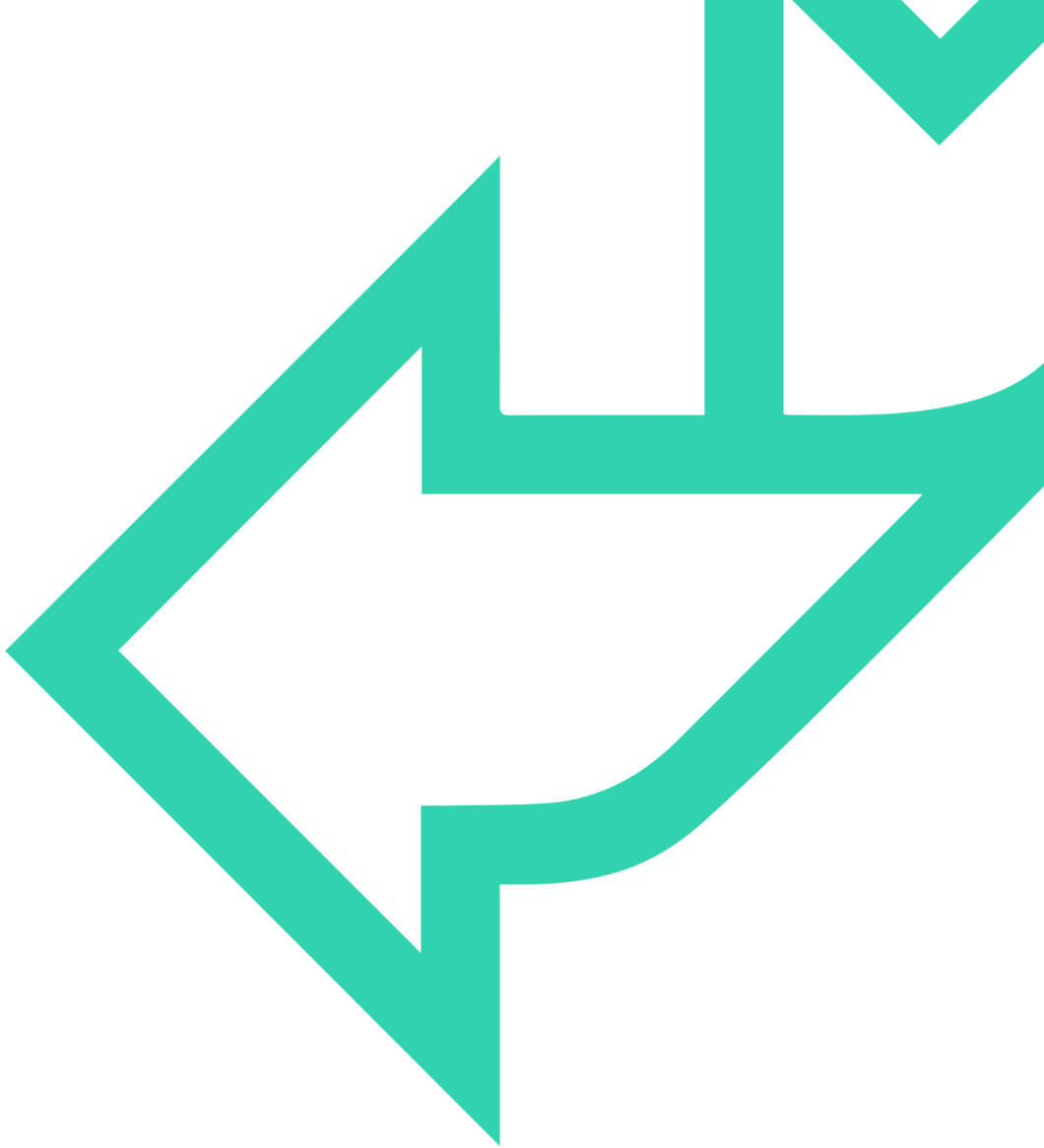
POWERING POTENTIAL

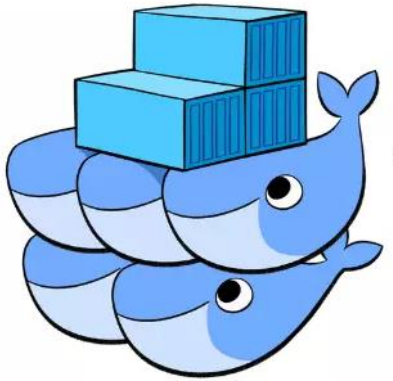
Docker Swarm



Container Orchestration

- Docker CLI can containerise apps and deploy them with ease
- In production serving 1000’s of users, using the CLI isn't a feasible option
- To deploy and manage containers at scale you’ll need to use a container orchestration

