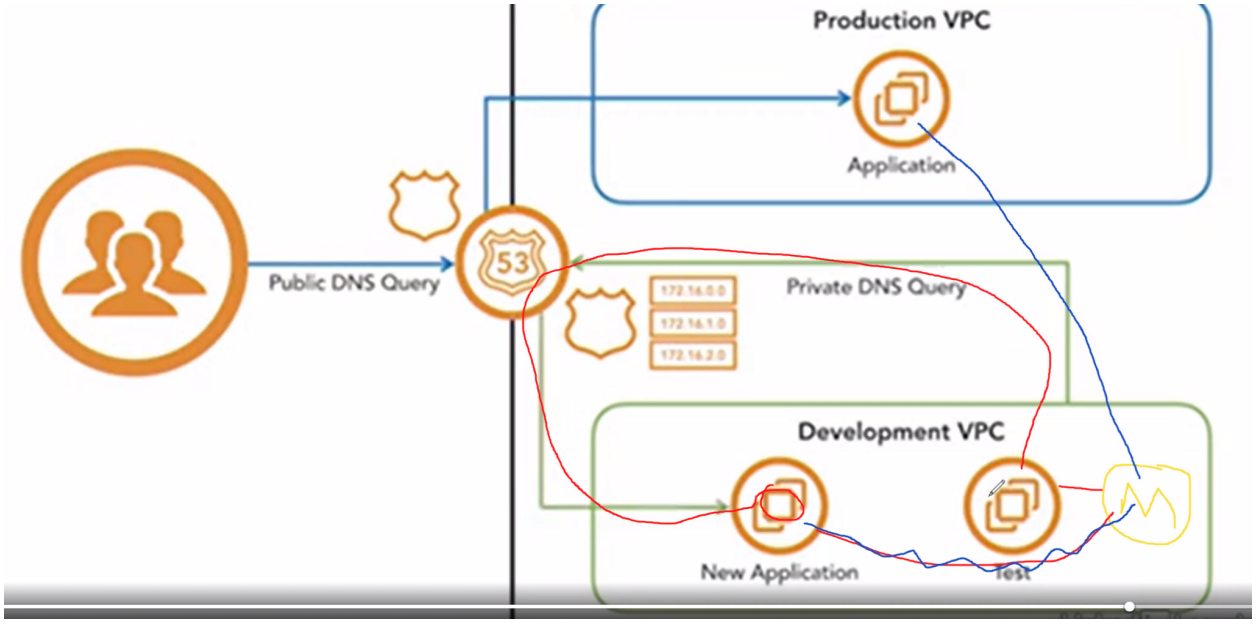
Deployment 6

What are we doing?

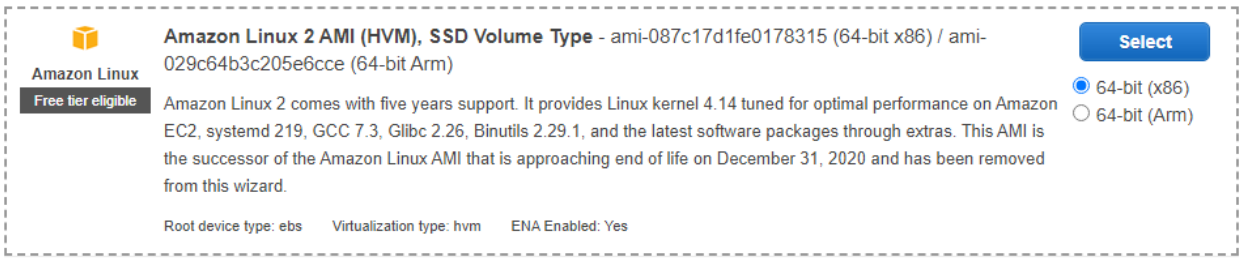
For this deployment, we are creating a development environment which has a jenkins master, test and an application. The jenkins master gives us instructions to both EC2s which is the test ec2 and application ec2. The cypress is the test and the application gets built and deployed.

The test will only happen when you give instructions to cypress. Once the test runs and is successful, you will have the files move to the master jenkins. The master jenkins will then push it up to the production environment.



Setting up the first EC2s

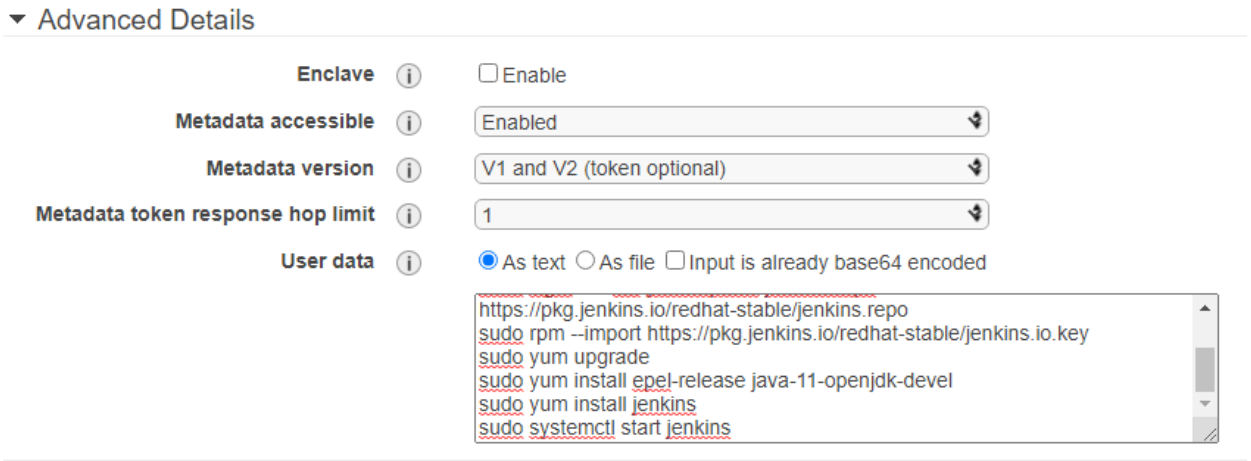
First EC2 will have the Jenkins application installed on it.
This instance will also use the Amazon Linux 2, AMI.



