Create a Cluster

This lesson will focus on creating a cluster again, the same way we created in the first chapter.

WE'LL COVER THE FOLLOWING

Pulling the code

We'll continue using definitions from the vfarcic/k8s-specs repository. To be on the safe side, we'll pull the latest version first.

> 🔍 All the commands from this chapter are available in the 02-ca.sh Gist.

```
cd k8s-specs

git pull
```

Next, we need a cluster. Please use the Gists below as inspiration to create a new cluster or to validate that the one you already fulfills all the requirements.

Gists and specifications

Choose the flavor you want and run the commands from its `.sh` file to create the cluster and the required specifications needed in this chapter.

> **NOTE:** In the end, you will see a command to `DELETE` the cluster too. Don't execute that command. Use the `DELETE` command only when you

need to delete the cluster, preferably at the end of the chapter.

GKE

- gke-scale.sh: **GKE** with 3 n1-standard-1 worker nodes, and with the `--enable-autoscaling` argument

EKS

- eks-ca.sh: **EKS** with 3 t2.small worker nodes, and with **Metrics Server**

AKS

- aks-scale.sh: **AKS** with 3 Standard_B2s worker nodes

Docker for Desktop and Minikube

When examining the Gists, you'll notice a few things. First of all, **Docker For Desktop** and **minikube** are not there. Both are single-node clusters that cannot be scaled. We need to run a cluster in a place where we can add and remove the nodes on demand. We'll have to use one of the cloud vendors (e.g., AWS, Azure, GCP). That does not mean that we cannot scale **on-prem clusters**. We can, but that depends on the vendor we're using. Some do have a solution, while others don't. For simplicity, we'll stick with one of the big three. Please choose between Google Kubernetes Engine (**GKE**), Amazon Elastic Container Service for Kubernetes (**EKS**), or Azure Kubernetes Service

(**AKS**). If you're not sure which one to pick, I suggest GKE, since it's the most stable and feature-rich managed Kubernetes cluster.

You'll also notice that GKE and AKS Gists are the same as in the previous chapter, while EKS changed. As you already know, the former already has the `Metrics Server` baked in. EKS doesn't, so I copied the Gist we used before and added the instructions to install `Metrics Server`. We might not need it in this chapter, but we will use it heavily later on. I want you to get used to having it at all times.

If you prefer running the examples locally, you might be devastated by the news that we won't use a local cluster in this chapter. Don't despair. The costs will be kept to a minimum (probably a few dollars in total), and we'll be back to local clusters in the next chapter (unless you choose to stay in clouds).

Now that we have a cluster in GKE, EKS, or AKS, our next step is to enable cluster auto-scaling, in the next lesson.