**Kubernetes Importance**

[**https://capgemini.udemy.com/course/learn-kubernetes/learn/lecture/14655916#overview**](https://capgemini.udemy.com/course/learn-kubernetes/learn/lecture/14655916#overview)[**https://kodekloud.com/**](https://kodekloud.com/) **(Lab details)  
  
BLP sessions for Shivakanta and pathways:**Pathway: <https://degreed.com/pathway/39zrvlmzpq/pathway>  
  
1) <https://capgemini-my.sharepoint.com/:v:/r/personal/harshada_gaikwad_capgemini_com1/Documents/Recordings/Blended%20Learning%20Program%20-Kubernetes_Kickoff%20Session-%2013%20July%202023-%2010_00%20AM%20TO%2011_00%20AM-%2023061912RCTechNext-20230713_100703-Meeting%20Recording.mp4?csf=1&web=1&e=GdADGm&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZy1MaW5rIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXcifSwicGxheWJhY2tPcHRpb25zIjp7InN0YXJ0VGltZUluU2Vjb25kcyI6MTQuMDJ9fQ%3D%3D>

2) <https://capgemini-my.sharepoint.com/:v:/r/personal/karanam-rajashekar_naidu_capgemini_com/Documents/Recordings/Blended%20Learning%20Program%20-Kubernetes_Mentoring%20Session%201-%2020%20July%202023-%2010_00%20AM%20TO%2012_00%20PM-%2023061912RCTechNext-20230720_100409-Meeting%20Recording.mp4?csf=1&web=1&e=Mt0cF8&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZy1MaW5rIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXcifX0%3D>

3) <https://capgemini-my.sharepoint.com/:v:/r/personal/arun_a_rai_capgemini_com/Documents/Recordings/Blended%20Learning%20Program%20-Kubernetes_Mentoring%20Session%202-%2027%20July%202023-%2010_00%20AM%20TO%2012_00%20PM-%2023061912RCTechNext-20230727_100331-Meeting%20Recording.mp4?csf=1&web=1&e=2i3amA&nav=eyJyZWZlcnJhbEluZm8iOnsicmVmZXJyYWxBcHAiOiJTdHJlYW1XZWJBcHAiLCJyZWZlcnJhbFZpZXciOiJTaGFyZURpYWxvZy1MaW5rIiwicmVmZXJyYWxBcHBQbGF0Zm9ybSI6IldlYiIsInJlZmVycmFsTW9kZSI6InZpZXcifX0%3D>

**One more BLP session:**

**1) Kick off session:** <https://capgemini-my.sharepoint.com/:v:/p/thurkkeeswaran_murugavel/Ed5GT-vAIe1EiFdHsF2lge4BP9fEo4RCsDOqrrMijvE_AQ?referrer=Teams.TEAMS-ELECTRON&referrerScenario=MeetingChicletGetLink.view.view>

**2) Mentoring session 1:** <https://capgemini-my.sharepoint.com/personal/kavya_madda_capgemini_com/_layouts/15/stream.aspx?id=%2Fpersonal%2Fkavya%5Fmadda%5Fcapgemini%5Fcom%2FDocuments%2FRecordings%2FBlended%20learning%20programs%20%2DKubernetes%5FMentoring%20Session%201%20%2D%2018%20January%2010%5F00%20AM%20TO%2001%5F00%20PM%20%2D%2023113048RCTechNext%2D20240118%5F100957%2DMeeting%20Recording%2Emp4&referrer=StreamWebApp%2EWeb&referrerScenario=AddressBarCopied%2Eview>

**3) Mentoring session 2:** <https://capgemini-my.sharepoint.com/:v:/p/deepak_sontakke/EUj1Q4MxngpBnVOjzonNA6QBwH8XgydMPZZHxye0arQ5dQ?referrer=Teams.TEAMS-ELECTRON&referrerScenario=MeetingChicletGetLink.view.view>

**4) Monitoring session 3:** <https://capgemini-my.sharepoint.com/personal/rajendra-kumar_meesala_capgemini_com/_layouts/15/stream.aspx?id=%2Fpersonal%2Frajendra%2Dkumar%5Fmeesala%5Fcapgemini%5Fcom%2FDocuments%2FRecordings%2FBlended%20learning%20programs%20%2DKubernetes%5FMentoring%20Session%203%20%2D%2031%20January%2010%5F00%20AM%20TO%2001%5F00%20PM%20%2D%2023113048RCTechNext%2D20240131%5F101040%2DMeeting%20Recording%2Emp4&referrer=StreamWebApp%2EWeb&referrerScenario=AddressBarCopied%2Eview>.

