IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

TRIGGERING BUILDS USING MASTER-AGENT IN JENKINS

[1]Shifal Shetty, [2]Chandrani Chakravorty
[1]PG Student, [2]Asst. Professor
[1]Department of Master of Computer Applications,
[1]*RV College of Engineering*, Bangalore, India

Abstract: Checking the equipment or product output manually takes time, and requires human help. This can be solved using Jenkins 'Master-Agent technique, it makes use of the Continuous Integration (CI) system, which is the developers' most commonly used tool for integrating their code into the repository. CI is the most powerful and speediest way of automating stuff. Jenkins is the most powerful method that allows for continuous integration. The paper covers the general Master-Agent setup architecture in Jenkins, explains the method of using CI technique. This paper explains how Jenkins originated as a pure Continuous Integration Framework and how it can be used to trigger builds using a master-agent configuration, as well as explaining the execution of tests, various Jenkins plugins, and what are their uses.

Introduction

Jenkins is an autonomous, open source automation platform that can be used to automate all sorts of tasks related to software development, testing, and distribution or deployment [1]. Jenkins can be built via Native System Packages, Docker, or even run by any Java Runtime Environment (JRE) built standalone computer. It provides numerous plugins that expand its use to projects in languages other than Java. In Jenkins, builds can be activated by a number of methods, such as committing in a version control system, scheduling through a cron-like process and requesting a particular build URL. It may also be activated after completion of the other builds in the queue. The DevOps automation framework is a continuous integration / continuous distribution and deployment (CI / CD) method written in the Java programming language. Used for the implementation of CI / CD workflows, called pipelines [2]. Jenkins runs on a number of platforms including Windows, MacOS, Unix and, in particular, Linux as a server. It includes a Java 8 VM and above and can be run on the OpenJDK or Oracle JRE. Jenkins normally operates inside a Jetty application server as a Java servlet. It will operate on other servers for the Java framework such as Apache Tomcat.

The most important stage in the software development process is checking the application or the program. By designing an automation model for the product company's test software or quality assurance the developer will verify the product quality [3]. Jenkins is an autonomous, open source Automation Platform that can be used to test, build and deploy the applications. The Jenkins' key role is to perform the job at a predefined time period, and the time interval will be fixed in Jenkins before execution. Jenkins retains the previous build outcome, and the build status can be submitted to the individual or team concerned.

Jenkins follows master- slave architecture, master runs the job on slaves running on multiple platforms, figure 1.1 demonstrates the block diagram of Jenkins Master-Agent node setup, where the master node consists of two services, that is labelled as Jenkins service and Jenkins slave service which is running in multiple platforms and these services are managed by the Jenkins slave pod. Jenkins route can be used to connect to different services which is outside the specific network [4].

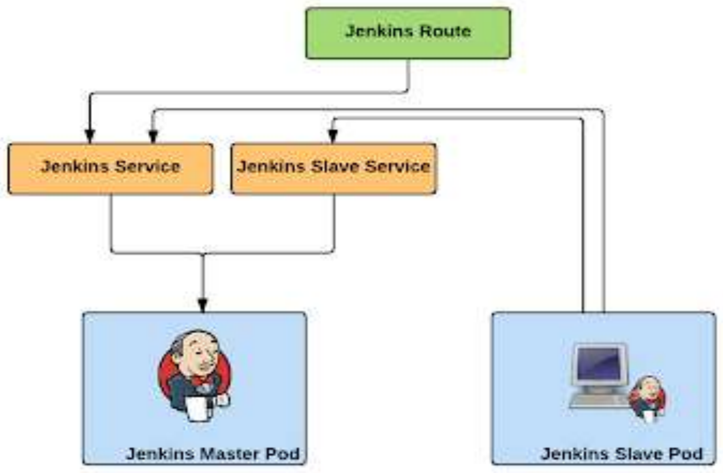


Figure 1.1: Block Diagram

I. JENKINS PLUGINS

A. Email Extension Plugin

This plugin is a swap for email publisher Jenkins. This is best done in class module with better elements for sending messages about the fabricate status. It would be best if you ensure that, as opposed to the entire party, you set up extremely granular and efficient methods for correspondence with important colleagues. One can design the body of the email as needed. Email plugin also offers an option to develop specific email body for failure and state of success [5].

B. Plugin git

This plugin incorporates GIT with Jenkins. The plugin often provides the aspect of informing the job on git code base changes via REST API. It is an incredibly helpful item because Git Hooks will easily inform the constructs, rather than constructing waiting for Git poll intervals [5].

C. Build Monitor plugin

Visibility is the essential aspect of the setup for the Jenkins CI. Understanding that CI is an item of collective ownership, everybody will know that if it fails, another portion of it doesn't work reliably. To make this a fact it is an admired step to see the awaited in a show visible to everyone. Build-Monitor Plugin provides a very apparent perspective on the status of Jenkins employments chosen. It basically needs different Windows frame sizes and is ideal for viewing on a computer on the office wall as an Extreme Feedback Device [5].

