



Helm Chart

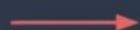
Chapter - 1

1. **What is Helm Chart?**
2. **Why we need Kubernetes?**
3. **How Helm Chart will help us to manage k8s?**
4. **How to install Helm Chart?**

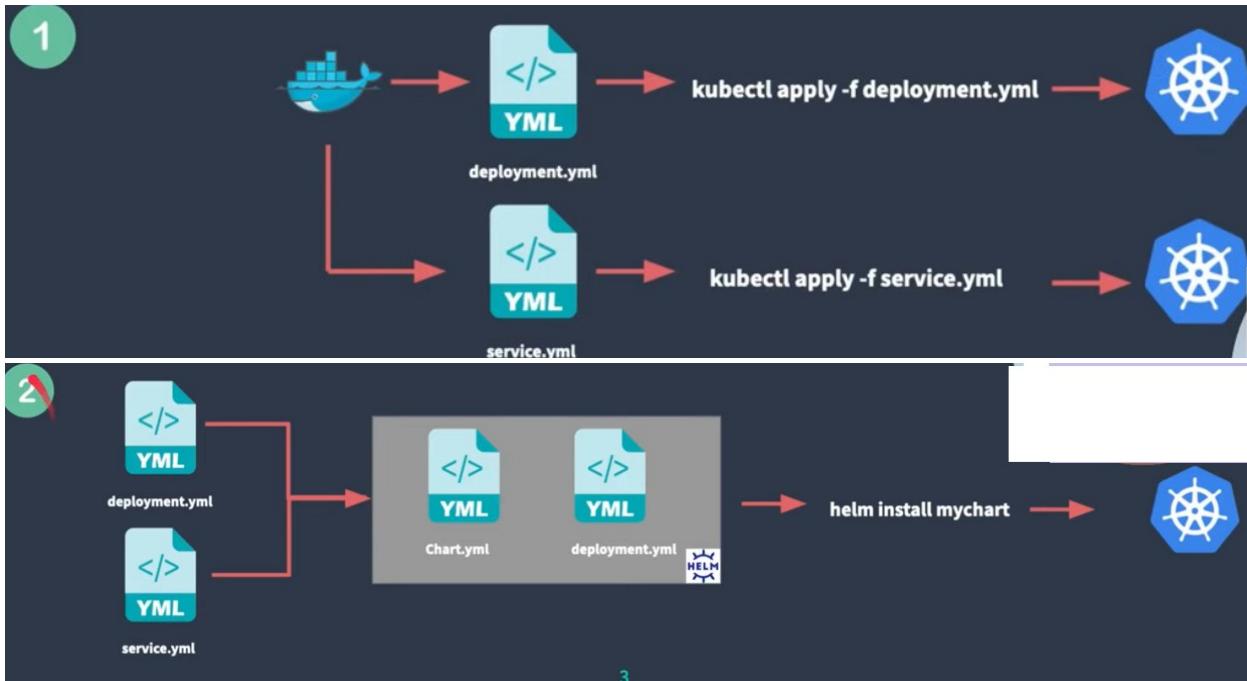
1. **Chapter 1** - Introduction to Helm Chart & Kubernetes
2. **What is Helm Chart?**
3. **Why is Kubernetes essential?**
4. **The role of Helm Chart in managing K8s Step-by-step guide to installing Helm Chart**



What is Helm Chart?



.



A Helm chart is used to simplify the deployment and management of applications on a Kubernetes cluster by packaging all the necessary resources (like deployments, services, and configurations) into a single, reusable unit, allowing for easier installation, upgrades, and sharing across different environments, significantly reducing manual configuration and potential errors in complex Kubernetes deployments. 🔗

Key benefits of using Helm charts:

Simplified deployment:

Instead of writing separate YAML files for each application component, a Helm chart bundles everything together, enabling deployment with a single command. 🔗

Versioning and reusability:

Charts can be versioned and shared across teams, allowing consistent deployments across different environments. 🔗

Configuration management:

Use "values files" to customize configurations for different environments without modifying the chart itself. 🔗

Easy upgrades and rollbacks:

Helm simplifies upgrading applications by managing changes between chart versions, and allows for easy rollbacks if needed. [🔗](#)

Reduced complexity:

Helm abstracts away the intricacies of Kubernetes YAML, making application management more accessible to developers and operations teams. [🔗](#)



Why we need K8S(Kubernetes)?



Example - A python app in a Docker container

How would you access the python application?

1. Build Docker image

```
docker build -t python-project .
```

2. Run Docker image

```
docker run -p 9001:9001 python-project
```

How Kubernetes will help you with Docker container?

1. **Scalability** - You can not scale single docker container to serve millions of request
2. **Automate Docker Deployment** - k8s can help you to manage automate the docker deployment
3. **Auto-Healing** - If a container is not healthy then k8s can auto replace the container
4. **Rollout & Rollback** - k8s monitors the unhealthy docker container and restart unresponsive.



How Helm Chart with help us to manage k8s?

Kubectl commands

```
kubectl apply -f deployment.yaml  
kubectl apply -f service.yaml  
kubectl apply -f ingress.yaml
```

```
kubectl delete -f deployment.yaml  
kubectl delete -f service.yaml  
kubectl delete -f ingress.yaml
```

helm commands

```
helm install my-app ./my-chart
```

```
helm uninstall my-app
```



How to install helm chart?

Pre-requisites

One running kubernetes cluster (any of the following)

- a. **EKS** - Amazon Elastic k8s service
- b. **GKE** - Google k8s engine
- c. **AKS** - Azure Kubernetes Service
- d. **microk8s** - k8s for development

(Disclaimer - We are gonna setup microk8s for the demo)

Installation of the Helm Chart

```
▶ curl -L https://git.io/get_helm.sh | bash -s -- --version v3.8.2  
▶ chmod 700 get_helm.sh  
▶ ./get_helm.sh
```

6

● Links - Microk8s - <https://microk8s.io/>

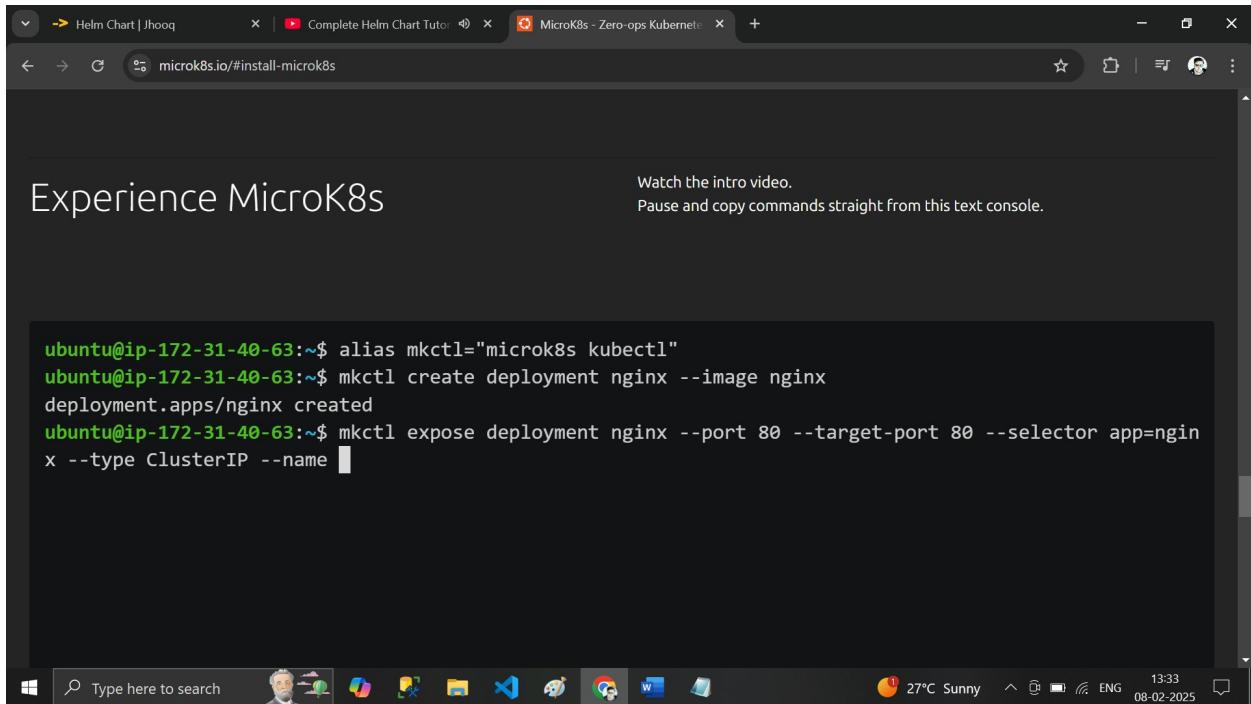
Install kubectl - <https://kubernetes.io/docs/tasks/tools...>

Install Microk8s -with Kubectl - <https://microk8s.io/docs/working-with-kubectl>

The screenshot shows a Microsoft Edge browser window with a dark theme. The address bar displays `microk8s.io/#install-microk8s`. The main content is a step-by-step guide for installing MicroK8s:

- 1. Install MicroK8s on Linux**
Command: `sudo snap install microk8s --classic`
Note: Don't have the `sudo` command? [Get set up for snaps](#)
- 2. Check the status while Kubernetes starts**
Command: `microk8s status --wait-ready`
- 3. Turn on the services you want**
Commands:
 - `microk8s enable dashboard`
 - `microk8s enable dns`
 - `microk8s enable registry`
 - `microk8s enable istio`
- 4. Start using Kubernetes**
Command: `microk8s kubectl get all --all-namespaces`
Text: If you mainly use MicroK8s you can make our `kubectl` the default one on your command-line with `alias mkctl="microk8s kubectl"`. Since it is a standard upstream `kubectl`, you can also drive other Kubernetes clusters with it by pointing to the respective `kubeconfig` file via the `--kubeconfig` argument.
- 5. Access the Kubernetes dashboard**
Command: `microk8s dashboard-proxy`
- 6. Start and stop Kubernetes to save battery**
Text: Kubernetes is a collection of system services that talk to each other all the time. If you don't need them running in the background then you will save battery by stopping them. `microk8s start` and `microk8s stop` will do the work for you.
[Read the docs to learn more](#)

At the bottom, there are links for "Join the community" and "Connect with our community and see what others are doing with MicroK8s". The taskbar at the bottom of the screen shows various pinned icons and the system tray with the date and time (08-02-2025) and battery level.



```
ubuntu@primary:~$ sudo snap install microk8s --classic
microk8s (1.27/stable) v1.27.2 from Canonical✓ installed
ubuntu@primary:~$
```

After this, reload the user groups either via a reboot or by running 'newgrp microk8s'.

```
ubuntu@primary:~$ sudo usermod -a -G microk8s ubuntu
ubuntu@primary:~$ newgrp microk8s
ubuntu@primary:~$
```

<https://kubernetes.io/docs/tasks/tools/install-kubectl-linux/>

The screenshot shows a Microsoft Edge browser window with the following details:

- Address Bar:** kubernetes.io/docs/tasks/tools/install-kubectl-linux/
- Page Title:** Install and Set Up kubectl on Linux
- Page Content:**
 - Section:** Install kubectl binary with curl on Linux
 - Text:** 1. Download the latest release with the command:
curl -LO "https://dl.k8s.io/release/\$(curl -L -s https://dl.k8s.io/release/stable.txt)"
 - Note:** To download a specific version, replace the \$(curl -L -s https://dl.k8s.io/release/stable.txt) portion of the command with the specific version. For example, to download version 1.32.0 on Linux x86-64, type:
curl -LO https://dl.k8s.io/release/v1.32.0/bin/linux/amd64/kubectl
 - And for Linux ARM64, type:
curl -LO https://dl.k8s.io/release/v1.32.0/bin/linux/arm64/kubectl
- Sidebar:** Includes links for Documentation, Getting started, Concepts, Tasks, and Install Tools (with sub-links for Linux, macOS, Windows, and more).
- System Tray:** Shows the date (08-02-2025), time (13:38), battery level (27°C Sunny), and network status.

```
ubuntu@primary:~$ curl -LO https://dl.k8s.io/release/v1.23.6/bin/linux/arm64/kubectl
% Total    % Received % Xferd  Average Speed   Time   Time     Current
          Dload  Upload Total Spent   Left Speed
100  138  100  138    0     0  536      0 --:--:-- --:--:-- 536
100 44.0M  100 44.0M   0     0 9765k      0 0:00:04 0:00:04 --:--:-- 11.5M
ubuntu@primary:~$ ls
Home  kubectl  snap
ubuntu@primary:~$ sudo install -o root -g root -m 0755 kubectl /usr/local/bin/kubectl
ubuntu@primary:~$
ubuntu@primary:~$ kubectl version
Client Version: version.Info{Major:"1", Minor:"23", GitVersion:"v1.23.6", GitCommit:"ad3338546da947756e8a88c", GitTreeState:"clean", BuildDate:"2022-04-14T08:49:13Z", GoVersion:"go1.17.9", Compiler:"gc", Platform:"linux/arm64"}
The connection to the server localhost:8080 was refused - did you specify the right host or port?
ubuntu@primary:~$
```

Install Microk8s -with Kubectl - <https://microk8s.io/docs/working-with-kubectl>

shakehand with kubectl + microk8s

If you have not already configured kubectl on the host, you can just open a terminal and generate the required config :

```
cd $HOME  
mkdir .kube  
cd .kube  
microk8s config > config
```

```
ubuntu@primary:~$ cd $HOME  
ubuntu@primary:~$ mkdir .kube  
ubuntu@primary:~$  
ubuntu@primary:~$ ls -lart  
total 45164  
-rw-r--r-- 1 ubuntu ubuntu      807 Jan  6 2022 .profile  
-rw-r--r-- 1 ubuntu ubuntu    3771 Jan  6 2022 .bashrc  
-rw-r--r-- 1 ubuntu ubuntu     220 Jan  6 2022 .bash_logout  
drwxr-x--- 1 ubuntu ubuntu    1824 Jul  8 18:17 Home  
drwxr-xr-x  3 root   root     4096 Jul  8 18:23 ..  
drwx----- 2 ubuntu ubuntu    4096 Jul  8 18:23 .ssh  
drwx----- 2 ubuntu ubuntu    4096 Jul  8 18:23 .cache  
-rw-r--r-- 1 ubuntu ubuntu      0 Jul  8 18:23 .sudo_as_admin_successful  
-rw----- 1 ubuntu ubuntu      5 Jul  8 18:23 .bash_history  
drwx----- 4 ubuntu ubuntu    4096 Jul  8 21:34 snap  
-rw-rw-r-- 1 ubuntu microk8s 46202880 Jul  8 21:47 kubectl  
drwxrwxr-x  2 ubuntu microk8s    4096 Jul  8 21:51 .kube  
drwxr-x--- 7 ubuntu ubuntu    4096 Jul  8 21:51 .  
ubuntu@primary:~$
```

```
ubuntu@primary:~$ cd .kube  
ubuntu@primary:~/.kube$ microk8s config > config  
ubuntu@primary:~/.kube$  
ubuntu@primary:~/.kube$  
ubuntu@primary:~/.kube$ ls -lart  
total 12  
drwxr-x--- 7 ubuntu ubuntu    4096 Jul  8 21:51 ..  
drwxrwxr-x  2 ubuntu microk8s 4096 Jul  8 21:51 .  
-rw-rw-r-- 1 ubuntu microk8s 1874 Jul  8 21:51 config  
ubuntu@primary:~/.kube$ cat con
```

