Experiment 06 - Jenkins Pipeline

Roll No.	37
Name	Mikil Lalwani
Class	D15-B
Subject	DevOps Lab
LO Mapped	LO1: To understand the fundamentals of DevOps engineering and be fully proficient with DevOps terminologies, concepts, benefits, and deployment options to meet your business requirements LO3: To understand the importance of Jenkins to Build and deploy Software Applications on server environment

Aim: To Build the pipeline of jobs using Maven / Gradle / Ant in Jenkins, create a pipeline script to build and deploy an application over the tomcat server.

Introduction:

(Introduction to pipeline and its advantages)

In Jenkins, a pipeline is a collection of events or jobs which are interlinked with one another in a sequence.
It is a combination of plugins that support the integration and implementation of continuous delivery pipelines using Jenkins.
In other words, a Jenkins Pipeline is a collection of jobs or events that brings the software from version control into the hands of the end users by using automation tools. It is used to incorporate continuous delivery in our software development workflow.
A pipeline has an extensible automation server for creating simple or even complex delivery pipelines "as code", via DSL (Domain-specific language).

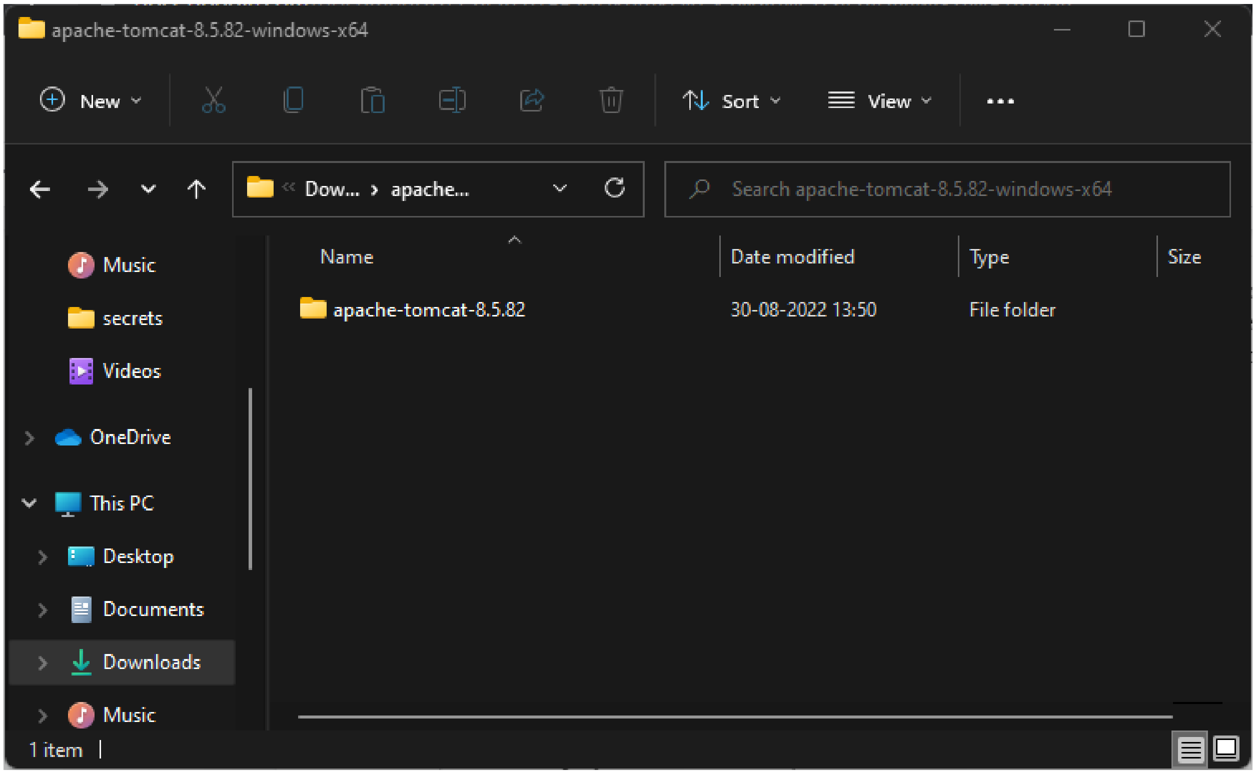
What is a Continuous Delivery Pipeline?
In a Jenkins Pipeline, every job has some sort of dependency on at least one or more jobs or events.
Jenkins Pipeline
The above diagram represents a continuous delivery pipeline in Jenkins. It contains a collection of states such as build, deploy, test and release. These jobs or events are interlinked with each other. Every state has its jobs, which work in a sequence called a continuous delivery pipeline.
A continuous delivery pipeline is an automated expression to show your process for getting software for version control. Thus, every change made in your software goes through a number of complex processes on its way to being released. It also involves developing the software in a repeatable and reliable manner, and progression of the built software through multiple stages of testing and deployment.

Build Tool:

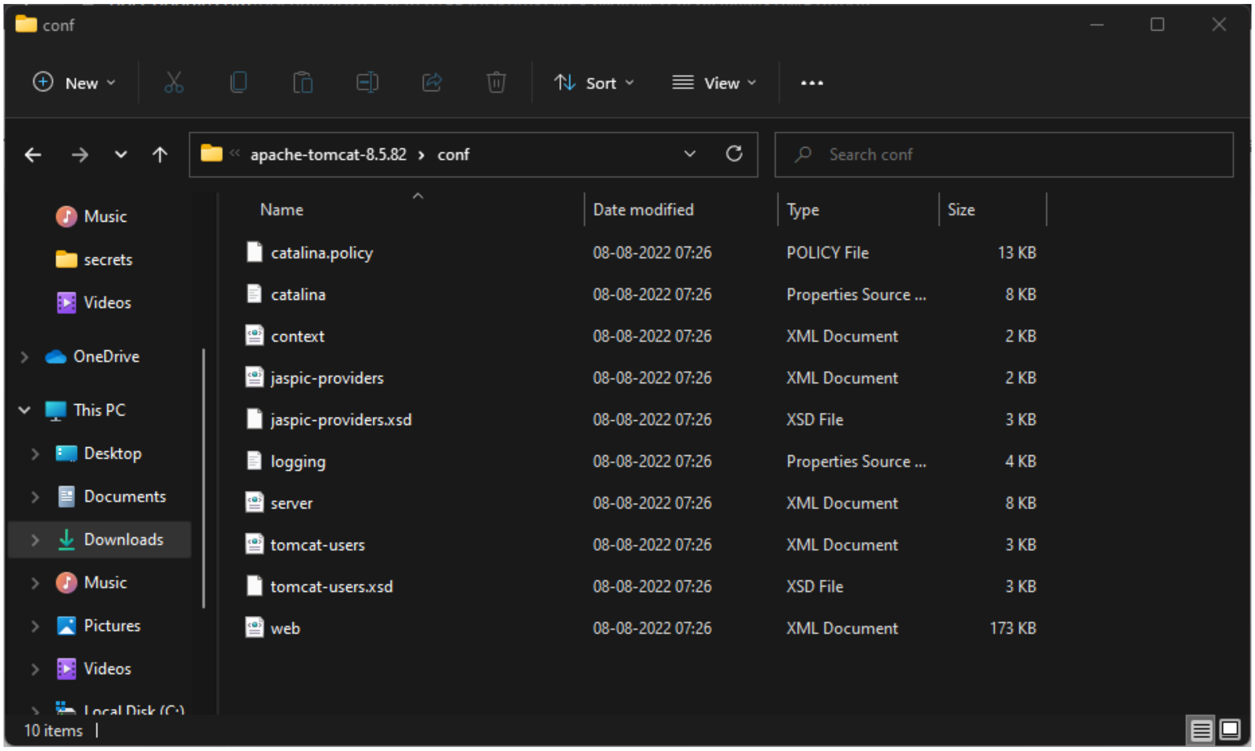
(Explain the build tool used)
Git
Tomcat
Jenkins

Pipeline Jobs:

1. Install Apache Tomcat.

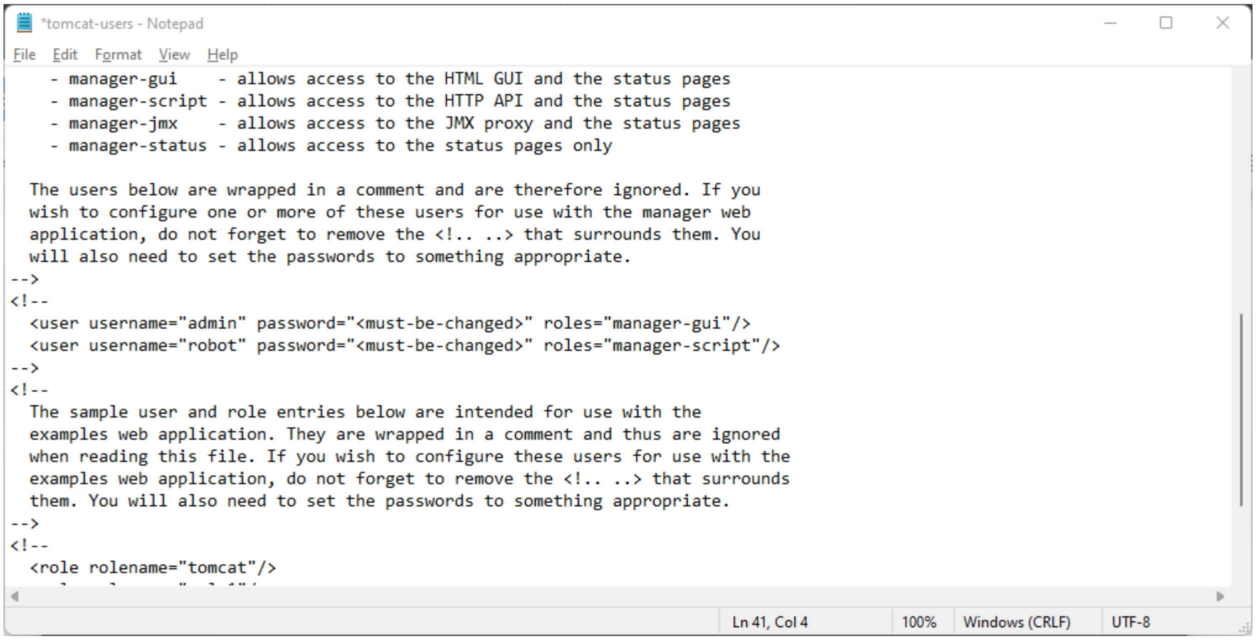


Name	Date modified	Type	Size
apache-tomcat-8.5.82	30-08-2022 13:50	File folder	

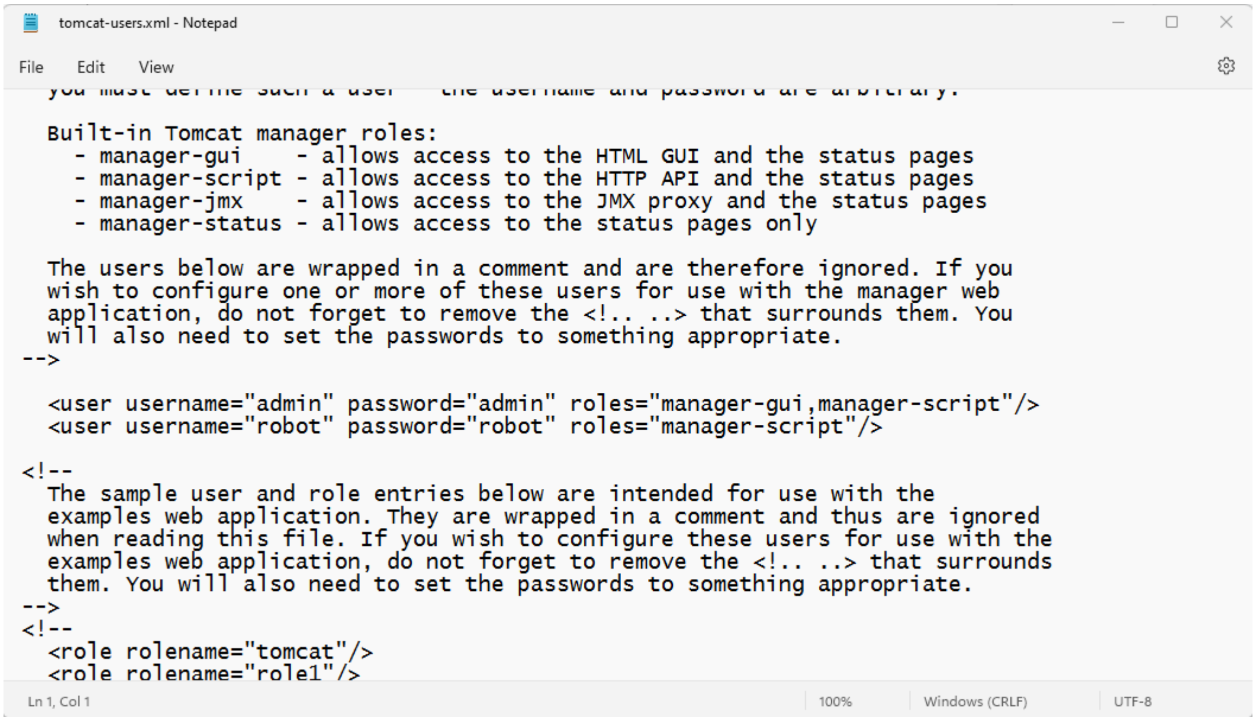


Name	Date modified	Type	Size
catalina.policy	08-08-2022 07:26	POLICY File	13 KB
catalina	08-08-2022 07:26	Properties Source ...	8 KB
context	08-08-2022 07:26	XML Document	2 KB
jaspic-providers	08-08-2022 07:26	XML Document	2 KB
jaspic-providers.xsd	08-08-2022 07:26	XSD File	3 KB
logging	08-08-2022 07:26	Properties Source ...	4 KB
server	08-08-2022 07:26	XML Document	8 KB
tomcat-users	08-08-2022 07:26	XML Document	3 KB
tomcat-users.xsd	08-08-2022 07:26	XSD File	3 KB
web	08-08-2022 07:26	XML Document	173 KB

2. Go to apache-tomcat-10/conf/tomacat-users.xml and uncomment username and set password as your choice.

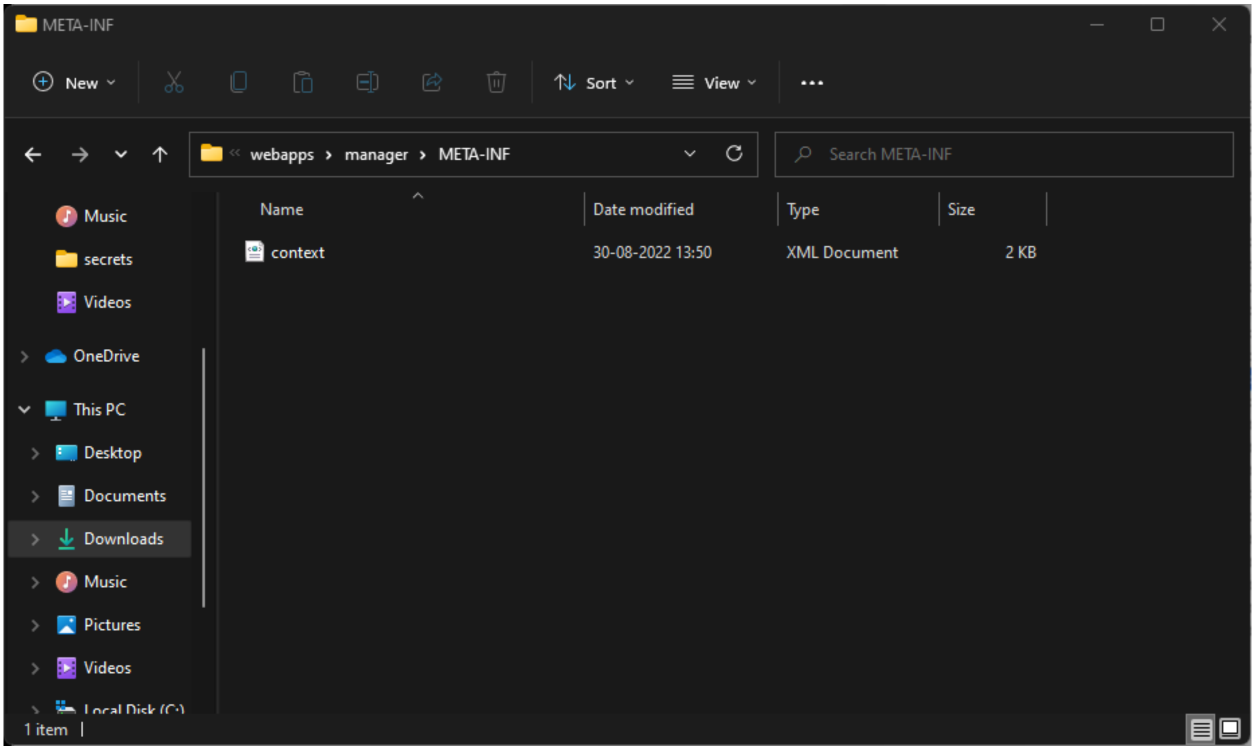


```
    - manager-gui    - allows access to the HTML GUI and the status pages
    - manager-script - allows access to the HTTP API and the status pages
    - manager-jmx    - allows access to the JMX proxy and the status pages
    - manager-status - allows access to the status pages only

  The users below are wrapped in a comment and are therefore ignored. If you
  wish to configure one or more of these users for use with the manager web
  application, do not forget to remove the <!.. ..> that surrounds them. You
  will also need to set the passwords to something appropriate.
-->
<!--
  <user username="admin" password="<must-be-changed>" roles="manager-gui"/>
  <user username="robot" password="<must-be-changed>" roles="manager-script"/>
-->
<!--
  The sample user and role entries below are intended for use with the
  examples web application. They are wrapped in a comment and thus are ignored
  when reading this file. If you wish to configure these users for use with the
  examples web application, do not forget to remove the <!.. ..> that surrounds
  them. You will also need to set the passwords to something appropriate.
-->
<!--
  <role rolename="tomcat"/>
```

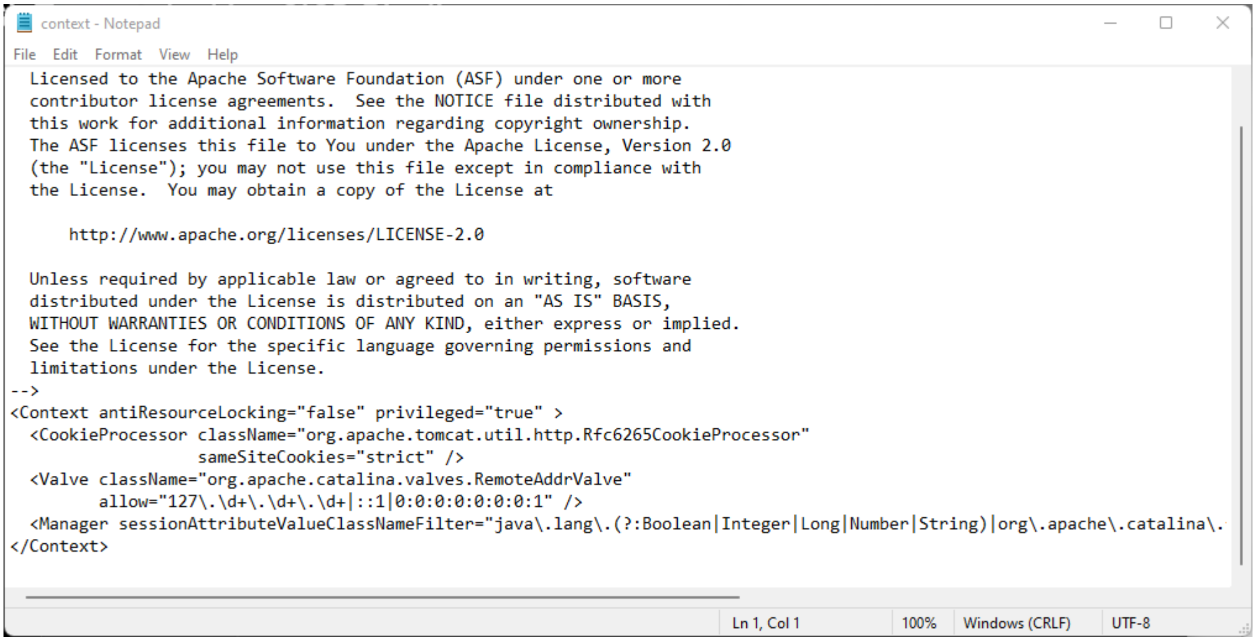


```
  you must define such a user - the username and password are arbitrary.

  Built-in Tomcat manager roles:
    - manager-gui    - allows access to the HTML GUI and the status pages
    - manager-script - allows access to the HTTP API and the status pages
    - manager-jmx    - allows access to the JMX proxy and the status pages
    - manager-status - allows access to the status pages only

  The users below are wrapped in a comment and are therefore ignored. If you
  wish to configure one or more of these users for use with the manager web
  application, do not forget to remove the <!.. ..> that surrounds them. You
  will also need to set the passwords to something appropriate.
-->

  <user username="admin" password="admin" roles="manager-gui,manager-script"/>
  <user username="robot" password="robot" roles="manager-script"/>

<!--
  The sample user and role entries below are intended for use with the
  examples web application. They are wrapped in a comment and thus are ignored
  when reading this file. If you wish to configure these users for use with the
  examples web application, do not forget to remove the <!.. ..> that surrounds
  them. You will also need to set the passwords to something appropriate.
-->
<!--
  <role rolename="tomcat"/>
  <role rolename="role1"/>
```

