Experiment: Working with k3d and Rancher

Create our cluster for this experiment. We'll be working with the Rancher configuration for k3d. Rancher Labs was purchased by Suse this year to continue to provide solutions for IoT and Edge usages for Kubernetes in cloud-native application development and modernization.

$> **k3d cluster create k3d-rancher --api-port 6550 --servers 1 --agents 3 --port 443:443@loadbalancer --wait**

[36mINFO[0m[0000] Created network 'k3d-k3d-rancher'
[36mINFO[0m[0000] Created volume 'k3d-k3d-rancher-images'
[36mINFO[0m[0001] Creating node 'k3d-k3d-rancher-server-0'
[36mINFO[0m[0001] Creating node 'k3d-k3d-rancher-agent-0'
[36mINFO[0m[0001] Creating node 'k3d-k3d-rancher-agent-1'
[36mINFO[0m[0005] Creating node 'k3d-k3d-rancher-agent-2'
[36mINFO[0m[0005] Creating LoadBalancer 'k3d-k3d-rancher-serverlb'
[36mINFO[0m[0013] Cluster 'k3d-rancher' created successfully!
[36mINFO[0m[0014] You can now use it like this:
kubectl cluster-info

This folder should have been created in our k3d getting started lab, but just to be sure

$> **mkdir~/.kube**

On Windows:

set KUBECONFIG_FILE=C:\k3d\.kube\k3d-rancher

k3d kubeconfig get k3d-rancher > %KUBECONFIG_FILE%

set KUBECONFIG=%KUBECONFIG_FILE%

On MacOS or Linux

~/k3d/.kube $ **export KUBECONFIG_FILE=~/.kube/k3d-rancher**

~/k3d/.kube $ **k3d kubeconfig get k3d-rancher > $KUBECONFIG_FILE**

~/k3d/.kube $ **export KUBECONFIG=$KUBECONFIG_FILE**

On Windows:

set | grep KUBE

KUBECONFIG_FILE=C:\k3d\.kube\k3d-rancher

On MacOS:

set | grep KUBE

KUBECONFIG_FILE=~/.kube/k3d-rancher

On Both MacOS and Windows:

kubectl cluster-info

Kubernetes master is running at https://0.0.0.0:6550
CoreDNS is running at https://0.0.0.0:6550/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
Metrics-server is running at https://0.0.0.0:6550/api/v1/namespaces/kube-system/services/https:metrics-server:/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.

k3d cluster list

NAME	SERVERS	AGENTS	LOADBALANCER
k3d-rancher	1/1	3/3	true

For Windows:

type %KUBECONFIG_FILE%

For MacOS:

echo $KUBECONFIG_FILE

---

apiVersion: v1
clusters:
- cluster:
    certificate-authority-data:
LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUJWekNCL3FBREFnRUNBZ0VBTUFvR
0NDcUdTTTQ5QkFNQ01DTXhJVEFmQmdOVkJBTU1HR3N6Y3kxelpYSjIKWlhJdFkyRkFNVF
U1T1RZM01qUTRPVEFlRncweU1EQTVNRGt4TnpJNE1EbGFGdzB6TURBNU1EY3hOekk0TU
RsYQpNQ014SVRBZkJnTlZCQU1NR0dzemN5MXpaWEoyWlhJdFkyRkFNVFU1T1RZM01qUT
RPVEJaTUJNR0J5cUdTTTQ5CkFnRUdDQ3FHU000OUF3RUhBMElBQkdmRm53RUtycFVtbV
h3ckVFUFdaYSsxZWdYQWhPV2ZUZEorZU94UWo4U3kKUDgzSTJQbDYrTUQ4OUNMTlRTb
E1Ebk5pM3FvS1N0ZHdGZFRhOFRHQUxTS2pJekFoTUE0R0ExVWREd0VCL3dRRQpBd0lDc
ERBUEJnTlZIUk1CQWY4RUJUQURBUUgvTUFvR0NDcUdTTTQ5QkFNQ0EwZ0FNRVVDSUF
VOGpaQ0RORkhMCkpDVkdOd2I2UXhxS0xPekp1NUtYV2JNdGZ0VVB4Ymc4QWlFQXNkQXF

```
JRm90R2JPcVk4OUxudU45eStrTU44M1AKU1pPWWRGMEIyNUV2dXgwPQotLS0tLUVORCB
DRVJUSUZJQ0FURS0tLS0tCg==
    server: https://0.0.0.0:6550
  name: k3d-k3d-rancher
contexts:
- context:
    cluster: k3d-k3d-rancher
    user: admin@k3d-k3d-rancher
  name: k3d-k3d-rancher
current-context: k3d-k3d-rancher
kind: Config
preferences: {}
users:
- name: admin@k3d-k3d-rancher
  user:
    password: dd79f910ebe64a30855bcd38b7425b98
    username: admin
```

set KUBECONFIG=%KUBECONFIG_FILE%

kubectl get nodes

```
NAME                     STATUS   ROLES    AGE     VERSION
k3d-k3d-rancher-agent-1    Ready    <none>   7m36s   v1.18.6+k3s1
k3d-k3d-rancher-agent-0    Ready    <none>   7m35s   v1.18.6+k3s1
k3d-k3d-rancher-agent-2    Ready    <none>   7m35s   v1.18.6+k3s1
k3d-k3d-rancher-server-0   Ready    master   7m34s   v1.18.6+k3s1
```

kubectl get pods

No resources found in default namespace.

kubectl config view -o jsonpath='{.users[*].name}'

'admin@k3d-k3d-rancher'

kubectl config get-contexts

```
CURRENT   NAME              CLUSTER           AUTHINFO                NAMESPACE
*         k3d-k3d-rancher   k3d-k3d-rancher   admin@k3d-k3d-rancher
```

kubectl config current-context

k3d-k3d-rancher

kubectl create namespace cattle-system

namespace/cattle-system created

kubectl create namespace cert-manager

namespace/cert-manager created

k3d node list

```
NAME                  ROLE          CLUSTER      STATUS
k3d-k3d-rancher-agent-0   agent         k3d-rancher   running
k3d-k3d-rancher-agent-1   agent         k3d-rancher   running
k3d-k3d-rancher-agent-2   agent         k3d-rancher   running
k3d-k3d-rancher-server-0  server        k3d-rancher   running
k3d-k3d-rancher-serverlb  loadbalancer  k3d-rancher   running
```

kubectl get nodes

```
NAME                  STATUS   ROLES    AGE   VERSION
k3d-k3d-rancher-agent-1   Ready    <none>   27m   v1.18.6+k3s1
k3d-k3d-rancher-agent-0   Ready    <none>   27m   v1.18.6+k3s1
k3d-k3d-rancher-agent-2   Ready    <none>   27m   v1.18.6+k3s1
k3d-k3d-rancher-server-0  Ready    master   27m   v1.18.6+k3s1
```

kubectl get namespaces

```
NAME              STATUS   AGE
p-2v6dj           Active   22h
p-2244b           Active   22h
local             Active   22h
kube-node-lease   Active   22h
default           Active   22h
cattle-global-data   Active   22h
```

```
cattle-global-nt     Active   22h
kube-public          Active   22h
cert-manager         Active   22h
kube-system          Active   22h
user-l7m6j           Active   21h
cattle-system        Active   22h
```

