**Workshop 2, CD part with kubernetes.**

**1, Install the Azure CLI for Windows**

<https://docs.microsoft.com/en-us/cli/azure/install-azure-cli-windows?view=azure-cli-latest>

in Powershell, az login (you can see I have multiple subscriptions)



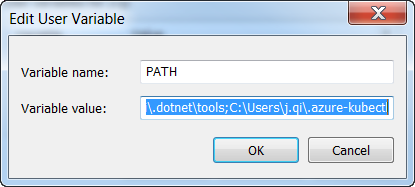
Chrome will launch, login with [j.qi@osram.com](mailto:j.qi@osram.com), then browser will disappear in 10 secs.

**2, Install KubeCtl and change environment parameters:**

In Powershell, install Kubernetes with a command “az aks install-cli”

“kubectl.exe” will be downloaded at “C:\Users\j.qi\.azure-kubectl”

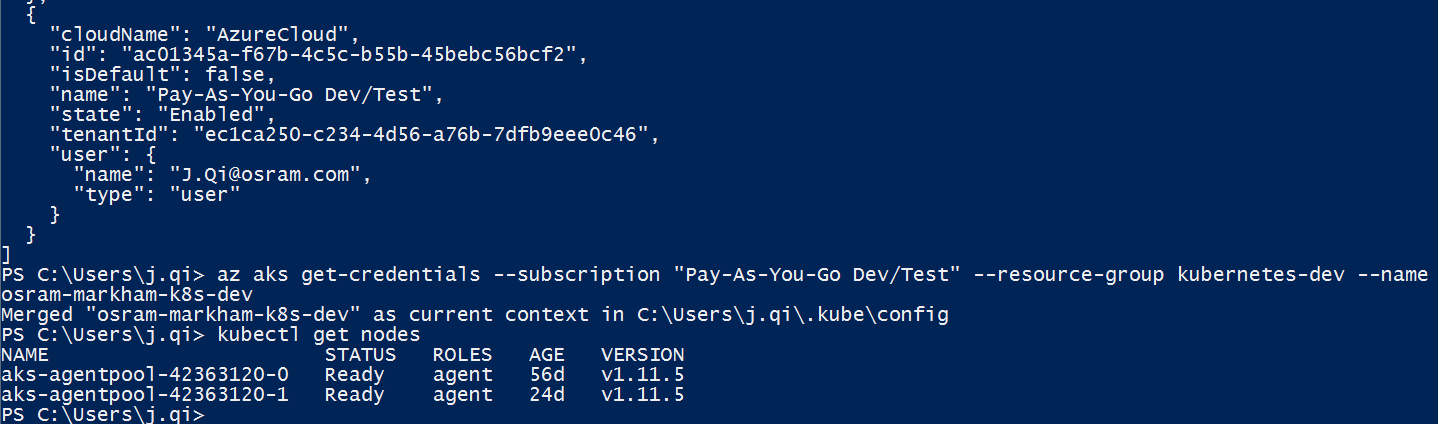
From Computer, right click, properties, Advanced System Settings, Advanced tab, Environment Variables, User variables for J.Qi, find PATH, add “C:\Users\j.qi\.azure-kubectl”



Use “kubctl” in powershell to verify kubctl is installed properly.

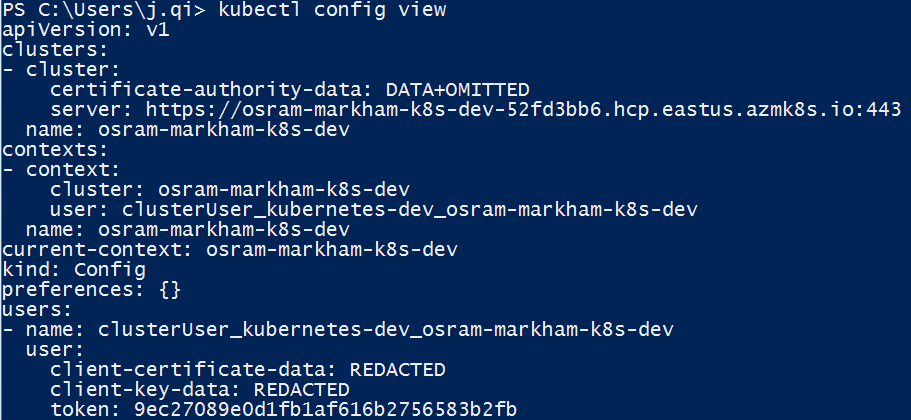
**3, Access to a kubernete cluster (assumed step 1 azure login is already done, and KubeCtl is already installed from step 2)**

