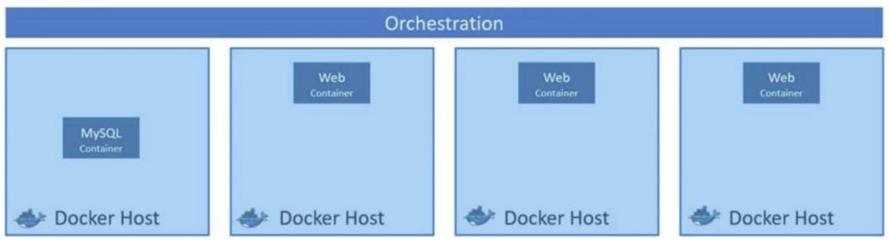
# Kubernetes or k8

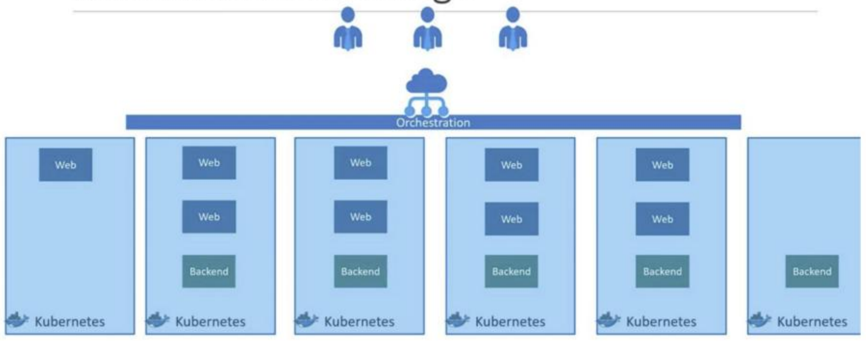
## Container + Orchestration

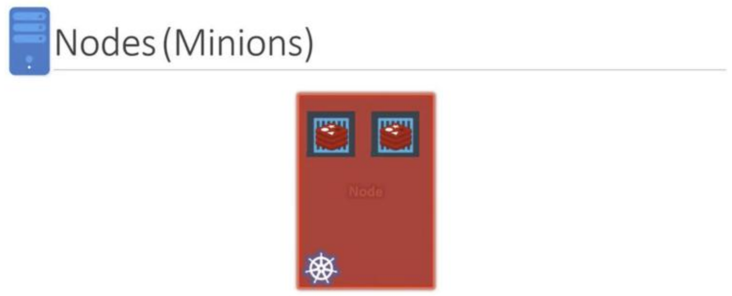
### Container orchestration

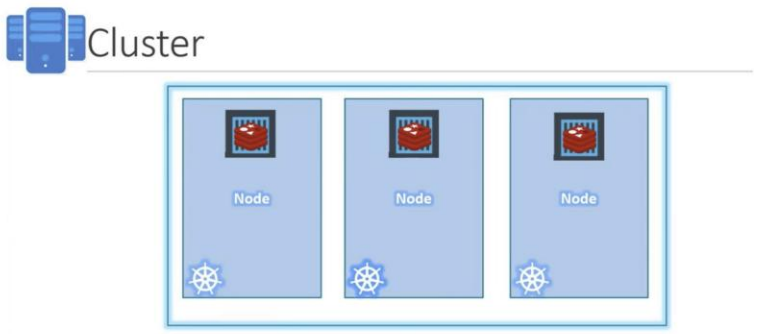


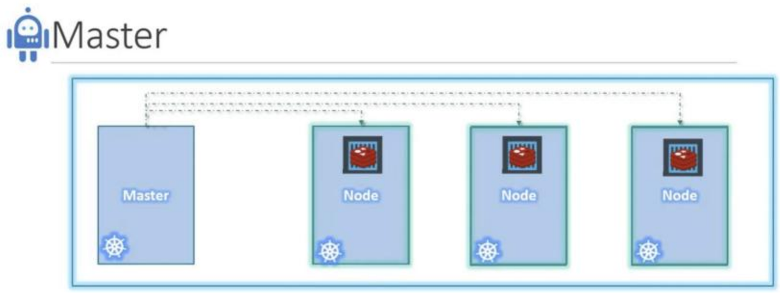
### Orchestration Technologies

### Kubernetes Advantage

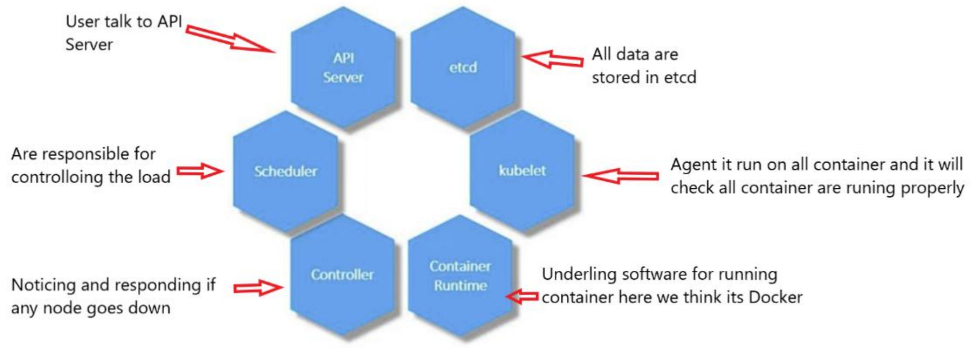



Nodes (Minions)
Node


Cluster
Node
Node
Node


Master
Master
Node
Node
Node

Components

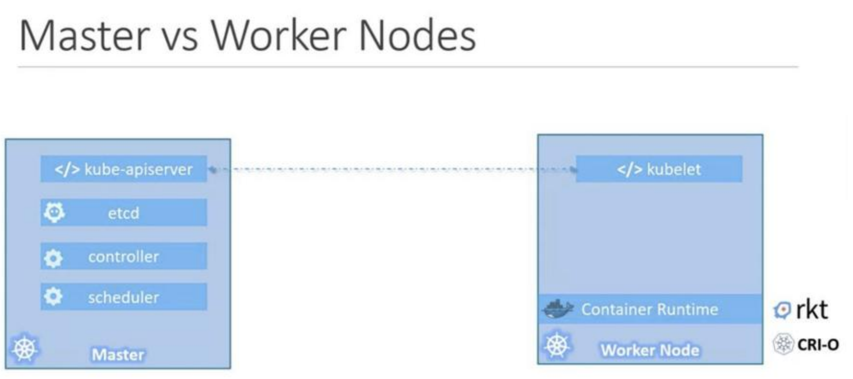

User talk to API Server
API Server
etcd
All data are stored in etcd
Are responsible for controlloing the load
Scheduler
kubelet
Agent it run on all container and it will check all container are runing properly
Noticing and responding if any node goes down
Controller
Container Runtime
Underling software for running container here we think its Docker

Kubernetes is a powerful container orchestration platform that automates the deployment, scaling, and management of containerized applications. It consists of several key components that work together to create a robust and scalable environment for running containers. Here are six essential components of Kubernetes:

1. **Master Node**: The master node is the control plane of the Kubernetes cluster. It manages and coordinates all cluster activities. It includes the following components:
    - **API Server**: The central management point for the Kubernetes cluster. It exposes the Kubernetes API and is the entry point for all administrative tasks and communication with the cluster.
    - **etcd**: A distributed key-value store that stores the configuration and state information of the cluster. It is used as Kubernetes' backing store for all cluster data.
    - **Controller Manager**: Maintains the desired state of resources in the cluster. It watches for changes and adjusts the cluster's state to match the desired state.
    - **Scheduler**: Assigns work (containers) to nodes based on resource availability and requirements.