certificate

```

clusters:
- cluster:
  certificate-authority-data: LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUREekNDQWZlZ0F3SUJBZ0lVT2ozTVFtb3JyWGYSVUSTMJJZZuk5QjlhLzNzd0RRWUpLb1pJaHZjTkFRRUwKQ1FBd0Z6RVZNQk1HQTFVRUF3d01NVEF1TVRVeUxqRTRNeTR4TUI0WERUSXpNRGN3T0RFNE16TXLNVm9YRFRNegrpNRGN3TLRFNE16TXLNVm93RnpFVkJ1CTUDBMVFQXd3TU1UQXVNPF5TgFNE15NHhNSULCSWpBTkJna3fao21HCj13MEJBUVVGQUPFQ0FR0EFSNLCoQ2dLQ0FRRUExv0NZdvRNWeE9IZTE2Q3UvRUVMH1NWVQ4UjFvQnYr1UxM0IK02ZMVTASFE8reUvWn1fleFBUbXIzRmN3cU9tcw5YY1J0Qj195cDF0bXR4QXNhdfpMby9xajdGbs91UmVuL2NSVgpMK2V0WS:tKODNGYnj5bhJpVmh6M0lxc0xPaH1S21Rzb01BWlh4QnIzaVVUVV1zL015WnBEZHIVnNmhta1BUClVPck0yNjh2bGnpWDNrUW43eUsvU1R6aFezcUI0NndrdEhLMDzsTaFQZkh60WSNVWj:jmhIMGjvaWzneXFcvnMKOTBLWTQwZk142nc3TFhqRE4zZWZKL0VnYTl0NnVhaYxeWpCxgxbeZhsFZINWU3anV1M0htL38SRDfQaDNB0Qp4Rk0vazdzWkfFKnl2b0hBdWZSMtDWnkNCWg96bxN3RWFb1Lf02ZE2NePdczN4VnJnU1eQVF80m8xTxdVVEFkCkJnTlZIUTRFrmdrVUdCbVRhaFV0dzkyTEFLN3hUVDZ1UXVJVDVi3dId1EVlIwakJCZ3dGb0FVR0JtVGf0U4KdzkyTEFLN3hUVDZ1UXVJVDVi3dEd1lEVlIwVEFRSC9CQV3QxDfQj960USCZ2txaGtpRz13MEJBUXNGQUPFQwpBUUVBTUtnl1B1SzrjeFNSzgyVytOM1B3zMVNFMrTZOWGLuVtLy0EhHcytEOFp2MXBGYkNUc3ZjCew3a3NiCnR3WFN4Z2J5SUowM0V4V0xwekl5RHftOGxRdhZNkzFuNHeRME5QTJqRGd3SGRubz80TFY3dVh1Q0tvWjF2UWsKd3Y4ekg5YlczXhxa1hm0ElJanN2U3QwWEpKTE04MndESTdIMDRSdnpWcVQ5SVFsazhXVkrhYnB5MwwwOlw0Ap1RGYdnFYNkdJUNEPNvg3cFMyewpXL2tZK1Vm1RhRExGU01c3J5aTk4dFTK25tMgfZMxU0ZnZmcwXUUzNKCrRk1VaVxi4RFSTzRIZXp0eTYzZmVUWU0yS1Q0bXIwNGtkZ1NNat1SEI00Fp0VVBKTH1aMzZuV1Qw0Wlz0HcKTG9GWkR6RE11TFFUvNFFMzEzK2FMYW9UQn9PQotLS0tLUVORCBDRVJUSUZJQ0FURS0tLS0tCg==server: https://192.168.64.5:16443
  name: microk8s-cluster
contexts:
- context:
  cluster: microk8s-cluster
  user: admin
  name: microk8s
current-context: microk8s
kind: Config
preferences: {}
users:
  -
```

>>>>>>>>>>>>>>>>>>>RESTART Your machine<<<<<<<<<<<<<<<<

<https://microk8s.io/>

The screenshot shows a browser window with several tabs open. The active tab is titled "MicroK8s - Zero-ops Kubernetes". The page content is as follows:

Try microk8s enable --help for a list of available services and optional features. microk8s disable <name> turns off a service.

4. Start using Kubernetes `microk8s kubectl get all --all-namespaces`

If you mainly use MicroK8s you can make our kubectl the default one on your command-line with `alias mkctl="microk8s kubectl"`. Since it is a standard upstream kubectl, you can also drive other Kubernetes clusters with it by pointing to the respective kubeconfig file via the `--kubeconfig` argument.

5. Access the Kubernetes dashboard `microk8s dashboard-proxy`

Kubernetes is a collection of system services that talk to each other all the time. If you don't need them running in the background then you will save battery by stopping them. `microk8s start` and `microk8s stop` will do the work for you.

6. Start and stop Kubernetes to save battery

Read the docs to learn more ›

A "Help us Improve" button is located on the right side of the page.

At the bottom of the screen, the Windows taskbar is visible with various icons and a system tray showing the date and time (08-02-2025), battery level (29°C), and network status.

```

ubuntu@primary:~$ kubectl get all --all-namespaces
NAMESPACE     NAME                                         READY   STATUS    RESTARTS   AGE
kube-system   pod/coredns-7745f9f87f-5xp7k                1/1     Running   1 (4m32s ago)   25m
kube-system   pod/calico-kube-controllers-6c99c8747f-wvvzw  1/1     Running   1 (4m32s ago)   25m
kube-system   pod/calico-node-f2dq4                         1/1     Running   1 (4m32s ago)   25m

NAMESPACE   NAME           TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
default     service/kubernetes   ClusterIP   10.152.183.1  <none>          443/TCP         25m
kube-system service/kube-dns   ClusterIP   10.152.183.10 <none>          53/UDP,53/TCP,9153/TCP 25m

NAMESPACE   NAME           DESIRED  CURRENT  READY  UP-TO-DATE  AVAILABLE  NODE SELECTOR  AGE
kube-system daemonset.apps/calico-node  1        1        1       1        1        1        kubernetes.io/os=linux  25m

NAMESPACE   NAME           READY  UP-TO-DATE  AVAILABLE  AGE
kube-system deployment.apps/coredns      1/1    1          1          25m
kube-system deployment.apps/calico-kube-controllers  1/1    1          1          25m

NAMESPACE   NAME           DESIRED  CURRENT  READY  AGE
kube-system replicaset.apps/coredns-7745f9f87f  1        1        1        25m
kube-system replicaset.apps/calico-kube-controllers-6c99c8747f  1        1        1        25m

```

helm Chart cmd / installation from Jhooq – Rahul Wagh Sir 

<https://jhooq.com/getting-start-with-helm-chart/>

```

ubuntu@primary:~$ curl -L https://git.io/get_helm.sh | bash -s -- --version v3.8.2
% Total    % Received % Xferd  Average Speed   Time     Time     Time  Current
          Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0  --::-- --::-- --::-- 0
100  6666  100  6666    0     0  8108  0  --::-- --::-- --::-- 95228
Downloading https://get.helm.sh/helm-v3.8.2-linux-arm64.tar.gz
Preparing to install helm and tiller into /usr/local/bin
helm installed into /usr/local/bin/helm
info: tiller binary was not found in this release; skipping tiller installation
Run 'helm init' to configure helm.

```

```

ubuntu@primary:~$ ls -lart
total 45164
-rw-r--r-- 1 ubuntu ubuntu      807 Jan  6 2022 .profile
-rw-r--r-- 1 ubuntu ubuntu    3771 Jan  6 2022 .bashrc
-rw-r--r-- 1 ubuntu ubuntu     220 Jan  6 2022 .bash_logout
drwxr-x--- 1 ubuntu ubuntu   1824 Jul  8 18:17 Home
drwxr-xr-x  3 root  root    4096 Jul  8 18:23 ..
drwx----- 2 ubuntu ubuntu    4096 Jul  8 18:23 .ssh
drwx----- 2 ubuntu ubuntu    4096 Jul  8 18:23 .cache
-rw-r--r-- 1 ubuntu ubuntu      0 Jul  8 18:23 .sudo_as_admin_successful
drwx----- 4 ubuntu ubuntu    4096 Jul  8 21:34 snap
-rw-rw-r-- 1 ubuntu mikrok8s 46202880 Jul  8 21:47 kubectl
drwxr-x--- 7 ubuntu ubuntu    4096 Jul  8 21:51 .
drwxrwxr-x  3 ubuntu mikrok8s  4096 Jul  8 21:53 .kube
-rw----- 1 ubuntu ubuntu     639 Jul  8 21:54 .bash_history
ubuntu@primary:~$ 

```

```
ubuntu@primary:~$ helm version
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
version.BuildInfo{Version:"v3.8.2", GitCommit:"6e3701edea09e5d55a8ca2aae03a68917630e91b", GitTreeState:"clean", GoVersion:"go1.17.5"}
ubuntu@primary:~$
```



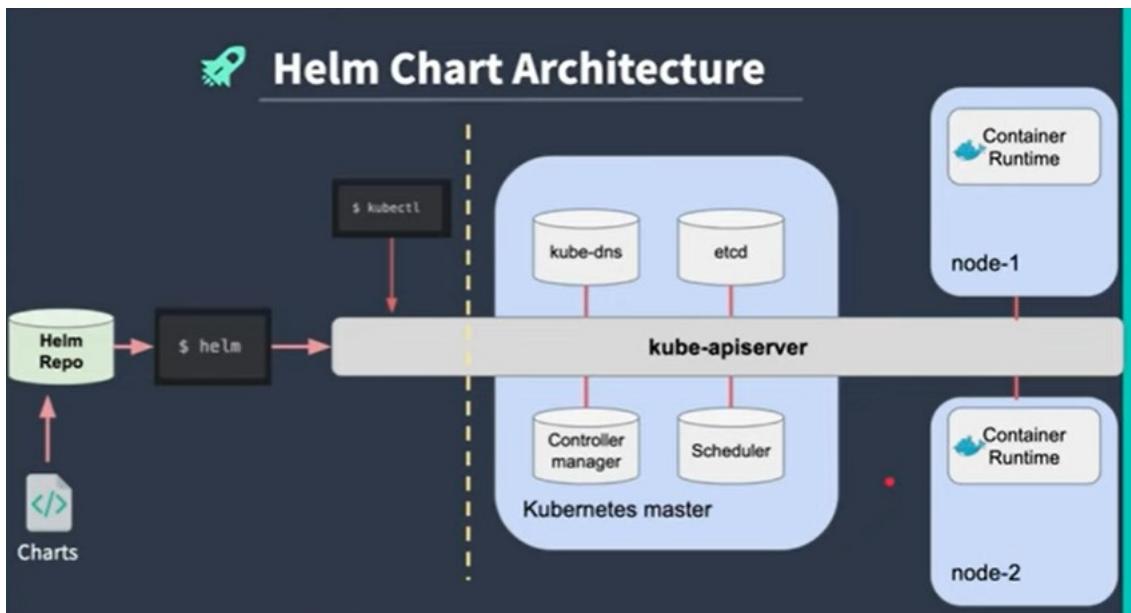
Chapter 2 - Diving Deeper into Helm Chart

Understand Helm Chart's significance ..The symbiotic relationship between Kubernetes and Helm Chart Streamlining .. K8s management using Helm Chart

complete Practical + notes + cmd execution mentioned below

<http://jhooq.com/getting-start-with-helm-chart/>

Helm chart Architecture





Helm CLI(Commands)

Basic Helm CLI Commands

```
▶ helm install myhelloworld helloworld
```

```
▶ helm delete myhelloworld
```

```
▶ helm list -a
```

2. Writing your first Helm Chart for "Hello World"

Now after you have done your Helm Chart installation, we can write our first "**Hello World**" Helm Chart.

To begin with -

2.1: Create your first Helm Chart

We are going to create our first **helloworld** Helm Chart using the following command

```
1 helm create helloworld
```

BASH

It should create a directory **helloworld**, you can verify it by using the following `ls -lart` command

```
1 ls -lart | grep helloworld
```

BASH

It should return you with -

```
1 drwxr-xr-x 4 vagrant vagrant 4096 Nov 7 19:57 helloworld
```

BASH

To verify the complete directory structure of the HelmChart please do run the command

```
1 tree helloworld
```

BASH

```
1 helloworld
2   |-- charts
3   |-- Chart.yaml
4   |-- templates
5   |   |-- deployment.yaml
6   |   |-- _helpers.tpl
7   |   |-- hpa.yaml
8   |   |-- ingress.yaml
9   |   |-- NOTES.txt
10  |   |-- serviceaccount.yaml
11  |   |-- service.yaml
12  |   |-- tests
13  |       └── test-connection.yaml
14  `-- values.yaml
```

Great now you created your first Helm Chart - **helloworld**.

In the next steps we are going to run the **helloworld** Helm Chart

2.2: Update the service.type from ClusterIP to NodePort inside the values.yml

Before you run your **helloworld** Helm Chart we need to update the `service.type` from `ClusterIP` to `NodePort`.

The reason for this change is - After installing/running the **helloworld** Helm Chart we should be able to access the service outside of the kubernetes cluster. And if you do not change the `service.type` then you will only be able to access the service within kubernetes cluster.

To update the `values.yml`, first go inside the directory `helloworld`

```
1 cd helloworld
```

BASH

2.2.1: Open `values.yml` in `vi`

After that open the `values.yml` in `vi`

```
1 vi values.yaml
```

2.2.2: Update `service.type` from `ClusterIP` to `NodePort`

Look for the `service.type` block and update its value to `NodePort`

```
1 service:  
2   type: NodePort  
3   port: 80
```

```
ubuntu@primary:~/helloworld$  
ubuntu@primary:~/helloworld$  
ubuntu@primary:~/helloworld$ pwd  
/home/ubuntu/helloworld  
ubuntu@primary:~/helloworld$ vi values.yaml █
```

change the service to NodePort ..

```
# runAsNonRoot: true
# runAsUser: 1000

service:
  type: NodePort
  port: 80

ingress:
  enabled: false
  className: ""
  annotations: {}
    # kubernetes.io/ingress.class: nginx
    # kubernetes.io/tls-acme: "true"
  hosts:
    - host: chart-example.local
      paths:
        - path: /
          pathType: ImplementationSpecific
  tls: []
#   - secretName: chart-example-tls
#     hosts:
#       - chart-example.local

resources: {}
# We usually recommend not to specify default resources and to leave this as a conscious
# choice for the user. This also increases chances charts run on environments with little
# resources, such as Minikube. If you do want to specify resources, uncomment the following
# lines, adjust them as necessary, and remove the curly braces after 'resources:'.
# limits:
#   cpu: 100m
#   memory: 128Mi
# requests:
```

2.3: Install the Helm Chart using command - `helm install`

Now after updating the `values.yaml`, you can install the Helm Chart.

Note : The `helm install` command take two arguments -

1. First argument - Release name that you pick
2. Second argument - Chart you want to install

It should look like -

```
1 helm install <FIRST_ARGUMENT_RELEASE_NAME> <SECOND_ARGUMENT_CHART_NAME>
```

```
1 helm install myhelloworld helloworld
```

After running the above command it should return you with –

```
1 NAME: myhellworld
2 LAST DEPLOYED: Sat Nov  7 21:48:08 2020
3 NAMESPACE: default
4 STATUS: deployed
5 REVISION: 1
6 NOTES:
7 1. Get the application URL by running these commands:
8   export NODE_PORT=$(kubectl get --namespace default -o jsonpath=".spec.ports[0].nodePort"} service myhelloworld)
9   export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath=".items[0].status.addresses[0].ip")
10  echo http://$NODE_IP:$NODE_PORT
```

2.4: Verify the helm install command

Now you need to verify your helm release i.e. `myhelloworld` and which can be done by running the `helm list` command.

```
1 helm list -a
```

It should return you with the release name which you have just installed i.e. `myhelloworld`

NAME	NAMESPACE	REVISION	UPDATED	STATUS
myhelloworld	default	1	2020-11-07 21:48:08.8550677 +0000 UTC	deployed

2.5: Get kubernetes Service details and port

Lets run the `kubectl get service` command to get the `NodePort`.

```
1 kubectl get service
```

And the above command should return you -

1	NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
2	kubernetes	ClusterIP	10.233.0.1	<none>	443/TCP	14d
3	myhellworld-helloworld	NodePort	10.233.14.134	<none>	80:30738/TCP	7m10s

Note: Keep in mind the `NodePort` number can vary in the range **30000-32767**, so you might get different `NodePort`.

