



Name	Java WebApp
URL	https://www.attackdefense.com/challengedetails?cid=2040
Type	DevOps Basics: Continuous Integration

Important Note: This document illustrates all the important steps required to complete this lab. This is by no means a comprehensive step-by-step solution for this exercise. This is only provided as a reference to various commands needed to complete this exercise and for your further research on this topic. Also, note that the IP addresses and domain names might be different in your lab.

Challenge Description

The continuous Integration development process dictates that developers push the code into the master development repository frequently and each (or a small number of) code pushes can trigger the automated build, tests, and deploy the latest build on the test server.

[Jenkins](#) is an open-source automation server that is used widely for continuous integration.

A Jenkins instance and a Gitlab instance are provided to the user. The source code of a Java Webapp is stored on the Gitlab instance.

Objective: Create a Jenkins job to package the web application into a deployable WAR package!

Instructions:

- The GitLab server is reachable with the name 'gitlab'
- Gitlab credentials:

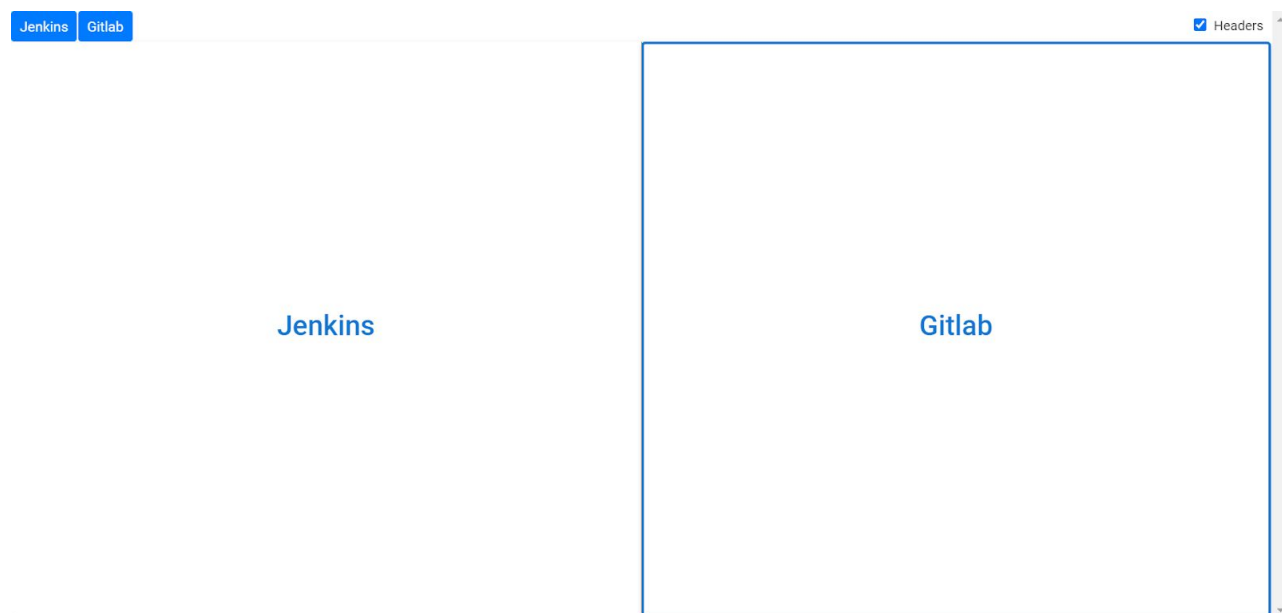
Username	Password
root	welcome123

- The Jenkins server is reachable with the name 'jenkins'
- Jenkins credentials:

Username	Password
admin	welcome123

Lab Setup

On starting the lab, the following interface will be accessible to the user.



On choosing (clicking the text in the center) left left panel, **Jenkins web UI** will open in a new tab



Welcome to Jenkins!

Sign in

☐ Keep me signed in

On selecting the right panel, a web UI of **Gitlab** will open in a new tab.



GitLab Community Edition

Open source software to collaborate on code

Manage Git repositories with fine-grained access controls that keep your code secure. Perform code reviews and enhance collaboration with merge requests. Each project can also have an issue tracker and a wiki.

Sign in	Register
Username or email	
<input type="text"/>	
Password	
<input type="password"/>	
<input type="checkbox"/> Remember me	Forgot your password?
<p>Sign in</p>	

The GitLab instance takes time to function properly. Hence, for some time the following wait page might be visible. This page reloads automatically.

PENTESTER ACADEMY

WebApp will appear once deployed!