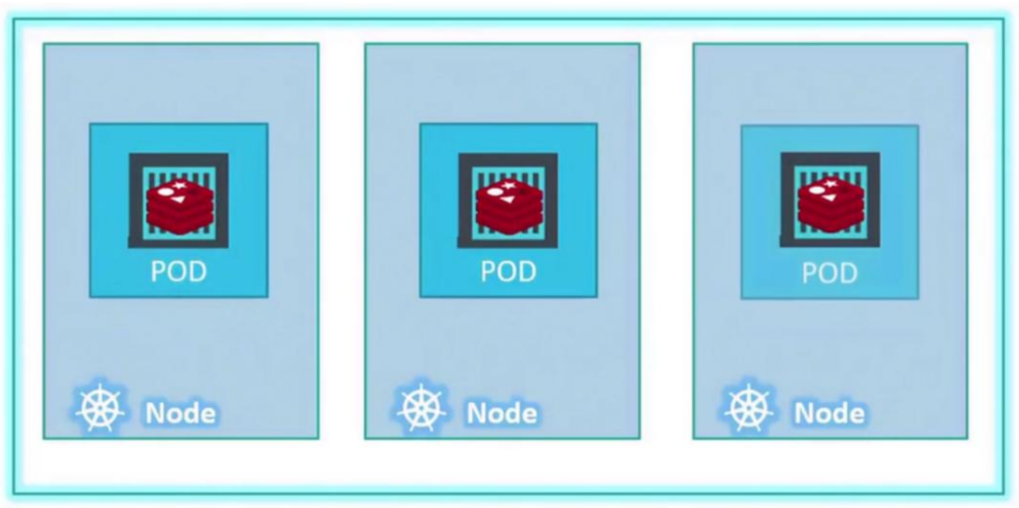
2. **Node** : Nodes are the worker machines in the cluster where containers are deployed and managed. Each node runs the following components:
    - **Kubelet**: Ensures containers are running in a Pod and manages the node's containers, networking, and storage.
    - **Container Runtime**: The software responsible for running containers. Docker and containerd are common container runtimes.
    - **Kube Proxy**: Maintains network rules to route traffic to the appropriate containers.

3. **Pod**: The smallest deployable unit in Kubernetes. A Pod can contain one or more containers that share the same network namespace and storage. Containers in the same Pod can communicate with each other using localhost.
4. **Service**: A way to expose a group of Pods as a network service. Services provide a stable IP address and DNS name, allowing other Pods and external clients to access the application.
5. **Volume**: A directory that exists within a Pod and provides persistent storage. Volumes can be used to store data that needs to persist beyond the lifetime of a container.
6. **Namespace**: A logical grouping mechanism that allows you to create virtual clusters within a physical cluster. Namespaces help organize and isolate resources, making it easier to manage large or multi-tenant clusters.

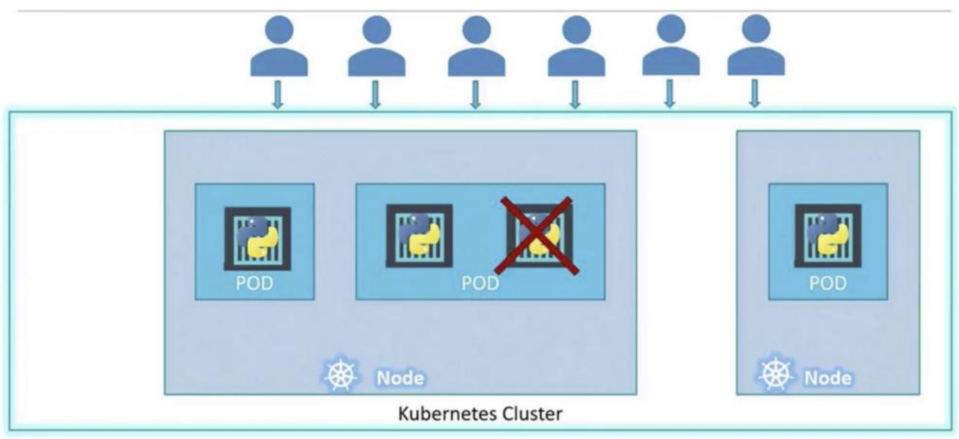

Master vs Worker Nodes

## Pods

Containers are run inside the pods.

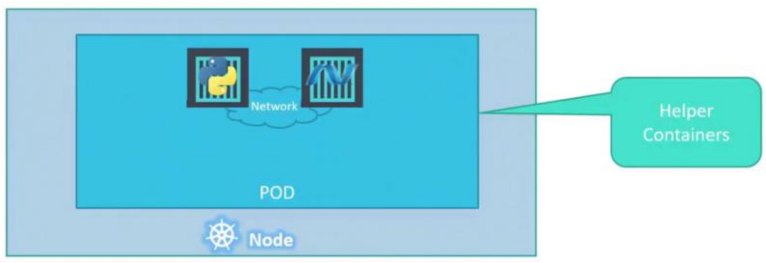


Increase the load
One container is run inside the pod.

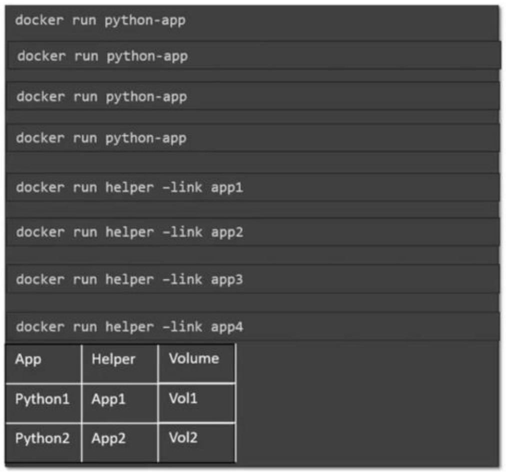


Sometimes one helper container can inside the pod to helps that container

## Multi-Container PODs

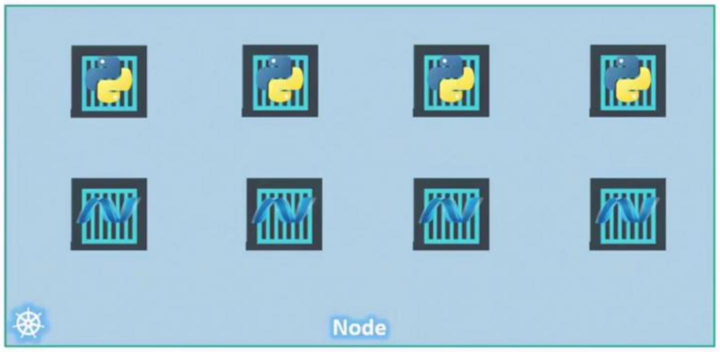


When load increase, we in create the container as we all as helper container and wen load down we need to delete the container as well as delete the helper container also



