

AWS-Jenkins CI/CD Pipeline Setup

Project Report

Project Title:

AWS-Jenkins CI/CD Pipeline Setup

Prepared By:

Shruti Maheshbhai Patel

202203103510074

Table of Contents

Section	Title	Page
1	Introduction	3
2	Objective	3
3	Tools and Technologies Used	3
4	Architecture Overview	3
5	Implementation Steps	3
6	Application Deployment	3
7	Monitoring & Validation	3
8	Acknowledgment & Walkthrough	4
9	Conclusion	18

1. Introduction

This report outlines the implementation of a CI/CD pipeline using Jenkins with AWS CodeBuild and AWS CodeDeploy.

2. Objective

To create a fully automated CI/CD pipeline integrated with GitHub, AWS CodeBuild, CodeDeploy, and Jenkins.

3. Tools and Technologies Used

- Jenkins
- AWS CodeBuild
- AWS CodeDeploy
- AWS CloudFormation
- Amazon EC2, S3, IAM, Load Balancer

4. Architecture Overview

The architecture includes source code from GitHub triggering builds in CodeBuild, deployments via CodeDeploy, and automation managed by Jenkins.

5. Implementation Steps

Steps:

1. Setup infrastructure using CloudFormation
2. Launch Jenkins EC2 instance
3. Configure IAM roles and S3
4. Integrate CodeBuild & CodeDeploy

6. Application Deployment

GitHub repo includes buildspec.yml and appspec.yml for CodeBuild and CodeDeploy, along with deployment scripts.

7. Monitoring & Validation

Validate deployment through Jenkins Console Output and AWS CodeDeploy logs.

8. Acknowledgment & Walkthrough

Step 1: Launching CloudFormation Stack

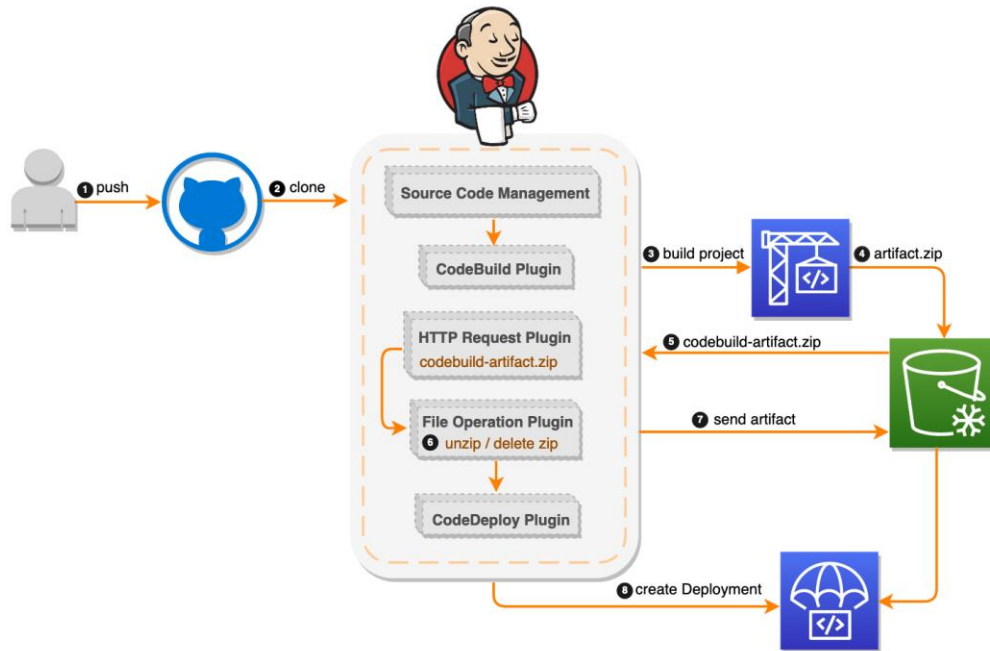


Figure 1: Screenshot of Launching CloudFormation Stack

Step 2: Jenkins EC2 Instance Setup

Response

Connection timeout	<input type="text" value="0"/>	?
Response codes expected	<input type="text" value="100:399"/>	?
Response content expected	<input type="text"/>	?
Output response to file	<input type="text" value="codebuild-artifact.zip"/>	?
Response body in console?	<input type="radio"/> Yes <input checked="" type="radio"/> No	?
Quiet all output?	<input type="radio"/> Yes <input checked="" type="radio"/> No	?

Figure 2: Screenshot of Jenkins EC2 Instance Setup

Step 3: Unlocking Jenkins

File Delete

Include File Pattern *

Exclude File Pattern

Add

HTTP Request

URL 1.amazonaws.com/jenkinscodedeploy-codedeploybucket-hnnb4ksa9rfb/codebuild-artifact.zip

HTTP mode GET

Ignore Ssl errors? ☒ Yes ☐ No

Http Proxy

Advanced...

