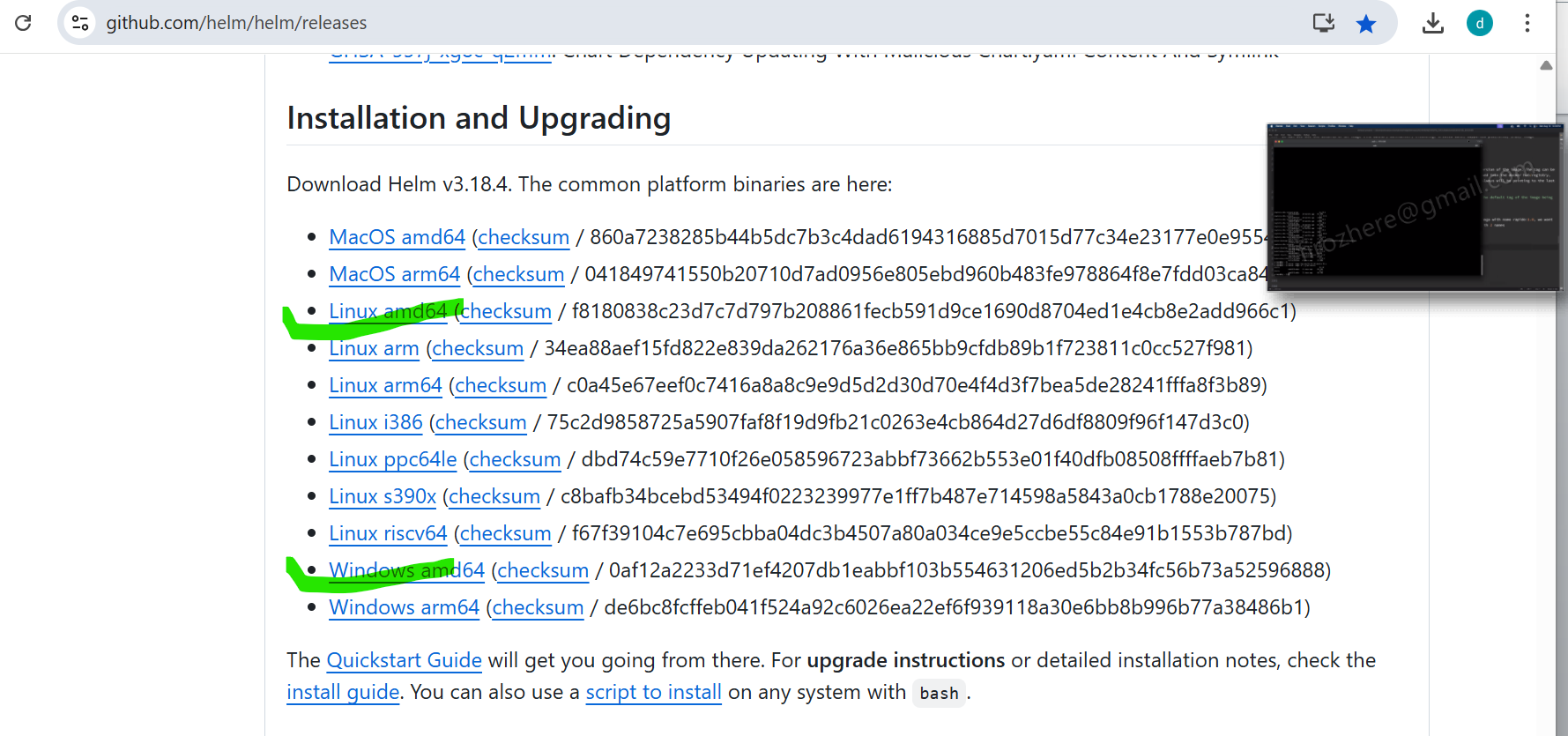
Helm: it is package manager of kubernates

#how to install on windows:

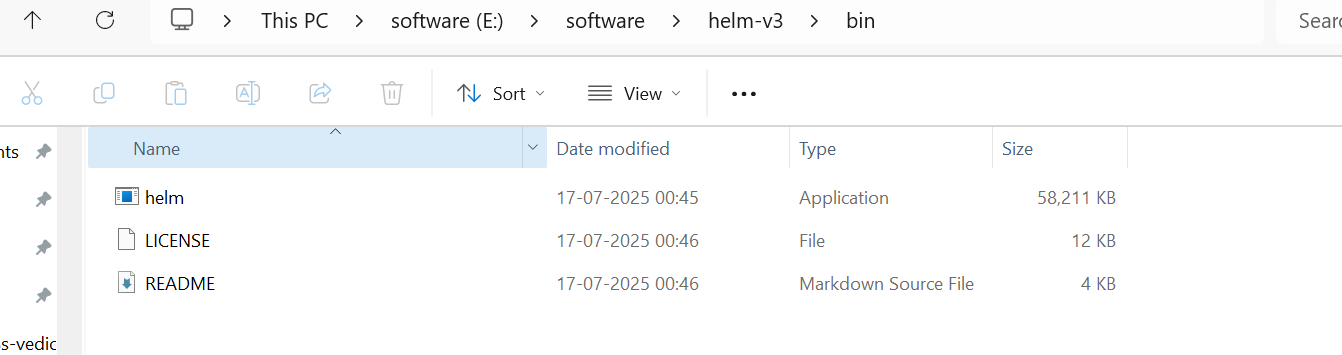
* Download binary of helm
* <https://github.com/helm/helm/releases>
* https://helm.sh/docs/intro/install/



