

INSTITUTO FEDERAL

Mato Grosso

Campus Cuiabá
Cel. Octayde Jorge da Silva

Pós Graduação Lato Sensu em Engenharia DevOps



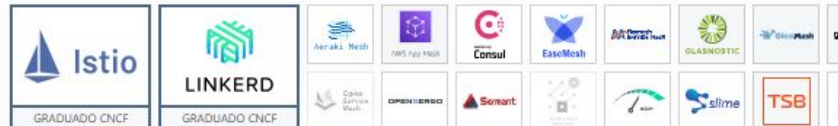
Dados

Disciplina: Arquitetura de Microsserviços e Containers

Docente: João Paulo Delgado Preti

Discentes: Marcos Roberto de Avila
Marcelo Barcelar Ricardo
Néftis Caroline Araujo da Silva
Wellington de Carvalho Batista

Malha de serviço



Armazenamento nativo em nuvem



Rede nativa da nuvem



MINIO

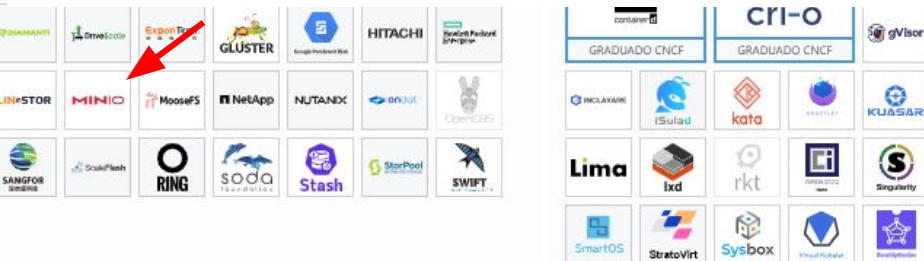
MinIO

MinIO

Tempo de execução

Armazenamento nativo em nuvem

O armazenamento de objetos para infraestrutura de dados de IA



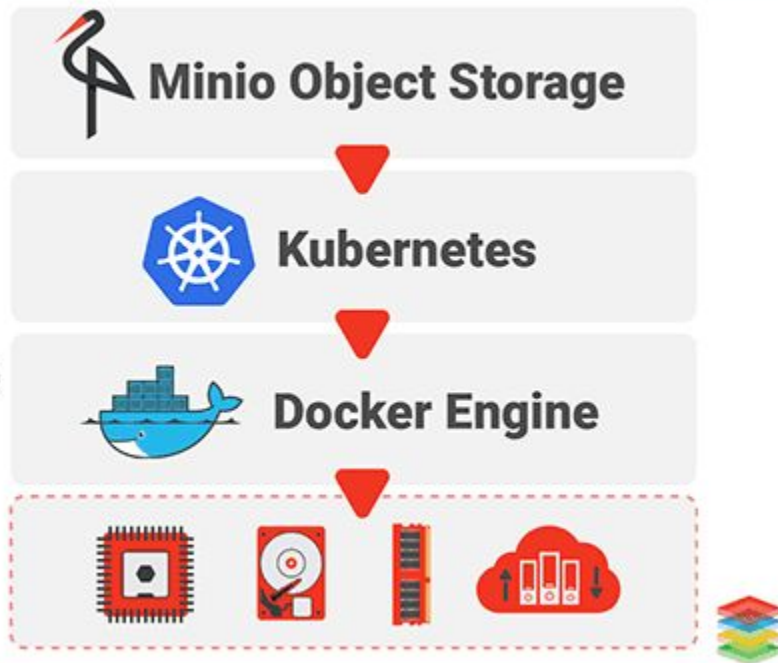
Conceitos



- O Minio é um servidor **Object Storage** distribuído de alto desempenho, que é projetado para infraestrutura de nuvem privada em larga escala. O MinIO **agrega volumes persistentes (PVs)** no Object Storage distribuído escalável usando as APIs de REST do Amazon S3. (IBM)
- MinIO é um servidor de armazenamento de objetos distribuídos(...) MinIO é o melhor servidor adequado para **armazenar dados não estruturados**, como fotos, vídeos, arquivos de log, backups e contêineres.