Since my `cluster ip` is **100.0.0.2** and `NodePort` is **30738**, so I can access my Nginx page of my `myhellworld` Helm Chart



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

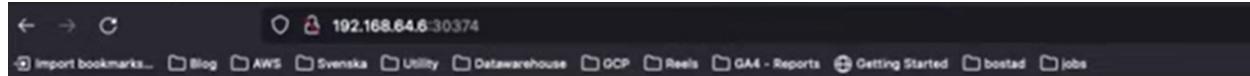
For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

Thank you for using nginx.

```
ubuntu@primary:~/helloworld$ pwd
/home/ubuntu/helloworld
ubuntu@primary:~/helloworld$ cd ..
ubuntu@primary:~$ 
ubuntu@primary:~$ pwd
/home/ubuntu
ubuntu@primary:~$ ls
Home helloworld kubectl snap
ubuntu@primary:~$
```

```
ubuntu@primary:~$ helm install myhelloworld helloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME: myhelloworld
LAST DEPLOYED: Sun Jul  9 16:33:35 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
  export NODE_PORT=$(kubectl get --namespace default -o jsonpath="{.spec.ports[0].nodePort}" services myhelloworld)
  export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath=".items[0].status.addresses[0].address")
  echo http://$NODE_IP:$NODE_PORT
ubuntu@primary:~$ 
ubuntu@primary:~$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME      NAMESPACE   REVISION   UPDATED             STATUS      CHART          APP V
VERSION
myhelloworld  default     1          2023-07-09 16:33:35.294021765 +0300 EEST    deployed   helloworld-0.1.0  1.16.
0
ubuntu@primary:~$
```

```
ubuntu@primary:~$ kubectl get service
NAME        TYPE        CLUSTER-IP      EXTERNAL-IP    PORT(S)        AGE
kubernetes  ClusterIP  10.152.183.1   <none>        443/TCP       97m
myhelloworld  NodePort  10.152.183.18  <none>        80:30374/TCP  4m7s
ubuntu@primary:~$ hostname -I
192.168.64.6 10.1.226.128 fd16:8590:7284:4e59:5054:ff:fe7f:b17d
ubuntu@primary:~$
```



Welcome to nginx!

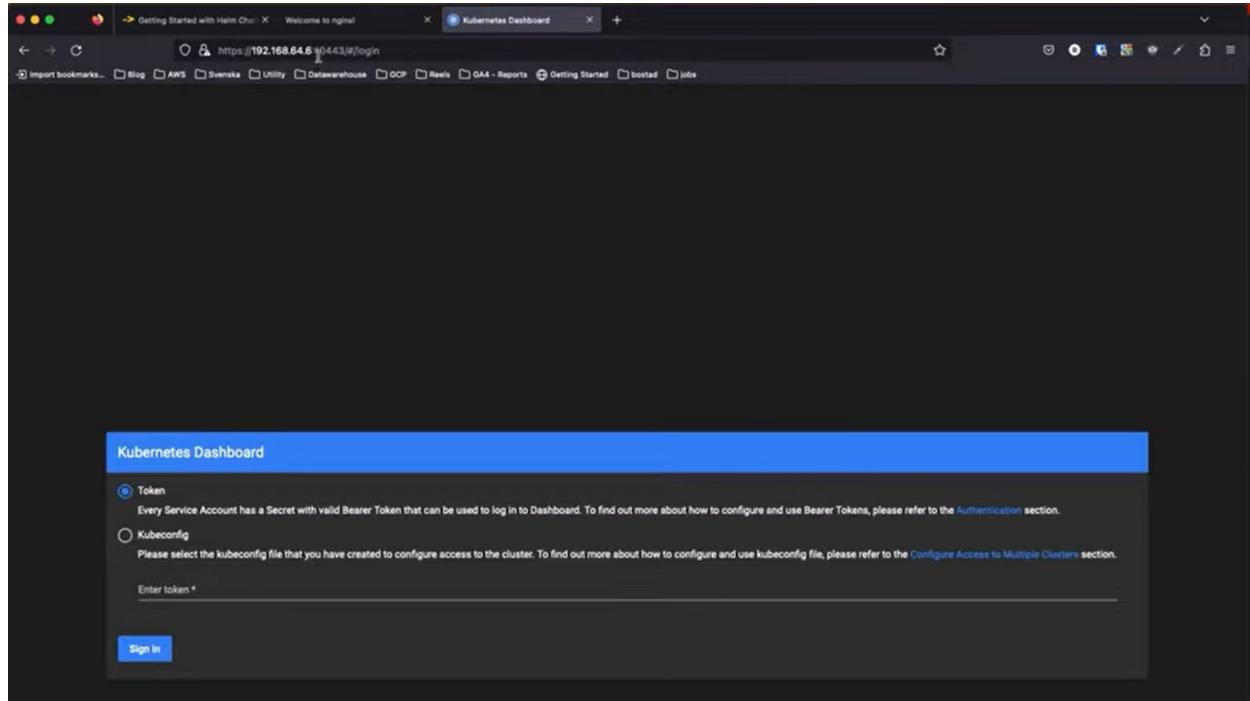
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

Thank you for using nginx.

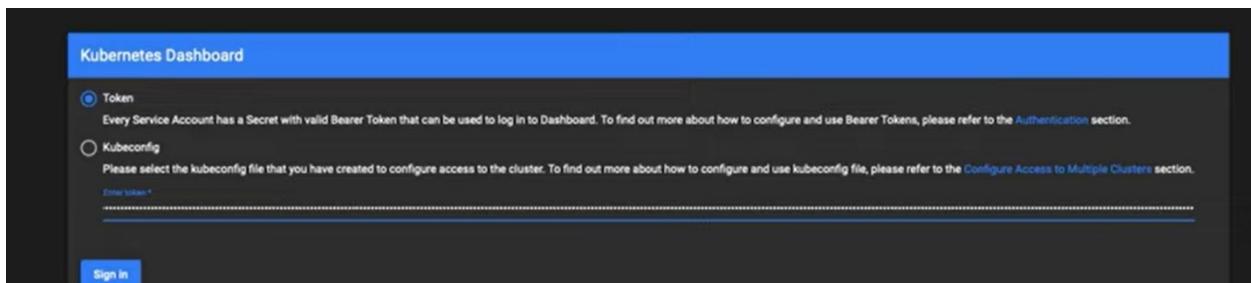
Kubernetes -dashboard

```
ubuntu@primary:~$ microk8s dashboard-proxy
Checking if Dashboard is running.
Infer repository core for addon dashboard
Waiting for Dashboard to come up.
Trying to get token from microk8s-dashboard-token
Waiting for secret token (attempt 0)
Dashboard will be available at https://127.0.0.1:10443
Use the following token to login:
eyJhbGciOiJSUzI1NiIsImtpZC16IjZqY3p1VWJYc1pyNE1MMkk3YkIyeWnSUpsdVNFMGVZQkppTjZheFJRN1UiFQ . eyJpc3MiOiJrdWJlc51dGVzL3NlcnZpY2VhY2NvdW50Iiwia3
VizXJuZXRLcy5pb9zZXJ2aWNLYWNjb3VudC9uYW1lcz8hY2Ui0iJrdWJllXN5c3RibsISImt1YmVybmv0ZXMuaW8vc2VydmljZWfjY291bnQvc2Vjcmv0Lm5hbWUi0iJtaWNyb2s4cy1
kYXNoYm9hcmQtgd9rZW4iLCJrdWJlc51dGVzLmlvL3NlcnZpY2VhY2NvdW50L3NlcnZpY2UtYWNjb3VudCSuYW1lijoIZGVmYXVsdcIsImt1YmVybmv0ZXMuaW8vc2VydmljZWfjY291
bnQvc2VydmljZS1hY2NvdW50lnVpZC16IjUyMGViNDfjLWJiZmUeNGRjNy04YTc1LTY4ZGjiYmE4NWm2MyIsInN1YiI6In5c3RlbTpzZXJ2aWNLYWNjb3VudDprdWJllXN5c3RlbTpkZ
WZhdx0In0.U0GSqYQvXvCjzYV-xw5HFsjYwKw4Mjn1IvaTH6oej_YmNkl18ovKLpzmYArh086jPxAcAgYxw6vvU5PQVKwrgavq9YE79TocYrFduWfbvPoy_RDn3WTk40mGjusftILS
S0pkkAWBv9YFajzHLMvx7eYymlsUHb1eei-zBhBkgtzlMkg9uVvH02LICMQ3E-t5T-yngaaB3Rv9jmFb-gwsazyk13wljkGCxqUlvdgr1vS5_t9gyAA1NYD6-Rsw31DAde00VpMzxHoP
WtZnbSqwRs78axp5mLZ20iBjWqPyn7lwq5uwH5t2a7wEJ00KAC1mVhQu0t2oAK9N1wK8w
```



cp the token + Sign in

```
ubuntu@primary:~$ microk8s dashboard-proxy
Checking if Dashboard is running.
Infer repository core for addon dashboard
Waiting for Dashboard to come up.
Trying to get token from microk8s-dashboard-token
Waiting for secret token (attempt 0)
Dashboard will be available at https://127.0.0.1:10443
Use the following token to login:
eyJhbGciOiJSUzI1NiSltmpTCI6IjZgY3IpWVJYCi1pyNE1MMkk3YkIyeWvnSupsdVNFMGVZQkppTjZheFJRJN1Uifq.eyJpc3Mi0iJrdWJlcml5dGVzL3NlcnZpY2VhY2NvdWS01iwi.a3
VzXJZkR1cy5pb9zXJ2oWNlyNjb3VudC9uYW1l3c8hYU0iJrdwJlLNx5c3RlbS1sImt1YmVybmv0ZXMuadW8vc2VydmljZWFyJY291bnQvc2Vjcmv0Lm5hbWUi0Jta0Nybz5c4yl
kYXNoYm9hcmTdg9rZW4iLCJrdwJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdWS0L3NlcnZpY2U0tYWNjb3VudCSuYW1lIjoizGVmYXVsdcIsImt1YmVybmv0ZXMuadW8vc2VydmljZWFyJY291
bnDn2VydmljZs1hY2NvdWS0LndVpZC16IjUyMGViNDfjlWLuijNtNGRjNy04YtclLTy4ZGjiYmE4NMzMyIsInN1YiI6InNSc3R1bTpzXJ2oWNlyNjb3VudDprdwJlLNx5c3R1bTpkZ
NzHwdVbYQvXcVjz-YW6oFkLpzmrArh086jPxAocAgYxw6vvU5PQVKwrgavaq9YE79TocYrFduWfbvPoy_RDn3WTk40mGjusftILS
S0pkkAWBv9YfajzHLmx7eYNNlSUHb1eei-zBhBkgtzLmc9g9Uvh02LICM03E-5ST-ymyaAB3RvjmFb-gwsazyk13wlkjKGcxqlvdgr1v55_t9gyAA1NYD6-Rsw31Dade00VpMzxHoP
WtZnbSqwRs78axp5mLZ20jBjWqPyn7lwq5uh5t2a7wEJ0QKAC1mVhQu0t2oAK9N1wK8w
```



deployment

The screenshot shows the Kubernetes Dashboard interface. On the left, a sidebar lists various cluster resources: Workloads, Cron Jobs, Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets, Service, Ingresses, Ingress Classes, Services, Config and Storage, Config Maps, Persistent Volume Claims, Secrets, Storage Classes, Cluster, and Cluster Role Bindings. The main content area has two charts: 'CPU Usage' and 'Memory Usage'. The 'CPU Usage' chart shows a single bar at approximately 0.001 cores from 16:34 to 16:42. The 'Memory Usage' chart shows a single bar at approximately 2 MiB from 16:34 to 16:42. Below the charts is a table titled 'Deployments' with one entry: 'myhelloworld'. The table columns are Name, Images, Labels, Pods, and Created. The 'Labels' column shows annotations like app.kubernetes.io/instance: myhelloworld, app.kubernetes.io/managed-by: Helm, and app.kubernetes.io/name: helloworld.

Name	Images	Labels	Pods	Created
myhelloworld	nginx:1.16.0	app.kubernetes.io/instance: myhelloworld app.kubernetes.io/managed-by: Helm app.kubernetes.io/name: helloworld	1 / 1	11 minutes ago

The screenshot shows the Kubernetes Dashboard interface. On the left is a sidebar with 'Workloads' expanded, showing 'Deployments' selected. The main area displays the 'Metadata' and 'Resource information' sections for a deployment named 'myhelloworld'. The deployment was created on Jul 9, 2023, and has an age of 11 minutes. It has a UID of feffdbec-c367-41a1-ab37-a082e84a44db. The resource information section shows a 'RollingUpdate' strategy with 0 min ready seconds and a revision history limit of 10. A selector is defined with labels: app.kubernetes.io/instance: myhelloworld, app.kubernetes.io/managed-by: Helm, app.kubernetes.io/name: helloworld, app.kubernetes.io/version: 1.16.0, and meta.helm.sh/chart: helloworld-0.1.0.

```
ubuntu@primary:~$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME      NAMESPACE   REVISION   UPDATED           STATUS      CHART          APP. V
myhelloworld  default     1          2023-07-09 16:33:35.294021765 +0300 EEST    deployed    helloworld-0.1.0    1.16.
ubuntu@primary:~$
```

```
# uninstall
```

```
ubuntu@primary:~$ helm uninstall myhelloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
release "myhelloworld" uninstalled
ubuntu@primary:~$
```

```
# go to K8s dashboard
```

The screenshot shows the Kubernetes dashboard interface. The URL in the address bar is <https://192.168.64.6:10443/#/deployment?namespace=default>. The dashboard has a dark theme with a blue header bar. On the left, there's a sidebar titled "Workloads" with a list of resources: Cron Jobs, Daemon Sets, Deployments (which is currently selected), Jobs, Pods, Replica Sets, Replication Controllers, Stateful Sets, Service, Ingresses, and Ingress Classes. The main content area is titled "Deployments" and displays the message "There is nothing to display here" and "No resources found." at the bottom.

The screenshot shows a presentation slide with a dark background. At the top, there's a logo for "HELM" featuring a crown icon and the word "Helm Chart". Below the logo, the title "Chapter - 3 (Helm CLI Command)" is displayed in a large, bold, blue font. Under the title, a list of Helm commands is presented in a numbered format: 1. `helm create`, 2. `helm install`, 3. `helm upgrade`, 4. `helm rollback`, 5. `helm --debug --dry-run`, 6. `helm template`, 7. `helm lint`, 8. `helm uninstall`. At the bottom center of the slide, the number "11" is visible.

Chapter 3 - Helm CLI Commands Mastery:

1. `helm create`
2. `helm install`
3. `helm upgrade`
4. `helm rollback`
5. Mastering debugging with `helm --debug --dry-run`
6. `helm template`, `helm lint`, `helm uninstall` and more!

