1. Create a namespace dev-environment and apply a resource-based quota that restricts the number of pods to 3 and services to 2.

Create a namespace

- **kubectl create namespace dev-environment**

```
[root@master ~]# kubectl create namespace dev-environment
namespace/dev-environment created
[root@master ~]# kubectl get ns
NAME              STATUS   AGE
default           Active   7d20h
dev-environment   Active   17s
kube-flannel      Active   7d20h
kube-node-lease   Active   7d20h
kube-public       Active   7d20h
kube-system       Active   7d20h
[root@master ~]#
```

- vi quota.yml

apiVersion: v1

kind: ResourceQuota

metadata:

name: dev-environment-quota

namespace: dev-environment

spec:

hard:

pods: "3"

services: "2"

```yaml
apiVersion: v1
kind: ResourceQuota
metadata:
  name: dev-environment-quota
  namespace: dev-environment
spec:
  hard:
    pods: "3"
    services: "2"
```

- **kubectl apply -f quota.yaml**
- **kubectl get resourcequota -n dev-environment**

```
[root@master ~]# kubectl apply -f quota.yml
resourcequota/dev-environment-quota created
[root@master ~]# kubectl get resourcequota -n dev-environment
NAME                    REQUEST                       LIMIT   AGE
dev-environment-quota   pods: 0/3, services: 0/2              16s
[root@master ~]#
```

2. Create a pod in the prod-environment namespace with 0.2 CPU and 200Mi memory requests, and 0.5 CPU and 500Mi memory limits.

- **kubectl create namespace prod-environment**

vi prod-pod.yml

apiVersion: v1

kind: Pod

metadata:

name: prod-resource-pod

namespace: prod-environment

spec:

containers:

- name: nginx

image: nginx

resources:

requests:

cpu: "200m"

memory: "200Mi"

limits:

cpu: "500m"

memory: "500Mi"

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: prod-resource-pod
  namespace: prod-environment
spec:
  containers:
  - name: nginx
    image: nginx
    resources:
      requests:
        cpu: "200m"
        memory: "200Mi"
      limits:
        cpu: "500m"
        memory: "500Mi"
```

- **kubectl apply -f prod-pod.yml**
- **kubectl get pod -n prod-environment**

```
[root@master ~]# kubectl apply -f prod-pod.yml
pod/prod-resource-pod created
[root@master ~]# kubectl get pod -n prod-environment
NAME                READY   STATUS    RESTARTS   AGE
prod-resource-pod   1/1     Running   0          66s
```

- **kubectl describe pod prod-resource-pod -n prod-environment**

```
[root@master ~]# kubectl describe pod prod-resource-pod -n prod-environment
Name:             prod-resource-pod
Namespace:        prod-environment
Priority:         0
Service Account:  default
Node:             worker-02/172.31.72.26
Start Time:       Wed, 24 Dec 2025 13:03:37 +0000
Labels:           <none>
Annotations:      <none>
Status:           Running
IP:               10.244.2.52
IPs:
  IP:  10.244.2.52
Containers:
  nginx:
    Container ID:   containerd://a6e9367817922524e88758338b7ffd71d6175670ef60688
    Image:          nginx
    Image ID:       docker.io/library/nginx@sha256:fb01117203ff38c2f9af91db1a740
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Wed, 24 Dec 2025 13:03:38 +0000
    Ready:          True
    Restart Count:  0
    Limits:
      cpu:     500m
      memory:  500Mi
    Requests:
      cpu:        200m
      memory:     200Mi
    Environment:  <none>
    Mounts:
```

3. In the staging-environment namespace, set a LimitRange that assigns default CPU and memory limits (300m CPU, 600Mi memory) and applies a minimum and maximum CPU.

- **kubectl create namespace staging-environment**

```
[root@master ~]# kubectl create namespace staging-environment
namespace/staging-environment created
[root@master ~]#
```

- **vi limitrange.yml**

```
apiVersion: v1
kind: LimitRange
metadata:
  name: staging-limitrange
  namespace: staging-environment
spec:
  limits:
  - type: Container
    default:
      cpu: "300m"
      memory: "600Mi"
    defaultRequest:
      cpu: "300m"
      memory: "600Mi"
    min:
      cpu: "100m"
    max:
```

cpu: "1000m"

```yaml
apiVersion: v1
kind: LimitRange
metadata:
  name: staging-limitrange
  namespace: staging-environment
spec:
  limits:
  - type: Container
    default:
      cpu: "300m"
      memory: "600Mi"
    defaultRequest:
      cpu: "300m"
      memory: "600Mi"
    min:
      cpu: "100m"
    max:
      cpu: "1000m"
```

- kubectl apply -f limitrange.yml
- kubectl describe limitrange staging-limitrange -n staging-environment

```
[root@master ~]# vi limitrange.yml
[root@master ~]# kubectl apply -f limitrange.yml
limitrange/staging-limitrange created
[root@master ~]# kubectl describe limitrange staging-limitrange -n staging-environment
Name:       staging-limitrange
Namespace:  staging-environment
Type        Resource  Min   Max  Default Request  Default Limit  Max Limit/Request Ratio
----        --------  ---   ---  ---------------  -------------  -----------------------
Container   cpu       100m  1    300m             300m           -
Container   memory    -     -    600Mi            600Mi          -
[root@master ~]#
```

4. Create a pod and a NodePort service in the default namespace, then create another pod in the test namespace and communicate between them using Service DNS.

- **kubectl create ns test**

