How you want to deploy your application In production environment?
Create deployment.yaml

deployment.yaml

```
deployments > ! deployment.yaml > {} spec > # replicas
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: myapp-deployment
5    labels:
6      tier: frontend
7      app: nginx
8  spec:
9    selector:
10     matchLabels:
11       app: myapp
12   replicas: 3
13   template:
14     metadata:
15       name: nginx-2
16       labels:
17         app: myapp
18     spec:
19       containers:
20         - name: nginx
21           image: nginx
```

replicaset.yaml

```
replicasets > ! replicaset.yaml > {} spec > {}
1  apiVersion: apps/v1
2  kind: ReplicaSet
3  metadata:
4    name: myapp-replicaset
5    labels:
6      app: myapp
7  spec:
8    selector:
9      matchLabels:
10       app: myapp
11   replicas: 4
12   template:
13     metadata:
14       name: nginx-2
15       labels:
16         app: myapp
17     spec:
18       containers:
19         - name: nginx
20           image: nginx
```

```
admin@ubuntu-server deployments # kubectl create -f deployment.yaml
deployment.apps/myapp-deployment created
admin@ubuntu-server deployments # kubectl get deployments
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
myapp-deployment    3/3     3            3           10s
admin@ubuntu-server deployments # kubectl get pods
NAME                          READY   STATUS    RESTARTS   AGE
myapp-replicaset-pjs89        1/1     Running   0          34m
myapp-replicaset-pwv6h        1/1     Running   0          34m
myapp-replicaset-zr6c7        1/1     Running   0          23s
```

```
controlplane ~ ➜ kubectl get all
NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
service/kubernetes   ClusterIP   10.43.0.1    <none>        443/TCP   7m36s

controlplane ~ ➜ kubectl get deployments
No resources found in default namespace.

controlplane ~ ➜ kubectl get deployments
NAME                 READY   UP-TO-DATE   AVAILABLE   AGE
frontend-deployment   0/4     4            0           5s

controlplane ~ ➜ kubectl get rs
NAME                            DESIRED   CURRENT   READY   AGE
frontend-deployment-7fbf4f5cd9   4         4         0       18s

controlplane ~ ➜ kubectl get pods
NAME                                  READY   STATUS             RESTARTS   AGE
frontend-deployment-7fbf4f5cd9-pnnlm   0/1     ImagePullBackOff   0          31s
frontend-deployment-7fbf4f5cd9-h4cb5   0/1     ErrImagePull       0          31s
frontend-deployment-7fbf4f5cd9-vqxbj   0/1     ErrImagePull       0          31s
frontend-deployment-7fbf4f5cd9-sxl7t   0/1     ErrImagePull       0          31s
```

What is the image used to create the pods in the new deployment? Busybox888

```
controlplane ~ ➜ kubectl describe deployments frontend-deployment
Name:                   frontend-deployment
Namespace:              default
CreationTimestamp:      Mon, 03 Apr 2023 06:37:12 +0000
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               name=busybox-pod
Replicas:               4 desired | 4 updated | 4 total | 0 availabl
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  name=busybox-pod
  Containers:
   busybox-container:
    Image:      busybox888
```

Why do you think the deployment is not ready? Image doesn't exist

```
Events:
  Type     Reason     Age                  From               Message
  ----     ------     ----                 ----               -------
  Normal   Scheduled  6m57s                default-scheduler  Successfully assigned default/frontend-deployment-7fbf4f5cd9-pnnlm to controlplane
  Normal   Pulling    5m32s (x4 over 6m56s)  kubelet          Pulling image "busybox888"
  Warning  Failed     5m32s (x4 over 6m56s)  kubelet          Failed to pull image "busybox888": rpc error: code = Unknown desc = failed to pull and unpack imag
.io/library/busybox888:latest": failed to resolve reference "docker.io/library/busybox888:latest": pull access denied, repository does not exist or may require au
on: server message: insufficient_scope: authorization failed
  Warning  Failed     5m32s (x4 over 6m56s)  kubelet          Error: ErrImagePull
  Warning  Failed     5m4s (x6 over 6m56s)   kubelet          Error: ImagePullBackOff
  Normal   BackOff    110s (x20 over 6m56s)  kubelet          Back-off pulling image "busybox888"
```