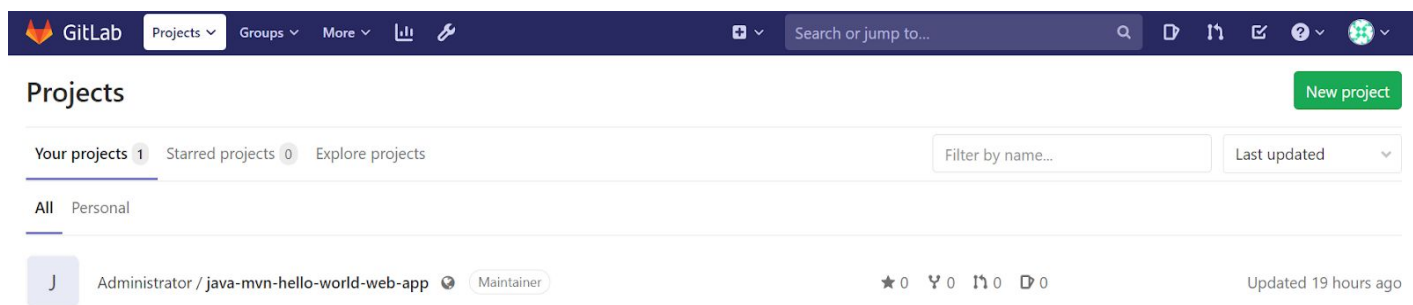
This page refreshes automatically.

Solution

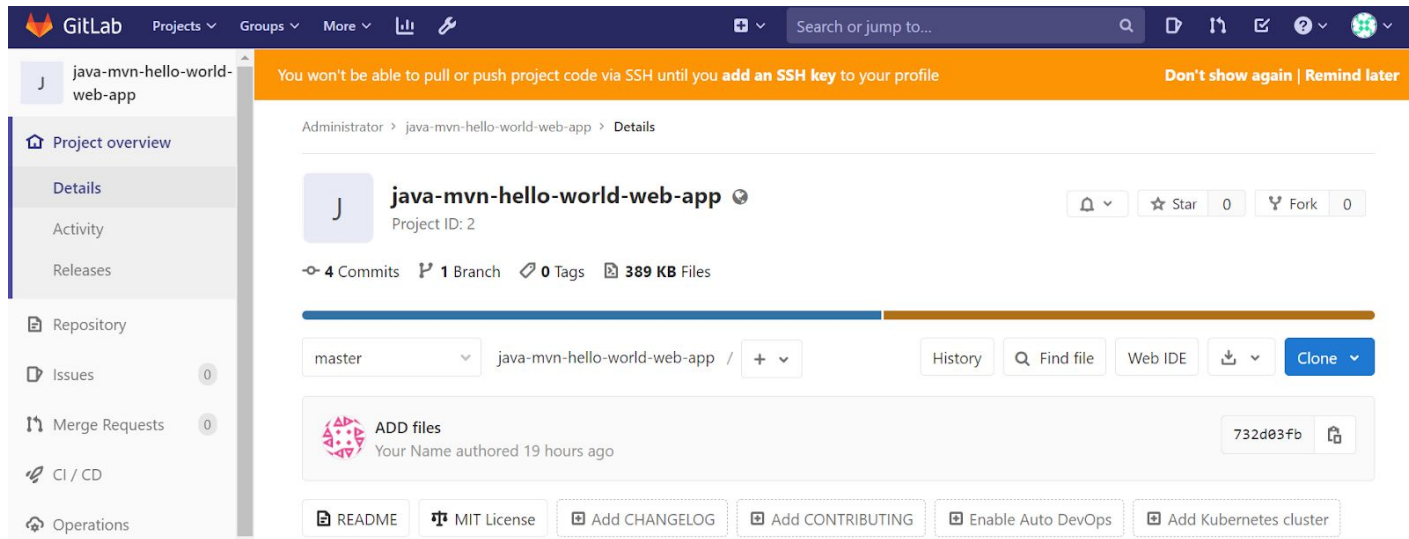
Step 1: Login into GitLab instance using the provided credentials:

Username: root

Password: welcome123



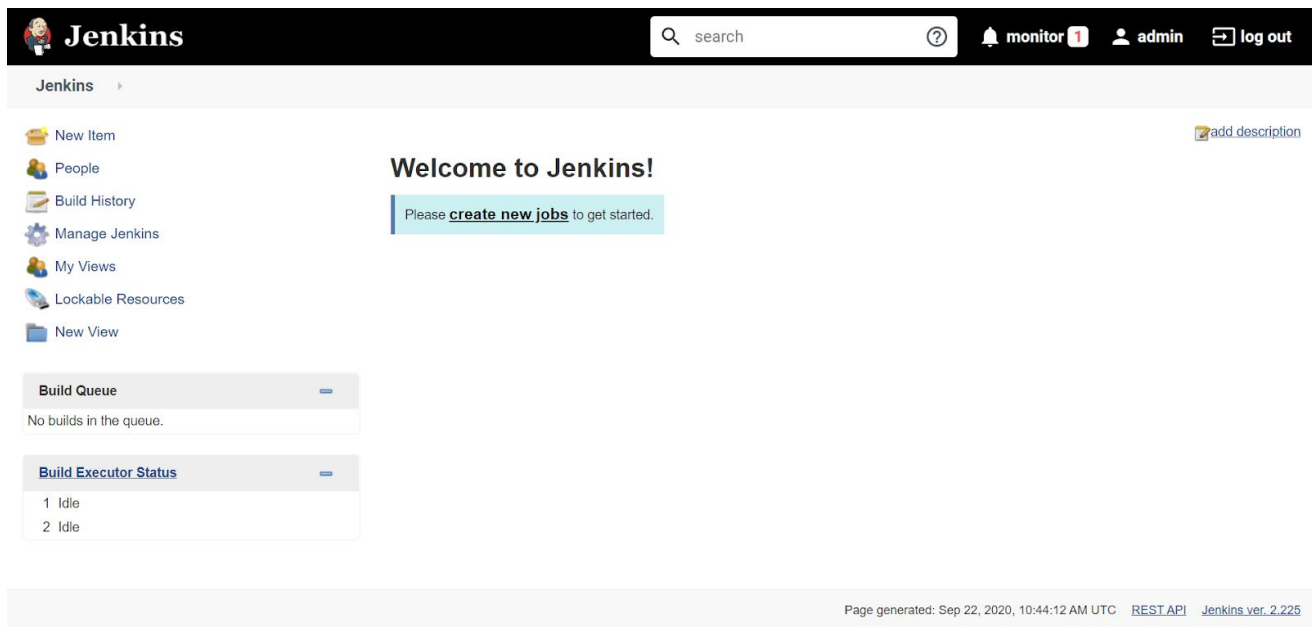
There is a repository present in the Administrator's account. Click on the repository link to open the repository page.



Step 2: Login into Jenkins instance using the provided credentials:

Username: admin

Password: welcome123



There is no job on this Jenkins server.


Step 3: Click on the “create new jobs” link. A job creation wizard will run. Enter the name of the job on the first page. You are free to select any name you like. We are using “java-webapp-build”.

Then select the “Freestyle project” option and click the “OK” button.


Jenkins >

Enter an item name


» Required field

**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**

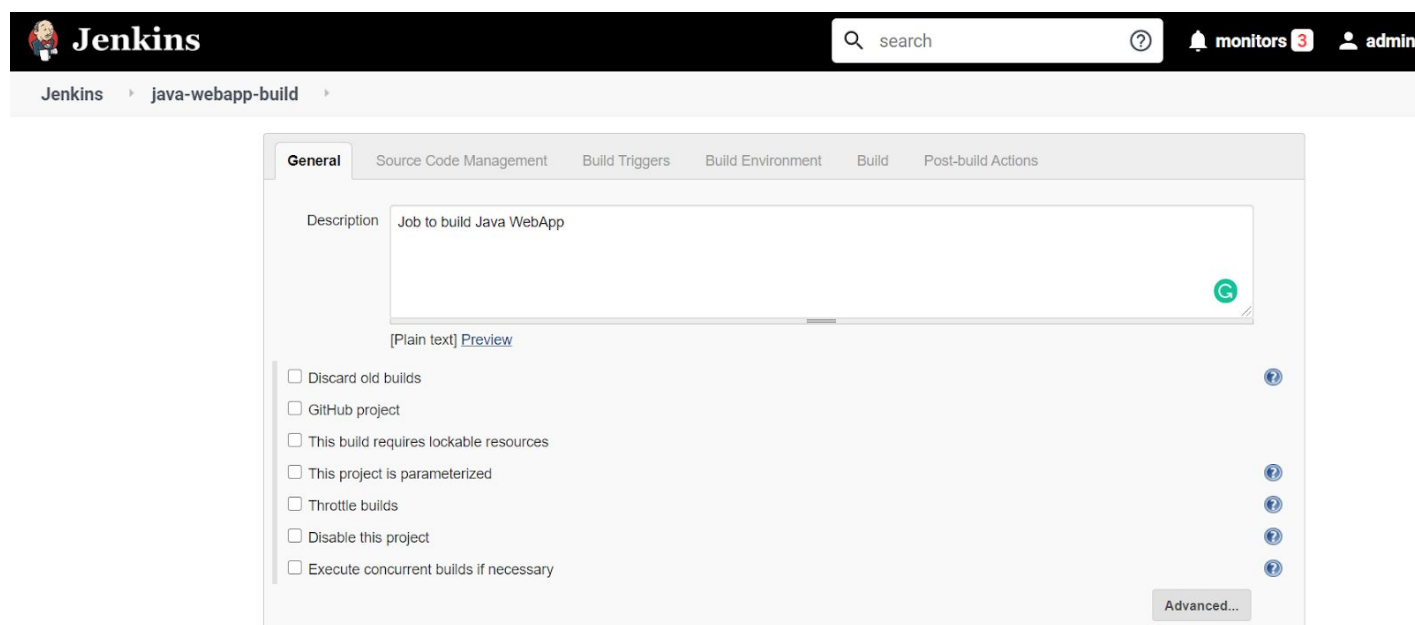
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