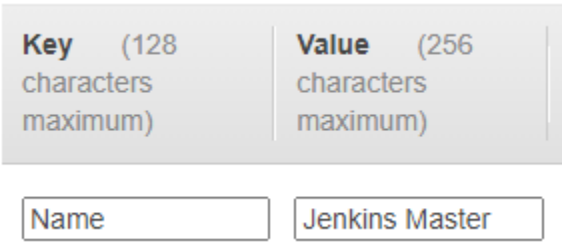
T2.micro for instance type

User data script

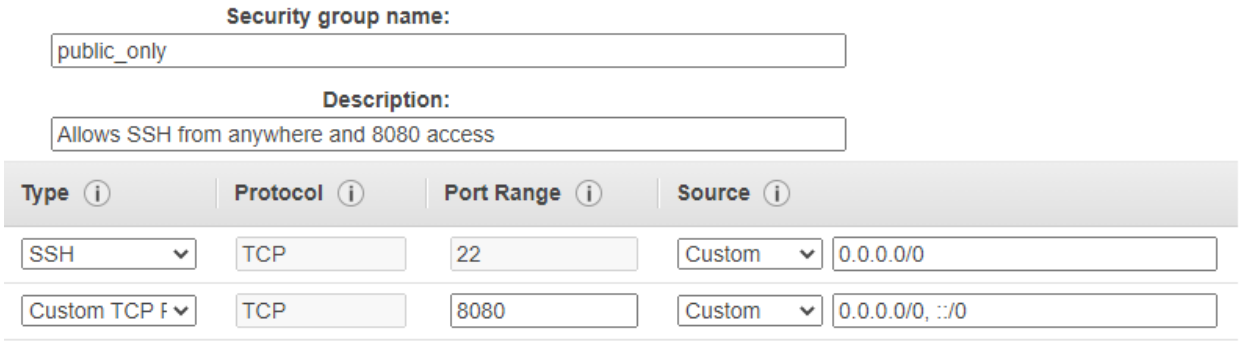
```
#!/bin/bash

sudo amazon-linux-extras install java-openjdk11
sudo amazon-linux-extras install epel
sudo wget -O /etc/yum.repos.d/jenkins.repo \
https://pkg.jenkins.io/redhat-stable/jenkins.repo
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
sudo yum upgrade -y
sudo yum install epel-release java-11-openjdk-devel -y
sudo yum install jenkins -y
sudo systemctl start jenkins
sudo yum install git -y
```



