sistemi-cloud-unict-2022

Progetto sviluppato per Sistemi Cloud (Università degli studi di Catania, Informatica Magistrale LM-18 A.A. 2021/2022). Si propone il refactoring ed il deploy su AWS di un progetto a microservizi tra cui l'integrazione le seguenti tecnologie:

- Autenticazione delle API gestita con Keycloak
- GitHub actions per il deploy delle immagini su DockerHub
- Kubernetes (EKS su AWS)

Script utili

Sviluppo

`setup-dev.sh` script bash che clona tutte le repository necessarie per avviare il progetto nella cartella `development`. Builda le immagini dei microservizi, ordina i file di configurazione di keycloak e dei database in modo tale che possano essere utilizzati dal docker compose.

`update-repo.sh` script bash da eseguire dentro la directory `development` che si occuperà di aggiornare all'ultima versione gli script, imports e le immagini di dei microservizi.

Kubernetes

`aws-db.sh` script bash utilizzato per effettuare il deploy dei database su AWS

`aws-be.sh` script bash utilizzato per effettuare il deploy dei microservizi su AWS

`init-db.sh` script bash utilizzato per effettuare il deploy dei database in locale (usando il namespace di `default`)

`init-be.sh` script bash utilizzato per effettuare il deploy dei microservizi in locale (usando il namespace di `default`)

`restore-k8.sh` script bash utilizzato per rimuovere tutte le risorse

Keycloack

Download latest image from docker

```
docker pull jboss/keycloak
```

By default there is no admin user created so you won't be able to login to the admin console. To create an admin account you need to use environment variables to pass in an initial username and password

```
docker run -e KEYCLOAK_USER=<USERNAME> -e KEYCLOAK_PASSWORD=<PASSWORD>
jboss/keycloak
```

Example

```
docker run -p 8180:8080 -e KEYCLOAK_USER=admin -e KEYCLOAK_PASSWORD=admin
jboss/keycloak
```

Andando su `localhost:8180` sarà possibile accedere alla console con username e password `admin`

- Getting started with Keycloack (Procedura per creazione utente, realm e client)
- Setup spring-boot keycloack

Docker

Il `docker-compose.yaml` presente nella directory principale della repository è *pronto all'uso'* in quanto utilizza le immagini presenti nel registry di docker.

In caso di errori frequenti del docker-compose: `docker-compose down --rmi all`

Enter inside docker container mysql

`docker exec -it CONTAINER_ID mysql -u admin -p`

Now it will be possible use `SQL` syntax: like `SHOW DATABASES; USE DB_NAME;`

Effettuare un dump dal database di un container docker

```
docker exec CONTAINER_ID /usr/bin/mysqldump -u admin --password=admin
DATABASE_NAME > backup.sql --no-tablespaces -y
```

Copia il file di backup in una directory locale

```
cp backup.sql DIR_DEST | docker exec -i CONTAINER_ID /usr/bin/mysql -u
admin --password=admin DATABASE
```

Attenzione: se si volesse eseguire l'intero progetto utilizzando le immagini buildate delle repository, è necessario sostituire le immagini del `docker-compose` della folder di `development` con quelle buildate localmente

Kubernetes & Minikube

Create and start the cluster

```
minikube start --driver docker
```

See if everything is running properly

```
minikube status
```

Display all nodes in cluster

```
kubectl get node
```

Apply configuration file (create secret and config first)

```
kubectl apply -f CONFIG_FILE.yaml
```

Get info:

```
kubectl get all

kubectl describe NAME_OF_SERVICE NAME_OF_SPECIFIC_SERVICE
```

examples:

```
kubectl describe service donor-service

kubectl describe pod donor-deployment-756b467d4d-2g7xk
```

Show logs:

```
kubectl logs POD_NAME -f
```

How to access service from browser?

```
kubectl get svc -o wide

minikube ip
```

How to get internal IP:

```
kubectl get node -o wide
```

Get minikube IP and listen to that port

Entrare in bash all'interno di un pod DB mysql:

```
kubectl exec --stdin --tty POD_NAME -- /bin/bash
```

`mysql -p` com password `root`

Get basic info about k8s components

```
kubectl get node
kubectl get pod
kubectl get svc
kubectl get all
```

Get extended info about components

```
kubectl get pod -o wide
kubectl get node -o wide
```

Get detailed info about a specific component

```
kubectl describe svc {svc-name}
kubectl describe pod {pod-name}
```

get application logs

```
kubectl logs {pod-name}
```

stop your Minikube cluster

```
minikube stop
```

> ⚠️ **Known issue - Minikube IP not accessible**

If you can't access the NodePort service webapp with `MinikubeIP:NodePort`, execute the following command:

```
minikube service pod-service
```

Tips: shortcuts!

```
alias k8='kubectl'
alias k8-all='kubectl get all'
alias k8-pod='kubectl get pods -o wide'
alias k8-svc='kubectl get svc -o wide'
alias k8-get-cnt='kubectl config get-contexts'
alias k8-log='kubectl logs'
alias aws-login='aws ecr get-login-password --region <REGION> | docker
login --username AWS --password-stdin <ID_AWS>.dkr.ecr.
<REGION>.amazonaws.com'
```

AWS

Login aws-cli

```
aws ecr get-login-password --region <REGION> | docker login --username AWS
--password-stdin <AWS_ID_ACCOUNT>.dkr.ecr.<REGION>.amazonaws.com
```

Creazione di un cluster

```
eksctl create cluster --name <CLUSTER-NAME> --version <VERSION-K8S> --
region <REGION> --nodegroup-name <NODEGROUP_NAME> --node-type <NODE-TYPE>
--nodes <N_NODES>
```

Example:

```
eksctl create cluster --name biobank-sprec-cluster --version 1.22 --region
us-east-1 --nodegroup-name linux-nodes --node-type t2.xlarge --nodes 4
```

Delete a cluster

```
eksctl delete nodegroup --cluster <CLUSTER_NAME> --region <CODE_REGION> --
name linux-nodes
```

Visualizzo i cluster disponibili

```
aws eks list-clusters
```

Create kubeconfig with aws command

```
aws eks --region <REGION> update-kubeconfig --name <CLUSTER_NAME>
```

Creo un namespace per ogni tipo di servizio:

```
kubectl create namespace <NAMESPACE_NAME>
```

Get current context:

```
kubectl config current-context
```

View all vailable context:

```
kubectl config get-contexts
```

Switch context:

```
kubectl config use-context <CONTEXT_NAME>
```

Creazione repository dove pushare le immagini (Facoltativo)

```
aws ecr create-repository \
    --repository-name biobank-sprec \
    --image-scanning-configuration scanOnPush=true \
    --region us-east-1
```

- Getting started AWS
- EKS cluster connection