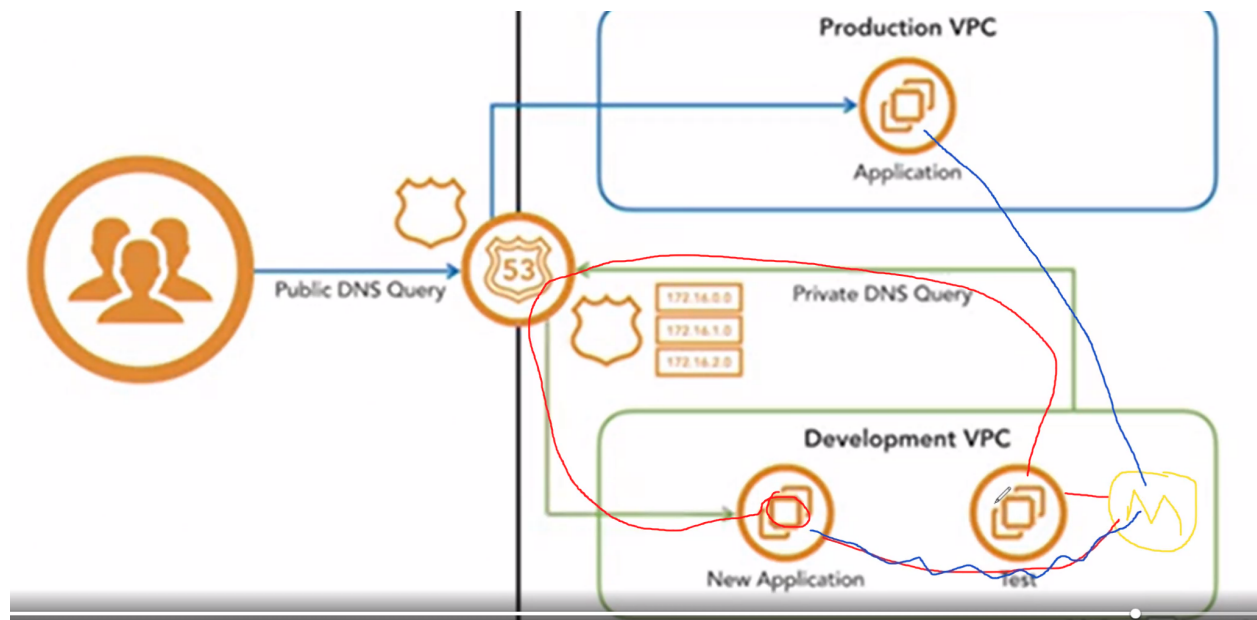


## Deployment 6

### 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 cyress is the test and the application gets built and deployed.

The test will only happen when you give instructions to cyress. 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.

**Amazon Linux**  
Free tier eligible

**Amazon Linux 2 AMI (HVM), SSD Volume Type** - ami-087c17d1fe0178315 (64-bit x86) / ami-029c64b3c205e6cce (64-bit Arm)  
  
Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is approaching end of life on December 31, 2020 and has been removed from this wizard.  
  
Root device type: ebs    Virtualization type: hvm    ENA Enabled: Yes

**Select**  
☒ 64-bit (x86)  
☐ 64-bit (Arm)

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
```

### ▼ Advanced Details

|  |   |
|--|---|
| <b>Enclave</b> ⓘ                           | <input type="checkbox"/> Enable   |
| <b>Metadata accessible</b> ⓘ               | <input type="text" value="Enabled"/>  |
| <b>Metadata version</b> ⓘ                  | <input type="text" value="V1 and V2 (token optional)"/>   |
| <b>Metadata token response hop limit</b> ⓘ | <input type="text" value="1"/>  |
| <b>User data</b> ⓘ                         | <input checked="" type="radio"/> As text <input type="radio"/> As file <input type="checkbox"/> Input is already base64 encoded |

