

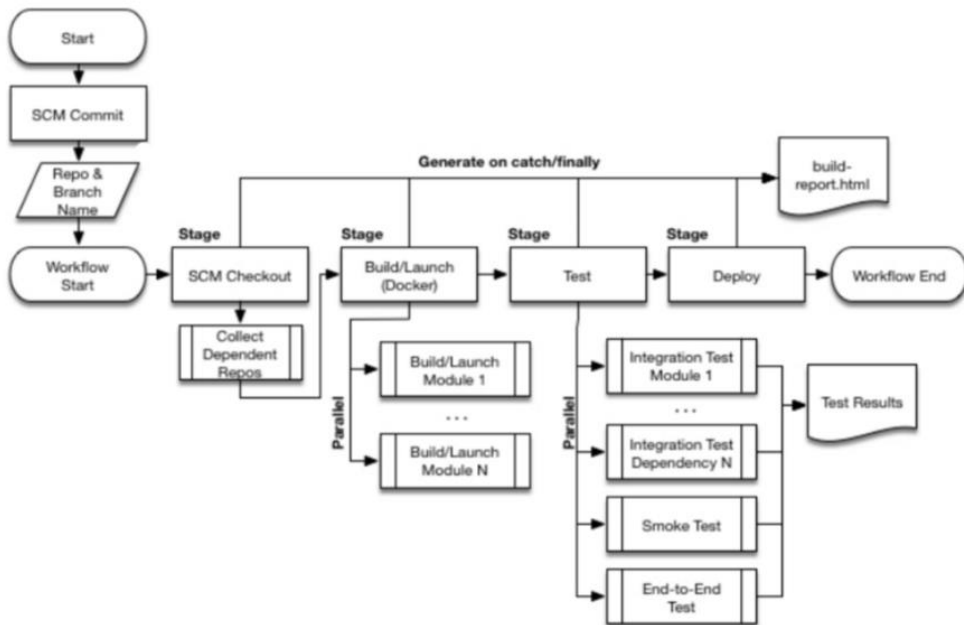
Jenkins – Continuous Integration

- Jenkins is an open source tool that allows continuous integration and continuous delivery of projects, regardless of the platform you are working on
- It is a free source that can handle any kind of build or continuous integration
- You can integrate Jenkins with a number of testing and deployment technologies

Why Jenkins

- Jenkins is a software that allows **continuous integration**
- Jenkins will be installed on a server where the central build will take place

Jenkins Workflow

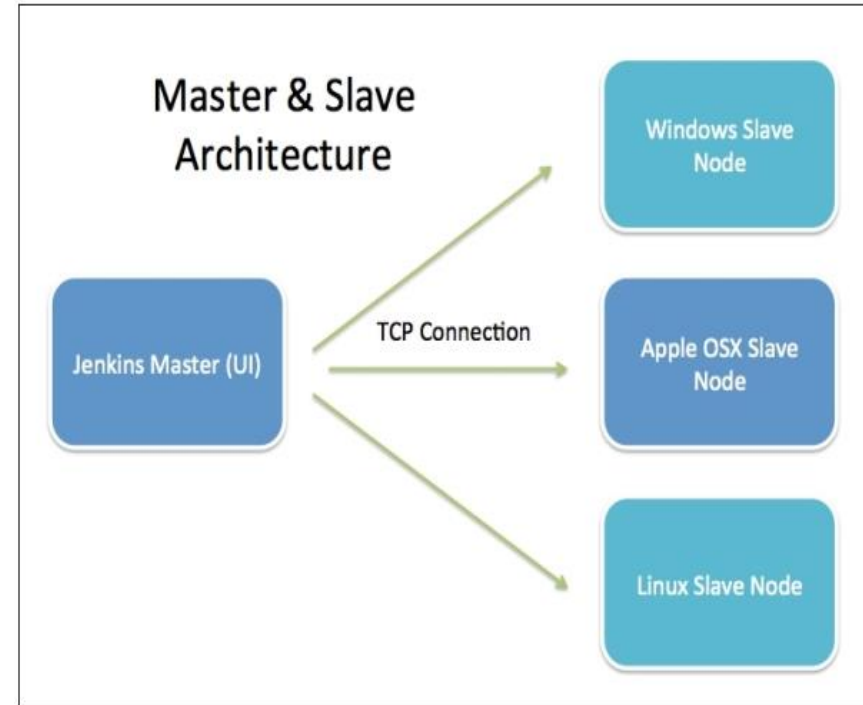


What is Continuous Integration

- Continuous Integration is a development practice that requires developers to integrate code into a shared repository at regular intervals
- This concept was meant to remove the problem of finding later occurrence of issues in the build lifecycle
- Continuous integration requires the developers to have frequent builds
- The common practice is that whenever a code commit occurs, a build should be triggered
- This can be done on any technology or platform


Jenkins Architecture

- Jenkins Architecture is based on the distributed. This has 2 components.
 - Jenkins Server
 - Jenkins Node/Slave/Build Server
- Your main Jenkins server is the master
- The master's job is to handle scheduling build jobs, dispatching builds to the slaves for the actual execution, monitor the slaves and recording and presenting the build results
- Even in a distributed architecture, a master instance of Jenkins can also execute build jobs directly
- The job of the slaves is to do as they are configured in the Jenkins Server, which involves executing build jobs dispatched by the master



Installing Jenkins

- Follow lab document to install and configure Jenkins


**Jenkins**


search


Admin | log out


Jenkins


ENABLE AUTO REFRESH


 New Item


 People

 Build History

 **Manage Jenkins**

 My Views

 Credentials

 New View

Build Queue

No builds in the queue.