Tag



Security Group

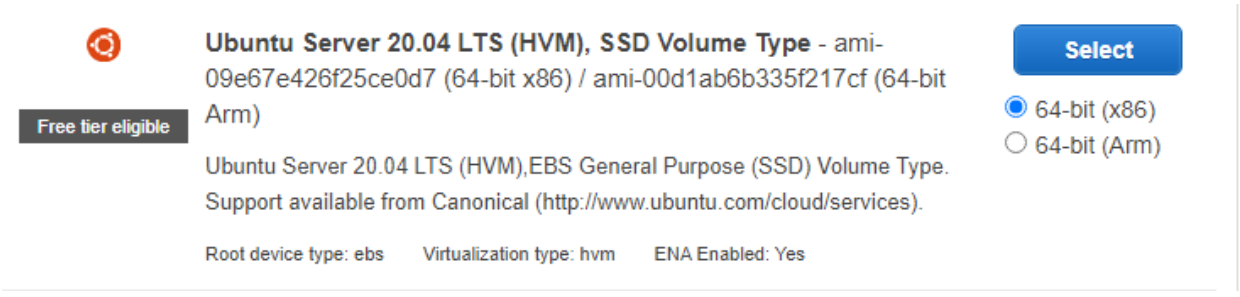


Select your keypair and Launch it

SSH into that EC2 instance
ssh -i key.pem **ec2-user**@PublicIPv4

<u>Setting up the second EC2s</u>

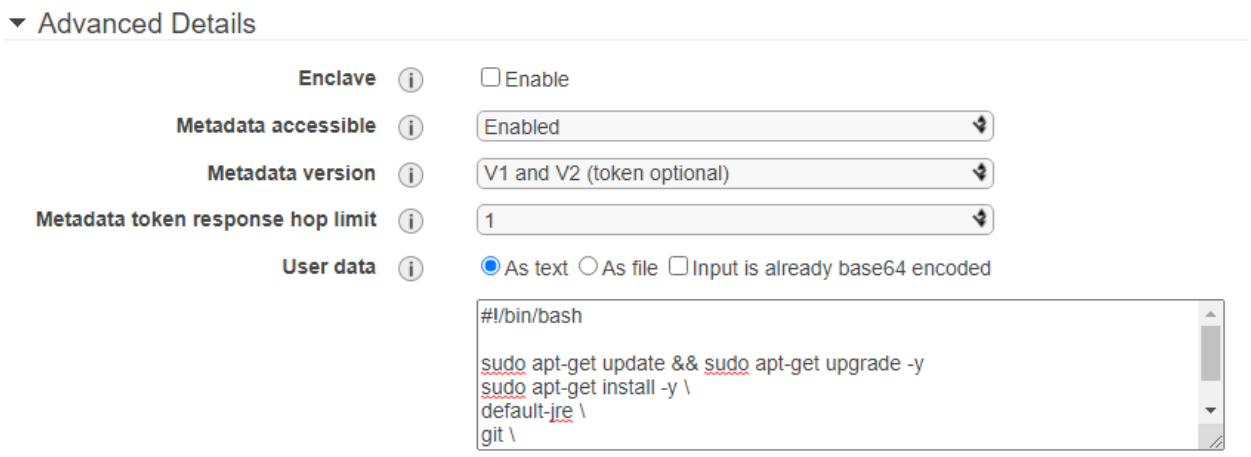
This instance will use Ubuntu, AMI.



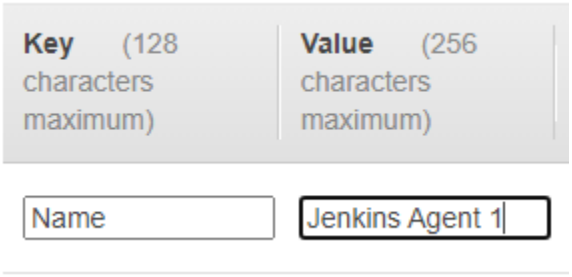
T2.micro for instance type

User data script

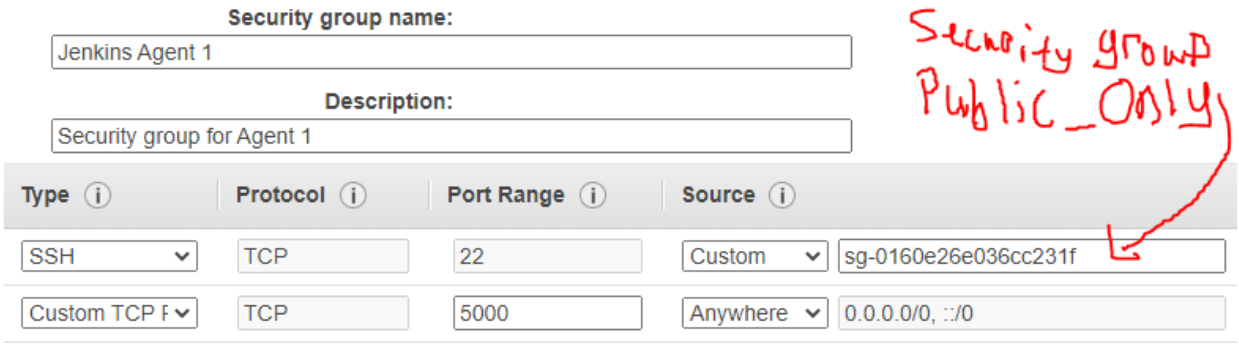
```
#!/bin/bash

sudo apt-get update && sudo apt-get upgrade -y
sudo apt-get install -y \
default-jre \
git \
nodejs -y \
npm -y
```



Tag



Security Group

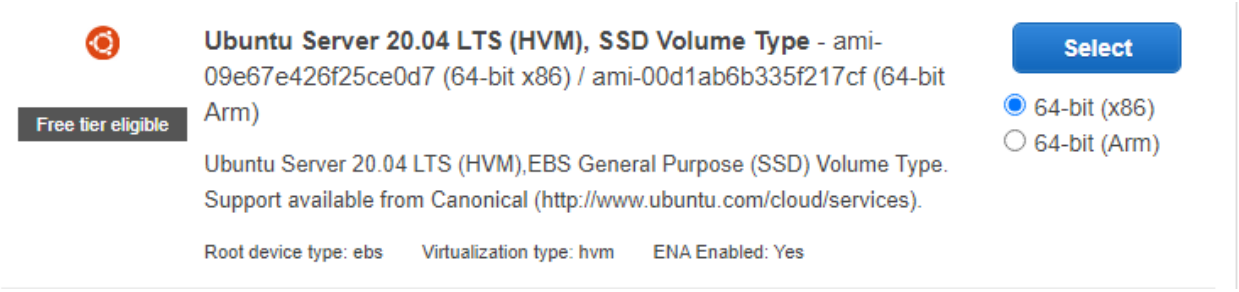


Select your keypair and Launch it

SSH into that EC2 instance
Create a file called linux.pem
Paste the RSA key into that file
Change the permissions using chmod 400 linux.pem
ssh -i linux.pem **ubutnu**@privateIPv4
privateIPv4 is the jenkins agent 1 private ipv4

Setting up the third EC2s

This instance will use Ubuntu, AMI.



