

Jenkins Overview



Jenkins is an open-source DevOps tool that has been popularly used for continuous integration and continuous delivery processes. It is a Java-based application and platform-independent. It is a build tool; used for running builds from the source code repository, running unit tests, and sending the build reports to the respective member or team.

This CI server runs in servlet containers such as Apache Tomcat. Jenkins facilitates [continuous integration and continuous delivery](#) in software projects by automating parts related to build, test, and deployment. This makes it easy for developers to continuously work on the betterment of the product by integrating changes to the project.

Jenkins automates the software builds in a continuous manner and lets the developers know about the errors at an early stage. A strong Jenkins community is one of the prime reasons for its popularity. Jenkins is not only extensible but also has a thriving plugin ecosystem.

Some of the possible steps that can be performed using Jenkins are:

- Software build using build systems such as Gradle, Maven, and more.
- Automation testing using test frameworks such as Nose2, PyTest, Robot, Selenium, and more.
- Execute test scripts (using Windows terminal, Linux shell, etc).
- Achieve test results and perform post actions such as printing test reports, and more.
- Execute test scenarios against different input combinations for obtaining improved test coverage.
- Continuous Integration (CI) where the artifacts are automatically created and tested. This aids in identification of issues in the product at an early stage of development.

Advantages of using Jenkins are:

- It is a cross-platform and can be used on Windows, Linux, Mac OS, and Solaris environments
- It is a free and open source tool
- Widely used and well documented
- Integration with a wide variety of tool and technologies
- Change Support: Jenkins generates the list of all changes done in repositories like SVN.
- Permanent links: Jenkins provides direct links to the latest build or failed build that can be used for easy communication
- Installation: Jenkins is easy to install either using direct installation file (exe) or war file to deploy using application server.
- Email integration: Jenkins can be configured to email the content of the status of the build.
- Easy Configuration: To configure various tasks on Jenkins is easy.
- TestNG test: Jenkins can be configured to run the automation test build on Testng after each build of SVN.
- Multiple VMs: Jenkins can be configured to distribute the build on multiple machines.
- Project build: Jenkins documents the details of jar, version of jar and mapping of build and jar numbers.
- Plugins: 3rd party plugin can be configured in Jenkins to use features and additional functionality.
-

Apart from Jenkins, we have many more tools in the market such as:

- Anthill
- Bamboo
- Cruise Control
- Team City and many more.

Why Jenkins and Selenium?

- Running Selenium tests in Jenkins allows you to run your tests every time your software changes and deploy the software to a new environment when the tests pass.
- Jenkins can schedule your tests to run at specific time.
- You can save the execution history and Test Reports.
- Jenkins supports Maven for building and Testing a project in continuous integration.

Following steps should be followed so that to install Jenkins successfully:

- **Step 1)** Got to <https://www.jenkins.io/download/> and select the platform. In our case Windows

Deploy Jenkins 2.121.1



Deploy to Azure

Download Jenkins 2.121.1 for:

Docker

FreeBSD

Gentoo

Mac OS X

OpenBSD

openSUSE

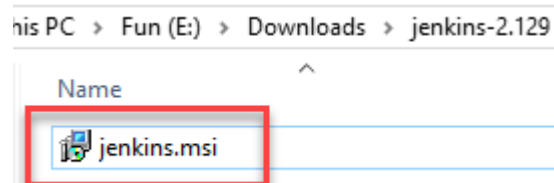
Red Hat/Fedora/CentOS

Ubuntu/Debian

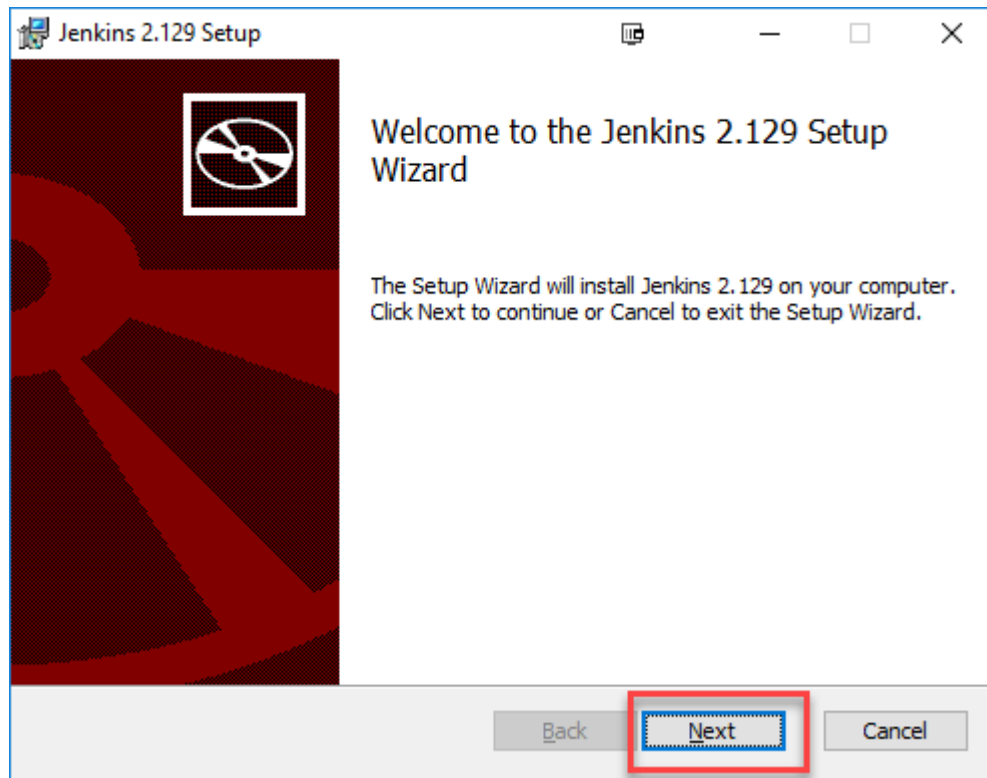
Windows

Generic Java package (war)

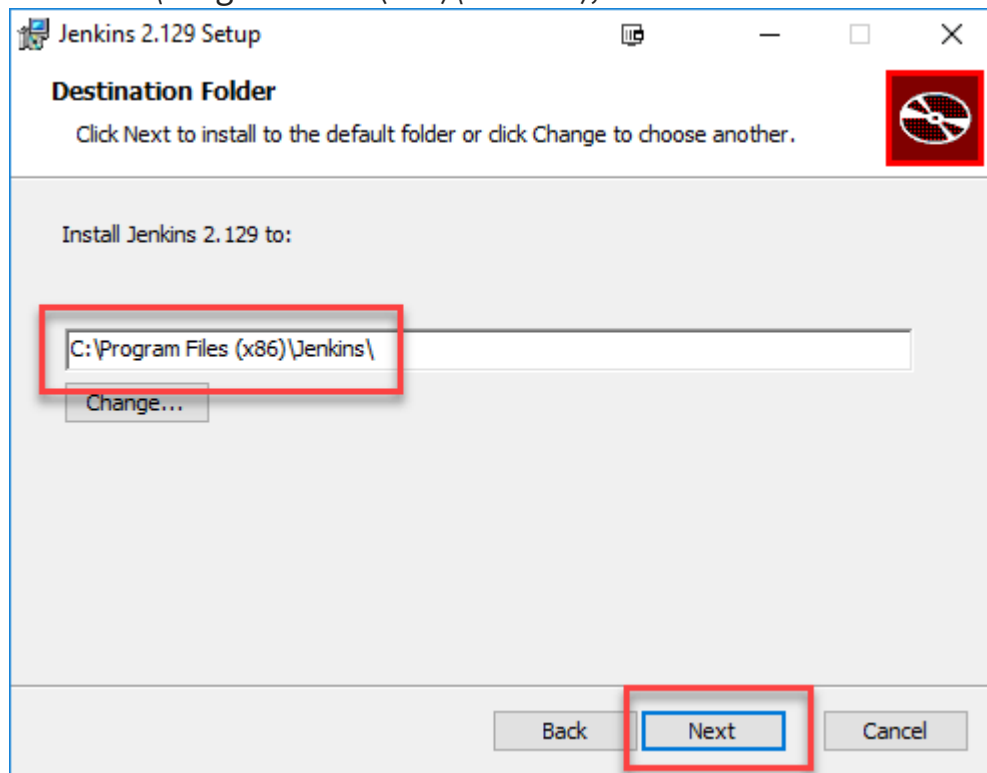
- **Step 2)** Go to download location from local computer and unzip the downloaded package. Double-click on unzipped **jenkins.msi**. You can also Jenkins using a WAR (Web application ARchive) but that is not recommended.