#Once install extract it and

* create on folder like

helm-version\bin and put here helm



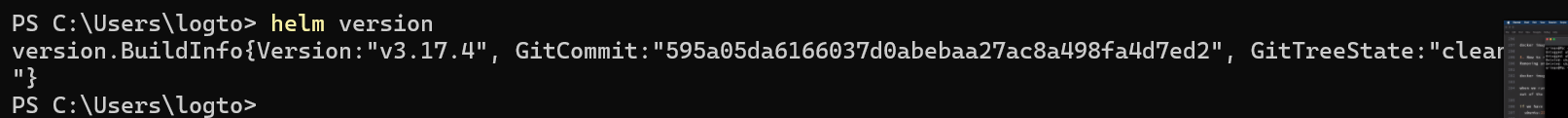
#Now set the path to wind path

HELM\_HOME=<LOCATION OF HELM BINARY>

PATH=$PATH:/HELM\_HOME/\bin

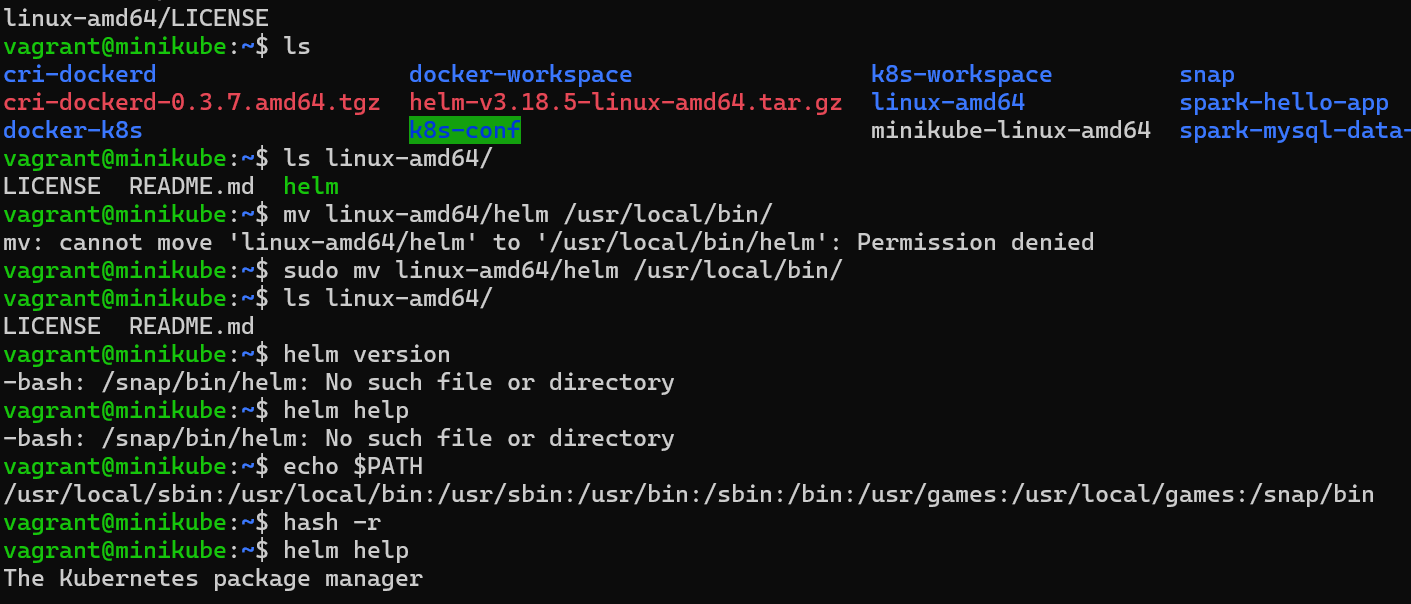
#now check the

Helm version



#Installation Ubuntu:

* download binary using above link
* wget <https://get.helm.sh/helm-v3.18.4-linux-amd64.tar.gz>
* extract tar –xzvf l helm-xxxxx
* mv helm /usr/local/bin



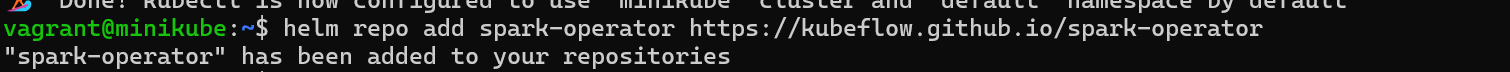
**✅ Most Useful Helm Commands (Cheat Sheet)**

| **Command** | **Description** |
| --- | --- |
| helm repo add <name> <url> | Add a new Helm chart repository |
| helm repo update | Update your local repo cache |
| helm search repo <keyword> | Search charts in added repositories |
| helm install <release> <chart> | Install a chart into Kubernetes |
| helm upgrade <release> <chart> | Upgrade a release to a new chart version |
| helm uninstall <release> | Uninstall/delete a release |
| helm list | List all Helm releases |
| helm list -A | List all releases in **all** namespaces |
| helm get all <release> | Show everything about a deployed release |
| helm status <release> | Show status of a Helm release |
| helm rollback <release> <rev> | Roll back to a previous release revision |
| helm template <chart> | Render chart to YAML (no install) |
| helm lint <chart> | Check chart for syntax and structure issues |
| helm show values <chart> | Show default values.yaml for a chart |
| helm pull <chart> --untar | Download and unpack a chart locally |
| helm create <chart-name> | Scaffold a new Helm chart project |

