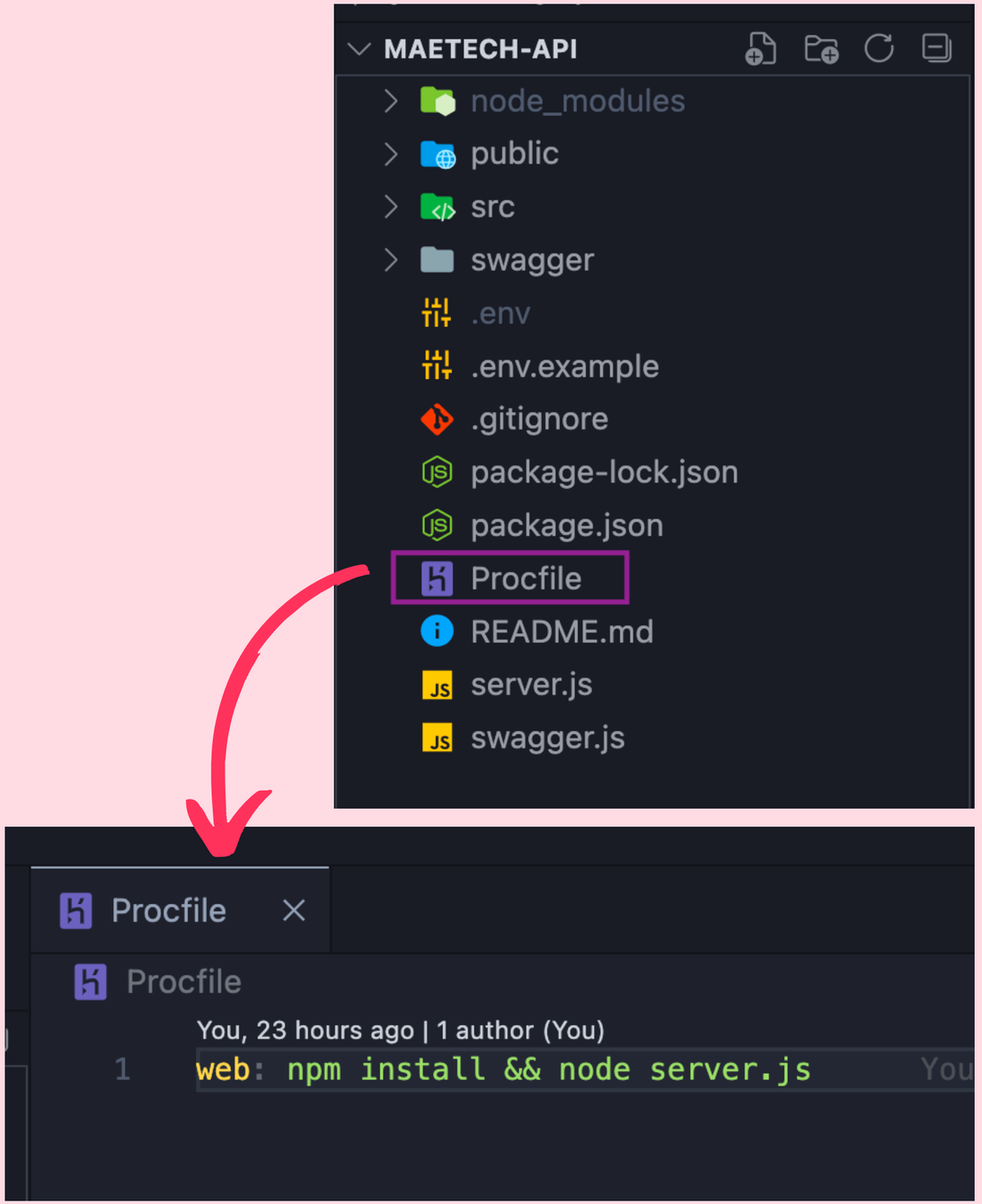
O heroku espera que esse arquivo esteja na raiz do seu projeto, onde fica o server.js, package.json e package-lock.json.
Então, vamos abrir o terminal na raiz do nosso projeto e digitar:

$ touch Procfile

O Procfile é que vai guiar os primeiros comandos para rodar seu projeto.

