Automating the Deployment of a React Application on NGINX on EC2 through Jenkins Pipeline

Introduction

This documentation outlines the process of automating the deployment of an React application on an NGINX web server hosted on an Amazon EC2 instance using Jenkins Pipeline. Automation streamlines the deployment process, reduces errors, and ensures consistency with integration of email notification.

Prerequisites

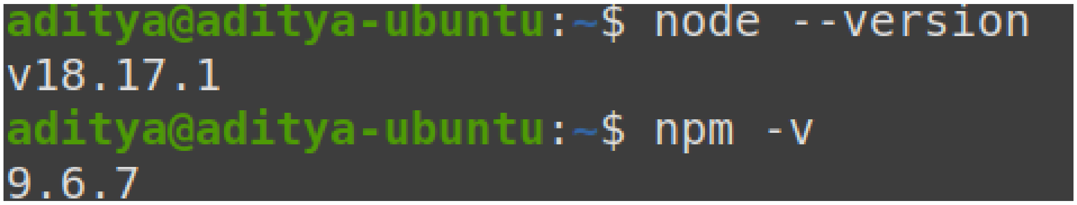
- Nodejs 18.17
- NPM 9.x.x
- Jenkins server
- React application source code in Github.
- Amazon EC2 Instance

Let's go through the steps to achieve this:

Step 1: Install NodeJs and NPM:

Run the following commands on the terminal to add Node.js PPA in your Ubuntu system and install it.

-> curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -
-> sudo apt install nodejs
-> node --version
-> npm -v



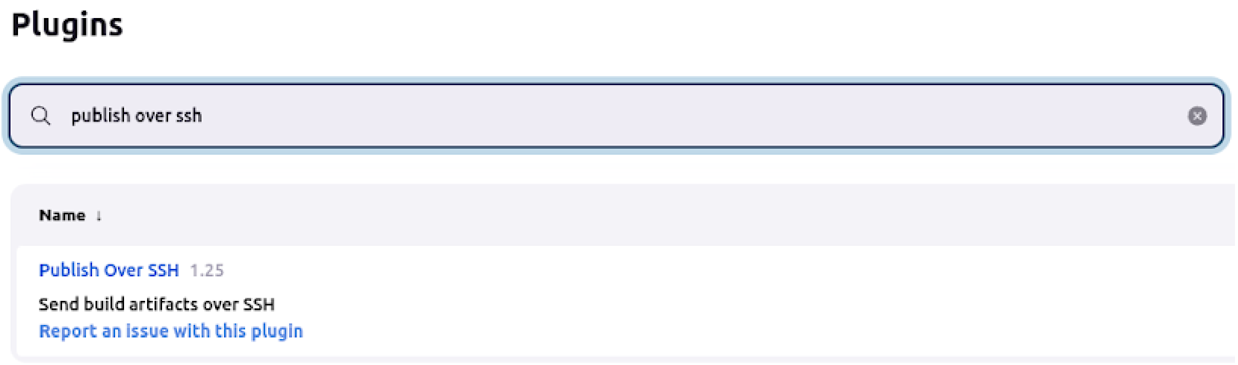
```
aditya@aditya-ubuntu:~$ node --version
v18.17.1
aditya@aditya-ubuntu:~$ npm -v
9.6.7
```

Step 2: Installing Jenkins:

1. To install Jenkins, follow this guide: [link to the page]

Step 3: Installing Plugins in jenkins :

