TRYBE
Modulo III – Back-end

Bloco 29: Deployment

1) Infraestrutura – Deploy com Heroku

Definições

DevOps: Dev (desenvolvimento) e Ops (operações) unindo pessoas, processos e tecnologia para entrega valor. Funções coordenadas.

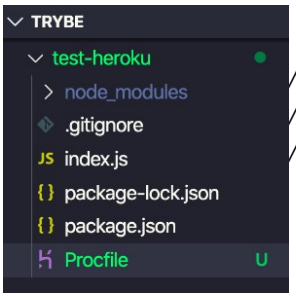
Deploy: Processo de publicar uma aplicação em um servidor, tornando-a disponível para ser acessada local ou externamente.

Serviços em nuvem
Vantagem: abstraem as complexidades de se administrar um servidor e suas diversas camadas.
Exemplos: Heroku, Google GCE, Amazon AWS, Microsoft Azure, IBM Cloud.

Intro - Heroku

Platform as a Service. *// vermelho é o que cuida para nós, azul temos que manipular*
Plataforma poliglota, precisa saber linguagem e framework usado no projeto.
Build: processo cujo código é preparado para posteriormente ser executado.

Procfile
Arquivo que especifica o comando que deve ser executado para iniciar o projeto.



// na pasta raiz do projeto
// <process type>: <command>
// ex: web: node index.js

Dynos
Containers dentro dos quais são posicionados os projetos a serem deployed. Escalabilidade.

Instalar
Snaped
sudo snap install hello-world
(opcional) *apt-get update && apt-get install snapd*
CLI (Command Line Interface)
sudo snap install heroku –classic
Login com comando *heroku login*

Usar Heroku

Inicializar
npm install #allDependenciesNeeded
heroku local web

Listar o remote (ou seja visualizar o origin, repositório remoto do git): *git remote -v*

Criar projeto para deploy
* *npx create-react-app meu-primeiro-deploy-heroku*
* Sequência *git init - git add . - git commit -m 'Initialize project using Create React App'*
* Criar repositório no github
* Linkar ao repositório

git remote add origin https://github.com/[SEU_USUARIO_GITHUB]/meu-primeiro-deploy-heroku.git

Criar remote heroku
heroku create
heroku create username –remote remotename
(exemplo: heroku create juliettebeaudet-bk --remote hawkins) (username opcional)
git remote rm heroku (para excluir)

Nomear
heroku create meu-deploy-de-testes-29302 --remote heroku-homolog
git remote rename heroku heroku-origin
→ Bom ter diversos apps do Heroku do mesmo código fonte (versões testes, staging, produção).

Vincular app existente a novo remote
heroku git:remote -a nome-do-seu-app-heroku --remote nome-do-seu-remote

Exemplo: *heroku git:remote -a meu-deploy-de-testes-29302 --remote heroku-test*

Buildpack
Conjunto de scripts para executar durante o deploy.
Útil para publicar apps React front-end sem precisar do back-end server-side.
Consultar mars, nginx e todo catálogo.

Criar o app heroku jà integrando buildpack:
heroku create $APP_NAME --buildpack mars/create-react-app
(exemplo: heroku create juliettebeaudet-ft --remote juliettebeaudet-ft --buildpack mars/create-react-app)

Fazer Deploy

Relação com a master
→ Dar git push do local para o remoto
git push heroku master
→ Se você trabalhar desde outra branch, colocar master como destino para funcionar
git push heroku branch-teste:master
→ Com nome diferente de de heroku como remote:
git push remotename branchname:master
→ *Cuidado: não será possível cancelar um deploy com ctrl c.*

Lidar com vários deploys
As versões serão publicadas *na ordem em que os processos forem concluídos*, e não na ordem em que os comandos push forem realizados.

Acompanhar a aplicação

Gerir apps
heroku apps – listar serviços em execução
heroku apps:info nome-do-seu-app-12345 – ver detalhes de um app em particular

Variáveis de ambiente
heroku config:set TESTE="texto qualquer" --app nome-do-seu-app-12345 – setar variáveis de ambiente
heroku config --app nome-do-seu-app-12345 - listar variáveis de ambiente

Logs
heroku logs --app nome-do-seu-app-12345 – monitorar os logs dos apps
*heroku logs **-n** 200 --app nome-do-seu-app-12345* - mudar o n° padrão (100) de lista de logs (ou –***num***)
*heroku logs **--tail** --app nome-do-seu-app-12345* – mostrar em tempo real os últimos logs (ou –***t***)

Remover app do Heroku
heroku destroy --app nome-do-app-12345 --confirm nome-do-app-12345
Exemplo: *heroku destroy --app meu-deploy-de-testes-29302 --confirm meu-deploy-de-testes-29302*

heroku destroy –app enigmatic-earth-26654 –confirm enigmatic-earth-26654

---

2) Deploy – Gerenciadores de processos

Importância de **gerenciar o ciclo de vida das aplicações num ambiente de produção**.

