Disciplina: **DEVOPS**
Professor: **Delano Medeiros Beder**
Aluna: Karys Cristina da Silva Barbosa **RA: 811871**
Aluno: Matheus dos Santos Sousa **RA: 812051**

JULHO 2025

1. APLICAÇÃO

Catlog é uma aplicação desenvolvida inicialmente na disciplina de Desenvolvimento para Web 2, utilizando as tecnologias Node.js no backend e React.js no frontend. Na etapa seguinte, aproveitamos a mesma aplicação como base para o projeto da disciplina atual, onde trabalhamos o conceito de conteinerização com Docker.

Para esta entrega, evoluímos a infraestrutura da aplicação com o uso do Minikube como orquestrador de containers, organizamos os recursos com Helm Charts, e configuramos o Ingress Controller, permitindo que a aplicação seja acessada de forma adequada dentro do ambiente Kubernetes.

2. ARTEFATOS

Explicando os contêineres utilizados

Ao longo do projeto criamos 5 contêineres e são eles:

Backend:
Responsável por toda a lógica da aplicação. Trata as requisições dos usuários, realiza a validação dos dados e interage com o banco de dados. Também é responsável por gerenciar o processo de autenticação e fornecer os dados necessários para o frontend via API REST.

Banco de Dados (BD):
Sistema de armazenamento dos dados da aplicação. Utiliza MySQL para persistir informações de usuários, postagens, imagens e credenciais. Está configurado com volume persistente (PVC) para garantir que os dados não sejam perdidos entre reinícios dos pods.

Frontend:
Interface gráfica da aplicação. Desenvolvido com ReactJS, permite que o usuário visualize e interaja com os conteúdos, como realizar cadastro, login, criar postagens e navegar entre seções. Toda comunicação com o backend é feita via requisições HTTP.

Send-email:

Serviço dedicado ao envio de e-mails. Assim que um novo usuário realiza o cadastro, esse serviço é acionado automaticamente para enviar um e-mail de confirmação ou boas-vindas. Funciona de forma desvinculada do backend principal, dando mais autonomia para este serviço.

Image-service:
Serviço especializado em lidar com o upload e armazenamento de imagens das postagens. Ele recebe imagens enviadas pelo frontend, realiza o processamento e as armazena, fornecendo as URLs para que sejam usadas nas postagens exibidas ao usuário.

Árvore dos diretórios e arquivos do chart da aplicação geral e dos sub charts

```
catlog-chart/
├── Chart.yaml
├── Chart.lock
├── values.yaml
├── .helmignore
├── charts/
│   ├── backend/
│   │   ├── Chart.yaml
│   │   ├── values.yaml
│   │   └── templates/
│   │       └── deployment.yaml
│   │       └── service.yaml
│   │       └── secret.yaml
│   ├── db/
│   │   └── ... (estrutura similar: Chart.yaml, values.yaml, templates/)
│   ├── frontend/
│   │   └── ... (idem)
│   ├── image-service/
│   │   └── ... (idem)
│   └── send-email/
│       └── ... (idem)
├── templates/
│   └── ... (templates globais, ex: ingress.yaml)
└── node/
```

3. **COMANDOS**

```
#!/bin/bash

set -e  # Interrompe em caso de erro
set -x  # Exibe os comandos sendo executados

# Inicia o Minikube
minikube start
```

```
# Configura o ambiente Docker para usar o Minikube
eval $(minikube docker-env)

# Faz o build das imagens Docker
docker build -t trabalho2devops-backend:latest ./backend
docker build -t trabalho2devops-frontend:latest ./frontend
docker build -t trabalho2devops-image-service:latest ./image-service
docker build -t trabalho2devops-send-email:latest ./send-email

# Carrega a imagem do Ingress Controller no Minikube (caso necessÃ¡rio)
minikube image load k8s.gcr.io/ingress-nginx/controller:v1.9.4

# Habilita os addons essenciais
minikube addons enable ingress
minikube addons enable dashboard
minikube addons enable metrics-server

# Exibe o IP do Minikube para ser colocado no /etc/hosts
minikube_ip=$(minikube ip)
echo -e "\nAdicione a seguinte linha ao seu /etc/hosts (como root):"
echo -e "${minikube_ip}\tcatlog.k8s.local\n"

# Aguarda o usuÃ¡rio confirmar ediÃ§Ã£o do hosts
read -p "Pressione ENTER apÃ³s editar o /etc/hosts..."

# Aplica o Helm chart
helm install catlog ./catlog-chart
```

4. **SCRIPT**

Deixamos disponível no nosso repositório do projeto no nome `deploy.sh` e pode ser executado seguindo as instruções do nosso ReadMe.md .

Link do nosso repositório:
https://github.com/mathdsousa/Trabalho2DevOps