**MAVEN JENKINS INTEGRATION**

**What is Jenkins?**

Jenkins is the leading open-source continuous integration tool developed by Hudson lab. It is cross-platform and can be used on Windows, Linux, Mac OS and Solaris environments. Jenkins is written in Java. It fires pre-configured actions when a particular step occurs in jobs.

**Important Features of Jenkins**

* Change Support: Jenkins generates the list of all changes done in repositories like SVN.
* 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.
* Plugins: 3rd party plugin can be configured in Jenkins to use features and additional functionality.

**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.

**Steps to Install Jenkins and configure it to Run Maven with TestNg Selenium**

**Installation**

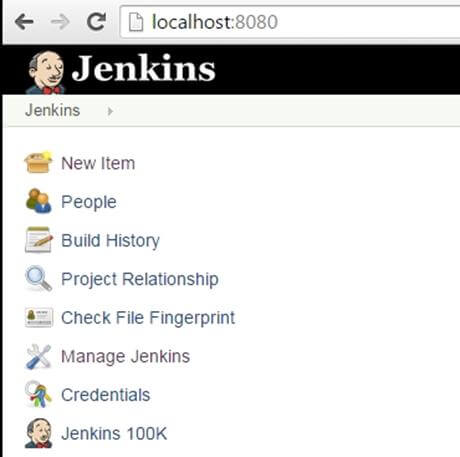
**Step 1)** Go to <https://www.jenkins.io/download/> and download correct package for your OS. Install Jenkins.

**Step 2)**Download the .war file as per your System at your specified location.

**Step 3)**Open cmd >Go the folder where jar is installed using cd ‘path of folder where Jenkins is installed’ and hit enter.

**Step 4)**Enter the command ‘java -jar jenkins.war’ and wait for the setup to complete and a default password will be created for you as an user> Note/Save the password.

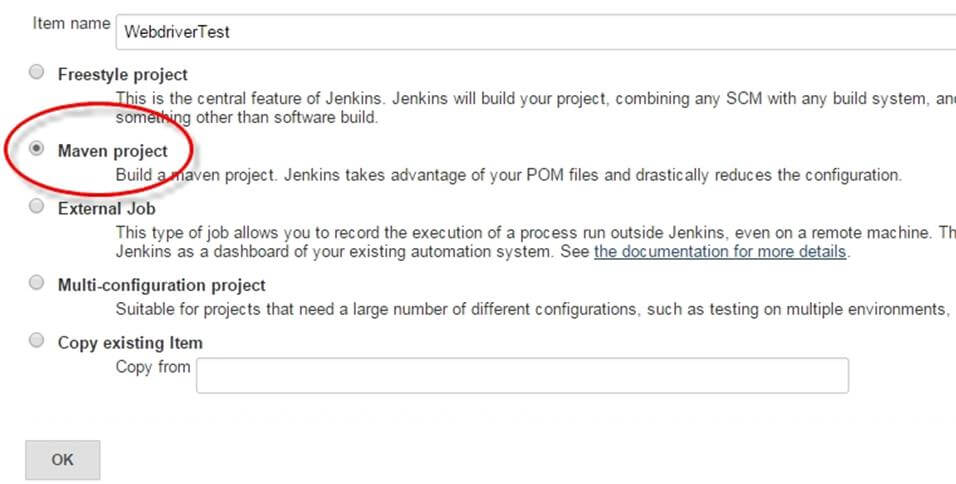
**Step 5)**Once installation is done, navigate to the Jenkins Dashboard (http://localhost:8080 by default) in the browser window after setting up your account with default setting using the password which was generated in the above step.

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin21.jpg)

**Step 6)**Click on the **New Item** link to create a CI job.

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin22.jpg)