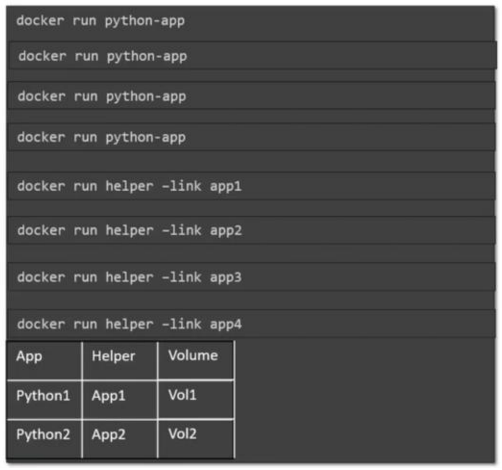
```
docker run python-app
docker run python-app
docker run python-app
docker run python-app
docker run helper -link app1
docker run helper -link app2
docker run helper -link app3
docker run helper -link app4
```

| App | Helper | Volume |
|---|---|---|
| Python1 | App1 | Vol1 |
| Python2 | App2 | Vol2 |



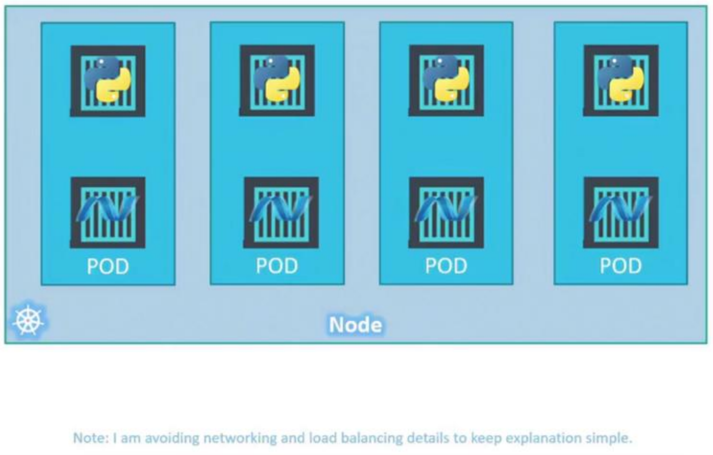
Note: I am avoiding networking and load balancing details to keep explanation simple.

When we use pod



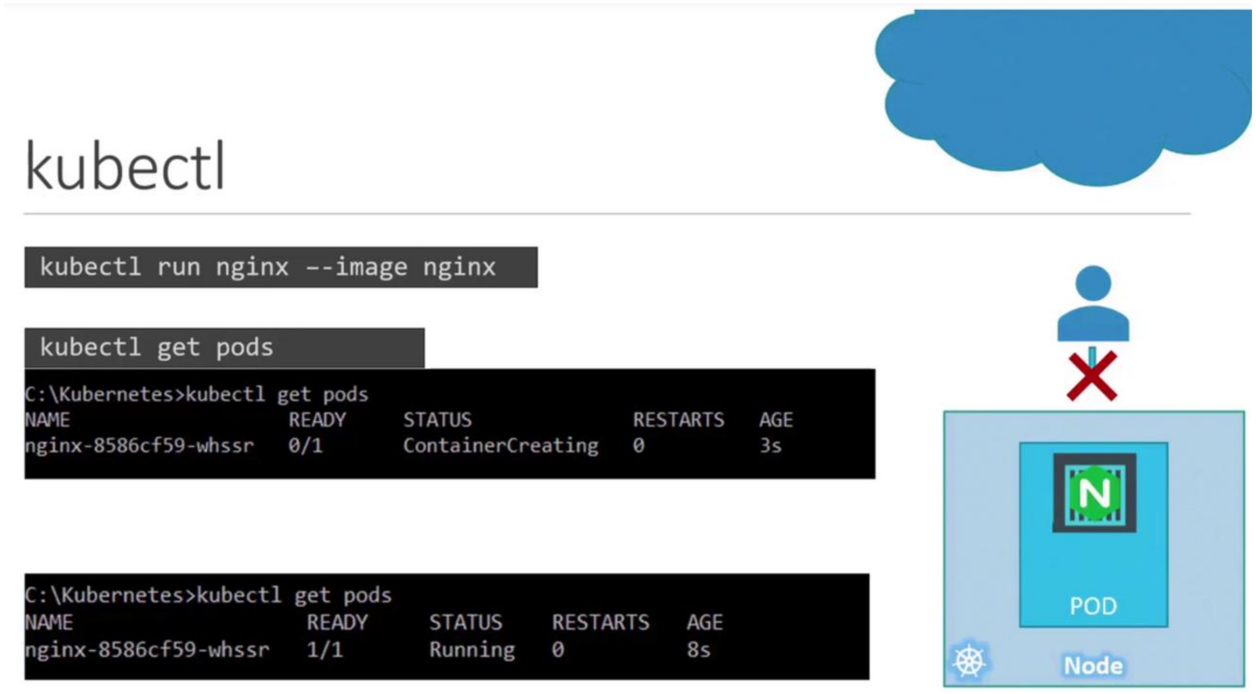
```
docker run python-app
docker run python-app
docker run python-app
docker run python-app
docker run helper -link app1
docker run helper -link app2
docker run helper -link app3
docker run helper -link app4
```

| App | Helper | Volume |
|---|---|---|
| Python1 | App1 | Vol1 |
| Python2 | App2 | Vol2 |



Note: I am avoiding networking and load balancing details to keep explanation simple.

**When run a container:**



kubectl

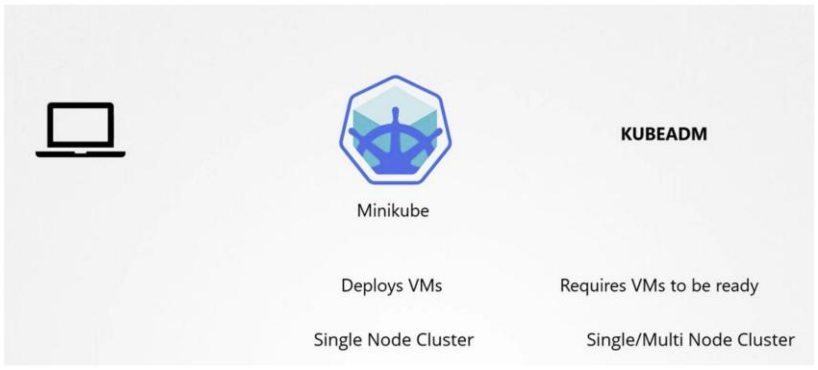
```
kubectl run nginx --image nginx
```

```
kubectl get pods
```

```
C:\Kubernetes>kubectl get pods
NAME                   READY    STATUS              RESTARTS   AGE
nginx-8586cf59-whssr   0/1      ContainerCreating   0          3s
```

```
C:\Kubernetes>kubectl get pods
NAME                   READY    STATUS    RESTARTS   AGE
nginx-8586cf59-whssr   1/1      Running   0          8s
```

