IBM Cloud Private 3.1.2

Lab Exercise # 2

Prepare Helm chart, deploy , upgrade and rollback

Duration: 60 mins

Table of Contents

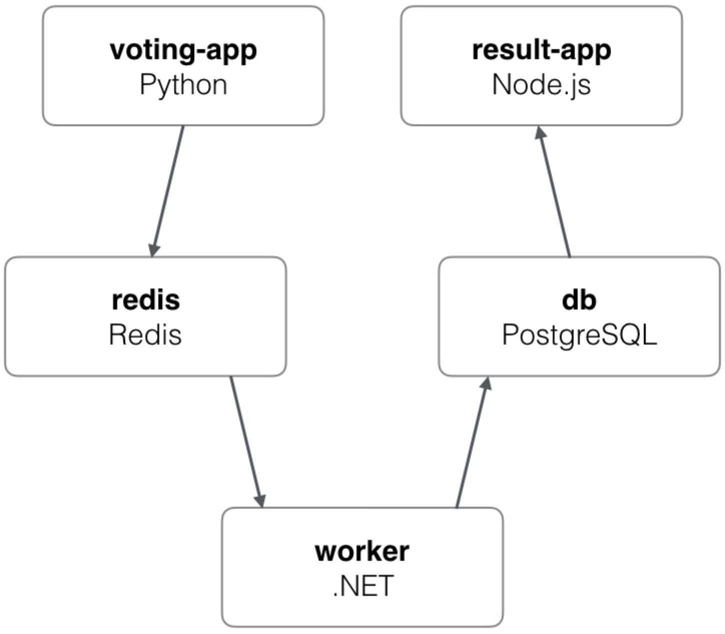
Objective

1. Deploy an application with multiple images/components into ICP
2. To create a helm chart for the same application
3. Deploy the chart from command line using helm
4. Access the deployed application
5. Upgrade the existing helm release with new version of the chart
6. Rollback the release to the older version.

Instructions

1. Introduction to the voting app to be deployed in this exercise

We are going to deploy a simple distributed application running across multiple containers.



Vote app : UI to vote for cats/dogs.

Redis db : Stores the vote from vote app.

Worker app: Pulls from redis and updates Postgress db

Postgress db: Stores the results of voting for the result app

Result app: UI to show results of voting

2. Deploy the sample app as is

Replace **<your-namespace>** with the namespace allocated for you for the duration of the lab exercises.

2.1 Set the target namespace to "vote"

```
$ kubectl config set-context mycluster-context --namespace=<your-namespace>
```

2.2 Create imagePullSecret.

Run the following command to create a docker-registry secret:

```
$ kubectl create secret docker-registry registry-secret-<your-index> --docker-username=<your-user-id> --docker-password=<your-user-password> --docker-server=mycluster.icp:8500 --docker-email=null
```

2.3 Patch default service account to use the imagePullSecret

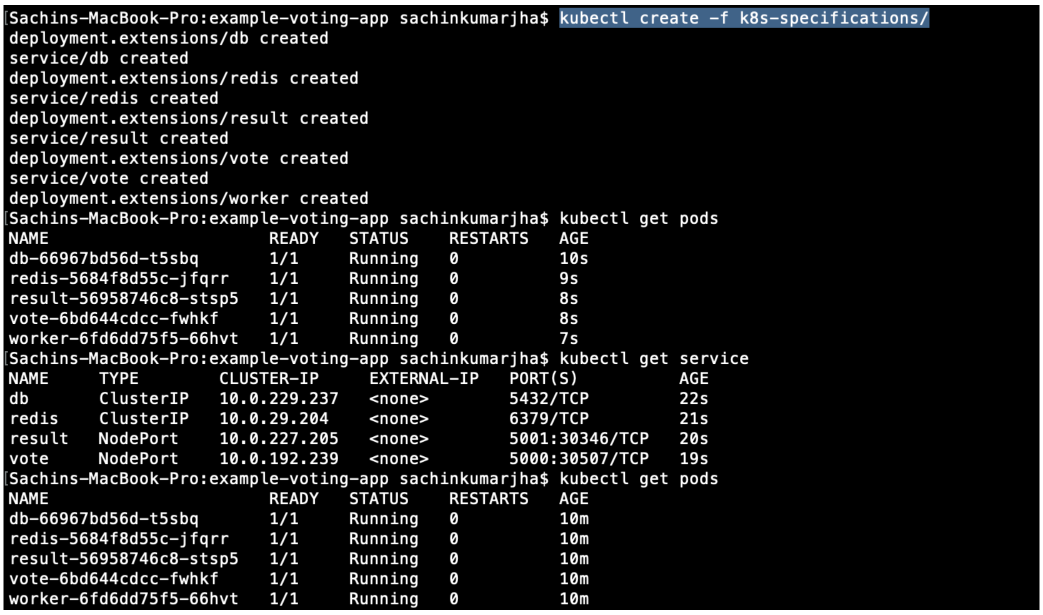
```
$ kubectl patch serviceaccount default -p '{"imagePullSecrets": [{"name": "registry-secret<your-index>" }]}'
```