**Step 7)**Select the Maven project radio button as shown in the following screenshot:

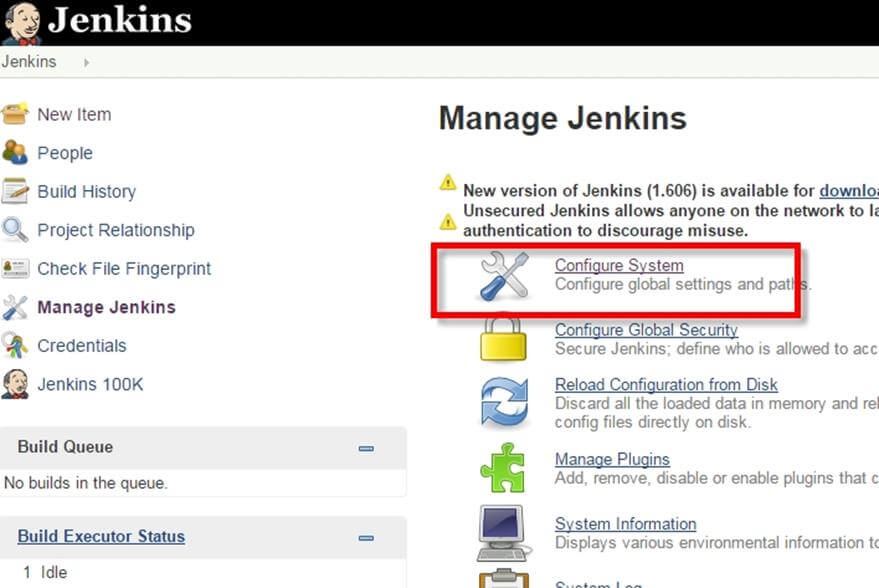
[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin23.jpg)

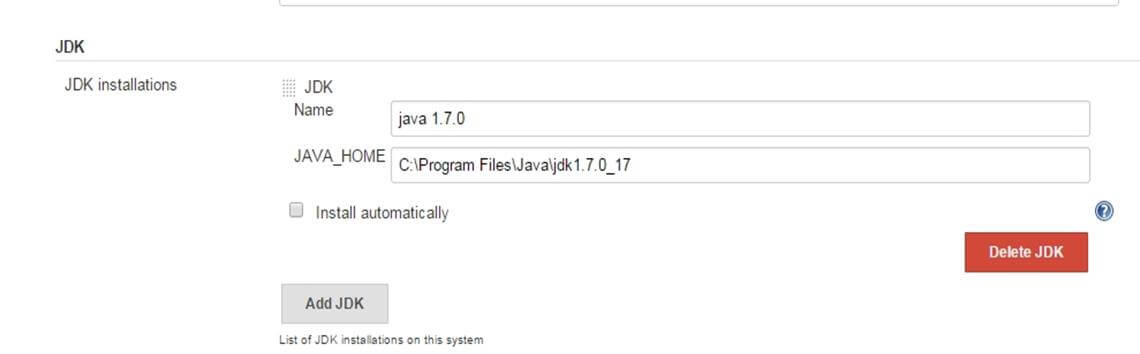
Using the Build a **Maven Project** option, Jenkins supports building and testing Maven projects.

**Step 6)**Click on OK button. A new job with name "WebdriverTest" is created in Jenkins Dashboard.

[Maven & Jenkins with Selenium: Complete Tutorial](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin24.jpg)

**Step 7)** Go to **Manage Jenkins** => **Configure System** as shown in the following screenshot.

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin25.jpg)

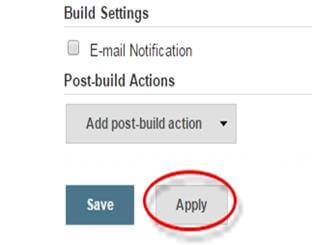
Click on JDK installations and configure JDK as in the following screenshot:  
  
[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin26.jpg)

**Step 8)**Go to the **Build** section of new job.

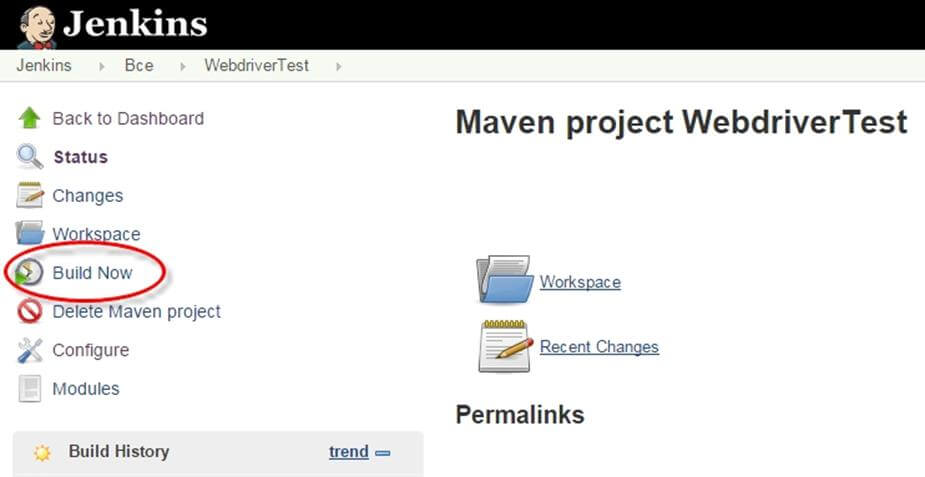
* In the **Root POM** textbox, enter full path to pom.xml
* In Goals and options section, enter "clean test"