Since I have multiple subscription, instead of using “az aks get-credentials --resource-group kubernetes-dev --name osram-markham-k8s-dev” (grab your credentials, point to resource group then resources), I have to force subscription to Dev/Test one: by adding –subscription “Pay-As-You-Go Dev/Test” in the command. i.e, point to the Subscription tire first then point to the resource group, then to the resource under the resource group.” “Pay-As-You-Go Dev/Test” was returned under the “name” field from az login.



To verify, use “kubectl get nodes” in powershell.

To see all the Kubernete clusters (cluster is higher level than node: cluster – master + nodes, in node – pods, in pod - containers):

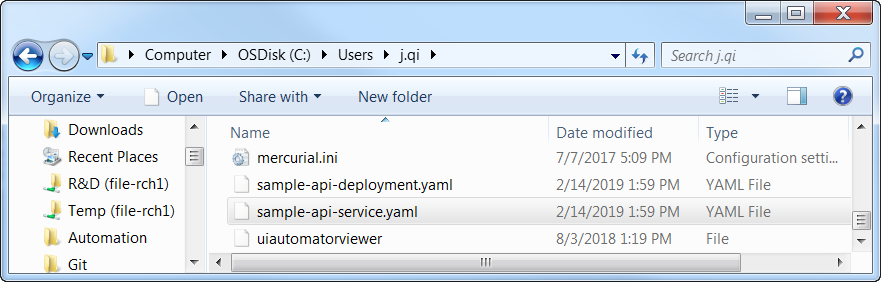


**To launch Kubenetes dashboard**:

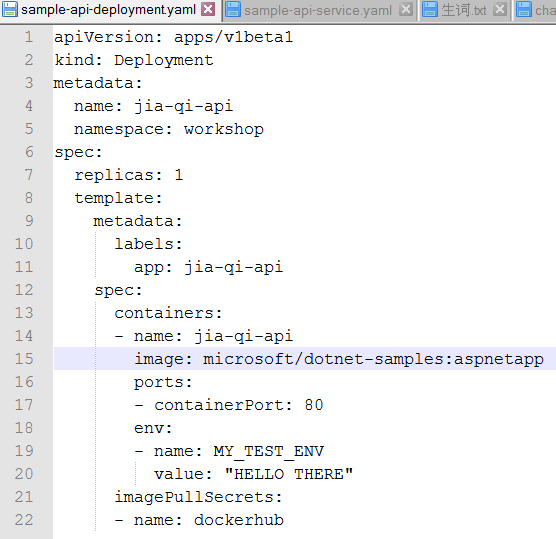
az aks browse --resource-group kubernetes-dev --name osram-markham-k8s-dev --subscription 'Pay-As-You-Go Dev/Test'

4, **Put 2 yaml files under Users/j.qi**:

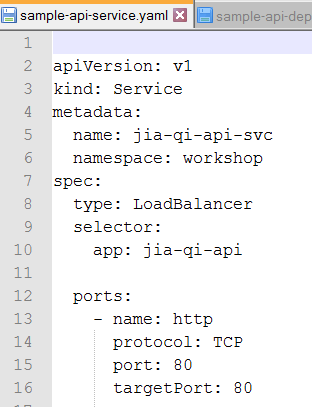
Kubernete manifest files can be in yaml or json format.



