V CUBE
Software Solutions Pvt. Ltd.

NAME: VOMMIDAPU CHINNI **BATCH NO:** 120

MAIL.ID: vommidapuchinni@gmail.com **DATE:** 04/05/24

TRAINER: Mr. Madhukar **MODULE:** JENKINS

ASSIGNMENT – 7

Write a brief explanation master slave and webhook

Jenkins, being a continuous integration and continuous delivery (CI/CD) tool, can be set up in a master-slave architecture to distribute the workload across multiple machines for improved performance and scalability. It is also called as Jenkins distributed builds.

Master: The Jenkins master is the central server that manages the entire Jenkins environment. It handles tasks such as job scheduling, monitoring, and distributing build jobs to the slave nodes.

Slave(node): A Jenkins slave is a separate machine that offloads build jobs from the master. Each slave connects to the master and waits for instructions to execute build jobs. Slaves can be configured to have specific environments, tools, or dependencies, allowing for parallel execution of builds on different platforms or configurations.

The master-slave architecture in Jenkins offers some benefits**:**

- Scalability: Additional slaves can be added to distribute workload and accommodate more build jobs.
- Performance: Builds can be executed in parallel across multiple nodes, reducing build times.
- Resource Isolation: Slaves can have dedicated environments, ensuring that builds do not interfere with each other.

WEBHOOK: A webhook in Jenkins allows external services to trigger Jenkins jobs automatically whenever certain events occur, such as code commits, pull requests, or issue updates in a version control system like Git.

If you wish to automate the build process in the multibranch pipeline we can use Webhook. This feature is not enabled until we install “Multibranch Scan Webhook Trigger”. This enables an option “scan by webhook” under “Scan Multibranch Pipeline Triggers”. Here we should give a token. I am giving it as “mytoken”. by this time your job looks something like below.

Integration: Webhooks facilitate seamless integration between Jenkins and other tools in the development ecosystem.

Automation: By triggering builds automatically based on events, webhooks help automate the CI/CD pipeline, reducing manual intervention and improving efficiency.

Realtime update: Builds are initiated immediately after the webhook receives the event notification, ensuring that the latest changes are tested and deployed promptly.

Setting up webhooks involves configuring Jenkins to listen for incoming HTTP requests from the external service and specifying the job to be triggered upon receiving the webhook payload.