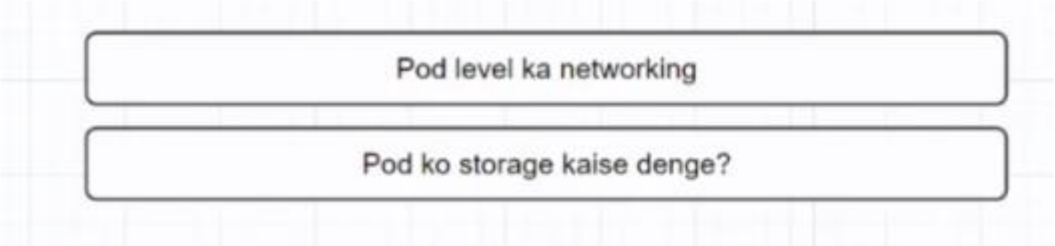
19 October 2024

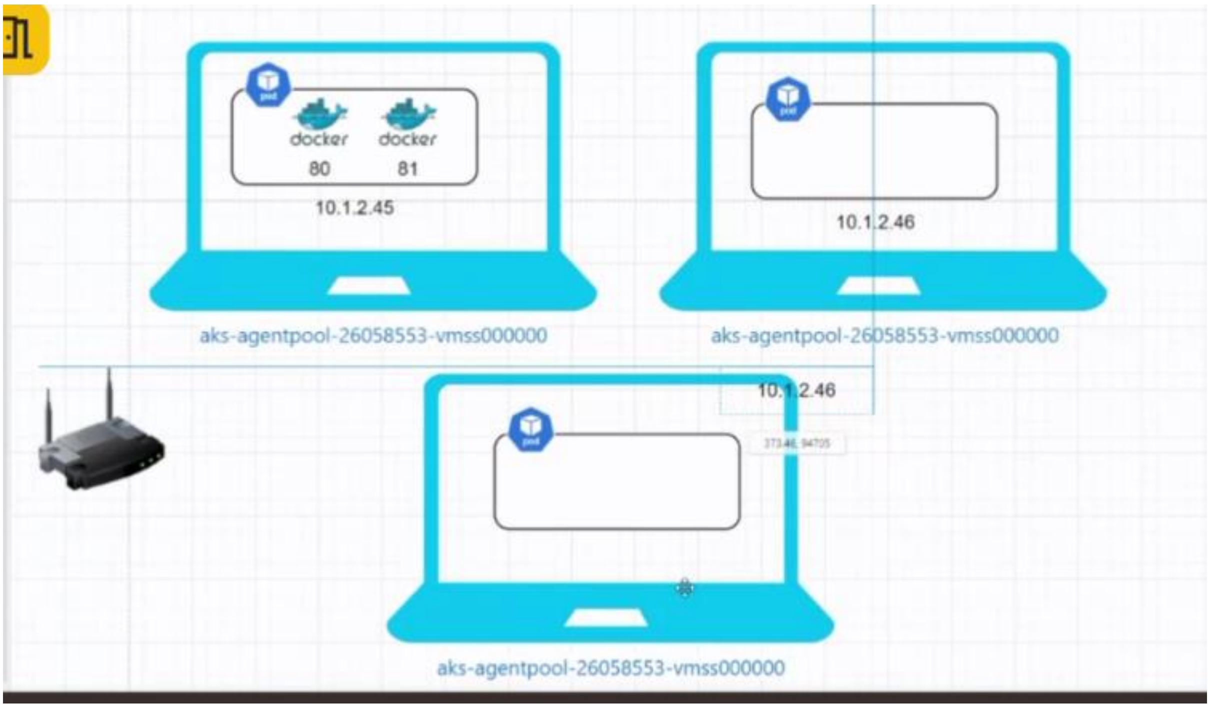
AGENDA – Create 2 containers inside a pod

Pod is just like a virtual machine.



