Final Report

Submitted to Dr.varsha

Lovely Professional University, Phagwara, Punjab.

Submitted By

Name : Sheikh Sajib Alom

Reg No: 12112802

Roll No: 29

Subject: INT334:ENTERPRISE APPLICATION AUTOMATION

Install Kubernetes

Prerequisites

- An Ubuntu 24.04 LTS system.
- Privileged access to the system (root or sudo user).
- Active internet connection.
- Minimum 2GB RAM or more.
- Minimum 2 CPU cores (or 2 vCPUs).

Note: You can use t3.small to fulfill all the requirements.

Step 1: Update and Upgrade Ubuntu (all nodes)

Begin by ensuring that your system is up to date. Open a terminal and execute the following commands:

```
sudo apt update && sudo apt upgrade -y
```

Step 2: Disable Swap (all nodes)

To enhance Kubernetes performance, disable swap and set essential kernel parameters. Run the following commands on all nodes to disable all swaps:

```
sudo swapoff -a
sudo sed -i '/ swap / s/^\(.*\)$/#\1/g' /etc/fstab
```

Step 3: Add Kernel Parameters (all nodes)

Load the required kernel modules on all nodes:

```
sudo tee /etc/modules-load.d/containerd.conf <<EOF
overlay
br_netfilter
EOF
sudo modprobe overlay
sudo modprobe br_netfilter
```

Configure the critical kernel parameters for Kubernetes using the following to all nodes:

```
sudo tee /etc/sysctl.d/kubernetes.conf <<EOF
net.bridge.bridge-nf-call-ip6tables = 1
net.bridge.bridge-nf-call-iptables = 1
net.ipv4.ip_forward = 1
EOF
```

Then, reload the changes:

sudo sysctl --system

Step 4: Install Containerd Runtime (all nodes)

We are using the containerd runtime. Install containerd and its dependencies with the following commands:
sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates

Enable the Docker repository (all nodes):
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmour -o /etc/apt/trusted.gpg.d/docker.gpg
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"

Update the package list and install containerd (all nodes):
sudo apt update
sudo apt install -y containerd.io

Configure containerd to start using systemd as cgroup (all nodes):
containerd config default | sudo tee /etc/containerd/config.toml >/dev/null 2>&1
sudo sed -i 's/SystemdCgroup = false/SystemdCgroup = true/g' /etc/containerd/config.toml

Restart and enable the containerd service (all nodes):
sudo systemctl restart containerd
sudo systemctl enable containerd

Step 5: Add Apt Repository for Kubernetes (all nodes)

Kubernetes packages are not available in the default Ubuntu 22.04 repositories. Add the Kubernetes repositories with the following commands:
echo "deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.30/deb/ /" | sudo tee /etc/apt/sources.list.d/kubernetes.list
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

Step 6: Install Kubectl, Kubeadm, and Kubelet (all nodes)

After adding the repositories, install essential Kubernetes components, including kubectl, kubelet, and kubeadm, on all nodes with the following commands:
sudo apt update
sudo apt install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl

Step 7: Initialize Kubernetes Cluster with Kubeadm (master node)

With all the prerequisites in place, initialize the Kubernetes cluster on the master node using the following Kubeadm command:
sudo kubeadm init

After the initialization is complete, make a note of the kubeadm join command for future reference. Run the following commands on the master node:
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

Next, use kubectl commands to check the cluster and node status (all nodes):
kubectl get nodes

```
root@master:~# kubectl get nodes
NAME      STATUS   ROLES           AGE     VERSION
master    Ready    control-plane   2m47s   v1.30.3
```

Step 8: Add Worker Nodes to the Cluster (worker nodes)