Helm Installation

Install Helm if not already present

https://github.com/helm/helm/releases

For Windows that would be

https://get.helm.sh/helm-v3.6.0-windows-amd64.zip

Unzip to the C:\helm folder or wherever you want the binary to live

For MacOS:

$ **brew install helm**

For Windows:

Open powershell

PS> **choco install helm**

Open a Windows Command Prompt (CMD)

C:\ > **mkdir helm**

$> **cd \helm**

$> **dir windows-amd64\.**

```
 Volume in drive C is OS
 Volume Serial Number is 5081-CA53

 Directory of C:\helm\windows-amd64

09/09/2020  12:37 PM    <DIR>          .
09/09/2020  12:37 PM    <DIR>          ..
09/09/2020  12:37 PM        39,836,672 helm.exe
09/09/2020  12:37 PM            11,373 LICENSE
09/09/2020  12:37 PM             3,308 README.md
               3 File(s)     39,851,353 bytes
               2 Dir(s)  173,093,220,352 bytes free
```

$> **move windows-amd64\helm.exe .**

1 file(s) moved.

For Both MacOS and Windows Environment

$> **helm version**

version.BuildInfo{Version:"v3.12.1",
GitCommit:"249e5215cde0c3fa72e27eb7a30e8d55c9696144", GitTreeState:"clean",
GoVersion:"go1.14.7"}

$> **helm repo add rancher-latest https://releases.rancher.com/server-charts/latest**

"rancher-latest" has been added to your repositories

$> **helm repo add jetstack https://charts.jetstack.io**

"jetstack" has been added to your repositories

$> **helm repo update**

Hang tight while we grab the latest from your chart repositories...
...Successfully got an update from the "rancher-latest" chart repository
...Successfully got an update from the "jetstack" chart repository
Update Complete. ⎈Happy Helming!⎈

helm install cert-manager jetstack/cert-manager --namespace cert-manager --version v1.12.0 --set installCRDs=true --wait

NAME: cert-manager
LAST DEPLOYED: Mon Jul 31 21:44:33 2023
NAMESPACE: cert-manager
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
cert-manager has been deployed successfully!

In order to begin issuing certificates, you will need to set up a ClusterIssuer

or Issuer resource (for example, by creating a 'letsencrypt-staging' issuer).

More information on the different types of issuers and how to configure them

can be found in our documentation:

https://cert-manager.io/docs/configuration/

For information on how to configure cert-manager to automatically provision

Certificates for Ingress resources, take a look at the `ingress-shim`

documentation:

https://cert-manager.io/docs/usage/ingress/

Rollout the cert-manager deployment

kubectl -n cert-manager rollout status deploy/cert-manager

deployment "cert-manager" successfully rolled out

Prime the container images we need to reduce the likelihood of timeout

docker pull rancher/rancher
Pulling from rancher/rancher
Digest: sha256:5a16a6a0611e49d55ff9d9fbf278b5ca2602575de8f52286b18158ee1a8a
5963

docker pull rancher/k3s
Pulling from rancher/k3s
Digest: sha256:a835d76608a2503af8b681bb5888499d7c3456902f6853c8c1031f4a884715
ca

docker pull rancher/server
latest: Pulling from rancher/server
Digest: sha256:95b55603122c28baea4e8d94663aa34ad770bbc624a9ed6ef986fb3ea5224d91
Status: Image is up to date for rancher/server:latest
docker.io/rancher/server:latest

docker pull rancher/k3d-proxy
latest: Pulling from rancher/k3d-proxy
Digest: sha256:2ff467bb4a25f904954f7f65e4c7c73134b53bd422f4229f106c7c202ee347e2
Status: Image is up to date for rancher/k3d-proxy:latest

docker.io/rancher/k3d-proxy:latest

Install Rancher with a Helm 3

chart

helm install rancher rancher-latest/rancher --namespace cattle-system --set hostname=rancher.k3d.localhost --wait --timeout 900s

NAME: rancher
LAST DEPLOYED: Mon Jul 31 21:34:47
2023 NAMESPACE: cattle-system
STATUS: deployed
REVISION: 1
TEST SUITE: None
NOTES:
Rancher Server has been installed.

NOTE: Rancher may take several minutes to fully initialize. Please standby while Certificates are being issued and Ingress comes up.

Check out our docs at https://rancher.com/docs/rancher/v2.x/en/

Browse to https://rancher.k3d.localhost

Happy Containering!

Check status of the rancher deployment

kubectl -n cattle-system rollout status deploy/rancher

deployment "rancher" successfully rolled out

Load the URL https://rancher.k3d.localhost

This requires that entry to be added to localhost 127.0.0.1 in our /etc/hosts file

On MacOS:

Edit the hosts file and add the following

sudo vim /etc/hosts

127.0.0.1 kubernetes.docker.internal rancher.k3d.localhost k3d.my.org sample.k3d.localhost

On Windows:

Edit the hosts file and add the following line

notepad c:\windows\system32\drivers\etc\hosts

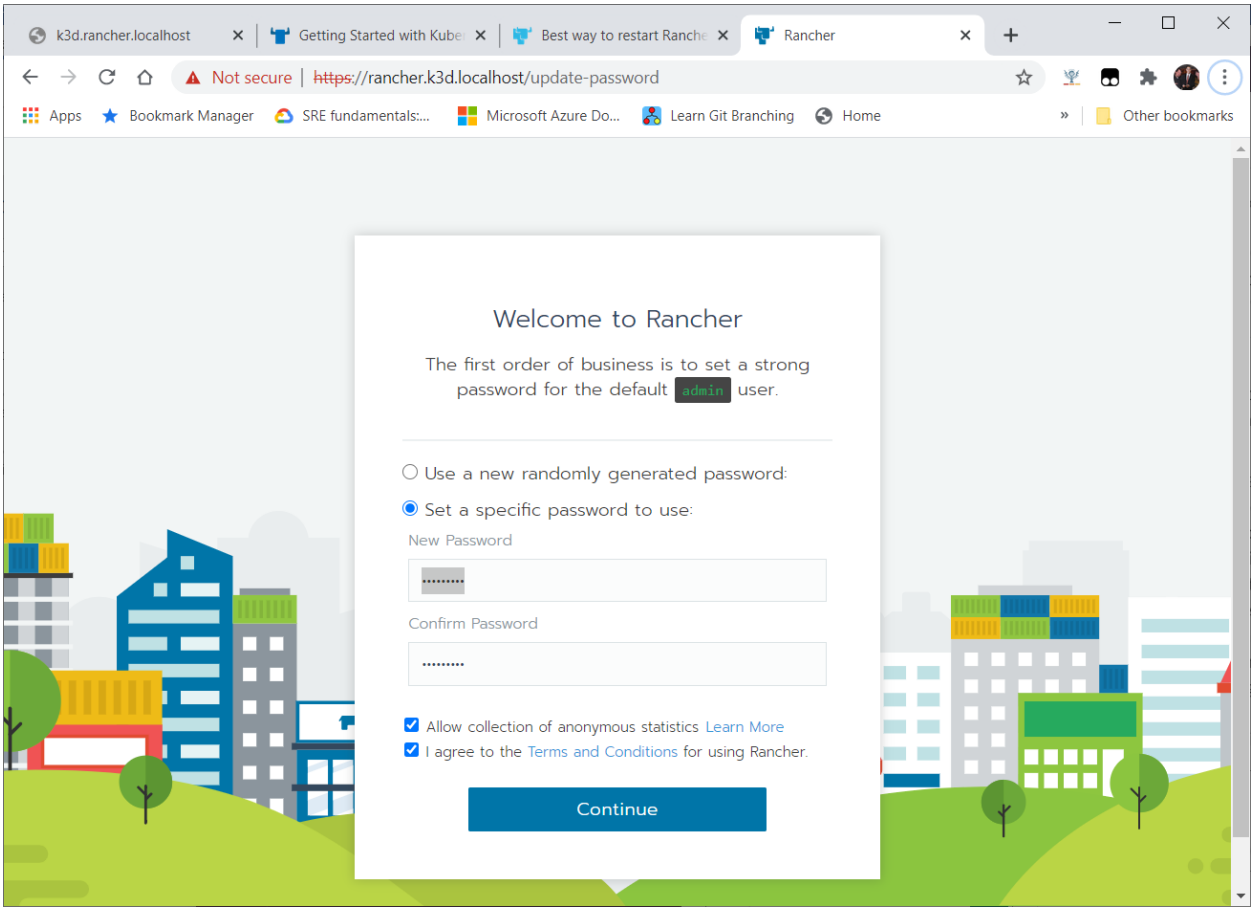
127.0.0.1 kubernetes.docker.internal rancher.k3d.localhost k3d.my.org sample.k3d.localhost

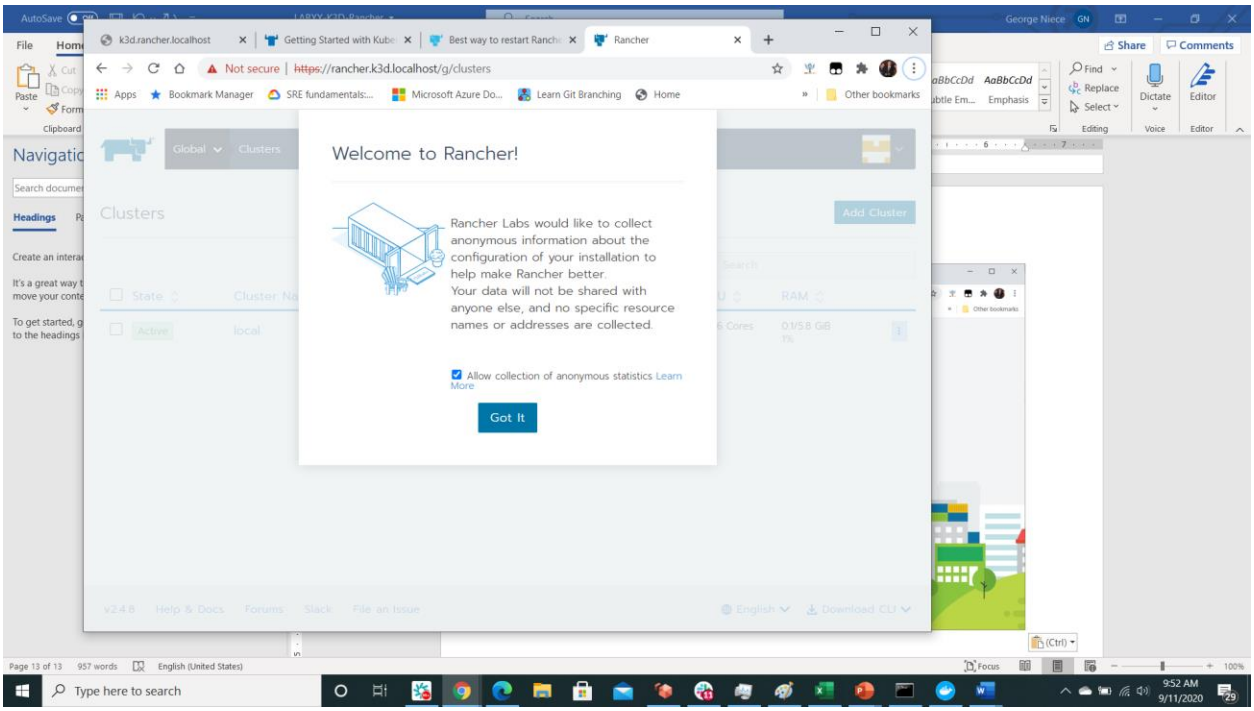
Load https://rancher.k3d.localhost in a browser

