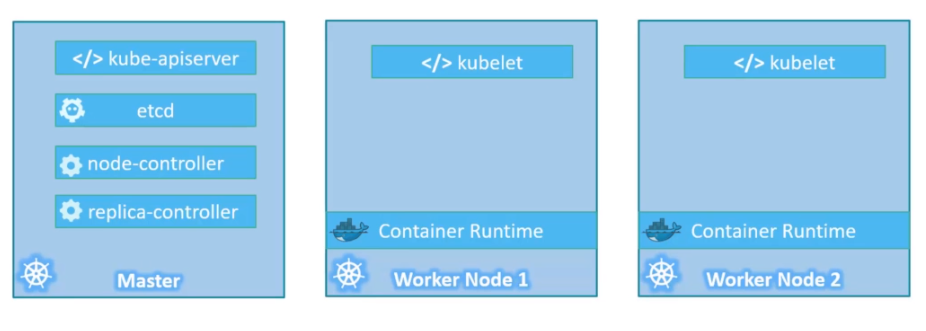
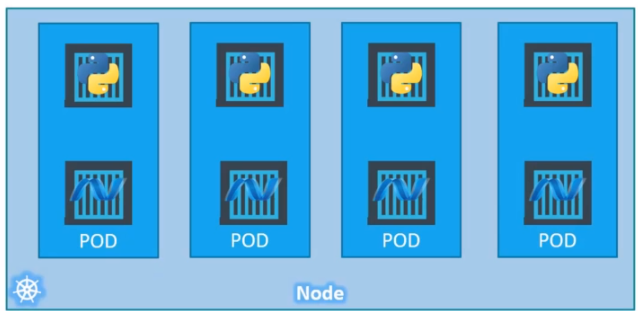
Master – Worker Node setup :



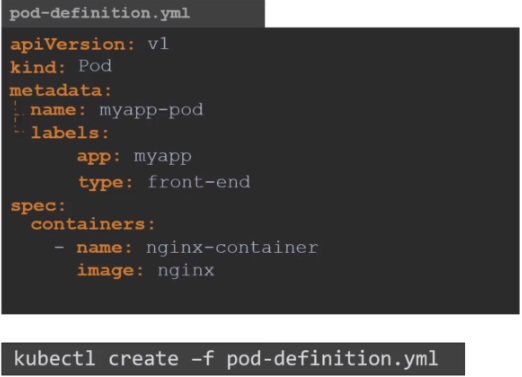
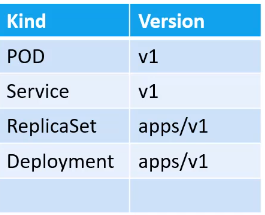
**Node and Pod :**

* A pod usually has one container in it (it can have multiple container in it as well)
* Pod is the smallest thing you can create in Kubernetes
* Node can have several pods in it. (one master and multiple nodes and each node can have multiple pods.)
* When user number on application increases > create more pods in node > when node reaches its max capacity , start creating more nodes. (do not create more containers in single pod)



**Kubernetes YAML file:**

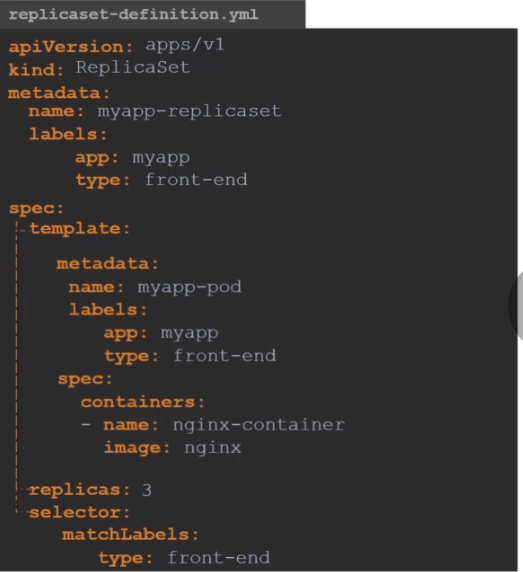
* Pod config YAML: **“pod-definition.yml”** Must have these 4 data in it.