1. In Jenkins, go to Manage Jenkins -> Plugins
2. Search for SSH plugin and Publish Over SSH under Available plugins



Plugins

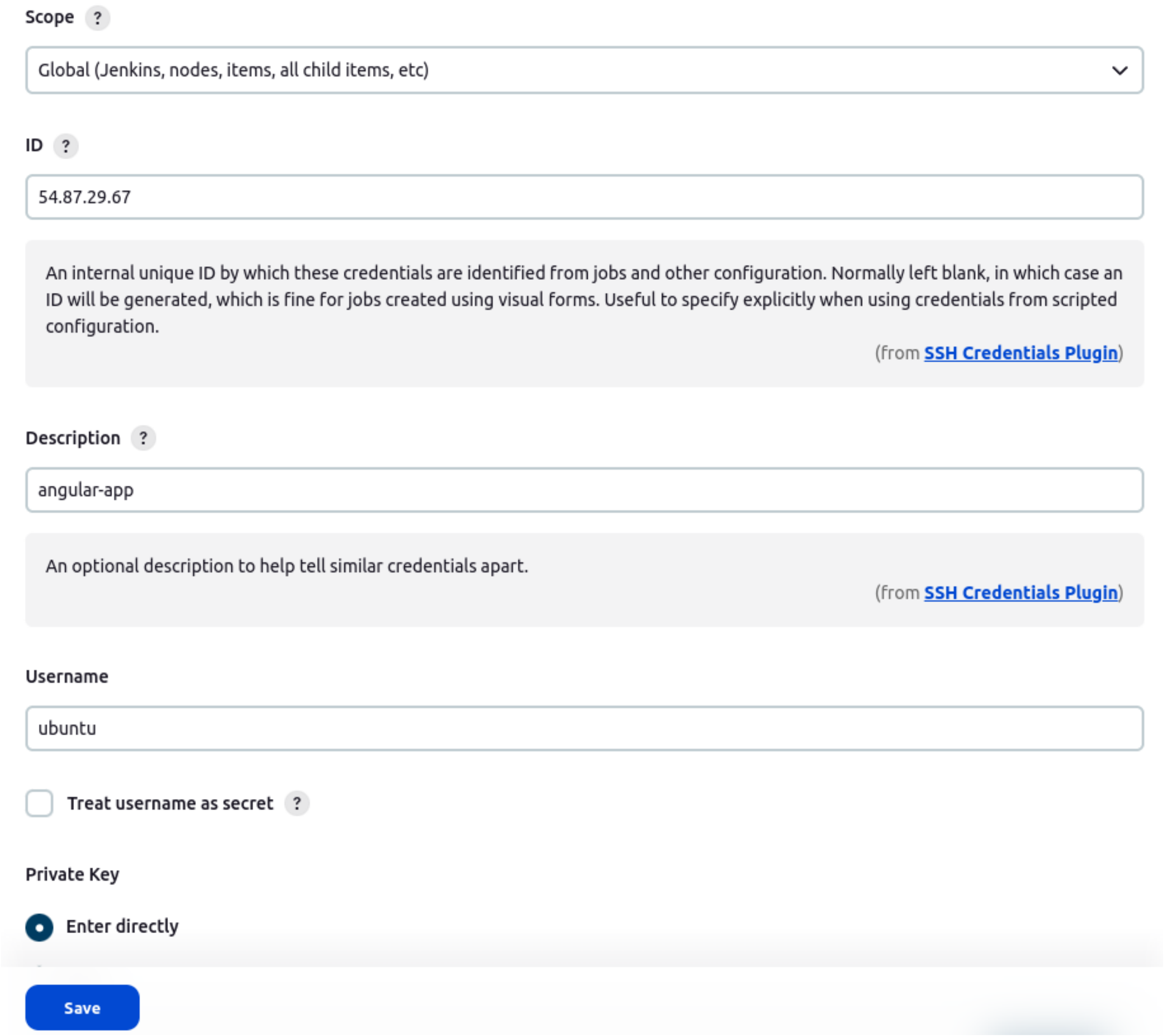
publish over ssh

Name ↓
Publish Over SSH 1.25 Send build artifacts over SSH Report an issue with this plugin

3. Click on Checkbox and Install

Step 3: Create Credentials

1. Navigate to **Manage Jenkins -> Credentials**
2. Click on **Add Credential**
3. In New Credentials, Select “**SSH Username with private key.**”



Scope ?

Global (Jenkins, nodes, items, all child items, etc)

ID ?

54.87.29.67

An internal unique ID by which these credentials are identified from jobs and other configuration. Normally left blank, in which case an ID will be generated, which is fine for jobs created using visual forms. Useful to specify explicitly when using credentials from scripted configuration.

(from SSH Credentials Plugin)

Description ?

angular-app

An optional description to help tell similar credentials apart.

(from SSH Credentials Plugin)

Username

ubuntu

☐ Treat username as secret ?

Private Key

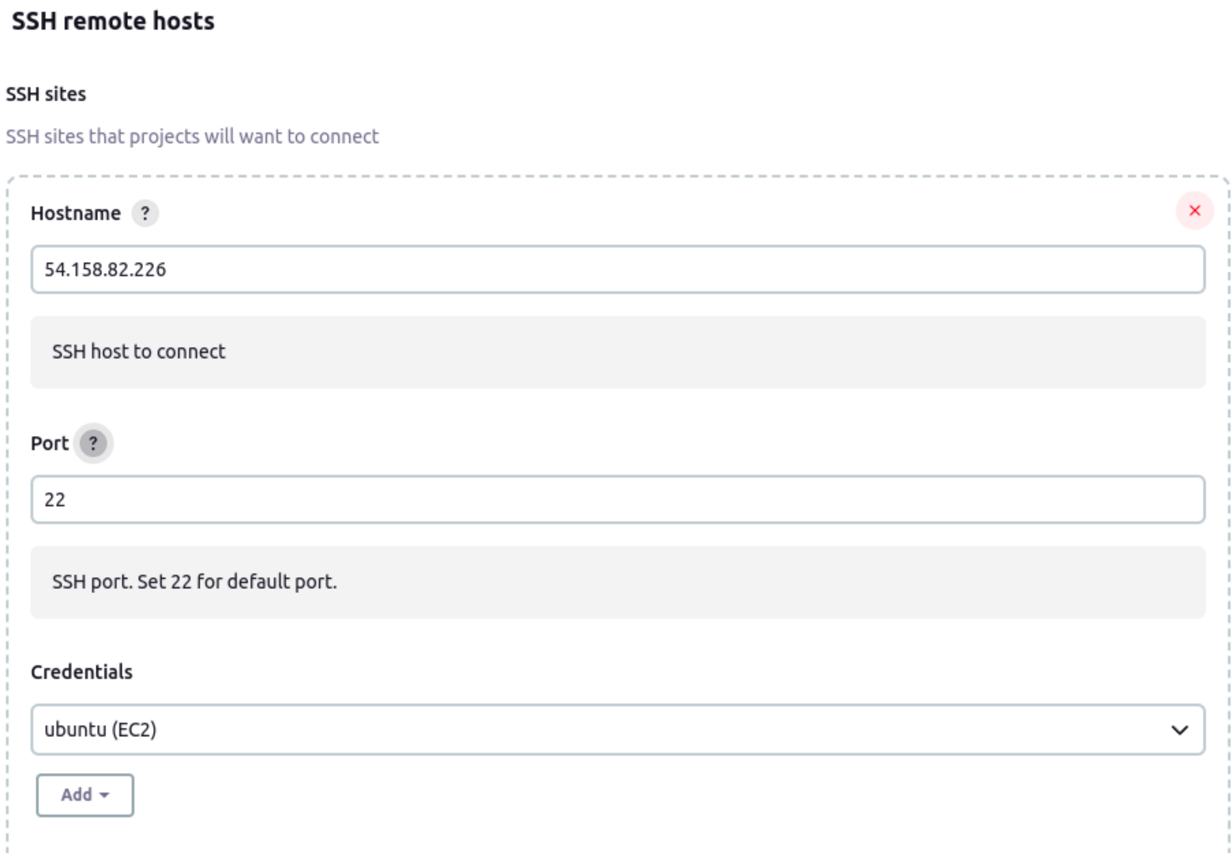
● Enter directly

4. Enter the following for:
    - ID : Provide ec2 public IP address
    - Description: angular-app
    - Username: ubuntu
    - In Private Key, Select directly
    - Then, Paste the key from .pem file.

