Jupyter Notebook on Kubernetes

Setting up Kubernetes on Google Cloud

Google Kubernetes Engine (GKE) is the simplest and most common way of setting up a Kubernetes Cluster. You may be able to receive free credits for trying it out. You will need to connect your credit card or other payment method to your google cloud account.

1. Go to `https://console.cloud.google.com` and log in.

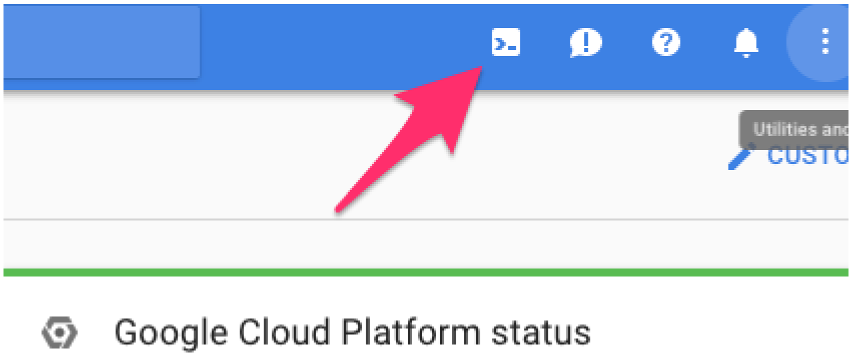
   2. Enable the Kubernetes Engine API.

   3. Use your preferred command line interface.

You have two options: a) use the Google Cloud Shell (no installation needed) or b) install and use the gcloud command-line tool. If you are unsure which to choose, we recommend beginning with option "a" and using the Google Cloud Shell. Instructions for each are detailed below:

1. Use the Google Cloud Shell. Start the Google Cloud Shell

   by clicking the button shown below. This will start an interactive shell session within Google Cloud.



See the Google Cloud Shell docs for more information.

2. Install and use the gcloud command line tool. This tool sends commands to Google Cloud and lets you do things like create and delete clusters.

- Go to the gcloud command line tool downloads page to download and install the gcloud command line tool.

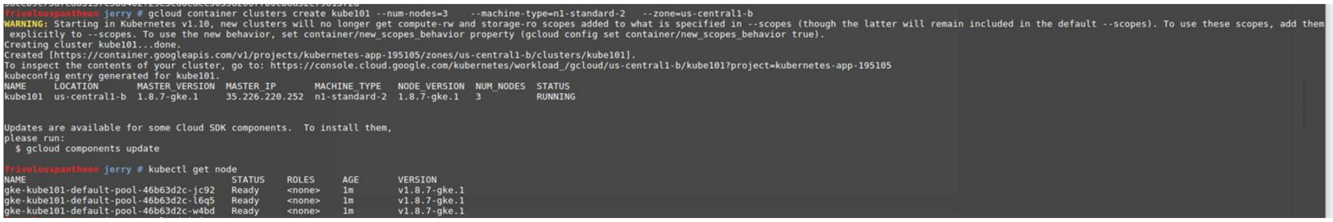
  - See the gcloud documentation for more information on the gcloud command line tool.

4. Install `kubectl`, which is a tool for controlling kubernetes. From the terminal, enter:

gcloud components install kubectl

5. Create a Kubernetes cluster on Google Cloud, by typing the following command into either the Google Cloud shell or the gcloud command-line tool:

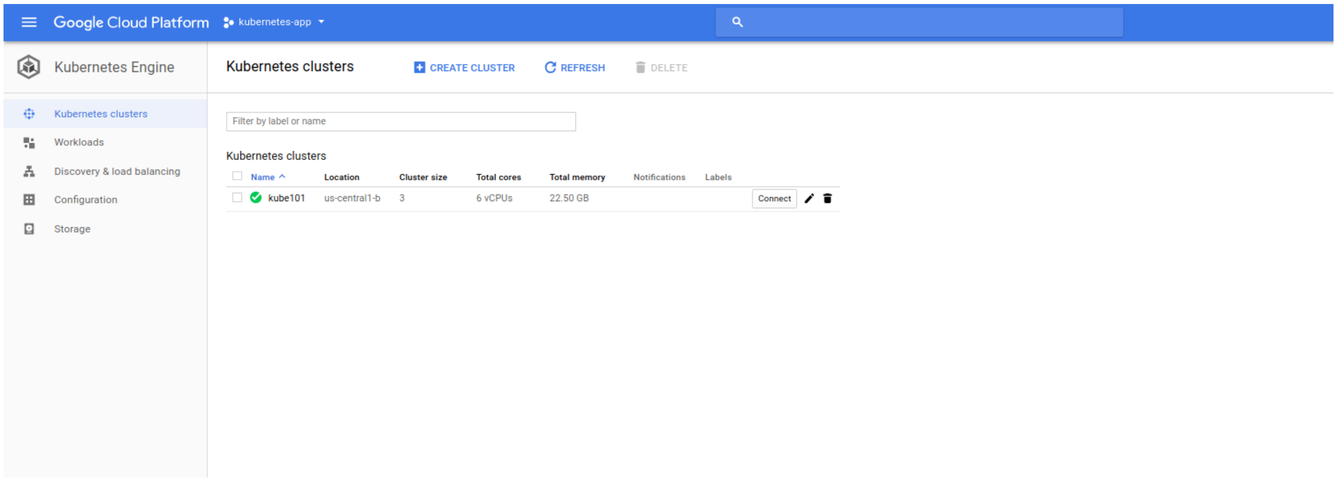
```
gcloud container clusters create kube101 --num-nodes=3    --machine-type=n1-standard-2   --zone=us-central1-b
```



```
frivolouspantheon jerry # gcloud container clusters create kube101 --num-nodes=3    --machine-type=n1-standard-2   --zone=us-central1-b
WARNING: Starting in Kubernetes v1.10, new clusters will no longer get compute-rw and storage-ro scopes added to what is specified in --scopes (though the latter will remain included in the default --scopes). To use these scopes, add them
 explicitly to --scopes. To use the new behavior, set container/new_scopes_behavior property (gcloud config set container/new_scopes_behavior true).
Creating cluster kube101...done.
Created [https://container.googleapis.com/v1/projects/kubernetes-app-195105/zones/us-central1-b/clusters/kube101].
To inspect the contents of your cluster, go to: https://console.cloud.google.com/kubernetes/workload_/gcloud/us-central1-b/kube101?project=kubernetes-app-195105
kubeconfig entry generated for kube101.
NAME     LOCATION       MASTER_VERSION  MASTER_IP       MACHINE_TYPE   NODE_VERSION  NUM_NODES  STATUS
kube101  us-central1-b  1.8.7-gke.1     35.226.220.252  n1-standard-2  1.8.7-gke.1   3          RUNNING


Updates are available for some Cloud SDK components.  To install them,
please run:
  $ gcloud components update

frivolouspantheon jerry # kubectl get node
NAME                                     STATUS    ROLES     AGE       VERSION
gke-kube101-default-pool-46b63d2c-jc92   Ready     <none>    1m        v1.8.7-gke.1
gke-kube101-default-pool-46b63d2c-l6q5   Ready     <none>    1m        v1.8.7-gke.1
gke-kube101-default-pool-46b63d2c-w4bd   Ready     <none>    1m        v1.8.7-gke.1
```

