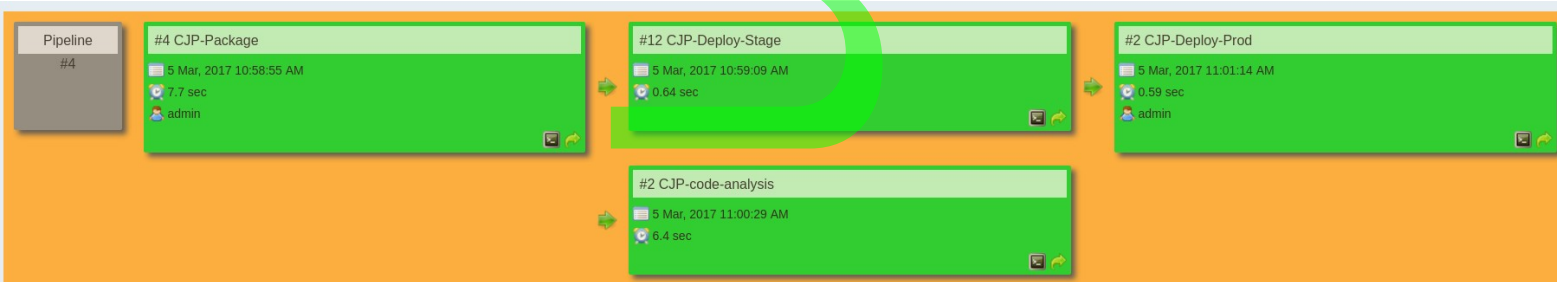


Complete Jenkins Project

This project is a complete build pipeline which will build java SC, create war package, run static code analysis, deploy package to staging tomcat server & deploy to prod tomcat server.

Pipeline consist of below mentioned jenkins job working together to create entire continous delivery pipeline.

1. **Package job** – This job will checkout source code from git repository, use Maven to build the code, archive the artifacts, trigger Static code anaysis job & staging deploy job
2. **Static code analysis** – This job will checkout source code from git repository, use Maven to run checkstyle code analysis and publish graph for STA.
3. **Deploy to staging Tomcat server** – This job will copy the war artifact from package job to deploy job, Deploy the artifacts to Staging tomcat server & trigger Prod deploy job.
4. **Deploy to Prod Tomcat Server** – This job will copy the war artifact from package job to deploy job, Deploy the artifacts to Staging tomcat server & trigger Prod deploy job.



Prerequisites:

1. Jenkins server: - Jenkins server should have openjdk 1.7, Git, Maven (Follow jenkins setup doc)
Plugins:- git, maven, copy artifacts, deploy to container, checkstyle, build pipeline
2. Staging tomcat server:- Create a vm or ec2 instance with ubuntu OS & Follow tomcat setup doc.
3. Prod tomcat server:- Create a vm or ec2 instance with ubuntu OS & Follow tomcat setup doc.

Create jenkins jobs:

1. CJP-package:- Create a new job with CJP-package name, freestyle project.
Refer below mentioned screenshots to setup the job

Source Code Management

☐ None
☒ Git

Repositories

Repository URL

Credentials

Branches to build

Branch Specifier (blank for 'any')

Repository browser

Additional Behaviours

☐ Subversion

Build

Goals

Post-build Actions

Files to archive

Projects to build

☒ Trigger only if build is stable
☐ Trigger even if the build is unstable
☐ Trigger even if the build fails

2. CJP-code-analysis:- Create a new job with CJP-code-analysis name, freestyle project.
Refer below mentioned screenshots to setup the job

The screenshot displays the Jenkins configuration interface for a new job. It is divided into three main sections: Source Code Management, Build, and Post-build Actions.

Source Code Management:

- None** (radio button) **Git** (radio button, selected)
- Repositories:**
 - Repository URL: `https://github.com/jleetutorial/maven-project.git`
 - Credentials: `- none -` (dropdown menu) **Add** (button)
 - Advanced...** (button)
 - Add Repository** (button)
- Branches to build:**
 - Branch Specifier (blank for 'any'): `*/master`
 - Add Branch** (button)
- Repository browser: `(Auto)` (dropdown menu)
- Additional Behaviours: **Add** (button)
- Subversion** (radio button)

Build:

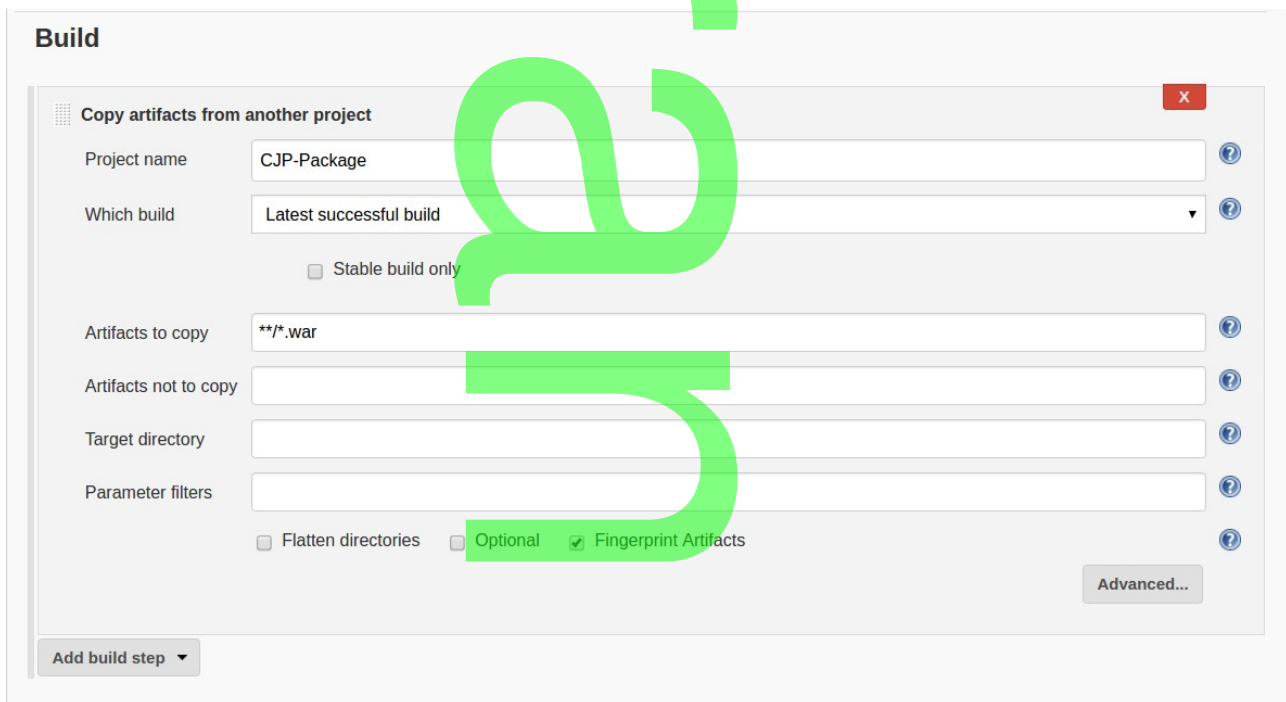
- Invoke top-level Maven targets** (checkbox, checked)
 - Goals: `checkstyle:checkstyle`
 - Advanced...** (button)
- Add build step** (button)

Post-build Actions:

- Publish Checkstyle analysis results** (checkbox, checked)
 - Checkstyle results:
 - Advanced...** (button)
- Add post-build action** (button)

[Fileset includes](#) setting that specifies the generated raw CheckStyle XML report files, such as `**/checkstyle-result.xml`. Basedir of the fileset is `the workspace root`. If no value is set, then the default `**/checkstyle-result.xml` is used. Be sure not to include any non-report files into this pattern.

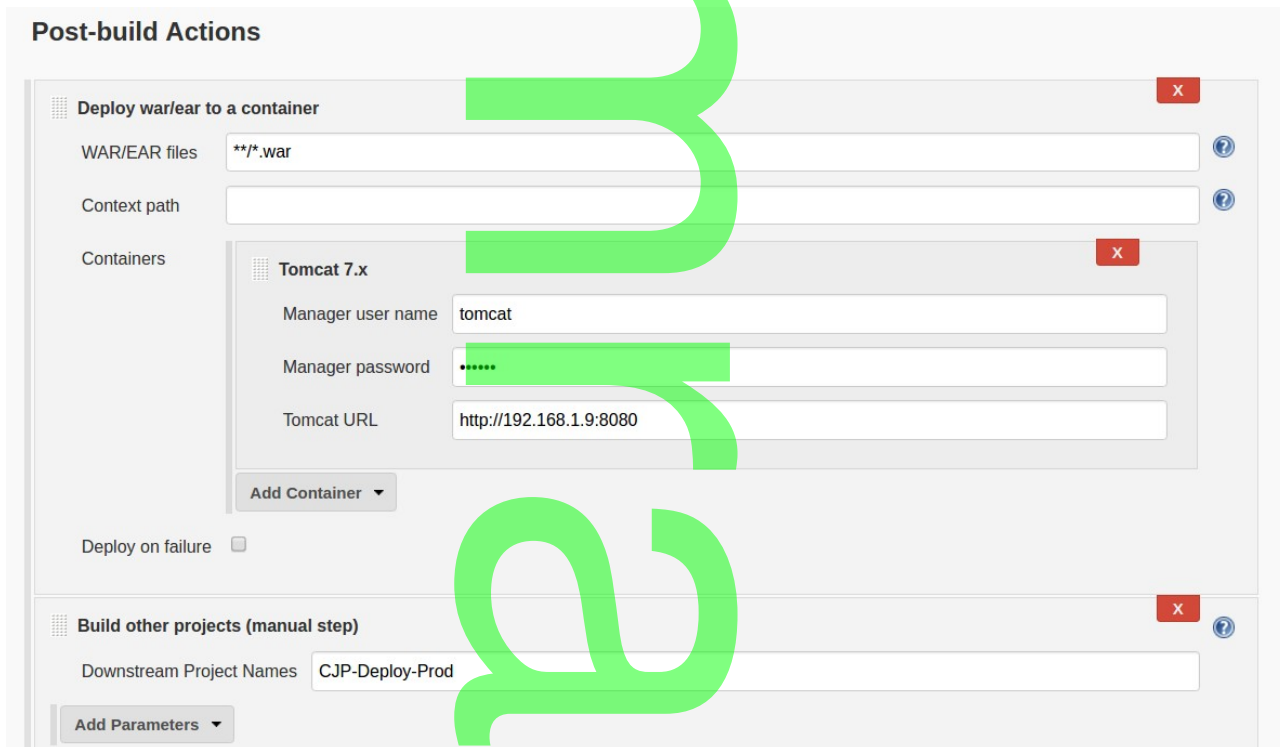
3. CJP-Deploy-Stage:- Create a new job with CJP-Deploy-Stage name, freestyle project.
Refer below mentioned screenshots to setup the job