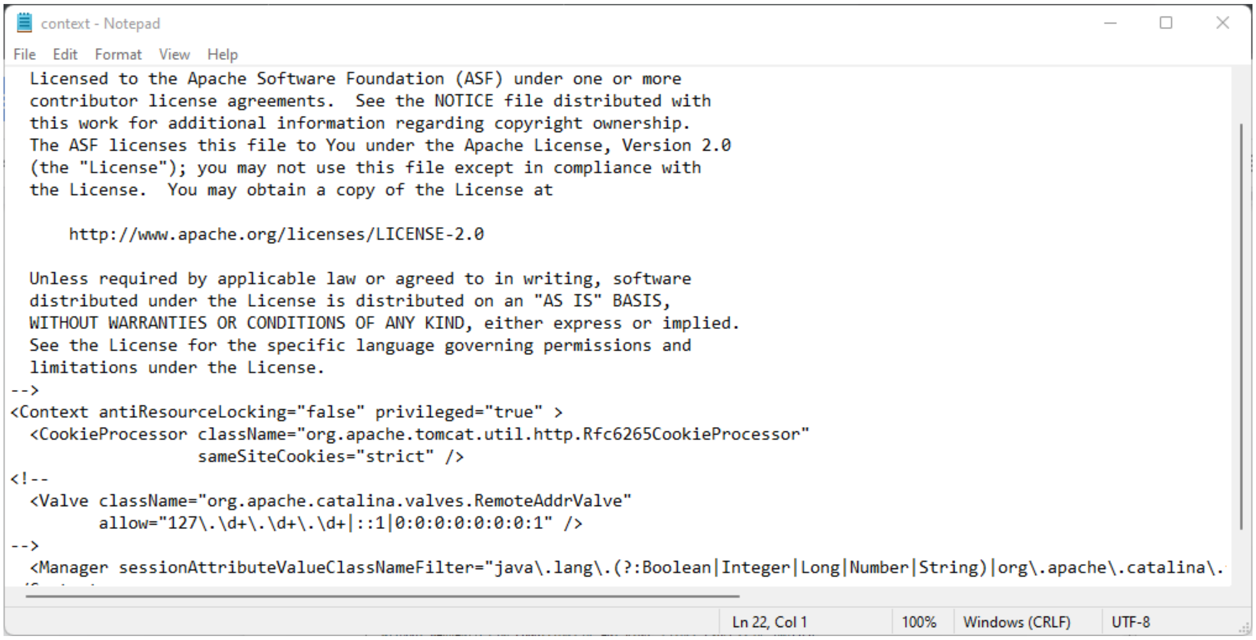
3. Goto apache-tomcat-10/webapps/manager/META-INF/context and comment Valve classname as shown below.



Name	Date modified	Type	Size
context	30-08-2022 13:50	XML Document	2 KB

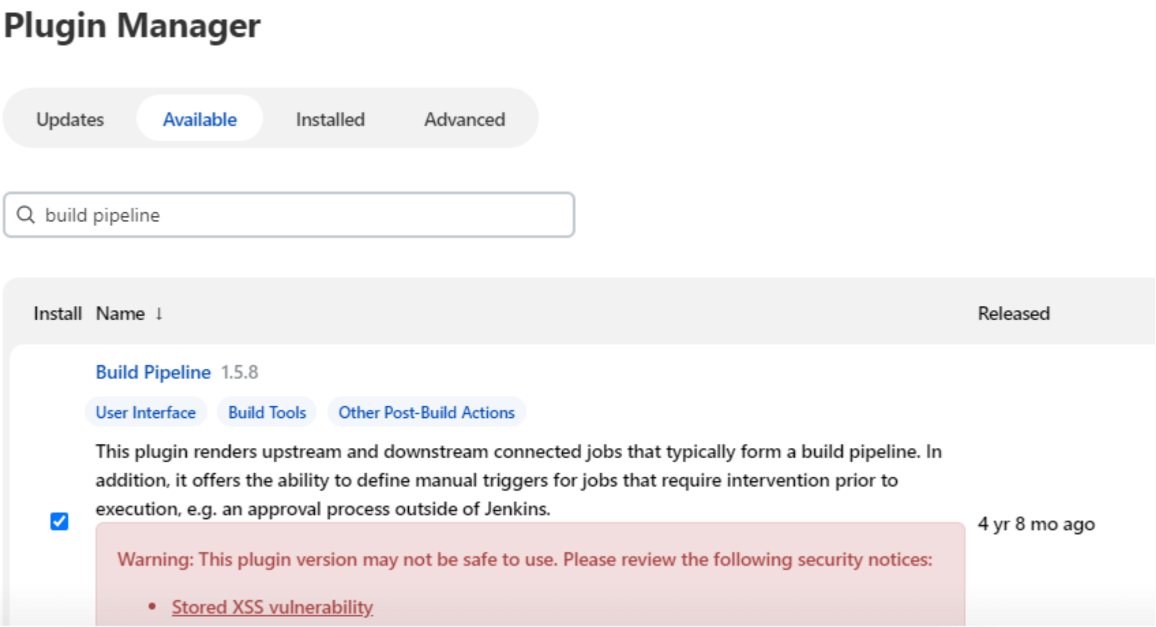


```
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
<Context antiResourceLocking="false" privileged="true" >
  <CookieProcessor className="org.apache.tomcat.util.http.Rfc6265CookieProcessor"
                   sameSiteCookies="strict" />
  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
         allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
  <Manager sessionAttributeValueClassNameFilter="java\.lang\.(?:Boolean|Integer|Long|Number|String)|org\.apache\.catalina\.
</Context>
```



```
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
<Context antiResourceLocking="false" privileged="true" >
  <CookieProcessor className="org.apache.tomcat.util.http.Rfc6265CookieProcessor"
                   sameSiteCookies="strict" />
<!--
  <Valve className="org.apache.catalina.valves.RemoteAddrValve"
         allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />
-->
  <Manager sessionAttributeValueClassNameFilter="java\.lang\.(?:Boolean|Integer|Long|Number|String)|org\.apache\.catalina\.
```

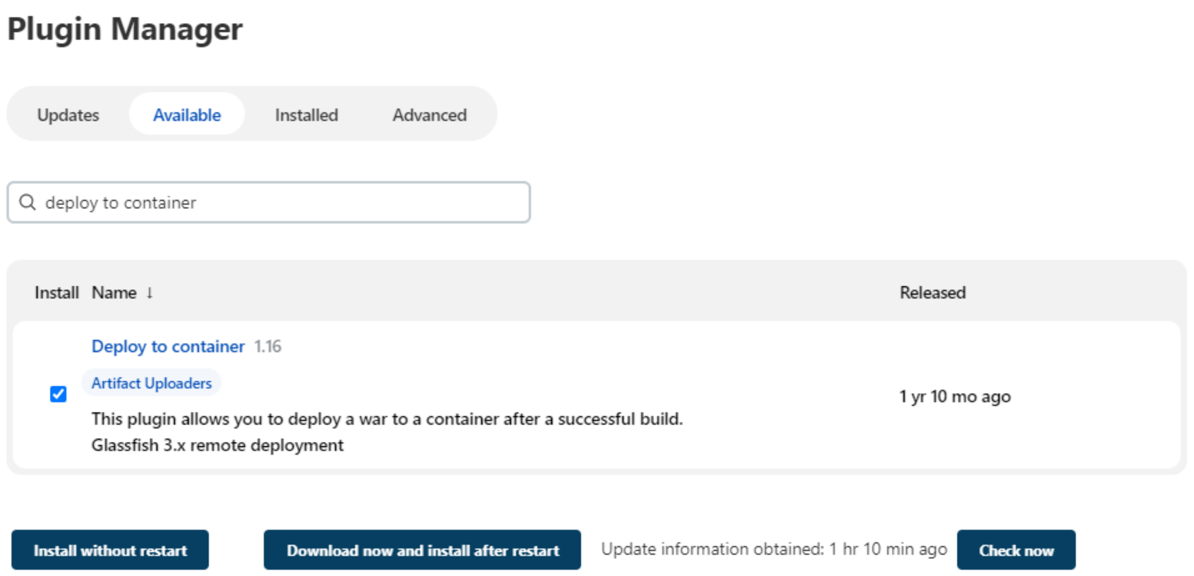
4. Install the build pipeline and deploy to container plugins.