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin27.jpg)

**Step 9)**Click on **Apply** button.

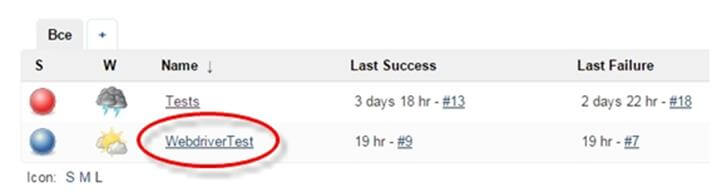
[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin28.jpg)

**Step 10)**On the WebdriverTest project page, click on the **Build Now** link.

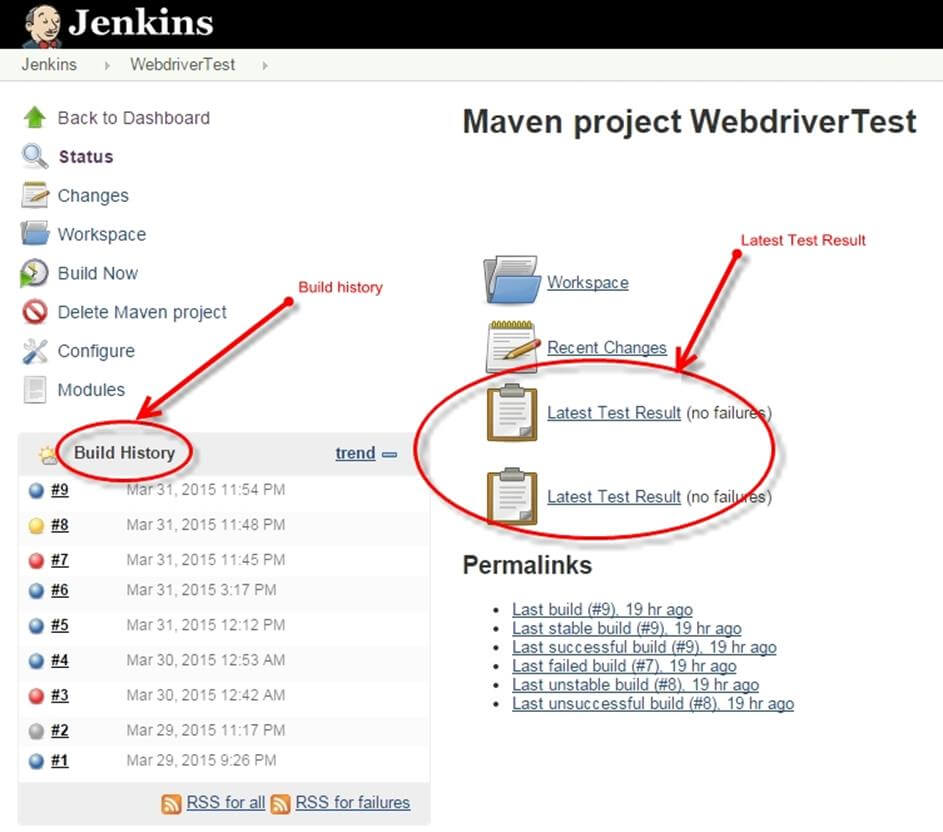
[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin29.jpg)

Maven will build the project. It will then have TestNG execute the test cases.

