

Practice for Lesson 6: Blue Ocean (Advanced Jenkins)

Practices for Lesson 6

Overview

In these practices, you will learn how to install the Blue Ocean plugin in Jenkins and further create a Pipeline in Jenkins using Blue Ocean.

.

Practice 6-1: Create Pipeline using Blue Ocean

Overview

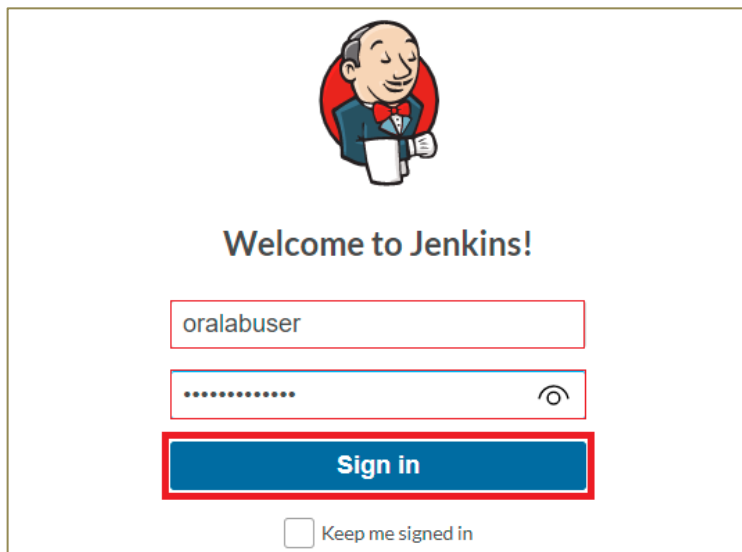
In this practice, you will learn how to create Pipeline in Jenkins using BlueOcean.

Assumptions

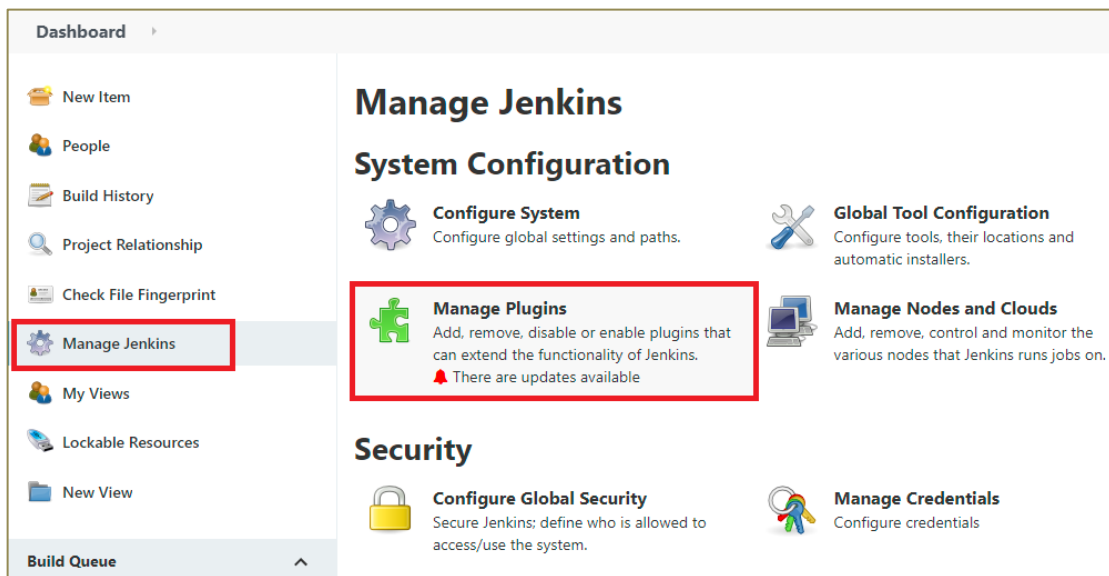
You should have completed the Practice of Lesson 5.

Tasks

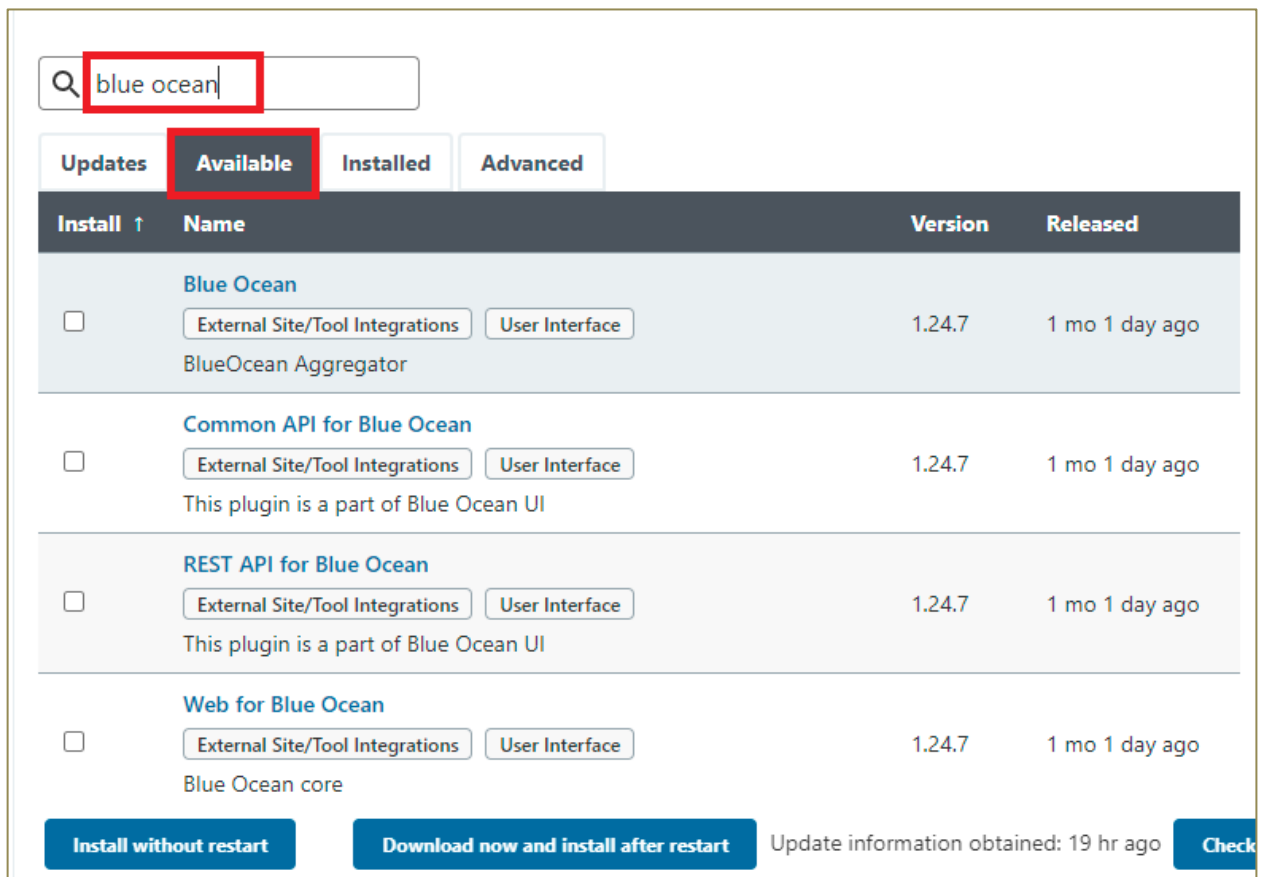
1. Sign in to **Jenkins Instance Dashboard**.
 - a. In a browser on your local machine, enter the **Public IP** address of the EC2 instance followed by the IP address to sign in to the Jenkins Dashboard (for example, **<Public-IP>:8080**).



- b. Enter the user name and password provided.
 - c. You will have access to the Jenkins Dashboard.
2. Install Blue Ocean on the **Linux instance**.
 - a. Navigate to Dashboard, select **Manage Jenkins** and click **Manage Plugins** as shown below.



- b. Select **Available** and search for **Blue Ocean**.



- c. Select the check box of the **Blue Ocean** plugin and click **Install without restart**.

Search: blue ocean

Updates Available Installed Advanced

| Install ↑ | Name | Version | Released |
|-------------------------------------|--|---------|----------------|
| <input checked="" type="checkbox"/> | Blue Ocean External Site/Tool Integrations User Interface BlueOcean Aggregator | 1.24.7 | 1 mo 1 day ago |
| <input type="checkbox"/> | Common API for Blue Ocean External Site/Tool Integrations User Interface This plugin is a part of Blue Ocean UI | 1.24.7 | 1 mo 1 day ago |
| <input type="checkbox"/> | REST API for Blue Ocean External Site/Tool Integrations User Interface This plugin is a part of Blue Ocean UI | 1.24.7 | 1 mo 1 day ago |
| <input type="checkbox"/> | Web for Blue Ocean External Site/Tool Integrations User Interface Blue Ocean core | 1.24.7 | 1 mo 1 day ago |

[Install without restart](#)
[Download now and install after restart](#)
 Update information obtained: 19 hr ago
 [Check for updates](#)

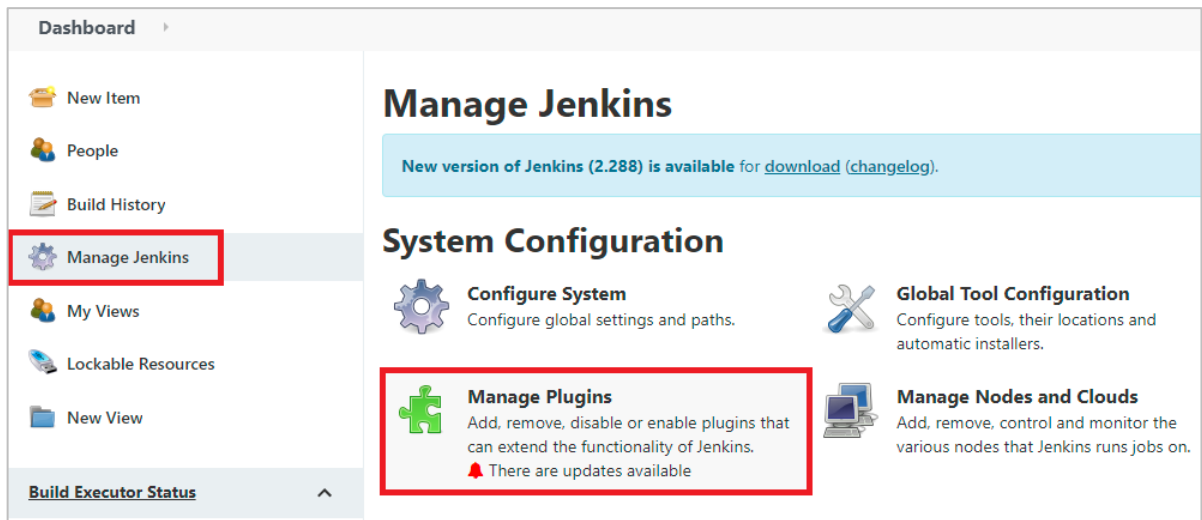
- d. The installation process will proceed and Success message will be displayed as shown below.

| | |
|-----------------------------------|-----------|
| Authentication Tokens API | ✓ Success |
| Handy Uri Templates 2.x API | ✓ Success |
| Bitbucket Branch Source | ✓ Success |
| Bitbucket Pipeline for Blue Ocean | ✓ Success |
| Dashboard for Blue Ocean | ✓ Success |
| Personalization for Blue Ocean | ✓ Success |
| Display URL for Blue Ocean | ✓ Success |
| Server Sent Events (SSE) Gateway | ✓ Success |
| Events API for Blue Ocean | ✓ Success |
| Blue Ocean Pipeline Editor | ✓ Success |
| i18n for Blue Ocean | ✓ Success |
| Autofavorite for Blue Ocean | ✓ Success |
| Blue Ocean | ✓ Success |
| Loading plugin extensions | ✓ Success |