## Options available Kubernetes:



Minikube

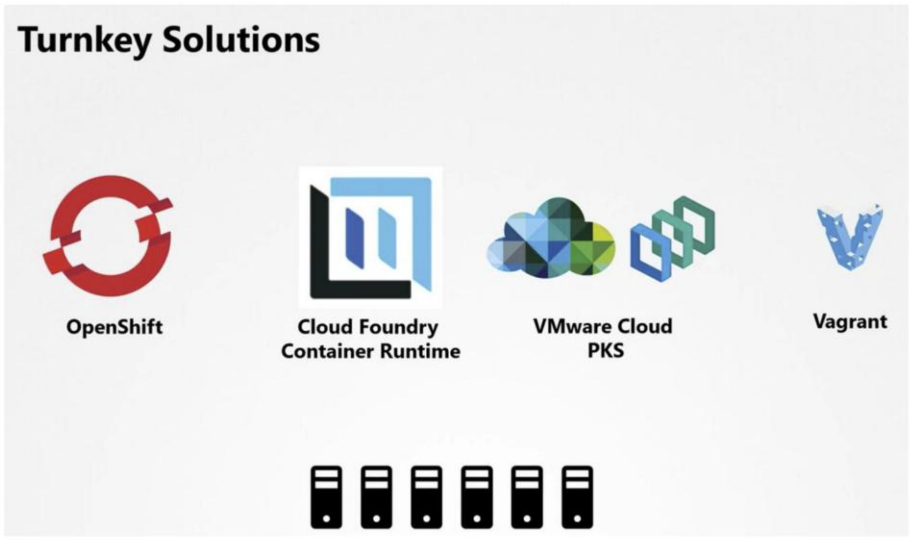
KUBEADM

Deploys VMs

Requires VMs to be ready

Single Node Cluster

Single/Multi Node Cluster



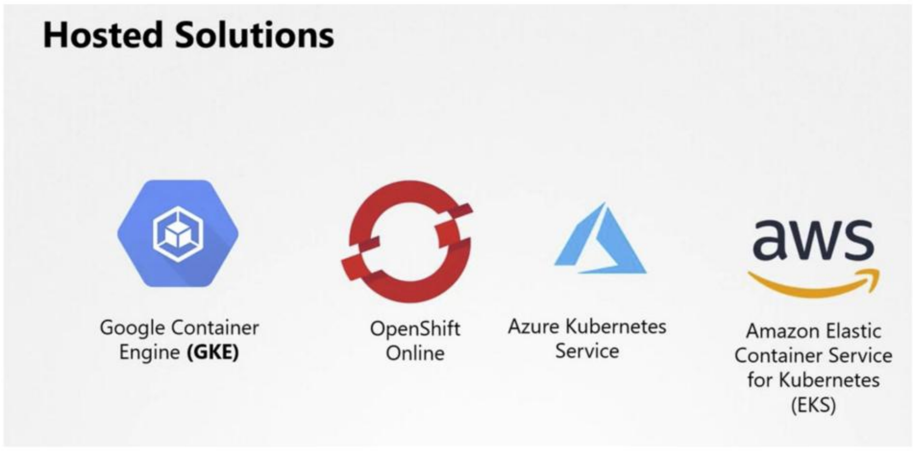
**Turnkey Solutions**

OpenShift

Cloud Foundry Container Runtime

VMware Cloud PKS

Vagrant



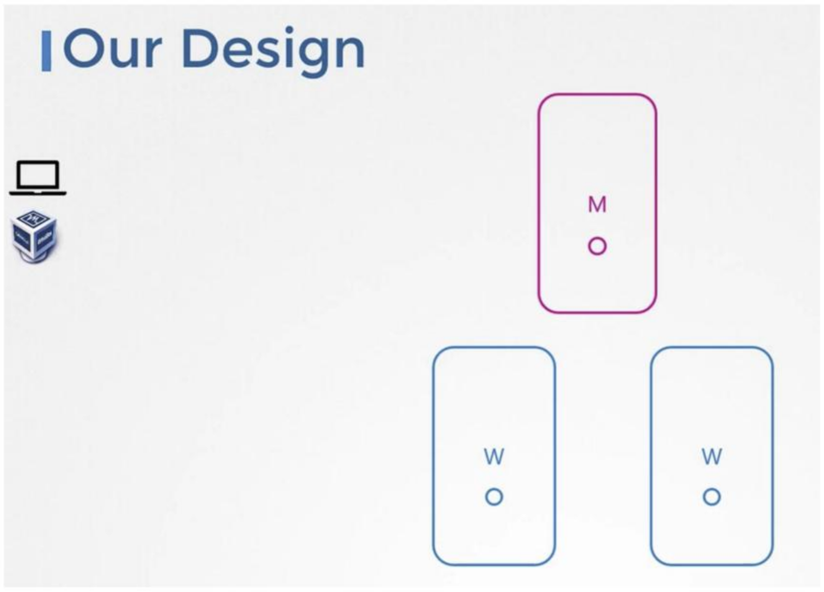
**Hosted Solutions**

Google Container Engine **(GKE)**

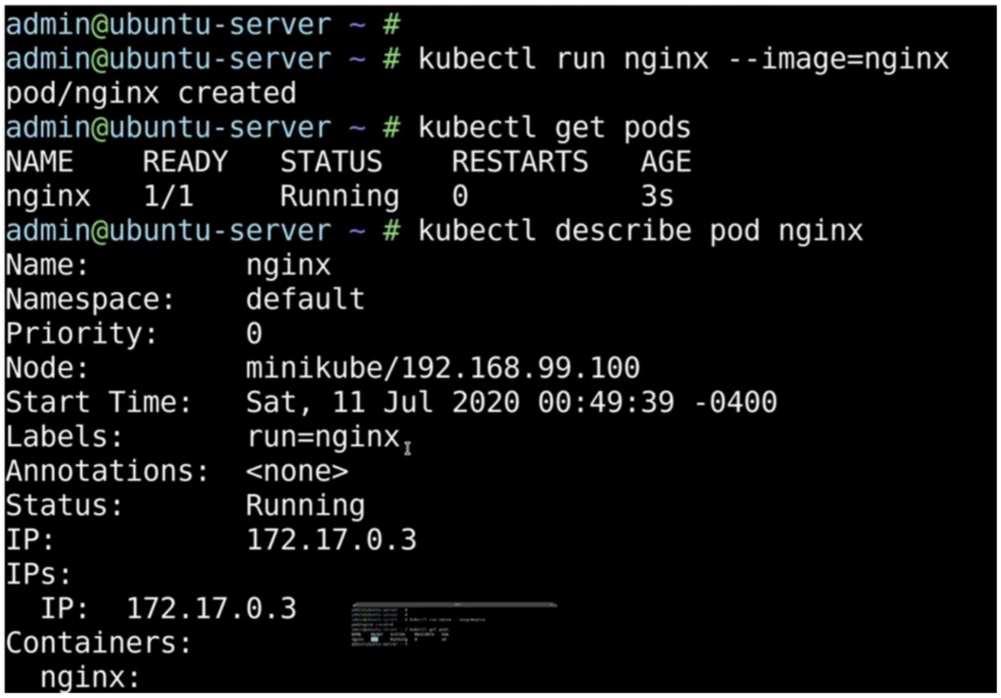
OpenShift Online

Azure Kubernetes Service

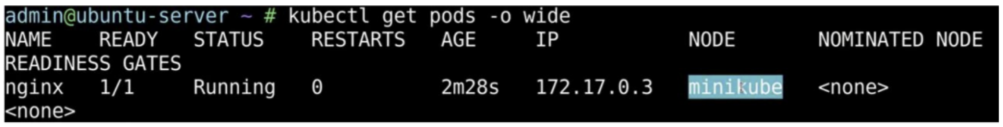
Amazon Elastic Container Service for Kubernetes (EKS)


