Project 2: DevOps (Step-by-step Implementation)  
Jenkins Declarative Pipeline.  
  
Blog Link: <https://lnkd.in/gHExnqmj>  
  
GitHub Repo: <https://lnkd.in/d_T3JVvf>

# Project 2 — DevOps (Implementation)

# Deploying web app using Jenkins CI/CD declarative pipeline.

Follow the steps:

1. First of all, go to the AWS portal, and create a new instance. As

· Name: jenkins-server

· AMI: ubuntu.

· Instance type: t2.micro (free tier).

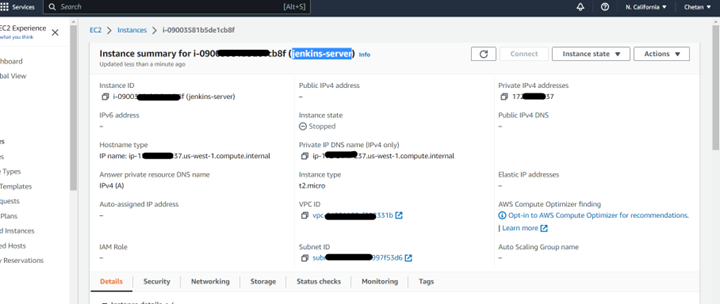
· Key pair login: Create > docker.pem.

· Allow HTTP.

· Allow HTTPS.

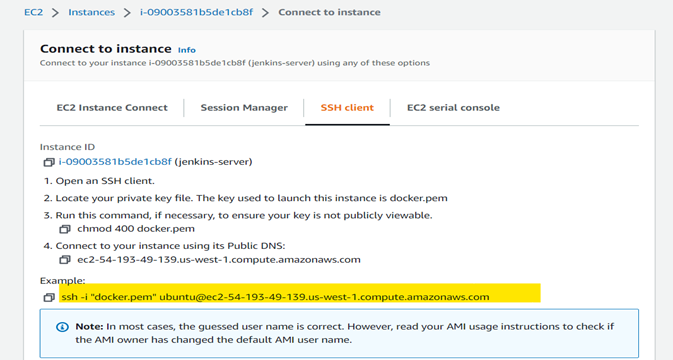
· (Download the .pem file.)

· Click on Launch Instance.



2. Now, we will allow ports 8080 and 8001 for this machine from a security group. We can find the security group in the VM description. Now, here we need to allow “Inbound Rule”

3. Now, connect to the EC2 instance that you have created. Copy the SSH from the server:



4. Go to the download folder, where the .pem file is placed and open the terminal in the same location, and paste the SSH.

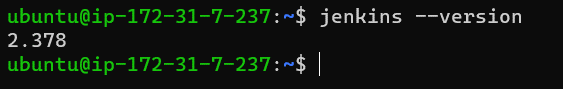
5. Install Jenkins from the following link:

<https://www.jenkins.io/doc/book/installing/linux/>

6. Install Docker by running:

“Sudo apt-install docker.io”

7. Now check if it got installed by running “jenkins — version” and “docker — version”



8. Goto Jenkins Dashboard and Click on “New Item”

· Name: declarative\_pipeline

· Select: Pipeline

Description: This is a demo pipeline project.

9. Pipeline Script:

pipeline{

agent any

stages{

stage{

steps{

git branch: ‘main’, URL: ‘https://github.com/chxtan/react\_django\_demo\_app.git'

}

}

stage{

steps{

echo “Testing”

}

}

stage{

steps{

script{

sh “docker build — no-cache -t react\_django\_demo\_app .”

}

}

}

stage{

steps{

script{

sh “docker run -p 8001:8001 -d react\_django\_demo\_app”

}

}

}

}

}

Or

pipeline {

    agent any

    stages {

        stage("Git clone") {

            steps {

                git branch: 'main', url: 'https://github.com/kshirsagarn/react\_django\_demo\_app.git'

            }

        }

         stage("Test") {

            steps {

                sh 'docker build --no-cache -t project2 .'