This is required since the images have been pushed to the default namespace and by default the scope of images is 'namespace'.

As you would be deploying the images in your own namespace, the default service account in your namespace need to use imagePullSecret.

2.4 Deploy using the individual k8s deployment and service definition files

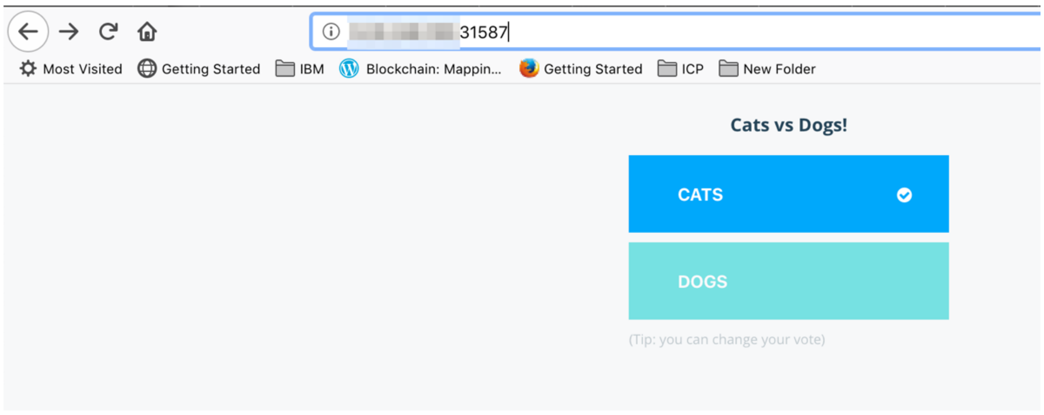
```
$ kubectl create -f k8s-specifications/
```



```
Sachins-MacBook-Pro:example-voting-app sachinkumarjha$ kubectl create -f k8s-specifications/
deployment.extensions/db created
service/db created
deployment.extensions/redis created
service/redis created
deployment.extensions/result created
service/result created
deployment.extensions/vote created
service/vote created
deployment.extensions/worker created
Sachins-MacBook-Pro:example-voting-app sachinkumarjha$ kubectl get pods
NAME                      READY   STATUS    RESTARTS   AGE
db-66967bd56d-t5sbq       1/1     Running   0          10s
redis-5684f8d55c-jfqrr    1/1     Running   0          9s
result-56958746c8-stsp5   1/1     Running   0          8s
vote-6bd644cdcc-fwhkf     1/1     Running   0          8s
worker-6fd6dd75f5-66hvt   1/1     Running   0          7s
Sachins-MacBook-Pro:example-voting-app sachinkumarjha$ kubectl get service
NAME     TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
db       ClusterIP   10.0.229.237   <none>        5432/TCP         22s
redis    ClusterIP   10.0.29.204    <none>        6379/TCP         21s
result   NodePort    10.0.227.205   <none>        5001:30346/TCP   20s
vote     NodePort    10.0.192.239   <none>        5000:30507/TCP   19s
Sachins-MacBook-Pro:example-voting-app sachinkumarjha$ kubectl get pods
NAME                      READY   STATUS    RESTARTS   AGE
db-66967bd56d-t5sbq       1/1     Running   0          10m
redis-5684f8d55c-jfqrr    1/1     Running   0          10m
result-56958746c8-stsp5   1/1     Running   0          10m
vote-6bd644cdcc-fwhkf     1/1     Running   0          10m
worker-6fd6dd75f5-66hvt   1/1     Running   0          10m
```

