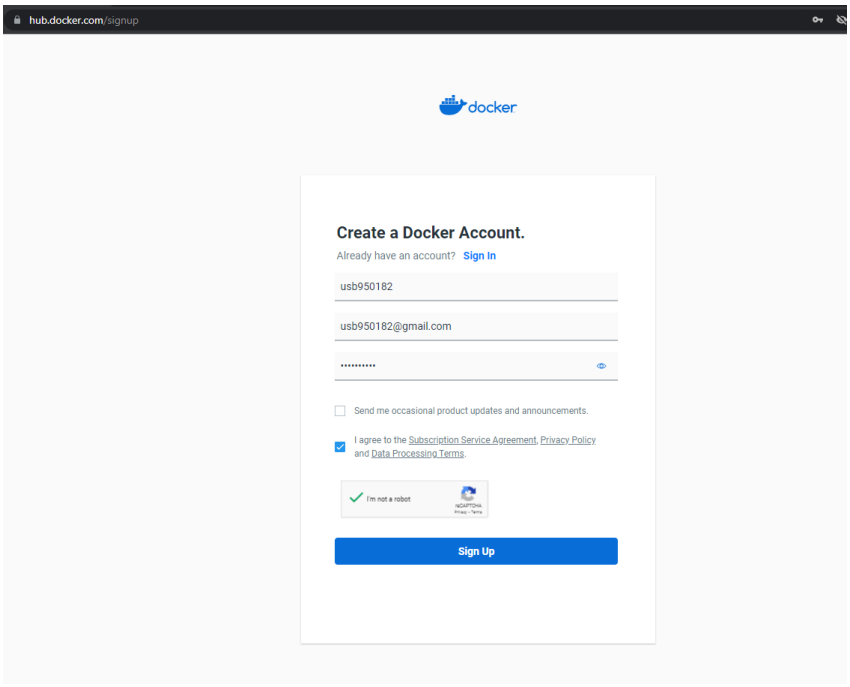
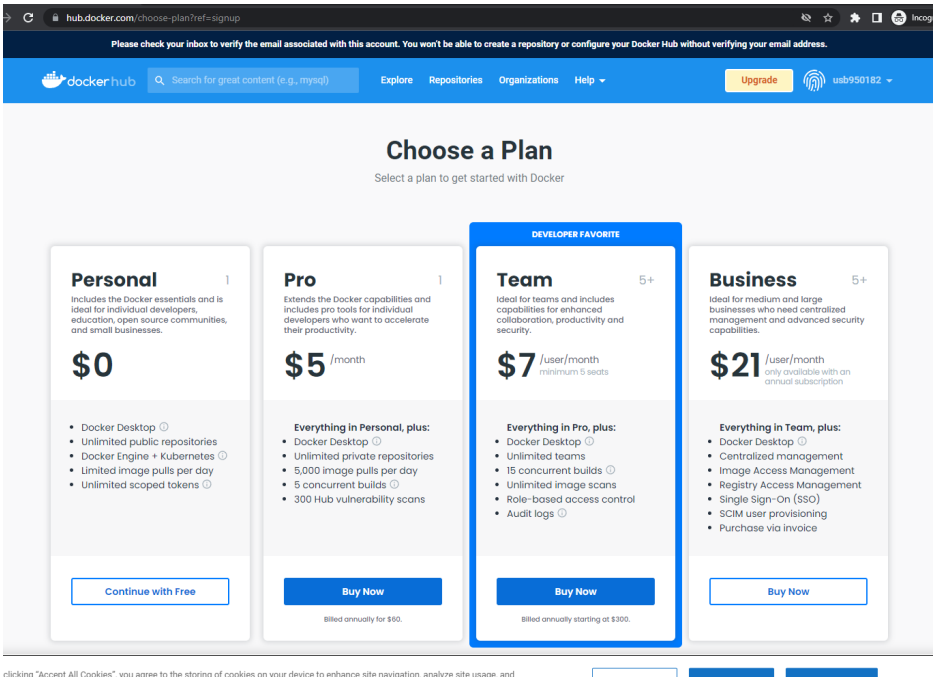
Setup/Installation Instructions

