What is a Helm Chart?

A **Helm Chart** is like a package manager for Kubernetes—just like `apt` for Ubuntu or `yum` for CentOS. It helps you define, install, and upgrade even complex Kubernetes applications in a repeatable way.

A Helm chart usually contains:

- **Chart.yaml** – Metadata about the chart.
- **values.yaml** – Default config values (like variables).
- **Templates folder** – Kubernetes YAML files, but with placeholders (using Go templating syntax like `{{ .Values.image.repository }}`).

Assignment:
Deploying Jenkins on Kubernetes using Helm.

Installation:

Step1: check helm version

```
helm version
```

Step2: Create folder structure:

```
mkdir jenkins-chart && cd jenkins-chart
helm create jenkins
```

Step3: add chart.yaml

```
apiVersion: v2
appVersion: 1.16.0
description: A Helm chart for Kubernetes
name: jenkins
type: application
version: 0.1.0
```

Values.yaml

```
labels:
  app: jenkins
replicas: 2
rollingupdate:
  maxsurge: 2
  unavailable: 0
volume:
  name: local-volume
  claimname: jenkins-pvc
  storage: 1Gi
image:
  registry: jenkins
  repository: jenkins
  tag: latest
ports:
  port: 8080
  targetport: 8080
  nodeport: 30080
```

Pv.yaml

```
apiVersion: v1                       # API version for this resource
kind: PersistentVolume               # Resource type: PersistentVolume
metadata:
  name: {{ .Values.volume.name }}                      # Name of the
PersistentVolume
spec:
  capacity:
    storage: {{ .Values.volume.storage }}                    # Storage capacity
of this volume
  accessModes:
    - ReadWriteOnce                  # RWO means volume can be mounted as
read-write by a single node
  persistentVolumeReclaimPolicy: Retain # What happens to PV when PVC is deleted
(Retain keeps data)
  storageClassName: manual           # Storage class name for matching with PVCs
  hostPath:
    path: /mnt/data                  # Path on the host node's filesystem to be
used for storage
  nodeAffinity:                      # Constraints for which nodes this PV can be
used on
    required:
      nodeSelectorTerms:
        - matchExpressions:
            - key: env               # The node label key to match
```

```
              operator: In             # Match if the node's label value is in the
following values
              values:
                - dev                  # The node must have label env=dev
```

Pvc.yaml

```
apiVersion: v1                       # API version for this resource
kind: PersistentVolumeClaim          # Resource type: PersistentVolumeClaim
metadata:
  name: {{ .Values.volume.claimname }}                   # Name of the PVC
spec:
  accessModes:
    - ReadWriteOnce                  # RWO means volume can be mounted as
read-write by a single node
  resources:
    requests:
      storage: {{ .Values.volume.storage }}                  # Amount of
storage requested by this claim
  storageClassName: manual           # Storage class name to match with
available PVs
```

templayes/Deployment.yml

```
apiVersion: apps/v1                    # API version for Deployment resources
kind: Deployment                       # Resource type: Deployment
metadata:
  name: {{ .Release.Name }}-deploy   # jenkins-deploy # Name of the deployment
  labels:
    app: {{ .Values.labels.app }}                   # Label applied to the deployment itself
spec:
  replicas: {{ .Values.replicas }}                       # Number of pod replicas to maintain
  strategy:
    type: RollingUpdate                # Update strategy: gradually replace pods
    rollingUpdate:
      maxSurge: {{ .Values.rollingupdate.maxsurge }}                      # Max number of pods that
can exceed the replica count during update
      maxUnavailable: {{ .Values.rollingupdate.unavailable }}                # Max number of pods
that can be unavailable during update
  selector:
    matchLabels:
      app: {{ .Values.labels.app }}                 # Selects which pods are managed by this
deployment
  template:
    metadata:
      labels:
        app: {{ .Values.labels.app }}              # Labels applied to the pods created by this
template
    spec:
      volumes:
        - name: {{ .Values.volume.name }}           # Name of the volume, referenced by volumeMounts
          persistentVolumeClaim:
            claimName: {{ .Values.volume.claimname }}     # Name of the PVC to use for this volume
      containers:
        - name: {{ .Release.Name }}               # Container name
          image: {{ .Values.image.registry }}/{{ .Values.image.repository }}:{{ .Values.image.tag }}
# Container image to use
          volumeMounts:
            - name: {{ .Values.volume.name }}       # Name of the volume to mount (must match volume
name above)
              mountPath: /var/jenkins_home # Where to mount the volume in the container
      securityContext:
        runAsUser: 0
        fsGroup: 0
```