Dentro desse arquivo vamos digitar:

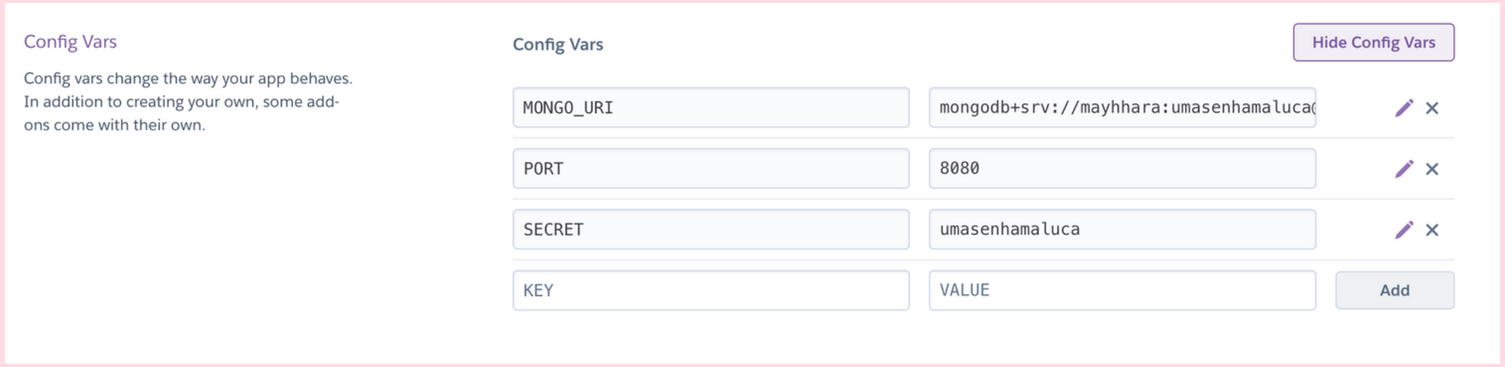
web: npm install && node server.js



Conectando o Cluster com o projeto no Heroku

Agora, lá no app do **Heroku** iremos na aba de ***Settings.***
Nós vamos adicionar as variáveis de ambiente necessárias para o projeto.
Em ***Settings*** vamos para a área de ***Config Vars***
Vamos criar as variáveis de ambiente do projeto, em ***Config Vars*** clicamos em ***Reveal Config Vars.***
Teremos então o campo de ***KEY (chave)*** e o campo de ***VALUE (valor)***, nele colocaremos a chave e o valor criado por nós no arquivo de configuração do banco de dados do seu projeto a ***MONGODB_URI***

Depois de adicionar o nome da variável e o valor clicamos em **add**
Se você tem outras variáveis de ambiente no seu .env, como o ***SECRET*** ou qualquer outra que seu projeto dependa pra funcionar você também deve fazer esse processo de adicionar nas **config vars** as chaves e os valores correspondentes.

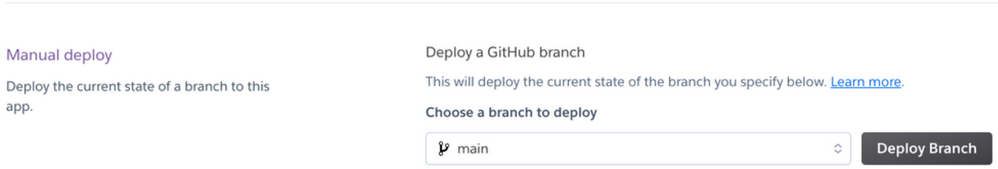


Fez todos os passos? Seu código ta sem erro?
Vamo deployar pra ver se ta tudo ok?

