

CENG 495 - Cloud Computing

2023 - 2

HW - 2

Ünlü, Berke Can
e2381028@ceng.metu.edu.tr

May 21, 2023

1 Installation

I've installed minikube and scaffold with homebrew in macOS. I did not face with a problem during installation.

I've used docker as driver for minikube. Docker desktop has already been installed in my machine.

I've decided to use [this project](#)

2 First Deployment

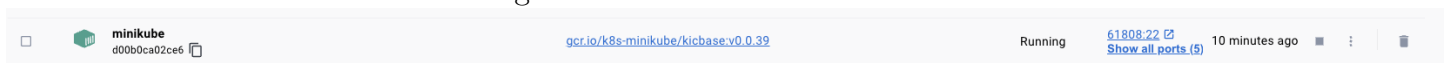
2.1 Minikube Start

I've run minikube start command.

Figure 1: Minikube Start

```
berkecanunlu@f131 microservice-kubernetes-demo % minikube start
minikube v1.30.1 on Darwin 13.3 (arm64)
Using the docker driver based on user configuration
Using Docker Desktop driver with root privileges
Starting control plane node minikube in cluster minikube
Pulling base image ...
Creating docker container (CPUs=4, Memory=6000MB) ...
Preparing Kubernetes v1.26.3 on Docker 23.0.2 ...
  Generating certificates and keys ...
  Booting up control plane ...
  Configuring RBAC rules ...
  Configuring bridge CNI (Container Networking Interface) ...
  Using image gcr.io/k8s-minikube/storage-provisioner:v5
Verifying Kubernetes components...
Enabled addons: default-storageclass, storage-provisioner
Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
berkecanunlu@f131 microservice-kubernetes-demo %
```

Figure 2: Docker view of Minikube



2.2 Skaffold init/dev

Then, I've run skaffold init to create "skaffold.yaml" file to run skaffold dev command. I've selected docker files inside the repository as builders in the first question. I've selected pom.xml files to create kubernetes resources. I've faced with an error as shown in below when I've run skaffold dev after skaffold init.

Figure 3: Skaffold init

```
Configuration skaffold.yaml was written
You can now run [skaffold build] to build the artifacts
or [skaffold run] to build and deploy
or [skaffold dev] to enter development mode, with auto-redeploy
berkecanunlu@f131 microservice-kubernetes-demo % skaffold dev
Generating tags...
- docker.io/ewolff/microservice-kubernetes-demo-apache -> docker.io/ewolff/microservice-kubernetes-demo-apache:db3af1b
- docker.io/ewolff/microservice-kubernetes-demo-catalog -> docker.io/ewolff/microservice-kubernetes-demo-catalog:db3af1b
- docker.io/ewolff/microservice-kubernetes-demo-customer -> docker.io/ewolff/microservice-kubernetes-demo-customer:db3af1b
- docker.io/ewolff/microservice-kubernetes-demo-order -> docker.io/ewolff/microservice-kubernetes-demo-order:db3af1b
Checking cache...
- docker.io/ewolff/microservice-kubernetes-demo-apache: Not found, Building
- docker.io/ewolff/microservice-kubernetes-demo-catalog: Error checking cache.
Cleaning up...
- No resources found
getting hash for artifact "docker.io/ewolff/microservice-kubernetes-demo-catalog": getting dependencies for "docker.io/ewolff/microservice-kubernetes-demo-catalog": file pattern [target/microservice-kubernetes-demo-catalog-0.0.1-SNAPSHOT.jar] must match at least one file
berkecanunlu@f131 microservice-kubernetes-demo %
```

The example is implemented in Java. Also, dockerfiles get JAR files so I had to run "mvnw clean package" command to build the project. It is stated in the readme file of the repository.

Figure 4: mvnw clean package

```
2023-05-17 01:07:29.431 INFO 0:133 --- [text@downloadmode] com.zaxxer.hikari.HikariDataSource : HikariPool-1 <- Shutdown completed.
[INFO] Results:
[INFO]
[INFO] Tests run: 0, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-jar-plugin:3.2.0:jar (default-jar) @ microservice-kubernetes-demo-order ---
[INFO] Building jar: /Users/berkecanunlu/Desktop/4.2/495/microservice-kubernetes/microservice-kubernetes-demo/microservice-kubernetes-demo-order/target/microservice-kubernetes-demo-order-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.3.4.RELEASE:repackage (repackage) @ microservice-kubernetes-demo-order ---
[INFO] Replacing main artifact with repackaged archive
[INFO] Reactor Summary:
[INFO]
[INFO] microservice-kubernetes-demo ..... SUCCESS [ 11.763 s]
[INFO] microservice-kubernetes-demo-customer ..... SUCCESS [ 53.764 s]
[INFO] microservice-kubernetes-demo-catalog ..... SUCCESS [ 6.421 s]
[INFO] microservice-kubernetes-demo-order ..... SUCCESS [ 6.934 s]
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 01:23 min
[INFO] Finished at: 2023-05-17T01:07:29+03:00
[INFO] Final Memory: 55M/214M
[INFO]
[INFO]
```