5. Click on **Create**.

Step 4: Configuring System

In, SSH remote hosts:



SSH remote hosts

SSH sites

SSH sites that projects will want to connect

Hostname: 54.158.82.226

SSH host to connect

Port: 22

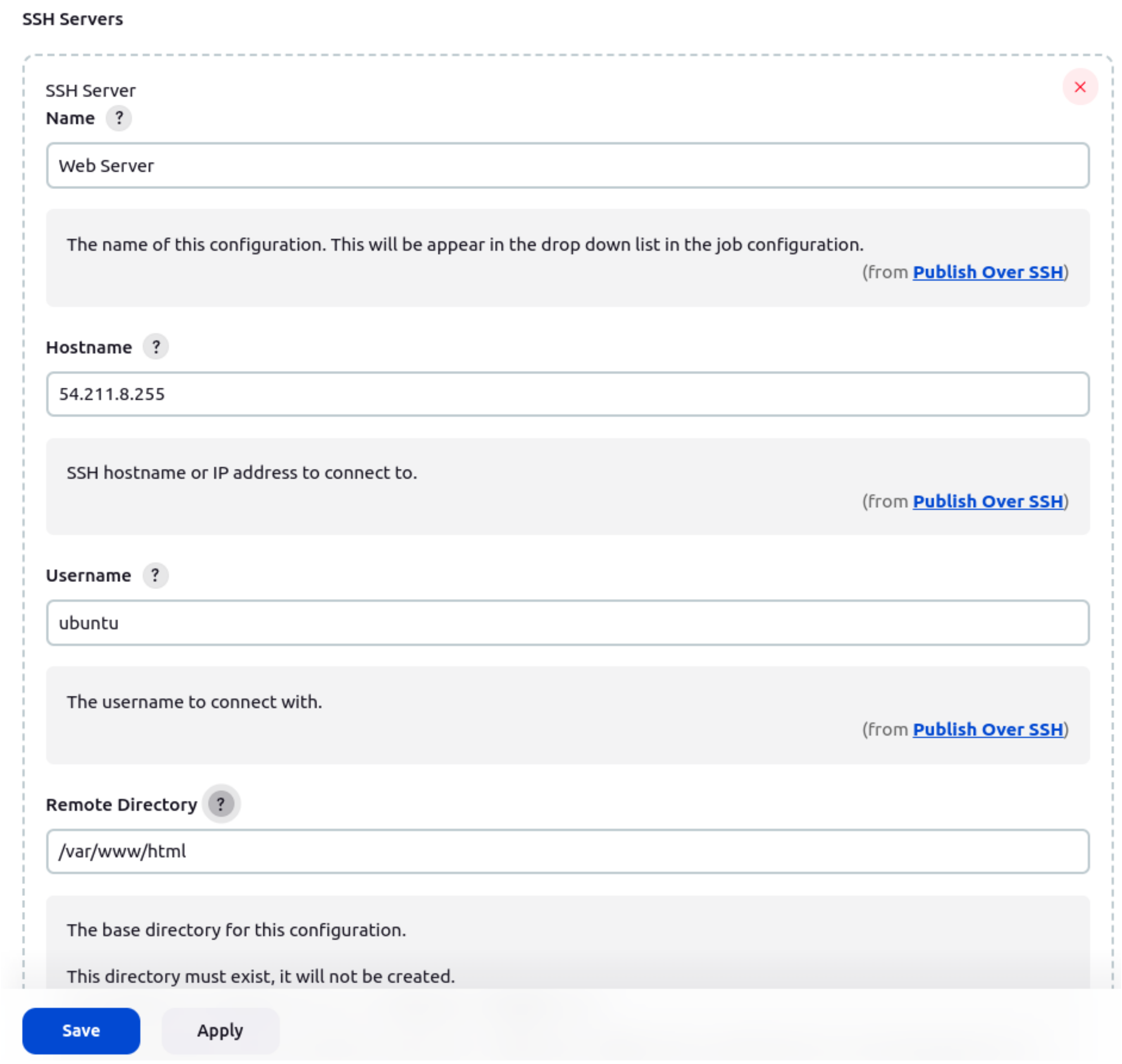
SSH port. Set 22 for default port.

Credentials: ubuntu (EC2)

- Enter **Public IP of EC2** in Hostname
- Port: **22**
- Select the credential created earlier
- Click on **Save**

In, Publish over SSH:

Under SSH Servers:



SSH Servers

SSH Server

Name ?

Web Server

The name of this configuration. This will be appear in the drop down list in the job configuration.

(from **Publish Over SSH**)

Hostname ?

54.211.8.255

SSH hostname or IP address to connect to.

(from **Publish Over SSH**)

Username ?

ubuntu

The username to connect with.

(from **Publish Over SSH**)

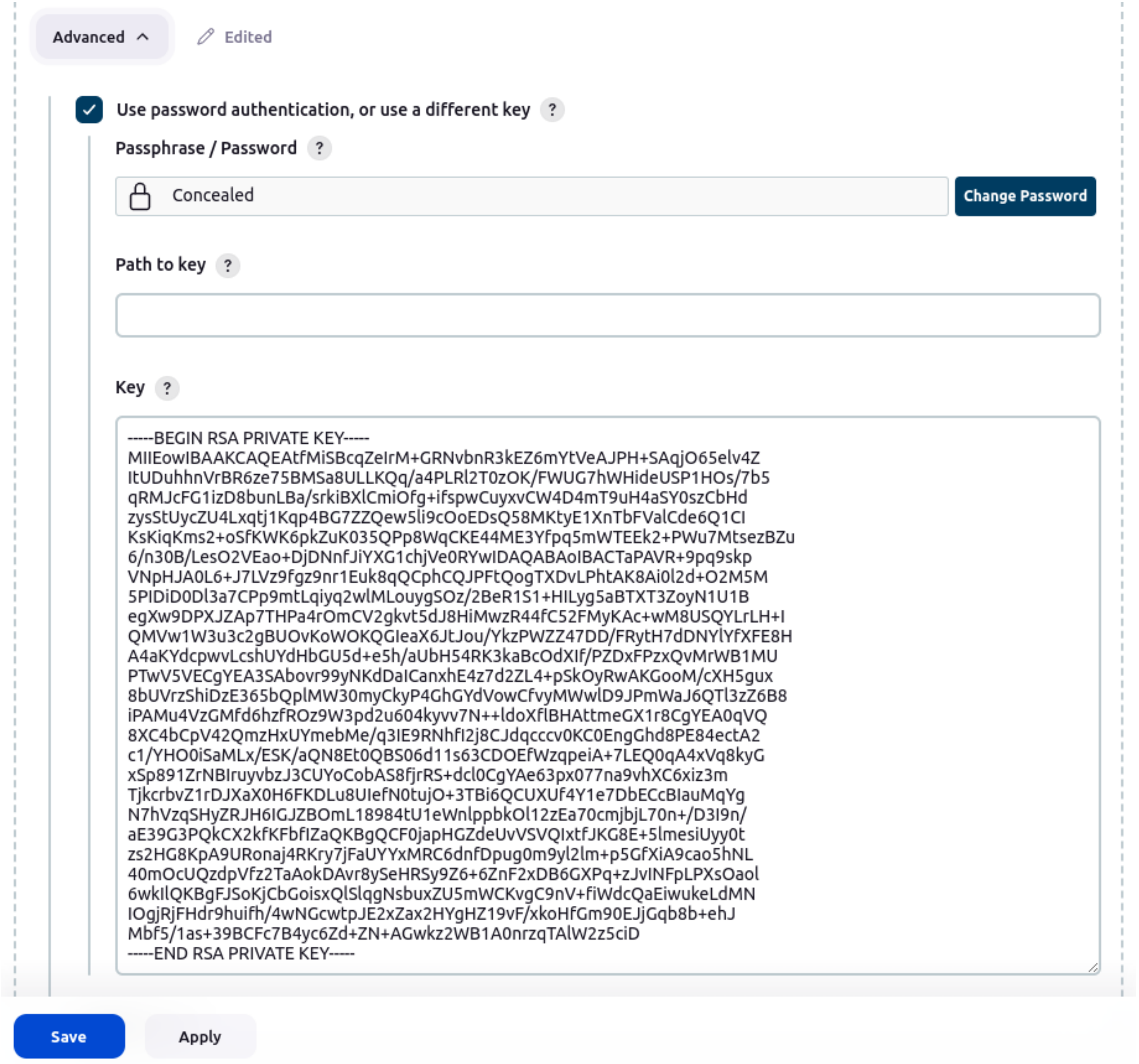
Remote Directory ?