Primeiro, você vai precisar subir as alterações feitas no seu projeto:

$ git add .
(pra adicionar os arquivos alterados)
$ git commit -m 'meu commit'
(pra commitar os arquivos alterados)
$ git push origin minha-branch
(pra subir pro repositorio os arquivos alterados)

Depois disso, voltaremos no app do **Heroku** e acessaremos a aba de **Deploy**. Seguiremos para **Manual Deploy,** selecionaremos a branch e entao clicaremos em **Deploy Branch**

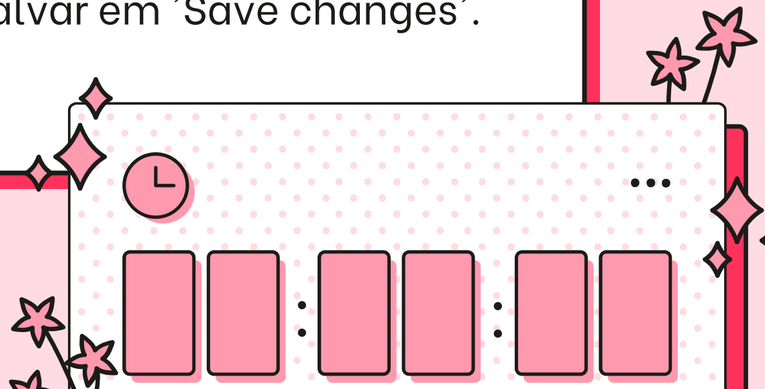


Depois disso, um terminalzinho aparecerá para você. Mas logo ele vai ficar todo verdinho e você pode clicar em **View**, para abrir sua linda api.

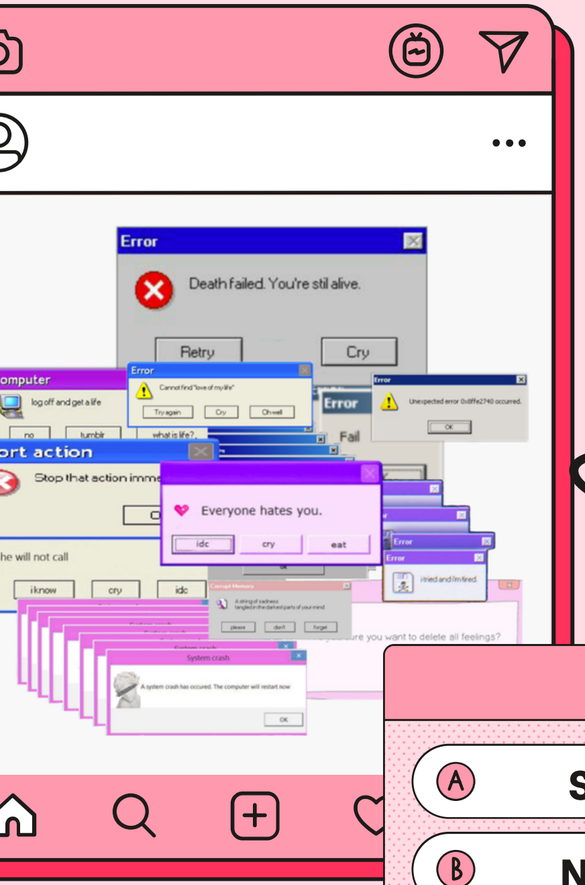
Deu certo? Parabéns! Nossa rota está no ar!

DICA1: Ative o deploy automático para que toda vez que vc der push nessa branch as modificações automaticamente sejam atualizadas

DICA2: Lembre-se de conferir se na aba de '**Settings**' tem 'Node.js' no seu buildpack.
Ih, May! Checkei e não tem, o que faço?
Você vai clicar em 'add buildpack' e selecionar um dos buildbacks oficiais ou a URL, no caso do Node.js é só selecionar o ícone dele e salvar em 'Save changes'.



Dúvidas?
Erros?
Questionamentos?



A SIM

B NÃO