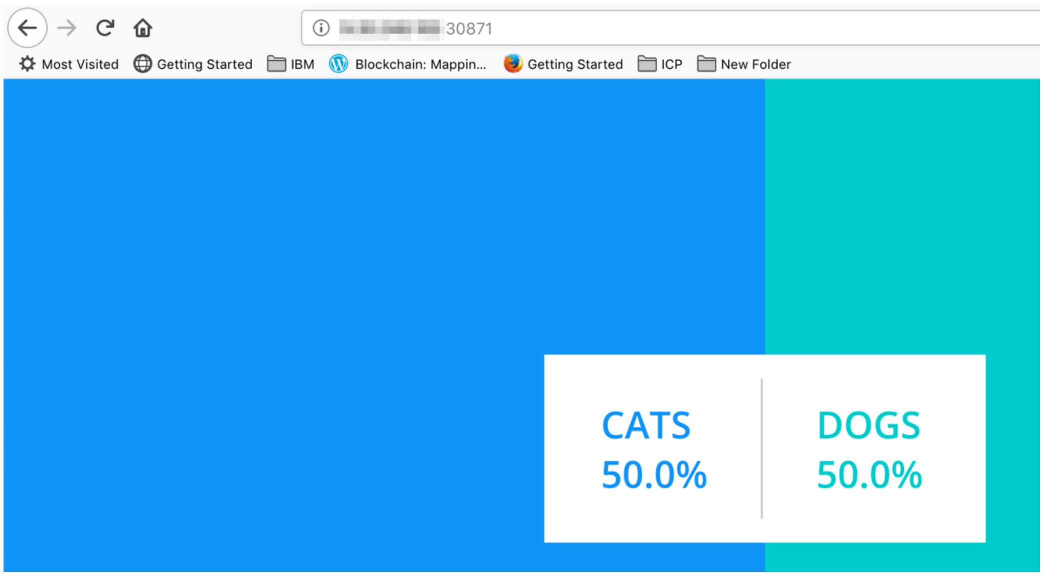
2.5 Access the vote and result app at given NodePorts

Vote app: http://<worker_ip>:<voteServiceNodePort>



[redacted] 31587

Result app: http://<worker_ip>:<resultServiceNodePort>



3. Create helm Chart

3.1 Create default helm chart

cd to <project root directory>

$ helm create voting-app-helm-charts

This creates the folder structure for a helm chart.

Follow the steps below. In case of doubts refer the modified chart in the sample repo here:

https://github.com/sachinjha/example-voting-app/tree/charts/voting-app-helm-charts

3.2 Create required charts for micro services.

Go to <projectroot>/voting-app-helm-charts/charts folder

$ helm create db

$ helm create redis

$ helm create result

$ helm create worker

$ helm create vote

3.3 Modify the db chart templates

3.4.1 Replace the current deployment.yaml and service.yaml files under db/templates/ with the contents from k8s-specification/db-deployment.yaml and k8s-specification/db-service.yaml

3.4.2 Delete ingress.yaml and NOTES.txt files

```
$ rm db/templates/ingress.yaml
$ rm db/templates/NOTES.txt
```

Add the file templates/pvc.yaml with the following contents:

```
# © Copyright IBM Corporation 2018
#
# Licensed under the Apache License, Version 2.0 (the "License");
# you may not use this file except in compliance with the License.
# You may obtain a copy of the License at
#
# http://www.apache.org/licenses/LICENSE-2.0
#
# Unless required by applicable law or agreed to in writing, software
# distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
# See the License for the specific language governing permissions and
# limitations under the License.

kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: postgres-pvc
  labels:
    app: {{ .Release.Name }}
    chart: "{{ .Chart.Name }}-{{ .Chart.Version }}"
    release: "{{ .Release.Name }}"
    heritage: "{{ .Release.Service }}"
spec:
  {{- if .Values.dataPVC.useDynamicProvisioning }}
  ## if present, use the storageClassName from the values.yaml, else use the
  ## default storageClass setup by Kubernetes Administrator
  ##
  ## setting storageClassName to nil means use the default storage class
  storageClassName: {{ default nil .Values.dataPVC.storageClassName | quote }}
  {{- else }}
  ## disable dynamic provisioning
  storageClassName: ""
  {{- end }}
  accessModes:
    - {{ .Values.dataPVC.accessMode | quote }}
  resources:
    requests:
      storage: {{ .Values.dataPVC.size | quote }}
```