**Instructor - Mumshad Mannambeth  
KodeKloud Training**

**KUBERNETES for the absolute beginners:**

1. Lab Environment 5) Services
2. Pre-requisites – YAML 6) Micro-service Architecture
3. PODs Deployment 7) Demos
4. Networking Basics 8) Coding Exercise

**KUBERNETES for Administrators:**

1. HA Development 6) Security
2. Kubernetes scheduler 7) Troubleshooting
3. Logging/Monitoring 8) Core concepts
4. Application Lifecycle 9) Demos
5. Maintenance 10) Coding exercise

**KUBERNETES for Developers:**

1. Core concepts 6) Pod design
2. ConfigMaps service accounts 7) Jobs
3. Multi-container pods 8) Services
4. Readiness and liveness pods 9) Demos
5. Logging and monitoring 10) Coding exercise

**Kubernetes:** It is also known as K8s was build on google based on their experience running containers in production, it is known as open-source project, and it is one of the best **container orchestration technologies** out there.

When we are understanding a Kubernetes, we must understand two things are containers + orchestration

**Containers** – Specifically we will look at the most popular container technology out there, that is **Docker**Why do we need containers?  
-> Compatibility/Dependency  
-> Long set-up time  
-> Different Dev/Test/Prod environments

What can it do?  
-> With Docker I was to run each component in a separate container with its own libraries and its own dependencies all on the same VM and the OS but separate environment or container, we just have to build the docker configuration once and all are developers could start with simple docker run command.

What are containers?  
-> Containers are completely isolated environment as in they have own processes or services there own networking interfaces just like virtual machines accept there is all shared the same operating system kernel.

**Container orchestration –** Our application into an docker container, then what next how can I run it in production what if the application relies on the other containers such as: databases, messaging services or other backend services and what are the number of users increasing and we need to scale our application and how it will scale down when load decreases, so to enabling these functionalities we need a underlining platform with the set of resources and capabilities the platform needs to orchestrate the connectivity between the containers and automatically scale up and down based on the load. This whole process automatically deploying and managing containers is known as container orchestration.

Kubernetes is just a container orchestration technology there are multiple technology available today docker as its own tool called **docker swarm** **Kubernetes from google** and **MESOS from Apache**.

**Docker swarm –** It is very easy to set-up and gets started it lacks some of the advance features required for complex applications.  
**MESOS –** it’s quite difficult to set-up and get started but supports many advanced features.

**Kubernetes –** It has most popular of all is a bit difficult to set-up but get started but provides a lot of option to customize deployments and supports to deployments to complex architectures.   
Kubernetes is now supporting all service providers like GCP, Azure and AWS and Kubernetes projects is one of the top ranked projects in GitHub

There are various advantages of Kubernetes container orchestration are:  
-> Our application is now highly available as hardware failures do not our application down, because I have multiple instances of our application running on different nodes user traffic is load balanced across the various containers when demand increases deploy more instances of the applications seamlessly and with in a matter of seconds and we have a ability to do that the service level when we run out of hardware resources scale the number of underlined nodes are up and down without having to take down the application. And all of these easily with the set of declarative objects configuration files and that is Kubernetes.

It is a container orchestration technology used to orchestrate the deployment and management of hundreds and thousands of containers in a clustered environment.

**Kubernetes architecture:**

**Nodes:** A node is a machine physical or virtual on which Kubernetes is installed, a node is a worker machine and that is where container will be launched by Kubernetes, it is also known as Minions in the past so we might here terms used interchange bling  
when the node is running fails abously our application goes down, so we need to have more than one node for this reason cluster comes.

**Cluster:** A cluster is a set of nodes grouped together, this way even one node fails we have application still accessible from the other nodes more ever having multiple nodes helps in sharing load as well.   
Now we have a cluster but who is responsible for managing the cluster where is the information about members of the cluster stored, how are the nodes monitored when a node fails how do we move the workload of the failed node to another worker node that why the Master comes in.

