**Kubernetes**

Kubernetes main components :

- Pod: It adds a layer on top of the container layer so that we can interact with k8s and not docker for example. Each pod runs 1 container inside of it.

- Node: Physical server

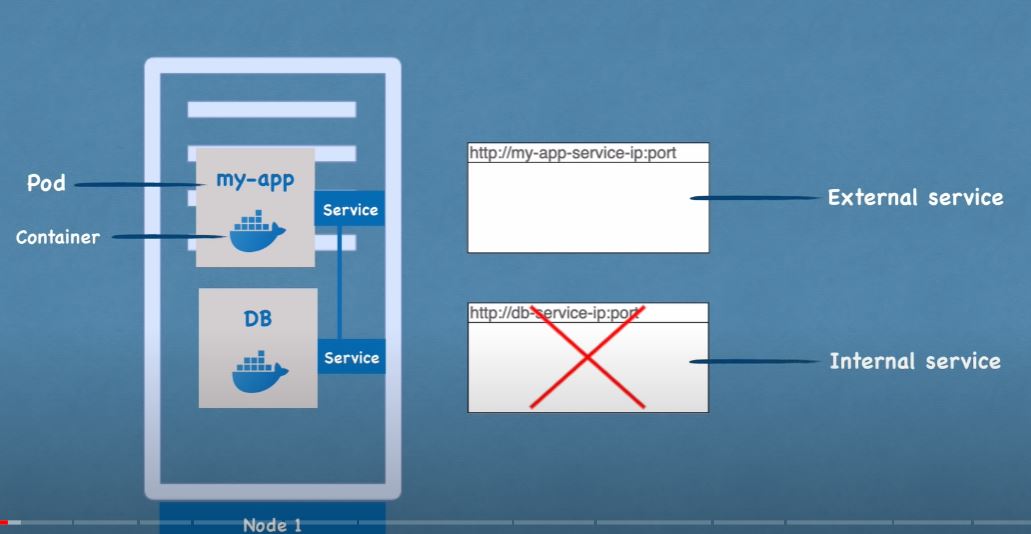
- Each pod has his own virtual network so each pod has his own IP address. If the pod has been re-created, a new IP address will be allocated for the pod which can sometimes be messy if the pod is a database container for example. To fix that problem, a component called service and ingress is used.

- Service and ingress: Permanent IP address that can be allocated to each pod. The good thing here is that the lifecycle of the pod and the service is not connected so if pod dies, the service and its IP address will stay.

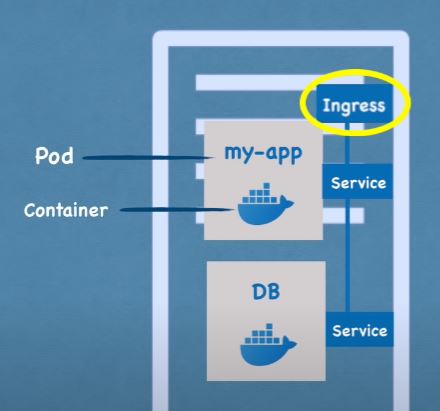
- Services: Used to route traffic from outside to our K8S cluster. To make our app inside of pods accessible to the outside, we must use external services. The service give us permanent IP address for our pod and act as a load balancer. We have 2 types of services :

External service: For our app container

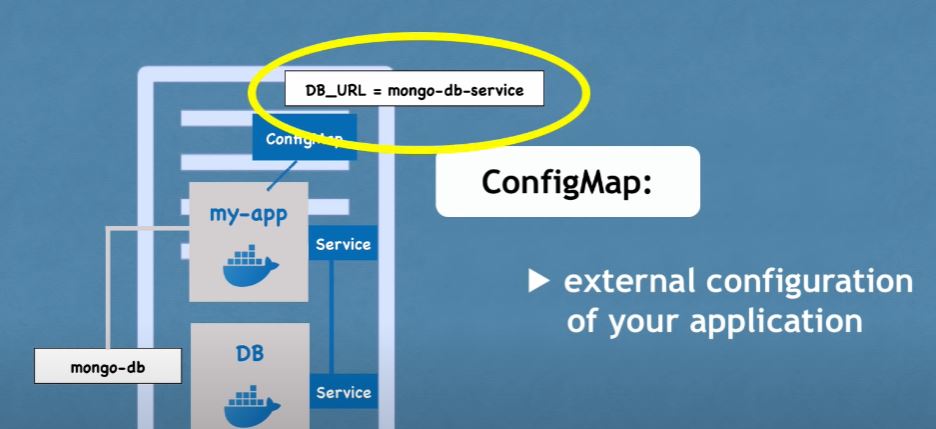
Internal service: For our database container