Replace the contents of values.yaml file with the following:

```
# Default values for result.
# This is a YAML-formatted file.
# Declare variables to be passed into your templates.

replicaCount: 1
dataPVC:
  name: db-pvc
  storageClassName: rbd
  useDynamicProvisioning: true
  accessMode: ReadWriteOnce
  size: 5Gi
```

3.4 Modify the redis chart templates

Replace the current deployment.yaml and service.yaml files under redis/templates/ with the contents from k8s-specification/redis-deployment.yaml and k8s-specification/redis-service.yaml

Delete ingress.yaml, NOTES.txt and values.yaml

```
$ rm db/templates/ingress.yaml
$ rm db/templates/NOTES.txt
$ rm db/values.yaml
```

3.5 Modify the vote chart

Modify the values.yaml under vote chart as follows:

```
replicaCount: 1
```

```
image:
  repository: mycluster.icp:8500/default/vote
  tag: 0.1.0
  pullPolicy: IfNotPresent

service:
  type: NodePort
  port: 80
```

3.6 Modify the result chart

Modify the values.yaml under result chart as follows:

```
replicaCount: 1
image:
  repository: mycluster.icp:8500/default/result
  tag: 0.1.0
  pullPolicy: IfNotPresent

service:
  type: NodePort
  port: 80
```

Leave other values as-is.

3.7 Modify the worker chart

Update values.yaml under worker chart as follows:

```
replicaCount: 1

image:
  repository: mycluster.icp:8500/default/worker
  tag: 0.1.0
  pullPolicy: IfNotPresent
```

Remove the section on service as its not required for worker app.

Delete the following files

- templates/service.yaml
- templates/ingress.yaml.

3.8 Update the top level values.yaml

Add parameters so that any of included chart parameters can be configured during install.
Update the voting-app-helm-charts/values.yaml as follows:

```yaml
# Default values for voting-app-helm-charts.
# This is a YAML-formatted file.
# Declare variables to be passed into your templates.

global:
  serviceAccountName: default

worker:
  replicaCount: 1

  image:
    repository: mycluster.icp:8500/default/worker
    tag: 0.1.0
    pullPolicy: IfNotPresent

vote:
  replicaCount: 1

  image:
    repository: mycluster.icp:8500/default/vote
    tag: 0.1.0
    pullPolicy: IfNotPresent

  service:
    type: NodePort
    port: 80

result:
  replicaCount: 1

  image:
    repository: mycluster.icp:8500/default/result
    tag: 0.1.0
    pullPolicy: IfNotPresent

  service:
    type: NodePort
    port: 80

db:
  dataPVC:
    name: db-pvc
    storageClassName:
    useDynamicProvisioning: true
    accessMode: ReadWriteOnce
    size: 5Gi
```

The above parameters are same as in the individual chart's values.yaml.

3.9 Validate the charts

```
$ helm lint charts/db
$ helm lint charts/result
$ helm lint charts/vote
$ helm lint charts/worker
$ helm lint charts/redis
$ cd ..  ( move to the project root folder )
$ helm lint voting-app-helm-charts
```

There should be no errors during validation.

3.10 Package the helm chart for distribution (optional)

In case you want to add the chart to a repository or share it with someone, there is a step to package the chart which creates a .tgz file