Create a new Deployment using the deployment-definition-1.yaml file located at /root/.
There is an issue with the file, so try to fix it.

```
controlplane ~ ✖ kubectl create -f deployment-definition-1.yaml
deployment.apps/deployment-1 created

controlplane ~ ➜ cat deployment-definition-1.yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: deployment-1
spec:
  replicas: 2
  selector:
    matchLabels:
      name: busybox-pod
  template:
    metadata:
      labels:
        name: busybox-pod
    spec:
      containers:
      - name: busybox-container
        image: busybox888
        command:
        - sh
        - "-c"
        - echo Hello Kubernetes! && sleep 3600
```

Create a new Deployment with the below attributes using your own deployment definition file.
Name: httpd-frontend;
Replicas: 3;
Image: httpd:2.4-alpine

```
controlplane ~ ➜ kubectl create -f dep-new.yaml
deployment.apps/httpd-frontend created

controlplane ~ ➜ cat dep-new.yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: httpd-frontend
spec:
  replicas: 3
  selector:
    matchLabels:
      name: httpd-frontend
  template:
    metadata:
      labels:
        name: httpd-frontend
    spec:
      containers:
      - name: httpd-frontend
        image: httpd:2.4-alpine

controlplane ~ ➜
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myapp-deployment
  labels:
    tier: frontend
spec:
  selector:
    matchLabels:
      app: myapp
  replicas: 6
  template:
    metadata:
      name: nginx-2
      labels:
        app: myapp
    spec:
      containers:
        - name: nginx
          image: nginx
~
```

Create deployment .

```
admin@ubuntu-server deployments # kubectl create -f deployment.yaml
deployment.apps/myapp-deployment created
```

Check the status of deployment .

```
admin@ubuntu-server deployments # kubectl rollout status deployment.apps/myapp-deploym
ent
deployment "myapp-deployment" successfully rolled out
```

Deployment is successfully rolled out .

In status command we can see that deployment brings pods one at a time .

```
admin@ubuntu-server deployments # kubectl delete deployment myapp-deployment
deployment.apps "myapp-deployment" deleted
admin@ubuntu-server deployments # kubectl create -f deployment.yaml
deployment.apps/myapp-deployment created
admin@ubuntu-server deployments # kubectl rollout status deployment.apps/myapp-deploym
ent
Waiting for deployment "myapp-deployment" rollout to finish: 0 of 6 updated replicas a
re available...
Waiting for deployment "myapp-deployment" rollout to finish: 1 of 6 updated replicas a
re available...
Waiting for deployment "myapp-deployment" rollout to finish: 2 of 6 updated replicas a
re available...
Waiting for deployment "myapp-deployment" rollout to finish: 3 of 6 updated replicas a
re available...
Waiting for deployment "myapp-deployment" rollout to finish: 4 of 6 updated replicas a
re available...
Waiting for deployment "myapp-deployment" rollout to finish: 5 of 6 updated replicas a
re available...
```

History of deployment .

```
admin@ubuntu-server deployments # kubectl rollout history  deployment.apps/myapp-deplo
yment
deployment.apps/myapp-deployment
REVISION  CHANGE-CAUSE
1         <none>
```

--record will instruct Kubernetes to record the cause of change in history of deployment .

```
admin@ubuntu-server deployments # kubectl create -f deployment.yaml --record
deployment.apps/myapp-deployment created
```

As we can see change cause if recorded .

```
admin@ubuntu-server deployments # kubectl rollout history  deployment.apps/myapp-deplo
yment
deployment.apps/myapp-deployment
REVISION  CHANGE-CAUSE
1         kubectl create --filename=deployment.yaml --record=true
```

Update image to nginx:1.18

```
admin@ubuntu-server deployments # kubectl edit deployment myapp-deployment --record
deployment.apps/myapp-deployment edited
admin@ubuntu-server deployments # kubectl rollout status deployment.apps/myapp-deploym
ent
Waiting for deployment "myapp-deployment" rollout to finish: 4 out of 6 new replicas h
ave been updated...
Waiting for deployment "myapp-deployment" rollout to finish: 4 out of 6 new replicas h
ave been updated...
Waiting for deployment "myapp-deployment" rollout to finish: 4 out of 6 new replicas h
ave been updated...
Waiting for deployment "myapp-deployment" rollout to finish: 4 out of 6 new replicas h
ave been updated...
Waiting for deployment "myapp-deployment" rollout to finish: 5 out of 6 new replicas h
ave been updated...
```

