# Kubernetes – Pods

A pod is the smallest unit that exists in Kubernetes. It is similar to that of tokens in [C](https://www.geeksforgeeks.org/c-programming-language/) or [C++](https://www.geeksforgeeks.org/c-plus-plus/) language. A specific pod can have one or more applications. The nature of Pods is ephemeral this means that in any case if a pod fails then Kubernetes can and will automatically create a new replica/ duplicate of the said pod and continue the operation. The pods have the capacity to include one or more containers based on the requirement. The containers can even be [Docker containers.](https://www.geeksforgeeks.org/introduction-to-docker/)The Pods in Kubernetes provide environmental dependencies which include[persistent storage volumes](https://www.geeksforgeeks.org/kubernetes-volumes/)which means it is permanent and is available to all pods in the said cluster and even configuration data that is required to run the container within the pod.

A picture containing diagram, text, circle

Description automatically generated

## What does Pod do?

A pod in a Kubernetes[cluster](https://www.geeksforgeeks.org/kubernetes-architecture/) indicates a process that is currently operating, and a pod may contain one or more containers. All of those containers share a single IP address, as well as the pod’s storage, network, and any other requirements. A pod is a collection of one or more running containers, allowing for simple container movement within a cluster.

The creation of a pod is due to a workload resource called controller, which means rollout, replicate, and health of the pods present in a cluster. If we consider that a node in a cluster fails then a controller detects that the pod on the mode is unresponsive and then replicates a pod or pods on other nodes to carry out the same function. The three mostly used controllers used are Jobs, Deployments, and Stateful Sets. Jobs are used for batch-type jobs that are mostly ephemeral and will run a task to completion. Deployments are used for applications that are stateless and persistent, for example, web services. StatefulSets is used for applications that are both stateful and Persistent like a database.

If any pod has any/multiple containers then all those are scheduled together on the same server in the cluster either a physical server or VM. All the containers present in the pods will share their resources and dependencies. All these clusters can coordinate their termination and execution.  For instance, if a pod contains an init container then it runs before the application container runs leveling or setting up the required environment for applications to follow.  Generally, pods are created by controllers that can automatically manage the pod lifecycle. The pod life cycle included replacing failed pods, replicating the pods when necessary, and eliminating the pod once the purpose was completed. Controllers use the information present in the pod templates to create the pods.

## Pod Lifecycle

We submit a pod request to the API server using the pod definition file, and the API server records the information for the pod in ETCD. The scheduler will schedule the unscheduled pod to the available node as soon as it is discovered. When a pod is scheduled, a running Kubelet on a node launches Docker, which then launches the container.

*The entire pod lifecycle stores in ETCD*

## Using Pods

Let’s create a [YAML](https://www.geeksforgeeks.org/yaml-comments/) file for an example image [(Nginx)](https://www.geeksforgeeks.org/what-is-nginx-web-server-and-how-to-install-it/) so that it can be deployed as a container.

apiVersion: v1

Kind: pod

metadata:

name: nginx

spec:

containers:

- name: nginx

image:

nginx:latest

ports:

- containers: 80

After creating the yaml file run the below command to deploy it as a container.

kubectl apply -f <name of yaml file>

## Types of Pods

* Single Container Pod
* [Multi Container Pod](https://www.geeksforgeeks.org/kubernetes-creating-multiple-container-in-a-pod/)
* Static Pod

A pod can be defined as the collection of containers and their storage inside a node of a Kubernetes cluster. There is a possibility to create a pod with multiple containers inside them.

Based on the number of pods present inside them they can be classified as a **Single Container Pod** or **Multi Container Pod**.

As the name suggests a single container contains only one container whereas the multi-container pod contains multiple containers. They can be used based on their function and use a case at the respective times. The methods of creation of both types of pods are different.

## How To Scheduled a Pod?

The pod will not be rescheduled to a different node when it expires because it is ephemeral (lasting only a very brief time). , we shouldn’t directly construct or use pods; instead, we should deploy pod-like deployment, replica sets, and daemon sets to maintain the pod with the aid of Kubernetes services. We have to deploy the pods with the help of objects. The main objects are

* [Deployment.](https://www.geeksforgeeks.org/kubernetes-deployments/)
* [Replication Controller.](https://www.geeksforgeeks.org/kubernetes-replication-controller/)
* [Replica Set.](https://www.geeksforgeeks.org/kubernetes-creating-a-replicaset/)
* DemonSet.

## Characteristics of Pods

A Pod represents the processes running on a cluster. If one pod is limited to a single process then it is possible to report and maintain the health of each process running within the cluster. Every Pod has some unique features like a Unique IP address, persistent storage volumes, and configuration information required to run the working. Mostly all the pods have a single container but many of them will have a few containers working closely together to execute a particular function or activity.

## Benefits of Pods

* If a pod contains many containers working towards a common goal then it is easy for them to communicate and share data among themselves.
* We know that all the containers in a pod will have the same network namespace due to which they can locate each other and communicate with the help o localhost.
* Pods can communicate with each other by using another pod’s IP address or even by referring to a resource that is located in another pod.
* Any pod can even include containers that run when the pod is started mainly to run any operation before the application containers run.
* The presence of pods has made it more Scalable as each pod and its replicas can be created and shut down automatically considering the changes in demand.