On each worker node, use the kubeadm join command you noted down earlier (you get the join command from last output of master):
kubeadm join 138.197.184.45:6443 --token 72ww2b.6orffywqcf5s4p2z \
--discovery-token-ca-cert-hash
sha256:aafb79cdd45a6e3b3fac01fb3efba0817360b01f90a4b6c3f11567108a36ba67

```
root@worker:~# kubeadm join 146.190.135.66:6443 --token f1h951.u4nkex9cwdd0g63w        --discovery-token-ca-cert-has
h sha256:6d15f2a790db38d1666af58c85f060b9fadc73f13c932e8e2a9eeef88f51f91a
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o yaml'
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

Step 9: Install Kubernetes Network Plugin (master node)

To enable communication between pods in the cluster, you need a network plugin. Install the Calico network plugin with the following command from the master node:
kubectl apply -f

https://raw.githubusercontent.com/projectcalico/calico/v3.25.0/manifests/calico.yaml

Step 10: Verify the Cluster and Test (master node)

Finally, we want to verify whether our cluster is successfully created.
kubectl get pods -n kube-system
kubectl get nodes

```
root@master:~# kubectl get nodes
NAME     STATUS   ROLES           AGE   VERSION
master   Ready    control-plane   17m   v1.30.3
worker   Ready    <none>          65s   v1.30.3
root@master:~# kubectl get pods -n kube-system
NAME                                       READY   STATUS    RESTARTS   AGE
calico-kube-controllers-5b9b456c66-dkvq5   1/1     Running   0          15m
calico-node-q8x86                          1/1     Running   0          68s
calico-node-rtllk                          1/1     Running   0          15m
coredns-7db6d8ff4d-ctdfh                   1/1     Running   0          17m
coredns-7db6d8ff4d-m6wxt                   1/1     Running   0          17m
etcd-master                                1/1     Running   0          17m
kube-apiserver-master                      1/1     Running   0          17m
kube-controller-manager-master             1/1     Running   0          17m
kube-proxy-756dt                           1/1     Running   0          17m
kube-proxy-qjh4n                           1/1     Running   0          68s
kube-scheduler-master                      1/1     Running   0          17m
root@master:~#
```

Install Minikube

Run the following commands to install Minikube:
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube && rm minikube-linux-amd64
minikube start
minikube status

```
root@LAPTOP-H6JIHC55:~# minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

root@LAPTOP-H6JIHC55:~#
```

Verify Kubernetes Cluster

Run the following commands to verify the cluster:
kubectl get nodes

```
root@LAPTOP-H6JIHC55:~# kubectl get nodes
NAME       STATUS   ROLES           AGE     VERSION
minikube   Ready    control-plane   5m19s   v1.31.0
root@LAPTOP-H6JIHC55:~#
```

kubectl get pods --namespace=kube-system

```
root@LAPTOP-H6JIHC55:~# kubectl get pods --namespace=kube-system
NAME                               READY   STATUS    RESTARTS        AGE
coredns-6f6b679f8f-659qz           1/1     Running   0               5m37s
etcd-minikube                      1/1     Running   0               5m42s
kube-apiserver-minikube            1/1     Running   0               5m42s
kube-controller-manager-minikube   1/1     Running   0               5m42s
kube-proxy-nr577                   1/1     Running   0               5m37s
kube-scheduler-minikube            1/1     Running   0               5m42s
storage-provisioner                1/1     Running   1 (5m4s ago)    5m40s
root@LAPTOP-H6JIHC55:~#
```

Kubernetes Deployment

Create a Deployment using the following command:
kubectl create deployment int334file --image=nginx

```
root@LAPTOP-H6JIHC55: ~
root@LAPTOP-H6JIHC55:~# kubectl create deployment int334file --image=nginx
deployment.apps/int334file created
root@LAPTOP-H6JIHC55:~#
```

kubectl expose deployment int334file --type=LoadBalancer --port=80 --target-port=80

```
root@LAPTOP-H6JIHC55:~# kubectl expose deployment int334file --type=LoadBalancer --port=80 --target-port=80
service/int334file exposed
root@LAPTOP-H6JIHC55:~#
```

View Deployments and update the image:
kubectl get deployments --namespace=kube-system

```
root@LAPTOP-H6JIHC55:~# kubectl get deployments --namespace=kube-system
NAME      READY   UP-TO-DATE   AVAILABLE   AGE
coredns   1/1     1            1           85m
root@LAPTOP-H6JIHC55:~#
```

kubectl set image deployment/int334file nginx=nginx:latest

