Scaling Services

Scaling our app to handle multiple requests

We'll cover the following

- Scaling app containers

Right now, you may be in a situation where you don't know how to handle these Docker services because, in swarm mode, most of the commands we used earlier won't work exactly as expected. For example, try removing an app container. As soon as you stop a container, the swarm manager will create a new one because the default state was to have one app and one database container.

So, if you want to stop your app, you can only do that by stopping the service itself.

Scaling app containers

You can deploy multiple containers on a different node. If there are multiple nodes in the cluster, the swarm manager will create container copy on every other node until it covers all the nodes or the defined number of containers are created.

And if there is only a single node in the cluster, all the containers are created in only that node.

We will scale our app to have three app containers and two database containers.

Let's see this in action. If you haven't stopped the services, then type `docker service scale <service ID/Name>=Replicas_number`.

```
$ docker service scale rnlrh4e262ae=3
rnlrh4e262ae scaled to 3
overall progress: 3 out of 3 tasks
1/3: running   [==================================================>]
2/3: running   [==================================================>]
3/3: running   [==================================================>]verify: Se
rvice converged

$ docker service scale database=2
```

```
database scaled to 2
overall progress: 2 out of 2 tasks1/2: running   [==========================
======================>]
2/2: running   [==================================================>]verify: Se
rvice converged
```

Great, you have scaled your app to handle the extra load. The swarm manager will take care of load balancing. To check whether the service is scaled or not, type `docker service ls` and check the replica column.

```
$ docker service ls
ID                  NAME                MODE                REPLICAS
            IMAGE                     PORTS
is4ct57yaqu0        database            replicated          2/2
                mysql/mysql-server:5.7
rnlrh4e262ae        priceless_napier    replicated          3/3
                flask_app:3.0             *:5000->5000/tcp
```

Type `docker ps` to see all the running containers.

Services can be scaled up and down using the same command. If you just provide the number of replicas, the swarm manager will create or stop the containers based on the number provided.

As every container handles the different requests, it might happen that a certain request can make a container exit the execution. If there are multiple hosts machines, how will you know which machine has stopped working and where the remaining containers created by the Swarm manager are?

For this, Docker has a GUI service that monitor's all the running containers and nodes in a swarm cluster.

In the next lesson, we will add a visualizer to our swarm cluster.