

## **Deployment #3**

Welcome to Deployment 3!! Time to step it up for this deployment. You will need to follow the steps below to set up a CI/CD pipeline from start to finish. We will still use Elastic Beanstalk and you will be in charge of adding to your pipeline!!

- 1. Install Jenkins on an EC2 if you haven't already.**
- 2. Activate the Jenkins user on the EC2 by running the commands below.**

```
$sudo passwd jenkins
```

```
$sudo su - jenkins -s /bin/bash
```

- 3. Create a Jenkins user in your AWS account:**

- Navigate to IAM in the AWS console. Next click on the Users option in Access management:

EC2

**Identity and Access Management (IAM)**

Search IAM

Dashboard

Access management

User groups

Users

Roles

Policies

Identity providers

Account settings

Access reports

Access analyzer

Archive rules

Analizers

Settings

Credential report

Organization activity

Service control policies (SCPs)

Related consoles

**There is a better way to connect your existing directory and give your users access across AWS**

AWS IAM Identity Center (successor to AWS Single Sign-On) offers a better way to connect or create a workforce directory, and to manage users' access to multiple AWS accounts, AWS applications, and SAML 2.0-based cloud applications. [Learn more](#)

[Go to IAM Identity Center](#)

## IAM dashboard

### Security recommendations

- ✓ **Root user has MFA**  
Having multi-factor authentication (MFA) for the root user improves security for this account.
- ✓ **Root user has no active access keys**  
Using access keys attached to an IAM user instead of the root user improves security.

### IAM resources

User groups	Users	Roles	Policies	Identity providers
0	5	16	1	0

### What's new

Updates for features in IAM

- Right-size permissions for more roles in your account using IAM Access Analyzer to generate 50 fine-grained IAM policies per day. 9 months ago
- Amazon S3 Object Ownership can now disable access control lists to simplify access management for data in S3. 10 months ago

[View all](#)

### AWS Account

Account ID  
266686430719

Account Alias  
tech-t [Edit](#) [Delete](#)

Sign-in URL for IAM users in this account  
<https://tech-t.signin.aws.amazon.com/console>

### Quick Links

[My security credentials](#)  
Manage your access keys, multi-factor authentication (MFA) and other credentials.

### Tools

[Policy simulator](#)

- Select add Users. Next, the username can be EB-user. Then select Programmatic access and then next:

EC2

**Identity and Access Management (IAM)**

Search IAM

Dashboard

Access management

User groups

Users

Roles

Policies

Identity providers

Account settings

Access reports

Access analyzer

Archive rules

Analizers

Settings

Credential report

Organization activity

Service control policies (SCPs)

**Introducing the new Users list experience**


We've redesigned the Users list experience to make it easier to use. [Let us know what you think.](#)

IAM > Users


**Managing human user access account by account? There's a better way.** [Dismiss](#) [Go to Identity Center](#)

Streamline human access to AWS and cloud apps when you enable Identity Center.


[Learn more](#) [Watch how it works](#)




One-time set up for workforce user access



Centrally manage access to multiple AWS accounts



Provide access centrally to the cloud applications your workforce uses



All with one-click access through a simple web portal

**Users (5)** [Info](#)

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

[Find users by username or access key](#)

[Refresh](#) [Delete](#) [Add users](#)

< 1 > ⚙️

The screenshot shows the AWS IAM 'Add user' console page. At the top, there's a navigation bar with the AWS logo, 'Services' link, a search bar, and a '[Alt+S]' shortcut. Below the navigation bar, the page title 'Add user' is displayed with a progress indicator showing five steps, with the first step '1' being active. The main section is titled 'Set user details' and includes a sub-header 'You can add multiple users at once with the same access type and permissions. [Learn more](#)'. There is a text input field for 'User name\*' containing 'EB-user' and a blue link 'Add another user'. Below this, the 'Select AWS access type' section provides instructions: 'Select how these users will primarily access AWS. If you choose only programmatic access, it does NOT prevent users from accessing the console using an assumed role. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)'. Two options are listed: 'Access key - Programmatic access' (selected with a checked checkbox) and 'Password - AWS Management Console access' (unchecked). The 'Access key' option includes a description: 'Enables an access key ID and secret access key for the AWS API, CLI, SDK, and other development tools.' The 'Password' option includes a description: 'Enables a password that allows users to sign-in to the AWS Management Console.' At the bottom of the form, there is a '\* Required' label, a 'Cancel' button, and a 'Next: Permissions' button.