```
root@LAPTOP-H6JIHC55: ~
root@LAPTOP-H6JIHC55:~# vim deployment.yaml
root@LAPTOP-H6JIHC55:~#
```

Deploy Using YAML: Save as deployment.yaml:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-depo
spec:
  replicas: 2
  selector:
    matchLabels:
      app: hello-depo
  template:
    metadata:
      labels:
        app: hello-depo
    spec:
      containers:
      - name: hello-depo
        image: nginx
        ports:
        - containerPort: 80
```

Apply the deployment:
kubectl apply -f deployment.yaml

```
root@LAPTOP-H6JIHC55:~# kubectl apply -f deployment.yaml
deployment.apps/hello-depo created
root@LAPTOP-H6JIHC55:~#
```

kubectl expose deployment hello-depo --type=ClusterIP --port=80 --target-port=80
kubectl get services

```
root@LAPTOP-H6JIHC55:~# kubectl get replicasets
NAME                    DESIRED   CURRENT   READY   AGE
hello-depo-d556bf558    3         3         3       60s
int334file-564488665f   1         1         1       4m15s
int334file-7d97c8c486   0         0         0       7m32s
root@LAPTOP-H6JIHC55:~#
```

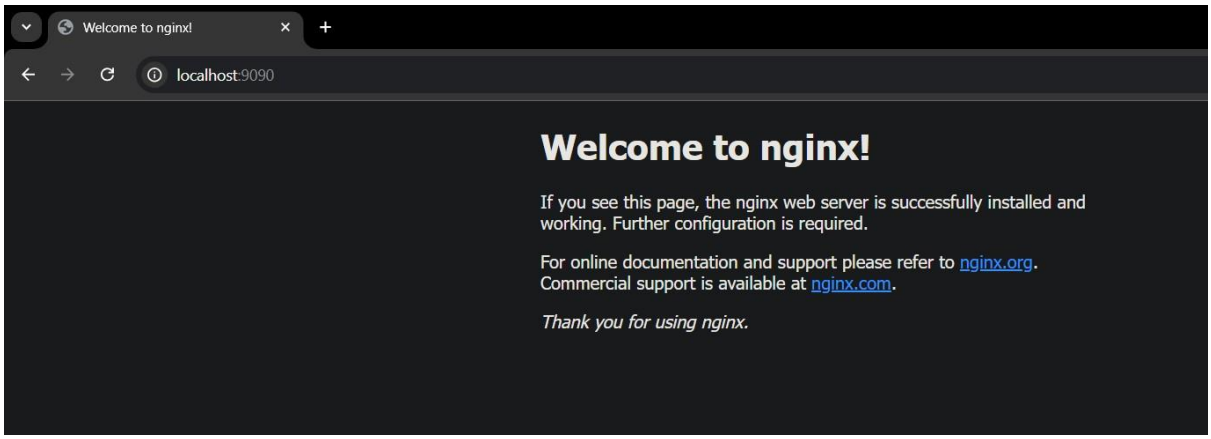
Kubernetes Services

Create a Service YAML for Nginx Deployment, save as nginx-deployment.yaml and apply:
kubectl apply -f nginx-deployment.yaml

```
root@LAPTOP-H6JIHC55:~# kubectl get deployments
NAME               READY   UP-TO-DATE   AVAILABLE   AGE
hello-depo         3/3     3            3           4h52m
int334file         1/1     1            1           4h59m
nginx-deployment   1/2     2            1           13s
```

kubectl port-forward service/nginx-service 9090:80

```
root@LAPTOP-H6JIHC55:~# kubectl port-forward service/nginx-service 9090:80
Forwarding from 127.0.0.1:9090 -> 80
Forwarding from [::1]:9090 -> 80
Handling connection for 9090
```



Kubernetes Ingress

Enable the Ingress Add-on:
minikube addons enable ingress

```
PS C:\WINDOWS\system32> minikube addons enable ingress
* ingress is an addon maintained by Kubernetes. For any concerns contact minikube on GitHub.
You can view the list of minikube maintainers at: https://github.com/kubernetes/minikube/blob/master/OWNERS
* After the addon is enabled, please run "minikube tunnel" and your ingress resources would be available at "127.0.0.1"
  - Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.3
  - Using image registry.k8s.io/ingress-nginx/kube-webhook-certgen:v1.4.3
  - Using image registry.k8s.io/ingress-nginx/controller:v1.11.2
* Verifying ingress addon...
```