- **Step 3)** In the Jenkin Setup screen, click Next.



- **Step 4)** Choose the location where you want to have the Jenkins instance installed (default location is C:\Program Files (x86)\Jenkins), then click on **Next** button.



Jenkins 2.249.2 Setup

Service Logon Credentials

Enter service credentials for the service.

Jenkins 2.249.2 installs and runs as an independent Windows service. To operate in this manner, you must supply the user account credentials for Jenkins 2.249.2 to run successfully.


Logon Type:

☐ Run service as LocalSystem (not recommended)

☒ Run service as local or domain user:

Account:

Password:

 Credentials must be tested to continue

Select 1st radio button


Jenkins 2.303.1 Setup

Port Selection

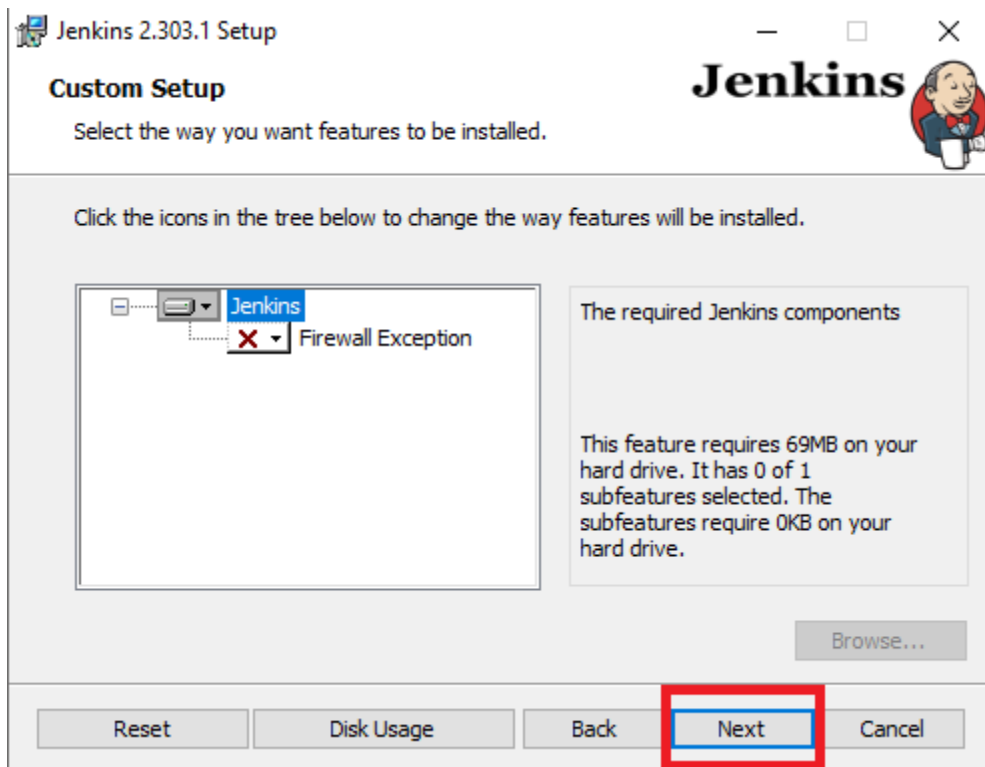
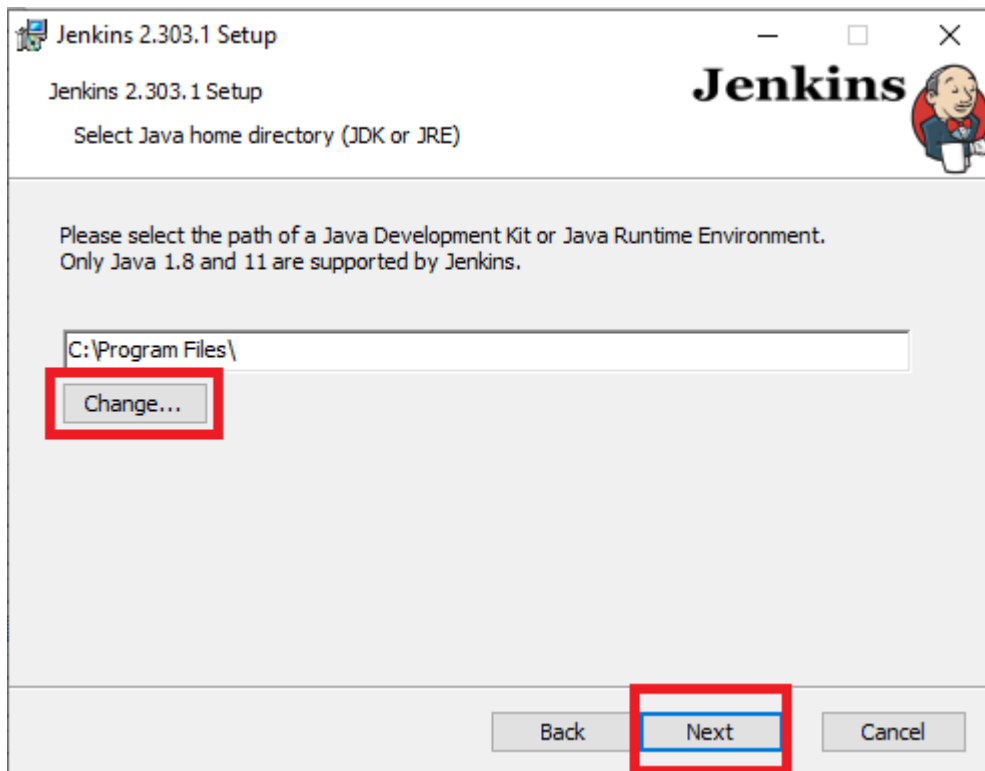
Choose a port for the service.

Please choose a port.

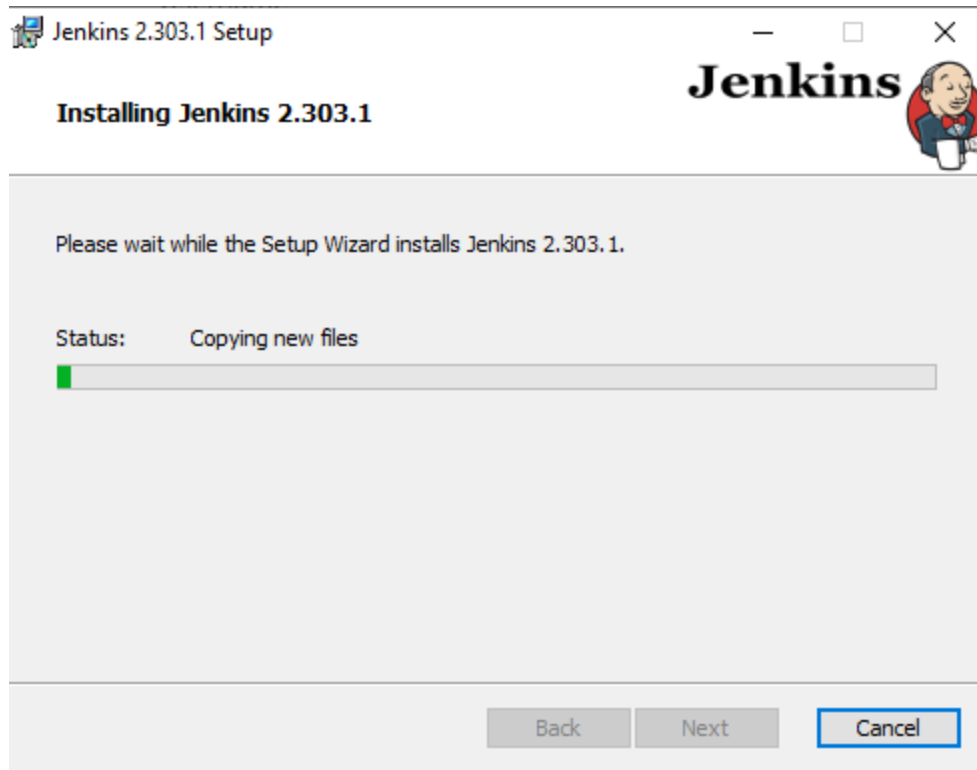
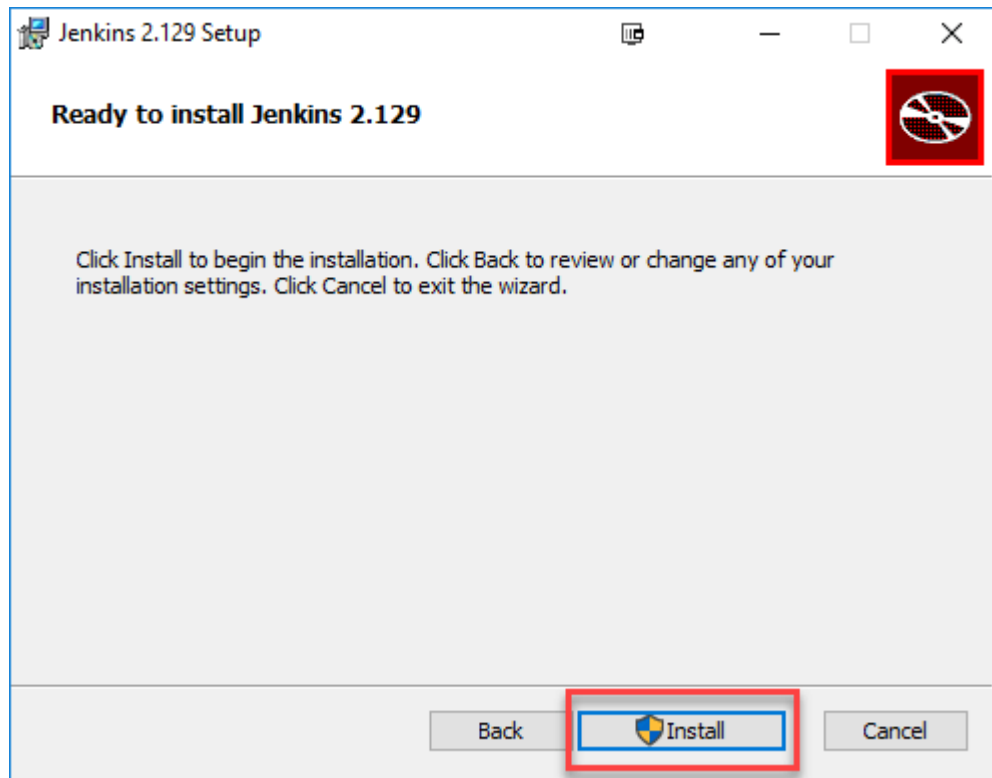
Port Number (1-65535):