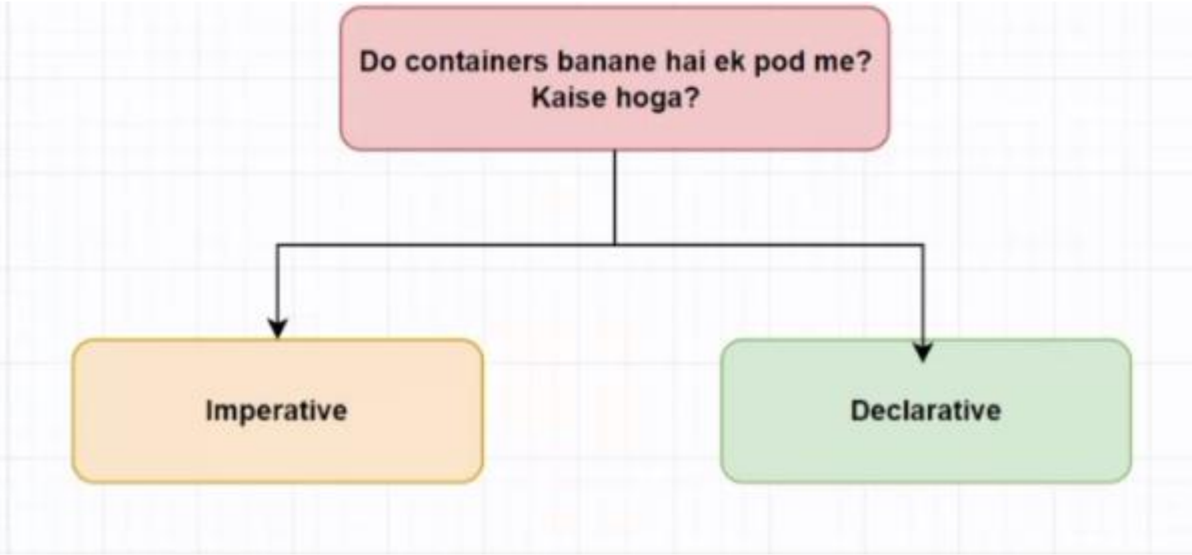
Pod level ka networking

Pod ko storage kaise denge?


docker
docker
80
81
10.1.2.45
aks-agentpool-26058553-vmss000000
10.1.2.46
aks-agentpool-26058553-vmss000000
10.1.2.46
aks-agentpool-26058553-vmss000000

1) Every pod is assigned an individual IP address

2) How 2 containers inside 1 pod can communicate to each other? Through local host



3) 3 formats of sending data from 1 computer to another computer are

i) JSON

ii) YAML

iii) PROTOBUF

4) Yml is mentioned as key value pair

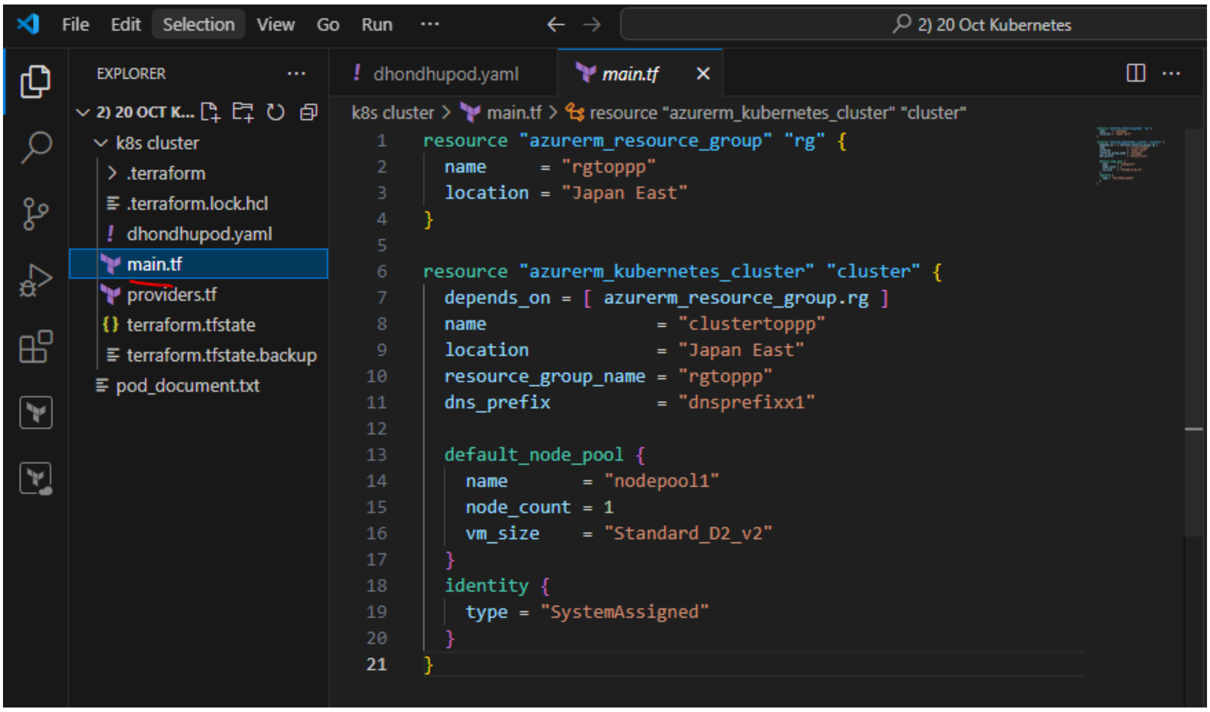
++

AGENDA – Create kubernetes cluster using terraform code

1) Create folder "2) 20 Oct Kubernetes" and open it with vscode.