Add build step

Figure 3: Screenshot of Unlocking Jenkins

Step 4: Jenkins Plugin Configuration

Parameter Store Variables Override

Timeout Override

File Operations

File Delete

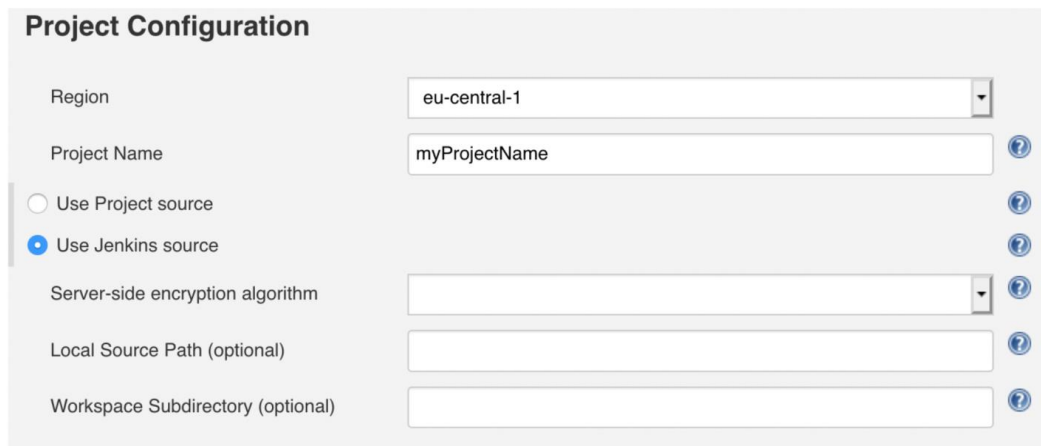
Include File Pattern *

Exclude File Pattern

Add

Figure 4: Screenshot of Jenkins Plugin Configuration

Step 5: Creating Jenkins Project



The screenshot shows the 'Project Configuration' page in Jenkins. It includes a 'Region' dropdown set to 'eu-central-1', a 'Project Name' text field with 'myProjectName', and two radio buttons for 'Use Project source' and 'Use Jenkins source' (the latter is selected). Below these are three more text fields: 'Server-side encryption algorithm', 'Local Source Path (optional)', and 'Workspace Subdirectory (optional)'. Each field has a help icon to its right.

Project Configuration

Region: eu-central-1

Project Name: myProjectName

☐ Use Project source

☒ Use Jenkins source

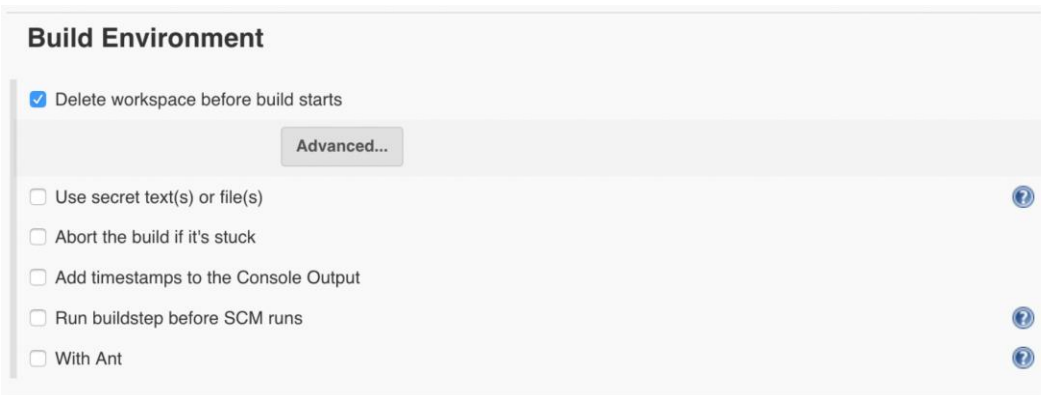
Server-side encryption algorithm:

Local Source Path (optional):

Workspace Subdirectory (optional):

Figure 5: Screenshot of Creating Jenkins Project

Step 6: GitHub Repository Setup



The screenshot shows the 'Build Environment' page. It features a checked checkbox for 'Delete workspace before build starts' and an 'Advanced...' button. Below this are several unchecked checkboxes: 'Use secret text(s) or file(s)', 'Abort the build if it's stuck', 'Add timestamps to the Console Output', 'Run buildstep before SCM runs', and 'With Ant'. Each checkbox has a help icon to its right.

Build Environment

☒ Delete workspace before build starts

Advanced...