As we can see the external service is used for our app but if we look deeply into the URL, we are using the ip address of the node (not the service) but In real life we do not need it to look like that we need to use names so DNS. So to fix that issue, we will use something called ingress. So instead of service the request comes to ingress and then forward to service.



- Config map: It’s the external configuration of our application so it will configuration data like URL for the database… So, we will connect it to the pod. In this case, now if we change the name of the database URL for example, we do not need to rebuild the image… all we have to do is modify the config map component.



We can use data from config map inside of our pod app by using environment variables or as properties file.

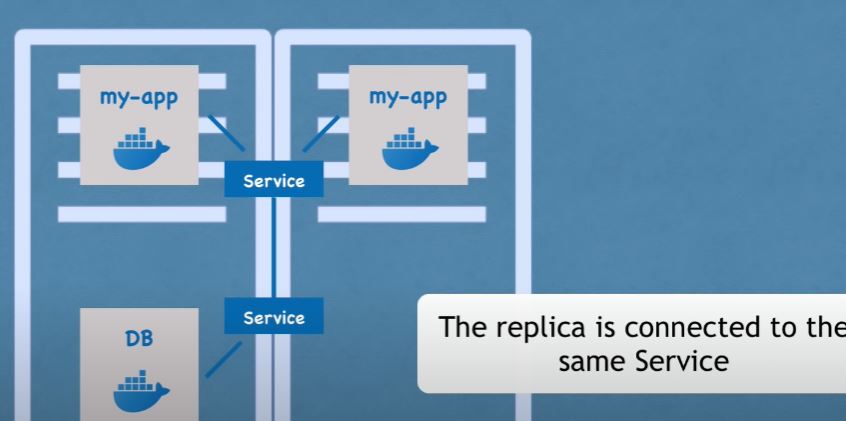
We can also use config map to change databases names or passwords… (But it is not good practice to put database passwords in the config map!) For this purpose, K8S has another service called secret.

- Secret: Used to store secret data

- Volumes: If our database pod is restarted, the data will be lost. That’s why we will use volumes. It attaches physical storage of machine to our pod files. The physical storage can be on our local machine or on a remote storage (outside of the K8S cluster)

Replicating pods :

If our pods die, we will lose all data. That’s why we will be replicating our pods on different nodes (servers). The replica will be connected to the same service.



The service has a permanent IP address and work as a load balancer so the service will catch the request and send it to the least busy pod. To create the replica, we will not create a second pod but we will write a blueprint for our pods and specify how many replicas of that pod. And this blueprint is called deployment.

- Deployment: Blueprint for the pods that will create and replicate. So in practice, we will not create pods but we will write deployments and there we will specify how many replicas we want.

N.B: Databases can NOT be replicated via deployment and the reason for that is that databases have a state which its data so they always need one central database to avoid data inconsistences. This mechanism is offered by another K8S component called stateful set.

- Stateful set: So, our database pods should be created using stateful sets and not deployment. And stateful set just like deployment will be taking care of replicating pods by making sure that the replicated databases are synchronized. Deploying stateful sets is more complicated than deploying deployment. That’s why a best practice is to host databases outside of the K8S cluster and just stateless (deployment) apps inside of the K8S cluster.

Explaining K8S architecture :

2 types of nodes that K8S operates on:

- master node

- slave node

**Worker machine :**

- Now we will consider we have 1 node (Worker machine) with 2 pods installed on it (application and database pod).

- 3 processes must be installed on every node that are used to schedule and manage those pods. Worker machine is a worker node that actually do the work.