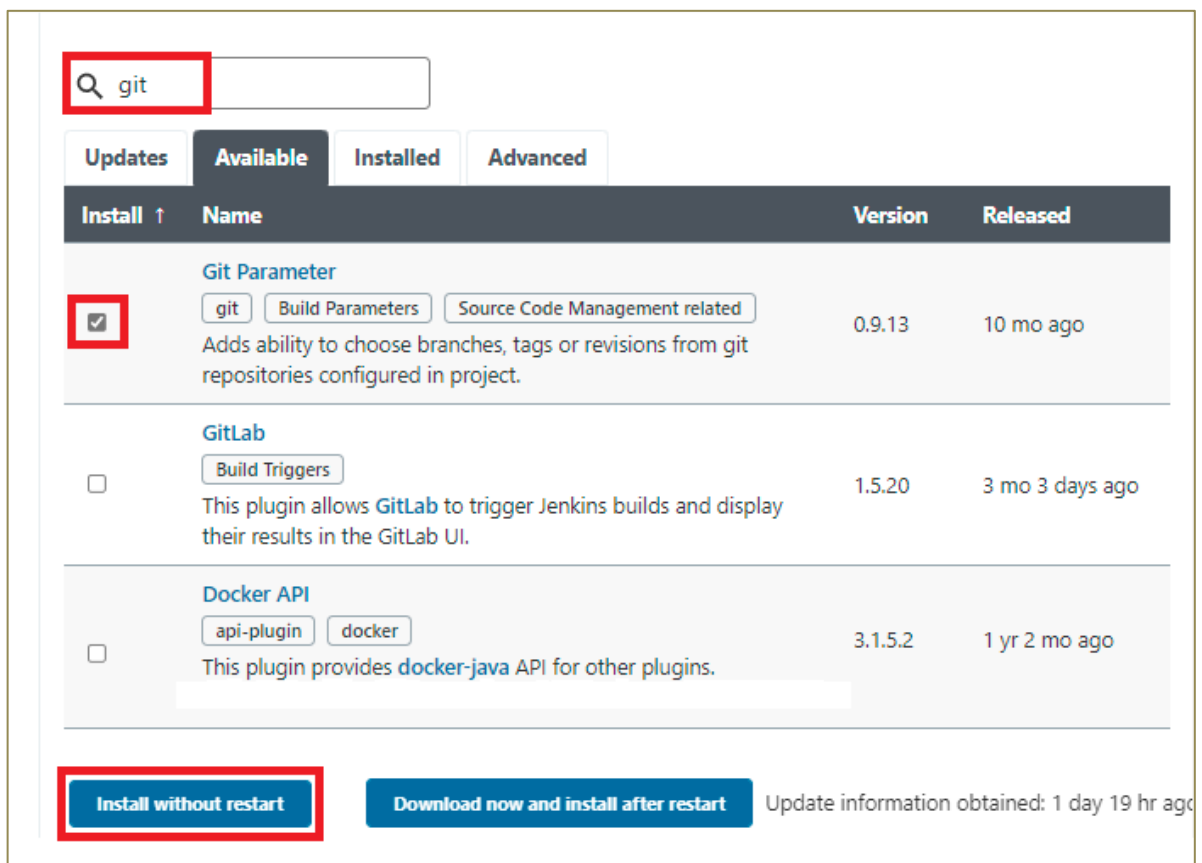
[Go back to the top page](#)
 (you can start using the installed plugins right away)

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

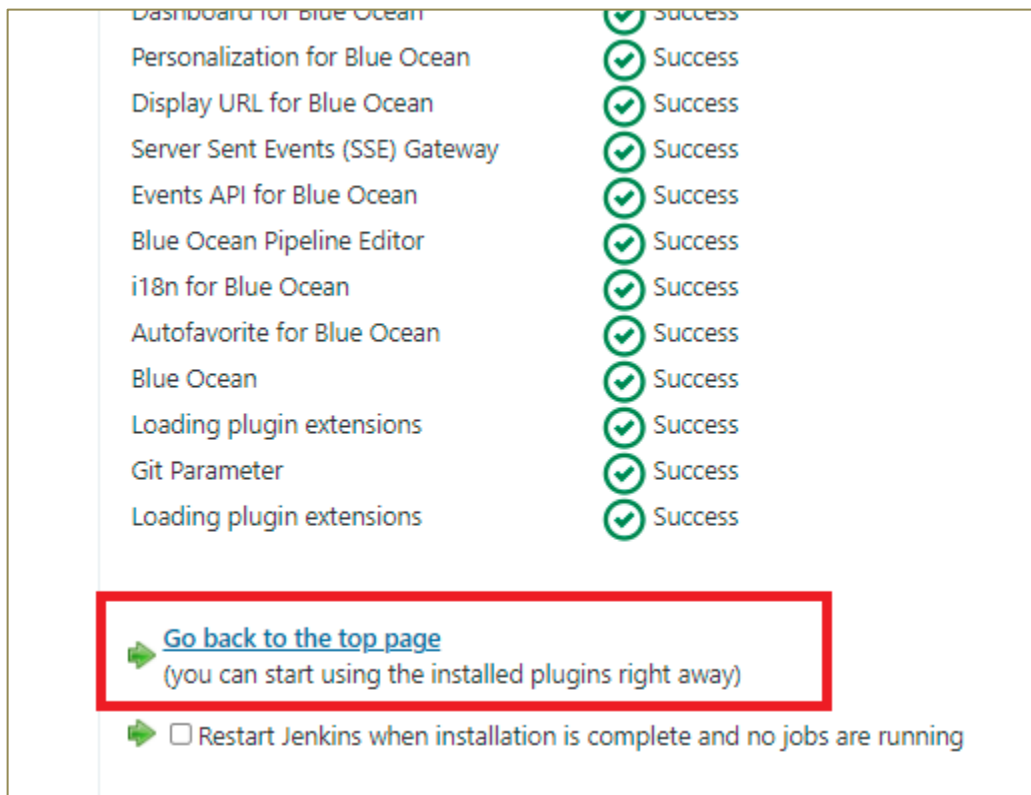
- e. Similarly, install **Git Parameter** plugin in the Jenkins. Navigate to **Main menu** and select **Manage Plugins** as shown below.



- f. Select **Available**, search for **Git** and select the check box of **Git Parameter**. Click **Install without restart**.

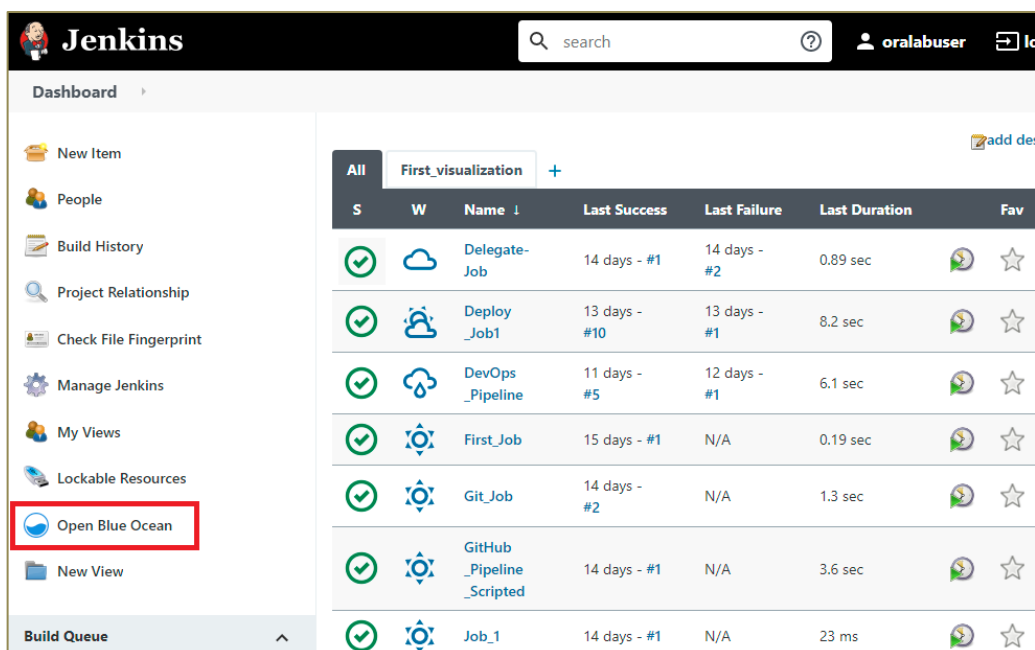


- g. As shown below the installation of the plugin is successful, click **Go back to the top page**.



3. Create Jenkins Pipeline using **Blue Ocean**.

- a. In the Dashboard click **Open Blue Ocean** to create the pipeline as shown below.



- b. View the Pipelines page and click on **New Pipeline** to create a pipeline in Jenkins.

The Jenkins interface shows the 'Pipelines' tab. At the top right, there is a 'Logout' button. Below the header, there is a search bar labeled 'Search pipelines...' and a 'New Pipeline' button, which is highlighted with a red rectangular box.

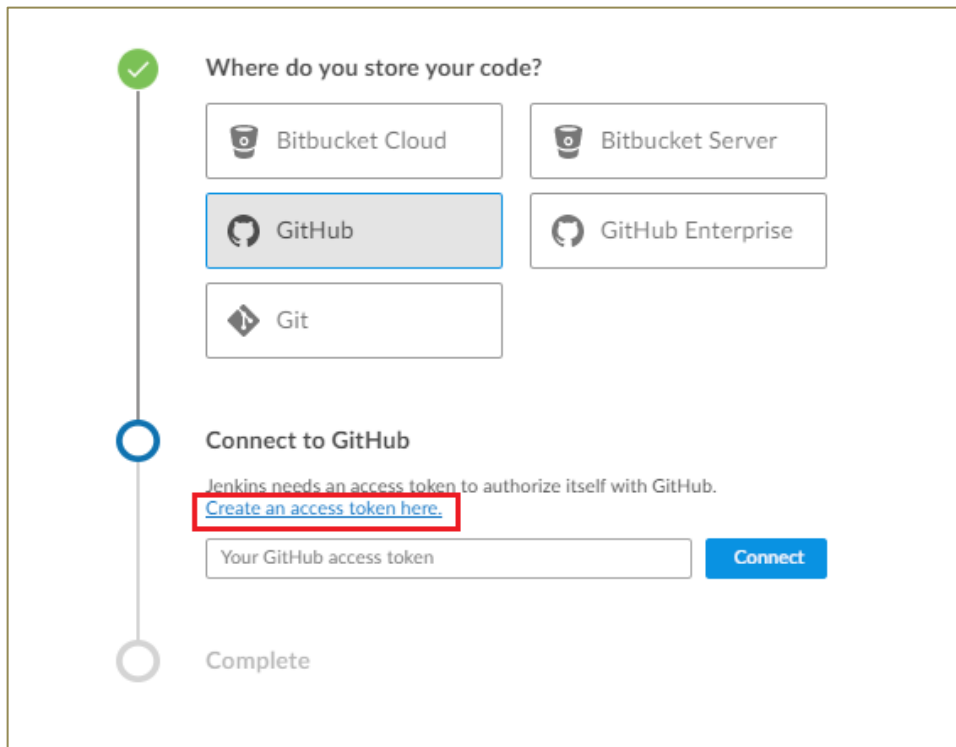
| NAME | HEALTH | BRANCHES | PR |
|--------------------------|--------|----------|----|
| Delegate-Job | | - | - |
| Deploy_Job1 | | - | - |
| DevOps_Pipeline | | - | - |
| First_Job | | - | - |
| Git_Job | | - | - |
| GitHub_Pipeline_Scripted | | - | - |
| Job_1 | | - | - |

- c. For “Where do you store your code”, select **GitHub** as shown below.

The Jenkins interface shows the 'Create Pipeline' page. The 'Pipelines' tab is selected. Below the header, there is a 'Classic Item Creation' link. The main content area is titled 'Where do you store your code?' and displays several options in a grid:

- Bitbucket Cloud
- Bitbucket Server
- GitHub** (highlighted with a red box)
- GitHub Enterprise
- Git

- d. To Connect to GitHub, an access token is required. Click on **Create an access token here**.



The image shows the Jenkins configuration interface for connecting to GitHub. A vertical progress bar on the left has three circles: the top one is green with a checkmark, the middle one is blue, and the bottom one is grey. The first step, 'Where do you store your code?', is completed. It lists five options: Bitbucket Cloud, Bitbucket Server, GitHub (highlighted with a blue border), GitHub Enterprise, and Git. The second step, 'Connect to GitHub', is active. It includes the text 'Jenkins needs an access token to authorize itself with GitHub.' and a red-bordered link 'Create an access token here.' Below this is a text input field labeled 'Your GitHub access token' and a blue 'Connect' button. The third step, 'Complete', is shown at the bottom.