Build Executor Status


1 Idle


2 Idle


Manage Jenkins


Restore the previous version of Jenkins


Downgrade to 2.138



**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

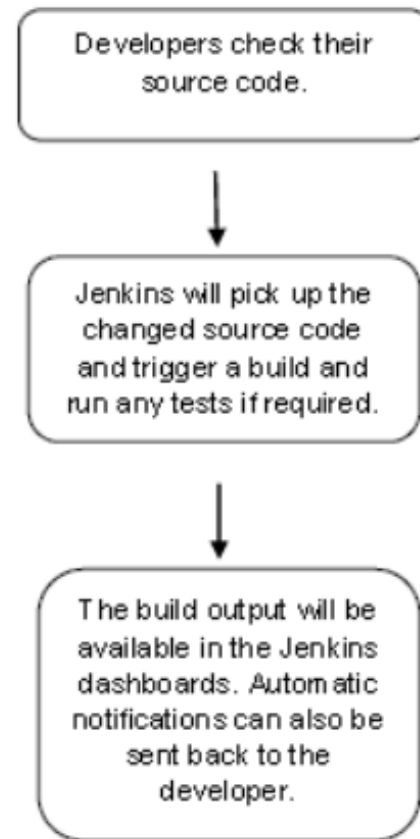
**Global Tool Configuration**
Configure tools, their locations and automatic installers.

**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.
 There are updates available

Why Jenkins?

- Jenkins is a software that allows **continuous integration**
- Jenkins will be installed on a server where the central build will take place
- The following flowchart demonstrates a very simple workflow of how Jenkins works:



Installing Git

- Download git and install on your VM (already done in Git lesson)

Installing Maven

- Maven is an open source build tool for many platforms
- Based on the concept of a project object model (POM), Maven can manage a project's build, reporting and documentation from a central piece of information
- Using maven we can build and manage any Java based project
- The primary goal of Maven is to provide developer with the following –
 - A comprehensive model for projects, which is reusable, maintainable, and easier to comprehend
 - Plugins or tools that interact with this declarative model
- To install maven run the following commands:
`apt-get update`
`apt-get install maven`
- To verify Maven installation run:
`mvn --version`

Install Tomcat

- Go to <http://tomcat.apache.org/>
- Download the Ubuntu distribution and install it on your VM

Jenkins Login



Welcome to Jenkins!

admin


.....

Sign in








☐ Keep me signed in

Jenkins Dashboard

[←](#) [→](#) [↻](#) [🏠](#) [📄](#) localhost:8080

 **Jenkins** 2 [?](#) [Admin](#) | [log out](#)

Jenkins [ENABLE AUTO REFRESH](#)

-  [New Item](#)
-  [People](#)
-  [Build History](#)
-  [Manage Jenkins](#)
-  [My Views](#)
-  [Credentials](#)
-  [New View](#)


 [add description](#)

All


addressbook

+

S	W	Name ↓	Last Success	Last Failure	Last Duration
---	---	--------	--------------	--------------	---------------

Build Queue 

No builds in the queue.

