## Jenkins Setup on EC2

* Create an EC2 instance (select Ubuntu for AMI and open ports 22, 80, and 8080):

Graphical user interface, text, application

Description automatically generated

Table

Description automatically generated

* SSH into the EC2 and run [**setup\_jenkins.sh**](https://github.com/cadenhong/kl_wk12_deployment2_organized/blob/main/setup_jenkins.sh) to install Jenkins, create a jenkins user, and activate the jenkins user by switching to it in Bash shell:

Graphical user interface, text

Description automatically generated

* Go to http://<ec2-public-ip>:8080 to set up Jenkins admin role – retrieve password by:
  + sudo cat /var/lib/Jenkins/secrets/initialAdminPassword
* Install suggested plugins
* Reset admin password then save

## Creating a Jenkins user on my AWS Account

* AWS > IAM > Access Management > Users > Add users
* Enter **EB-user** as username and click **Access key - Programmatic access** for access type:Graphical user interface, text, application, email

  Description automatically generated
* Select **AdministratorAccess** under “Attach existing policies directly”:Graphical user interface, application

  Description automatically generated
* Review user info then create user:Graphical user interface, text, application, email

  Description automatically generated
* Download generated credentials CSV (contains Access key ID and Secret access key):Graphical user interface, text, application, Teams

  Description automatically generated

## AWS CLI Setup on EC2

* As the Ubuntu user (**NOT** Jenkins user) on the EC2, run [**setup\_awscli.sh**](https://github.com/cadenhong/kl_wk12_deployment2_organized/blob/main/setup_awscli.sh)**:** Text

  Description automatically generated

Text

Description automatically generated

* Configure AWS using credentials generated during the IAM user creation:Graphical user interface, text

  Description automatically generated
* cd .aws – read the config and credentials files to confirm:Text

  Description automatically generated

## Installing EB CLI on jenkins EC2 User

* sudo su - jenkins -s /bin/bash to switch to the jenkins user’s bash terminal
* Run [**setup\_ebcli.sh**](https://github.com/cadenhong/kl_wk12_deployment2_organized/blob/main/setup_ebcli.sh) to set up the EB CLI on jenkins user – it will run the following:
  + pip install awsebcli --upgrade --userText

    Description automatically generated
  + Installation will complete, but there may be warning messages about the PATH variable not including the bin folder the package is installed in: Text

    Description automatically generated
  + To resolve the warning message, run:

export PATH=”/var/lib/jenkins/.local/bin:$PATH”Text

Description automatically generated with low confidence

## Generate Access Token to Connect GitHub to Jenkins Server

* First, fork the [Deployment 2 repo](https://github.com/kura-labs-org/kuralabs_deployment_2)
* Then, navigate to:
  + Settings > Developer settings > Personal access tokens > Generate new token
* Under “Select scopes” - select **repo** and **admin:repo\_hook:**Graphical user interface, text, application

  Description automatically generated
* Generate and copy the personal access token:Graphical user interface, text, application

  Description automatically generated

## Connect GitHub Repo to Jenkins via Multibranch Build

* Go to http://<ec2-public-ip>:8080 and login as admin user
* Dashboard > New Item > Multibranch pipeline
* Under Branch sources, add GitHub credentials:Graphical user interface, application

  Description automatically generated
* Enter GitHub username and generated access token as the password:Graphical user interface, text, application, email

  Description automatically generated
* Select the entered credentials and enter the [forked repo URL](https://github.com/cadenhong/kl_wk12_deployment2), then validate connection:Graphical user interface, text, application, Teams

  Description automatically generated
* After clicking Apply then Save, there will be a build happening:Graphical user interface, application

  Description automatically generated
* **\*\*\* The test stage kept failing, so I had to go into Jenkinsfile and change line 20 to activate the pytest module for it to recognize the test file\*\*\*:**

Graphical user interface, text

Description automatically generated with medium confidenceTimeline

Description automatically generated with medium confidence

*BEFORE AFTER*

## Deploy the url-shortener Application Using Elastic Beanstalk CLI (as a Jenkins user)

* sudo su - jenkins -s /bin/bash to switch to the jenkins user’s bash terminal
* cd workspace/<project-name>
* eb init
* Continue with configuration as described below:

Graphical user interface, text, application

Description automatically generated

**Text

Description automatically generated**

**Text

Description automatically generated**

* eb create – select default for first 3 questions and no for Spot Fleet:Text

  Description automatically generated
* Elastic Beanstalk creating the environment and application:A screenshot of a computer

  Description automatically generated with medium confidence
* **Application deployment successful:**Graphical user interface, text, application

  Description automatically generated
* You can check Elastic Beanstalk > Environments to check the status of the running environment and application**:Graphical user interface, text, application

  Description automatically generated**

## Adding “Deploy” Stage to Pipeline

* Edit the forked repo’s [Jenkinsfile](https://github.com/cadenhong/kl_wk12_deployment2/blob/main/Jenkinsfile) to include a Deploy stage:Graphical user interface, text, application, email

  Description automatically generated
* Confirm that build was successful on Jenkins pipeline:**Table, calendar

  Description automatically generated**

# Modifying/Adding to the Pipeline

## 1. Adding Webhook to Automate Deployment

* Navigate to GitHub repo > Settings > Webhooks
  + For Payload URL, enter http://<ec2-public-ip>:8080/github-webhook/
  + Content type should be application/json

Graphical user interface, text, application, email

Description automatically generated

* Scan Repository on Jenkins to activate webhook: Graphical user interface, text, application

  Description automatically generated with medium confidence
* To test the automated deployment pipeline, make changes to the application – in this case, I changed the text displayed on the front-end:Graphical user interface, text, application, email

  Description automatically generated
* Once code change is added, committed, and pushed to the repository, Jenkins will automatically start going through the different stages to build again:Graphical user interface, application

  Description automatically generated

Graphical user interface, application

Description automatically generated

* Once the Deploy stage is completed, refresh the app URL and see the changes made:

Graphical user interface, text, application

Description automatically generated

## 2A. Adding Slack Notification on Jenkins

* On [Slack app directory](https://kura-labs.slack.com/apps/A0F7VRFKN-jenkins-ci), search Jenkins CI and add to Slack (to the specific Channel you want):Graphical user interface, text, application, email

  Description automatically generated
* Follow instructions (take note of the **Integration Token Credential ID** provided in Step 3:Graphical user interface, text, application, Teams

  Description automatically generatedGraphical user interface, text, application

  Description automatically generated
  + Go to Jenkins > Dashboard > Manage Jenkins > Plugin Manager > Available > Search “Slack Notification”
  + Install the plugin – it should show up under Installed after completed:Graphical user interface, application

    Description automatically generated
  + Dashboard > Manage Jenkins > Configure System > Slack > Add workspace name > Add credential as “Secret Text”
  + Enter the provided **Integration Token Credential ID**:**Graphical user interface, text, application, email

    Description automatically generated**
  + You will receive a Slack notification once it is all set:A picture containing logo

    Description automatically generated

## 2B. Slack Notification Based on the Status of Build Pipeline

* Add slackSend function inside JenkinsfileGraphical user interface, text, application, email

  Description automatically generated
* Slack notifications will be sent each time Jenkins scans the repository: Application

  Description automatically generated with low confidence