I. Dockerhub Setup

A. Signup for a docker account at: https://hub.docker.com/

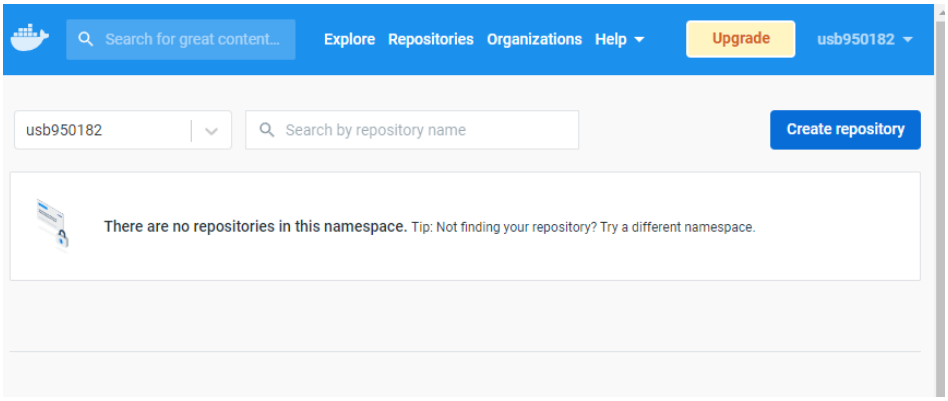


B. Continue with the free tier

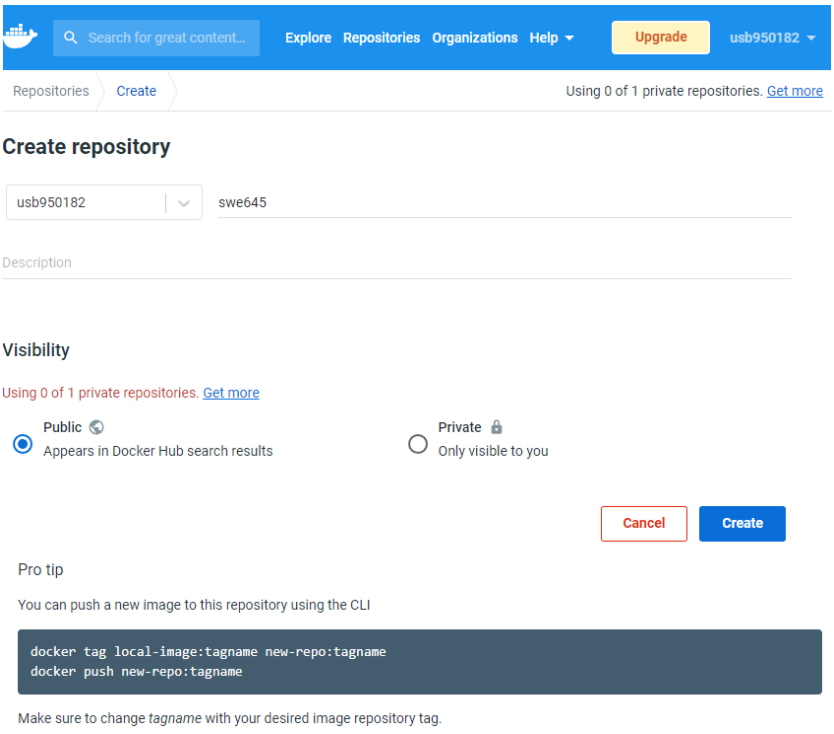


C. Verify your email address

D. Create a Dockerhub repository

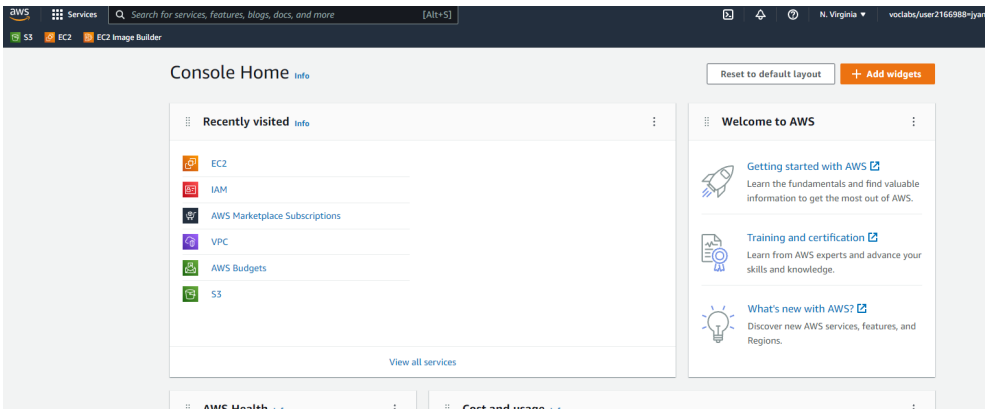


E. Enter a Repository name

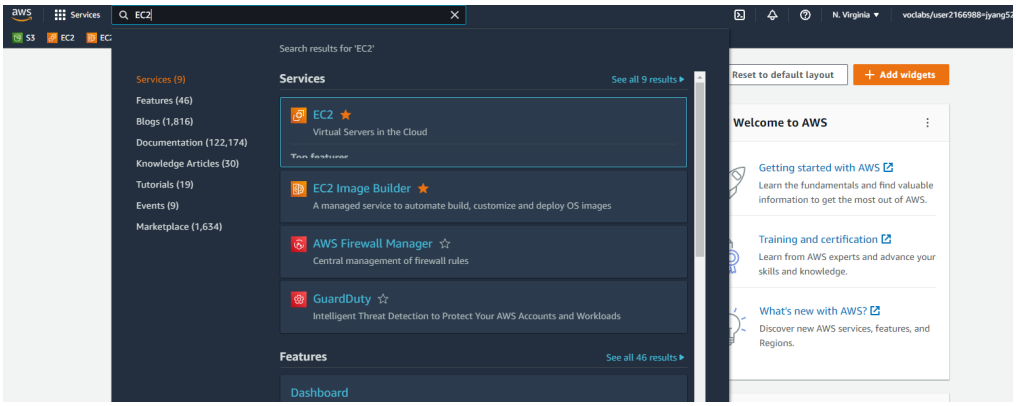


II. EC2 Instance Setup

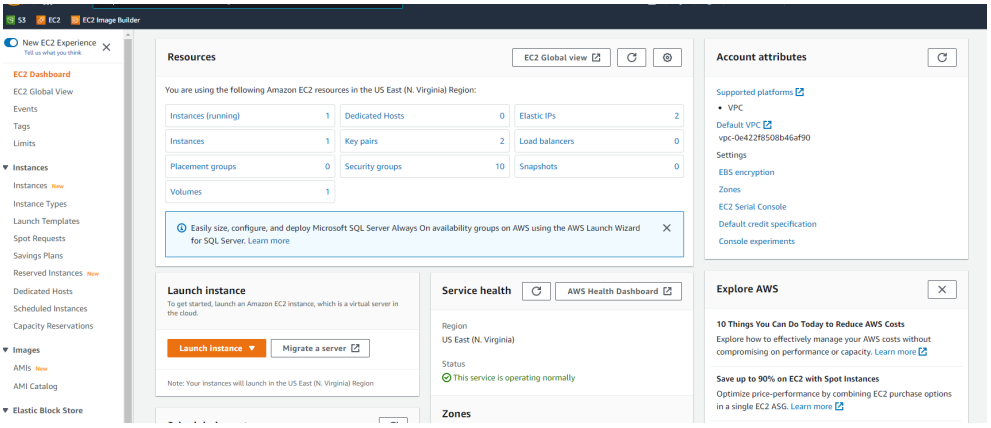
A. Log in to the AWS console



B. Search for and select “EC2” in the search bar

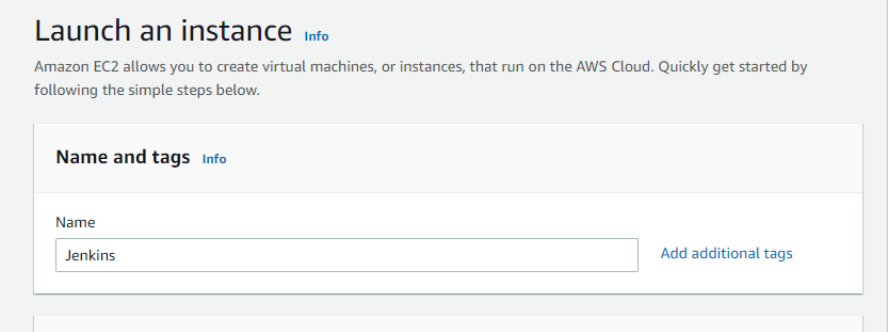


C. Select the Launch Instance button

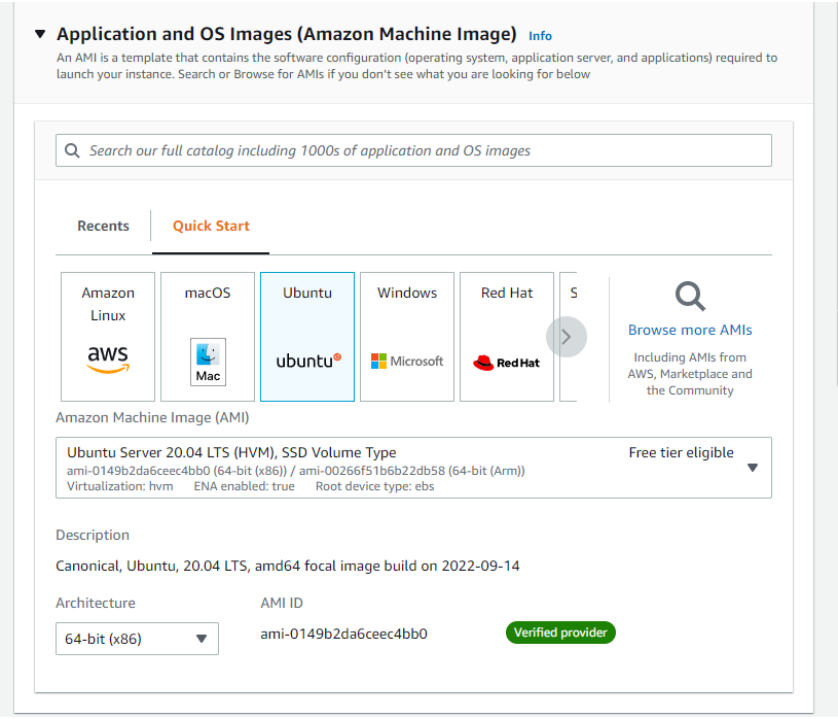


D. Create the Instance

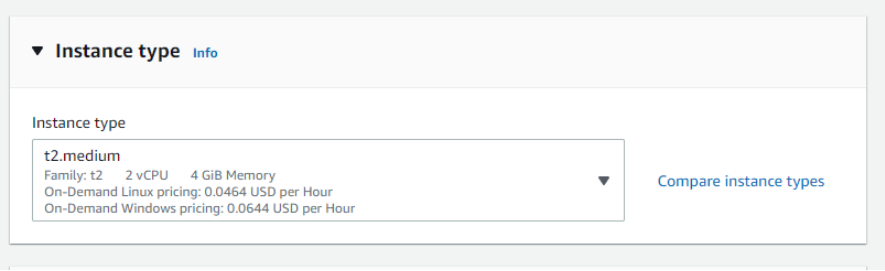
1. Enter a name



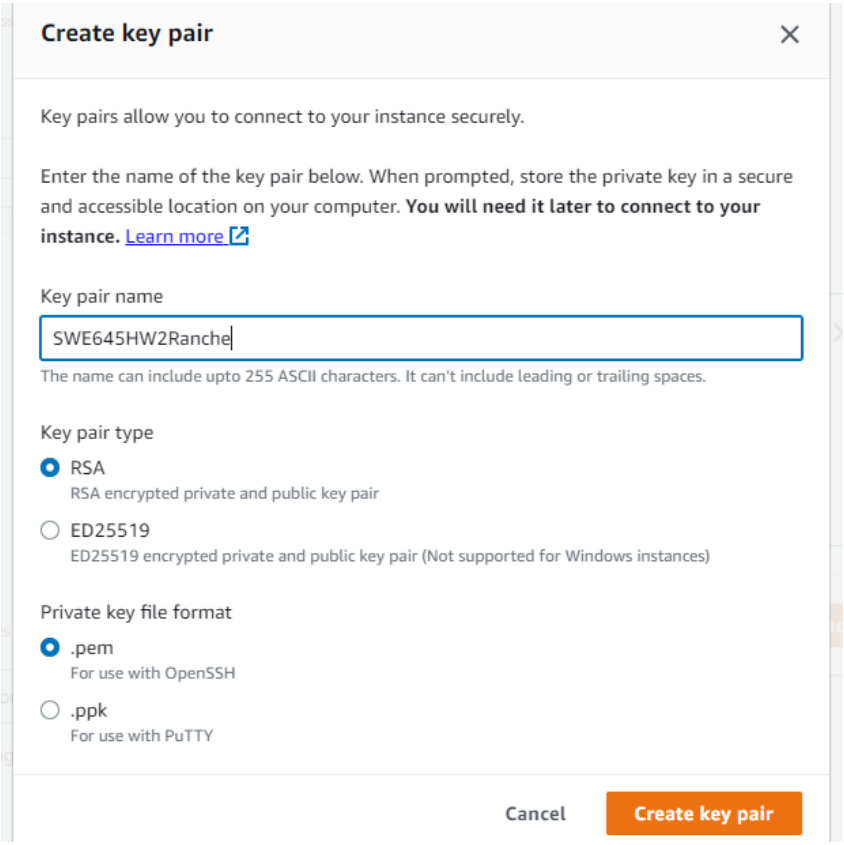
2. Select Ubuntu Server 20.04 LTS (HVM)



3. Select the “t2.medium” instance type

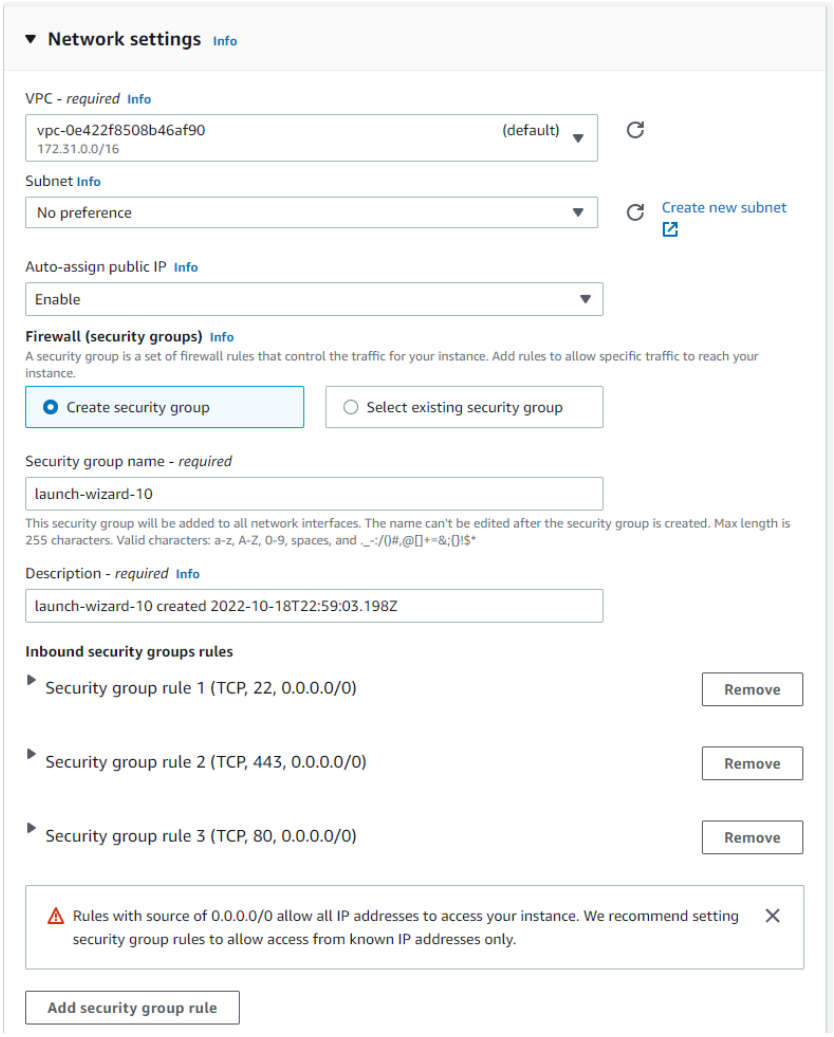


4. Create a Key pair (login). Use a key type of “RSA” and a private key file format of “.pem” Make sure to save it onto your local machine.



5. Under Network Settings, ensure that “ssh”, “HTTPS”, and “HTTP” are all checked, and the source type is from “Anywhere - 0.0.0.0/0”, with ports

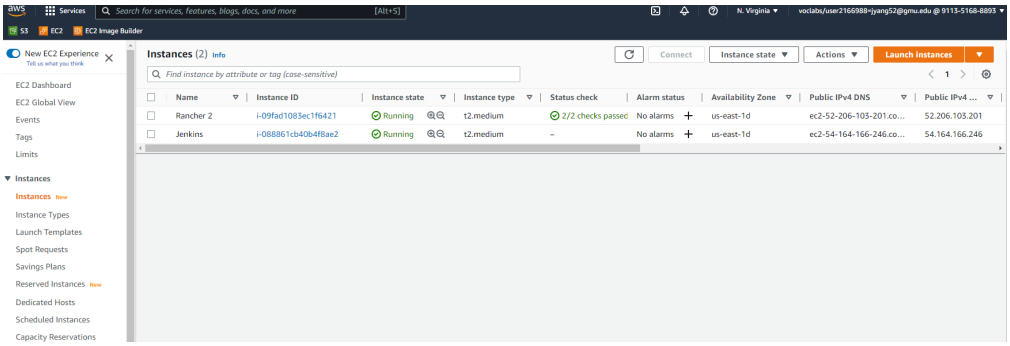
22, 443, and 80 exposed



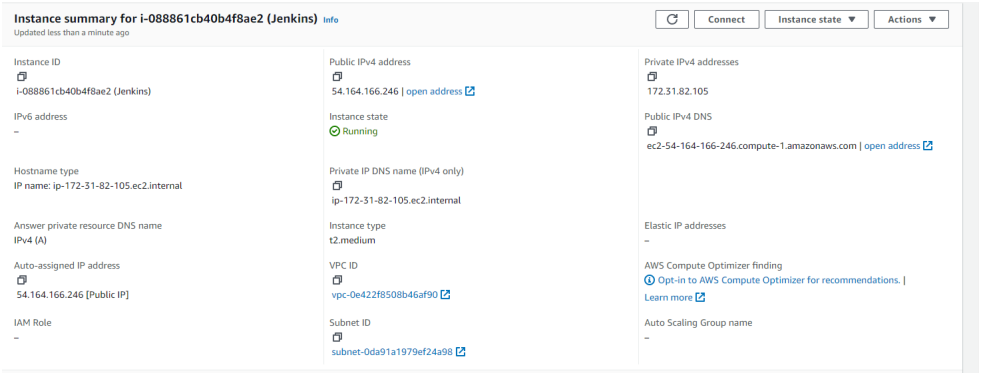
6. Launch your new instance

E. Click “View all Instances”

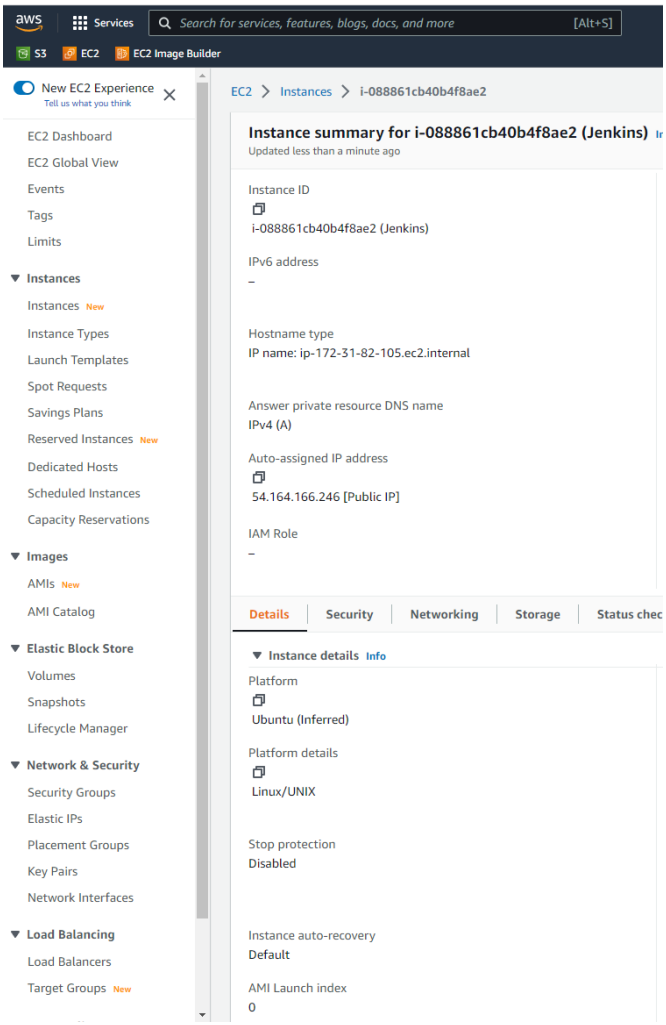
F. Select the blue link under “Instance ID” to make note of the Instance Info



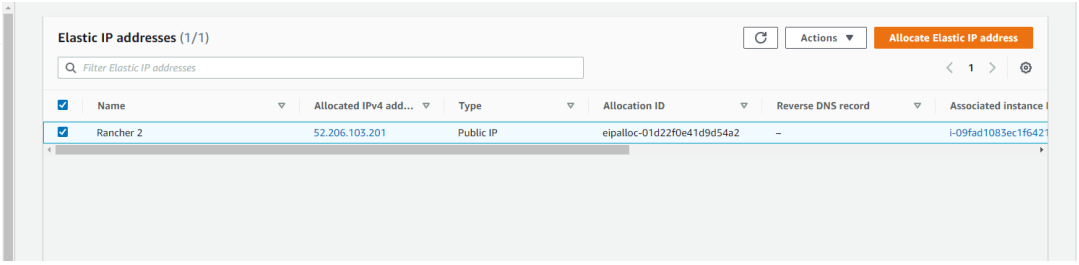
G. You'll see the instance summary for your instance, take note of the Public IPv4 DNS