Sample-api-deployment.yaml grabs a smple docker image from dockerhub with dockerhub’s credential, to create a container named jia-qi-api



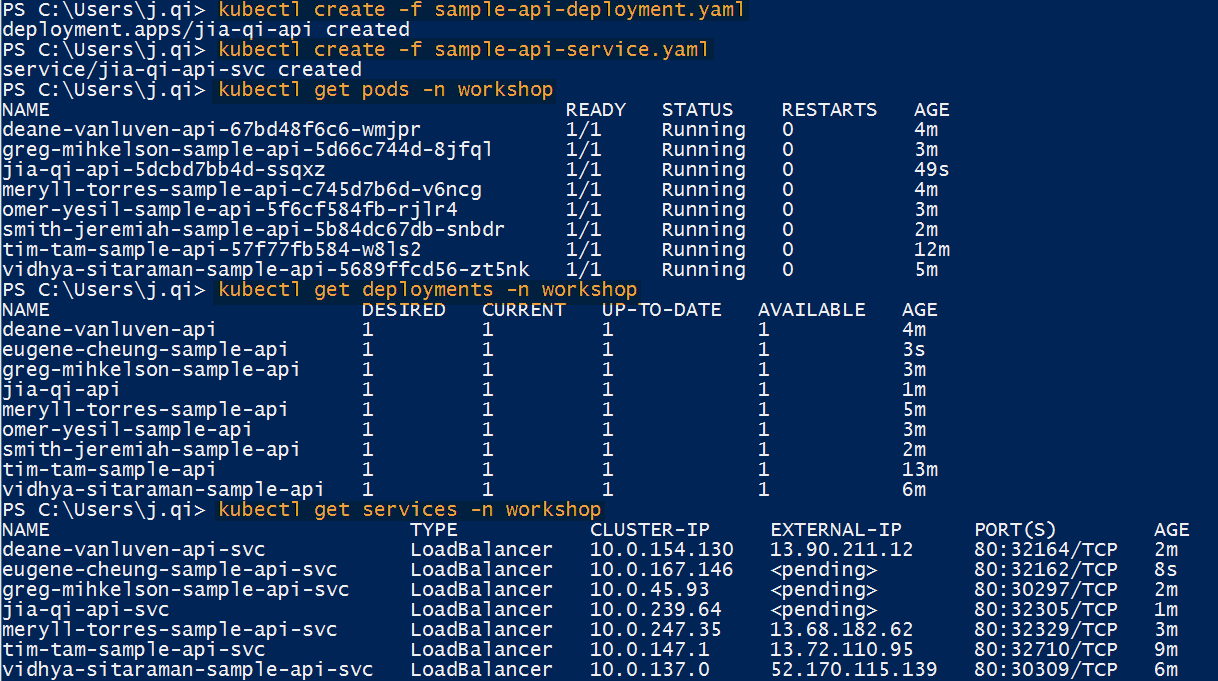
Sample-api-service.yaml: Services is the proxy between human visitors and the pods. 把人的访问request按load balance的方法分流到合适的pod上（可能有多个replicate pods有同一个label做同样的事情）



5, **Deploying to Kubernetes**:

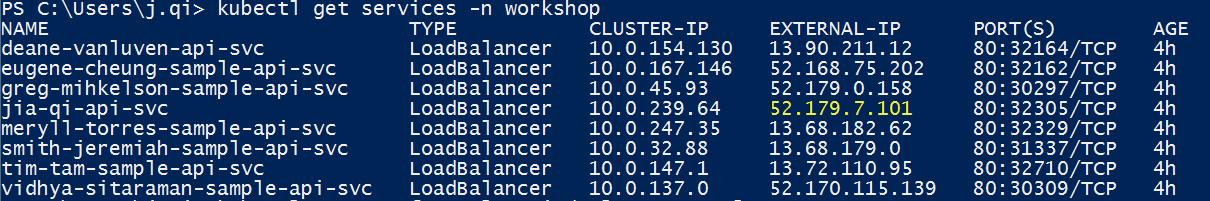
Deploy the application with deployment yaml, deploy the load balancer service with service yaml.