Plugin Manager

build pipeline

Install	Name ↓	Released
☑	**Build Pipeline** 1.5.8 User Interface · Build Tools · Other Post-Build Actions This plugin renders upstream and downstream connected jobs that typically form a build pipeline. In addition, it offers the ability to define manual triggers for jobs that require intervention prior to execution, e.g. an approval process outside of Jenkins. Warning: This plugin version may not be safe to use. Please review the following security notices: • Stored XSS vulnerability	4 yr 8 mo ago



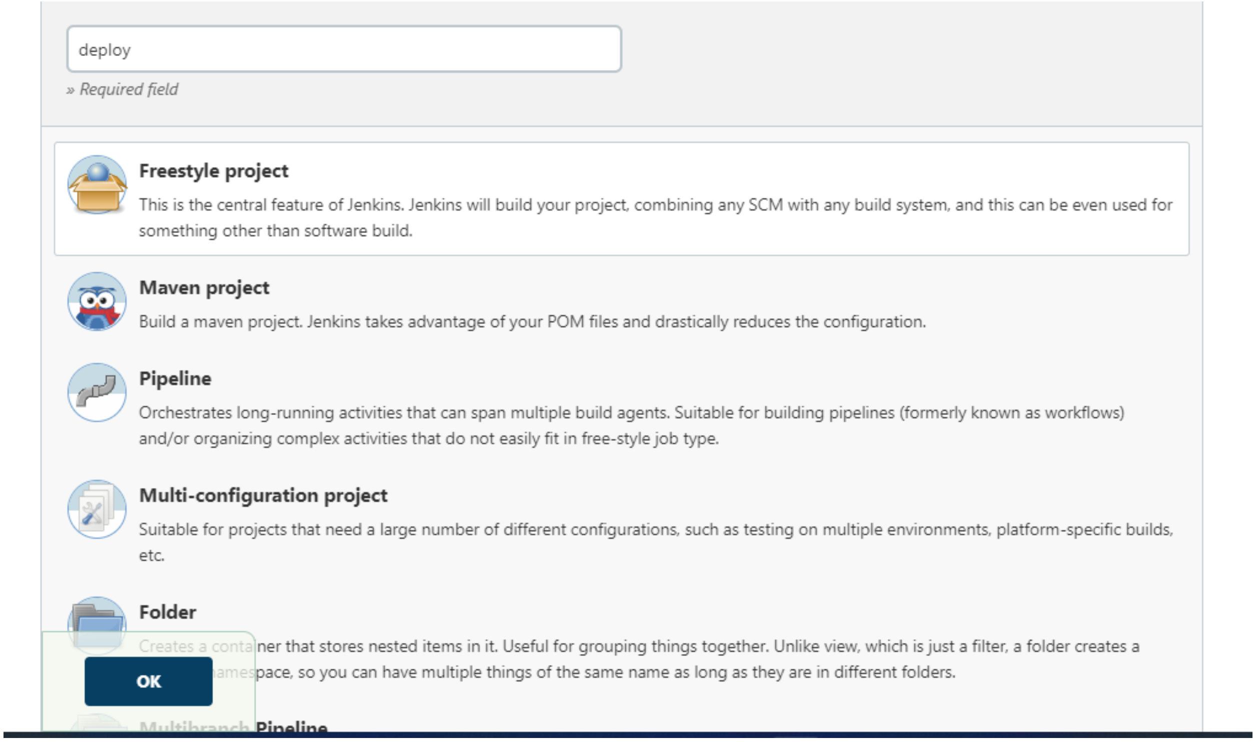
Plugin Manager

deploy to container

Install	Name ↓	Released
☑	**Deploy to container** 1.16 Artifact Uploaders This plugin allows you to deploy a war to a container after a successful build. Glassfish 3.x remote deployment	1 yr 10 mo ago

Update information obtained: 1 hr 10 min ago

5. Create a new freestyle project and name DeployWar and enter details as shown below.



deploy

» 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.

Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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.

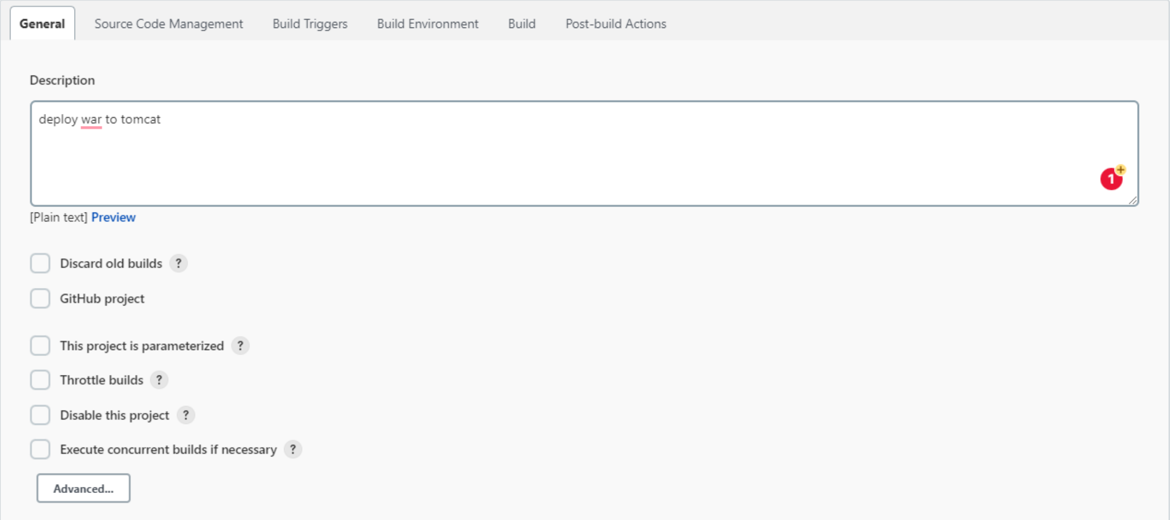
Multi-configuration project

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 namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline



Description

deploy war to tomcat

[Plain text] Preview

□ Discard old builds ?

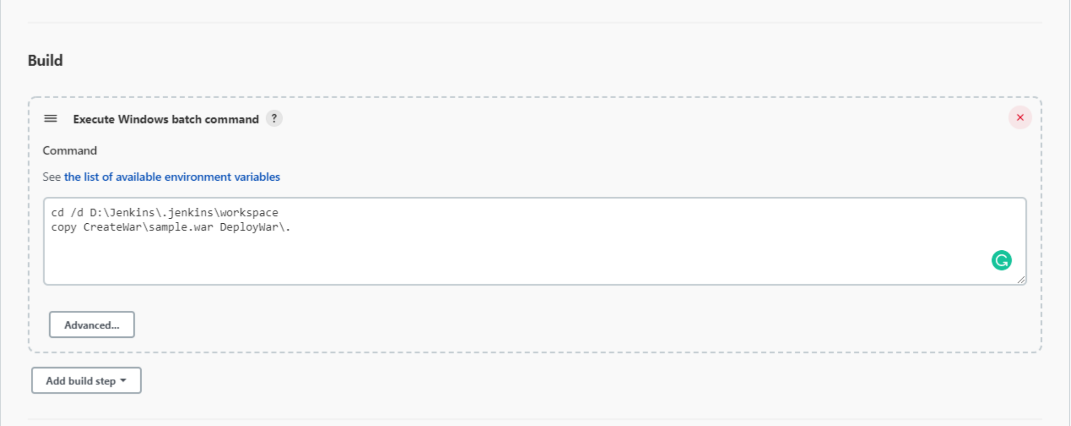
□ GitHub project

□ This project is parameterized ?

□ Throttle builds ?

□ Disable this project ?

□ Execute concurrent builds if necessary ?



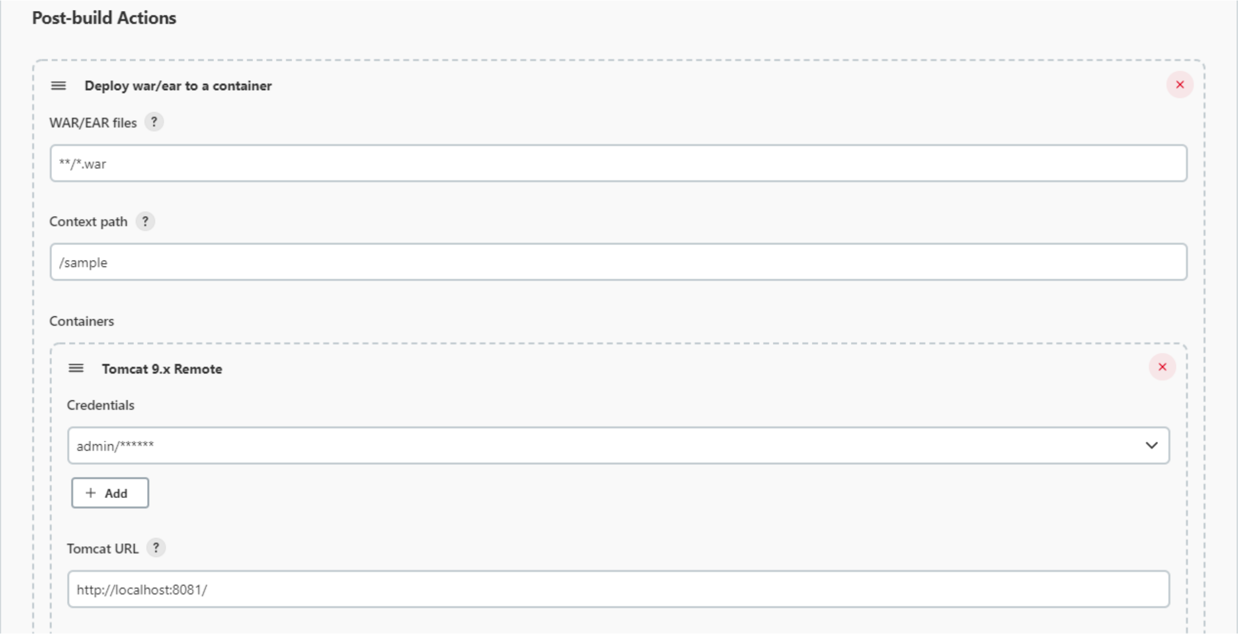
Build

Execute Windows batch command ?

Command

See the list of available environment variables

```
cd /d D:\Jenkins\.jenkins\workspace
copy CreateWar\sample.war DeployWar\.
```



Post-build Actions

Deploy war/ear to a container

WAR/EAR files ?

**/*.war

Context path ?