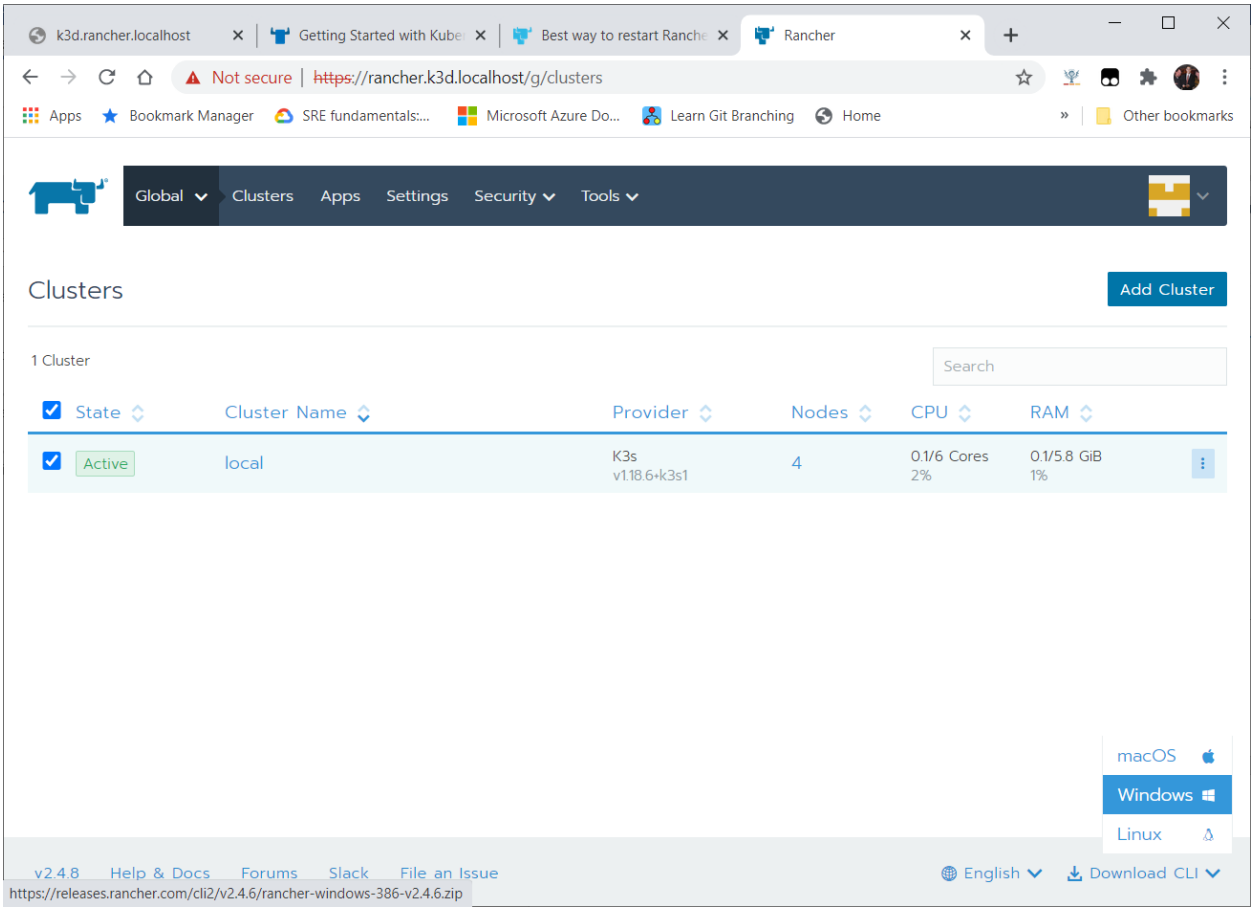
Execute the command for a Helm installation, to get the password from the Kubernetes cluster.

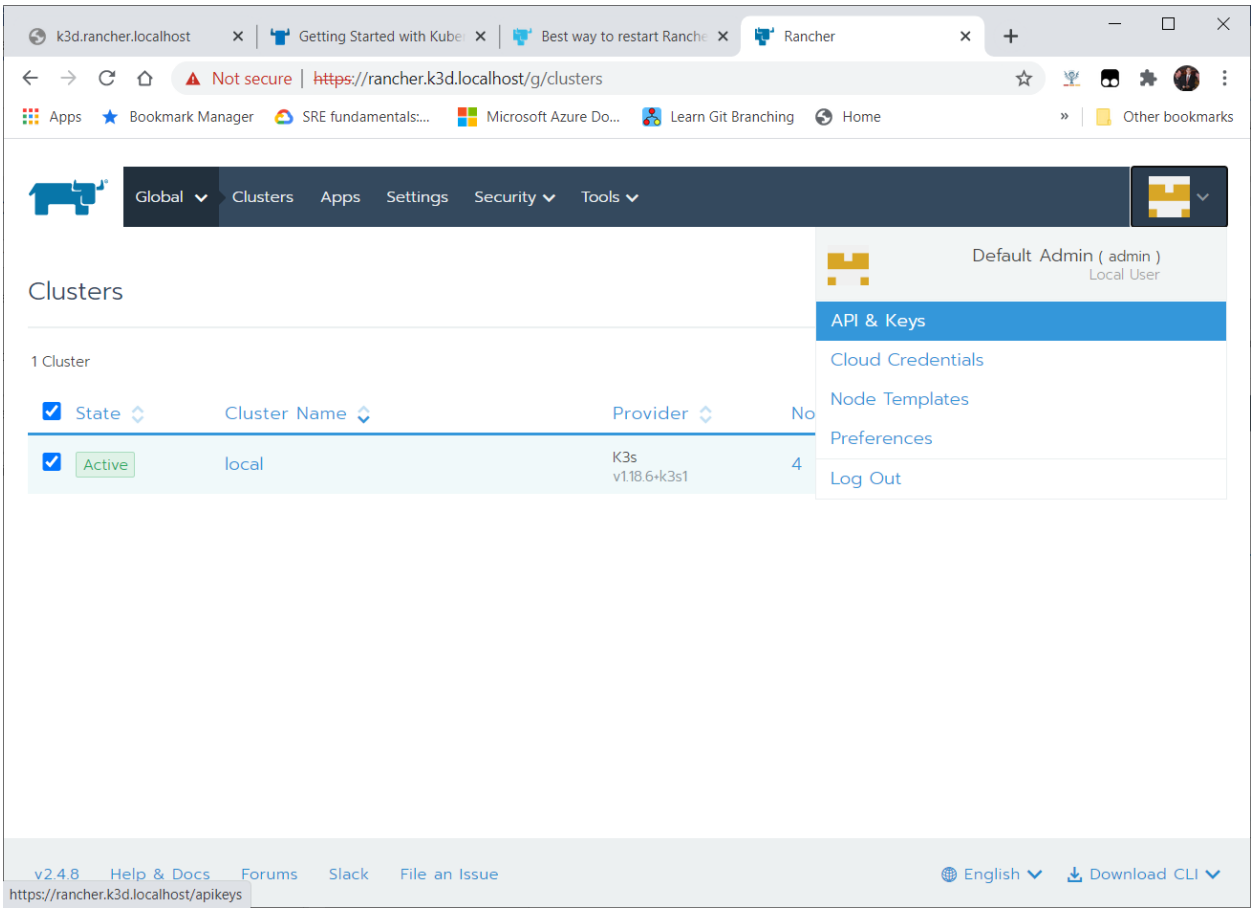
Supply the password and then set a specific password to use.