Then, I've reinitialized skaffold.yaml and selected builders. After that, I've run skaffold dev and microservices become available.

Figure 5: Skaffold init

```
berkecanunlu@f131 microservice-kubernetes-demo % skaffold init
? Choose the builder to build image docker.io/ewolff/microservice-kubernetes-demo-apache Buildpacks (pom.xml)
? Choose the builder to build image docker.io/ewolff/microservice-kubernetes-demo-catalog Buildpacks (microservice-kubernetes-demo-catalog/pom.xml)
? Choose the builder to build image docker.io/ewolff/microservice-kubernetes-demo-customer Buildpacks (microservice-kubernetes-demo-customer/pom.xml)
? Choose the builder to build image docker.io/ewolff/microservice-kubernetes-demo-order Buildpacks (microservice-kubernetes-demo-order/pom.xml)
? Which builders would you like to create kubernetes resources for? Docker (apache/Dockerfile), Docker (microservice-kubernetes-demo-catalog/Dockerfile), Docker (microservice-kubernetes-demo-customer/Dockerfile), Docker (microservice-kubernetes-demo-order/Dockerfile)
apiVersion: skaffold/v4beta5
kind: Config
metadata:
  name: microservice-kubernetes-demo
build:
  artifacts:
    - image: docker.io/ewolff/microservice-kubernetes-demo-apache
      buildpacks:
        builder: gcr.io/buildpacks/builder:v1
    - image: docker.io/ewolff/microservice-kubernetes-demo-catalog
      context: microservice-kubernetes-demo-catalog
      buildpacks:
        builder: gcr.io/buildpacks/builder:v1
    - image: docker.io/ewolff/microservice-kubernetes-demo-customer
      context: microservice-kubernetes-demo-customer
      buildpacks:
        builder: gcr.io/buildpacks/builder:v1
    - image: docker.io/ewolff/microservice-kubernetes-demo-order
      context: microservice-kubernetes-demo-order
      buildpacks:
        builder: gcr.io/buildpacks/builder:v1
manifests:
  rawYaml:
    - microservices.yaml

? Do you want to write this configuration to skaffold.yaml? Yes
Configuration skaffold.yaml was written
You can now run [skaffold build] to build the artifacts
or [skaffold run] to build and deploy
or [skaffold dev] to enter development mode, with auto-redeploy
berkecanunlu@f131 microservice-kubernetes-demo %
```

Figure 6: Skaffold dev builds

```
berkecanunlu@f131 microservice-kubernetes-demo % skaffold dev
Generating tags...
- docker.io/ewolff/microservice-kubernetes-demo-apache -> docker.io/ewolff/microservice-kubernetes-demo-apache:db3af1b-dirty
- docker.io/ewolff/microservice-kubernetes-demo-catalog -> docker.io/ewolff/microservice-kubernetes-demo-catalog:db3af1b-dirty
- docker.io/ewolff/microservice-kubernetes-demo-customer -> docker.io/ewolff/microservice-kubernetes-demo-customer:db3af1b-dirty
- docker.io/ewolff/microservice-kubernetes-demo-order -> docker.io/ewolff/microservice-kubernetes-demo-order:db3af1b-dirty
Checking cache...
- docker.io/ewolff/microservice-kubernetes-demo-apache: Not found. Building
- docker.io/ewolff/microservice-kubernetes-demo-catalog: Not found. Building
- docker.io/ewolff/microservice-kubernetes-demo-customer: Not found. Building
- docker.io/ewolff/microservice-kubernetes-demo-order: Not found. Building
Starting build...
Found [minikube] context, using local docker daemon.
Building [docker.io/ewolff/microservice-kubernetes-demo-order]...
Target platforms: [linux/arm64]
```