/sample

Containers

Tomcat 9.x Remote

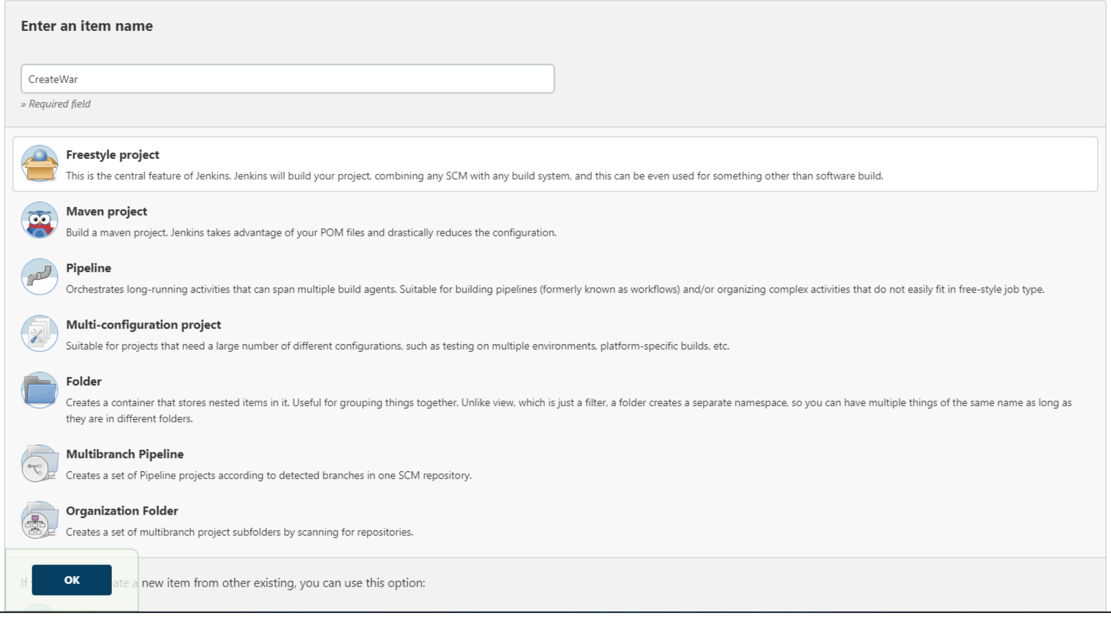
Credentials

admin/******

Tomcat URL ?

http://localhost:8081/

6. Create a new freestyle project and name CreateWar and enter details as shown below.



Enter an item name

CreateWar

» 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.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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.

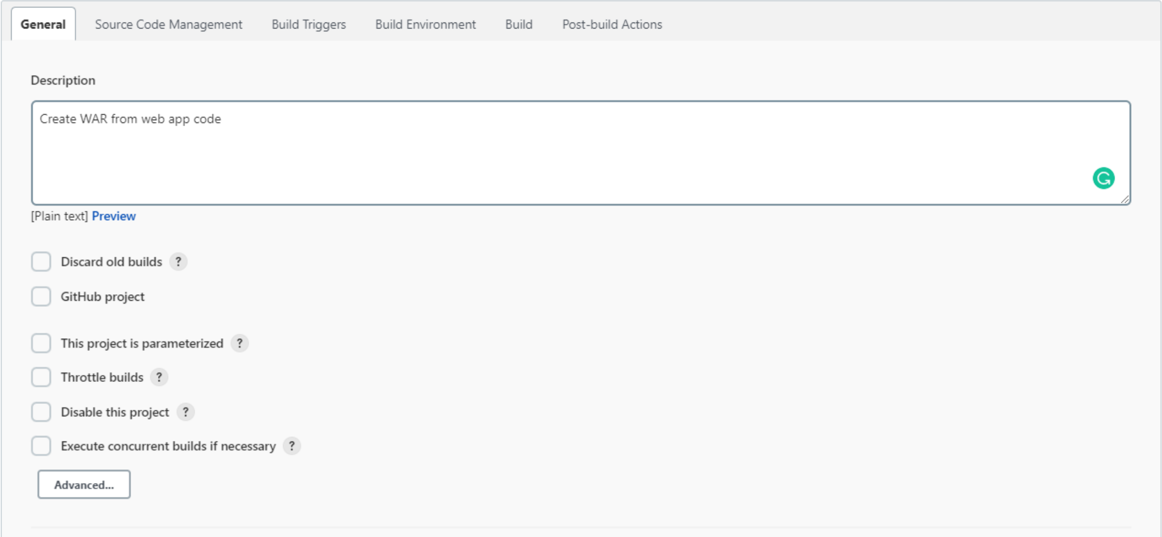
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

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.

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.

If OK ate a new item from other existing, you can use this option:



Description

Create WAR from web app code

[Plain text] Preview

□ Discard old builds ?

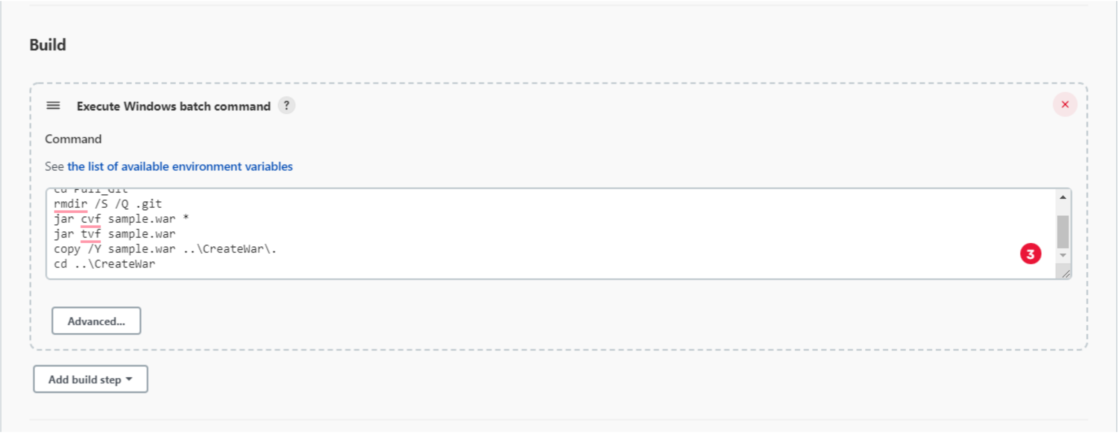
□ GitHub project

□ This project is parameterized ?

□ Throttle builds ?

□ Disable this project ?

□ Execute concurrent builds if necessary ?



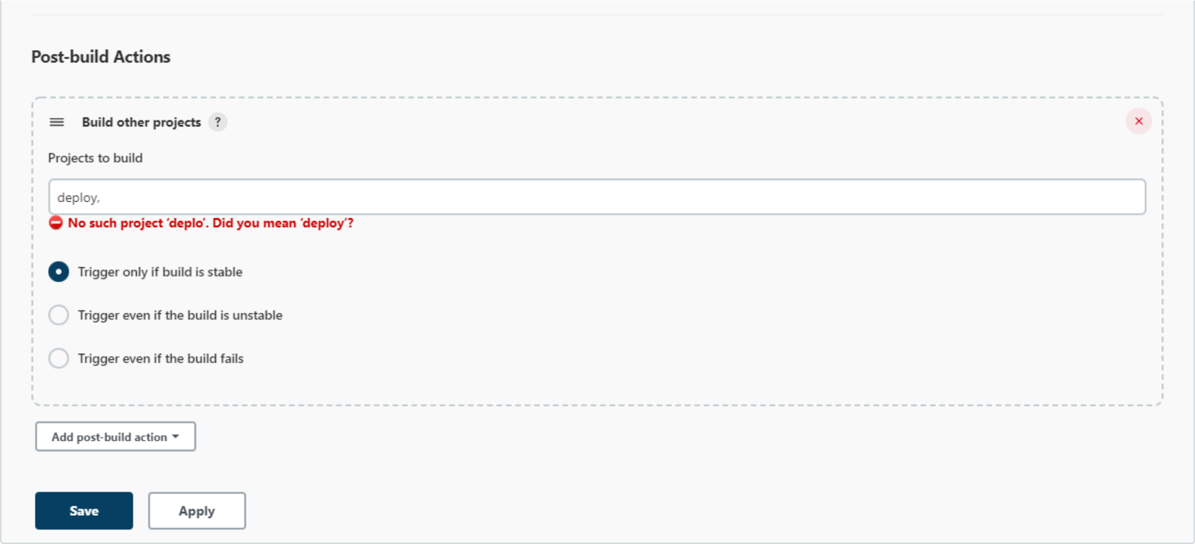
Build

Execute Windows batch command ?

Command

See the list of available environment variables

```
cd Pull_Git
rmdir /S /Q .git
jar cvf sample.war *
jar tvf sample.war
copy /Y sample.war ..\CreateWar\.
cd ..\CreateWar
```



Post-build Actions

Build other projects ?

Projects to build

deploy,

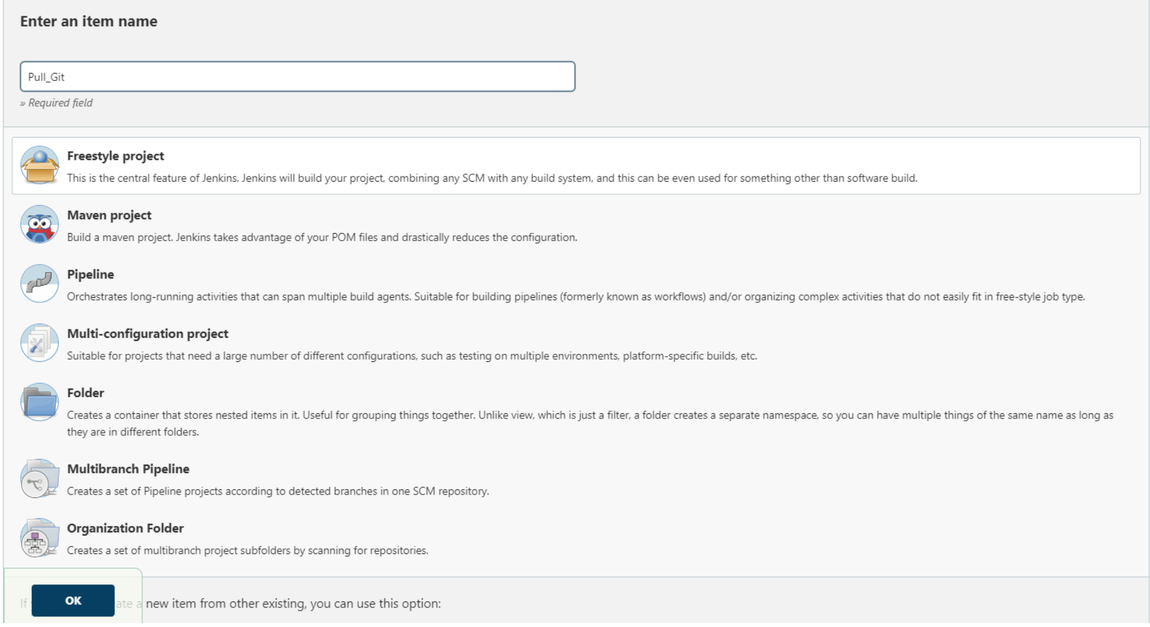
No such project 'deplo'. Did you mean 'deploy'?

● Trigger only if build is stable

○ Trigger even if the build is unstable

○ Trigger even if the build fails

7. Create a new freestyle project and name Pull_Git and enter details as shown below.



Enter an item name

Pull_Git

» 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.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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.