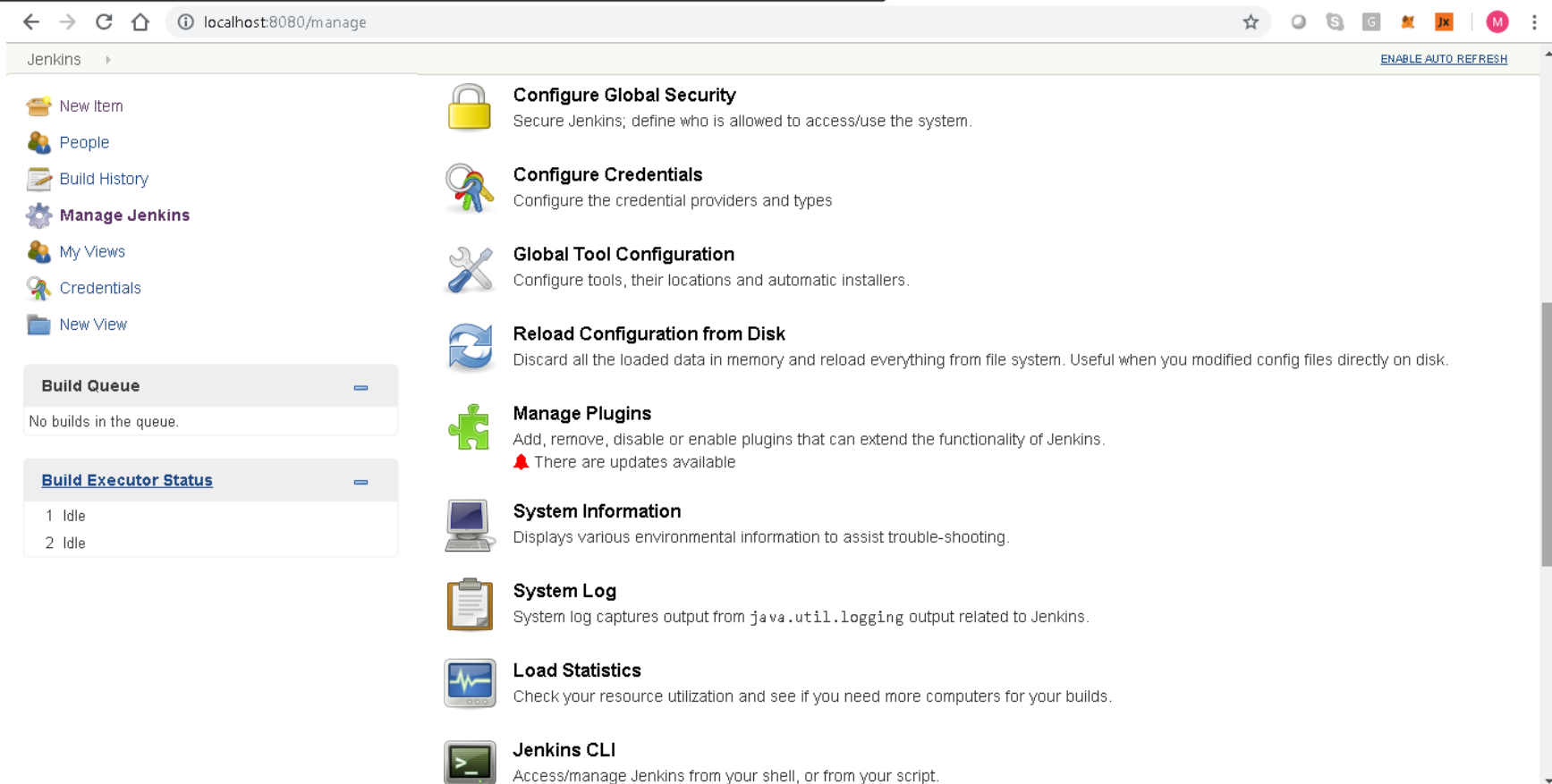
Build Executor Status 

1 Idle

2 Idle

[Legend](#)  [RSS for all](#)  [RSS for failures](#)  [RSS for just latest builds](#)

Manage Jenkins



Manage Plugins

- Click the 'Manage Plugins' option
- Click the Available tab
- This tab will give a list of plugins which are available for downloading
- In the 'Filter' tab type 'Git plugin'
- The list will then be filtered
- Check the Git Plugin option and click on the button 'Install without restart'
- The installation will then begin and the screen will be refreshed to show the status of the download

Install Plugins

localhost:8080/pluginManager/available



2

search

Admin

log out

Jenkins > Plugin Manager

Back to Dashboard

Manage Jenkins

Filter: git

Updates

Available

Installed

Advanced

Install ↓

Name

Version



[GitHub Authentication](#)

0.31

Authentication plugin using GitHub OAuth to provide authentication and authorization capabilities for GitHub and GitHub Enterprise.



[Gitlab Authentication](#)

1.4

This is the an authentication plugin using gitlab OAuth.



[Gitcolony Build Notification](#)

1.1

This plugin updates live branch build status in [Gitcolony](#).



[GitHub Issues](#)

1.2.4

This plugin creates GitHub issues when builds fail, and automatically closes the issue when the build starts passing again.



[Pipeline GitHub Notify Step](#)

1.0.4

Plugin that provides a GitHub status notification step



[Git Parameter](#)

0.9.6

Adds ability to choose branches, tags or revisions from git repositories configured in project.



[bootstrapped-multi-test-results-report](#)

2.1.3

This plugin generates HTML reports using handlebars templates with bootstrap components. [Join chat](#) if you have questions/suggestions



[Git Changelog](#)

2.14

Creates a highly configurable changelog, or relasenotes, from Git.

