DevOps

NEXUS

What is Artifactory
What is Nexus
Nexus Setup
Integrating Nexus with Jenkins

SWIPE LEFT

An artifact in software development refers to a byproduct of the software development process. It can be any file, like binaries, executables, JAR/WAR files, scripts, or configuration files, that are produced during the development lifecycle.

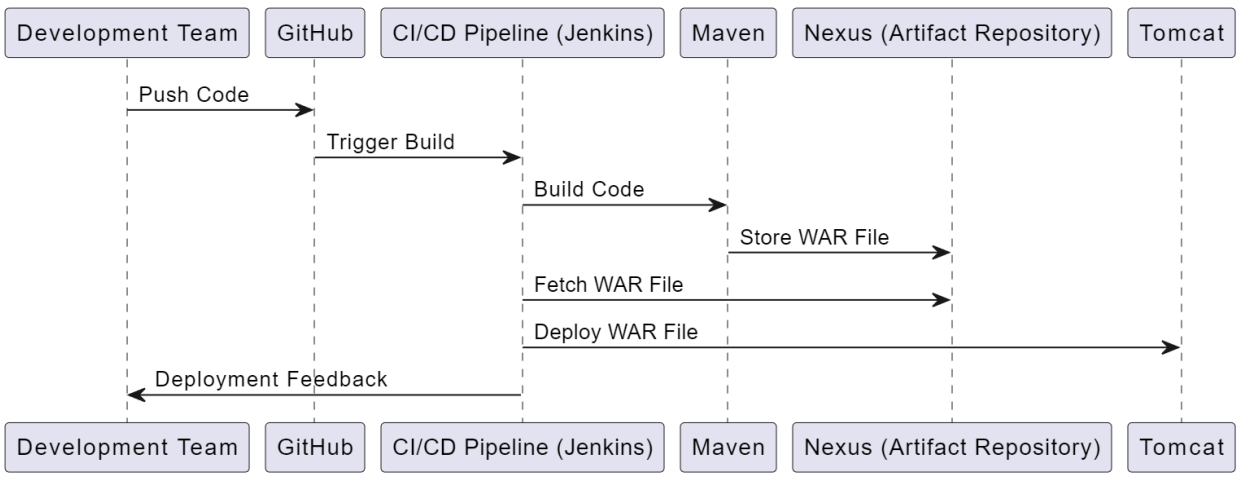
In the context of Maven (a build tool), an artifact typically refers to the files generated after a build process, such as:

- JAR (Java Archive)
- WAR (Web Archive)
- EAR (Enterprise Archive)

Importance of Artifacts in DevOps:

As a **DevOps engineer**, understanding and managing artifacts is crucial for the following reasons:

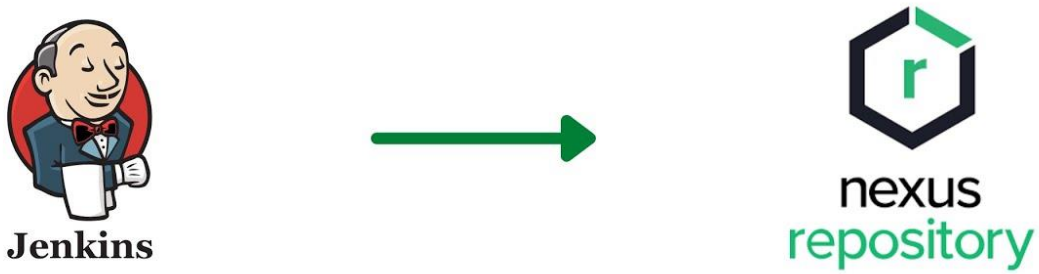
1. **Continuous Integration/Continuous Deployment (CI/CD):** Artifacts are the output of your build process and managing them effectively ensures they are correctly deployed across different environments.
2. **Versioning & Traceability:** Artifacts are versioned, and proper management ensures traceability.
3. **Storage & Distribution:** DevOps tools (e.g., Artifactory, Nexus) are used to store and distribute artifacts across environments or teams.



Nexus is a repository manager used in software development for managing and storing software artifacts. It helps developers and DevOps teams by serving as a central hub for storing, retrieving, and managing dependencies, build artifacts, and releases.