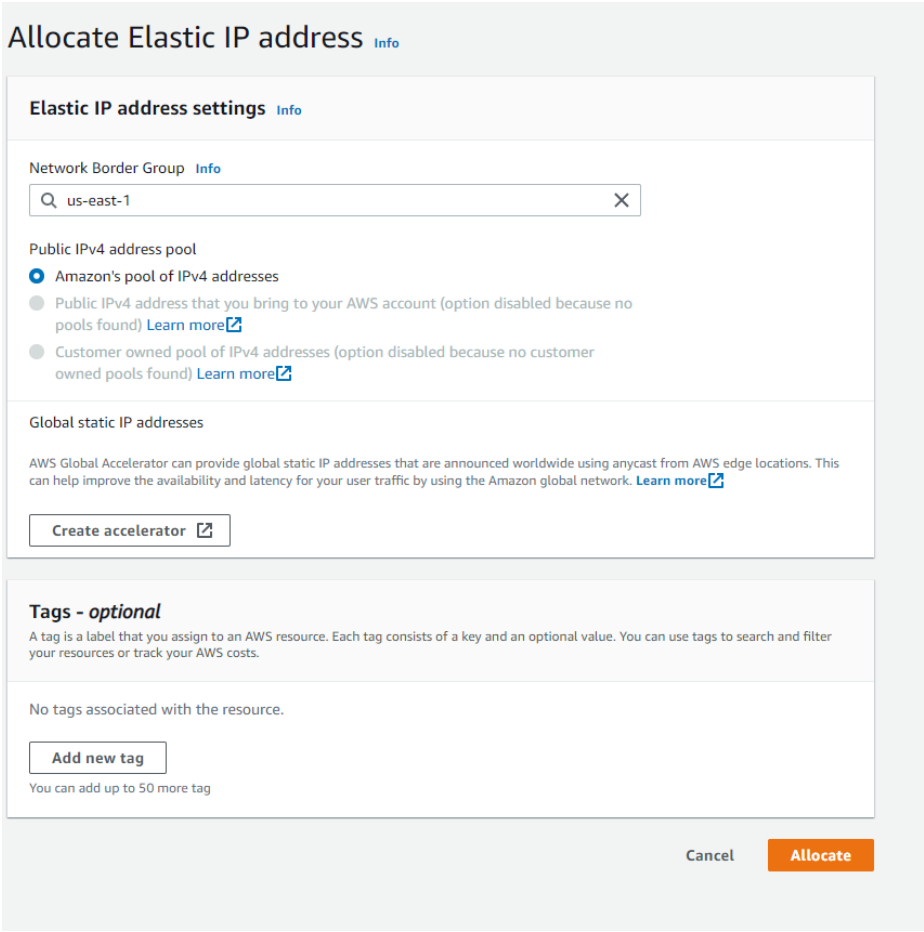
H. On the same page, in the left side bar menu, you'll see an entry called "Elastic IPs" under Network & Security. Select it.



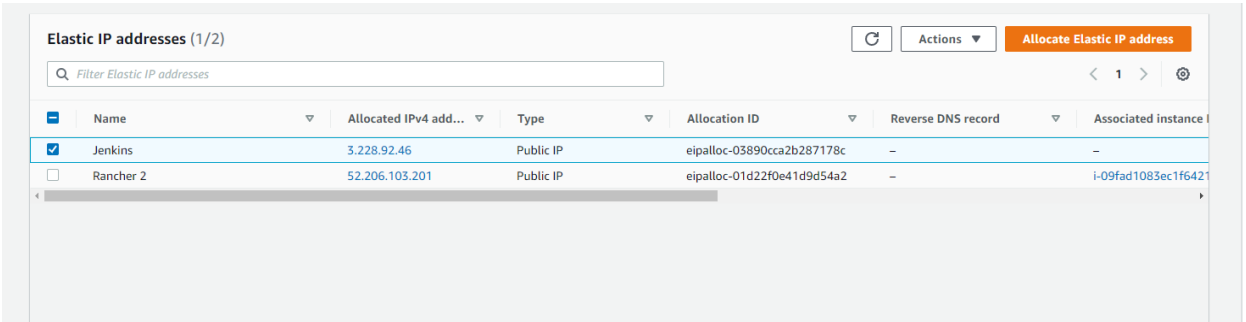
I. Select “Allocate Elastic IP Address”



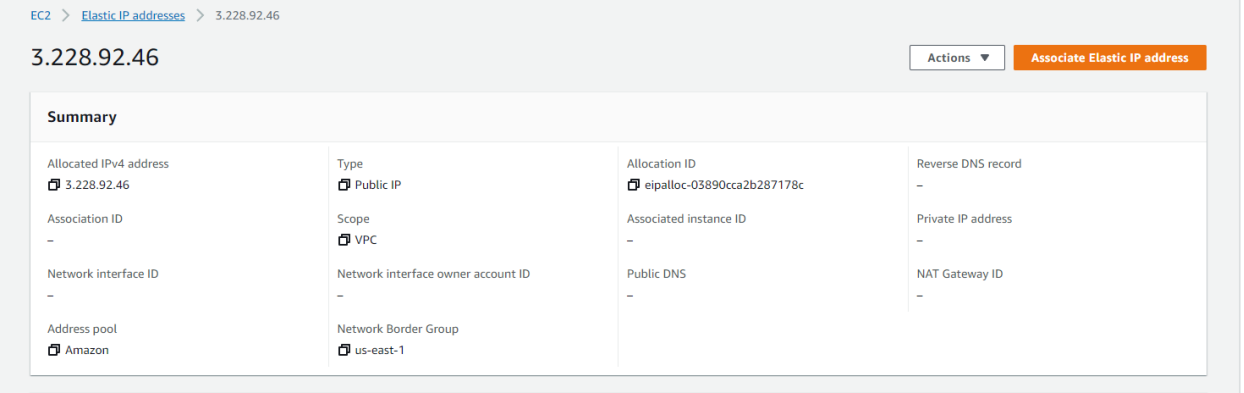
J. Ensure that the Network Border Group matches the instance’s, and click Allocate



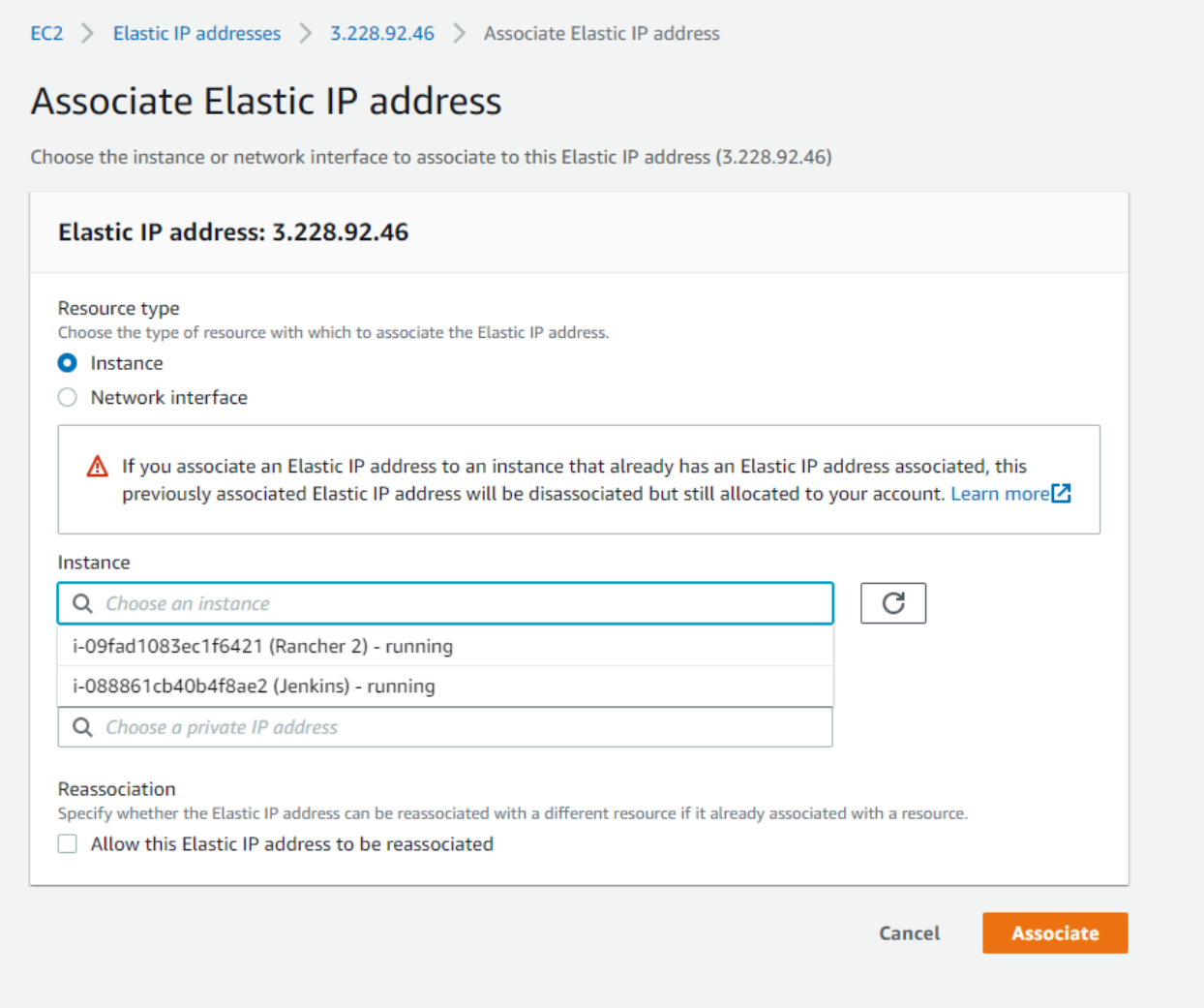
K. Edit the name of your new Elastic IP address and click the blue link under Allocated IPv4 Address



L. Click Associate Elastic IP Address



M. Select the new Instance name and click Associate



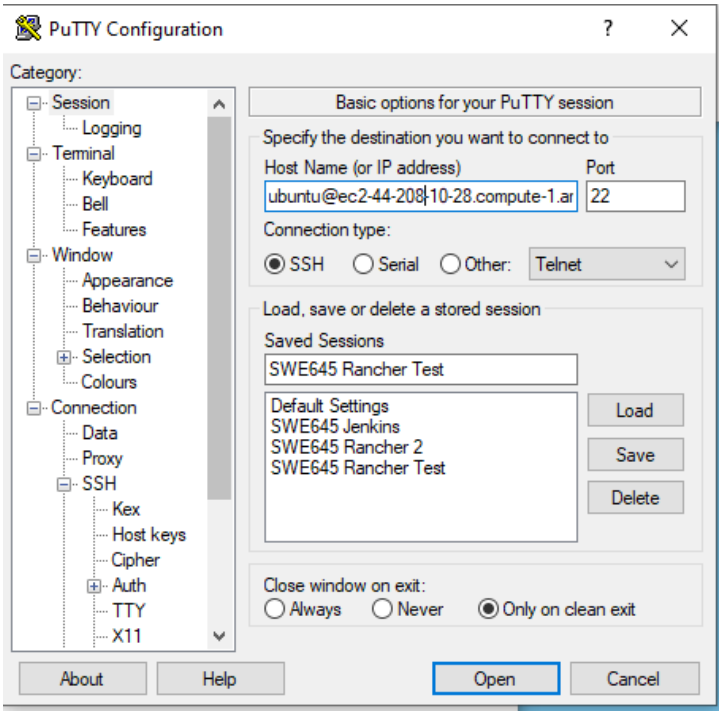
N. Enjoy the new EC2 instance! Associating an Elastic IP address hardened the public IPv4 DNS address by providing a permanent reference instead of creating a new one every time the instance stops and restarts

III. GKE Cluster Setup

A. Creating Kubernetes Cluster on GCP
B. Login to cloud.google.com using your credentials and then navigate to the Console
C. Create a project – In GCP, any work is done inside a project
D. Enable Kubernetes Engine API
E. Under the Navigation menu, select Kubernetes Engine and then select Project1, Click on Create to create a Kubernetes Cluster
F. Configure using the standard option and create the name of the cluster, zone(US-east-1)

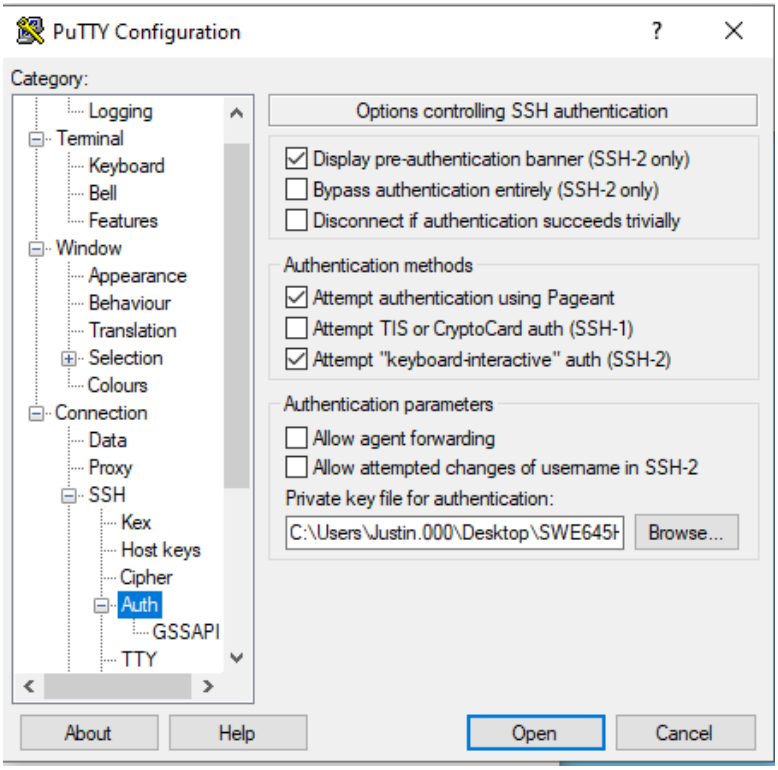
IV. Rancher Setup

A. Create and EC2 instance with the above steps
B. Connect to your new EC2 Instance
    1. If on Windows, download and install PuTTY.
        a) For the Host Name, enter 'ubuntu@<EC2 PUBLIC IPv4 DNS>'
        b) Use Port: 22
        c) Connection type: SSH
        d)



        e) On the left side menu, Click "Connection", then "SSH", then "Auth"

f) In the “Private key file for Authentication” field, Browse for the .pem file from the key login section of the EC2 instance setup

