Day 28 Task: Jenkins Agents

Jenkins Master (Server)

The Jenkins Master acts as the control centre for all the activities that take place within a Jenkins environment. It manages the distribution of tasks to different Jenkins agents (also called "slaves" or "nodes"), which are responsible for performing the actual builds, tests, and deployments.

The Jenkins Master also provides a web-based user interface (UI) for managing the Jenkins environment, configuring jobs, and viewing build logs and reports. It allows users to define different job types, such as freestyle jobs, pipeline jobs, and multi-configuration jobs, to automate different aspects of the software development process.

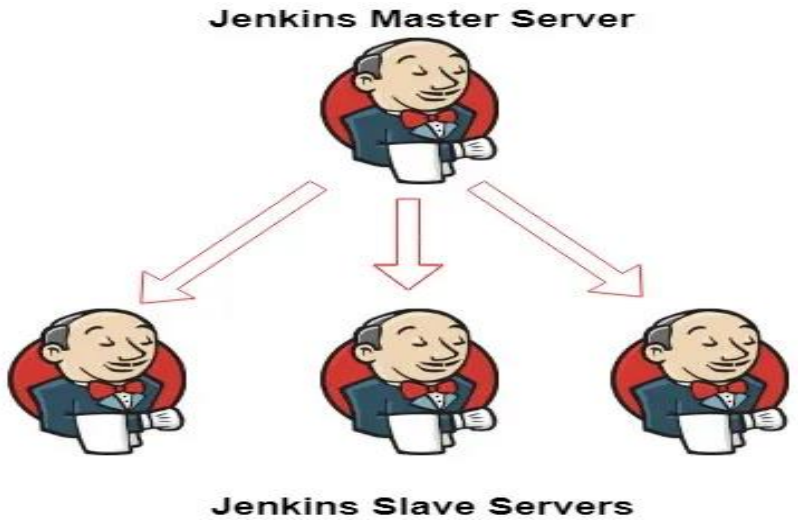
Jenkins Agent

Jenkins Agent, also known as a "slave" or "node", is a separate software component that works with the Jenkins Master (Server) to execute build, test, and deployment tasks in a distributed manner.

When a build or test job is submitted to Jenkins, the Jenkins Master assigns the task to an available Agent that meets the job's requirements. The Agent then downloads the necessary code, dependencies, and configurations from the Jenkins Master and performs the build or test task in a separate workspace.

After the task is complete, the Agent reports the result back to the Jenkins Master, which consolidates the results from all Agents and generates a report that can be viewed through the web-based user interface.

Jenkins Agents provide the ability to distribute build and test workloads across multiple machines, which can significantly improve the performance and scalability of a Jenkins environment. By allowing for parallel execution of tasks, Jenkins Agents can help to reduce the time required to complete builds and tests, and improve the overall productivity of development teams.



Pre-requisites

Let's say we're starting with a fresh Ubuntu 22.04 Linux installation. To get an agent working make sure you install Java (same version as jenkins master server) and Docker on it.

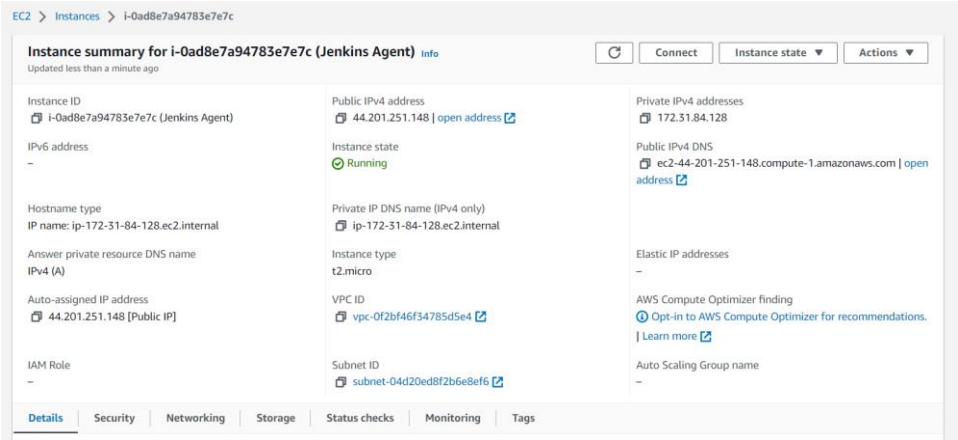
Note:- While creating an agent, be sure to separate rights, permissions, and ownership for jenkins users.