Service.yaml

```
apiVersion: v1                       # API version for Service resources
kind: Service                        # Resource type: Service
metadata:
  name: {{ .Release.Name }}-service               # Name of the service
spec:
  type: NodePort                     # Service type: makes service accessible on a port on each
node
  selector:
    app: {{ .Values.labels.app }}                  # Selects pods with this label to send traffic
to
  ports:
    - port: {{ .Values.ports.port }}                      # Port the service exposes
      targetPort: {{ .Values.ports.targetport }}                # Port on the pod that traffic is
sent to
      nodePort: {{ .Values.ports.nodeport }}                  # Port on each node where service is
accessible (30000-32767)
```

Dry run

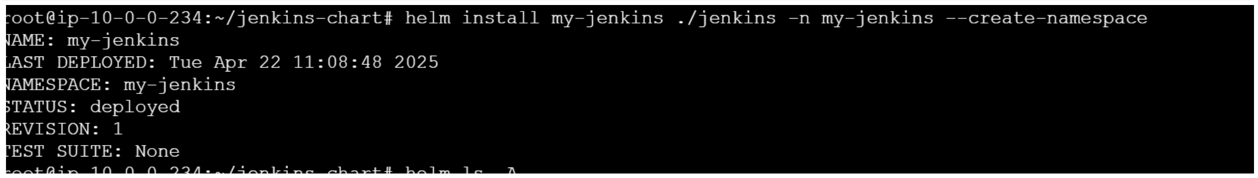
```
helm install my-jenkins ./jenkins -n my-jenkins
--create-namespace --dry-run
```

If you get errors: delete these files

```
rm -rf
./jenkins/templates/{ingress.yaml,serviceaccount.yaml,hpa.y
aml,NOTES.txt,tests}
```

Actual Install

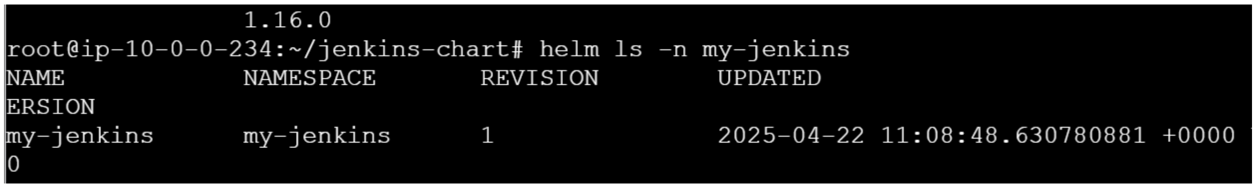
```
helm install my-jenkins ./jenkins -n my-jenkins
--create-namespace
```



```
oot@ip-10-0-0-234:~/jenkins-chart# helm install my-jenkins ./jenkins -n my-jenkins --create-namespace
AME: my-jenkins
AST DEPLOYED: Tue Apr 22 11:08:48 2025
AMESPACE: my-jenkins
TATUS: deployed
EVISION: 1
EST SUITE: None
oot@ip-10-0-0-234:~/jenkins-chart# helm ls -A
```

