Experiment No. 6

Aim: To understand Jenkins Master-Slave Architecture and Scale your Jenkins standalone implementation by implementing slave nodes.

LO No. & Statement: (LO4): Examine the importance of Selenium and Jenkins to test software Applications.

Theory:

Jenkins is one of the most important tools in DevOps. Jenkins is used in the Continuous Integration stage of DevOps. In this blog, I am going to talk about Jenkins's Master and Slave architecture. The pointers that I will cover are as follows:

- What is Jenkins?
- Jenkins Architecture
- How does Jenkins Master-Slave architecture work?
- Setting up Slaves with Jenkins Master

What is Jenkins?

- Jenkins is an open-source automation tool written in Java with plugins built for Continuous Integration purposes. Jenkins is used to building and testing 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.

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 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.

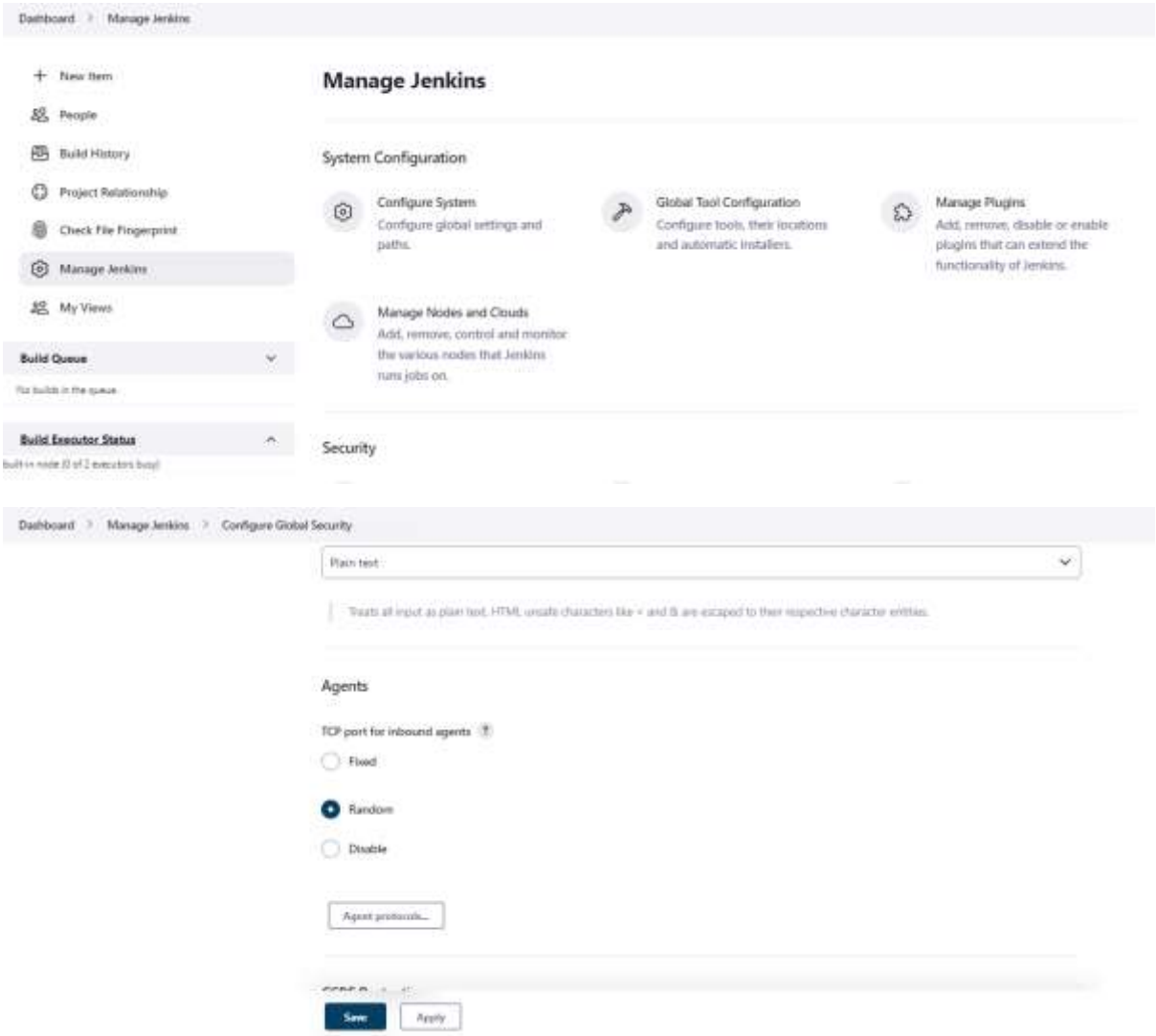
Jenkins Slave:

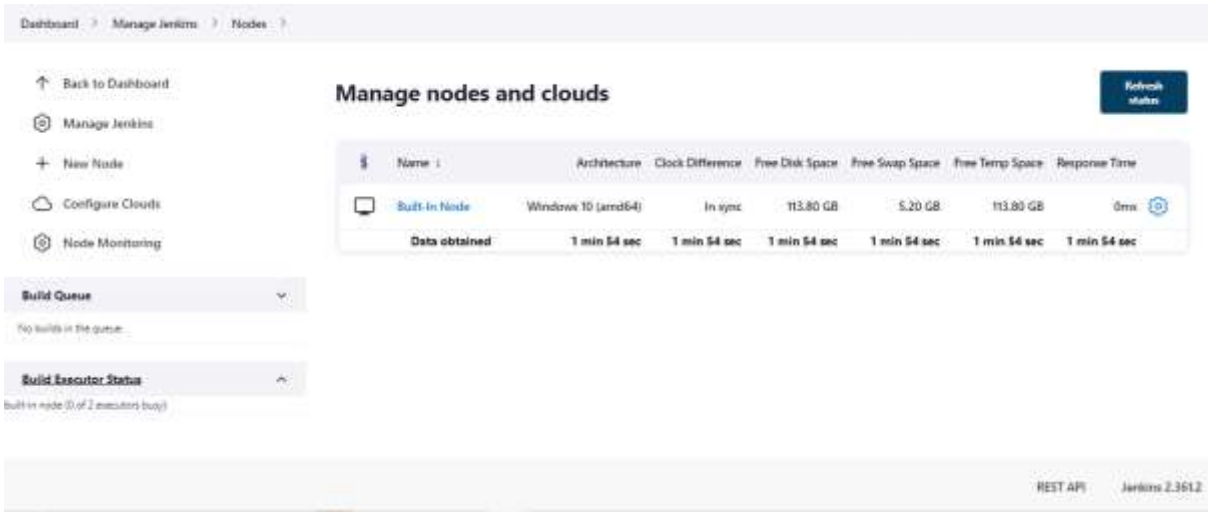
A Slave is a Java executable that runs on a remote machine. The following are the characteristics of Jenkins's Slaves:

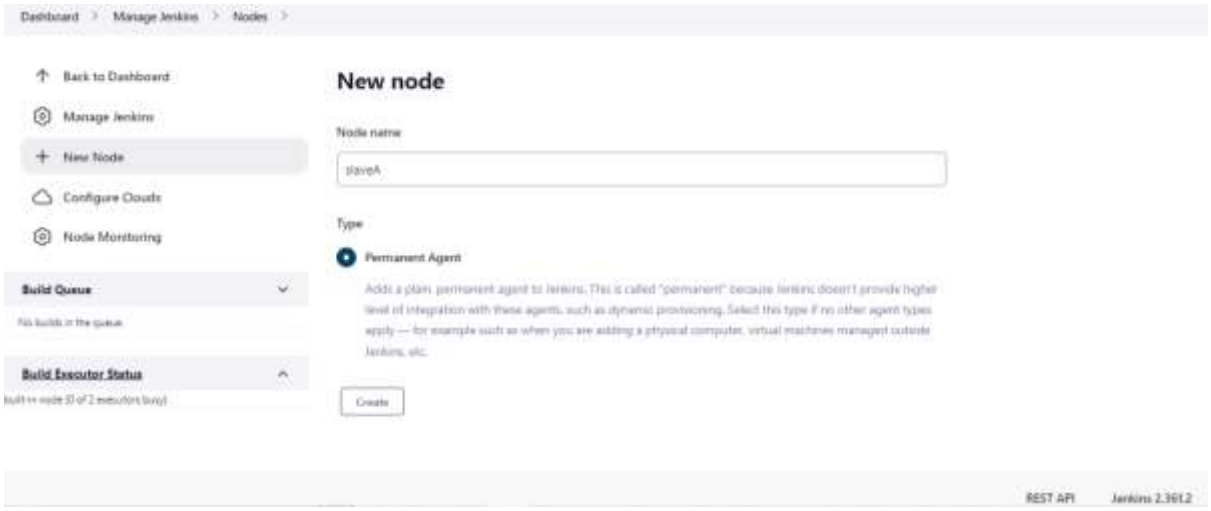
- It hears requests from the Jenkins Master instance.
- Slaves can run on a variety of operating systems.
- The job of a Slave is to do as they are told to, which involves executing build jobs dispatched by the Master.
- You can configure a project to always run on a particular Slave machine or a particular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

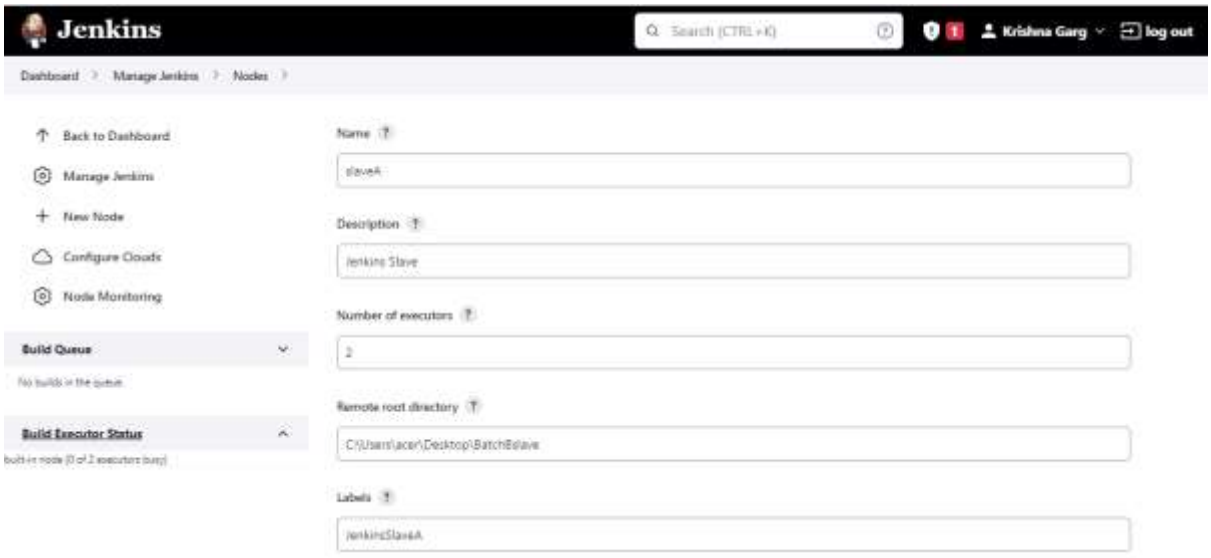
Output:

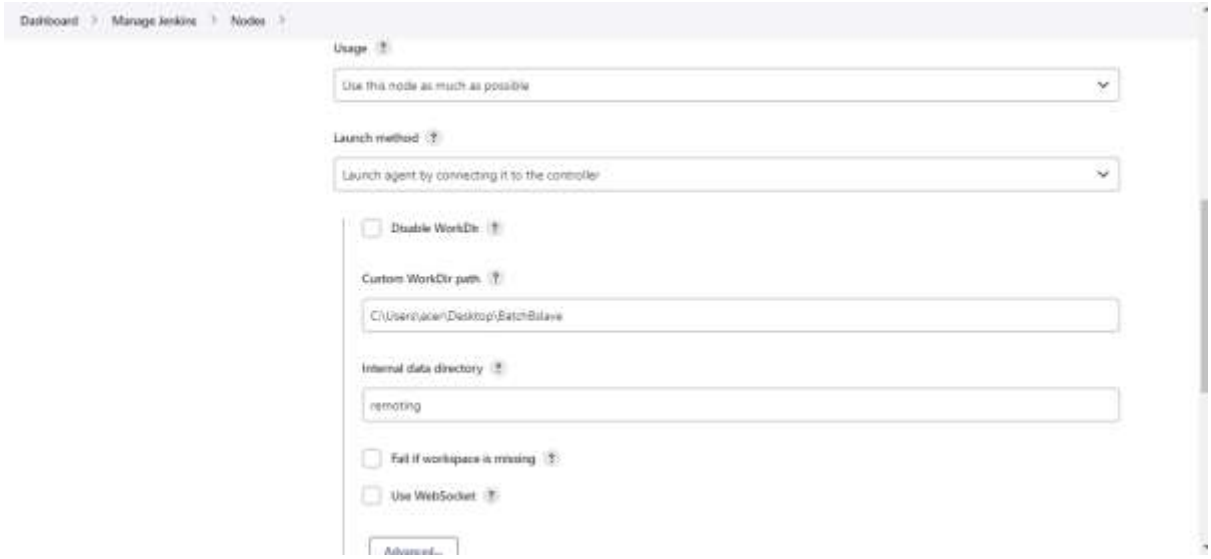
Step 1: Create a node by clicking onto manage nodes and clouds

