minikube and kubectl

- What is Minikube
  - Production Cluster Setup - https://www.youtube.com/watch?v=X48VuDVv0do&t=1393s
  - Let's say if you want to test something quickly or test the new build or a component, setting up a cluster like production will be pretty difficult or maybe even impossible if you don't have enough resources like cpu, memory etc exactly for this use case there is this open source tool called "Minikube"
  - It is basically one node cluster, Both master and node process will run on one machine/node, this node will have docker container runtime pre installed.
  - The way its gonna run on your laptop is through VB
  - Minikube will crate a create Virtual Box on you laptop
  - Node runs in that Virtual Box
  - To summerize, it will be a 1 node K8s cluster
  - For testing purposes
- Since now you created a cluster but now you need something to interact with the cluster, so you create component, configure etc, that's where "kubectl" comes in the picture

- What is kubectl
    - Command line tool for k8s cluster to interact.
    - It is for to interact with any type of cluster, be it Minikube or Cloud cluster
- Installation
  - Virtualisation on your machine needed.
  - As its gonna run in VB or some hypervisor , so you will need to install some type of hypervisor

- Kubectl CLI
  ...for configuring the Minikube cluster

- Minikube CLI
  ...for start up/deleting the cluster

- Create and debug pods in a minikube cluster
- CRUD commands
  - Create deployment
    - kubectl create deployment [name]
  - Edit deployment
    - kubectl edit deployment [name]
  - Delete deployment

        - kubectl delete deployment [name]

- Status of different K8s components
    - kubectl get nodes I pod | services I replicaset I deployment

- Debugging pods
    - Log to console - kubectl logs [pod name]
    - Get Interactive Terminal - kubectl exec -it [pod name] -- bin/bash

- Use configuration file for CRUD
    - Apply a configuration file - kubectl apply -f [file name]
    - Delete with kubectl delete -f [file name]

- Some info
    - Pod is the smallest unit , you are not going to create the pod but you will create with the help of deployment (blueprint for creating pods) as it is abstraction over pods, so this what we are gonna be creating that's gonna create the pod underneath
    - cmd - kubectl create deployment NAME —image [—dry-run] [options]
    - The option that required while creating is image, because pods will be base out of that.
    - cmd - kubectl create deployment nginx-depl --image=nginx
        - kubectl get deployment
        - kubectl get pod
            - NAME READY STATUS RESTARTS AGE
            - nginx-deply-7585b7cc98-w5s6b 1/1 Running 0 29s
        - blueprint for creating pods
        - most basic configuration for deployment
        - (name and image to use)
        - rest defaults
        - And there is another layer which is automatically managed by K8s deployment called "replicaset", which is managing the replicas of a Pod
            - kubectl get replicaset
            - kubectl get replicaset
                - NAME DESIRED CURRENT READY AGE
                - nginx-deply-7585b7cc98 1 1 1 5m9s
        - When deployment is created and when the pod is created, the

pod name will have a prefix of deployment_name-replicaset-_and it's own id_
    - kubectl get all - will list all the components that are inside the cluster
    - Kubectl get secret - will show you the secret that are created
    - kubectl get pod --watch
    - minikube service _service—_name url
- Again if we see the Layers of abstraction
    - Deployment manages a Replicaset
                |
    - Replicaset manages a Pod
                |
    - Pod is an abstraction of Container
                |
    - Container
    - So, everything below the deployment is handled by Kubernetes, we don't have to worry about any of it, just have to mention to the deployment as a blueprint
    - And for ex: if you want to edit the image that pod uses, you will have to edit in the deployment directly and not in the pod
        - kubectl edit deployment _NAME_
            - It will have autogenerated configuration file with default values
            - And once you edit, save and exit, it will terminate the old pod and will start the new pod with the changes and also in replicaset, the old pods will be deleted and new pods will be added
- Cmds which help for debugging
    - kubect describe pod _pod_name_, will describe the pod stages/events
    - kubectl exec -it _pod_name — bin/bash (somewhat similar to docker), with this command I'll get the mongoDb application container
    - For such short configs/options to use in coommand line is practical but if you have lot of options or configs that you wanted to have while creating the deployment ,
      kubectl create deployment [name] —image=image_name Option1 Option2 Option3 ….
      doing it with cmd line is impractical, so you will work with K8s config files (YAML) and you just tell kubectl to execute that file, like
        - kubectl apply -f _config_file.yaml_
        - When you updated the file and when you run the cmd, K8s will know when it was created and update, basically it will know when to create or update deployment.

        - With kubectl apply .., you can both create and update a component , you can do this with any other components like services, volumes any other K8s components.

- Summary - Create and debug pods in a minikube cluster
    - CRUD commands
        - Create deployment
            - kubectl create deployment [name]
        - Edit deployment
            - kubectl edit deployment [name]
        - Delete deployment
            - kubectl delete deployment [name]

    - Status of different K8s components
        - kubectl get nodes I pod | services I replicaset I deployment

    - Debugging pods
        - Log to console - kubectl logs [pod name]
        - Get Interactive Terminal - kubectl exec -it [pod name] -- bin/bash
        - Get info about pod - kubectl describe pod [pod name]

    - Use configuration file for CRUD
        - Apply a configuration file - kubectl apply -f [file name]
        - Delete with kubectl delete -f [file name]