Task-01

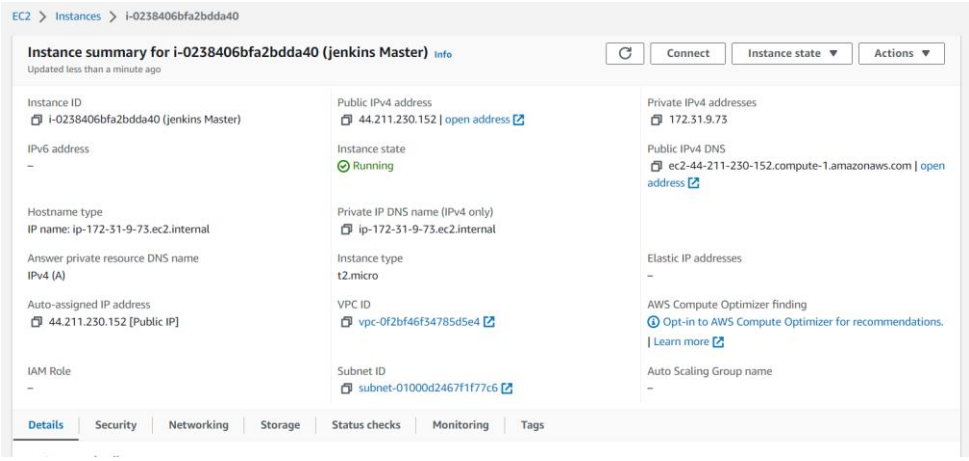
- Create a new AWS EC2 Instance and connect it to master (Where Jenkins is installed)

Here I have created a new EC2 instance called Jenkins Agent which I am going to connect it to another instance called Jenkins master where Jenkins and docker are already installed

1. Jenkins Agent



2. Jenkins Master

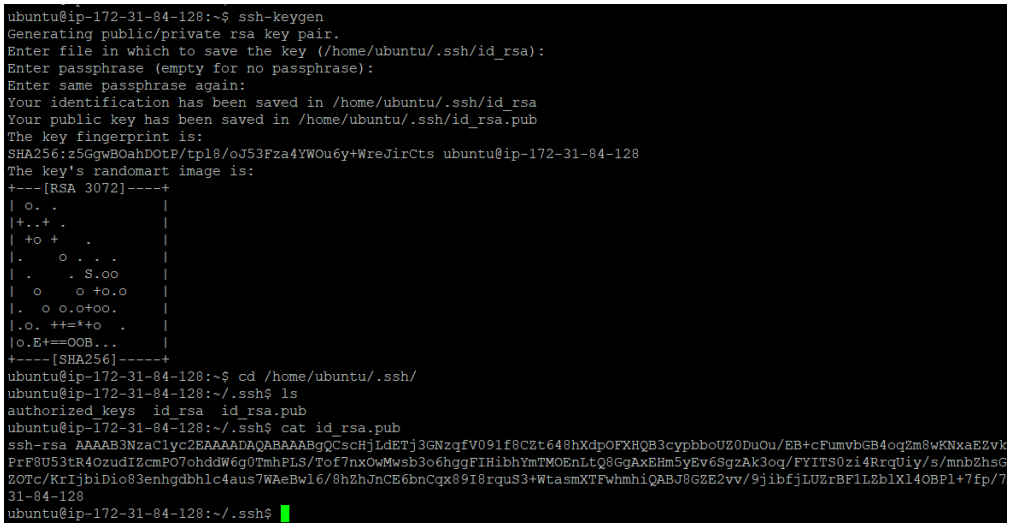


- The connection of master and agent requires SSH and the public-private key pair exchange.