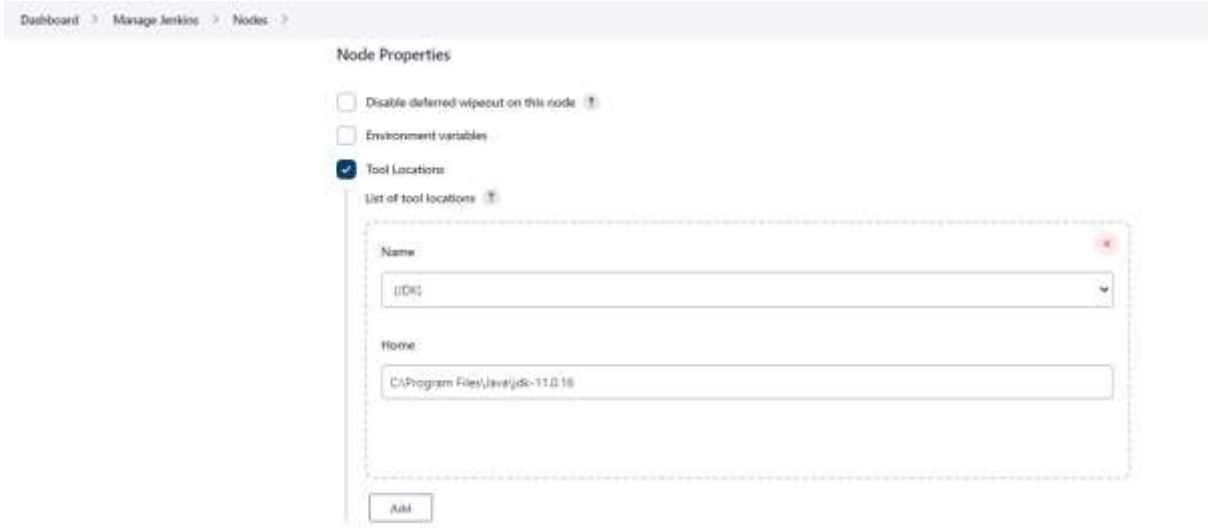


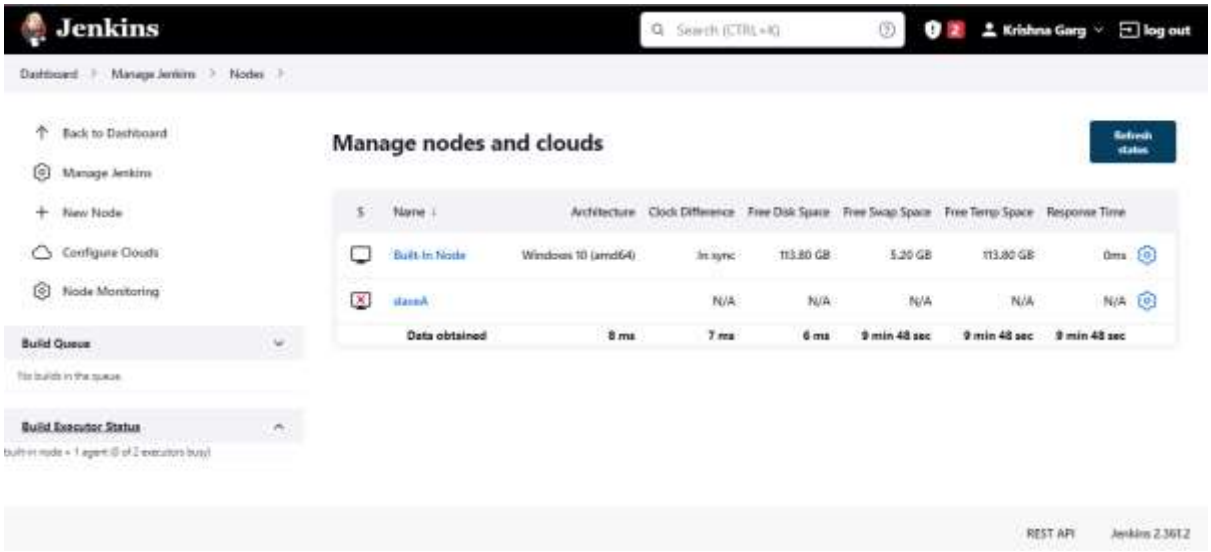

Dashboard > Manage Jenkins > Nodes >
Back to Dashboard
Manage Jenkins
New Node
Configure Clouds
Node Monitoring
Build Queue
No builds in the queue
Build Executor Status
Manage nodes and clouds
Refresh status
Name
Architecture
Clock Difference
Free Disk Space
Free Swap Space
Free Temp Space
Response Time
Built-In Node
Windows 10 (amd64)
In sync
113.80 GB
5.20 GB
113.80 GB
0ms
Data obtained
1 min 54 sec
REST API
Jenkins 2.361.2


Dashboard > Manage Jenkins > Nodes >
Back to Dashboard
Manage Jenkins
New Node
Configure Clouds
Node Monitoring
Build Queue
No builds in the queue
Build Executor Status
New node
Node name
slaveA
Type
Permanent Agent
Create
REST API
Jenkins 2.361.2