Figure 7: Project is ready

```
Starting deploy...
- deployment.apps/apache created
- deployment.apps/catalog created
- deployment.apps/customer created
- deployment.apps/order created
- service/apache created
- service/catalog created
- service/customer created
- service/order created
Waiting for deployments to stabilize...
- deployment/apache is ready. [3/4 deployment(s) still pending]
- deployment/order is ready. [2/4 deployment(s) still pending]
- deployment/catalog is ready. [1/4 deployment(s) still pending]
- deployment/customer is ready.
Deployments stabilized in 3.094 seconds
Listing files to watch...
- docker.io/ewolff/microservice-kubernetes-demo-apache
- docker.io/ewolff/microservice-kubernetes-demo-catalog
- docker.io/ewolff/microservice-kubernetes-demo-customer
- docker.io/ewolff/microservice-kubernetes-demo-order
Press Ctrl+C to exit
Watching for changes...
[apache] AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 10.244.0.4. Set the 'ServerName' directive globally to suppress this message
[catalog]
[catalog]      . ----          -       ---
[order]
[catalog] /\ / ___'_ _ _ _ _ _ _ _ _ _ \\\
[customer]
[customer]   .
[customer]  /\_/_/_/_/_/_/_/_/_/_/_/_/_/_/_\
[catalog] ( ()_/_/_/_/_/_/_/_/_/_/_/_/_/_/_\
[customer] ( ()_/_/_/_/_/_/_/_/_/_/_/_/_/__\
[customer] w _ _ | | | | | | | | | | ) ) ) )
[order]
[customer]   .
[order]  /\_/_/_/_/_/_/_/_/_/_/_/_/_/_/_\
[catalog] w _ _ | | | | | | | | | | ) ) ) )
[customer] ==|_|_|_|_|_|_|_|_|_|_|_|_|_|_|/
[order] ( ()_/_/_/_/_/_/_/_/_/_/_/_/_/__\
[catalog]   .
[catalog]  /\_/_/_/_/_/_/_/_/_/_/_/_/_/__\
[catalog] ==|_|_|_|_|_|_|_|_|_|_|_|_|_|_|/
```

When I tried to access localhost:8080 nothing came to the web browser. Then, I've followed the **readme of the repository again**. It is stated that "Run **kubectl port-forward deployment/apache 8081:80** to create a proxy to the Apache httpd server on your local machine. Then open `http://localhost:8081` to see the web page of the Apache httpd server in the web browser."

Figure 8: Project is ready

```
berkecanunlu@f131 microservice-kubernetes-demo % kubectl port-forward deployment/apache 8081:80
Forwarding from 127.0.0.1:8081 -> 80
Forwarding from [::1]:8081 -> 80
Handling connection for 8081
Handling connection for 8081
Handling connection for 8081
Handling connection for 8081
```

When I did this, localhost:8081 become available and I saw the web page.

Figure 9: Skaffold init

Order Processing

Customer
Catalog
Catalog
Order

List / add / remove customers
List / add / remove items
Search Items
Create an order

2.3 kubectl get

Figure 10: Services running

```
berkecanunlu@f131 microservice-kubernetes-demo % kubectl get services
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
apache	LoadBalancer	10.96.14.74	<pending>	80:30969/TCP	4m45s
catalog	LoadBalancer	10.103.162.69	<pending>	8080:31578/TCP	4m45s
customer	LoadBalancer	10.105.82.3	<pending>	8080:31106/TCP	4m45s
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	19m
order	LoadBalancer	10.103.226.204	<pending>	8080:30740/TCP	4m45s

Figure 11: Pods

```
berkecanunlu@f131 microservice-kubernetes-demo % kubectl get pods
```

NAME	READY	STATUS	RESTARTS	AGE
apache-7674df88f7-krwlf	1/1	Running	0	3m41s
catalog-6695d6648c-j6sjn	1/1	Running	0	3m41s
customer-79f9b765f5-wvqz4	1/1	Running	0	3m41s
order-685f575955-nxjrl	1/1	Running	0	3m41s

```
berkecanunlu@f131 microservice-kubernetes-demo %
```

Figure 12: Deployment

```
berkecanunlu@f131 microservice-kubernetes-demo % kubectl get deployments
```

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
apache	1/1	1	1	5m10s
catalog	1/1	1	1	5m10s
customer	1/1	1	1	5m10s
order	1/1	1	1	5m10s

3 Configuring Frontend

I’ve changed index.html with bootstrap.

Figure 13: Index after changes

Order Processing

[Customer](#)
List / add / remove customers

[Catalog](#)
List / add / remove items

[Order](#)
Create an order

Figure 14: kubectl get after index change