MINIO

Minio Architecture



O que você pode fazer com a MinIO?



Benefícios

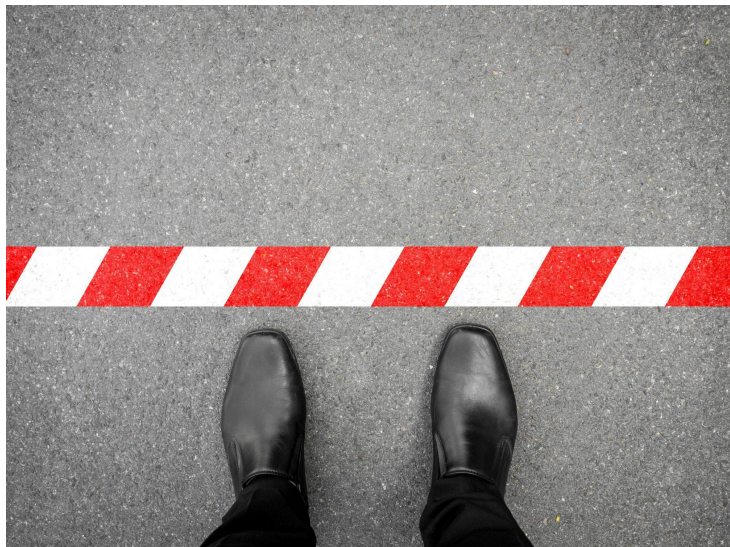


Escalabilidade

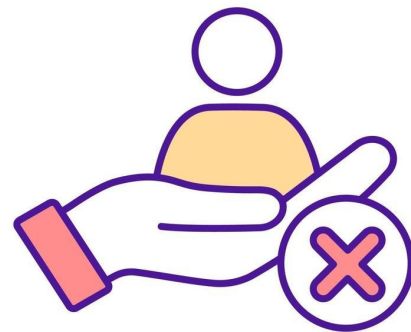


Segurança de Dados

Desafios



Limite de Recursos



Falta de Suporte

Recursos e outras Tecnologias?



IntelliJ



Visual Studio



#Instalação Docker

#Instalação do docker

```
``plaintext
```

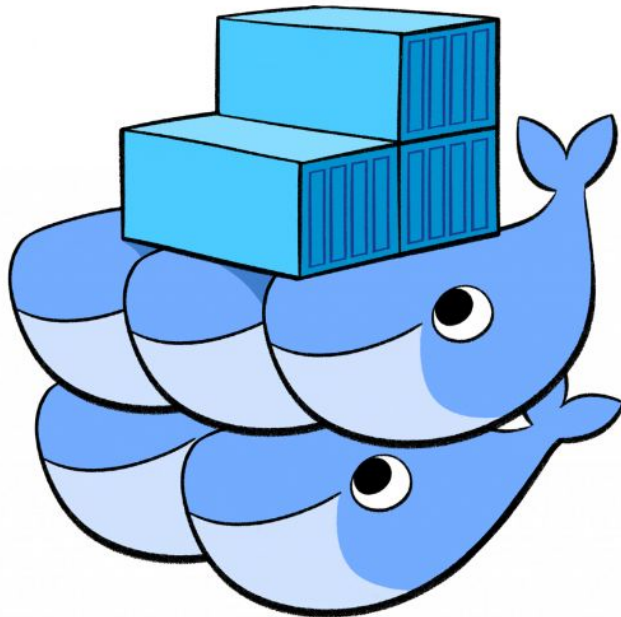
```
curl -fsSL
```

```
https://get.docker.com -o
```

```
get-docker.sh
```

```
sh get-docker.sh
```

```
```
```



## #Instalação Kind e criação de cluster

---