☐ Use secret text(s) or file(s)

☐ Abort the build if it's stuck

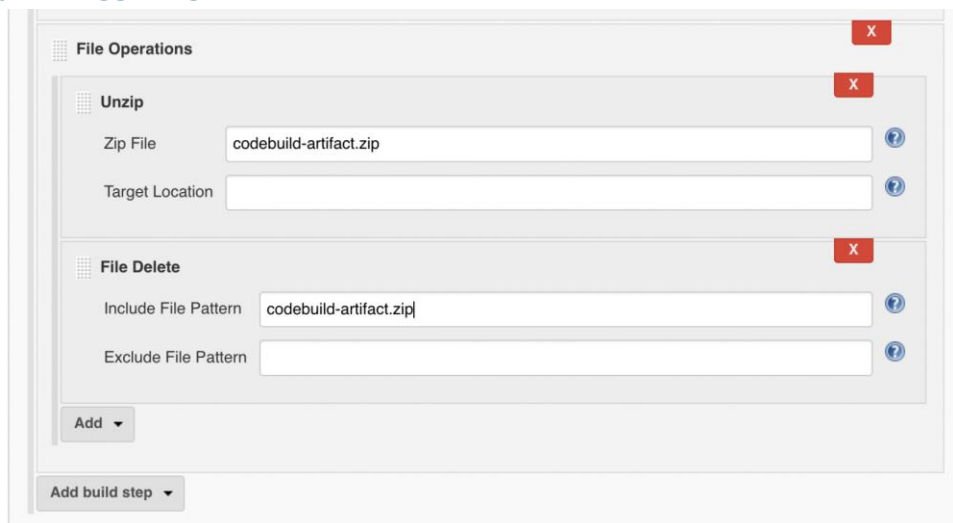
☐ Add timestamps to the Console Output

☐ Run buildstep before SCM runs

☐ With Ant

Figure 6: Screenshot of GitHub Repository Setup

Step 7: Triggering Build from GitHub



The screenshot shows the 'Build Steps' configuration page. It has a 'File Operations' section with an 'Unzip' step and a 'File Delete' step. The 'Unzip' step has a 'Zip File' field with 'codebuild-artifact.zip' and a 'Target Location' field. The 'File Delete' step has an 'Include File Pattern' field with 'codebuild-artifact.zip' and an 'Exclude File Pattern' field. There are 'Add' and 'Add build step' buttons at the bottom.

File Operations

Unzip

Zip File: codebuild-artifact.zip

Target Location:

File Delete

Include File Pattern: codebuild-artifact.zip

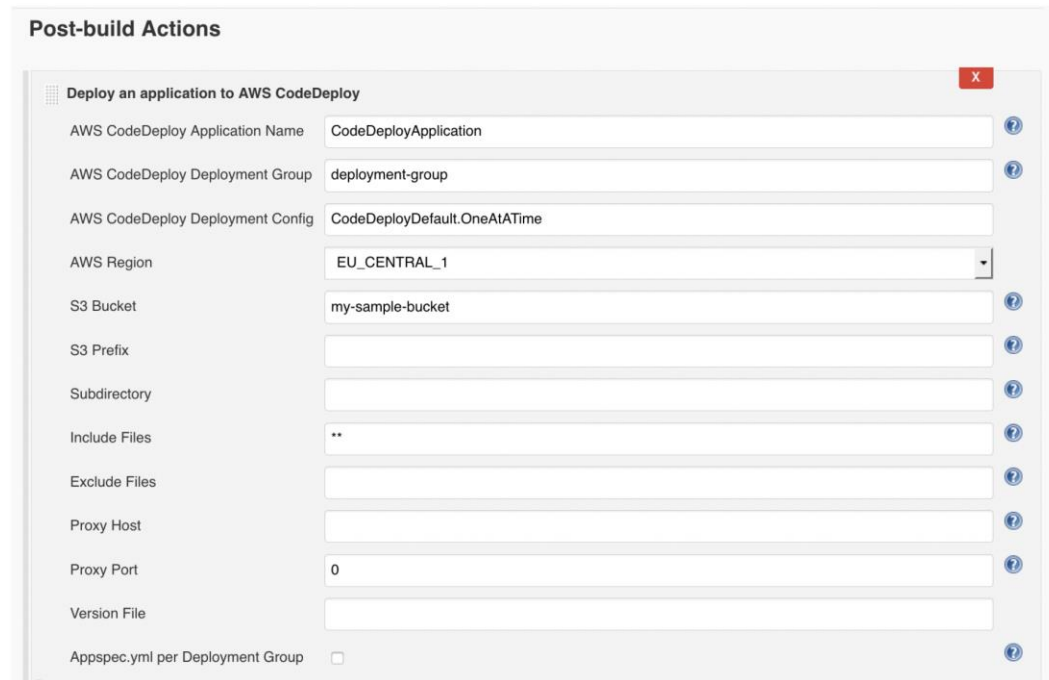
Exclude File Pattern:

Add

Add build step

Figure 7: Screenshot of Triggering Build from GitHub

Step 8: CodeBuild Execution



Post-build Actions

Deploy an application to AWS CodeDeploy

AWS CodeDeploy Application Name: CodeDeployApplication

AWS CodeDeploy Deployment Group: deployment-group

AWS CodeDeploy Deployment Config: CodeDeployDefault.OneAtATime

AWS Region: EU_CENTRAL_1

S3 Bucket: my-sample-bucket

S3 Prefix:

Subdirectory:

Include Files: **

Exclude Files:

Proxy Host:

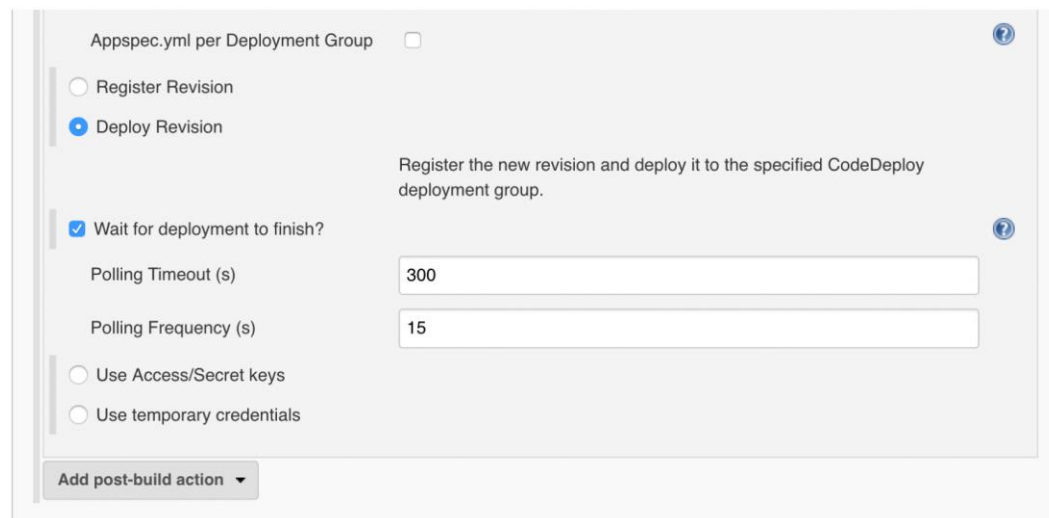
Proxy Port: 0

Version File:

Appspec.yml per Deployment Group: ☐

Figure 8: Screenshot of CodeBuild Execution

Step 9: CodeDeploy Deployment Monitoring



Appspec.yml per Deployment Group: ☐

☐ Register Revision

☒ Deploy Revision

☒ Wait for deployment to finish?

Register the new revision and deploy it to the specified CodeDeploy deployment group.

Polling Timeout (s): 300

Polling Frequency (s): 15

☐ Use Access/Secret keys

☐ Use temporary credentials

Add post-build action

Figure 9: Screenshot of CodeDeploy Deployment Monitoring

Step 10: Verifying Application on ELB



Figure 10: Screenshot of Verifying Application on ELB

Step 11: Additional Screenshot

```
Response Code: HTTP/1.1 200 OK
Success code from [100..399]
Saving response body to /var/lib/jenkins/workspace/SampleCodeDeployApp/codebuild-artifact.zip
Unzip File Operation:
Unzipping codebuild-artifact.zip to /var/lib/jenkins/workspace/SampleCodeDeployApp
Unzip completed.
File Delete Operation:
/var/lib/jenkins/workspace/SampleCodeDeployApp/codebuild-artifact.zip deleting...
Success.
Zipping files into /tmp/#6-4298943850890137367.zip
Uploading zip to s3://codedeployblog-codedeploybucket-#6-4298 43850890137367.zip
Registering revision for application 'CodeDeployBlog-DemoApplication-1GJ0EG LMKRV7'
Creating deployment with revision at {RevisionType: 'S3', S3Location: {Bucket: codedeployblog-codedeploybucket-#6-4298943850890137367.zip, BundleType: zip, ETag: 3c08a5a095e9955f63d842ab3 ad4f7}, }, Key:
Monitoring deployment with ID d-HCEIP4LMC...
Deployment status: Created; instances: null
Deployment status: InProgress; instances: {Pending: 1, InProgress: 1, Succeeded: 1, Failed: 0, Skipped: 0, Ready: 0}
Deployment status: InProgress; instances: {Pending: 0, InProgress: 1, Succeeded: 2, Failed: 0, Skipped: 0, Ready: 0}
Deployment status: Succeeded; instances: {Pending: 0, InProgress: 0, Succeeded: 3, Failed: 0, Skipped: 0, Ready: 0}
Finished: SUCCESS
```