Check release status

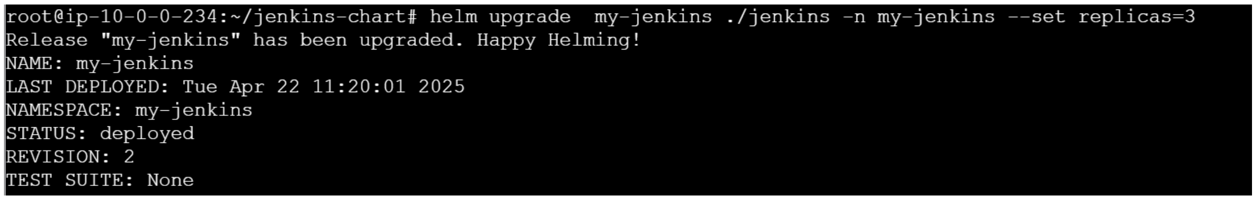
```
helm ls -n my-jenkins
```



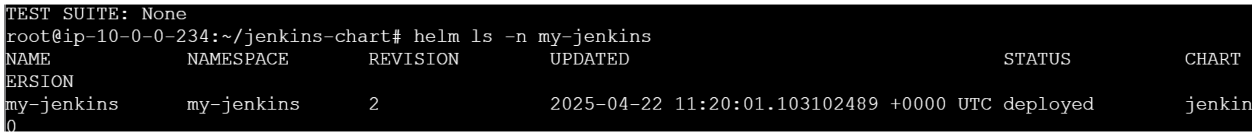
```
                1.16.0
root@ip-10-0-0-234:~/jenkins-chart# helm ls -n my-jenkins
NAME            NAMESPACE       REVISION        UPDATED
ERSION
my-jenkins      my-jenkins      1               2025-04-22 11:08:48.630780881 +0000
0
```

To upgrade:

```
helm upgrade  my-jenkins ./ -n my-jenkins --set replicas=3
```



```
root@ip-10-0-0-234:~/jenkins-chart# helm upgrade  my-jenkins ./jenkins -n my-jenkins --set replicas=3
Release "my-jenkins" has been upgraded. Happy Helming!
NAME: my-jenkins
LAST DEPLOYED: Tue Apr 22 11:20:01 2025
NAMESPACE: my-jenkins
STATUS: deployed
REVISION: 2
TEST SUITE: None
```



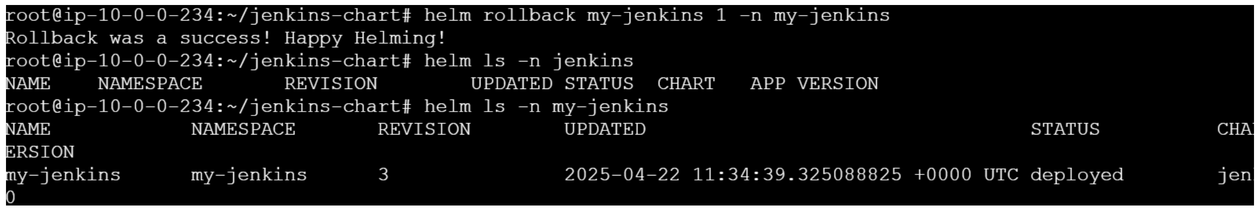
```
TEST SUITE: None
root@ip-10-0-0-234:~/jenkins-chart# helm ls -n my-jenkins
NAME            NAMESPACE       REVISION        UPDATED                                 STATUS          CHART
ERSION
my-jenkins      my-jenkins      2               2025-04-22 11:20:01.103102489 +0000 UTC deployed        jenkin
0
```

To fetch everything Helm knows about the installation.

```
helm get all my-jenkins -n my-jenkins
```

Rollback to previous version:

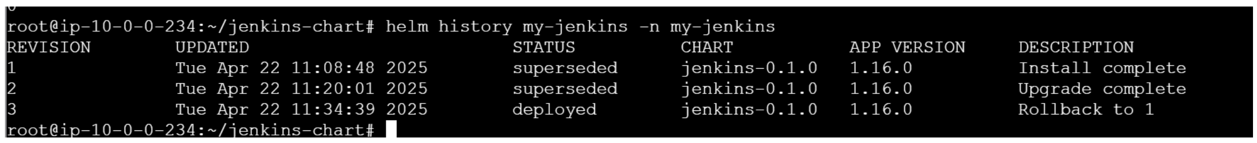
```
helm rollback my-jenkins 1 -n my-jenkins
```



```
root@ip-10-0-0-234:~/jenkins-chart# helm rollback my-jenkins 1 -n my-jenkins
Rollback was a success! Happy Helming!
root@ip-10-0-0-234:~/jenkins-chart# helm ls -n jenkins
NAME    NAMESPACE       REVISION        UPDATED STATUS  CHART   APP VERSION
root@ip-10-0-0-234:~/jenkins-chart# helm ls -n my-jenkins
NAME            NAMESPACE       REVISION        UPDATED                                 STATUS          CHA
ERSION
my-jenkins      my-jenkins      3               2025-04-22 11:34:39.325088825 +0000 UTC deployed        jen
0
```

To check history:

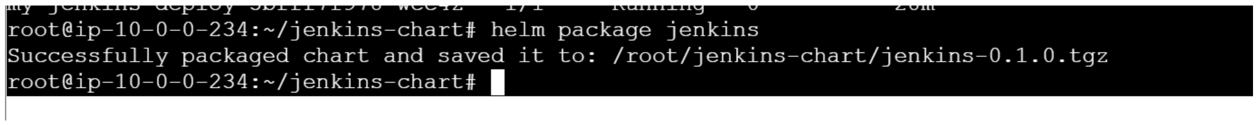
```
helm history my-jenkins -n my-jenkins
```



```
root@ip-10-0-0-234:~/jenkins-chart# helm history my-jenkins -n my-jenkins
REVISION        UPDATED                         STATUS          CHART           APP VERSION     DESCRIPTION
1               Tue Apr 22 11:08:48 2025        superseded      jenkins-0.1.0   1.16.0          Install complete
2               Tue Apr 22 11:20:01 2025        superseded      jenkins-0.1.0   1.16.0          Upgrade complete
3               Tue Apr 22 11:34:39 2025        deployed        jenkins-0.1.0   1.16.0          Rollback to 1
root@ip-10-0-0-234:~/jenkins-chart#
```

To Package: Package your chart (creates `.tgz` file)