/var/www/html

The base directory for this configuration.

This directory must exist, it will not be created.

- Enter the above configurations.
- Then, Click on **Advanced**.



☑ Use password authentication, or use a different key

Passphrase / Password

Concealed

Path to key

Key

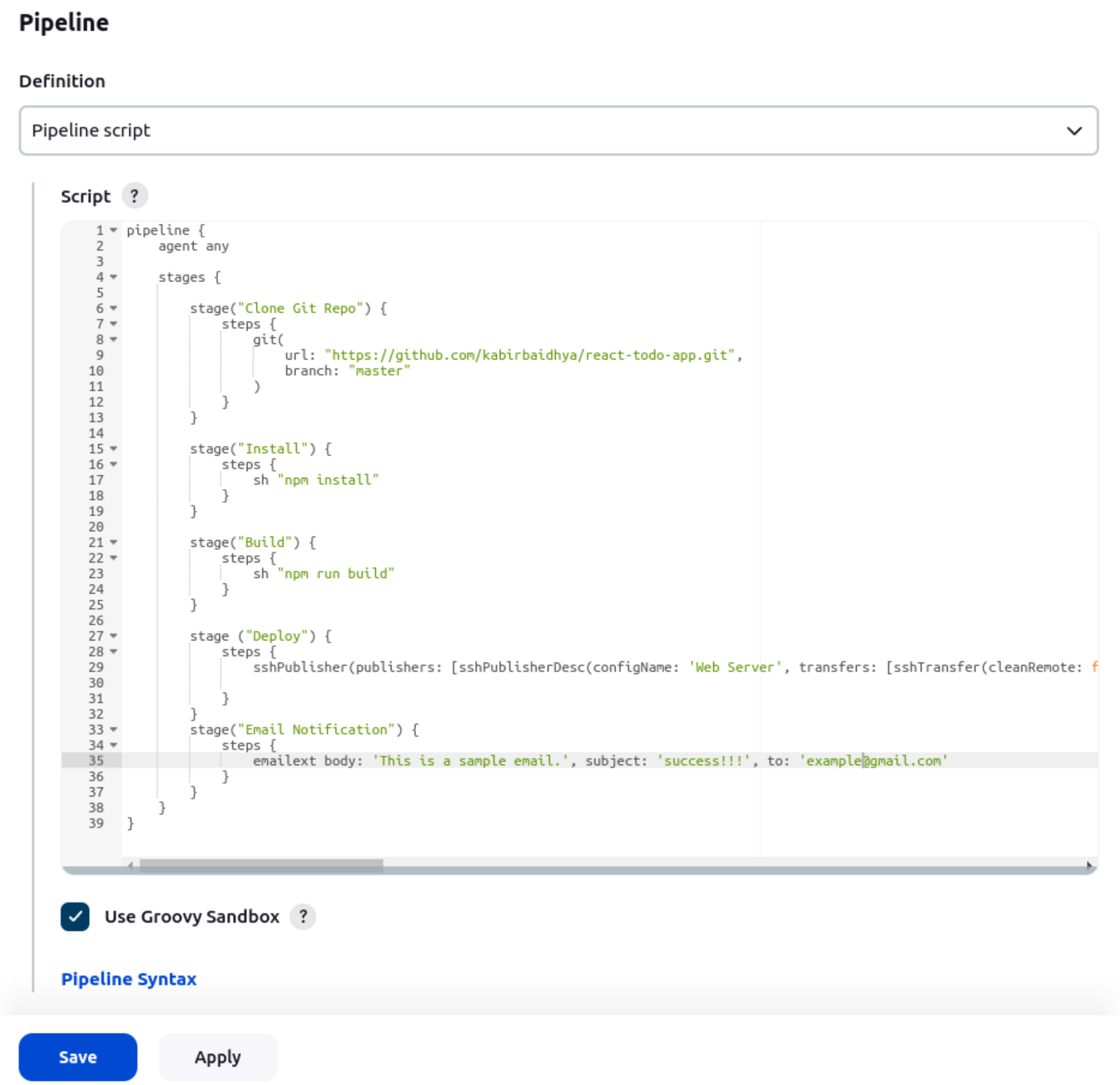
```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAtfMiSBcqZeIrM+GRNvbnR3kEZ6mYtVeAJPH+SAqjO65elv4Z
ItUDuhhnVrBR6ze75BMSa8ULLKQq/a4PLRl2T0zOK/FWUG7hWHideUSP1HOs/7b5
qRMJcFG1izD8bunLBa/srkiBXlCmiOfg+ifspwCuyxvCW4D4mT9uH4aSY0szCbHd
zysStUycZU4Lxqtj1Kqp4BG7ZZQew5li9cOoEDsQ58MKtyE1XnTbFValCde6Q1CI
KsKiqKms2+oSfKWK6pkZuK035QPp8WqCKE44ME3Yfpq5mWTEEk2+PWu7MtsezBZu
6/n30B/LesO2VEao+DjDNnfJiYXG1chjVe0RYwIDAQABAoIBACTaPAVR+9pq9skp
VNpHJA0L6+J7LVz9fgz9nr1Euk8qQCphCQJPFtQogTXDvLPhtAK8Ai0l2d+O2M5M
5PIDiD0Dl3a7CPp9mtLqiyq2wlMLouygSOz/2BeR1S1+HILyg5aBTXT3ZoyN1U1B
egXw9DPXJZAp7THPa4rOmCV2gkvt5dJ8HiMwzR44fC52FMyKAc+wM8USQYLrLH+I
QMVw1W3u3c2gBUOvKoWOKQGIeaX6JtJou/YkzPWZZ47DD/FRytH7dDNYlYfXFE8H
A4aKYdcpwvLcshUYdHbGU5d+e5h/aUbH54RK3kaBcOdXIf/PZDxFPzxQvMrWB1MU
PTwV5VECgYEA3SAbovr99yNKdDaICanxhE4z7d2ZL4+pSkOyRwAKGooM/cXH5gux
8bUVrzShiDzE365bQplMW30myCkyP4GhGYdVowCfvyMWwlD9JPmWaJ6QTl3zZ6B8
iPAMu4VzGMfd6hzfROz9W3pd2u604kyvv7N++ldoXflBHAttmeGX1r8CgYEA0qVQ
8XC4bCpV42QmzHxUYmebMe/q3IE9RNhfI2j8CJdqcccv0KC0EngGhd8PE84ectA2
c1/YHO0iSaMLx/ESK/aQN8Et0QBS06d11s63CDOEfWzqpeiA+7LEQ0qA4xVq8kyG
xSp891ZrNBIruyvbzJ3CUYoCobAS8fjrRS+dcl0CgYAe63px077na9vhXC6xiz3m
TjkcrbvZ1rDJXaX0H6FKDLu8UIefN0tujO+3TBi6QCUXUf4Y1e7DbECcBIauMqYg
N7hVzqSHyZRJH6IGJZBOmL18984tU1eWnlppbkOl12zEa70cmjbjL70n+/D3I9n/
aE39G3PQkCX2kfKFbfIZaQKBgQCF0japHGZdeUvVSVQIxtfJKG8E+5lmesiUyy0t
zs2HG8KpA9URonaj4RKry7jFaUYYxMRC6dnfDpug0m9yl2lm+p5GfXiA9cao5hNL
40mOcUQzdpVfz2TaAokDAvr8ySeHRSy9Z6+6ZnF2xDB6GXPq+zJvINFpLPXsOaol
6wkIlQKBgFJSoKjCbGoisxQlSlqgNsbuxZU5mWCKvgC9nV+fiWdcQaEiwukeLdMN
IOgjRjFHdr9huifh/4wNGcwtpJE2xZax2HYgHZ19vF/xkoHfGm90EJjGqb8b+ehJ
Mbf5/1as+39BCFc7B4yc6Zd+ZN+AGwkz2WB1A0nrzqTAlW2z5ciD
-----END RSA PRIVATE KEY-----
```