2) Create folder "k8s cluster" and create main.tf and providers.tf file

3) In main.tf file write code of rg and k8s cluster



```
resource "azurerm_resource_group" "rg" {
  name     = "rgtoppp"
  location = "Japan East"
}

resource "azurerm_kubernetes_cluster" "cluster" {
  depends_on = [ azurerm_resource_group.rg ]
  name                = "clustertoppp"
  location            = "Japan East"
  resource_group_name = "rgtoppp"
  dns_prefix          = "dnsprefixx1"

  default_node_pool {
    name       = "nodepool1"
    node_count = 1
    vm_size    = "Standard_D2_v2"
  }
  identity {
    type = "SystemAssigned"
  }
}
```

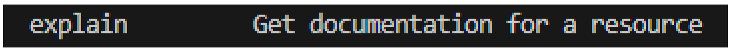
4) Run terraform init, validate, fmt, az login, plan, apply.

++

AGENDA – Create dhondhupod file

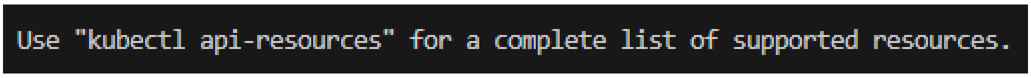
5) Now writing dhondhupod.yaml file in vscode.

Kubectl



```
 explain         Get documentation for a resource
```

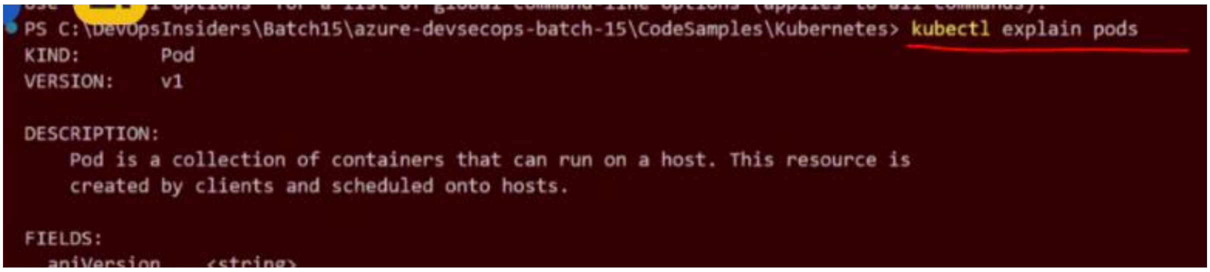
kubectl explain --help



```
Use "kubectl api-resources" for a complete list of supported resources.
```

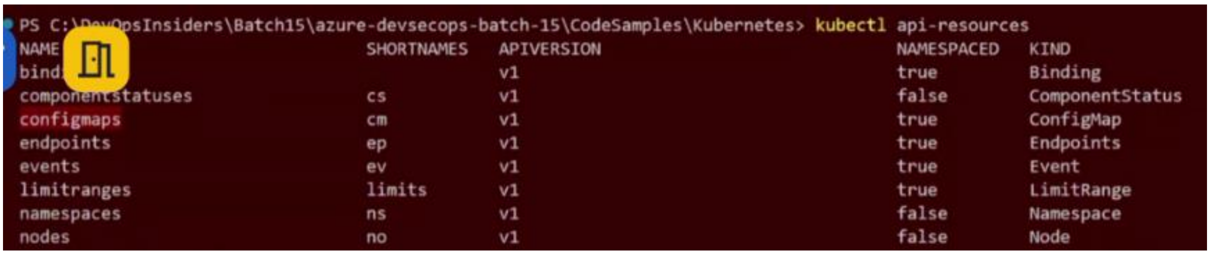
6) Now make whole cluster and connect as like last class

Kubectl explain pods – command for document of pod



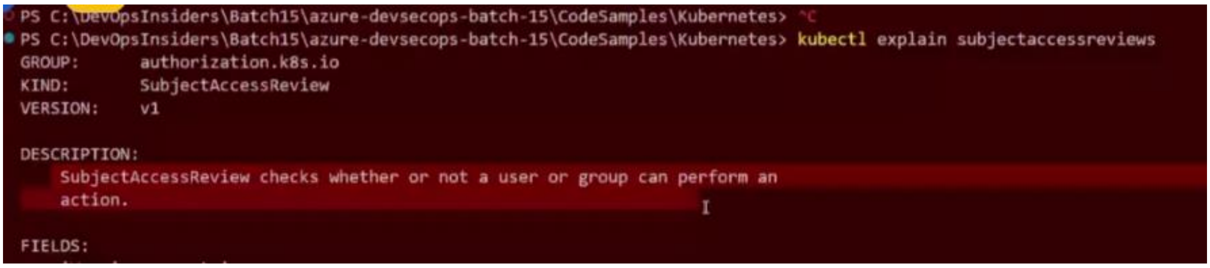
```
Use "kubectl options" for a list of global command-line options (applies to all commands).
PS C:\DevOpsInsiders\Batch15\azure-devsecops-batch-15\CodeSamples\Kubernetes> kubectl explain pods
KIND:     Pod
VERSION:  v1

DESCRIPTION:
    Pod is a collection of containers that can run on a host. This resource is
    created by clients and scheduled onto hosts.

FIELDS:
  apiVersion    <string>
```