Figure 11: Screenshot of pipeline process

Step 12: Additional Screenshot

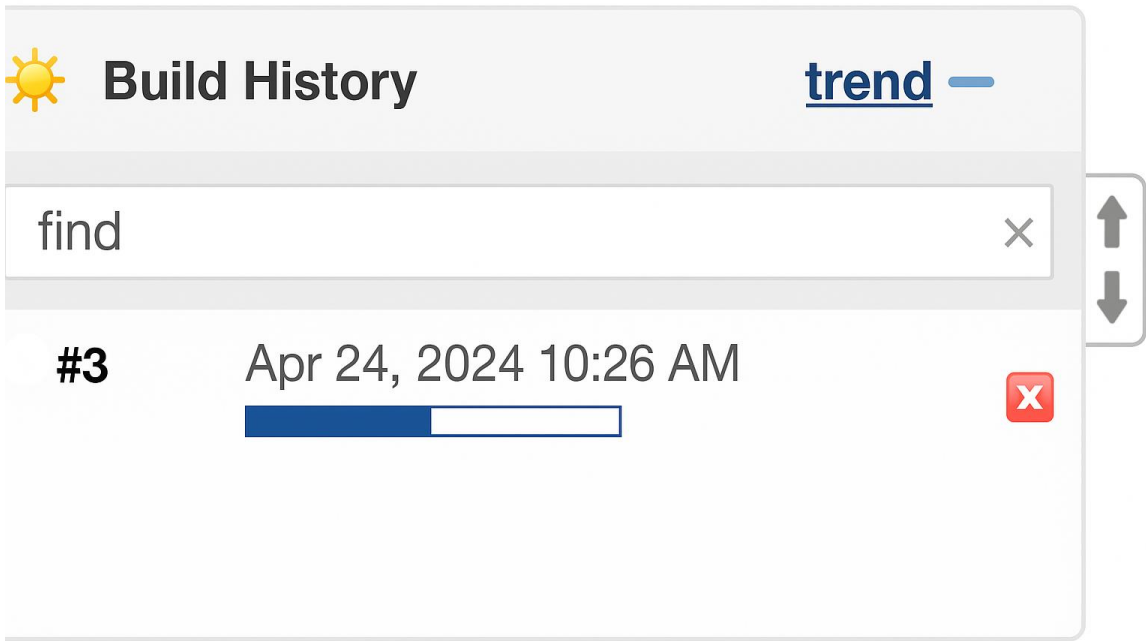