## Our Design

**Run a container:**



```
admin@ubuntu-server ~ #
admin@ubuntu-server ~ # kubectl run nginx --image=nginx
pod/nginx created
admin@ubuntu-server ~ # kubectl get pods
NAME     READY    STATUS     RESTARTS    AGE
nginx    1/1      Running    0           3s
admin@ubuntu-server ~ # kubectl describe pod nginx
Name:          nginx
Namespace:     default
Priority:      0
Node:          minikube/192.168.99.100
Start Time:    Sat, 11 Jul 2020 00:49:39 -0400
Labels:        run=nginx
Annotations:   <none>
Status:        Running
IP:            172.17.0.3
IPs:
  IP:  172.17.0.3
Containers:
  nginx:
```

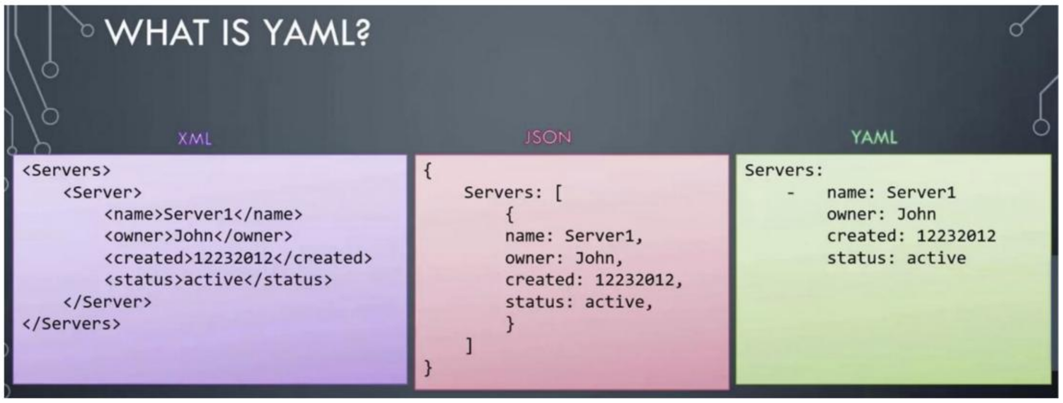


```
admin@ubuntu-server ~ # kubectl get pods -o wide
NAME     READY   STATUS     RESTARTS   AGE     IP           NODE       NOMINATED NODE
READINESS GATES
nginx    1/1     Running    0          2m28s   172.17.0.3   minikube   <none>
<none>
```

URL
https://kubernetes.io/docs/concepts/

https://kubernetes.io/docs/concepts/workloads/pods/

YAML



## WHAT IS YAML?

XML

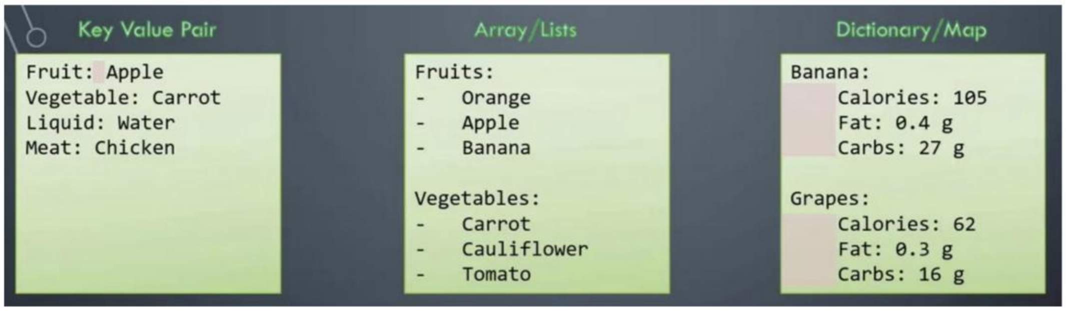
```
<Servers>
    <Server>
        <name>Server1</name>
        <owner>John</owner>
        <created>12232012</created>
        <status>active</status>
    </Server>
</Servers>
```

JSON

```
{
    Servers: [
        {
        name: Server1,
        owner: John,
        created: 12232012,
        status: active,
        }
    ]
}
```

YAML

```
Servers:
    -   name: Server1
        owner: John
        created: 12232012
        status: active
```



Key Value Pair

```
Fruit: Apple
Vegetable: Carrot
Liquid: Water
Meat: Chicken
```

Array/Lists

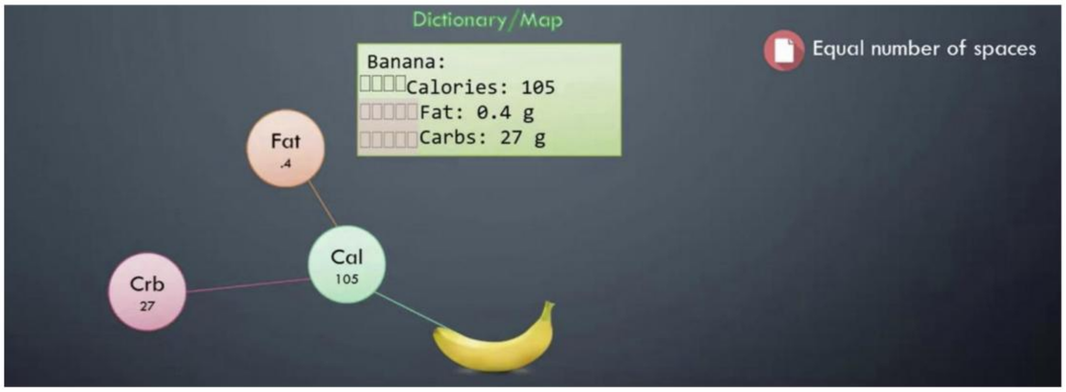
```
Fruits:
-    Orange
-    Apple
-    Banana

Vegetables:
-    Carrot
-    Cauliflower
-    Tomato
```

