

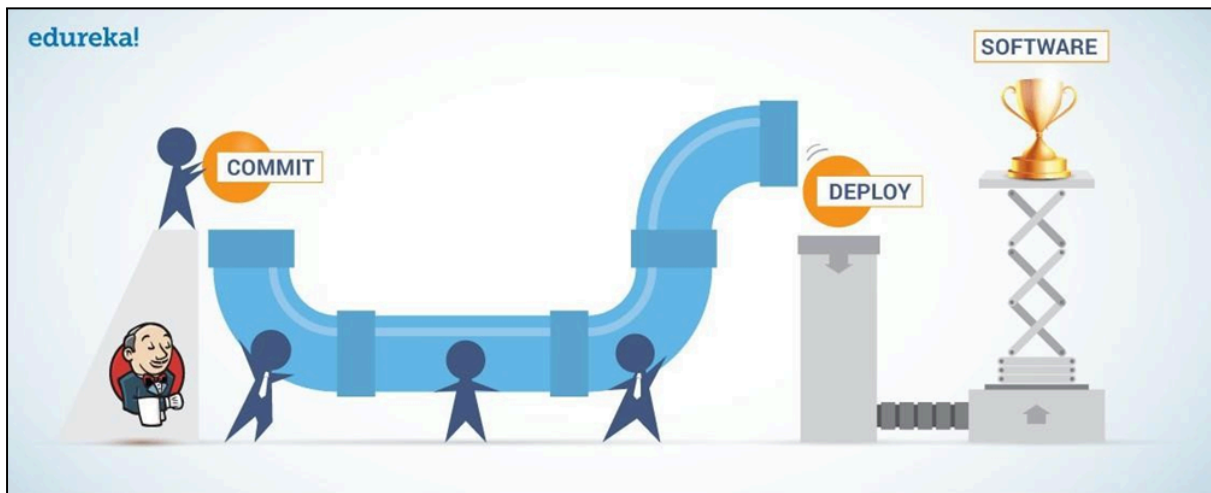
Name of Student: Pushkar Sane		
Roll Number: 45		Lab Assignment Number: 10
Title of Lab Assignment: To implement Jenkins Master/Slave architecture.		
DOP: 28-03-2024		DOS: 05-04-2024
CO Mapped: CO3	PO Mapped: PO2, PO3, PO5, PS01, PS02	Signature:

Practical No. 10

Aim: To implement Jenkins Master/Slave architecture.

Introduction:

What is Jenkins?



Jenkins is an open-source automation tool written in Java with plugins built for Continuous Integration purposes. Jenkins is used to build and test your software projects continuously making it easier for developers to integrate changes to the project, and making it easier for users to obtain a fresh build. It also allows you to continuously deliver your software by integrating with a large number of testing and deployment technologies.

With Jenkins, organizations can accelerate the software development process through automation. Jenkins integrates development life-cycle processes of all kinds, including build, document, test, package, stage, deploy, static analysis and much more.

Jenkins achieves Continuous Integration with the help of plugins. Plugins allow the integration of Various DevOps stages. If you want to integrate a particular tool, you need to install the plugins for that tool. For example Git, Maven 2 project, Amazon EC2, HTML publisher, etc.

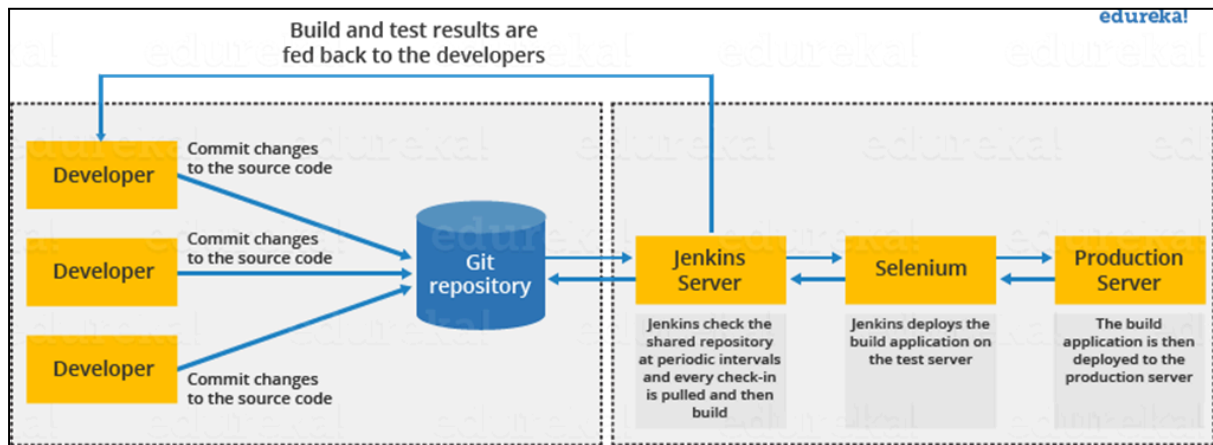
Advantages of Jenkins include:

- It is an open-source tool with great community support.
- Too easy to install.

- It has 1000+ plugins to ease your work. If a plugin does not exist, you can code it and share it with the community.
- It is free of cost.
- It is built with Java and hence, it is portable to all the major platforms.

Jenkins Architecture

Let us have a look at the Jenkins Architecture below diagram depicts the same.