--machine-type specifies the amount of CPU and RAM in each node. There is a variety of types to choose from. Picking something appropriate here will have a large effect on how much you pay - smaller machines restrict the max amount of RAM each user can have access to but allow more fine-grained scaling, reducing cost. The default (n1-standard-2) has 2CPUs and 7.5G of RAM each, and might not be a good fit for all use cases!

- --zone specifies which data center to use. Pick something that is not too far away from your users. You can find a list of them here.

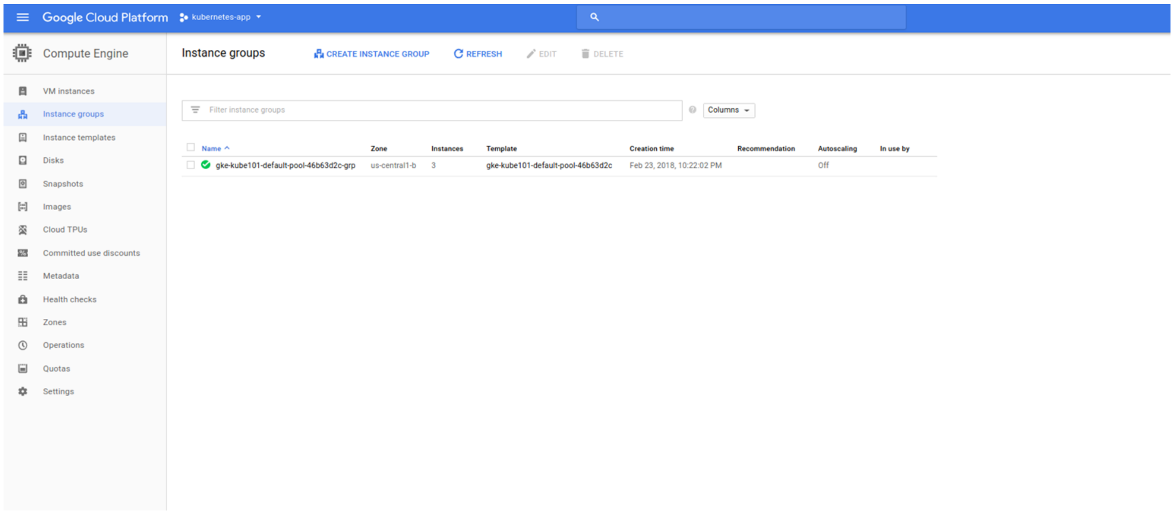


Kubernetes clusters

□	Name	Location	Cluster size	Total cores	Total memory	Notifications	Labels	
□	✓ kube101	us-central1-b	3	6 vCPUs	22.50 GB			Connect

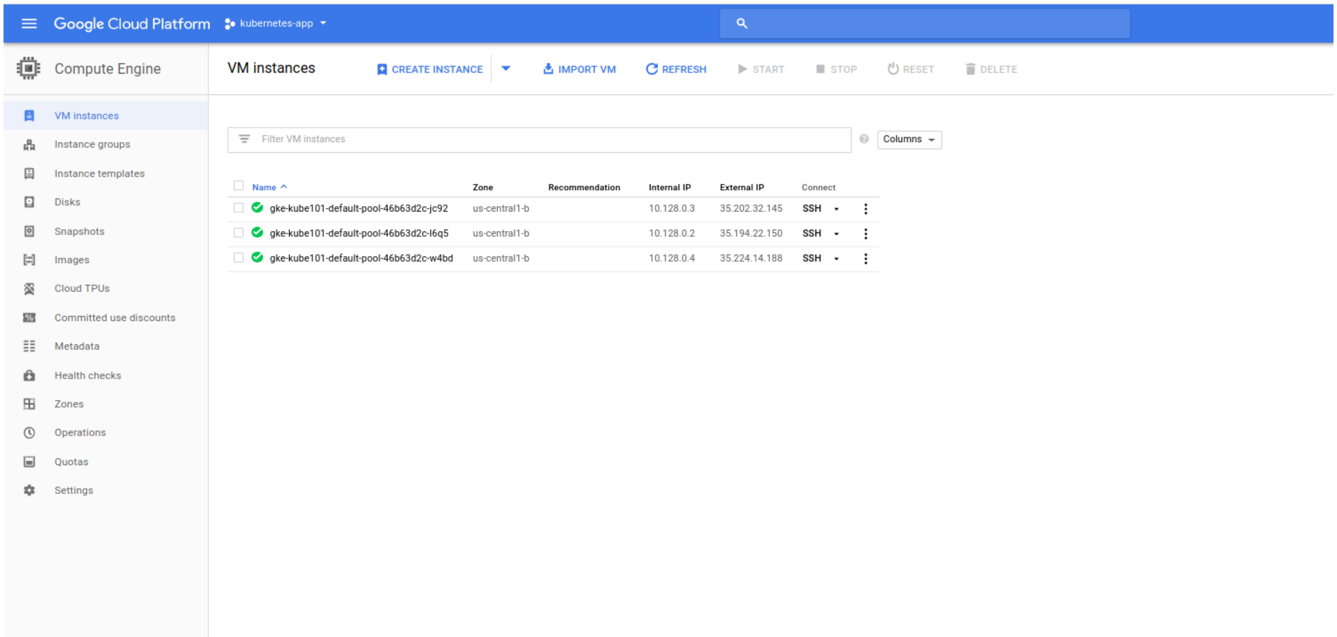
2. To test if your cluster is initialized, run:

kubectl get node



Name	Zone	Instances	Template	Creation time	Recommendation	Autoscaling	In use by
gke-kube101-default-pool-46b63d2c-grp	us-central1-b	3	gke-kube101-default-pool-46b63d2c	Feb 23, 2018, 10:22:02 PM		Off	

The response should list three running nodes.