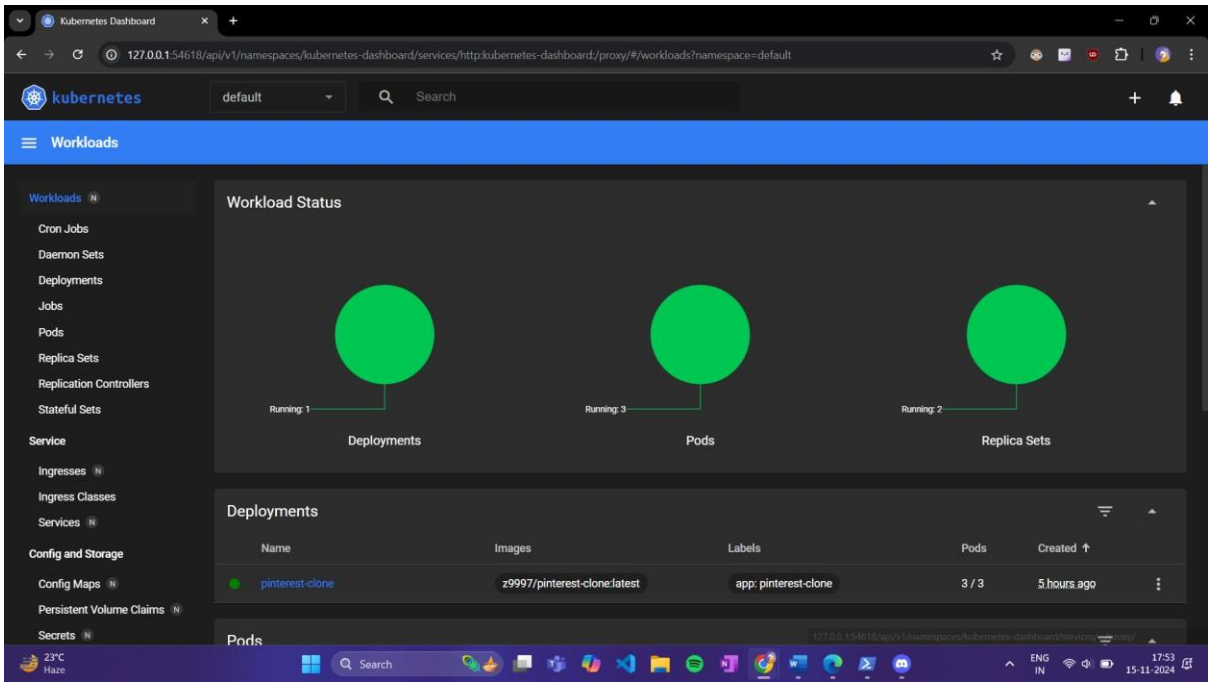
Install NGINX Ingress Controller:
kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/main/deploy/static/provider/cloud/deploy.yaml
kubectl expose deployment ingress-nginx-controller --port=80 --target-port=80 --name=nginx-ingress-service

