

School: CLOUDBOOSTA	Course: DevOps
Name: AKOSA SAMUEL ONYEJEKWE	Assignment-2_ Jenkins-2
Student No: CBALC0168	Issue Date: May 14, 2026

Tasks:

1. Create Jenkins Continuous Integration pipeline job
2. Configure Jenkins pipeline project settings
3. Configure GitHub repository integration
4. Configure Jenkins Git credentials
5. Configure Jenkins pipeline script from SCM
6. Configure Jenkins branch specification
7. Configure Jenkinsfile pipeline path
8. Trigger Jenkins CI pipeline build
9. Verify Jenkins CI pipeline execution
10. Modify GitHub repository files
11. Trigger GitHub webhook automation
12. Verify automated CI build execution
13. Create Jenkins Continuous Deployment pipeline job
14. Configure deployment pipeline script
15. Configure deployment environment stages
16. Configure AWS deployment integration
17. Execute deployment pipeline workflow
18. Verify deployment approval workflow
19. Validate successful CI/CD pipeline execution
20. Verify Jenkins build history and deployment status

Goal:

To configure Jenkins Continuous Integration and Continuous Deployment (CI/CD) pipeline jobs integrated with GitHub repositories, automate source code build and deployment workflows, configure AWS deployment integration, and validate automated DevOps delivery pipelines within a cloud-native Jenkins environment.

Result:

- Jenkins Continuous Integration pipeline created successfully
- Jenkins pipeline project settings configured successfully
- GitHub repository integration configured successfully
- Jenkins Git credentials configured successfully
- Jenkins pipeline script from SCM configured successfully
- Jenkins branch specification configured successfully
- Jenkinsfile path configured successfully
- Jenkins CI pipeline executed successfully
- GitHub webhook automation triggered successfully
- Automated CI build execution validated successfully
- Jenkins Continuous Deployment pipeline created successfully
- Deployment pipeline stages configured successfully
- AWS deployment integration configured successfully
- Deployment approval workflow executed successfully
- Jenkins CI/CD pipeline validated successfully
- Jenkins build history and deployment workflow verified successfully

Action:

Step 1: Created Jenkins Continuous Integration Pipeline Job

- Opened: Jenkins Dashboard → New Item
- Configured item name: continuous-integration-job
- Selected: Pipeline
- Clicked: OK
- Verified Jenkins pipeline job created successfully

Step 2: Verified GitHub Repository Configuration

- Opened GitHub repository: ci-demo
- Verified repository forked successfully from: cloudboostauk/ci-demo
- Confirmed:
 - Main branch available

- Repository accessible successfully

Step 3: Configured Jenkins Pipeline Project Settings

- Opened: Jenkins → continuous-integration-job → Configure
- Configured:
 - GitHub project option enabled
- Configured project URL: GitHub repository URL
- Enabled: GitHub hook trigger for GITScm polling
- Verified webhook trigger configured successfully

Step 4: Configured Pipeline Script from SCM

- Opened: Pipeline section
- Configured:
 - Definition: Pipeline script from SCM
- Selected: Git
- Configured repository URL: GitHub repository clone URL
- Configured Git credentials successfully

Step 5: Configured Branch Specification

- Configured: Branches to build
- Configured branch specifier: */main
- Verified Jenkins branch configuration completed successfully

Step 6: Configured Jenkinsfile Script Path

- Configured: Script path: Jenkinsfile
- Enabled: Lightweight checkout

- Saved Jenkins pipeline configuration successfully

Step 7: Triggered Jenkins CI Pipeline Build

- Opened: continuous-integration-job
- Selected: Build Now
- Jenkins pipeline stages executed:
 - Checkout SCM
 - Clone repository
 - Execute pipeline stages
 - Cleanup workspace
- Verified Git repository cloned successfully