Where do you store your code?

- Bitbucket Cloud
- Bitbucket Server
- GitHub
- GitHub Enterprise
- Git

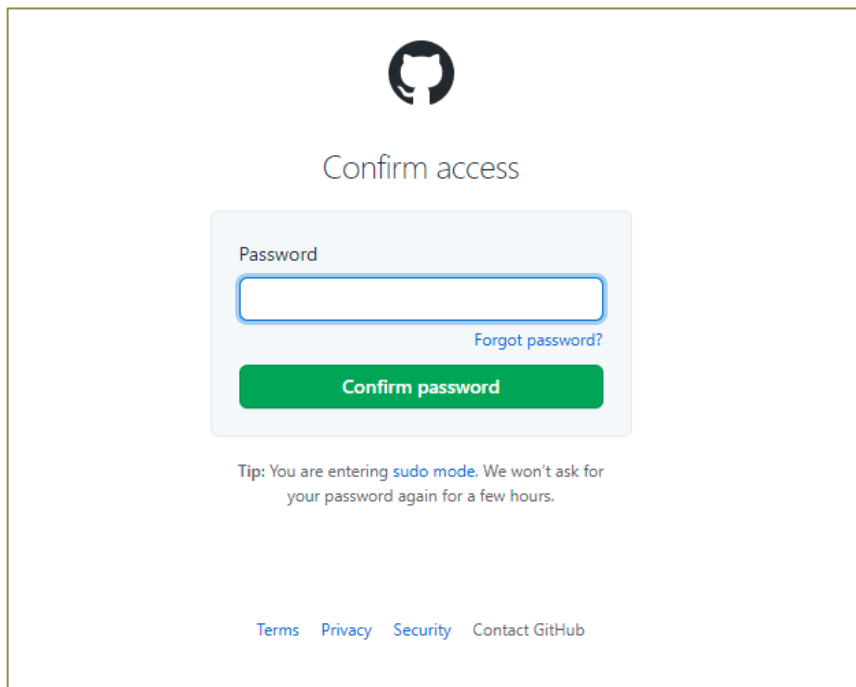
Connect to GitHub

Jenkins needs an access token to authorize itself with GitHub.
[Create an access token here.](#)

Your GitHub access token

Complete

- e. Login to the respective GitHub repository with the credentials as shown below.



The image shows the GitHub 'Confirm access' screen. At the top is the GitHub logo. Below it is the title 'Confirm access'. There is a password input field with the label 'Password' above it. To the right of the input field is a link 'Forgot password?'. Below the input field is a green button labeled 'Confirm password'. At the bottom, there is a tip: 'Tip: You are entering sudo mode. We won't ask for your password again for a few hours.' At the very bottom are links for 'Terms', 'Privacy', 'Security', and 'Contact GitHub'.

Confirm access

Password

[Forgot password?](#)

Tip: You are entering `sudo mode`. We won't ask for your password again for a few hours.

[Terms](#) [Privacy](#) [Security](#) [Contact GitHub](#)

- f. Provide the Name for the **Token** as shown below.

Settings / Developer settings

GitHub Apps
OAuth Apps
Personal access tokens

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

My_Token

What's this token for?

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 |

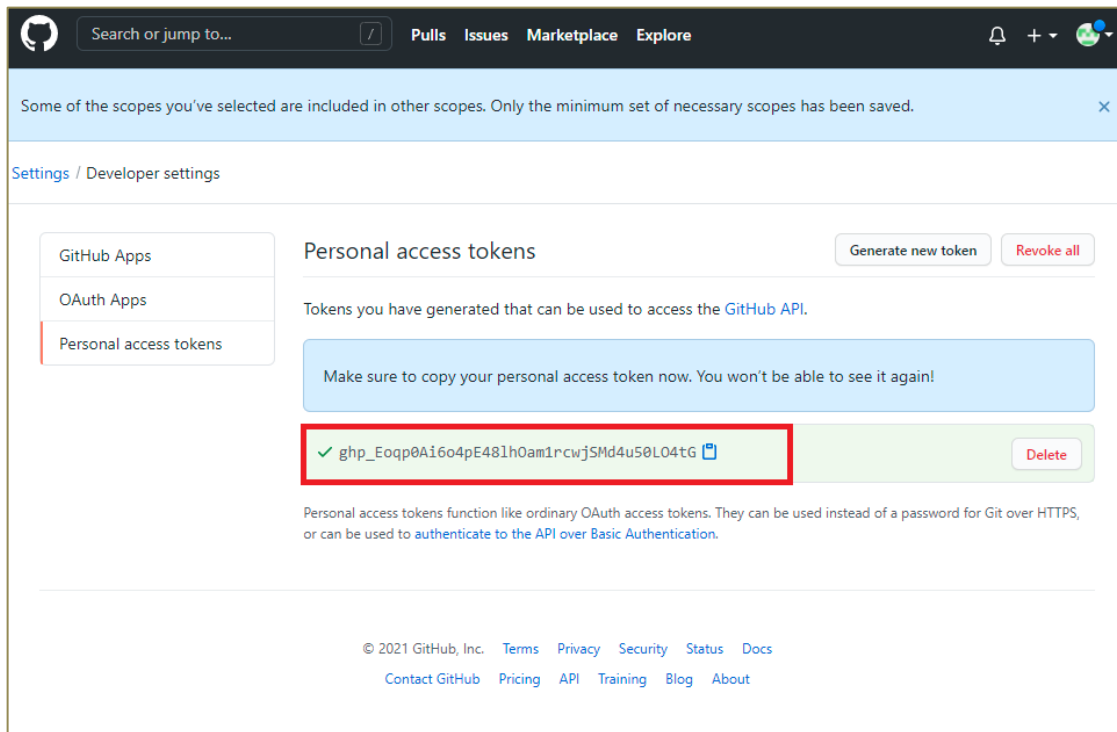
- g. Accept the default values and click **Generate token** as shown below.

| | |
|--|--|
| <input type="checkbox"/> user | Update ALL user data |
| <input checked="" type="checkbox"/> read:user | Read ALL user profile data |
| <input checked="" type="checkbox"/> user:email | Access user email addresses (read-only) |
| <input type="checkbox"/> user:follow | Follow and unfollow users |
| <input type="checkbox"/> delete_repo | Delete repositories |
| <input type="checkbox"/> write:discussion | Read and write team discussions |
| <input type="checkbox"/> read:discussion | Read team discussions |
| <input type="checkbox"/> admin:enterprise | Full control of enterprises |
| <input type="checkbox"/> manage_billing:enterprise | Read and write enterprise billing data |
| <input type="checkbox"/> read:enterprise | Read enterprise profile data |
| <input type="checkbox"/> admin:gpg_key | Full control of public user GPG keys (Developer Preview) |
| <input type="checkbox"/> write:gpg_key | Write public user GPG keys |
| <input type="checkbox"/> read:gpg_key | Read public user GPG keys |

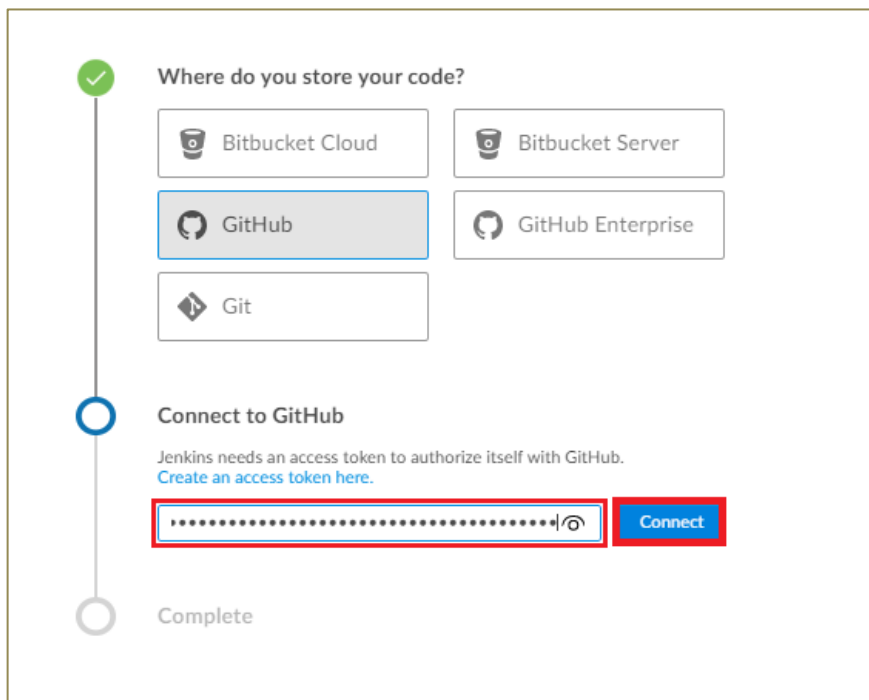
Generate token

Cancel

- h. The Token is generated successfully, copy the **Token**.



- i. Paste the **Token** as shown below and click **Connect**.



- j. The **GitHub** account is linked to the **Jenkins**. Choose the repository from the GitHub account and click Create **Pipeline**.

Which organization does the repository belong to?

oralabuser

Choose a repository

Loaded 5 repositories

Search...

BlueOcean_Pipeline

HelloWorld

JenkinsFiles

PipelineScript

...

Create Pipeline

- k. The Jenkins try to search for the **Jenkinsfile** in the GitHub repository, if not found it creates a new **Jenkinsfile**. As shown below is the Jenkins Pipeline UI.