[Gitolite Build Report Github Status](#)

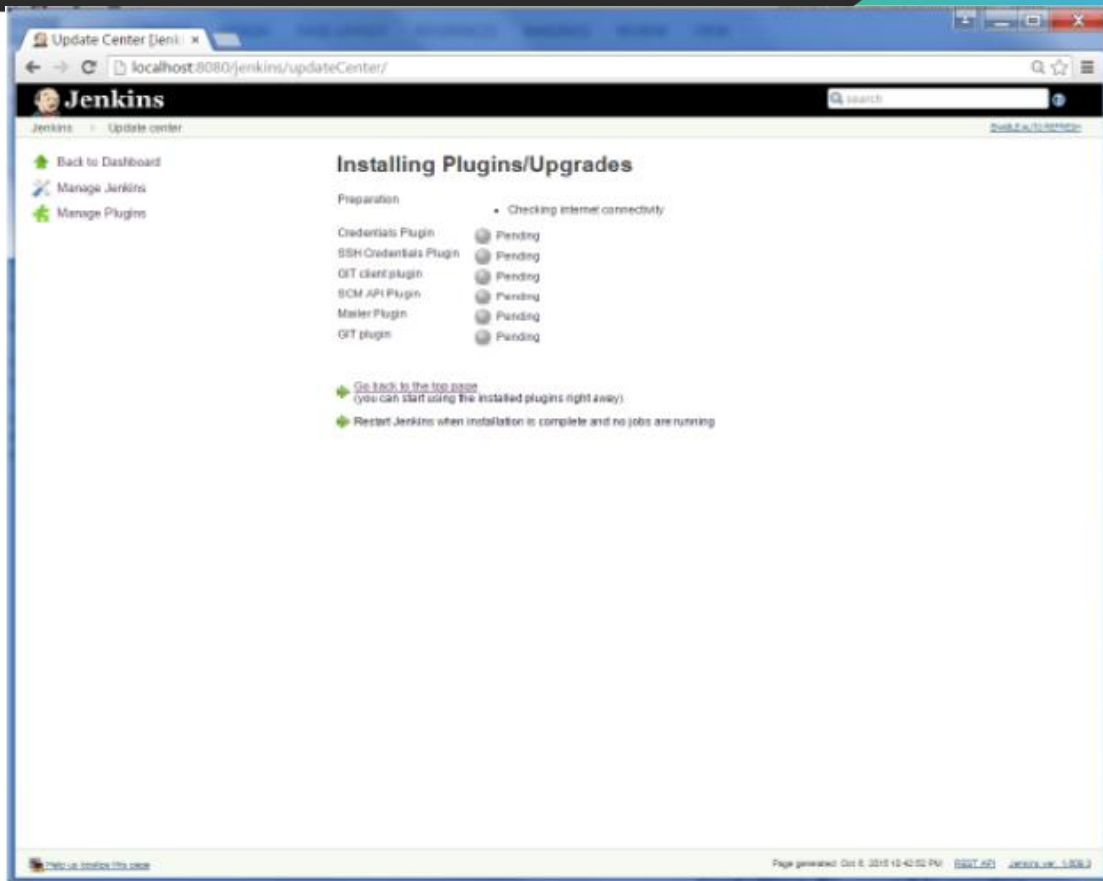
Install without restart

Download now and install after restart

Update information obtained: 17 min ago

Check now

Install Plugins










Creating Jobs

← → ↺ 🏠 ⓘ localhost:8080/view/all/newJob ☆ ⚙️ S G ✨ JX | M ⋮

Jenkins ▸ All ▸

» 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.
-  **External Job**
This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.
-  **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.

OK

Creating Jobs

localhost:8080/job/new_project/configure

Jenkins > new_project >

General Source Code Management Build Triggers Build Environment Build Post-build Actions

[Plain text] [Preview](#)

- ☐ Discard old builds
- ☐ GitHub project
- ☐ This project is parameterized
- ☐ Throttle builds
- ☐ Disable this project
- ☐ Execute concurrent builds if necessary

Advanced...

Source Code Management

- ☒ None
- ☐ Git
- ☐ Subversion

Build Triggers

- ☐ Trigger builds remotely (e.g., from scripts)
- ☐ Build after other projects are built
- ☐ Build periodically
- ☐ GitHub hook trigger for GITScm polling
- ☐ Poll SCM

Save Apply

Setting Up Maven in Jenkins

- Click Manage Jenkins
- Click Global Tool Configuration
- Click add Maven and enter Maven home path from your VM

Global Tool Configuration

← → ↻ 🏠 ⓘ localhost:8080/configureTools/ 🔍 ☆ 🌐 S G 🚀 JX M ⋮

Jenkins > Global Tool Configuration

⚙️ Manage Jenkins

🔧 Global Tool Configuration

Maven Configuration

Default settings provider Use default maven settings ▼

Default global settings provider Use default maven global settings ▼

JDK

JDK installations...

Git

Git installations

Git

Name

Default

Path to Git executable

git.exe

?

☐ Install automatically

?

