

Block 4: Container Orchestration

Auftrag 4.1: Container-Orchestrierung

1. Warum braucht man Container-Orchestrierung? Container-Orchestrierung wird benötigt, um komplexe Anwendungen mit vielen Containern zu verwalten, zu skalieren, zu überwachen und hochverfügbar zu machen.
 2. Wie funktioniert Container-Orchestrierung? Bei der Container-Orchestrierung werden Tools und Plattformen verwendet, um die Bereitstellung, Verwaltung und Kommunikation von Containern zu automatisieren und zu koordinieren.
 3. Welche Container-Orchestrierung Technologien kennen Sie? Einige bekannte Container-Orchestrierungstechnologien sind Docker Swarm, Kubernetes, Apache Mesos und Amazon ECS (Elastic Container Service).
 4. Was versteht man unter "Scaling Containers"? "Scaling Containers" bezieht sich auf die Fähigkeit, die Anzahl der Containerinstanzen, die eine Anwendung ausführen, dynamisch zu erhöhen oder zu verringern, um die Anforderungen des Workloads zu erfüllen.
 5. Was gibt es für Deployment-Strategien? Es gibt verschiedene Deployment-Strategien wie Rolling Update, Blue-Green Deployment, Canary Deployment und A/B-Testing, die es ermöglichen, Anwendungen schrittweise zu aktualisieren oder neue Versionen bereitzustellen, um Ausfallzeiten zu minimieren und einen reibungslosen Übergang sicherzustellen.
-

Kubernetes - minikube - Was ist das?

minikube ist eine lokale Kubernetes Installation mit dem Ziel, einfach und schnell Kubernetes zu lernen, und simple Kubernetes Umgebungen zu entwickeln und auszutesten.

Systemvoraussetzungen: 2 CPUs oder mehr Mindestens 2GB RAM Mindestens 20GB Disk Space
Internet Verbindung Container oder virtuelle Umgebung: Docker, Hyperkit, Hyper-V, KVM, Parallels, Podman, VirtualBox, oder VMware Fusion/Workstation

Installation minikube

Dependencies

```
sudo apt install curl wget apt-transport-https -y
```

Download und Installation minikube

```
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64  
sudo install minikube-linux-amd64 /usr/local/bin/minikube
```

Download und Installation kubectl

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"  
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
```

Ein paar minikube Addons

```
minikube start --driver=docker  
minikube addons enable ingress  
minikube addons enable dashboard  
minikube addons enable metrics-server
```

Reboot Linux

```
sudo reboot
```

Start minikube und Dashboard

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

```
minikube dashboard --url
```

Um das Dashboard von einem anderen Computer zu öffnen kann man den folgenden Befehl benutzen um die Ports zu Mappen:

```
ssh -L <Port>:127.0.0.1:<Port> <User>@<IP>
```

Auftrag 4.2 Container Orchestration mit Docker Compose

1. Folgen Sie dem Tutorial <https://docs.docker.com/compose/gettingstarted/>
2. Erstellen Sie ein Diagramm und beschreiben Sie die einzelnen verwendeten Container
3. Beantworten Sie folgende Fragen
 1. Was ist Redis? Redis ist eine Open-Source-Datenbank, die als Schlüssel-Wert-Speicher dient und schnellen Zugriff auf Daten ermöglicht. Es wird häufig als Zwischenspeicher oder Caching-Mechanismus eingesetzt.
 2. Welche Ports werden genutzt? Redis verwendet standardmäßig den Port 6379 für die Kommunikation zwischen Client und Server. Dieser Port wird in der Regel für den Zugriff auf Redis-Dienste genutzt.
 3. Was ist die Bedeutung von ENV im DOCKERFILE? ENV (Environment) in einem Dockerfile ermöglicht das Setzen von Umgebungsvariablen innerhalb des Container-Images. Diese Variablen können während der Laufzeit verwendet werden, um bestimmte Konfigurationswerte oder Parameter anzupassen, wie z.B. Datenbankverbindungszeichenfolgen oder API-Schlüssel.

Auftrag 4.3: minikube - Kubernetes ganz einfach (Crash Kurs)

Die installierte Kubernetes Umgebung muss nach jedem Reboot/Neustart des Systems mit den folgenden Befehlen gestartet werden.

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

```
minikube dashboard --url
```

- Erstellen Sie in ihrem privaten Git Account ein Repository mit dem Namen dev_minikube
- Erstellen Sie im Homeverzeichnis des Benutzers ein Verzeichnis dev_minikube
- Initialisieren Sie ihr Git Repository in diesem Verzeichnis, erstellen die 4 benötigten Files (Git Repository der Konfigurationsfiles) für Deployment und Services und stellen Sie sicher, dass diese Konfigurationsfiles in ihrem Git Account auf github.com gespeichert sind.

Sind die Konfigurationsfiles einmal erstellt, gestaltet sich das eigentliche Deployment sichtlich einfach:

```
kubectl apply -f mongo-config.yaml
kubectl apply -f mongo-secret.yaml
kubectl apply -f mongo.yaml
kubectl apply -f webapp.yaml
```

Das wars auch schon. Jetzt müssen Sie nur noch herausfinden, über welchen URL bzw. IP Adresse und Port auf Ihren neu deployten Service zugegriffen werden kann.

IP Adresse des minikube Nodes herausfinden:

```
minikube ip
```

Port des Webapp Services (Nodeport) herausfinden:

```
kubectl describe service webapp
```

URL wäre dann zusammengesetzt: http://192.168.49.2:30100

Falls die Webseite nicht erreichbar ist, kann der folgende Befehl ausgeführt werden: `

```
minikube service webapp-service
```

Revision #1

Created 15 December 2023 14:11:57 by Manuel Regli

Updated 15 December 2023 14:12:38 by Manuel Regli