1. Helm create & Install

Create Helm chart

```
> helm create helloworld
```

Install Helm chart

```
> helm install myhelloworld release helloworld
```

```
ubuntu@primary:~$ helm create helloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
Creating helloworld
ubuntu@primary:~$ ls -lart
total 56
-rw-r--r-- 1 ubuntu ubuntu 807 Jan  6 2022 .profile
-rw-r--r-- 1 ubuntu ubuntu 3771 Jan  6 2022 .bashrc
-rw-r--r-- 1 ubuntu ubuntu 220 Jan  6 2022 .bash_logout
drwxr-xr-x 3 root  root 4096 Jul  9 14:15 ..
drwx----- 2 ubuntu ubuntu 4096 Jul  9 14:15 .ssh
drwx----- 2 ubuntu ubuntu 4096 Jul  9 14:15 .cache
-rw-r--r-- 1 ubuntu ubuntu  0 Jul  9 14:15 .sudo_as_admin_successful
drwxr-xr-x 1 ubuntu ubuntu  96 Jul  9 14:55 common-work-directory
drwx----- 4 ubuntu ubuntu 4096 Jul  9 15:00 snap
drwxrwxr-x 3 ubuntu ubuntu 4096 Jul  9 15:01 .kube
-rw----- 1 ubuntu ubuntu 2303 Jul  9 16:32 .viminfo
-rw----- 1 ubuntu ubuntu 1920 Jul 11 20:10 .bash_history
drwxr-x--- 1 ubuntu ubuntu 1760 Jul 11 21:06 Home
drwxr-xr-x 4 ubuntu ubuntu 4096 Jul 11 21:25 helloworld
drwxr-x--- 9 ubuntu ubuntu 4096 Jul 11 21:25 .
ubuntu@primary:~$ ls -lart helloworld/
```

```
ubuntu@primary:~$ ls -lart helloworld/
total 28
-rw-r--r-- 1 ubuntu ubuntu 1877 Jul 11 21:25 values.yaml
drwxr-xr-x 3 ubuntu ubuntu 4096 Jul 11 21:25 templates
drwxr-xr-x 2 ubuntu ubuntu 4096 Jul 11 21:25 charts
-rw-r--r-- 1 ubuntu ubuntu 1146 Jul 11 21:25 Chart.yaml
-rw-r--r-- 1 ubuntu ubuntu  349 Jul 11 21:25 .helmignore
drwxr-x--- 9 ubuntu ubuntu 4096 Jul 11 21:25 ..
drwxr-xr-x 4 ubuntu ubuntu 4096 Jul 11 21:25 .
ubuntu@primary:~$ 
```

```
ubuntu@primary:~$ helm install myhelloworldrelease helloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME: myhelloworldrelease
LAST DEPLOYED: Tue Jul 11 21:29:56 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
   export POD_NAME=$(kubectl get pods --namespace default -l "app.kubernetes.io/name=helloworld,app.kubernetes.io/instance=myhelloworld"
   " -o jsonpath="{.items[0].metadata.name}")
   export CONTAINER_PORT=$(kubectl get pod --namespace default $POD_NAME -o jsonpath=".spec.containers[0].ports[0].containerPort")
   echo "Visit http://127.0.0.1:8080 to use your application"
   kubectl --namespace default port-forward $POD_NAME 8080:$CONTAINER_PORT
ubuntu@primary:~$
```

```
ubuntu@primary:~$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME          NAMESPACE     REVISION      UPDATED           STATUS        CHART
PP VERSION
myhelloworldrelease  default       1            2023-07-11 21:29:56.004749205 +0300 EEST    deployed    helloworld-0.1.0
.16.0
ubuntu@primary:~$
```

3. Helm Upgrade

Upgrade the helm chart

> **helm upgrade myhelloworldrelease helloworld**

4. Helm Rollback

Rollback the helm chart

> **helm rollback myhelloworldrelease helloworld**

```
# vim values.yaml
```

```
values.yaml + (D:\3_Devops-SI) + - +   
# Default values for helloworld.  
# This is a YAML-formatted file.  
# Declare variables to be passed into your templates.  
  
replicaCount: 2  
  
image:  
  repository: nginx  
  pullPolicy: IfNotPresent  
  # Overrides the image tag whose default is the chart appVersion.  
  tag: ""  
  
imagePullSecrets: []  
nameOverride: ""  
fullnameOverride: ""  
  
serviceAccount:  
  # Specifies whether a service account should be created  
  create: true  
  # Annotations to add to the service account  
  annotations: {}  
  # The name of the service account to use.  
  # If not set and create is true, a name is generated using the fullname template
```

```
ubuntu@primary:~$ helm upgrade myhelloworldrelease helloworld  
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config  
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config  
Release "myhelloworldrelease" has been upgraded. Happy Helming!  
NAME: myhelloworldrelease  
LAST DEPLOYED: Tue Jul 11 21:33:31 2023  
NAMESPACE: default  
STATUS: deployed  
REVISION: 2  
NOTES:
```

```
ubuntu@primary:~$ helm list -a  
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config  
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config  
NAME          NAMESPACE   REVISION      UPDATED           STATUS        CHART  
PP VERSION  
myhelloworldrelease    default     2            2023-07-11 21:33:31.869754402 +0300 EEST   deployed    helloworld-0.1.0  
ubuntu@primary:~$
```

🚀 4. Helm Rollback

Rollback the helm chart

```
> helm rollback myhelloworldrelease helloworld
```

```
ubuntu@primary:~$ helm rollback myhelloworldrelease 1
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
Rollback was a success! Happy Helming!
ubuntu@primary:~$
```

```
ubuntu@primary:~$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME          NAMESPACE      REVISION      UPDATED           STATUS
ERSION
myhelloworldrelease    default      3           2023-07-11 21:36:37.74212636 +0300 EEST deployed
0
ubuntu@primary:~$
```

🚀 5. Helm -- debug – dry-run

Validate your helm chart before install

```
helm install --dry-run  
--debug
```

Kubernetes API
Server

(validate and verify)



(Kubernetes
Resources)

```
> helm install myhelloworld --debug --dry-run helloworld
```

```
ubuntu@primary:~$ helm install myhelloworldrelease --debug --dry-run helloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
install.go:178: [debug] Original chart version: ""
install.go:195: [debug] CHART PATH: /home/ubuntu/helloworld

NAME: myhelloworldrelease
LAST DEPLOYED: Tue Jul 11 21:43:32 2023
NAMESPACE: default
STATUS: pending-install
REVISION: 1
USER-SUPPLIED VALUES:
{}

COMPUTED VALUES:
affinity: {}
autoscaling:
  enabled: false
  maxReplicas: 100
  minReplicas: 1
  targetCPUUtilizationPercentage: 80
fullnameOverride: ""
image:
```

```
# Source: helloworld/templates/deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: myhelloworldrelease
  labels:
    helm.sh/chart: helloworld-0.1.0
    app.kubernetes.io/name: helloworld
    app.kubernetes.io/instance: myhelloworldrelease
    app.kubernetes.io/version: "1.16.0"
    app.kubernetes.io/managed-by: Helm
spec:
  replicas: 2
  selector:
    matchLabels:
      app.kubernetes.io/name: helloworld
      app.kubernetes.io/instance: myhelloworldrelease
  template:
    metadata:
      labels:
        app.kubernetes.io/name: helloworld
```



6. Helm template

Renders the chart templates locally



> **helm install helloworld**

```
ubuntu@primary:~$ helm template helloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
---
# Source: helloworld/templates/serviceaccount.yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: release-name-helloworld
  labels:
    helm.sh/chart: helloworld-0.1.0
    app.kubernetes.io/name: helloworld
    app.kubernetes.io/instance: release-name
    app.kubernetes.io/version: "1.16.0"
    app.kubernetes.io/managed-by: Helm
---
# Source: helloworld/templates/service.yaml
apiVersion: v1
kind: Service
metadata:
  name: release-name-helloworld
  labels:
    helm.sh/chart: helloworld-0.1.0
    app.kubernetes.io/name: helloworld
    app.kubernetes.io/instance: release-name
    app.kubernetes.io/version: "1.16.0"
    app.kubernetes.io/managed-by: Helm
spec:
  type: ClusterIP
  ports:
```

```
name: release-name-helloworld
labels:
  helm.sh/chart: helloworld-0.1.0
  app.kubernetes.io/name: helloworld
  app.kubernetes.io/instance: release-name
  app.kubernetes.io/version: "1.16.0"
  app.kubernetes.io/managed-by: Helm
spec:
  type: ClusterIP
  ports:
    - port: 80
      targetPort: http
      protocol: TCP
      name: http
  selector:
    app.kubernetes.io/name: helloworld
    app.kubernetes.io/instance: release-name
  ...
# Source: helloworld/templates/deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: release-name-helloworld
  labels:
    helm.sh/chart: helloworld-0.1.0
    app.kubernetes.io/name: helloworld
    app.kubernetes.io/instance: release-name
    app.kubernetes.io/version: "1.16.0"
    app.kubernetes.io/managed-by: Helm
spec:
  replicas: 2
  selector:
    matchLabels:
      app.kubernetes.io/name: helloworld
```



7. Helm Lint

Find any errors or misconfiguration

> **helm lint helloworld**

```
ubuntu@primary:~$ helm lint helloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube
--> Linting helloworld
[INFO] Chart.yaml: icon is recommended

1 chart(s) linted, 0 chart(s) failed
ubuntu@primary:~$
```



8. Helm Uninstall

Remove the chart

```
> helm uninstall myhelloworldrelease
```

```
ubuntu@primary:~$ helm uninstall myhelloworldrelease
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
release "myhelloworldrelease" uninstalled
ubuntu@primary:~$ 
ubuntu@primary:~$ 
ubuntu@primary:~$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME      NAMESPACE      REVISION      UPDATED STATUS      CHART      APP VERSION
ubuntu@primary:~$
```



Chapter - 4

(Create your own custom chart)

1. Python Application with REST API
2. Create Docker Container
3. Push Docker image to docker repository
4. Create HelmChart for Python REST API app
5. Install the Helm chart
6. Verify the Helm Chart after installation



Chapter 4 - Creating Custom Helm Charts:

1. Designing a Python application with REST API
2. Docker Container creation and management
3. Pushing Docker images to repositories
4. Crafting Helm Charts for your Python REST API app
5. Helm chart installation and post-installation verification

A screenshot of the Visual Studio Code (VS Code) interface. The title bar shows the project name "python-flask-rest-api-project". The left sidebar contains icons for file operations like Open, Save, and Find. The main editor area displays the "main.py" file:

```
1 from flask import Flask, jsonify, request
2
3 app = Flask(__name__)
4
5
6 @app.route('/hello', methods=['GET'])
7 def helloworld():
8     if(request.method == 'GET'):
9         data = {"data": "Hello World"}
10    return jsonify(data)
11
12
13 if __name__ == '__main__':
14     app.run(host='0.0.0.0', port=9001)
```

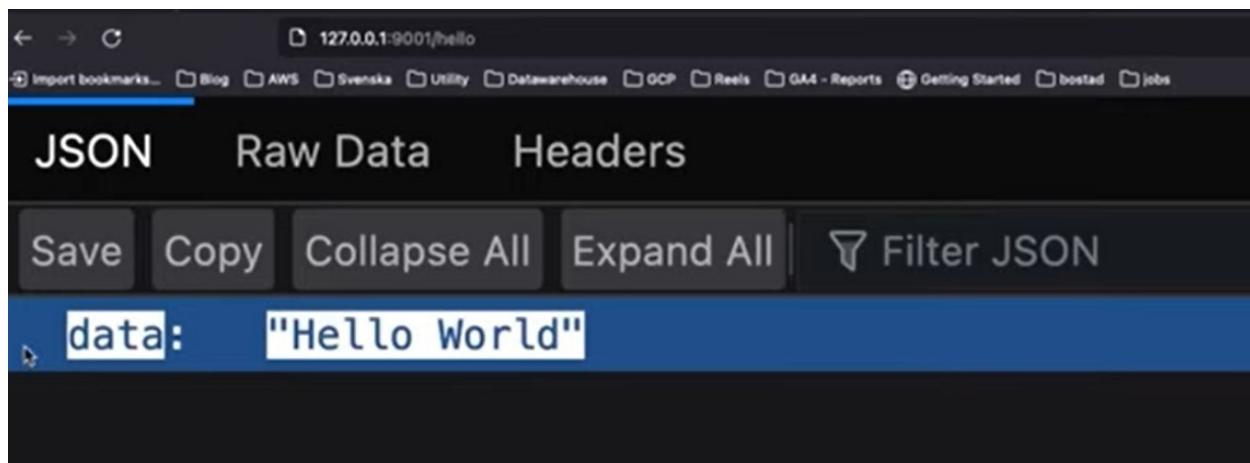
The status bar at the bottom shows "AWS: profile:default", "Ln 1, Col 1", "Tab Size: 4", "UTF-8", "CRLF", "Python", "Select Interpreter", "Go Live", and system information like "29°C Partly sunny" and "17:50 08-02-2025".

A screenshot of the Visual Studio Code (VS Code) interface, similar to the first one but with a terminal tab open. The title bar shows the project name "python-flask-rest-api-project". The left sidebar contains icons for file operations like Open, Save, and Find. The main editor area displays the "main.py" file, identical to the first screenshot.

The terminal tab is active and shows the following output:

```
* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
siddhartha@Siddhartha:/mnt/d/3_DevOps-SRE_Projects___/0006_DevOps_notes_Proj----/06-1-_HELM/helmchart/helloworld/chapter4/python-flask-rest-api-project$ py ma
in.py
siddhartha@Siddhartha:/mnt/d/3_DevOps-SRE_Projects___/0006_DevOps_notes_Proj----/06-1-_HELM/helmchart/helloworld/chapter4/python-flask-rest-api-project$ pytho
n3 main.py
* Serving Flask app 'main'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:9001
* Running on http://172.24.31.46:9001
Press CTRL+C to quit
172.24.16.1 - - [08/Feb/2025 17:54:38] "GET / HTTP/1.1" 404 -
172.24.16.1 - - [08/Feb/2025 17:54:38] "GET /favicon.ico HTTP/1.1" 404 -
```

The status bar at the bottom shows "AWS: profile:default", "Ln 9, Col 39", "Tab Size: 4", "UTF-8", "CRLF", "Python", "Select Interpreter", "Go Live", and system information like "29°C Partly sunny" and "17:56 08-02-2025".



```
# run the docker container locally run the docker img & push the docker image
```

1. Build Docker image

```
docker build -t python-project .
```

2. Build and tag according to docker hub

```
▶ docker tag python-rest-api rahulwagh17/python-flask-rest-api-project:python-rest-api
```

3. Docker push

```
▶ docker push rahulwagh17/python-flask-rest-api-project:python-rest-api
```

4. Docker pull

```
▶ docker pull rahulwagh17/python-flask-rest-api-project:python-rest-api
```

5. Run Docker image

```
# Practicals
```

OPEN EDITORS | 1 unsaved |

- main.py
- dockerfile
- Microsoft.PowerShell_profile.ps1

PYTHON-FLASK-REST-API-PROJECT

- dockerfile
- main.py
- README.md
- requirements.txt

```

1 FROM python:3.8
2 WORKDIR /app
3 COPY . /app
4 RUN pip --no-cache-dir install -r requirements.txt
5 CMD ["python3", "main.py"]
6

```

TERMINAL

```

PS> docker build -t python-project .
[+] Building 15.2s (5/9)
=> [internal] load build definition from dockerfile
=> => transferring dockerfile: 161B
=> [internal] load metadata for docker.io/library/python:3.8
=> [auth] library/python:pull token for registry-1.docker.io
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/4] FROM docker.io/library/python:3.8@sha256:d411270700143fa2683cc8264d9fa5d3279fd3b6afff62ae81ea2f9d070e390c
=> => resolve docker.io/library/python:3.8@sha256:d411270700143fa2683cc8264d9fa5d3279fd3b6afff62ae81ea2f9d070e390c
=> => sha256:5a98c896c047f960c5fd29d44fa778899a68e7ebfb6a6a4f2a3fb7baa902f6a 249B / 249B
=> => sha256:c48f13b5ff0f4b2e298de83a94a99fe7abdfb3335fe9b7811bf764abb1a4ac 18.06MB / 18.06MB
=> => sha256:cddc73ede6c704bfa2325e53c32db3553c8fc3a91dab6c092bb353f82098b09 6.16MB / 6.16MB
=> => sha256:01272fe8adbacc44af2d2b92994b31c40a151f4324ca392050d9e8d580927dd32 41.94MB / 211.27MB
=> => sha256:a173f2ae08962ea19db1e418ae84a0c9f71480b51f768a19332dfa83d7722a5 39.85MB / 64.39MB
=> => sha256:a47cff731e941e78bf63ca19f0811b675283e2c00dde10c5f78d93b2bc343 24.05MB / 24.05MB
=> => sha256:cdd62bf3913c498a16f7a7b1b6555ba43d02b2511c508fa4c0a9b1975ffe20e 23.07MB / 49.56MB
=> [internal] load build context
=> => transferring context: 32.70kB