Jenkins

Pipelines Administration

Logout

BlueOcean_Pipeline

Cancel Save

Pipeline Settings

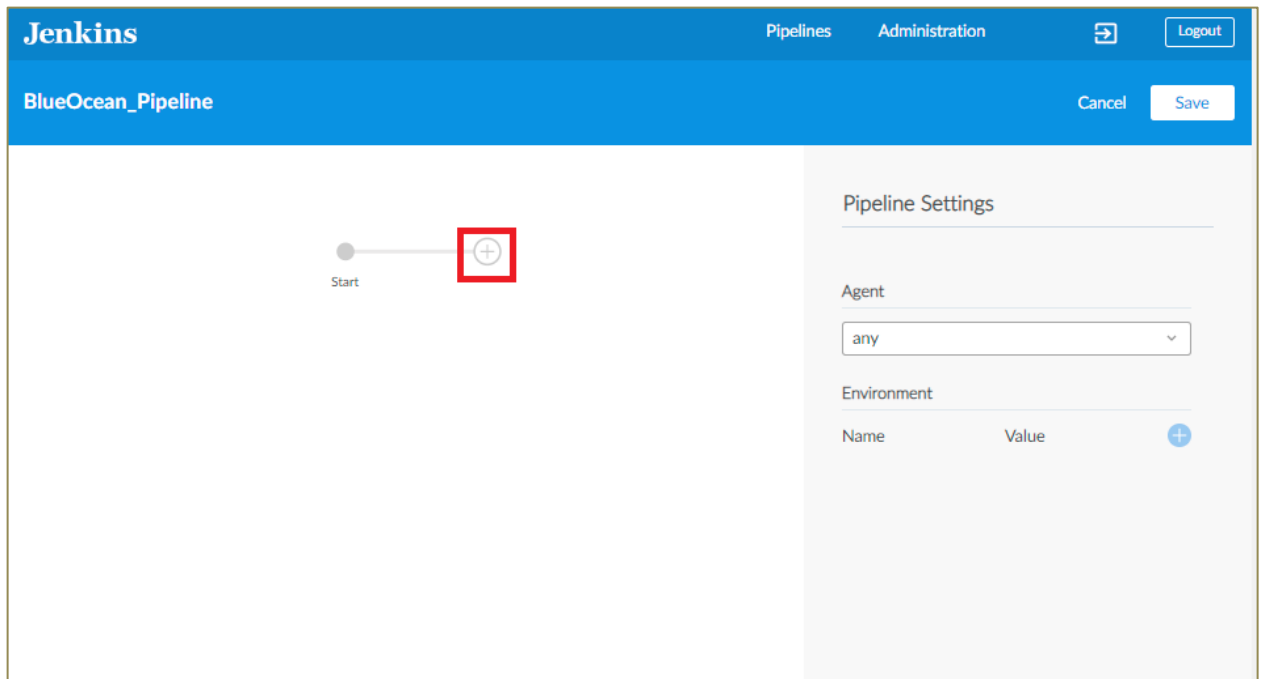
Agent

any

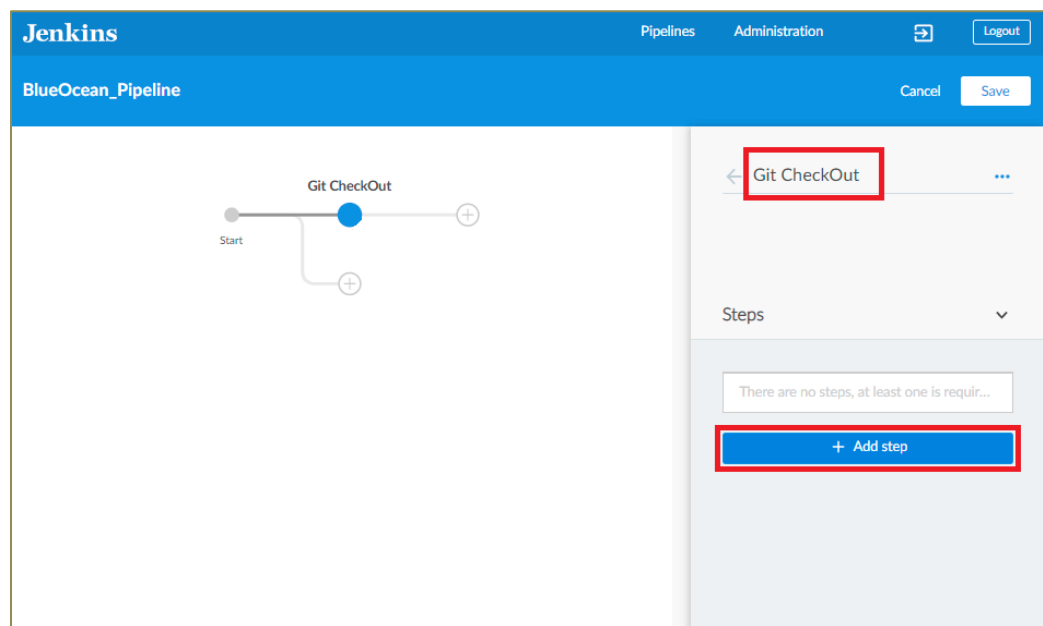
Environment

| Name | Value |
|------|-------|
| | |

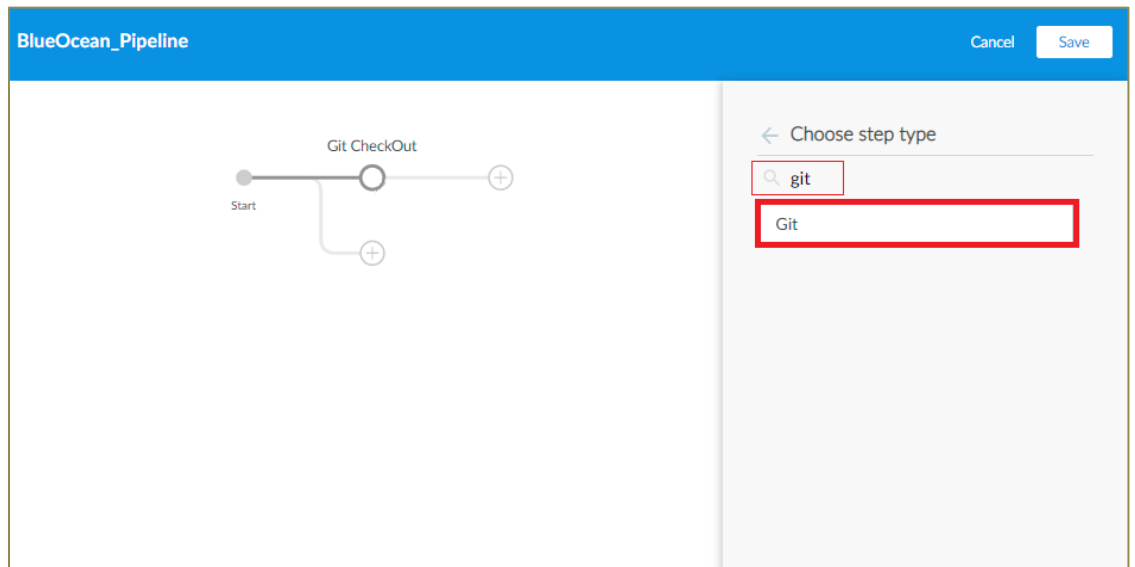
- l. Select the symbol as shown below to add the pipeline step.



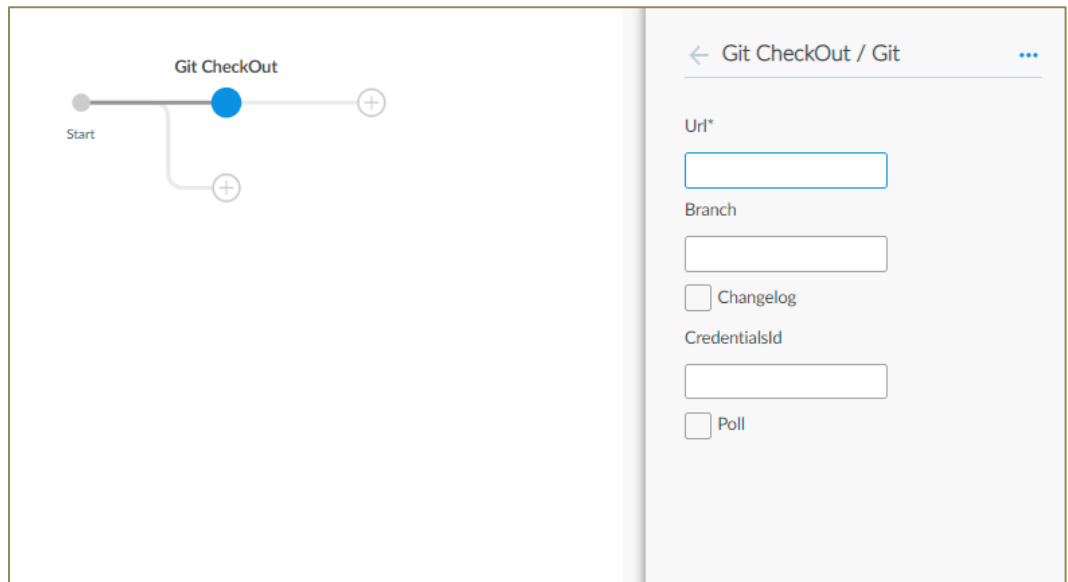
- i) Provide the name for the Stage as shown below and click on **Add Step** to add steps to the Stage.



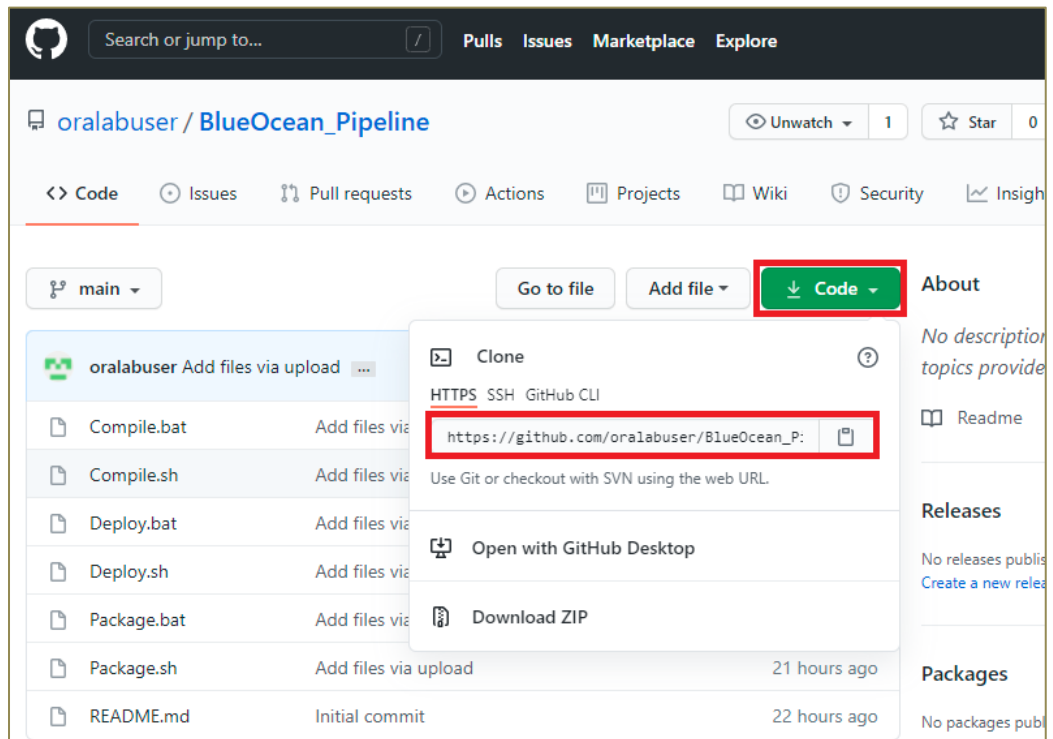
- ii) Search for **Git** as shown below and select the **Git**.



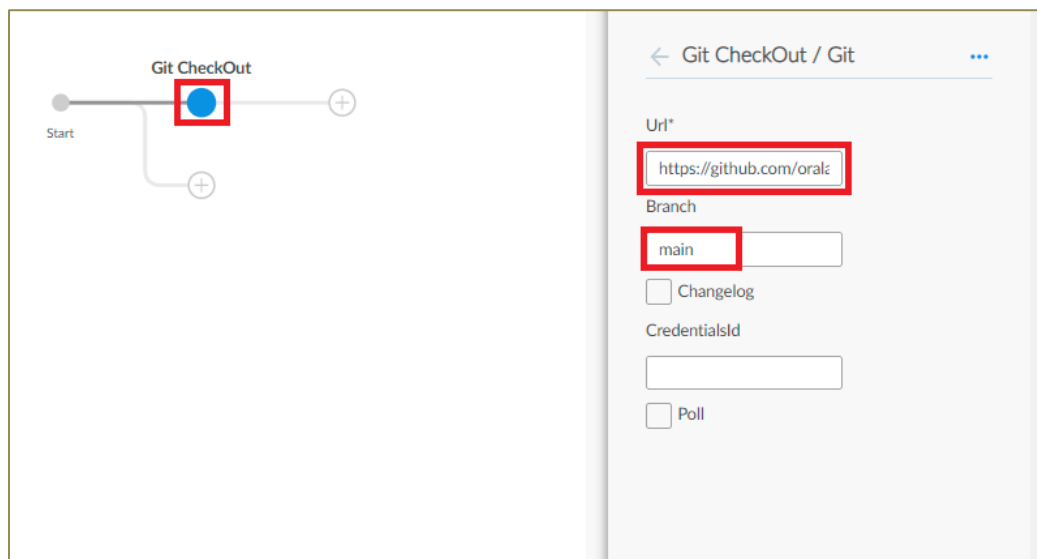
- iii) Provide the **URL** and the Branch name for the respective GitHub repository.