Step 1 - Generate SSH keys on "Jenkins-agent" EC2 instance

Step 2 - Add public key from "Jenkins-agent" instance to "Jenkins-master" instance under location ".ssh/authorized_keys"

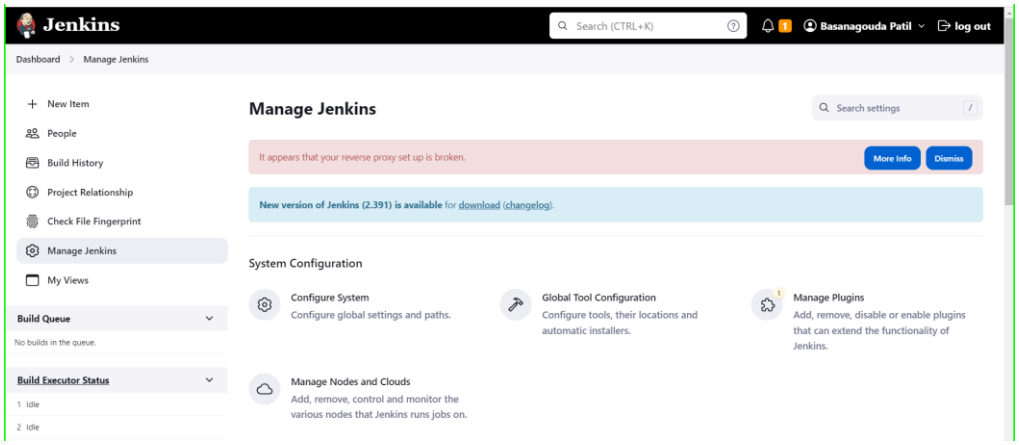
Id_rsa.pub key from Jenkins agent :



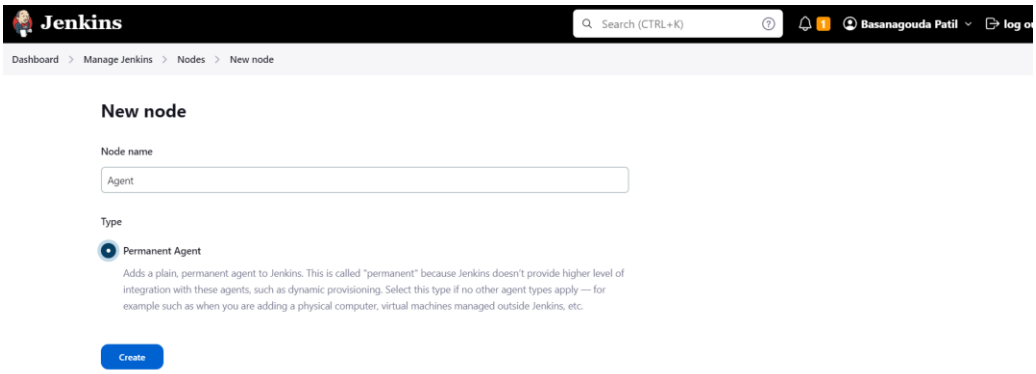
authorized key from Jenkins master :

- **The connection of master and agent requires SSH and the public-private key pair exchange.**

Step 1 – Login to the Jenkins and go to dashboard, and click on “Manage Jenkins” then click on “Manage Nodes and Clouds”



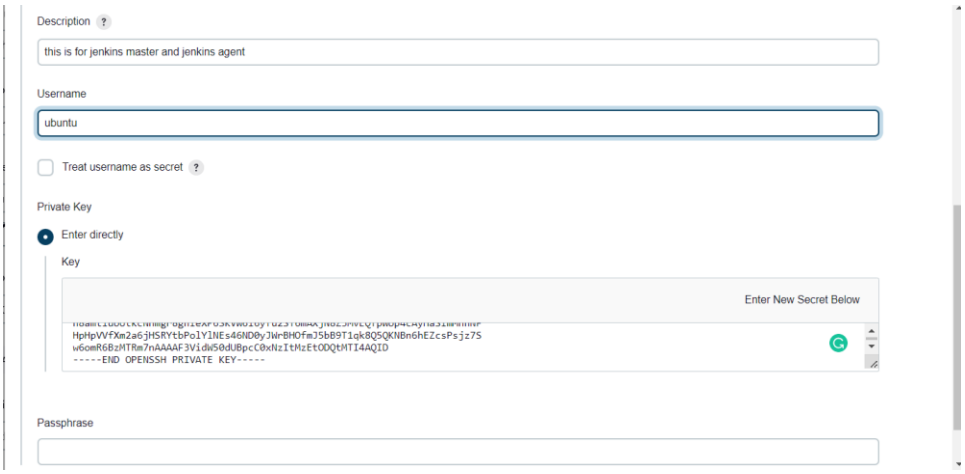
Step 2 - To create a node click on "New Node" and give name to it and make it permanent agent then click on create.