D. Build plugin

If a person is considering setting up a parallel task(job) in the pipeline, he may end up relying on the project in a situation referred to regularly as "diamond" form. It means there is a single parent position which will launch a few downstream workers. A solitary aggregation job runs after those jobs are done. This plugin allows all the prompt downstream occupations to be completed to the job. The execution can stretch along these lines and perform many steps in parallel, and only once after all the parallel work is done run a last collection step. With this plugin more complex connections are impractical [5].

E. Rebuild plugin

Often a stage in the pipeline(build) can fail due to some specific problem not detected with the associated repair, such as restarting Jenkins, or a memory question. The pipeline parameters shift number and produce number should be physically demonstrated with a clear end target to cause a revamp of the flopped downstream occupation which is quite unbalanced. The Rebuild plugin is proving useful here. It encourages modifying work from the failed assembly with an indistinguishable parameter [5].

F. SSH & SSH agent

Using SSH and SSH agent module, one can use the SSH Module to execute shell commands on remote computers, enabling you to control slaves operating on Unix computers over SSH. It requires some kind of tactic to disperse slaves. This dispatch technique will create an SSH link as the predefined username to the predetermined host. If it has a fair java implementation, the most current slave.jar is duplicated by SFTP. It begins the cycle of slaving [5].

G. Parameterized trigger plugin

This plugin allows you the ability to activate new types after the manufacture has ended, with different methods to signify parameters for the new type. You may have various arrangements: each has a collection of undertakings to trigger, a criterion for when to activate them (given the impact of the present form), and a section of parameters [5].

II. JENKINS MASTER-AGENT ARCHITECTURE

Following are the key components in Jenkins:
1. Jenkins Master Node
2. Jenkins Agent Node

`

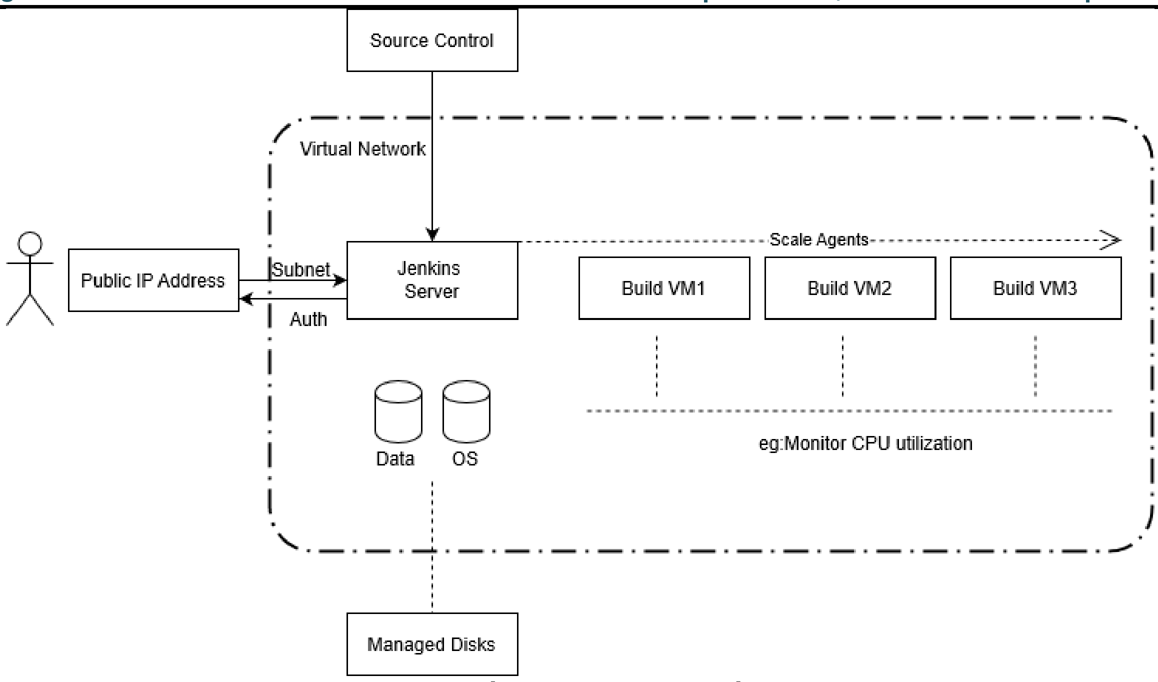


Figure 3.1: Jenkins Master-Agent Architecture

1. Jenkins Master Node: Jenkins's server or master node holds all key configuration. The following are the key Jenkins master components.

