

知能情報実験実習 1 手順書

前提条件: バージョン Linux12

実行環境 raspberry Pi OS

利用ツール Docker, Minikube

1.目的と完成条件の定義

目的: Docker Compose を用いて Web+データベース(DB)の 2 コンテナ構成を実装する。

完成条件: DB コンテナのみを停止させたときにデータベース接続確立エラーが発生し、復活させたときに元のブログ画面が表示される

2.構築手順

- Docker をインストールする。

```
curl -fsSL https://get.docker.com -o get-docker.sh
# インストールスクリプトをダウンロード

sudo sh get-docker.sh
# ダウンロードしたインストールスクリプトの実行
```

- kubectl のインストール

```
curl -LO "https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/arm64/kubectl"
# 最新版のダウンロード
sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
# インストール
```

- スワップを停止させる

```
sudo swapoff -a
#一時的にスワップを停止

sudo sed -i '/swap/d' /etc/fstab
#再起動後もスワップを無効化させる
```

- Minikube のインストールと起動

```
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-  
linux-arm64  
# Minikube のダウンロード  
  
sudo install minikube-linux-arm64 /usr/local/bin/minikube  
# Minikube のインストール  
  
minikube start --driver=docker  
# Docker ドライバを使用して Minikube を起動
```

- Apache デプロイメントの作成と管理
マニフェストファイルとして、yaml ファイルを作成し、apache-custom.yaml という名前で保存する。内容は以下の通り

```
# -----  
# 1. DB のサービス（内部通信用）  
# -----  
apiVersion: v1  
kind: Service  
metadata:  
name: wordpress-mysql  
spec:  
ports:  
- port: 3306  
selector:  
app: wordpress  
tier: mysql  
clusterIP: None  
---  
# -----  
# 2. DB のデプロイメント（本体）  
# -----  
apiVersion: apps/v1  
kind: Deployment  
metadata:  
name: wordpress-mysql
```

```
spec:
selector:
matchLabels:
app: wordpress
tier: mysql
template:
metadata:
labels:
app: wordpress
tier: mysql
spec:
containers:
- name: mysql
image: mariadb:10.6
env:
- name: MYSQL_ROOT_PASSWORD
value: secret_password
- name: MYSQL_DATABASE
value: wordpress_db
- name: MYSQL_USER
value: wp_user
- name: MYSQL_PASSWORD
value: wp_password
---
# -----
# 3. Web のサービス（外部公開用）
# -----
apiVersion: v1
kind: Service
metadata:
name: wordpress-web
spec:
type: NodePort
ports:
- port: 80
targetPort: 80      # ここが ratgetPort になっていました
```

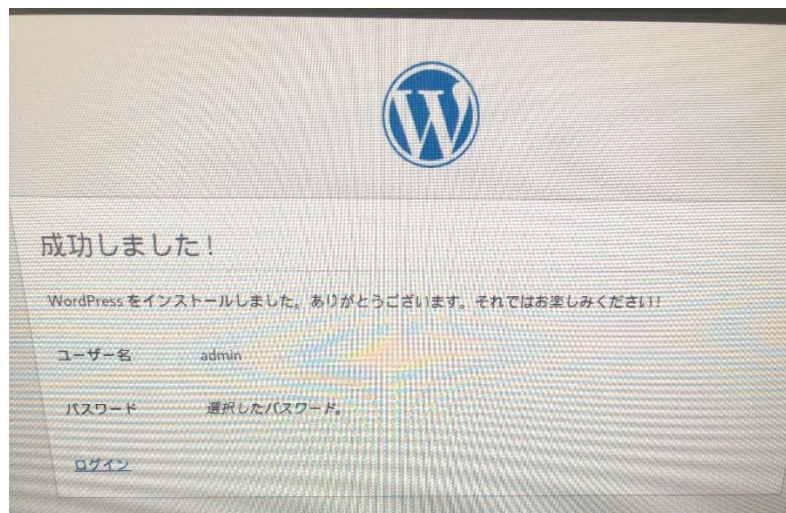
```
nodePort: 30080      # 外部からアクセスするポート番号
selector:
app: wordpress
tier: frontend
---
# -----
# 4. Web のデプロイメント（本体）
# -----
apiVersion: apps/v1
kind: Deployment
metadata:
name: wordpress-web
spec:
replicas: 1
selector:
matchLabels:
app: wordpress
tier: frontend
template:
metadata:
labels:
app: wordpress
tier: frontend
spec:
containers:
- name: wordpress
image: wordpress:latest
env:
- name: WORDPRESS_DB_HOST
value: wordpress-mysql
- name: WORDPRESS_DB_USER
value: wp_user
- name: WORDPRESS_DB_PASSWORD
value: wp_password
- name: WORDPRESS_DB_NAME
value: wordpress_db
```

```
ports:  
- containerPort: 80
```

- ・ yaml ファイルの適用

```
kubectl apply -f apache-custom.yaml  
# yaml ファイルの内容をクラスタに適用する
```

- ・ 公開したサービス (WordPress) でコンテンツを作成する
言語選択画面が表示されるので、日本語を選択後にユーザー名とパスワード、メールアドレスを入力する。



- ・ サービス内での記事作成
サービス内で Kubernetes 接続テストという記事を作成する。



- ・ DB コンテナのみを停止させる

```
kubectl scale deployment wordpress-mysql --replicas=0
# DB の数を 0 にする
```

- ・リロードする
リロードしてデータベース接続確認エラーが発生することを確認する。
- ・DB を復活させる

```
kubectl scale deployment wordpress-mysql --replicas=1
# DB の数を 1 にする
```

- ・リロードする
リロードして、元のブログ画面が表示されることを確かめる。
- ### 3.トラブルシューティング
- 公開したサービスのリンクに行っても画面が表示されない
 - ・サービスを表示するためにはサービスのリンクのうち、:以下の数字も入力する必要があります。
 - アプリカスタマイズの際、エラーが発生する
 - ・Kubectl create configmap において、文章中で半角の!が入っているとエラーが発生することを確認しています。 !は全角で入力してください。
- ### 4.参考資料:
- ・-bash とは:!:イベントが見つかりません
ja.unixlinux.online/ex/1007045565.html
 - ・Kubernetes クラスタを構築する方法
https://zenn.dev/programing_gym/articles/faa0ff78caa19b
 - ・jemini を使用