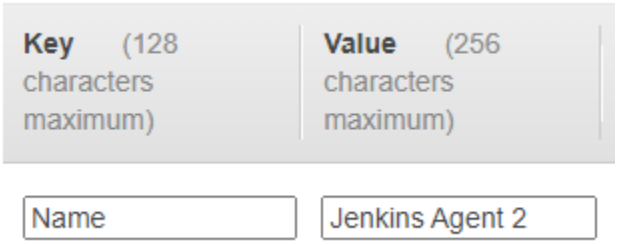
T2.micro for instance type

User data script

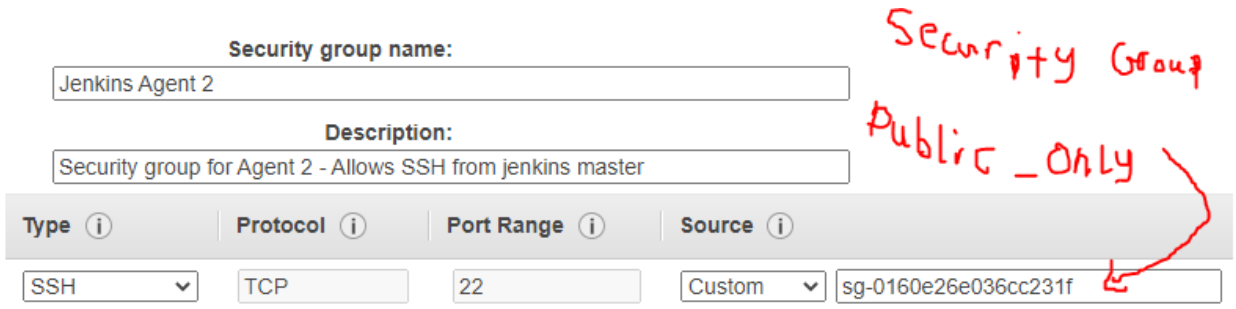
```
#!/bin/bash

sudo apt-get update && sudo apt-get upgrade -y
sudo apt-get install -y \
default-jre \
git \
nodejs -y \
npm -y \
maven \
libgtk2.0-0 \
libgtk-3-0 \
libgbm-dev \
libnotify-dev \
libgconf-2-4 \
libnss3 \
libxss1 \
```

```
libasound2 \
libxtst6 \
xauth \
xvfb
```

Tag



Security Group



Select your keypair and Launch it

SSH into the Jenkins Master EC2 instance
Create a file called linux.pem
Paste the RSA key into that file
Change the permissions using chmod 400 linux.pem
ssh -i linux.pem ubutnu@privateIPv4
privateIPv4 is the jenkins agent 2 private ipv4
Once inside the AMI, it will take a couple of minutes to install all dependencies.

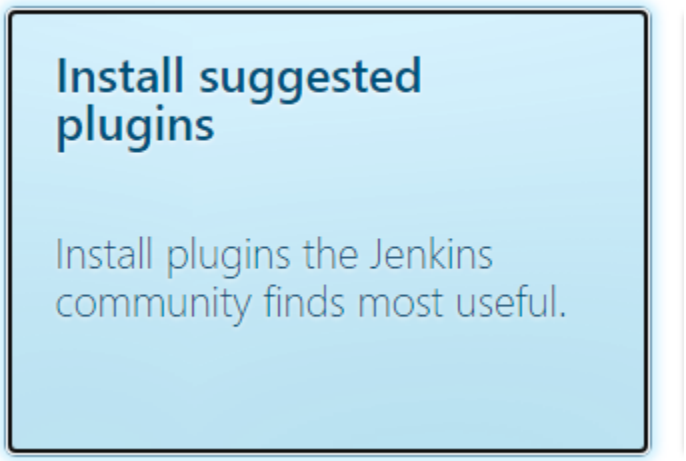
<u>After setting up EC2</u>

Once all the EC2's are set up, connect to your Jenkins application using the public ipv4 of Jenkins Master followed by the port 8080 -> ipv4:8080

SSH into the jenkins master
ssh -i .\key.pem ec2-user@publicIPv4

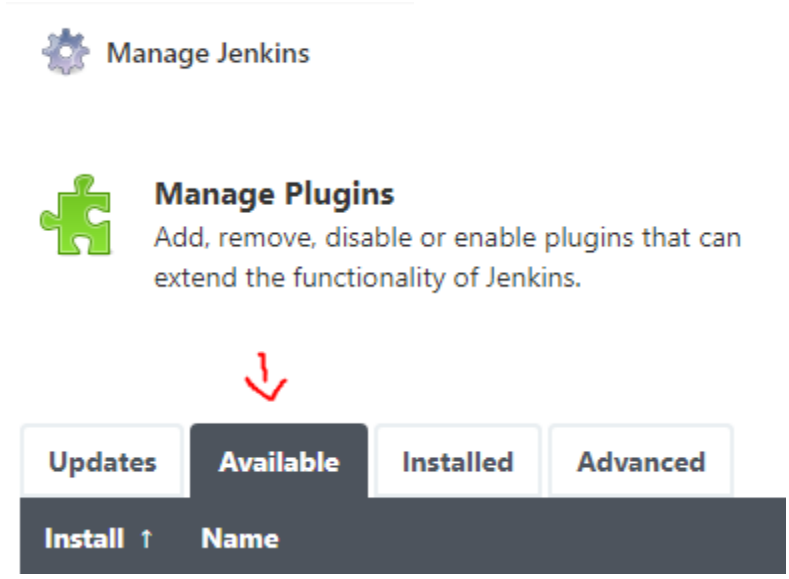
Obtain the Jenkins password and paste the value into jenkins
sudo cat /var/lib/jenkins/secrets/initialAdminPassword