```

AWS

```
# docker img
```

```

P> docker image ls
REPOSITORY          TAG      IMAGE ID      CREATED        SIZE
python-project      latest   463f7785835b  About a minute ago  1.46GB
siddhartha082/go-web-app    v1      9170efd33aa6   3 months ago   45.9MB
gcr.io/k8s-minikube/kicbase <none>  81df28859520   5 months ago   1.81GB
PS> docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS
NAMES
e13d821d8b64      gcr.io/k8s-minikube/kicbase:v0.0.45   "/usr/local/bin/entr..."   3 months ago     Exited (130) 3 months ago
minikube
PS>

```

5. Run Docker image

```

docker run -p 9001:9001 python-project

```

```

helm install my-app ./my-chart

```

```

helm uninstall my-app

```

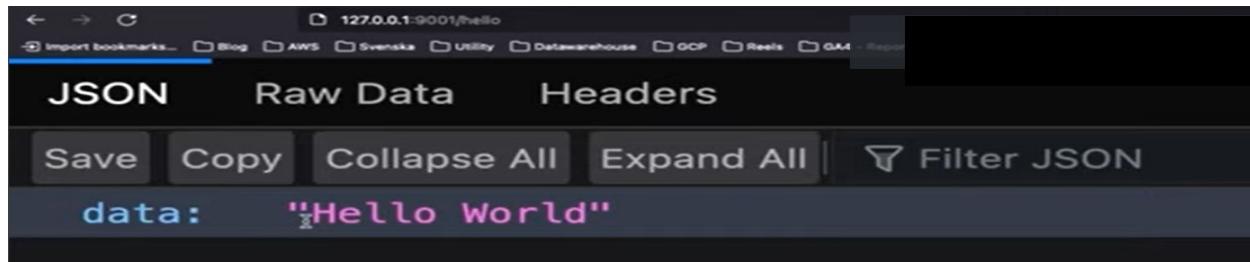
```

1 FROM python:3.8
2 WORKDIR /app
3 COPY . /app
4 RUN pip --no-cache-dir install -r requirements.txt
5 CMD ["python3", "main.py"]
6

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ESP-IDF

P> docker run -p 9001:9001 python-project
* Serving Flask app 'main'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:9001
* Running on http://172.17.0.2:9001
Press CTRL+C to quit
172.17.0.1 - - [08/Feb/2025 12:50:52] "GET /hello HTTP/1.1" 200 -
172.17.0.1 - - [08/Feb/2025 12:50:52] "GET /favicon.ico HTTP/1.1" 404 -

```



Push the docker img to docker hub

```

P> docker image ls
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
python-project      latest   463f7785835b  11 minutes ago  1.46GB
siddhartha082/go-web-app    v1      9170efd33aa6  3 months ago   45.9MB
gcr.io/k8s-minikube/kicbase <none>  81df28859520  5 months ago   1.81GB
P> docker login
Authenticating with existing credentials...
Login Succeeded
P> docker tag python-project siddhartha082/helm-chart-py-project:helm-chartv1

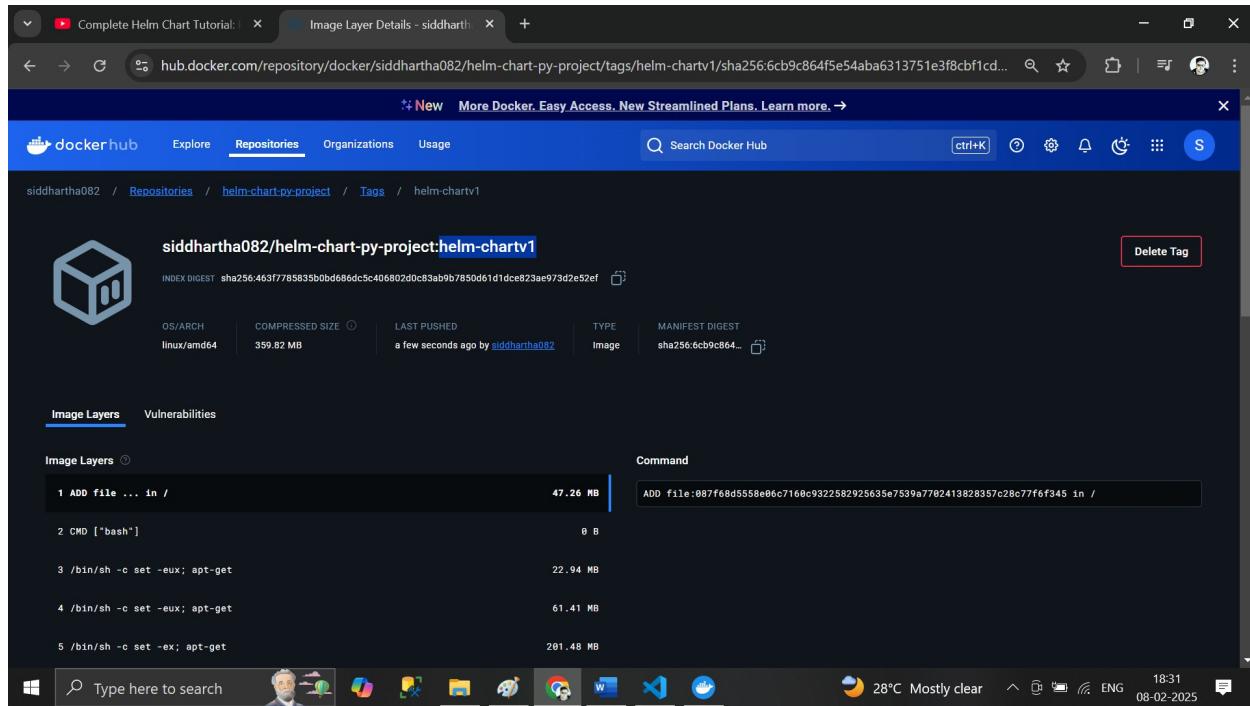
```

```

PS>docker push siddhartha082/helm-chart-py-project:helm-chartv1
The push refers to repository [docker.io/siddhartha082/helm-chart-py-project]
ff9518d9817c: Pushed
a173f2aeee8e9: Pushed
01272fe8adba: Pushed
cc48f13b5f0f: Pushed
5a98c896c047: Pushed
2c08678988eb: Pushed
e50e07929188: Pushed
cdd62bf39133: Pushed
a47cff7f31e9: Pushed
cdcc73e4e6c7: Pushed
511f39350be2: Pushed
helm-chartv1: digest: sha256:463f7785835b0bd686dc5c406802d0c83ab9b7850d61d1dce823ae973d2e52ef size: 856
P>

```

successfully img pushed in docker Hub



File Explorer

- OPEN EDITORS 1 unsaved
- main.py
- dockerfile
- Microsoft.PowerShell_profile.ps1

THON-FLASK-REST-API-PROJECT

- dockerfile
- main.py
- README.md
- requirements.txt

```

1 FROM python:3.8
2 WORKDIR /app
3 COPY . /app
4 RUN pip --no-cache-dir install -r requirements.txt
5 CMD ["python3", "main.py"]
6

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ESP-IDF

```

PS>docker image ls
REPOSITORY TAG IMAGE ID CREATED SIZE
python-project latest 463f7785835b 17 minutes ago 1.46GB
siddhartha082/helm-chart-py-project helm-chartv1 463f7785835b 17 minutes ago 1.46GB
siddhartha082/go-web-app v1 9170efd33aa6 3 months ago 45.9MB
gcr.io/k8s-minikube/kicbase <none> 81df28859520 5 months ago 1.81GB
PS>

```

create helm chart

VS Code Terminal Output:

```
PS>helm version
version.BuildInfo{Version:"v3.16+unreleased", GitCommit:"7b622102069925a22030cc650d77985a9f8cf36a", GitTreeState:"clean", GoVersion:"go1.22.8"}
PS>helm create helm-chart-py-project
Creating helm-chart-py-project
PS>ls

Directory: D:\3_Devops-SRE_Projects___\0006_DevOps_notes_Proj---\06-1_-HELM\helmchart\helloworld\chapter4\python-flask-rest-api-project

Mode LastWriteTime Length Name
---- -- -- -- --
d---- 08-02-2025 18:37 helm-chart-py-project
-a--- 08-02-2025 17:44 124 dockerfile
-a--- 08-02-2025 17:44 289 main.py
-a--- 08-02-2025 17:44 169 README.md
-a--- 08-02-2025 17:44 12 requirements.txt
```

VS Code Terminal Output:

```
PS>helm create helm-chart-py-project
Creating helm-chart-py-project
PS>ls

Directory: D:\3_Devops-SRE_Projects___\0006_DevOps_notes_Proj---\06-1_-HELM\helmchart\helloworld\chapter4\python-flask-rest-api-project\helm-chart-py-project

Mode LastWriteTime Length Name
---- -- -- -- --
d---- 08-02-2025 18:37 charts
d---- 08-02-2025 18:37 templates
-a--- 08-02-2025 18:37 349 .helmignore
-a--- 08-02-2025 18:37 1157 Chart.yaml
-a--- 08-02-2025 18:37 4387 values.yaml
```

```

FROM python:3.8
WORKDIR /app
PS>ls

Directory: D:\_3_Devops-SRE_Projects___\0006_DevOps_notes_Proj---\06-1_HELM\helmchart\helloworld\chapter4\python-flask-rest-api-project\helm-chart-py-project

Mode LastWriteTime Length Name
---- ----- ----- 
d---- 08-02-2025 18:37 helm-chart-py-project
-a--- 08-02-2025 17:44 124 dockerfile

PS>

Mode LastWriteTime Length Name
---- ----- ----- 
d---- 08-02-2025 18:37 charts
d---- 08-02-2025 18:37 templates
-a--- 08-02-2025 18:37 349 .helmignore
-a--- 08-02-2025 18:37 1157 Chart.yaml
-a--- 08-02-2025 18:37 4307 values.yaml

```

update the chart.yaml edit mode

```

apiVersion: v2
name: helm-chart-py-project
description: A Helm chart for Kubernetes

# A chart can be either an 'application' or a 'library' chart.
#
# Application charts are a collection of templates that can be packaged into versioned archives
# to be deployed.
#
# Library charts provide useful utilities or functions for the chart developer. They're included as
# a dependency of application charts to inject those utilities and functions into the rendering
# pipeline. Library charts do not define any templates and therefore cannot be deployed.
type: application

# This is the chart version. This version number should be incremented each time you make changes
# to the chart and its templates, including the app version.
# Versions are expected to follow Semantic Versioning (https://semver.org/)
version: 0.1.0

# This is the version number of the application being deployed. This version number should be
# incremented each time you make changes to the application. Versions are not expected to
# follow Semantic Versioning. They should reflect the version the application is using.
# It is recommended to use it with quotes.
appVersion: "1.16.0"

```

set comment the app_version .save + quit

now update the same for values.yaml file

```

FROM python:3.8
WORKDIR /app
# Default values for helm-chart-py-project.
# This is a YAML-formatted file.
# Declare variables to be passed into your templates.

# This will set the replicaset count more information can be found here: https://kubernetes.io/docs/concepts/workloads/controllers/replicaset/
replicaCount: 1

# This sets the container image more information can be found here: https://kubernetes.io/docs/concepts/containers/images/
image:
  repository: nginx
  # This sets the pull policy for images.
  pullPolicy: IfNotPresent
  # Overrides the image tag whose default is the chart appVersion.
  tag: ""

# This is for the secrets for pulling an image from a private repository more information can be found here: https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry/
imagePullSecrets: []

# This is to override the chart name.
nameOverride: ""
fullnameOverride: ""

# This section builds out the service account more information can be found here: https://kubernetes.io/docs/concepts/security/service-accounts/
serviceAccount:
  # Specifies whether a service account should be created
  create: true
  "values.yaml" [unix] 123L, 4307B

```

change the repo – ngnix to your docker repo name : tagname too save + quit

```

# Default values for helm-chart-py-project.
# This is a YAML-formatted file.
# Declare variables to be passed into your templates.

# This will set the replicaset count more information can be found here: https://kubernetes.io/docs/concepts/workloads/controllers/replicaset/
replicaCount: 1

# This sets the container image more information can be found here: https://kubernetes.io/docs/concepts/containers/images/
image:
  repository: siddhartha082/helm-chart-py-project:helm-chartv1
  # This sets the pull policy for images.
  pullPolicy: IfNotPresent
  # Overrides the image tag whose default is the chart appVersion.
  tag: ""

# This is for the secrets for pulling an image from a private repository more information can be found here: https://kubernetes.io/docs/tasks/configure-pod-container/pull-image-private-registry/
imagePullSecrets: []

# This is to override the chart name.
nameOverride: ""
fullnameOverride: ""

# This section builds out the service account more information can be found here: https://kubernetes.io/docs/concepts/security/service-accounts/
serviceAccount:
  # Specifies whether a service account should be created
  create: true
  "values.yaml" [unix] 123L, 4307B

```

we have updated helm chart + values yaml ..

in Values.yaml change the clusterIP to NodePort .. for port forwarding

```
● Microsoft.PowerShell_profile...
PYTHON-FLASK-REST-API-PROJECT
> helm-chart-py-project
  dockerfile
  main.py
  README.md
  requirements.txt

# This is for setting up a service more information can be found here: https://kubernetes.io/docs/concepts/services-networking/service/
service:
  # This sets the service type more information can be found here: https://kubernetes.io/docs/concepts/services-networking/service/#publishing-services-service-types
  type: ClusterIP
  # This sets the ports more information can be found here: https://kubernetes.io/docs/concepts/services-networking/service/#field-spec-ports
  port: 80

# This block is for setting up the ingress for more information can be found here: https://kubernetes.io/docs/concepts/services-networking/ingress/
ingress:
  enabled: false
  className: ""
  annotations: {}
    # kubernetes.io/ingress.class: nginx
    # kubernetes.io/tls-acme: "true"
  hosts:
    - host: chart-example.local
      paths:
        - path: /
          pathType: ImplementationSpecific
  tls: []
    # - secretName: chart-example-tls
    #   hosts: [chart-example.local]
```

NodePort

```
● Microsoft.PowerShell_profile...
PYTHON-FLASK-REST-API-PROJECT
> helm-chart-py-project
  dockerfile
  main.py
  README.md
  requirements.txt

# This is for setting up a service more information can be found here: https://kubernetes.io/docs/concepts/services-networking/service/
service:
  # This sets the service type more information can be found here: https://kubernetes.io/docs/concepts/services-networking/service/#publishing-services-service-types
  type: NodePort
  # This sets the ports more information can be found here: https://kubernetes.io/docs/concepts/services-networking/service/#field-spec-ports
  port: 80

# This block is for setting up the ingress for more information can be found here: https://kubernetes.io/docs/concepts/services-networking/ingress/
ingress:
  enabled: false
  className: ""
  annotations: {}
    # kubernetes.io/ingress.class: nginx
    # kubernetes.io/tls-acme: "true"
  hosts:
    - host: chart-example.local
      paths:
        - path: /
          pathType: ImplementationSpecific
  tls: []
```

Go inside the template . deployment.yml file

The screenshot shows a Windows desktop environment with the Visual Studio Code (VS Code) application open. The title bar reads "python-flask-rest-api-project". The left sidebar displays the "EXPLORER" view with a tree structure showing a project named "PYTHON-FLASK-REST-API-PROJECT" containing files like "main.py", "dockerfile", "README.md", and "requirements.txt". The main area shows a terminal window with the following content:

```
PS>vim values.yaml
PS>ls

Directory: D:\_3_Devops-SRE_Projects_\0006_DevOps_notes_Proj---\06-1-_HELM\helmchart\helloworld\chapter4\python-flask-rest-api-project\helm-chart-py-project

Mode LastWriteTime Length Name
---- -- -- -- --
d---- 08-02-2025 18:37 charts
d---- 08-02-2025 18:37 templates
-a---- 08-02-2025 18:42 563 .Chart.yaml.un~
-a---- 08-02-2025 18:37 349 .helmignore
-a---- 08-02-2025 18:51 999 .values.yaml.un~
-a---- 08-02-2025 18:42 1158 Chart.yaml
-a---- 08-02-2025 18:37 1157 Chart.yaml~
-a---- 08-02-2025 18:51 4349 values.yaml
-a---- 08-02-2025 18:46 4350 values.yaml~

PS>cd
```

This screenshot is nearly identical to the one above, but the terminal output has changed. The user has navigated into the "templates" directory:

```
PS>cd templates
PS>ls

Directory: D:\_3_Devops-SRE_Projects_\0006_DevOps_notes_Proj---\06-1-_HELM\helmchart\helloworld\chapter4\python-flask-rest-api-project\helm-chart-py-project\templates

Mode LastWriteTime Length Name
---- -- -- -- --
d---- 08-02-2025 18:37 tests
-a---- 08-02-2025 18:37 2233 deployment.yaml
-a---- 08-02-2025 18:37 1033 hpa.yaml
-a---- 08-02-2025 18:37 1130 ingress.yaml
-a---- 08-02-2025 18:37 1800 NOTES.txt
-a---- 08-02-2025 18:37 403 service.yaml
-a---- 08-02-2025 18:37 417 serviceaccount.yaml
-a---- 08-02-2025 18:37 1922 _helpers.tpl

PS>
```

```

apiVersion: apps/v1
kind: Deployment
metadata:
  name: {{ include "helm-chart-py-project.fullname" . }}
  labels:
    {{- include "helm-chart-py-project.labels" . | nindent 4 }}
spec:
  {{- if not .Values.autoscaling.enabled }}
  replicas: {{ .Values.replicaCount }}
  {{- end }}
  selector:
    matchLabels:
      {{- include "helm-chart-py-project.selectorLabels" . | nindent 6 }}
  template:
    metadata:
      {{- with .Values.podAnnotations }}
      annotations:
        {{- toYaml . | nindent 8 }}
      {{- end }}
      labels:
        {{- include "helm-chart-py-project.labels" . | nindent 8 }}
        {{- with .Values.podLabels }}
        {{- toYaml . | nindent 8 }}
        {{- end }}
    spec:
      {{- with .Values.imagePullSecrets }}
      imagePullSecrets:
        {{- toYaml . | nindent 8 }}
      {{- end }}

```

-- INSERT --

```

spec:
  {{- with .Values.imagePullSecrets }}
  imagePullSecrets:
    {{- toYaml . | nindent 8 }}
  {{- end }}
  serviceAccountName: {{ include "helm-chart-py-project.serviceAccountName" . }}
  securityContext:
    {{- toYaml .Values.podSecurityContext | nindent 8 }}
  containers:
    - name: {{ .Chart.Name }}
      securityContext:
        {{- toYaml .Values.securityContext | nindent 12 }}
        image: "{{ .Values.image.repository }}:{{ .Values.image.tag | default .Chart.AppVersion }}"
        imagePullPolicy: {{ .Values.image.pullPolicy }}
      ports:
        - name: http
          containerPort: 9001
          protocol: TCP
          livenessProbe:
            {{- toYaml .Values.livenessProbe | nindent 12 }}
          readinessProbe:
            {{- toYaml .Values.readinessProbe | nindent 12 }}
        resources:
          {{- toYaml .Values.resources | nindent 12 }}
        {{- with .Values.volumeMounts }}
        volumeMounts:

```

Now Comment out probes .. liveness + readiness probes

WORKDIR /app

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL	PORTS	ESP-IDF
----------	--------	---------------	-----------------	-------	---------

```

securityContext:
  {{- toYaml .Values.podSecurityContext | nindent 8 }}
containers:
- name: {{ .Chart.Name }}
  securityContext:
    {{- toYaml .Values.securityContext | nindent 12 }}
  image: "{{ .Values.image.repository }}:{{ .Values.image.tag | default .Chart.AppVersion }}"
  imagePullPolicy: {{ .Values.image.pullPolicy }}
  ports:
    - name: http
      containerPort: 9001
      protocol: TCP
      #livenessProbe:
      #  {{- toYaml .Values.livenessProbe | nindent 12 }}
      #readinessProbe:
      #  {{- toYaml .Values.readinessProbe | nindent 12 }}
  resources:
    {{- toYaml .Values.resources | nindent 12 }}
  {{- with .Values.volumeMounts --}}
  volumeMounts:
    {{- toYaml . | nindent 12 }}
  {{- end --}}
  {{- with .Values.volumes --}}

```

as in values.yml file we have commented the app_version

WORKDIR /app

PROBLEMS	OUTPUT	DEBUG CONSOLE	TERMINAL	PORTS	ESP-IDF
----------	--------	---------------	-----------------	-------	---------

```

securityContext:
  {{- toYaml .Values.podSecurityContext | nindent 8 }}
containers:
- name: {{ .Chart.Name }}
  securityContext:
    {{- toYaml .Values.securityContext | nindent 12 }}
  image: "{{ .Values.image.repository }}"
  imagePullPolicy: {{ .Values.image.pullPolicy }}
  ports:
    - name: http
      containerPort: 9001
      protocol: TCP
      #livenessProbe:
      #  {{- toYaml .Values.livenessProbe | nindent 12 }}
      #readinessProbe:
      #  {{- toYaml .Values.readinessProbe | nindent 12 }}
  resources:
    {{- toYaml .Values.resources | nindent 12 }}

```

```

File Edit Selection View Go Run ... ← → python-flask-rest-api-project
EXPLORER ... main.py dockerfile Microsoft.PowerShell_profile.ps1 ...
OPEN EDITORS 1 unsaved
main.py
dockerfile
Microsoft.PowerShell_profile...
PYTHON-FLASK-REST-API-PROJECT helm-chart-py-project
main.py
README.md
requirements.txt
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS ESP-IDF
powershell + ... ×
PS>ls
Directory: D:\3_Devops-SRE_Projects_\0006_DevOps_notes_Proj---\06-1_-_HELM\helmchart\helloworld\chapter4\python-flask-rest-api-project\helm-chart-py-project

Mode LastWriteTime Length Name
---- ----- ---- -
d---- 08-02-2025 18:37 charts
d---- 08-02-2025 19:04 templates
-a--- 08-02-2025 18:42 563 .Chart.yaml.un~
-a--- 08-02-2025 18:37 349 .helmignore
-a--- 08-02-2025 18:51 999 .values.yaml.un~
-a--- 08-02-2025 18:42 1158 Chart.yaml
-a--- 08-02-2025 18:37 1157 Chart.yaml~
-a--- 08-02-2025 18:51 4349 values.yaml
-a--- 08-02-2025 18:46 4350 values.yaml~

PS>cd ...
PS>helm show chart helm-chart-py-project
apiVersion: v2
description: A Helm chart for Kubernetes
name: helm-chart-py-project
type: application
version: 0.1.0
PS>

```

```

ubuntu@primary:~$ helm install mypythonhelmchart python-flask-rest-api-project
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME: mypythonhelmchart
LAST DEPLOYED: Tue Jul 11 23:29:34 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
  export NODE_PORT=$(kubectl get --namespace default -o jsonpath=".spec.ports[0].nodePort" services mypythonhelmchart)
  export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath=".items[0].status.addresses[0].address")
  echo http://$NODE_IP:$NODE_PORT
ubuntu@primary:~$ 

```

```

ubuntu@primary:~$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME           NAMESPACE      REVISION      UPDATED             STATUS      CHART
APP VERSION
mypythonhelmchart   default      1            2023-07-11 23:29:34.825975291 +0300 EEST    deployed   python-flask-rest-
-project-0.1.0
ubuntu@primary:~$ 

```

```

ubuntu@primary:~$ kubectl get deployments
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
mypythonhelmchart-python-flask-rest-api-project   1/1     1           1           64s
ubuntu@primary:~$ 
ubuntu@primary:~$ kubectl get service
NAME          TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)      AGE
kubernetes    ClusterIP  10.152.183.1  <none>        443/TCP     2d8h
mypythonhelmchart-python-flask-rest-api-project   NodePort   10.152.183.123  <none>        80:31866/TCP  87s
ubuntu@primary:~$ 

```

```
ubuntu@primary:~$ hostname -I  
192.168.64.6 10.1.226.128 fd16:8590:7284:4e59:5054:ff:fe7f:b17d  
ubuntu@primary:~$
```

A screenshot of a web browser window. The address bar shows the URL `192.168.64.6:31886/hello`. Below the address bar, there are tabs for "JSON", "Raw Data", and "Headers". Underneath these tabs is a row of buttons: "Save", "Copy", "Collapse All", "Expand All", and "Filter JSON". The main content area displays a single line of JSON data: `data: "Hello World"`.



Chapter - 5

(Helmfile)

1. What is **Helmfile** and Why we need it?
2. Installing the Helmfile?
3. Manage your Helmchart using **Helmfile**?
4. Install Helmchart from remote **Git Repo** using Helmfile
5. Install **multiple HelmChart**



1. What is Helmfile and why we need it?

helm commands

```
helm install my-app ./my-chart
```

```
helm uninstall my-app
```



\$ **helmfile sync**



1. What is Helmfile and why we need it?

helmfile

```
---
```

```
releases:
```

```
  - name: helloworld
```

```
    chart: ./helloworld
```

```
    installed: true
```

→ Name of helmchart
→ Set true for install



1. What is Helmfile and why we need it?

helmfile

```
---
```

```
releases:
```

```
  - name: helloworld
```

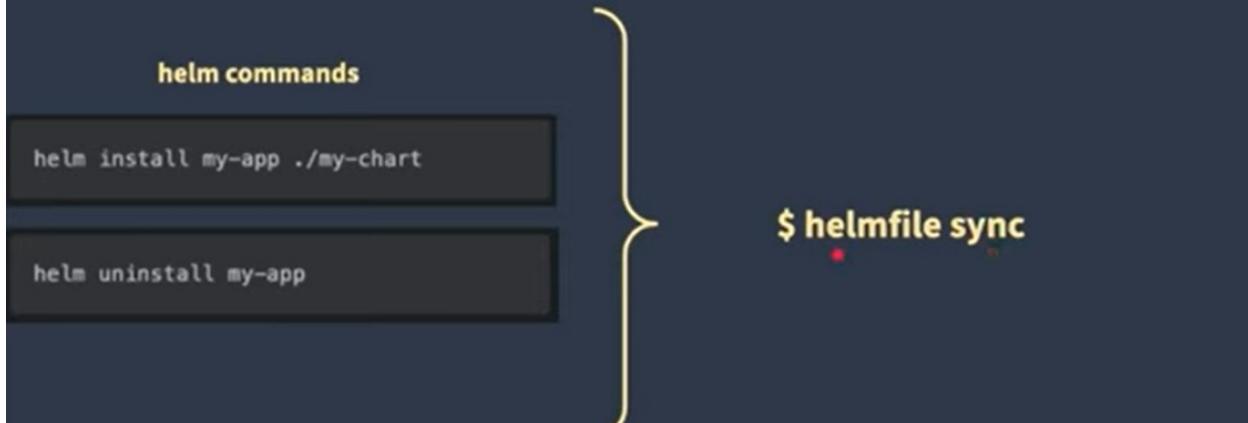
```
    chart: ./helloworld
```

```
    installed: true
```

→ Name of helmchart
→ Set true for install



1. What is Helmfile and why we need it?



<https://jhooq.com/helmfile-manage-helmchart/>

How to use Helmfile for managing helm chart?

Helmfile is another wrapper working on top of Helm Chart. Just like Helm Chart, Helmfile also uses the YAML for writing the configurations.

But why do you need Helmfile if you are working with Kubernetes and Helm Chart?

Here are some key benefits of using Helmfile -

1. You can bundle several Helm Charts into a Single Helmfile to manage your kubernetes eco system
2. Helmfile helps you to keep isolation between the different environments(developemnt, staging, production)
3. It can help you to identify the differences between the new changes which you want to apply against the existing running deployment inside kubernetes cluster
4. Helmfile uses the Go Templates which lets you templatify your Helmfile and also you can use Sprig Library with functions - requiredEnv, exec, readFile, toYaml, fromYaml, setValueAtPath, get, tpl, required, fetchSecretValue, expandSecretRefs
5. With the help of HelmFile you can deploy multi-tier applications inside kubernetes cluster.

In this guide I am assuming that you have not used the Helmfile before so the first most appropriate step which I think would be to *Installing Helmfile*?

(Note :- But there is a prerequisite before installing the Helmfile, [you must install the helmchart](#))

1. Download the Helmfile from [GitHub](#)
2. Rename the file from - **helmfile_linux_amd64** to **helmfile**

```
1 mv helmfile_linux_amd64 helmfile
```

BASH

3. Change the permission of the file and make it executable

```
1 chmod 777 helmfile
```

BASH

4. Move the file from current location to **/usr/local/bin**

```
1 mv helmfile /usr/local/bin
```

5. After moving the file verify the Helmfile installation by running the command - **\$ helmfile --version**

```
$ helmfile --version
helmfile version v0.141.0
```

Helmfile version after installation

https://github.com/roboll/helmfile/releases/download/v0.144.0/helmfile_windows_amd64.exe

```
ubuntu@primary:~$ wget https://github.com/roboll/helmfile/releases/download/v0.144.0/helmfile_linux_arm64
--2023-07-17 22:20:33-- https://github.com/roboll/helmfile/releases/download/v0.144.0/helmfile_linux_arm64
Resolving github.com (github.com)... 140.82.121.3
Connecting to github.com (github.com)|140.82.121.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/74499101/933a6501-5232-466b-a39f-75697ec7ec
b2?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAINNJA4CSEH53AX2F20230717%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20
230717T192033Z&X-Amz-Expires=300&X-Amz-Signature=bef381931436f0319fb49f3e8e6fa3b0ff69ad7539044726fbaf520d4b268fdb&X-Amz-SignedHe
ders=host&actor_id=0&key_id=0&repo_id=74499101&response-content-disposition=attachment%3B%20filename%3Dhelmfile_linux_arm64&respons
e-content-type=application%2Foctet-stream [following]
--2023-07-17 22:20:34-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/74499101/933a6501-5232-466b-
a39f-75697ec7ecb2?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAINNJA4CSEH53AX2F20230717%2Fus-east-1%2Fs3%2Faws4_reques
t&X-Amz-Date=20230717T192033Z&X-Amz-Expires=300&X-Amz-Signature=bef381931436f0319fb49f3e8e6fa3b0ff69ad7539044726fbaf520d4b268fdb&X-
Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=74499101&response-content-disposition=attachment%3B%20filename%3Dhelmfile_linu
x_arm64&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.108
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.108|
HTTP request sent, awaiting response... 200 OK
Length: 45861899 (44M) [application/octet-stream]
Saving to: 'helmfile_linux_arm64'
```