Select **“Use password authentication, or use a different key”**
Key: Paste EC2 .pem file contents

Step 4: Create a Pipeline project

In Dashboard, Click on **New Item** ->

1. Enter an **item name**
2. Select **Pipeline project**
3. **Click** on **OK.**



Pipeline

Definition

Pipeline script

Script ?

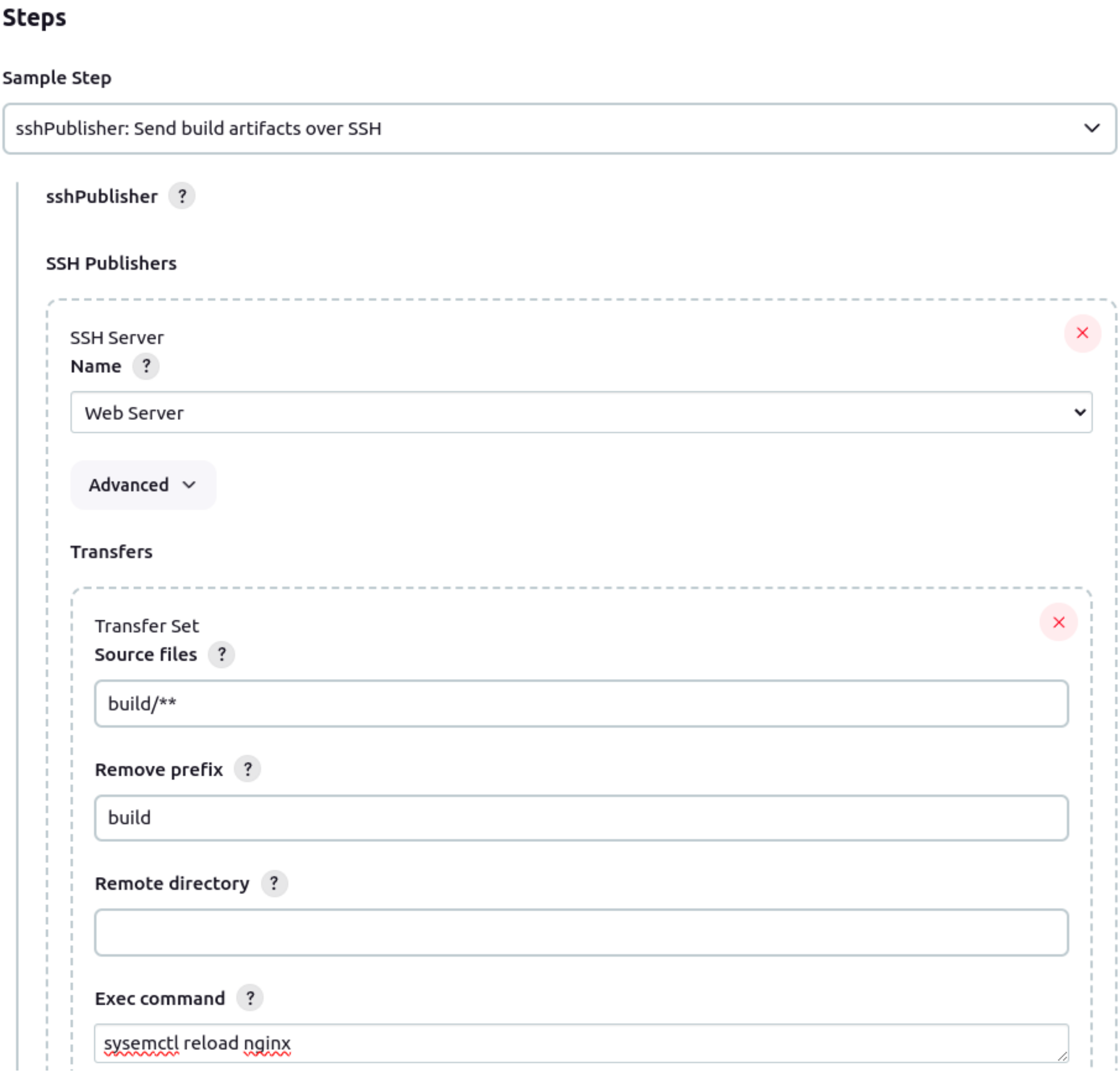
```
pipeline {
    agent any

    stages {

        stage("Clone Git Repo") {
            steps {
                git(
                    url: "https://github.com/kabirbaidhya/react-todo-app.git",
                    branch: "master"
                )
            }
        }

        stage("Install") {
            steps {
                sh "npm install"
            }
        }

        stage("Build") {
            steps {
                sh "npm run build"
            }
        }

        stage ("Deploy") {
            steps {
                sshPublisher(publishers: [sshPublisherDesc(configName: 'Web Server', transfers: [sshTransfer(cleanRemote: f

            }
        }
        stage("Email Notification") {
            steps {
                emailext body: 'This is a sample email.', subject: 'success!!!', to: 'example@gmail.com'
            }
        }
    }
}
```