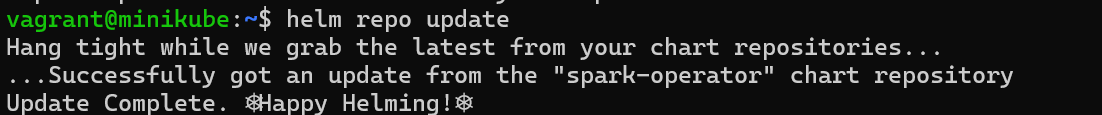
##Install spark-operator using helm

#first the spark repos

helm repo add spark-operator <https://kubeflow.github.io/spark-operator>



#Update the repo

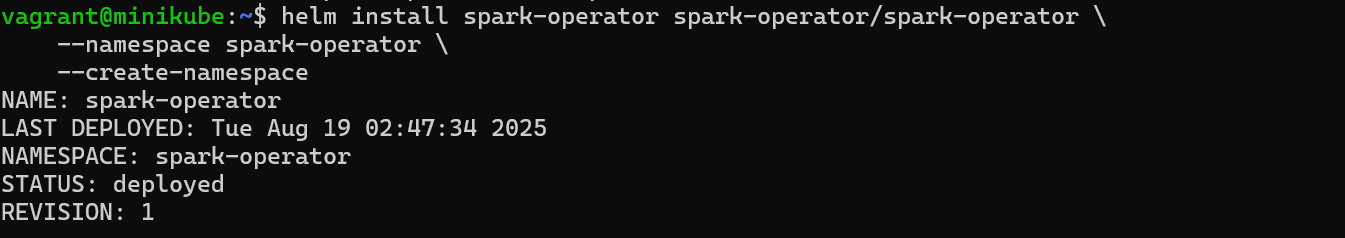
* helm repo update
* 

#Create spark operator name-space

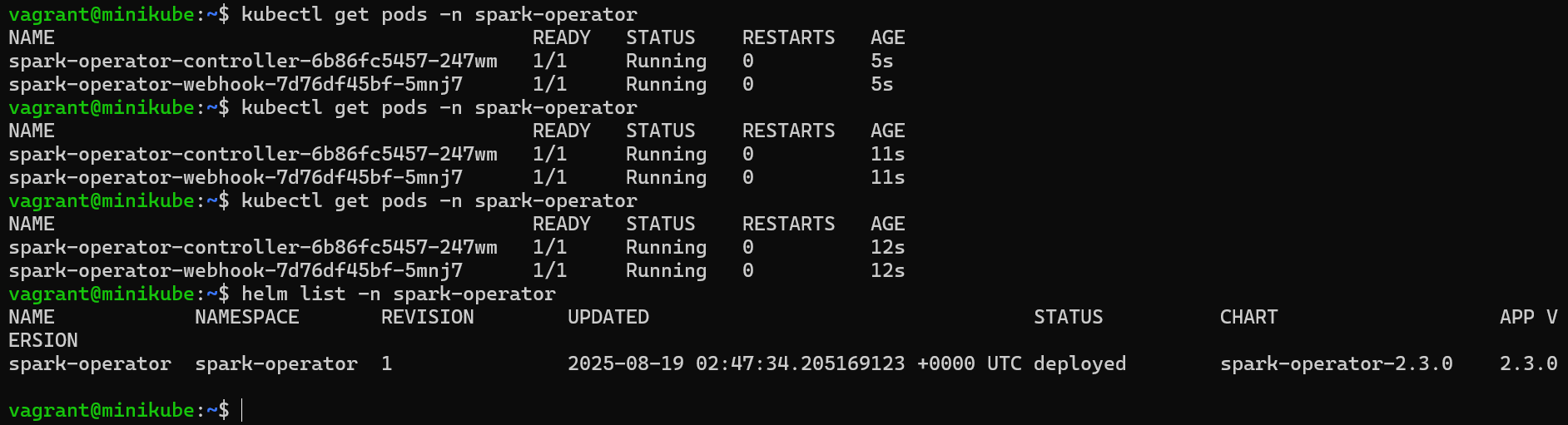
* helm install spark-operator spark-operator/spark-operator \

--namespace spark-operator \

--create-namespace



#now check the version and operator



wnode1.vm.box = "ubuntu/jammy64"

#start the minikube servere

sudo minikube start --driver=none --apiserver-ips=192.168.1.112

minikube start --driver=docker --apiserver-ips=192.168.1.112

#Stop the server

sudo minikube stop

minikube delete

##check status

minikube status

###create the pods:

##TO deployment on minikube

docker build -t spark-mysql-dataload:2.00 .

Apply the RBAC and Service Account:

# create CRD (custom resource defination) / its one time installation and it provide kind as sparkapplication

helm install spark-operator spark-operator/spark-operator \

--namespace spark-operator \

--create-namespace \

--set sparkJobNamespace=default \

--set webhook.enable=true

#Verify CRDS

kubectl get crd sparkapplications.sparkoperator.k8s.io

OR

kubectl get crd

#create the RBA or service-account

kubectl apply -f service-account.yml

#Verify

kubectl get sa

# deplpy the spark-app

kubectl apply -f spark-app.yml

#Monitor the application:

kubectl get sparkapplications

kubectl logs spark-mysql-dataload-driver

#for delete the pod

#call

kubectl get pod

#once get then delete that pods

kubectl delete pod <pod\_name>

# to delete the driver

#kubectl get sparkapplication

Once get the name then delete the driver pod

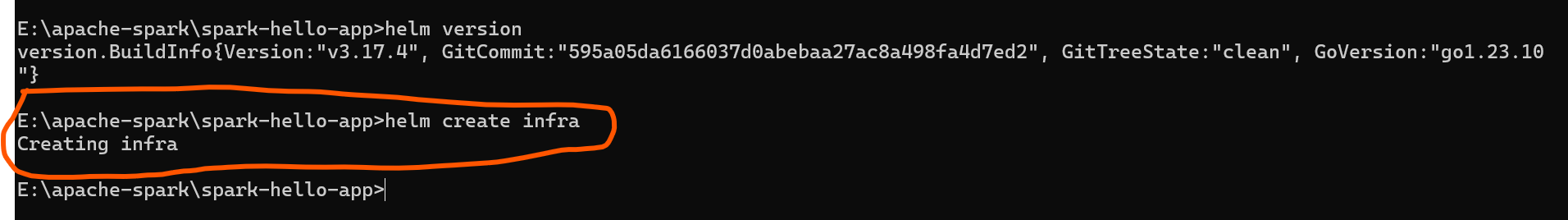
kubectl delete sparkapplications <pods\_anme>

