

Kubernetes Rancher

Kubernetes Rancher

2.1 Install Kubernetes rancher

Untuk menginstall kubernetes dapat menggunakan vagrant. <https://github.com/adisaputra10/vagrant/blob/master/Vagrantfile>, setelah file tersebut di download. Untuk menjalankan cukup menjalankan perintah

```
vagrant up
```

Selanjutnya untuk jalankan virtual master

```
vagrant ssh master
```

Setelah berhasil menjalankan vagrant maka install docker seperti di bawah ini

```
apt-get update && apt-get install docker.io
```

Setelah docker terinstall maka jalankan perintah docker rancher seperti di bawah ini

```
sudo docker run -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher:stable
```

```

root@ubuntu-xenial:/vagrant# sudo docker run -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher:latest
Unable to find image 'rancher/rancher:latest' locally
latest: Pulling from rancher/rancher
38e2e6cd5626: Pull complete
705054bc3f5b: Pull complete
c7051e069564: Pull complete
7308e914506c: Pull complete
0cfcb3cfb94b: Pull complete
49cb3551f487: Pull complete
8856f5defe68: Pull complete
d50abe29b623: Pull complete
297145c80f79: Pull complete
b6b66b1777e8: Pull complete
7edf8d37c2b7: Pull complete
511c877e7916: Pull complete
Digest: sha256:924b8acaa169821c86b840c33e1d79d87db0dfbb84dae6c102cc7c196811230f
Status: Downloaded newer image for rancher/rancher:latest
6413dcae5c53e94084cd8d4b719a645537964a1e1ddac2a7a43cc14300819d1
root@ubuntu-xenial:/vagrant#

```

Cek ip address dengan perintah **ifconfig**

```

root@ubuntu-xenial:/vagrant# ifconfig
docker0    Link encap:Ethernet  HWaddr 02:42:24:10:92:59
            inet addr:172.17.0.1  Bcast:172.17.255.255  Mask:255.255.0.0
            inet6 addr: fe80::42:24ff:fe10:9259/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:957 errors:0 dropped:0 overruns:0 frame:0
            TX packets:993 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0
            RX bytes:54480 (54.4 KB)  TX bytes:2101863 (2.1 MB)

enp0s3     Link encap:Ethernet  HWaddr 02:46:93:03:7f:ca
            inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
            inet6 addr: fe80::46:93ff:fe03:7fca/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:244845 errors:0 dropped:0 overruns:0 frame:0
            TX packets:74607 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:252167911 (252.1 MB)  TX bytes:4632895 (4.6 MB)

enp0s8     Link encap:Ethernet  HWaddr 08:00:27:67:4b:b6
            inet addr:192.168.100.3  Bcast:192.168.100.255  Mask:255.255.255.0
            inet6 addr: fe80::a00:27ff:fe67:4bb6/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:246 errors:0 dropped:0 overruns:0 frame:0
            TX packets:14 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:22614 (22.6 KB)  TX bytes:1156 (1.1 KB)

lo         Link encap:Local Loopback
            inet addr:127.0.0.1  Mask:255.0.0.0
            inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING  MTU:65536  Metric:1
            RX packets:0 errors:0 dropped:0 overruns:0 frame:0
            TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1
            RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

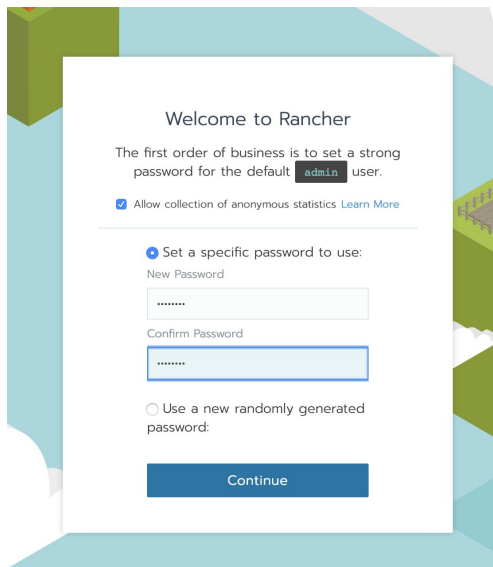
veth780b5dd Link encap:Ethernet  HWaddr 76:a7:b9:21:c6:98
            inet6 addr: fe80::74a7:b9ff:fe21:c698/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:957 errors:0 dropped:0 overruns:0 frame:0
            TX packets:1001 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0
            RX bytes:67878 (67.8 KB)  TX bytes:2102511 (2.1 MB)

root@ubuntu-xenial:/vagrant#

```

Selanjutnya buka di browser <https://192.168.100.3/update-password>

Halaman pertama ada untuk set password seperti di bawah ini



Welcome to Rancher

The first order of business is to set a strong password for the default `admin` user.

☒ Allow collection of anonymous statistics [Learn More](#)

☒ Set a specific password to use:

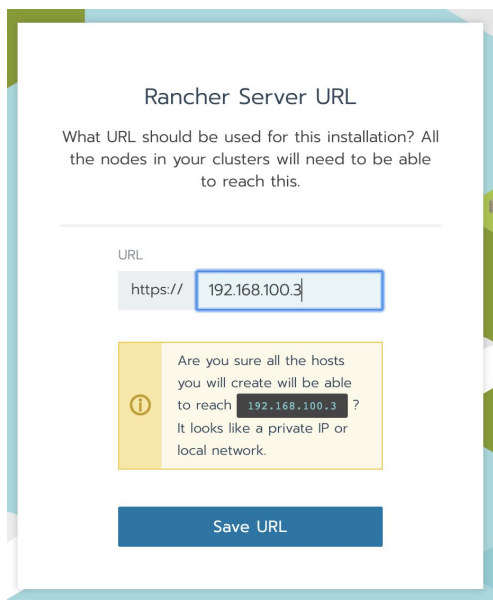
New Password

Confirm Password

☐ Use a new randomly generated password:

[Continue](#)

Selanjutnya setting url seperti di bawah ini




Rancher Server URL

What URL should be used for this installation? All the nodes in your clusters will need to be able to reach this.

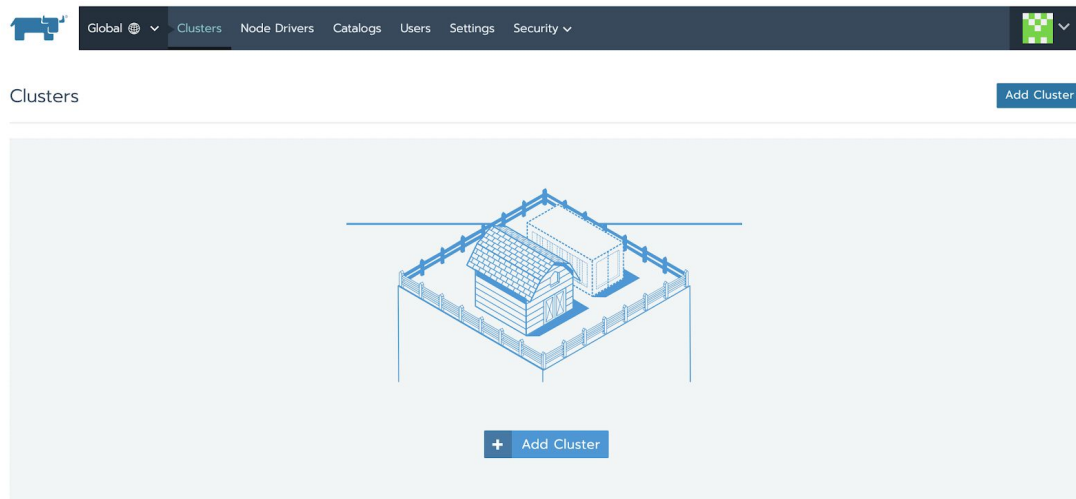
URL

[https://](#)

 Are you sure all the hosts you will create will be able to reach `192.168.100.3` ? It looks like a private IP or local network.

[Save URL](#)

Halaman kubernetes rancher seperti di bawah ini, untuk mengatur kubernetes maka dapat klik Add Cluster seperti di bawah ini



Selanjutnya pilih custom seperti di bawah ini dan ketik nama cluster seperti kotak merah di bawah ini. Kemudian klik Next.

Add Cluster

In a hosted Kubernetes provider

Google Container Engine

Amazon EKS

Azure Kubernetes Service

Import existing cluster

IMPORT

From nodes in an infrastructure provider

Amazon EC2

Microsoft Azure

DigitalOcean

vSphere

From my own existing nodes

CUSTOM

Cluster Name *

Add a Description

Member Roles

Control who has access to the cluster and what permission they have to change it.