☑ Use Groovy Sandbox ?

- Refer from the above code.

- In **stage("Deploy"),** generate syntax to send the artifacts over SSH:
- Click on **Pipeline Syntax**,

- Select **sshPublisher: Send build artifacts over SSH** in Sample Step.



Steps

Sample Step: sshPublisher: Send build artifacts over SSH

sshPublisher ?

SSH Publishers

SSH Server

Name ?: Web Server

Transfers

Transfer Set

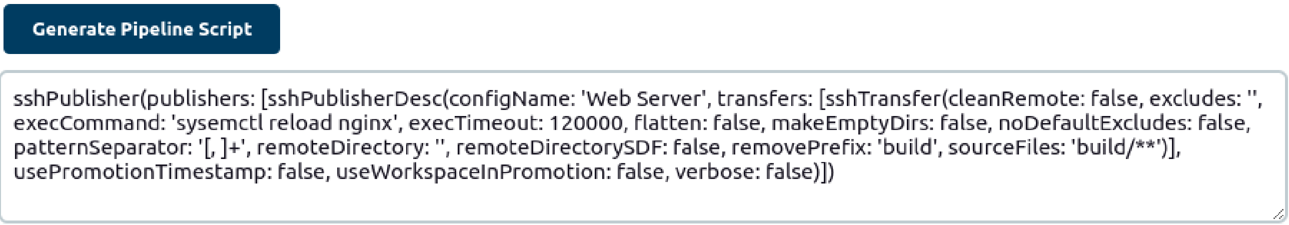
Source files ?: build/**

Remove prefix ?: build

Remote directory ?:

Exec command ?: sysemctl reload nginx

- Refer the above and then.
- Click on **Generate Pipeline Script.**

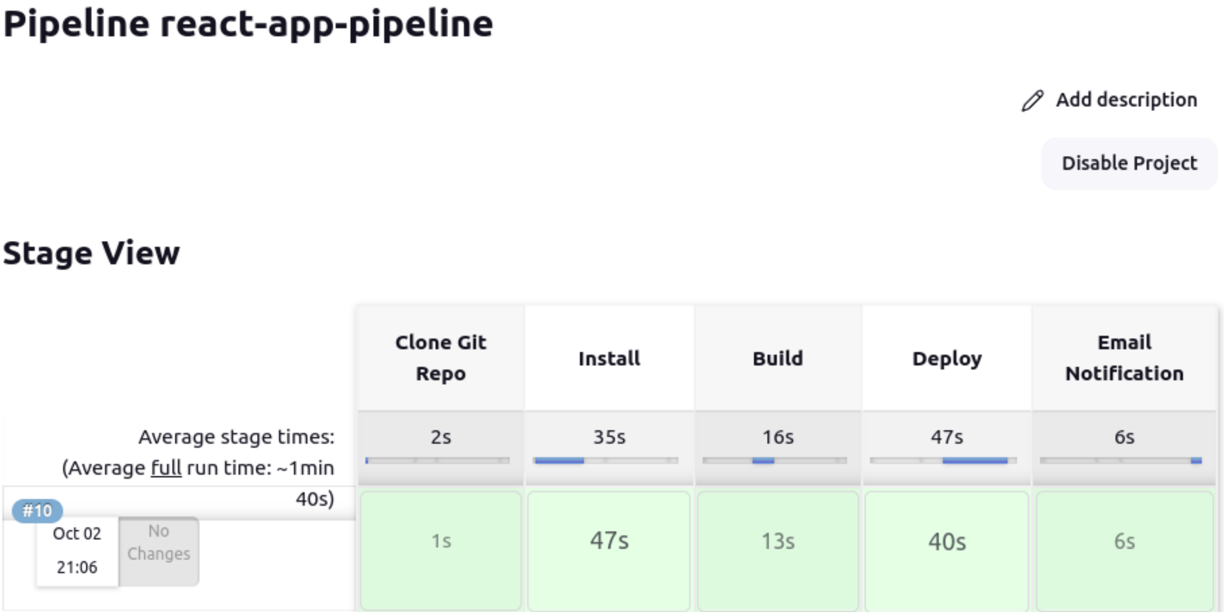


```
sshPublisher(publishers: [sshPublisherDesc(configName: 'Web Server', transfers: [sshTransfer(cleanRemote: false, excludes: '',
execCommand: 'sysemctl reload nginx', execTimeout: 120000, flatten: false, makeEmptyDirs: false, noDefaultExcludes: false,
patternSeparator: '[, ]+', remoteDirectory: '', remoteDirectorySDF: false, removePrefix: 'build', sourceFiles: 'build/**')],
usePromotionTimestamp: false, useWorkspaceInPromotion: false, verbose: false)])
```