Figure 12: Screenshot of pipeline process

Step 13: Additional Screenshot

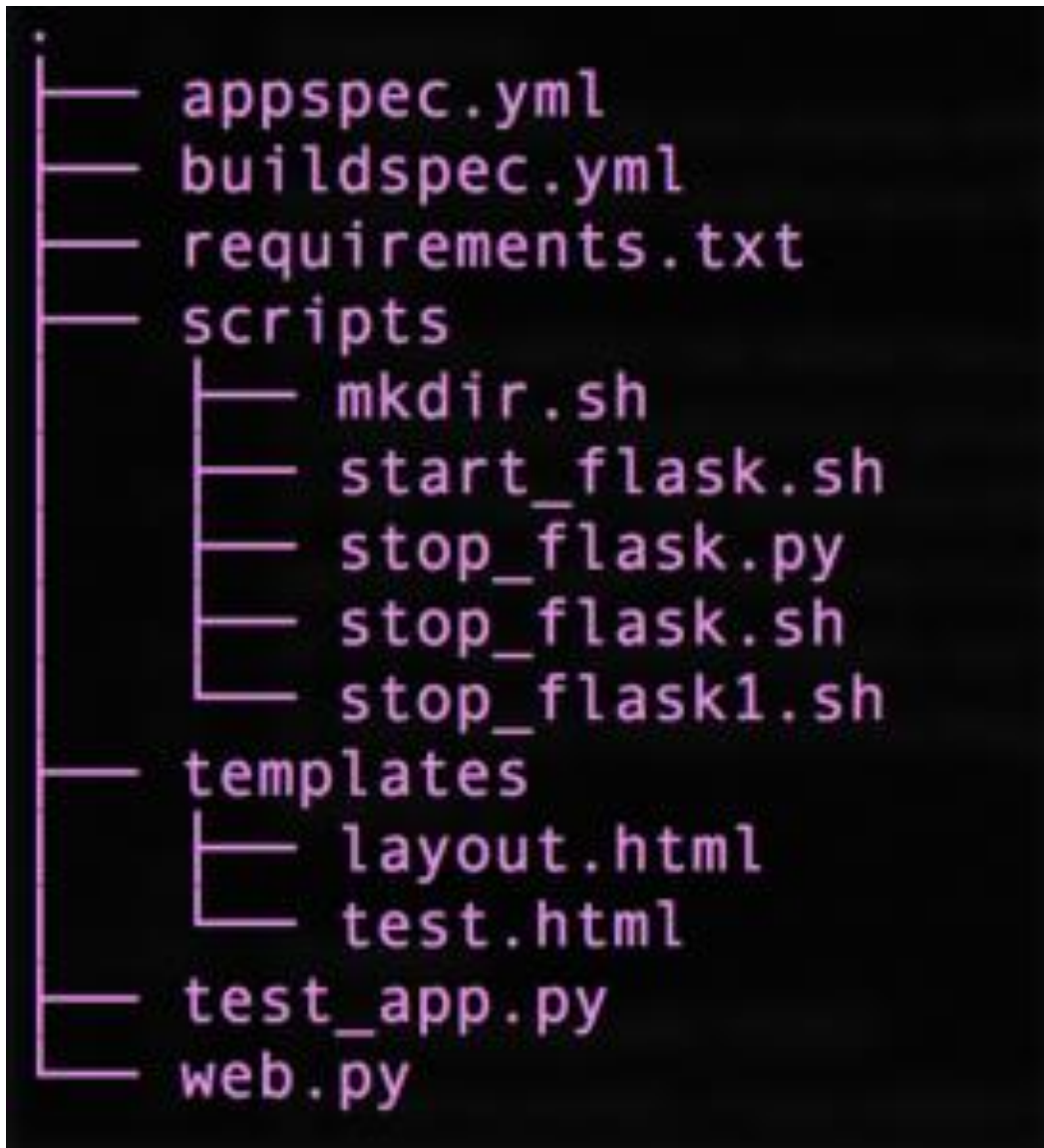
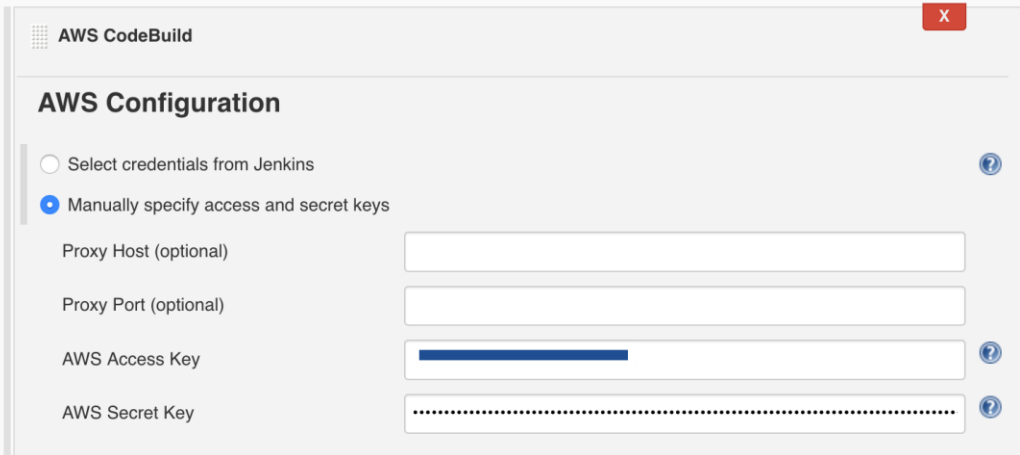