We can see cause of change , updated image etc .

```
admin@ubuntu-server deployments # kubectl describe deployment myapp-deployment
Name:                   myapp-deployment
Namespace:              default
CreationTimestamp:      Sun, 12 Jul 2020 16:09:55 -0400
Labels:                 tier=frontend
Annotations:            deployment.kubernetes.io/revision: 2
                        kubernetes.io/change-cause: kubectl edit deployment myapp-depl
oyment --record=true
Selector:               app=myapp
Replicas:               6 desired | 6 updated | 6 total | 6 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=myapp
  Containers:
   nginx:
    Image:        nginx:1.18
    Port:         <none>
    Host Port:    <none>
admin@ubuntu-server deployments # kubectl set image deployment myapp-deployment nginx=
nginx:1.18-perl --record
deployment.apps/myapp-deployment image_updated
admin@ubuntu-server deployments # kubectl rollout status deployment/myapp-deployment
Waiting for deployment "myapp-deployment" rollout to finish: 2 old replicas are pendin
g termination...
Waiting for deployment "myapp-deployment" rollout to finish: 2 old replicas are pendin
g termination...
Waiting for deployment "myapp-deployment" rollout to finish: 2 old replicas are pendin
g termination...
Waiting for deployment "myapp-deployment" rollout to finish: 1 old replicas are pendin
g termination...
Waiting for deployment "myapp-deployment" rollout to finish: 1 old replicas are pendin
g termination...
Waiting for deployment "myapp-deployment" rollout to finish: 1 old replicas are pendin
g termination...
Waiting for deployment "myapp-deployment" rollout to finish: 5 of 6 updated replicas a
re available...
deployment "myapp-deployment" successfully rolled out
admin@ubuntu-server deployments #
admin@ubuntu-server deployments # kubectl rollout history  deployment/myapp-deployment
deployment.apps/myapp-deployment
REVISION  CHANGE-CAUSE
1         kubectl create --filename=deployment.yaml --record=true
2         kubectl edit deployment myapp-deployment --record=true
3         kubectl set image deployment myapp-deployment nginx=nginx:1.18-perl --record
=true

admin@ubuntu-server deployments # kubectl rollout undo deployment/myapp-deployment
deployment.apps/myapp-deployment rolled back
admin@ubuntu-server deployments # kubectl rollout status deployment/myapp-deployment
Waiting for deployment "myapp-deployment" rollout to finish: 4 out of 6 new replicas h
ave been updated...
```

which is kubectl edit deployment,

Since we have roll back to revision 2 , revision 2 have become latest with revision 4 .

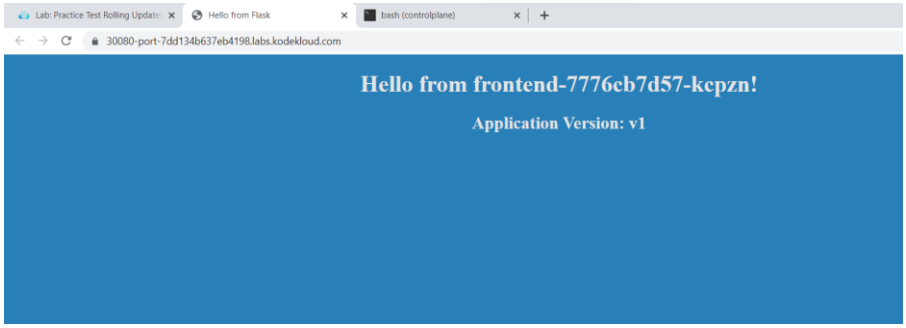
```
admin@ubuntu-server deployments #
admin@ubuntu-server deployments # kubectl rollout history  deployment/myapp-deployment
deployment.apps/myapp-deployment
REVISION  CHANGE-CAUSE
1         kubectl create --filename=deployment.yaml --record=true
3         kubectl set image deployment myapp-deployment nginx=nginx:1.18-perl --record
=true
4         kubectl edit deployment myapp-deployment --record=true
```