Name	Zone	Recommendation	Internal IP	External IP	Connect
gke-kube101-default-pool-46b63d2c-jc92	us-central1-b		10.128.0.3	35.202.32.145	SSH
gke-kube101-default-pool-46b63d2c-l6q5	us-central1-b		10.128.0.2	35.194.22.150	SSH
gke-kube101-default-pool-46b63d2c-w4bd	us-central1-b		10.128.0.4	35.224.14.188	SSH

Ref :http://zero-to-jupyterhub.readthedocs.io/en/latest/create-k8s-cluster.html#create-k8s-cluster

Setting up Helm

Helm, the package manager for Kubernetes, is a useful tool to install, upgrade and manage applications on a Kubernetes cluster. We will be using Helm to install and manage JupyterHub on our cluster.

Installation

The simplest way to install helm is to run Helm’s installer script at a terminal:

```
curl https://raw.githubusercontent.com/kubernetes/helm/master/scripts/get | bash
```

Initialization

After installing helm on your machine, initialize helm on your Kubernetes cluster. At the terminal, enter:

1. Set up a ServiceAccount for use by Tiller, the server side component of helm.

```
kubectl --namespace kube-system create serviceaccount jerry
```

2. Give the ServiceAccount RBAC full permissions to manage the cluser.

While most clusters have RBAC enabled and you need this line, you must skip this step if your kubernetes cluster does not have RBAC enabled (for example, if you are using Azure AKS).

```
kubectl create clusterrolebinding jerry --clusterrole cluster-admin –serviceaccount=kube-system:jerry
```

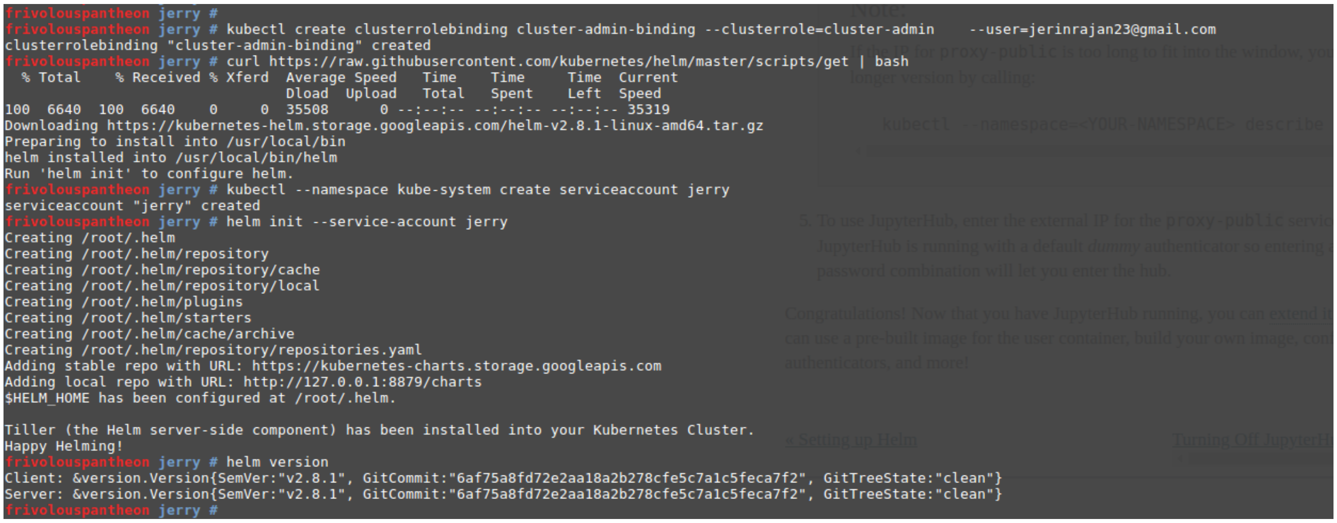
3. Set up Helm on the cluster.

```
helm init --service-account jerry
```

4.Verify

You can verify that you have the correct version and that it installed properly by running:

```
helm version
```



```
frivolouspantheon jerry #
frivolouspantheon jerry # kubectl create clusterrolebinding cluster-admin-binding --clusterrole=cluster-admin    --user=jerinrajan23@gmail.com
clusterrolebinding "cluster-admin-binding" created
frivolouspantheon jerry # curl https://raw.githubusercontent.com/kubernetes/helm/master/scripts/get | bash
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  6640  100  6640    0     0  35508      0 --:--:-- --:--:-- --:--:-- 35319
Downloading https://kubernetes-helm.storage.googleapis.com/helm-v2.8.1-linux-amd64.tar.gz
Preparing to install into /usr/local/bin
helm installed into /usr/local/bin/helm
Run 'helm init' to configure helm.
frivolouspantheon jerry # kubectl --namespace kube-system create serviceaccount jerry
serviceaccount "jerry" created
frivolouspantheon jerry # helm init --service-account jerry
Creating /root/.helm
Creating /root/.helm/repository
Creating /root/.helm/repository/cache
Creating /root/.helm/repository/local
Creating /root/.helm/plugins
Creating /root/.helm/starters
Creating /root/.helm/cache/archive
Creating /root/.helm/repository/repositories.yaml
Adding stable repo with URL: https://kubernetes-charts.storage.googleapis.com
Adding local repo with URL: http://127.0.0.1:8879/charts
$HELM_HOME has been configured at /root/.helm.

Tiller (the Helm server-side component) has been installed into your Kubernetes Cluster.
Happy Helming!
frivolouspantheon jerry # helm version
Client: &version.Version{SemVer:"v2.8.1", GitCommit:"6af75a8fd72e2aa18a2b278cfe5c7a1c5feca7f2", GitTreeState:"clean"}
Server: &version.Version{SemVer:"v2.8.1", GitCommit:"6af75a8fd72e2aa18a2b278cfe5c7a1c5feca7f2", GitTreeState:"clean"}
frivolouspantheon jerry #
```

Prepare configuration file

This step prepares a configuration file (config file). We will use the YAML file format to specify JupyterHub's configuration.

