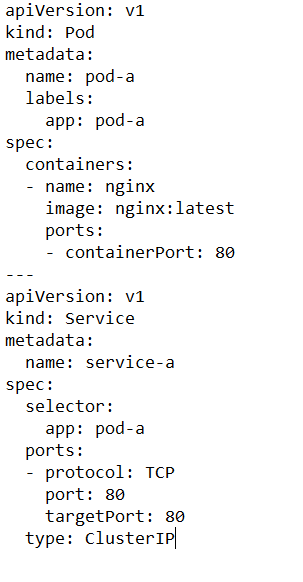
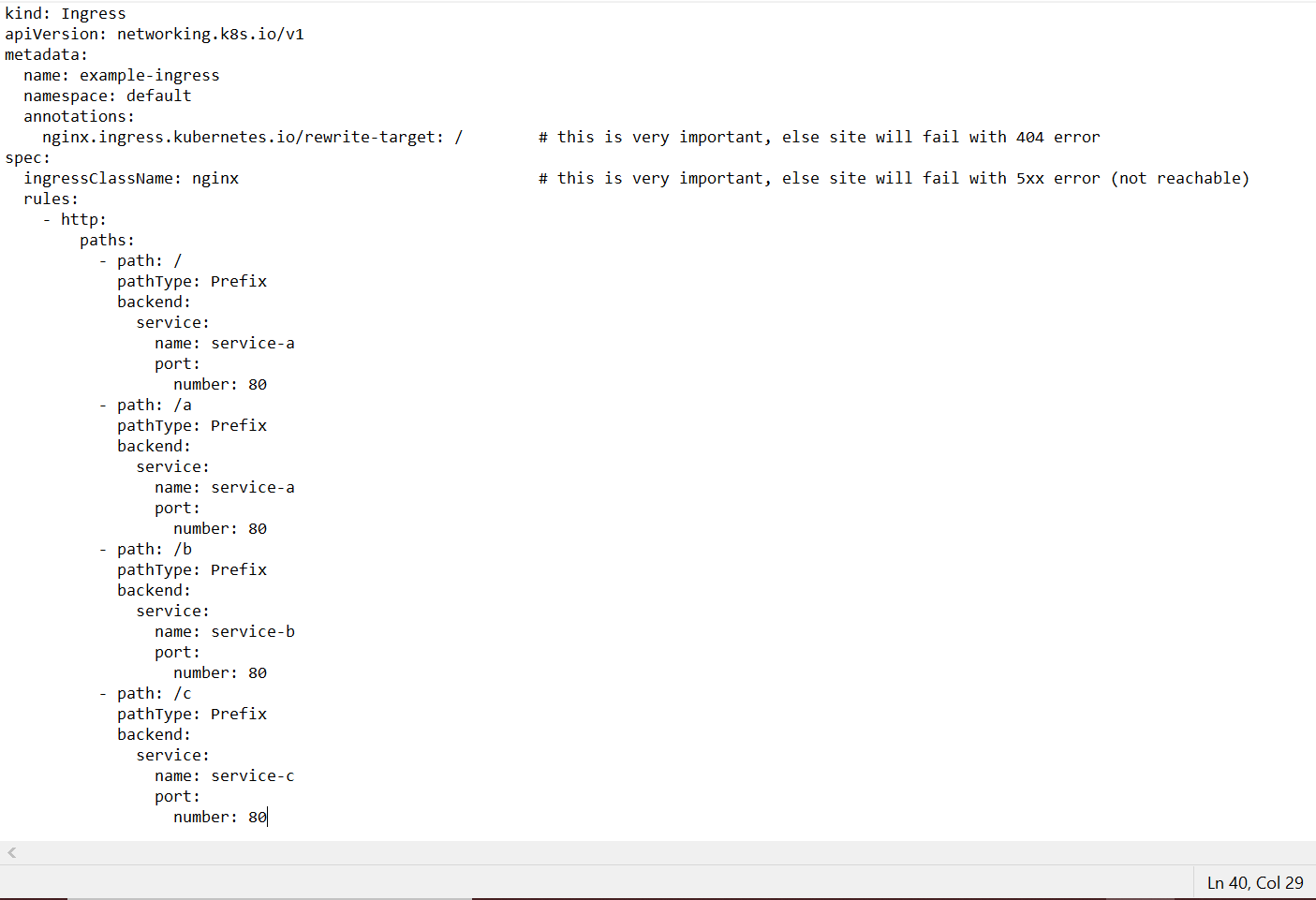
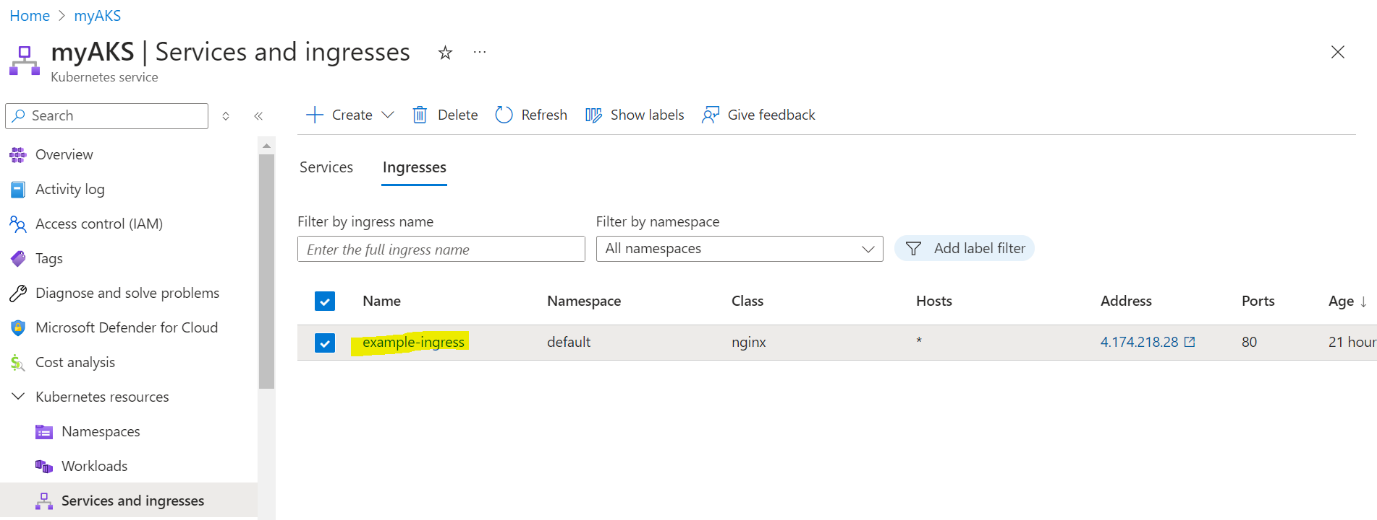
**Ingress**

1. Ingress means rather than creating load balancer for each pod, you can make a single ingress service to expose all the pods.
2. Ingress has path-based routing, like below,
   1. [www.host.com/a](https://www.host.com/a)
   2. [www.host.com/b](https://www.host.com/b)
   3. [www.host.com/c](https://www.host.com/c)
3. How to create Ingress? First create three pods, pod-a (image: nginx), pod-b (image: httpd), pod-c (image: caddy), w.r.t services service-a, service-b, service-c. Below is the screenshot of one.
4. After that in AKS use Helm, to install ingress-controller. Without an ingress controller the ingress resources won’t work. Many ingresses can use a single ingress controller. The Ingress Controller most widely used are Nginx and httpd (tomcat Apache).
5. In Azure CLI, use below commands to install helm and ingress controller,
   1. helm repo add ingress-nginx <https://kubernetes.github.io/ingress-nginx>
   2. helm repo update
   3. helm install installing-ingress14 ingress-nginx/ingress-nginx `  
      — namespace ingressmy4  
      — create-namespace  
      — set controller.replicaCount=1 `  
      — set controller.nodeSelector.”beta\.kubernetes\.io/os”=linux  
      — set defaultBackend.nodeSelector.”beta\.kubernetes\.io/os”=linux
6. After this is done you can see the ingress service running in the services.
7. Now you need to write ingress resource file, which is the most important part, in which I stumbled a lot, got a lot of errors before succeeding. Below is the ingress.yaml file which you would ultimately apply in aks (you can use azure portal directly in case of aks)
8. 
9. Results:
   1. IP/c --> caddy is of C
   2. IP/b --> it works (httpd server)
   3. IP/ --> nginx server for root