7) **kubectl api-resources** – this command gives us the list of all resources



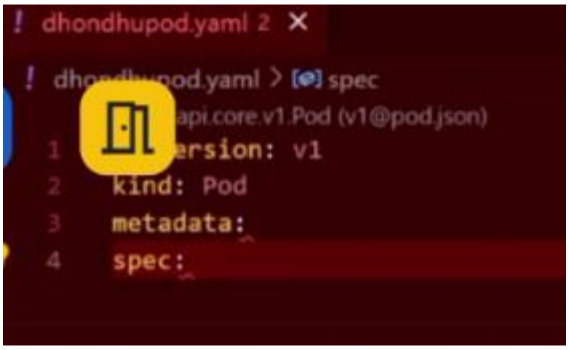
```
PS C:\DevOpsInsiders\Batch15\azure-devsecops-batch-15\CodeSamples\Kubernetes> kubectl api-resources
NAME                              SHORTNAMES   APIVERSION                      NAMESPACED   KIND
bind                                           v1                              true         Binding
componentstatuses                 cs           v1                              false        ComponentStatus
configmaps                        cm           v1                              true         ConfigMap
endpoints                         ep           v1                              true         Endpoints
events                            ev           v1                              true         Event
limitranges                       limits       v1                              true         LimitRange
namespaces                        ns           v1                              false        Namespace
nodes                             no           v1                              false        Node
```

8) **kubectl explain subjectaccessreview**



```
PS C:\DevOpsInsiders\Batch15\azure-devsecops-batch-15\CodeSamples\Kubernetes> ^C
PS C:\DevOpsInsiders\Batch15\azure-devsecops-batch-15\CodeSamples\Kubernetes> kubectl explain subjectaccessreviews
GROUP:      authorization.k8s.io
KIND:       SubjectAccessReview
VERSION:    v1

DESCRIPTION:
    SubjectAccessReview checks whether or not a user or group can perform an
    action.

FIELDS:
```

9)

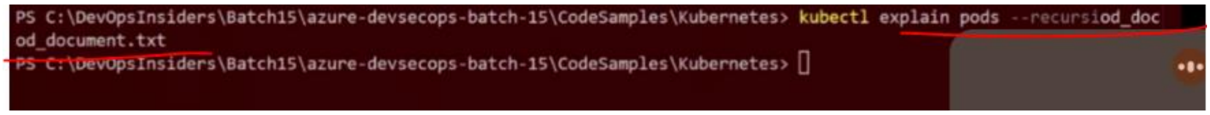


```
apiVersion: v1
kind: Pod
metadata:
spec:
```

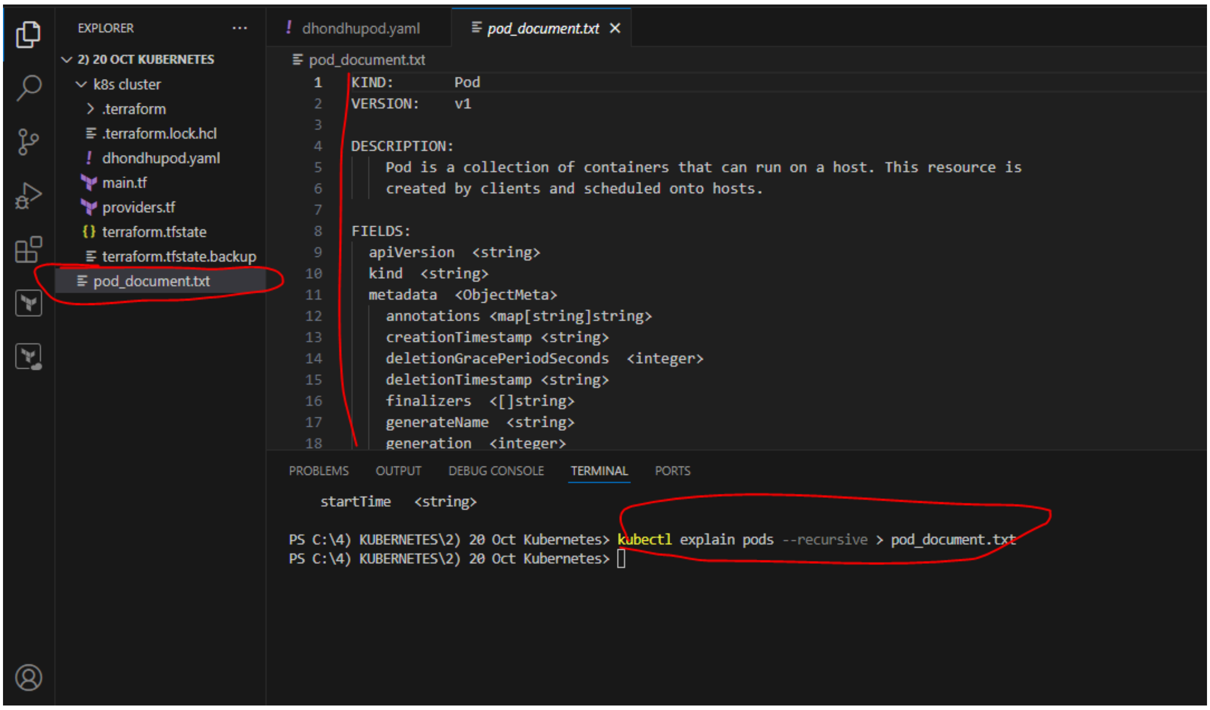
10) To know about meta data and spec, we have to run below command

