***JENKINS***

***Jenkins*** is an open source tool with a plugin built for continuous integration purposes. The principal functionality of Jenkins is to keep a track of the version control system and to initiate and monitor a build system if changes occur. It monitors the whole process and provides reports and notifications to alert.

**CI/CD Definition :**

***“****CI/CD stands for Continuous Integration and Continuous Delivery (or Continuous Deployment), it is about how an Integrated code on a shared repository is used to release software to production multiple times a day with the help of automation.****”***

**Continuous Integration :**

Continuous Integration is about how developers integrate code using a shared repository multiple times a day with the help of automation.

**Continuous Delivery :**

Continuous Delivery is about automatically releasing software to the test or production environment.

**Continuous Integration and Continuous Delivery :** Continuous delivery is an extension of continuous integration since it automatically deploys all code changes to a testing and/or production environment after the build stage.

>To use Jenkins we require:

-- A source code repository which is accessible, for instance, a Git repository A working build script, e.g., a Maven script, checked into the repository.

>To start Jenkins manually, you can use either of the following:

(Jenkins\_url)/restart: Forces a restart without waiting for builds to complete

(Jenkin\_url)/safeRestart: Allows all running builds to complete

>Some of the important plugins in Jenkin includes:

* Maven 2 project
* Amazon EC2
* HTML publisher
* Copy artifact
* Join
* Green Balls

>To deploy a custom field of a core plugin, you have to do following things:

-- Stop Jenkins

-- Copy the custom HPI to $Jenkins\_Home/plugins

-- Delete the previously expanded plugin directory

-- Make an empty file called <plugin>.hpi.pinned

-- Start Jenkins

**>Advantage of Jenkins include:**

-- At integration stage, build failures are cached.

-- For each code commit change an automatic build report notification is generated.

-- To notify developers about build report success or failure, it is integrated with LDAP mail server.

-- Achieves continuous integration agile development and test driven development.

-- With simple steps, the maven release project is automated.

-- Easy tracking of bugs at an early stage in the development environment than production.

**CI/CD within the SDLC:**

1.SDLC starts with the developer, the developer will write some codes, some software.

2.Then the software needs to be Built, the Developer will use Feature and master branches (if using git), he will properly develop his features in separate branches over the code. Feature and master branches both need to build and compile.

What is going to be happening now is, we are going to use Jenkins to do the Build, Jenkins will compile Feature and master branches any these fails it can sense the developer with some feedback. It will say that the builds have failed because of some compile error/syntax error, So the developer can make some changes to commit those changes to version control and build it again.

3. After the build process is going to have the test process, there are lots of tests to test the software/feature which they really work or not.

4. If build & test are successful, then the Release process can start First, to release the software it needs to be Package, it can be Package in the archive, zip or package in a container in a VM based on the requirements.

5. Once it Packaged then move to Provision servers and Deploy our software on the servers

6. Then Customer can use the application

7,8. This process needs to be Planned and Monitored, we need to make sure that we ensure that the Process whenever developers do some codes we want to release it quickly as possible to the customer.

**Jenkins Features:**

1.Easy Installation

2.Easy Configuration

3.Available Plugins

4.Extensible

5.Easy Distribution

6.Free Open Source

-- Jenkins Freestyle Project is a repeatable build job, script, or pipeline that contains steps and post-build actions. It is an improved job or task that can span multiple operations. It allows you to configure build triggers and offers project-based security for your Jenkins project.

**Jenkins Pipeline:** In Jenkins, a pipeline is a collection of events or jobs which are interlinked with one another in a sequence.It is a combination of plugins that support the integration and implementation of continuous delivery pipelines using Jenkins.

**JenkinsFile:** Jenkins Pipeline can be defined by a text file called JenkinsFile. You can implement the pipeline as code using JenkinsFile, and this can be defined by using a DSL (Domain Specific Language).

**Pipeline syntax:**

>Two types of syntax are used for defining your JenkinsFile.

1.Declarative: Declarative pipeline syntax offers a simple way to create pipelines. It consists of a predefined hierarchy to create Jenkins pipelines. It provides you the ability to control all aspects of a pipeline execution in a simple, straightforward manner.

2.Scripted:Scripted Jenkins pipeline syntax runs on the Jenkins master with the help of a lightweight executor. It uses very few resources to convert the pipeline into atomic commands.

**Jenkins Pipeline Concepts:**

**Pipeline:** This is the user-defined block, which contains all the processes such as build, test, deploy, etc. it is a group of all the stages in a JenkinsFile. All the stages and steps are defined in this block. It is used in declarative pipeline syntax.

pipeline { }

**Node:** The node is a machine on which Jenkins runs is called a node. A node block is used in scripted pipeline syntax.

node{ }

**Stage:** This block contains a series of steps in a pipeline. i.e., build, test, and deploy processes all come together in a stage. Generally, a stage block visualizes the Jenkins pipeline process.

**Step:** A step is a single task that executes a specific process at a defined time. A pipeline involves a series of steps defined within a stage block.

**>**The Gradle Artifactory Plugin allows you to deploy your build artifacts and build information to Artifactory and also to resolve your build dependencies from Artifactory. The minimum supported Gradle version is 4.10.

**To create a backup and copy files in Jenkins:** Jenkins saves all the settings, builds artifacts and logs in its home directory, to create a back-up of your Jenkins setup, just copy this directory. You can also copy a job directory to clone or replicate a job or rename the directory.

**Clone a Git repository via Jenkins:** To clone a Git repository via Jenkins, you have to enter the email and username for your Jenkins system. For that, you have to switch into your job directory and execute the “git config” command.

**To set up Jenkins job:** To create a project that is handled via jobs in Jenkins. Select New item from the menu, once this done enter a name for the job and select free-style job. Then click OK to create a new job in Jenkins..

**>**Database library plugin defines a common abstraction to connect to relational databases. By doing so, it serves three purposes:

-- It allows database driver plugins (such as H2 Database Plugin) to be developed, improving the user experience for administrators.

--It simplifies other plugins that use RDBMS by eliminating the need to code up a configuration UI to let administrators select a database.The key class in this plugin is the Database class, which acts as a holder for JDBC DataSource instances. The Database class is an extension point to be implemented by database driver plugins.

--The calling code can use this DataSource instance to obtain a connection to the database.

**Jenkins security:** Access Control is the primary mechanism for securing a Jenkins environment against unauthorized usage.

>Two facets of configuration are necessary for configuring Access Control in Jenkins:

A Security Realm which informs the Jenkins environment how and where to pull user (or identity) information from. Also commonly known as "authentication."

>Authorization configuration which informs the Jenkins environment as to which users and/or groups can access which aspects of Jenkins, and to what extent.