```
[root@master ~]# kubectl create namespace test
namespace/test created
[root@master ~]#
```

vi web-pod.yml

apiVersion: v1

kind: Pod

metadata:

name: web-pod

labels:

app: web

spec:

containers:

- name: nginx

image: nginx

ports:

- containerPort: 80

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: web-pod
  labels:
    app: web
spec:
  containers:
  - name: nginx
    image: nginx
    ports:
    - containerPort: 80
```

- **kubectl apply -f web-pod.yml**

```
[root@master ~]# vi web-pod.yml
[root@master ~]# kubectl apply -f web-pod.yml
pod/web-pod created
[root@master ~]#
```

vi web-service.yml

apiVersion: v1

kind: Service

metadata:

name: web-service

spec:

type: NodePort

selector:

app: web

ports:

- port: 80

targetPort: 80

nodePort: 30080

```
apiVersion: v1
kind: Service
metadata:
  name: web-service
spec:
  type: NodePort
  selector:
    app: web
  ports:
    - port: 80
      targetPort: 80
      nodePort: 30080
```

- **kubectl apply -f web-service.yml**

```
[root@master ~]# kubectl apply -f web-service.yml
service/web-service created
[root@master ~]#
```

vi client-pod.yml

apiVersion: v1

kind: Pod

metadata:

name: client-pod

namespace: test

spec:

containers:

- name: busybox

image: busybox

command: ["sleep", "3600"]

```
apiVersion: v1
kind: Pod
metadata:
  name: client-pod
  namespace: test
spec:
  containers:
  - name: busybox
    image: busybox
    command: ["sleep", "3600"]
```

- **kubectl apply -f client-pod.yml**

```
[root@master ~]# kubectl apply -f client-pod.yml
pod/client-pod created
[root@master ~]#
```

- **kubectl exec -it client-pod -n test – sh**
- **wget -qO- web-service.default.svc.cluster.local**

```
[root@master ~]# kubectl exec -it client-pod -n test -- sh
/ # wget -qO- web-service.default.svc.cluster.local
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
html { color-scheme: light dark; }
body { width: 35em; margin: 0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif; }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

5. Apply a LimitRange with a max limit/request ratio of 2 for memory in the performance-environment namespace, and test by creating a pod with mismatched resource requests and limits.

- **kubectl create namespace performance-environment**

```
[root@master ~]# kubectl create namespace performance-environment
namespace/performance-environment created
[root@master ~]#
```

vi memory-limitrange.yml

apiVersion: v1

kind: LimitRange

metadata:

name: memory-ratio-limit

namespace: performance-environment

spec:

limits:

- type: Container

maxLimitRequestRatio:

memory: "2"

```yaml
apiVersion: v1
kind: LimitRange
metadata:
  name: memory-ratio-limit
  namespace: performance-environment
spec:
  limits:
  - type: Container
    maxLimitRequestRatio:
      memory: "2"
```

- **kubectl apply -f memory-limitrange.yaml**

```
[root@master ~]# kubectl apply -f memory-limitrange.yml
limitrange/memory-ratio-limit created
[root@master ~]#
```

- **kubectl describe limitrange memory-ratio-limit -n performance-environment**

```
[root@master ~]# kubectl describe limitrange memory-ratio-limit -n performance-environment
Name:       memory-ratio-limit
Namespace:  performance-environment
Type        Resource  Min  Max  Default Request  Default Limit  Max Limit/Request Ratio
----        --------  ---  ---  ---------------  -------------  -----------------------
Container   memory    -    -    -                -              2
[root@master ~]#
```

Test with valid pod

vi valid.yml

apiVersion: v1

kind: Pod

metadata:

name: good-memory-pod

namespace: performance-environment

spec:

containers:

- name: nginx

image: nginx

resources:

requests:

memory: "200Mi"

limits:

memory: "400Mi"

```
apiVersion: v1
kind: Pod
metadata:
  name: good-memory-pod
  namespace: performance-environment
spec:
  containers:
  - name: nginx
    image: nginx
    resources:
      requests:
        memory: "200Mi"
      limits:
        memory: "400Mi"
```

- **kubectl apply -f valid.yml**

```
[root@master ~]# kubectl apply -f valid.yml
pod/good-memory-pod created
```

if we test with the bad pod it don't create

vi badpod.yml

apiVersion: v1

kind: Pod

metadata:

name: bad-memory-pod

namespace: performance-environment

spec:

containers:

- name: nginx

image: nginx

resources:

requests:

memory: "200Mi"

limits:

memory: "600Mi"

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: bad-memory-pod
  namespace: performance-environment
spec:
  containers:
  - name: nginx
    image: nginx
    resources:
      requests:
        memory: "200Mi"
      limits:
        memory: "600Mi"
```

- **kubectl apply -f badpod.yml**

```
[root@master ~]# kubectl apply -f badpod.yml
Error from server (Forbidden): error when creating "badpod.yml": pods "bad-memory-pod" is forbidden: memory max limit to request ratio per Contai
ner is 2, but provided ratio is 3.000000
[root@master ~]#
```