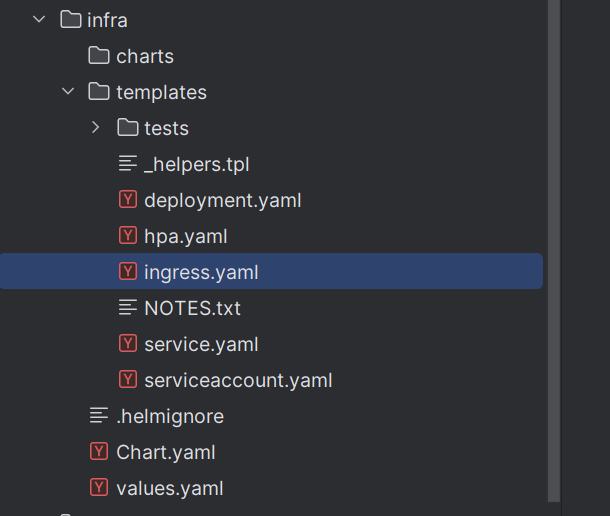
#for describe

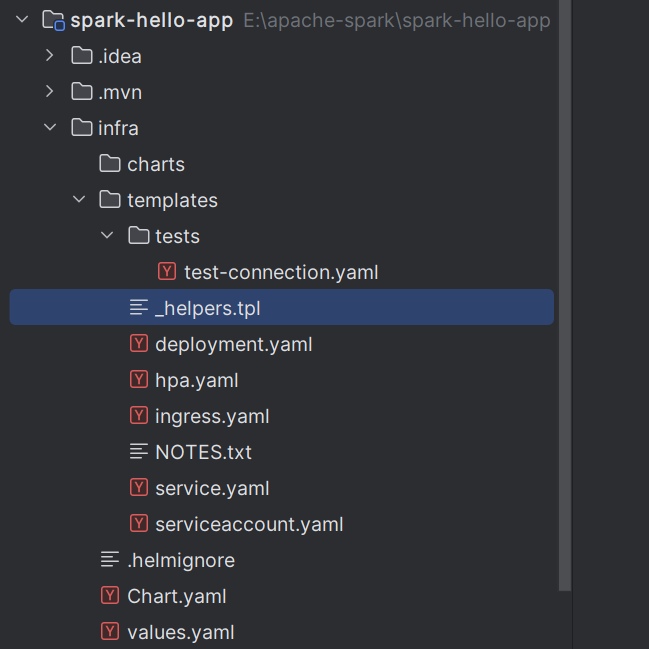
kubectl describe sparkapplication spark-hello-app -n default