This single Jenkins server was not enough to meet certain requirements like:

- Sometimes you might need several different environments to test your builds. This cannot be done by a single Jenkins server.
- If larger and heavier projects get built on a regular basis then a single Jenkins server cannot simply handle the entire load.

To address the above-stated needs, Jenkins distributed architecture came into the picture.

Jenkins Distributed Architecture

Jenkins uses a Master-Slave architecture to manage distributed builds. In this architecture, Master and Slave communicate through TCP/IP protocol.

Jenkins Master

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 (possibly taking them online and offline as required).
- Recording and presenting the build results.
- A Master instance of Jenkins can also execute build jobs directly.

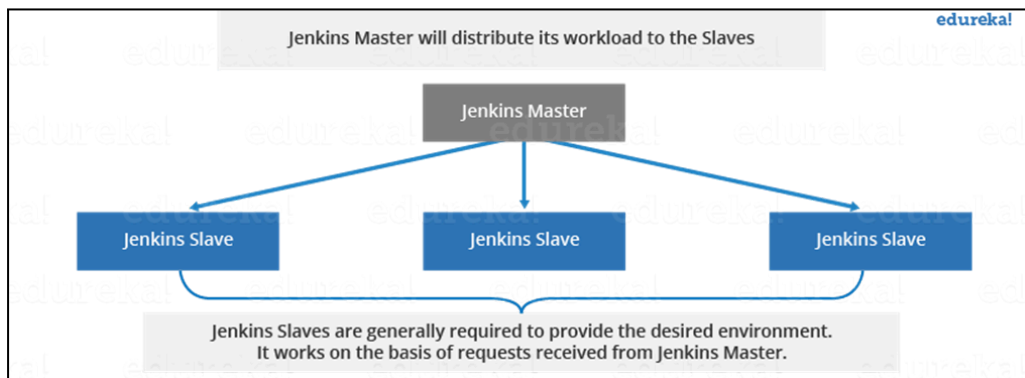
Jenkins Slave

A Slave is a Java executable that runs on a remote machine. Following are the characteristics of Jenkins

Slaves:

- It hears requests from the Jenkins Master instance.
- Slaves can run on a variety of operating systems.
- The job of a Slave is to do as they are told to, which involves executing build jobs dispatched by the Master.
- You can configure a project to always run on a particular Slave machine or a particular type of Slave machine, or simply let Jenkins pick the next available Slave.

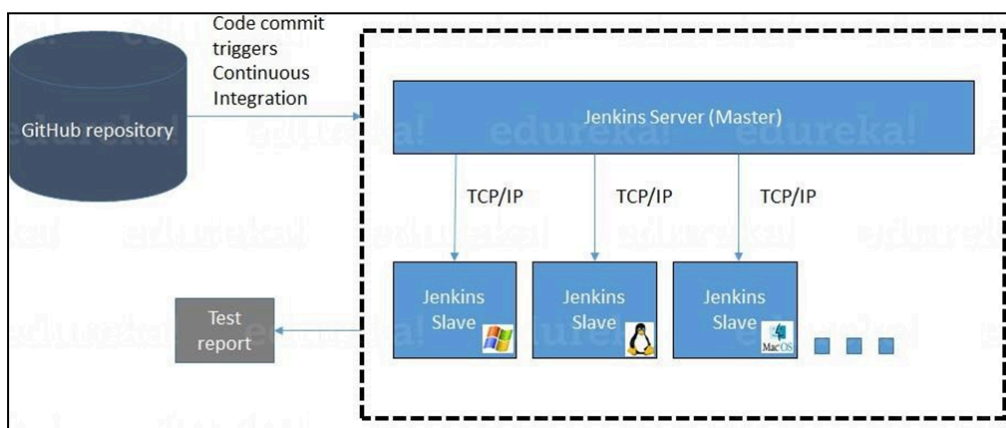
The diagram below is self-explanatory. It consists of a Jenkins Master which is managing three Jenkins Slave.



How does Jenkins Master and Slave Architecture work?

Now let us look at an example in which we use Jenkins for testing in different environments like Ubuntu, MAC, Windows, etc.

The diagram below represents the same:

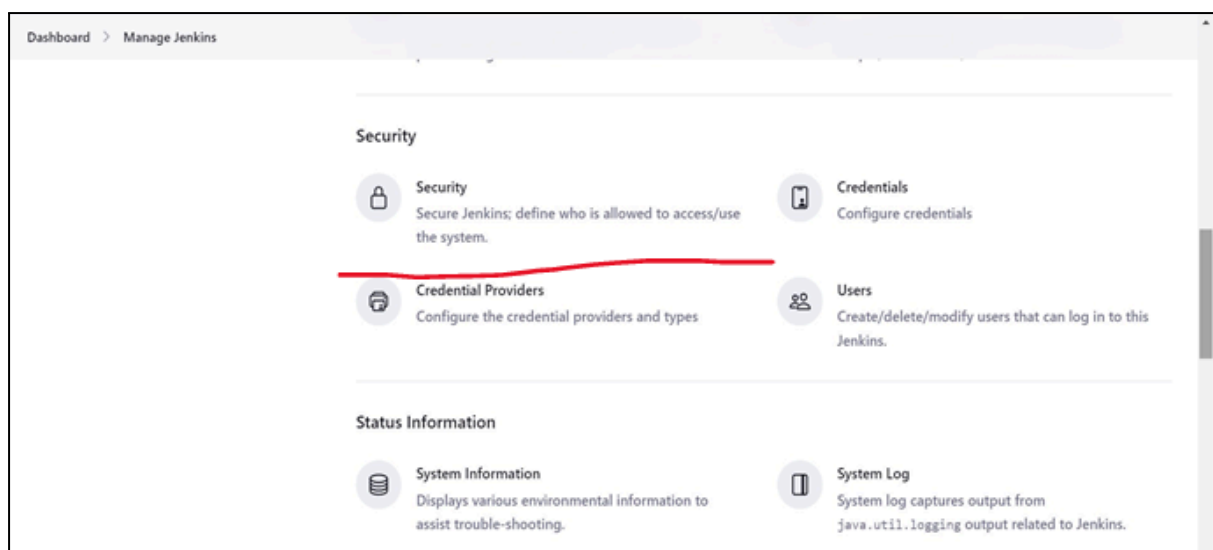
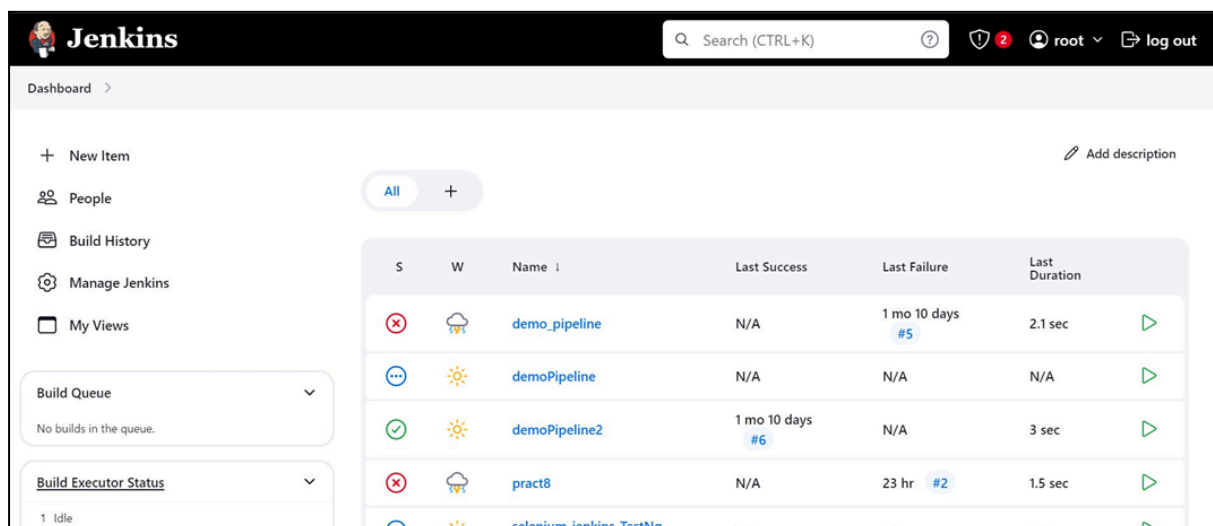


The above image represents the following functions:

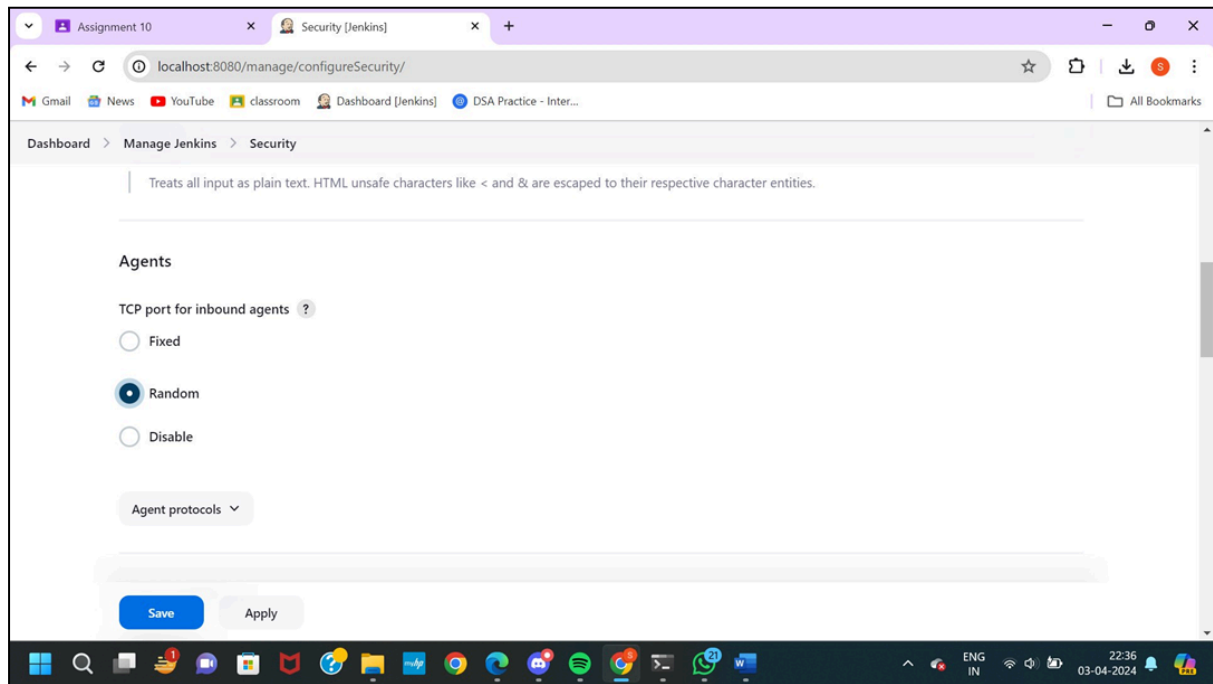
- Jenkins checks the Git repository at periodic intervals for any changes made in the source code.
- Each build requires a different testing environment which is not possible for a single Jenkins server. In order to perform testing in different environments, Jenkins uses various Slaves as shown in the diagram.
- Jenkins Master requests these Slaves to perform