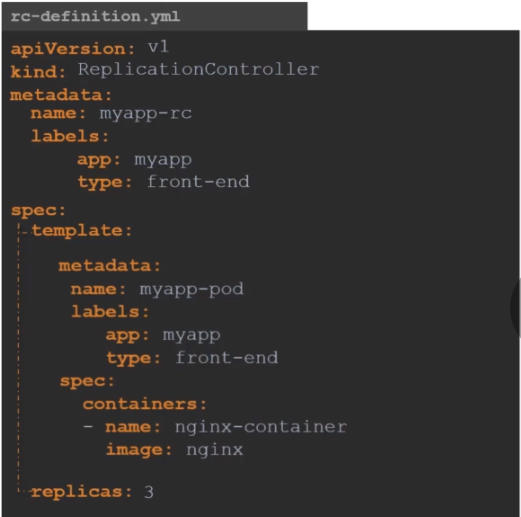
 

**Replication controler & Replica set**: **“rc-definition.yml”** or **“replicaset-definition.yml”**

* Replication controler is the old technology and is going to be replaced with replica set. They both have same functionality.
* it make sures that specified number of pods are running all the time. Means if we loose one pod, replication controler will detect it and will create a new one.
* It is also usefull for load balancing. As the demand increases , it will create new pods to distribute the load.

(replication controler spans across multiple nodes in cluster.)



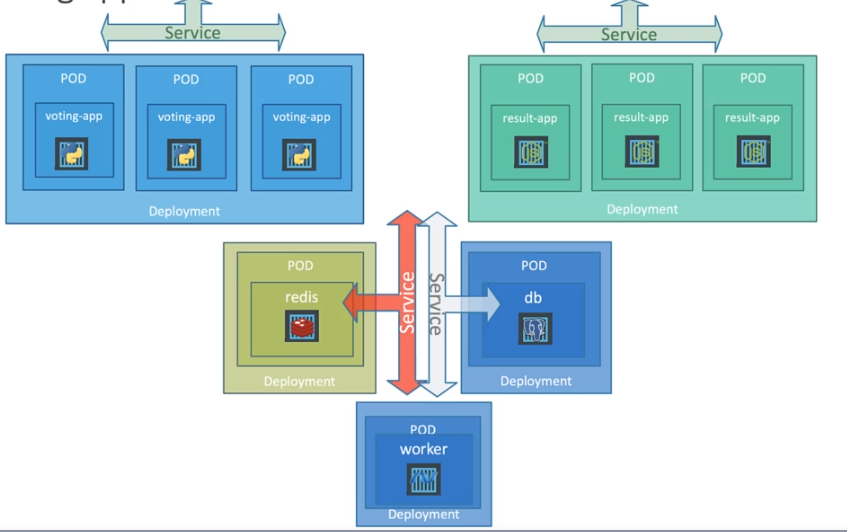
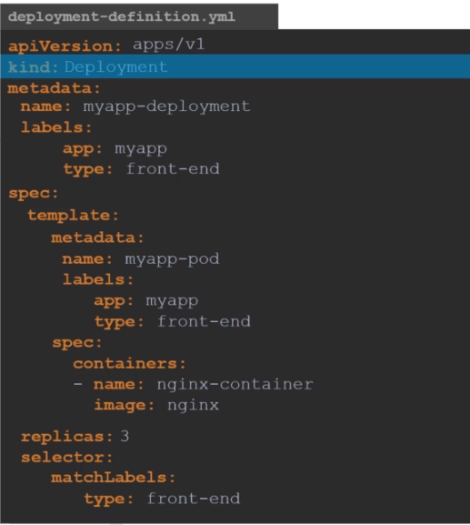


$ kubectl create -f rc-definition.yml $ kubectl create -f replicaset-definition.yml

$ kubectl get replicationcontroller

**Deployment**: **“deployment-definition.yaml”**

* Deployment provides us with the capability to upgrade the underlying instances seamlessly, using rolling updates, undo changes, pause and resume changes.
* $ kubectl create -f deployment-definition.yml
* $ kubectl get deployments