aws Services Search for services, features, blogs, docs, and more [Alt+S] Global Tech-t

EC2

## Add user

1 2 3 4 5

### Set user details

You can add multiple users at once with the same access type and permissions. [Learn more](#)

User name\* EB-user

[Add another user](#)

### Select AWS access type

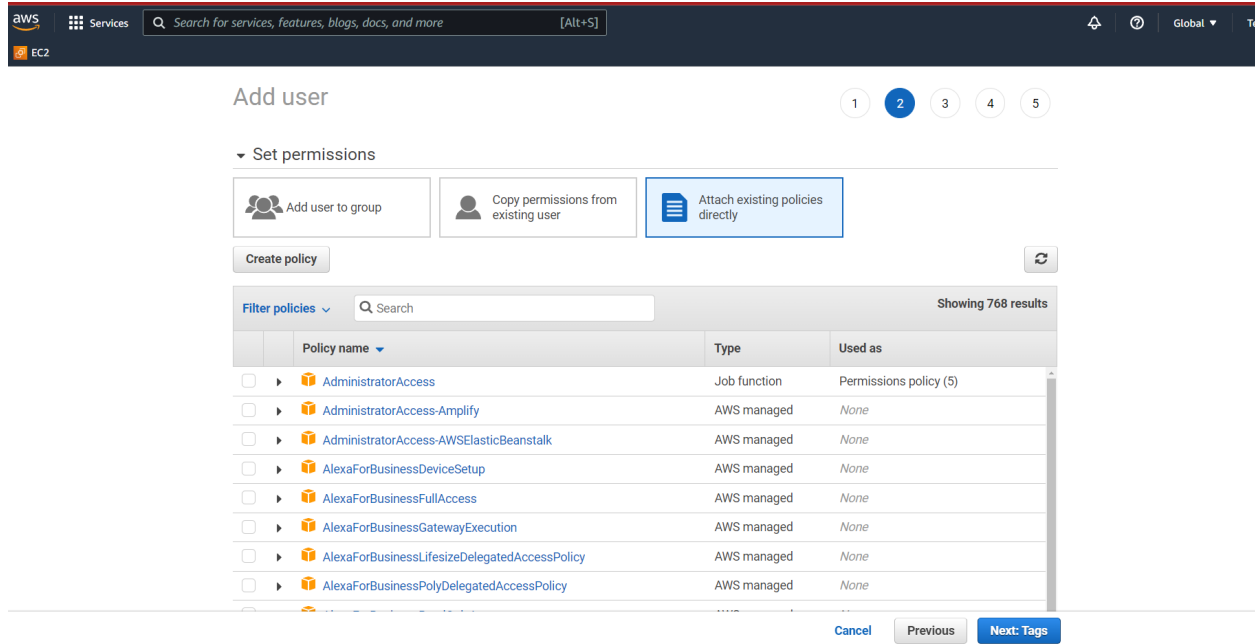
Select how these users will primarily access AWS. If you choose only programmatic access, it does NOT prevent users from accessing the console using an assumed role. Access keys and autogenerated passwords are provided in the last step. [Learn more](#)

Select AWS credential type\*

- ☒ **Access key - Programmatic access**  
Enables an **access key ID** and **secret access key** for the AWS API, CLI, SDK, and other development tools.
- ☐ **Password - AWS Management Console access**  
Enables a **password** that allows users to sign-in to the AWS Management Console.

\* Required Cancel **Next: Permissions**

- Select “Attach existing policies directly” and select administrator access. Now choose Next for this page and the following page:



- Finally, create the user and then copy and save the “access key ID” and the “secret access key”:

aws

Services

Search for services, features, blogs, docs, and more

[Alt+S]

EC2

Add user

12345

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name

EB-user

AWS access type

Programmatic access - with an access key

Permissions boundary

Permissions boundary is not set

Permissions summary

The following policies will be attached to the user shown above.

Type	Name
Managed policy	<a href="#">AdministratorAccess</a>

Tags

No tags were added.