It is recommended that you accept the selected default port.



- **Step 5)**Click on the Install button.



- **Step 6)** Once install is complete, click Finish.

Completed the Jenkins 2.303.1 Setup Wizard

Click the Finish button to exit the Setup Wizard.



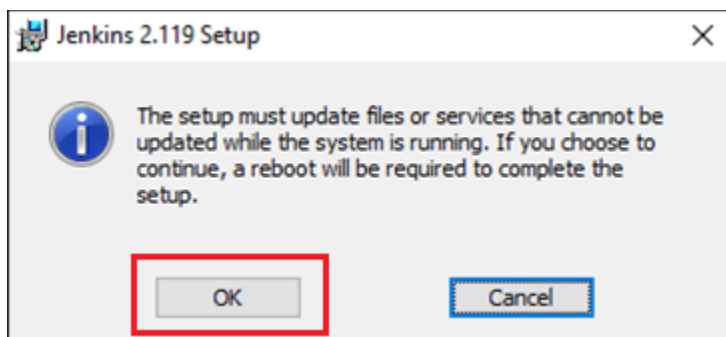
Back

Finish

Cancel

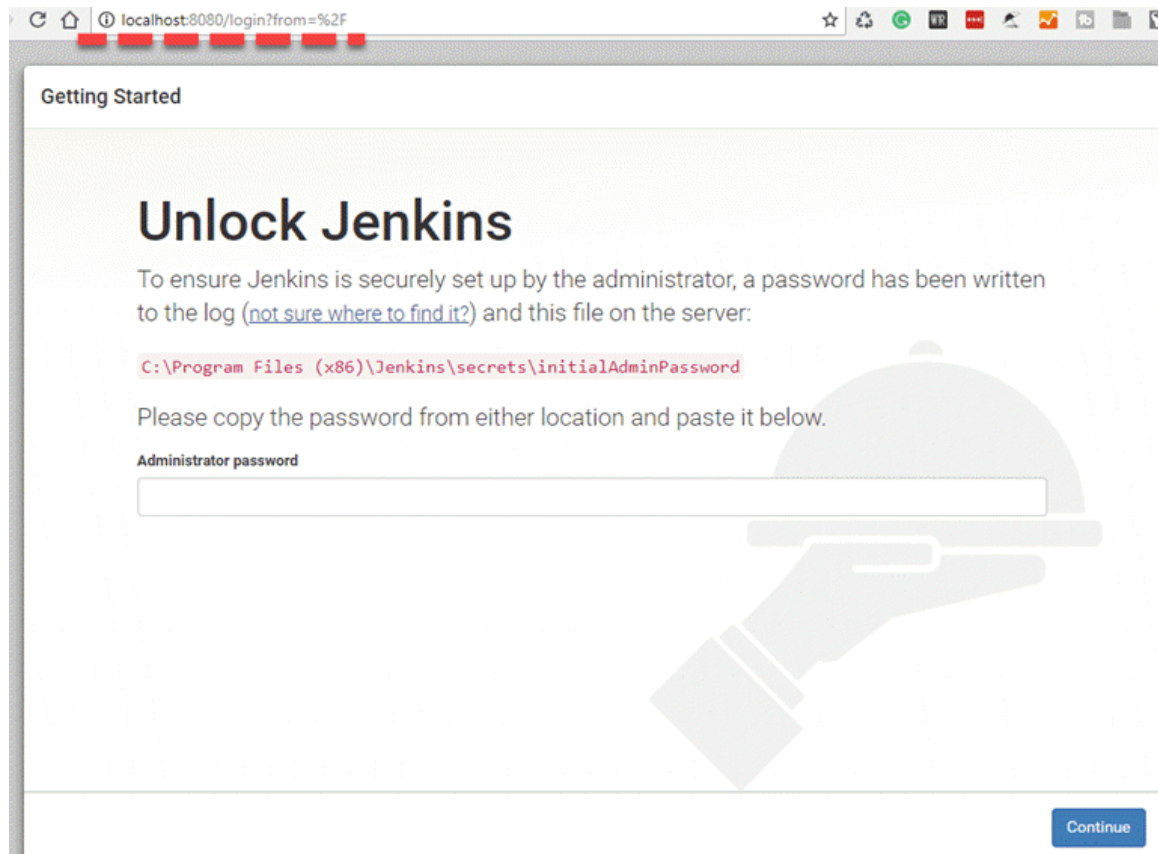
X

- **Step 7)** During the installation process an info panel may pop-up to inform the user that for a complete setup, the system should be rebooted at the end of the current installation. Click on OK button when the Info panel is popping-up:

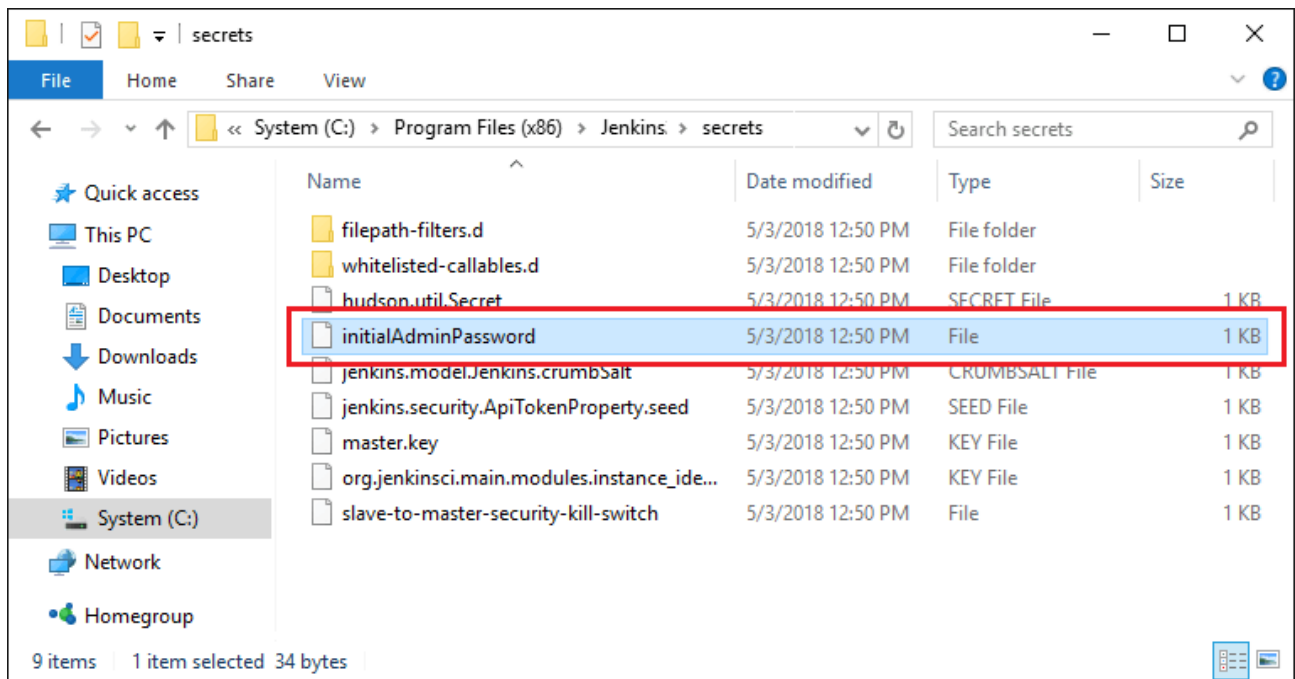


How to Unblock Jenkins?

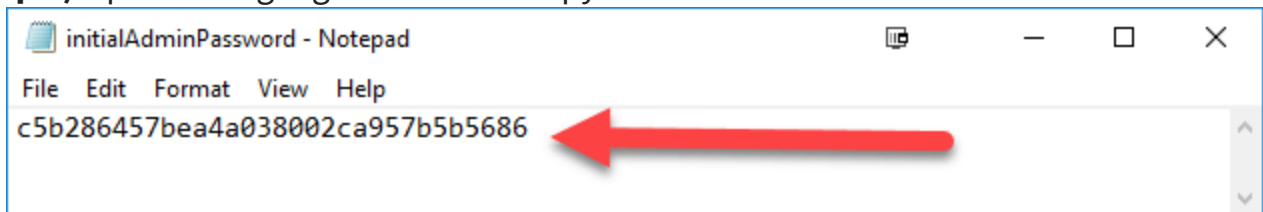
- After completing the Jenkins installation phase, you should proceed further and start its configuration. Next steps will guide you how you can unblock Jenkins application:
- **Step 1)** After completing the Jenkins installation process, a browser tab will pop-up asking for the initial Administrator password. To access Jenkins, you need to go to browse the following path in your web browser.
- <http://localhost:8080>
- If you can access the above URL, then it confirms that Jenkins is successfully installed in your system.



- **Step 2)** The initial Administrator password should be found under the Jenkins installation path (set at Step 4 in Jenkins Installation).
- For default installation location to C:\Program Files (x86)\Jenkins, a file called **initialAdminPassword** can be found under C:\Program Files (x86)\Jenkins\secrets.
- However, If a custom path for Jenkins installation was selected, then you should check that location for **initialAdminPassword** file.



- **Step 3)** Open the highlighted file and copy the content of the **initialAdminPassword** file.

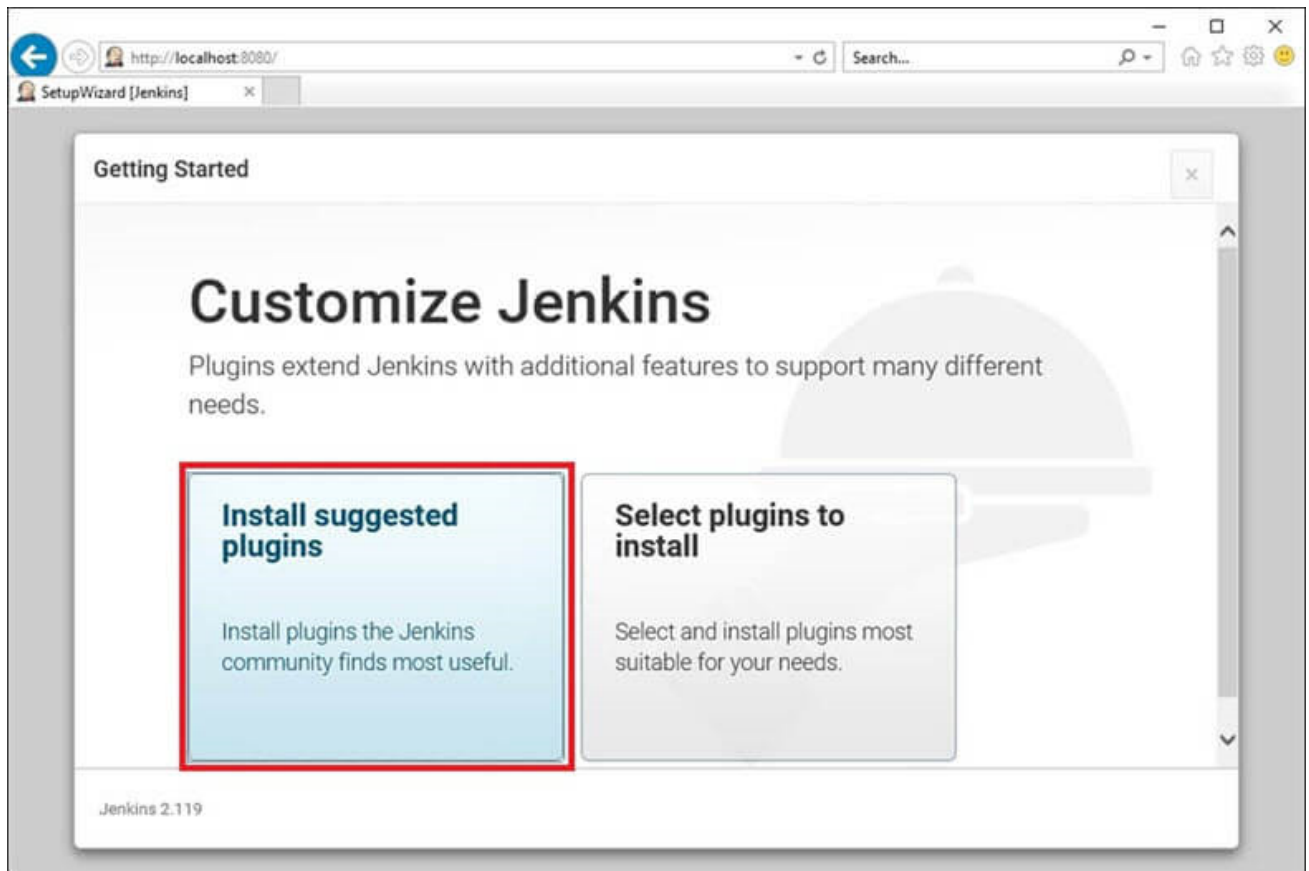


- **Step 4)** Paste the password it into browser's pop-up tab (<http://localhost:8080/login?form=%2F>) and click on Continue button.