```
```plaintext
curl -Lo ./kind
https://kind.sigs.k8s.io/dl/v0.22.0/
kind-linux-amd64
chmod +x ./kind
sudo mv ./kind /usr/local/bin/kind
```



```
```plaintext
kind create cluster --name kind-k8s
--config kind.yml
```
```


```



## #Persistent Volume e Persistent Volume Claim



```
kubectl apply -f pv.yml
```

```
kubectl get PersistentVolume
```

```
kubectl apply -f pvc.yml
```

```
kubectl get PersistentVolumeClaim
```

## #Manifesto para implantar MINIO

---

```
kubectl apply -f deploy.yml
```

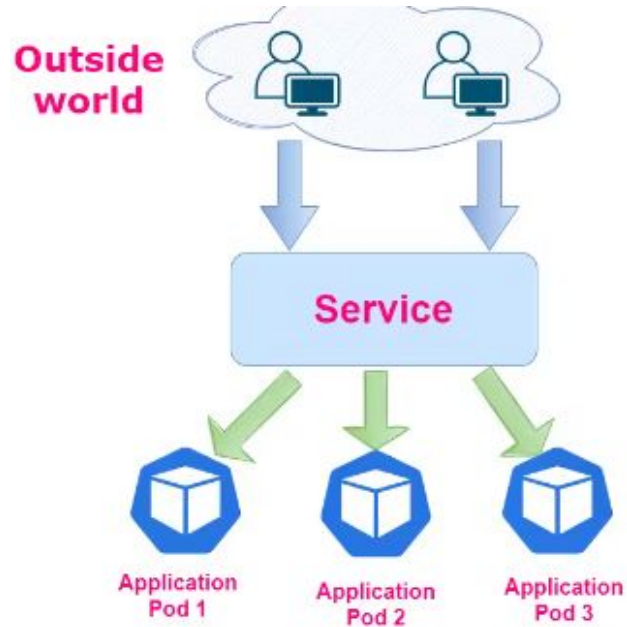
```
kubectl get pods
```

```
kubectl logs minio-6d9cd468cb-5279f
```



## #Expor a porta e aplicação do service

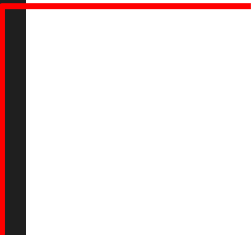
```
kubectl apply -f svc.yml
kubectl port-forward svc/minio
--address 0.0.0.0 9000:9000 & \
kubectl port-forward svc/minio
--address 0.0.0.0 41795:41795
```



## Service: Portas

```
! svc.yml X
C: > Users > Usuário_01 > Desktop > ! svc.yml
1 apiVersion: v1
2 kind: Service
3 metadata:
4 name: minio
5 spec:
6 selector:
7 app: minio
8 ports:
9 - protocol: TCP
10 port: 9000
11 targetPort: 9000
12 name: minio-port-9000
13 - protocol: TCP
14 port: 41795
15 targetPort: 41795
16 name: minio-port-random
17
18
```