Imaginar problema:
Aplicação caiu de madrugada e não tinha um process manager que automaticamente tratou para que não parasse para o usuário.

Vantagens dos Process Managers (PMs) →

- Reload automático;
- Abstração da complexidade de gerenciadores nativos;
- Gerenciamento de sessões;
- Facilidade de gerenciamento de múltiplos cores;
- Responsabilidade do uso de cores delegados ao PM;
- Gerenciamento de múltiplas aplicações no servidor;
- Escalonamento dos processos;
- Balanceamento de carga;
- Monitoramento;
- Gerenciamento de logs.

PMs populares:
- PM2;
- StrongLoop's PM ;
- Forever ;
- SystemD do Linux.

PM2

Popular particularmente na comunidade Node.js.
Instalar, checkar e atualizar
\$ npm install pm2@latest -g ou *sudo npm install pm2@latest -g*
\$ pm2 –version
\$ pm2 update

Gerenciar processos

Comandos básicos da PM2

Start
\$ pm2 start index.js
\$ pm2 start index.js --name <NOME_DO_PROCESSO>
Todo processo também contem id: *pm2 start 0* (para começar o processo do id 0).

Stop
\$ pm2 stop <NOME_DO_PROCESSO> (e pode dar start novamente)
\$ pm2 stop all (em todos os processos)

Delete, Restart, Reload
\$ pm2 delete <NOME_DO_PROCESSO> (excluir da lista de processo do PM2)
\$ pm2 restart <NOME_DO_PROCESSO>
\$ pm2 reload <NOME_DO_PROCESSO> (primeiro sobe o novo processo e depois finaliza o anterior, 0-second-downtime comparado com restart)

Monitorar processos

List
\$ pm2 list
\$ pm2 ls
\$ pm2 l

```
>>> pm2 ls
┌────┬─────────┬─────────┬─────────┬─────────┬──────────┬────┬──────────┬──────┬──────────┬──────────┬──────────┐
│ id │ name    │ version │ mode    │ pid     │ uptime   │ ↺  │ status   │ cpu  │ mem      │ user     │ watching │
├────┼─────────┼─────────┼─────────┼─────────┼──────────┼────┼──────────┼──────┼──────────┼──────────┼──────────┤
│ 0  │ Worker  │ 4.0.0   │ fork    │ 40209   │ 2s       │ 0  │ online   │ 6.7% │ 32.2mb   │ alex     │ disabled │
│ 1  │ HTTP-API│ 4.0.0   │ cluster │ 40208   │ 2s       │ 0  │ online   │ 4.7% │ 34.1mb   │ alex     │ enabled  │
│ 2  │ HTTP-API│ 4.0.0   │ cluster │ 40211   │ 2s       │ 0  │ online   │ 4.7% │ 34.3mb   │ alex     │ enabled  │
└────┴─────────┴─────────┴─────────┴─────────┴──────────┴────┴──────────┴──────┴──────────┴──────────┴──────────┘
host metrics | cpu: 6.8% | mem: 63.9% | net: 12.5ms ↓0.009mb/s ↑0.001mb/s | disk: ↓6.898mb/s ↑0.01mb/s /dev/disk1s1 89% |
```