Dictionary/Map

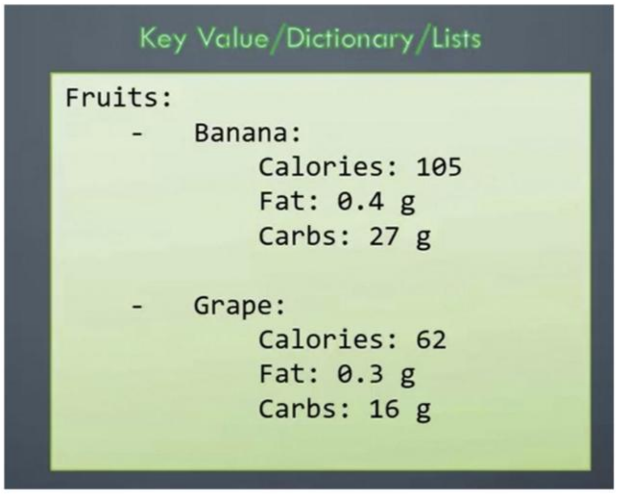
```
Banana:
    Calories: 105
    Fat: 0.4 g
    Carbs: 27 g

Grapes:
    Calories: 62
    Fat: 0.3 g
    Carbs: 16 g
```



Dictionary/Map

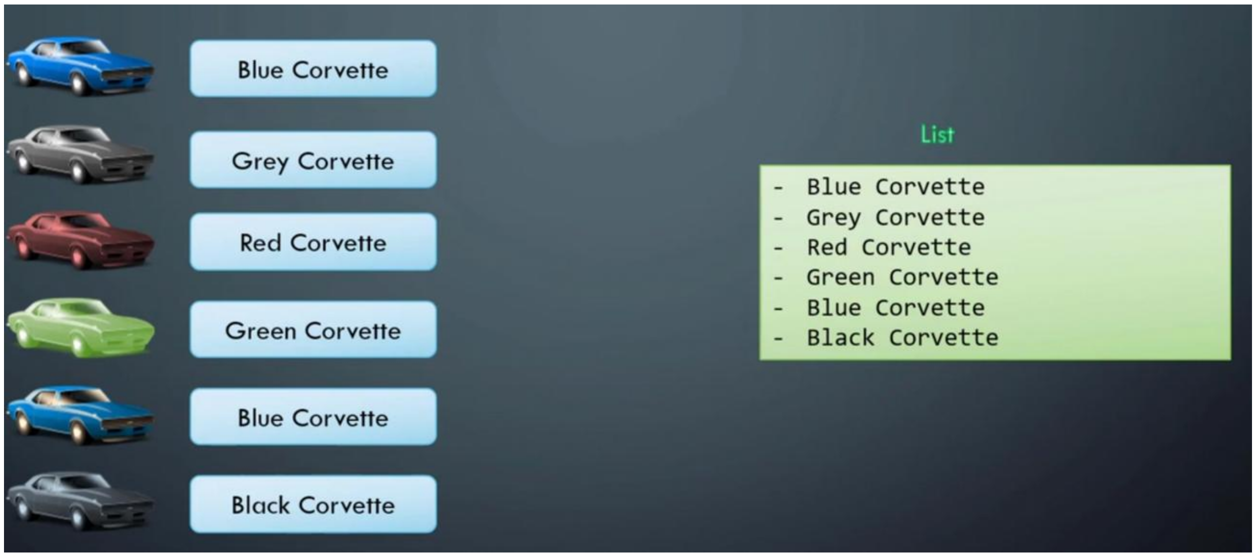
```
Banana:
    Calories: 105
     Fat: 0.4 g
     Carbs: 27 g
```



Key Value/Dictionary/Lists

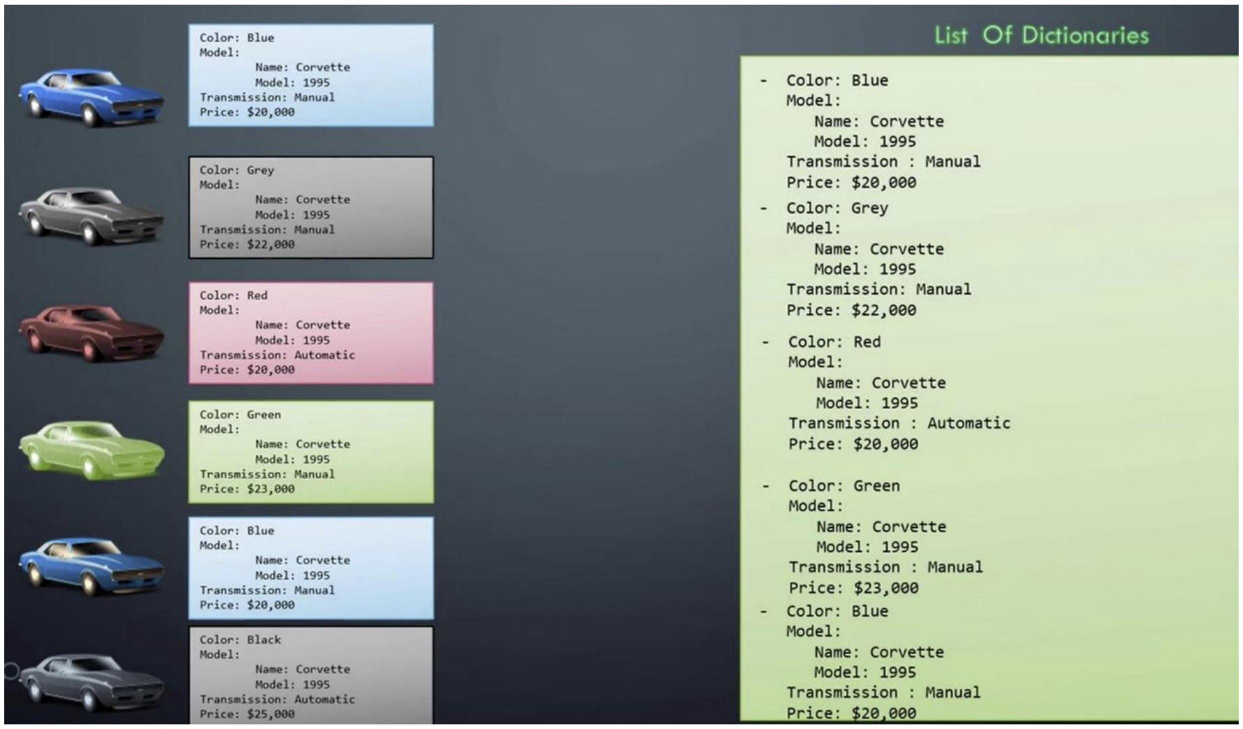
```
Fruits:
    -   Banana:
            Calories: 105
            Fat: 0.4 g
            Carbs: 27 g

    -   Grape:
            Calories: 62
            Fat: 0.3 g
            Carbs: 16 g
```



