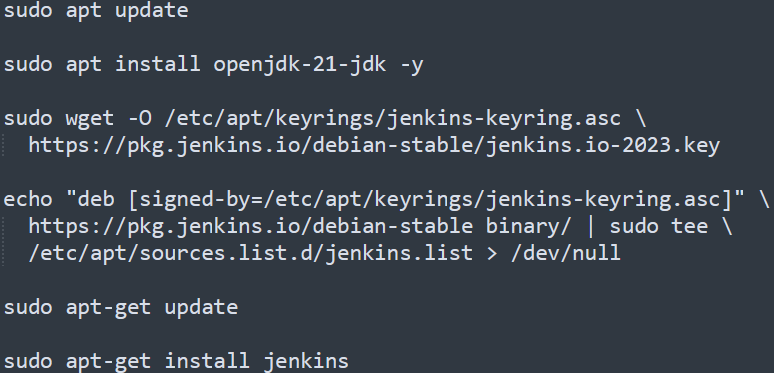
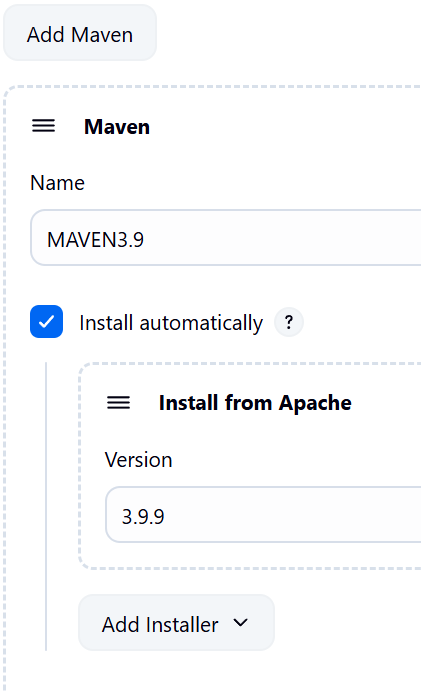
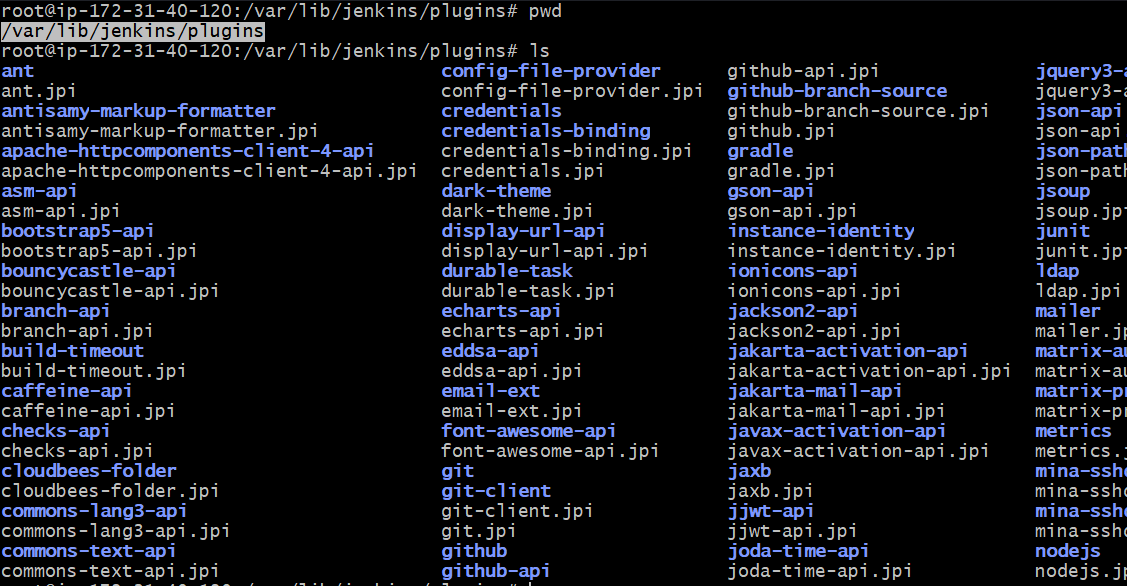
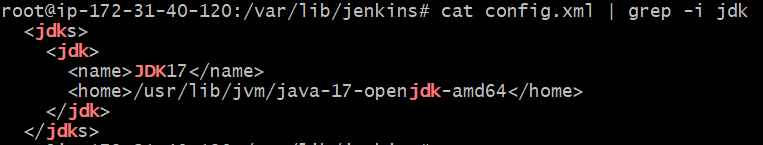
**CI/CD With Jenkins**

* To install Jenkins in Ubuntu:
  + You need to install Java because *Jenkins is written in Java*.