**Multibranch Pipeline**

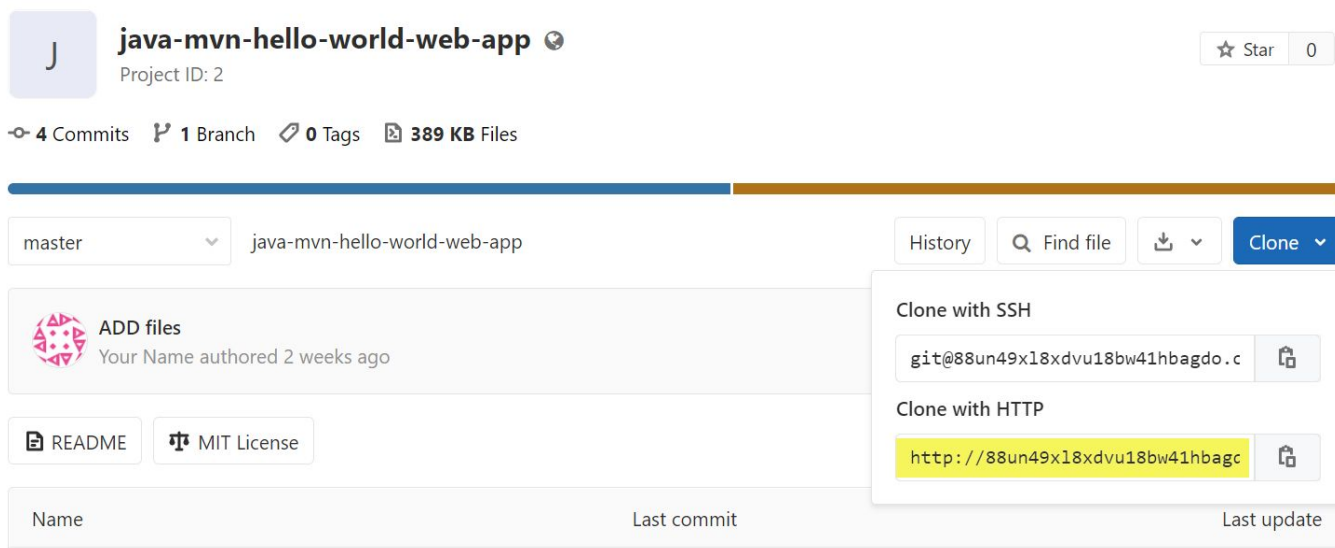
Creates a set of Pipeline projects according to detected branches in one SCM repository.

OK

On the next page, enter some description of the job.



Step 4: Open the Java project repository page on GitLab and copy the “Clone with HTTP” path.



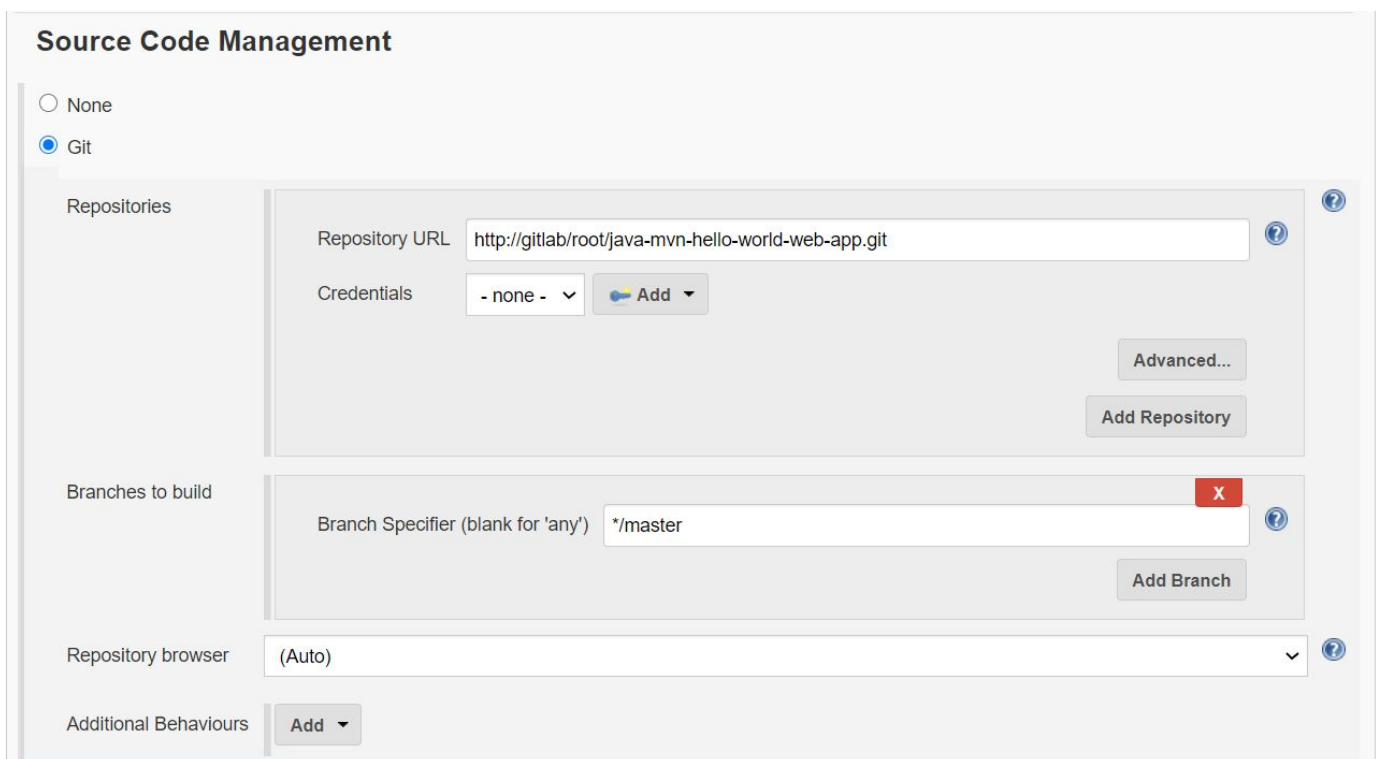
Step 5: Select “Git” in the Source Code Management section and paste this path in the “Repository URL” field.