$ helm package voting-app-helm-charts

3.11 **Install the chart**

Delete the existing deployment and services created from step 2

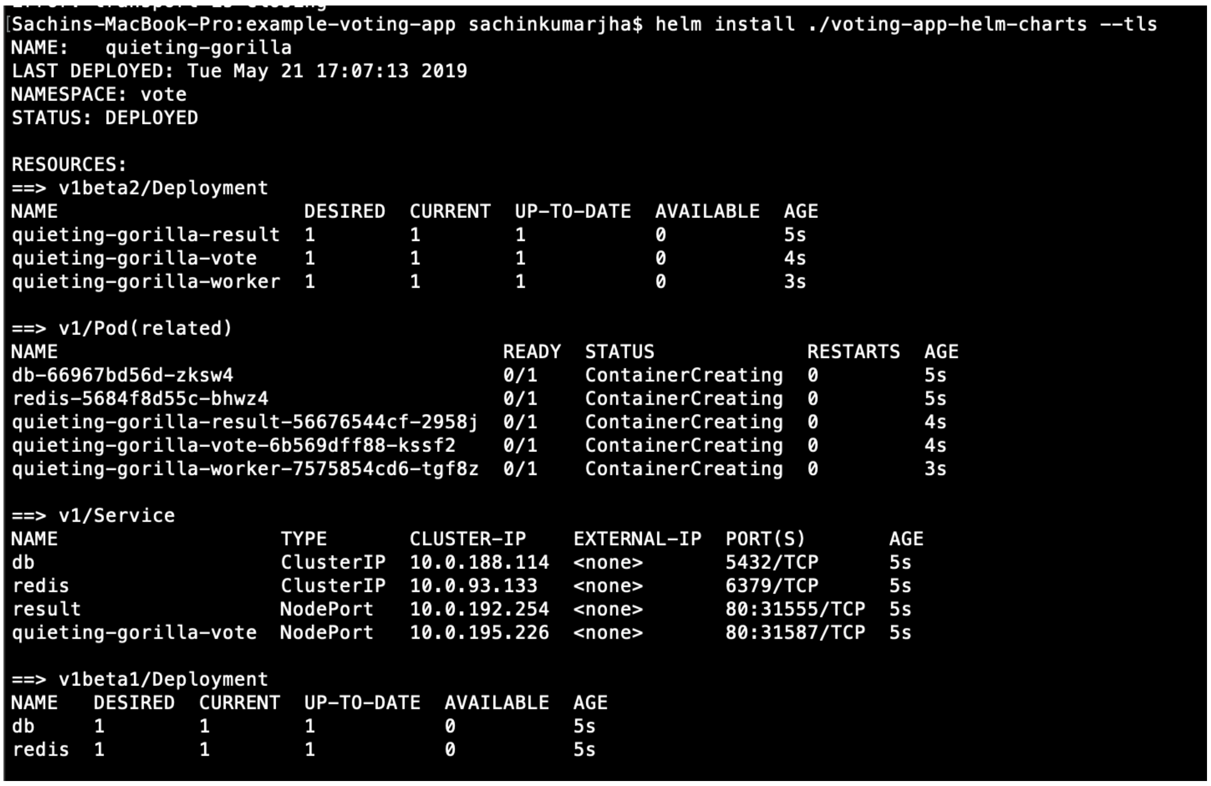
$ kubectl delete -f ./k8s-specifications

Set the current namespace to ***<your-namespace>***

$ *kubectl config set-context mycluster-context --namespace=**<your-namespace>***

Install the voting app

$ helm install ./voting-app-helm-charts --name voting-app



```
[Sachins-MacBook-Pro:example-voting-app sachinkumarjha$ helm install ./voting-app-helm-charts --tls
NAME:   quieting-gorilla
LAST DEPLOYED: Tue May 21 17:07:13 2019
NAMESPACE: vote
STATUS: DEPLOYED

RESOURCES:
==> v1beta2/Deployment
NAME                     DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
quieting-gorilla-result  1        1        1           0          5s
quieting-gorilla-vote    1        1        1           0          4s
quieting-gorilla-worker  1        1        1           0          3s

==> v1/Pod(related)
NAME                                      READY  STATUS             RESTARTS  AGE
db-66967bd56d-zksw4                       0/1    ContainerCreating  0         5s
redis-5684f8d55c-bhwz4                    0/1    ContainerCreating  0         5s
quieting-gorilla-result-56676544cf-2958j  0/1    ContainerCreating  0         4s
quieting-gorilla-vote-6b569dff88-kssf2    0/1    ContainerCreating  0         4s
quieting-gorilla-worker-7575854cd6-tgf8z  0/1    ContainerCreating  0         3s

==> v1/Service
NAME                   TYPE       CLUSTER-IP    EXTERNAL-IP  PORT(S)       AGE
db                     ClusterIP  10.0.188.114  <none>       5432/TCP      5s
redis                  ClusterIP  10.0.93.133   <none>       6379/TCP      5s
result                 NodePort   10.0.192.254  <none>       80:31555/TCP  5s
quieting-gorilla-vote  NodePort   10.0.195.226  <none>       80:31587/TCP  5s

==> v1beta1/Deployment
NAME   DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
db     1        1        1           0          5s
redis  1        1        1           0          5s
```

You should see results as shown above. After some time, the pvc should be bound to a pv and pods should be up and running.
Access the vote application and result application at the respective nodePorts.
In this example (based on screen shot above)
Vote app : http://<worker_ip>:31587
Result app: http://<worker_ip>:31555

4. Modify the charts and upgrade to a new version.

Let's update the helm release with the newer version of chart.