Delete Git

Add Git ▼

Gradle

Gradle installations

Add Gradle

List of Gradle installations on this system

SonarScanner for MSBuild

SonarScanner for MSBuild installations

Add SonarScanner for MSBuild

List of SonarScanner for MSBuild installations on this system

SonarQube Scanner

SonarQube Scanner installations

Add SonarQube Scanner

List of SonarQube Scanner installations on this system

Save Apply

Running Job

localhost:8080/job/address-compile/10/console

Jenkins

2

search

Admin | log out

Jenkins > address-compile > #10

Back to Project

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete Build

Polling Log

Git Build Data

No Tags


Previous Build

Console Output

```
Started by an SCM change
Building in workspace C:\Program Files (x86)\Jenkins\workspace\address-compile
> git.exe rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url https://github.com/manikabedi/devops.git # timeout=10
Fetching upstream changes from https://github.com/manikabedi/devops.git
> git.exe --version # timeout=10
> git.exe fetch --tags --progress https://github.com/manikabedi/devops.git +refs/heads/*:refs/remotes/origin/*
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
> git.exe rev-parse "refs/remotes/origin/origin/master^{commit}" # timeout=10
Checking out Revision 9507ba7633275aa69434249588b136714fc0d662 (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f 9507ba7633275aa69434249588b136714fc0d662
Commit message: "Update build.xml"
> git.exe rev-list --no-walk 5732dedad5f418a62499e94fdd01d8cf90bc4977 # timeout=10
[address-compile] $ cmd.exe /C "mvn compile && exit %%ERRORLEVEL%%"
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] Building Vaadin Addressbook example 2.0
[INFO] -----
[INFO]
[INFO] --- maven-enforcer-plugin:1.0:enforce (enforce-versions) @ addressbook ---
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ addressbook ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] skip non existing resourceDirectory C:\Program Files (x86)\Jenkins\workspace\address-compile\src\main\resources
[INFO]
[INFO] --- maven-compiler-plugin:3.2:compile (default-compile) @ addressbook ---
[INFO] Nothing to compile - all classes are up to date
[INFO]
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 7.817 s
[INFO] Finished at: 2018-10-27T21:01:57+05:30
```

Jenkins Deployment Plugin

← → ↻ 🏠 ⓘ localhost:8080/pluginManager/installed ☆ ⚙ Ⓢ Ⓜ ⚡ ⚙ | 🔴 ⋮

 **Jenkins**

2

?

Admin

| log out

Jenkins ▶ Plugin Manager

📈 Back to Dashboard

⚙ Manage Jenkins

Filter:

Updates Available **Installed** Advanced

Enabled	Name ↓	Version	Previously installed version	Uninstall
<input checked="" type="checkbox"/>	bouncycastle API Plugin This plugin provides an stable API to Bouncy Castle related tasks.	2.17		<button>Uninstall</button>
<input checked="" type="checkbox"/>	Command Agent Launcher Plugin Allows agents to be launched using a specified command.	1.2		<button>Uninstall</button>
<input checked="" type="checkbox"/>	Credentials Plugin This plugin allows you to store credentials in Jenkins.	2.1.18		<button>Uninstall</button>
<input checked="" type="checkbox"/>	Deploy to container Plugin This plugin allows you to deploy a war to a container after a successful build. Glassfish 3.x remote deployment	1.13		<button>Uninstall</button>
<input checked="" type="checkbox"/>	JDK Tool Allows the JDK tool to be installed via download from Oracle's website.	1.1		<button>Uninstall</button>

Page generated: Dec 11, 2018 12:22:12 PM IST [REST API](#) [Jenkins ver. 2.147](#)

Jenkins Deployment

← → ↻ 🏠 ⓘ localhost:8080/job/address-package/configure ☆ ⚙️ S G Jx M ⋮

Jenkins ▶ address-package ▶

General Source Code Management Build Triggers Build Environment Build **Post-build Actions**

Add build step ▼

Post-build Actions

Archive the artifacts ✕ ⓘ

Files to archive ⓘ

Advanced...