The copied URL will look something like this:

`http://<random_server_URL>/root/java-mvn-hello-world-web-app.git`

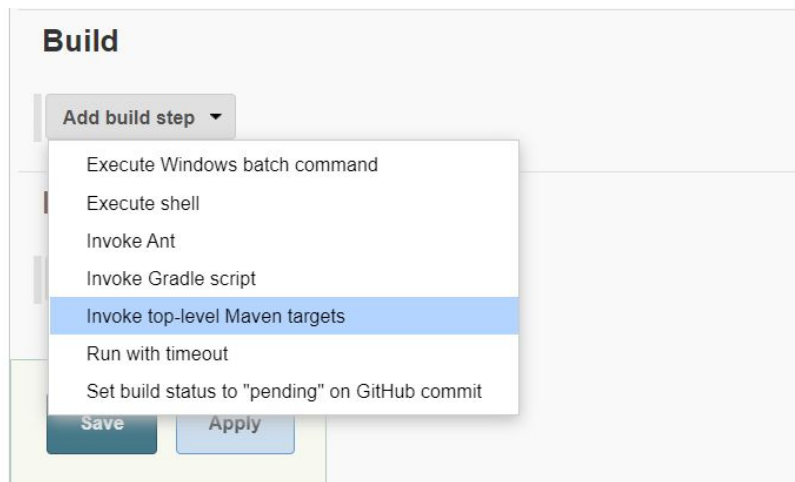
This URL is to make sure that the GitLab files can be accessed through web UI. However, to clone it to the Jenkins server, change it to:

`http://gitlab/root/java-mvn-hello-world-web-app.git`

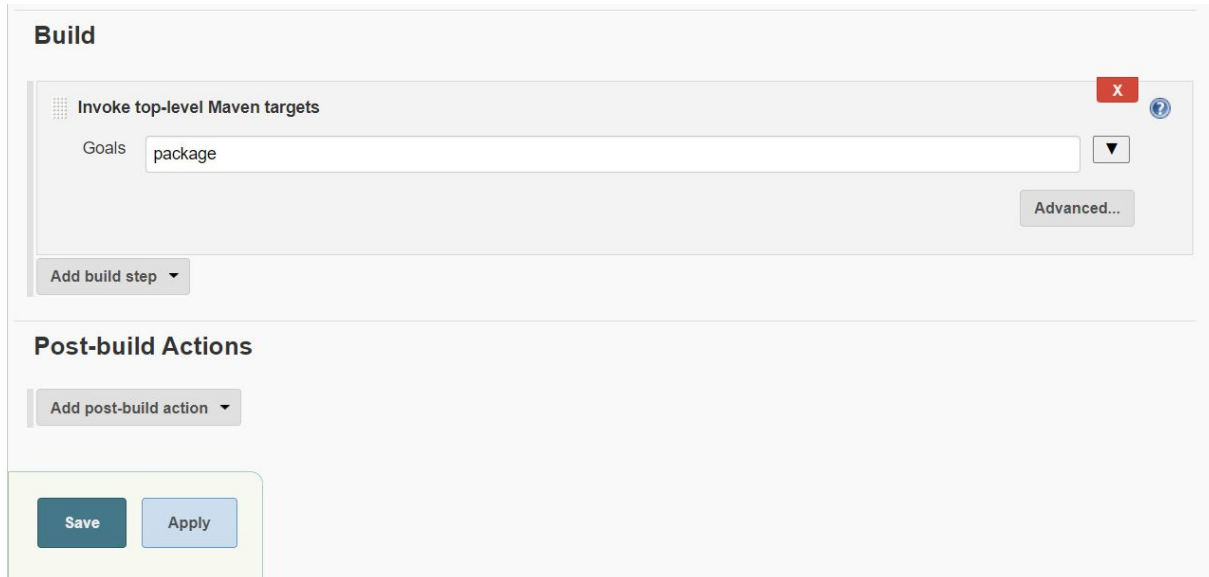


The screenshot shows the 'Source Code Management' configuration page in Jenkins. The 'Git' radio button is selected under the 'Source Code Management' section. The 'Repositories' section contains a 'Repository URL' field with the value 'http://gitlab/root/java-mvn-hello-world-web-app.git' and a 'Credentials' dropdown menu set to '- none -'. There are 'Advanced...' and 'Add Repository' buttons. The 'Branches to build' section has a 'Branch Specifier (blank for \'any\')' field with the value '*/master' and an 'Add Branch' button. The 'Repository browser' dropdown is set to '(Auto)'. At the bottom, there is an 'Additional Behaviours' section with an 'Add' button.

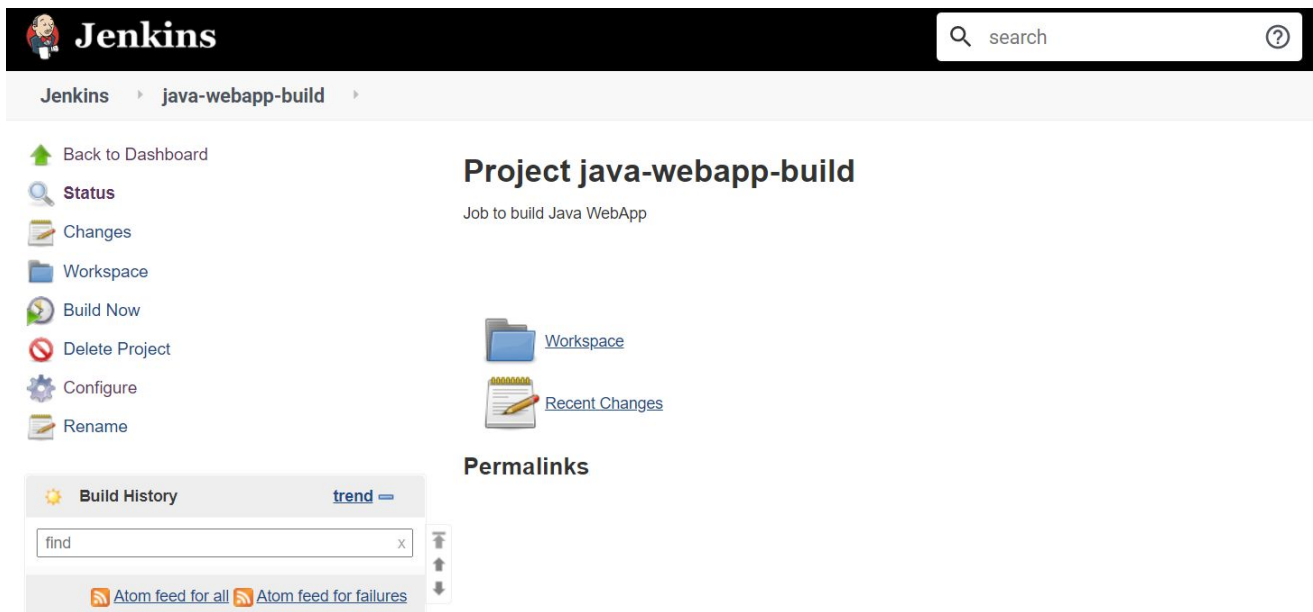
Step 6: In the build section, select “Invoke top-level Maven targets”.



Step 7: In order to create a deployable WAR archive, one has to run “mvn package” command. So, add “package” in the “Goals” field. Save the changes.