Verify app deployment with get pods and get deployments, verify service deployment with get services.

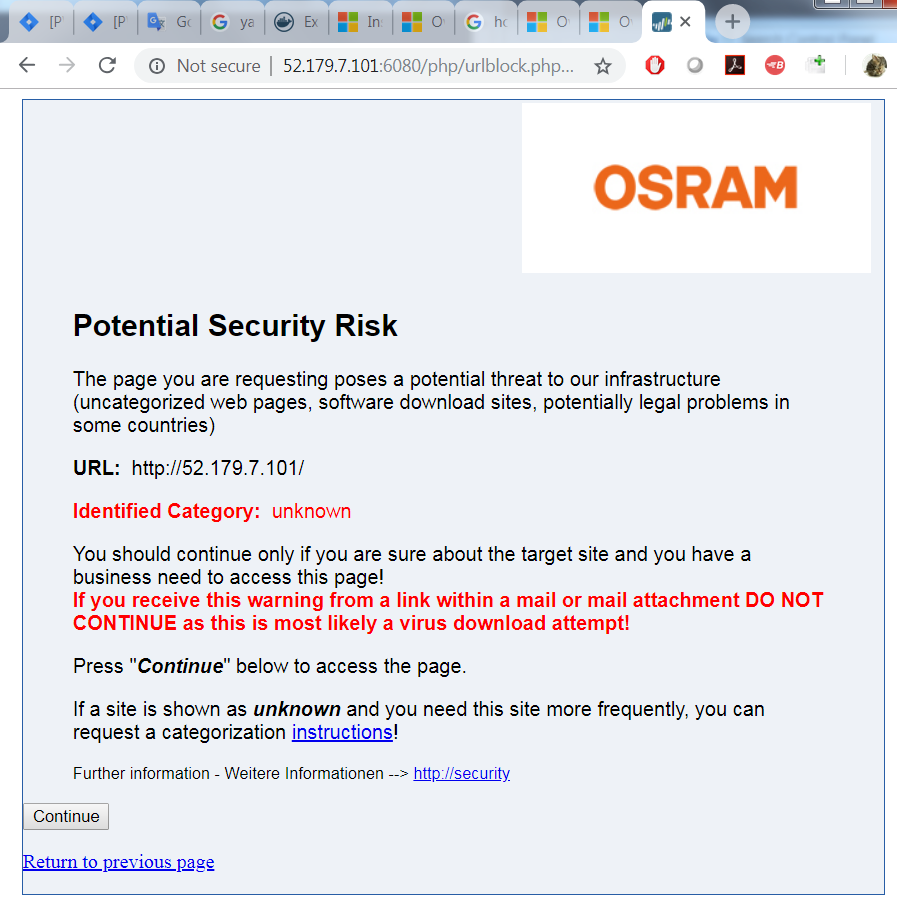


**6, view app from browser**

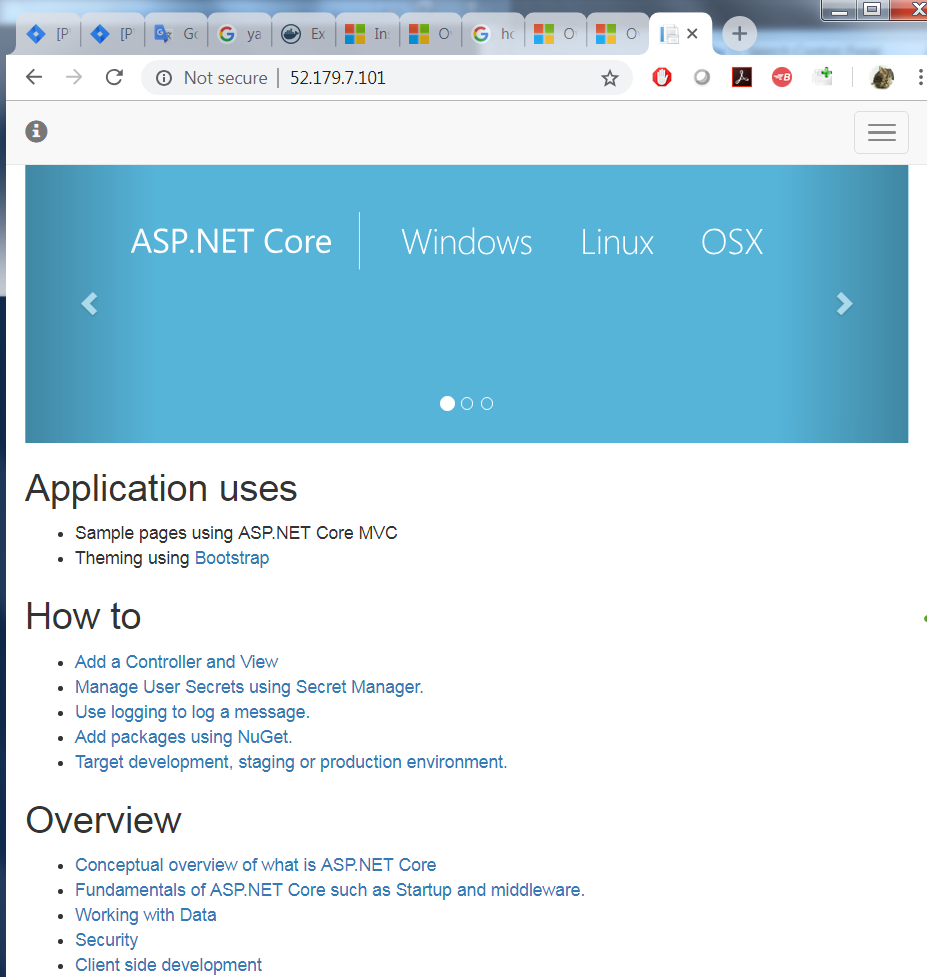
Wait a minute and verify get service again, the External\_IP will be populated for your service.



Copy paste the external IP on Chrome:

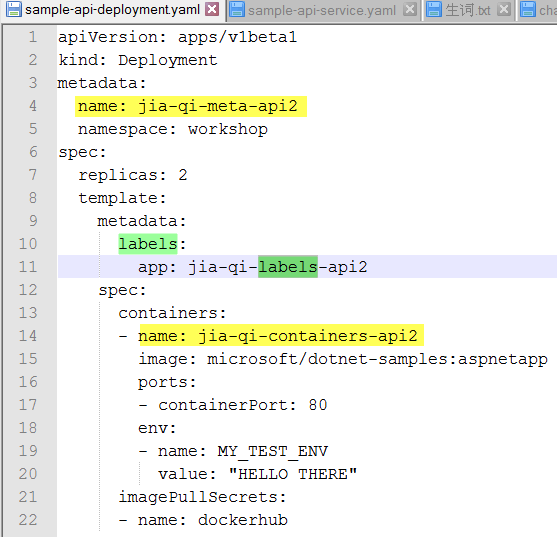


To visit your web application:

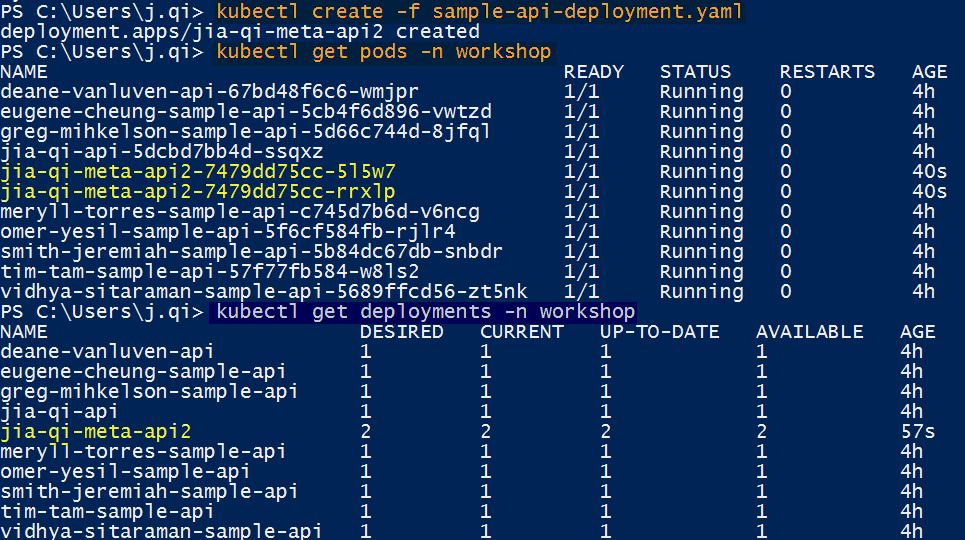


7, **The labels name from deployment yaml is used as selector app in the service yaml file**.

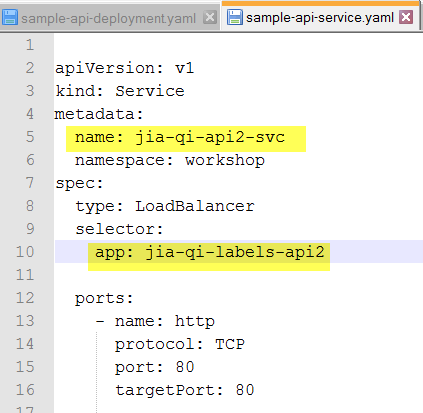
I changed replicas to 2 in the deployment yaml file, this will result 2 same pods being deployment, but each pod still have only 1 container.



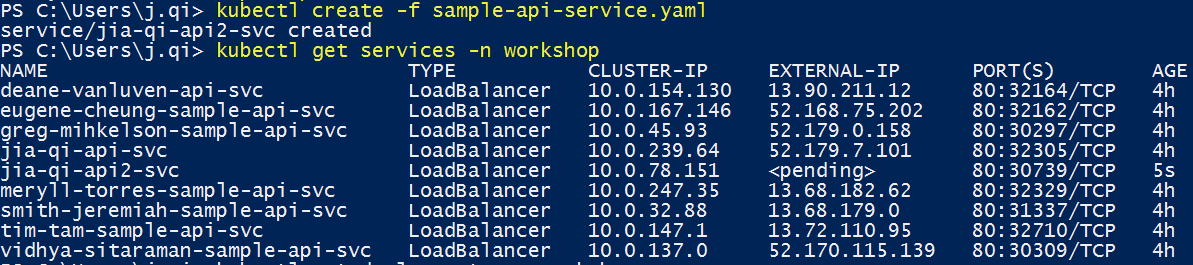
In powershell, kubectl create -f sample-api-deployment.yaml, then check pods and deployments. I noticed that 1 deployment and 2 pods are created. The deployment name is jia-qi-meta-api2, and the pods names are jia-qi-meta-api2-random numbers.



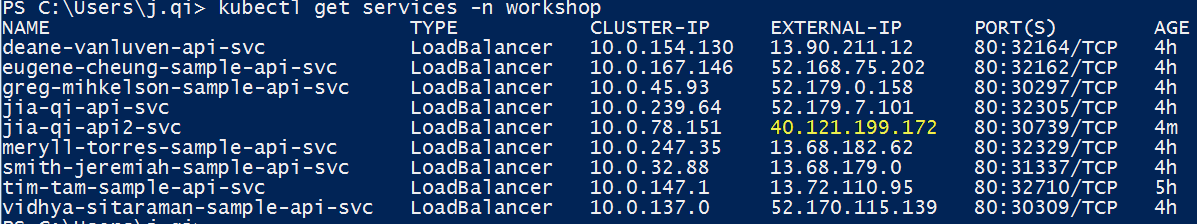
I changed the name and app fields of service yaml:



Then redeployed service:



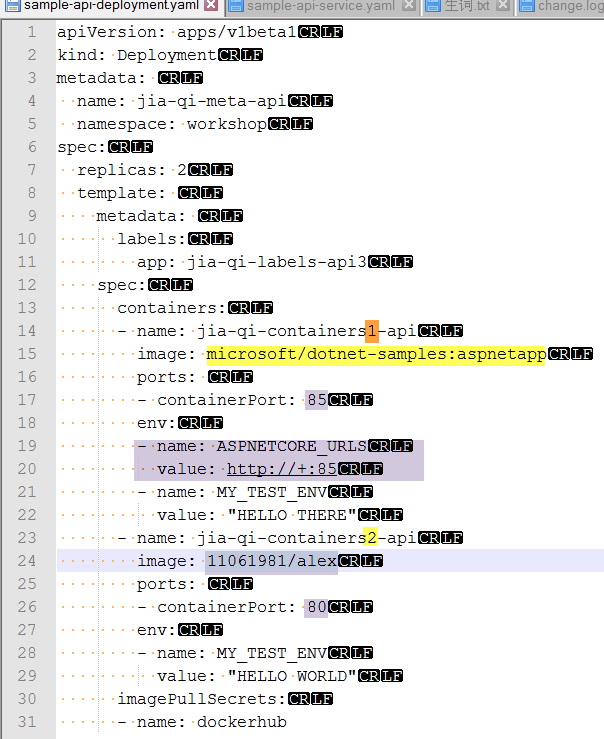
Wait a minute, see the external IP, able to visit from Chrome with the external IP.

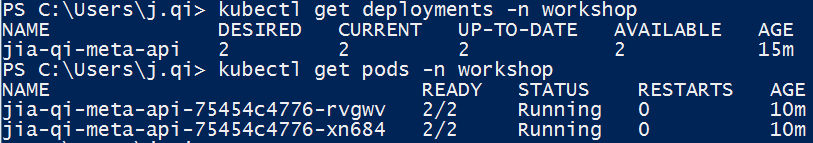


8, **To deploy 2 replicate pods with 2 containers in each pot**:

In the deployment yaml, the Containers are actually a list, so you can add more containers into the list as the screenshot below.

When modify file, please note Tab not going to work, space works. Each container needs to have different ports.



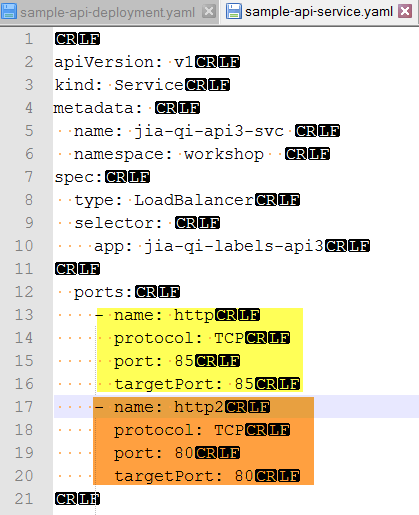


Note pod READY is 2/2 instead of 1/1, this is because now each pod has 2 containers, and both of them are ready. If one of the container has problem, you will see READY 1/2.

If your app is not a web browser app, you may have a container show up as not working, i.e., READY 1/2. For example, I used hello-world as the second app from Docker hub, and I encountered above problem. So basically, you need to find both apps displayable in browsers to use.

Use “kubectl describe pod/jia-qi-meta-api-75454c4776-rvgwv -n workshop” to see both containers in the pod.