##to create starndard helm folder or structure use below command

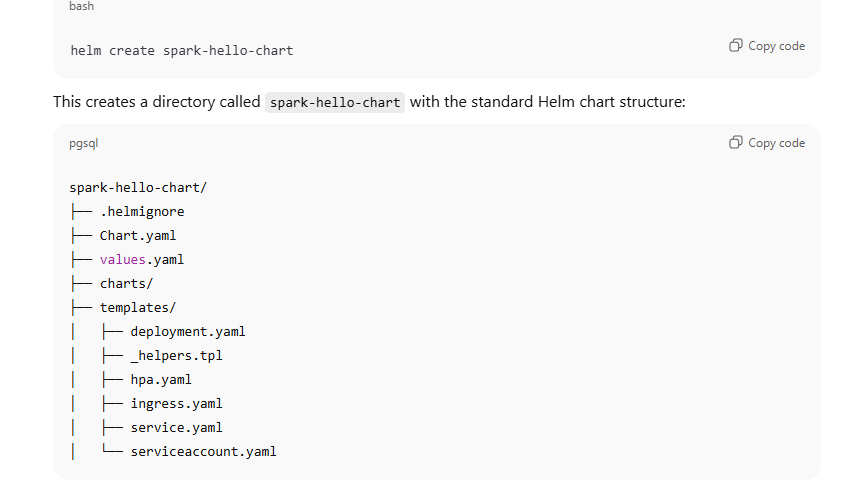
* helm create infra

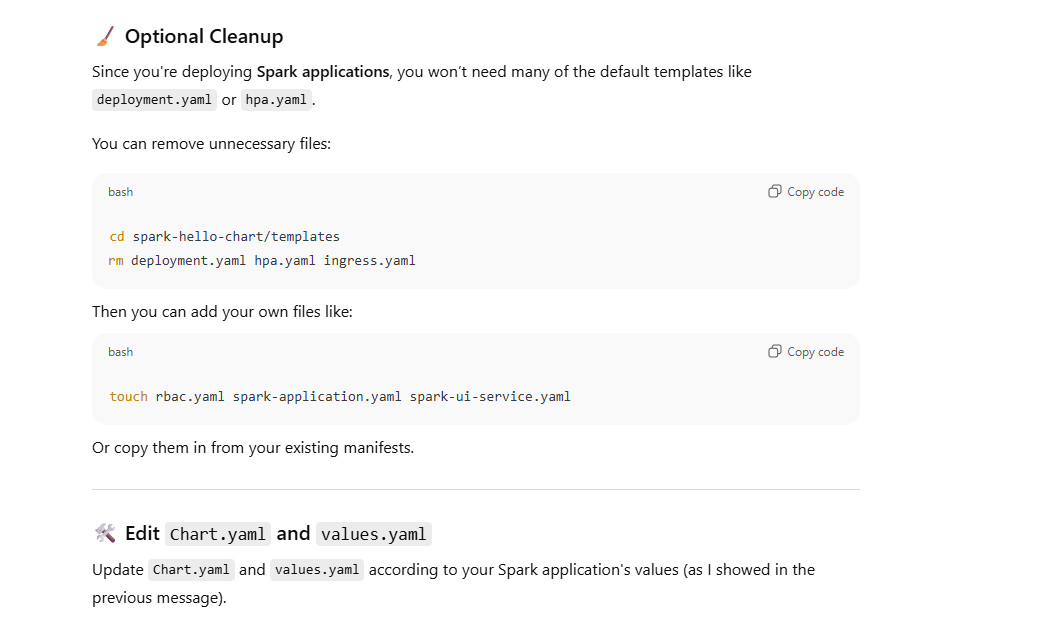
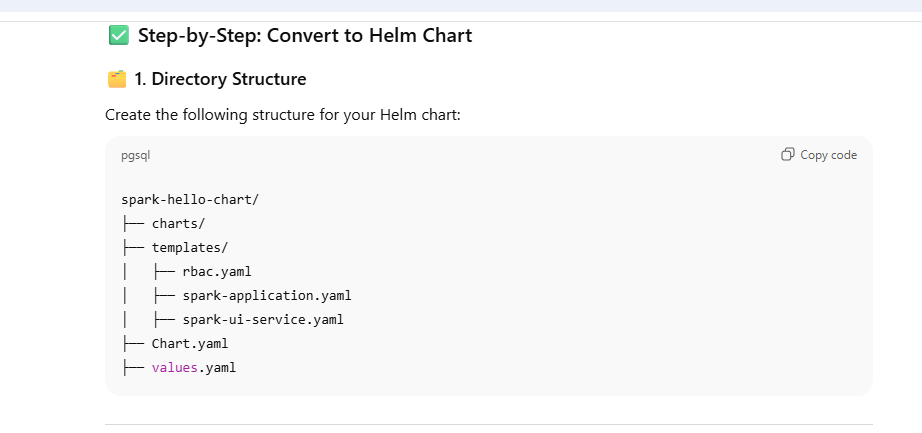


* structure
* 
* Expendended structure

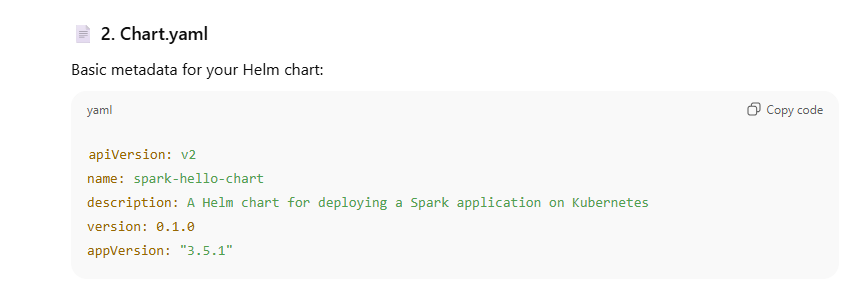


>Standard structure



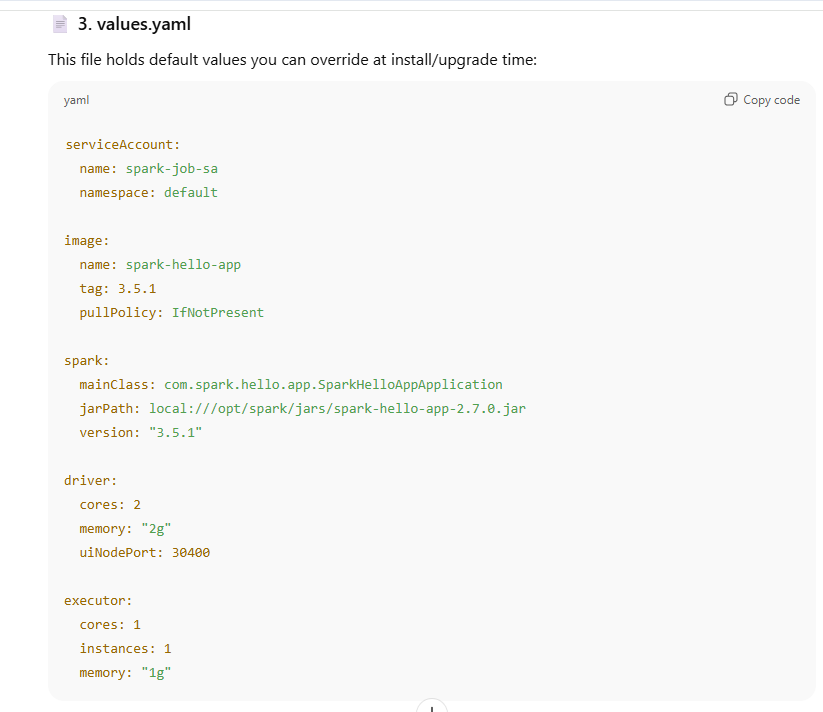
* Spark related file
* For Spark deployment we don’t need all file
* Required file are below
* 

#Chart contains only metadata information



#Value.yaml

* Its contains default values we can override as per reuirement

#

# now are file under template folder

* 📄 4. templates/rbac.yaml

apiVersion: v1

kind: ServiceAccount

metadata:

name: {{ .Values.serviceAccount.name }}

namespace: {{ .Values.serviceAccount.namespace }}

---

apiVersion: rbac.authorization.k8s.io/v1

kind: Role

metadata:

name: spark-role

namespace: {{ .Values.serviceAccount.namespace }}

rules:

- apiGroups: [""]

resources: ["pods", "pods/exec", "pods/log", "services", "configmaps", "events"]

verbs: ["get", "list", "watch", "create", "delete", "patch", "update"]