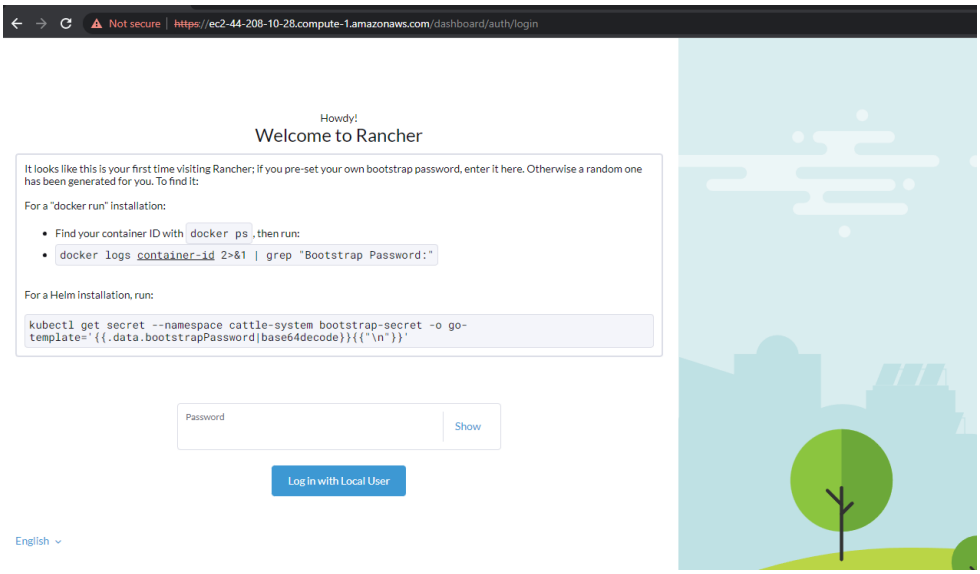


2. If on Mac or Linux, open a terminal window and enter the following command with the appropriate field:
    a) ssh -i <YOUR .PEM FILE LOCATION> ubuntu@<EC2 INSTANCE PUBLIC IPv4 DNS>

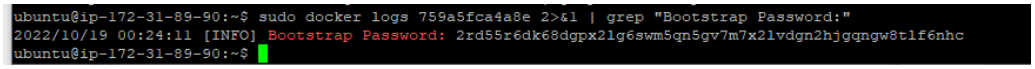
C. Once ssh’d into the instance, update it by running ‘sudo apt-get update’
D. Install docker with ‘sudo apt install docker.io’
E. Verify docker is installed by running ‘docker -v’
F. Install Rancher by running ‘sudo docker run --privileged=true -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher’
G. Enter the command ‘sudo docker ps’ and note the Container ID. This will be important later

```
ubuntu@ip-172-31-89-90:~$ sudo docker ps
CONTAINER ID   IMAGE             COMMAND            CR
759a5fca4a8e   rancher/rancher   "entrypoint.sh"    6
ubuntu@ip-172-31-89-90:~$
```

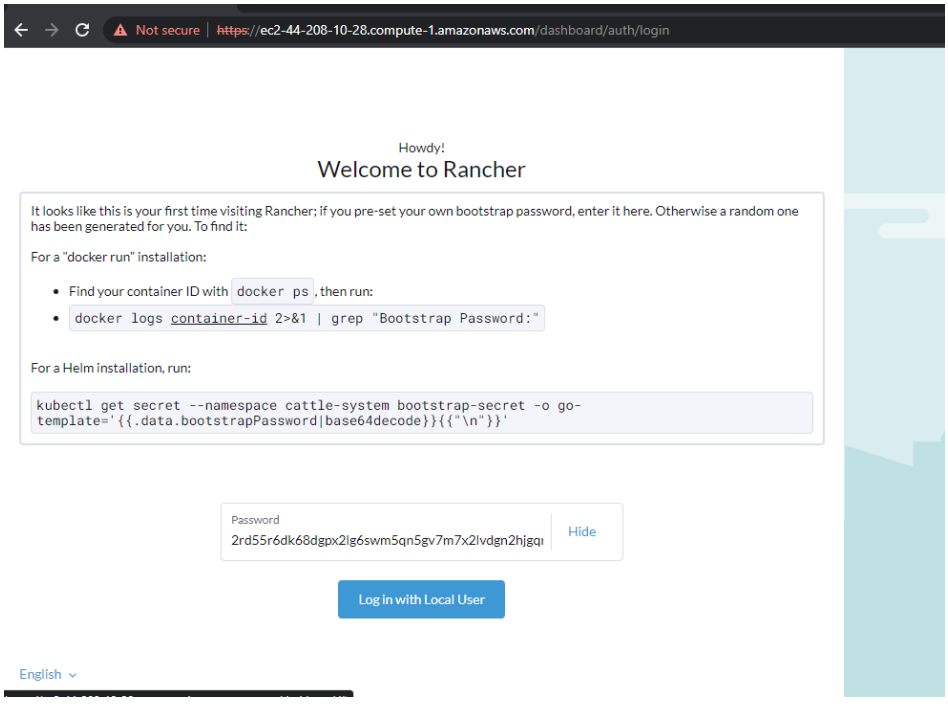
H. Open your browser and copy/paste your EC2 public IPv4 address. Accept the browser privacy warning and you should land on this page



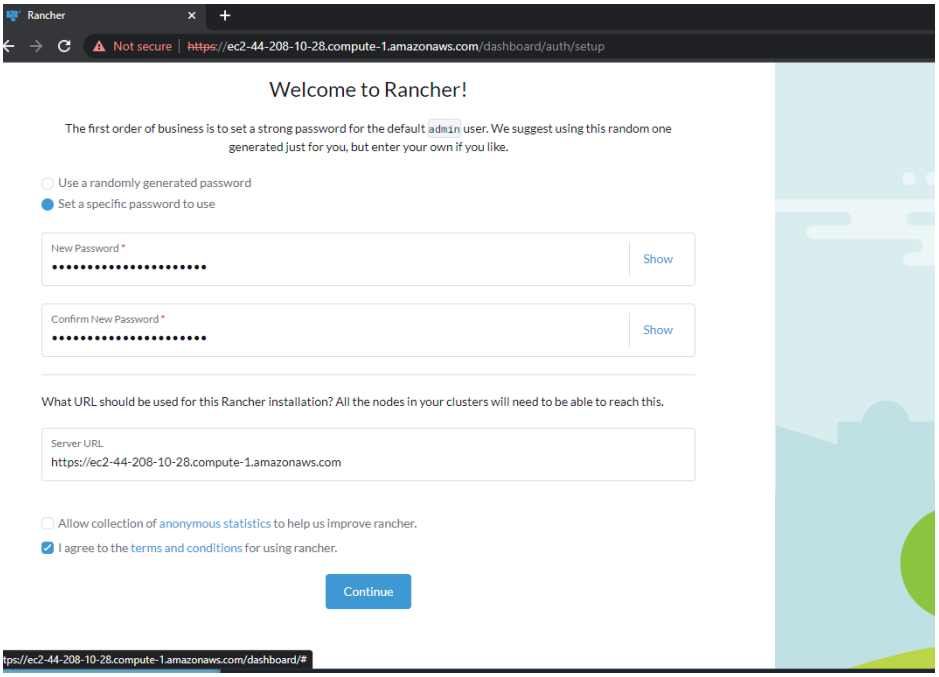
I. Enter 'sudo docker logs <CONTAINER ID> 2>&1 | grep "Bootstrap Password:"', where <CONTAINER ID> is the value from step 7. You'll see the Bootstrap Password.



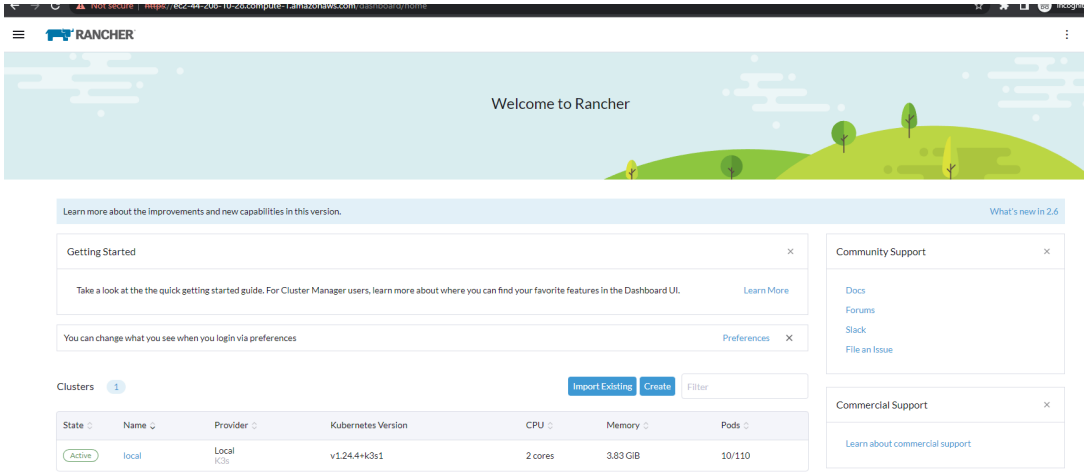
J. Copy and paste the Bootstrap Password into the Rancher UI Password field and login



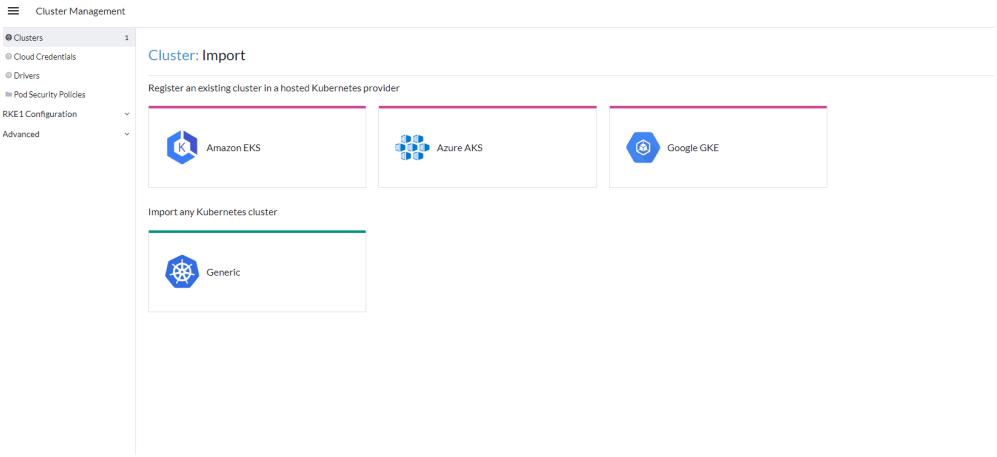
K. Create a new password and agree to the terms, then hit ‘Continue’



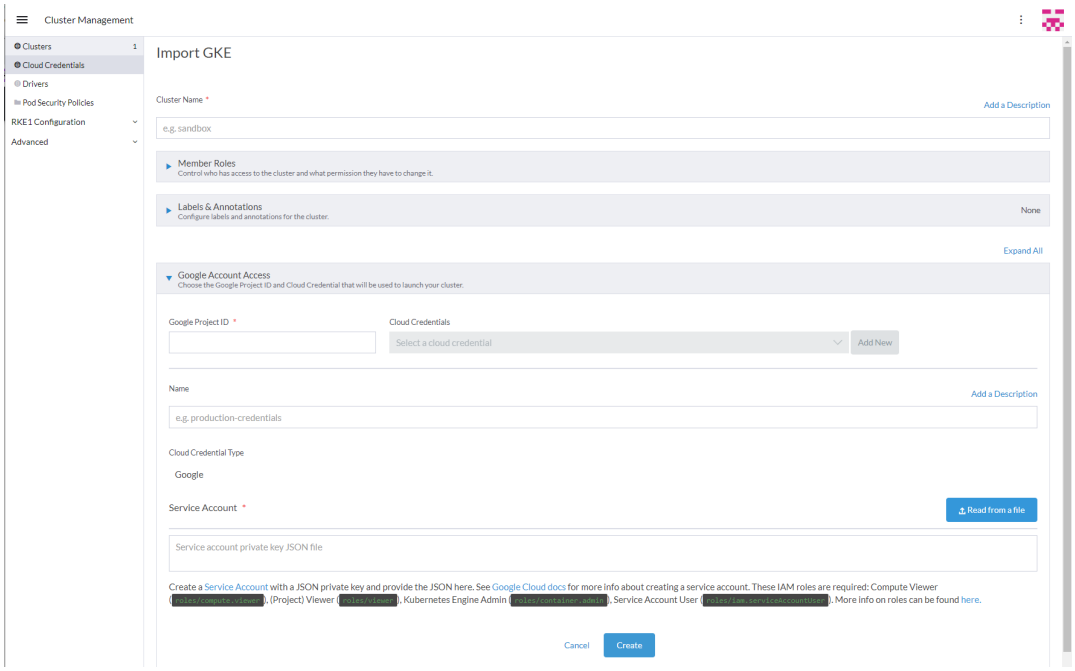
L. You’ll be taken to the home screen!