```
https://pkg.jenkins.io/redhat-stable/jenkins.repo
sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key
sudo yum upgrade
sudo yum install epel-release java-11-openjdk-devel
sudo yum install jenkins
sudo systemctl start jenkins
```

## Tag

| Key (128 characters maximum) | Value (256 characters maximum) |
|------------------------------|--------------------------------|
| Name                         | Jenkins Master                 |

## Security Group

Security group name:

Description:

| Type ⓘ         | Protocol ⓘ | Port Range ⓘ | Source ⓘ                 |
|----------------|------------|--------------|--------------------------|
| SSH ▾          | TCP        | 22           | Custom ▾ 0.0.0.0/0       |
| Custom TCP F ▾ | TCP        | 8080         | Custom ▾ 0.0.0.0/0, ::/0 |


Select your keypair and Launch it

SSH into that EC2 instance

```
ssh -i key.pem ec2-user@PublicIPv4
```

## Setting up the second EC2s

This instance will use Ubuntu, AMI.

  
**Free tier eligible**

**Ubuntu Server 20.04 LTS (HVM), SSD Volume Type** - ami-09e67e426f25ce0d7 (64-bit x86) / ami-00d1ab6b335f217cf (64-bit Arm)  
Ubuntu Server 20.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).  
Root device type: ebs    Virtualization type: hvm    ENA Enabled: Yes

**Select**  
☒ 64-bit (x86)  
☐ 64-bit (Arm)

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
```

#### ▼ Advanced Details

|                                   |   |
|-----------------------------------|---|
| Enclave                           | <input type="checkbox"/> Enable   |
| Metadata accessible               | Enabled   |
| Metadata version                  | V1 and V2 (token optional)  |
| Metadata token response hop limit | 1   |
| User data                         | <input checked="" type="radio"/> As text <input type="radio"/> As file <input type="checkbox"/> Input is already base64 encoded |

```
#!/bin/bash  
  
sudo apt-get update && sudo apt-get upgrade -y  
sudo apt-get install -y \  
default-jre \  
git \
```

#### Tag

|                                     |                                       |
|-------------------------------------|---------------------------------------|
| <b>Key</b> (128 characters maximum) | <b>Value</b> (256 characters maximum) |
| Name                                | Jenkins Agent 1                       |

#### Security Group

|                            |          |            |          |                      |
|----------------------------|----------|------------|----------|----------------------|
| Security group name:       |          |            |          |                      |
| Jenkins Agent 1            |          |            |          |                      |
| Description:               |          |            |          |                      |
| Security group for Agent 1 |          |            |          |                      |
| Type                       | Protocol | Port Range | Source   |                      |
| SSH                        | TCP      | 22         | Custom   | sg-0160e26e036cc231f |
| Custom TCP F               | TCP      | 5000       | Anywhere | 0.0.0.0/0, ::/0      |


Security group  
Public Only

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 ubuntu@privateIPv4`  
privateIPv4 is the jenkins agent 1 private ipv4

### Setting up the third EC2s

This instance will use Ubuntu, AMI.

**Ubuntu Server 20.04 LTS (HVM), SSD Volume Type** - ami-09e67e426f25ce0d7 (64-bit x86) / ami-00d1ab6b335f217cf (64-bit Arm)  
**Free tier eligible**

Ubuntu Server 20.04 LTS (HVM),EBS General Purpose (SSD) Volume Type.  
Support available from Canonical (<http://www.ubuntu.com/cloud/services>).  
Root device type: ebs    Virtualization type: hvm    ENA Enabled: Yes

**Select**  
☒ 64-bit (x86)  
☐ 64-bit (Arm)

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

| Key (128 characters maximum) | Value (256 characters maximum) |
|------------------------------|--------------------------------|
| Name                         | Jenkins Agent 2                |

Security Group

|   |            |              |          |                      |
|---|------------|--------------|----------|----------------------|
| Security group name:  |            |              |          |                      |
| Jenkins Agent 2   |            |              |          |                      |
| Description:  |            |              |          |                      |
| Security group for Agent 2 - Allows SSH from jenkins master |            |              |          |                      |
| Type ⓘ  | Protocol ⓘ | Port Range ⓘ | Source ⓘ |                      |
| SSH ▼   | TCP        | 22           | Custom ▼ | sg-0160e26e036cc231f |

*Security Group Public Only*

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 ubuntu@privateIPv4`**

privateIPv4 is the jenkins agent 2 private ipv4

Once inside the AMI, it will take a couple of minutes to install all dependencies.