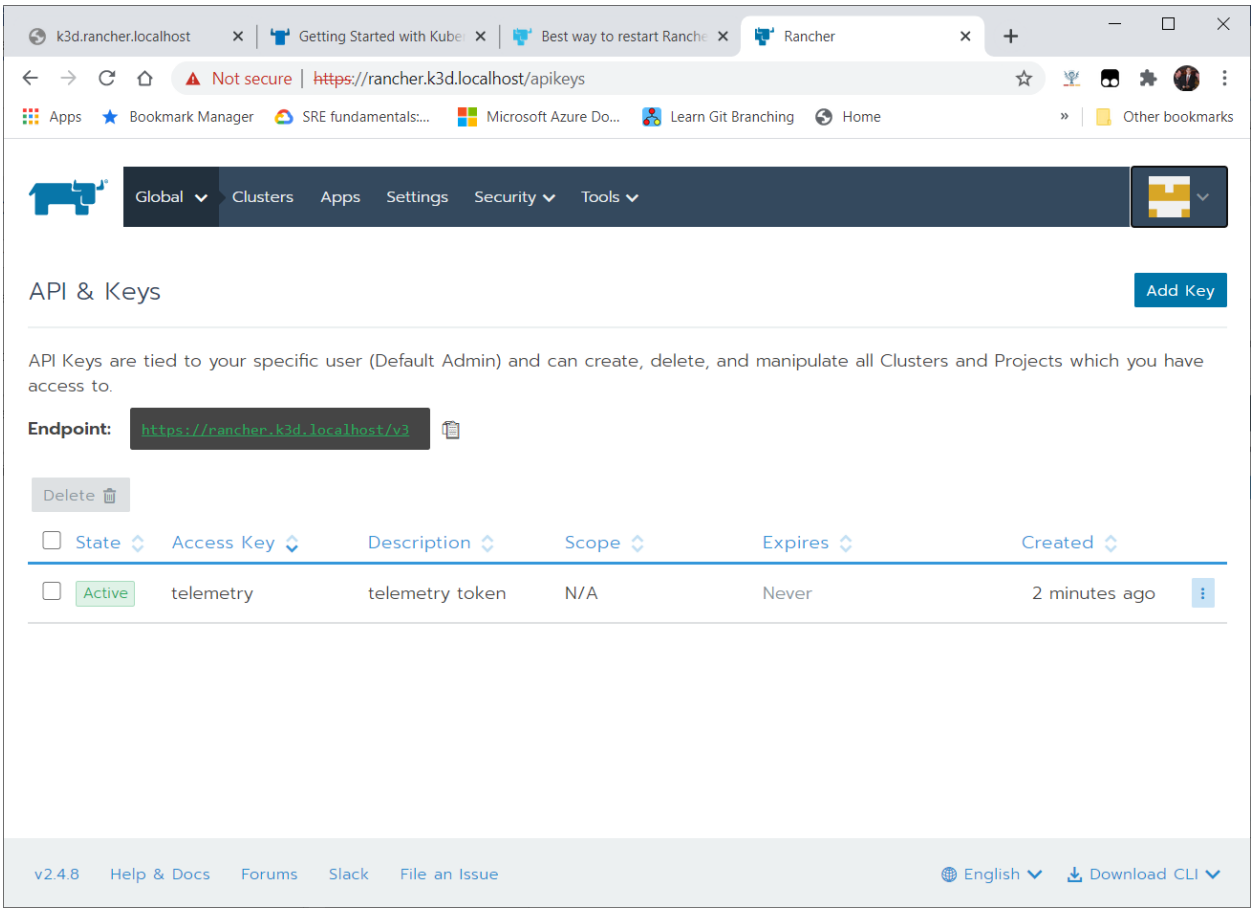
Leave the Server URL as https://rancher.k3d.localhost