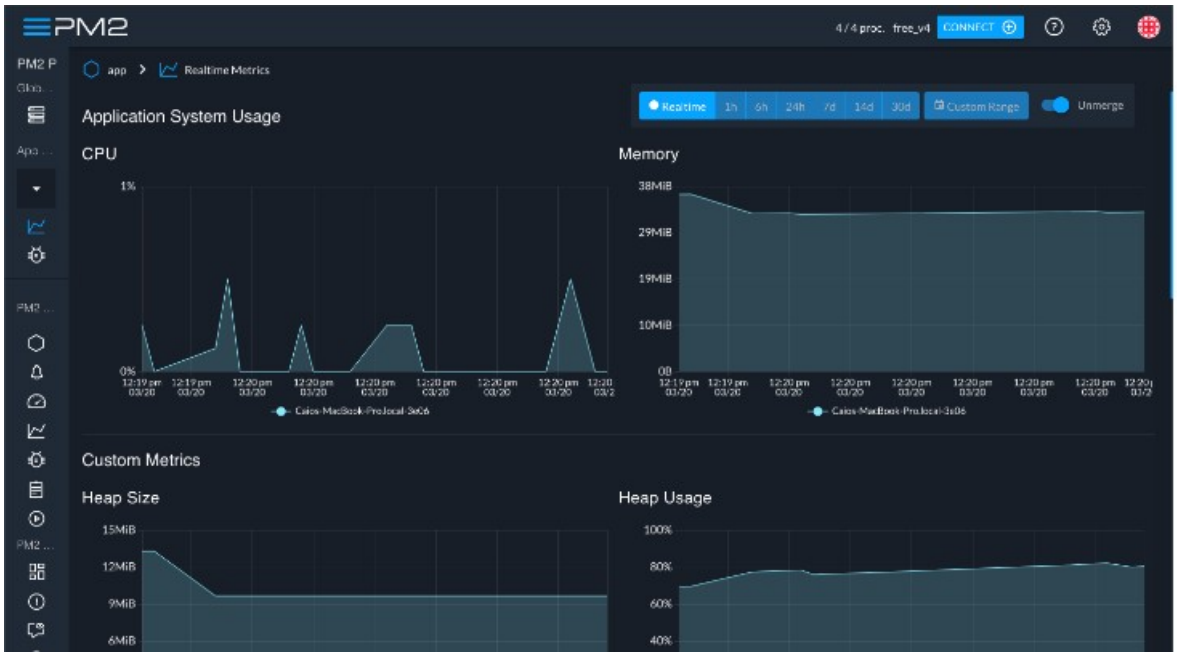
Com **sort**, por name , id , pid , memory , cpu , status ou uptim:
\$ pm2 list --sort name:desc

Show (para mais detalhes sobre um processo)
\$ pm2 show <NOME_DO_PROCESSO>

Logs
\$ pm2 logs <NOME_DO_PROCESSO>

Monit (ver dashboard em tempo real)
\$ pm2 monit

Interface Web
Monitorar usando um dashboard do PM2, plano free.
- Criar conta
- rodar *\$ pm2 plus*



//informações exibidas: consumo de recursos e configurações da máquina

Conectar chave no heroku:
Criar bucket, ver chaves
Terminal:
*\$ heroku config:set *
*PM2_PUBLIC_KEY=6bwm0n7mef7lb3o *
*PM2_SECRET_KEY=m4k1wg2bkn25m36 *
*PM2_MACHINE_NAME=NOME_DO_SERVER *
--app NOME_DO_APP_NO_HEROKU

Modo Cluster

Permite **escalar a aplicação Node entre as CPUs disponíveis** na máquina, via **load balancing** (carga repartida entre CPUs – um processo para um CPU). Aumenta performance e resiliência.
Uso do Node.js cluster module.

Chamar via ***–instances*** ou ***-i*** em comandos de start, reload ou restart.
Pode definir o n° de processos em paralelo via número, max ou 0:
\$ pm2 start index.js --instances 2 --name <NOME_DO_PROCESSO>
\$ pm2 start index.js -i max --name <NOME_DO_PROCESSO>

Outro jeito de aumentar n° de processos: **Scaling**
\$ pm2 scale <NOME_DO_PROCESSO> 3
\$ pm2 scale <NOME_DO_PROCESSO> +3
(com total ou adição de processos)

Stateless
Para escalar horizontalmente, cachear e ter menos complexidade dos storages.

Ecosystem file

Arquivo de configuração para o PM2 executar aplicações.
Bom para padronizar dentro da mesma equipe.

Usar
\$ pm2 [start|restart|stop|delete] ecosystem.config.js|exosystem.config.yml

Escrever: escolha entre linguagens
Javascript
```
module.exports = {
  apps: [
    {
      name: 'app',
      script: './index.js'
    },
    //...
  ]
};
```

YAML
```
apps:

- name: app-1
  script: .app-1/index.js
- name: app-2
  script: .app-2/index.js
  exec_mode: cluster
  instances: 4
  env_prod:
    ENVIRONMENT: PRODUCTION
  env_homolog:
    ENVIRONMENT: HOMOLOG
```

// com multiaplicativos, instâncias, variáveis de ambiente que deve passar assim:
// \$ pm2 start ecosystem.config.yml --env homolog

JSON.

Auto restart

PM2 reinicia automaticamente processos que falharam.
Possibilidade de outras **configurações para esses restarts**.