Steps:

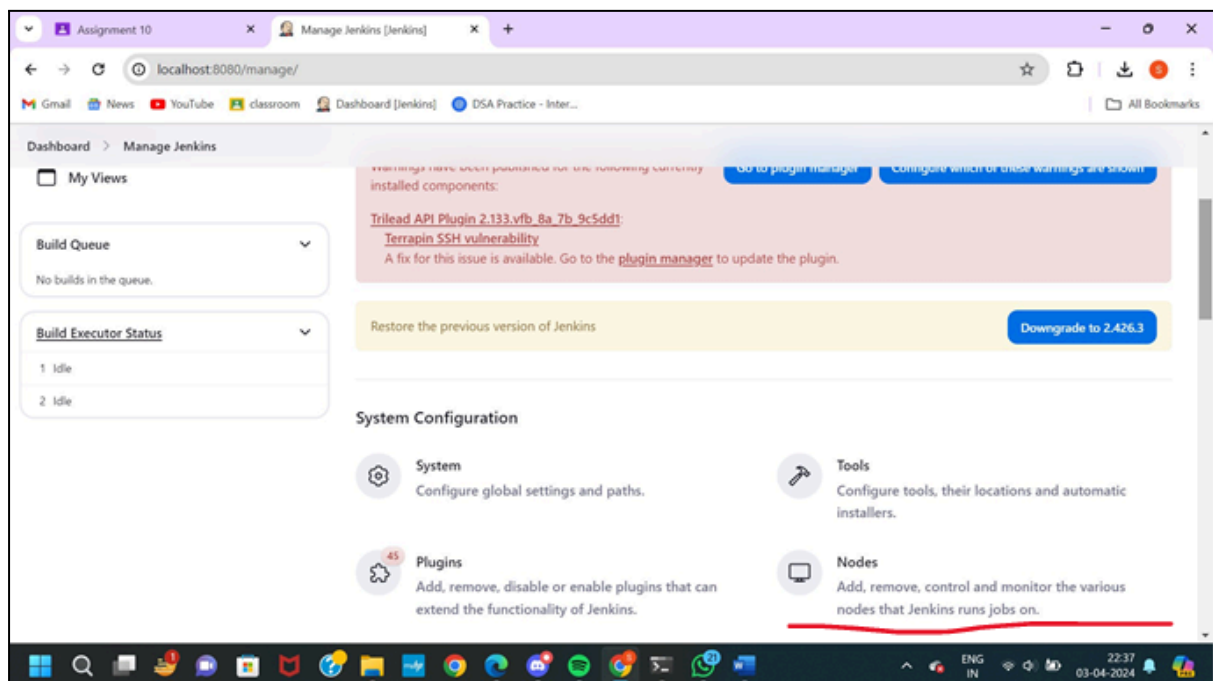
Step 1: Open the Jenkins (localhost:8080) and navigate to the “Manage Jenkins” menu. Then select the “Security” option.



Step 2: Search for the “Agents” option and then select “Random” under the TCP ports for inbound agents menu and then scroll down and click on apply and save.

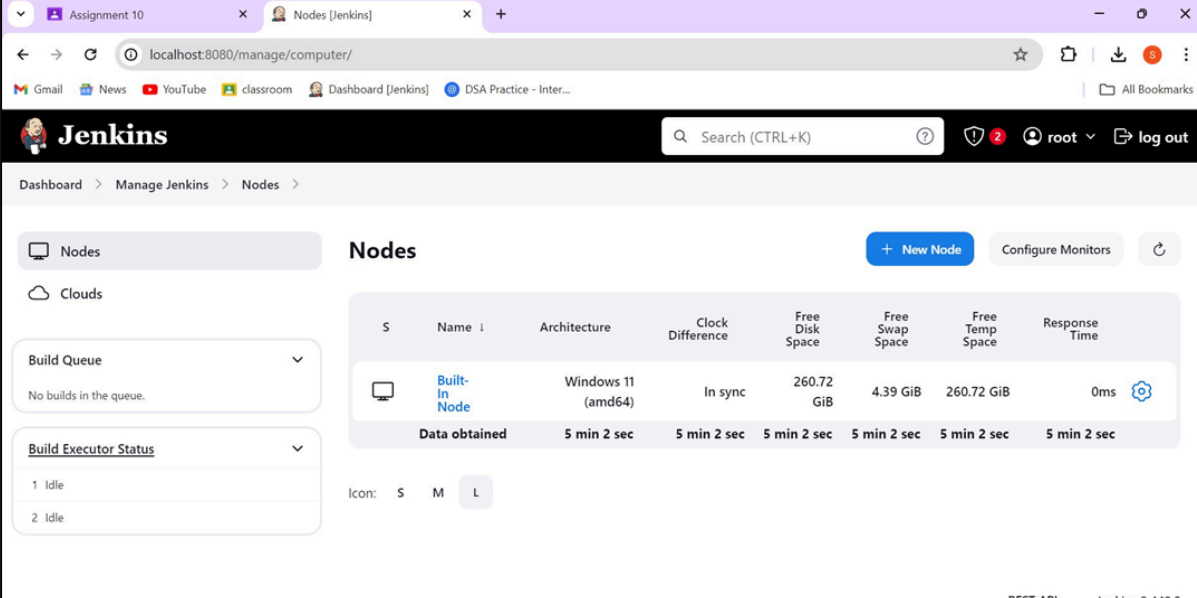


Step 3: Go to the dashboard and again navigate to the Manage Jenkins Menu, and then select the “Nodes” menu.