kubectl explain pods –recursive

11) **kubectl explain pods --recursive > pod_document.txt**



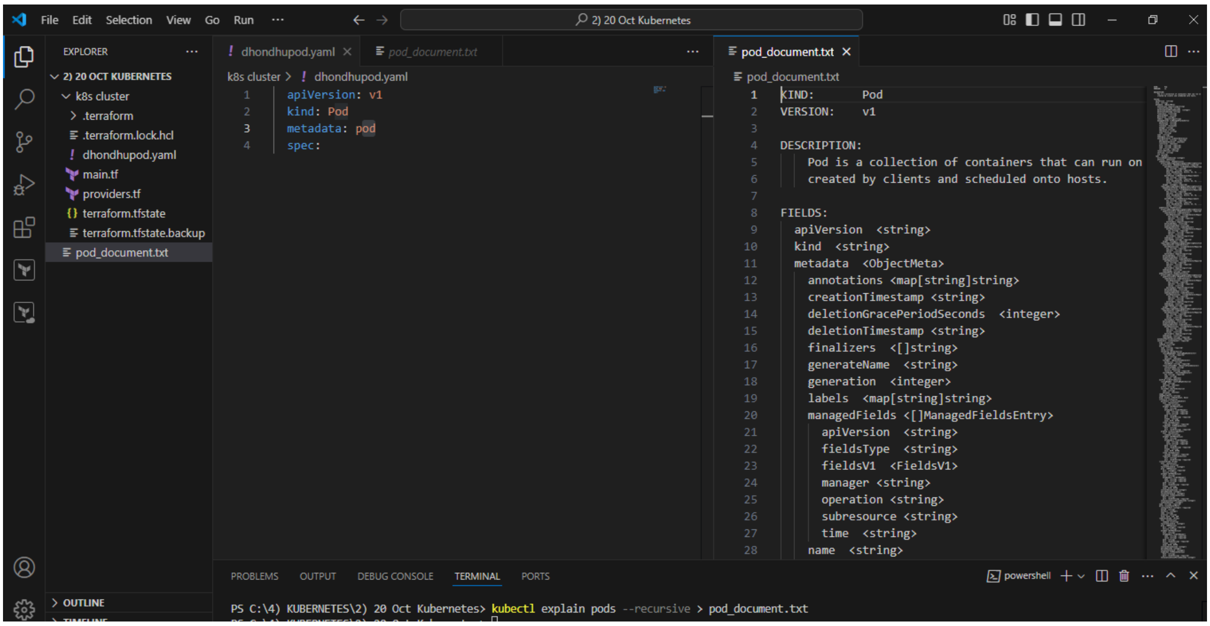
```
PS C:\DevOpsInsiders\Batch15\azure-devsecops-batch-15\CodeSamples\Kubernetes> kubectl explain pods --recursiod_doc
od_document.txt
PS C:\DevOpsInsiders\Batch15\azure-devsecops-batch-15\CodeSamples\Kubernetes> 
```



```
KIND:       Pod
VERSION:    v1

DESCRIPTION:
    Pod is a collection of containers that can run on a host. This resource is
    created by clients and scheduled onto hosts.

FIELDS:
  apiVersion  <string>
  kind  <string>
  metadata  <ObjectMeta>
    annotations <map[string]string>
    creationTimestamp <string>
    deletionGracePeriodSeconds  <integer>
    deletionTimestamp <string>
    finalizers  <[]string>
    generateName  <string>
    generation  <integer>
```

```
    startTime   <string>

PS C:\4) KUBERNETES\2) 20 Oct Kubernetes> kubectl explain pods --recursive > pod_document.txt
PS C:\4) KUBERNETES\2) 20 Oct Kubernetes>
```

