Troubleshooting Kubernetes deployments

Read the blog article at
https://learnk8s.io/troubleshooting-deployments

START

```
~$ kubectl get pods
```

Is there any **PENDING** Pod? **?** — YES

NO

```
~$ kubectl describe pod <pod-name>
```

Is the cluster *full*? **?** — NO

YES → Provision a bigger cluster

Are you hitting the ResourceQuota limits? **?** — NO

YES → Relax the ResourceQuota limits

Are you mounting a **PENDING** PersistentVolumeClaim? **?** — NO

YES → Fix the PersistentVolumeClaim

```
~$ kubectl get pods -o wide
```

Is he Pod assigned to the Node **?** — YES → There is an issue with the Kubelet

NO → There is an issue with the Scheduler

Are the Pods **RUNNING**? **?** — NO →

```
~$ kubectl logs <pod-name>
```

YES

Can you see the logs for the app? **?** — YES → Fix the issue in the application

NO

Did it the container died too quickly? **?** — NO →

YES

```
~$ kubectl logs <pod-name> --previous
```

```
~$ kubectl describe pod <pod-name>
```

Is the Pod status **ImagePullBackOff**? **?** — NO → Is the Pod status **CrashLoopBackOff**? **?** — NO → Is the Pod status **RunContainerError**? **?**

YES

Is the name of the image correct? **?** — NO → Fix the image name

YES

Is the image tag valid? Does it exist? **?** — NO → Fix the tag

YES

Are you pulling images from a private registry? **?** — YES → Configure pulling images from a private registry

NO → The issue could be with the CRI or Kubelet

CrashLoopBackOff? YES

Did you inspect the logs and fixed app crashes? **?** — NO → Fix the app crashes

YES

Did you forget the **CMD** instruction in the Dockerfile? **?** — YES → Fix the Dockerfile

NO

Is the Pod restarting frequently? Cycling between **Running** and **CrashLoopBackoff**? **?** — YES → Fix the liveness probe

NO → Unknown state

RunContainerError? **?** — NO → Consult StackOverflow

YES → The issue is likely to be with mounting volumes

Are the Pods **READY**? **?** — NO →

```
~$ kubectl describe pod <pod-name>
```

YES

Is the Readiness probe failing? **?** — YES → Fix the Readiness probe

NO → Unknown state

```
~$ kubectl port-forward <pod-name> 8080:<pod-port>
```

Can you access the app? **?** — NO → Is the port exposed by container correct and listening on 0.0.0.0? **?** — NO → Fix the app. It should listen on 0.0.0.0. Update the containerPort

YES → Unknown state

YES

Pods are running correctly

```
~$ kubectl describe ingress <ingress-name>
```

Can you see a list of Backends? **?** — NO → Are the serviceName and servicePort matching the Service? **?** — NO → Fix the ingress serviceName and servicePort

YES

```
~$ kubectl port-forward <ingress-pod-name> 8080:<ingress-port>
```

Can you visit the app? **?** — NO → The issue is specific to the Ingress controller. Consult the docs for your Ingress.

YES

The Ingress is running correctly

The app should be working. Can you visit it from the public internet? **?** — NO → The issue is likely to be with the infrastructure and how the cluster is exposed.

YES

END

```
~$ kubectl describe service <service-name>
```

Can you see a list of endpoints? **?** — NO → Is the Selector matching the right Pod label? **?** — NO → Fix the Service selector. It has to match the Pod labels

YES

Does the Pod have an IP address assigned? **?** — NO → There is an issue with the Controller manager

YES → There is an issue with the Kubelet

```
~$ kubectl port-forward service/<service-name> 8080:<service-port>
```

Can you visit the app? **?** — NO → Is the targetPort on the Service matching the containerPort in the Pod? **?** — NO → Fix the Service targetPort and the containerPort

YES → The issue could be with Kube Proxy

YES

The Service is running correctly

learnk8s