### After setting up EC2

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`

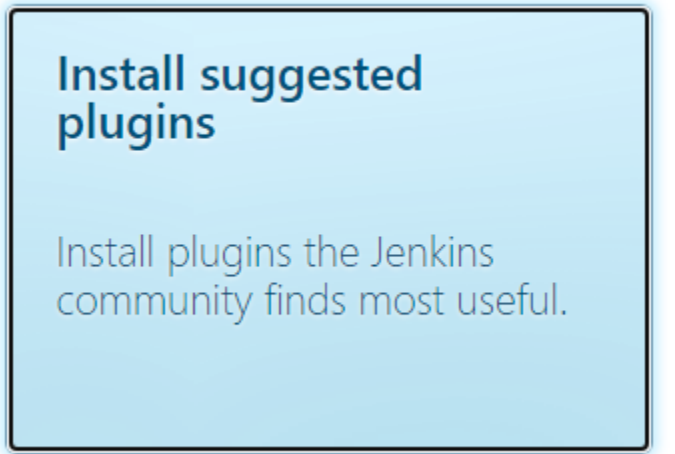
SSH into the jenkins master

```
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



Manage Jenkins



### Manage Plugins

Add, remove, disable or enable plugins that can extend the functionality of Jenkins.



| Updates   | Available | Installed | Advanced |
|-----------|-----------|-----------|----------|
| Install ↑ | Name      |           |          |


|                                     |   |       |                  |
|-------------------------------------|---|-------|------------------|
| <input checked="" type="checkbox"/> | <b>NodeJS</b><br><div>npm</div><br>NodeJS Plugin executes <b>NodeJS</b> script as a build step.   | 1.4.0 | 6 mo 15 days ago |
| <input checked="" type="checkbox"/> | <b>Amazon EC2</b><br><div>agent   aws   Cloud Providers</div> <div>Cluster Management and Distributed Build</div> <div>spotinst</div><br>This plugin integrates Jenkins with <b>Amazon EC2</b> or anything implementing the EC2 API's such as an Ubuntu.                        | 1.64  | 3 days 5 hr ago  |
| <input checked="" type="checkbox"/> | <b>Maven Integration</b><br><div>Build Tools</div><br>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

**Download now and install after restart**

➡ ☒ Restart Jenkins when installation is complete and no jobs are running

Create two different Agents on Jenkins

 **Manage Jenkins**



### **Manage Nodes and Clouds**

Add, remove, control and monitor the various nodes that Jenkins runs jobs on.





## New Node

Node name

agent1

☒ **Permanent Agent**

Adds a plain, permanent agent.  
Select this type if no other agent is present.

OK

Name

agent1