List

```
-  Blue Corvette
-  Grey Corvette
-  Red Corvette
-  Green Corvette
-  Blue Corvette
-  Black Corvette
```



```
Color: Blue
Model:
        Name: Corvette
        Model: 1995
Transmission: Manual
Price: $20,000
```

```
Color: Grey
Model:
        Name: Corvette
        Model: 1995
Transmission: Manual
Price: $22,000
```

```
Color: Red
Model:
        Name: Corvette
        Model: 1995
Transmission: Automatic
Price: $20,000
```

```
Color: Green
Model:
        Name: Corvette
        Model: 1995
Transmission: Manual
Price: $23,000
```

```
Color: Blue
Model:
        Name: Corvette
        Model: 1995
Transmission: Manual
Price: $20,000
```

```
Color: Black
Model:
        Name: Corvette
        Model: 1995
Transmission: Automatic
Price: $25,000
```

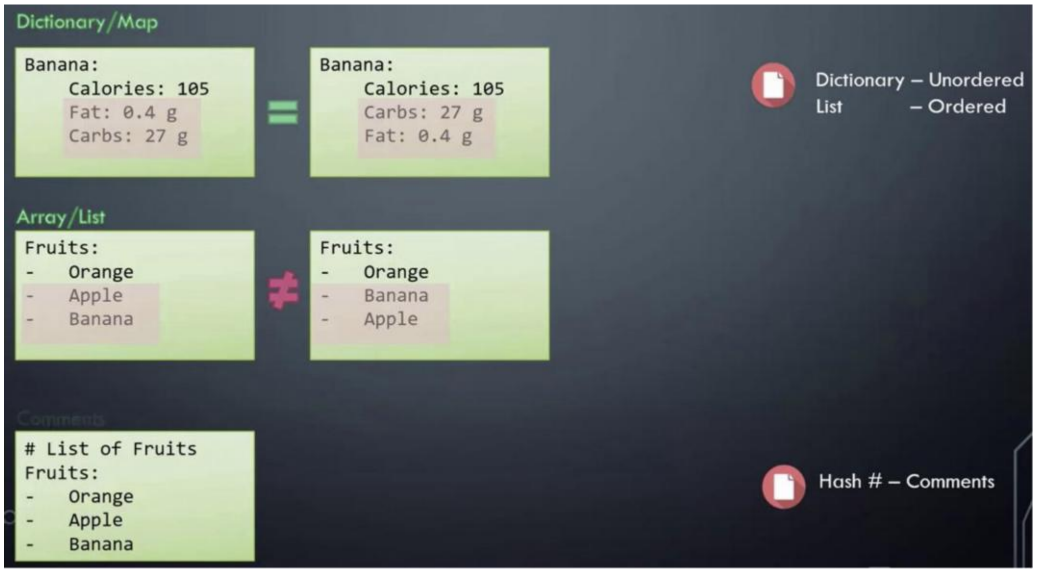
List Of Dictionaries

```
-  Color: Blue
   Model:
      Name: Corvette
      Model: 1995
   Transmission : Manual
   Price: $20,000
-  Color: Grey
   Model:
      Name: Corvette
      Model: 1995
   Transmission: Manual
   Price: $22,000

-  Color: Red
   Model:
      Name: Corvette
      Model: 1995
   Transmission : Automatic
   Price: $20,000

-  Color: Green
   Model:
      Name: Corvette
      Model: 1995
   Transmission : Manual
   Price: $23,000
-  Color: Blue
   Model:
      Name: Corvette
      Model: 1995
   Transmission : Manual
   Price: $20,000
```

Directory and array/list ordering



```
Banana:
    Calories: 105
    Fat: 0.4 g
    Carbs: 27 g
```

```
Banana:
    Calories: 105
    Carbs: 27 g
    Fat: 0.4 g
```

```
Fruits:
-   Orange
-   Apple
-   Banana
```

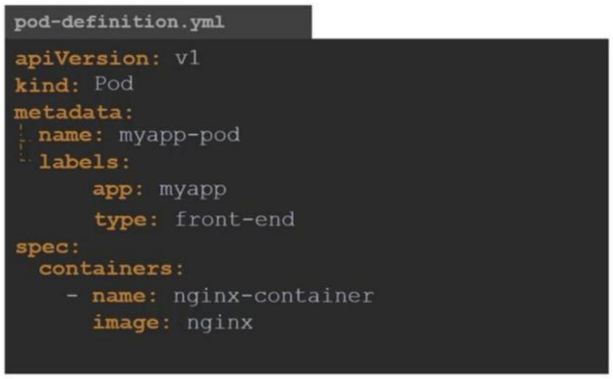
```
Fruits:
-   Orange
-   Banana
-   Apple
```

```
# List of Fruits
Fruits:
-   Orange
-   Apple
-   Banana
```

Dictionary – Unordered
List – Ordered

Hash # – Comments

## YAML in Kubernetes

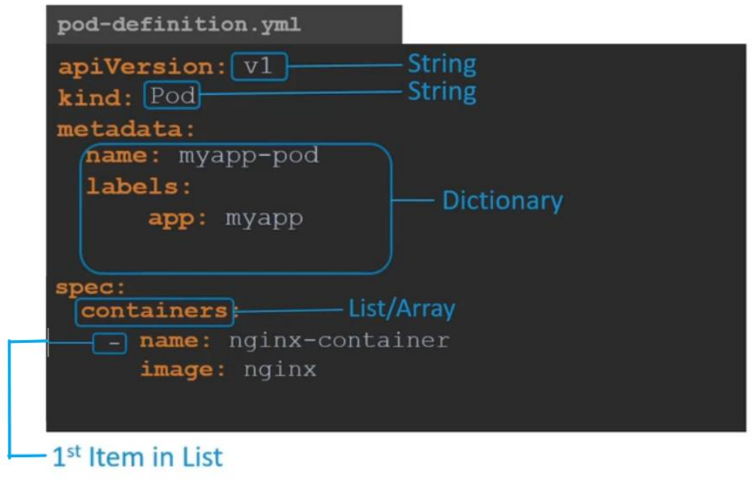


pod-definition.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: myapp-pod
  labels:
       app: myapp
       type: front-end
spec:
  containers:
    - name: nginx-container
      image: nginx
```

| Kind | Version |
|---|---|
| POD | v1 |
| Service | v1 |
| ReplicaSet | apps/v1 |
| Deployment | apps/v1 |
| | |