A. Jenkins Jobs: A job is a collection of steps which can be used to build source code, test code, run a shell script, or execute a remote host Ansible role [6]. There are several forms of employment required to sustain the software development & quality production process.

B. Jenkins Plugins: Plugins are modules developed by the community which can be installed on the Jenkins server. It can incorporate further functionality that aren't present in Jenkins natively.

C. Jenkins User: Jenkins has its own user database. This can be used for authenticating Jenkins users.
D. Jenkins Global Security: Jenkins has the two primary authentication methods.

a) Jenkins's own user database: Set of users that Jenkins own account contains.
b) LDAP Integration: Jenkins authentication using corporate Lightweight Directory Access Protocol (LDAP) configuration.

E. Jenkins Nodes / Clouds: Used for configuring several slave nodes (Linux / Windows) or clouds (docker, Kubernetes) to operate Jenkins work.

F. Jenkins Global Settings (Configure System): Under Jenkins' global setup, all combinations of activated plugins and native regional Jenkins settings can be identified and regional environment variables may also be modified under this portion.

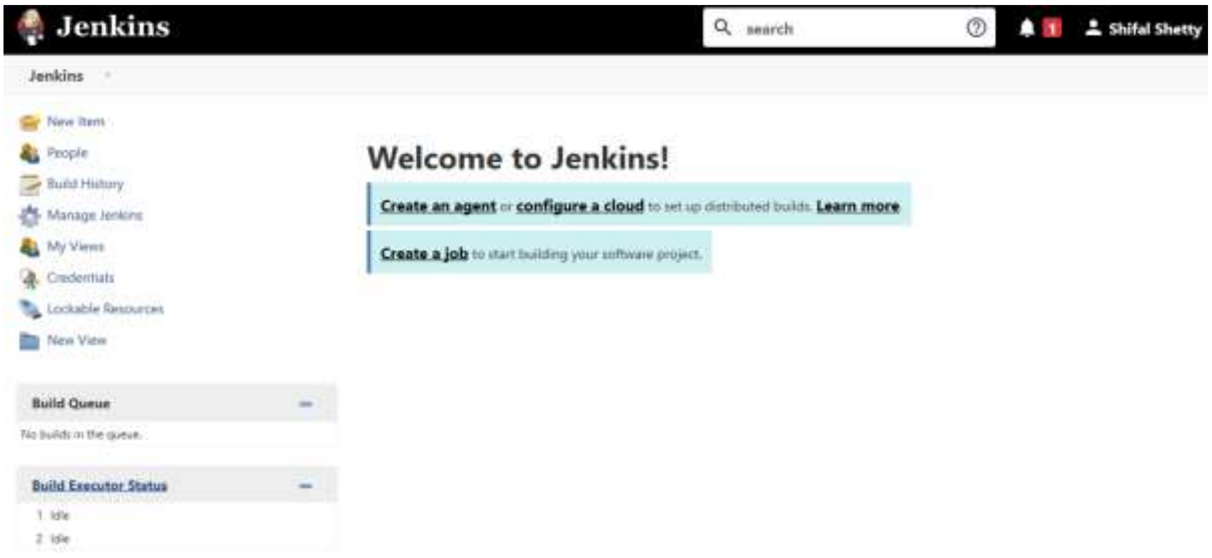
G. Jenkins Logs: Provides logging information on all Jenkins server actions including job logs, plugin logs, webhook logs, etc. [6].

2. Jenkins Slave Node: Jenkins slaves are the worker nodes in Jenkins system optimized for the workers. With a combination of Windows & Linux servers one can have any amount of Jenkins slaves connected to a boss. Even, depending on the usage case, one may limit work to operate on different slaves. For starters, if a user has a Java 8 configuration slave, then he may delegate this slave to jobs involving Java 8 setting. There is no universal method for utilizing the slaves, so a process so plan may be developed depending on the project needs [7].

III. Steps to Setup Master-Agent and Trigger Builds

Following are the steps to setup Master-Agent in Jenkins:

1. Open Jenkins on any browser by providing the URL: htpp://localhost:8080/(port number depends on setup)



Welcome to Jenkins!

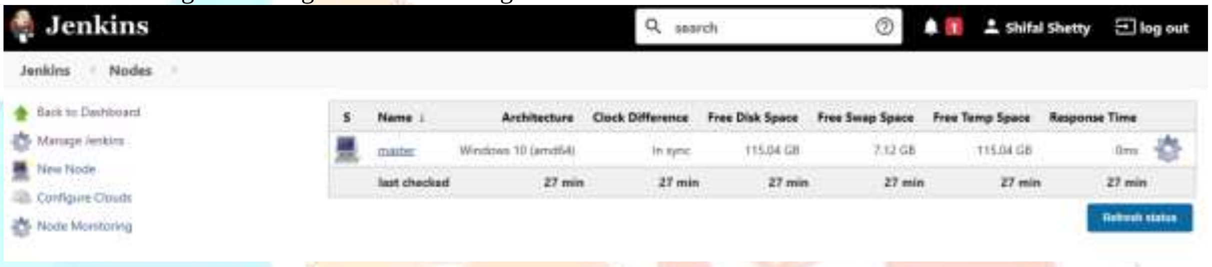
Create an agent or configure a cloud to set up distributed builds. Learn more

Create a job to start building your software project.