Cancel

Previous

Create user

aws

Services

Search for services, features, blogs, docs, and more

[Alt+S]

EC2

Add user

12345

Success

You successfully created the users shown below. You can view and download user security credentials. You can also email users instructions for signing in to the AWS Management Console. This is the last time these credentials will be available to download. However, you can create new credentials at any time.

Users with AWS Management Console access can sign-in at: <https://tech-t.signin.aws.amazon.com/console>

Download .csv

	User	Access key ID	Secret access key
▶	EB-user	AKIAT4F577X7WOLMS673	***** <a href="#">Show</a>

Close

#### 4. Install AWS CLI on the Jenkins EC2 and configure:

```
$curl \
"https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" \
-o "awscliv2.zip"

$unzip awscliv2.zip

$sudo ./aws/install

$aws --version

$sudo su - jenkins -s /bin/bash

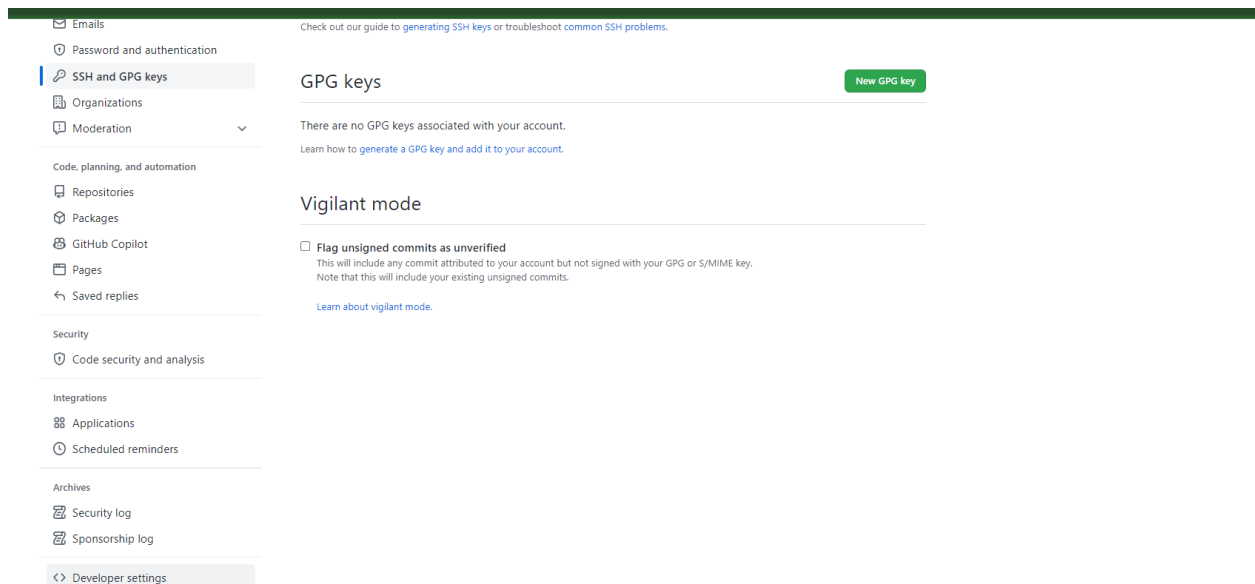
$aws configure
- Set Access Key ID
- Set Secret Access Key
- Set region to: us-east-1
- Set Output format: json
```

#### 5. Install EB CLI in the Jenkins EC2 user:

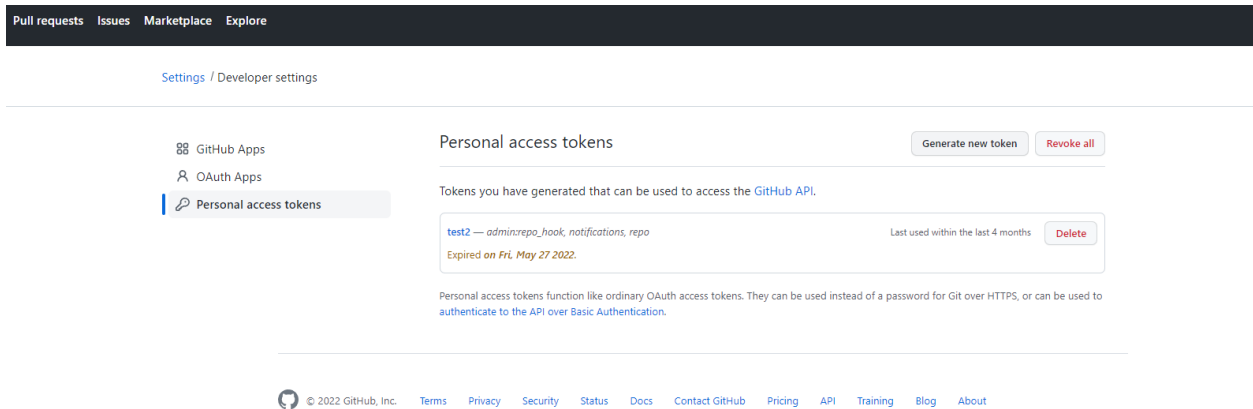
```
$pip install awsecli --upgrade --user
$eb --version
```