12) Open our file and whole doc on another side

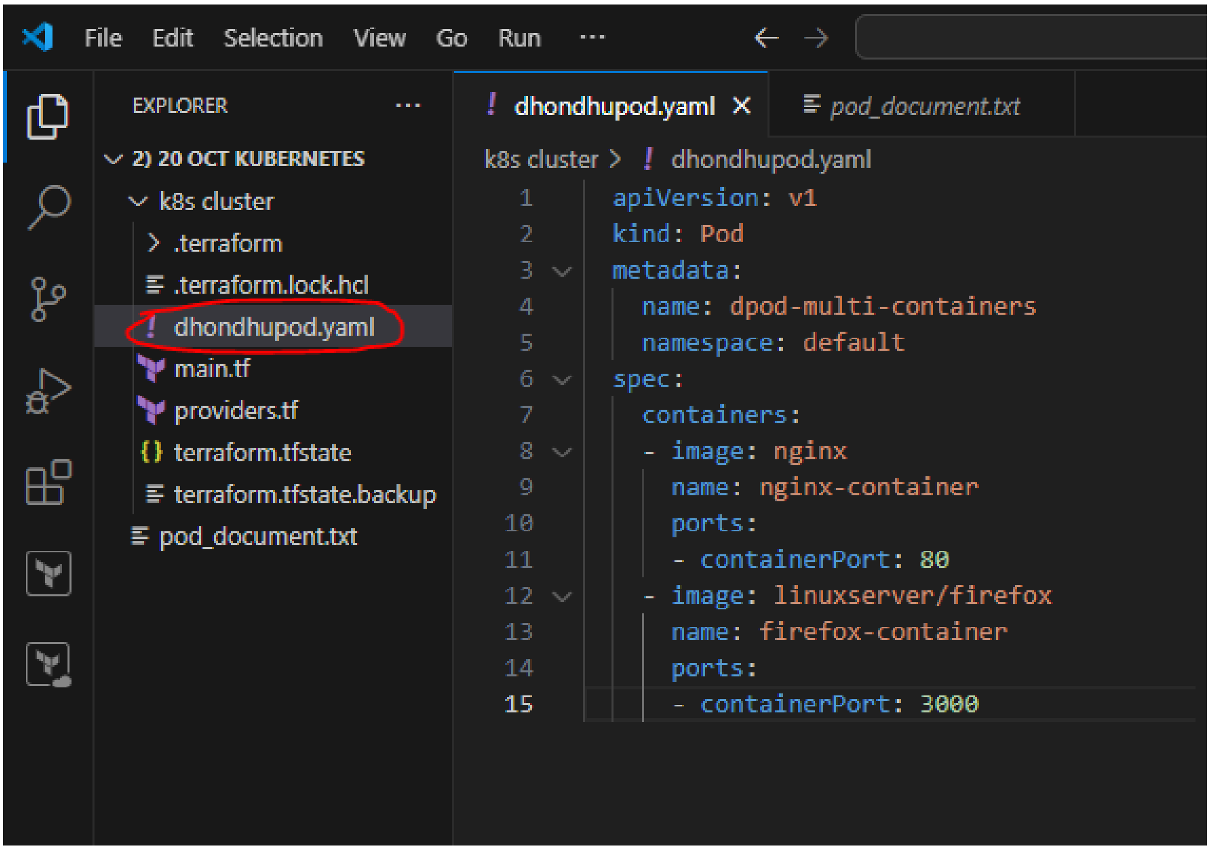


```
apiVersion: v1
kind: Pod
metadata: pod
spec:
```

```
KIND:       Pod
VERSION:    v1

DESCRIPTION:
    Pod is a collection of containers that can run on
    created by clients and scheduled onto hosts.

FIELDS:
  apiVersion  <string>
  kind  <string>
  metadata  <ObjectMeta>
    annotations <map[string]string>
    creationTimestamp <string>
    deletionGracePeriodSeconds  <integer>
    deletionTimestamp <string>
    finalizers  <[]string>
    generateName  <string>
    generation  <integer>
    labels  <map[string]string>
    managedFields <[]ManagedFieldsEntry>
      apiVersion  <string>
      fieldsType  <string>
      fieldsV1  <FieldsV1>
      manager <string>
      operation <string>
      subresource <string>
      time  <string>
    name  <string>
```

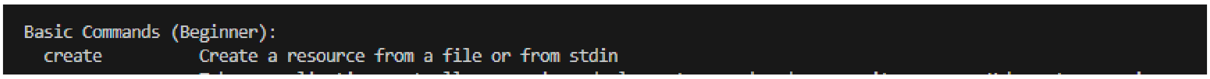
```
PS C:\4) KUBERNETES\2) 20 Oct Kubernetes> kubectl explain pods --recursive > pod_document.txt
```