```

● berkecanunlu@f131 microservice-kubernetes-demo % kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
apache-868bdb997-6zt2s              1/1     Running   0           97s
catalog-67c7c758c-274pb             1/1     Running   0           37m
customer-687d496bf6-bsrcg           1/1     Running   0           37m
order-58c79ffb4b-mrxc5              1/1     Running   0           37m
● berkecanunlu@f131 microservice-kubernetes-demo % kubectl get services
NAME          TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
apache        LoadBalancer 10.97.120.160  <pending>      80:31623/TCP     38m
catalog        LoadBalancer 10.103.227.228 <pending>      8080:30288/TCP   38m
customer       LoadBalancer 10.111.116.97  <pending>      8080:31164/TCP   38m
kubernetes    ClusterIP      10.96.0.1      <none>         443/TCP          49m
order         LoadBalancer 10.104.44.29   <pending>      8080:31611/TCP   38m
● berkecanunlu@f131 microservice-kubernetes-demo % kubectl get endpoints
NAME          ENDPOINTS          AGE
apache        10.244.0.8:80      38m
catalog        10.244.0.5:8080    38m
customer       10.244.0.7:8080    38m
kubernetes    192.168.49.2:8443  49m
order         10.244.0.6:8080    38m
● berkecanunlu@f131 microservice-kubernetes-demo % kubectl get deployments
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
apache        1/1     1             1           38m
catalog        1/1     1             1           38m
customer       1/1     1             1           38m
order         1/1     1             1           38m

```

When I tried to change the customer/catalog/order pages, I've noticed that changes cannot be applied. When I looked into Dockerfile of these services, it copies JAR file. Then, I've run `./mvnw clean package` to build the project again. When I run it, skaffold built the images again and changes are applied.

Also, whenever a change is occurred in the containers, port forwarding becomes broken so I had to run the `"kubectl port-forward deployment/apache 8081:80"` command again.

Figure 15: Customer page changes

[Home](#) [List](#) [Form](#)

Customer View All

id	Name	Firstname	
1	Wolff	Eberhard	delete
2	Johnson	Rod	delete

[Add Customer](#)

Figure 16: Order page changes

[Home](#) [List](#)

Order View All

ID	Customer	Total Price	
No orders			

[Add Order](#)

Figure 17: Item page changes

[Home](#) [List](#) [Search](#) [Form](#)

Item View All

id	Name	Price	
1	iPod	42.0	delete
2	iPod touch	21.0	delete
3	iPod nano	1.0	delete
4	Apple TV	100.0	delete

[Add Item](#)

Figure 18: kubectl get after other pages changes'

```

● berkecanunlu@f131 microservice-kubernetes-demo % kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
apache-786d7d46f9-zthbc             1/1     Running   0           21s
catalog-694b7f6597-vcvcr            1/1     Running   0           21s
customer-658cdb9c7c-7rtqp           1/1     Running   0           21s
order-77758b98cf-2rgt7              1/1     Running   0           21s
● berkecanunlu@f131 microservice-kubernetes-demo % kubectl get services
NAME      TYPE          CLUSTER-IP    EXTERNAL-IP   PORT(S)          AGE
apache    LoadBalancer 10.110.175.249 <pending>     80:30663/TCP     25s
catalog   LoadBalancer 10.109.60.50   <pending>     8080:32531/TCP   25s
customer  LoadBalancer 10.111.74.239  <pending>     8080:30867/TCP   25s
kubernetes ClusterIP      10.96.0.1     <none>        443/TCP         55m
order     LoadBalancer 10.103.212.131 <pending>     8080:32471/TCP   25s
● berkecanunlu@f131 microservice-kubernetes-demo % kubectl get endpoints
NAME      ENDPOINTS          AGE
apache    10.244.0.9:80      31s
catalog    10.244.0.10:8080   31s
customer   10.244.0.12:8080   31s
kubernetes 192.168.49.2:8443  55m
order      10.244.0.11:8080   31s
● berkecanunlu@f131 microservice-kubernetes-demo % kubectl get deployments
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
apache    1/1     1             1           40s
catalog    1/1     1             1           40s
customer   1/1     1             1           40s
order     1/1     1             1           40s
○ berkecanunlu@f131 microservice-kubernetes-demo % 

```