Install the suggested plugins



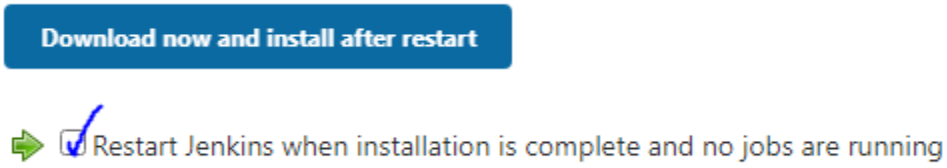
Create an account

Once Jenkins is setup, you have to download some plugins

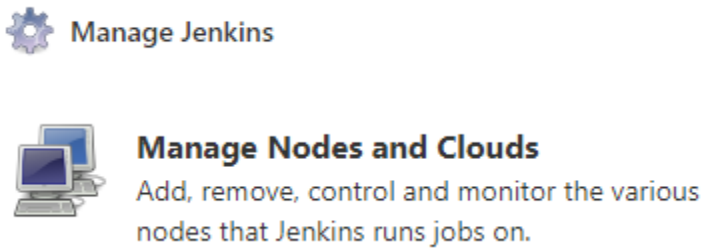


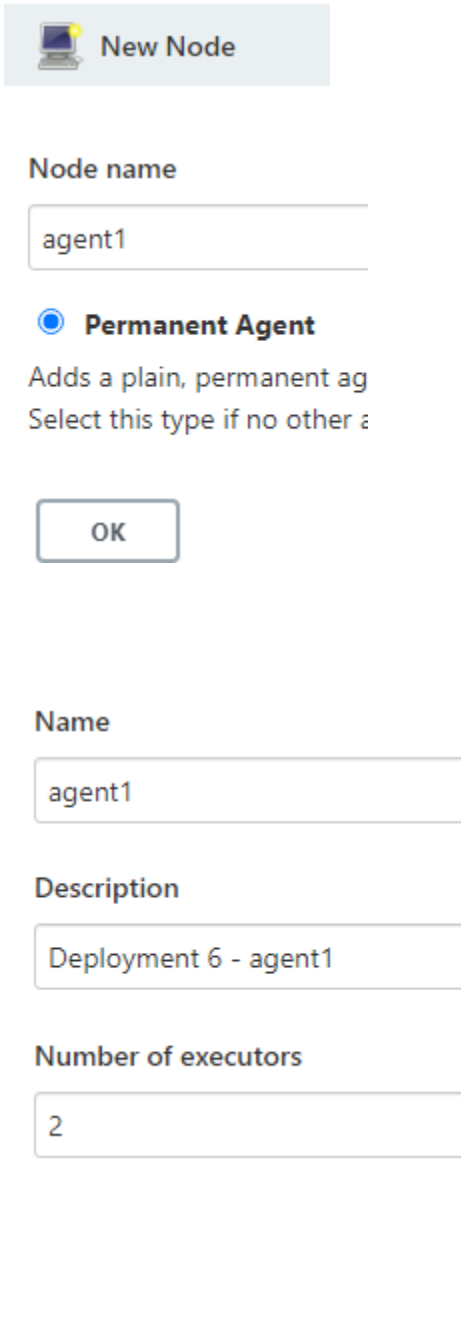
☑	**NodeJS** npm NodeJS Plugin executes NodeJS script as a build step.	1.4.0	6 mo 15 days ago
☑	**Amazon EC2** agent aws Cloud Providers Cluster Management and Distributed Build spotinst This plugin integrates Jenkins with Amazon EC2 or anything implementing the EC2 API's such as an Ubuntu.	1.64	3 days 5 hr ago
☑	**Maven Integration** Build Tools This plug-in provides, for better and for worse, a deep integration of Jenkins and Maven: Automatic triggers between projects depending on SNAPSHOTs, automated configuration of various Jenkins publishers (Junit, ...).	3.12	3 mo 11 days ago

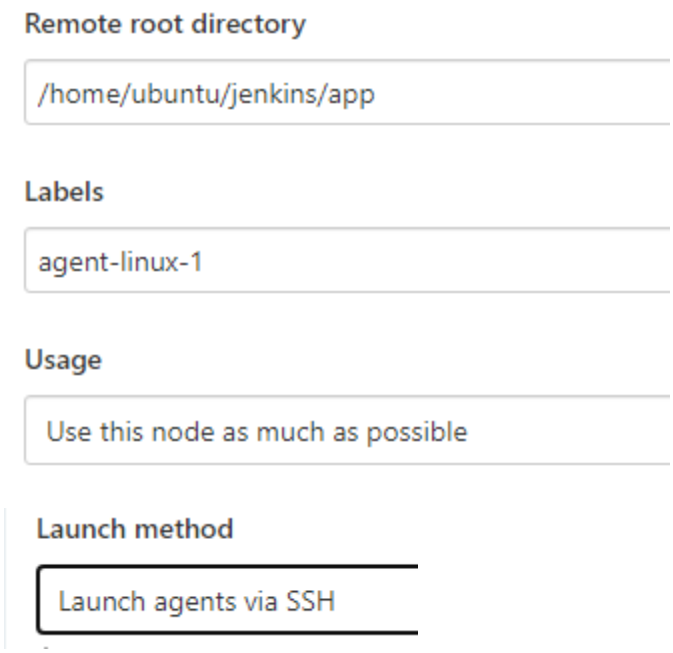
Once you selected all 3 plugins, download now and install after restart