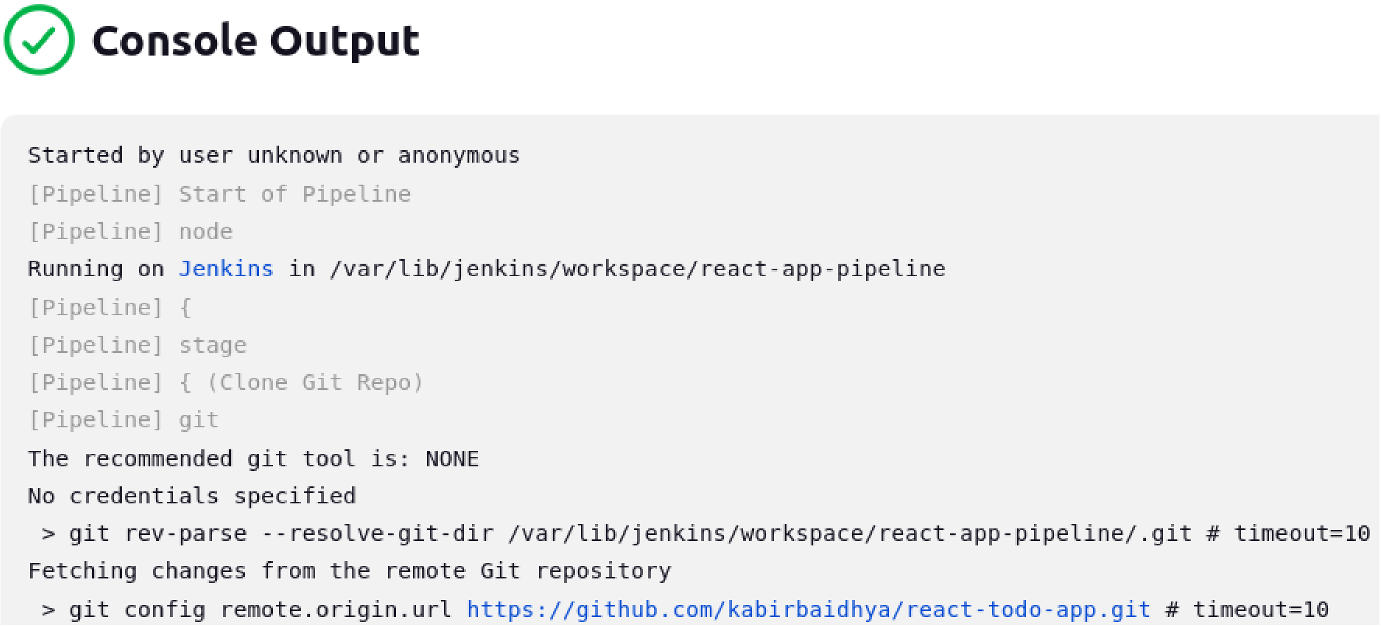
- Copy the generated pipeline script.
- Paste it under **stage("Deploy")**.
- Click on **Apply & Save.**

Step 5: Build the project

- After saving the project, click on " **Build Now** " and check the console output.

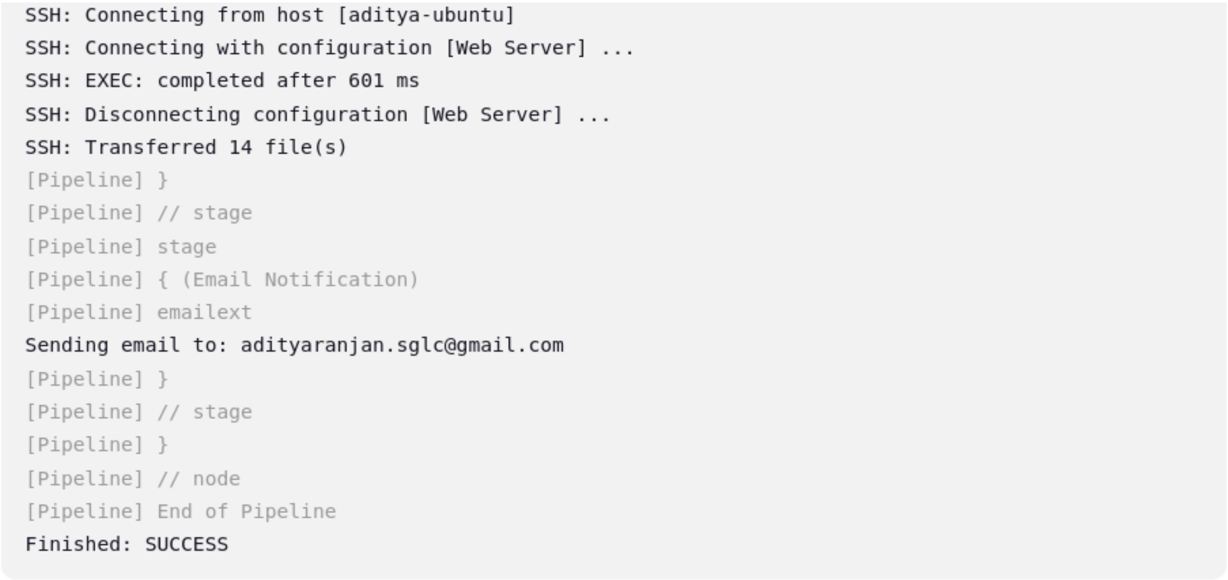


	Clone Git Repo	Install	Build	Deploy	Email Notification
Average stage times: (Average full run time: ~1min 40s)	2s	35s	16s	47s	6s
#10 Oct 02 21:06 No Changes	1s	47s	13s	40s	6s