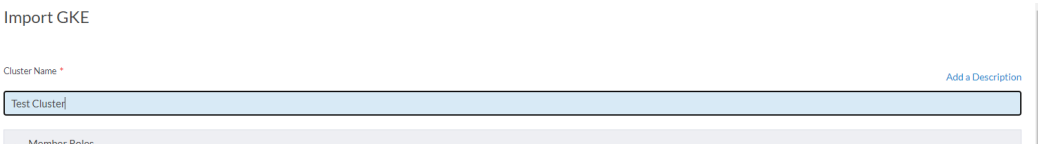
M. Next to the Clusters section of the Home Page, click ‘Import Existing’



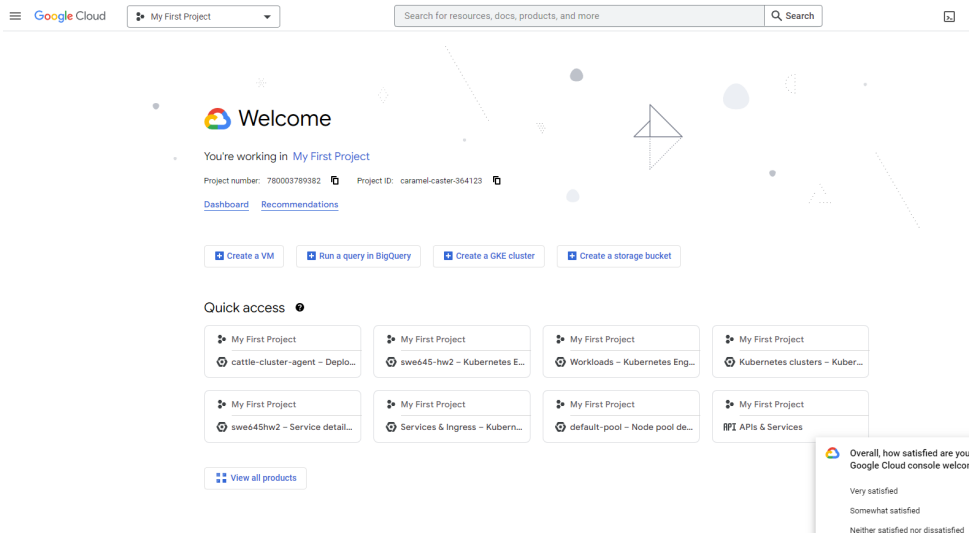
N. Select your Cluster to import. I’ll be continuing with GKE



O. Insert a name for your cluster



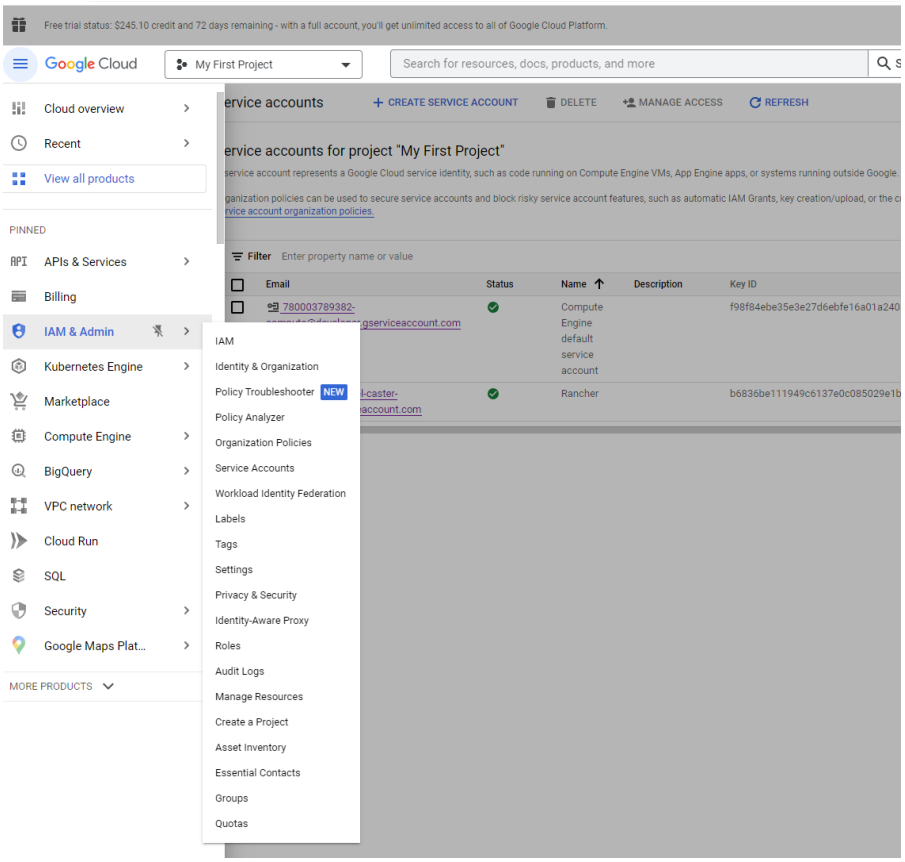
P. From the Google Cloud Project Page, find the Project ID and copy it



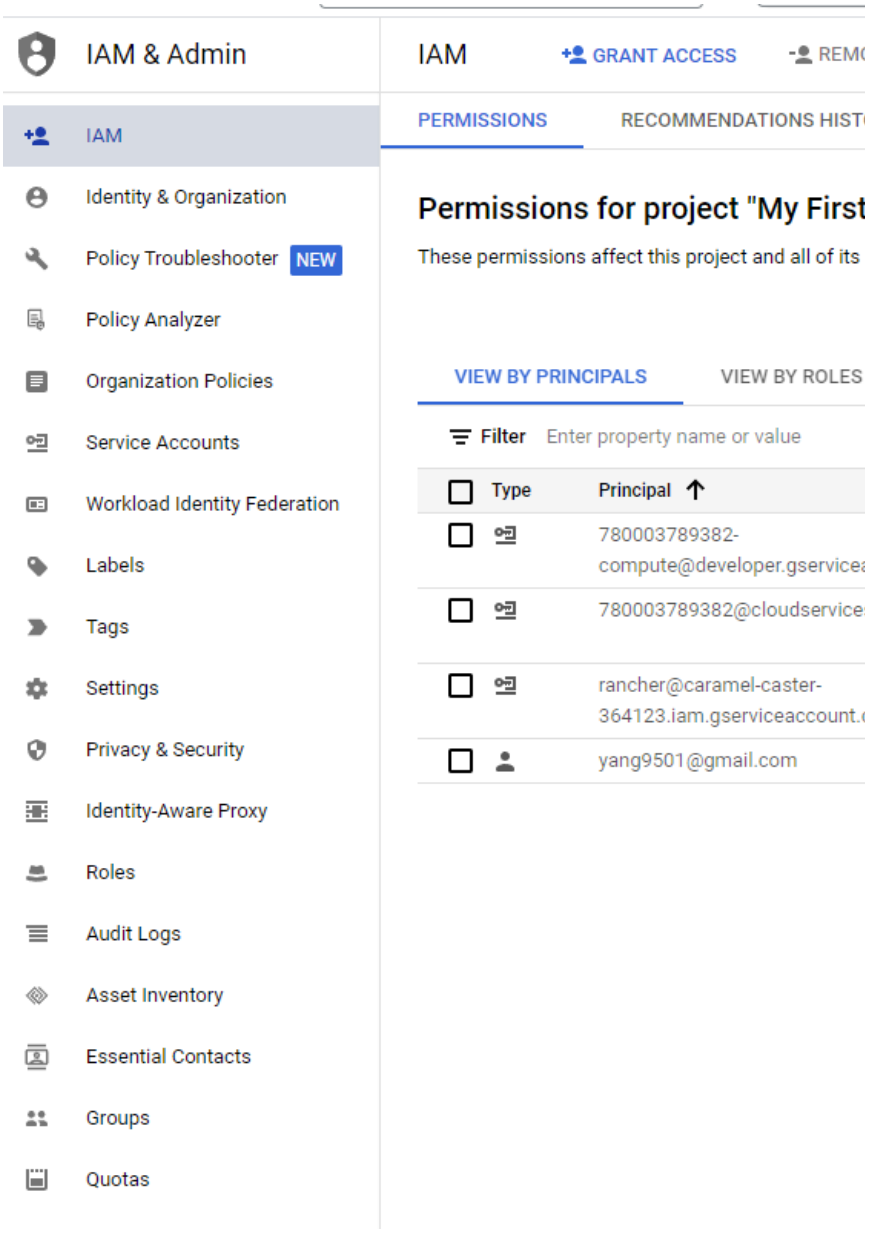
Q. Within the Google Account Access section, copy that Project ID into the Google Project ID field

R. Create a Service Account within GKE

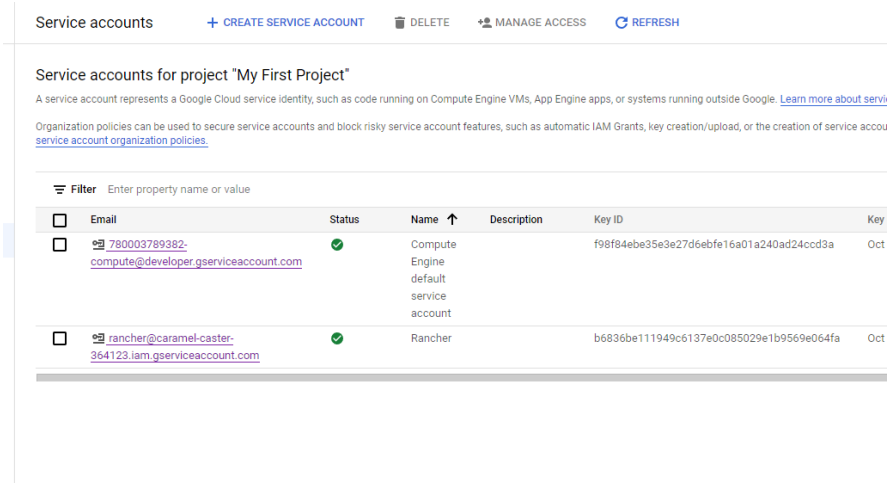
1. Within the Google Cloud Console, open the left sidebar. Select the IAM & Admin option



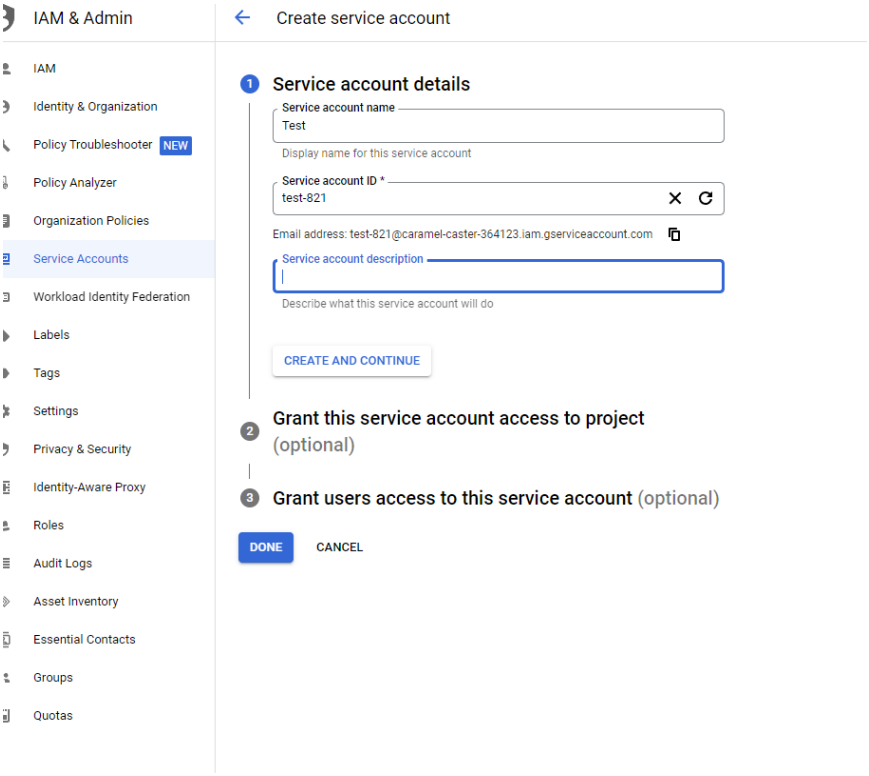
2. Within the left sidebar, select Service Accounts