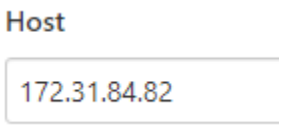
Create two different Agents on Jenkins



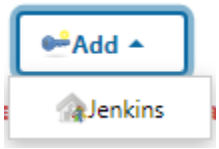

New Node
Node name
agent1
Permanent Agent
Adds a plain, permanent ag
Select this type if no other a
OK
Name
agent1
Description
Deployment 6 - agent1
Number of executors
2

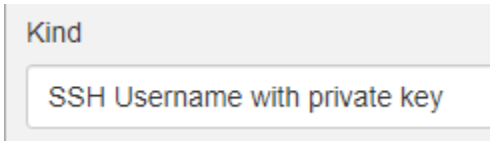


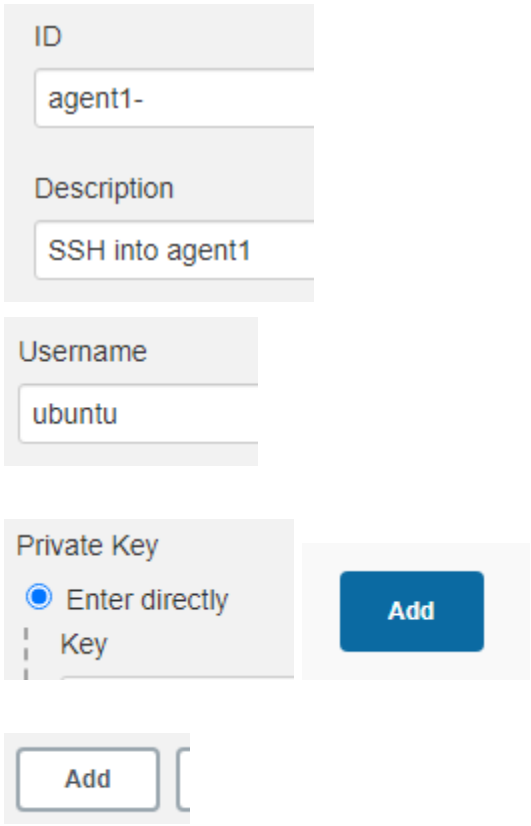
So the host should be the private IPv4 of the agent you are creating
Private IPv4 of agent1