Also, **the services file also need to be changed due to port change**:

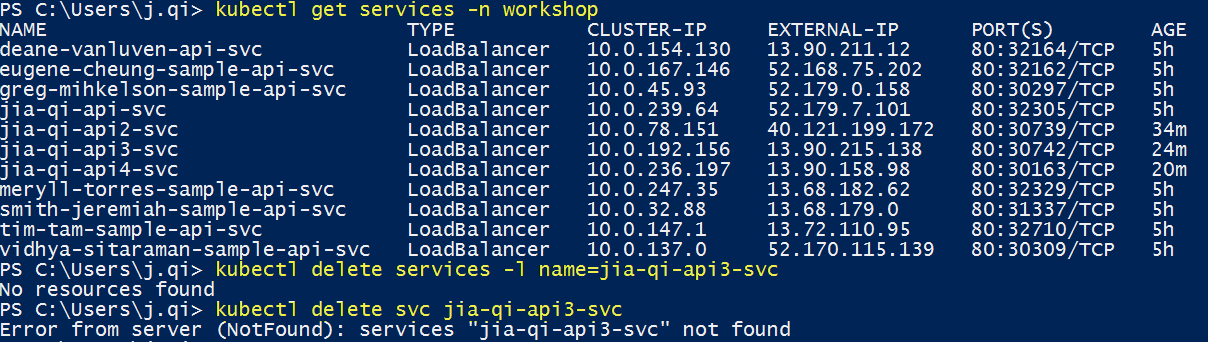




11061981/alex can be visited from browser at: <http://104.45.134.200/api/values> (note port is 80, to visit this .NET Core web app which we created from workshop 1, it is IP:port/api/values)

microsoft/dotnet-samples:aspnetapp can be visited at <http://104.45.134.200:85/> (note port is 85)

**9. How to delete deployment and services.**

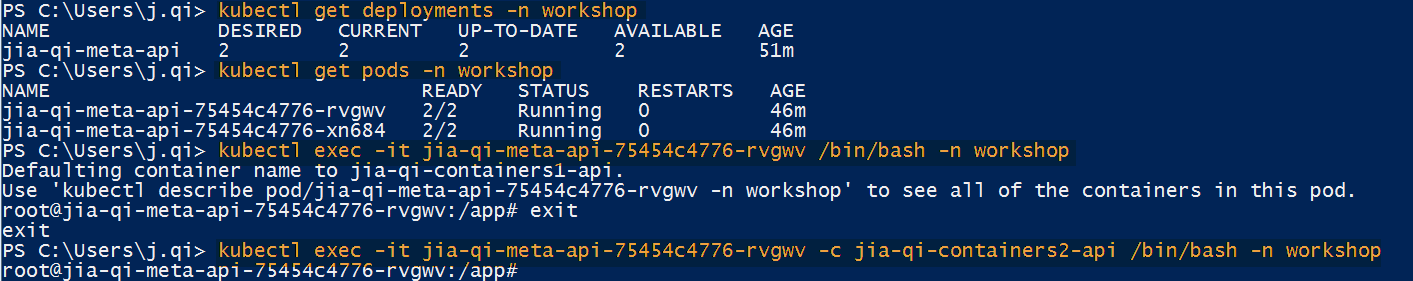


Both above not working because you forgot the namespace –n workshop, which is a logical grouping, not a physical grouping. Physical grouping are more for cluster, nodes, pods, containers, images.

* Kubectl delete deployment <deployment name> -n workshop
* Kubectl delete pod <pod name> -n workshop
* Kubectl delete services <service name> - n workshop

Note: if you delete pod, as long as the deloployment exists, it will auto regenerate pods, so basically you do not need to use “delete pod” at all, just “delete deployment”, which will auto delete all pods belong to it too.

**10, Get in to the container**



2 duplicate pods, each pod has 2 containers, container 1 is named as “jia-qi-containers1-api”, and container 2 is named as “jia-qi-containers2-api” in the deployment yaml file.

To get into the default container 1, you just need to mention pod name.

To get into container 2, you also need to mentioned container name after tag –c (same as --container)