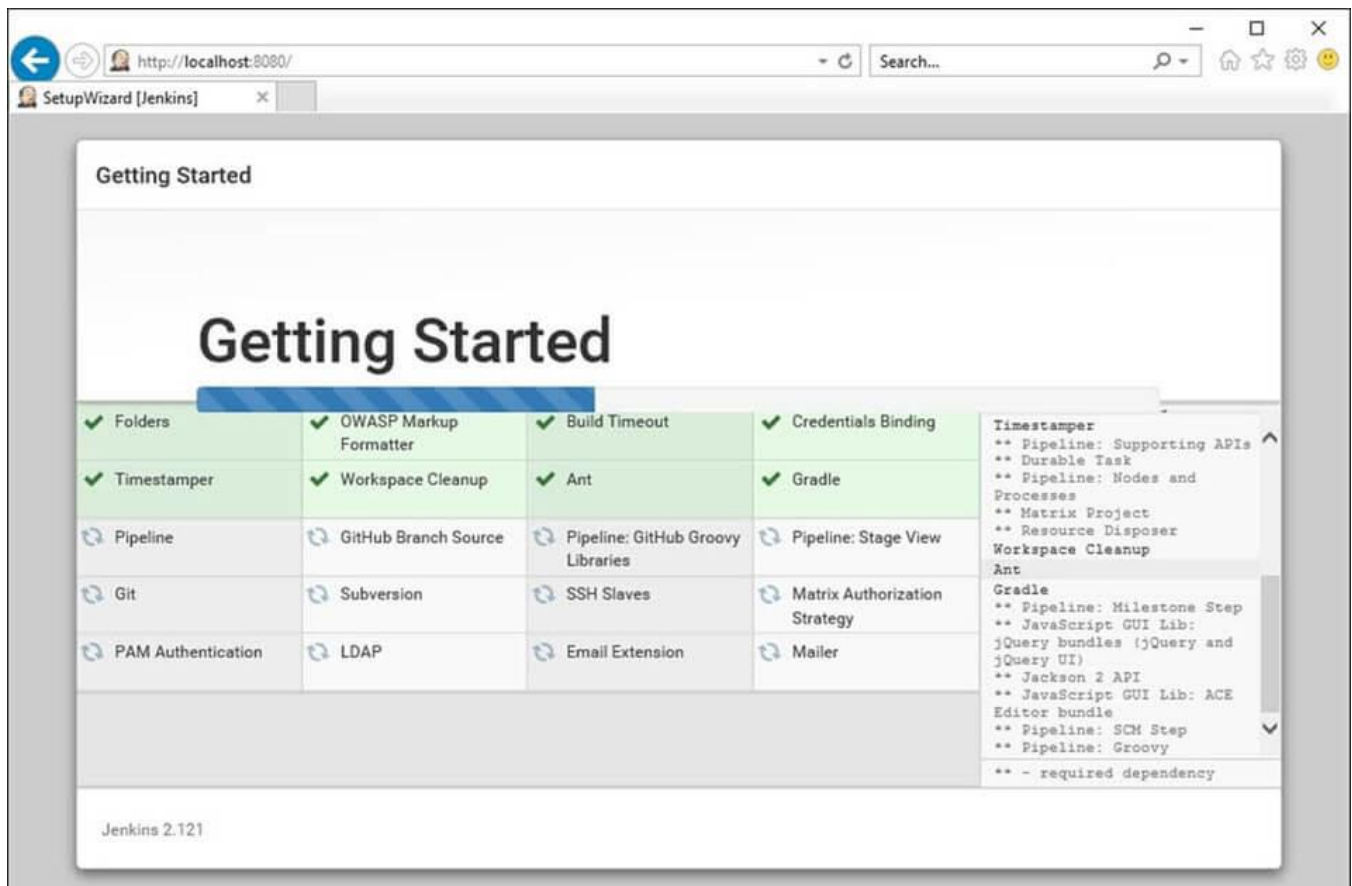


Customize Jenkins

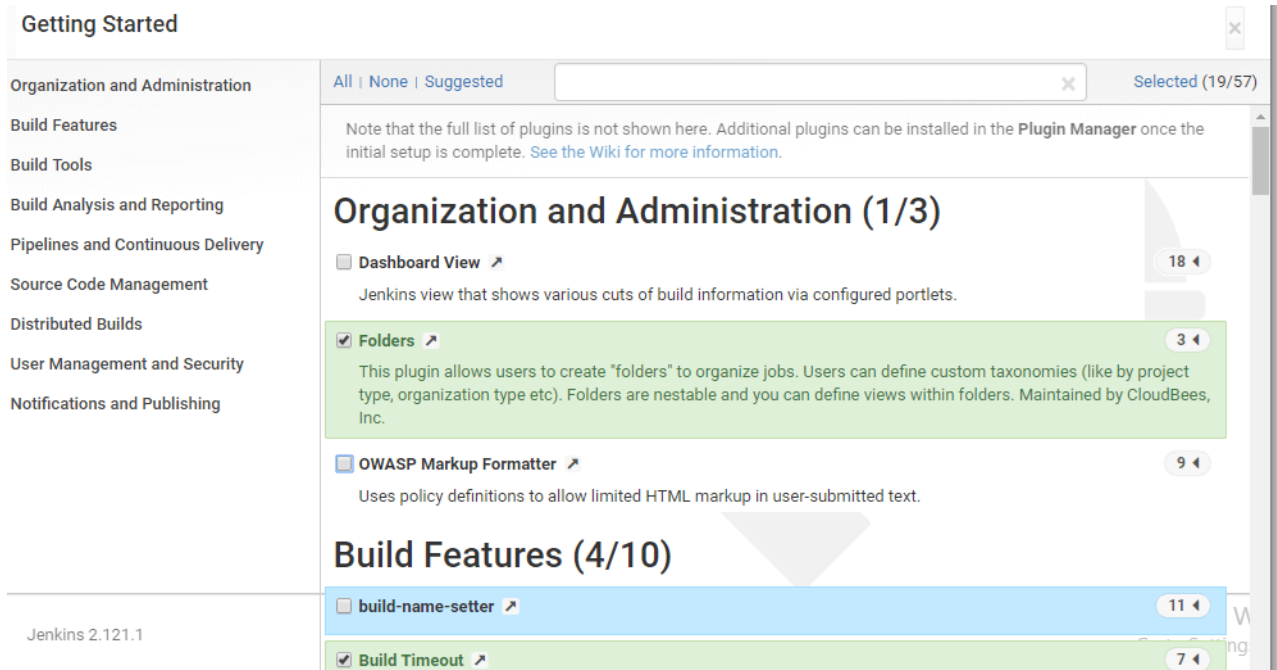
- You can also customize your Jenkins environment by below-given steps:
- **Step 1)** Click on the “Install suggested plugins button” so Jenkins will retrieve and install the essential plugins



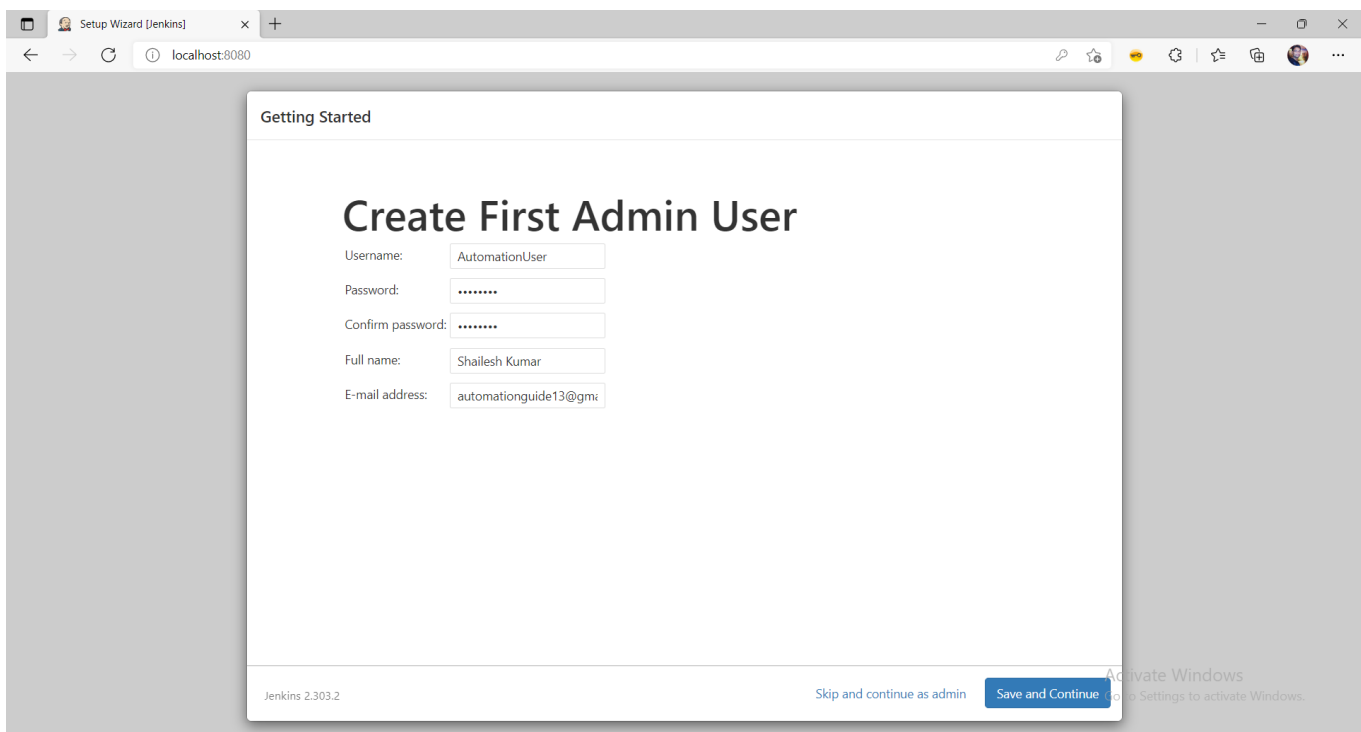
- Jenkins will start to download and install all the necessary plugins needed to create new Jenkins Jobs.



