**Step 1:**

* docker pull nginx

**Step 2:**

* **Build**  
  docker build -t repository/nginx:1.16.1 --target ui -f .\nginxDockerfile .  
  docker build -t repository/nginx:1.19.4 --target ui -f .\nginxDockerfile .
* **Push**docker push repository/nginx:1.16.1  
  docker push repository/nginx:1.19.4

**Step 3:**  
**Kubernetes**

* Create Secret for private docker registry  
  kubectl create secret docker-registry regcred --docker-server=<your-registry-server> --docker-username=<your-name> --docker-password=<your-pword> --docker-email=<your-email>
* Create K8s v1 Deployment  
  kubectl create -f nginx-deployment-defination-v1.yaml –record
* Create K8s service for v1 Deployment  
  kubecly apply -f webapp-service-v1.yaml
* Check running Pods  
  kubectl get pods
* Check version of app in UI
* Create Canary Deployment  
  kubectl create -f nginx-canary-deployment-defination-v2.yaml –record
* Update the old service by removing the uncommon selector between both deployments

Kubectl apply -f webapp-service-canary-v2.yaml

* Check the traffic, it will be split between v1.0 and v2.0
* Rollback Canary Deployment  
  kubectl delete deployment webserver-canary-deployment

**Step 4:**

* Scaling Deployments to 4 replicas  
  kubectl scale --current-replicas=1 --replicas=4 deployment/webserver-deployment
* Update image in deployment using kubectl  
  kubectl set image deployment/webserver-deployment nginx-container=repository/nginx:1.19.4 –record
* Check Rollout Status for deployment using kubectl  
  kubectl rollout status deployment/webserver-deployment
* Check rollout history using kubectl  
  kubectl rollout history deployment.v1.apps/webserver-deployment
* Rollback apps to specific version using kubectl  
  kubectl rollout undo deployment.v1.apps/webserver-deployment --to-revision=1
* Scale Down Replicas to 2  
  kubectl scale --current-replicas=4 --replicas=2 deployment/webserver-deployment