The screenshot displays the 'Build' configuration page in Jenkins. The 'Copy artifacts from another project' section is active, showing the following settings:

- Project name:** CJP-Package
- Which build:** Latest successful build
- ☐ Stable build only
- Artifacts to copy:** **/*.war
- Artifacts not to copy:** (empty field)
- Target directory:** (empty field)
- Parameter filters:** (empty field)
- ☐ Flatten directories
- ☐ Optional
- ☒ Fingerprint Artifacts

At the bottom left, there is a button labeled 'Add build step'. At the bottom right, there is a button labeled 'Advanced...'. The top right corner of the configuration area has a red 'X' button.



The screenshot shows the 'Post-build Actions' configuration in Jenkins. It contains two main sections: 'Deploy war/ear to a container' and 'Build other projects (manual step)'. The first section has fields for 'WAR/EAR files' (set to '**/*.war'), 'Context path' (empty), and a list of 'Containers'. A container named 'Tomcat 7.x' is selected, showing fields for 'Manager user name' (tomcat), 'Manager password' (masked with dots), and 'Tomcat URL' (http://192.168.1.9:8080). There is an 'Add Container' button and a 'Deploy on failure' checkbox. The second section, 'Build other projects (manual step)', has a 'Downstream Project Names' field set to 'CJP-Deploy-Prod' and an 'Add Parameters' button. Red close buttons are visible in the top right of each section.

Post-build Actions

Deploy war/ear to a container

WAR/EAR files:

Context path:

Containers:

- Tomcat 7.x**
 - Manager user name:
 - Manager password:
 - Tomcat URL:

Add Container ▾

Deploy on failure ☐

Build other projects (manual step)

Downstream Project Names:

Add Parameters ▾

4. CJP-Deploy-Prod:- Create a new job with CJP-Deploy-Prod name, freestyle project.
Refer below mentioned screenshots to setup the job

Build

Copy artifacts from another project

Project name

CJP-Package

Which build

Latest successful build

☐ Stable build only

Artifacts to copy

**/*.war

Artifacts not to copy

Target directory

Parameter filters

☐ Flatten directories

☐ Optional

☒ Fingerprint Artifacts

Advanced...