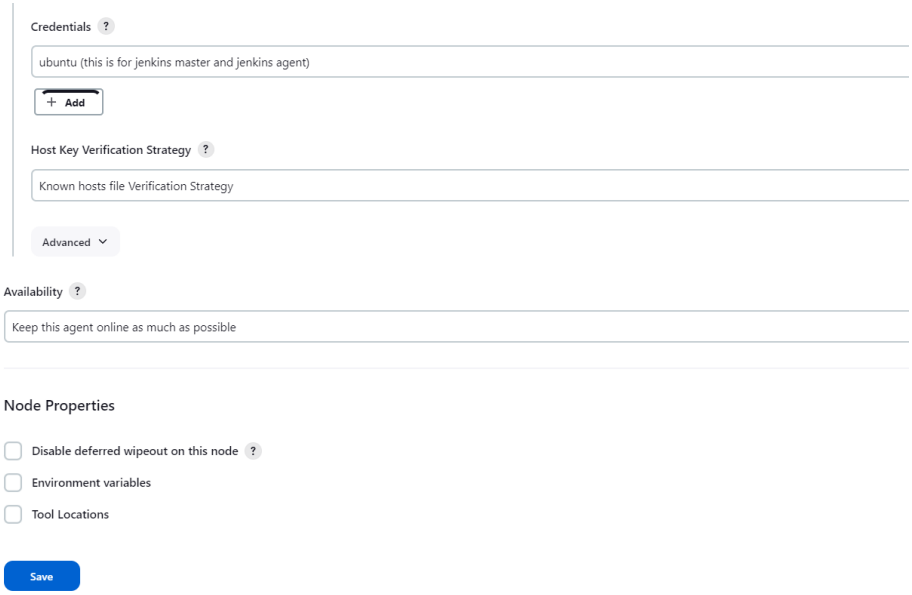
Step 3 - Add details to the node, accordingly.

Step 4 – In launch method select Launch agents via SSH and click on ADD Credentials

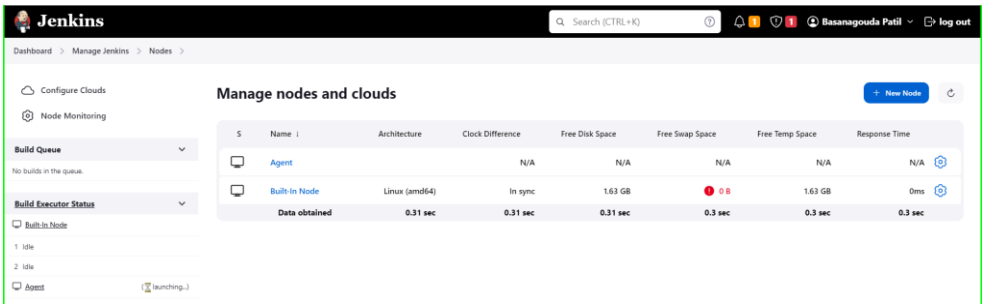
Step 5 – Give description according and past the private key that we created in 'jenkins-agent' instance using ssh-keygen.



Step 6 - Below we select ubuntu for credentials and click on 'save' that will create node.



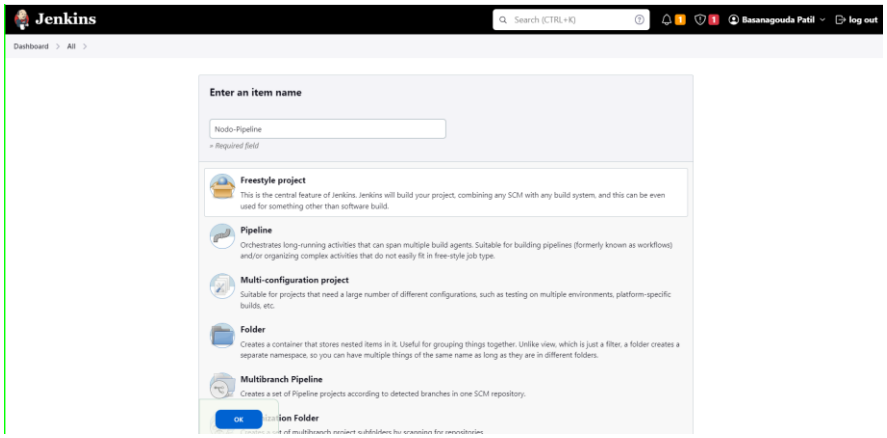
- **Verify its status under "Nodes" section.**