```
ubuntu@primary:~$ ls
Home common-work-directory helmfile_linux_arm64 python-flask-rest-api-project snap
ubuntu@primary:~$
ubuntu@primary:~$ mv helmfile_linux_arm64 helmfile
ubuntu@primary:~$ ls
Home common-work-directory helmfile python-flask-rest-api-project snap
ubuntu@primary:~$
```

```
ubuntu@primary:~$ ls -lart
total 44860
-rw-r--r-- 1 ubuntu ubuntu      807 Jan  6 2022 .profile
-rw-r--r-- 1 ubuntu ubuntu    3771 Jan  6 2022 .bashrc
-rw-r--r-- 1 ubuntu ubuntu     220 Jan  6 2022 .bash_logout
-rw-rw-r-- 1 ubuntu ubuntu 45861899 Mar 31 2022 helmfile
drwxr-xr-x 3 root  root   4096 Jul  9 14:15 ..
drwx----- 2 ubuntu ubuntu   4096 Jul  9 14:15 .ssh
-rw-r--r-- 1 ubuntu ubuntu        0 Jul  9 14:15 .sudo_as_admin_successful
drwxr-xr-x 1 ubuntu ubuntu      96 Jul  9 14:55 common-work-directory
drwx----- 4 ubuntu ubuntu   4096 Jul  9 15:00 snap
drwxrwxr-x 3 ubuntu ubuntu   4096 Jul  9 15:01 .kube
drwx----- 3 ubuntu ubuntu   4096 Jul 11 21:28 .cache
drwxr-xr-x 4 ubuntu ubuntu   4096 Jul 11 23:28 python-flask-rest-api-project
-rw----- 1 ubuntu ubuntu 14252 Jul 11 23:28 .viminfo
-rw----- 1 ubuntu ubuntu    3249 Jul 11 23:37 .bash_history
drwxr-x--- 1 ubuntu ubuntu   1760 Jul 17 19:15 Home
-rw-rw-r-- 1 ubuntu ubuntu     165 Jul 17 22:20 .wget-hsts
drwxr-x--- 9 ubuntu ubuntu   4096 Jul 17 22:21 .
ubuntu@primary:~$
```

```
ubuntu@primary:~$ chmod 777 helmfile
ubuntu@primary:~$
ubuntu@primary:~$ ls -lart
total 44860
-rw-r--r-- 1 ubuntu ubuntu      807 Jan  6 2022 .profile
-rw-r--r-- 1 ubuntu ubuntu    3771 Jan  6 2022 .bashrc
-rw-r--r-- 1 ubuntu ubuntu     220 Jan  6 2022 .bash_logout
-rwxrwxrwx 1 ubuntu ubuntu 45861899 Mar 31 2022 helmfile
drwxr-xr-x 3 root  root      4096 Jul  9 14:15 ..
drwx----- 2 ubuntu ubuntu      4096 Jul  9 14:15 .ssh
-rw-r--r-- 1 ubuntu ubuntu       0 Jul  9 14:15 .sudo_as_admin_successful
drwxr-xr-x 1 ubuntu ubuntu      96 Jul  9 14:55 common-work-directory
drwx----- 4 ubuntu ubuntu     4096 Jul  9 15:00 snap
drwxrwxr-x 3 ubuntu ubuntu     4096 Jul  9 15:01 .kube
drwx----- 3 ubuntu ubuntu     4096 Jul 11 21:28 .cache
drwxr-xr-x 4 ubuntu ubuntu     4096 Jul 11 23:28 python-flask-rest-api-project
-rw----- 1 ubuntu ubuntu   14252 Jul 11 23:28 .viminfo
-rw----- 1 ubuntu ubuntu     3249 Jul 11 23:37 .bash_history
drwxr-x--- 1 ubuntu ubuntu    1760 Jul 17 19:15 Home
-rw-rw-r-- 1 ubuntu ubuntu      165 Jul 17 22:20 .wget-hsts
drwxr-x--- 9 ubuntu ubuntu     4096 Jul 17 22:21 .
ubuntu@primary:~$
```

#Mv this file usr/local/bin dir

```
ubuntu@primary:~$ mv helmfile /usr/local/bin
mv: cannot move 'helmfile' to '/usr/local/bin/helmfile': Permission denied
ubuntu@primary:~$ sudo mv helmfile /usr/local/bin
ubuntu@primary:~$ ls -lart /usr/local/bin
total 133500
-rwxrwxrwx 1 ubuntu ubuntu 45861899 Mar 31 2022 helmfile
drwxr-xr-x 10 root  root     4096 Jun 16 05:11 ..
-rwxr-xr-x 1 root  root  46202880 Jul  9 14:59 kubectl
-rwxr-xr-x 1 root  root  44630016 Jul  9 15:18 helm
drwxr-xr-x 2 root  root     4096 Jul 17 22:23 .
ubuntu@primary:~$
```

```
ubuntu@primary:~$ helmfile -version
helmfile version v0.144.0
ubuntu@primary:~$
```

```
ubuntu@primary:~$ helm create helloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
Creating helloworld
ubuntu@primary:~$ ls
Home  common-work-directory  helloworld  python-flask-rest-api-project  snap
ubuntu@primary:~$
```

create helmfile .yaml

```
ubuntu@primary:~$ vi helmfile.yaml
```

Complete Helm Chart Tutorial | How to use Helmfile for managing charts | Releases - roboll/helmfile | Image Layer Details - siddharth | - +

[jhoqq.com/helmfile-manage-helmchart/](#)

J->Hooq Home Ansible Kubernetes Terraform YouTube About Contact

Creating helloworld

helm create helloworld helmchart

2. Let's create a Helmfile for the helmchart we have just created and we are going to create **helmfile.yaml** at the same location where we have created **helloworld** helmchart.

Here are the contents of **helmfile.yaml**-

```
1 ---  
2 releases:  
3  
4   - name: helloworld  
5     chart: ./helloworld  
6     installed: true
```

Installed: true - If you want to install the helmchart using helmfile then you must set this flag to **true**

3. After creating the helloworld helmchart and its respective helmfile let's try to install the helm chart by running the command **\$ helmfile sync**

```
---  
releases:  
  
- name: helloworldreleas[  
  chart: ./helloworld  
  installed: true  
  
~  
~
```

```
ubuntu@primary:~$ ls
Home common-work-directory helloworld python-flask-rest-api-project snap
ubuntu@primary:~$ vi helmfile.yaml
ubuntu@primary:~$ ls
Home common-work-directory helloworld helmfile.yaml python-flask-rest-api-project snap
ubuntu@primary:~$ cat helmfile.yaml
---
releases:

- name: helloworldreleas
  chart: ./helloworld
  installed: true
```

```
ubuntu@primary:~$ helmfile sync
Building dependency release=helloworldreleas, chart=helloworld
Affected releases are:
  helloworldreleas (helloworld) UPDATED

Upgrading release=helloworldreleas, chart=helloworld
Release "helloworldreleas" does not exist. Installing it now.
NAME: helloworldreleas
LAST DEPLOYED: Mon Jul 17 22:28:07 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
  export POD_NAME=$(kubectl get pods --namespace default -l "app.kubernetes.io/name=helloworldreleas" -o jsonpath="{.items[0].metadata.name}")
  export CONTAINER_PORT=$(kubectl get pod --namespace default $POD_NAME -o jsonpath=".spec.containers[0].port.number")
  echo "Visit http://127.0.0.1:8080 to use your application"
  kubectl --namespace default port-forward $POD_NAME 8080:$CONTAINER_PORT

Listing releases matching ^helloworldreleas$
helloworldreleas      default      1          2023-07-17 22:28:07.696860344 +0300
0.1.0  1.16.0

UPATED RELEASES:
NAME        CHART      VERSION
helloworldreleas  helloworld  0.1.0
```

```
ubuntu@primary:~$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME        NAMESPACE   APP VERSION    REVISION    UPDATED           STATUS      CHART
helloworldreleas  default    0.1.0  1.16.0       1          2023-07-17 22:28:07.696860344 +0300 EEST  I  deployed  helloworld
```

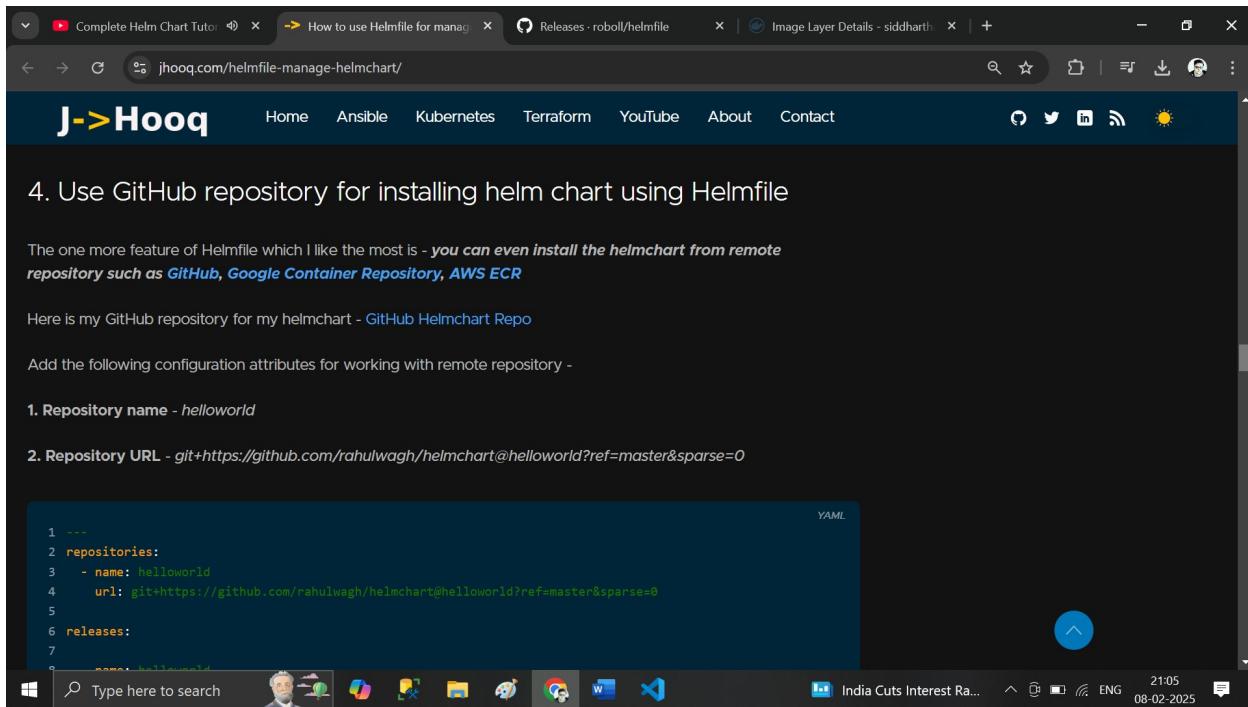
```
# del
```

```
ubuntu@primary:~$ vi helmfile.yaml
```

Make it false

```
---  
releases:  
  
- name: helloworldreleas  
  chart: ./helloworld  
  installed: fals■
```

```
ubuntu@primary:~$ helmfile sync  
Listing releases matching ^helloworldreleas$  
helloworldreleas      default      1          2023-07-17 22:28:07.696860344 +0300 EEST      deployed      helloworld  
0.1.0  1.16.0  
  
Affected releases are:  
  helloworldreleas (./helloworld) DELETED  
  
Deleting helloworldreleas  
release "helloworldreleas" uninstalled  
  
DELETED RELEASES:  
NAME  
helloworldreleas  
ubuntu@primary:~$ ■
```



The screenshot shows a Microsoft Edge browser window with the following details:

- Address Bar:** jhooq.com/helmfile-manage-helmchart/
- Page Title:** How to use Helmfile for managing charts
- Content:** A step-by-step guide. Step 4 is titled "4. Use GitHub repository for installing helm chart using Helmfile". It discusses the ability to install charts from remote repositories like GitHub, Google Container Repository, and AWS ECR. It provides a GitHub repository link for a helmchart named "helloworld".
- Code Snippet:** A code editor window shows a Helmfile snippet with the following YAML:

```
1 ---  
2 repositories:  
3   - name: helloworld  
4     url: git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0  
5  
6 releases:  
7   - name: helloworld
```

The browser interface includes a search bar, taskbar icons, and system status indicators at the bottom.

Add the following configuration attributes for working with remote repository -

1. Repository name - `helloworld`
2. Repository URL - `git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0`

```
YAML
1 ---
2 repositories:
3   - name: helloworld
4     url: git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0
5
6 releases:
7
8   - name: helloworld
9     chart: helloworld/helloworld
10    installed: false
```

Now you can run the `$ helmfile sync` command for installing the helm chart from remote repository

<https://github.com/helm/helm>

```
ubuntu@primary:~$ helm plugin install https://github.com/aslafy-z/helm-git --version 0.15.1
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
Installed plugin: helm-git
ubuntu@primary:~$
```

```
ubuntu@primary:~$ vi helmfile.yaml
ubuntu@primary:~$ vi helmfile.yaml
ubuntu@primary:~$
```

```
---
repositories:
  - name: helloworld
    url: git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0

releases:
  - name: helloworld
    chart: helloworld/helloworld
    installed: true
```

```

ubuntu@primary:~$ helmfile sync
Adding repo helloworld git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0
"helloworld" has been added to your repositories

Affected releases are:
  helloworld (helloworld/helloworld) UPDATED

Upgrading release=helloworld, chart=helloworld/helloworld
Release "helloworld" does not exist. Installing it now.
NAME: helloworld
LAST DEPLOYED: Mon Jul 17 22:38:18 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
  export NODE_PORT=$(kubectl get --namespace default -o jsonpath=".spec.ports[0].nodePort" services helloworld)
  export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath=".items[0].status.addresses[0].ip")
  echo http://$NODE_IP:$NODE_PORT

Listing releases matching ^helloworld$
helloworld      default      1          2023-07-17 22:38:18.313915226 +0300 EEST
.16.0

UPDATED RELEASES:
NAME      CHART      VERSION
helloworld  helloworld/helloworld  0.1.0

ubuntu@primary:~$ 

```



```

ubuntu@primary:~$ helmfile sync
Adding repo helloworld git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0
"helloworld" has been added to your repositories

Affected releases are:
  helloworld (helloworld/helloworld) UPDATED

Upgrading release=helloworld, chart=helloworld/helloworld
Release "helloworld" does not exist. Installing it now.
NAME: helloworld
LAST DEPLOYED: Mon Jul 17 22:38:18 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
  export NODE_PORT=$(kubectl get --namespace default -o jsonpath=".spec.ports[0].nodePort" services helloworld)
  export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath=".items[0].status.addresses[0].ip")
  echo http://$NODE_IP:$NODE_PORT

Listing releases matching ^helloworld$
helloworld      default      1          2023-07-17 22:38:18.313915226 +0300 EEST
.16.0

UPDATED RELEASES:
NAME      CHART      VERSION
helloworld  helloworld/helloworld  0.1.0

ubuntu@primary:~$ 

```

```

ubuntu@primary:~$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME        NAMESPACE   REVISION   UPDATED             STATUS    CHART
PP VERSION
helloworld  default     1          2023-07-17 22:38:18.313915226 +0300 EEST  deployed  helloworld
.16.0

```

```

repositories:
  - name: helloworld
    url: git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0

releases:
  - name: helloworld
    chart: helloworld/helloworld
    installed: false
  -
  -
  -

```

uninstall

```

ubuntu@primary:~$ helmfile sync
Adding repo helloworld git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0
"helloworld" has been added to your repositories

Listing releases matching ^helloworld$
helloworld      default      1          2023-07-17 22:38:18.313915226 +0300 EEST      deployed      helloworld-0.1
|.16.0