13) Write code as below



```yaml
apiVersion: v1
kind: Pod
metadata:
  name: dpod-multi-containers
  namespace: default
spec:
  containers:
  - image: nginx
    name: nginx-container
    ports:
    - containerPort: 80
  - image: linuxserver/firefox
    name: firefox-container
    ports:
    - containerPort: 3000
```

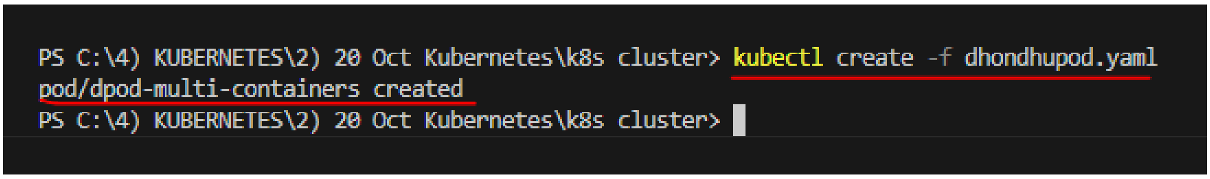
14) **kubectl --help**



```
Basic Commands (Beginner):
  create         Create a resource from a file or from stdin
```

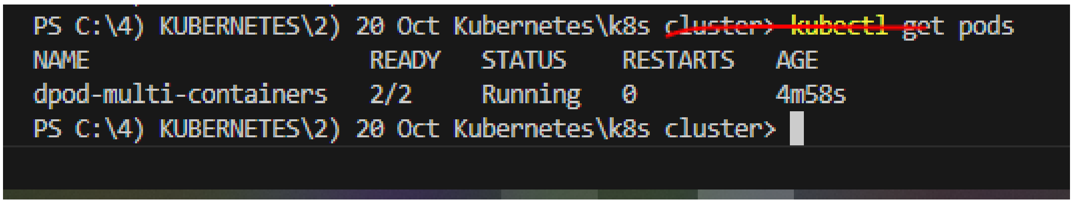
15) **kubectl create --help**

kubectl create -f dhondhupod.yaml



```
PS C:\4) KUBERNETES\2) 20 Oct Kubernetes\k8s cluster> kubectl create -f dhondhupod.yaml
pod/dpod-multi-containers created
PS C:\4) KUBERNETES\2) 20 Oct Kubernetes\k8s cluster>
```

16) **kubectl get pods**

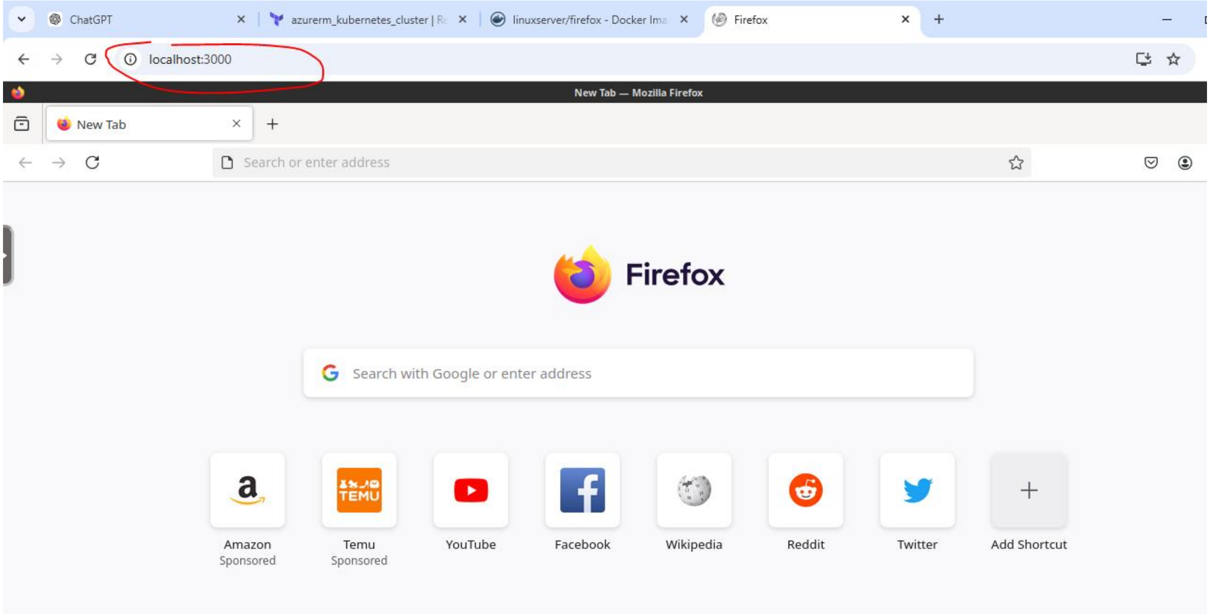


```
PS C:\4) KUBERNETES\2) 20 Oct Kubernetes\k8s cluster> kubectl get pods
NAME                    READY   STATUS    RESTARTS   AGE
dpod-multi-containers   2/2     Running   0          4m58s
PS C:\4) KUBERNETES\2) 20 Oct Kubernetes\k8s cluster>
```