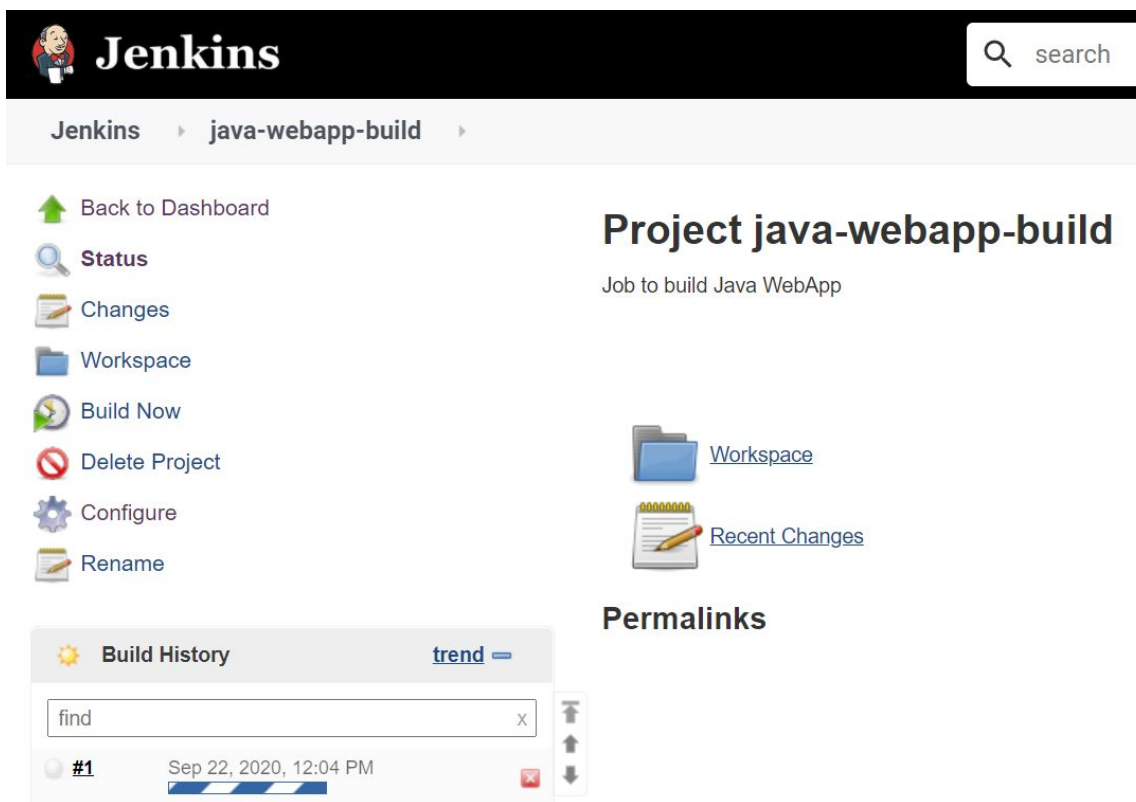


The job will be saved and the page will redirect to the Job page.



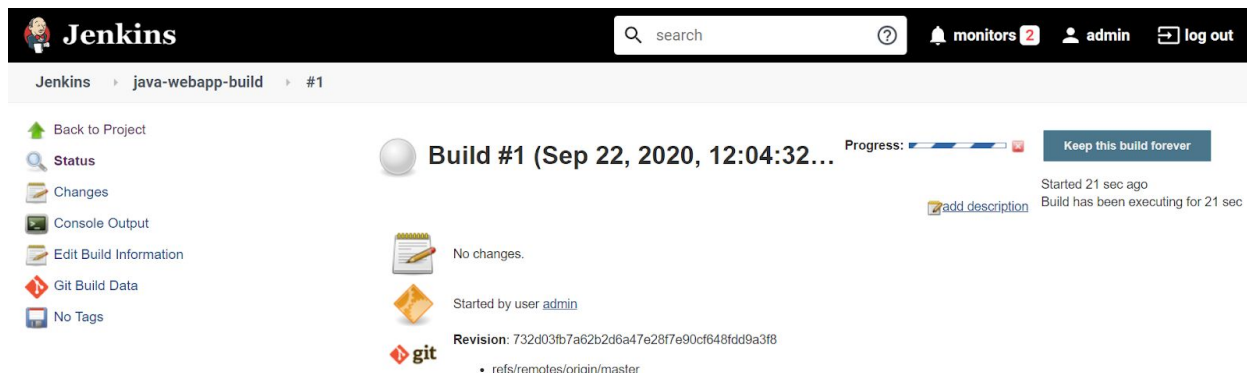
The screenshot shows the Jenkins web interface. At the top is a black header with the Jenkins logo and a search bar. Below the header is a breadcrumb trail: Jenkins > java-webapp-build. On the left is a sidebar menu with icons and labels: Back to Dashboard, Status, Changes, Workspace, Build Now, Delete Project, Configure, and Rename. The main content area has the title "Project java-webapp-build" and the subtitle "Job to build Java WebApp". Below the title are two links: "Workspace" and "Recent Changes". Further down is a "Permalinks" section. At the bottom left is a "Build History" widget with a search bar containing "find", a "trend" link, and two Atom feed links: "Atom feed for all" and "Atom feed for failures".

Step 8: Click “Build now” from the left-hand menu to fire the job.