            }

        }

         stage("Test") {

            steps {

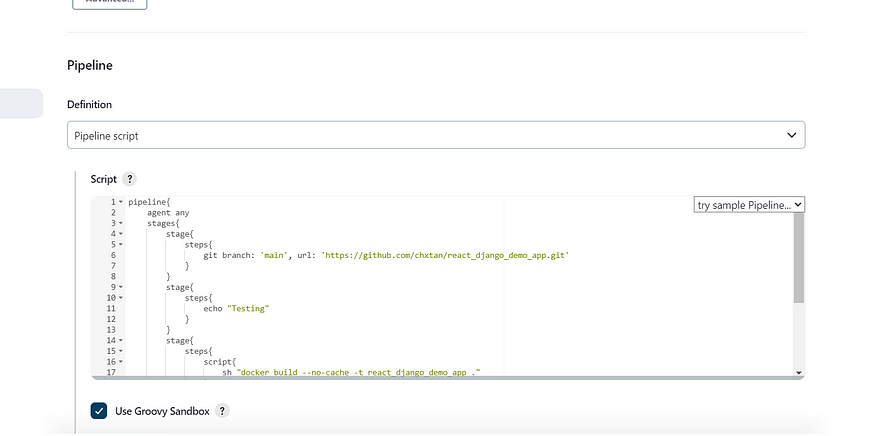
                sh 'docker run -d -p 8001:8001 project2 '