It's important to save the config file in a safe place. The config file is needed for future changes to JupyterHub's settings.

For the following steps, use your favorite code editor. We'll use the nano editor as an example.

1. Create a file called config.yaml. Using the nano editor, for example, entering nanoconfig.yaml at the terminal will start the editor and open the config file.

2. Create a random hex string to use as a security token. Run this command in a terminal

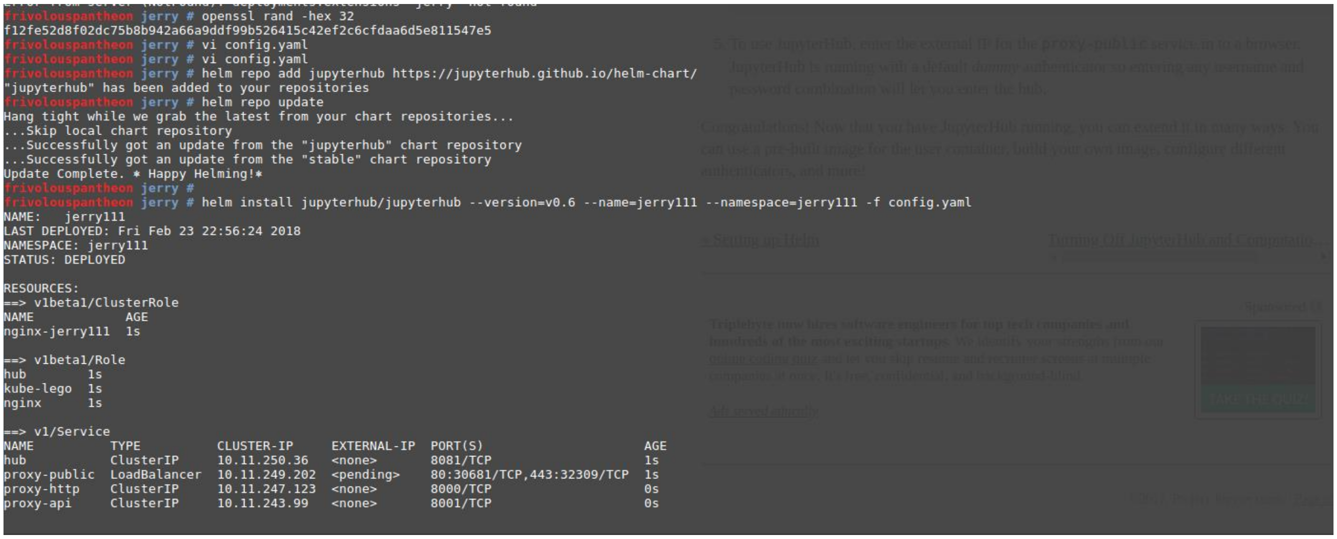
```
openssl rand -hex 32
```

Copy the output for use in the next step

3. Insert these lines into the config.yaml file. When editing YAML files, use straight quotes and spaces and avoid using curly quotes or tabs. Substitute RANDOM_STRING below with the output of openssl rand -hex 32 from step 2.

```
vi config.yaml
```

```
proxy:

secretToken: 'f12fe52d8f02dc75b8b942a66a9ddf99b526415c42ef2c6cfdaa6d5e811547e5'
```



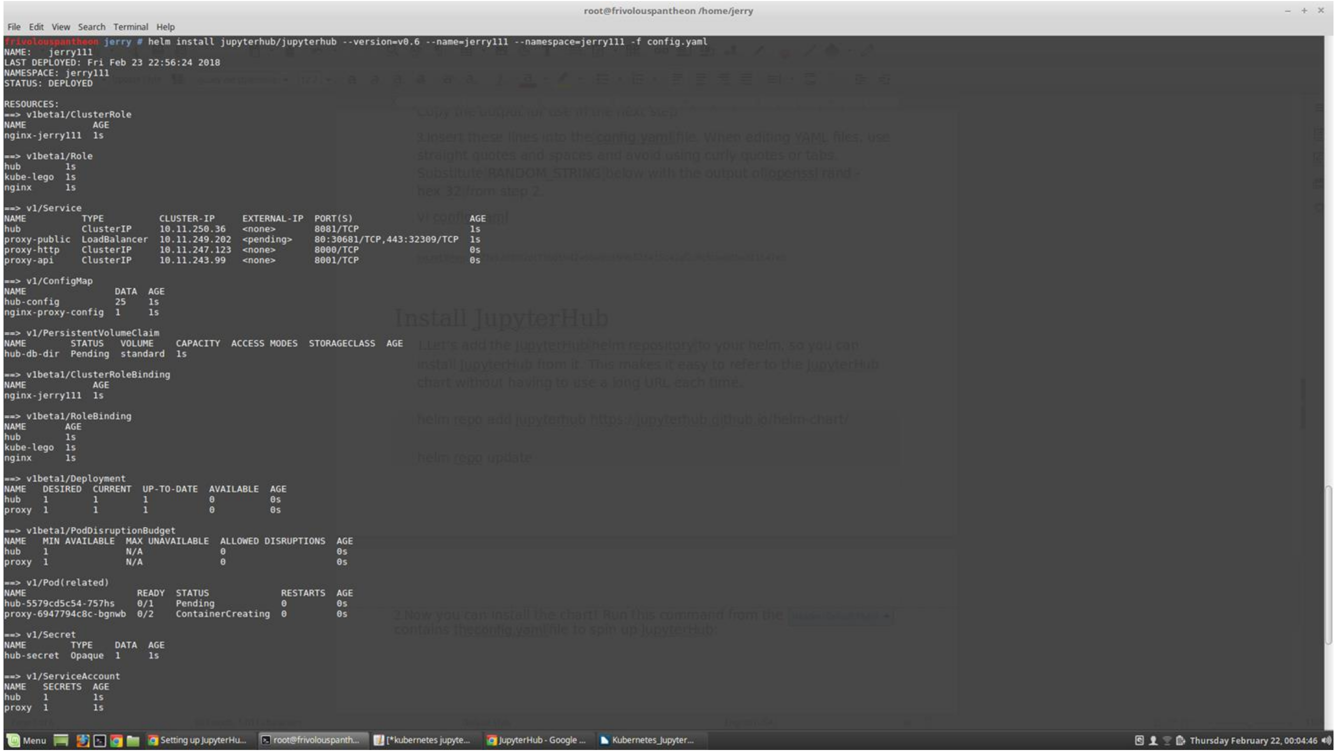
```
Error from server (NotFound): deployments.extensions "jerry" not found
frivolouspantheon jerry # openssl rand -hex 32
f12fe52d8f02dc75b8b942a66a9ddf99b526415c42ef2c6cfdaa6d5e811547e5
frivolouspantheon jerry # vi config.yaml
frivolouspantheon jerry # vi config.yaml
frivolouspantheon jerry # helm repo add jupyterhub https://jupyterhub.github.io/helm-chart/
"jupyterhub" has been added to your repositories
frivolouspantheon jerry # helm repo update
Hang tight while we grab the latest from your chart repositories...
...Skip local chart repository
...Successfully got an update from the "jupyterhub" chart repository
...Successfully got an update from the "stable" chart repository
Update Complete. * Happy Helming!*
frivolouspantheon jerry #
frivolouspantheon jerry # helm install jupyterhub/jupyterhub --version=v0.6 --name=jerry111 --namespace=jerry111 -f config.yaml
NAME:   jerry111
LAST DEPLOYED: Fri Feb 23 22:56:24 2018
NAMESPACE: jerry111
STATUS: DEPLOYED

RESOURCES:
==> v1beta1/ClusterRole
NAME            AGE
nginx-jerry111  1s

==> v1beta1/Role
hub        1s
kube-lego  1s
nginx      1s

==> v1/Service
NAME          TYPE          CLUSTER-IP     EXTERNAL-IP  PORT(S)                     AGE
hub           ClusterIP     10.11.250.36   <none>       8081/TCP                    1s
proxy-public  LoadBalancer  10.11.249.202  <pending>    80:30681/TCP,443:32309/TCP  1s
proxy-http    ClusterIP     10.11.247.123  <none>       8000/TCP                    0s
proxy-api     ClusterIP     10.11.243.99   <none>       8001/TCP                    0s
```