3. Select ‘Create Service Account’ near the top of the window

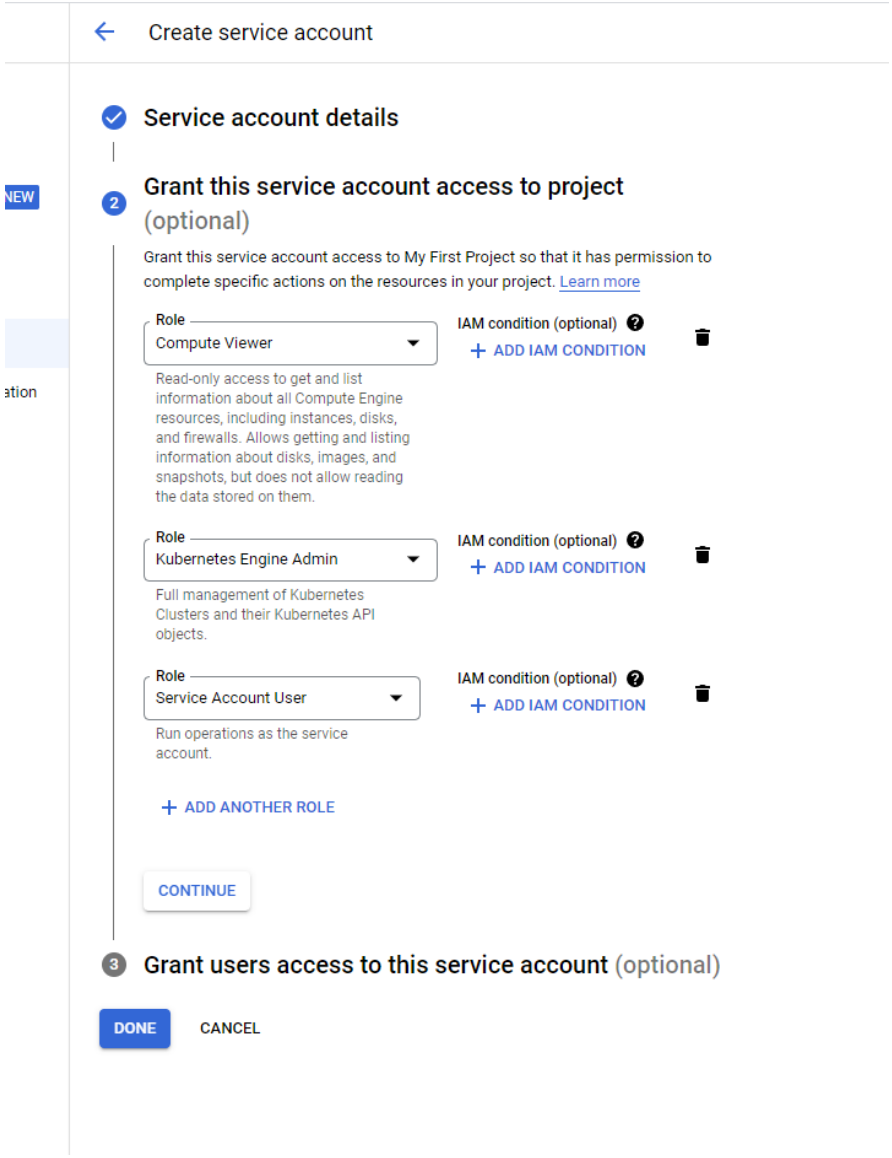


4. Enter the Service Account Details



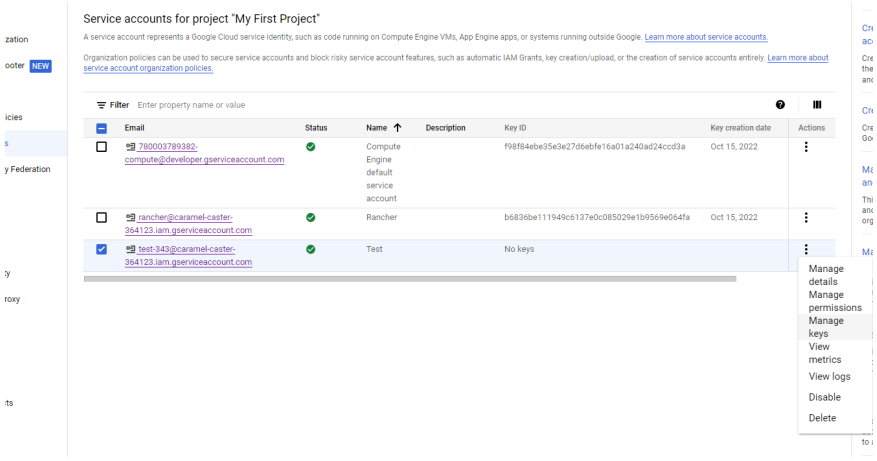
5. Add the Compute Viewer, (Project) Viewer, Kubernetes Engine Admin , and Service Account User roles to the Service Account. Or alternatively, if you can’t find all the necessary roles, just make the Service Account an

Owner.

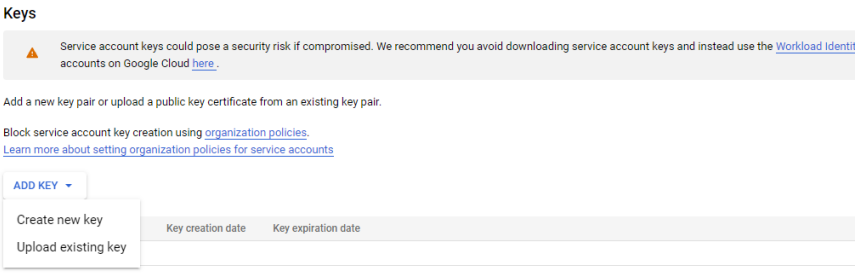


6. Within the Service Account management page, click the Actions button(the three vertical dots) associated with your new Service Account

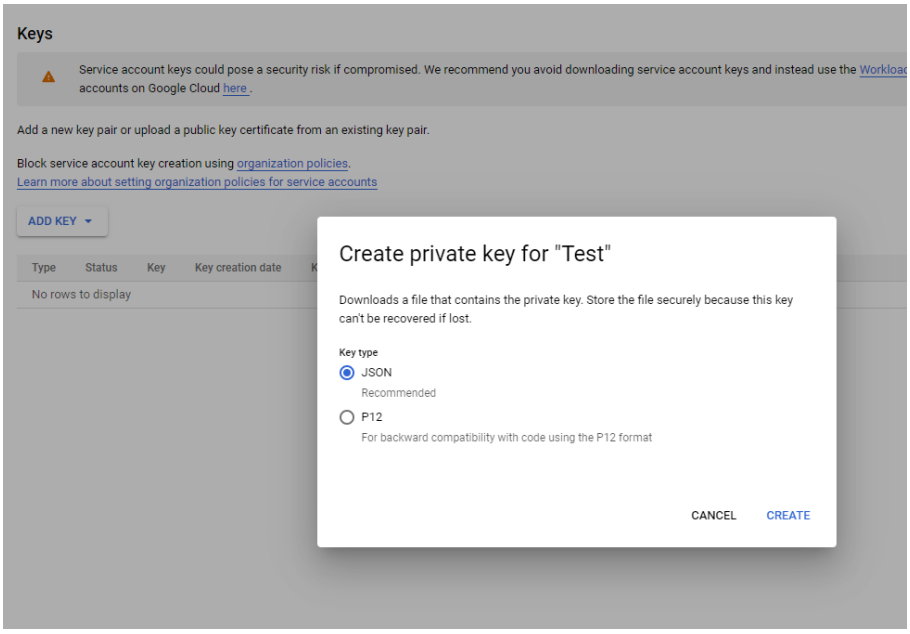
and select “Manage Keys”



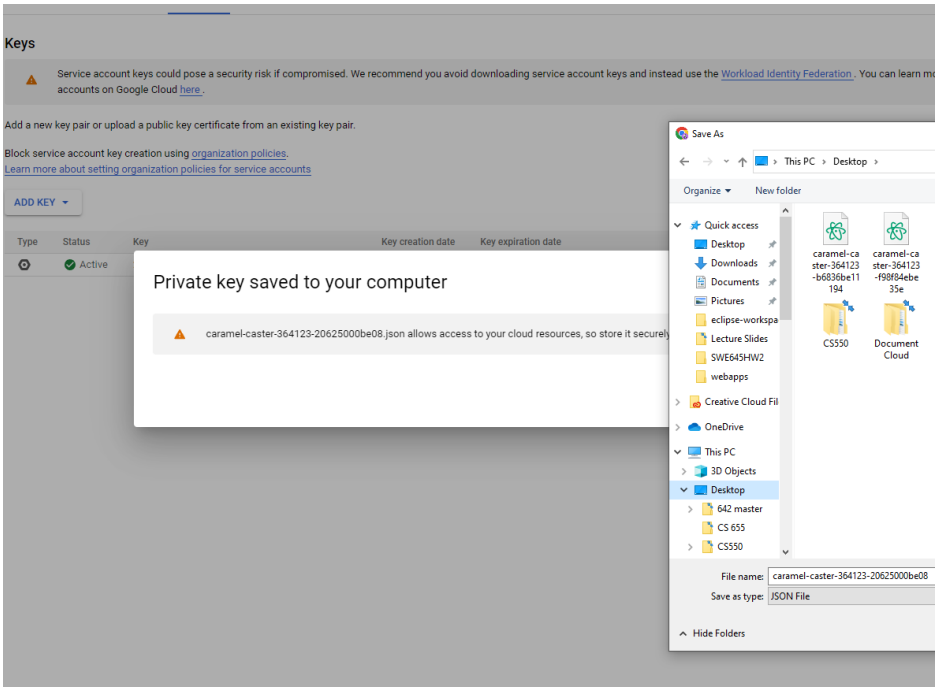
7. Add a new key by selecting the Create New Key option from the Add Key dropdown menu



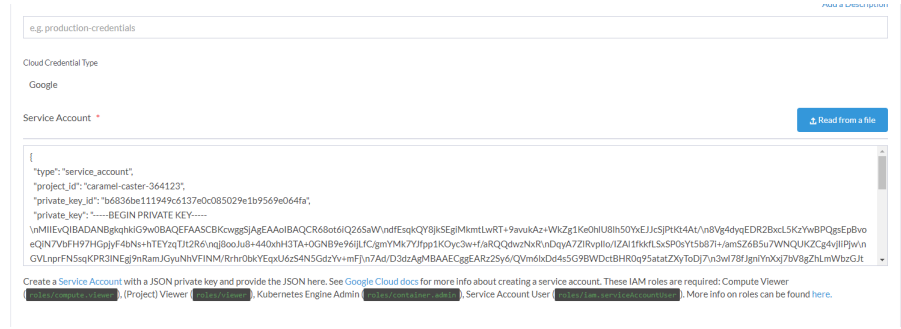
8. Select JSON and Create



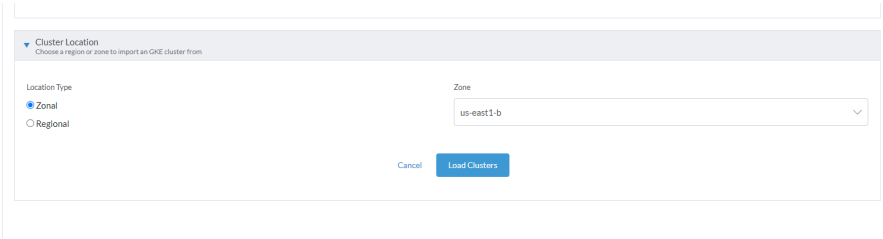
9. Save the key file on your local machine



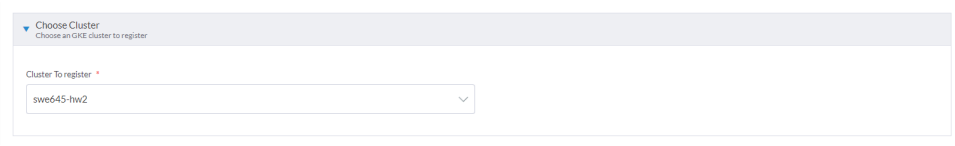
10. Back within Rancher, upload the key file into the Service account field using the “Read from a file” button



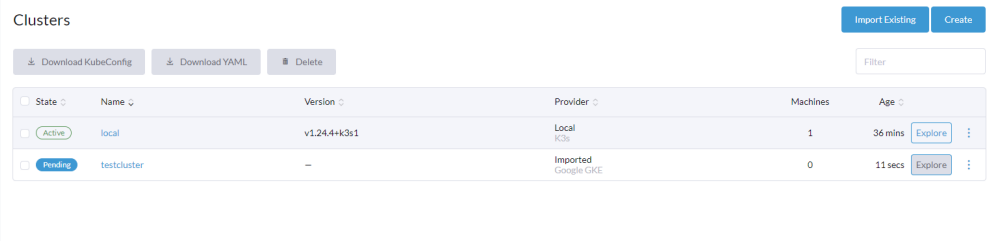
11. Select the appropriate Zone for your Cluster Location



12. Click Load Clusters
13. Choose the appropriate cluster to load from your cloud provider in the Choose Cluster Section



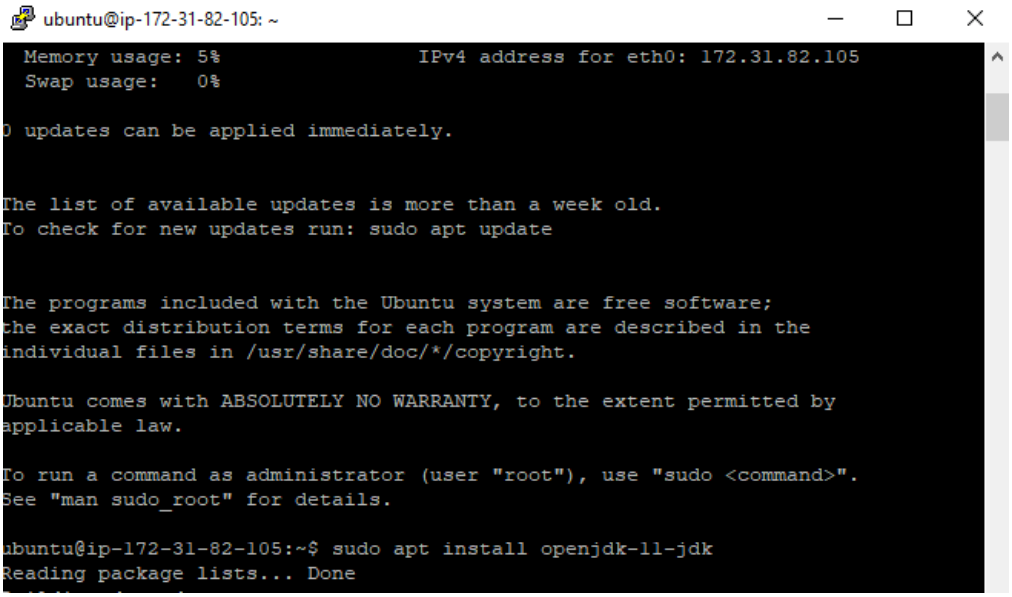
14. Register the cluster!
15. It will take a while to get ready