For the purpose of this lab, small modifications are done to the images and they are uploaded into ICP private registry with versions 1.1.

We shall update the chart details to pull the updated version of images and then upgrade the helm release version.

4.1 **Update the vote image version in different files** , in helm chart

4.1.1 Update <project root>/voting-app-helm-charts/Chart.yaml

```
apiVersion: v1
appVersion: "1.0"
description: A Helm chart for Kubernetes
name: voting-app-helm-charts
version: 0.1.1
```

4.1.2 Update <project root>/voting-app-helm-charts/charts/vote/Chart.yaml

```
apiVersion: v1
appVersion: "1.0"
description: A Helm chart for Kubernetes
name: vote
version: 0.1.1
```

4.1.3 Update <project root>/ voting-app-helm-charts/charts/vote/values.yaml

```
image:
  repository: mycluster.icp:8500/default/vote
  tag: 0.1.1
  pullPolicy: IfNotPresent
```

4.1.4 Update <project root>/ voting-app-helm-charts/values.yaml

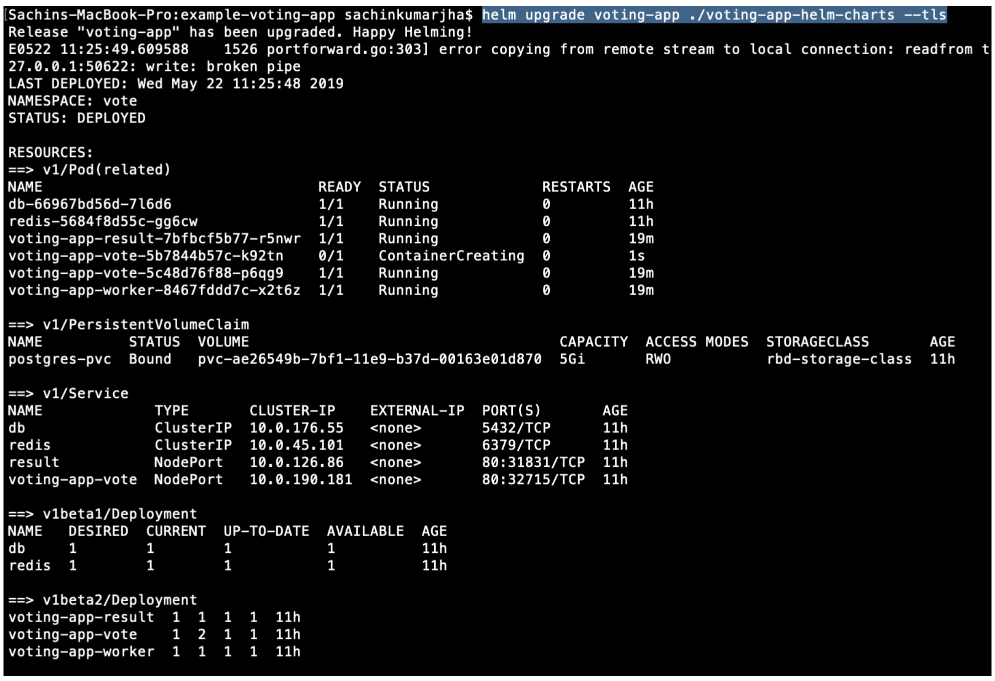
```
vote:
 replicaCount: 1

 image:
  repository: mycluster.icp:8500/default/vote
  tag: 0.1.1
  pullPolicy: IfNotPresent
```

4.1.5 Package charts and observe that the new version of archive is v0.1.1

```
$ helm package voting-app-helm-charts
```

