Getting connect to the cluster

1. Install Helm 3 and kubectl. Latest versions are fine
    a. https://helm.sh/docs/intro/install/
    b. https://kubernetes.io/docs/tasks/tools/install-kubectl-macos/
2. Login to the purdue VPN to reach geddes
3. Login to geddes and navigate to the geddes cluster with the top left dropdown
4. Click **kubeconfig File** at the top left and copy context to clipboard
5. Create a directory called **.kube** in your homedir (make sure to include the .)
6. Create a file called config in the new .**kube** directory called **config** and copy the kubectl file context here
7. Test to see if kubectl is properly connected to the cluster with the command **kubectl get nodes**

Launching Jupyterhub

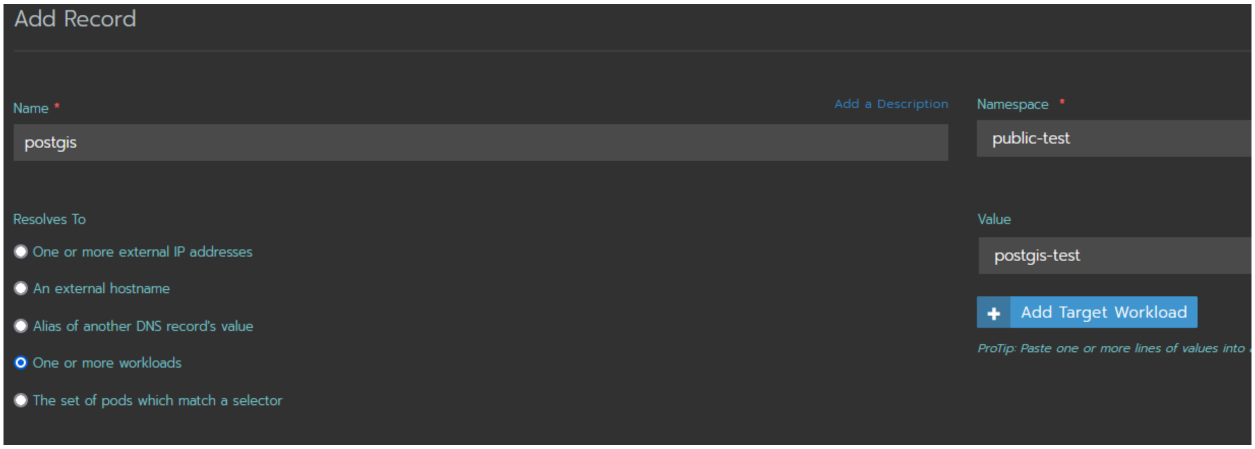
1. Create Namespace to deploy Jupyterhub to (remember that the name of your namespace will be part of your doman name in the format **<service-name>.<namespace>.geddes.rcac.purdue.edu** so name appropriately)
    a. In the Rancher UI Navigate to your ironhacks project using the top left dropdown
    b. At the top click **namespaces**
    c. At top left click **add namespace**
    d. Name and create, leave everything else default
2. Copy context into a file called config.yaml

```
singleuser:
  image:
    name: geddes-registry.rcac.purdue.edu/docker-hub-cache/jupyter/datascience-notebook
    tag: latest
  networkTools:
    image:
      name: geddes-registry.rcac.purdue.edu/docker-hub-cache/jupyterhub/k8s-network-tools

hub:
  image:
    name: geddes-registry.rcac.purdue.edu/docker-hub-cache/jupyterhub/k8s-hub

proxy:
  chp:
    image:
      name: geddes-registry.rcac.purdue.edu/docker-hub-cache/jupyterhub/configurable-http-proxy
      tag: 4.4.0

prePuller:
  hook:
    image:
      name: geddes-registry.rcac.purdue.edu/docker-hub-cache/jupyterhub/k8s-image-awaiter
```