```
kubectl apply -f svc.yml
kubectl port-forward
svc/minio --address 0.0.0.0
9000:9000 & \
kubectl port-forward
svc/minio --address 0.0.0.0
41795:41795
```



# Resultado :

svc.yml - projeto\_ifmt - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

- PROJETO\_IFMT
  - 43069
  - ! deploy.yml
  - \$ get-docker.sh
  - ! kind.yml
  - ! pv.yml
  - ! pvc.yml
  - ! Readme.md
  - ! svc.yml

! svc.yml

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4 name: minio
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

clusters:

- cluster:
  - certificate-authority-data: DATA+OMITTED
  - server: https://127.0.0.1:36251
  - name: kind-kind
- contexts:
  - context:
    - cluster: kind-kind
    - user: kind-kind
    - name: kind-kind
  - current-context: kind-kind
  - kind: Config
  - preferences: {}
  - users:
    - name: kind-kind
    - user:
      - client-certificate-data: DATA+OMITTED
      - client-key-data: DATA+OMITTED

o linux@linux-VirtualBox:~/Área de Trabalho/projeto\_ifmt\$ Handling connection for 9000

Handling connection for 9000

E0318 16:15:20.710620 18740 portforward.go:394] error copying from local connection to remote stream: read tcp4 127.0.0.1:9000->127.0.0.1:42636: read: connection reset by peer

E0318 16:15:20.714902 18740 portforward.go:394] error copying from local connection to remote stream: read tcp4 127.0.0.1:9000->127.0.0.1:42648: read: connection reset by peer

^C

• linux@linux-VirtualBox:~/Área de Trabalho/projeto\_ifmt\$ kubectl get svc

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	65m
minio	ClusterIP	10.96.90.115	<none>	9000/TCP,43069/TCP	23m

• linux@linux-VirtualBox:~/Área de Trabalho/projeto\_ifmt\$ kind get clusters

kind

• linux@linux-VirtualBox:~/Área de Trabalho/projeto\_ifmt\$ kubectl get pods

NAME	READY	STATUS	RESTARTS	AGE
minio-75b47687fc-lclhp	1/1	Running	0	63m

• linux@linux-VirtualBox:~/Área de Trabalho/projeto\_ifmt\$ kubectl logs minio-75b47687fc-lclhp

Formatting 1st pool, 1 set(s), 1 drives per set.

WARNING: Host local has more than 0 drives of set. A host failure will result in data becoming unavailable.

MinIO Object Storage Server

Copyright: 2015-2024 MinIO, Inc.

License: GNU AGPLv3 <https://www.gnu.org/licenses/agpl-3.0.html>

Version: RELEASE.2024-03-15T01-07-19Z (go1.21.8 linux/amd64)

API: http://10.244.0.5:9000 http://127.0.0.1:9000

WebUI: http://10.244.0.5:36131 http://127.0.0.1:36131

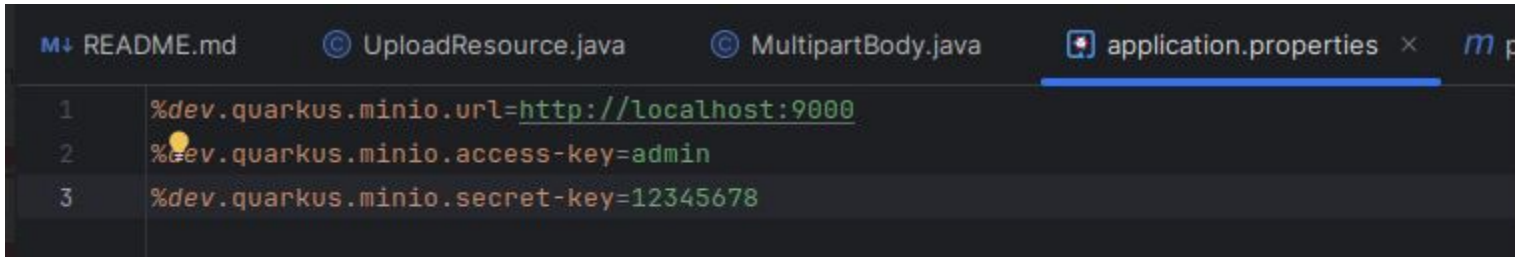
Docs: https://min.io/docs/minio/linux/index.html

Status: 1 Online, 0 Offline.



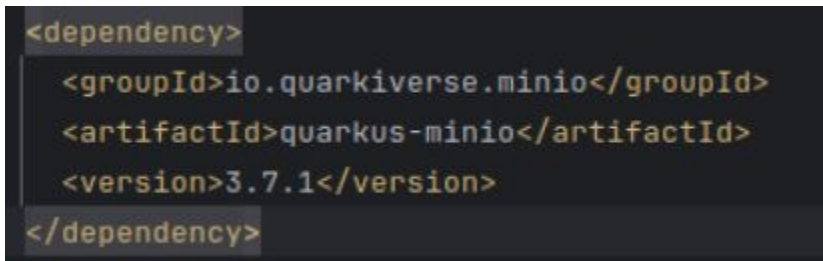
# Aplicação/Configurações:

Application.properties:



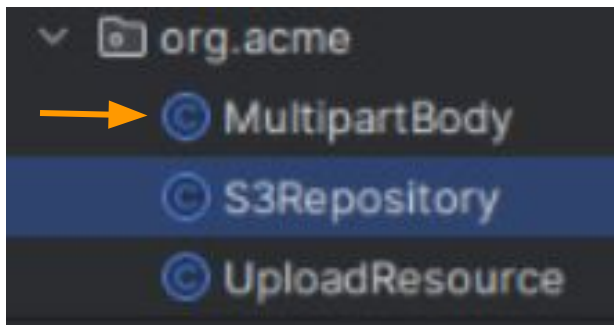
```
1 %dev.quarkus.minio.url=http://localhost:9000
2 %dev.quarkus.minio.access-key=admin
3 %dev.quarkus.minio.secret-key=12345678
```

pom.xml