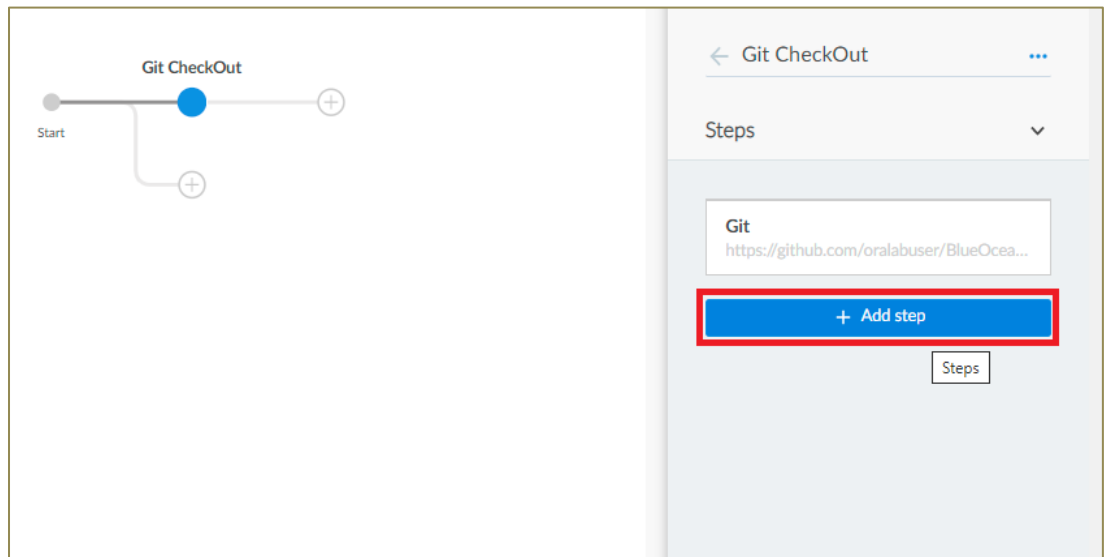
- iv) Open the GitHub repository, click on **Code** and copy the **HTTPS** path as shown below.



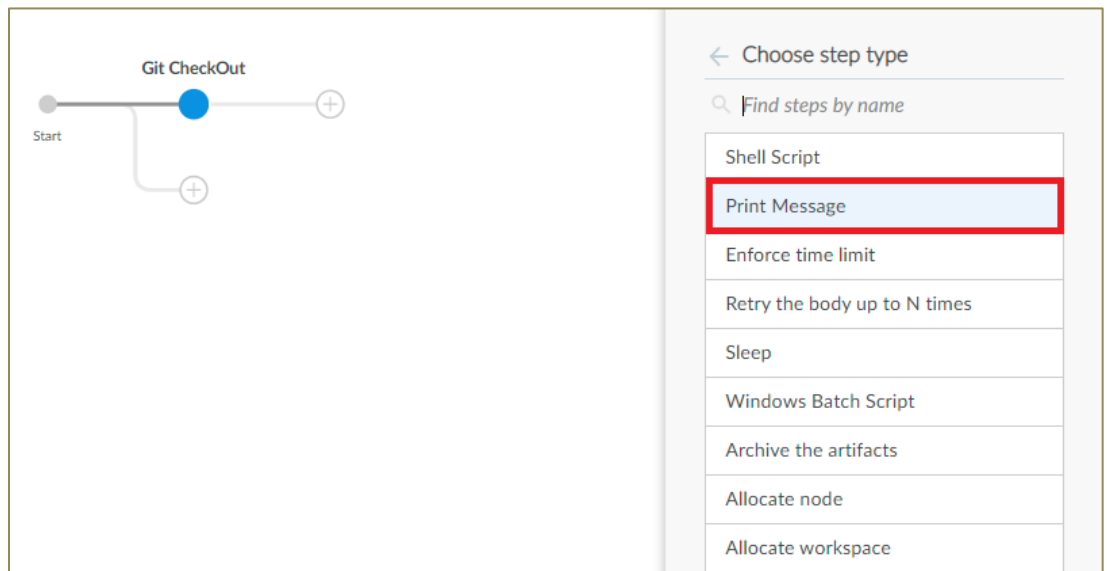
- v) Navigate to Jenkins pipeline, paste the URL and provide **Branch** as **main** as shown below.



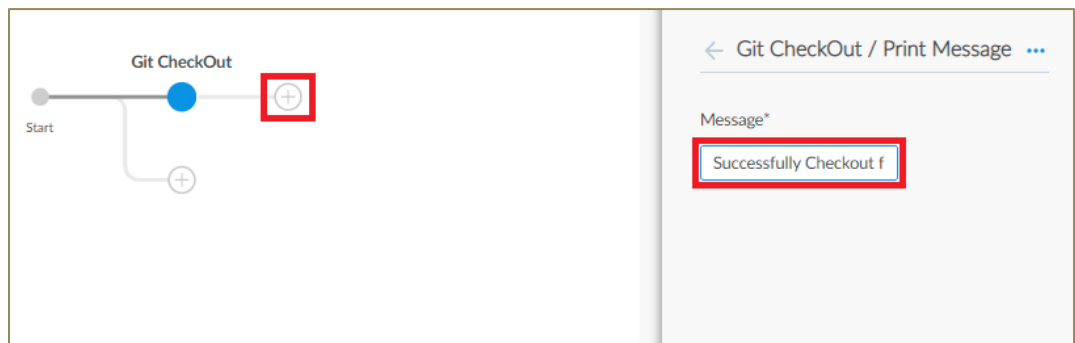
- vi) Click **Add Step** to further add new steps to the Stage as shown below.



vii) Select **Print Message** to display the message while execution.

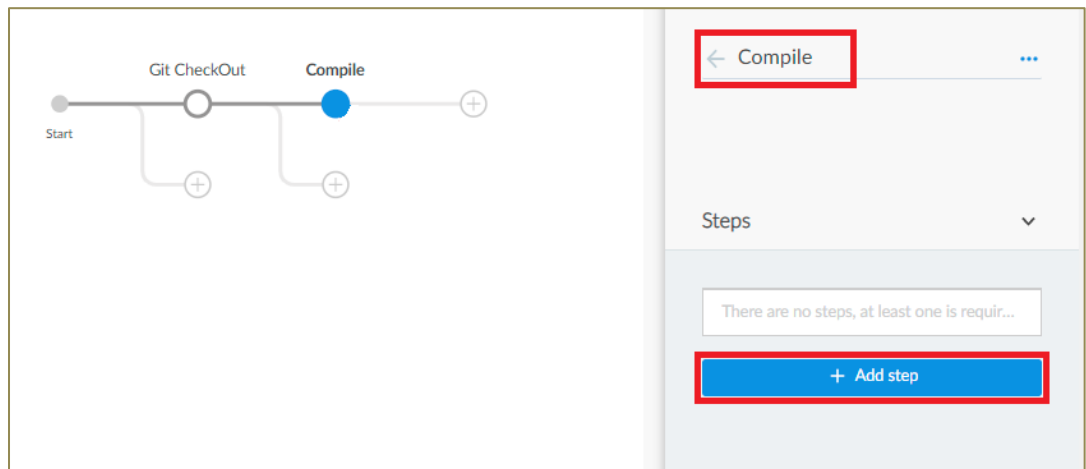


viii) Type **Successfully Checkout from GitHub!!** in the **Message** box as shown below.

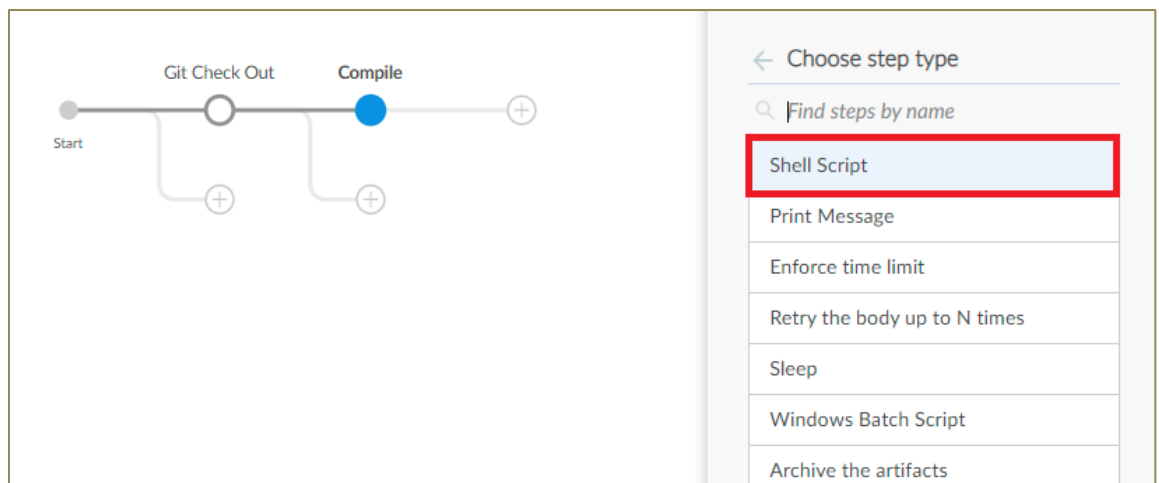


m. Add next Stage **Compile** in the Jenkins pipeline.

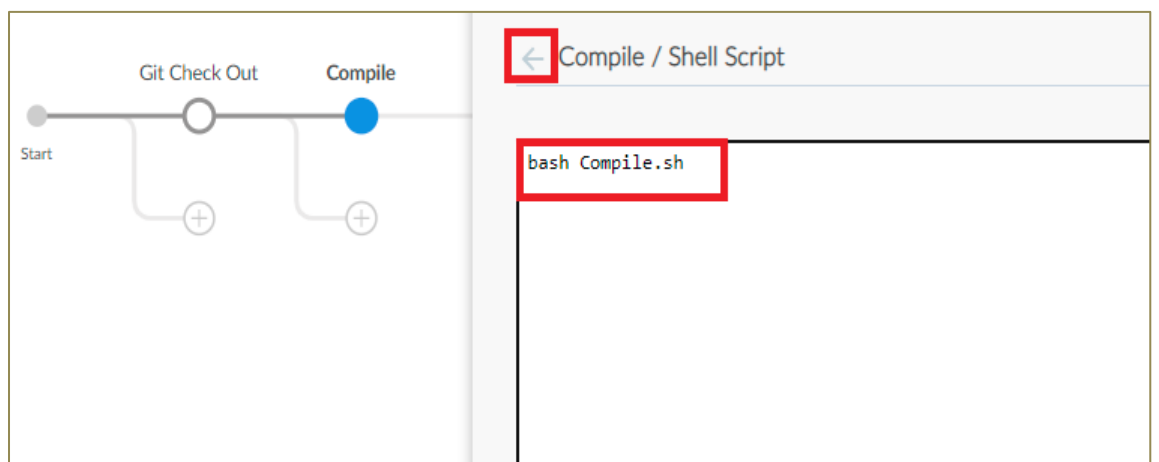
- i) Provide the name for the Stage as shown below and click on **Add Step** to add steps to the Stage.



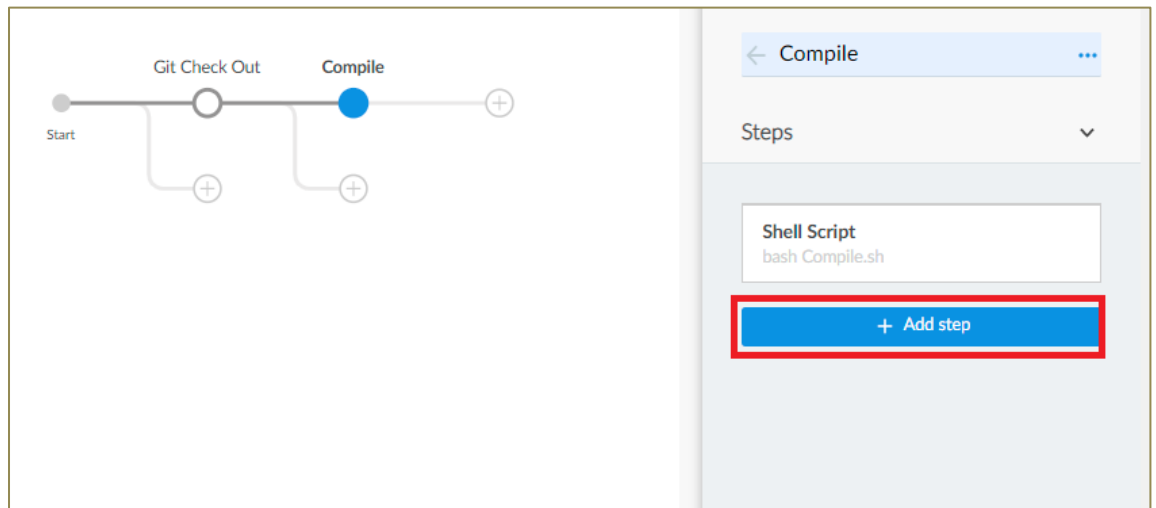
- ii) Choose the step type as **Shell Script** as shown below.