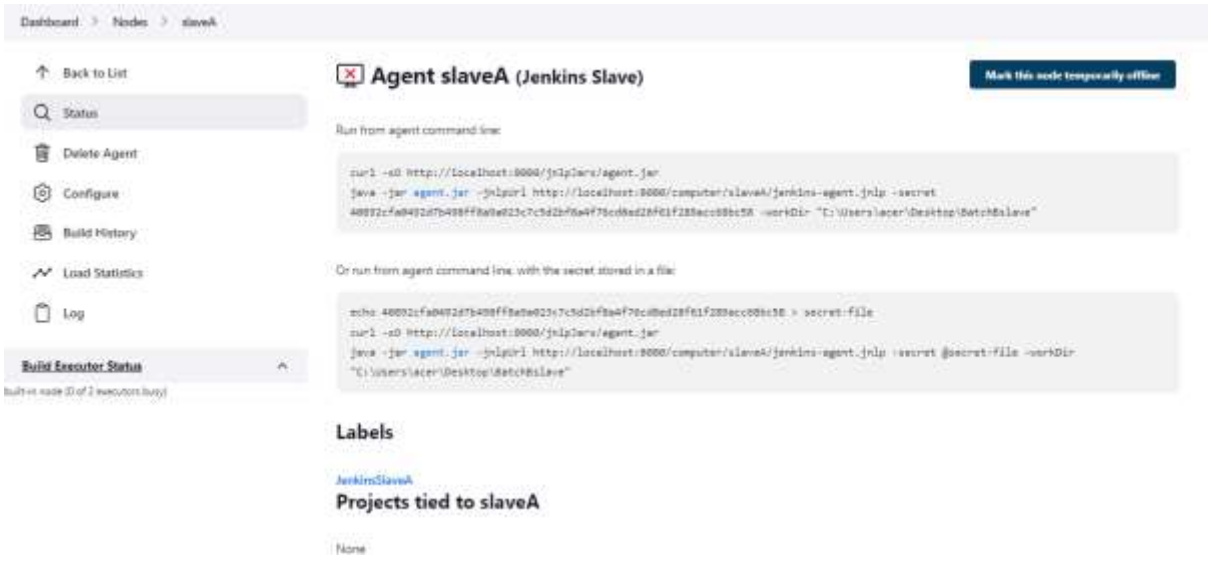

Jenkins
Search (CTRL+K)
Krishna Garg
log out
Dashboard > Manage Jenkins > Nodes >
Back to Dashboard
Manage Jenkins
New Node
Configure Clouds
Node Monitoring
Build Queue
No builds in the queue
Build Executor Status
Name
slaveA
Description
Jenkins Slave
Number of executors
2
Remote root directory
C:\Users\acer\Desktop\BatchSlave
Labels
JenkinsSlaveA


Dashboard > Manage Jenkins > Nodes >
Usage
Use this node as much as possible
Launch method
Launch agent by connecting it to the controller
Disable WorkDir
Custom WorkDir path
C:\Users\acer\Desktop\BatchBslave
Internal data directory
remoting
Fail if workspace is missing
Use WebSocket
Advanced...