No builds in the queue.

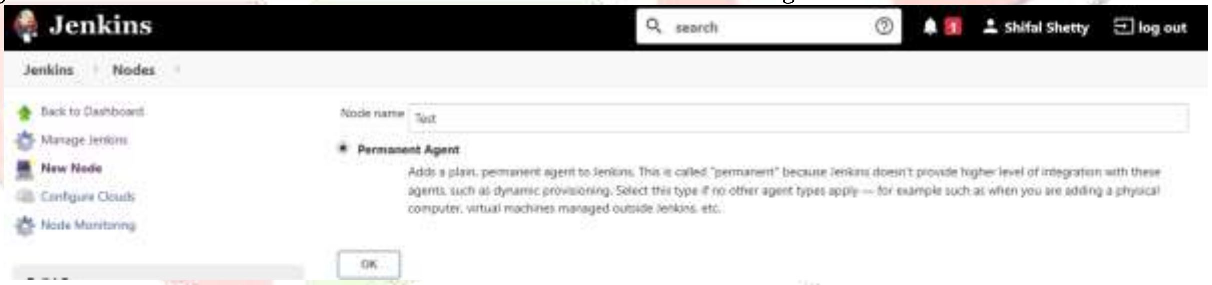
1 Idle
2 Idle

2. On your master machine go to Manage Jenkins > Manage Nodes



S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	master	Windows 10 (amd64)	In sync	115.04 GB	7.12 GB	115.04 GB	0ms
	last checked	27 min	27 min	27 min	27 min	27 min	27 min