Step 8: Verified Jenkins CI Pipeline Execution

- Opened: Console Output
- Verified:
 - Jenkinsfile loaded successfully
 - Git credentials authenticated successfully
 - Repository checkout completed successfully
- Pipeline execution status: SUCCESS
- Confirmed CI workflow executed successfully

Step 9: Modified GitHub Repository File

- Opened GitHub repository: ci-demo
- Edited file: trigger-file.txt
- Added additional content to file
- Configured commit message: Update trigger file with additional content

- Clicked: Commit changes
- Verified GitHub repository updated successfully

Step 10: Verified GitHub Webhook Automation

- Opened: Jenkins → continuous-integration-job
- Verified:
 - Automated build triggered successfully after GitHub commit
- Confirmed multiple builds executed automatically
- Verified GitHub webhook integration functioning successfully

Step 11: Created Jenkins Continuous Deployment Pipeline Job

- Opened: Jenkins Dashboard → New Item
- Configured item name: continuous-deployment-job
- Selected: Pipeline
- Clicked: OK
- Verified deployment pipeline job created successfully

Step 12: Configured Deployment Pipeline Script

- Opened: continuous-deployment-job → Configure
- Configured deployment pipeline stages within Jenkins pipeline script
- Configured:
 - Environment stages
 - Build stages
 - Deployment stages
- Enabled: Use Groovy Sandbox
- Saved deployment pipeline successfully

Step 13: Configured AWS Deployment Integration

- Configured AWS deployment stage within Jenkins pipeline
- Configured:
 - AWS credentials
 - AWS region: eu-west-2
- Configured deployment artifact upload to AWS infrastructure
- Verified AWS deployment configuration completed successfully

Step 14: Executed Deployment Pipeline Workflow

- Opened: continuous-deployment-job
- Selected: Build Now
- Jenkins deployment stages executed:
 - Environment setup
 - Build execution
 - Deployment execution
 - Cleanup operations
- Verified deployment workflow executed successfully

Step 15: Verified Deployment Approval Workflow

- Verified Jenkins manual approval stage displayed successfully
- Confirmed deployment approval request executed successfully
- Verified deployment progression to production environment
- Confirmed deployment completed successfully

Step 16: Verified Successful Deployment Pipeline Execution

- Opened: Console Output
- Verified:
 - Maven build execution completed successfully
 - Artifact generated successfully
 - AWS deployment completed successfully
 - Workspace cleanup executed successfully
- Verified pipeline execution status: SUCCESS

Step 17: Verified Jenkins Build History

- Opened: continuous-deployment-job
- Verified build history:
 - Build #1
 - Build #2
 - Build #3
- Verified deployment targets:
 - UAT
 - PROD
- Confirmed successful deployment pipeline execution history available

Summary

- Jenkins Continuous Integration pipeline configured successfully
- GitHub repository integration configured successfully
- Jenkins SCM and Jenkinsfile pipeline integration validated successfully
- Automated GitHub webhook triggering configured successfully
- Jenkins Continuous Deployment pipeline configured successfully
- AWS deployment integration configured successfully
- Manual deployment approval workflow validated successfully

- Jenkins automated CI/CD delivery workflow executed successfully

Lab objective achieved:

Implemented Jenkins Continuous Integration and Continuous Deployment pipeline configuration, GitHub repository integration, Jenkinsfile automation, AWS deployment integration, webhook automation, and cloud-native DevOps workflow orchestration successfully within Jenkins CI/CD infrastructure.

New Item

Enter an item name

continuous-integration-job

Select an item type



Pipeline

Build, test, and deploy using pipelines. Supports stages, parallel work, and running on multiple agents.



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



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.



Organization Folder

Creates a set of multibranch project subfolders by scanning for repositories.

OK



ci-demo

Public

forked from [cloudboostauk/ci-demo](#)



main



1 Branch



0 Tags

Configure

- General
- Triggers
- Pipeline
- Advanced

General

Enabled ☒

Description

CI job to build, test and upload jar to S3 from Dev environments.