Figure 13: Screenshot of pipeline process

Step 14: Additional Screenshot

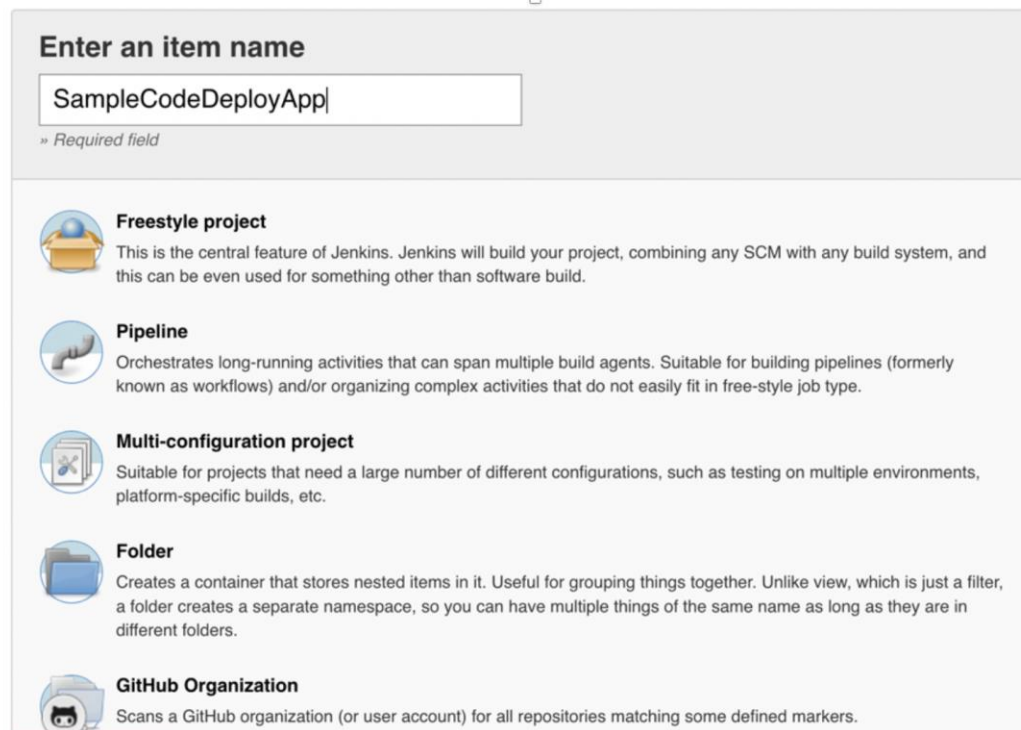
Build



The screenshot shows the 'AWS CodeBuild' configuration window. At the top, there's a title bar with 'AWS CodeBuild' and a red close button. Below the title bar, the 'AWS Configuration' section is visible. It has two radio buttons: 'Select credentials from Jenkins' (unselected) and 'Manually specify access and secret keys' (selected). Below these are four input fields: 'Proxy Host (optional)', 'Proxy Port (optional)', 'AWS Access Key', and 'AWS Secret Key'. The 'AWS Access Key' field has a blue bar, and the 'AWS Secret Key' field has a dotted line. There are help icons (question marks) next to the 'Manually specify access and secret keys' option and the 'AWS Access Key' and 'AWS Secret Key' fields.

Figure 14: Screenshot of pipeline process

Step 15: Additional Screenshot



The screenshot shows the 'Enter an item name' dialog in Jenkins. The title is 'Enter an item name'. Below the title is a text input field containing 'SampleCodeDeployApp'. Below the input field is a small text label '» Required field'. Below the input field is a list of project types with icons and 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.
- GitHub Organization**: Scans a GitHub organization (or user account) for all repositories matching some defined markers.

Figure 15: Screenshot of pipeline process

Step 16: Additional Screenshot

Source Code Management

☐ None
☐ CA Harvest
☐ File System
☒ Git

Repositories

Repository URL

Credentials

Branches to build

Branch Specifier (blank for 'any')

Figure 16: Screenshot of pipeline process

Step 17: Additional Screenshot

Build Triggers

☐ Trigger builds remotely (e.g., from scripts)
☐ Build after other projects are built
☐ Build periodically
☐ Build when a change is pushed to GitBucket
☐ GitHub Branches
☐ GitHub Pull Requests
☐ GitHub hook trigger for GITScm polling
☒ Poll SCM

Schedule

Would last have run at Saturday, September 22, 2018 9:43:23 PM UTC; would next run at Saturday, September 22, 2018 9:43:23 PM UTC.

Ignore post-commit hooks ☐

Figure 17: Screenshot of pipeline process

Step 18: Additional Screenshot

Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Figure 18: Screenshot of pipeline process