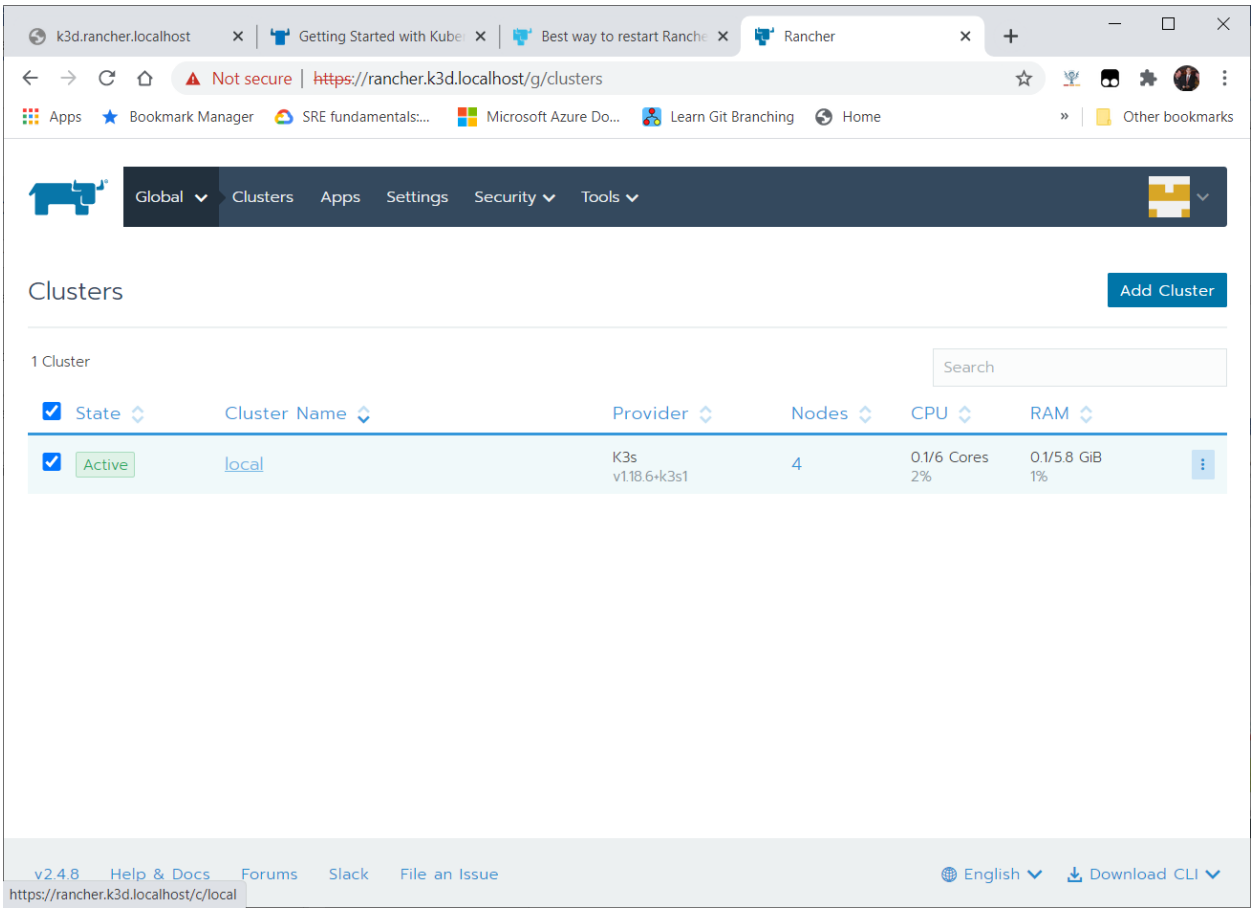


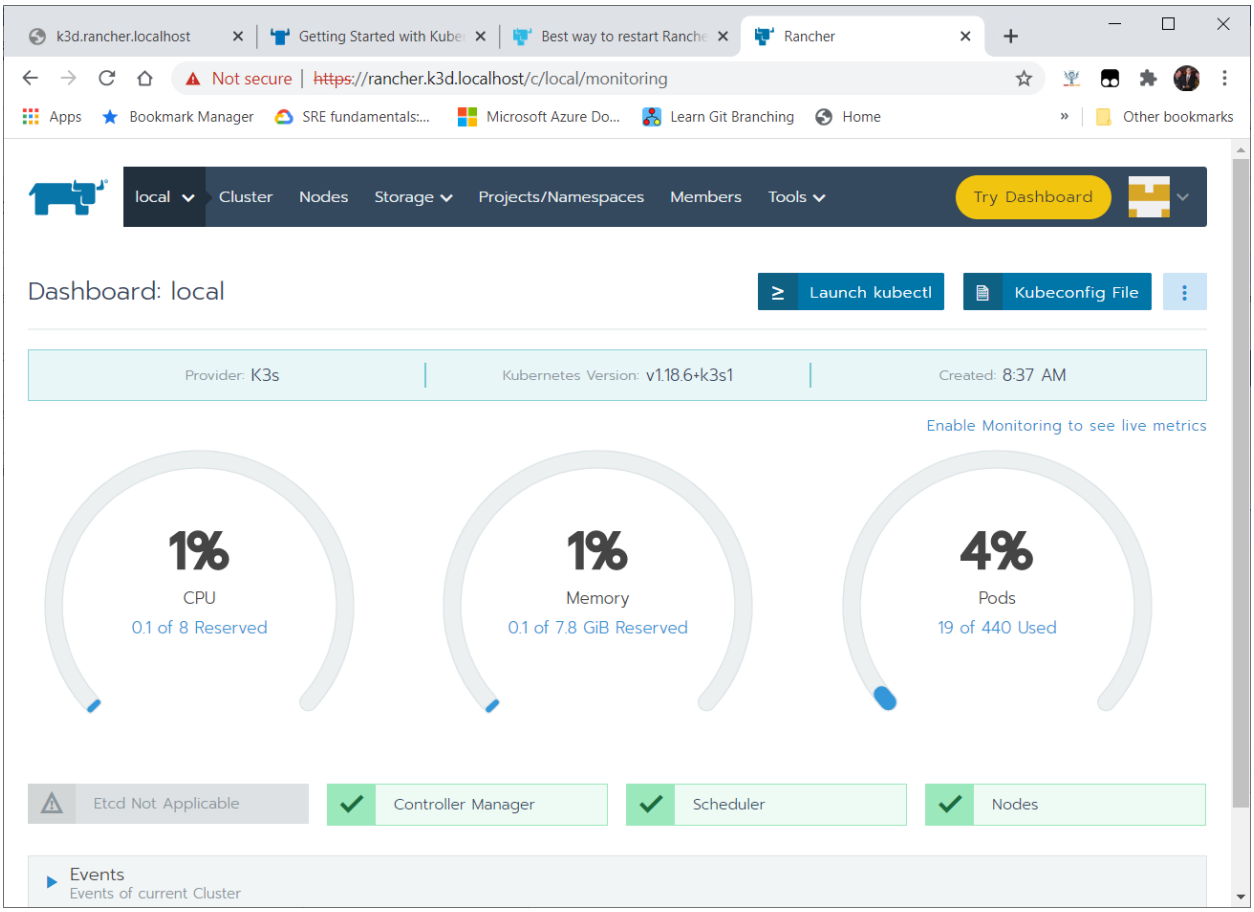

Welcome to Rancher!
Rancher Labs would like to collect anonymous information about the configuration of your installation to help make Rancher better.
Your data will not be shared with anyone else, and no specific resource names or addresses are collected.
Allow collection of anonymous statistics Learn More
Got It
Clusters
Add Cluster
Type here to search