3. Add the jupyterhub helm repo with command
    a. **helm repo add jupyterhub https://jupyterhub.github.io/helm-chart/**
4. Update repo
    a. **helm repo update**
5. Use the command **helm upgrade --cleanup-on-fail --install <release-name> jupyterhub/jupyterhub --namespace <your-namespace> --values config.yaml** to launch jupyterhub
    a. **<release-name>** is the release name (this is an arbitrary name to keep track of your helm deployment)
    b. **<your namespace>** name of the namespace you just created

Domain Name Generation and Service Creation

1. In the Rancher UI create a service
    a. From the workloads section click on the "service discovery" tab
    b. Click "add record" at the top
    c. Give name
        i. This will act as the service-name portion of the kubernetes dns formatting <service-name>.<namespace>.geddes.rcac.purdue.edu. so in my case my dns name will be postgis.public-test.geddes.rcac.purdue.edu when finished
    d. Chose the namespace you deployed to.
    e. Click on "one or more workloads"
    f. Click on "add target workload"
    g. Select the **proxy** workload deployed with jupyterhub



Add Record

Name * postgis

Namespace * public-test

Resolves To
○ One or more external IP addresses
○ An external hostname
○ Alias of another DNS record's value
● One or more workloads
○ The set of pods which match a selector

Value postgis-test

ProTip: Paste one or more lines of values into

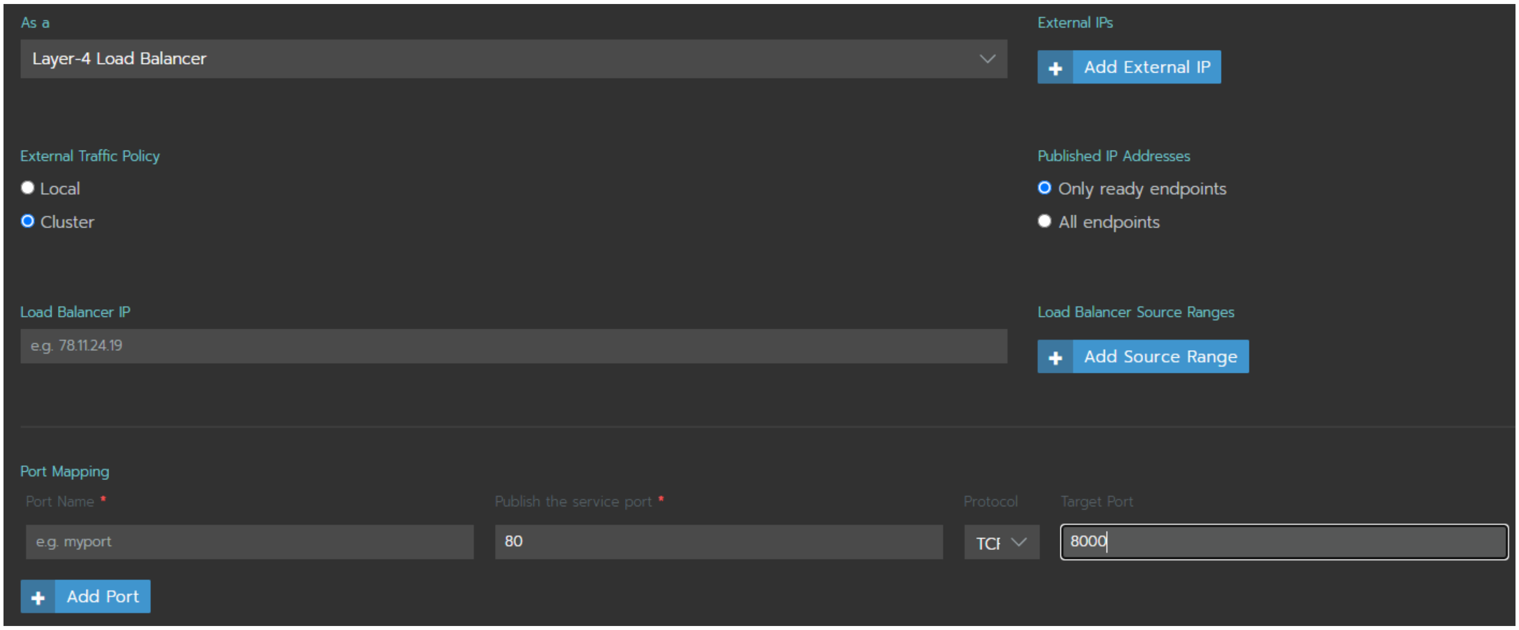
    h. Click on "show advanced options" at the bottum right
        i. It's a bit hard to see, it's above the create/cancel button line ro the right
    i. Select "layer-4 load balancer" in the "as a' dropdown
        i. By default this will give your container a campus private address so it can be reached via the DNS name like a normal server deployment but only internally to Purdue. You can leave all the extra configs default that pop up in this section