## 6. Connect GitHub to Jenkins Server:

- First Fork the Deployment repo:  
[https://github.com/kura-labs-org/kuralabs\\_deployment\\_2.git](https://github.com/kura-labs-org/kuralabs_deployment_2.git)
- Next, create an access token from GitHub:
  - Navigate to your GitHub settings, select developer settings



- Select personal access token and create a new token.



- Select the settings you see below for access token permissions.



#### Expiration \*

30 days

The token will expire on Sun, Sep 25 2022

#### Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes.](#)

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> write:org	Read and write org and team membership, read and write org projects
<input type="checkbox"/> read:org	Read org and team membership, read org projects
<input type="checkbox"/> manage_runners:org	Manage org runners and runner groups
<input type="checkbox"/> admin:public_key	Full control of user public keys
<input type="checkbox"/> write:public_key	Write user public keys
<input type="checkbox"/> read:public_key	Read user public keys
<input checked="" type="checkbox"/> admin:repo_hook	Full control of repository hooks
<input checked="" type="checkbox"/> write:repo_hook	Write repository hooks
<input checked="" type="checkbox"/> read:repo_hook	Read repository hooks
<input type="checkbox"/> admin:org_hook	Full control of organization hooks
<input type="checkbox"/> gist	Create gists
<input type="checkbox"/> notifications	Access notifications

## 7. Create a multibranch build:


- Log back into Jenkins and select “New item”



# Jenkins

Dashboard >

+ New Item

 People

 Build History

 Project Relationship

 Check File Fingerprint

 Manage Jenkins

 My Views

 New View

All

+

S

W

Na



Bui



tes



url.

## Build Queue



No builds in the queue.

## Build Executor Status



1 Idle

2 Idle

Icon:

S

M

L

- Select multibranch pipeline

The screenshot shows the Jenkins 'New Item' page. At the top, there's a navigation bar with the Jenkins logo and a search bar. Below the navigation bar, there's a breadcrumb trail: 'board > All >'. The main content area is titled 'Enter an item name' and contains a text input field with the value 'url-shortener'. Below the input field, there's a note: '» Required field'. The main content area also lists several item types with their descriptions:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- Multibranch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository. (This option is highlighted with a blue border.)
- Organization Folder**: Creates a set of multibranch project subfolders by scanning for repositories.

Below the list of item types, there's a section titled 'If you want to create a new item from other existing, you can use this option:'. It contains a 'Copy from' section with a text input field and an 'OK' button.

- Enter a display name and brief description

**General** Branch Sources Build Configuration Scan Repository Triggers Orphaned Item Strategy Appearance Health metrics Properties Pipeline Libraries

Display Name ?  
Build Flask

Description  
CI/CD pipeline deployment 1  
[Plain text] [Preview](#)

Disable  
☐ (No new builds within this Multibranch Pipeline will be executed until it is re-enabled)

Branch Sources

GitHub Credentials ? ✕

- Add a Branch source by selecting Add source and select GitHub

>

General **Branch Sources** Build Configuration Scan Multibranch Pipeline Triggers Orphaned Item Strategy Appearance Health metrics Properties Pipeline Libraries

☐ Disable (No new builds within this Multibranch Pipeline will be executed until it is re-enabled)

---

**Branch Sources**

[Add source ▾](#)

---

**Build Configuration**

Mode

by Jenkinsfile ▾

Script Path ?

Jenkinsfile

---

**Scan Multibranch Pipeline Triggers**

☐ Periodically if not otherwise run ?