## How do Pods Communicate With Each Other?

The creation of a pod has made it easy for communication between various components. If a pod contains multiple containers then they can communicate with each other by using a local host. Communication with outside pods can be made by exposing a pod. Communication within the clusters of the same pod is easy because Kubernetes assign a cluster private IP address to each pod in a cluster.

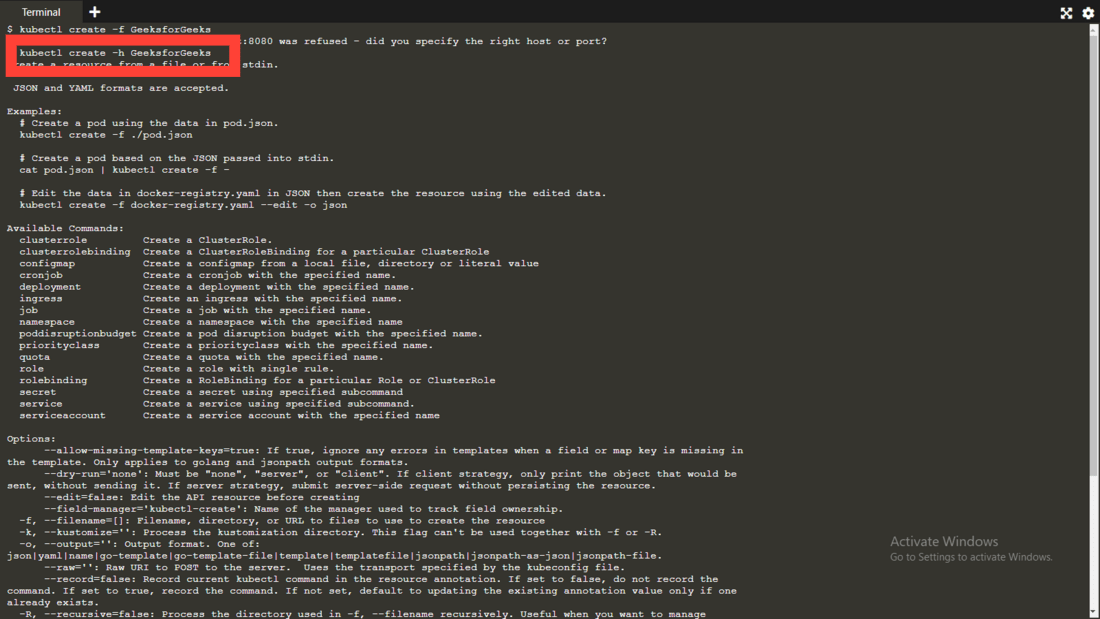
## What are the Basic Kubectl Commands for Pods?

 A pod can be created by using the create command format.

### Create Pod

$ kubectl create -f [FILENAME]

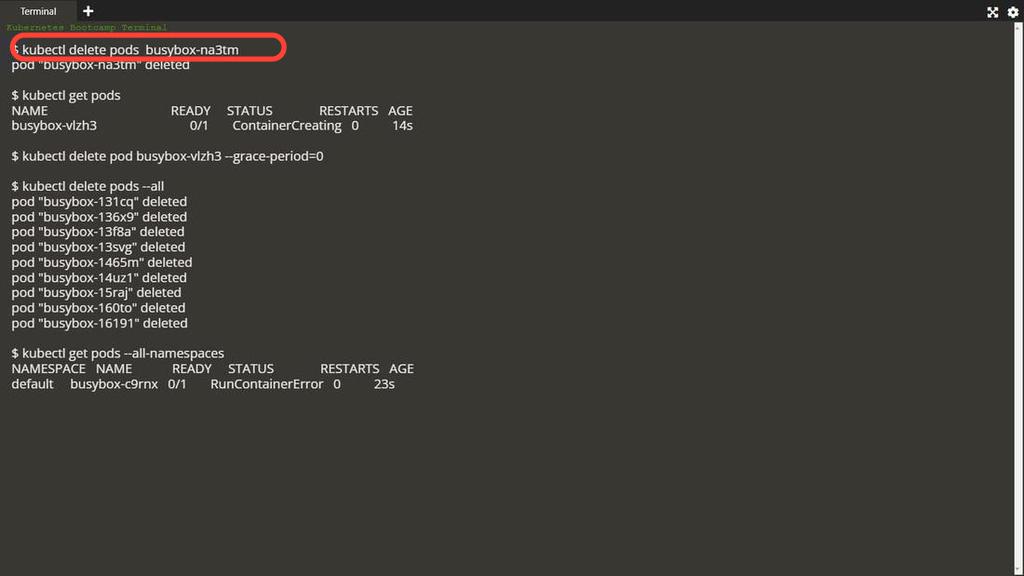
In the [FILENAME], you need to insert the required filename with which you want to create your file, then a new pod with the name GeeksforGeeks will be created.



### Delete Pod

$ kubectl delete -f FILENAME

here the pod named GeeksforGeeks will be deleted



### Get The Pod

To see the no.of pods available in the particular namespace can be seen using the below command.

Kubectl get pod <name> --namespace