3. Creating new Node: click on new node > Enter Node Name > Select Permanent Agent and Press OK.

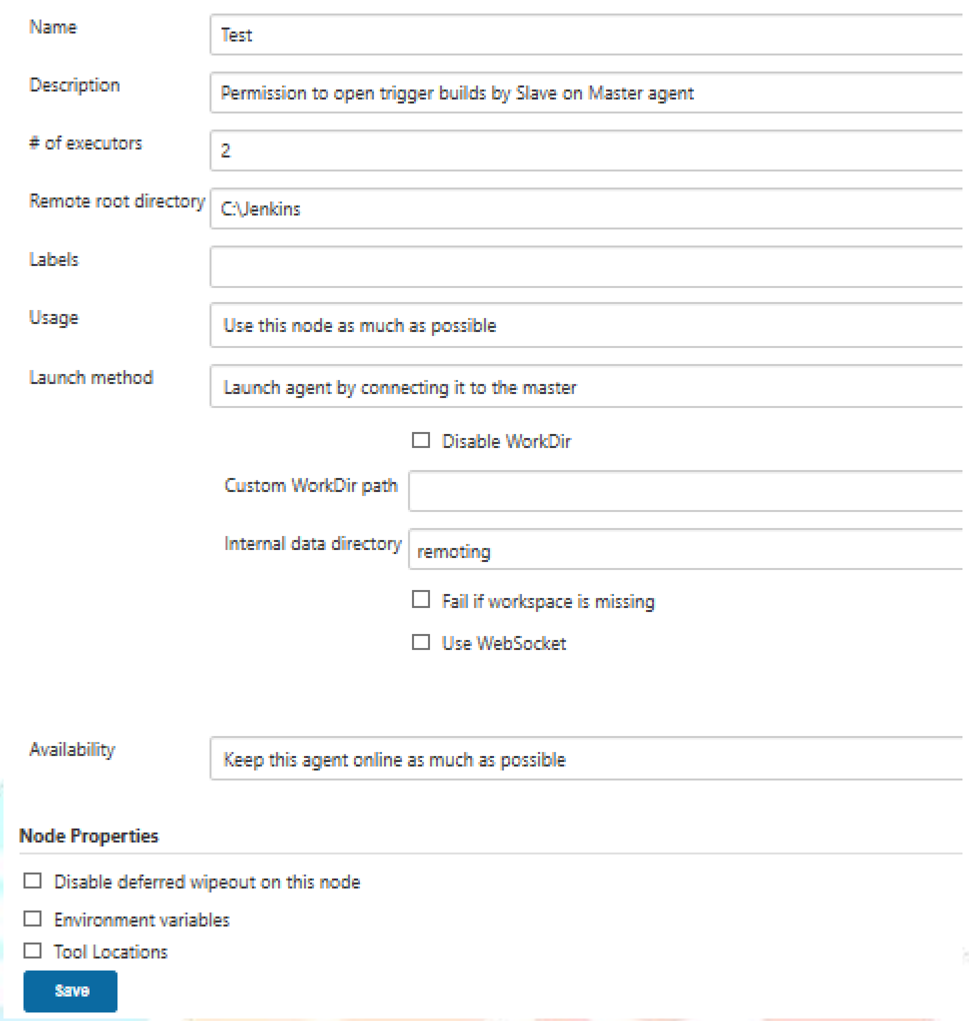


Node name: Test

● Permanent Agent

Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins doesn't provide higher level of integration with these agents, such as dynamic provisioning. Select this type if no other agent types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.

4. Fill it up as follows:
   a. Place an executor amount
      I. (One or more) Where appropriate
   b. Set a Remote Root Address
      i. Master's home directory on an agent computer
      ii. Using something like, for a Windows agent: "C:\Jenkins\" Select the appropriate Usage setting:
         i. To an additional worker: Make full use of this node
         ii. For professional work: Use this unit just for linked workers
   c. Launch Method: A simple way to monitor a Windows agent is via Java Web Start (Recommended for Windows) using the Launch command
   d. Availability: Keep this agent online and add the required information in the provided space and press OK



Name	Test
Description	Permission to open trigger builds by Slave on Master agent
# of executors	2
Remote root directory	C:\Jenkins
Labels	
Usage	Use this node as much as possible
Launch method	Launch agent by connecting it to the master

☐ Disable WorkDir

Custom WorkDir path:

Internal data directory: remoting

☐ Fail if workspace is missing

☐ Use WebSocket

Availability: Keep this agent online as much as possible

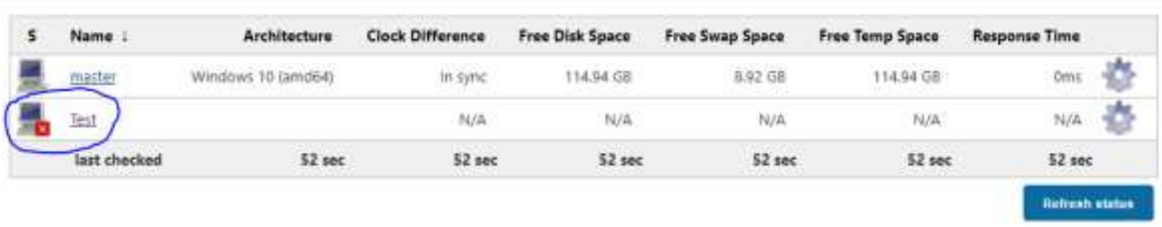
Node Properties

☐ Disable deferred wipeout on this node

☐ Environment variables

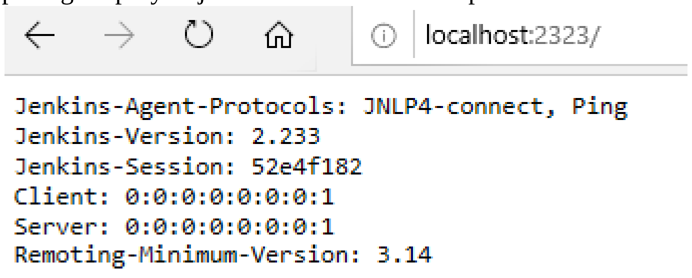
☐ Tool Locations

5. Now link to the master's agent computer using the measures below.
   a. Open a browser on the agent machine and go to the Jenkins master server URL http://yourjenkinsmaster:8080
   b. Go to Manage Jenkins > Manage Nodes,
   c. Click on the screen for freshly generated agent. When you have setup global protection, you may need to login to someone who has the "Link Agent" permission.
   d. To start agent from tab on server, press the Launch icon.

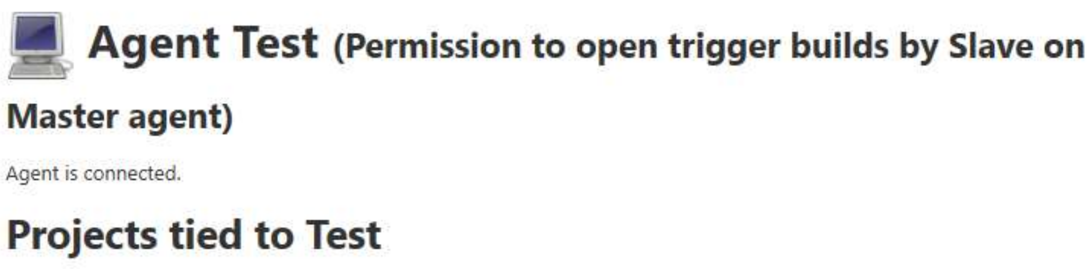


S	Name ↓	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	master	Windows 10 (amd64)	In sync	114.94 GB	8.92 GB	114.94 GB	0ms
	Test		N/A	N/A	N/A	N/A	N/A
	last checked	52 sec	52 sec	52 sec	52 sec	52 sec	52 sec