V. Jenkins Setup

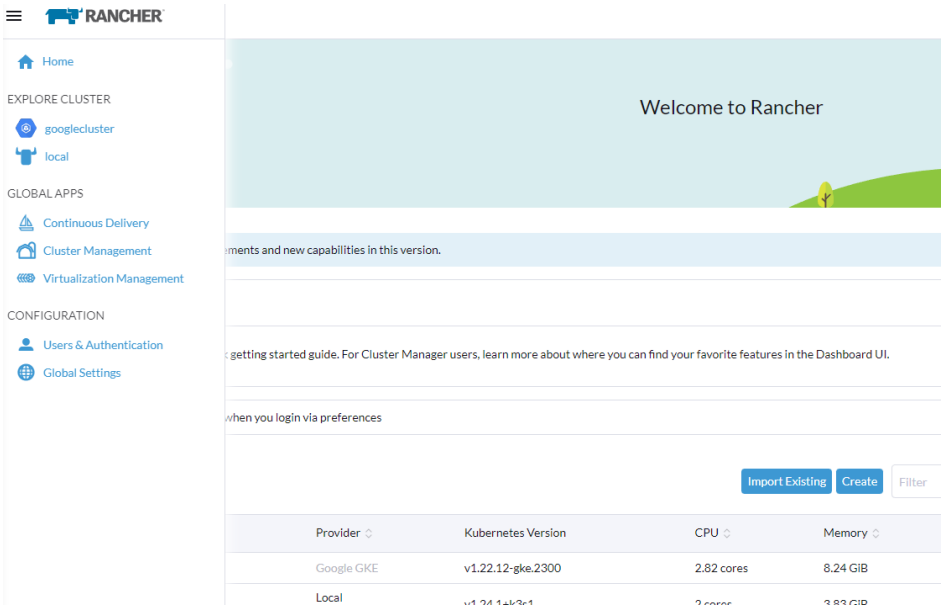
A. Create another EC2 instance using the steps above
   1. Make sure to add an additional Inbound Rule under the Security Groups section with Custom TCP type, Port Range of 8080, and Source of Anywhere IPv4.

B. Connect to the EC2 instance using the terminal on Mac or Linux, or Putty on Windows
C. Install JDK 11: ‘sudo apt install openjdk-11-jdk’

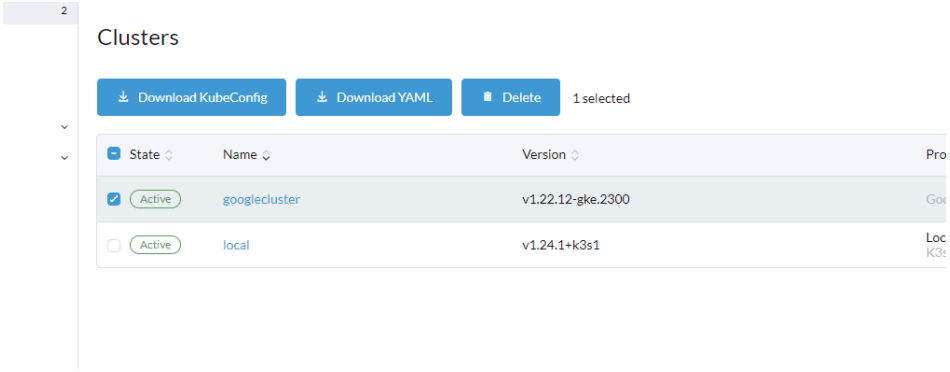


D. Import the GPG keys of the Jenkins Repository
    1. wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -
E. Add the Jenkins repository to the system
    1. sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
F. Once ssh’d into the instance, update it by running ‘sudo apt-get update’
G. Install docker with ‘sudo apt install docker.io’
H. Verify docker is installed by running ‘docker -v’
I. Install Jenkins with the following commands:
    1. sudo apt update
    2. sudo apt install jenkins
J. The Jenkins service will automatically start after installation. Check status with
    1. Systemctl status jenkins
K. Install Kubectl
    1. sudo apt install snapd
    2. sudo snap install kubectl --classic
L. Add the jenkins user to the docker user group
    1. sudo usermod -a -G docker jenkins
M. Reboot the Jenkins Host in the EC2 console for the user group changes to take effect
N. We also need to add the kubeconfig file into the Jenkins host for reference when using kubectl commands in the Pipeline job
    1. Visit your Rancher dashboard

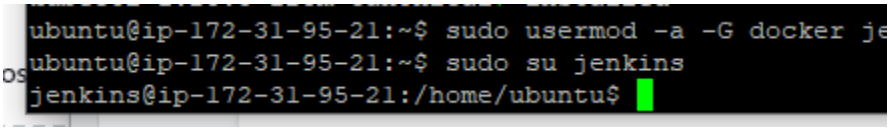
2. On the expandable menu on the left side, click Cluster Management



3. Check the cluster corresponding to your deployment and click the Download KubeConfig button



4. You'll download a yaml file onto your local machine. Make note of the directory location
5. Enter the following command to switch to the jenkins user in the command line of the Jenkins host
    a) sudo su jenkins



6. Navigate to the working directory of the Jenkins user, in this case /var/lib/jenkins. If you're not sure which directory it is, skip this step and perform it after Step 19. The Home directory is found in the Jenkins UI

under Dashboard -> Manage Jenkins -> Configure System.

```
jenkins@ip-172-31-95-21:/home/ubuntu$ cd ..
jenkins@ip-172-31-95-21:/home$ cd ..
jenkins@ip-172-31-95-21:/$ cd var/lib/jenkins/
jenkins@ip-172-31-95-21:~$
```

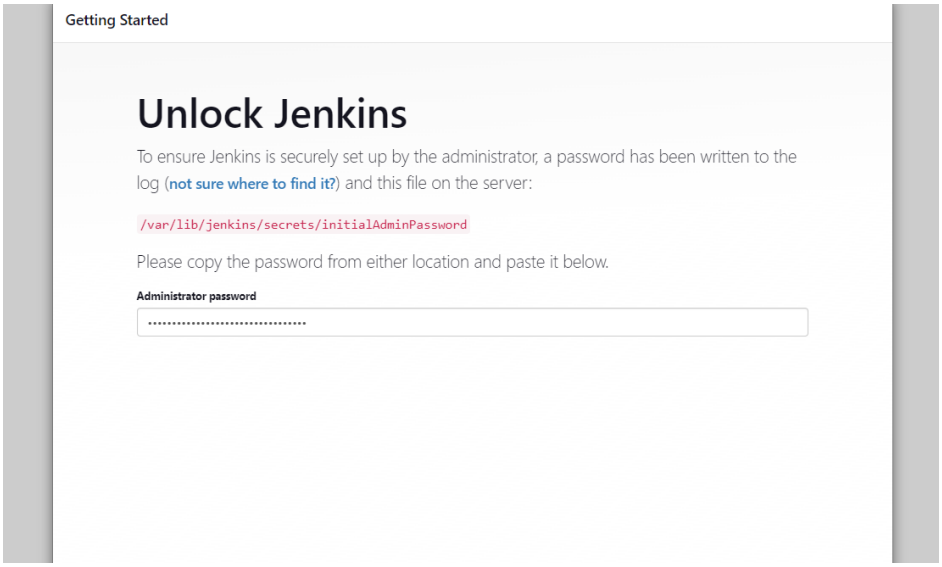
7. Create a new directory called .kube
    a) mkdir .kube
8. Enter the new .kube directory and create a file called ‘config’. Copy paste the contents of the downloaded yaml file into this new config file.

```
jenkins@ip-172-31-95-21:~/.kube$ touch config
jenkins@ip-172-31-95-21:~/.kube$ vi config
jenkins@ip-172-31-95-21:~/.kube$
```

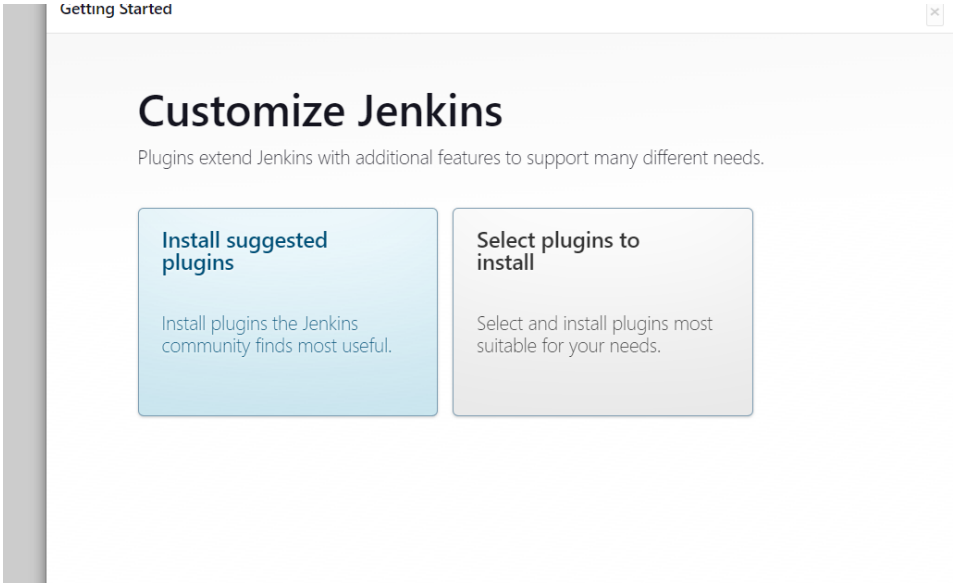
O. Get the password on the terminal with this command
1. sudo cat /var/lib/jenkins/secrets/initialAdminPassword

```
ubuntu@ip-172-31-95-21:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
1b41a7254afc4f92b136ca2da469ff8d
ubuntu@ip-172-31-95-21:~$
```

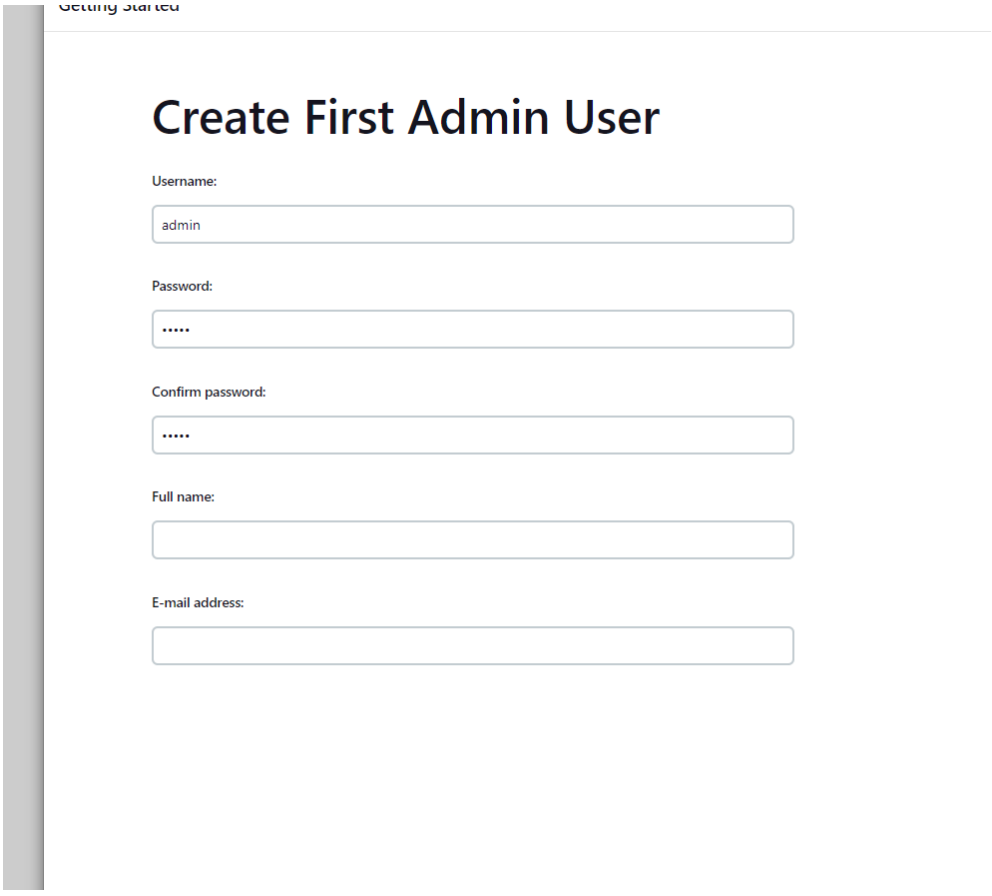
P. In your browser, navigate to http://your_ip_or_domain:8080 and past the password into the appropriate field
1. Make sure to use the http prefix, NOT https