Edit as YAML

Cluster Options

Customize Kubernetes options for the cluster


Show advanced options

Next

Cancel

Centang semua etcd , control plane dan worker kemudian klik kotak merah untuk mengcopy kode docker kemudian paste pada server 192.168.100.3, sesuai dengan server yang di install docker rancher sebelumnya

Add Cluster: training

Edit as YAML 

Customize Node Run Command
Editing node options will update the command you will run on your existing machines


1 Node Options
Choose what roles the node will have in the cluster

Node Role

☒ etcd ☒ Control Plane ☒ Worker [Show advanced options](#)

2 Run this command on one or more existing machines already running a supported version of Docker.

```
sudo docker run -d --privileged --restart=unless-stopped --net=host -v /etc/kubernetes:/etc/kubernetes -v /var/run:/var/run rancher/rancher-agent:v2.1.6 --server https://192.168.100.3 --token m1c7dt5rc5g7d2sxm8rnp7dpxfxfd61pn4nll2mb7vzr76c9xcnjkh --ca-checksum 8b9873b9dda6fe96c88e18d6a4517cee639229b720d50240c2494477ccd6b8d9 --etcd --controlplane --worker
```



Done

Pada server akan muncul seperti di bawah ini pada saat menjalankan docker rancher saat membuat cluster

```
root@ubuntu-venailt:~# sudo docker run -d --privileged --restart=unless-stopped --net=host -v /etc/kubernetes:/etc/kubernetes -v /var/run:/var/run rancher/rancher-agent:v2.1.6 --server https://192.168.100.3 --token m1c7dt5rc5g7d2sxm8rnp7dpxfxfd61pn4nll2mb7vzr76c9xcnjkh --ca-checksum 8b9873b9dda6fe96c88e18d6a4517cee639229b720d50240c2494477ccd6b8d9 --etcd --controlplane --worker
Unable to find image 'rancher/rancher-agent:v2.1.6' locally
v2.1.6: Pulling from rancher/rancher-agent
8b9873b9dda6fe96c88e18d6a4517cee639229b720d50240c2494477ccd6b8d9: Already exists
76b8d6c3f5b: Already exists
76b8d6c3f5b: Already exists
76b8d6c3f5b: Already exists
76b8d6c3f5b: Downloading [=====] 8.75MB/31.33MB
76b8d6c3f5b: Download complete
76b8d6c3f5b: Downloading [=====] 5.10MB/16.91MB
76b8d6c3f5b: Downloading [=====] 3.96MB/12.8MB
```

Setelah di jalankan pada browser maka akan muncul seperti kotak merah di bawah ini kemudian klik **Done**

Add Cluster: training

Edit as YAML

Customize Node Run Command
Editing node options will update the command you will run on your existing machines

1 Node Options
Choose what roles the node will have in the cluster

Node Role

☒ etcd ☒ Control Plane ☒ Worker

Show advanced options

2 Run this command on one or more existing machines already running a supported version of Docker.

```
sudo docker run -d --privileged --restart=unless-stopped --net=host -v /etc/kubernetes:/etc/kubernetes -v /var/run:/var/run rancher/rancher-agent:v2.1.6 --server https://192.168.100.3 --token mlc7dt5rc5g7d2sxm8rkp7dpfxfd61pn4n112mb7vzr76c9xcnjkh --ca-checksum 8b9873b9dda6fe96c88e18d6a4517cee639229b720d50240c2494477ccd6b8d9 --etcd --controlplane --worker
```

1 new node has registered

Done

Proses pembuatan cluster kubernetes akan muncul seperti di bawah ini, tunggu hingga Aktif

Clusters Add Cluster

Delete

Search

State	Cluster Name	Provider	Nodes	CPU	RAM
Provisioning	training	Custom	1	n/a	n/a

[healthcheck] Start Healthcheck on service [kube-scheduler] on host [10.0.2.15]

Server Master cluster Kubernetes selesai di setting jika muncul seperti di bawah ini

Delete

Search

State	Cluster Name	Provider	Nodes	CPU	RAM
Active	training	Custom v1.11.6	1	0.3/2 Cores 13%	0/18 GiB 0%

Untuk melihat detail cluster kubernetes maka klik training seperti di bawah ini

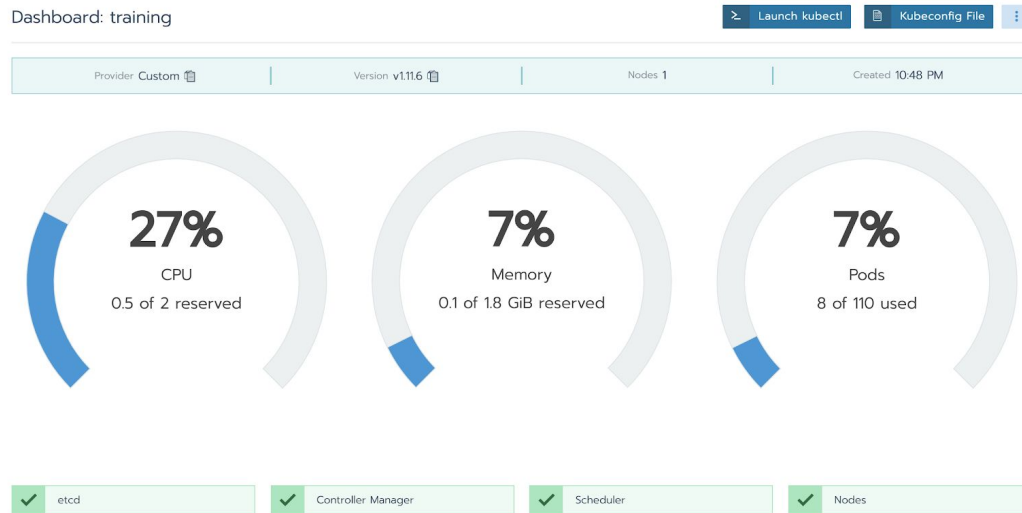
Clusters Add Cluster

Delete

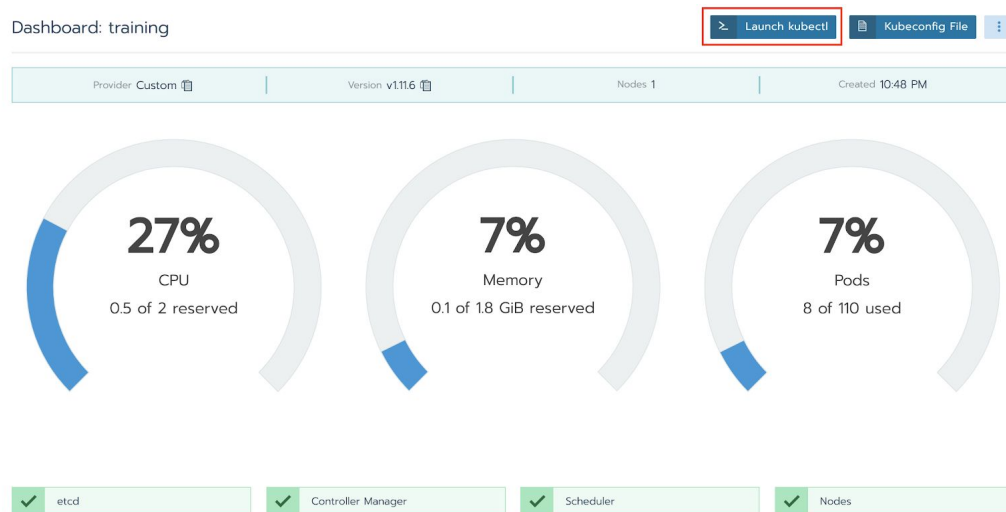
Search

<input type="checkbox"/> State	Cluster Name	Provider	Nodes	CPU	RAM	
<input type="checkbox"/> Active	training	Custom v1.11.6	1	0.3/2 Cores 13%	0/1.8 GiB 0%	

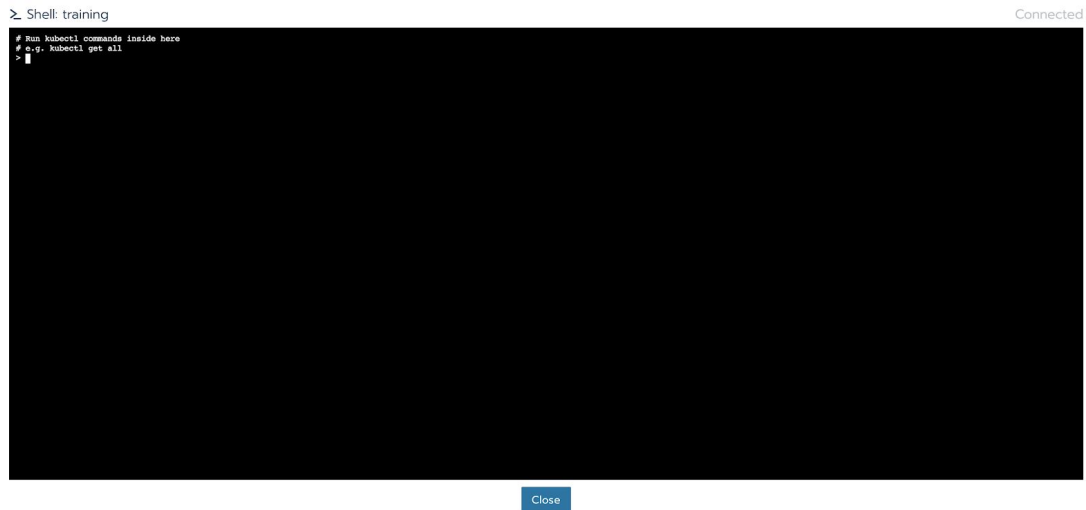
Akan muncul usage dan proses pada cluster kubernetes



Kubernetes cli dapat di akses dengan klik menu launch kubectl seperti pada kotak merah di bawah ini