Step 4: You will be able to see the “Built-in Node” in the Nodes list. This node is also known as Master Node which is basically the current OS.

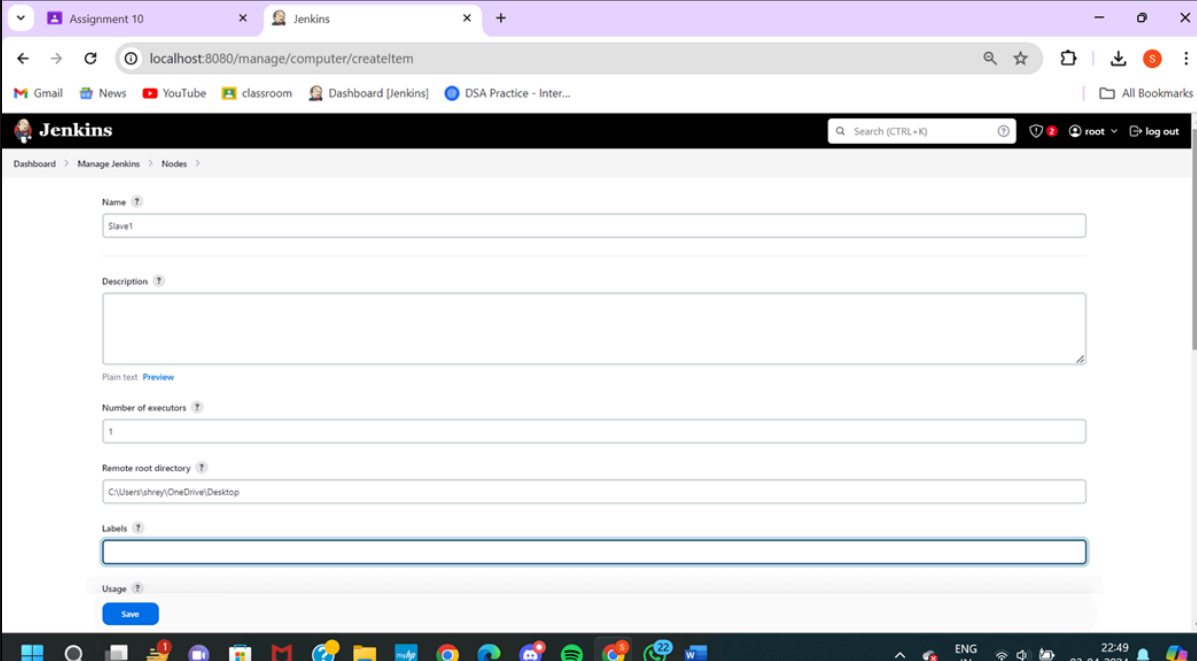
Now click on the “New Node” to create a new Slave node.



The screenshot shows the Jenkins web interface at the URL `localhost:8080/manage/computer/`. The page title is "Nodes". On the left sidebar, there are links for "Nodes" and "Clouds". The main content area displays a table of nodes. The first node is the "Built-in Node" with architecture "Windows 11 (amd64)". Below the table, there are icons for "S", "M", and "L".

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-in Node	Windows 11 (amd64)	In sync	260.72 GiB	4.39 GiB	260.72 GiB	0ms
Data obtained		5 min 2 sec	5 min 2 sec	5 min 2 sec	5 min 2 sec	5 min 2 sec	5 min 2 sec

Step 5: Configure the Slave node.



The screenshot shows the Jenkins web interface at the URL `localhost:8080/manage/computer/createitem`. The page is titled "createitem". It contains several form fields for configuring a new node:

- Name:** A text input field with the value "Slave1".
- Description:** A large text area for describing the node.
- Number of executors:** A text input field with the value "1".
- Remote root directory:** A text input field with the value "C:/Users/shrey/OneDrive/Desktop".
- Labels:** A text input field for specifying labels.
- Usage:** A checkbox labeled "Save" is checked.

For Remote root directory, create a new folder on your device (Empty) and copy paste the path of that folder.

Number of executors ?

2

The maximum number of concurrent builds that Jenkins may perform on this node.
A good value to start with would be the number of CPU cores on the machine. Setting a higher value would cause each build to take longer, but could increase the overall throughput. For example, one build might be CPU-bound, while a second build running at the same time might be I/O-bound — so the second build could take advantage of the spare I/O capacity at that moment.

Agents (nodes that are not the built-in node) must have at least one executor. To temporarily prevent any builds from being executed on an agent, use the *Mark this node temporarily offline* button on the agent's page.