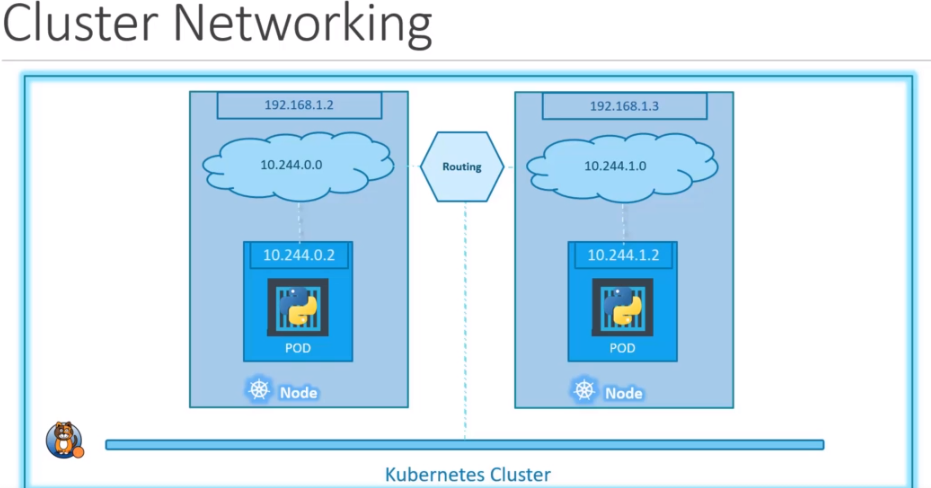


**Update & Rollback :**

* $ kubectl rollout status deployment/myapp-deployment : to see the status of rollout
* $ kubectl rollout history deployment/myapp-deployment : to check history
* $ kubectl rollout undo deployment/myapp-deployment : incase if we want to roll back to the previous version.

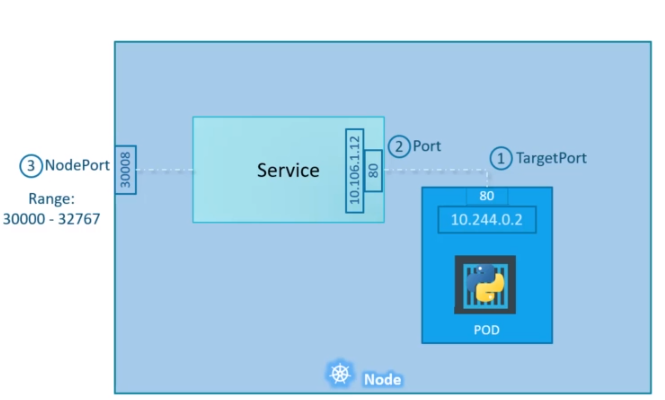
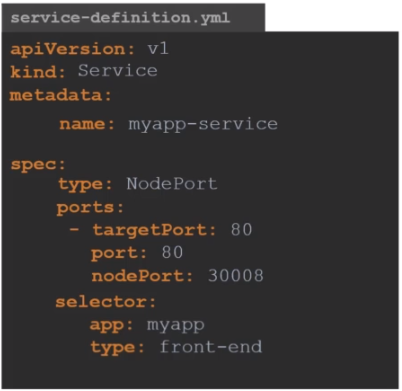
**Networking in Kubernetes:**

* Each Node has its own IP address.
* Each pod has its own IP address.

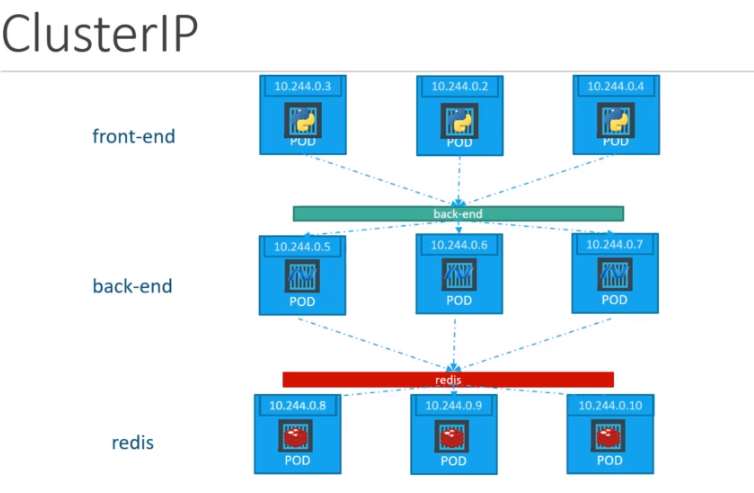
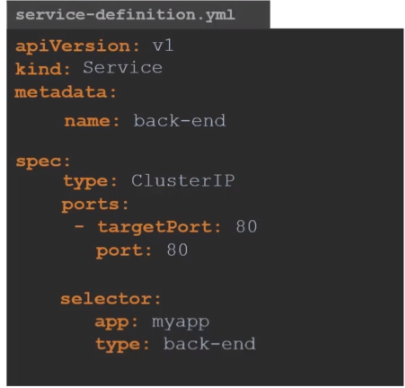