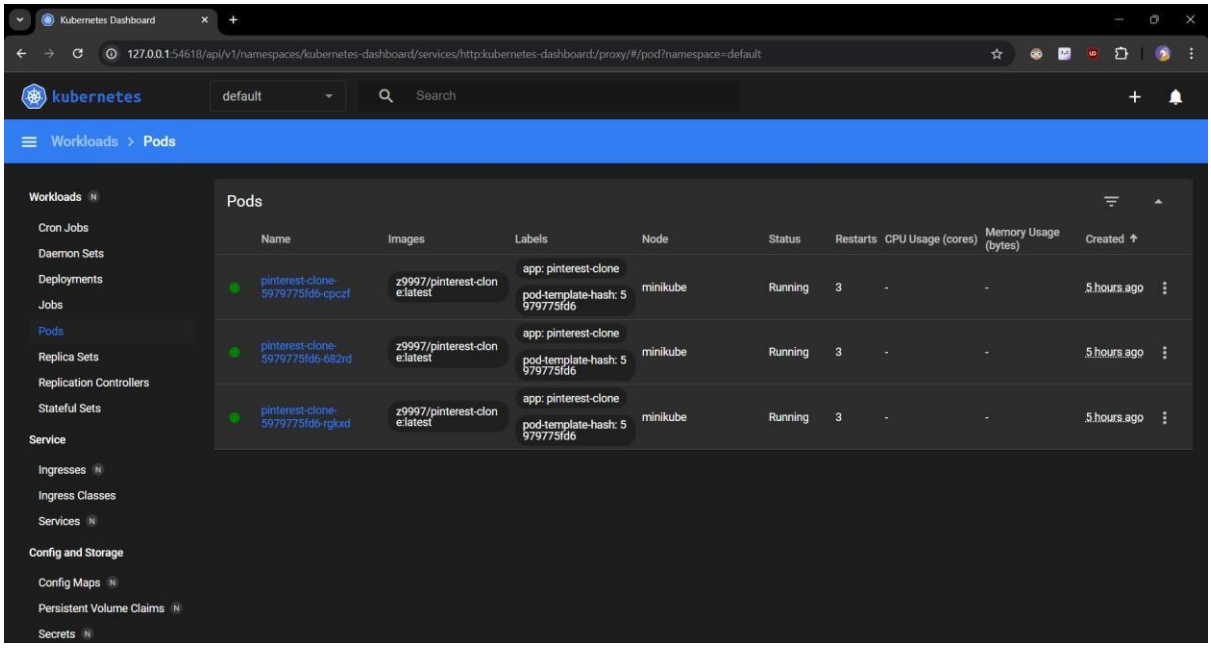
Create an Ingress Rule Using YAML, save as ingress.yaml and apply:
kubectl apply -f ingress.yaml
kubectl get svc -n ingress-nginx