Description

Deployment 6 - agent1

Number of executors

2

### Remote root directory

/home/ubuntu/jenkins/app

### Labels

agent-linux-1

### Usage

Use this node as much as possible

### Launch method

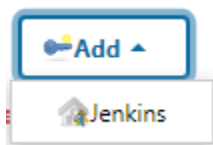
Launch agents via SSH

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

### Host

172.31.84.82

### Add credentials



### Kind

SSH Username with private key

ID

agent1-

Description

SSH into agent1

Username

ubuntu

Private Key

☒ Enter directly

Key

Add

Add

Select the credentials

Credentials

ubuntu (SSH into agent1) ▼

Availability

Keep this agent online as much as possible

## Node Properties

- ☐ Disable deferred wipeout on this node
- ☐ Environment variables
- ☐ Tool Locations

Save



### Host Key Verification Strategy

Non verifying Verification Strategy

Create the second node

#### Node name

agent2

☐ **Permanent Agent**

Adds a plain, permanent agent to Jer  
Select this type if no other agent type

☒ **Copy Existing Node**

Copy from agent1

OK

#### Description

Deployment 6 - agent2

#### Labels

agent-linux-2

Change the host IP to agent 2's private IPv4

#### Host

172.31.92.244

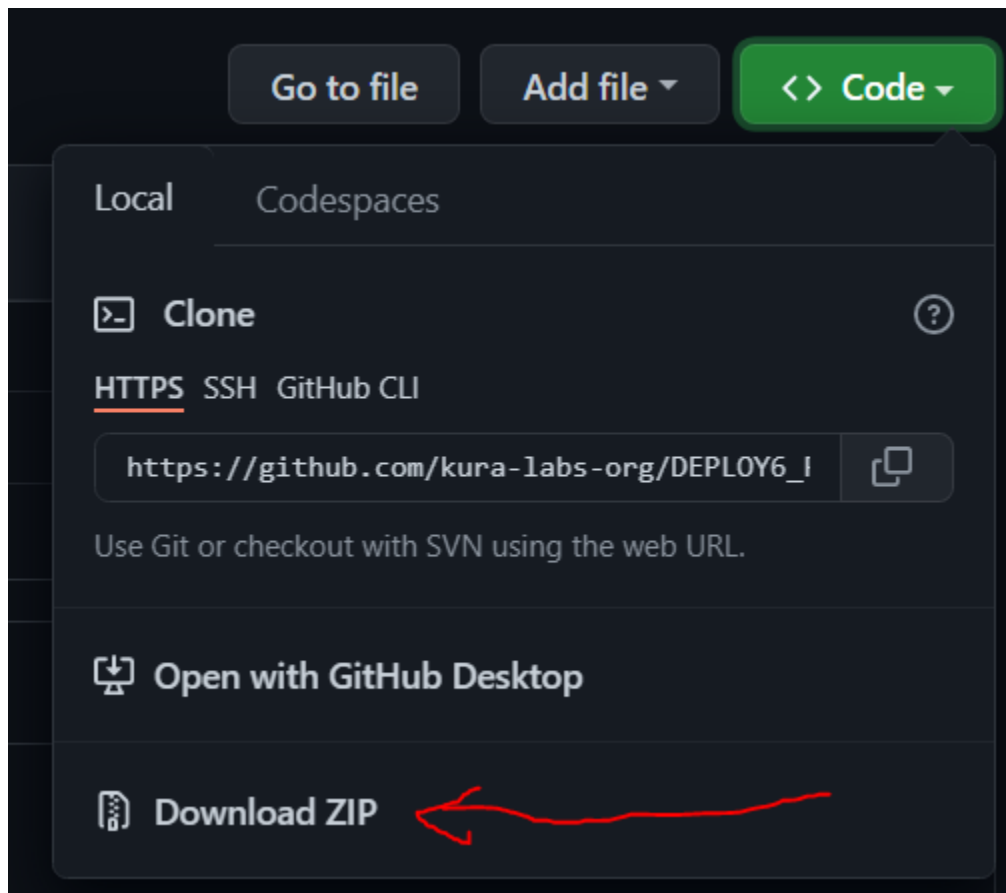
Save it

Go into master



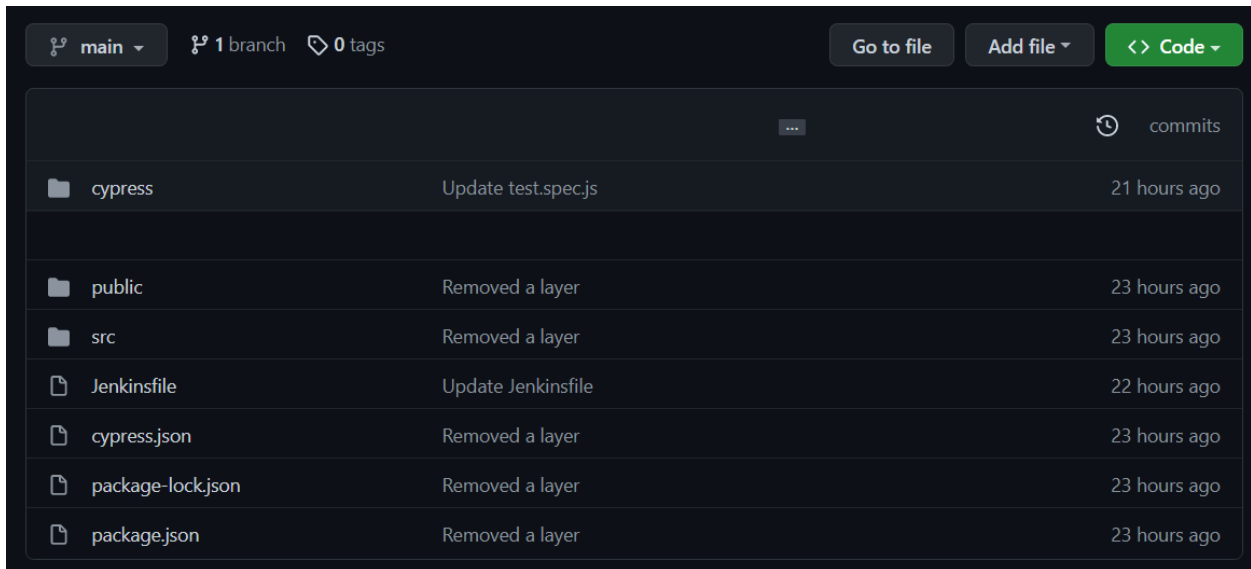
Number of executors

Go to Kura GitHub -> [https://github.com/kura-labs-org/DEPLOY6\\_FE](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



The screenshot shows a GitHub repository interface. At the top, there are buttons for 'main', '1 branch', '0 tags', 'Go to file', 'Add file', and 'Code'. Below this is a table listing the repository's contents. The table has three columns: the file or folder name, the commit message, and the time since the last commit. The