Memória máxima
\$ pm2 start index.js --name <NOME_DO_PROCESSO> --max-memory-restart 20M
OU
apps:

- name: app
script: ./index.js
max_memory_restart: 20M

Delay de restart
\$ pm2 start index.js --name <NOME_DO_PROCESSO> --restart-delay 100
OU
apps:

- name: app
script: ./index.js
restart_delay: 100

Estratégias de backoff
Configurar a aplicação para reiniciar de maneira mais inteligente, em vez de somente ficar reiniciando sempre que houver uma exceção.

Exponential backoff para aumentar prazo entre cada tentativa (status *waiting restart)*:
Adicionar a tag *--exp-backoff-restart-delay* mais o tempo de delay no start
OU
apps:

- name: app
script: ./index.js
exp_backoff_restart_delay: 100

Assistir alterações

Ficar observando um diretório específico e, caso haja alterações nos arquivos, ele automaticamente reinicia os processos.

\$ pm2 start index.js --name <NOME_DO_PROCESSO> ***--watch***

OU

apps:

- name: app
* script: ./index.js*
* watch: ./*
(especificando quais diretórios deverão ser observados)

PM2 com outras linguagens

PM2 consegue inferir a linguagem.

Lista default
{
".sh": "bash",
".py": "python",
".rb": "ruby",
".coffee" : "coffee",
".php": "php",
".pl" : "perl",
".js" : "node"
}

Configurar interpretador desejado
\$ pm2 start hello-world.py --interpreter=python

PM2 com Heroku

Podemos fazer um deploy no Heroku utilizando os recursos disponíveis do PM2!

Adicionar módulo ao projeto
\$ npm install pm2

No package.json
// ...
"scripts": {
* "start": "pm2-runtime start ecosystem.config.yml"*
}
// …
// pm2-runtime agrupa os aplicativos em um ambiente de produção adequado do Node.js, resolve problemas de execução de aplicativos Node.js dentro dos containers, como controle de fluxo de processo, monitoramento automático de aplicativos.

No ecosystem.config.yml

apps:

- name: app
* script: ./index.js*

→ Finalmente, seguir com o deploy com Heroku.

Aprofundar modos completos

Cluster + Heroku

apps:

- name: app
script: ./index.js
exec_mode: cluster
instances: max

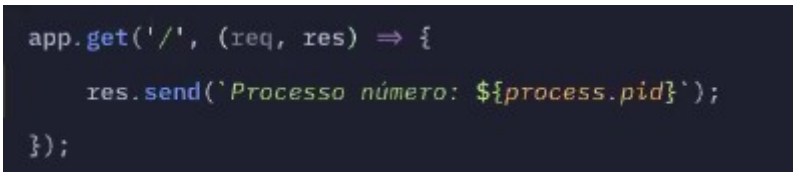
Cluster + Heroku + Dashboard

Configurar e adicionar o dashboard com chaves e credenciais dadas.

*\$ heroku config:set *
*PM2_PUBLIC_KEY=CHAVE_PUBLICA *
*PM2_SECRET_KEY=CHAVE_PRIVADA *
*PM2_MACHINE_NAME=NOME_DO_SERVER *
--app NOME_DO_APP_NO_HEROKU

Dicas diversas

AWS em alguns casos tem gerenciador próprio.
Essa parte também pode não ser mais do desenvolvedor, fora devOps.
–
Node tem acesso ao process.pid ou seja elementos de processos do meu sistema operacional:



–
random-hex-color biblioteca js que retorna cor
–
Docker é parecido mas não gerencia processo, gerencia container.
–
Escolher entre YAML, JS e JSON: yaml mais legível que Json e estático, diferença do Js dinâmico.
–
env do ecosystem file VS do .env : ecosystem tem configurações pra vários ambientes (permite versionar), .env tem segredos pro seu ambiente local.

---

3) Projeto Stranger Things

Dicas diversas aprendidas no projeto: comandos Heroku (invisiveis no repo)

heroku config:set UPSIDEDOWN_MODE=false --app juliettebeaudet-bk
heroku config:set UPSIDEDOWN_MODE=true --app juliettebeaudet-bd

heroku config --app juliettebeaudet-bk

git remote rename juliettebeaudet-ft development
heroku create juliettebeaudet-pd --remote juliettebeaudet-pd --buildpack mars/create-react-app

heroku config:set ENV_MODE='dvlpt' --app juliettebeaudet-ft

heroku config:set REACT_APP_UPSIDEDOWN_TIMEOUT=30000 --app juliettebeaudet-pd

git push development juliette-st-frontend:master
git push juliettebeaudet-pd juliette-st-frontend:master