Discover Workshop: Modern Cloud Apps with Micro Service architecture on Azure with Immeo: Lab help guide

This guide is intended to be used for speeding up the process of reaching the lab parts related to Kubernetes and to provide help during the lab if there are challenges.

Lab setup

Start the lab using the provided invite.
Follow the Lab setup guide (separate document) if you have not already completed it.
If you have completed the Lab setup guide you can skip the “Before lab” section and go to Exercise 1.

Exercise 1

ssh to build agent either from Cloud shell (https://shell.azure.com) or from VS Code using the remote ssh extension (see Lab setup guide, “Install VS Code and Remote - SSH extension”).

Open portal and note down name of VM resource, AKS cluster and container registry.
[SUFFIX] is what comes after fabmedical-[SUFFIX]
[SHORT_SUFFIX] is fabmedical or sometimes equal to [SUFFIX] e.g. in content-web.yml

Skip Task 1 and 2. These steps cover retrieving mongodb docker image and running api and web on the build agent.

Start the mongo container:
docker network create fabmedical
docker container run --name mongo --net fabmedical -p 27017:27017 -d mongo
docker container list
docker container logs mongo

Initialize mongo db
cd ~/Fabmedical/content-init
npm install
nodejs server.js

Task 3: Create Dockerfile

Task 4: Create Docker images
Skip step 4 (if your git repository is based on the Immeo source repository)

Task 5: Run a containerized application

Task 6: Setup environment variables
[BUILDAGENTIP] is the public ip of the VM. Go to https://portal.azure.com and find the fabmedical VM under Virtual machines and note down the ip.
Skip step 14 (see Lab setup guide, “Install VS Code and Remote - SSH extension”)

Task 7: Run several containers with Docker compose

Task 8: Push images to Azure Container Registry
Skip step 7 (if your git repository is based on the Immeo source repository)

Task 9: Setup CI Pipeline to Push Images

Exercise 2

Task 1: Tunnel into the Azure Kubernetes Service cluster
Step 1-3

Task 2: Deploy a service using the Kubernetes management dashboard
Step 7-11
Create new cosmosdb-secret.yml file

- code cosmosdb-secret.yml
- Paste in content from step 12 with base64 encoded value from 11 inserted.
- kubectl create –f cosmosdb-secret.yml
- Create deployment


api.deployment.yml

- Create service


api.service.yml

- kubectl create --save-config=true -f api.deployment.yml -f api.service.yml
- kubectl get services
- kubectl get deployments

Task 4: Deploy a service using a Helm chart
Open a new Azure Cloud Shell console
Step 6-23
kubectl get services
Note down the public ip of the web app
Open a browser and enter http://[web-app-ip]
Step 25
Step 26 (if your git repository is based on the Immeo source repository)

Task 5: Initialize database with a Kubernetes Job

Step 1-5
Step 7 by running

- kubectl describe jobs/init
- kubectl logs jobs/init

Step 8

Go to http://[web-app-ip] and navigate to Speakers and Sessions.
Confirm that data is now shown.

Task 7: Configure Continuous Delivery to the Kubernetes Cluster
Task 8: Review Azure Monitor for Containers

Exercise 3: Scale the application and test HA

To open Kubernetes dashboard perform Exercise 2 step 6+7.
az aks browse --name fabmedical-226960 --resource-group fabmedical-226960

Alternative is to use kubectl
Useful kubectl commands
Scale: kubectl scale --replicas=3 deployment/api
List replica sets: kubectl get rs
List deployments: kubectl get deployments
Edit deployment: kubctl edit deployment api
List services: kubectl get services
List pods: kubectl get pods
Delete pod: kubectl pod <pod name>

Exercise 4: Working with services and routing application traffic