For the built-in node, set the number of executors to zero to prevent it from executing builds locally on the controller. *Note: The built-in node will always be able to run flyweight tasks including Pipeline's top-level task.*

localhost:8080/computer/Slave1/configure

Dashboard > Nodes > Slave1 > Configure

Usage ?
Use this node as much as possible

Launch method ?
Launch agent by connecting it to the controller

Availability ?
Keep this agent online as much as possible

Node Properties

- ☐ Disable deferred wipeout on this node ?
- ☐ Disk Space Monitoring Thresholds
- ☐ Environment variables
- ☐ Tool Locations

Save

Labels can be anything according to your choice

Select the options exactly as above, if you are not able to see the above options in the dropdown list then go to step 1 and select TCP inbound as random and check again.

Click on save.

Step 6: Once the node is saved, you will see the cross sign on the slave node which basically means that the slave is not yet connected or running.

To connect the slave, click on the slave node and then select the status option. On the same page you will see multiple commands for different operating systems. Copy the command for windows

Dashboard > Manage Jenkins > Nodes >

Nodes

+ New Node Configure Monitors

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-In Node	Windows 11 (amd64)	In sync	10.03 GiB	7.45 GiB	10.03 GiB	0ms
	Slave1	Windows 11 (amd64)	In sync	10.03 GiB	7.46 GiB	10.03 GiB	100ms
Data obtained			41 min	41 min	41 min	41 min	41 min

Icon: S M L

REST API Jenkins 2.440.2

and run it in the command prompt.

localhost:8080/computer/Slave1/

Dashboard > Nodes > Slave1

Status

Delete Agent

Configure

Build History

Load Statistics

Log

Build Executor Status

1 Idle

2 Idle

Agent Slave1

Mark this node temporarily offline

This is the first demo of Master-slave demo using Jenkins

Connection was broken

Run from agent command line: (Unix)

```
curl -s0 http://localhost:8080/jnlpJars/agent.jar
java -jar agent.jar -url http://localhost:8080/ -secret f8d8d223c7ccfd34dc83b85a3e58c7fcc2dc1382adb7ab1efc5f045e9227346e -name Slave1 -workDir "C:\Users\prasa\Desktop\Master_Slave_Jenkins"
```

Run from agent command line: (Windows)

```
curl.exe -s0 http://localhost:8080/jnlpJars/agent.jar
java -jar agent.jar -url http://localhost:8080/ -secret f8d8d223c7ccfd34dc83b85a3e58c7fcc2dc1382adb7ab1efc5f045e9227346e -name Slave1 -workDir "C:\Users\prasa\Desktop\Master_Slave_Jenkins"
```

Or run from agent command line, with the secret stored in a file: (Unix)

```
echo f8d8d223c7ccfd34dc83b85a3e58c7fcc2dc1382adb7ab1efc5f045e9227346e > secret-file
curl -s0 http://localhost:8080/jnlpJars/agent.jar
```

```

Command Prompt - java -jar X + v
Apr 01, 2024 8:14:53 PM hudson.remoting.Launcher createEngine
INFO: Setting up agent: Slave1
Apr 01, 2024 8:14:53 PM hudson.remoting.Engine startEngine
INFO: Using Remoting version: 3206.vb_15dcf73f6a_9
Apr 01, 2024 8:14:53 PM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
INFO: Using C:\Users\prasa\Desktop\Master_Slave_Jenkins\remoting as a remoting work directory
Apr 01, 2024 8:14:53 PM hudson.remoting.Launcher$Cuilistener status
INFO: Locating server among [http://localhost:8080/]
Apr 01, 2024 8:14:53 PM org.jenkinsci.remoting.engine.JnlpAgentEndpointResolver resolve
INFO: Remoting server accepts the following protocols: [JNLP4-connect, Ping]
Apr 01, 2024 8:14:53 PM hudson.remoting.Launcher$Cuilistener status
INFO: Agent discovery successful
Agent address: localhost
Agent port: 53285
Identity: 86:28:2c:d6:b0:51:dd:6b:52:6a:b0:17:37:e3:31:38
Apr 01, 2024 8:14:53 PM hudson.remoting.Launcher$Cuilistener status
INFO: Handshaking
Apr 01, 2024 8:14:53 PM hudson.remoting.Launcher$Cuilistener status
INFO: Connecting to localhost:53285
Apr 01, 2024 8:14:53 PM hudson.remoting.Launcher$Cuilistener status
INFO: Server reports protocol JNLP4-connect-proxy not supported, skipping
Apr 01, 2024 8:14:53 PM hudson.remoting.Launcher$Cuilistener status
INFO: Trying protocol: JNLP4-connect
Apr 01, 2024 8:14:54 PM org.jenkinsci.remoting.protocol.impl.BIONetworkLayer$Reader run
INFO: Waiting for ProtocolStack to start.
Apr 01, 2024 8:14:54 PM hudson.remoting.Launcher$Cuilistener status
INFO: Remote identity confirmed: 86:28:2c:d6:b0:51:dd:6b:52:6a:b0:17:37:e3:31:38
Apr 01, 2024 8:14:54 PM hudson.remoting.Launcher$Cuilistener status
INFO: Connected

```

Step 7: Once the slave node is connected and running, we can create a new job for the slave node to perform remotely.