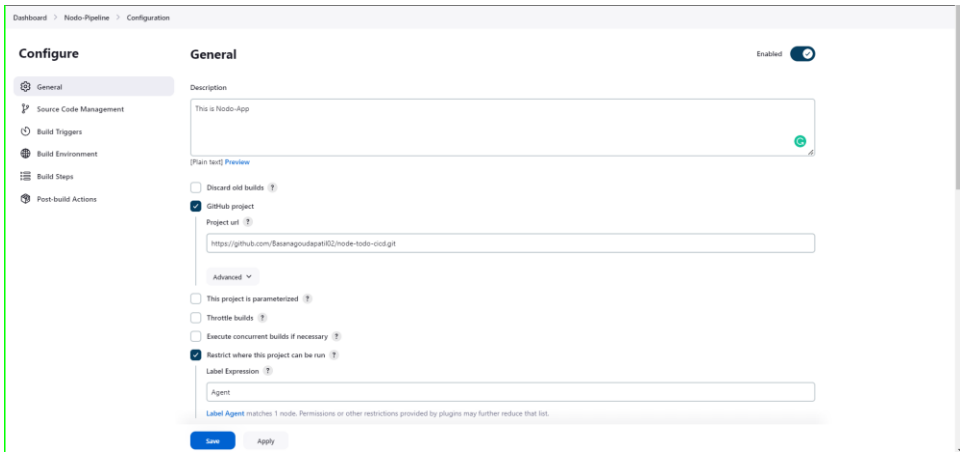
Task-02

- Run your previous Jobs (which you built on Day 26, and Day 27) on the new agent
- Use labels for the agent, your master server should trigger builds for the agent server.

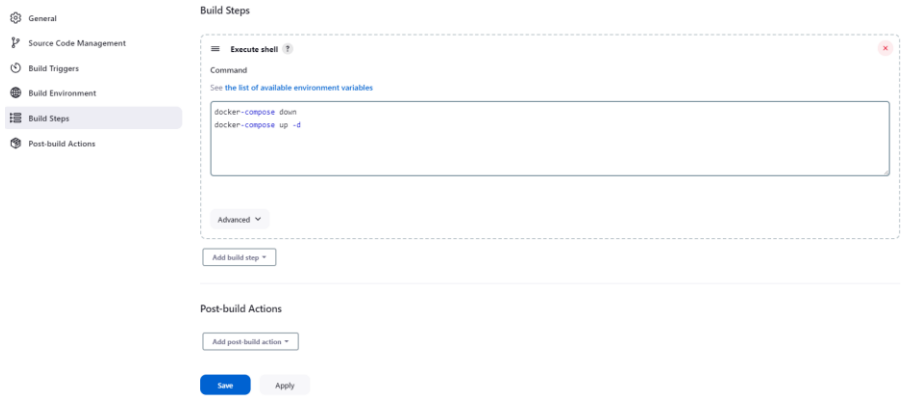
Step -1 Click on “New Item” and enter desired name and select “Freestyle project” and click on Ok.



Step 2 – Give description, select GitHub project and enter your repository url from where you will clone all files of your project and add label expressions, in that add your node label that you created.

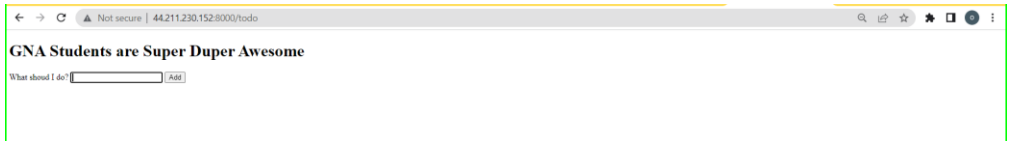


Step 3 – In build Steps select Execute shell and write commands to run , Click on save.



Step 4 - Click on build and check your project console output.

Past your instance public Ip in browse to check web app is running



Happy Learning:)