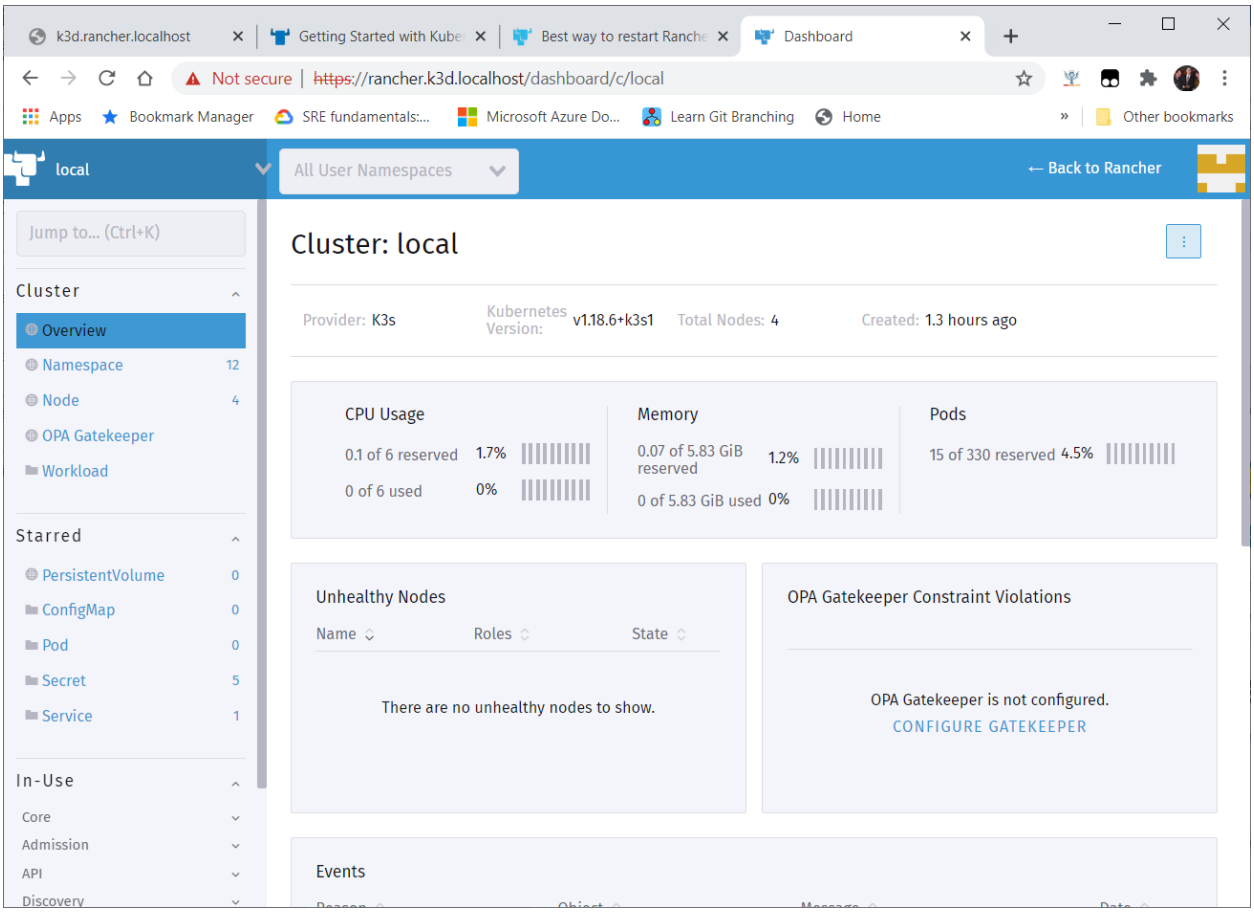

k3d.rancher.localhost
Getting Started with Kube
Best way to restart Ranche
Rancher
Not secure | https://rancher.k3d.localhost/g/clusters
Apps
Bookmark Manager
SRE fundamentals:...
Microsoft Azure Do...
Learn Git Branching
Home
Other bookmarks
Global
Clusters
Apps
Settings
Security
Tools
Clusters
Add Cluster
1 Cluster
Search
State
Cluster Name
Provider
Nodes
CPU
RAM
Active
local
K3s
v1.18.6+k3s1
4
0.1/6 Cores
2%
0.1/5.8 GiB
1%
macOS
Windows
Linux
v2.4.8
Help & Docs
Forums
Slack
File an Issue
English
Download CLI
https://releases.rancher.com/cli2/v2.4.6/rancher-windows-386-v2.4.6.zip


k3d.rancher.localhost
Getting Started with Kube
Best way to restart Ranche
Rancher
Not secure
https://rancher.k3d.localhost/g/clusters
Apps
Bookmark Manager
SRE fundamentals:...
Microsoft Azure Do...
Learn Git Branching
Home
Other bookmarks
Global
Clusters
Apps
Settings
Security
Tools
Default Admin (admin)
Local User
API & Keys
Cloud Credentials
Node Templates
Preferences
Log Out
Clusters
1 Cluster
State
Cluster Name
Provider
No
Active
local
K3s
v1.18.6+k3s1
4
v2.4.8
Help & Docs
Forums
Slack
File an Issue
English
Download CLI
https://rancher.k3d.localhost/apikeys


k3d.rancher.localhost
Getting Started with Kube
Best way to restart Ranche
Rancher
Not secure | https://rancher.k3d.localhost/apikeys
Apps
Bookmark Manager
SRE fundamentals:...
Microsoft Azure Do...
Learn Git Branching
Home
Other bookmarks
Global
Clusters
Apps
Settings
Security
Tools
API & Keys
Add Key
API Keys are tied to your specific user (Default Admin) and can create, delete, and manipulate all Clusters and Projects which you have access to.
Endpoint:
https://rancher.k3d.localhost/v3
Delete
State
Access Key
Description
Scope
Expires
Created
Active
telemetry
telemetry token
N/A
Never
2 minutes ago
v2.4.8
Help & Docs
Forums
Slack
File an Issue
English
Download CLI


k3d.rancher.localhost
Getting Started with Kube
Best way to restart Ranche
Rancher
Not secure | https://rancher.k3d.localhost/g/clusters
Apps
Bookmark Manager
SRE fundamentals:...
Microsoft Azure Do...
Learn Git Branching
Home
Other bookmarks
Global
Clusters
Apps
Settings
Security
Tools
Clusters
Add Cluster
1 Cluster
Search
State
Cluster Name
Provider
Nodes
CPU
RAM
Active
local
K3s
v1.18.6+k3s1
4
0.1/6 Cores
2%
0.1/5.8 GiB
1%
v2.4.8
Help & Docs
Forums
Slack
File an Issue
English
Download CLI
https://rancher.k3d.localhost/c/local

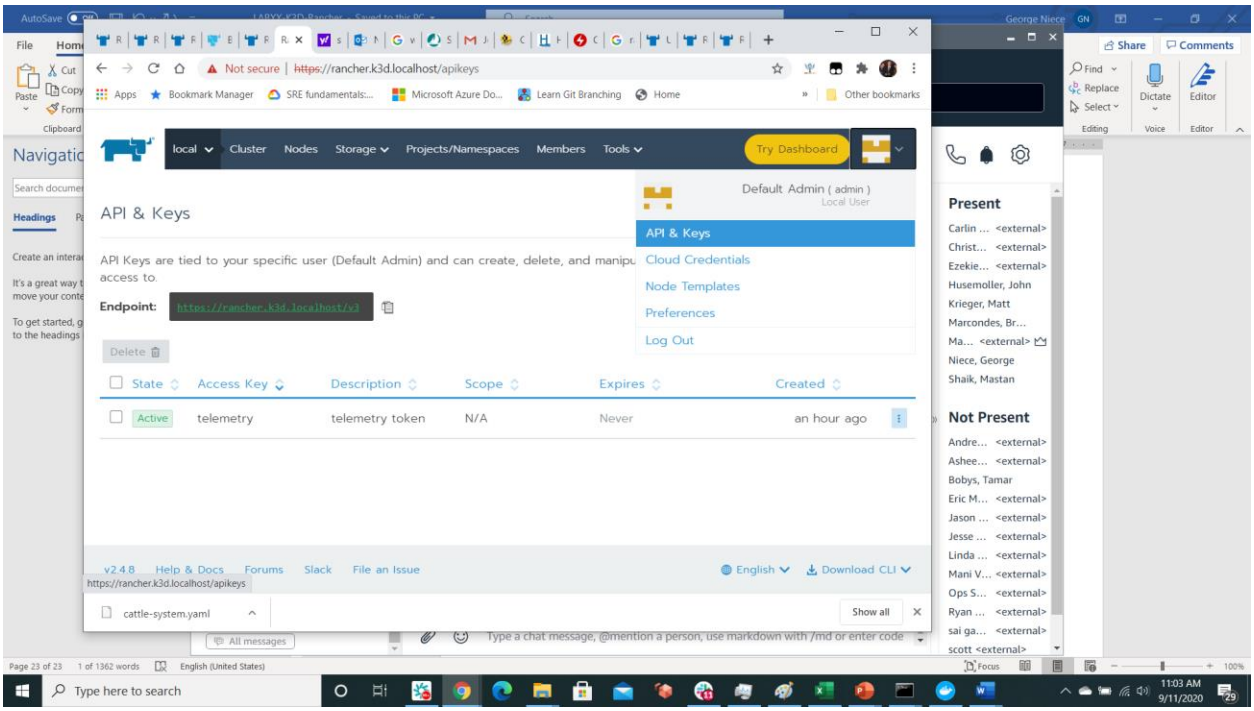

k3d.rancher.localhost
Getting Started with Kube
Best way to restart Ranche
Rancher
Not secure
https://rancher.k3d.localhost/c/local/monitoring
Apps
Bookmark Manager
SRE fundamentals:...
Microsoft Azure Do...
Learn Git Branching
Home
Other bookmarks
local
Cluster
Nodes
Storage
Projects/Namespaces
Members
Tools
Try Dashboard
Dashboard: local
Launch kubectl
Kubeconfig File
Provider: K3s
Kubernetes Version: v1.18.6+k3s1
Created: 8:37 AM
Enable Monitoring to see live metrics
1%
CPU
0.1 of 8 Reserved
1%
Memory
0.1 of 7.8 GiB Reserved
4%
Pods
19 of 440 Used
Etcd Not Applicable
Controller Manager
Scheduler
Nodes
Events
Events of current Cluster