6. Test the connectivity of agent by opening: http://yourjenkinsmaster:8080 or http://localhost:2323



```
Jenkins-Agent-Protocols: JNLP4-connect, Ping
Jenkins-Version: 2.233
Jenkins-Session: 52e4f182
Client: 0:0:0:0:0:0:0:1
Server: 0:0:0:0:0:0:0:1
Remoting-Minimum-Version: 3.14
```



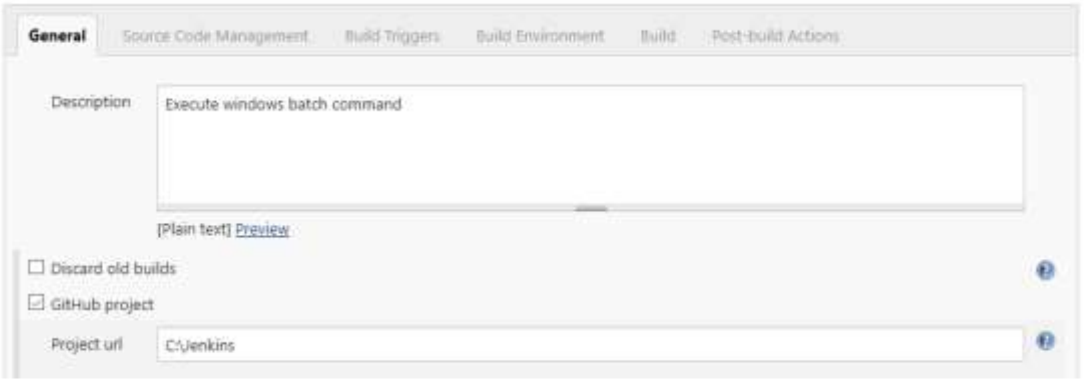
Agent Test (Permission to open trigger builds by Slave on Master agent)

Agent is connected.

Projects tied to Test

Following is an example to setup Builds to Trigger using Master-Agent in Jenkins:

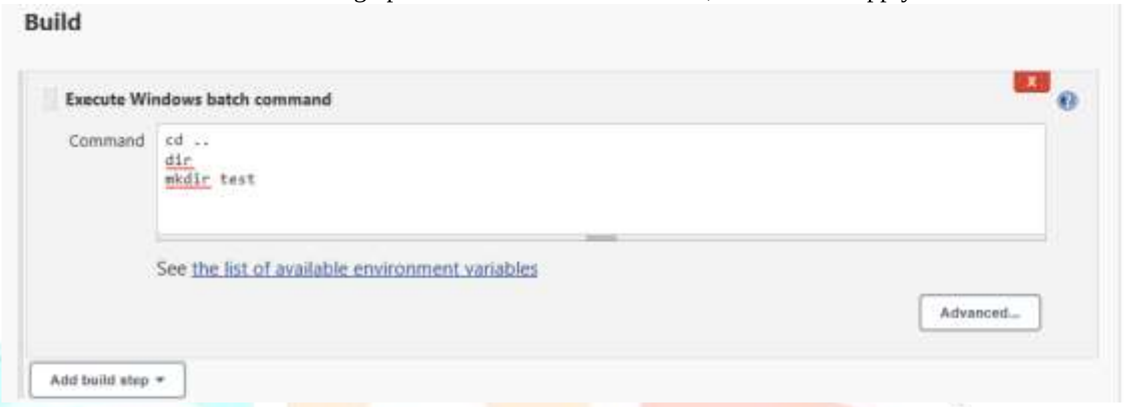
1. Go to Jenkins dashboard and press New Element to determine the Task info



Description: Execute windows batch command

Project url: C:\Jenkins

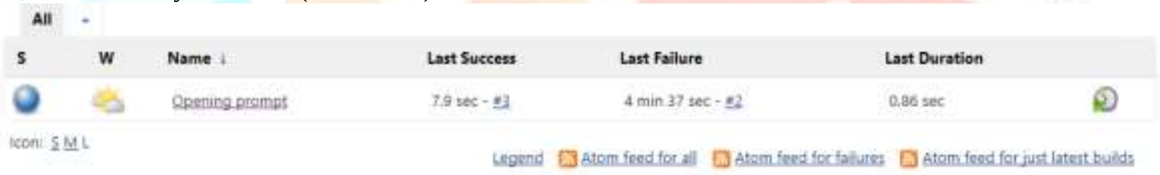
2. Specify the windows batch command for setting up builds in Build Environment, and click on apply and save:


Build
Execute Windows batch command
See the list of available environment variables
Advanced...
Add build step

Command:

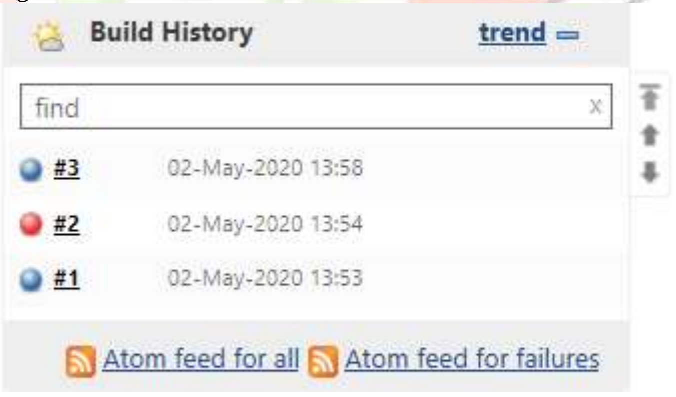
```
cd ..
dir
mkdir test
```

3. To cause builds, click on the symbol Run (bottom left):



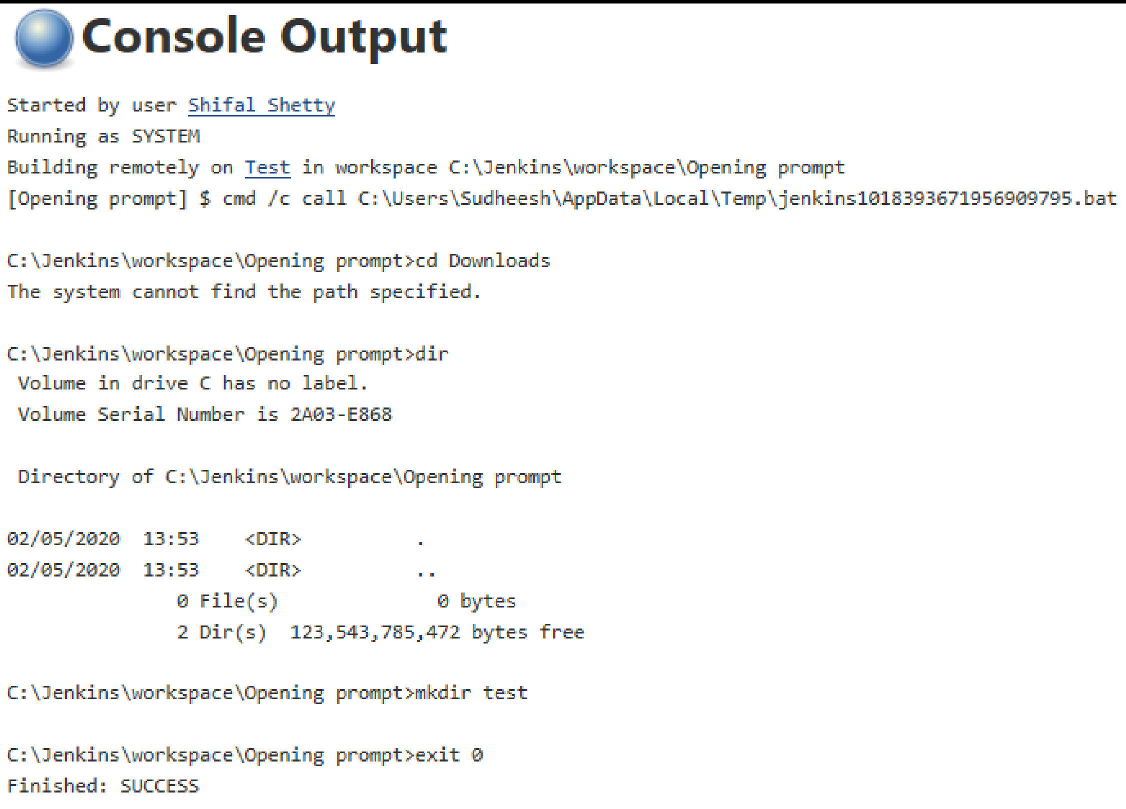
S	W	Name ↓	Last Success	Last Failure	Last Duration
●		Opening prompt	7.9 sec - #3	4 min 37 sec - #2	0.86 sec

4. To see the results of the build, click on Build History on the right-side menu: Red color indicates failure in builds, Blue indicates build completed, Green indicates build in progress.