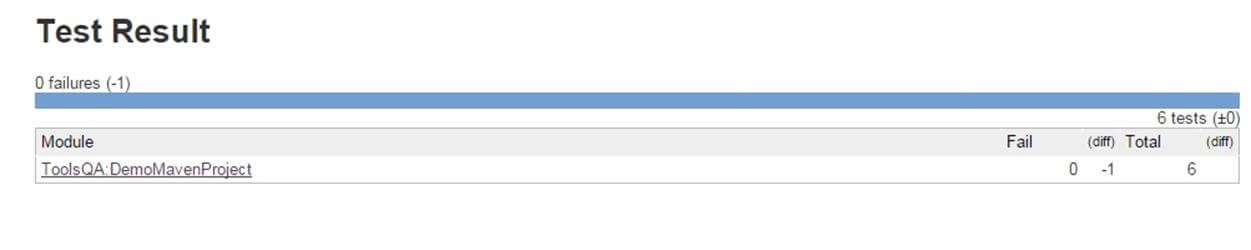
**Step 11)**Once the build process is completed, in Jenkins Dashboard click on the **WebdriverTest** project

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin30.jpg)

**Step 12)**The WebdriverTest project page displays the build history and links to the results as shown in the following screenshot:

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin31.jpg)

**Step 13)**Click on the "Latest Test Result" link to view the test results as shown in the following screenshot:

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin32.jpg)

**Step 14)**. Select specific build, and you will see the current status by clicking on "**console output**".

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin33.jpg)

**Scheduling Jenkins for automatic execution.**

Scheduling builds (Selenium Tests) is one of the important features of Jenkins where it automatically triggers the build, based on defined criteria. Jenkins provides multiple ways to trigger the build process under the Build Trigger configuration.

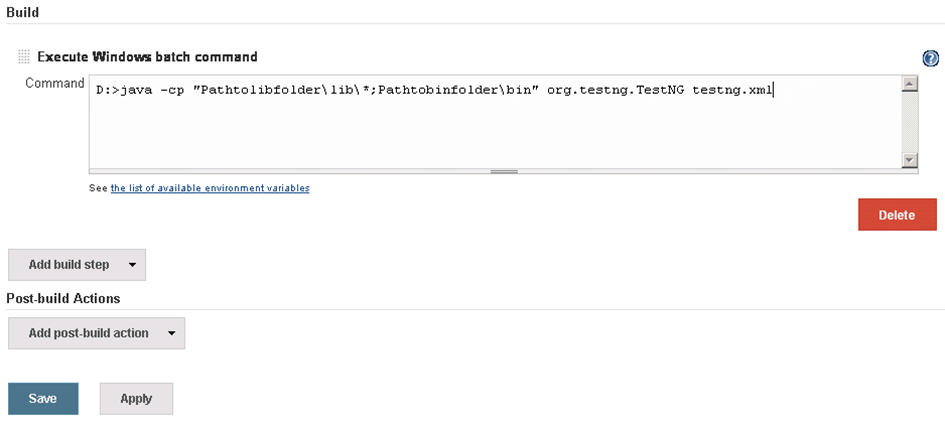
For example:  
Enter 0 23 \* \* \* in the Schedule textbox as shown in the following screenshot. This will trigger the build process every day at 11 p.m.

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin34.jpg)