Why Nexus is Advantageous Over Other Artifact Repositories:

1. **Broad Format Support:** Nexus supports multiple artifact types (e.g., Maven, npm, Docker, etc.), whereas some alternatives (like Artifactory or Sonatype Nexus 2) might specialize more in certain formats.
2. **User-Friendly Interface**: Nexus has an intuitive web-based UI that makes managing repositories, artifacts, and permissions easy for users.
3. **Enterprise-Ready**: Nexus Repository Pro (the enterprise edition) provides robust features for enterprise needs, such as high availability, disaster recovery, and performance improvements.
4. **Integration with CI/CD Pipelines**: Nexus integrates seamlessly with CI/CD tools like Jenkins, GitLab, or Bamboo, making it ideal for automating the artifact management lifecycle, including building, storing, and deploying artifacts.
5. **Cost-Effectiveness**: Nexus has a free open-source version that includes many of the features needed by small to medium-sized teams. For larger enterprises, the paid version offers advanced capabilities at a competitive price compared to alternatives like **JFrog Artifactory**.



Upload Artifact to Nexus using Jenkins

FOSSTechNix.com

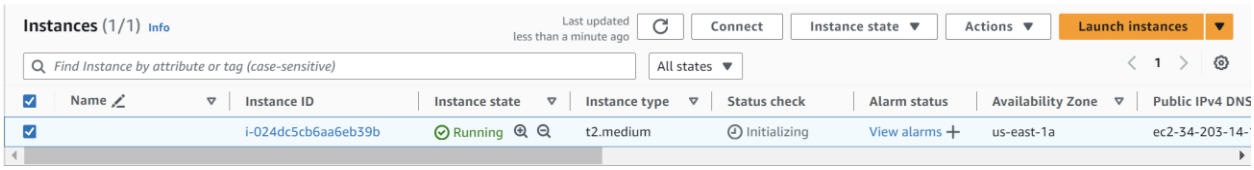
Nexus Setup and Integrated to Jenkins

Step 1: Launch an Instance with the following configurations:

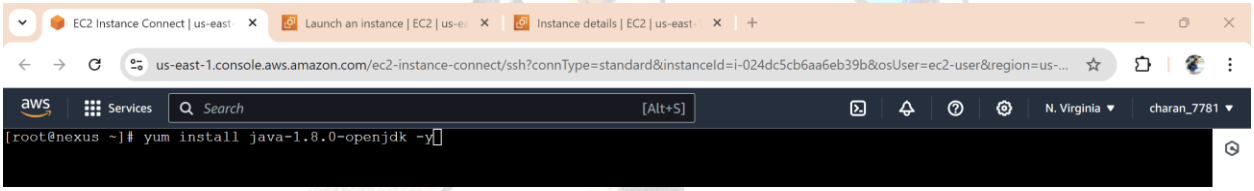
Storage: 25 GiB

Security Group: Allow port 9000

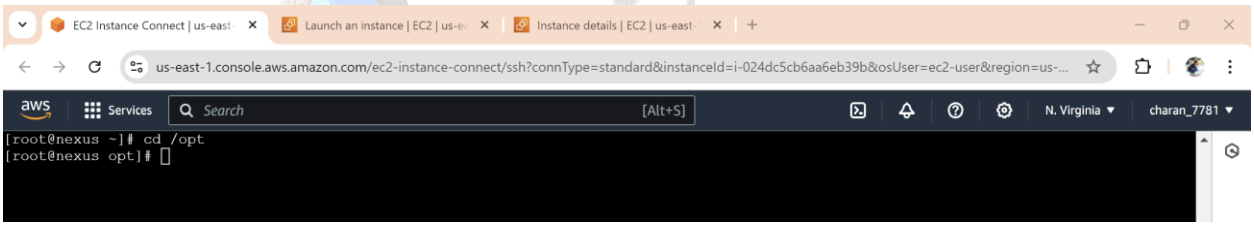
Instance Type: t2.medium



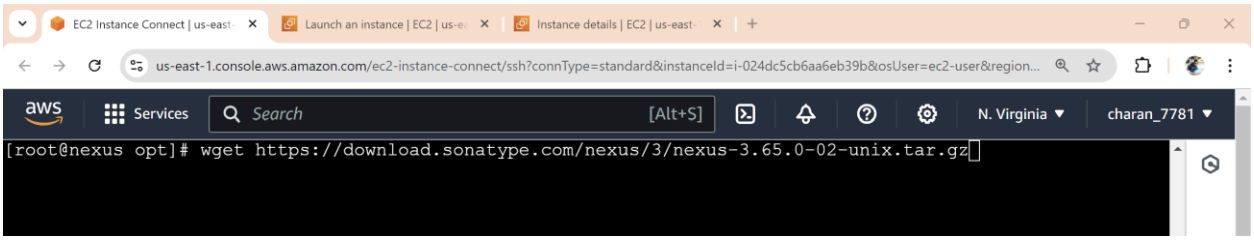
Step-2: Install java 1.8.0