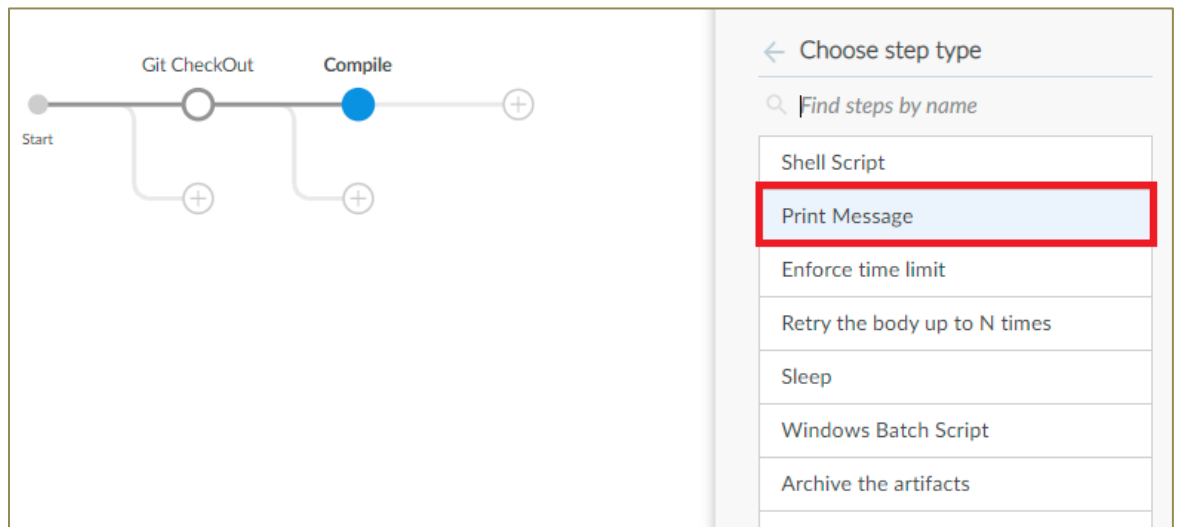
- iii) Type the command in the **Shell Script** to execute the file.



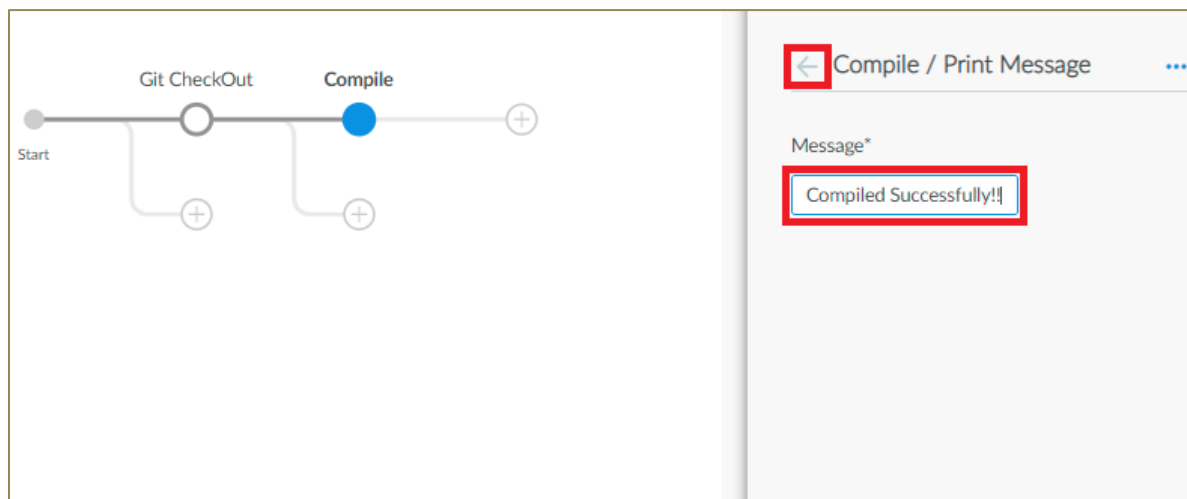
- iv) Click **Add step** to further add the step to the Stage as shown below.



v) Select **Print Message** to display the message while execution.



vi) Type **Compiled Successfully!!** in the **Message** box as shown below.



- i) Verify the Steps added to the **Compile** stage and click on the **+** symbol to add the new stage.

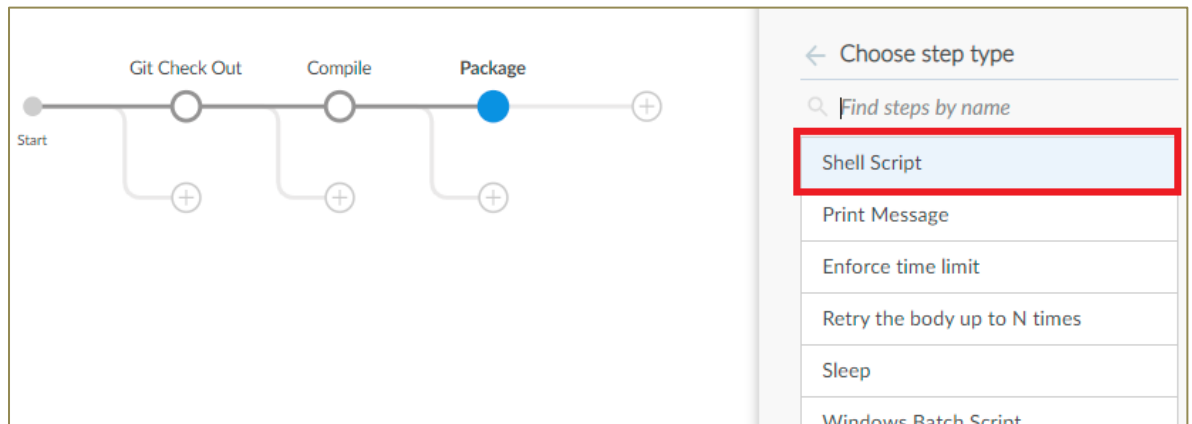
The screenshot shows the Jenkins pipeline configuration interface. On the left, a pipeline graph displays the stages: 'Start' (grey circle), 'Git Check Out' (white circle), 'Compile' (blue circle), and a final stage represented by a white circle with a plus sign. The 'Compile' stage is selected. On the right, the 'Steps' section for the 'Compile' stage is shown. It contains two steps: 'Shell Script' with the command 'bash Compile.sh' and 'Print Message' with the message 'Compiled Successfully!!'. A blue button labeled '+ Add step' is at the bottom of the steps list. A red box highlights the plus sign icon in the pipeline graph and the 'Add step' button.

- n. Add next Stage **Package** in the Jenkins pipeline.

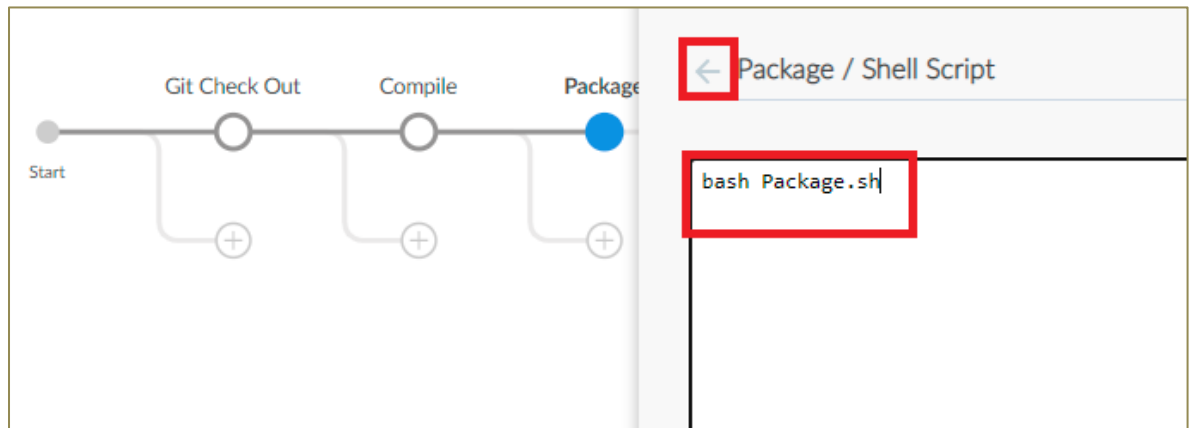
- i) Provide the name for the Stage as shown below and click on **Add Step** to add steps to the Stage.

The screenshot shows the Jenkins pipeline configuration interface. On the left, the pipeline graph now includes the 'Package' stage (blue circle) after the 'Compile' stage. On the right, the 'Steps' section for the 'Package' stage is shown. The stage name 'Package' is entered in the top field. Below it, a message states 'There are no steps, at least one is requir...'. A blue button labeled '+ Add step' is at the bottom. A red box highlights the 'Package' stage name and the '+ Add step' button.

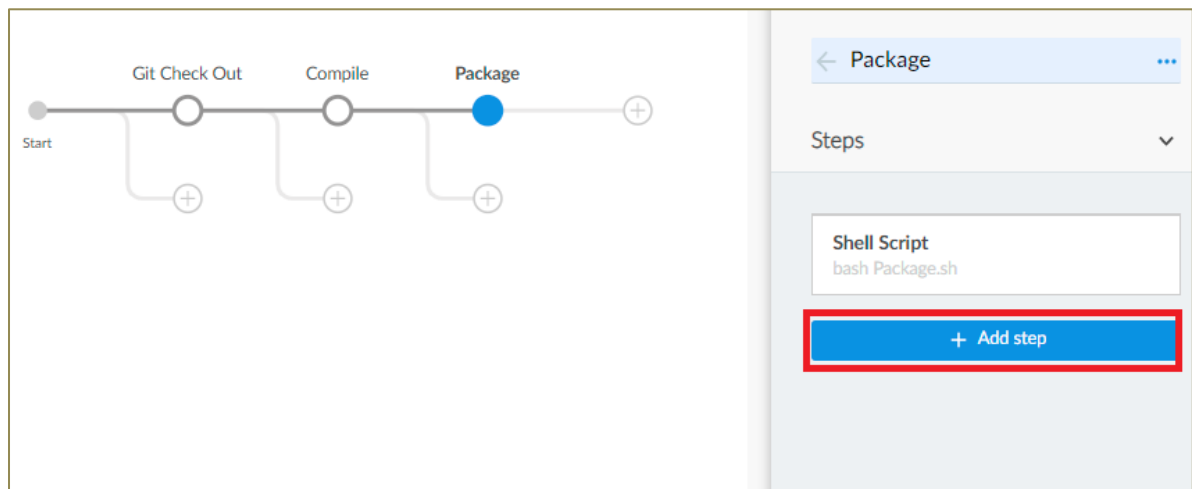
- ii) Choose the step type as **Shell Script** as shown below.



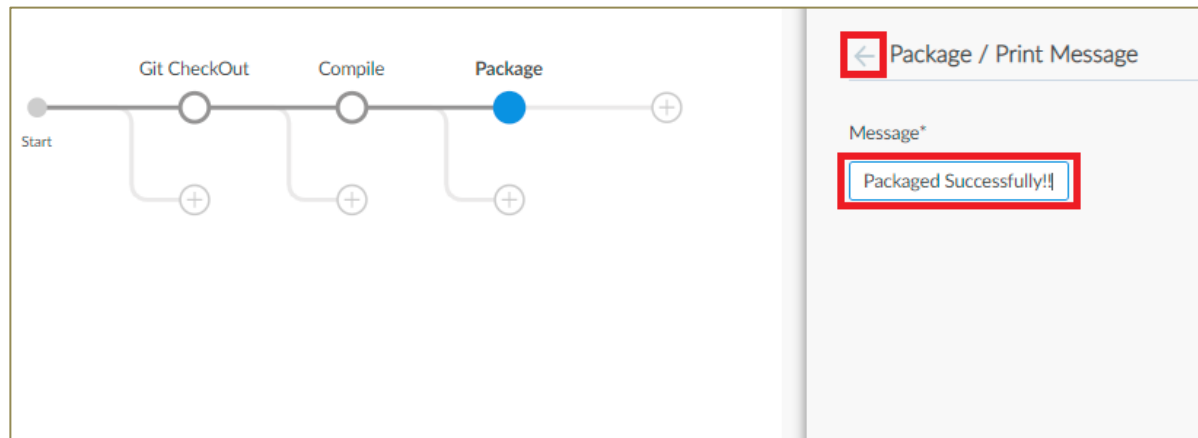
iii) Type the command in the **Shell Script** to execute the file.