Kubernetes



Docker Swarm

Benefits of Container Orchestration tools

- **Easily scale containers (up/down)**
  - Control how many instances of a container are running,
    Easily increase the scale of our operation when the user base increases

- **Increased redundancy and improved resiliency**
  - if one instance of your container dies, there will be others to take its place

- **Deploy containers across multiple machines (nodes)**
  - make use of more compute resources and improve system resiliency

- **Load balancing between containers to network traffic across nodes**
  - Dynamically re-allocate containers across nodes -
    - if a node dies, containers on that node will be reallocated to the remaining nodes automatically
- **Rolling updates** can be done without stopping or restarting any containers
  - Updates can be implemented without requiring any downtime for the end user

DOCKER SWARM

A swarm is a group of containers running across a group of machine (nodes) called a cluster

- **Is Docker's in-built container orchestration tool**.
  - No extra installation requirements

- **Swarm has many components:**
  - clusters, manager nodes, worker nodes and a network mesh for ingress traffic.

MANAGER AND WORKER NODES

Manager Nodes

- This node can execute commands for managing the Swarm
- Can have more than one Manager node in a cluster
    - for high availability, fault tolerance, lead election

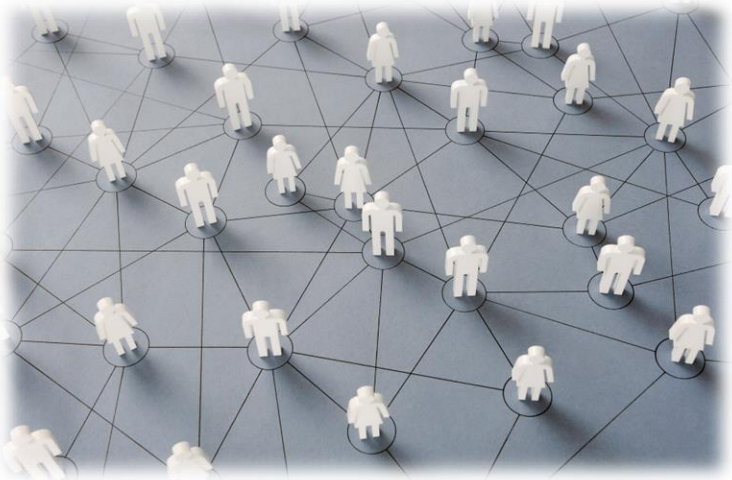
- When a cluster is initialised, the first node becomes the first manager.
    - It can produce a join token to allow more manager or worker nodes to join the cluster

Worker Nodes

- Worker just run containers, complying with the manager nodes requests.
- Containers for deployed services will be automatically provisioned evenly across worker nodes.

Communication between nodes

- **Nodes within a cluster communicate with one another**
  - Balances the containers within the swarm across the cluster
  - making maximum use of compute resources
  - Also ensuring no node is being overworked



Docker Swarm vs Kubernetes

Feature	Docker Swarm	Kubernetes
Ease of Use	Simpler setup and management	More complex, requires deeper knowledge
Integration	Native Docker integration	Extensive ecosystem, integrates with many tools
Scalability	Good for smaller-scale deployments	Highly scalable, suited for large-scale deployments
Feature Set	Basic orchestration features	Extensive features, supports complex use cases
Community Support	Smaller community	Large, active community
Resource Requirements	Lightweight	More resource-intensive
Customizability	Less customizable	Highly customizable
Self-Healing	Basic failover capabilities	Advanced self-healing capabilities
Networking	Basic built-in networking	Advanced networking features

LAB

DOCKER SWARM PLAYGROUND

https://app.qa.com/lab/docker-swarm-playground/

Duration: 4h

This Docker swarm playground lab provides you with a Docker swarm cluster running in Microsoft Azure.

The cluster is comprised of one manager node and two worker nodes.

The nodes have docker, docker-compose, and command-line completions pre-installed and ready for you to play with.

The cluster allows public access on ports 80, 443, and 8080.

Go further! **Docker and K8s**

https://app.qa.com/learning-paths/building-deploying-and-running-containers-in-production-1-888/

Hope you enjoyed our session on Dockers