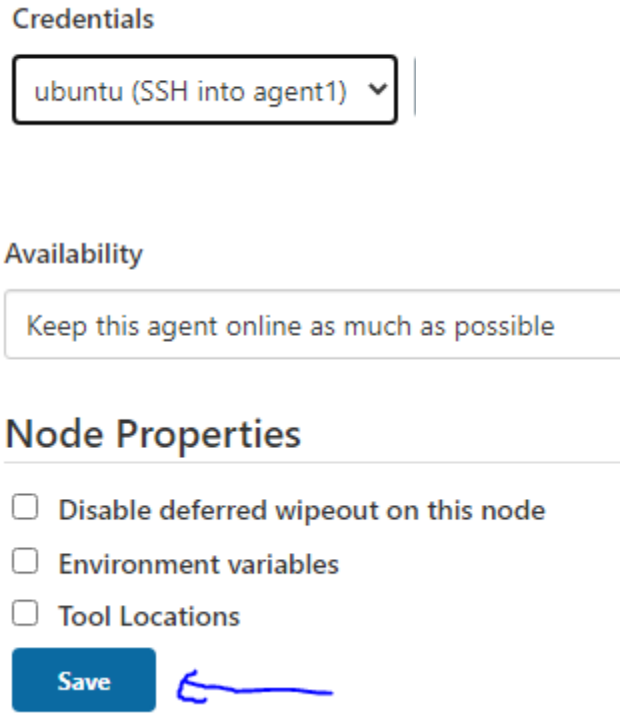
Add credentials

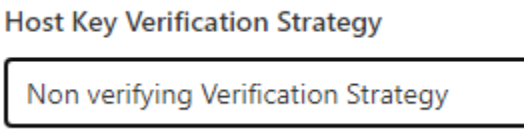




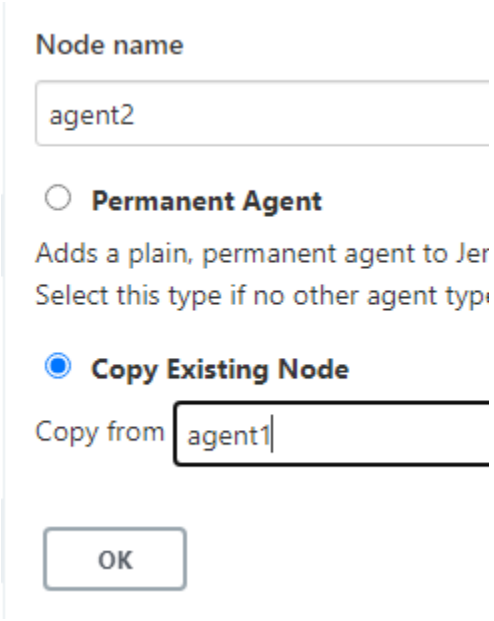


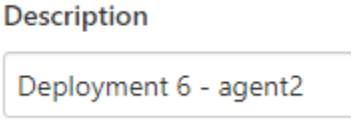
Select the credentials

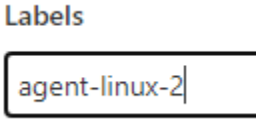




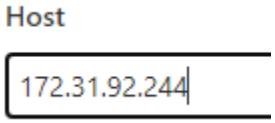
Create the second node





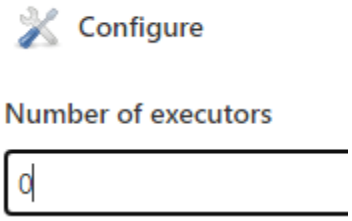


Change the host IP to agent 2’s private IPv4

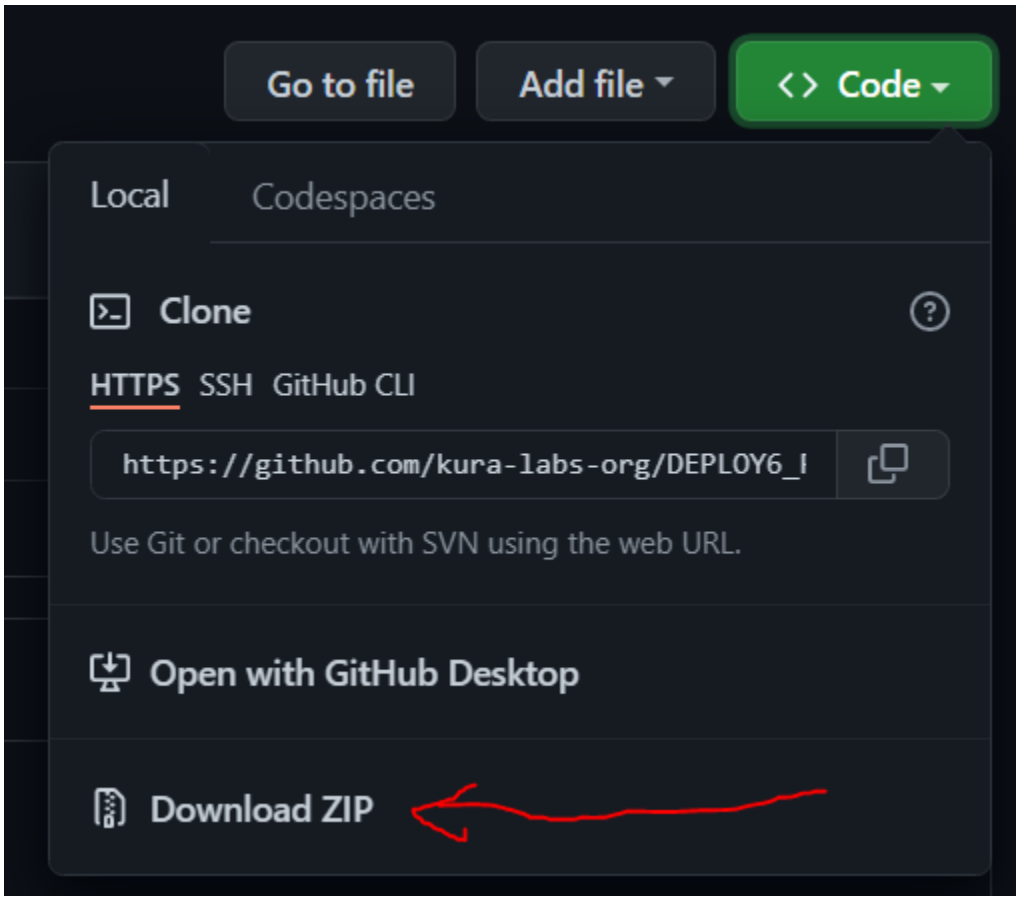


Save it

Go into master

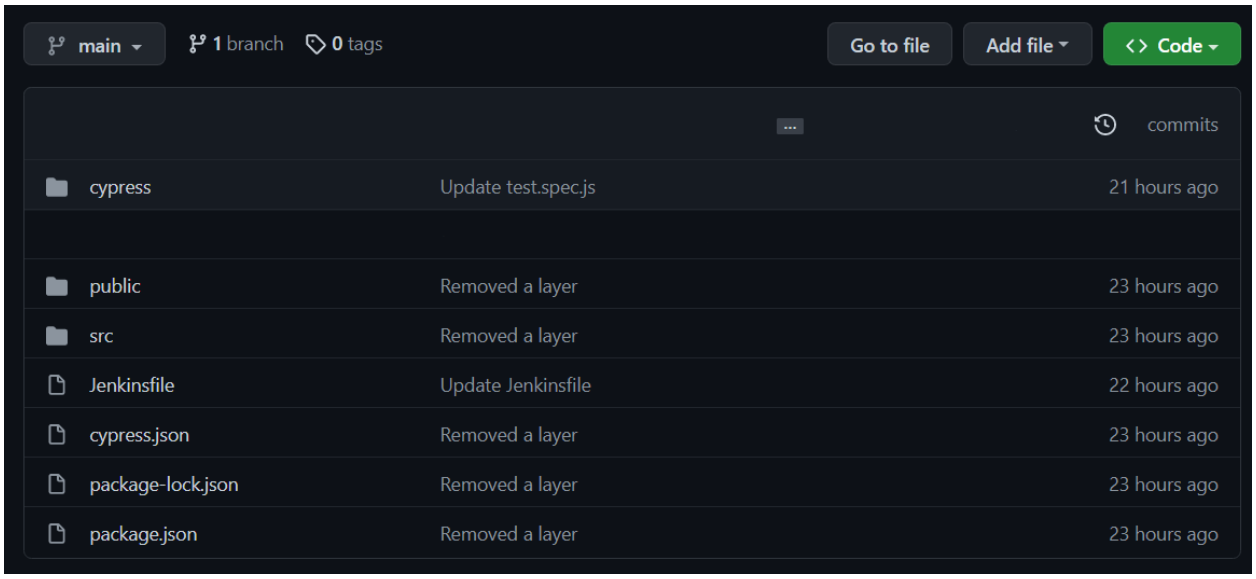


Go to Kura GitHub -> https://github.com/kura-labs-org/DEPLOY6_FE
Download the repository as a ZIP



Once downloaded, extract it to your desktop
Delete the README.md and Deployment#6.pdf

Create a new repository on GitHub and upload the contents in kura_test_repo folder to that repository



Navigate to the Jenkinsfile inside the folder and edit it.

```
pipeline {
  agent {
      label 'agent-linux-1'
  }
  stages {
    stage ('Build') {
      steps {
      sh 'rm -rf ./kura_test_repo/cypress2'
      sh '''
        npm install
        npm run build
        sudo npm install -g serve
        serve -s build &
        '''
      }
    }
    stage ('Second') {
      agent {
        label 'agent-linux-2'
      }
      steps {
```

```
        sh '''
          npm install cypress
          npm install mocha
          npx cypress run --spec ./cypress/integration/test.spec.js