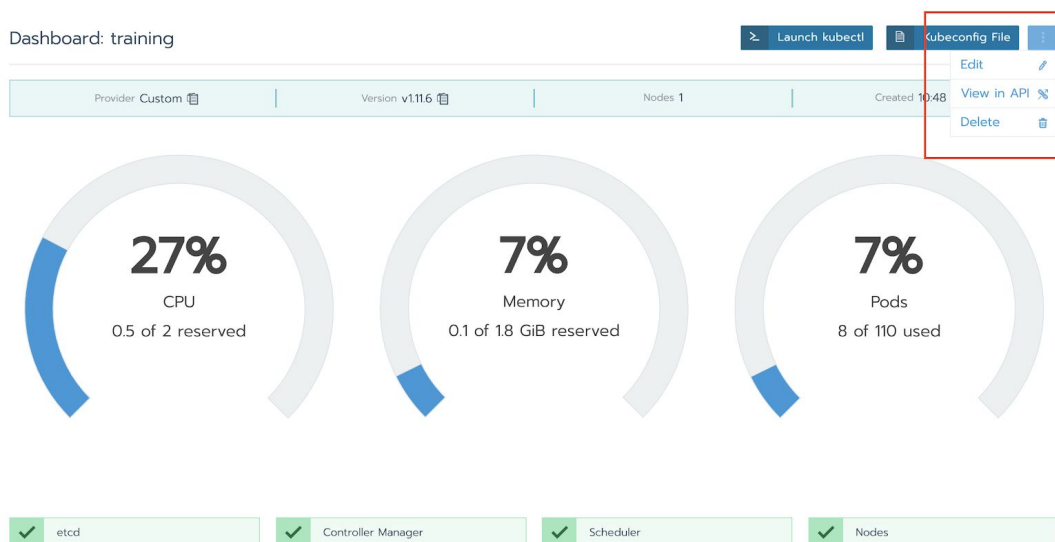
Maka akan muncul kubernetes cli



Pada kubernetes dapat add node server

2.2 Add Node Server

Untuk add node server pastikan hostname berbeda, hostname dapat di cek pada file `/etc/hostname`. langkah selanjutnya adalah dengan klik titik 3 pada kanan atas seperti gambar di bawah ini kemudian klik **Edit**



Selanjutnya scroll ke bawah hingga ketemu gambar seperti di bawah ini, pastikan yang di centang adalah worker kemudina klik show advance option seperti kotak merah di bawah ini

Cloud Provider

① If your cloud provider is not listed, please use the **Custom** option.

☒ None
☐ Amazon
☐ Azure
☐ Custom

[Read more about the Kubernetes cloud providers](#)

[Show advanced options](#)

Customize Node Run Command

Editing node options will update the command you will run on your existing machines

① Node Options

Choose what roles the node will have in the cluster

Node Role

☐ etcd
 ☐ Control Plane
 ☒ Worker

[Show advanced options](#)

② Run this command on one or more existing machines already running a supported version of Docker.

```

sudo docker run -d --privileged --restart=unless-stopped --net=host -v /etc/kubernetes:/etc/kubernetes -v /var/run:/var/run rancher/rancher-agent:v2.1.6 --server https://192.168.100.3 --token wt2nt5hsw19f6nkwjngxwp7csgfdgv5g4pr4bhtfs9fc9dgd27n4 --ca-checksum fa3e05261b37d85b026bc77565998f067fd06ee9df5a6b709a228e807ea3ee0 --worker
  
```

[Save](#) [Cancel](#)

Kemudian masukkan ip address untuk node server yang akan di tambahkan

Customize Node Run Command

Editing node options will update the command you will run on your existing machines

① Node Options

Choose what roles the node will have in the cluster

Node Role

☐ etcd
 ☐ Control Plane
 ☒ Worker

Node Address

Optionally configure the public address and internal address for machines

Public Address

192.168.100.4

Internal Address

e.g. 123.4

Node Name

Optionally configure the node name as identification instead of the actual hostname

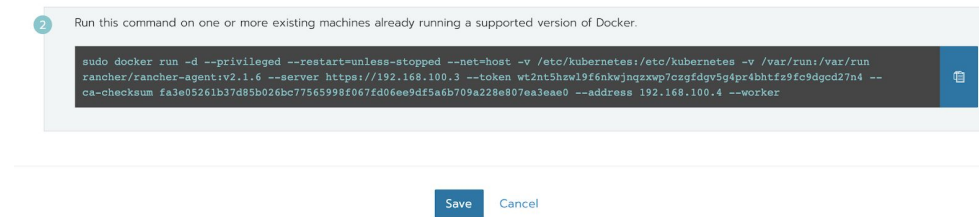
e.g. my-worker-node

Node Labels

Optional labels to be applied to the node

[+ Add Label](#)

Kemudian akan muncul code untuk membuat docker pada node server seperti di bawah ini



Selanjutnya ubah koding docker seperti kotak merah di bawah ini

```
sudo docker run -d --privileged --restart=unless-stopped --net=host -v /etc/kubernetes:/etc/kubernetes -v /var/run:/var/run rancher/rancher-agent:v2.1.6 --server https://192.168.100.3 --token m1c7dt5rc5g7d2sxn8rpk7dpfxfd61pn4n112mb7vzr76c9xcnjkh --ca-checksum 8b9873b9dda6fe96c88e18d6a4517cee639229b720d50240c2494477ccd6b8d9 --worker
```

--restart=unless-stopped di ubah menjadi --restart=always

Langkah selanjutnya adalah dengan masuk ke server node yang akan di tambahkan ke cluster kubernetes dengan perintah vagrant seperti di bawah ini

```
bmdusers-MacBook-Pro-2:training adisaputra$ vagrant ssh node1
Welcome to Ubuntu 16.04.5 LTS (GNU/Linux 4.4.0-138-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Get cloud support with Ubuntu Advantage Cloud Guest:
http://www.ubuntu.com/business/services/cloud

0 packages can be updated.
0 updates are security updates.

New release '18.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

-----
WARNING! Your environment specifies an invalid locale.
The unknown environment variables are:
LC_CTYPE=UTF-8 LC_ALL=
This can affect your user experience significantly, including the
ability to manage packages. You may install the locales by running:

    sudo apt-get install language-pack-UTF-8
    or
    sudo locale-gen UTF-8

To see all available language packs, run:
apt-cache search "^language-pack-[a-z][a-z]$"
To disable this message for all users, run:
    sudo touch /var/lib/cloud/instance/locale-check.skip
-----

vagrant@ubuntu-xenial:~$ sudo su
root@ubuntu-xenial:/home/vagrant#
root@ubuntu-xenial:/home/vagrant#
```

Kemudian jalan kan perintah docker seperti di bawah ini

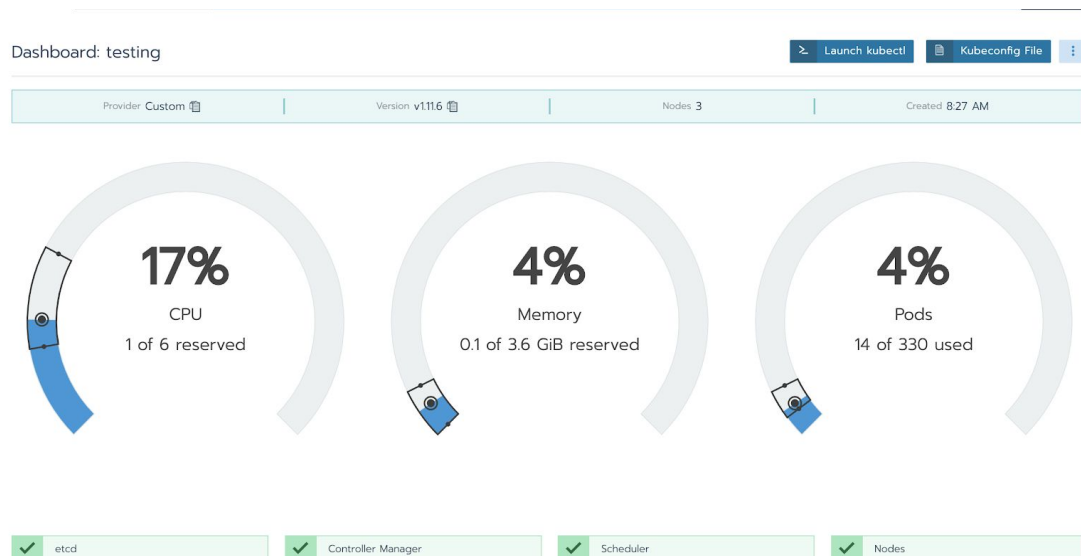
Jangan lupa install docker dengan perintah seperti di bawah ini

```
apt-get update && apt-get install docker.io
```