Deploy war/ear to a container ✕ ⓘ

WAR/EAR files ⓘ

Context path ⓘ

Containers

Tomcat 8.x ✕

Credentials ⓘ Add

Tomcat URL

Add Container ▼

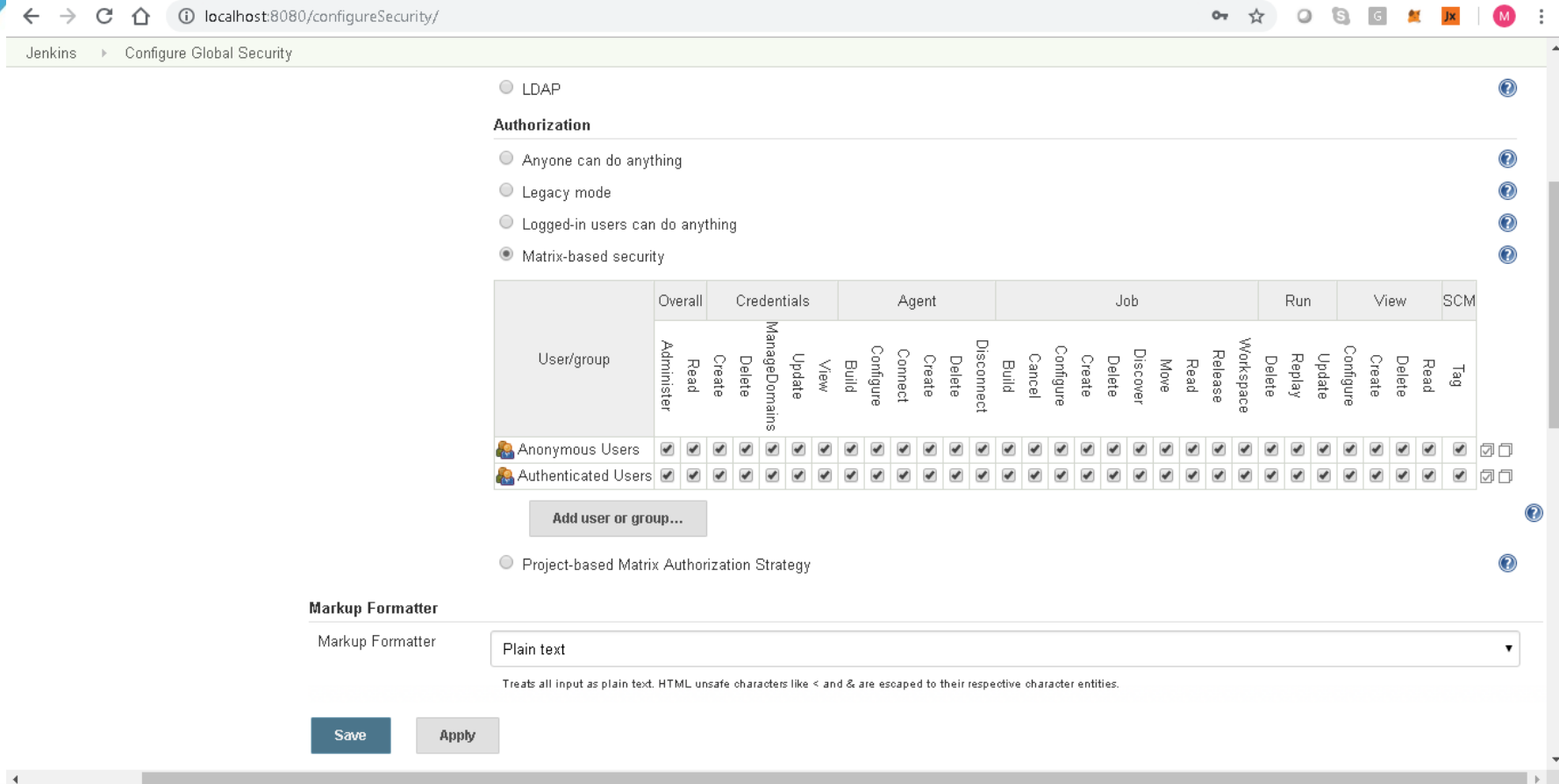
Save

Apply


Integrate Jenkins with Ant

- Install Ant plugin in Jenkins (if not installed as part of initial Jenkins Setup)
- Install Ant on your VM using commands:
 `apt-get update`
 `apt-get install ant`
 `ant --version`
- Go to Global Tool Configuration in Jenkins
- Click on Add Ant and enter your ant home folder

Authentication




← → ↺ 🏠 ⓘ localhost:8080/configureSecurity/ 🔑 🔍 ☆ 🌐 S G 🌟 JX | M ⋮

 **Jenkins**

2 🔍 search ? Admin | log out

Jenkins > Configure Global Security

 **Configure Global Security**

☒ Enable security

☐ Disable remember me

Access Control

Security Realm

☐ Delegate to servlet container

☒ Jenkins' own user database

- ☐ Allow users to sign up

☐ LDAP

Authorization

☐ Anyone can do anything

☐ Legacy mode

☒ Logged-in users can do anything

- ☐ Allow anonymous read access

☐ Matrix-based security

☐ Project-based Matrix Authorization Strategy

Markup Formatter

Markup Formatter

Plain text

Treats all input as plain text. HTML unsafe characters like < and & are escaped to their respective character entities.

Agents

TCP port for JNLP agents

☐ Fixed :

☐ Random

☒ Disable

Agent protocols...