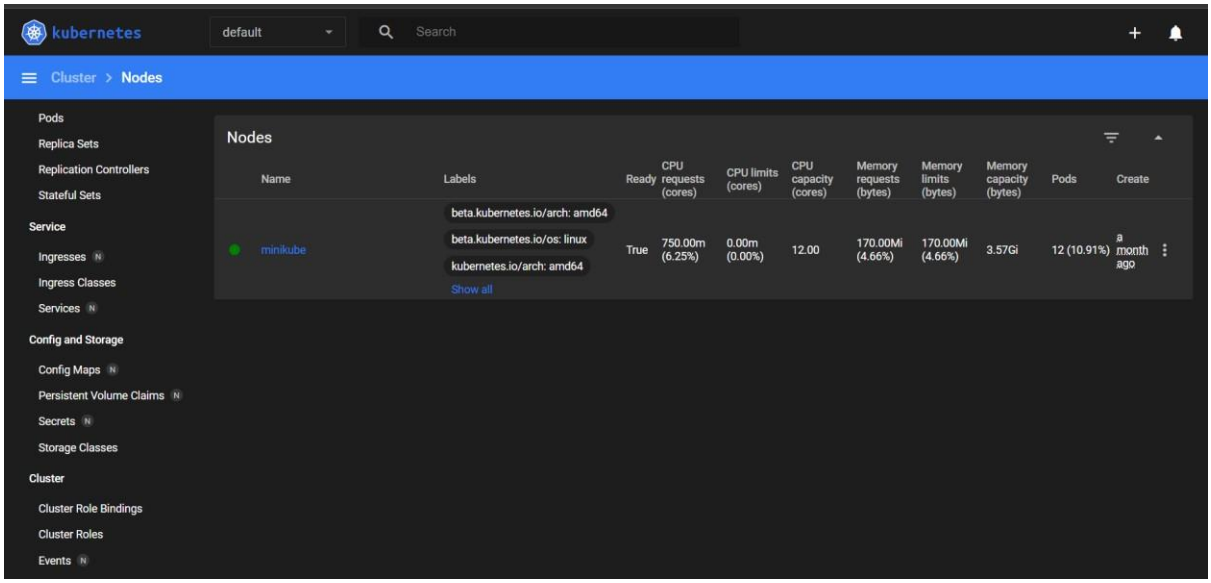
Kubernetes UI

Enable the Dashboard:
minikube start
minikube addons enable dashboard
minikube dashboard

```
PS C:\WINDOWS\system32> minikube start
* minikube v1.34.0 on Microsoft Windows 11 Home Single Language 10.0.22631.4460 Build 22631.4460
* Using the docker driver based on existing profile
* Starting "minikube" primary control-plane node in "minikube" cluster
* Pulling base image v0.0.45 ...
* Restarting existing docker container for "minikube" ...
! Failing to connect to https://registry.k8s.io/ from inside the minikube container
* To pull new external images, you may need to configure a proxy: https://minikube.sigs.k8s.io/docs/reference/networking/proxy/
* Preparing Kubernetes v1.31.0 on Docker 27.2.0 ...
* Verifying Kubernetes components...
  - Using image gcr.io/k8s-minikube/storage-provisioner:v5
  - Using image docker.io/kubernetesui/dashboard:v2.7.0
  - Using image docker.io/kubernetesui/metrics-scraper:v1.0.8
* Some dashboard features require the metrics-server addon. To enable all features please run:

        minikube addons enable metrics-server

* Enabled addons: storage-provisioner, dashboard, default-storageclass
* Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
PS C:\WINDOWS\system32>
```







Splunk Installation

Download and Install Splunk:
sudo su
cd /opt/
apt update
wget -O splunk-9.2.1-78803f08aabb-Linux-x86_64.tgz
"https://download.splunk.com/products/splunk/releases/9.2.1/linux/splunk-9.2.1-78803f08aabb-Linux-x86_64.tgz"
tar -xvf splunk-9.2.1-78803f08aabb-Linux-x86_64.tgz
chmod 777 splunk