Console Output

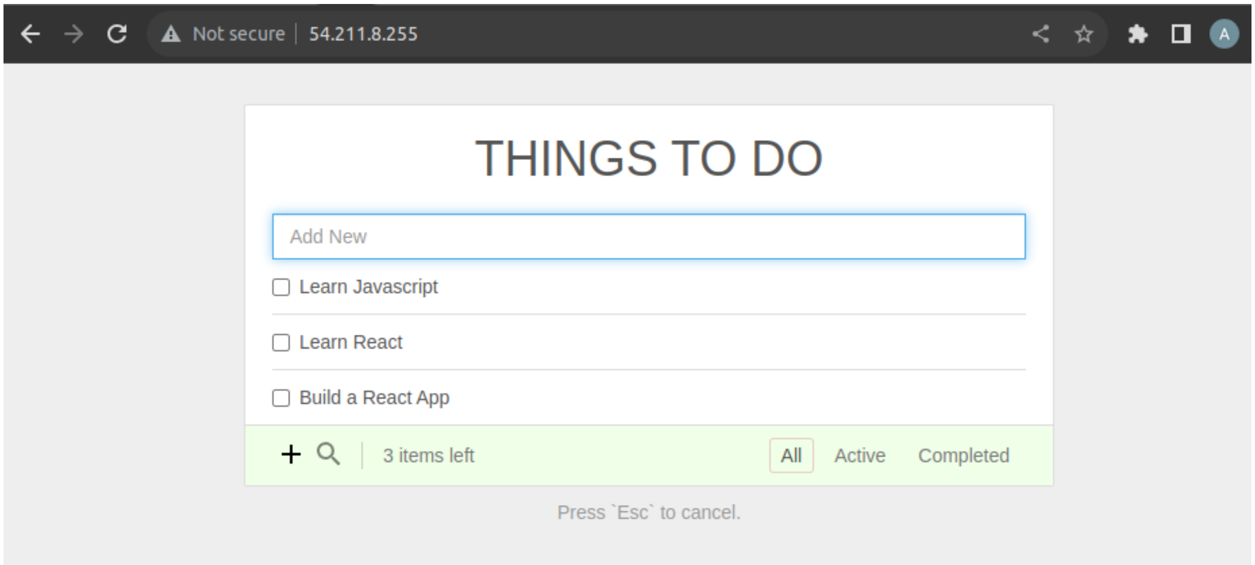
```
Started by user unknown or anonymous
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/react-app-pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Clone Git Repo)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/react-app-pipeline/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/kabirbaidhya/react-todo-app.git # timeout=10
```



```
SSH: Connecting from host [aditya-ubuntu]
SSH: Connecting with configuration [Web Server] ...
SSH: EXEC: completed after 601 ms
SSH: Disconnecting configuration [Web Server] ...
SSH: Transferred 14 file(s)
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Email Notification)
[Pipeline] emailext
Sending email to: adityaranjan.sglc@gmail.com
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

- After successful build, the files will be transferred to the EC2 specified location.

- Copy the public ip-address of the EC2 and paste it in the browser, we should see that the application has been hosted on nginx.



THINGS TO DO

Add New

□ Learn Javascript

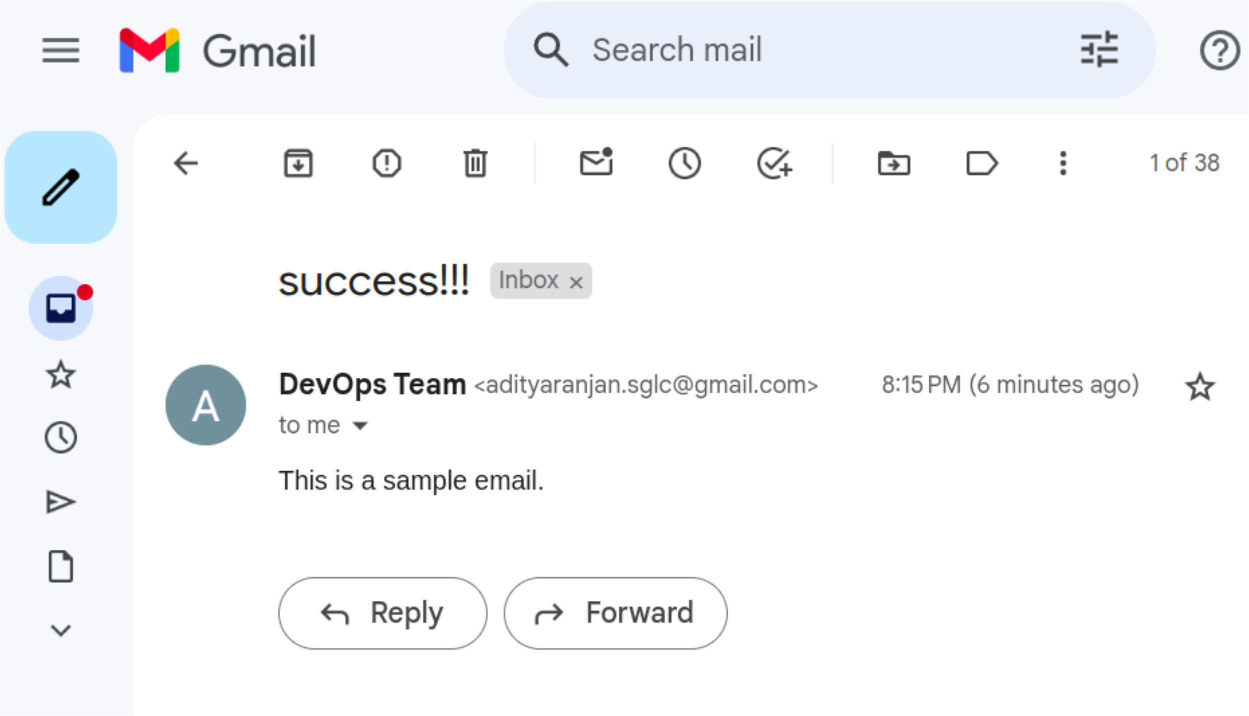
□ Learn React

□ Build a React App

3 items left

Press `Esc` to cancel.

- Verify if the email is received to the specified email address.



success!!!

DevOps Team <adityaranjan.sglc@gmail.com> 8:15 PM (6 minutes ago)

to me

This is a sample email.