Now lets update image to some imaginary which doesn't exists .

```
admin@ubuntu-server deployments # kubectl edit deployment myapp-deployment --record
deployment.apps/myapp-deployment edited
admin@ubuntu-server deployments # kubectl rollout status deployment/myapp-deployment
Waiting for deployment "myapp-deployment" rollout to finish: 3 out of 6 new replicas h
ave been updated...
^Cadmin@ubuntu-server deployments # c
admin@ubuntu-server deployments # kubectl get deployment myapp-deployment
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
myapp-deployment   5/6     3            5           8m15s
admin@ubuntu-server deployments # kubectl get pods
NAME                                READY   STATUS         RESTARTS   AGE
myapp-deployment-789c649f95-9xs8q   1/1     Running        0          5m28s
myapp-deployment-789c649f95-dkfm4   1/1     Running        0          5m30s
myapp-deployment-789c649f95-qtngw   1/1     Running        0          5m31s
myapp-deployment-789c649f95-rktrd   1/1     Running        0          5m31s
myapp-deployment-789c649f95-x9jf5   1/1     Running        0          5m27s
myapp-deployment-84cfd5697c-5f7tg   0/1     ErrImagePull   0          69s
myapp-deployment-84cfd5697c-wtsjz   0/1     ErrImagePull   0          69s
myapp-deployment-84cfd5697c-xjgmp   0/1     ErrImagePull   0          69s
admin@ubuntu-server deployments #
admin@ubuntu-server deployments # kubectl rollout history  deployment/myapp-deployment
deployment.apps/myapp-deployment
REVISION  CHANGE-CAUSE
1         kubectl create --filename=deployment.yaml --record=true
3         kubectl set image deployment myapp-deployment nginx=nginx:1.18-perl --record
=true
4         kubectl edit deployment myapp-deployment --record=true
5         kubectl edit deployment myapp-deployment --record=true

admin@ubuntu-server deployments # kubectl rollout undo deployment/myapp-deployment
deployment.apps/myapp-deployment rolled back
admin@ubuntu-server deployments # kubectl rollout status deployment/myapp-deployment
deployment "myapp-deployment" successfully rolled out
admin@ubuntu-server deployments # kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
myapp-deployment-789c649f95-8s9gk   1/1     Running   0          12s
myapp-deployment-789c649f95-9xs8q   1/1     Running   0          9m5s
myapp-deployment-789c649f95-dkfm4   1/1     Running   0          9m7s
myapp-deployment-789c649f95-qtngw   1/1     Running   0          9m8s
myapp-deployment-789c649f95-rktrd   1/1     Running   0          9m8s
myapp-deployment-789c649f95-x9jf5   1/1     Running   0          9m4s
```

We have deployed a simple web application. Inspect the PODs and the Services
Wait for the application to fully deploy and view the application using the link called Webapp Portal above your terminal.



```
controlplane ~ ➜ kubectl get pods
NAME                        READY   STATUS    RESTARTS   AGE
frontend-7776cb7d57-p82nv   1/1     Running   0          50s
frontend-7776cb7d57-2w9h6   1/1     Running   0          50s
frontend-7776cb7d57-nt7rs   1/1     Running   0          50s
frontend-7776cb7d57-kcpzn   1/1     Running   0          50s
```

What is the current color of the web application? Blue

Run the script named curl-test.sh to send multiple requests to test the web application. Take a note of the output.
Execute the script at /root/curl-test.sh.

```
controlplane ~ ➜ ./curl-test.sh
Hello, Application Version: v1 ; Color: blue OK

Hello, Application Version: v1 ; Color: blue OK

Hello, Application Version: v1 ; Color: blue OK

Hello, Application Version: v1 ; Color: blue OK

Hello, Application Version: v1 ; Color: blue OK

Hello, Application Version: v1 ; Color: blue OK

Hello, Application Version: v1 ; Color: blue OK

Hello, Application Version: v1 ; Color: blue OK

Hello, Application Version: v1 ; Color: blue OK

Hello, Application Version: v1 ; Color: blue OK
```