---

**Orphaned Item Strategy**

Jobs for removed SCM heads (i.e. deleted branches) can be removed immediately or kept based on a desired retention strategy. By default, jobs will be removed as soon as Jenkins determines their associated SCM head no longer exists. As an example, it may be useful to configure a different retention strategy to be able to examine build results of a branch after it has been removed.

☐ Abort builds ?

[Save](#) [Apply](#)

- Select the Add button and select GitHub

er >

General **Branch Sources** Build Configuration Scan Multibranch Pipeline Triggers Orphaned Item Strategy Appearance Health metrics Properties Pipeline Libraries

☐ Disable  
(No new builds within this Multibranch Pipeline will be executed until it is re-enabled)

---

**Branch Sources**

Filter

- Git
- GitHub
- Single repository & branch

Wodle

by Jenkinsfile

Script Path ?

Jenkinsfile

---

**Scan Multibranch Pipeline Triggers**

☐ Periodically if not otherwise run ?

---

**Orphaned Item Strategy**

Jobs for removed SCM heads (i.e. deleted branches) can be removed immediately or kept based on a desired retention strategy. By default, jobs will be removed as soon as Jenkins determines their associated SCM head no longer exists. As an example, it may be useful to configure a different retention strategy to be able to examine build results of a branch after it has been removed.

☐ Abort builds ?

://configure#

- Click on Add and then select Jenkins

General **Branch Sources** Build Configuration Scan Multibranch Pipeline Triggers Orphaned Item Strategy Appearance Health metrics Properties Pipeline Libraries

☐ Disable (No new builds within this Multibranch Pipeline will be executed until it is re-enabled)

### Branch Sources

GitHub Credentials ?

- none -

+ Add

url-shortener

Jenkins

Jenkins Credentials Provider

Repository URL

Validate

☐ Repository Scan - Depreciated Visualization

### Behaviors

Discover branches ?

Strategy ?

Exclude branches that are also filed as PRs

**Save** Apply

- Under username enter your GitHub username
- Under password enter your token

The screenshot shows the 'Global Credentials (unrestricted)' configuration page in Jenkins. The 'Kind' is set to 'Username with password'. The 'Scope' is 'Global (Jenkins, nodes, items, all child items, etc)'. The 'Username' is 'kura-labs01'. The 'Password' field is masked with dots. The 'ID' field is empty. The 'Description' field contains 'GitHub Token'. At the bottom, there are 'Add' and 'Cancel' buttons.

Global credentials (unrestricted)

Kind

Username with password

Scope ?

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

Username ?

kura-labs01

☐ Treat username as secret ?

Password ?

.....

ID ?

Description ?

GitHub Token

Add Cancel

- **(Optional)** under ID and Description enter GitHub repo
- Enter your URL to the repository and you can validate by selecting validate.



ener

General **Branch Sources** Build Configuration Scan Multibranch Pipeline Triggers Orphaned Item Strategy Appearance Health metrics Properties Pipeline Libraries

☐ Disable  
(No new builds within this Multibranch Pipeline will be executed until it is re-enabled)

### Branch Sources

**GitHub** ?

Credentials ?

kura-labs01/\*\*\*\*\* (GitHub Token) ▼

+ Add

☒ Repository HTTPS URL

Repository HTTPS URL ?

https://github.com/kura-labs-org/kuralabs\_deployment\_1.git

Credentials ok. Connected to https://github.com/kura-labs-org/kuralabs\_deployment\_1. Validate

☐ Repository Scan - Depreciated Visualization

### Behaviors

**Discover branches** ?

Strategy ?

Exclude branches that are also filed as PRs ▼

Save Apply

- Make sure this says “Jenkinsfile”

>

General **Branch Sources** Build Configuration Scan Multibranch Pipeline Triggers Orphaned Item Strategy Appearance Health metrics Properties Pipeline Libraries

☐ Disable (No new builds within this Multibranch Pipeline will be executed until it is re-enabled)

---

**Branch Sources**

[Add source](#)

---

**Build Configuration**

Mode

by Jenkinsfile

Script Path ?

Jenkinsfile

---

**Scan Multibranch Pipeline Triggers**

☐ Periodically if not otherwise run ?