Plain text [Preview](#)

- ☐ Discard old builds ?
- ☐ Do not allow concurrent builds
- ☐ Do not allow the pipeline to resume if the controller restarts

☒ GitHub project

Project url ?

https://github.com/samuelakosaonyekwe/ci-demo.git

Advanced ▾

- ☐ Permission to Copy Artifact
- ☐ Pipeline speed/durability override ?
- ☐ Preserve stashes from completed builds ?
- ☐ This project is parameterized ?
- ☐ Throttle builds ?

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☒ GitHub hook trigger for GITScm polling ?
- ☐ Poll SCM ?
- ☐ Trigger builds remotely (e.g., from scripts) ?

Pipeline

Define your Pipeline using Groovy directly or pull it from source control.

Definition

Pipeline script from SCM ▾

SCM ?

Git ▾ ?

Repositories ?

Repository URL ?

https://github.com/samuelakosaonyekwe/ci-demo.git

Credentials ?

samuelakosaonyejekwe/***** (github access)

▼

+ Add

Advanced ▼

+ Add Repository

Branches to build ?

Branch Specifier (blank for 'any') ?

*/main

+ Add Branch

Repository browser ?

(Auto) ▼

Additional Behaviours

+ Add

Script Path ?

jenkinsfile

☒ Lightweight checkout ?

[Pipeline Syntax](#)

Advanced

Advanced ▼

Save

Apply

REST API

Jenkins 2.555.1



Jenkins

/ continuous-integration-job ▾ / #1 ▾



Status



Changes



Console Output



Edit Build Information



Delete build '#1'



Timings



Git Build Data



Pipeline Overview



Restart from Stage



Replay



Pipeline Steps

Console

Download

Copy

View as plain text

```
Started by user Akosa Samuel Onyejekwe
Obtained Jenkinsfile from git https://github.com/samuelakosaonyejekwe/ci-demo.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/continuous-integration-job
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential github-access
Cloning the remote Git repository
Cloning repository https://github.com/samuelakosaonyejekwe/ci-demo.git
> git init /var/lib/jenkins/workspace/continuous-integration-job # timeout=10
```