Inspect the deployment and identify the number of PODs deployed by it
What container image is used to deploy the applications? kodekloud/webapp-color:v1

```
controlplane ~ ➜ kubectl describe deployments
Name:                   frontend
Namespace:              default
CreationTimestamp:      Mon, 03 Apr 2023 08:27:10 +0000
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               name=webapp
Replicas:               4 desired | 4 updated | 4 total | 4 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        20
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  name=webapp
  Containers:
   simple-webapp:
    Image:        kodekloud/webapp-color:v1
    Port:         8080/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  <none>
NewReplicaSet:   frontend-7776cb7d57 (4/4 replicas created)
Events:
  Type    Reason             Age    From                   Message
  ----    ------             ----   ----                   -------
  Normal  ScalingReplicaSet  3m52s  deployment-controller  Scaled up replica set frontend-7776cb7d57 to 4
```

Inspect the deployment and identify the current strategy - RollingUpdate

```
controlplane ~ ➜ kubectl describe deployments
Name:                   frontend
Namespace:              default
CreationTimestamp:      Mon, 03 Apr 2023 08:27:10 +0000
Labels:                 <none>
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               name=webapp
Replicas:               4 desired | 4 updated | 4 total | 4 available | 0 unavailable
StrategyType:           RollingUpdate
```

If you were to upgrade the application now what would happen? Pods upgraded few at a time .

Let us try that. Upgrade the application by setting the image on the deployment to kodekloud/webapp-color:v2
Do not delete and re-create the deployment. Only set the new image name for the existing deployment.

```
controlplane ~ ➜ kubectl edit deployment frontend
deployment.apps/frontend edited
```

Update the image .

Run the script curl-test.sh again. Notice the requests now hit both the old and newer versions. However none of them fail.
Execute the script at /root/curl-test.sh.

```
controlplane ~ ➜ ./curl-test.sh
Hello, Application Version: v2 ; Color: green OK

Hello, Application Version: v2 ; Color: green OK

Hello, Application Version: v2 ; Color: green OK

Hello, Application Version: v2 ; Color: green OK

Hello, Application Version: v2 ; Color: green OK

Hello, Application Version: v2 ; Color: green OK

Hello, Application Version: v2 ; Color: green OK

Hello, Application Version: v2 ; Color: green OK

Hello, Application Version: v2 ; Color: green OK
```

Up to how many PODs can be down for upgrade at a time
Consider the current strategy settings and number of PODs - 4
1

Change the deployment strategy to Recreate
Delete and re-create the deployment if necessary. Only update the strategy type for the existing deployment.

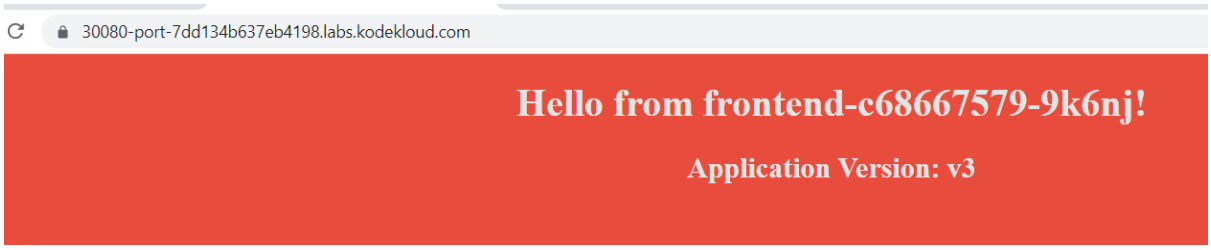
```
controlplane ~ ➜ kubectl edit deployment frontend
deployment.apps/frontend edited
```

---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend
  namespace: default
spec:
  replicas: 4
  selector:
    matchLabels:
      name: webapp
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        name: webapp
    spec:
      containers:
      - image: kodekloud/webapp-color:v2
        name: simple-webapp
        ports:
        - containerPort: 8080
          protocol: TCP

Upgrade the application by setting the image on the deployment to kodekloud/webapp-color:v3
Do not delete and re-create the deployment. Only set the new image name for the existing deployment.

```
controlplane ~ ➜ kubectl edit deployment frontend
deployment.apps/frontend edited
```

Update image url .



Application updated to v3 .