Student Name: Nguyen Tuan Hai

Student ID: 20161311

LABWORK
COURSE: DISTRIBUTED SYSTEMS
CHAPTER 2: ARCHITECTURES

Question 1: What are the commands did you use?

docker build --tag=microservice-kubernetes-demo-catalog microservice-kubernetes-demo-catalog

docker tag microservice-kubernetes-demo-catalog ntuanhai/microservice-kubernetes-demo-catalog:latest

docker push ntuanhai/microservice-kubernetes-demo-catalog

docker build --tag=microservice-kubernetes-demo-customer microservice-kubernetes-demo-customer

docker tag microservice-kubernetes-demo-customer ntuanhai/microservice-kubernetes-demo-customer:latest

docker push ntuanhai/microservice-kubernetes-demo-customer

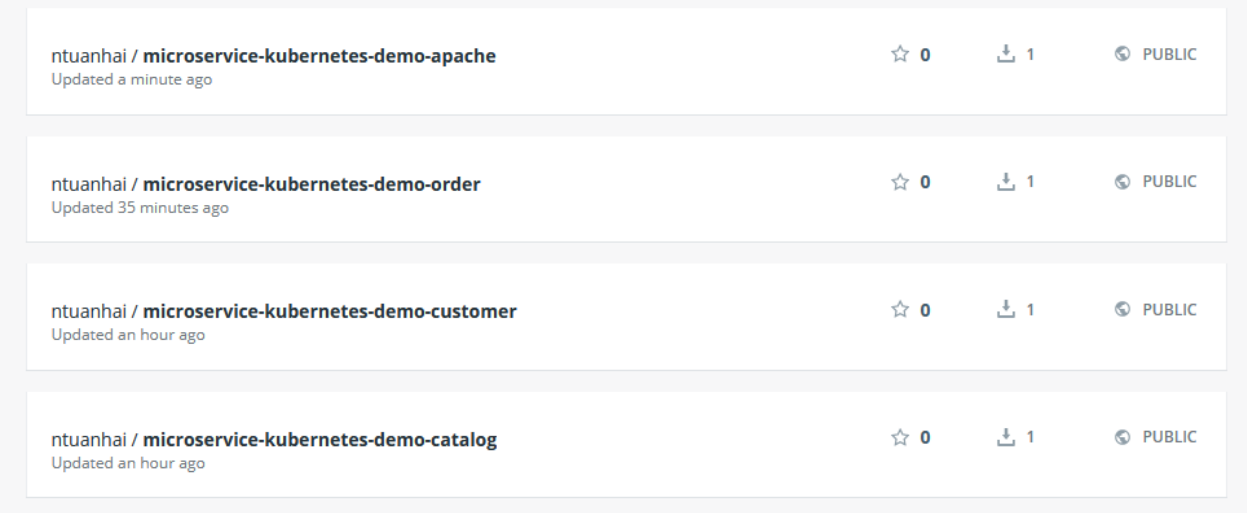
docker build --tag=microservice-kubernetes-demo-order microservice-kubernetes-demo-order

docker tag microservice-kubernetes-demo-order ntuanhai/microservice-kubernetes-demo-order:latest

docker push ntuanhai/microservice-kubernetes-demo-order

Question 2: Open the website Docker Hub and login with your account. What's new in your docker hub repository?

I see 4 new repositories



Question 3: What is the status of these created pods? Now, wait few minutes and re-type this command, what is the new status of these pods?

The status of created pod is ContainerCreating
After waiting for few minutes, the new status of these pods is Running

Question 4: What is the role of application server glassfish?

Glassfish serves as an application server mostly served for Java EE application, which can be used as HTTP Server, handling HTTP requests, handle servlets & JSP for java web development.

Question 5: Why do we need to create the 2 JNDI above?

Two JNDI are used because we need to provide services for both MyReceiver and MySender

The firs JNDI is for sender or receiver to look up in order to create TopicConnection by a JMS provider
The second JNDI is for sender or receiver to create TopicPublisher/TopicSubscriber

Question 6: Explain the message passing method of Sender and Receiver in basing on the theory of event-based architecture.

Event-based architecture consists of subscriber and publisher.

Sender here acts as publisher and uses myTopicConnectionFactory to create myTopic, then creates a publisher to publish message to myTopic

Receiver here acts as subscriber, subscribes to myTopic, then have a listener listen to the message when it is published and trigger onMessage event.

Question 7: Compare the JMS and DDS.

Feature	JMS	DDS
Architecture	Publish subscribe	Publish subscribe (multicast)
Platform Independence	Same API is exposed for all HW, OS, and languages supported	Same API is exposed for all HW, OS, and languages supported
Discovery of endpoints	JNDI and JMS servers must be specified and configured	Dynamic Discovery, no need to specify where endpoints reside
Type Safety	Generic Objects and XML are not type safe	Strong type safety, application calls write() and read() with a specific data type
Tailoring communication behavior	Limited ability to tailor communications	QoS policies allow for easy tailoring of communication behaviors
Interoperability	None	Open standard with proven interoperability