cd splunk/bin

./splunk start --accept-license

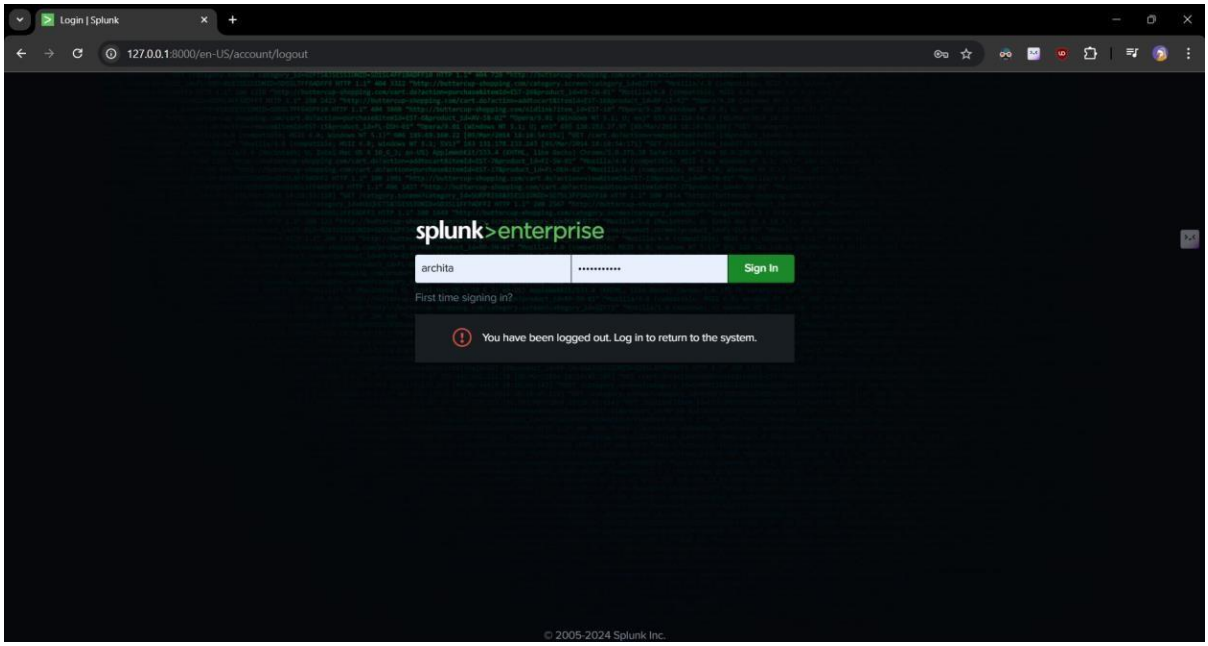
```
root@LAPTOP-H6JIHC55:~# ls
my-web-service  my_web-service  nginx-scale-demo  scale-demo  splunk  splunk-9.2.1-78803f08aabb-Linux-x86_64.tgz
root@LAPTOP-H6JIHC55:~# chmod 777 splunk
root@LAPTOP-H6JIHC55:~# cd splunk/bin
root@LAPTOP-H6JIHC55:~/splunk/bin# ls
2to3-3.7                    idle3                          priforgepng                    setSplunkEnv
ColdStorageArchiver.py      idle3.7                        prigreypng                     shc_upgrade_template.py
ColdStorageArchiver_GCP.py  importtool                     pripalpng                      signtool
S3benchmark                 installit.py                   pripamtopng                    slim
bloom                       jars                           pripnglsch                     spl-lang-server-sockets
bottle.py                   jsmin                          pripngtopam                    spl2-orchestrator
btool                       locktest                       priweavepng                    splunk
btprobe                     locktool                       pydoc3                         splunk-optimize
bzip2                       mongod                         pydoc3.7                       splunk-optimize-lex
classify                    mongod-3.6                     python                         splunk-tlsd
coldToFrozenExample.py      mongod-4.0                     python3                        splunkd
compsup                     mongodump                      python3.7                      splunkmon
copyright.txt               mongorestore                   python3.7m                     supervisor-simulator
dbmanipulator.py            noah_self_storage_archiver.py  pyvenv                         tarit.py
easy_install-3.7            node                           pyvenv-3.7                     tocsv.py
exporttool                  openssl                        rapidDiag                      tsidx_scan.py
fill_summary_index.py       parse_xml_buckets.py           recover-metadata               tsidxprobe
genAuditKeys.py             pcre2-config                   rest_handler.py                tsidxprobe_plo
genRootCA.sh                pcregextest                    runScript.py                   untarit.py
genSignedServerCert.py      pid_check.sh                   safe_restart_cluster_master.py walklex
genSignedServerCert.sh      pip3                           scripts                        wheel
genWebCert.py               pip3.7                         scrubber.py
genWebCert.sh               prichunkpng                    searchtest
```