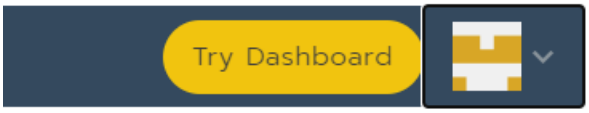


https://rancher.com/docs/rancher/v2.x/en/quick-start-guide/workload/

https://rancher.com/docs/rancher/v2.x/en/quick-start-guide/workload/quickstart-deploy-workload-ingress/

https://rancher.com/docs/rancher/v2.x/en/quick-start-guide/workload/quickstart-deploy-workload-nodeport/





Clusters and Projects which you have

Add API Key

Description

ExperimentKey

Automatically Expire

- Never
- A day from now
- A month from now
- A year from now
- Custom - 0 minutes

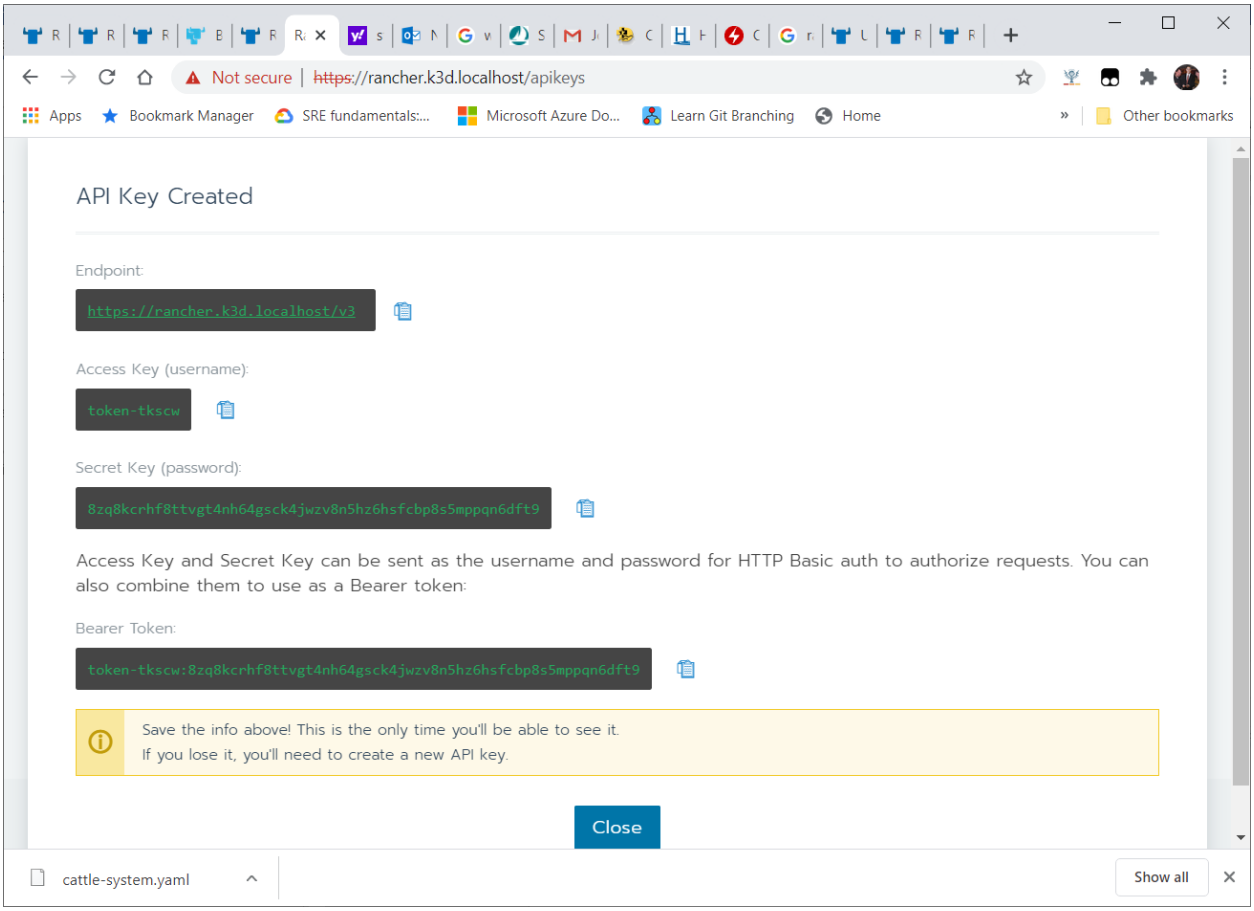
Scope

no scope

Cluster-scoped tokens can be used to interact directly with the Kubernetes API of clusters configured with an Authorized Cluster Endpoint

Create Cancel

Information for API Key is displayed for Endpoint, Access Key, Secret Key and Bearer Token



API Key Created

Endpoint:https://rancher.k3d.localhost/v3

Access Key (username):token-tkscw

Secret Key (password):8zq8kcrhf8ttvgt4nh64gsck4jwzv8n5hz6hsfcbp8s5mppqn6dft9

Access Key and Secret Key can be sent as the username and password for HTTP Basic auth to authorize requests. You can also combine them to use as a Bearer token:

Bearer Token:token-tkscw:8zq8kcrhf8ttvgt4nh64gsck4jwzv8n5hz6hsfcbp8s5mppqn6dft9

Save the info above! This is the only time you'll be able to see it.
If you lose it, you'll need to create a new API key.

Delete our Rancher cluster

k3d cluster delete k3d-rancher

[36mINFO[0m[0000] Deleting cluster 'k3d-rancher'
[36mINFO[0m[0000] Deleted k3d-k3d-rancher-serverlb
[36mINFO[0m[0001] Deleted k3d-k3d-rancher-agent-2
[36mINFO[0m[0002] Deleted k3d-k3d-rancher-agent-1
[36mINFO[0m[0005] Deleted k3d-k3d-rancher-agent-0
[36mINFO[0m[0007] Deleted k3d-k3d-rancher-server-0
[36mINFO[0m[0007] Deleting cluster network
[36mINFO[0m[0007] Deleting image volume 'k3d-k3d-rancher-images'
[36mINFO[0m[0007] Removing cluster details from default kubeconfig...
[36mINFO[0m[0007] Removing standalone kubeconfig file (if there is one)...
[36mINFO[0m[0007] Successfully deleted cluster k3d-rancher!