Jenkins

/ continuous-integration-job



/ #1



```
[WS-CLEANUP] done
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

samuelakosaonyejekwe / ci-demo

🔍 Type to search

<> Code

🔗 Pull requests

👤 Agents

🕒 Actions

📁 Projects

📖 Wiki

🛡️ Security and quality

📊 Insights

⚙️ Settings

📘 On April 24 we'll start using GitHub Copilot interaction data for AI model training unless you opt out. [Review this update](#) and manage your preferences in your [GitHub account settings](#).

Files

main

Go to file

> maven_project

Jenkinsfile

README.md

trigger_file.txt

ci-demo / trigger_file.txt in main

Cancel changes

Commit changes...

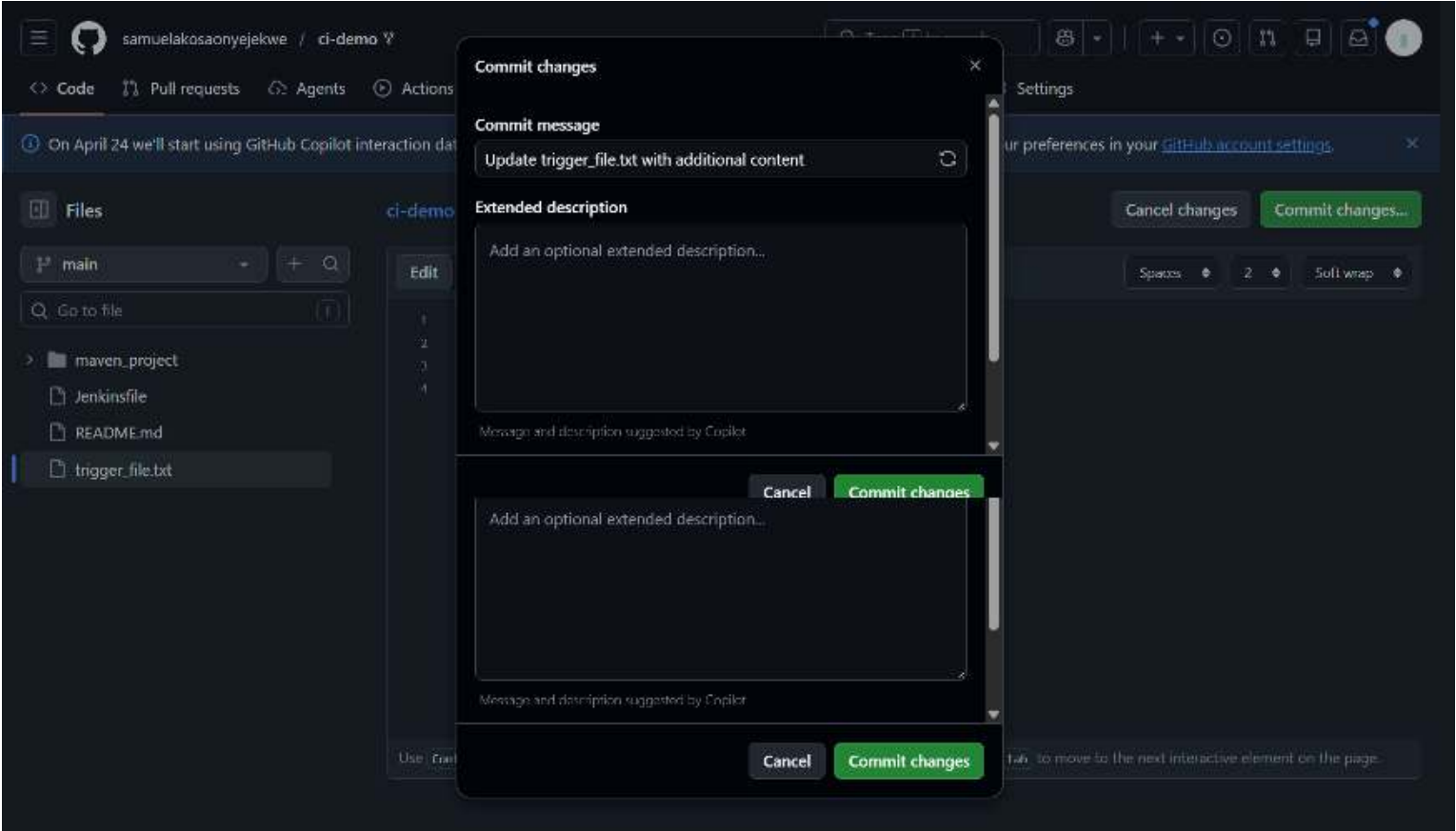
Edit

Preview

Spaces 2 Soft wrap

```
1 Hi this is sample file ...
2 hi ...
3 Testing 1 2 3...
4
```

Use `Control + Shift + n` to toggle the `tab` key moving focus. Alternatively, use `esc` then `Tab` to move to the next interactive element on the page.





Jenkins

/ continuous-integration-job



Stages



Rename



Pipeline Syntax



GitHub Hook Log

Builds >



Filter



Today



#2 11:46 AM



#1 11:40 AM



New Item

Enter an item name

continuous-deployment-job

Select an item type



Pipeline

Build, test, and deploy using pipelines. Supports stages, parallel work, and running on multiple agents.



Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.



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.



Organization Folder

Creates a set of multibranch project subfolders by scanning for repositories.



Duplicate an existing item

Create a new item by copying the configuration of an existing one.

OK

Configure

⚙️ General

🕒 Triggers

🔗 Pipeline

🔧 Advanced

General

Enabled 

Description

CD job to deploy jar to higher environments.

Plain text [Preview](#)

- ☐ Discard old builds ?
- ☐ Do not allow concurrent builds
- ☐ Do not allow the pipeline to resume if the controller restarts

☒ GitHub project

Project url ?

https://github.com/cloudboostauk/my-CICD-app

Advanced ▾

- ☐ Permission to Copy Artifact
- ☐ Pipeline speed/durability override ?
- ☐ Preserve stashes from completed builds ?
- ☐ This project is parameterized ?
- ☐ Throttle builds ?

Triggers

Set up automated actions that start your build based on specific events, like code changes or scheduled times.

- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☐ GitHub hook trigger for GITScm polling ?
- ☐ Poll SCM ?
- ☐ Trigger builds remotely (e.g., from scripts) ?

Pipeline

Define your Pipeline using Groovy directly or pull it from source control.

Definition

Pipeline script ▾

Script ?

[try sample Pipeline...](#) ▾