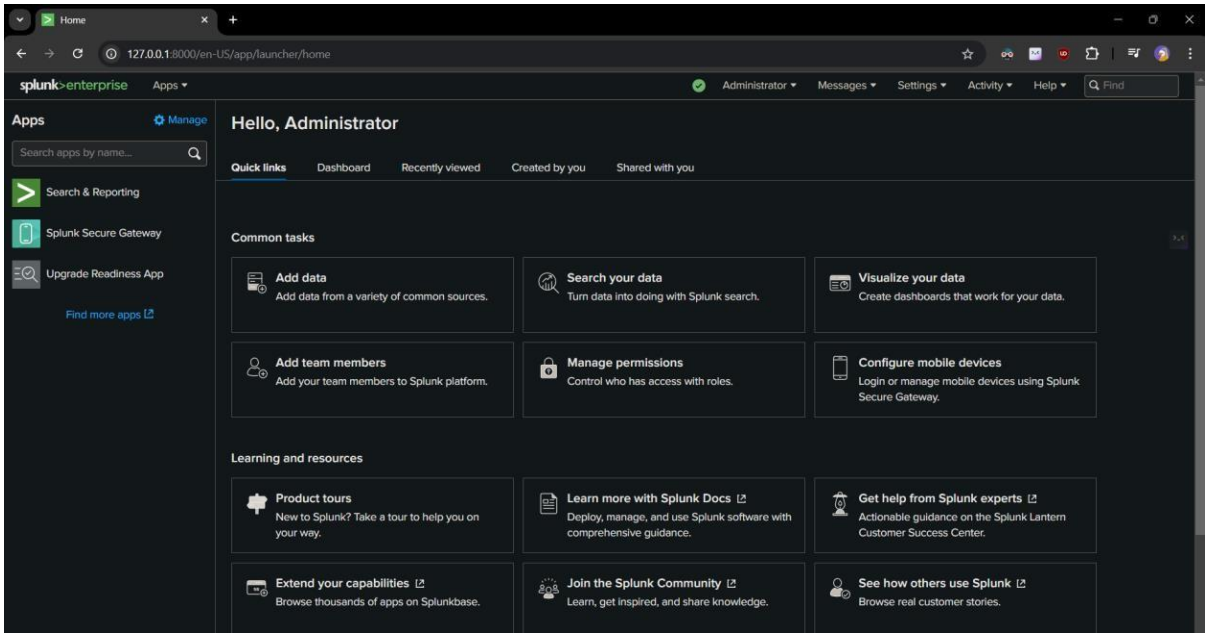
```
root@LAPTOP-H6JIHC55:~/splunk/bin# ./splunk start --accept-license

This appears to be your first time running this version of Splunk.

Splunk software must create an administrator account during startup. Otherwise, you cannot log in.
Create credentials for the administrator account.
Characters do not appear on the screen when you type in credentials.

Please enter an administrator username: archita
Password must contain at least:
   * 8 total printable ASCII character(s).
Please enter a new password:
Please confirm new password:
Copying '/root/splunk/etc/openldap/ldap.conf.default' to '/root/splunk/etc/openldap/ldap.conf'.
Generating RSA private key, 2048 bit long modulus
.........................+++++
.........................+++++
e is 65537 (0x10001)
writing RSA key

Generating RSA private key, 2048 bit long modulus
.................................................................+++++
..................+++++
e is 65537 (0x10001)
writing RSA key

Moving '/root/splunk/share/splunk/search_mrsparkle/modules.new' to '/root/splunk/share/splunk/search_mrsparkle/modules'.

Splunk> See your world.  Maybe wish you hadn't.

Checking prerequisites...
        Checking http port [8000]: open
        Checking mgmt port [8089]: open
```

```
Waiting for web server at http://127.0.0.1:8000 to be available..................... Done


If you get stuck, we're here to help.
Look for answers here: http://docs.splunk.com

The Splunk web interface is at http://LAPTOP-H6JIHC55:8000

root@LAPTOP-H6JIHC55:~/splunk/bin#
```

Access the Splunk Web Interface:

http://127.0.0.1:8000


Login | Splunk
127.0.0.1:8000/en-US/account/logout
splunk>enterprise
archita
Sign In
First time signing in?
You have been logged out. Log in to return to the system.
© 2005-2024 Splunk Inc.


Home
127.0.0.1:8000/en-US/app/launcher/home
splunk>enterprise
Apps
Administrator
Messages
Settings
Activity
Help
Find
Apps
Manage
Search apps by name...
Search & Reporting
Splunk Secure Gateway
Upgrade Readiness App
Find more apps
Hello, Administrator
Quick links
Dashboard
Recently viewed
Created by you
Shared with you
Common tasks
Add data
Add data from a variety of common sources.
Search your data
Turn data into doing with Splunk search.
Visualize your data
Create dashboards that work for your data.
Add team members
Add your team members to Splunk platform.
Manage permissions
Control who has access with roles.
Configure mobile devices
Login or manage mobile devices using Splunk Secure Gateway.
Learning and resources
Product tours
New to Splunk? Take a tour to help you on your way.
Learn more with Splunk Docs
Deploy, manage, and use Splunk software with comprehensive guidance.
Get help from Splunk experts
Actionable guidance on the Splunk Lantern Customer Success Center.
Extend your capabilities
Browse thousands of apps on Splunkbase.
Join the Splunk Community
Learn, get inspired, and share knowledge.
See how others use Splunk
Browse real customer stories.