#3	02-May-2020 13:58
#2	02-May-2020 13:54
#1	02-May-2020 13:53

5. To See the build logs, click on Console output screen:



Console Output

```
Started by user Shifal Shetty
Running as SYSTEM
Building remotely on Test in workspace C:\Jenkins\workspace\Opening prompt
[Opening prompt] $ cmd /c call C:\Users\Sudheesh\AppData\Local\Temp\jenkins1018393671956909795.bat

C:\Jenkins\workspace\Opening prompt>cd Downloads
The system cannot find the path specified.

C:\Jenkins\workspace\Opening prompt>dir
 Volume in drive C has no label.
 Volume Serial Number is 2A03-E868

 Directory of C:\Jenkins\workspace\Opening prompt

02/05/2020  13:53    <DIR>          .
02/05/2020  13:53    <DIR>          ..
               0 File(s)              0 bytes
               2 Dir(s)  123,543,785,472 bytes free

C:\Jenkins\workspace\Opening prompt>mkdir test

C:\Jenkins\workspace\Opening prompt>exit 0
Finished: SUCCESS
```

V. ADVANTAGES OF TRIGERING BUILDS IN JENKINS

1. Jenkins reduces the commitment of repetitive coding: A command prompt file may be translated to a press of a GUI button with the Jenkins uses. That can be achieved by getting the script rolled up as a Jenkins work. It is possible to build parameterized Jenkins jobs for configuration or user feedback. qaSo, one can save hundreds of lines of computer writing.

2. Integration of Individual Jobs: Jenkins workers are usually tiny devices. They serve low, often very basic purposes. Jenkins offers pipeline plugin which can be used to merge several works. Pipelining provides such advantages that Linux users understand more than anyone else. These may be paired sequentially or in tandem.

3. Greater data support for project management: For project management, each activity is wrapped as a Jenkins job. For each Jenkins job, success or failure can be identified, and job completion time can be measured.

4. Manual Tests option: Often things perform well geographically, but when put into a central structure they collapse. It is as circumstances shift around the time they drive. Continuous Integration checks the technology against the existing condition of a data base which is conducted in a production setting.

5. Decrease Code Review Time: CI systems such as Jenkins and Version Control System such as Git are capable of interacting with each other and advising users when a merge request is necessary. Typically, that occurs because all the exams are taken, and no other conditions are fulfilled. Furthermore, the disparity in code coverage can often be stated in the request for merge itself. This greatly decreases the time required to process a proposal for a merge.

6. Increased Code Coverage: CI servers like Jenkins will search the application for coverage of checks. Tests expand accuracy of the language. That inspires transparency and accountability in members of the team. Test results are shown on the development pipeline, ensuring that team leaders obey the appropriate guild lines. Similar to application analysis, code coverage guarantees that checking is a straightforward mechanism amongst team members.

VI. CONCLUSION

Triggering Builds using Master-Agent method in Jenkins by using Continuous integration technique is a need on complex tasks because of the advantages it gives on early detection of the problems. Since the process is automated, we would have fewer reliance on test engineers as it reduces the preparation period that test engineers need. This often reduces the expected human mistakes because test engineers are far less interested. The execution and reporting are not done manually, this reduces the cycle time of test execution. Thus, using Jenkins to trigger builds will give better performance and results.

VII. BIBLIOGRAPHY

[1] Arpitha R & Mrs. Kavitha S, "Automation Using Jenkins: Plugins, TestDesign", Imperial Journal of Interdisciplinary Research, Vol-3, Issue-5, 2017 ISSN: 2454-1362

[2] Jen-Hao Teng; Shun-Yu Chan; Jin-Chang Lee; R. Lee, "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices", PowerCon 2017. 2017 International Conference on Power System Technology. Proceedings (Cat. No.00EX409), ISBN: 0-7803-6338-8

[3] CHEN Jian, WANG Ying, ZHOU Q, IA Rui wu, PENG Guang zheng and FAN Wei, "A Survey of Continuous Integration, Continuous Delivery and Continuous Deployment", (Dept. of Automatic Control, Beijing Institute of Technology, Beijing100081, Journal of Beijing Institute of Technology;2002-01

[4] Valentina Armenise, "Continuous delivery with Jenkins: Jenkins solutions to implement continuous delivery", RELENG '15: Proceedings of the Third International Workshop on Release Engineering

[5] Mojtaba Shahina, Muhammad Ali Babara and Liming Zhub, "Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices", The Centre for Research on Engineering Software Technologies, The University of Adelaide, Australia b Data61, Commonwealth Scientific and Industrial Research Organisation, Sydney, NSW 2015, Australia

[6] Rahman Jamal, "Continuous Delivery with Jenkins: Jenkins Solutions to Implement Continuous Delivery", Conference:2015 IEEE/ACM 3rd International Workshop on Release Engineering (RELENG)

[7] Pavol Spanik, Libor Hargas, Miroslav Hrianka and Ivan Kozehuba, "Build Optimization Using Jenkins", Dept of E&C, Sri JC College of Engineering, JSS Science and Technology University, Mysuru, India