```
<dependency>
 <groupId>io.quarkiverse.minio</groupId>
 <artifactId>quarkus-minio</artifactId>
 <version>3.7.1</version>
</dependency>
```

# Aplicação/Classes:



```
import java.io.InputStream;

1 usage
public class MultipartBody {

 2 usages
 @FormParam("anexo")
 @PartType(MediaType.APPLICATION_OCTET_STREAM)
 @Schema(type = SchemaType.STRING, format = "binary", description = "file")
 public InputStream anexo;

 2 usages
 @FormParam("fileName")
 @PartType(MediaType.TEXT_PLAIN)
 public String fileName;

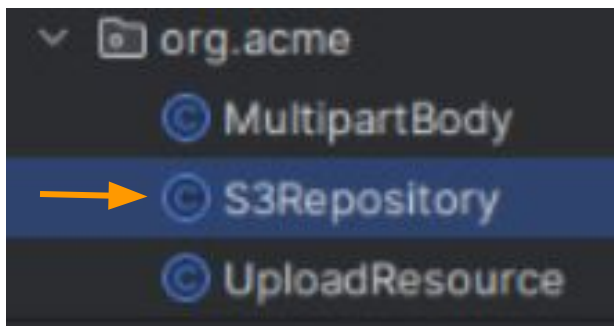
 1 usage
 public InputStream getAnexo() {
 return anexo;
 }

 no usages
 public void setAnexo(InputStream anexo) {
 this.anexo = anexo;
 }

 2 usages
 public String getFileName() {
 return fileName;
 }

 no usages
 public void setFileName(String fileName) {
 this.fileName = fileName;
 }
}
```

# Aplicação/Classes:



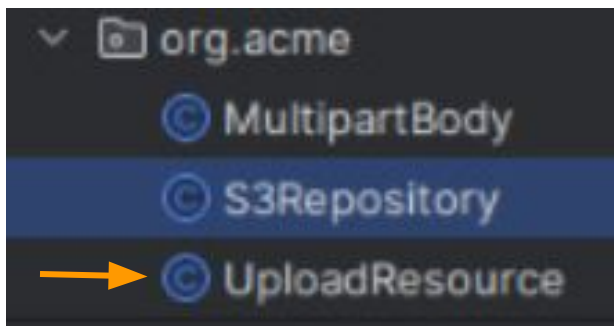
```
package org.acme;

> import ...
1 usage
@ApplicationScoped
public class S3Repository {

 1 usage
 @Inject
 MinioClient minioClient;

