Hands-on Jenkins Day-1 : Installing Jenkins

Purpose of the this hands-on is to learn how to install Jenkins Server and build simple jobs.

Learning Steps

At the end of the this hands-on training, students will be able to;

- install and configure Jenkins Server on Amazon Linux 2 EC2 instance using `yum` repo.
- install plugins
- create view
- create simple Free Style jobs
- create simple Pipeline jobs
- create simple Pipeline with jenkinsfile

Outline

- Part 1 - Installing Jenkins Server on Amazon Linux 2 with `yum` Repo
- Part 2 - learning Jenkins Dashboard
- Part 3 - Installing Plugins
- Part 4 - Creating a view
- Part 5 - Creating First Jenkins Job
- Part 6 - Creating a Simple Pipeline with Jenkins
- Part 7 - Creating a Jenkins Pipeline with Jenkinsfile

Part 1 - Installing Jenkins Server on Amazon Linux 2 with `yum` Repo

- Launch an AWS EC2 instance of Amazon Linux 2 AMI with security group allowing SSH(22) and TCP 8080 ports.
- Connect to the instance with SSH.

```
ssh -i "key.pem" ec2-user@<IP.ADRESS.>
```

- Update the installed packages and package cache on your instance.

```
sudo yum update -y
```

- Install `Java 11 openjdk` Java Development Kit.

```
sudo amazon-linux-extras install java-openjdk11 -y
```

- Check the java version.

```
java -version
```

- Add Jenkins repo to the `yum` repository.

```
sudo wget -O /etc/yum.repos.d/jenkins.repo
https://pkg.jenkins.io/redhat/jenkins.repo
```

yes

- Import a key file from Jenkins-CI to enable installation from the package.

```
sudo rpm --import https://pkg.jenkins.io/redhat/jenkins.io.key
```

- Enable the EPEL repository for Amazon EC2 instance.

```
sudo amazon-linux-extras install epel -y
```

- Install Jenkins.

```
sudo yum install jenkins -y
```

- Start Jenkins service.

```
sudo systemctl start jenkins
```

- Enable Jenkins service so that Jenkins service can restart automatically after reboots.

```
sudo systemctl enable jenkins
```

- Check if the Jenkins service is up and running.

```
sudo systemctl status jenkins
```

- Get the initial administrative password.

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

- Open your browser, get your ec2 instance Public IPv4 DNS and paste it at address bar with 8080. **"http://[ec2-public-dns-name]:8080"**
- Enter the temporary password to unlock the Jenkins.
- Install suggested plugins.
- Create first **admin user (admin:Jenkins1234)**.
- Check the URL, then save and finish the installation.

Part 2 - Learning Jenkins Dashboard

- Explain `Jenkins` dashboard.
- Explain `Items` options.
- Explain `Manage Jenkins` dashboard.
- Explain other Jenkins terminology.

Part 3 - Installing Plugins

- Follow `Manage Jenkins` -> `Manage Plugins` path and install the plugins (install without restart):
    - `AnsiColor`
    - `Copy Artifact`
    - `Deploy to container`

Part 4 - Creating a view

- Click `My Views` on the left menu items or click `+` on the jobs tabs.
- Select `New View`
- Give a name like `my view`
- Select `List View` option
- Select `OK` option

Part 5 - Creating First Jenkins Job

- We will create a job in Jenkins which picks up a simple "Hello World" bash script and runs it. The freestyle build job is a highly flexible and easy-to-use option. To create a Jenkins freestyle job;
    - Open your Jenkins dashboard and click on `New Item` to create a new job item.
    - Enter `my-first-job` then select `free style project` and click `OK`.
    - Enter `My first jenkins job` in the description field.
    - Explain `Source Code Management`, `Build Triggers`, `Build Environment`, `Build`, `Post-build Actions` tabs.

```
1. Source Code Management Tab : optional SCM, such as CVS or Subversion
where your source code resides.
2. Build Triggers : Optional triggers to control when Jenkins will
perform builds.
3. Build Environment: Some sort of build script that performs the build
(ant, maven, shell script, batch file, etc.) where the real work
happens
4. Build : Optional steps to collect information out of the build,
such as archiving the artifacts and/or recording javadoc and test
results.
5. Post-build Actions : Optional steps to notify other people/systems
with the build result, such as sending e-mails, IMs, updating issue
tracker, etc.
```

    - Go to `Build` section and choose "Execute Shell Command" step from `Add build step` dropdown menu.
    - Write down just `echo "Hello World, This is my first job"` to execute shell command, in text area shown.
    - Click `apply` and `save` buttons.
- On the Project job page, click to `Build now`.
- Show and explain the result of this build action under the `Build History`
    - Click to the build number to reach build page.
    - Show the console results from `Console Output`.

Part 6 - Creating a Simple Pipeline with Jenkins

- Go to the Jenkins dashboard and click on `New Item` to create a pipeline.
- Enter `simple-pipeline` then select `Pipeline` and click `OK`.
- Enter `My first simple pipeline` in the description field.
- Go to the `Pipeline` section, enter following script, then click `apply` and `save`.

```
pipeline {
    agent any
    stages {
        stage('build') {
            steps {
                echo "Welcome to Jenkins Enviroment"
                sh 'echo second step'
                sh 'echo another step'
                sh '''
                echo 'Multiline'
                echo 'Example'
                '''
                echo 'not using shell'
            }
        }
    }
}
```

- Go to the project page and click `Build Now`.
- Explain the built results.
- Explain the pipeline script.

Part 7 - Creating a Jenkins Pipeline with Jenkinsfile

- Create a public project repository `jenkinsfile-pipeline-project` on your GitHub account.
- Clone the `jenkinsfile-pipeline-project` repository on local computer.

```
git clone <your-repo-url>
```

- Create a `Jenkinsfile` within your local `jenkinsfile-pipeline-project` repo and save following pipeline script. Consider that filename 'Jenkinsfile' is case sensitive.

```
pipeline {
    agent any
    stages {
        stage('build') {
            steps {
                echo "Welcome to Jenkins Enviroment"
                sh 'echo using shell within Jenkinsfile'
                echo 'not using shell in the Jenkinsfile'
            }
        }
    }
}
```

- Commit and push the local changes to update the remote repo on GitHub.

```
git add .
git commit -m 'added Jenkinsfile'
git push
```

- Go to the Jenkins dashboard and click on `New Item` to create a pipeline.
- Enter `pipeline-from-jenkinsfile` then select `Pipeline` and click `OK`.
- Enter `Simple pipeline configured with Jenkinsfile` in the description field.
- Go to the `Pipeline` section, and select `Pipeline script from SCM` in the `Definition` field.
- Select `Git` in the `SCM` field.
- Enter URL of the project repository, and let others be default.

```
https://github.com/<your_github_account_name>/jenkinsfile-pipeline-project/
```

- Click `apply` and `save`. Note that the script `Jenkinsfile` should be placed under root folder of repo.
- Go to the Jenkins project page and click `Build Now`.
- Explain the role of `Jenkinsfile` and the built results.