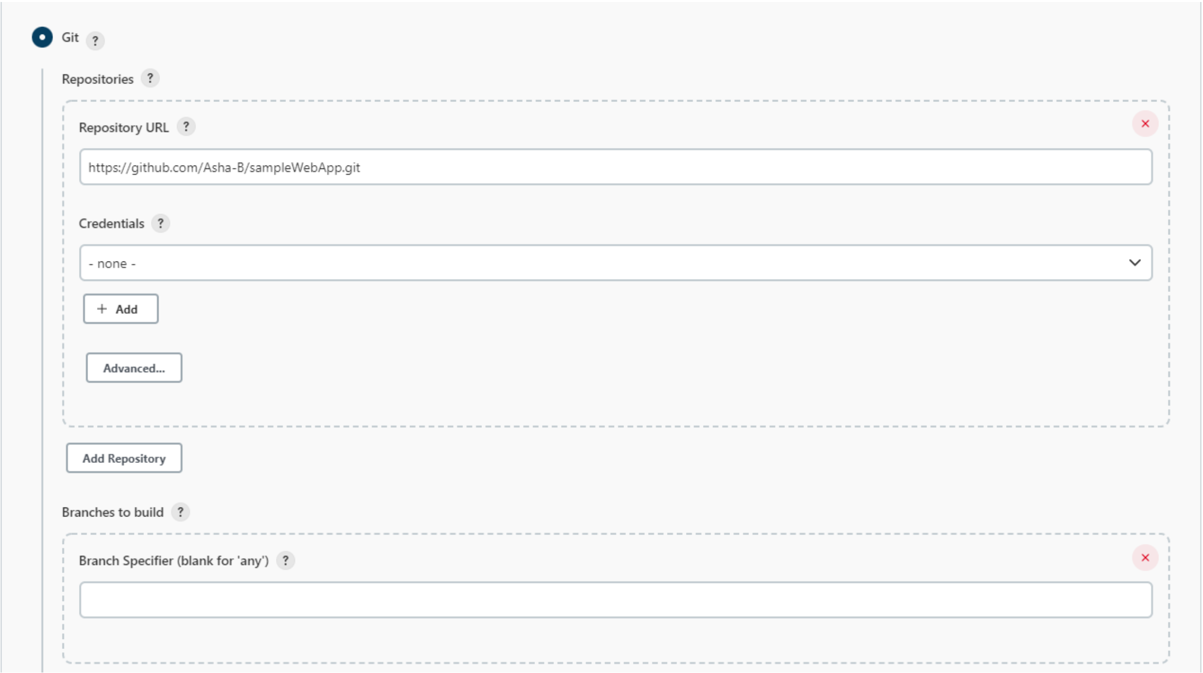
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

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.

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.

If ate a new item from other existing, you can use this option:



● Git

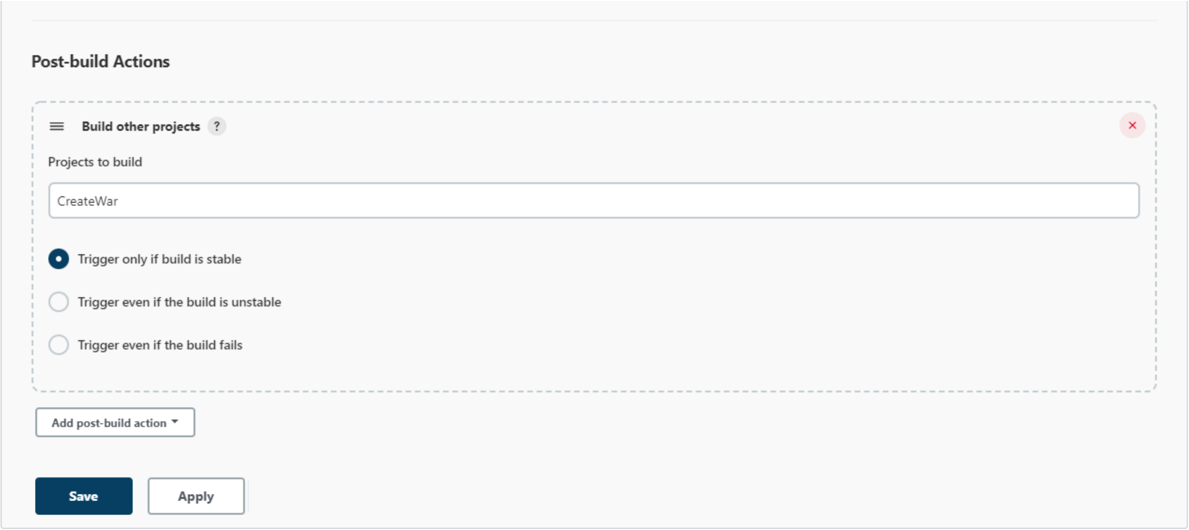
Repositories

Repository URL: https://github.com/Asha-B/sampleWebApp.git

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'):



Post-build Actions

Build other projects

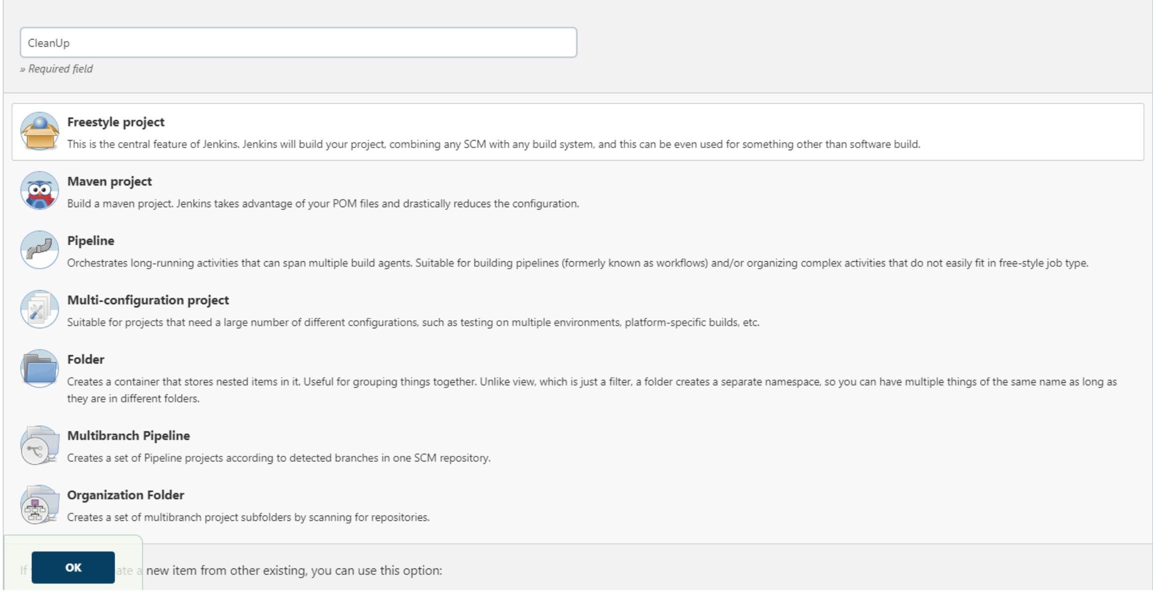
Projects to build: CreateWar

● Trigger only if build is stable

○ Trigger even if the build is unstable

○ Trigger even if the build fails

8. Create a new freestyle project and name CleanUp and enter details as shown below.



CleanUp

» 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.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

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.

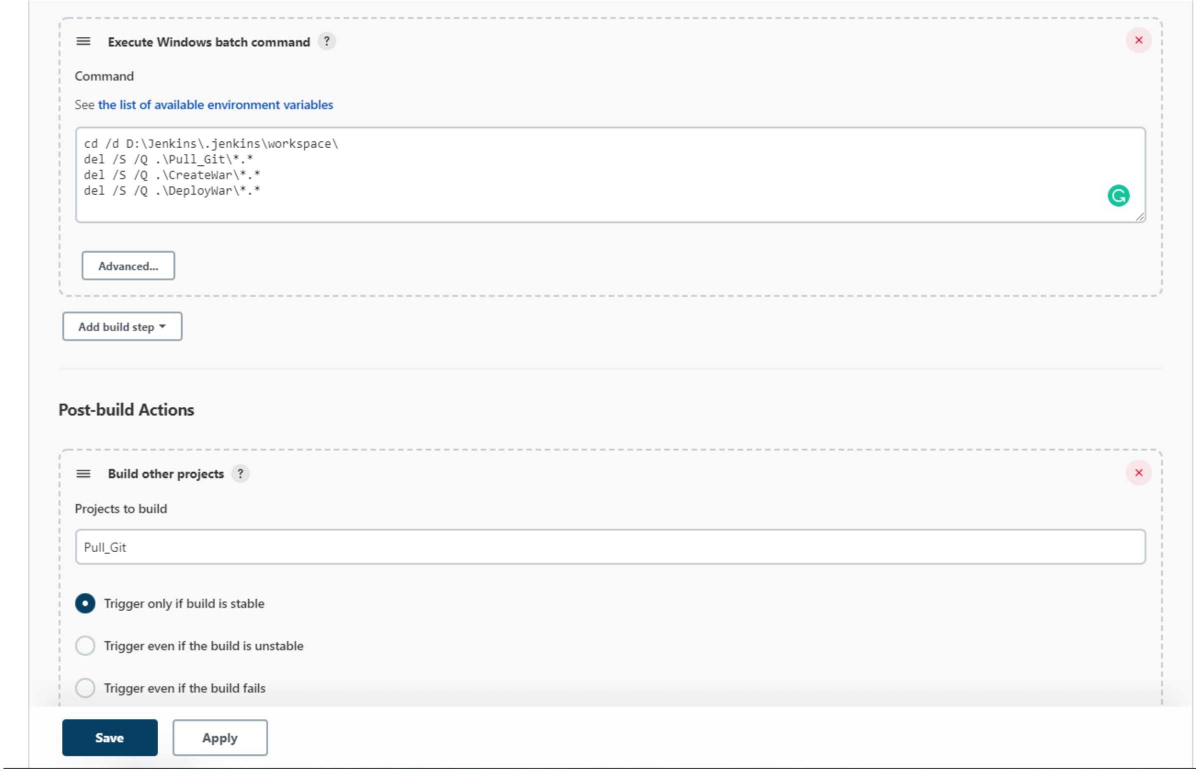
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

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.

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.

If ate a new item from other existing, you can use this option:



Execute Windows batch command ?