- **Note:** You can choose the Option “Select Plugins to Install” and select the plugins you want to install

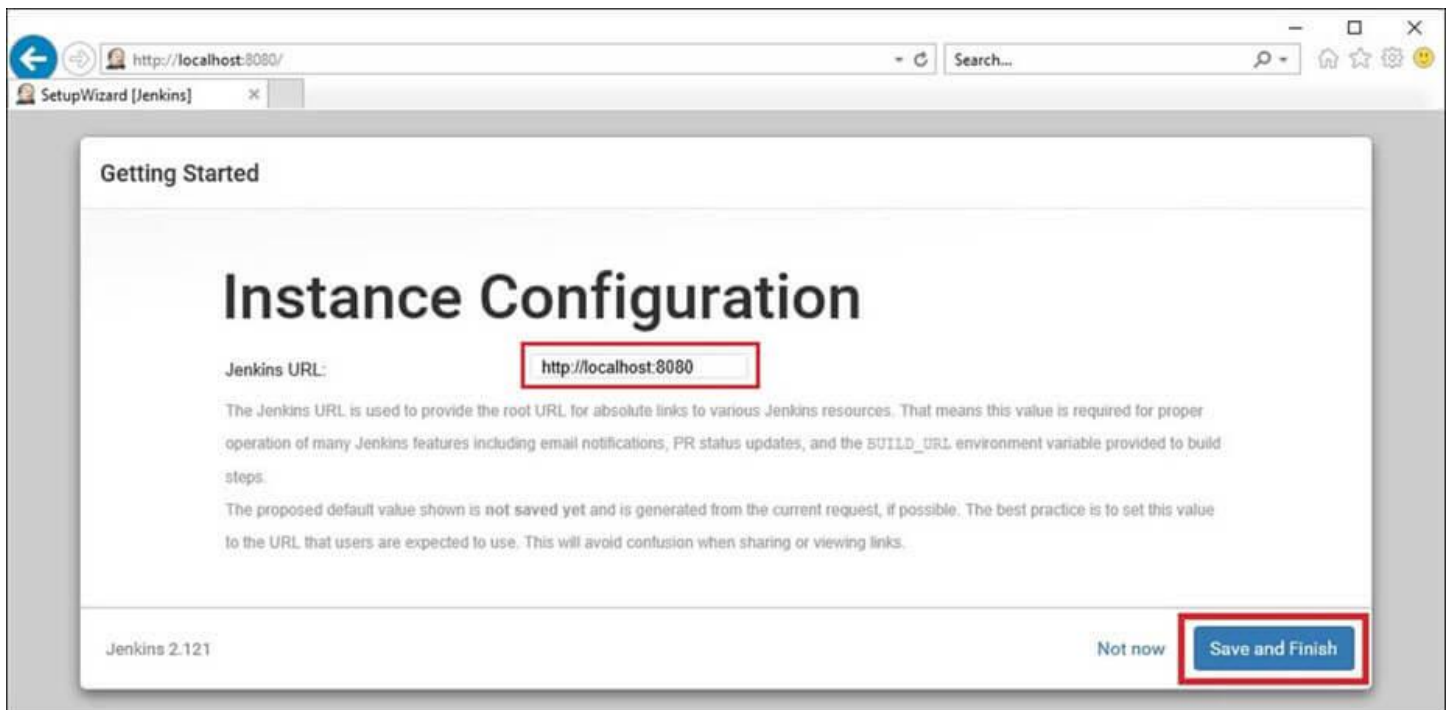


- **Step 2)** After all suggested plugins were installed, the “Create First Admin User” panel will show up. Fill all the fields with desired account details and hit the “**Save and Finish**” button.

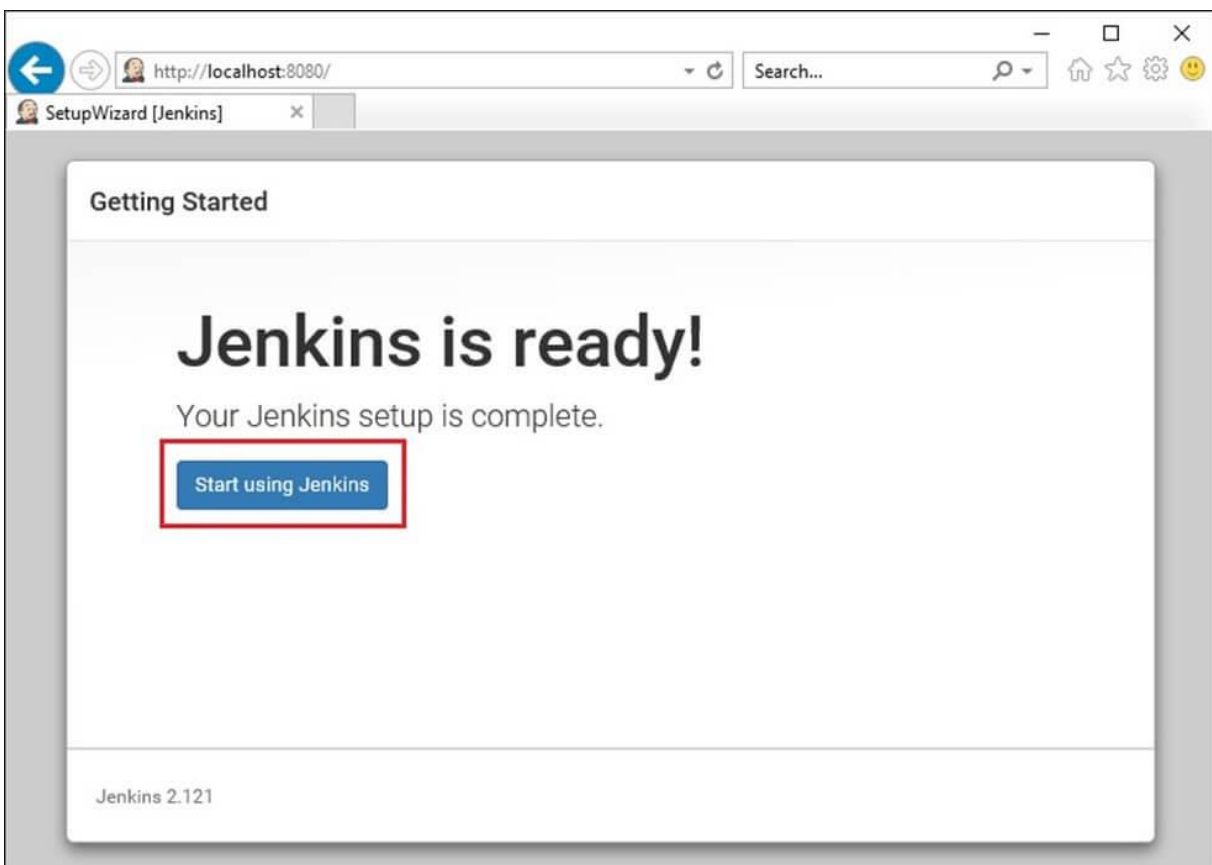


Step 3) Once you have filled the above data, finally it will ask for URL information where you can configure the default instance path for Jenkins. Leave it as it is to avoid any confusions later. However, if another application is already using 8080 port, you can use another port for Jenkins