Install JupyterHub

1. Let's add the JupyterHub helm repository to your helm, so you can install JupyterHub from it. This makes it easy to refer to the JupyterHub chart without having to use a long URL each time.

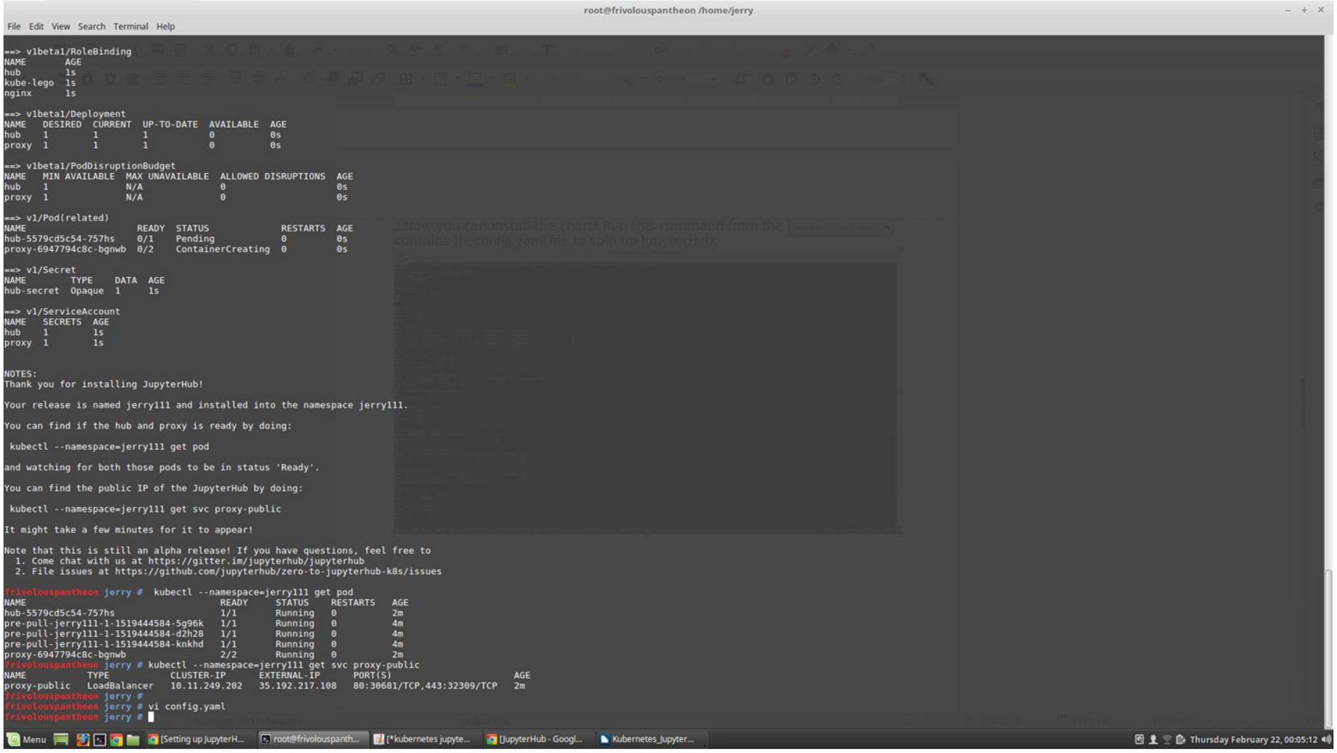
helm repo add jupyterhub https://jupyterhub.github.io/helm-chart/