Command

See the list of available environment variables

```
cd /d D:\Jenkins\.jenkins\workspace\
del /S /Q .\Pull_Git\*.*
del /S /Q .\CreateWar\*.*
del /S /Q .\DeployWar\*.*
```

Post-build Actions

Build other projects ?

Projects to build

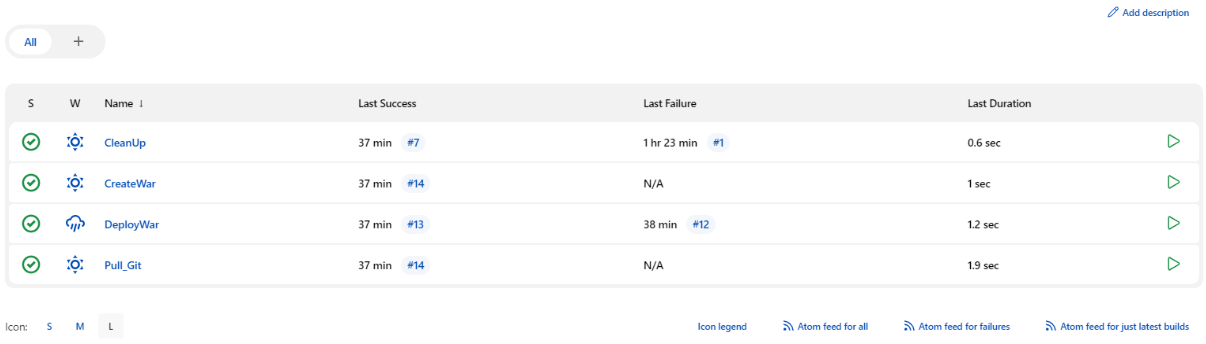
Pull_Git

● Trigger only if build is stable

○ Trigger even if the build is unstable

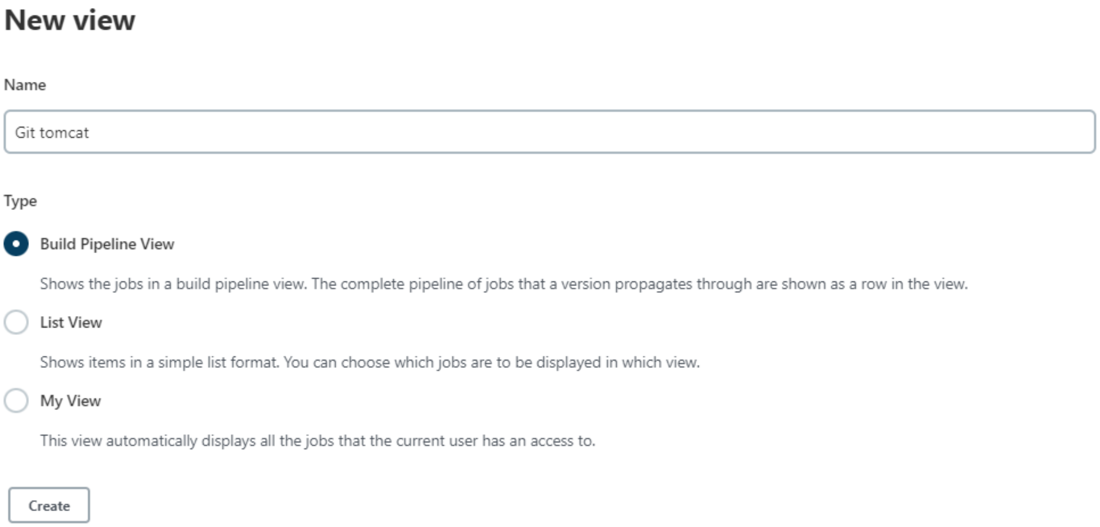
○ Trigger even if the build fails

9. Now build the Pull_Git project.



S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☀	CleanUp	37 min #7	1 hr 23 min #1	0.6 sec	▷
✓	☀	CreateWar	37 min #14	N/A	1 sec	▷
✓	☂	DeployWar	37 min #13	38 min #12	1.2 sec	▷
✓	☀	Pull_Git	37 min #14	N/A	1.9 sec	▷

10. Create a new Pipeline.


New view

Name: Git tomcat

Type

● Build Pipeline View

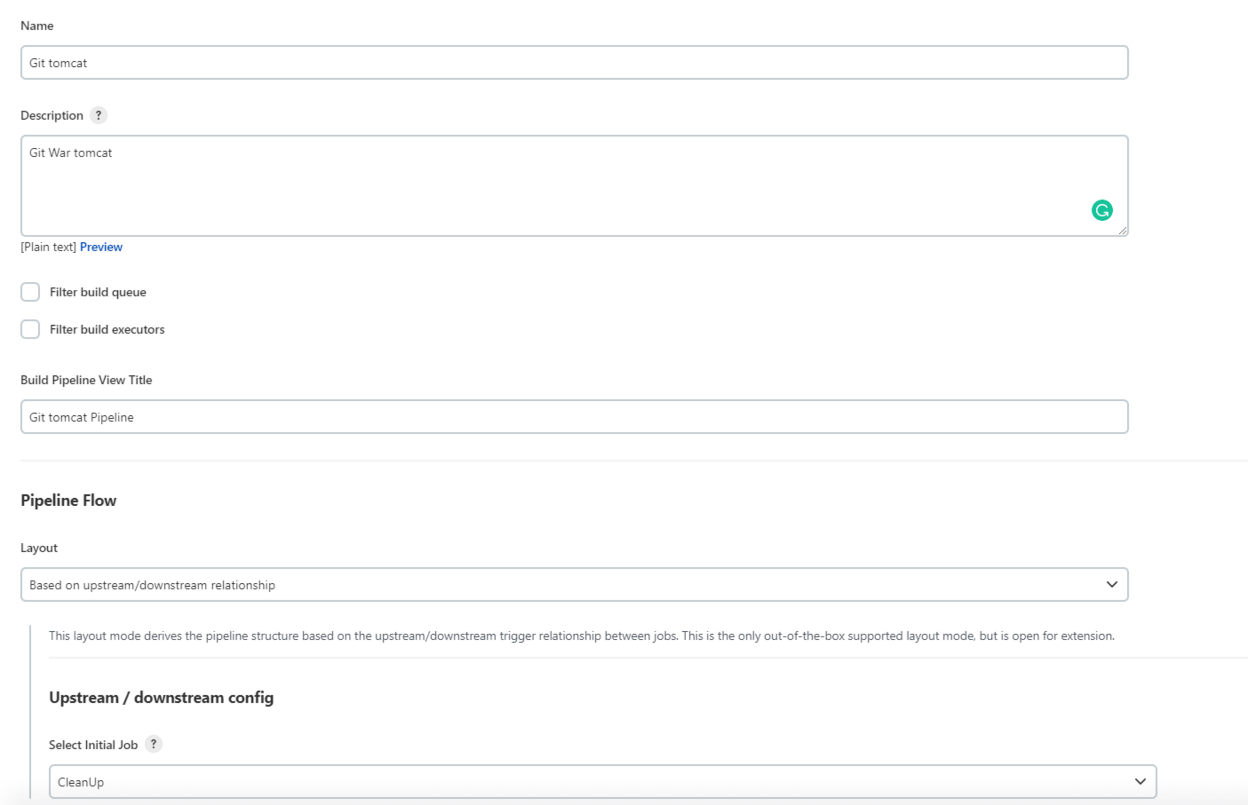
Shows the jobs in a build pipeline view. The complete pipeline of jobs that a version propagates through are shown as a row in the view.

○ List View

Shows items in a simple list format. You can choose which jobs are to be displayed in which view.

○ My View

This view automatically displays all the jobs that the current user has an access to.



Name

Git tomcat

Description ?

Git War tomcat

[Plain text] Preview

□ Filter build queue

□ Filter build executors

Build Pipeline View Title

Git tomcat Pipeline

Pipeline Flow

Layout

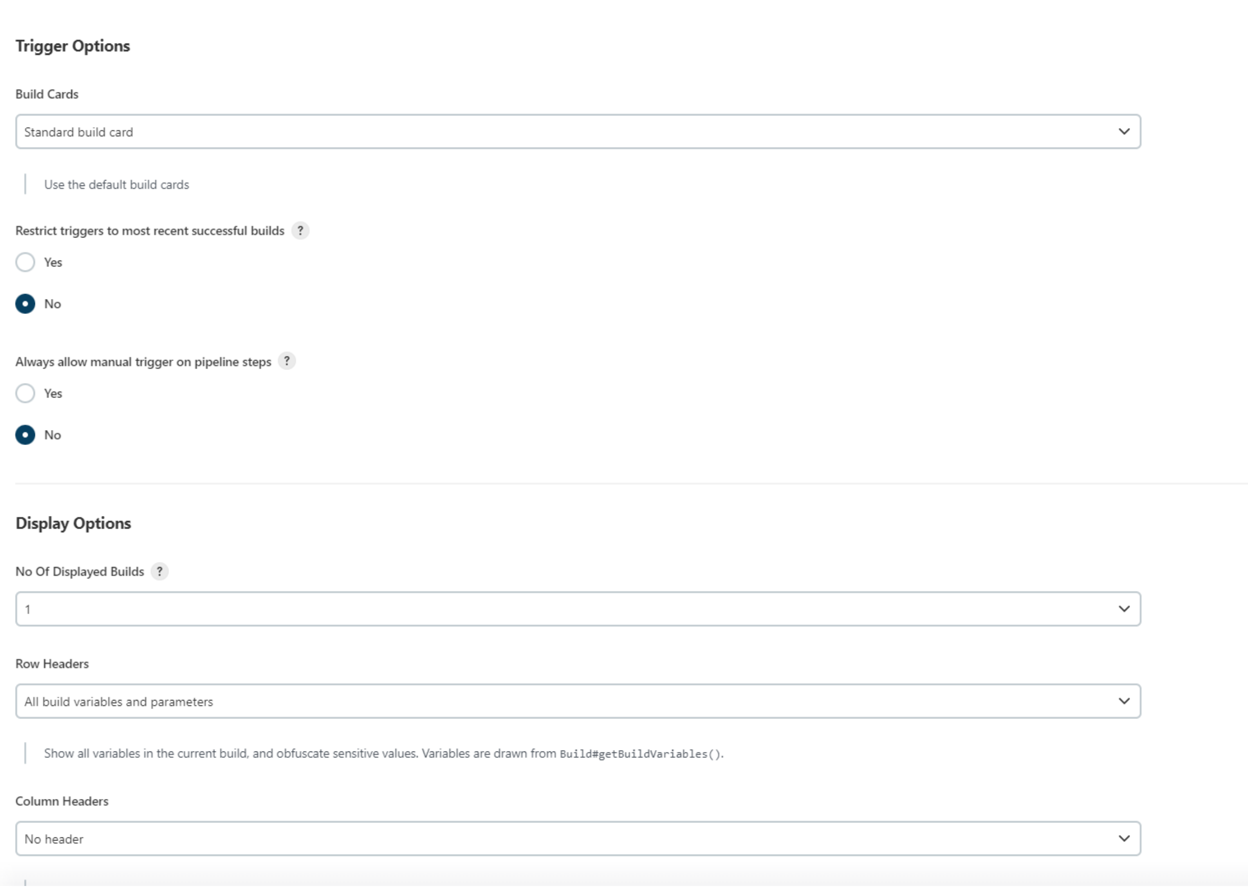
Based on upstream/downstream relationship

This layout mode derives the pipeline structure based on the upstream/downstream trigger relationship between jobs. This is the only out-of-the-box supported layout mode, but is open for extension.