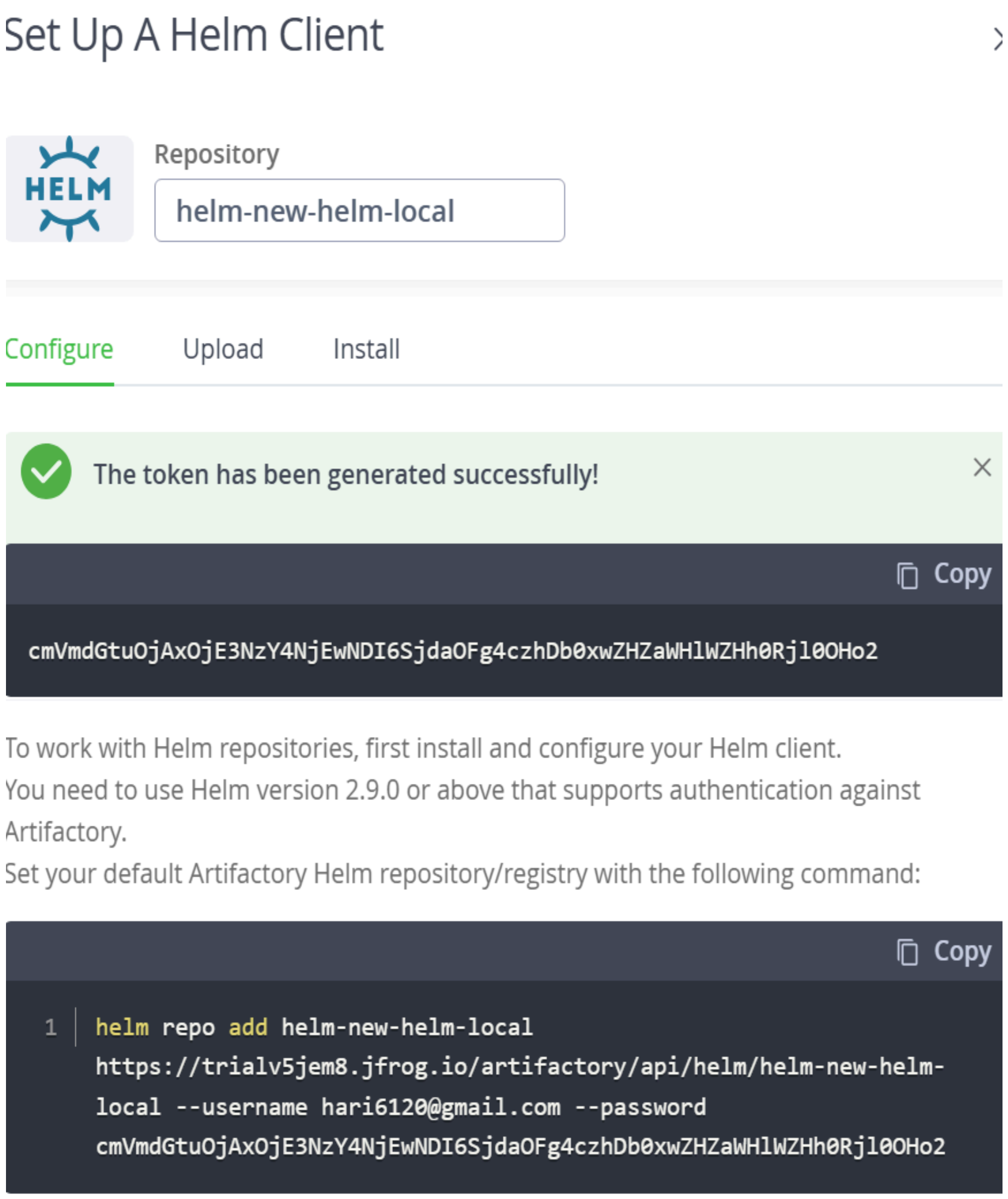
```
helm package jenkins
```



```
root@ip-10-0-0-234:~/jenkins-chart# helm package jenkins
Successfully packaged chart and saved it to: /root/jenkins-chart/jenkins-0.1.0.tgz
root@ip-10-0-0-234:~/jenkins-chart#
```

Uploading to JFrog Artifactory:

> Create a repository with helm
> Click on Set me up and get commands to setup the repo