  + Its not a native program (like .exe or .bin), rather it’s a **.war** file (Java Web Application Archive).
  + To run it, you need JVM (Java Virtual Machine), which comes from JDE/JRE.
  + 
  + Inside **/var/lib/jenkins** the jenkins configuration (**config.xml**) file exists.
  + After installing jenkins, you can copy the *public IP* of the instance and open in the browser with port *8080* (remember: TCP with port 8080 should be present in the security group attached to the ubuntu instance).
    - After opening the browser, it’ll show one path where the initial password is present.
    - **/var/lib/jenkins/secrets/initialAdminPassword** : In this file the initial password is stored.
  + *If you can’t open the jenkins ui through browser, then try updating the security inbound rule for TCP 8080 traffic for all IPv4. sometimes, My IP doesn’t work.*
* **Jobs in Jenkins**
  + **Freestyle Job**
    - In freestyle, everything is configured in the Jenkins UI.
    - ***Graphical Jobs***.
    - Each job has a GUI form where you define:
      * Where to get code (GitHub, SVN, etc.)
      * Build steps (e.g., mvn clean install, npm build)
      * Post-build actions (e.g., deploy, send email)
    - **Pros:**
      * Easy to create (beginner friendly)
      * Great for simple projects
      * No need to learn syntax.
    - **Cons:**
      * Hard to maintain (if there are many jobs, have to edit each of them manually)
      * Not portable (configs only stay in Jenkins server, not git repo)
      * Limited flexibility (complex workflows are difficult to manage)
      * If jenkins crashes, you loose job definitions (unless backed up)
  + **Pipeline As A Code** 
    - Instead of configuring Jobs in UI, **Jenkinsfile** is used.
    - Jenkins read the file and runs the pipeline automatically.
    - Written in Groovy based DSL (Domain Specific Language)
* **Plugins vs Tools**
  + Simple analogy:
    - Keywords:
      * Programmer (Jenkins)
      * Programming Language (Plugin)
      * Tools (Laptop with compiler installed)
    - If a programmer knows the language (jenkins have plugins installed) but doesn’t have a laptop (the server where jenkins present, doesn’t have that tool): then it’ll be of no use
    - If a programmer doesn’t know the language (jenkins don’t have the plugin) and he is given a laptop (the server where jenkins is present, have the tools installed): then it’ll be of no use
  + ***Plugins*** *tell Jenkins how to do things;* ***Tools*** *let Jenkins actually do the work.*
  + You can install the tools in the server directly executing the command like **apt install maven** ..etc. OR you can do from the Jenkins GUI as well.
    - Note:
      * In GUI, it’ll display only those Tools, whose Plugins are installed.
      * If you don’t see the particular Tool you want, then install its Plugin first.
      * If you’ll install the Tools via system CLI directly; then also it’ll be of no use if the Plugin is not installed in Jenkins.
  + Ex: I am installing Maven (tool) via GUI
    - 
    - Its simple, just give a name and select the version.
  + Ex-2: I am installing JDK via GUI. Its little different
    - 
    - Its little different. Installed java-17 version in cli, then gave its home directory path in GUI.
  + The tools whose multiple versions can be installed at once in a system *(multiple versions of JDK can be installed in a system)*, we need to tell jenkins that which version is to be used by giving that version's home directory path.
  + The installed plugins stay in the directory: /var/lib/jenkins/plugins
    - 
  + All global tools configurations (JDK, Maven, Git, Node.js etc) are stored inside: /var/lib/jenkins/hudson.tasks.\*
    - Exception: JDKs are stored inside **/var/lib/jenkins/config.xml** because Jenkins treats them as a core runtime tool
    - If you have not updated the JDK in Jenkins UI, then you can’t see the JDK inside that **config.xml**. And Jenkins will use the default JDK that is present globally (in my case, global default was JDK version 21).
      * 
    - If multiples JDKs are configured inside this, then whatever version mentioned in the Job will be used while running the Job inside pipeline.