Upstream / downstream config

Select Initial Job ?

CleanUp



Trigger Options

Build Cards

Standard build card

Use the default build cards

Restrict triggers to most recent successful builds ?

○ Yes

● No

Always allow manual trigger on pipeline steps ?

○ Yes

● No

Display Options

No Of Displayed Builds ?

1

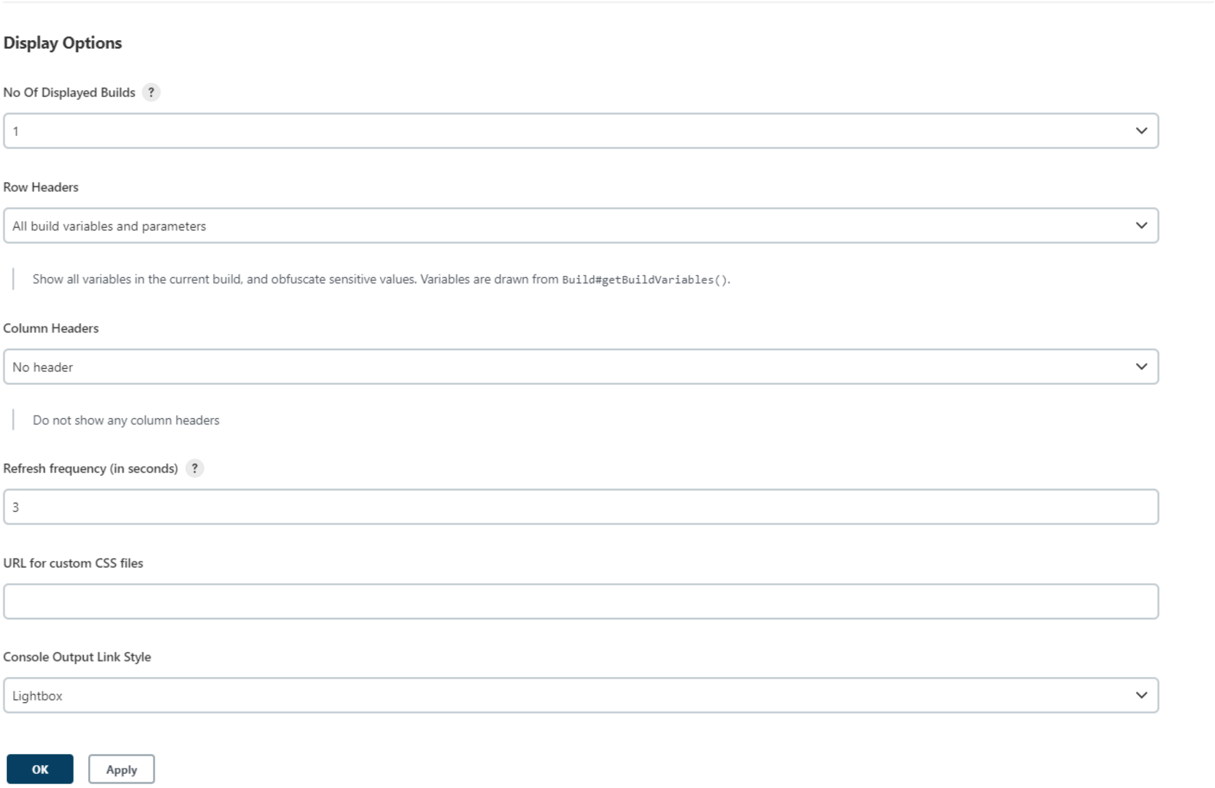
Row Headers

All build variables and parameters

Show all variables in the current build, and obfuscate sensitive values. Variables are drawn from `Build#getBuildVariables()`.

Column Headers

No header



Display Options

No Of Displayed Builds ?

1

Row Headers

All build variables and parameters

Show all variables in the current build, and obfuscate sensitive values. Variables are drawn from `Build#getBuildVariables()`.

Column Headers

No header

Do not show any column headers

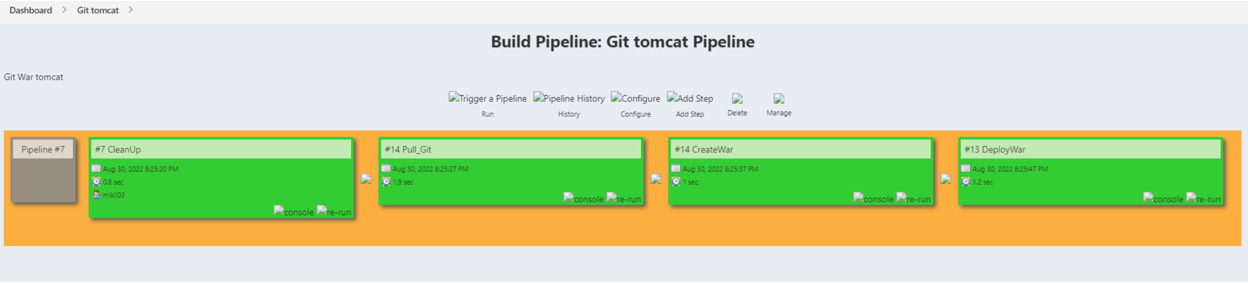
Refresh frequency (in seconds) ?

3

URL for custom CSS files

Console Output Link Style

Lightbox



Dashboard > Git tomcat >

Build Pipeline: Git tomcat Pipeline

Git War tomcat

Pipeline #7

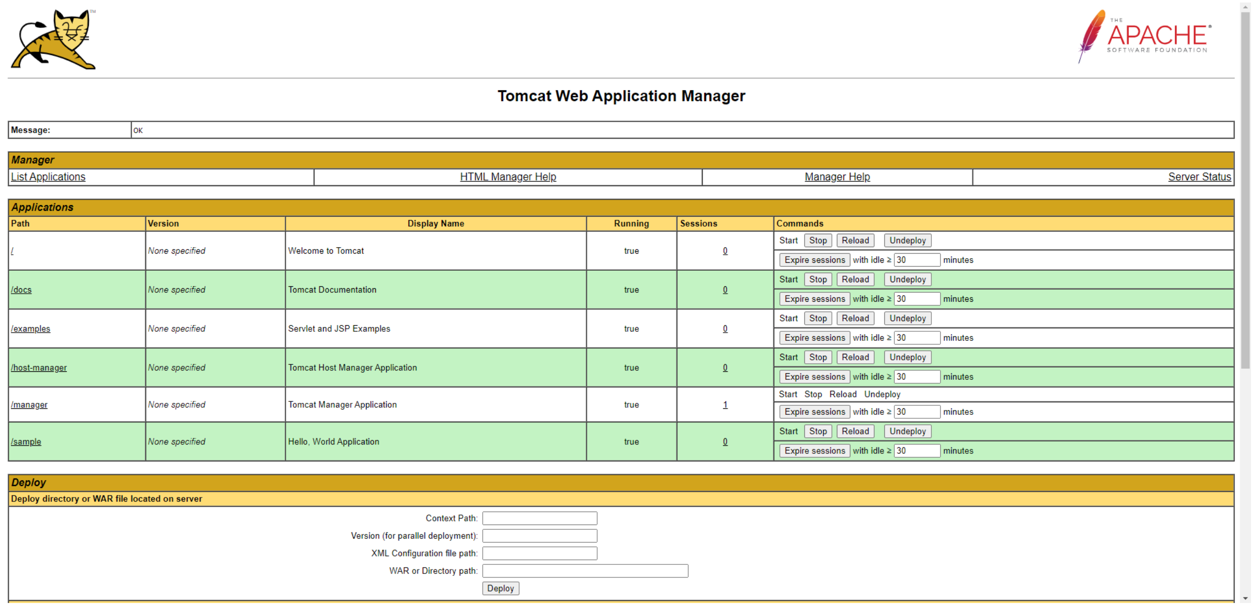
#7 CleanUp
Aug 30, 2022 8:25:20 PM
0.6 sec
mikil03

#14 Pull_Git
Aug 30, 2022 8:25:27 PM
1.9 sec

#14 CreateWar
Aug 30, 2022 8:25:37 PM
1 sec

#13 DeployWar
Aug 30, 2022 8:25:47 PM
1.2 sec

11. Now when the pipeline execution is complete we can see a sample folder in tomcat server at localhost 8081. In this folder we can see our webapp.



Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	*None specified*	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	*None specified*	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	*None specified*	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	*None specified*	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	*None specified*	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/sample	*None specified*	Hello, World Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

Deploy

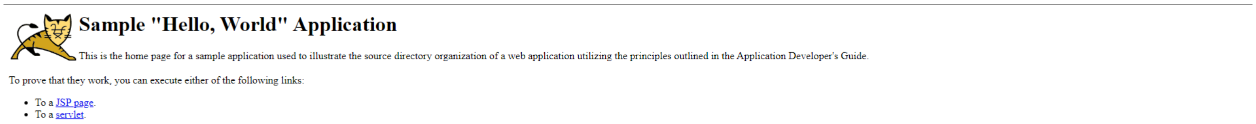
Deploy directory or WAR file located on server

Context Path:

Version (for parallel deployment):

XML Configuration file path:

WAR or Directory path:



Sample "Hello, World" Application

This is the home page for a sample application used to illustrate the source directory organization of a web application utilizing the principles outlined in the Application Developer's Guide.

To prove that they work, you can execute either of the following links:

- To a JSP page.
- To a servlet.

Conclusion

Thus we successfully built a Jenkins Pipeline.