Step 19: Additional Screenshot

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log ([not sure where to find it?](#)) and this file on the server:

```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

 This connection is not secure. Logins entered here could be compromised. [Learn More](#)

Figure 19: Screenshot of pipeline process

Step 20: Additional Screenshot

Specify Details

Specify a stack name and parameter values. You can use or change the default parameter values, which are defined in the AWS CloudFormation template. [Learn more.](#)

Stack name

Parameters

CodeBuildProject	<input type="text"/>	Enter the project name of the CodeBuild
CodedeployInstanceType	<input type="text" value="t2.medium"/>	EC2 instance type for CodeDeploy Web Servers
InstanceCount	<input type="text" value="3"/>	Number of CodeDeploy Web Server EC2 instances
JenkinsInstanceType	<input type="text" value="t2.medium"/>	EC2 instance type for Jenkins Server
KeyName	<input type="text" value="Search"/>	The EC2 Key Pair to allow SSH access to CodeDeploy EC2 instances and Jenkins Server
PublicSubnet1	<input type="text" value="Search by ID, or Name tag value"/>	The first public subnet where the Jenkins EC2 instance, ELB and CodeDeploy Web Servers will be launched
PublicSubnet2	<input type="text" value="Search by ID, or Name tag value"/>	The second public subnet where the ELB and CodeDeploy Web Servers will be launched
VpcId	<input type="text" value="Search by ID, or Name tag value"/>	The VPC Id where the EC2 instances will be launched.
YourIPRange	<input type="text"/>	CIDR block of the network from where you will connect to the Jenkins server using HTTP and SSH

Figure 20: Screenshot of pipeline process

Step 21: Additional Screenshot

CodeDeployBlog

Delete

Update

Stack actions ▾

Create stack

Stack info

Events

Resources

Outputs

Parameters

Template

Change sets

Outputs (6)

Key	Value	Description	Export name
CodeBuildProjectName	CBprojectName	CodeBuild Project Name	-
CodeDeployApplicationName	CodeDeployBlog-DemoApplication	CodeDeploy Application Name	-
CodeDeployDeploymentGroup	CodeDeployBlog-DemoFleet	CodeDeploy Deployment Group Name	-
ELBDNSName	CodeDeployBlog-ELB- m eu-central-1.elb.amazonaws.co	DNS Name of the ELB	-
JenkinsServerDNSName	ec2- eu-central-1.compute.amazonaws.com	DNS Name of Jenkins Server	-
S3BucketName	codedeployblog-codedeploybucket	S3 Bucket Name	-

Figure 21: Screenshot of pipeline process

Step 22: Additional Screenshot

Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Figure 22: Screenshot of pipeline process

Step 23: Additional Screenshot

Getting Started

Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Figure 23: Screenshot of pipeline process

Step 24: Additional Screenshot

Getting Started

Getting Started

✓ Folders Plugin	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding
✓ Timestamper	✓ Workspace Cleanup	✓ Ant	✓ Gradle
✓ Pipeline	🔄 GitHub Branch Source	🔄 Pipeline: GitHub Groovy Libraries	✓ Pipeline: Stage View
✓ Git	🔄 Subversion	🔄 SSH Slaves	<input type="radio"/> Matrix Authorization Strategy
🔄 PAM Authentication	🔄 LDAP	🔄 Email Extension	✓ Mailer

```
bundle
Pipeline: Stage View
** Pipeline: Build Step
** Pipeline: Model API
** Pipeline: Declarative
Extension Points API
** JSch dependency
** Git client
** GIT server
** Pipeline: Shared Groovy
Libraries
** Display URL API
Mailer
** Branch API
** Pipeline: Multibranch
** Authentication Tokens API
** Docker Commons
** Pipeline: Basic Steps
** Docker Pipeline
** Pipeline: Stage Tags Metadata
** Pipeline: Declarative Agent
API
** Pipeline: Declarative
Pipeline
** GitHub API
Git
** GitHub
** - required dependency
```

Figure 24: Screenshot of pipeline process

Step 25: Additional Screenshot



Please wait while Jenkins is getting ready to work ...

Your browser will reload automatically when Jenkins is ready.

Figure 25: Screenshot of pipeline process

Step 26: Additional Screenshot

Installing Plugins/Upgrades



Preparation

- Checking internet connectivity
- Checking update center connectivity
- Success

HTTP Request

 Success

Amazon Web Services SDK

 Installing



AWS CodeDeploy

 Pending

AWS CodeBuild

 Pending


File Operations


 Pending

Loading plugin extensions

 Pending

Restarting Jenkins

 Pending

 [Go back to the top page](#)
(you can start using the installed plugins right away)

 ☒ Restart Jenkins when installation is complete and no jobs are running

Figure 26: Screenshot of pipeline process

Step 27: Additional Screenshot

Getting Started

Jenkins is ready!

You have skipped the **setup of an admin user**.

To log in, use the username: "admin" and the administrator password you used to access the setup wizard.

Your Jenkins setup is complete.

[Start using Jenkins](#)

Figure 27: Screenshot of pipeline process

Step 28: Additional Screenshot

Manage Jenkins



Configure System

Configure global settings and paths.



Configure Global Security

Secure Jenkins; define who is allowed to access/use the system.



Configure Credentials

Configure the credential providers and types



Global Tool Configuration

Configure tools, their locations and automatic installers.



Reload Configuration from Disk

Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.



Manage Plugins

Add, remove, disable or enable plugins that can extend the functionality of Jenkins.



System Information

Displays various environmental information to assist trouble-shooting.

Figure 28: Screenshot of pipeline process

Step 29: Additional Screenshot

Filter:

Updates Available Installed Advanced

Install ↓	Name	Version
<input checked="" type="checkbox"/>	AWS CodeDeploy This plugin provides a "post-build" step for AWS CodeDeploy.	1.21
<input type="checkbox"/>	JDCloud CodeDeploy This plugin provides a "post-build" step for JDCloud CodeDeploy.	1.0.0

Install without restart Download now and install after restart Update information obtained: 7 min 5 sec ago Check

Figure 29: Screenshot of pipeline process

9. Conclusion

A fully functional CI/CD pipeline was successfully built using Jenkins, CodeBuild, and CodeDeploy. This setup ensures automation, faster release cycles, and high reliability in application deployments.