Gabarito dos exercícios

A seguir, encontra-se sugestões de solução para os exercícios propostos.

Exercício 1 : Crie uma API simples que responda com "Está vivo!!!" utilizando *express* e faça o deploy no `Heroku` utilizando o CLI.

Resolução

1. Crie uma nova pasta para o projeto.

2. Inicialize o projeto com `npm init` . Adicione o script `"start": "node index.js"` ao seu `pacakge.json` . Ele deverá ficar parecido com esse:

```
{
  "name": "test-heroku",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "node index.js",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

3. Inicialize um novo repositório git:

Copiar

```
git init
git add .
git commit -m 'First commit'
```

4. Instale o express com o npm:

Copiar

```
npm install express
```

5. Adicione um arquivo `index.js` na raiz do projeto. Ele deverá ser parecido com o abaixo:

Copiar

```
const express = require('express');

const app = express();

app.get('/', (req, res) => {
  res.send('Está vivo!!!')
});

const port = process.env.PORT || 3000;

app.listen(port);
console.log(`Escutando na porta ${port}`);
```

5. Inicialize o Heroku com o comando `heroku create` .

6. Crie um arquivo `.gitignore` na raiz do projeto com o conteúdo `node_modules/` .

7. Commite as alterações:

Copiar

```
git add .
git commit -m 'Install express and add index.js'
```

8. Publicar no Heroku

Copiar

```
git push heroku master
```

10. Aguarde o deploy terminar e acesse o link exibido no terminal. Ao abri-lo no browser, deverá aparecer a mensagem `Está vivo!!!` .

Exercício 2 : Agora, modifique a API para responder uma nova mensagem:

1. Crie uma nova variável de ambiente com um texto qualquer;
2. Modifique o código da sua API para que ela responda com o texto contido na variável criada no passo anterior.

Resolução

1. Adicione a variável com o seguinte comando no terminal:

Copiar

```
heroku config:set MESSAGE='Variáveis funcionam!!!' --app nome-do-seu-app-12345
```

2. Modifique o arquivo `index.js` como abaixo:

Copiar

```
const express = require('express');

const app = express();

const port = process.env.PORT || 3000;
const message = process.env.MESSAGE || 'Está vivo!!!';

app.get('/', (req, res) => {
  res.send(message);
});

app.listen(port);
console.log(`Escutando na porta ${port}`);
```

3. Adicione e commite as alterações:

Copiar

```
git add .
git commit -m "Adiciona mensagem de response por variável de ambiente"
```

4. Faça deploy das alterações:

Copiar

```
git push heroku master
```

5. Após o deploy terminar, recarregue a página e deverá aparecer no navegador a mensagem `Variáveis funcionam!!!` .

Exercício 3 . Simule o deploy em multi ambientes (produção e homolação). Para isso:

1. Renomeie o *remote* do *deploy* dos exercícios anteriores para `homolog` ;
2. Crie um novo *remote* a partir do mesmo projeto. Dessa vez, o *remote* deverá se chamar `prod` ;
3. Em seguida, configure as variáveis de ambiente para terem valores diferentes por ambiente.

Resolução

1. Renomeie o *remote* padrão:

Copiar

```
git remote rename heroku homolog
```

2. Criar um novo *remote* chamado `prod`

Copiar

```
heroku create --remote prod
```

3. Altere a variável de ambiente de `homolog` para uma mensagem específica para o ambiente:

Copiar

```
heroku config:set MESSAGE="HOMOLOG: Variáveis de ambiente funcionam" --app NOME-DO-APP-DE-HOMOLOG
```

4. Crie a variável de ambiente de `prod` com uma mensagem específica para o ambiente:

Copiar

```
heroku config:set MESSAGE="PROD: Variáveis de ambiente funcionam" --app NOME-DO-APP-DE-PROD
```

5. Faça deploy do *app* de `prod` :

Copiar

```
git push prod master
```

6. Abre no navegador os dois apps. Cada um vai exibir uma mensagem diferente, descrevendo qual ambiente está: `homolog` ou `PROD` .

Exercício 4 : Faça deploy de uma aplicação React.

1. Crie uma aplicação React utilizando `create-react-app` ;
2. Crie um novo *app* utilizando o *buildpack* mars/create-react-app;
3. Então, faça o deploy do *app* no Heroku.

Resolução

1. Com o `create-react-app` devidamente instalado, inicie um novo app:

Copiar

```
npx create-react-app my-app
```

2. Entre na pasta do projeto. Se necessário, inicie o um novo repositório git e commite os arquivos. Enfim, crie um novo *Heroku app* :

Copiar

```
cd my-app

# Só necessário se CRA não criar automaticamente um novo repositório
git init
git add .
git commit -m 'react-create-app on Heroku'
```

```
→ heroku-react git:(master) heroku create -b mars/create-react-app
Creating app... done, ⬢ salty-garden-31796
Setting buildpack to mars/create-react-app... done
https://salty-garden-31796.herokuapp.com/ | https://git.heroku.com/salty-garden-31796.git
→ heroku-react git:(master) git remove -v
git: 'remove' is not a git command. See 'git --help'.

The most similar command is
        remote
→ heroku-react git:(master) git remote -v
heroku  https://git.heroku.com/salty-garden-31796.git (fetch)
heroku  https://git.heroku.com/salty-garden-31796.git (push)
→ heroku-react git:(master)
```

3. Publique o *app*

Copiar

```
git push heroku master
```