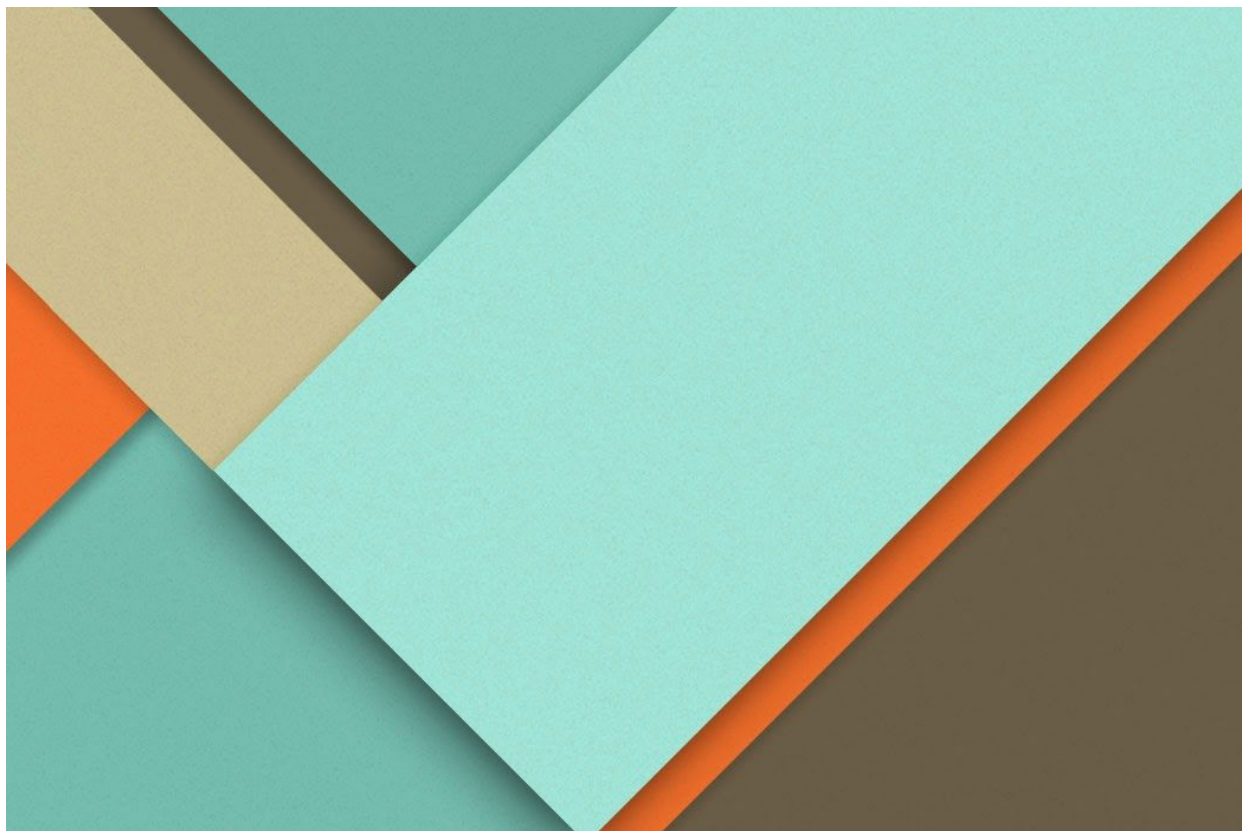
```
root@ubuntu-xenial:/home/vagrant# sudo docker run -d --privileged --restart=always --net=host -v /etc/kubernetes:/etc/kubernetes -v /var/run:/var/run rancher/rancher-agent:v2.1.6 --server https://192.168.100.3 --token m1c7dt5rc6g7d2sxn0k
c7629f1fd41penn12a7vz77a9cm1ab --ca-checksum B098739d0adfe9c88e18d0e4517ee539229b726d524e2c94a77cdd08d9 --worker
Unable to find image 'rancher/rancher-agent:v2.1.6' locally
v2.1.6: Pulling from rancher/rancher-agent
88a3dc0d52a1: Pull complete
785854bc3f5b: Pull complete
c7601ee092d4: Pull complete
7380e9f1d86c: Pull complete
e271e11baba: Pull complete
f9a674f0b0a1: Pull complete
727b31f9d0d1: Pull complete
c2cd6220e43: Pull complete
d1c27a3b0907: Pull complete
Digest: sha256:b413986a3c92440b2d4d13928989b3e77f83501f4dc88f9e7491e2d4d273bc
Status: Downloaded newer image for rancher/rancher-agent:v2.1.6
#97ee9e26c9511e317f5964640be7e62c27cd81d4e451fa59fe38f64971ef2
root@ubuntu-xenial:/home/vagrant#
```

Setelah menjalankan perintah seperti di atas maka

pada browser akan muncul halaman seperti di bawah ini



Dapat terlihat bahwa cpu yang di gunakan pada cluster beserta memori, cpu dan memori yang terlihat adalah total dari semua server node yang ada, jika memiliki 2 node server dengan kapasitas 2 Gb maka pada cluster akan muncul 4 Gb, untuk resource cluster server dapat di tambah atau scaling secara horizontal dengan penambahan server



Deploy

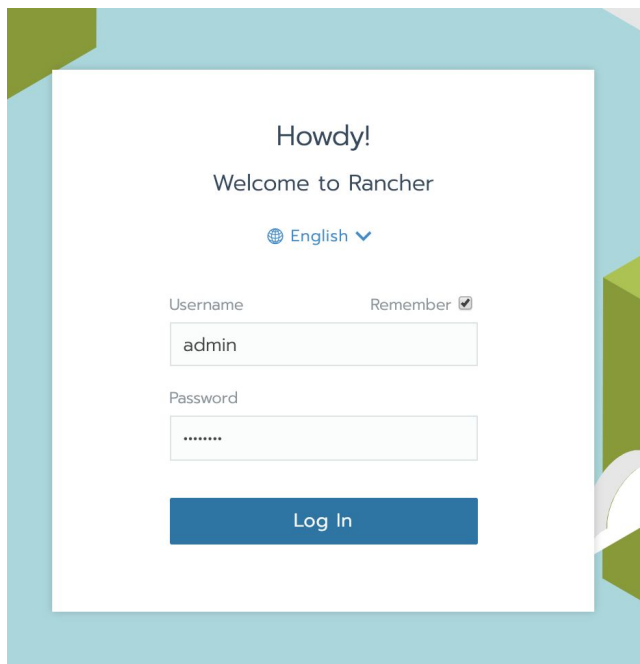
Docker

Kubernetes

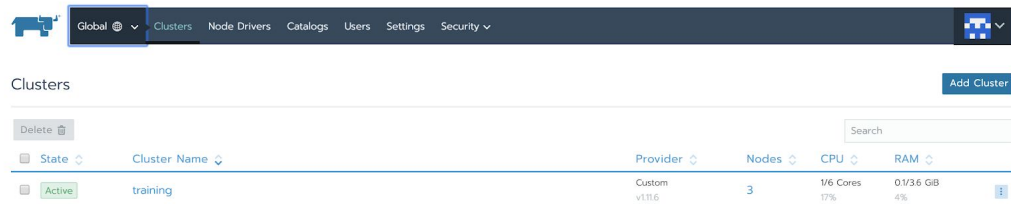
Deploy Docker Kubernetes

3.1 Deploy Web server pada kubernetes

Untuk mendeploy aplikasi pada kubernetes akan di lakukan pada workload. Workload adalah server untuk menjalankan docker. Untuk mendeploy docker cara nya adalah sebagai berikut



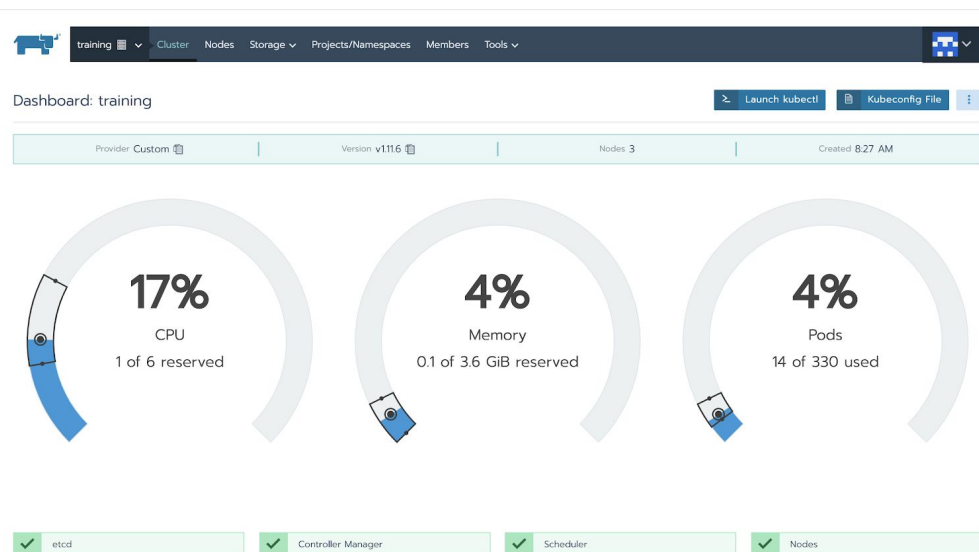
Selanjutnya akan muncul pilihan cluster pada kubernetes seperti di bawah ini