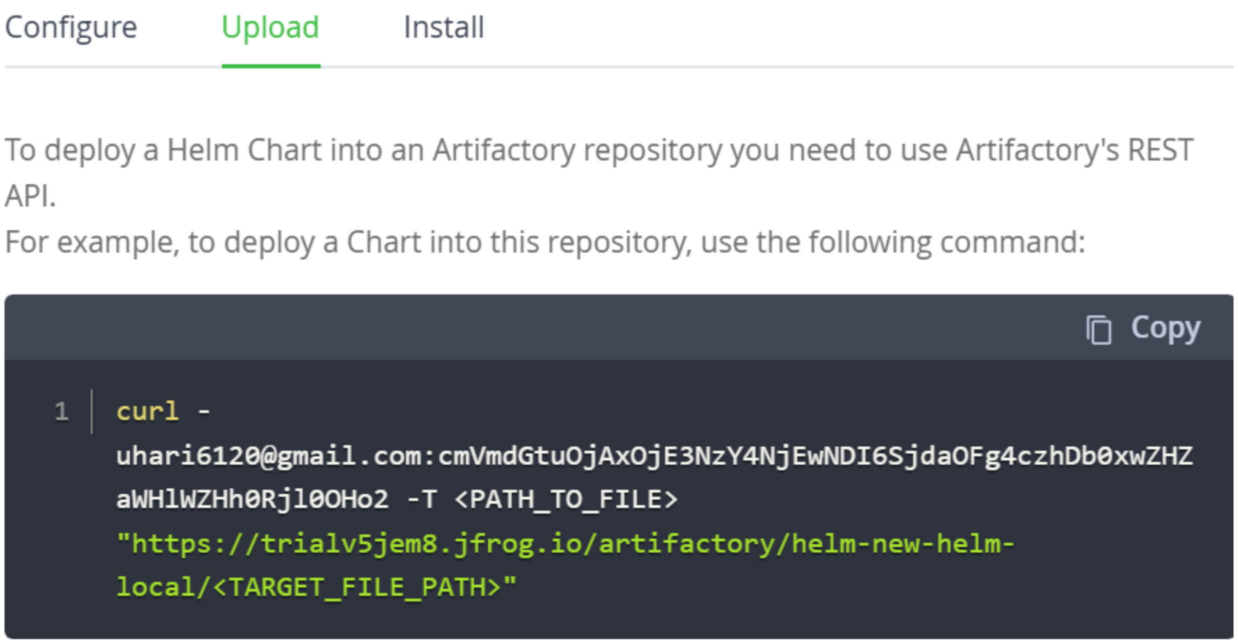
The token has been generated successfully!

```
cmVmdGtuOjAxOjE3NzY4NjEwNDI6SjdaOFg4czhDb0xwZHZaWHlWZHh0Rjl0OHo2
```

To work with Helm repositories, first install and configure your Helm client.
You need to use Helm version 2.9.0 or above that supports authentication against Artifactory.
Set your default Artifactory Helm repository/registry with the following command:

```
helm repo add helm-new-helm-local
https://trialv5jem8.jfrog.io/artifactory/api/helm/helm-new-helm-
local --username hari6120@gmail.com --password
cmVmdGtuOjAxOjE3NzY4NjEwNDI6SjdaOFg4czhDb0xwZHZaWHlWZHh0Rjl0OHo2
```