files listed are 'cypress', 'public', 'src', 'Jenkinsfile', 'cypress.json', 'package-lock.json', and 'package.json'. The commit messages for 'cypress', 'Jenkinsfile', and 'package.json' are 'Update test.spec.js', 'Update Jenkinsfile', and 'Removed a layer' respectively. The other files have the message 'Removed a layer'. The times range from '21 hours ago' to '23 hours ago'.

| 📁 cypress           | Update test.spec.js | 21 hours ago |
|---------------------|---------------------|--------------|
| 📁 public            | Removed a layer     | 23 hours ago |
| 📁 src               | Removed a layer     | 23 hours ago |
| 📄 Jenkinsfile       | Update Jenkinsfile  | 22 hours ago |
| 📄 cypress.json      | Removed a layer     | 23 hours ago |
| 📄 package-lock.json | Removed a layer     | 23 hours ago |
| 📄 package.json      | Removed a layer     | 23 hours ago |

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

```

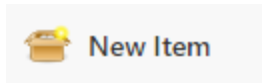
10 lines (8 sloc) | 252 Bytes

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

Display Name

Deployment 6

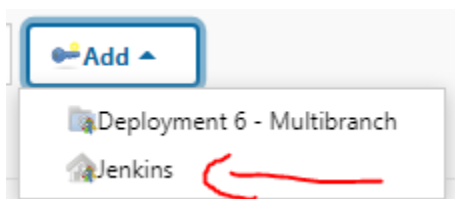
Specify the Git

Git

Project Repository ?

`https://github.com/Deodutt/DEPLOY6_REACT`

Create a credentials



Kind

Username with password

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



Username

Deodutt

☐ Treat username as secret

Password

.....

ID

jenkins-webhook-id

Add

Select the credentials from dropdown

Deodutt/\*\*\*\*\*

- none -

Deodutt/\*\*\*\*\*

ubuntu (SSH into agent1)

Make sure the Script path matches your repository layout.

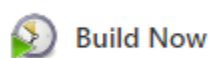
Mode

by Jenkinsfile

Script Path

Jenkinsfile

Save



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