 1 usage
 public void enviarArquivo(String bucket, String hash, InputStream file) {
 try {
 minioClient.putObject(PutObjectArgs.builder().bucket(bucket).object(hash).stream(
 file, objectSize: -1, partSize: 10485760)
 .contentType(MediaType.APPLICATION_OCTET_STREAM)
 .build());
 } catch (Exception e) {
 System.out.println("Falha ao enviar o arquivo");
 e.printStackTrace();
 }
 }
}
```

# Aplicação/Classes:



```
21 public class UploadResource {
22
23 // Injetar dependências necessárias (ex: S3 repository)
24 @Inject
25 S3Repository s3Repository;
26
27 // Definir método para upload de arquivo
28 @POST
29 @Consumes(MediaType.MULTIPART_FORM_DATA)
30 @Produces(MediaType.APPLICATION_JSON)
31 public Response uploadFile(@MultipartForm MultipartBody anexo) throws IOException {
32 try {
33 // String fileName = file.getName();
34 // long fileSize = file.getSize();
35
36 // Salvar arquivo no bucket S3
37 s3Repository.enviarArquivo(bucket: "arquivos", anexo.getFileName() != null ? anexo.getFileName() : UUID.randomUUID().toString(), anexo.getFile());
38
39 // Retornar resposta com sucesso
40 return Response.ok().status(201).build();
41 } catch (Exception e) {
42 // Tratar erros e retornar resposta de erro
43 return Response.serverError().build();
44 }
45 }
46 }
```

# Upload de Arquivo

anexo  
string(\$binary)

file

Procurar... 2024\_02\_ANEX 23...77072033(1).p7s

☐ Send empty value

fileName  
string

teste ifmt

☐ Send empty value

Execute

Clear

## Responses

Curl

```
curl -X 'POST' \
 'http://localhost:8080/arquivos' \
 -H 'accept: */*' \
 -H 'Content-Type: multipart/form-data' \
 -F 'anexo=@2024_02_ANEX 2394f9410d98af0bab83e4286312533f10f0d95a3ed4a8b72a387239d241ee00-1709177072033(1).p7s;type=application/pkcs7-signature' \
 -F 'fileName=teste ifmt;type=text/plain'
```

Request URL

http://localhost:8080/arquivos

Server response

Code

Details

201

Undocumented

Response headers

content-length: 0

Responses

Code

Description

Links

# MINIO:

localhost:41795/browser/arquivos

Object Browser

Start typing to filter objects in the bucket



arquivos

Created on: Mon, Mar 18 2024 13:21:14 (GMT-4) Access: PRIVATE 296.8 KiB - 1 Object

Rewind ↶

Refresh ↻

Upload ↗



arquivos



Create new path ↗



Name

Last Modified

Size



e4744e99-24f8-483f-acd1-21a848166cdc

Today, 14:16

296.8 KiB



teste ifmt

Today, 14:18

302.1 KiB

User



Object Browser



Access Keys



Documentation

Administrator



Buckets



Policies



Identity



Monitoring



Events



Tiering



Site Replication



Configuration

Subnet



License



Health



Performance



Profile

---

## Conclusão

A disciplina de Arquitetura de Microserviços e Containers fomentou a curiosidade de explorar novas formas de implementações da tecnologia no mercado. Sendo possível vislumbrar as diversas ferramentas que auxiliam nessa implementação dentre elas o MinIO que torna possível o armazenamento de volume de dados e seu gerenciamento.

---

## Referências

- [Red Hat](#)(Acesso na data 18 Março de 2024)
- [CI9 Tecnologia](#) (Acesso na data 18 Março de 2024)
- [Openlbes](#) (Acesso na data 18 Março de 2024)
- [Medium](#) (Acesso 18 Março de 2024)
- [Joel Garcia Júnior](#) (Acesso 18 Março 2024)
- <https://kind.sigs.k8s.io/>(Acesso 18 Março de 2024)
- [Documentação MINIO](#)(Acesso 18 Março de 2024)