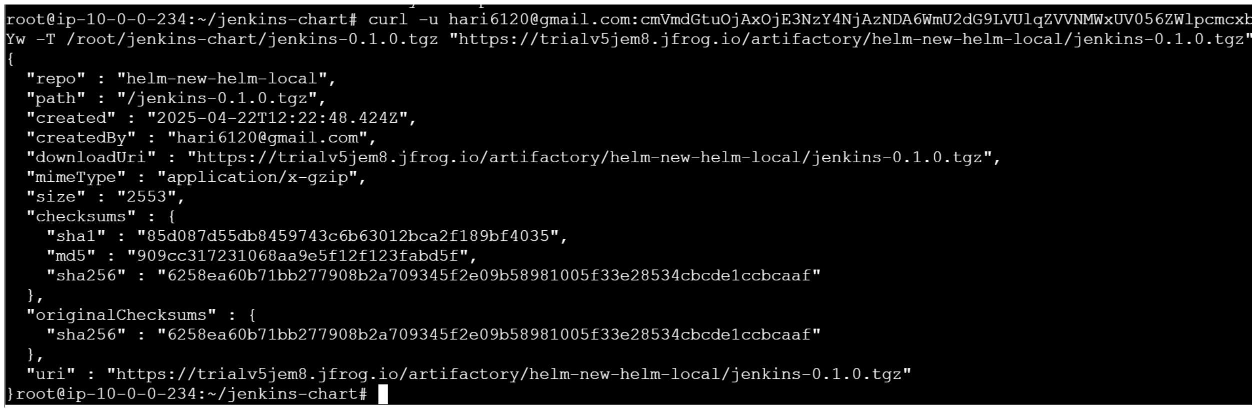
Use the curl command to upload



Configure Upload Install

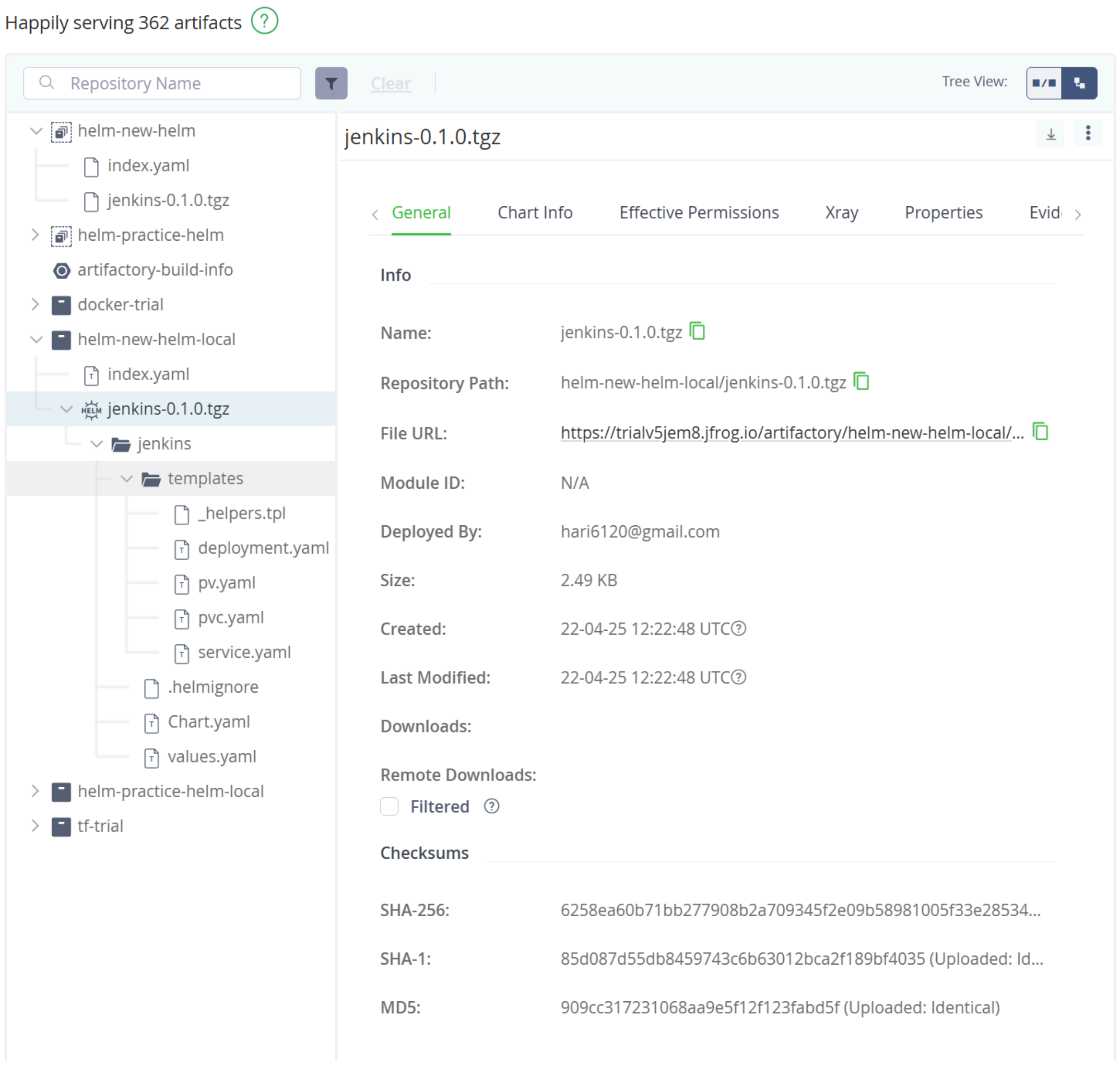
To deploy a Helm Chart into an Artifactory repository you need to use Artifactory's REST API.
For example, to deploy a Chart into this repository, use the following command:

```
curl -
uhari6120@gmail.com:cmVmdGtuOjAxOjE3NzY4NjEwNDI6SjdaOFg4czhDb0xwZHZ
aWHlWZHh0Rjl0OHo2 -T <PATH_TO_FILE>
"https://trialv5jem8.jfrog.io/artifactory/helm-new-helm-
local/<TARGET_FILE_PATH>"
```



```
root@ip-10-0-0-234:~/jenkins-chart# curl -u hari6120@gmail.com:cmVmdGtuOjAxOjE3NzY4NjAzNDA6WmU2dG9LVUlqZVVNMWxUV056ZWlpcmcxb
Yw -T /root/jenkins-chart/jenkins-0.1.0.tgz "https://trialv5jem8.jfrog.io/artifactory/helm-new-helm-local/jenkins-0.1.0.tgz"
{
  "repo" : "helm-new-helm-local",
  "path" : "/jenkins-0.1.0.tgz",
  "created" : "2025-04-22T12:22:48.424Z",
  "createdBy" : "hari6120@gmail.com",
  "downloadUri" : "https://trialv5jem8.jfrog.io/artifactory/helm-new-helm-local/jenkins-0.1.0.tgz",
  "mimeType" : "application/x-gzip",
  "size" : "2553",
  "checksums" : {
    "sha1" : "85d087d55db8459743c6b63012bca2f189bf4035",
    "md5" : "909cc317231068aa9e5f12f123fabd5f",
    "sha256" : "6258ea60b71bb277908b2a709345f2e09b58981005f33e28534cbcde1ccbcaaf"
  },
  "originalChecksums" : {
    "sha256" : "6258ea60b71bb277908b2a709345f2e09b58981005f33e28534cbcde1ccbcaaf"
  },
  "uri" : "https://trialv5jem8.jfrog.io/artifactory/helm-new-helm-local/jenkins-0.1.0.tgz"
}root@ip-10-0-0-234:~/jenkins-chart#
```