- apiGroups: ["batch"]

resources: ["jobs"]

verbs: ["get", "list", "watch", "create", "delete"]

---

apiVersion: rbac.authorization.k8s.io/v1

kind: RoleBinding

metadata:

name: spark-role-binding

namespace: {{ .Values.serviceAccount.namespace }}

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: Role

name: spark-role

subjects:

- kind: ServiceAccount

name: {{ .Values.serviceAccount.name }}

namespace: {{ .Values.serviceAccount.namespace }}

---

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRoleBinding

metadata:

name: spark-cluster-role-binding

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole

name: cluster-admin

subjects:

- kind: ServiceAccount

name: {{ .Values.serviceAccount.name }}

namespace: {{ .Values.serviceAccount.namespace }}

📄 5. templates/spark-application.yaml

apiVersion: sparkoperator.k8s.io/v1beta2

kind: SparkApplication

metadata:

name: spark-hello-app

namespace: {{ .Values.serviceAccount.namespace }}

spec:

type: Java

mode: cluster

image: {{ .Values.image.name }}:{{ .Values.image.tag }}

imagePullPolicy: {{ .Values.image.pullPolicy }}

mainClass: {{ .Values.spark.mainClass }}

mainApplicationFile: '{{ .Values.spark.jarPath }}'

sparkVersion: {{ .Values.spark.version }}

restartPolicy:

type: Never

sparkConf:

spark.kubernetes.container.image.pullPolicy: {{ .Values.image.pullPolicy }}

spark.kubernetes.executor.podNamePrefix: spark-hello-app-exec

spark.kubernetes.authenticate.driver.serviceAccountName: {{ .Values.serviceAccount.name }}

spark.kubernetes.api.timeout: "60000"

spark.kubernetes.executor.uploadLocalJars: "false"

spark.kubernetes.driver.uploadLocalJars: "false"

spark.hadoop.fs.file.impl: org.apache.hadoop.fs.LocalFileSystem

driver:

cores: {{ .Values.driver.cores }}

memory: "{{ .Values.driver.memory }}"

labels:

version: 1.0.0

serviceAccount: {{ .Values.serviceAccount.name }}

annotations:

sparkoperator.k8s.io/spark-ui-port: "4040"

sparkoperator.k8s.io/spark-ui-service-type: "NodePort"

sparkoperator.k8s.io/spark-ui-node-port: "{{ .Values.driver.uiNodePort }}"

executor:

cores: {{ .Values.executor.cores }}

instances: {{ .Values.executor.instances }}

memory: "{{ .Values.executor.memory }}"

labels:

version: 1.0.0

#📄 6. templates/spark-ui-service.yaml

apiVersion: v1

kind: Service

metadata:

name: spark-ui-service

namespace: {{ .Values.serviceAccount.namespace }}

spec:

type: NodePort

selector:

spark-role: driver

spark-app-selector: spark-hello-app

ports:

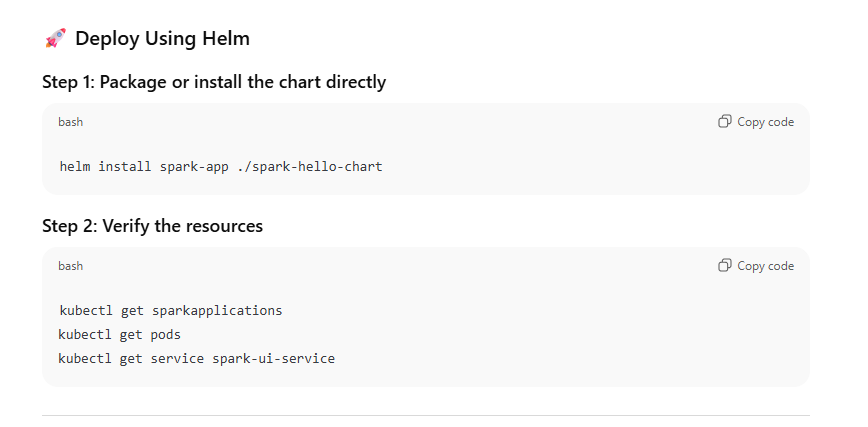
- name: webui

protocol: TCP

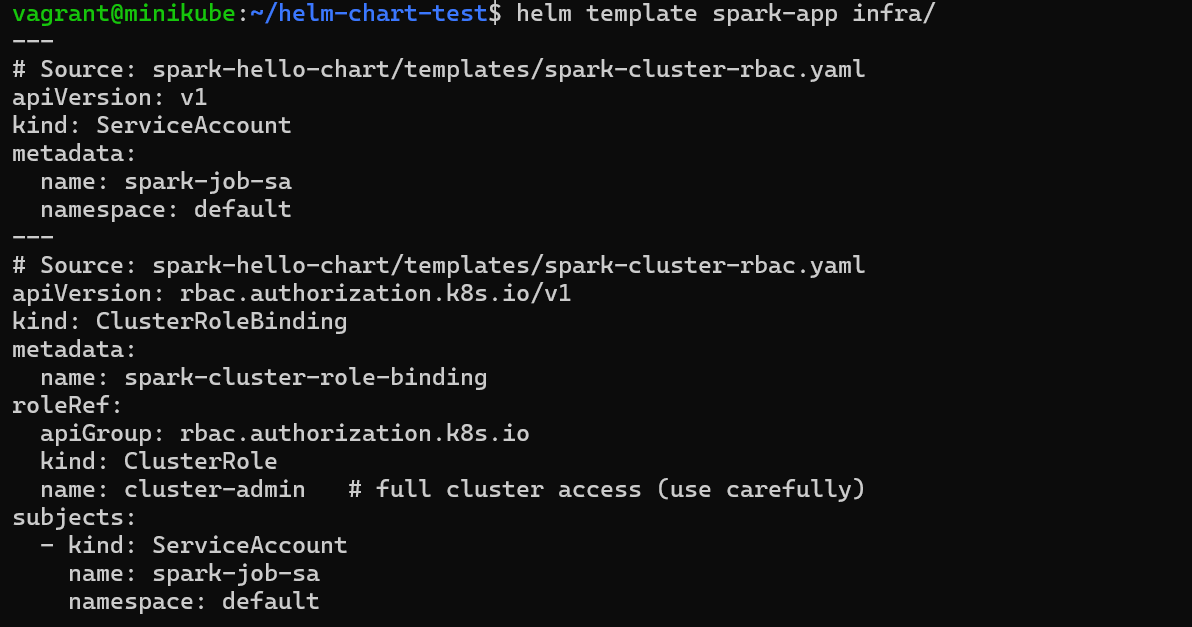
port: 4040

targetPort: 4040

nodePort: {{ .Values.driver.uiNodePort }}



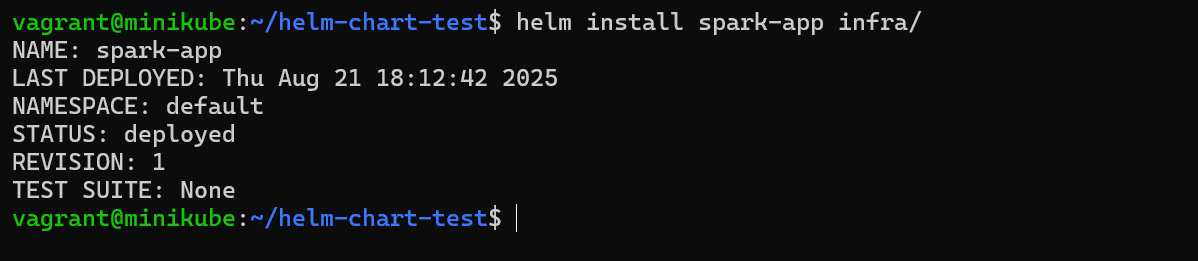
##to test first verify the value of chart

* helm template spark-app ./spark-hello-chart
* 

# create the chat use below command

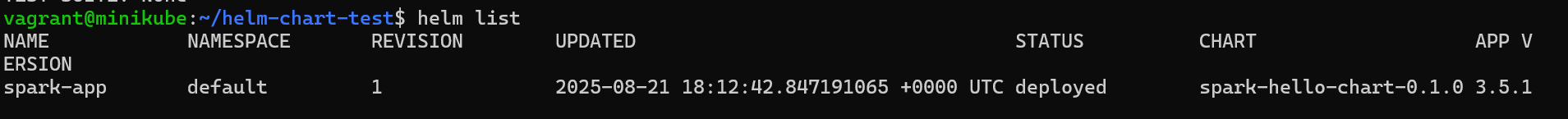
* helm install spark-app ./spark-hello-chart

here <spark-name> cloud be any think



#To List and UnInstall

helm list # See running releases



helm upgrade spark-app ... # Upgrade it

helm uninstall spark-app # Remove it 

##final Notes

# check helm install or not

* helm version

#To install spark-operator and CRD (custome resource definition) use below command

* helm repo add spark-operator <https://kubeflow.github.io/spark-operator>

# Update the repose

* helm repo update

#install spark-operator or CRD

* helm install spark-operator spark-operator/spark-operator \

--namespace spark-operator \

--create-namespace

OR

helm install spark-operator spark-operator/spark-operator \

--namespace spark-operator \

--create-namespace \

--set sparkJobNamespace=default \

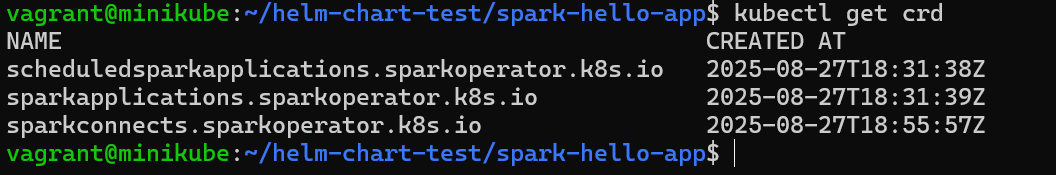
--set webhook.enable=true

# Verify the CRD

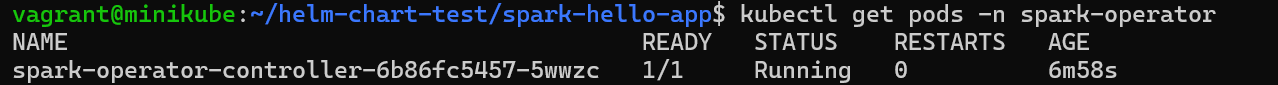
* kubectl get crd sparkapplications.sparkoperator.k8s.io

OR

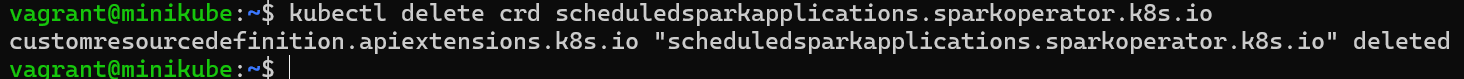
kubectl get crd



#to check spark operator running or not

* kubectl get pods -n spark-operator

#to delete crd

* kubectl delete crd <name\_crd> eg. Kubectl delete crd sparkApplication

# To create helm folder structure

* helm create <folderName> like helm create helm-chart

Note here folder and lots of file will generate so kept only req file ,delete unreq

#Once file the \*.yaml file using value.yaml file then verify that value properly file or not