            }

        }

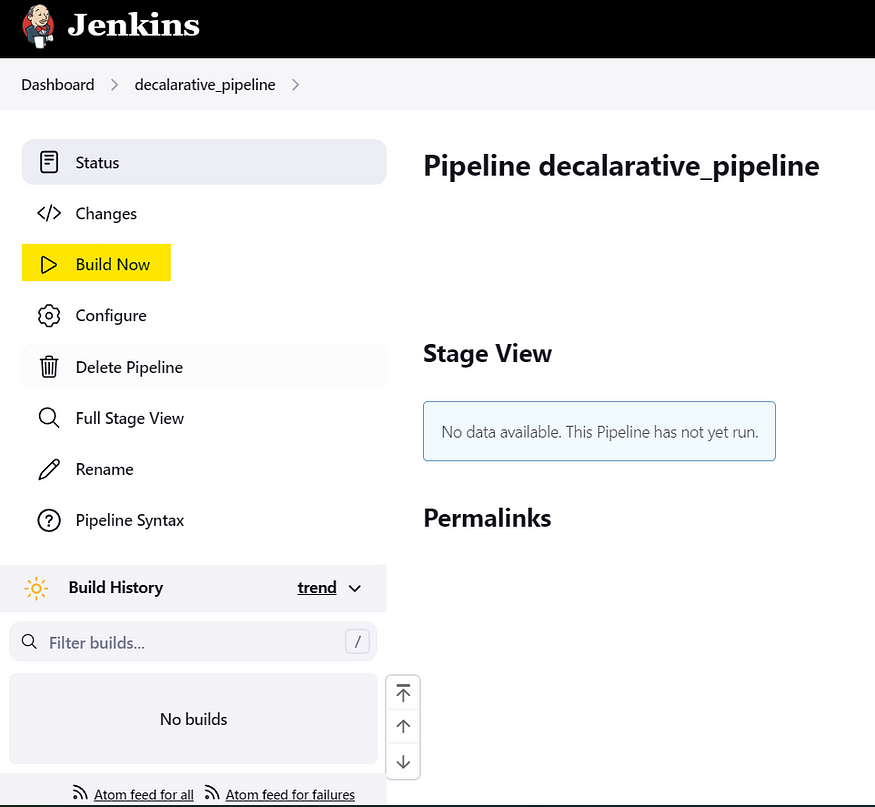
    }

}

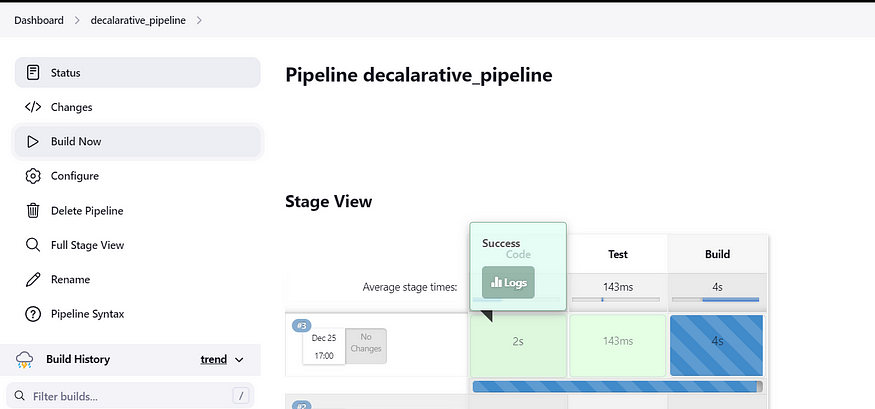


10. Click on “Save”.

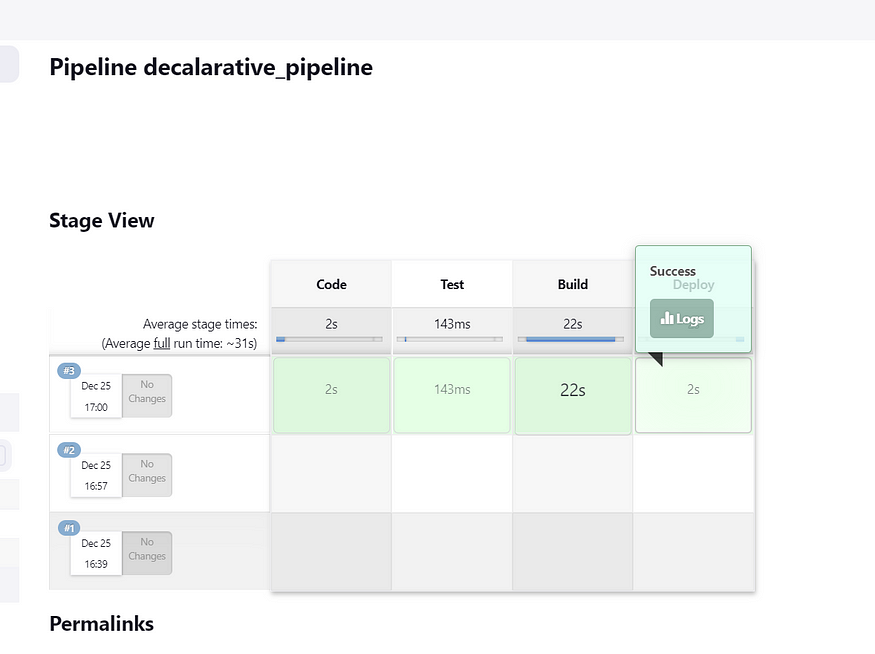
11. Click on “Build Now”.



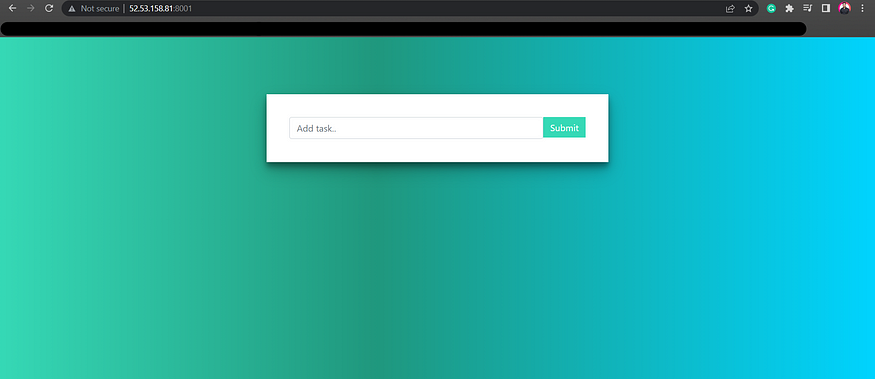
12. We can click on “Stage View” to check the results if it is still under process.



13. Once it will be completed, It will return as “Success”.



14. Now, Search for the Public IP with port 8001 in the browser,



We have built the image of a Web app source code, and deployed that image as a container, using Jenkins Pipelines.

Hope you found this helpful. Do connect/ follow for more such content.

~ Chetan Rakhra.