```
kubectl create -f pod-definition.yml
```



pod-definition.yml

```
apiVersion: v1
kind: Pod
metadata:
  name: myapp-pod
  labels:
       app: myapp
spec:
  containers:
    - name: nginx-container
      image: nginx
```

| Kind | Version |
|---|---|
| POD | v1 |
| Service | v1 |
| ReplicaSet | apps/v1 |
| Deployment | apps/v1 |
| | |

In Kubernetes, a Pod is the smallest deployable unit that represents a collection of one or more containers that share networking and storage resources. The definition of a Pod is typically described in a YAML file, which specifies its configuration and the containers it should run. Here's a brief description of the components in a Kubernetes Pod definition YAML file:

**API Version and Kind:**

- **apiVersion**: Specifies the version of the Kubernetes API used to create the object.
- **kind**: Defines the type of object being created, in this case, **Pod**.

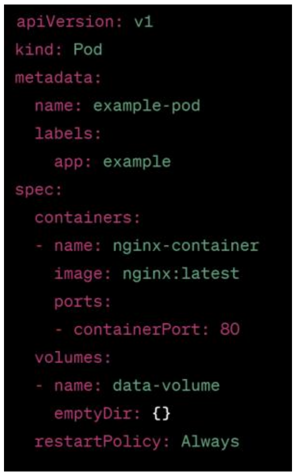
**Metadata:**

- **metadata**: Contains information about the Pod such as its name, labels, annotations, and namespace.

**Pod Specification:**

- **spec**: Describes the desired state for the Pod.
    - **containers**: Specifies the list of containers to be launched within the Pod.
        - **name**: Name of the container.
        - **image**: Container image to be used.
        - **ports**: Exposes specific ports for communication.
        - **env**: Defines environment variables for the container.
        - **volumeMounts**: Mounts storage volumes into the container's filesystem.
    - **volumes**: Defines the list of volumes that can be mounted by containers within the Pod.
    - **restartPolicy**: Defines the Pod's restart policy ('Always', 'OnFailure', or 'Never').
    - **dnsPolicy**: Specifies how the Pod should resolve DNS names.

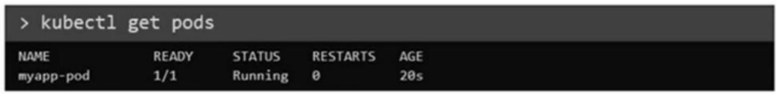
**Example YAML Structure:**



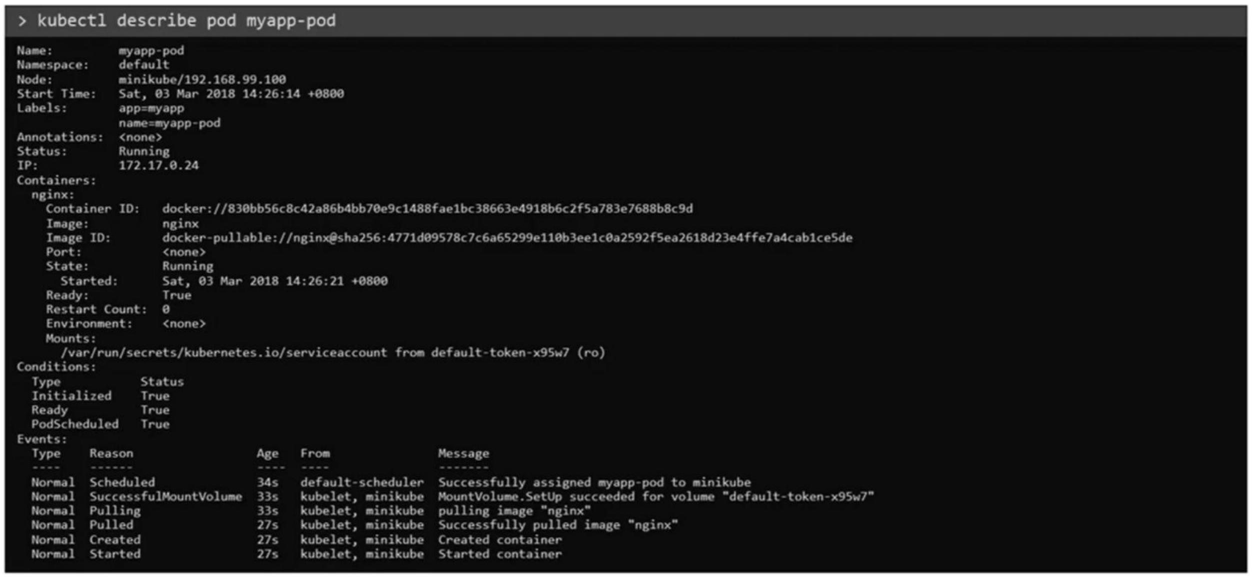
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: example-pod
  labels:
    app: example
spec:
  containers:
  - name: nginx-container
    image: nginx:latest
    ports:
    - containerPort: 80
  volumes:
  - name: data-volume
    emptyDir: {}
  restartPolicy: Always
```

This is a basic example of a Pod definition YAML file. It describes a Pod named **example-pod** running an **nginx** container, exposing port 80, and with an associated empty volume named **data-volume**. Adjustments and additional configurations can be made based on specific requirements and the desired behavior of the Pod.

## Command



```
> kubectl get pods
NAME         READY    STATUS    RESTARTS   AGE
myapp-pod    1/1      Running   0          20s
```



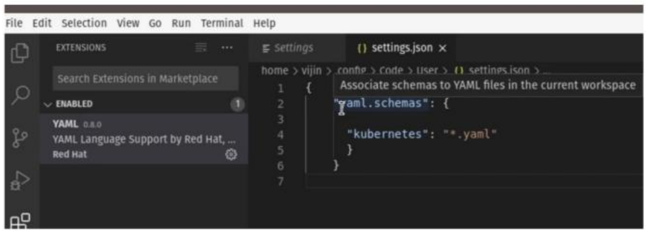
```
> kubectl describe pod myapp-pod
Name:          myapp-pod
Namespace:     default
Node:          minikube/192.168.99.100
Start Time:    Sat, 03 Mar 2018 14:26:14 +0800
Labels:        app=myapp
               name=myapp-pod
Annotations:   <none>
Status:        Running
IP:            172.17.0.24
Containers:
  nginx:
    Container ID:   docker://830bb56c8c42a86b4bb70e9c1488fae1bc38663e4918b6c2f5a783e7688b8c9d
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:4771d09578c7c6a65299e110b3ee1c0a2592f5ea2618d23e4ffe7a4cab1ce5de
    Port:           <none>
    State:          Running
      Started:      Sat, 03 Mar 2018 14:26:21 +0800
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-x95w7 (ro)
Conditions:
  Type           Status
  Initialized    True
  Ready          True
  PodScheduled   True
Events:
  Type    Reason                 Age   From               Message
  ----    ------                 ----  ----               -------
  Normal  Scheduled              34s   default-scheduler  Successfully assigned myapp-pod to minikube
  Normal  SuccessfulMountVolume  33s   kubelet, minikube  MountVolume.SetUp succeeded for volume "default-token-x95w7"
  Normal  Pulling                33s   kubelet, minikube  pulling image "nginx"
  Normal  Pulled                 27s   kubelet, minikube  Successfully pulled image "nginx"
  Normal  Created                27s   kubelet, minikube  Created container
  Normal  Started                27s   kubelet, minikube  Started container
```

Sample nginx pod create by yaml file

```
apiVersion: v1
kind: Pod
metadata:
  name: nginx
spec:
  containers:
  - name: nginx
    image: nginx:1.14.2
    ports:
    - containerPort: 80
```

## Editor for creating yaml file

Visual studio code download URL for all type of OS
https://code.visualstudio.com/download
Schema need to add in VS code for kubernetes


File Edit Selection View Go Run Terminal Help
EXTENSIONS
Search Extensions in Marketplace
ENABLED 1
YAML 0.8.0
YAML Language Support by Red Hat, ...
Red Hat
Settings
settings.json
home > vijin > .config > Code > User > settings.json > ...
Associate schemas to YAML files in the current workspace

```
{
    "yaml.schemas": {

     "kubernetes": "*.yaml"
     }
    }

```