* helm template <name> /folder like
* helm template spark-hello-app infra

#check list of helm release

* helm list or helm list –A

# To create/ deploy chart using helm

* helm install <name> folder where all file is there or helm install infra/

# To check release

* helm list or helm list –A

# To delete or uninstall release

* helm uninstall <release\_name> -n <namespace>like helm uninstall spark-hello –n default

# Verify release install or not

* helm list –A

##Not bydefault minikube or kubernate wont load local image need to run these command

By default, Minikube runs Kubernetes in a **VM or container** with its **own Docker environment**. So docker build on your local machine doesn’t make images available **inside Minikube** unless you do one of:

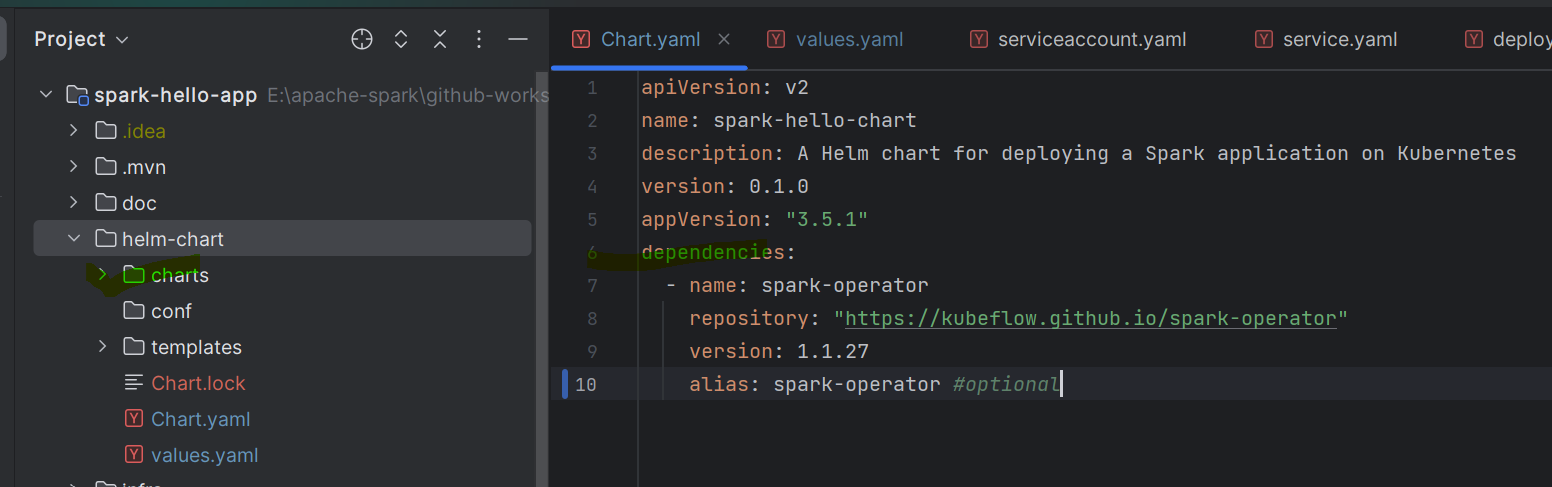
* Use minikube image load
* Use a shared Docker daemon via:

eval $(minikube docker-env)

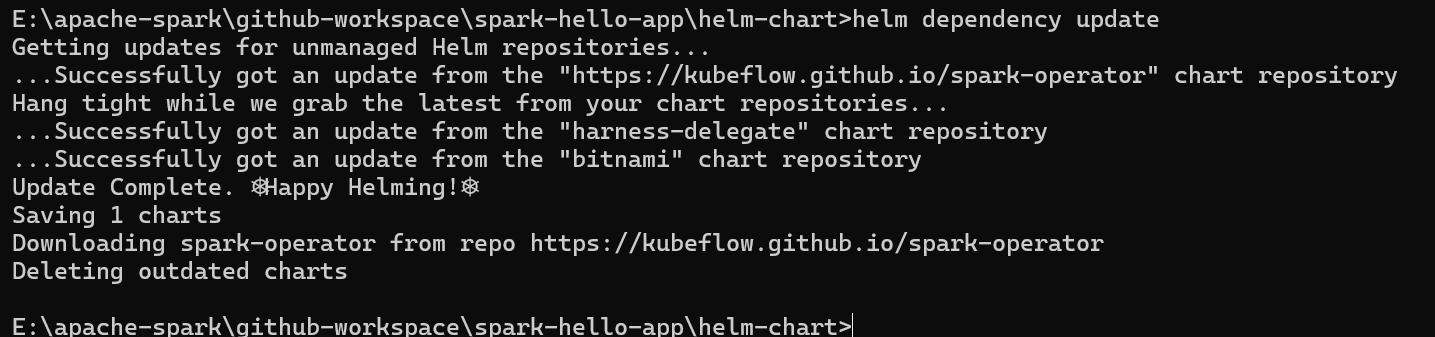
##Added spark-operator as dependency

#create folder under helm-chart

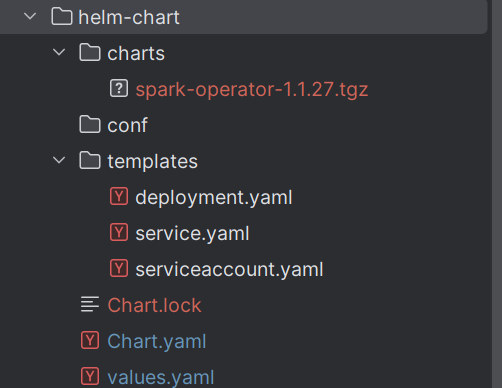
* Helm-chart/charts

#Add dependency tag on Charts.yaml  


# goto helm-chart dir and run command

* helm dependency update 

#final folder look like



#finally commit all these file on git

Note if getting error like role already exist , Added role name tag on value.yaml file And use it all place