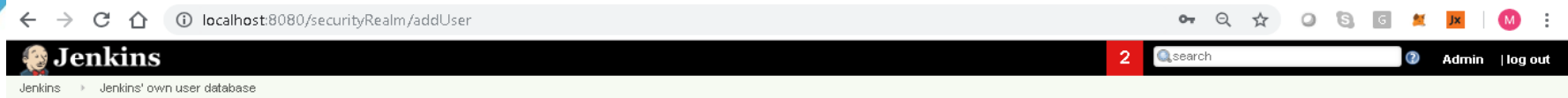
CSRF Protection

☒ Prevent Cross Site Request Forgery exploits

Save

Apply

Create a New User



[Back to Dashboard](#)

 Manage Jenkins **Create User**

Create User

Username:	<input type="text"/>
Password:	<input type="password"/>
Confirm password:	<input type="password"/>
Full name:	<input type="text"/>
E-mail address:	<input type="text"/>

Create User

Page generated: Dec 11, 2018 12:10:44 PM IST Jenkins ver. 2.147

- Always secure Jenkins
- In larger systems, don't build on the master
- Backup Jenkins Home regularly
- Limit project names to a sane (e.g. alphanumeric) character set
- Use "file fingerprinting" to manage dependencies
- The most reliable builds will be clean builds, which are built fully from Source Code Control
- Integrate tightly with your issue tracking system, like JIRA or bugzilla, to reduce the need for maintaining a Change Log
- Integrate tightly with a repository browsing tool like FishEye if you are using Subversion as source code management tool
- Always configure your job to generate trend reports and automated testing when running a Java build

- Set up Jenkins on the partition that has the most free disk-space
- Archive unused jobs before removing them
- Setup a different job/project for each maintenance or development branch you create
- Prevent resource collisions in jobs that are running in parallel
- Avoid scheduling all jobs to start at the same time
- Set up email notifications mapping to ALL developers in the project, so that everyone on the team has his pulse on the project's current status
- Take steps to ensure failures are reported as soon as possible
- Write jobs for your maintenance tasks, such as clean up operations to avoid full disk problems
- Tag, label, or baseline the codebase after the successful build

Jenkins Parameterized Builds

The screenshot shows the Jenkins web interface. At the top, the Jenkins logo is on the left, a red tab with the number '2' is in the center, and a search bar with the text 'search' is on the right. Below the header, the breadcrumb 'Jenkins > address-package' is visible. The main content area has a tabbed interface with 'General' selected. Under the 'General' tab, there is a 'Description' field with a text area and a '[Plain text] Preview' link. Below this, there are several checkboxes: 'Discard old builds', 'GitHub project', 'This project is parameterized' (which is checked), 'Throttle builds', 'Disable this project', and 'Execute concurrent builds'. An 'Add Parameter' dropdown menu is open, showing a list of parameter types: 'Boolean Parameter', 'Choice Parameter', 'Credentials Parameter', 'File Parameter', 'List Subversion tags (and more)', 'Multi-line String Parameter', 'Password Parameter', 'Run Parameter', and 'String Parameter'. The 'Boolean Parameter' option is highlighted. To the right of the dropdown, there are three question mark icons. At the bottom right of the configuration area, there is an 'Advanced...' button. Below the configuration area, there are 'Save' and 'Apply' buttons. The browser's address bar at the bottom shows 'localhost:8080/job/address-package/configure#'. The footer of the page contains the URL 'www.cognixia.com'.

Jenkins

2

search

Admin | log out

Jenkins > address-package

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Description

[Plain text] [Preview](#)

☐ Discard old builds

☐ GitHub project

☒ This project is parameterized

Add Parameter

- Boolean Parameter
- Choice Parameter
- Credentials Parameter
- File Parameter
- List Subversion tags (and more)
- Multi-line String Parameter
- Password Parameter
- Run Parameter
- String Parameter

☐ Throttle builds

☐ Disable this project

☐ Execute concurrent builds

Advanced...


Source Code Management

Save Apply


localhost:8080/job/address-package/configure#

www.cognixia.com

Environment Inject Plugin

 **Jenkins**

2

 search

[Admin](#) | [log out](#)

[Jenkins](#) > [Plugin Manager](#)

[Back to Dashboard](#)

[Manage Jenkins](#)

Filter:

Updates

Available

Installed

Advanced

Install ↓	Name	Version
<input type="checkbox"/>	meliara-testlab Publishes results of automated tests to Meliora Testlab, and optionally, injects needed CORS response headers for allowing Testlab -> Jenkins API calls.	1.15
<input checked="" type="checkbox"/>	Environment Injector This plugin makes it possible to set an environment for the builds. Warning: This plugin version may not be safe to use. Please review the following security notices: <ul style="list-style-type: none">Exposure of sensitive build variables stored by EnvInject 1.90 and earlier	2.1.6
<input type="checkbox"/>	EnvInject API Stores shared logic for Environment Injection management	1.5
<input type="checkbox"/>	Message Injector This plugin the ability to inject messages into a Gerrit Trigger message.	0.1.1
<input type="checkbox"/>	Shared Objects This plugin makes it possible to populate as environment variables some objects such as public file paths, Clearcase objects, locations of installed tools and so on. This plugin contributes to the EnvInject plugin. Warning: This plugin version may not be safe to use. Please review the following security notices: <ul style="list-style-type: none">Arbitrary code execution vulnerability in rare circumstances	0.44