**Master:** A Master is another node with Kubernetes installed in it and it is configured as a master, the master watches over the node in the cluster and is responsible for actual orchestration of containers on the worker nodes.  
When we install Kubernetes on a system actually installing the following components are:  
  
 **APIs server**, **etcd service**, **kublet service**, **Container runtime**, **controllers** and **schedulers**   
  
The **API server** acts as the front-end for Kubernetes, the user management devices, command line interfaces all talk to the APIs server to interact with the Kubernetes cluster.

**etcd service:** It is a distributed reliable key-value store used by Kubernetes to store all data used to manage the cluster, think of this way when we have multiple nodes, multiple masters in our clusters etcd store that all the information and all the nodes in the cluster in a distributed manner, responsible for implementing logs to with in the cluster to ensure that there are no complex between the masters.

**Schedulers:** The schedular is responsible for distributing work or containers across multiple nodes it looks for newly created containers and assigns them to nodes.

**Controllers:** The controller are brain behind the orchestration there responsible for no chasing and responding when nodes containers and end points goes down, the controllers make decision to bring up new container in such cases.

**Container runtime:** It is the underlining software that is used to run containers in our case it happens to be docker but there are other options as well.

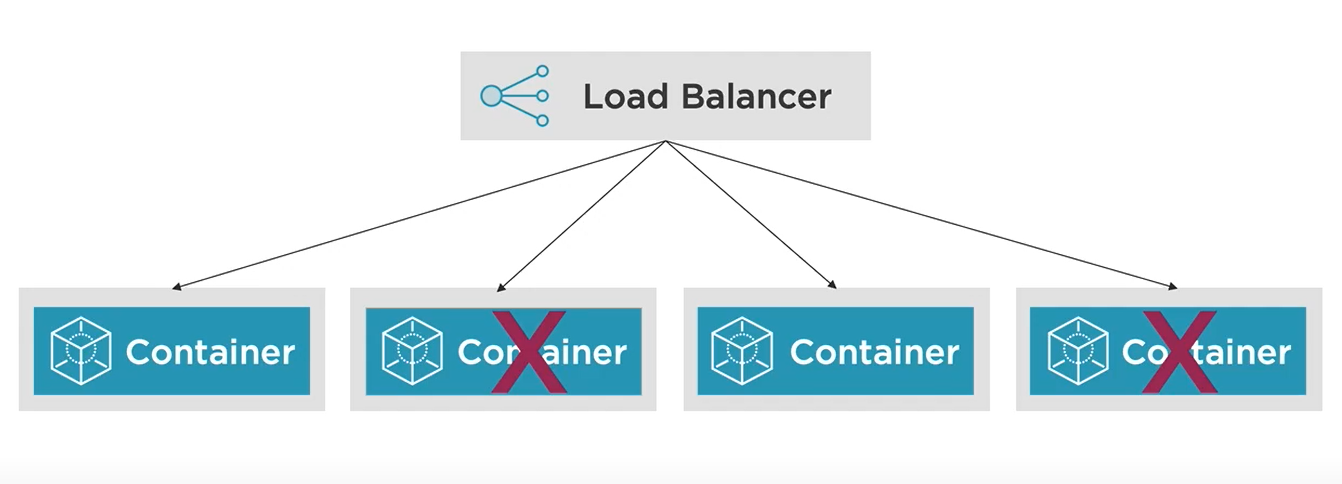
**Kublet:** Finally, kublet is the agent that runs each node in the cluster, the agent is responsible for making shores that the containers are running on the node as accepted.

One of the command line utilities is called **kubectl.**

**Kubernetes from developer perspective  
Creating Pods  
Deployments of pods  
Creating services   
Understanding storage options  
Creating ConfigMaps and secrets  
Putting it all together   
  
Course perspective:**1) Comfortable using command-line tools and virtual machines  
2) General familiarity with software development  
3) Understanding of docker containers and how they work

**Module Overview**

1. Kubernetes overview
2. The big picture
3. Benefits and use cases
4. Running Kubernetes locally
5. Getting started with kubectl
6. Web UI dashboard
7. **Kubernetes overview:** (K8s) Kubernetes is an open-source system for automating development, scaling and management of containerized applications. **(**[**https://kubernetes.io**](https://kubernetes.io)**)**

How are we managing containers?  
  
-> Suppose we set-up a load balancer and different nodes or virtual machines even physical servers all the servers are run on the different containers it works fine, but what happens is one or more containers goes down now we have lot of issue is comes.