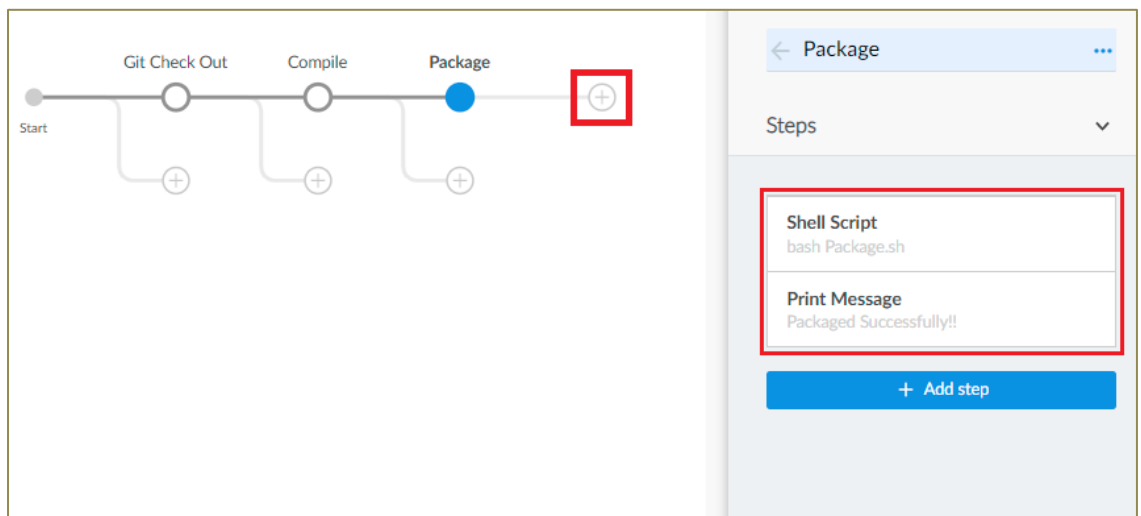
iv) Click **Add step** to add the further step to the Stage as shown below.



v) Select **Print Message** and type **Packaged Successfully!!** in the **Message** box as shown below.

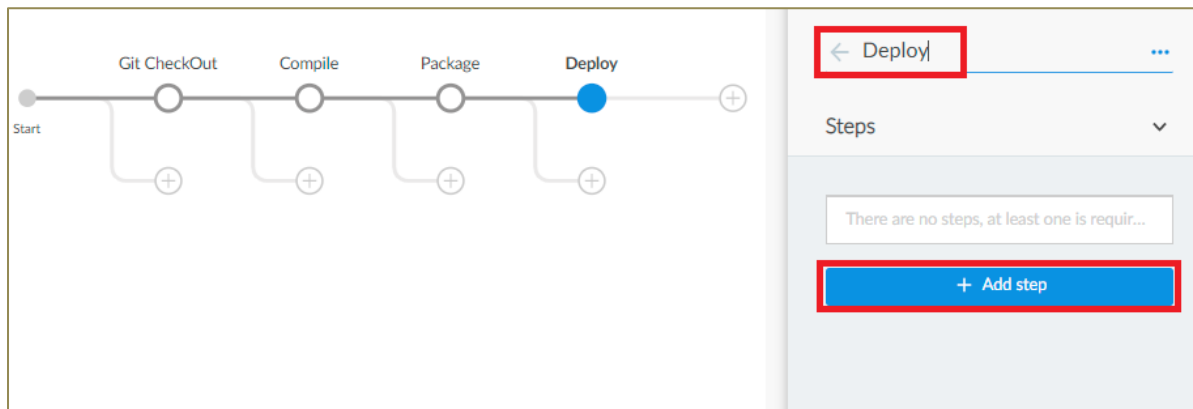


- vi) Verify the Steps added to the **Package** stage and click on the **+** symbol to add the new stage.

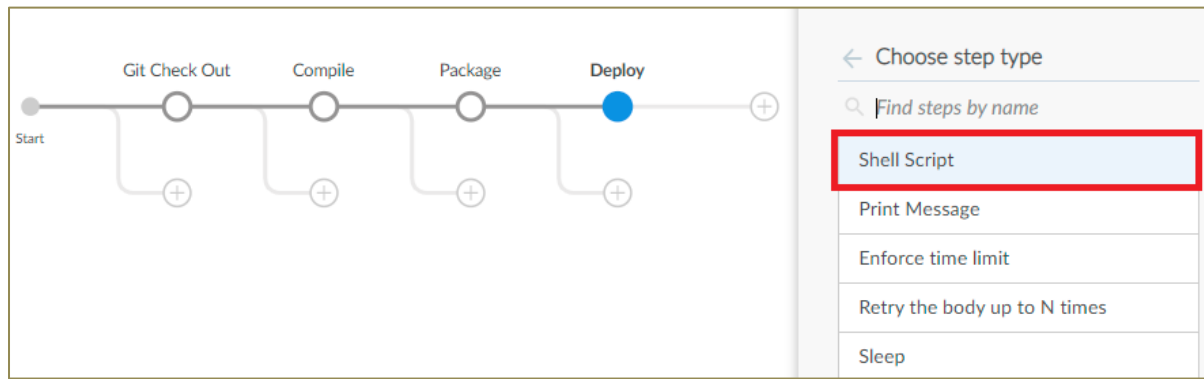


- o. Add next Stage **Deploy** in the Jenkins pipeline.

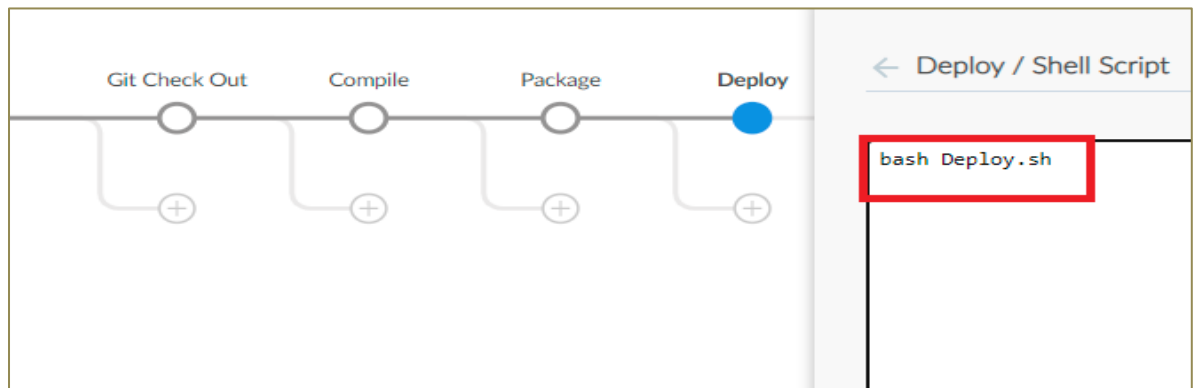
- i) Provide the name for the Stage as shown below and click on **Add Step** to add steps to the Stage.



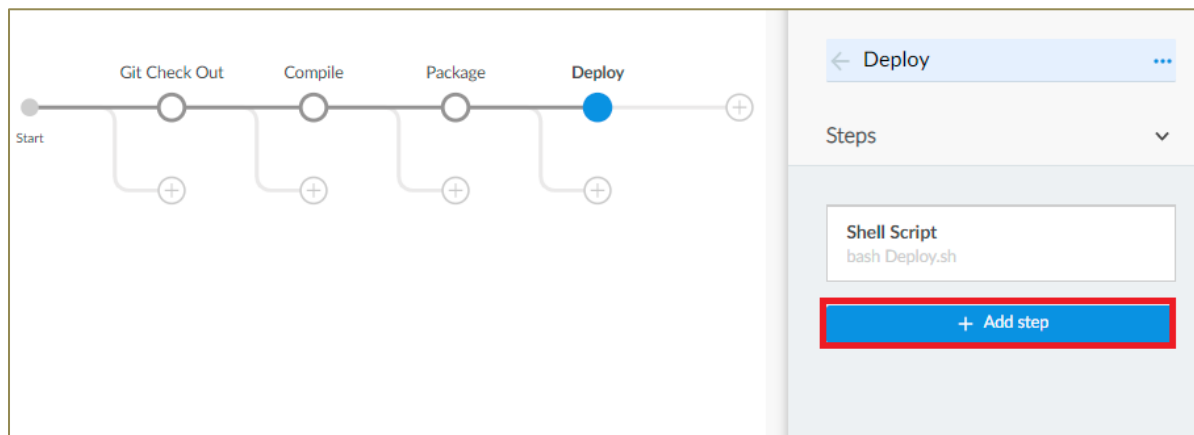
- ii) Choose the step type as **Shell Script** as shown below.



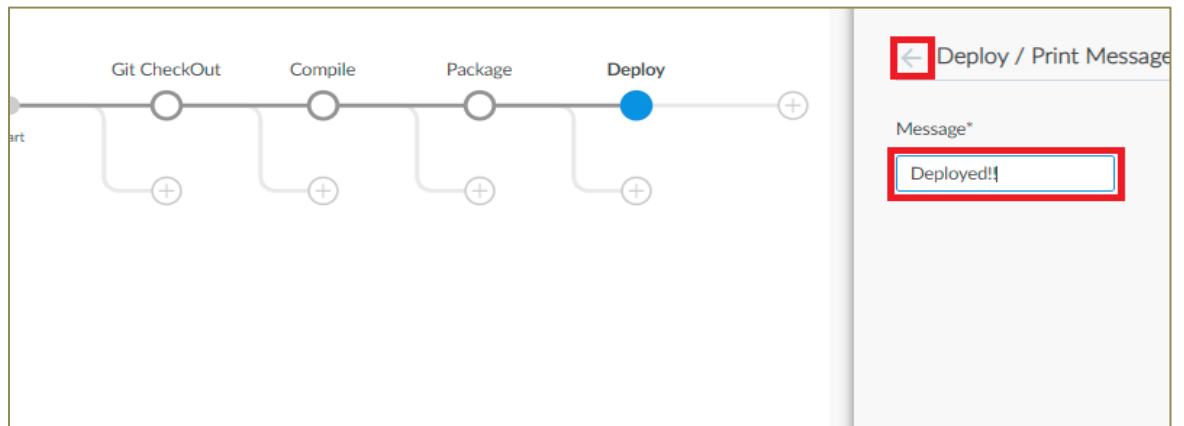
iii) Type the command in the Shell Script to execute the file.



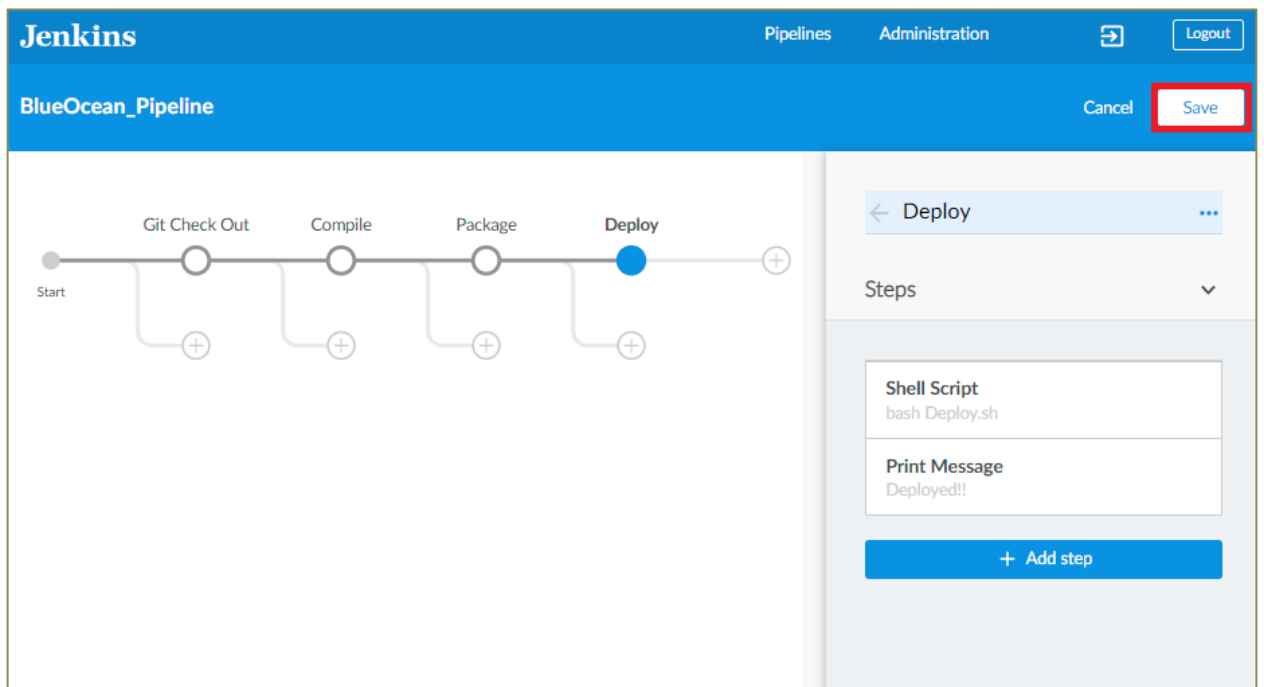
iv) Click **Add step** to add the further step to the Stage as shown below.