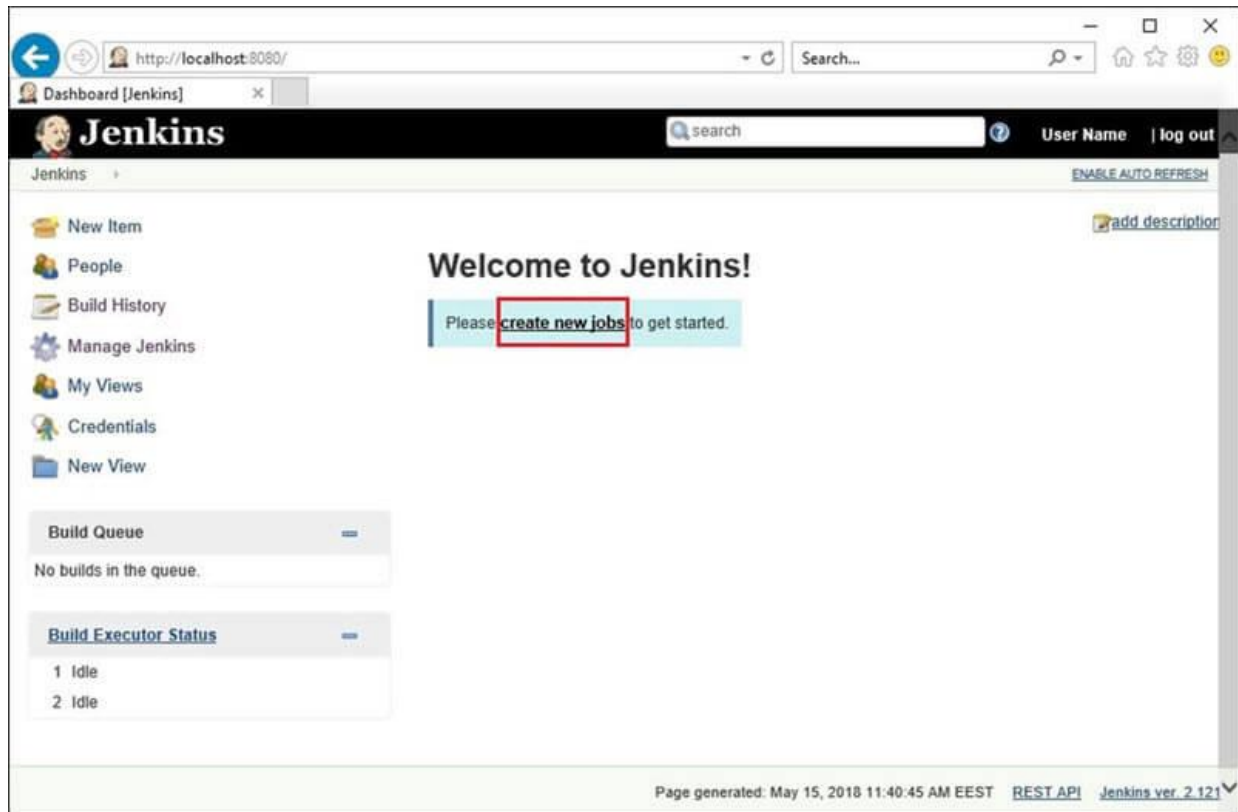
and finally save the settings, and you are done with installation of Jenkins. Hit the “**Save and Continue**” button:



Congratulations! We have successfully installed a new Jenkins Server. Hit the “Start using Jenkins” button.

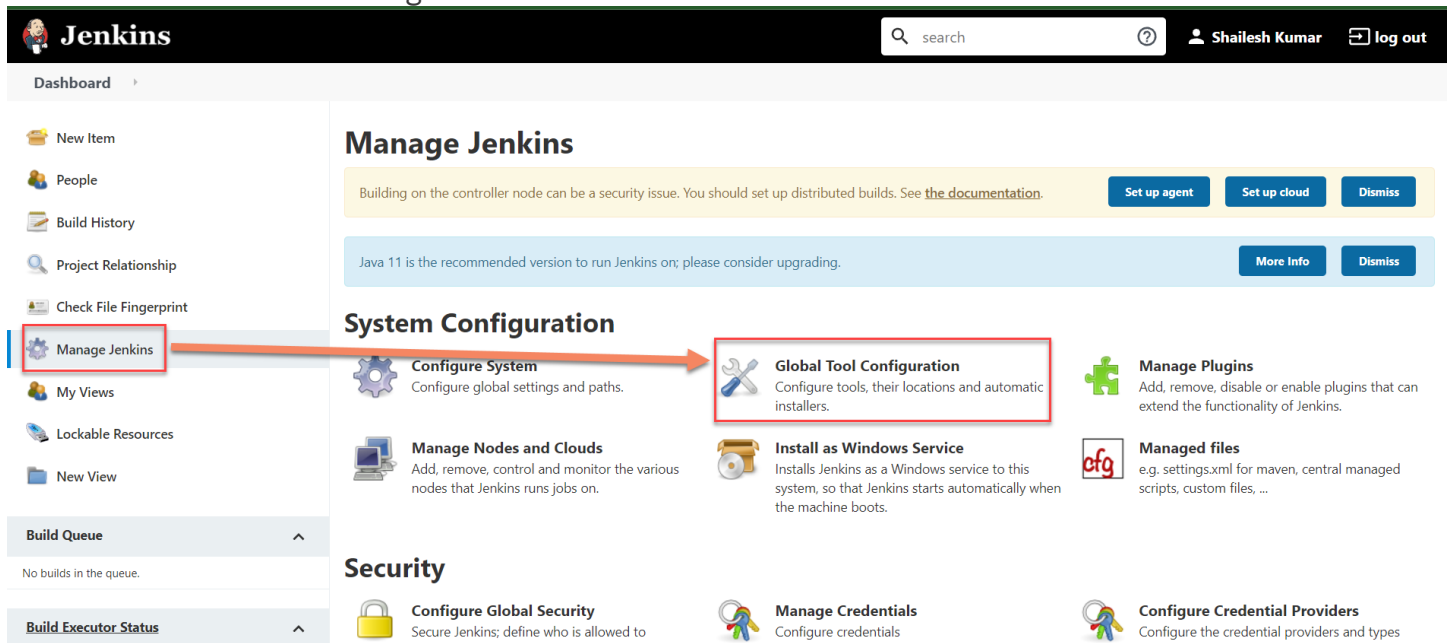


Below you can find the Jenkins instance up and run, ready to create first Jenkins jobs:



Follow the below steps:

- Go to Jenkins dashboard
- Click on manage Jenkins
- Click on configure Jenkins



- Click on JDK installation – In JDK name section enter the name, under Java Home section – give your java path

JDK installations

JDK Name

JAVA_HOME

C:\Program Files\Java\jdk1.8.0_66

☐ Install automatically

Add JDK

Delete JDK

List of JDK installations on this system

- Click on Maven installation – In Maven name section enter the name as “MAVEN_HOME”, under Java Home section – give your Maven path, apply and save.

Dashboard > Global Tool Configuration

Add Gradle

List of Gradle installations on this system

Ant

Ant installations

Add Ant

List of Ant installations on this system

Maven

Maven installations

Add Maven

Maven

Name

MAVEN_HOME

MAVEN_HOME

C:\apache-maven-3.8.3

☐ Install automatically

Add Maven

Delete Maven

Activate Windows

Jenkins Plugin needs to added before start

Open Jenkins-> Manage Jenkins ->Manage Plugin -> click on available tab -> search below plugins and click on “install without restart”