helm repo update

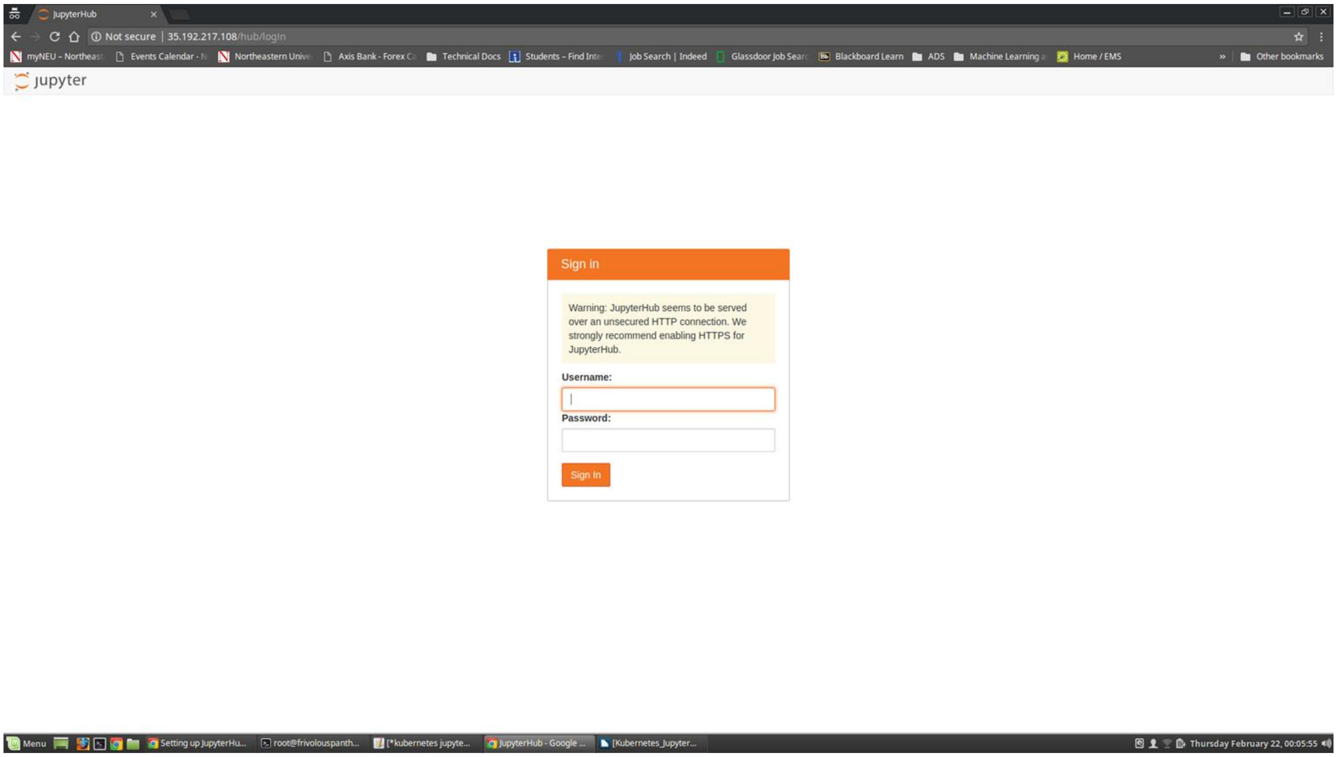


```
frivolouspantheon jerry # helm install jupyterhub/jupyterhub --version=v0.6 --name=jerry111 --namespace=jerry111 -f config.yaml
NAME:   jerry111
LAST DEPLOYED: Fri Feb 23 22:56:24 2018
NAMESPACE: jerry111
STATUS: DEPLOYED

RESOURCES:
==> v1beta1/ClusterRole
NAME            AGE
nginx-jerry111  1s

==> v1beta1/Role
hub        1s
kube-lego  1s
nginx      1s

==> v1/Service
NAME          TYPE          CLUSTER-IP     EXTERNAL-IP  PORT(S)                       AGE
hub           ClusterIP     10.11.250.36   <none>       8081/TCP                      1s
proxy-public  LoadBalancer  10.11.249.202  <pending>    80:30681/TCP,443:32309/TCP  1s
proxy-http    ClusterIP     10.11.247.123  <none>       8000/TCP                      0s
proxy-api     ClusterIP     10.11.243.99   <none>       8001/TCP                      0s

==> v1/ConfigMap
NAME                DATA  AGE
hub-config          25    1s
nginx-proxy-config  1     1s

==> v1/PersistentVolumeClaim
NAME        STATUS   VOLUME    CAPACITY  ACCESS MODES  STORAGECLASS  AGE
hub-db-dir  Pending  standard  1s

==> v1beta1/ClusterRoleBinding
NAME            AGE
nginx-jerry111  1s

==> v1beta1/RoleBinding
NAME       AGE
hub        1s
kube-lego  1s
nginx      1s

==> v1beta1/Deployment
NAME   DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
hub    1        1        1           0          0s
proxy  1        1        1           0          0s

==> v1beta1/PodDisruptionBudget
NAME   MIN AVAILABLE  MAX UNAVAILABLE  ALLOWED DISRUPTIONS  AGE
hub    1              N/A              0                    0s
proxy  1              N/A              0                    0s

==> v1/Pod(related)
NAME                      READY  STATUS             RESTARTS  AGE
hub-5579cd5c54-757hs      0/1    Pending            0         0s
proxy-6947794c8c-bgnwb  0/2    ContainerCreating  0         0s

==> v1/Secret
NAME        TYPE    DATA  AGE
hub-secret  Opaque  1     1s

==> v1/ServiceAccount
NAME   SECRETS  AGE
hub    1        1s
proxy  1        1s
```

2.Now you can install the chart! Run this command from the directory that contains theconfig.yaml file to spin up JupyterHub:

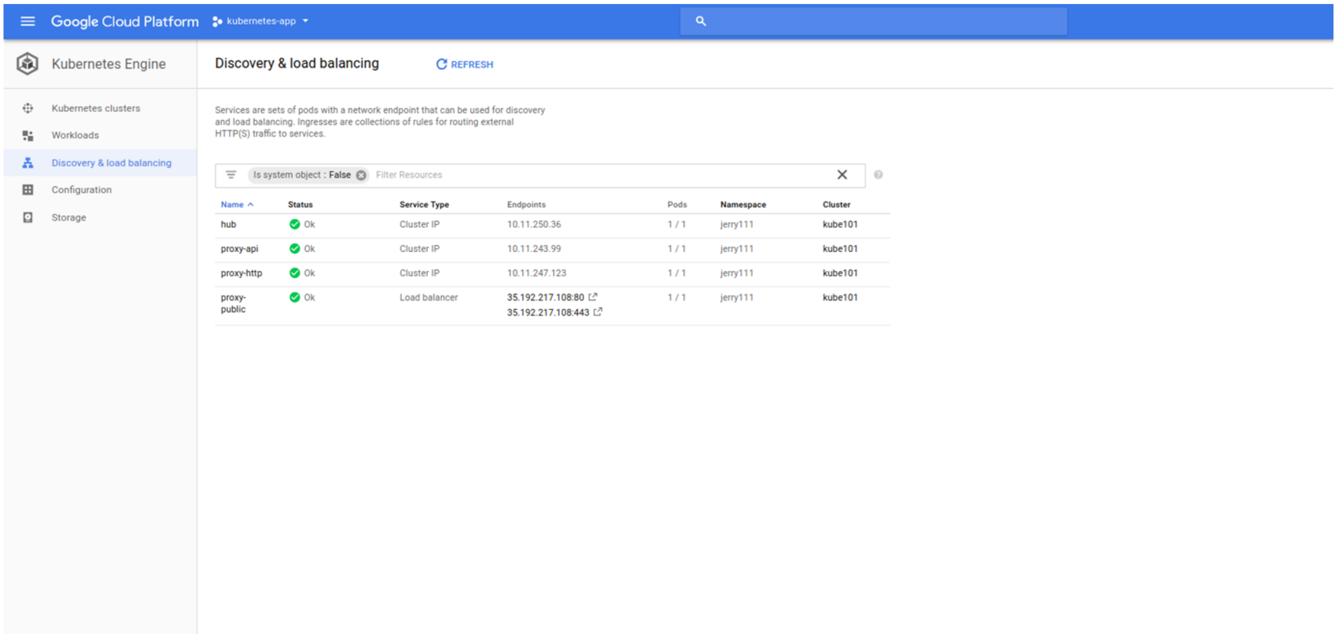


```
==> v1beta1/RoleBinding
NAME       AGE
hub        1s
kube-lego  1s
nginx      1s

==> v1beta1/Deployment
NAME   DESIRED  CURRENT  UP-TO-DATE  AVAILABLE  AGE
hub    1        1        1           0          0s
proxy  1        1        1           0          0s

==> v1beta1/PodDisruptionBudget
NAME   MIN AVAILABLE  MAX UNAVAILABLE  ALLOWED DISRUPTIONS  AGE
hub    1              N/A              0                    0s
proxy  1              N/A              0                    0s

==> v1/Pod(related)
NAME                      READY  STATUS             RESTARTS  AGE
hub-5579cd5c54-757hs      0/1    Pending            0         0s
proxy-6947794c8c-bgnwb  0/2    ContainerCreating  0         0s

==> v1/Secret
NAME        TYPE    DATA  AGE
hub-secret  Opaque  1     1s

==> v1/ServiceAccount
NAME   SECRETS  AGE
hub    1        1s
proxy  1        1s


NOTES:
Thank you for installing JupyterHub!

Your release is named jerry111 and installed into the namespace jerry111.

You can find if the hub and proxy is ready by doing:

 kubectl --namespace=jerry111 get pod

and watching for both those pods to be in status 'Ready'.

You can find the public IP of the JupyterHub by doing:

 kubectl --namespace=jerry111 get svc proxy-public

It might take a few minutes for it to appear!

Note that this is still an alpha release! If you have questions, feel free to
  1. Come chat with us at https://gitter.im/jupyterhub/jupyterhub
  2. File issues at https://github.com/jupyterhub/zero-to-jupyterhub-k8s/issues

frivolouspantheon jerry #  kubectl --namespace=jerry111 get pod
NAME                                     READY     STATUS    RESTARTS   AGE
hub-5579cd5c54-757hs                     1/1       Running   0          2m
pre-pull-jerry111-1-1519444584-5g96k   1/1       Running   0          4m
pre-pull-jerry111-1-1519444584-d2h28   1/1       Running   0          4m
pre-pull-jerry111-1-1519444584-knkhd   1/1       Running   0          4m
proxy-6947794c8c-bgnwb                   2/2       Running   0          2m
frivolouspantheon jerry # kubectl --namespace=jerry111 get svc proxy-public
NAME           TYPE           CLUSTER-IP      EXTERNAL-IP      PORT(S)                       AGE
proxy-public   LoadBalancer   10.11.249.202   35.192.217.108   80:30681/TCP,443:32309/TCP   2m
frivolouspantheon jerry #
frivolouspantheon jerry # vi config.yaml
frivolouspantheon jerry # █
```



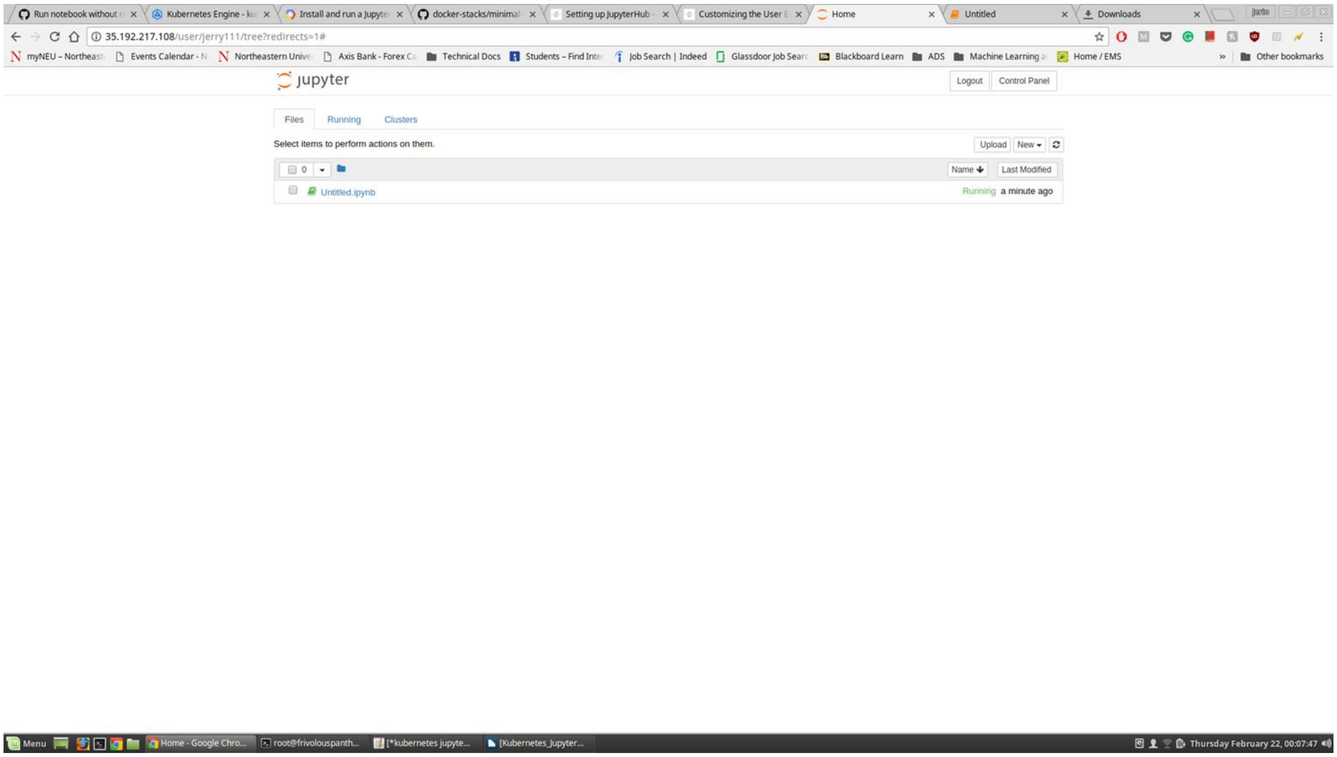
Warning: JupyterHub seems to be served over an unsecured HTTP connection. We strongly recommend enabling HTTPS for JupyterHub.

