**Lesson 3 Demo 2**

**Understanding the Best Practices of Kubernetes Cluster**



Steps to be followed:

1. Placing all the configuration files in the same directory
2. Creating a single configuration file for related objects

**Step 1: Placing all the configuration files in the same directory**

1. Create a directory to place all configuration files

***mkdir configfiles***

1. Navigate to the **configfiles** folder to create config files

***cd configfiles***



**Step 2: Creating a single configuration file for related objects**

1. Inside the **configfiles** folder, create a sample config file

***vi kubesample.yaml***



1. Write the following code in the **kubesample.yaml** file:

***apiVersion: v1***

***kind: Service***

***metadata:***

***name: redis***

***labels:***

***app: hello***

***tier: backend***

***role: master***

***spec:***

***ports:***

***- port: 6379***

***targetPort: 80***

***selector:***

***app: redis***

***tier: backend***

***role: master***

***---***

***apiVersion: apps/v1***

***kind: Deployment***

***metadata:***

***name: redis-master***

***spec:***

***selector:***

***matchLabels:***

***app: redis***

***role: master***

***tier: backend***

***replicas: 1***

***template:***

***metadata:***

***labels:***

***app: redis***

***role: master***

***tier: backend***

***spec:***

***containers:***

***- name: master***

***image: redis***

***resources:***

***requests:***

***cpu: 100m***

***memory: 100Mi***

***ports:***

***- containerPort: 6379***

***---***

***apiVersion: v1***

***kind: Service***

***metadata:***

***name: redis-slave***

***labels:***

***app: redis***

***tier: backend***

***role: slave***

***spec:***

***ports:***

***- port: 6379***

***selector:***

***app: redis***

***tier: backend***

***role: slave***

***---***

***apiVersion: apps/v1***

***kind: Deployment***

***metadata:***

***name: redis-slave***

***spec:***

***selector:***

***matchLabels:***

***app: redis***

***role: slave***

***tier: backend***

***replicas: 2***

***template:***

***metadata:***

***labels:***

***app: redis***

***role: slave***

***tier: backend***

***spec:***

***containers:***

***- name: slave***

***image: gcr.io/google\_samples/gb-redisslave:v1***

***resources:***

***requests:***

***cpu: 100m***

***memory: 100Mi***

***env:***

***- name: GET\_HOSTS\_FROM***

***value: dns***

***ports:***

***- containerPort: 6379***

***---***

***apiVersion: v1***

***kind: Service***

***metadata:***

***name: frontend***

***labels:***

***app: kubesample***

***tier: frontend***

***spec:***

***# comment or delete the following line if you want to use a LoadBalancer***

***type: NodePort***

***# if your cluster supports it, uncomment the following to automatically create***

***# an external load-balanced IP for the frontend service.***

***# type: LoadBalancer***

***ports:***

***- port: 80***

***selector:***

***app: kubesample***

***tier: frontend***

***---***

***apiVersion: apps/v1***

***kind: Deployment***

***metadata:***

***name: frontend***

***spec:***

***selector:***

***matchLabels:***

***app: kubesample***

***tier: frontend***

***replicas: 3***

***template:***

***metadata:***

***labels:***

***app: kubesample***

***tier: frontend***

***spec:***

***containers:***

***- name: php-redis***

***image: gcr.io/google-samples/gb-frontend:v4***

***resources:***

***requests:***

***cpu: 100m***

***memory: 100Mi***

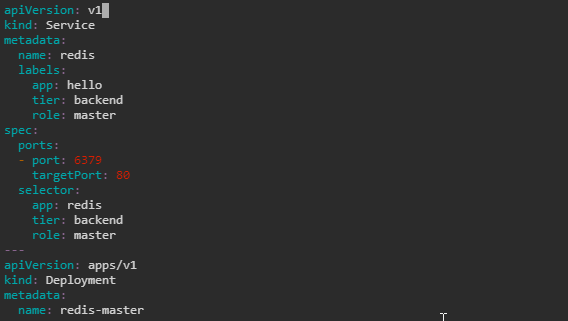
***env:***

***- name: GET\_HOSTS\_FROM***

***value: dns***

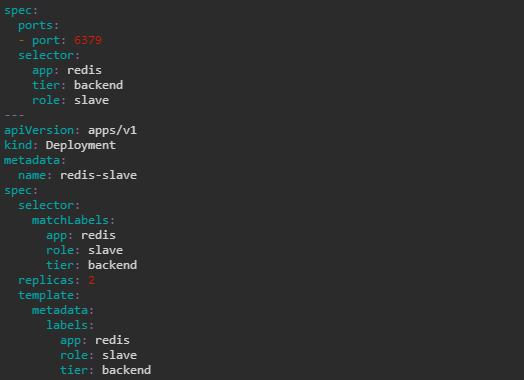
***ports:***

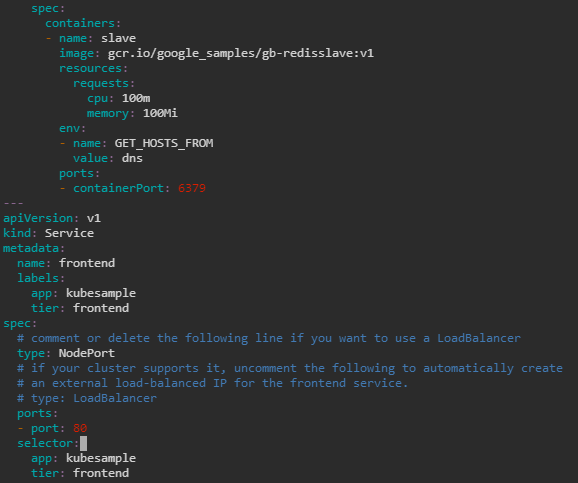
***- containerPort: 80***

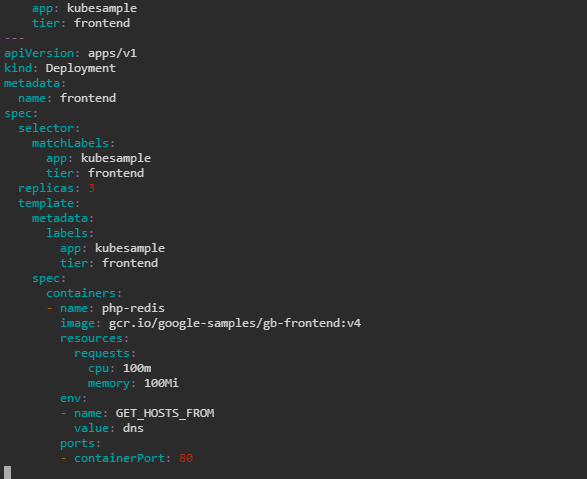








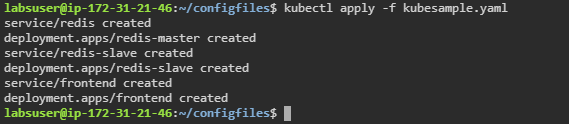




| **Note:** Press the **esc** button and type **:wq** to save and exit the file |
| --- |

1. Write the following command to apply the configuration file to deploy:

***kubectl apply -f kubesample.yaml***



1. Write the following command to verify the resources created using the config files:

***kubectl get all***