Install without restart

Download now and install after restart

Update information obtained: 45 min ago

Check now

Use of Jenkins Environment Variables

- When a Jenkins job executes, it sets some environment variables that you may use in your shell script, batch command, Ant script or Maven POM
- <https://wiki.jenkins.io/display/JENKINS/Building+a+software+project#Buildingasoftwa+reproject-belowJenkinsSetEnvironmentVariables>

Jenkins Environment Variables

Environment Variable	Description
BUILD_NUMBER	The current build number, such as "153"
BUILD_ID	The current build id, such as "2005-08-22_23-59-59" (YYYY-MM-DD_hh-mm-ss, defunct since version 1.597)
BUILD_URL	The URL where the results of this build can be found (e.g. http://buildserver/jenkins/job/MyJobName/666/)
NODE_NAME	The name of the node the current build is running on. Equals 'master' for master node.
JOB_NAME	Name of the project of this build. This is the name you gave your job when you first set it up. It's the third column of the Jenkins Dashboard main page.
BUILD_TAG	String of <code>jenkins-\${JOB_NAME}-\${BUILD_NUMBER}</code> . Convenient to put into a resource file, a jar file, etc for easier identification.
JENKINS_URL	Set to the URL of the Jenkins master that's running the build. This value is used by Jenkins CLI for example
EXECUTOR_NUMBER	The unique number that identifies the current executor (among executors of the same machine) that's carrying out this build. This is the number you see in the "build executor status", except that the number starts from 0, not 1.
JAVA_HOME	If your job is configured to use a specific JDK, this variable is set to the JAVA_HOME of the specified JDK. When this variable is set, PATH is also updated to have <code>\$JAVA_HOME/bin</code> .
WORKSPACE	The absolute path of the workspace.
SVN_REVISION	For Subversion-based projects, this variable contains the revision number of the module. If you have more than one module specified, this won't be set.
CVS_BRANCH	For CVS-based projects, this variable contains the branch of the module. If CVS is configured to check out the trunk, this environment variable will not be set.
GIT_COMMIT	For Git-based projects, this variable contains the Git hash of the commit checked out for the build (like <code>ce9a3c1404e8c91be604088670e93434c4253f03</code>) (all the <code>GIT_*</code> variables require git plugin)
GIT_URL	For Git-based projects, this variable contains the Git url (like <code>git@github.com:user/repo.git</code> or <code>[https://github.com/user/repo.git]</code>)
GIT_BRANCH	For Git-based projects, this variable contains the Git branch that was checked out for the build (normally origin/master)

Project Based Matrix Plugin

Jenkins ▸ Plugin Manager			
<input checked="" type="checkbox"/>	Git plugin This plugin integrates Git with Jenkins.	3.9.1	Uninstall
<input checked="" type="checkbox"/>	HTML Publisher This plugin publishes HTML reports.	1.16	Uninstall
<input checked="" type="checkbox"/>	JDK Tool Allows the JDK tool to be installed via download from Oracle's website.	1.1	Uninstall
<input checked="" type="checkbox"/>	JUnit Plugin Allows JUnit-format test results to be published.	1.26.1	Downgrade to 1.24 Uninstall
<input checked="" type="checkbox"/>	Matrix Authorization Strategy Plugin Offers matrix-based security authorization strategies (global and per-project).	2.3	Uninstall
<input checked="" type="checkbox"/>	Matrix Project Plugin Multi-configuration (matrix) project type.	1.13	Uninstall
<input checked="" type="checkbox"/>	Maven Release Plug-in Plug-in A plug-in that enables you to perform releases using the maven-release-plugin from Jenkins.	0.14.0	Uninstall
<input checked="" type="checkbox"/>	Parameterized Trigger plugin	2.35.2	Uninstall
<input checked="" type="checkbox"/>	PMD Plug-in This plug-in collects the PMD analysis results of the project modules and visualizes the found warnings.	3.50	Uninstall
<input checked="" type="checkbox"/>	Run Condition Define conditions for the execution of build steps	1.0	Uninstall
<input checked="" type="checkbox"/>	Script Security Allows Jenkins administrators to control what in-process scripts can be run by less-privileged users.	1.44	Uninstall
<input checked="" type="checkbox"/>	Static Analysis Utilities This plug-in provides utilities for the static code analysis plug-ins.	1.95	Uninstall

<https://wiki.jenkins-ci.org/display/JENKINS/Matrix+Project+Plugin>

Configuring Jenkins Hub and Node in the cloud (AWS)

Jenkins 2

Jenkins > Nodes >

- [Back to Dashboard](#)
- [Manage Jenkins](#)
- [New Node](#)
- [Configure](#)

S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	master	Windows 10 (x86)	In sync	717.26 GB	4.05 GB	717.26 GB	0ms
	Data obtained	53 min	53 min	53 min	53 min	53 min	53 min

[Refresh status](#)

Build Queue

No builds in the queue.

Build Executor Status

- Idle
- Idle

Case Study

- Follow Lab Exercise 2 for the complete Case Study



THANK YOU