Add build step

Post-build Actions

Deploy war/ear to a container

WAR/EAR files

**/*.war

Context path

Containers

Tomcat 7.x

Manager user name

tomcat

Manager password

Tomcat URL

http://192.168.1.10:8080

Add Container

Deploy on failure

☐

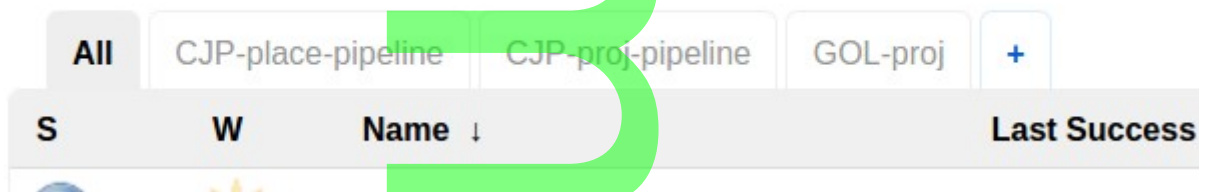
Add post-build action

Setup build pipeline view

Go to jenkins main dashboard

Click on the plus symbol

Visualpath Imran Teli



Select Build Pipeline View, give pipeline view a name and click OK

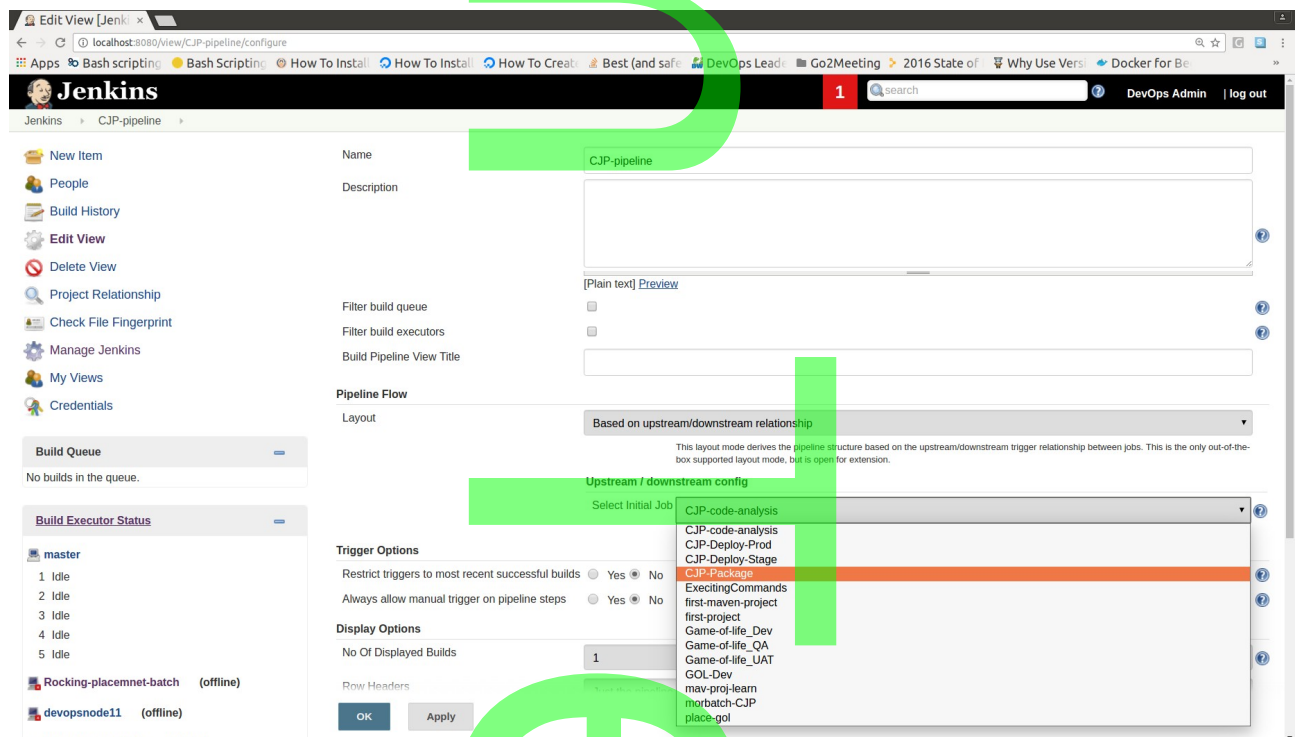
View name

☒ **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.

Select initial Job – CJP-Package & click OK



The screenshot displays the Jenkins web interface for a pipeline named 'CJP-pipeline'. The 'Build Pipeline' section shows a sequence of four steps: #4 CJP-Package, #12 CJP-Deploy-Stage, #2 CJP-Deploy-Prod, and #2 CJP-code-analysis. Each step is represented by a green box with a status icon (a green circle with a white checkmark) indicating successful completion. The steps are connected by arrows, showing a sequential flow. The interface includes a top navigation bar with the Jenkins logo, a search bar, and a 'DevOps Admin' link. A footer at the bottom indicates the page was generated on 05-Mar-2017 at 12:39:07 IST, using the REST API, and the Jenkins version is 2.32.2.

Step	Name	Status	Start Time	Duration	User
#4	CJP-Package	Success	5 Mar, 2017 10:58:55 AM	7.7 sec	admin
#12	CJP-Deploy-Stage	Success	5 Mar, 2017 10:59:09 AM	0.64 sec	admin
#2	CJP-Deploy-Prod	Success	5 Mar, 2017 11:01:14 AM	0.59 sec	admin
#2	CJP-code-analysis	Success	5 Mar, 2017 11:00:29 AM	6.4 sec	admin

Click run and verify if the entire pipeline works.