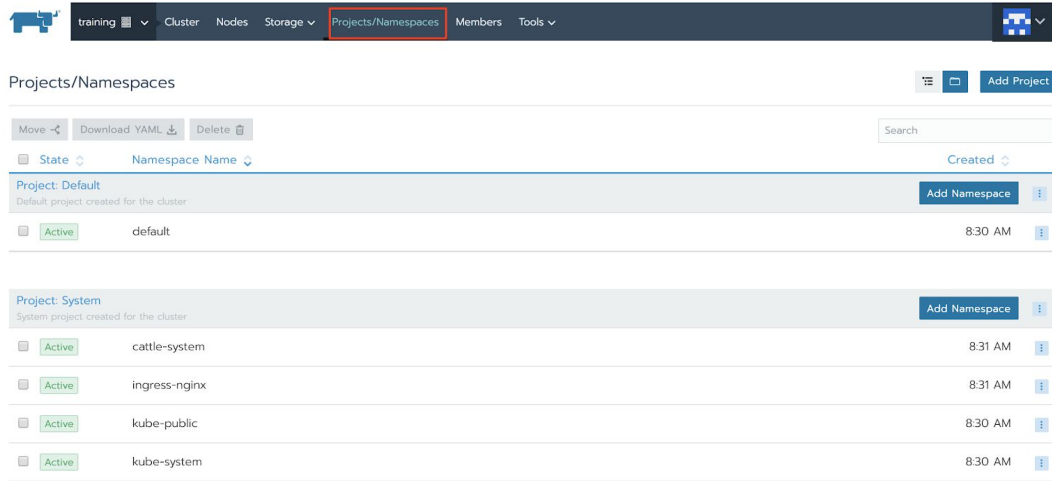
The screenshot shows the 'Clusters' page in the Kubernetes Dashboard. The top navigation bar includes 'Global', 'Clusters', 'Node Drivers', 'Catalogs', 'Users', 'Settings', and 'Security'. The 'Clusters' section has a search bar and a table of clusters. One cluster named 'training' is listed with a status of 'Active'. The table columns are State, Cluster Name, Provider, Nodes, CPU, and RAM.

State	Cluster Name	Provider	Nodes	CPU	RAM
Active	training	Custom v1.11.6	3	1/6 Cores 17%	0.1/3.6 GiB 4%

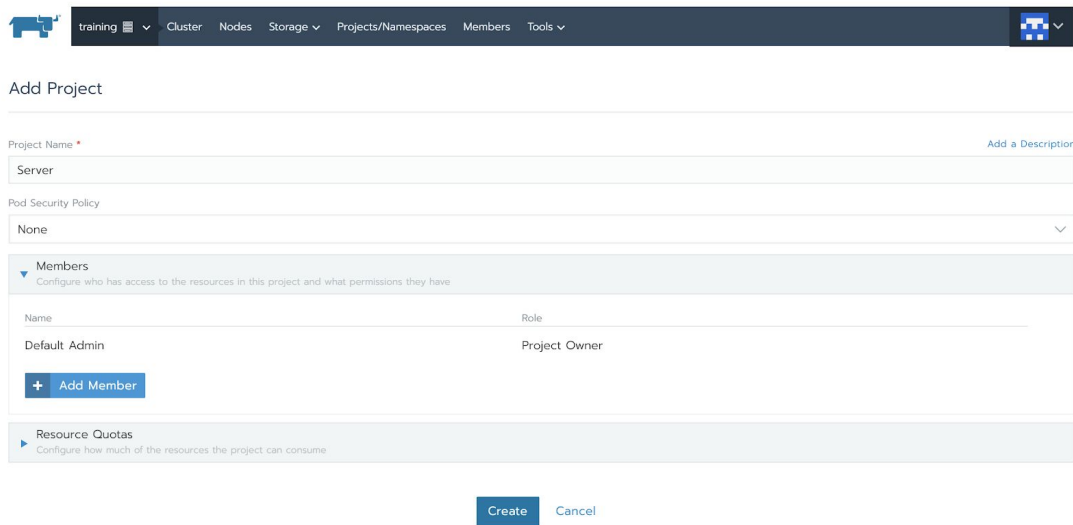
Pada halaman dashboard akan di tampilkan usage resource dari server



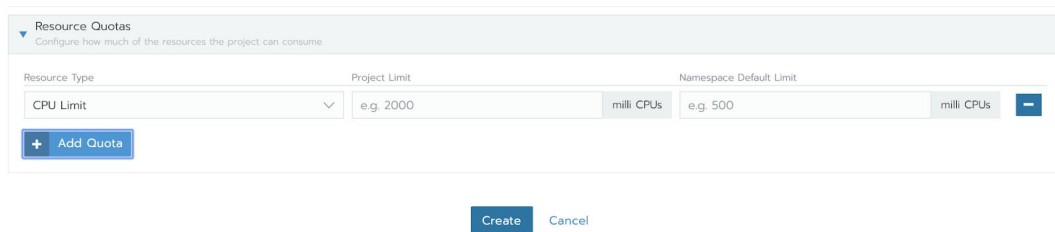
Untuk deploy aplikasi docker sebaiknya membuat project atau namespace sehingga docker yang di deploy akan terlihat baik , dapat klik tanda merah seperti di bawah ini



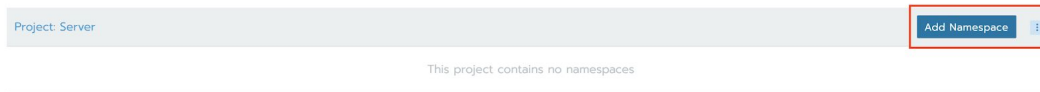
Selanjutnya Add Project seperti di bawah ini



Saat pembuatan project dapat di lakukan pembatasan atau limitasi seperti di bawah ini



Limitasi dapat di gunakan untuk environment development dan staging Setelah membuat project maka membuat namespace seperti di bawah ini



Selanjutnya penamaan namespace

Add Namespace

Name Add a Description

Webserver

Project

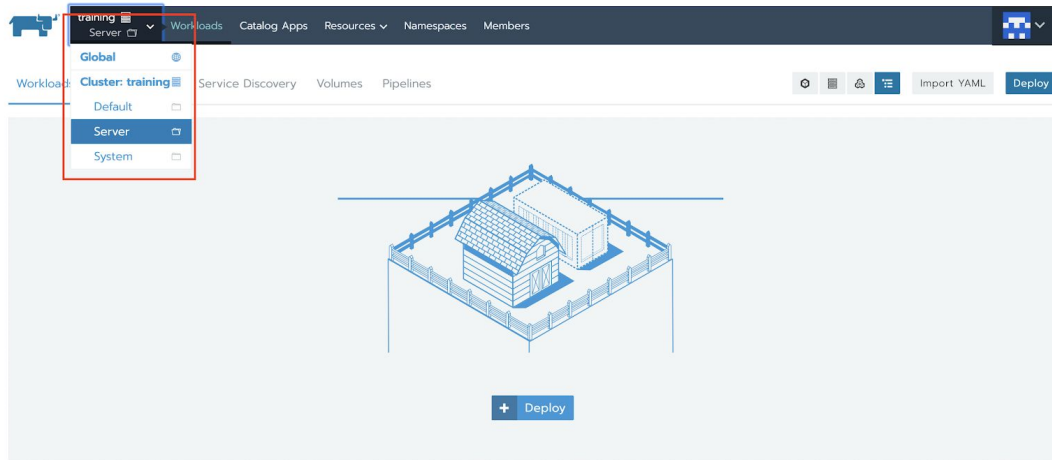
Server

Resource Quotas
Configure how much of the resources the namespace as a whole can consume

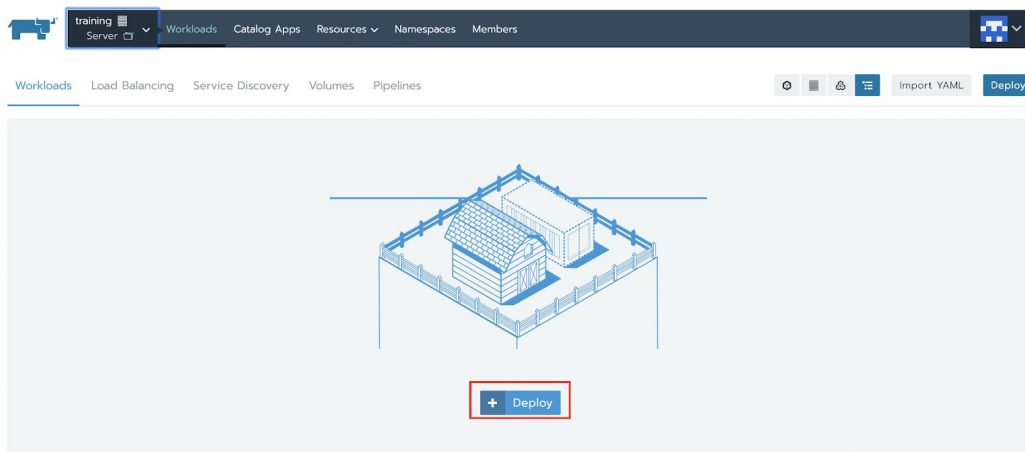
Resource Type	Project Resource Availability	Limit
CPU Limit	<div><div></div></div>	5000 milli CPUs

Create Cancel

Setelah berhasil membuat project dan namespace , pada menu dapat di temukan menu seperti di bawah ini



Untuk deploy docker maka kita dapat klik kotak merah seperti di bawah ini



Saat akan deploy akan muncul gambar seperti di bawah ini

Deploy Workload

Name * 1 [Add a Description](#)
web_server

Workload Type 2 [More options](#)
Scalable deployment of 1 pod

Docker Image * 3
nginx

Namespace * 4 [Add to a new namespace](#)
webserver

Port Mapping
Publish the container port * 5
80
+ Add Port

Protocol TCP
As a NodePort (On every node)
On listening port * Random

Environment Variables
Set the environment that will be visible to the container, including injecting values from other resources like Secrets.

