**Udemy learning part-4**

* **Meaning of cherry pick:**
* Cherry pick is a command that allows you to apply a specific commit from one branch to another.

Git cherrypick <commit id>

* **Git tags:**
* Git tags are used to mark specific points in a repository’s history, making it easier to reference and track changes.
* Tags are used to mark a commit stage as relevant. we can tag a commit for future reference.
* Tags are much like branches, and they do not change once initiated. We can have any number of tags on a branch or different branches.
* Primarily it is used to mark a projects initial point like V1.1.
* **To see all tags**
* Git tag -l
* **Developer is done with the project, and they want to release it. How can they release it.**
* Git tag -a “Release\_V1.0” -m “Tagged Project A Version1.0”
* **To check the version**
* Git tag Release\_V1.0 (this is done in our local machine we want to push into remote repository)
* Git push origin tag Release\_V1.0
* **How to delete the release**
* Git tag -d <version> (it deletes only in local machine)
* **How to delete in remote repository**
* Git push --delete origin <version> (it deletes the version in remotely)
* **What is rebase and what is the difference between rebase and merge.**
* Rebase is a git command that replays your commits on top of another branch.
* Merge is a git command that combines the changes from two branches by creating a new merge commit.
* The purpose of both the commands is same. The purpose is to merge one branch content into another branch.
* **What is Jenkins?**
* Jenkins a is a free and open-source automation tool.
* It helps automate the parts of software deployment related to building, testing, and deployment which supports continuous integration and continuous delivery.
* With Jenkins, organizations can accelerate the software development process through automation.
* Jenkins integrate development life cycle processes of all kinds, including build, document package, stage, deploy, static analysis and much more.
* Jenkins achieve CI with the help of plugins. Plugins allow the integration of various devops stages. If you want to integrate a particular tool, you need to install the plugins for that tool.
* For example: git, maven2 project, amazon ec2, HTML publisher etc.
* **Integrate jenkins with GitHub repository:**
* **How to get the code from github to jenkins**
* Username and password approach
* Ssh keys approach
* Personal access tocken approach
* Then it will build the project (using maven for java)

**Prerequisites:**

* Git
* Java
* Apache maven
* Create one ec2 instance
* Execute Sudo su(to switch root user)
* Yum install git (installed git)
* Install java
* Install maven (sudo wget <url of maven>)
* Install jenkins(go to google- install jenkins-install on linux-select long term support release-copy all urls one by one).
* **Configure jenkins on ec2**
* Enter administrator password
* Click on installed jenkins plugins
* Create username, password, confirm password, full name, and email.
* Click on save and continue
* Click on save and finish
* Jenkins dashboard displayed
* **Configure GitHub credentials and maven location on jenkins:**
* Click on manage jenkins
* Under security click on manage credentials
* click on jenkins
* click on global credentials
* Click on add credentials
* Click on down arrow select username and password
* Enter username and password (GitHub username and password)
* Enter id (GitHub credentials)
* Description (added GitHub username and password)
* Click on create
* Go to dashboard (need to provide the location of maven)
* Click on manage jenkins
* Click on global tool configuration
* Click on add maven
* Give name(maven-3.2.5)
* Enter home directory of maven
* Click on apply and save.
* **I have to create job**
* Click on new item
* Enter job name
* Click on freestyle project
* Click on ok
* Give description (about project)
* Go to source code management
* Select git
* Go to github select repository and copy url and paste it in jenkins
* Click on down arrow add credentials
* Enter branch \*/main
* Click on add timestamps to the console output under build environments
* Click on build steps
* Select invoke top level maven targets
* Select maven version
* Enter package name(clean package)
* Click on apply and save.
* Clcik on build now.(success)
* **Add maven plugins on jenkins**
* Click on manage jenkins
* Click on plugins
* Click on available plugins
* Search maven integration plugin
* Click on install without restart
* Go to new item
* Enter job name(mavenproject)
* Select maven project
* Click on ok
* Give description (about project)
* Go to source code management
* Select git
* Go to github select repository and copy URL and paste it in jenkins
* Click on down arrow add credentials
* Enter branch \*/main
* Enter clean install
* Click on apply and save
* Click on build now(success).