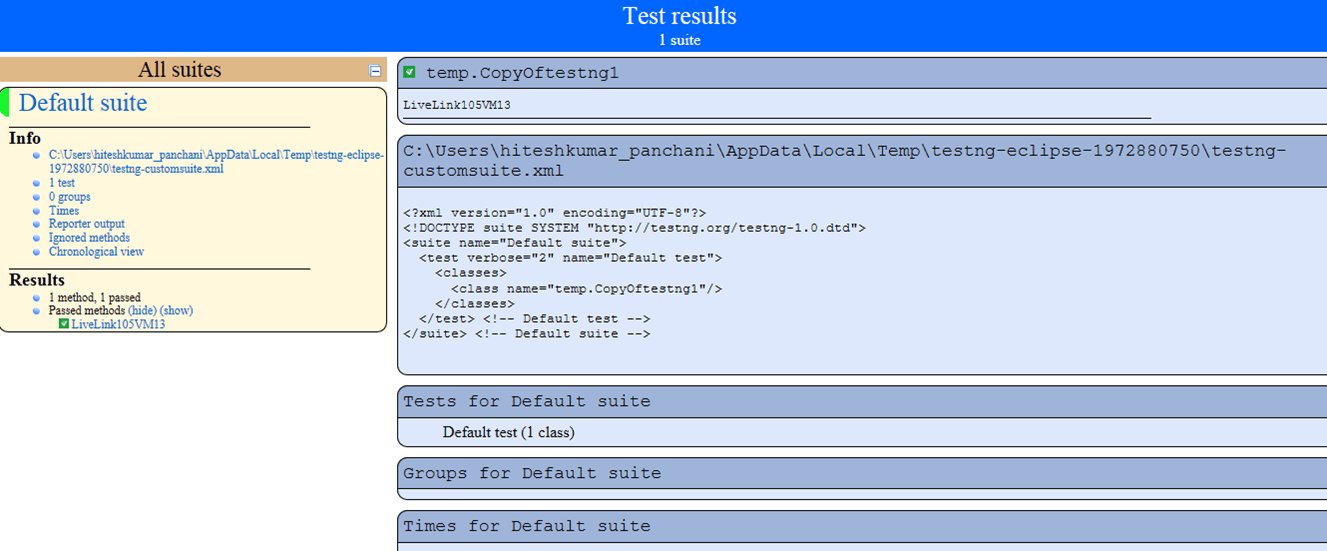
**Using Jenkings without Maven**

To run pure TestNg script in Jenkins, enter the following in build.

**D:>java -cp "Pathtolibfolder\lib\\*;Pathtobinfolder\bin" org.testng.TestNG testng.xml**

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin35.png)

* Click on Save button.
* Note: The actual path of lib and bin folder need to add in above command.
* After saving the command, Jenkins will build project in predefined time, and this command will be run using TestNG.
* Result will be stored in custom report HTML file that can be sent via email with a Jenkin configuration
* Output of the code will be

[](https://www.guru99.com/images/5-2015/050115_1023_MavenJenkin36.png)

**Benefits of using Jenkins**

1. Early issue finding – Bug can be detected in early phase of the software development.
2. Automatic integration – no separate effort required to integrate all changes.
3. Installer – a deployable system available at any point of development
4. Records – past build records maintained.
5. Support and Plugins: One of the reasons for Jenkin's popularity is the availability of large community support. Also, lots of ready-made plugins are available which help you expand its functionality.

**------------------------------------------------------------------------------------------------------------------------------------------------------------------------------**