The screenshot shows the Jenkins 'Nodes' page. On the left, there are sections for 'Nodes' (with a 'New Node' button), 'Build Queue' (showing 'No builds in the queue'), and 'Build Executor Status' (showing 'Built-In Node' with 1 idle executor and 'Slave1' with 2 idle executors). The main table lists the nodes:

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-In Node	Windows 11 (amd64)	In sync	9.89 GiB	6.13 GiB	9.89 GiB	0ms
	Slave1	Windows 11 (amd64)	In sync	9.89 GiB	6.10 GiB	9.89 GiB	92ms
Data obtained			45 sec	45 sec	44 sec	45 sec	45 sec

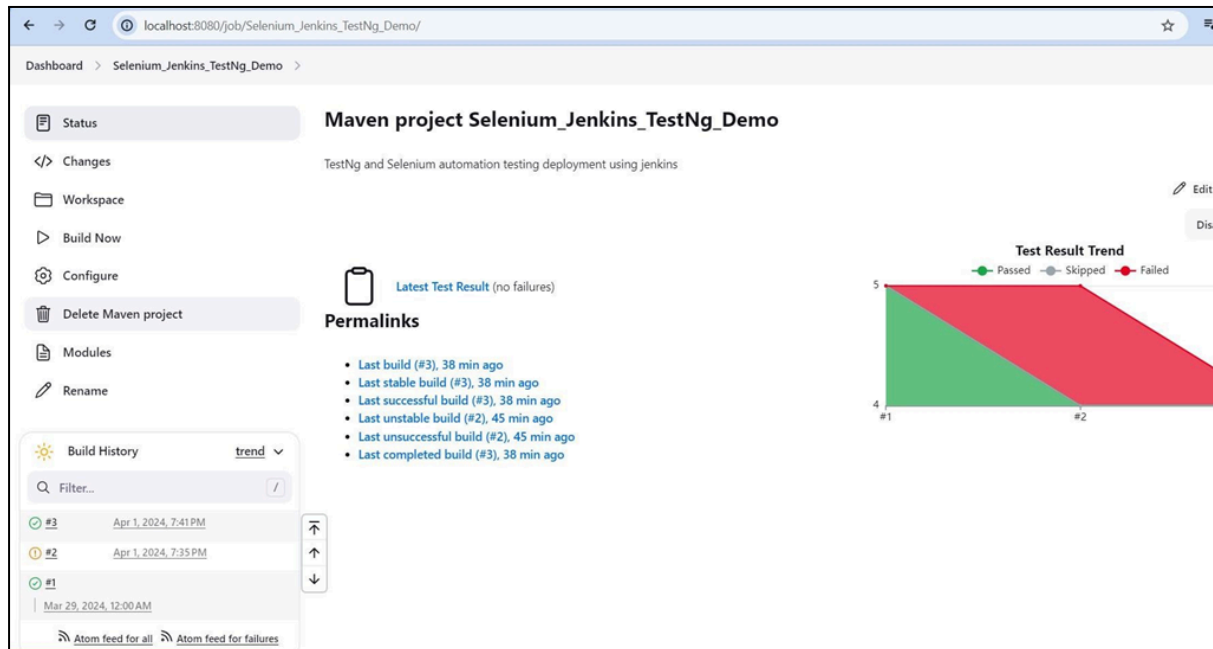
At the bottom, there are icons for 'S', 'M', and 'L'.

Go to any previously built Jenkins project and click on configure.

The screenshot shows the Jenkins 'Configuration' page for a project named 'Selenium_Jenkins_TestNg_Demo'. The 'General' tab is selected. The 'Discard old builds' checkbox is checked. The 'GitHub project' checkbox is unchecked. The 'GitLab Connection' dropdown is set to 'Use alternative credential'. The 'This project is parameterized' checkbox is unchecked. The 'Throttle builds' checkbox is unchecked. The 'Execute concurrent builds if necessary' checkbox is unchecked. The 'Restrict where this project can be run' checkbox is checked, and the 'Label Expression' is set to 'Slave1'. A note below states: 'Label Slave1 matches 1 node. Permissions or other restrictions provided by plugins may further reduce that list.' At the bottom, there are 'Save' and 'Apply' buttons.

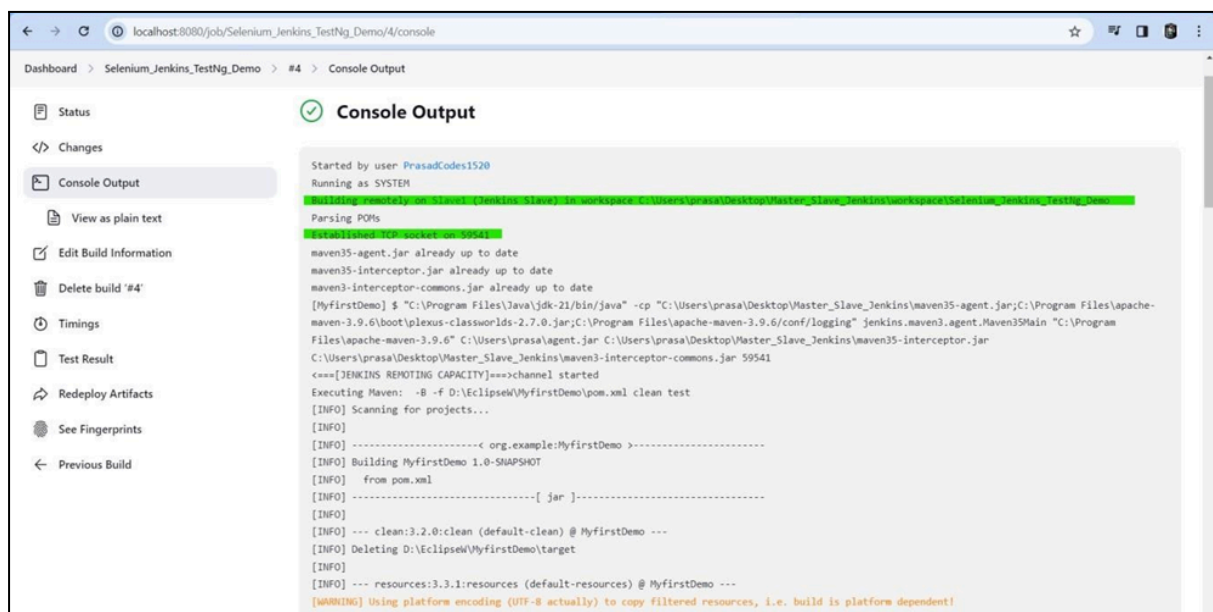
- Then search for the “Restrict where this project can be run” option select it.
- And enter the name of the slave node in the empty field.