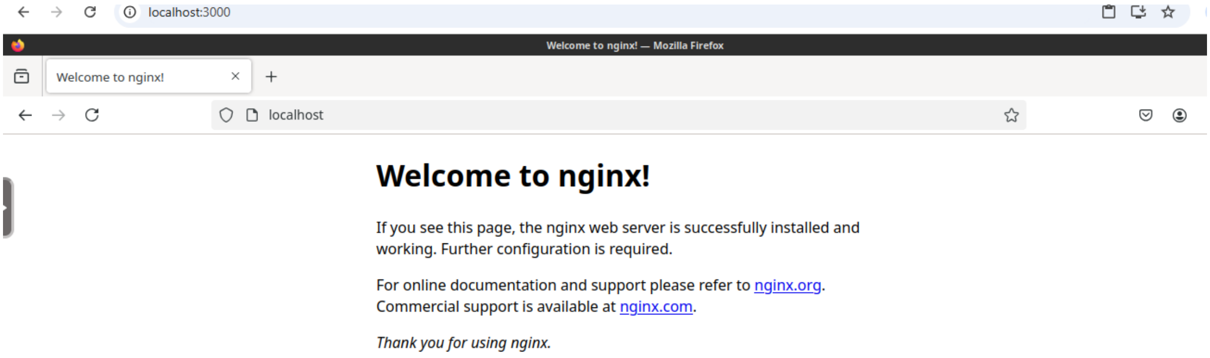
Here 2/2 = both pods running

17) **kubectl port-forward dpod-multi-containers 3000:3000**

18) localhost:3000



19) localhost:80



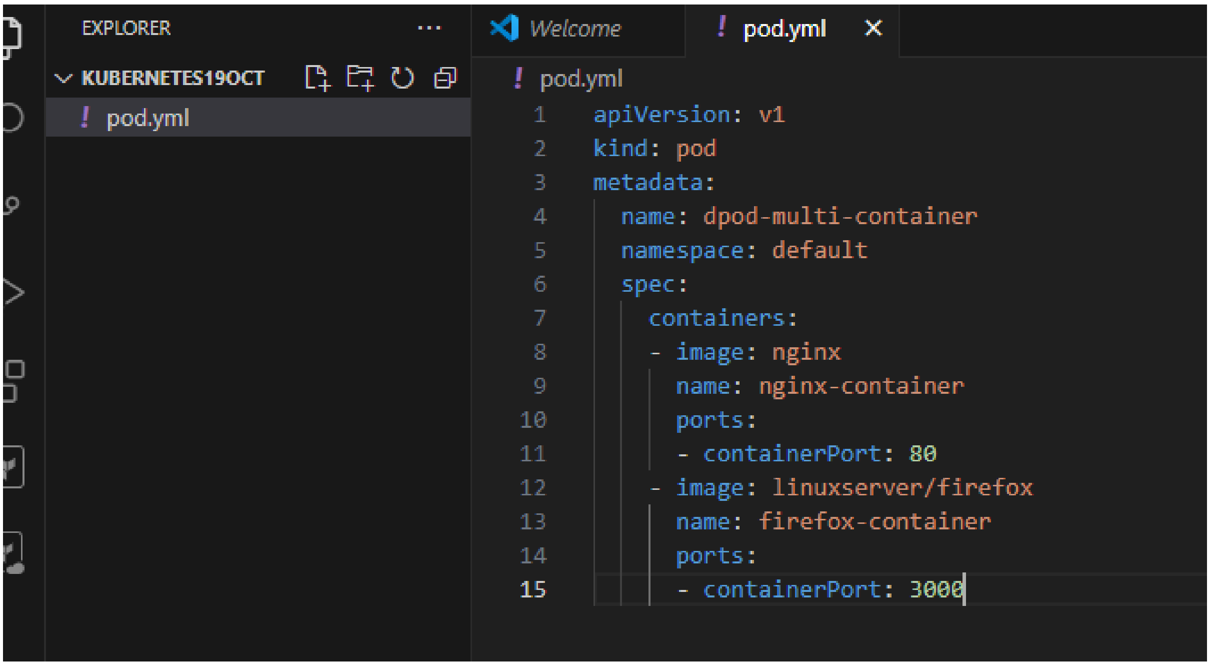
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

3) Open vscode and write yaml



```yaml
apiVersion: v1
kind: pod
metadata:
  name: dpod-multi-container
  namespace: default
  spec:
    containers:
    - image: nginx
      name: nginx-container
      ports:
      - containerPort: 80
    - image: linuxserver/firefox
      name: firefox-container
      ports:
      - containerPort: 3000
```

4) Run command