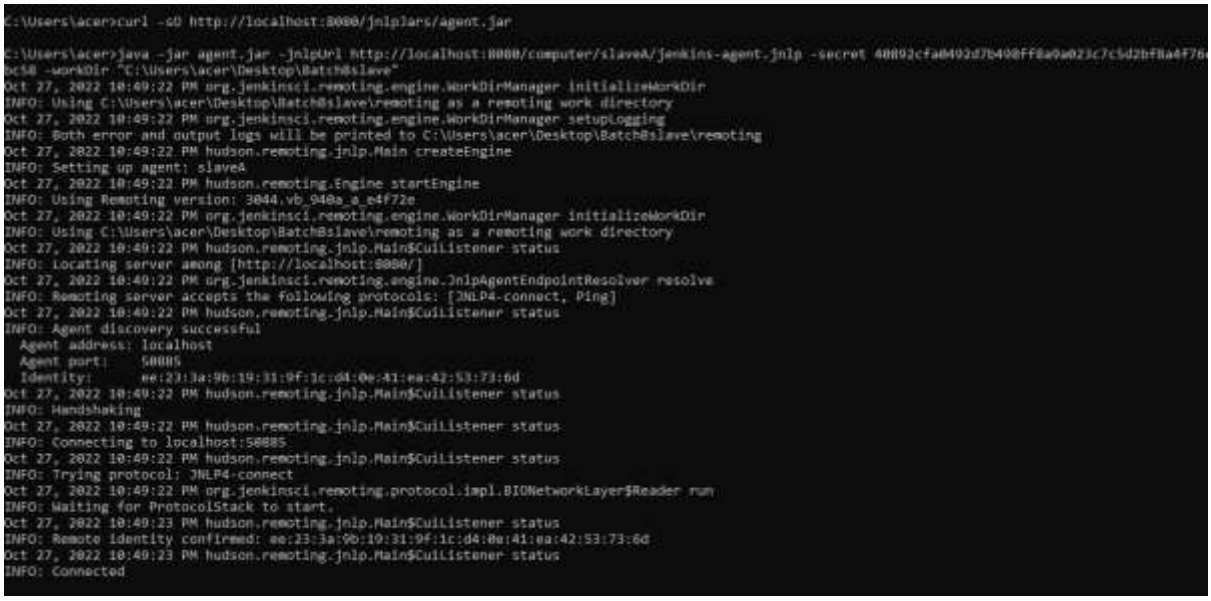

Dashboard > Manage Jenkins > Nodes >
Node Properties
Disable deferred wipeout on this node
Environment variables
Tool Locations
List of tool locations
Name
(JDK)
Home
C:\Program Files\Java\jdk-11.0.16
Add

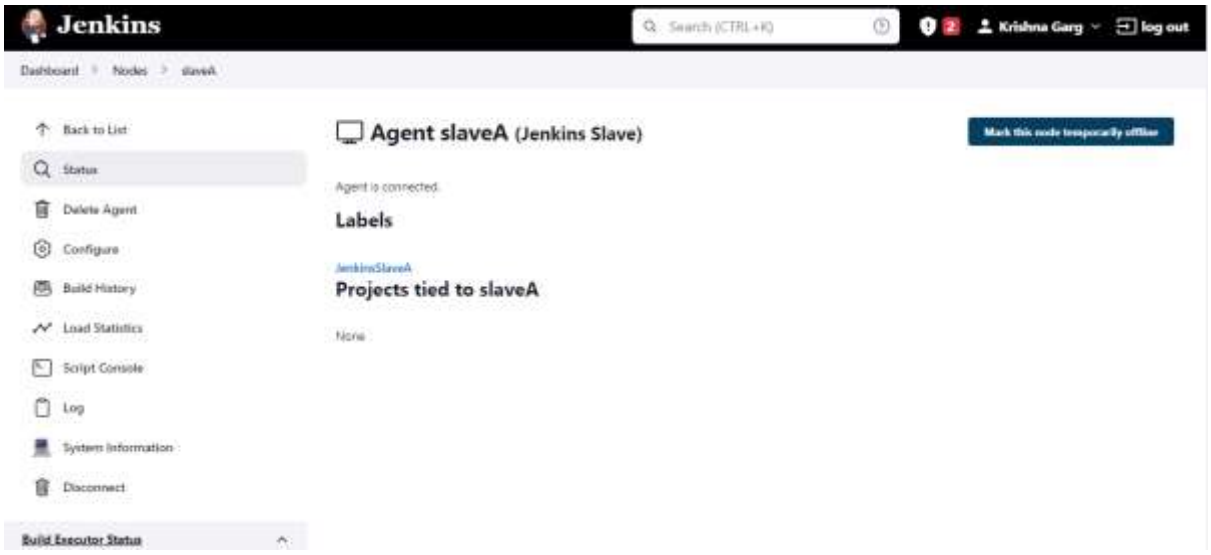

Jenkins
Search (CTRL+K)
Krishna Garg
log out
Dashboard > Manage Jenkins > Nodes >
Back to Dashboard
Manage Jenkins
New Node
Configure Clouds
Node Monitoring
Build Queue
No builds in the queue.
Build Executor Status
Built-in node + 1 agent (0 of 2 executors busy)
Manage nodes and clouds
Refresh status
S
Name
Architecture
Clock Difference
Free Disk Space
Free Swap Space
Free Temp Space
Response Time
Built-In Node
Windows 10 (amd64)
In sync
113.80 GB
5.20 GB
113.80 GB
0ms
slaveA
N/A
N/A
N/A
N/A
N/A
Data obtained
8 ms
7 ms
6 ms
9 min 48 sec
9 min 48 sec
9 min 48 sec
REST API
Jenkins 2.361.2