ii. If you want to deploy to a public internet addres see below steps **l** and **m**

j. Click “add port”

k. Fill in “80” under “publish the service port and “8000” under target port

i. This says that connection hitting port 80 (default http port) on your service IP routes to your container port 8000 where the proxy server is listening


As a: Layer-4 Load Balancer

External Traffic Policy: ○ Local ● Cluster

Published IP Addresses: ● Only ready endpoints ○ All endpoints

Load Balancer IP: e.g. 78.11.24.19

Port Mapping

Port Name *: e.g. myport

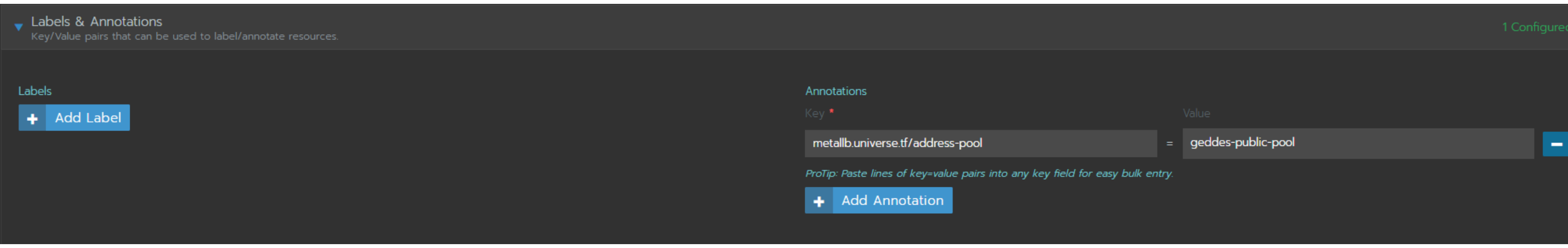
Publish the service port *: 80

Protocol: TCP

Target Port: 8000

l. (optional public address config) click the **labels & annotations** dropdown and click **add annotoation**

m. (optional public address config) copy **metallb.universe.tf/address-pool: geddes-public-pool** into the key box


Labels & Annotations

Key/Value pairs that can be used to label/annotate resources.

Annotations

Key *: metallb.universe.tf/address-pool = Value: geddes-public-pool

ProTip: Paste lines of key=value pairs into any key field for easy bulk entry.

n. Click create

2. You should now be able to navigate to your new service URL **<service-name>.<namespace>.geddes.rcac.purdue.edu** and see the jupyterhub login

Updating jupyterhub with more advanced config/auth

1. Update your current deployment by making changes to your current config.yaml and reuse the same command you used initially to launch jupyterhub from section **Launching Jupyterhub** step **5**

Documentation

Main jupyterhub docs for customizing deployments:
https://zero-to-jupyterhub.readthedocs.io/en/latest/jupyterhub/customization.html

Google auth documentation example:
https://zero-to-jupyterhub.readthedocs.io/en/latest/administrator/authentication.html?highlight=authentication#google

How to customize notebook environments:
https://zero-to-jupyterhub.readthedocs.io/en/latest/jupyterhub/customizing/user-environment.html