4.2 Upgrade existing helm release with new version of the chart.

$ helm upgrade voting-app ./voting-app-helm-charts –tls

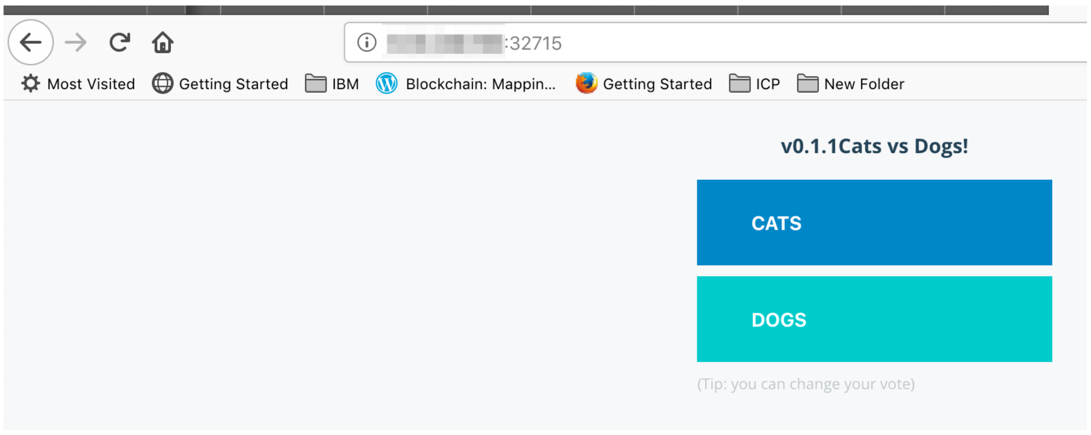


```
Sachins-MacBook-Pro:example-voting-app sachinkumarjha$ helm upgrade voting-app ./voting-app-helm-charts --tls
Release "voting-app" has been upgraded. Happy Helming!
E0522 11:25:49.609588    1526 portforward.go:303] error copying from remote stream to local connection: readfrom t
27.0.0.1:50622: write: broken pipe
LAST DEPLOYED: Wed May 22 11:25:48 2019
NAMESPACE: vote
STATUS: DEPLOYED

RESOURCES:
==> v1/Pod(related)
NAME                                READY  STATUS             RESTARTS  AGE
db-66967bd56d-7l6d6                 1/1    Running            0         11h
redis-5684f8d55c-gg6cw              1/1    Running            0         11h
voting-app-result-7bfbcf5b77-r5nwr  1/1    Running            0         19m
voting-app-vote-5b7844b57c-k92tn    0/1    ContainerCreating  0         1s
voting-app-vote-5c48d76f88-p6qg9    1/1    Running            0         19m
voting-app-worker-8467fddd7c-x2t6z  1/1    Running            0         19m

==> v1/PersistentVolumeClaim
NAME          STATUS  VOLUME                                    CAPACITY  ACCESS MODES  STORAGECLASS       AGE
postgres-pvc  Bound   pvc-ae26549b-7bf1-11e9-b37d-00163e01d870  5Gi       RWO           rbd-storage-class  11h

==> v1/Service
NAME             TYPE       CLUSTER-IP    EXTERNAL-IP  PORT(S)       AGE
db               ClusterIP  10.0.176.55   <none>       5432/TCP      11h
redis            ClusterIP  10.0.45.101   <none>       6379/TCP      11h
result           NodePort   10.0.126.86   <none>       80:31831/TCP  11h
voting-app-vote  NodePort   10.0.190.181  <none>       80:32715/TCP  11h

==> v1beta1/Deployment
NAME   DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
db     1        1        1           1          11h
redis  1        1        1           1          11h

==> v1beta2/Deployment
voting-app-result  1  1  1  1  11h
voting-app-vote    1  2  1  1  11h
voting-app-worker  1  1  1  1  11h
```

Notice that the new voting-app pod is getting created and in some time the existing one will be deleted.

4.3 Access the app now (URL is still the same)

Observe that the voting app now shows version v0.1.1



5. Rollback to older version

5.1 Check the history of the versions available.

```
$ helm history voting-app –tls
```

1.2 Rollback to the desired version.

```
$ helm rollback voting-app 1 –tls
```

1.3 Observe the chart version via CLI and application in UI

```
$ helm list –tls
```

UI would show the voting app page without version.

6. Clean up

```
$ helm delete –purge voting-app
```

Summary

We have gone through the following steps:

1) Looked at the existing voting app as is.
2) Deployed the existing app using individual deployment files.
3) Created the helm chart with dependencies
4) Validate the helm chart via Lint
5) Installed the initial version of the chart
6) Upgraded the helm release to a new version and rolled back to older version.