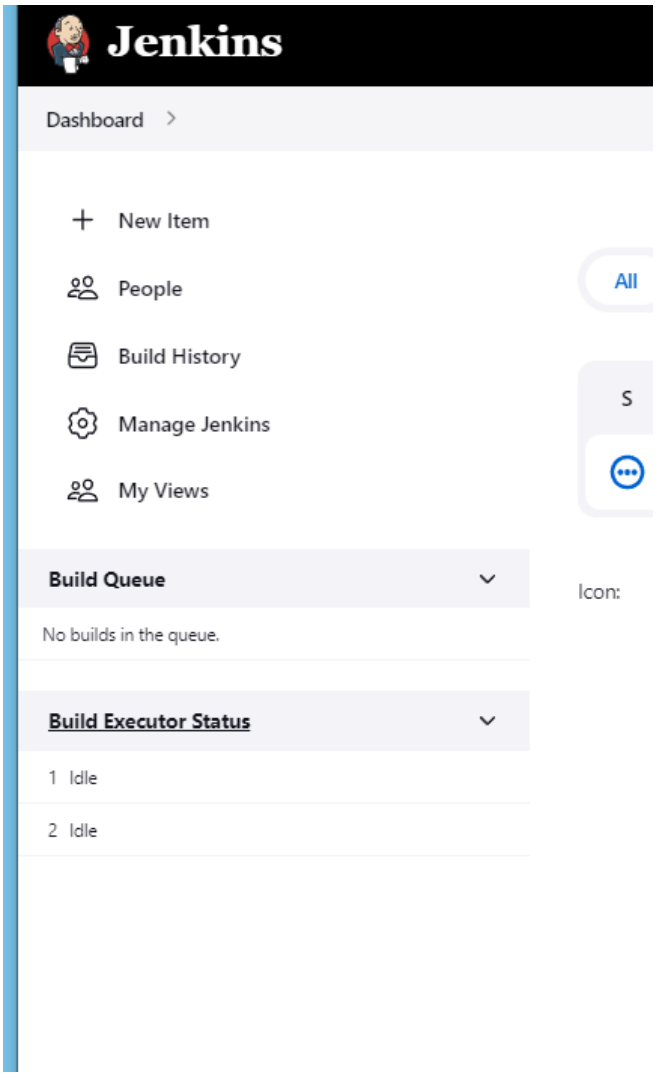
Q. Install the suggested plugins



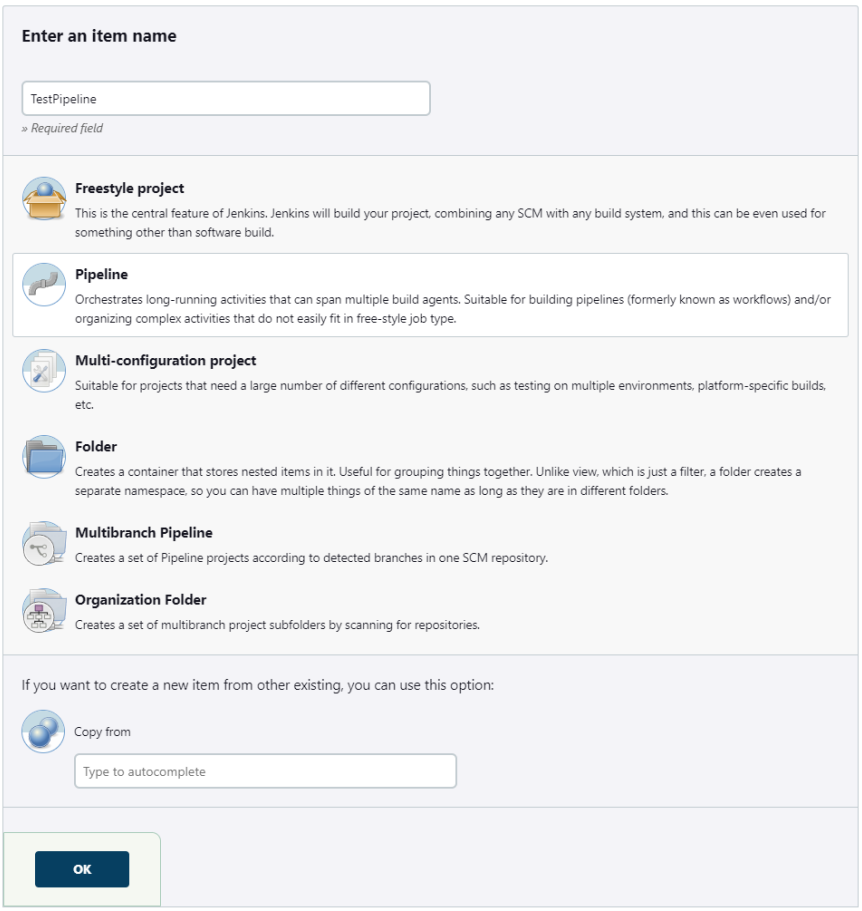
R. Create an admin user



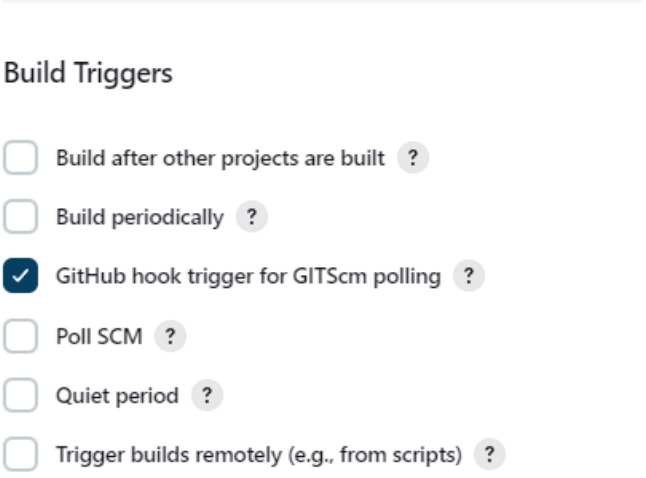
S. You should see a dashboard at some point once you've completed initial setup. Click the "New Item" option in the left sidebar to create your first pipeline



T. Enter a name and select the Pipeline option



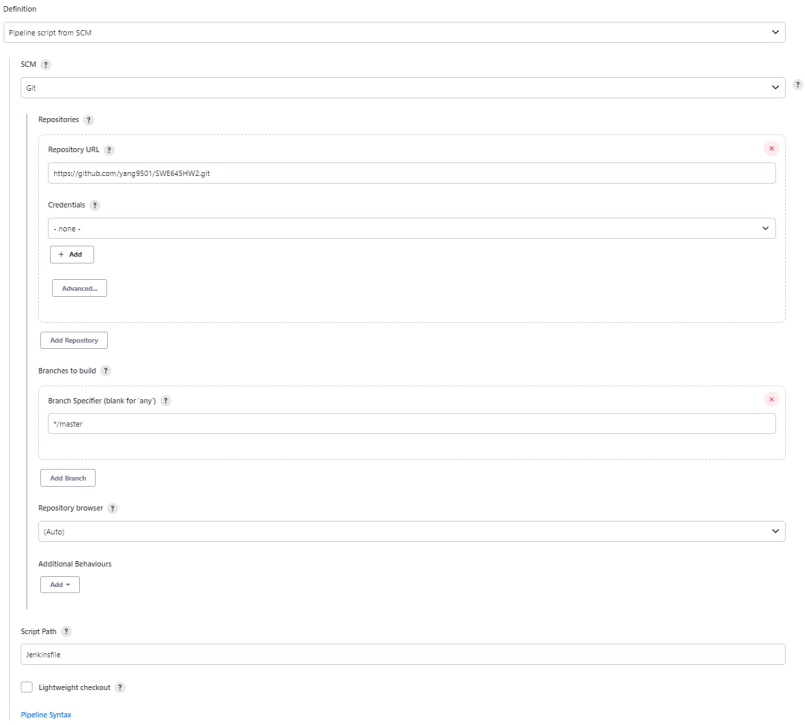
U. Select the Github hook trigger for GITScm polling option in the Build Triggers



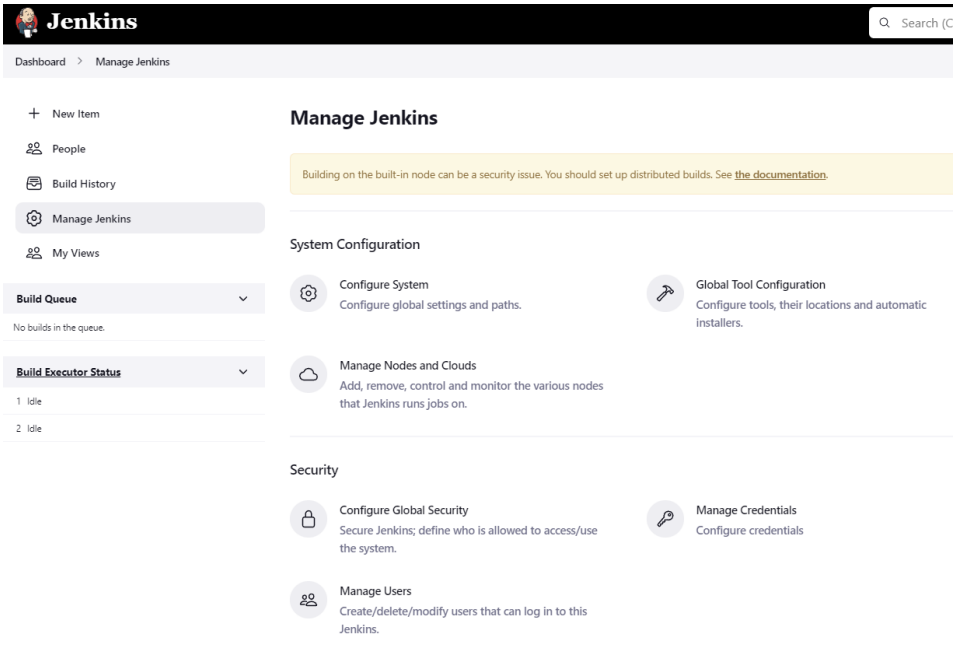
section

V. Within the Pipeline section, choose the following
    1. Definition: Pipeline script from SCM
    2. SCM: Git
    3. RepositoryURL: Your git repository URL

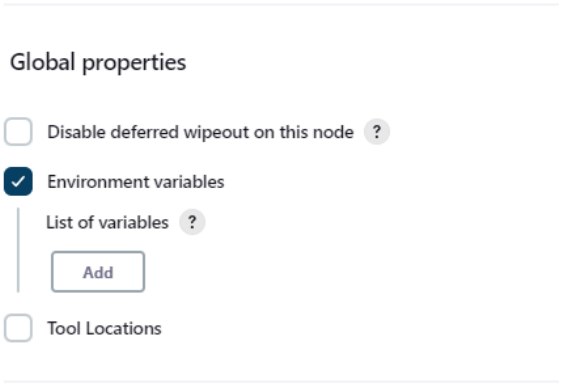
4. Branch Specifier: Your git branch you want to build
5. Script Path: Jenkinsfile
6. Lightweight checkout: UNCHECKED



W. Exit back to the Dashboard. Select Manage Jenkins from the left sidebar, then Configure System.



X. Under Global Properties, check Environment Variables box and click Add



Y. Add the following environment variable for use in the Jenkinsfile
   1. Name: DOCKERHUB_PASS
   2. Value: swe645hw2docker

Z. Open the project repository in Github
   1. Enter the settings tab for that repository
   2. Click the “Webhooks” option in the left sidebar menu
   3. Add a webhook with the Payload URL pointed towards: <AWS URL>:8080/github_webhook/ and leave the other settings as default.

AA. We’re ready to test the pipeline. Committing and pushing a trivial change in the codebase with git will trigger the webhook to be sent to the Jenkins /github_webhook/ API, which will trigger our new pipeline to run.

Pipeline SWE645HW2

Stage View

	Declarative: Checkout SCM	Building the Student Survey Image	Pushing Image to Dockerhub	Deploying on cluster through Rancher
Average stage times: (Average full run time: ~25s)	337ms	6s	948ms	1s
#3 Oct 19 03:56 No Changes	366ms	14s	2s	2s

VI. Project Configuration Setup(Dockerfile/Jenkinsfile)

A. Dockerfile

Dockerfile

```
FROM tomcat:9.0-jdk11
COPY SWE645HW2.war /usr/local/tomcat/webapps/
```

B. Jenkinsfile

Jenkinsfile

```
pipeline {
  agent any
  stages {
    stage("Building the Student Survey Image"){
      steps{
        script {
          git 'https://github.com/yang9501/SWE645HW2.git'
          sh 'pwd'
          sh 'ls'
          sh 'rm -rf *.war'
          //The .war file creation process breaks if there are any foreign files in the folder structure.  Have to isolate the filepath w
          sh 'jar -cvf SWE645HW2.war -C src/main/webapp/ .'
          sh 'ls'
          sh 'echo ${BUILD_TIMESTAMP}'
          sh 'docker login -u yang9501 -p ${DOCKERHUB_PASS}'
          sh 'docker build -t yang9501/swe645hw2:latest .'
        }
      }
    }
    stage("Pushing Image to Dockerhub"){
      steps{
        script {
          sh 'docker push yang9501/swe645hw2:latest'
        }
      }
    }
    stage("Deploying on cluster through Rancher"){
      steps{
        script {
          //Jenkins uses the 'jenkins' user on the host to perform actions on the host.  .kube/config is a folder that grants access to p
          sh 'kubectl --kubeconfig /var/lib/jenkins/.kube/config version'
          sh 'kubectl --kubeconfig /var/lib/jenkins/.kube/config set image deployment/swe645hw2 container-0=yang9501/swe645hw2:latest'
          //After the cluster receives the image, you have to redeploy the pods in order for the new image to take effect
          sh 'kubectl --kubeconfig /var/lib/jenkins/.kube/config rollout restart deployment/swe645hw2'
        }
      }
    }
  }
}
```