          '''
        }
        post {
          always {
            junit 'results/cypress-report.xml'
          }

        }
      }
    }
  }
```

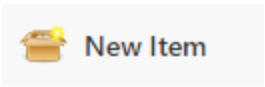
Navigate to the test.specs.js file inside the repository
Root -> cypress -> integration -> test.spec.js

10 lines (8 sloc) | 252 Bytes

```
1  describe('Heading', () => {
2      it('has the right title', () => {
3          cy.visit('http://3.83.219.67:5000/example-1')
4
5          cy.get('h1')
6              .invoke('text')
7              .should("equal", "My Awesome Web Application")
8      });
9
10 });
```

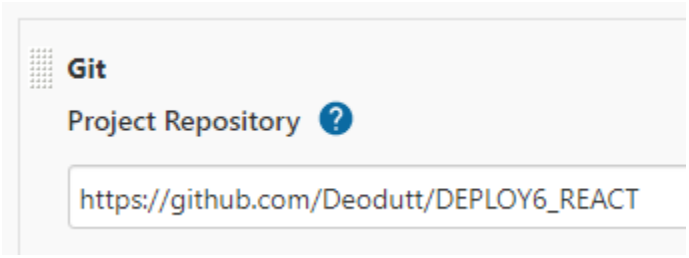
Edit the IP address in line 3.
It should be the private IP address of the agent 1.

Go back into Jenkins and create a new item

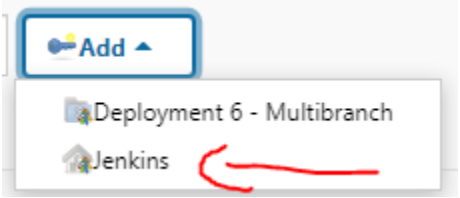


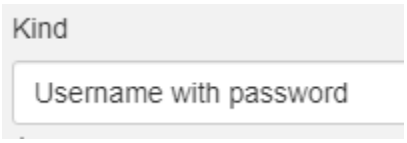
Name it and select Multibranch pipeline project

Specify the Git

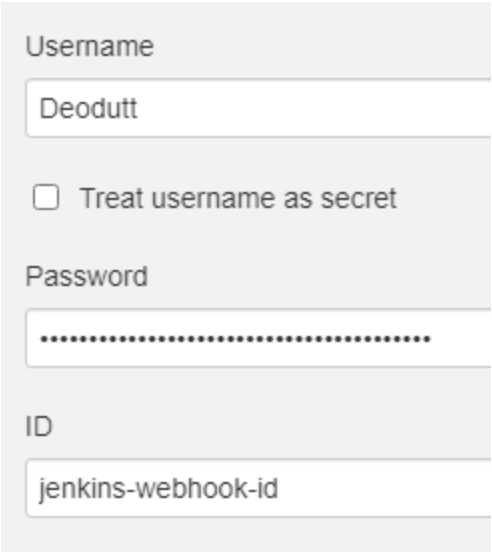


Create a credentials



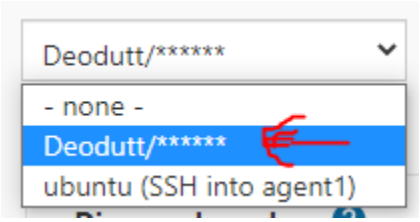


For Username put your GitHub username
For password put your GitHub Personal Access Token
If yours expired you can create a new one -> here

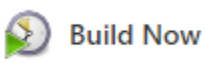


Add

Select the credentials from dropdown



Make sure the Script path matches your repository layout.



To build, test and deploy you will use the build file that's located inside agent 1.