Uploaded Artifact:


Happily serving 362 artifacts

```
helm-new-helm
    index.yaml
    jenkins-0.1.0.tgz
helm-practice-helm
artifactory-build-info
docker-trial
helm-new-helm-local
    index.yaml
    jenkins-0.1.0.tgz
        jenkins
            templates
                _helpers.tpl
                deployment.yaml
                pv.yaml
                pvc.yaml
                service.yaml
            .helmignore
            Chart.yaml
            values.yaml
helm-practice-helm-local
tf-trial
```

jenkins-0.1.0.tgz

Info

Name:	jenkins-0.1.0.tgz
Repository Path:	helm-new-helm-local/jenkins-0.1.0.tgz
File URL:	https://trialv5jem8.jfrog.io/artifactory/helm-new-helm-local/...
Module ID:	N/A
Deployed By:	hari6120@gmail.com
Size:	2.49 KB
Created:	22-04-25 12:22:48 UTC
Last Modified:	22-04-25 12:22:48 UTC
Downloads:	
Remote Downloads:	

☐ Filtered

Checksums

SHA-256:	6258ea60b71bb277908b2a709345f2e09b58981005f33e28534...
SHA-1:	85d087d55db8459743c6b63012bca2f189bf4035 (Uploaded: Id...
MD5:	909cc317231068aa9e5f12f123fabd5f (Uploaded: Identical)