```
21 }
22 stage('download jar from s3') {
23     steps{
24         script{
25             withAWS(credentials: 'my-cba-aws-credential', region: 'eu-west-2') {
26                 sh '''echo "Downloading jar from s3 for deployment to ${ENV}" '''
27                 s3Download bucket: 'materialsfromlocalmachinesam', file: 'myproject.jar', path:
28             }
29         }
30     }
31 }
```

```
32 ~stage('maven deploy') {  
33 ~  steps {  
34 ~  script{  
35 ~
```

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

Advanced

Advanced ▾

Save

Apply

[REST API](#)

Jenkins 2.555.1

Status

Changes

Console Output

Edit Build Information

Parameters

Timings

Paused for Input

Pipeline Overview

Thread Dump

Pause/resume

Replay

Pipeline Steps

Workspaces

Previous Build

Console

Download

Copy

View as plain text

```
Started by user Akosa Samuel Onyejekwe
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/continuous-deployment-job
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Tool Install)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (init)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] cleanWs
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] done
[Pipeline] echo
Preparing job with variables
[Pipeline] echo
deploying in PROD
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (download jar from s3)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] withAWS
Constructing AWS CredentialsSetting AWS region eu-west-2
[Pipeline] {
[Pipeline] sh
+ echo Downloading jar from s3 for deployment to
Downloading jar from s3 for deployment to
[Pipeline] s3Download
Downloading s3://materialsfromlocalmachinesam/ci-demo/javaapp/myapp.jar to
file:/var/lib/jenkins/workspace/continuous-deployment-job/myproject.jar
Finished: Downloading from materialsfromlocalmachinesam/ci-demo/javaapp/myapp.jar
Download complete
[Pipeline] }
[Pipeline] // withAWS
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
```

```

[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (maven deploy)
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] tool
[Pipeline] envVarsForTool
[Pipeline] withEnv
[Pipeline] {
[Pipeline] script
[Pipeline] {
[Pipeline] input
Do you want to proceed deployment to production?
Proceed or Abort
Approved by Akosa Samuel Onyajakwe
[Pipeline] echo
Deploying....
Maven deploy....
[Pipeline] echo
mvn deploy:deploy-file -DgroupId=<group-id> -DartifactId=myapp.jar
-Dversion=<version> -Dpackaging=<type-of-packaging>
-Dfile=<path-to-file> -DrepositoryId=<id-to-map-on-server-section-of-settings.xml>
-Durl=<url-of-the-repository-to-deploy>
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Declarative: Post Actions)
[Pipeline] script
[Pipeline] {
[Pipeline] echo
this is always executed
[Pipeline] cleanWs
[WS-CLEANUP] Deleting project workspace...
[WS-CLEANUP] Deferred wipeout is used...
[WS-CLEANUP] done
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

```



Jenkins

/ continuous-deployment-j...

Recent

ⓘ Pipeline Syntax

Builds >

⋮

🔍 Filter

/

Today



#3 Started by phoenixsamuel 12:14 PM



| Target Env is: PROD



#2 Started by phoenixsamuel 12:11 PM



| Target Env is: UAT



#1 11:59 AM