Enter Username : jerry111 -→{namespace or identity}



Services are sets of pods with a network endpoint that can be used for discovery and load balancing. Ingresses are collections of rules for routing external HTTP(S) traffic to services.

Name ^	Status	Service Type	Endpoints	Pods	Namespace	Cluster
hub	✓ Ok	Cluster IP	10.11.250.36	1 / 1	jerry111	kube101
proxy-api	✓ Ok	Cluster IP	10.11.243.99	1 / 1	jerry111	kube101
proxy-http	✓ Ok	Cluster IP	10.11.247.123	1 / 1	jerry111	kube101
proxy-public	✓ Ok	Load balancer	35.192.217.108:80 35.192.217.108:443	1 / 1	jerry111	kube101

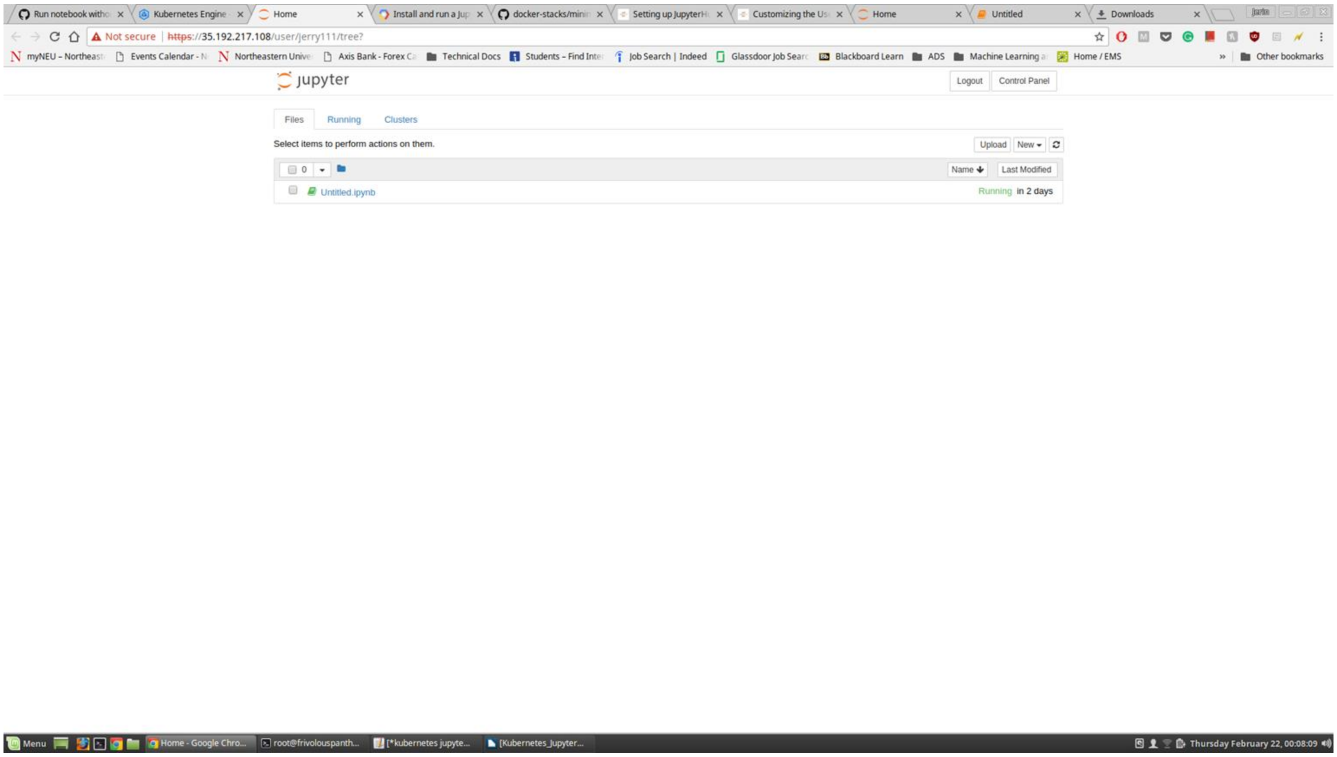


35.192.217.108/user/jerry111/tree?redirects=1#

Select items to perform actions on them.

Untitled.ipynb Running a minute ago

Thursday February 22, 00:07:47



Not secure | https://35.192.217.108/user/jerry111/tree?

Select items to perform actions on them.

Untitled.ipynb Running in 2 days

Thursday February 22, 00:08:09

Ref: http://zero-to-jupyterhub.readthedocs.io/en/latest/setup-jupyterhub.html#setup-jupyterhub

Another method
https://cloud.google.com/dataproc/docs/tutorials/jupyter-notebook