Node Scheduling
Configure what nodes the pods can be deployed to.

Health Check
Periodically make a request to the container to see if it is alive and responding correctly.

Volumes
Persist and share data separate from the lifecycle of an individual container.

Scaling/Upgrade Policy
Configure how pods are replaced when performing an upgrade.

[Show advanced options](#)

Penjelasan

1. Name adalah nama untuk aplikasi yang akan di deploy
2. Scalable adalah jumlah container yang ingin di buat berapa banyak, saat traffic besar maka dapat di perbesar sesuai dengan kebutuhan
3. Docker image adalah nama image yang ada di registry, untuk saat ini kita akan mencoba image dari server yang bernama nginx
4. Namespace adalah group dari project yang akan di buat
5. Port Mapping berguna untuk mengakses docker yang berada pada kubernetes rancher

Pada saat membuat docker pada workload terdapat juga health check yang berguna untuk mengecek kondisi docker adapun parameter untuk mengecek adalah sebagai berikut

Health Check
Periodically make a request to the container to see if it is alive and responding correctly.

Readiness Check [Define a separate liveness check](#)

☐ None
☒ TCP connection opens successfully
☐ HTTP request returns a successful status (2xx or 3xx)
☐ HTTPS request returns a successful status (2xx or 3xx)
☐ Command run inside the container exits with status 0

Target Container Port ^{*}

Start Checking After seconds

Check Interval seconds

Check Timeout seconds

Healthy After successes

Unhealthy After failures

Selain dari health cek terdapat fitur autoscaling sehingga dapat membantu jika sewaktu-waktu terjadi lonjakan traffic seperti di bawah ini

Scaling/Upgrade Policy
Configure how pods are replaced when performing an upgrade.

☒ Rolling: start new pods, then stop old
☐ Rolling: stop old pods, then start new
☐ Kill ALL pods, then start new
☐ Custom

Batch Size Pod
Pods will be started and stopped this many at a time

Minimum Ready Time seconds
Containers in the pods must be up for at least this long before the pod is considered available.

Progress Deadline seconds
How long to wait without seeing progress before marking the deployment as stalled.

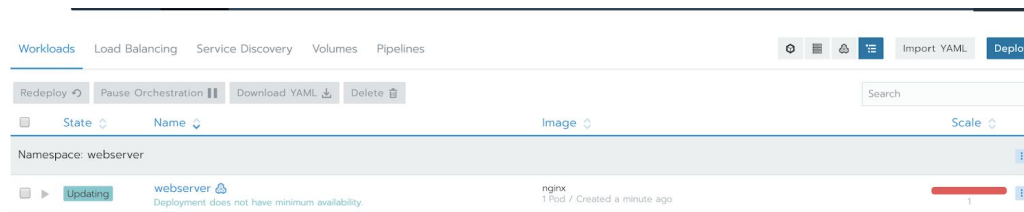
[Show advanced options](#)

[Launch](#) [Cancel](#)

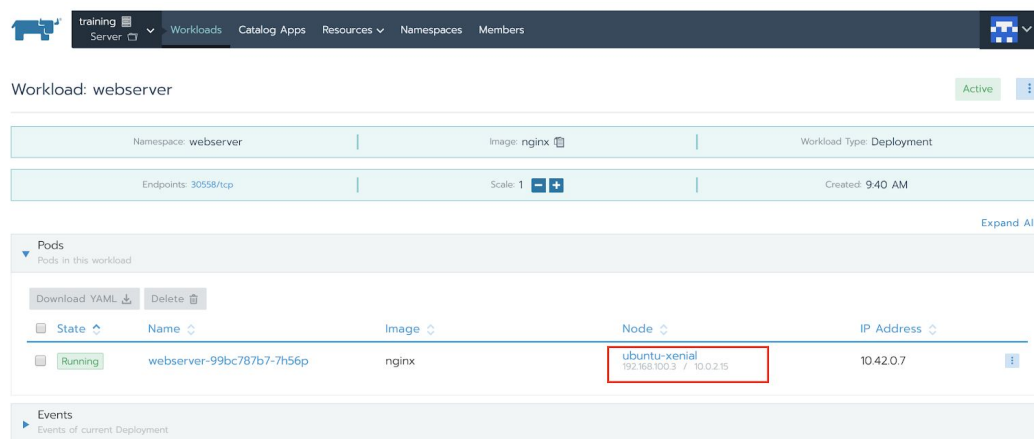
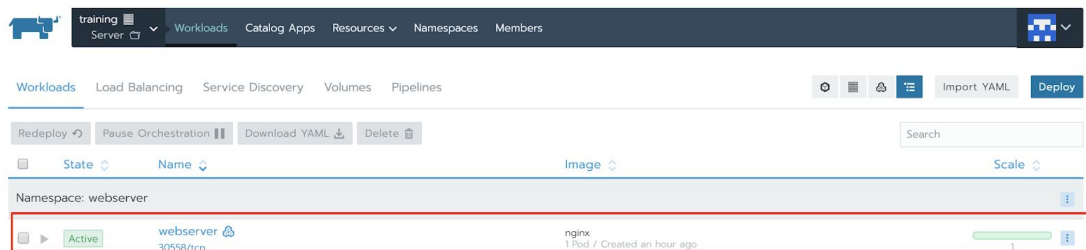
Hal yang perlu di perhatikan adalah Progress Deadline seperti kotak merah di atas, progress deadline menentukan waktu untuk membuat replikasi baru

Setelah semua proses di lakukan selanjutnya klik tombol launch

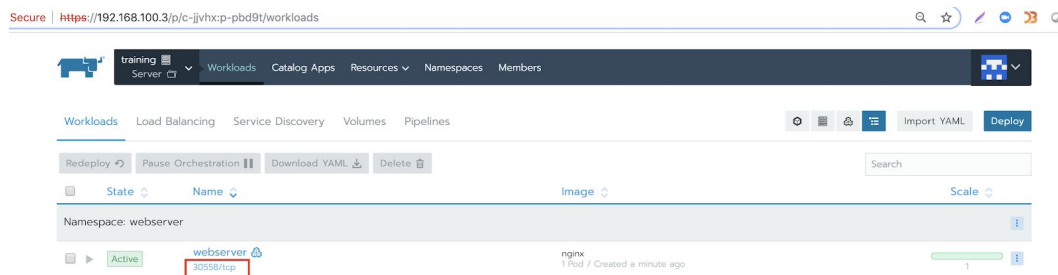
Progress pembuatan workload docker pada kubernetes



Docker yang telah di deploy pada kubernetes dapat di akses dengan port seperti pada kotak merah di bawah ini port 30558 dengan untuk melihat ip address dengan masuk ke docker yang telah di jalan kan seperti di bawah ini



Ip address adalah 192.168.100.3 sehingga aplikasi server dapat di akses menggunakan alamat <http://192.168.100.3:30558>



Tampilan saat di akses adalah sebagai berikut

Not Secure | 192.168.100.3:30558

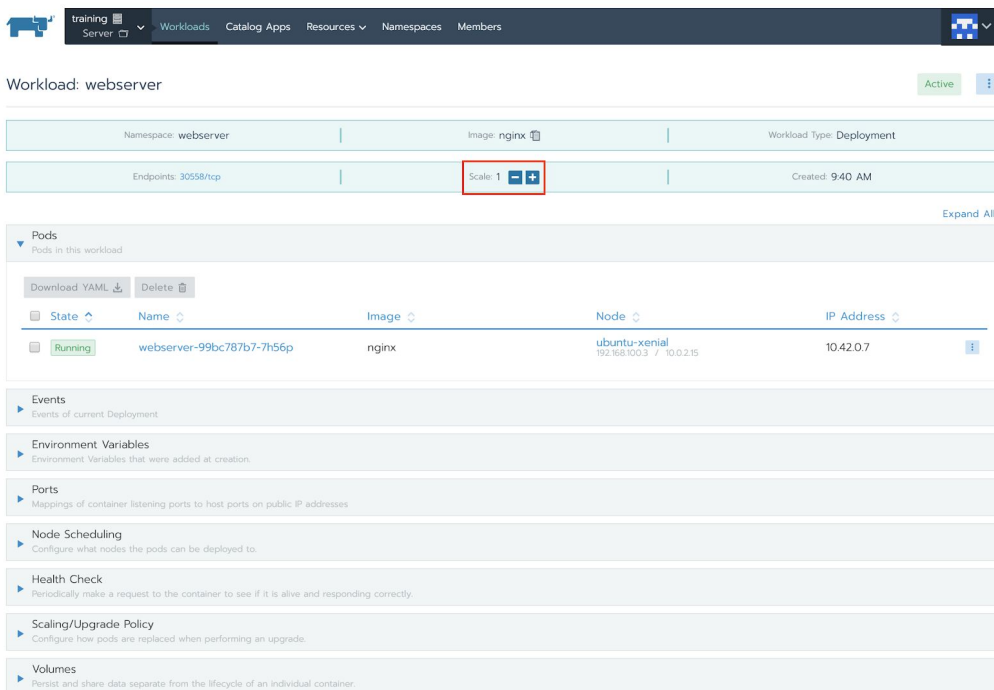
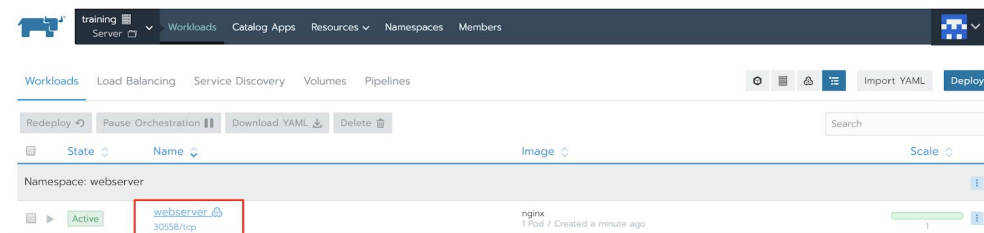
Welcome to nginx!