Affected releases are:
helloworld (helloworld/helloworld) DELETED

Deleting helloworld
release "helloworld" uninstalled

DELETED RELEASES:
NAME
helloworld
ubuntu@primary:~$ 

```

5. Deploy multiple Helmcharts using Helmfile

Now we have seen in the previous steps how to deploy a local Helmchart as well as helmchart deployment from repository such as GitHub. In this section we will deploy multiple helmcharts using a Helmfile.

Use the following steps -

1. Create your first helm chart -

```
BASH
1 helm create helloworld1
```

2. Create your second helm chart

```
BASH
1 helm create helloworld2
```

3. Create a Helmfile for deploying multiple helmchart, i.e., helloworld1, helloworld2

The screenshot shows a web browser with multiple tabs open, including a video player, a GitHub repository page, and several search results. The main content area displays a tutorial on the J->Hooq website. It includes a code editor showing a Helmfile snippet and a terminal window showing the command `helmfile sync`.

3. Create a Helmfile for deploying multiple helmchart i.e. helloworld1, helloworld2

```
1 ...
2 releases:
3   - name: helloworld1
4     chart: ./helloworld1
5     installed: true
6   - name: helloworld2
7     chart: ./helloworld2
8     installed: true
9
10
```

4. Install the helmchart using Helmfile

```
1 helmfile sync
```

The screenshot shows a web browser with multiple tabs open, including a video player, a GitHub repository page, and several search results. The main content area displays a tutorial on the J->Hooq website. It includes a terminal window showing the output of the `helmfile sync` command, which details the deployment of two charts.

5. After the successful deployment you should see following message onto your console

```
1 $ helmfile sync
2 Building dependency release=helloworld1, chart=helloworld1
3 Building dependency release=helloworld2, chart=helloworld2
4 Affected releases are:
5   helloworld1 (helloworld1) UPDATED
6   helloworld2 (helloworld2) UPDATED
7
8 Upgrading release=helloworld1, chart=helloworld1
9 Upgrading release=helloworld2, chart=helloworld2
10 Release "helloworld2" does not exist. Installing it now.
11 NAME: helloworld2
12 LAST DEPLOYED: Mon Oct 18 21:05:18 2021
13 NAMESPACE: default
14 STATUS: deployed
15 REVISION: 1
16 NOTES:
```

The same approach can be taken to deploy the multiple helmchart from GitHub repository also. You need to push all of

The screenshot shows a web browser window with multiple tabs open. The active tab is titled "jhooc.com/helmfile-manage-helmchart/". The page content is from the "J->Hooq" website, which has a dark blue header with links for Home, Ansible, Kubernetes, Terraform, YouTube, About, and Contact. Below the header, there is a snippet of YAML code:

```
1 ---  
2 repositories:  
3   - name: helloworld  
4     url: git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0  
5  
6 releases:  
7  
8   - name: helloworld1  
9     chart: ./helloworld1  
10    installed: true  
11  
12   - name: helloworld2  
13     chart: ./helloworld2  
14    installed: true
```

The code defines a repository named "helloworld" with a specific URL and two releases, "helloworld1" and "helloworld2", each with their respective charts and installed status.



Chapter - 6

(Helm repo)

1. How to use helm repo CLI?
2. How to search different charts?

3.1 Search for the 'wordpress' repo

```
1 helm search hub wordpress
```

BASH

After running the above command it should return you with the list of repos available on the Helm Hub.

1 URL	CHART VERSION	APP VERSION	DESCRIPTION
2 https://hub.helm.sh/charts/groundhog2k/wordpress	0.1.3	5.5.1-apache	A Helm chart for Wordpress
3 https://hub.helm.sh/charts/bitnami/wordpress	10.0.3	5.5.3	Web publishing platform for
4 https://hub.helm.sh/charts/seccurecodebox/old-w...	2.1.0	4.0	Insecure & Outdated Wordpre
5 https://hub.helm.sh/charts/fasterbytecharts/wor...	0.8.4	v0.8.4	FasterBytes WordPress Opera
6 https://hub.helm.sh/charts/presslabs/wordpress-...	0.10.5	0.10.5	Presslabs WordPress Operato
7 https://hub.helm.sh/charts/presslabs/wordpress-...	0.10.3	v0.10.3	A Helm chart for deploying
8 https://hub.helm.sh/charts/fasterbytecharts/wor...	0.10.2	v0.10.2	A Helm chart for deploying
9 https://hub.helm.sh/charts/seccurecodebox/wpscan	2.1.0	latest	A Helm chart for the WordPr
10 https://hub.helm.sh/charts/presslabs/stack	0.10.3	v0.10.3	Open-Source WordPress Infra
11 https://hub.helm.sh/charts/fasterbytecharts/stack	0.10.2	v0.10.2	Open-Source WordPress Infra

BASH

If you look carefully at the results then we are interested in <https://hub.helm.sh/charts/bitnami/wordpress>.

In case if the URL is too long to see then you can put `--max-col-width=0`, so that you can view the complete URL

```
1 helm search hub wordpress --max-col-width=0
```

≡

BASH

```

is is my wordpress package

https://artifacthub.io/packages/helm/bitnami-aks/wordpress           15.2.13   6.1.0   W
WordPress is the world's most popular blogging and content management platform. Powerful yet simple, everyone from students to global corporations use it to build beautiful, functional websites.
https://artifacthub.io/packages/helm/shubham-wordpress/wordpress      0.1.0    1.16.0   A
Helm chart for Kubernetes

https://artifacthub.io/packages/helm/bitnami/wordpress                16.1.27   6.2.2   W
WordPress is the world's most popular blogging and content management platform. Powerful yet simple, everyone from students to global corporations use it to build beautiful, functional websites.
https://artifacthub.io/packages/helm/truecharts/wordpress               2.0.39   6.2.2   T
The WordPress rich content management system can utilize plugins, widgets, and themes.

```

```

ubuntu@primary:~$ helm repo list
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME          URL
helloworld    git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0
ubuntu@primary:~$ 

```

Connect to Wi-Fi | Complete Helm Chart Tutorial: | How to use Helmfile for managing multiple charts: | Helm chart - Wordpress Installation: | +

jhoqq.com/helm-chart-wordpress-installation/

J->Hooq Home Ansible Kubernetes Terraform YouTube About repos 0/0

3.2 Add 'bitnami/wordpress' to your repo list of Helm Chart

After knowing the `repo url` now you can add it to your local Helm Chart repo list.

But before adding the `bitnami/wordpress` first check whether it already exists on your repo list or not?

```

1 helm repo list

```

If you haven't added the `bitnami/wordpress` before then it should not show in the list.

Alright, let us add it to your repo list -

```

1 helm repo add bitnami https://charts.bitnami.com/bitnami

```

Once you add it successfully then you should see the following message.

Windows Taskbar: Type here to search, Start button, File Explorer, Google Chrome, Microsoft Edge, Visual Studio Code, File Explorer, Task View, Taskbar icons, Weather (19°C), Clear, Battery (80%), ENG, 05:29, 09-02-2025

```

ubuntu@primary:~$ helm repo add bitnami https://charts.bitnami.com/bitnami
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
"bitnami" has been added to your repositories
ubuntu@primary:~$ 
ubuntu@primary:~$ helm repo list
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME          URL
helloworld    git+https://github.com/rahulwagh/helmchart@helloworld?ref=master&sparse=0
bitnami       https://charts.bitnami.com/bitnami
ubuntu@primary:~$ 

```

The screenshot shows a Microsoft Edge browser window with the following details:

- Address Bar:** jhoq.com/helm-chart-wordpress-installation/
- Page Content:**
 - J-Hooq Header:** Home, Ansible, Kubernetes, Terraform, YouTube, About, repos
 - Section:** 3.4 Readme and Values
 - Text:** There are a few more details which are provided along with the helm chart package.
 - Link:** [Readme.md](#)
 - Text:** This Readme.md contains the installation instructions and it can be viewed using the following command
 - Terminal Mockup:** 1 helm show readme bitnami/wordpress --version 10.0.3
 - Section:** Values
 - Text:** If you are familiar with WordPress before then you need `username` and `password` to access the WordPress CMS, so you can view the default values
 - Terminal Mockup:** 1 helm show values bitnami/wordpress --version 10.0.3
 - Note:** Note - Here you will get a long list of values but you can skip this part because we are going to set up the username and password in the next step.
- System Tray:** Shows the Windows Start button, a search bar, pinned icons for File Explorer, Task View, and others, and system status including battery level, temperature (19°C), and date/time (05:31, 09-02-2025).

```

### To 11.0.0

The [Bitnami WordPress](https://github.com/bitnami/containers/tree/main/bitnami/wordpress) image was refactored and
now the source code is published in GitHub in the `rootfs` folder of the container image.

Compatibility is not guaranteed due to the amount of involved changes, however no breaking changes are expected.

### To 10.0.0

[On November 13, 2020, Helm v2 support was formally finished](https://github.com/helm/charts#status-of-the-project),
this major version is the result of the required changes applied to the Helm Chart to be able to incorporate the different features added in Helm v3 and to be consistent with the Helm project itself regarding the Helm v2 EOL.

[Learn more about this change and related upgrade considerations](https://docs.bitnami.com/kubernetes/apps/wordpress-administration/upgrade-helm3/).

```



Chapter - 7

(Hooks & Test)

1. What is Helm Hooks ?
2. When to use the Helm Hooks?
3. Demo - Helm Hooks?
4. What is Helm Test?
5. How to use Helm Test?



1. What is Helm Hooks?



Helm hooks can be any k8s resource?

```
apiVersion: batch/v1
kind: Job
metadata:
  name: pi
spec:
  template:
    spec:
      containers:
        - name: pi
          image: perl:5.34.0
```

Batch

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  selector:
    matchLabels:
```

Deployment

```
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app.kubernetes.io/name: MyApp
  ports:
    - protocol: TCP
      port: 80
```

Service

```
$ ls
helloworld
$ tree helloworld/
helloworld/
├── Chart.yaml
├── charts
└── templates
    ├── NOTES.txt
    ├── _helpers.tpl
    ├── deployment.yaml
    ├── hooks
    │   └── pre-install.yaml
    ├── hpa.yaml
    ├── ingress.yaml
    ├── service.yaml
    ├── serviceaccount.yaml
    └── tests
        └── test-connection.yaml
values.yaml

4 directories, 11 files
$
```

```

apiVersion: batch/v1
kind: Job
metadata:
  name: "{{ include "helloworld.fullname" . }}-pre-install-job-hook"
  labels:
    {{- include "helloworld.labels" . | indent 4 }}
  annotations:
    "helm.sh/hook": "pre-install"
    "helm.sh/hook-weight": "0"
    "helm.sh/hook-delete-policy": hook-succeeded
spec:
  template:
    spec:
      containers:
        - name: pre-install
          image: busybox
          imagePullPolicy: IfNotPresent
          command: ['sh', '-c', 'echo pre-install Pod is Running ; sleep 10']
          restartPolicy: OnFailure
          terminationGracePeriodSeconds: 0
  backoffLimit: 3
  completions: 1
  parallelism: 1

```

```

ubuntu@democratic-sole: ~ (multipass)
Every 2.0s: kubectl get pod
              I
NAME           READY   STATUS    RESTARTS   AGE
myhelloworld-release-helloworld-test-connection   0/1     Completed   0          24h

```

```

$ helm install myhelloworld-release helloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config

```

NAME	READY	STATUS	RESTARTS	AGE
myhelloworld-release-helloworld-pre-install-job-hook	1/1	Running	0	7s

```
$ helm install myhelloworld helloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME: myhelloworld
LAST DEPLOYED: Fri Jul 21 23:02:51 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
  export NODE_PORT=$(kubectl get --namespace default -o jsonpath=".spec.ports[0].nodePort" services myhelloworld-hello-world)
  export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath=".items[0].status.addresses[0].address")
  echo http://$NODE_IP:$NODE_PORT
$ 
```

```
ubuntu@democratic-sole: ~ (multipass)
Every 2.0s: kubectl get pod
                                         den

NAME                      READY   STATUS    RESTARTS   AGE
myhelloworld-helloworld-6f5c8c9fd6-t6sml   1/1     Running   0          6s
```

```
$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME        NAMESPACE      REVISION      UPDATED             STATUS      CHART
myhelloworld  default       1           2023-07-21 23:02:51.836700922 +0300 EEST  deployed  helloworld-0.1.0
.16.0
$ 
```

```
$ helm uninstall myhelloworld
WARNING: Kubernetes configuration file is group-re
WARNING: Kubernetes configuration file is world-re
release "myhelloworld" uninstalled
$ 
```

```
ubuntu@democratic-sole: ~ (multipass)
Every 2.0s: kubectl get pod
No resources found in default namespace.
```

```
$ tree helloworld
helloworld
├── Chart.yaml
├── charts
└── templates
    ├── NOTES.txt
    ├── _helpers.tpl
    ├── deployment.yaml
    ├── hooks
    │   └── pre-install.yaml
    ├── hpa.yaml
    ├── ingress.yaml
    ├── service.yaml
    ├── serviceaccount.yaml
    └── tests
        └── test-connection.yaml
└── values.yaml

4 directories, 11 files
$
```

```
$ cat helloworld/templates/tests/test-connection.yaml
apiVersion: v1
kind: Pod
metadata:
  name: "{{ include "helloworld.fullname" . }}-test-connection"
  labels:
    {{- include "helloworld.labels" . | indent 4 }}
  annotations:
    "helm.sh/hook": test
spec:
  containers:
    - name: wget
      image: busybox
      command: ['wget']
      args: ['{{ include "helloworld.fullname" . }}:{{ .Values.service.port }}']
  restartPolicy: Never
$
```

```
$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME    NAMESPACE      REVISION      UPDATED STATUS  CHART   APP VERSION
$
```

```
$ helm install myhelloworld helloworld
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME: myhelloworld
LAST DEPLOYED: Fri Jul 21 23:12:01 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
NOTES:
1. Get the application URL by running these commands:
  export NODE_PORT=$(kubectl get --namespace default -o jsonpath=".spec.ports[0].nodePort" services myhelloworld-helloworld)
  export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath=".items[0].status.addresses[0].address")
  echo http://$NODE_IP:$NODE_PORT
$
```

```
$ kubectl get service
NAME           TYPE      CLUSTER-IP   EXTERNAL-IP  PORT(S)        AGE
kubernetes     ClusterIP  10.152.183.1 <none>       443/TCP       25h
myhelloworld-helloworld  NodePort   10.152.183.217 <none>       80:31836/TCP  23s
$
```

```
$ hostname -I
192.168.64.7 10.1.242.64 fd16:8590:7284:4e59:5054:ff:fe59:58f5
$
```

```
① 192.168.64.7:31836
```

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

```
$ helm list -a
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME      NAMESPACE   REVISION   UPDATED             STATUS    CHART
PP VERSION
myhelloworldrelease  default     1          2023-07-21 23:12:01.304746807 +0300 EEST    deployed   helloworld-16.0
$ 
$ helm test myhelloworldrelease
WARNING: Kubernetes configuration file is group-readable. This is insecure. Location: /home/ubuntu/.kube/config
WARNING: Kubernetes configuration file is world-readable. This is insecure. Location: /home/ubuntu/.kube/config
NAME: myhelloworldrelease
LAST DEPLOYED: Fri Jul 21 23:12:01 2023
NAMESPACE: default
STATUS: deployed
REVISION: 1
TEST SUITE: myhelloworldrelease-helloworld-test-connection
Last Started: Fri Jul 21 23:14:19 2023
Last Completed: Fri Jul 21 23:14:22 2023
Phase: Succeeded
NOTES:
1. Get the application URL by running these commands:
  export NODE_PORT=$(kubectl get --namespace default -o jsonpath="{.spec.ports[0].nodePort}")
  export NODE_IP=$(kubectl get nodes --namespace default -o jsonpath=".items[0].status.addresses[0].ip")
  echo http://$NODE_IP:$NODE_PORT
$
```