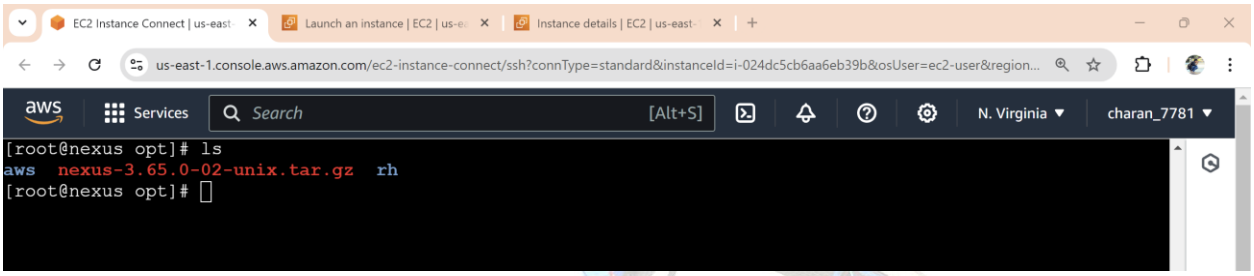


Step-3: Change directory to /opt

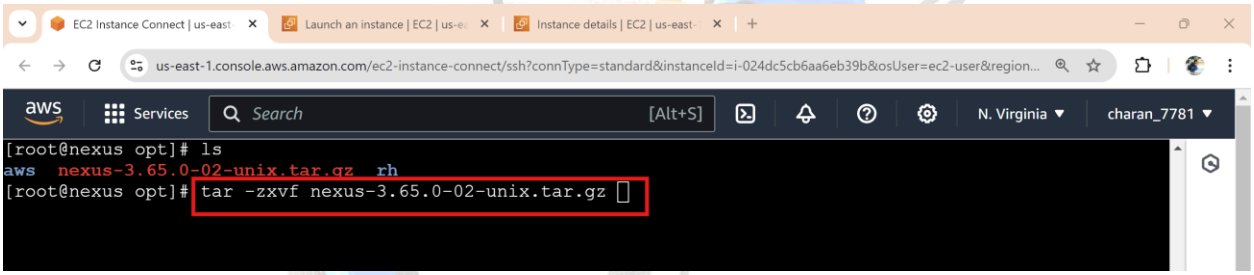


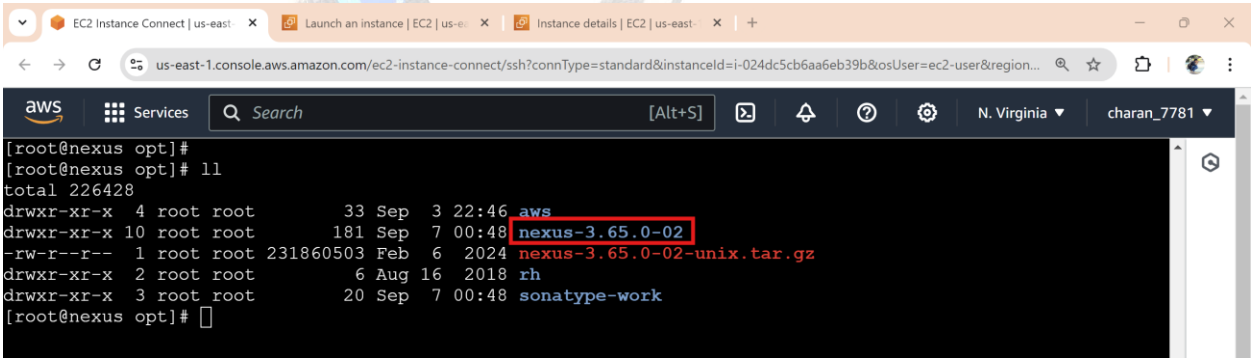
Step-4: Download the nexus from their official https://help.sonatype.com/en/download.html