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.

Kita dapat melihat detail docker dengan klik seperti kotak merah di bawah ini

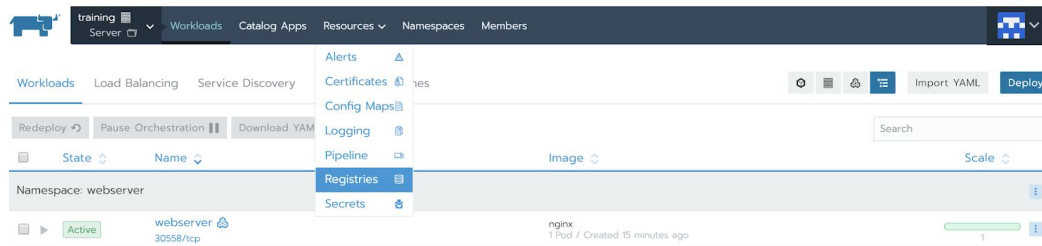


Gambar di atas merupakan detail dari docker yang jalan pada kubernetes, untuk menaikin docker hanya dengan klik kotak merah seperti di atas, Scale berguna saat traffic yang di gunakan sedang tinggi

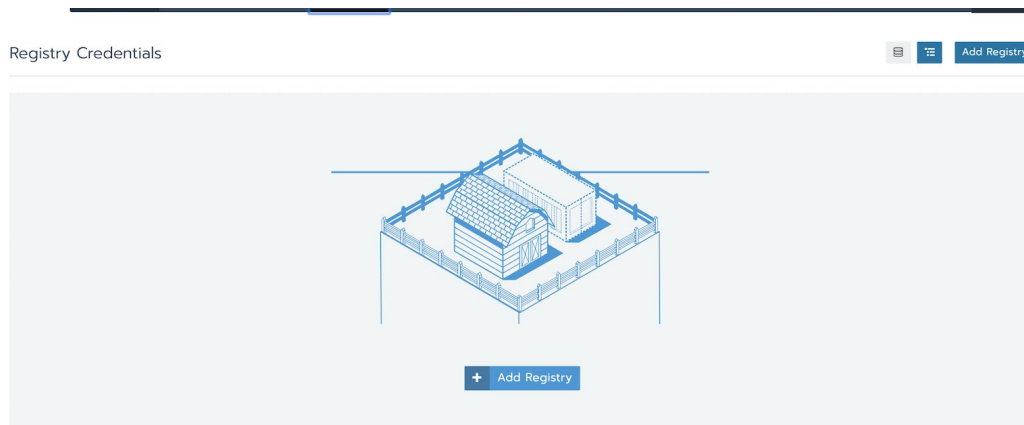
3.2 Deploy Aplikasi dari Docker repository

Pada contoh sebelumnya kita sudah mencoba menjalankan webserver , pada kali ini kita akan mencoba menjalankan aplikasi yang berasal dari image yang kita buat atau build menggunakan docker file, hal ini di lakukan karena proses build aplikasi akan di lakukan saat di lokal atau di server Jenkins

Sebelum kita build aplikasi server maka kita harus mengatur docker registry seperti di bawah ini



Selanjutnya add registry seperti gambar di bawah ini



Maka Selanjutnya akan muncul halaman untuk login

Add Registry

Name Add a Description

dockerregistry

Scope

☒ Available to all namespaces in this project
☐ Available to a single namespace

Address

☒ DockerHub
☐ Quay.io
☐ Custom

Username

adisaputra10

Password

.....

Save Cancel

Kemudian di save

Status docker registry aktif seperti gambar di bawah ini

Registry Credentials Add Registry

Delete

Search

State	Name	Namespace	Address	Username
Namespace: All				
Active	dockerregistry	All	DockerHub	adisaputra10

Langkah selanjutnya adalah membuild source code yang kita miliki menjadi docker images, contoh dapat di downlod pada <https://github.com/adisaputra10/nginx/>

File Dockerfile seperti di bawah ini

```
FROM nginx
```

```
COPY . /usr/share/nginx/html/
```

Selanjutnya pada folder yang sama buat file HTML , seperti di bawah ini


```
<br> <center><strong><h2>Web Server </h2></strong>
</center>
```

Kemudian build dan push dengan perintah **docker build . -t username/image:latest**

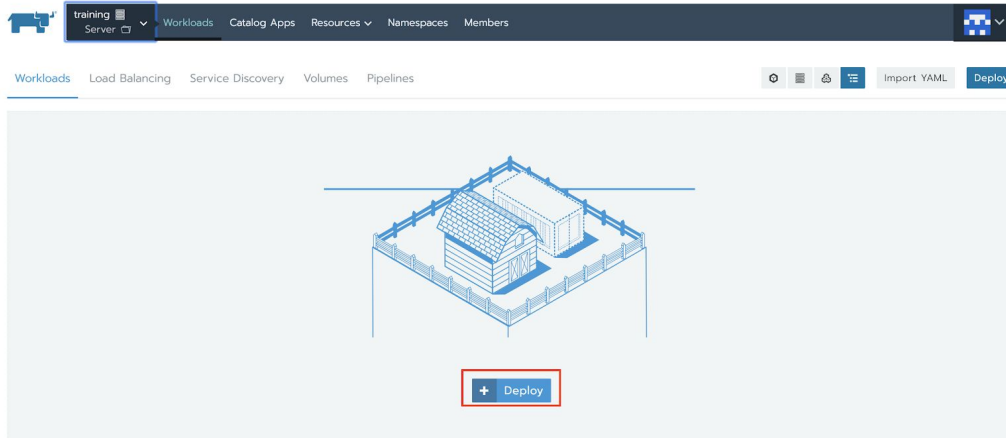
Username menggunakan username yang telah di daftarkan pada hub.docker.com

Contoh seperti ini

```
docker build . -t adisaputra10/nginx
docker push adisaputra10/nginx
```

```
root@ubuntu-xenial:/home/vagrant/nginx# docker build . -t adisaputra10/nginx
Sending build context to Docker daemon 60.42kB
Step 1/2 : FROM nginx
----> 42b4762643dc
Step 2/2 : COPY . /usr/share/nginx/html/
----> Using cache
----> 4a32822cdaf3
Successfully built 4a32822cdaf3
Successfully tagged adisaputra10/nginx:latest
root@ubuntu-xenial:/home/vagrant/nginx# docker push adisaputra10/nginx
The push refers to repository [docker.io/adisaputra10/nginx]
2d58e1ef6c2e: Pushed
89decddf7fb7: Layer already exists
787822cf1b17: Layer already exists
3c816b4ead84: Layer already exists
latest: digest: sha256:b719bfbcb86e603c0ded5bc360d5a875d8cf709b0310a599a63b916c8321c4b4f size: 1157
root@ubuntu-xenial:/home/vagrant/nginx#
```

Selanjutnya jika telah di lakukan kita mencoba untuk deploy ke kubernetes



training Server Workloads Catalog Apps Resources Namespaces Members

Workloads Load Balancing Service Discovery Volumes Pipelines Import YAML Deploy

Deploy Workload

Name * [Add a Description](#)

Workload Type [More options](#)
Scalable deployment of 1 pod

Docker Image * [Add to a new namespace](#)

Namespace *

Port Mapping

Publish the container port * Protocol As a On listening port * [Random](#)

+ Add Port

Perbedaan dari docker yang sebelum nya adalah image, image berasal dari registry yang telah di setup sebelumnya

Untuk health cek juga sama dengan sebelumnya

Health Check
Periodically make a request to the container to see if it is alive and responding correctly.

Readiness Check [Define a separate liveness check](#)

☐ None
☒ TCP connection opens successfully
☐ HTTP request returns a successful status (2xx or 3xx)
☐ HTTPS request returns a successful status (2xx or 3xx)
☐ Command run inside the container exits with status 0

Target Container Port ^{*}

80

Start Checking After

10 seconds

Check Interval

2 seconds

Check Timeout

2 seconds

Healthy After

2 successes

Unhealthy After

3 failures

Untuk scale juga sama dengan pengaturan sebelumnya

Scaling/Upgrade Policy
Configure how pods are replaced when performing an upgrade.

☒ Rolling: start new pods, then stop old
☐ Rolling: stop old pods, then start new
☐ Kill ALL pods, then start new
☐ Custom

Batch Size

1 Pod

Pods will be started and stopped this many at a time

Minimum Ready Time

0 seconds

Containers in the pods must be up for at least this long before the pod is considered available.

