Documentação de Deployment do Jupyter Notebook

Este documento descreve o processo completo para configurar e implantar o **Jupyter Notebook** em um cluster Kubernetes local utilizando **Kind** (Kubernetes in Docker). Abaixo estão os detalhes sobre a criação do cluster, configuração do namespace, deployment do Jupyter Notebook, e acesso ao ambiente.

`jupyter-deployer.sh`

Este script automatiza a criação do cluster Kind, configuração do namespace e deployment do Jupyter Notebook, e a configuração de port-forwarding para acessar o Jupyter Notebook no navegador.

```
# Criando cluster Kind
kind create cluster --name jupyter-cluster
```

- **Ação**: Cria um novo cluster Kubernetes local usando Kind com o nome `jupyter-cluster`.
- **Por que**: O cluster Kind fornece um ambiente Kubernetes local para desenvolvimento e teste. O nome do cluster `jupyter-cluster` ajuda a identificar o ambiente específico para o Jupyter Notebook.

```
# Verificando se o cluster foi criado com sucesso
kubectl cluster-info --context kind-jupyter-cluster
```

- **Ação**: Verifica as informações do cluster para garantir que o cluster Kind foi criado corretamente.
- **Por que**: Confirmar que o cluster está funcionando e acessível é crucial antes de prosseguir com outras configurações.

```
# Criando o namespace Jupyter
kubectl apply -f ./main/deployer/jupyter/jupyter-namespace.yaml
```

- **Ação**: Aplica a configuração do namespace para o Jupyter Notebook a partir do arquivo `jupyter-namespace.yaml`.
- **Por que**: O namespace `jupyter` isola os recursos do Jupyter Notebook dentro do cluster, evitando conflitos com outros recursos e ajudando na organização.

```
# Verificando se o namespace foi criado com sucesso
kubectl get namespaces
```

- **Ação**: Lista todos os namespaces no cluster para confirmar a criação do namespace `jupyter`.
- **Por que**: Certifica-se de que o namespace foi criado e está ativo.

```
# Criando o deployment do Jupyter Notebook
kubectl apply -f ./main/deployer/jupyter/jupyter-deployment.yaml
```

- **Ação**: Aplica a configuração do deployment do Jupyter Notebook a partir do arquivo `jupyter-deployment.yaml`.
- **Por que**: O deployment configura o Jupyter Notebook como um pod no cluster, definindo o container e suas especificações.

```
# Verificando se o deployment foi criado com sucesso
kubectl get deployments -n jupyter
```

- **Ação**: Lista os deployments no namespace `jupyter` para garantir que o deployment do Jupyter Notebook está em execução.
- **Por que**: Verificar o estado do deployment ajuda a confirmar que o Jupyter Notebook está corretamente implantado e funcionando.

```
# Capturando o token do Jupyter Notebook
JUPYTER_TOKEN=$(kubectl exec -it $(kubectl get pods -n jupyter -o name | cut -d/ -f2) -n jupyter -- bash -c "jupyter notebook list" | grep http | cut -d' ' -f1 | cut -d'=' -f2)
```

- **Ação**: Executa um comando dentro do pod do Jupyter Notebook para recuperar o token de autenticação necessário para acessar a interface web.
- **Por que**: O token é necessário para autenticar o acesso ao Jupyter Notebook via navegador.

```
# Configurando port-forward para acessar o Jupyter Notebook
echo "Para acessar o Jupyter, abra o navegador e acesse http://localhost:8888?token=$JUPYTER_TOKEN"
kubectl port-forward deployment/jupyter -n jupyter 8888:8888
```

- **Ação**: Configura o port-forwarding para mapear a porta 8888 do deployment Jupyter Notebook para a porta 8888 da máquina local.
- **Por que**: O port-forwarding permite acessar a interface do Jupyter Notebook em `http://localhost:8888` a partir do navegador local.

jupyter-deployment.yaml

Arquivo de configuração para o deployment e serviço do Jupyter Notebook.

```
apiVersion: apps/v1
kind: Deployment
metadata:
```

```
  name: jupyter-deployment
  namespace: jupyter
spec:
  replicas: 1
  selector:
    matchLabels:
      app: jupyter
  template:
    metadata:
      labels:
        app: jupyter
    spec:
      containers:
      - name: jupyter
        image: jupyter/pyspark-notebook:latest
        ports:
        - containerPort: 8888
        resources:
          limits:
            memory: "2Gi"
            cpu: "1000m"
        volumeMounts:
        - name: notebook-storage
          mountPath: /home/jovyan/work
      volumes:
      - name: notebook-storage
        emptyDir: {}
---
apiVersion: v1
kind: Service
metadata:
  name: jupyter-service
  namespace: jupyter
spec:
  type: NodePort
  ports:
  - port: 8888
    targetPort: 8888
    nodePort: 30001
  selector:
    app: jupyter
```

- **Deployment**:
  - `apiVersion`: Define a versão da API usada para o deployment.
  - `kind`: Tipo de recurso (Deployment).
  - `metadata`: Informações sobre o deployment, incluindo nome e namespace.
  - `spec`: Define a especificação do deployment, como número de réplicas, seletor de labels, template do pod, e configurações do container.
  - `containers`: Especifica o container do Jupyter Notebook, incluindo a imagem Docker, portas, e limites de recursos.
  - `volumes`: Define volumes para armazenamento temporário dos notebooks.

- **Service**:
  - `apiVersion`: Define a versão da API usada para o serviço.
  - `kind`: Tipo de recurso (Service).
  - `metadata`: Informações sobre o serviço, incluindo nome e namespace.
  - `spec`: Define a especificação do serviço, incluindo o tipo `NodePort`, portas, e seletor de labels.

`jupyter-namespace.yaml`

Arquivo de configuração para o namespace onde o Jupyter Notebook será implantado.

```
apiVersion: v1
kind: Namespace
metadata:
  name: jupyter
  labels:
    name: jupyter
```

- `apiVersion`: Define a versão da API usada para o namespace.
- `kind`: Tipo de recurso (Namespace).
- `metadata`: Informações sobre o namespace, incluindo nome e labels.

Este guia fornece uma visão abrangente de cada etapa do processo de configuração e deploy do Jupyter Notebook, detalhando o propósito e a importância de cada comando e configuração.