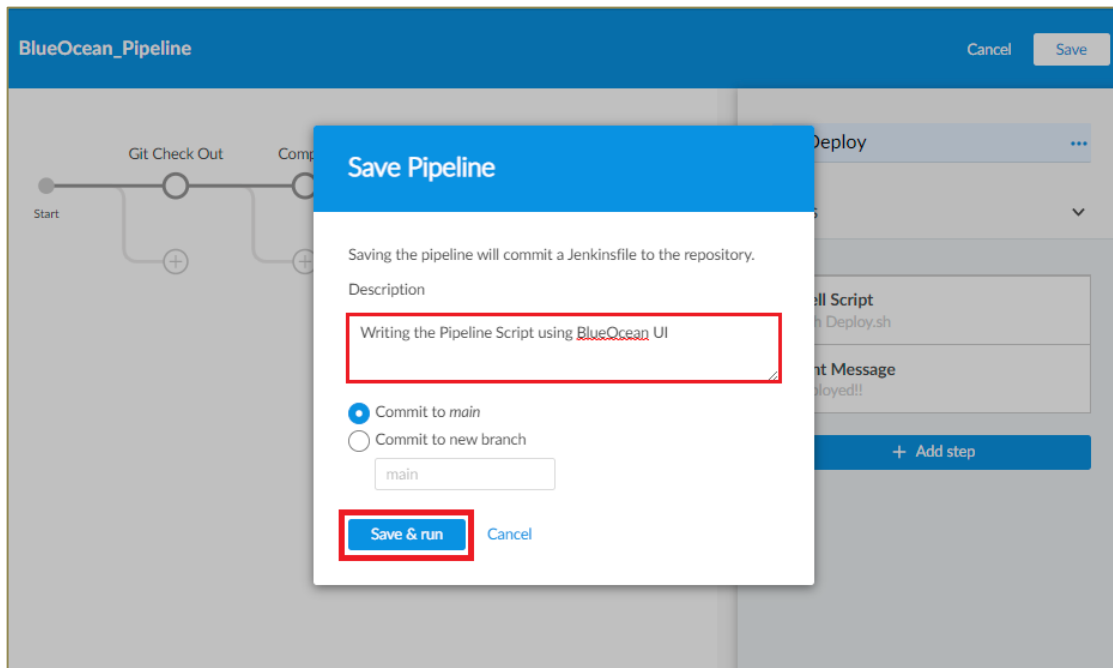
v) Select **Print Message** and type **Deployed!!** in the **Message** box as shown below.



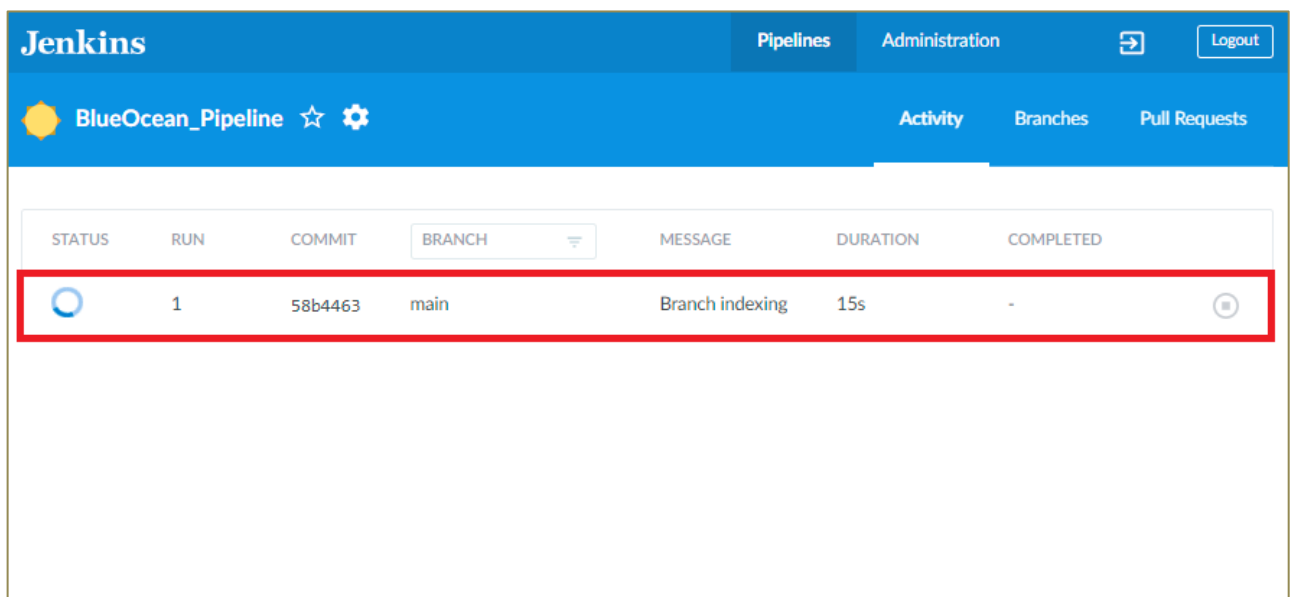
- p. All the stages for the **Pipeline** is added successfully. Click **Save** as shown below.



- q. In the Save Pipeline dialog box type the description “**Writing the Pipeline Script using BlueOcean UI**” and click **Save & run** as shown below.



- r. Verify that the **Pipeline** starts creating as shown below.



- s. Verify that the **Pipeline** is created successfully as shown below and click on the pipeline.

| Jenkins | | | | | | | Pipelines | Administration | Logout |
|------------------------|-----|---------|--------|-----------------|----------|-------------------|-----------|----------------|---------------|
| BlueOcean_Pipeline ☆ ⚙ | | | | | | | Activity | Branches | Pull Requests |
| STATUS | RUN | COMMIT | BRANCH | MESSAGE | DURATION | COMPLETED | | | |
| ✓ | 1 | 58b4463 | main | Branch indexing | 14s | a few seconds ago | | | |

- t. View the **Pipeline** stages which are created and verify the out messages from the stages as shown below.

✓ BlueOcean_Pipeline 1
Pipeline
Changes
Tests
Artifacts
Refresh
Edit
Settings
Share
Logout

Branch: main
14s
No changes
Commit: 58b4463
11 minutes ago
Branch indexing

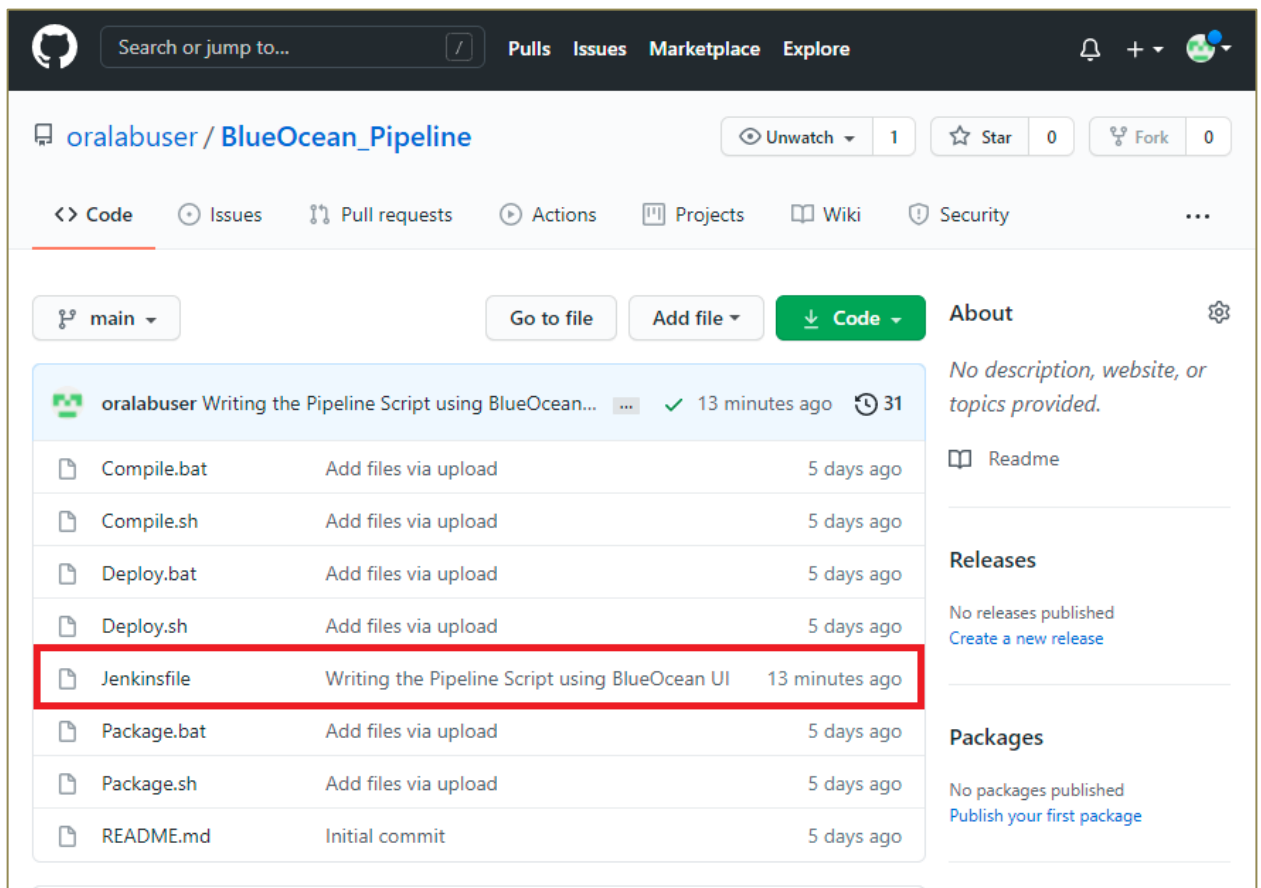
Start
Git Check Out
Compile
Package
Deploy
End

Deploy - <1s
Restart Deploy


> bash Deploy.sh — Shell Script
<1s


> Deployed!!! — Print Message
<1s


- u. Open Jenkins repository and verify the **Jenkinsfile** being created as shown below.
Note: Refresh the page.



- v. Click on the **Jenkinsfile** and view the Groovy script for the Pipeline creation as shown below.

 main ▾ BlueOcean_Pipeline / Jenkinsfile

 oralabuser Writing the Pipeline Script using BlueOcean UI

Latest commit 82f8e76 3 days ago  History

🔍 1 contributor

33 lines (29 sloc) | 570 Bytes

RawBlame🖨️✏️🗑️

```
1 pipeline {
2   agent any
3   stages {
4     stage('Git Check Out') {
5       steps {
6         git(url: 'https://github.com/oralabuser/BlueOcean_Pipeline.git', branch: 'main')
7         echo 'Successfully Checkout!!'
8       }
9     }
10
11    stage('Compile') {
12      steps {
13        sh 'bash Compile.sh'
14        echo 'Compiled Successfully!!'
15      }
16    }
17
18    stage('Package') {
19      steps {
20        sh 'bash Package.sh'
21        echo 'Packaged Successfully!!'
22      }
23    }
24
25    stage('Deploy') {
```

4. Close the terminal, Logout from the AWS Management console and Jenkins Dashboard.