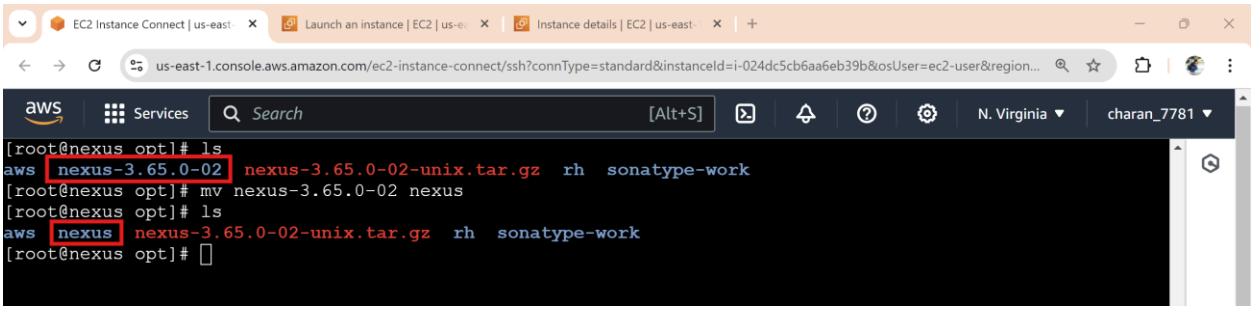


Step-5: Extract the tar file



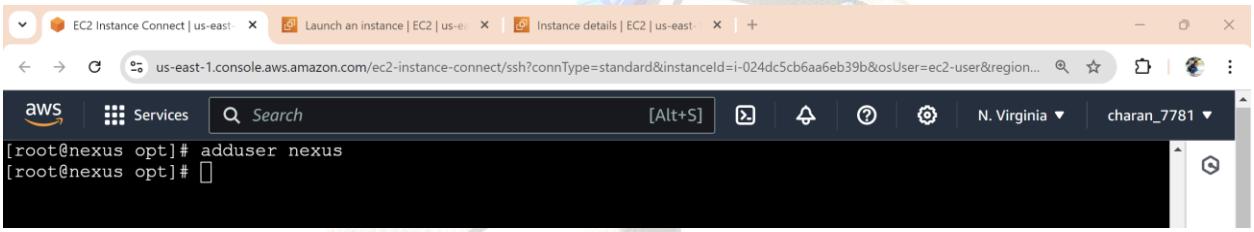


Step-6: Chang name of nexus directory

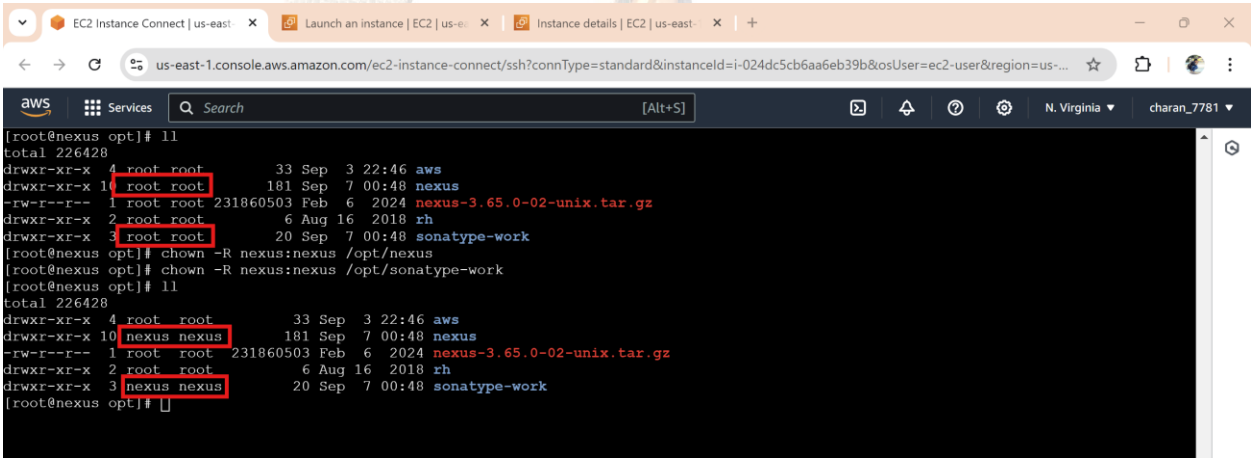


Step-7: Add nexus user

adduser nexus

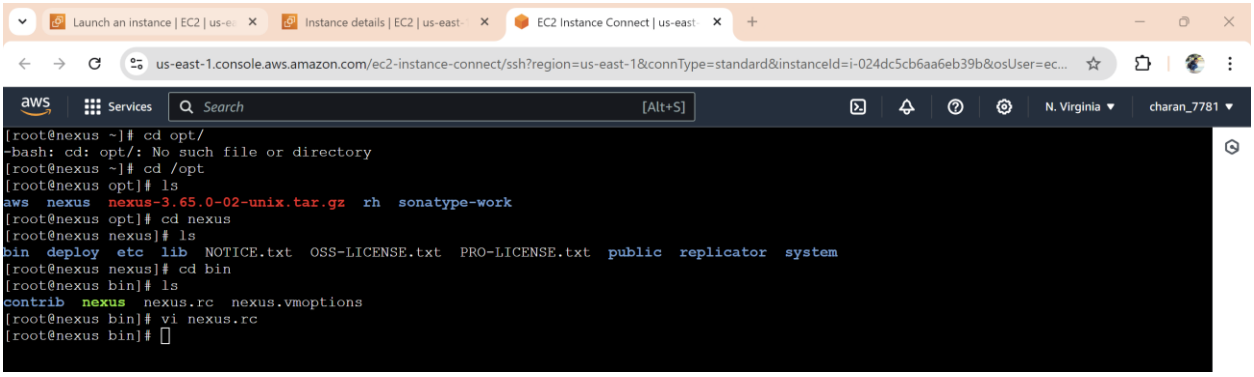


Step-8: Change the ownership of the directory's nexus and Sonatype.