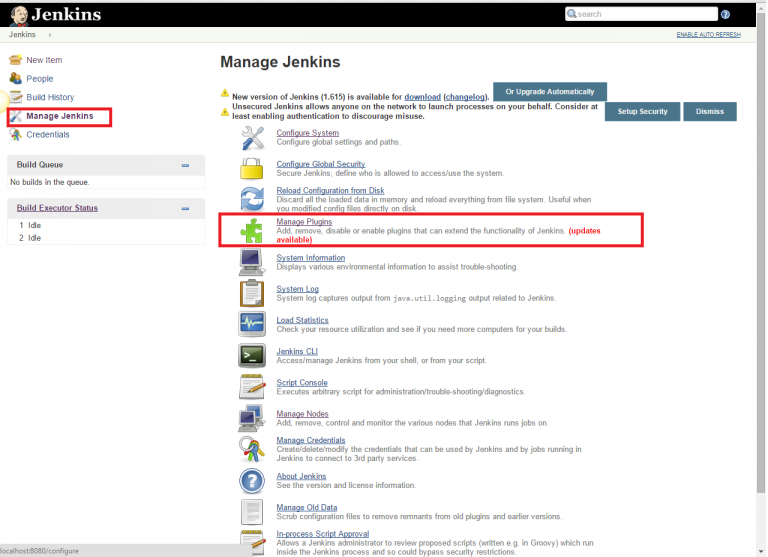
**CONFIGURE EMAIL SETTINGS**

1. Open Jenkins using the following URL: http://localhost:8080/ on any browser.

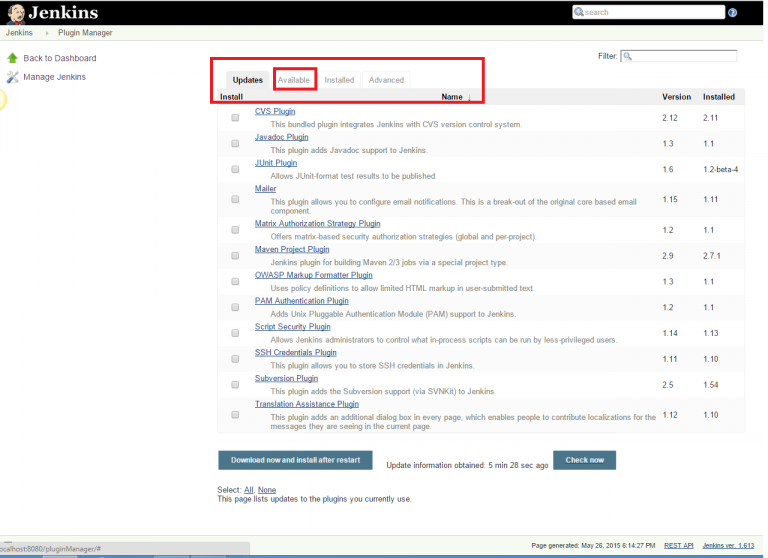
Graphical user interface, text, application, email

Description automatically generated

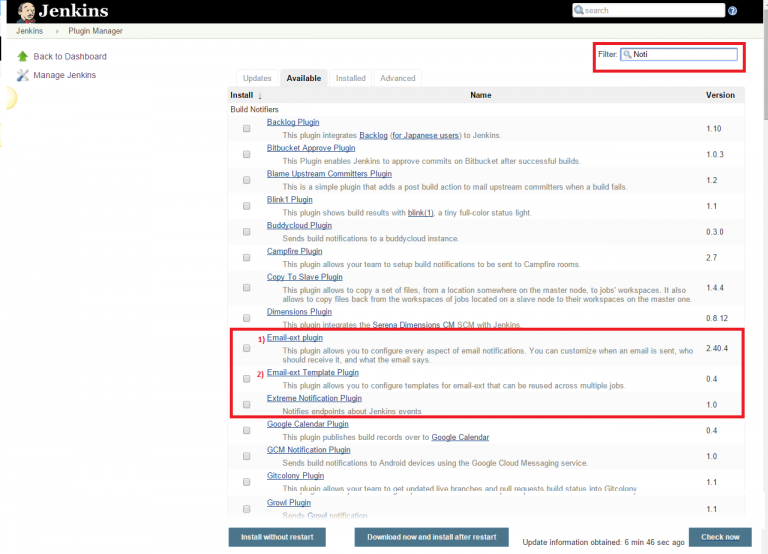
2. Click the ‘Manage Jenkins’ menu option displayed at the right side of the screen. You will be redirected to the ‘Manage Jenkins’ page, where you need to select the ‘Manage Plugin’ option.



3. Click the ‘Available’ tab present at the top of the ‘Manage Plugin’ page.



4. Start typing ‘Notification’ in the ‘Filter’ field displayed at the top-right side of the ‘Manage Plugin’ page. Click the checkbox next to the ‘**Email-ext plugin’** option. Click the ‘Install without restart’ button.