**Services:**  “**service-definition.yaml”**

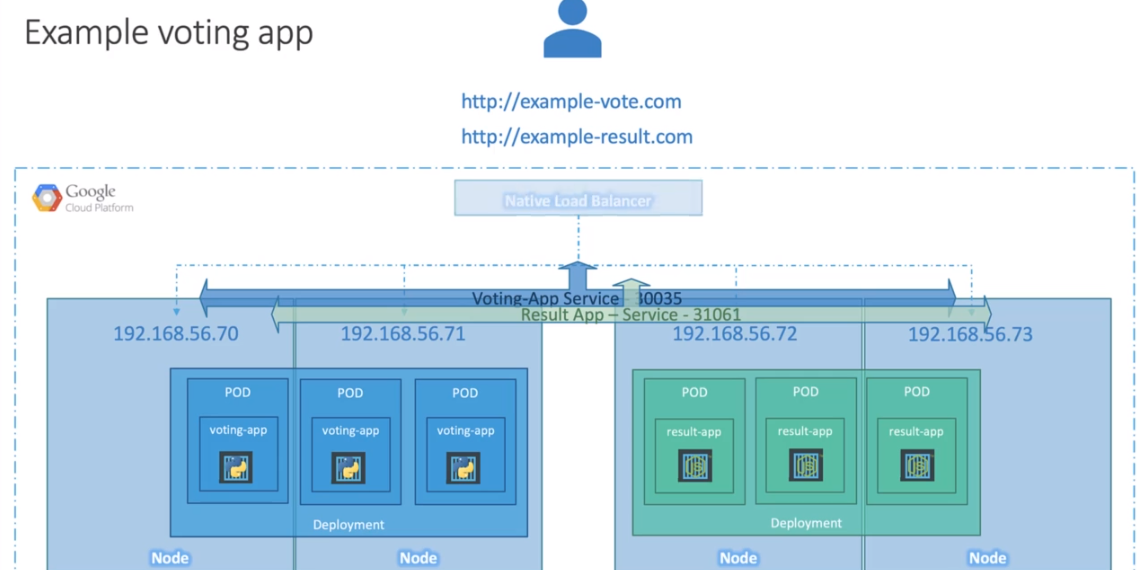
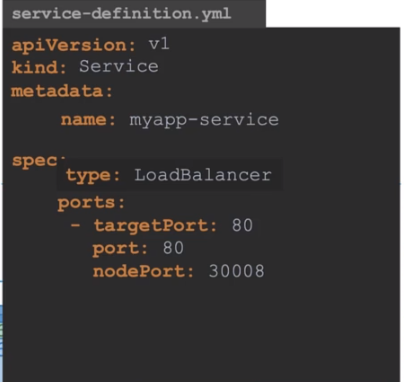
* It enables the connectivity b/w sets of pods.
* It Helps enabling communication b/w frontend application and Users.
* It helps enabling communication b/w frontend and backend pods.
* **Nodeport service:**

* $ kubectl create -f service-definition.yaml
* $ kubectl get services
* $ kubectl describe services myapp-service
* You can access the application form browser using the port number (Not targetport ) <http://localhost:80> (port)
* Targetport is the container port and port is Host port
* **Cluster IP:**

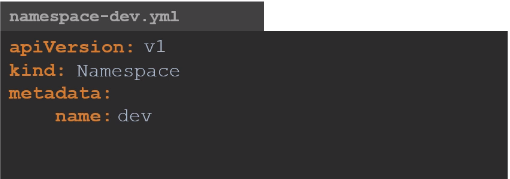
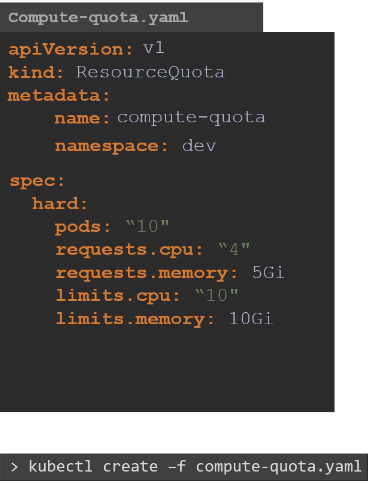
 

* **Load-Balancer:**



**Namespaces:**  **“namespace-dev.yaml” & “compute-quota.yaml”**

* All the Pods, Services and Deployments that we create are created under certain namespaces
* By default Kubernetes has “**Default**” namespace
* Each of these namespaces has sets of policies which defines who can do what
* Compute-quota defines how many pods, cpu, memory can namespace have.
* $ kubectl create -f namespace-dev.yaml
* $ kubectl get pods –namespace=kube-system : to get the pods of specific namespace
* $ kubectl create -f pod-definition.yml –namespace=dev : to create pod in specific namespace
* $ kubectl config set-contex $(kubectl config current contex) –namespace=dev : to move from ‘default’ namespace to ‘dev’
* $ kubectl get pods –all-namespaces
* $ kubectl create -f compute-quota.yaml

**Overall scenario:**