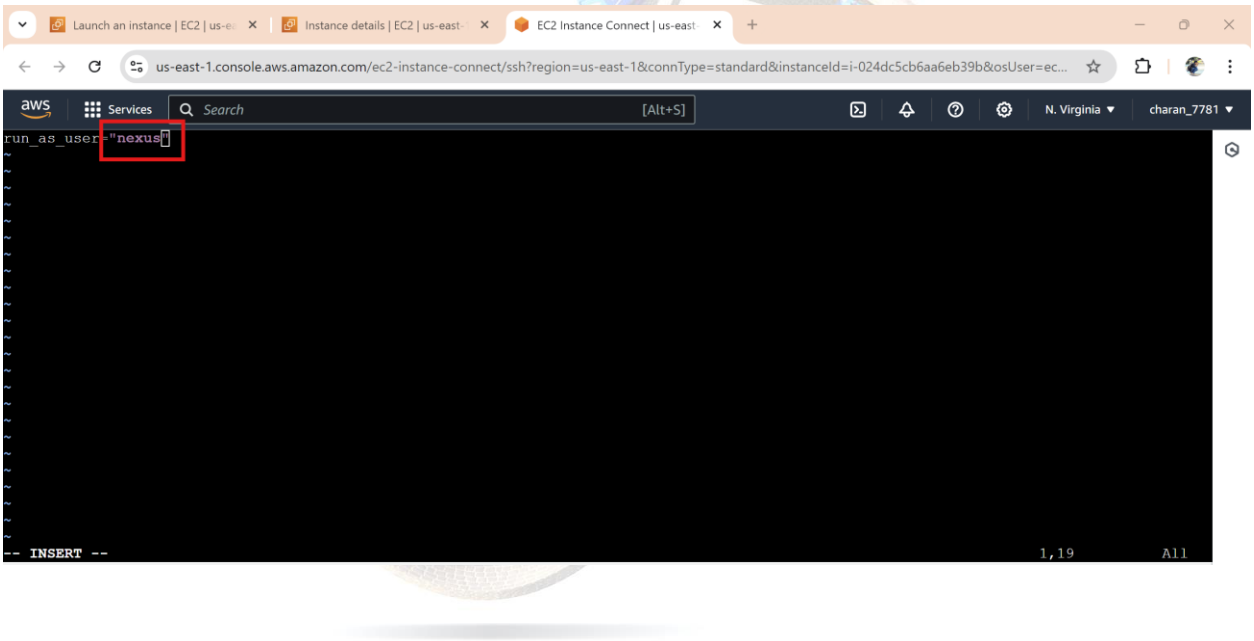


Step-9: Change Directory to /opt/nexus/bin/nexus.rc

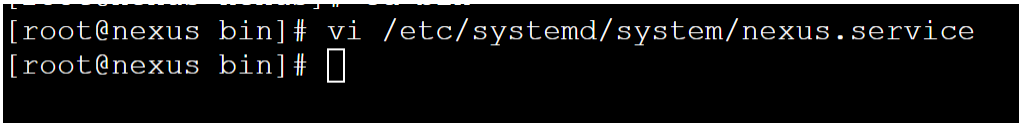
Remove **#** and add **nexus**



Add **nexus** as user



Step 10: Configure Nexus to run as a service.



Paste the below script:

[Unit]

Description=nexus service

After=network.target

[Service]

Type=forking

LimitNOFILE=65536

User=nexus

Group=nexus

ExecStart=/opt/nexus/bin/nexus start

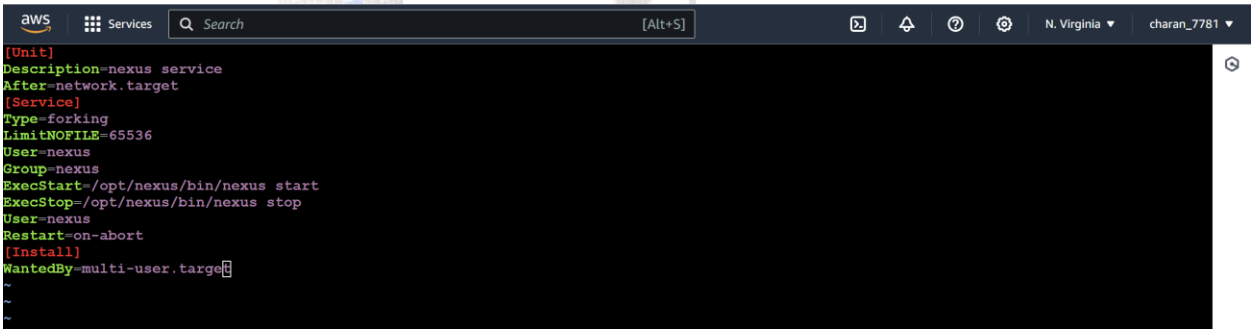
ExecStop=/opt/nexus/bin/nexus stop

User=nexus

Restart=on-abort

[Install]