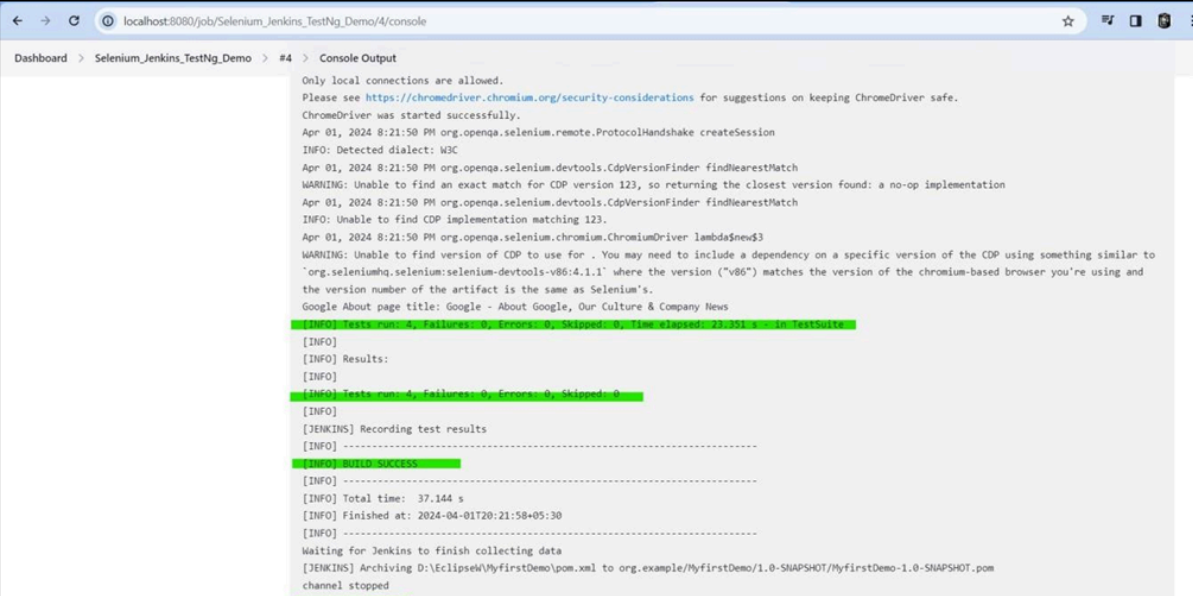
Step 8: Click on save, and then build the Jenkins project. (Build Now).



Once you click on build now, the Jenkins project will be remotely handled by the slave node which we configured.

Step 9: To check if the project has been remotely handled by slave or not, go to the console output of the recently built project. (Step 8)





The screenshot shows a web browser window with the address bar displaying 'localhost:8080/job/Selenium_Jenkins_TestNg_Demo/4/console'. The page content is titled 'Console Output' and shows the following log entries:

```
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
Apr 01, 2024 8:21:50 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected dialect: WC
Apr 01, 2024 8:21:50 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 123, so returning the closest version found: a no-op implementation
Apr 01, 2024 8:21:50 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Unable to find CDP implementation matching 123.
Apr 01, 2024 8:21:50 PM org.openqa.selenium.chromium.ChromiumDriver lambda$new$3
WARNING: Unable to find version of CDP to use for . You may need to include a dependency on a specific version of the CDP using something similar to
'org.seleniumhq.selenium:selenium-devtools-v86:4.1.1' where the version ("v86") matches the version of the chromium-based browser you're using and
the version number of the artifact is the same as Selenium's.
Google About page title: Google - About Google, Our Culture & Company News
[INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 23.351 s in TestSuite
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 4, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[JENKINS] Recording test results
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 37.144 s
[INFO] Finished at: 2024-04-01T20:21:58+05:30
[INFO] -----
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving D:\Eclipse\MyFirstDemo\pom.xml to org.example/MyFirstDemo/1.0-SNAPSHOT/MyFirstDemo-1.0-SNAPSHOT.pom
channel stopped
Finished: SUCCESS
```

Conclusion: Implementing Jenkins with a master-slave architecture optimizes resource utilization, scalability, and fault tolerance. By distributing tasks across multiple nodes, it enhances efficiency, reduces build times, and ensures continuous operation. This setup fosters flexibility and agility in software development, promoting faster and more reliable delivery. Overall, it facilitates enhanced productivity and innovation within the organization.