---

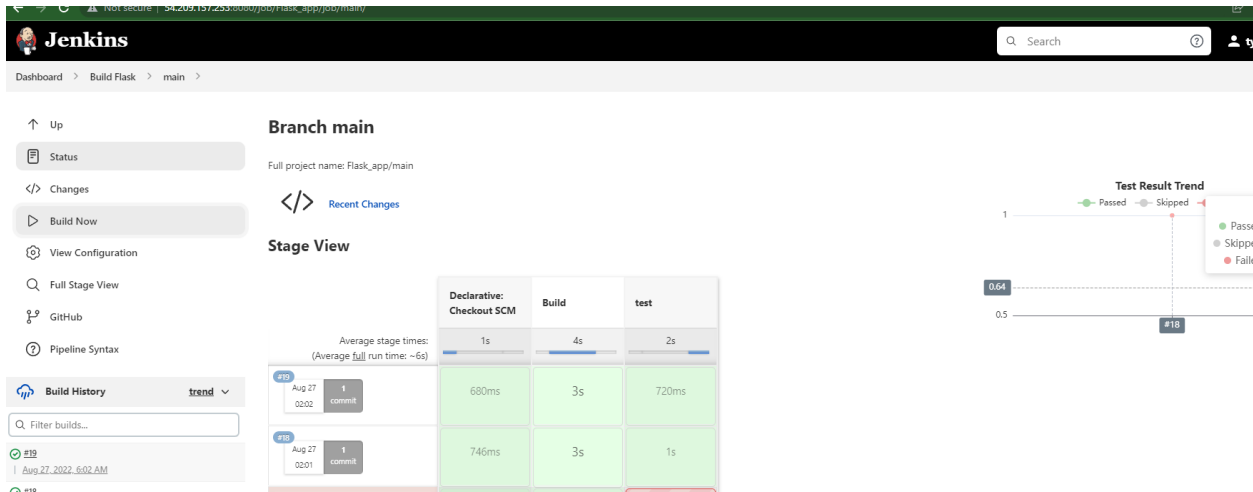
**Orphaned Item Strategy**

Jobs for removed SCM heads (i.e. deleted branches) can be removed immediately or kept based on a desired retention strategy. By default, jobs will be removed as soon as Jenkins determines their associated SCM head no longer exists. As an example, it may be useful to configure a different retention strategy to be able to examine build results of a branch after it has been removed.

☐ Abort builds ?

[Save](#) [Apply](#)

- Select Apply and then Save
- You should see a build happening. If you don't, select Scan Repository.



## 8. Now deploy the application from Elastic Beanstalk CLI:

```
$sudo su - jenkins -s /bin/bash
```

```
$cd /var/workspace/{{The name of your project}}/
```

```
$eb init
```

- Select: us-east-1
- Press enter
- Select: Python
- Select: (The latest version of python available)
- Select: N (for CodeCommit)

```
$eb create
```

- Take the default for the next 3 questions by hitting enter (remember the environment name)
- Spot Fleet: No
- Wait for the environment to be made!! And then check it

## 9. Now add a deployment stage to the pipeline in your Jenkinsfile:

```
pipeline {
  agent any
  stages {
    stage ('Build') {
      steps {
        sh '''#!/bin/bash
        python3 -m venv test3
        source test3/bin/activate
        pip install pip --upgrade
        pip install -r requirements.txt
        export FLASK_APP=application
        flask run &
        '''
      }
    }
    stage ('test') {
      steps {
        sh '''#!/bin/bash
        source test3/bin/activate
        py.test --verbose --junit-xml test-reports/results.xml
        '''
      }
    }
  }
}
```

```

    }

    post{
        always {
            junit 'test-reports/results.xml'
        }
    }
}
stage ('Deploy') {
    steps {
        sh '/var/lib/jenkins/.local/bin/eb deploy {{Your environment
name}}'
    }
}
}
}

```

## Now add to the pipeline:

- Add a Webhook to the pipeline
- Create email notification to the pipeline

Once you have done that, apply your changes by building the pipeline again

## Create documentation:

Document your process for adding a webhook and email notification. Also, include a summary of what you configured in each part of your setup. Lastly, include any issues you ran into and what you did to fix them.

**Note:** Submit your work by uploading your work to a repo or the forked repo. Then submit the link to the repo via LMS.