1. Maven Integration
2. TestNG Results
3. GitHub Integration
4. Green Ball

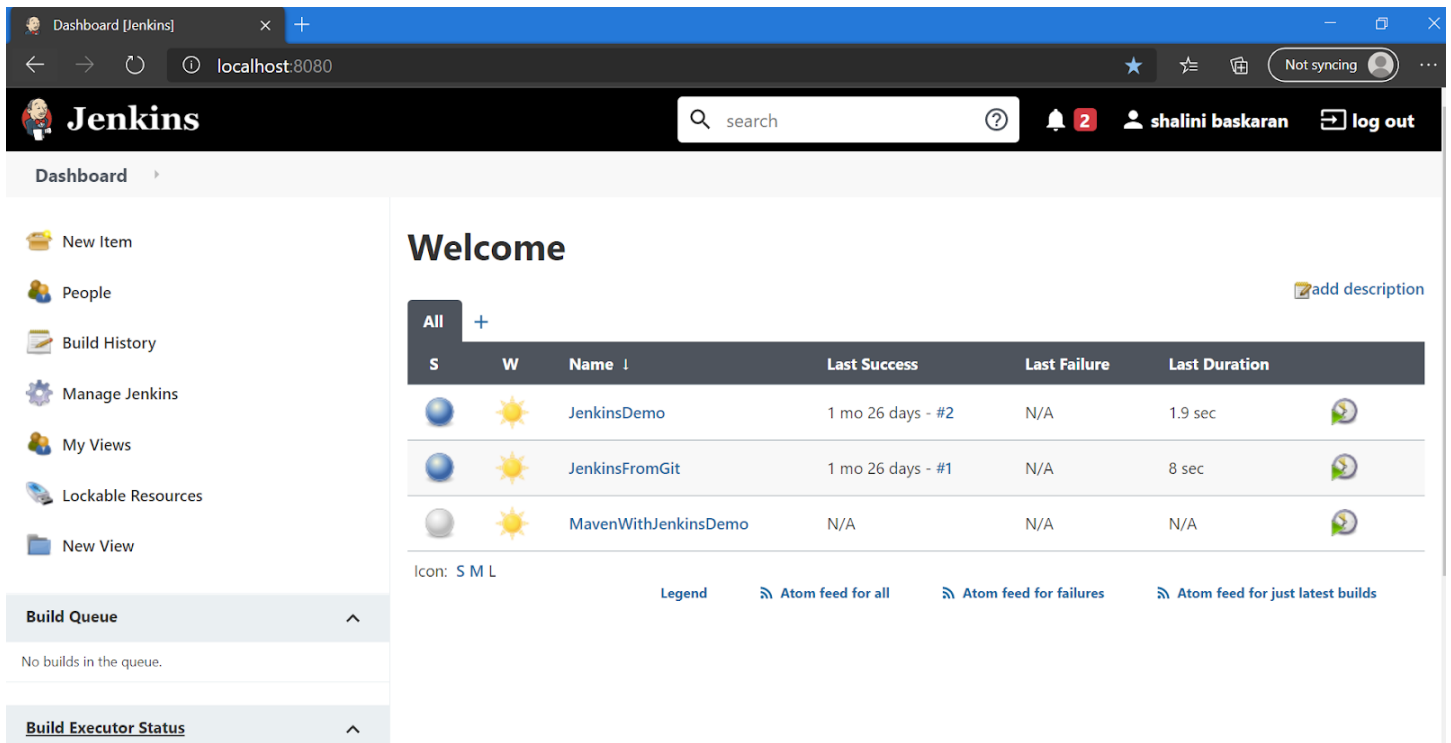
How To Integrate Selenium Tests In Maven With Jenkins?

We saw how to integrate Jenkins with Selenium WebDriver in the sections above. The best use case of this integration is to implement it for Selenium test automation. Jenkins integration with Selenium

WebDriver makes your [cross browser testing](#) scripts sturdier than ever. In this section, I will explain how to integrate Jenkins with Selenium test scripts with the help of Maven.

Step 1: Start the Jenkins server.

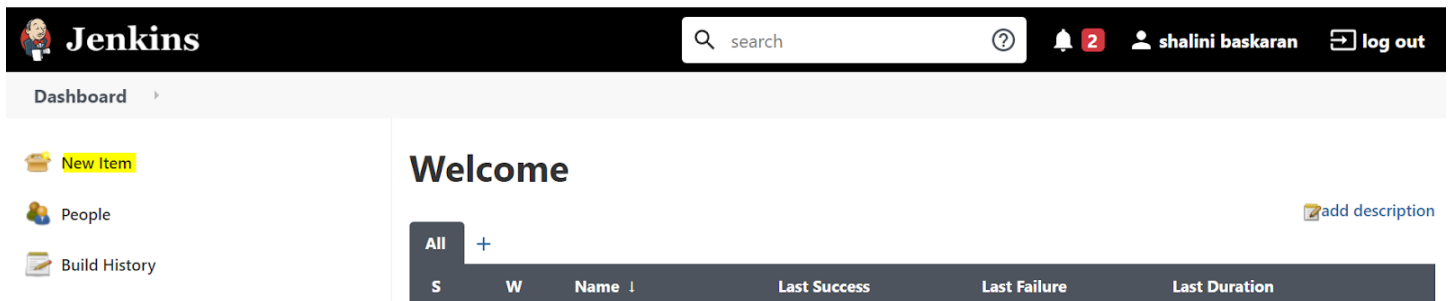
Step 2: Open the browser and navigate to the localhost and the port in which Jenkins is running.



The screenshot shows the Jenkins Dashboard in a web browser. The browser's address bar displays 'localhost:8080'. The Jenkins logo and name are at the top left. A search bar and user information 'shalini baskaran' with a 'log out' button are at the top right. The left sidebar contains a 'Dashboard' menu and a list of links: 'New Item', 'People', 'Build History', 'Manage Jenkins', 'My Views', 'Lockable Resources', and 'New View'. The main content area is titled 'Welcome' and features a table of builds. The table has columns for 'S' (Status), 'W' (Icon), 'Name', 'Last Success', 'Last Failure', and 'Last Duration'. There are three rows of builds: 'JenkinsDemo', 'JenkinsFromGit', and 'MavenWithJenkinsDemo'. Below the table, there are links for 'Icon: S M L', 'Legend', and three Atom feed links: 'Atom feed for all', 'Atom feed for failures', and 'Atom feed for just latest builds'. The 'Build Queue' section shows 'No builds in the queue.' and the 'Build Executor Status' section is also visible.

S	W	Name ↓	Last Success	Last Failure	Last Duration
		JenkinsDemo	1 mo 26 days - #2	N/A	1.9 sec
		JenkinsFromGit	1 mo 26 days - #1	N/A	8 sec
		MavenWithJenkinsDemo	N/A	N/A	N/A

Step 3: Click **New Item** in the dashboard.



This screenshot is similar to the previous one, but the 'New Item' link in the left sidebar is highlighted with a yellow background. The rest of the dashboard content remains the same.

Step 4: Enter the project name and select the project type as **Maven project**.

Enter an item name

» This field cannot be empty, please enter a valid name



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.

Step 5: Click **Ok**. Now you could see a job being created successfully in the dashboard.

Jenkins

search

?

2

shalini baskaran

log out

Dashboard

New Item

People

Build History

Manage Jenkins

My Views

Lockable Resources

New View

Build Queue

Welcome

add description

All

+

S	W	Name ↓	Last Success	Last Failure	Last Duration	
		JenkinsDemo	1 mo 11 days - #2	N/A	1.9 sec	
		JenkinsFromGit	1 mo 11 days - #1	N/A	8 sec	
		MavenWithJenkinsDemo	N/A	N/A	N/A	

Icon: S M L

Legend

Atom feed for all

Atom feed for failures

Atom feed for just latest builds

Step 6: Click the project and click **Configure**.

The screenshot shows the Jenkins web interface. At the top is the Jenkins logo and a search bar. Below the navigation bar, the breadcrumb trail reads 'Dashboard > MavenWithJenkinsDemo >'. On the left sidebar, the 'Configure' option (represented by a gear icon) is highlighted with a red rectangular box. Other sidebar options include 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Maven project', 'Modules', and 'Rename'. The main content area is titled 'Maven project MavenWithJenkinsDemo' and contains links for 'Workspace' and 'Recent Changes', along with a 'Permalinks' section.

Step 7: Under the Build section, enter the complete path of your pom.xml . In the Goals and options, enter the command clean test.

This screenshot displays the 'Build' configuration tab within the Jenkins project settings for 'MavenWithJenkinsDemo'. The tab is highlighted with a white background. Below the tab navigation (General, Source Code Management, Build Triggers, Build Environment, Pre Steps, Build, Post Steps, Build Settings), there is a 'Post-build Actions' section. Within this section, the 'Root POM' field is populated with the file path 'C:\Users\Shalin\IdeaProjects\First\pom.xml'. The 'Goals and options' field contains the text 'clean test'. To the right of each input field is a blue circular help icon. At the bottom right of the configuration area, there is a button labeled 'Advanced...'.

Step 8: Click **Apply** and then **Save**.

Step 9: Click **Build Now**.

Jenkins

search

?

2

shalini baskaran

log out

Dashboard > MavenWithJenkinsDemo >

Back to Dashboard

Status

Changes

Workspace

Build Now

Configure

Delete Maven project

Modules

Rename

Maven project MavenWithJenkinsDemo

add description

Disable Project

Workspace

Recent Changes

Permalinks

Step 10: Now the build will run and, after successful completion of the build, the results would be displayed. To view the complete logs, click the console output.

JenkinsDemo #3 Console [Jenkin...]

localhost:8080/job/JenkinsDemo/3/console

?

2

shalini baskaran

log out

Dashboard > JenkinsDemo > #3

Back to Project

Status

Changes

Console Output

Console Output

Started by user shalini baskaran

Running as SYSTEM

Building in workspace C:\Windows\system32\config\systemprofile\AppData\Local\Jenkins.jenkins\workspace\JenkinsDemo

[JenkinsDemo] \$ cmd /c call C:\Windows\TEMP\jenkins5665269370162719102.bat

C:\Windows\system32\config\systemprofile\AppData\Local\Jenkins.jenkins\workspace\JenkinsDemo>cd