Process

1) Created a simple Spring Boot application with PostgreSQL Database

2) Package the application to a jar file using ‘mvn clean package -DskipTests ‘

3) Created the DockerFile with the details of java sdk and simple copy commands to create a docker image ‘timestamp-api’

Command : docker build -t timestamp-api .

4) Tag the image as per the standards. For example v1

Command : docker tag <ImageID> <TargetACR/TargetRepo:v1>

Help: ImageID can be obtained by running the command "docker image ls" - which lists the images on the local machine

Target ACR is the URL of the target azure container registry where the image is intended to be uploaded

Target Repo is the repository where the image is intended to be uploaded in the Target ACR

5) Login to Azure

Command : az login (uses the Azure CLI)

6) Login to the Target ACR

Command : az acr login -n <Target ACR>

7) Push the WebAPI-Image to ACR

Command : docker image push <TargetACR/TargetRepo:v1>

8) Use the file "webapi-deployment.yml" to create the webAPI container

Command : kubectl apply -f "webapi-deployment.yml"

9) Use the file "db-configmap.yml" to create a configuration map for the PostgeSQL DB that needs to be created. This file contains the environment variables for Database name, username, password

Command : kubectl apply -f "db-configmap.yml"

10) Use the file "webapi-svc.yml" and "db-deployment.yml" to create the webapi and DB as services that can be accessed.

Command : kubectl apply -f "webapi-service.yml"

kubectl apply -f "db-deployment.yml"

11) Get a list of all the services that are currently running on the server

Command : kubectl get service

12) From the output of #11, find the External IP (created because of deploying the web api with Load Balancer) to access the API

13) Make a POST call to the URL “http://ExternalIP:<Port>/api/timestamp with an empty body to make an entry of the current timestamp to the database

14) Make a GET call to the URL [http://ExternalIP:<Port>/api/all](http://ExternalIP:%3cPort%3e/api/all) to list all the entries that exist in the database

Note: <Port> has been defined as 8002 for this application