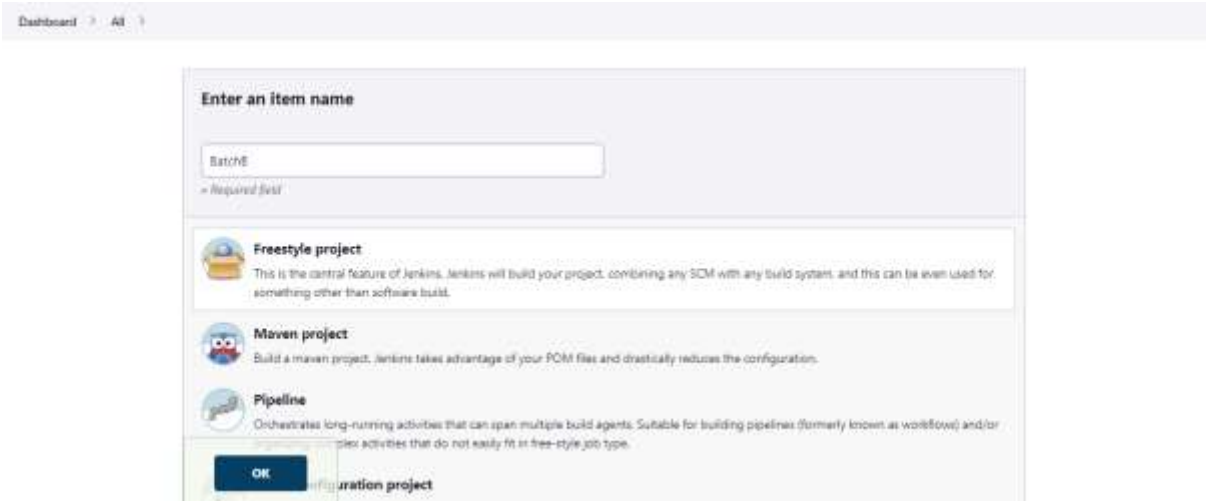
Connected to the Slave

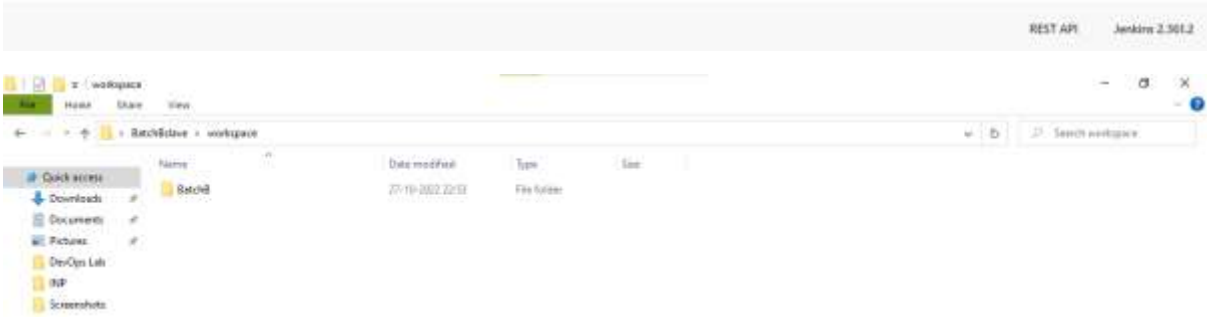


Now you can assign the task to the slave









Conclusion:

From this experiment, it is concluded that we have successfully understood the Jenkins Master – Slave Architecture and scaled our Jenkins standalone implementation by implementing slave nodes.

Hence, we have successfully achieved Lab Outcome (LO4).

Also, we have achieved PO1, PO2, PO3, PO4, PO5, and PO12 from this experiment.