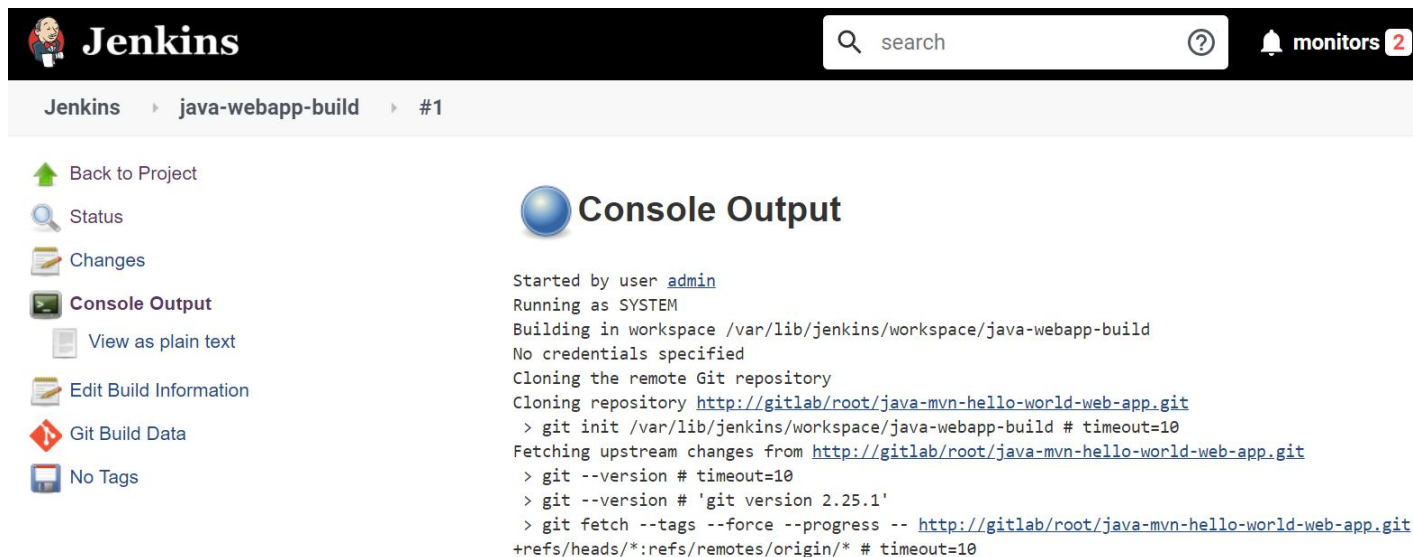
This screenshot shows the same Jenkins interface as the previous one, but after a build has been executed. The "Build History" widget at the bottom left now displays a list of builds. The first build is highlighted with a blue bar and shows the following details: a grey circle icon, the build number "#1", and the timestamp "Sep 22, 2020, 12:04 PM". The rest of the interface, including the sidebar menu and the main content area, remains the same.

An in-progress will appear under the “Build History”. Click on the number #1 to visit the build page.



The screenshot shows the Jenkins interface for a build named "Build #1 (Sep 22, 2020, 12:04:32...)". The top navigation bar includes the Jenkins logo, a search bar, and links for monitors (2), admin, and log out. The breadcrumb trail is "Jenkins > java-webapp-build > #1". On the left sidebar, there are links for "Back to Project", "Status", "Changes", "Console Output", "Edit Build Information", "Git Build Data", and "No Tags". The main content area shows a progress bar, a "Keep this build forever" button, and a timestamp "Started 21 sec ago". Below this, it says "Build has been executing for 21 sec" with an "add description" link. The build status is "No changes." and it was "Started by user admin". The revision is "732d03fb7a62b2d6a47e28f7e90cf648fd9a3f8" from the "refs/remotes/origin/master" branch.

Step 9: Click on “Console Output” to view the build logs.



The screenshot shows the Jenkins interface for the "Console Output" of "Build #1". The top navigation bar is the same as the previous screenshot. The breadcrumb trail is "Jenkins > java-webapp-build > #1". On the left sidebar, the "Console Output" link is highlighted, and there is a "View as plain text" link below it. The main content area has a large "Console Output" header. Below the header, it shows the build logs starting with "Started by user admin" and "Running as SYSTEM". The logs show the build process in the workspace "/var/lib/jenkins/workspace/java-webapp-build", including cloning the repository from "http://gitlab/root/java-mvn-hello-world-web-app.git" and fetching upstream changes.

```

[1;34mINFO[m]
[1;34mINFO[m] [1m--- [0;32mmaven-war-plugin:2.4:war[m [1m(default-war)[m @ [36mmvn-hello-world[0;1m ---[m
[1;34mINFO[m] Packaging webapp
[1;34mINFO[m] Assembling webapp [mvn-hello-world] in [/var/lib/jenkins/workspace/java-webapp-build/target/mvn-hello-world]
[1;34mINFO[m] Processing war project
[1;34mINFO[m] Copying webapp resources [/var/lib/jenkins/workspace/java-webapp-build/src/main/webapp]
[1;34mINFO[m] Webapp assembled in [334 msecs]
[1;34mINFO[m] Building war: /var/lib/jenkins/workspace/java-webapp-build/target/mvn-hello-world.war
[1;34mINFO[m] [1m-----[m
[1;34mINFO[m] [1;32mBUILD SUCCESS[m
[1;34mINFO[m] [1m-----[m
[1;34mINFO[m] Total time: 5.058 s
[1;34mINFO[m] Finished at: 2020-09-22T16:38:10Z
[1;34mINFO[m] [1m-----[m
Finished: SUCCESS

```

The job completed successfully and the WAR archive is available for deployment.

Learning

Building and packaging Java web app using Jenkins.