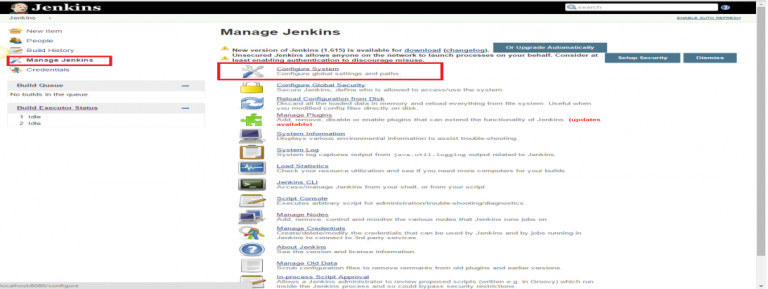


5. Now, click the checkbox next to the ‘Email-ext Template Plugin’ option. Click the ‘Install without restart’ button.

Graphical user interface, application

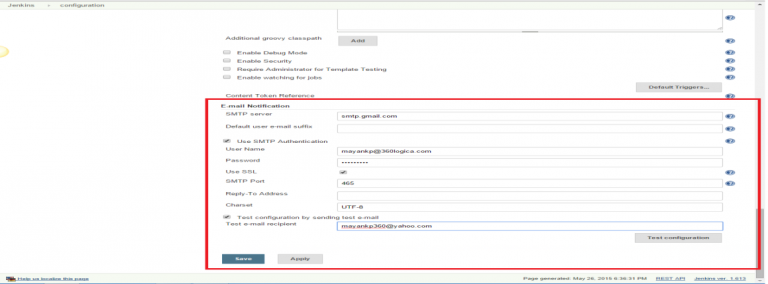
Description automatically generated

6. Go to the Jenkins home page and click the ‘Manage Jenkins’ menu option. Then, select the ‘Configure System’ option.

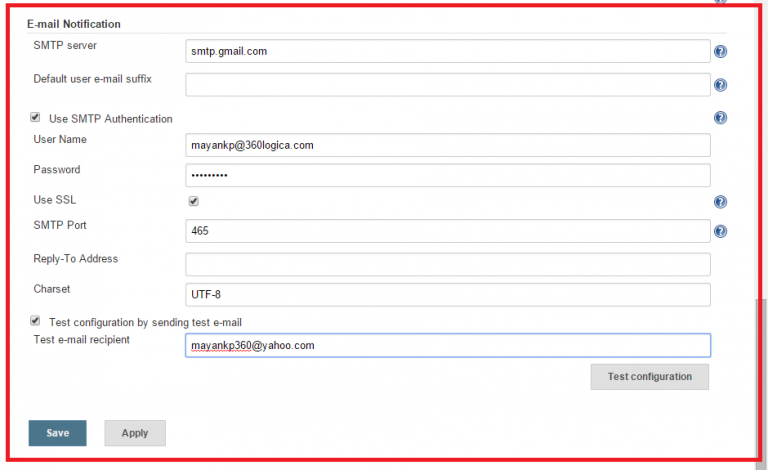


7. Enter the SMTP server name under ‘Email Notification’. Click the ‘Advanced’ button and then click the checkbox next to the ‘Use SMTP Authentication’ option. Now, set the following fields.(2 step authetication)

* + **SMTP server name** : smtp.gmail.com
  + **User name**: user\_email\_id@gmail.com
  + **Password**: 123456
  + **Use SSL**: Checked
  + **SMTP Port**: 465



8. Check the email notification functionality by clicking the checkbox next to the ‘Test configuration by sending Test e-mail recipient’ option. Enter a valid email id and click the ‘Test configuration’ button to check whether the email id is valid or not.



9. Go to the home page and click on a created job, like Homes. Then, click the ‘Configure’ option.

Graphical user interface, text, application

Description automatically generated

Graphical user interface, application

Description automatically generated

10. Click the ‘Add post-build action’ drop-down.

Graphical user interface, application, Teams

Description automatically generated

11. Select the ‘E-mail Notification’ value.

Graphical user interface, application

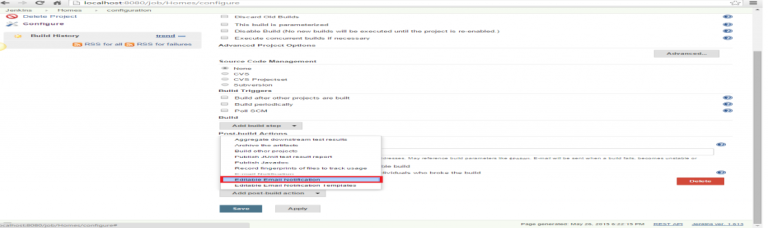
Description automatically generated

12. Enter the recipient email id in the ‘E-mail Notification’ box and select the checkbox next to the ‘**Send e-mail for every unstable build’** option.

Graphical user interface, text, application, email

Description automatically generated

13. Click the ‘Add post-build action’ drop-down and select the ‘Editable Email Notification’ value.



14. Fill the ‘Editable Email Notification’ fields.

* Project Recipient List : [email\_id@gmail.com](mailto:email_id@gmail.com)

Graphical user interface, application

Description automatically generated

15. Click the ‘Advance Settings…’ button in the ‘Editable Email Notification’ box.

16. Click the ‘Add Trigger’ drop-down and select the ‘Always’ option.

A computer screen shot

Description automatically generated with low confidence

17. Click the ‘Save’ button.

18. Go to the home page and click on the job, like Homes.

19. Click the ‘Build now’ link and check the email id after the job execution.