Progress Deadline

60 seconds

How long to wait without seeing progress before marking the deployment as stalled.

[Show advanced options](#)

[Launch](#) [Cancel](#)

Proses deploy docker seperti di bawah ini

State	Name	Image	Scale
Active	webserver	nginx	1
Updating	webserverbuild	adisaputra10/nginx	1

Namespace: webserver

Deployment does not have minimum availability.

Setelah deploy berhasil kita dapat mengakses dengan port seperti kotak merah di bawah ini

Active	webserverbuild	adisaputra10/nginx	1
--------	----------------	--------------------	---

32269/tcp

Port 32269 untuk ip dapat kita lihat dengan klik webserverbuild lalu akan muncul seperti gambar di bawah ini

Workload: webserverbuild Active

Namespace: webserver	Image: adisaputra10/nginx	Workload Type: Deployment
Endpoints: 32269/http	Scale: 1	Created: 10:15 AM

Expand All

Pods
Pods in this workload

Download YAML Delete

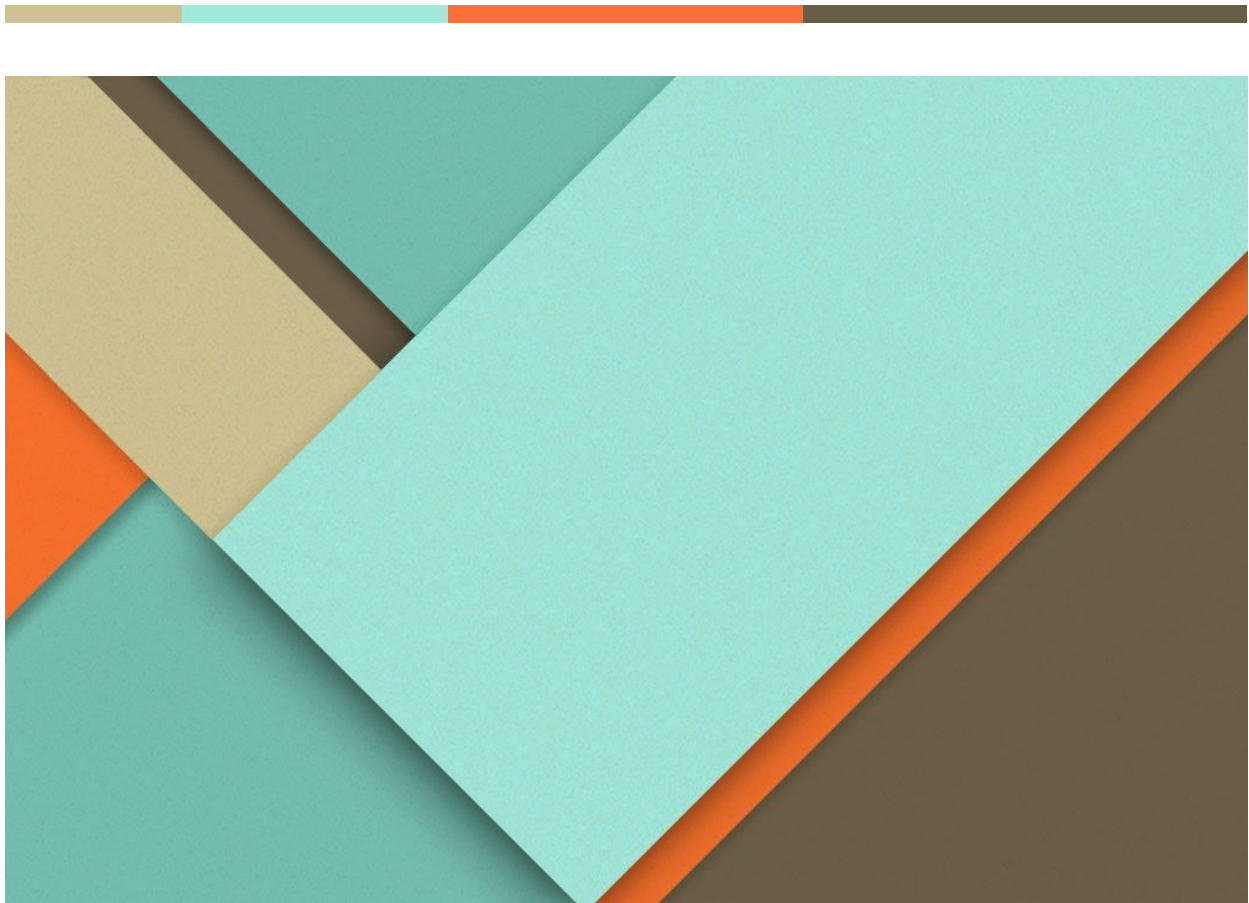
State	Name	Image	Node	IP Address
Running	webserverbuild-58fcb6b5d-lcbrw	adisaputra10/nginx	node1 192.168.100.4 / 10.0.2.15	10.42.3.2

Ip yang di gunakan adalah 192.168.100.4. Sehingga alamat yang bisa di akses adalah 192.168.100.4:32269

Tampilan akan seperti di bawah ini

Not Secure | 192.168.100.4:32269

Web Server



Auto Scaling dan Health Check

Workload: webserverbuild

Namespace: webserver

Image: adisaputra10/nginx

Workload Type: Deployment

Endpoints: 32269/http

Scale: 1

Created: 10:15 AM

Pods

Pods in this workload

Download YAML

Delete

State	Name	Image	Node	IP Address
Running	webserverbuild-58fcbdb5d-1cbrw	adisaputra10/nginx	node1 192.168.100.4 / 10.0.2.15	10.42.3.2

Active

Edit

Clone

Redeploy

Add a Sidecar

Rollback

Execute Shell

Pause Orchestration

View/Edit YAML

View in API

Delete

Akan muncul halaman untuk mengatur health check

Upgrade Service

Name: webserverbuidl [Add a Description](#) Workload Type: Scalable deployment of 1 pod

Docker Image: Namespace: webserver

Port Mapping: Publish the container port: Protocol: TCP As a: NodePort (On every node) On listening port: Random

[+ Add Port](#)

[Expand All](#)

- Environment Variables: Set the environment that will be visible to the container, including injecting values from other resources like Secrets.
- Node Scheduling: Configure what nodes the pods can be deployed to.
- Health Check: Periodically make a request to the container to see if it is alive and responding correctly.**
- Volumes: Persist and share data separate from the lifecycle of an individual container.
- Scaling/Upgrade Policy: Configure how pods are replaced when performing an upgrade.

[Show advanced options](#)

[Upgrade](#) [Cancel](#)

Health Check
Periodically make a request to the container to see if it is alive and responding correctly.

Readiness Check		Liveness Check	
<input type="radio"/> None <input checked="" type="radio"/> TCP connection opens successfully <input type="radio"/> HTTP request returns a successful status (2xx or 3xx) <input type="radio"/> HTTPS request returns a successful status (2xx or 3xx) <input type="radio"/> Command run inside the container exits with status 0		<input type="radio"/> None <input checked="" type="radio"/> TCP connection opens successfully <input type="radio"/> HTTP request returns a successful status (2xx or 3xx) <input type="radio"/> HTTPS request returns a successful status (2xx or 3xx) <input type="radio"/> Command run inside the container exits with status 0	
Target Container Port: <input type="text" value="80"/>	Start Checking After: <input type="text" value="10"/> seconds	Target Container Port: <input type="text" value="80"/>	Start Checking After: <input type="text" value="10"/> seconds
Check Interval: <input type="text" value="2"/> seconds	Check Timeout: <input type="text" value="2"/> seconds	Check Interval: <input type="text" value="2"/> seconds	Check Timeout: <input type="text" value="2"/> seconds
Healthy After: <input type="text" value="2"/> successes	Unhealthy After: <input type="text" value="3"/> failures	Healthy After: <input type="text" value="1"/>	Unhealthy After: <input type="text" value="3"/> failures

Jika di lihat dari gambar di atas, maka port dari docker akan di cek terus-menerus selama 2 detik, port yang di cek adalah port 80, selain itu akan ada pengecekan jika time out dalam 2 detik akan di nyatakan unhealthy , jumlah unhealthy adalah sebanyak 3 kali, jika 3 kali status unhealthy maka kubernetes akan memberikan perintah pada auto scaling untuk mentrigger membuat docker baru

Di sisi sebelah kanan dan kiri sebenarnya sama saja

Penjelasan Autto Scaling

Scaling/Upgrade Policy

Configure how pods are replaced when performing an upgrade.

☒ Rolling: start new pods, then stop old

☐ Rolling: stop old pods, then start new

☐ Kill ALL pods, then start new

☐ Custom

Batch Size

1

Pod

Pods will be started and stopped this many at a time

Minimum Ready Time

0

seconds

Containers in the pods must be up for at least this long before the pod is considered available.

Progress Deadline

60

seconds

How long to wait without seeing progress before marking the deployment as stalled.

Show advanced options

Upgrade

Cancel

Untuk auto scaling proses pembuatan docker yang baru di tentukan pada Scaling / Upgrade Policy. Proses auto scaling adalah akan mencoba membuat docker baru baru selanjutnya docker lama akan langsung di delete, Scaling / Upgarde Policy ini berguna juga untuk proses deploy baru, sehingga aplikasi yang lama tetap akan jalan hingga benar-benar aplikasi baru dapat di gunakan