WantedBy=multi-user.target

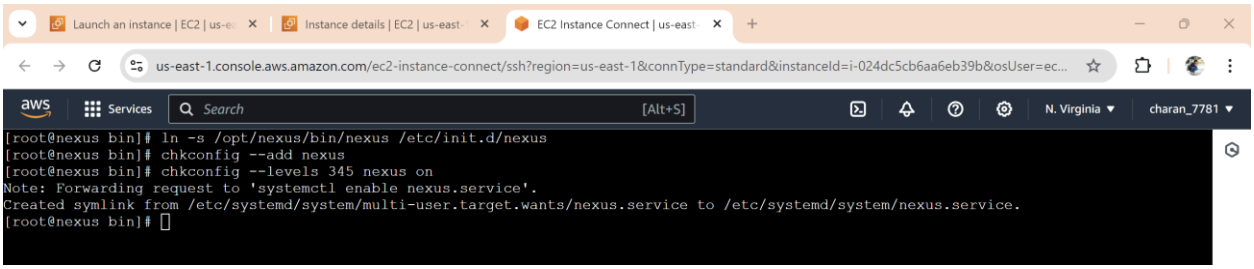


Step-11: ln -s /opt/nexus/bin/nexus /etc/init.d/nexus

Step-12: Add service to boot

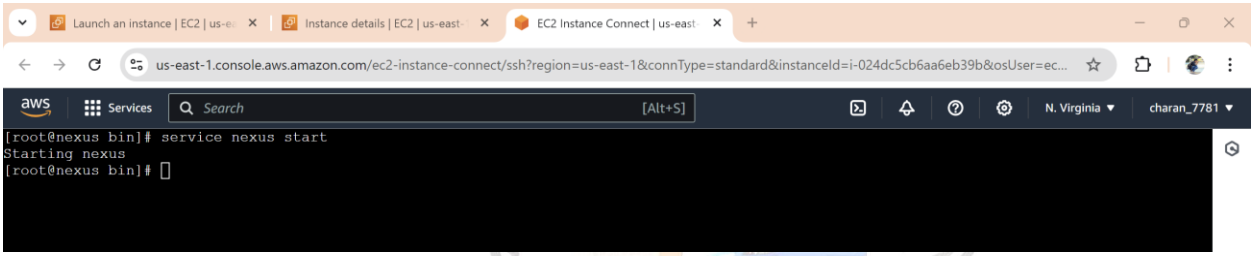
chkconfig --add nexus

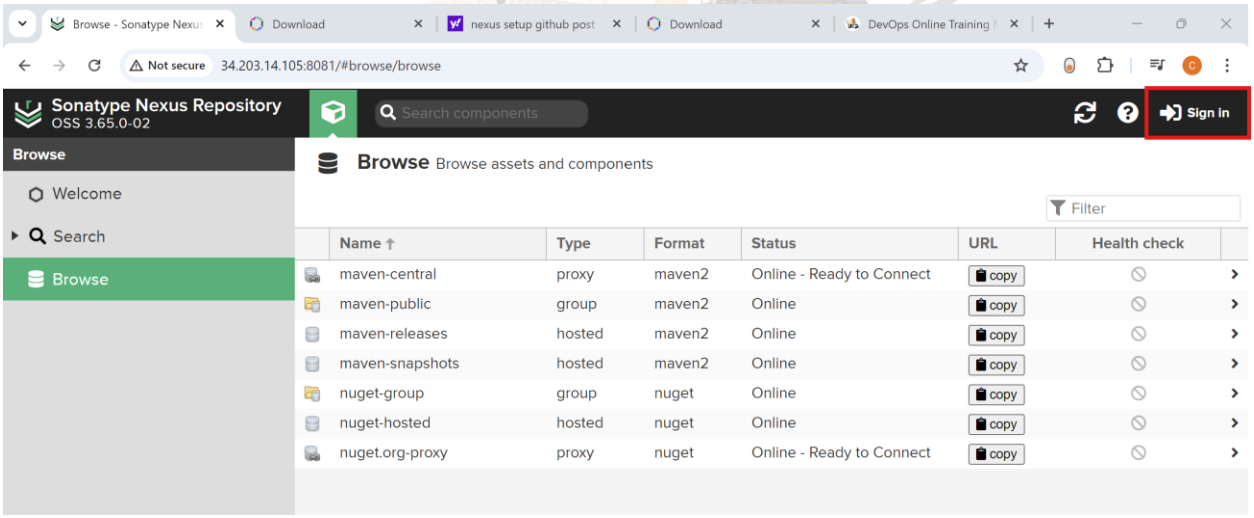
chkconfig --levels 345 nexus on



Step-13: Start service

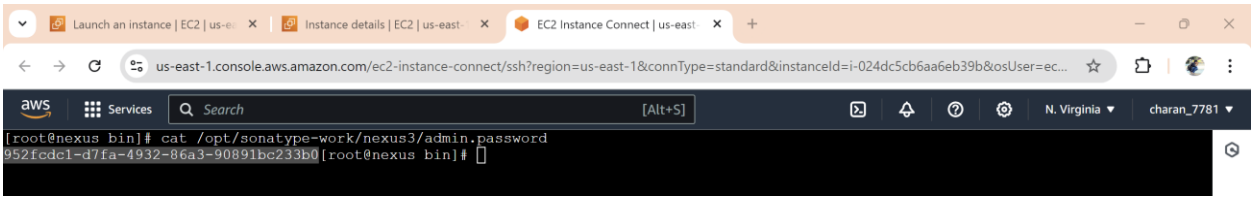
service nexus start



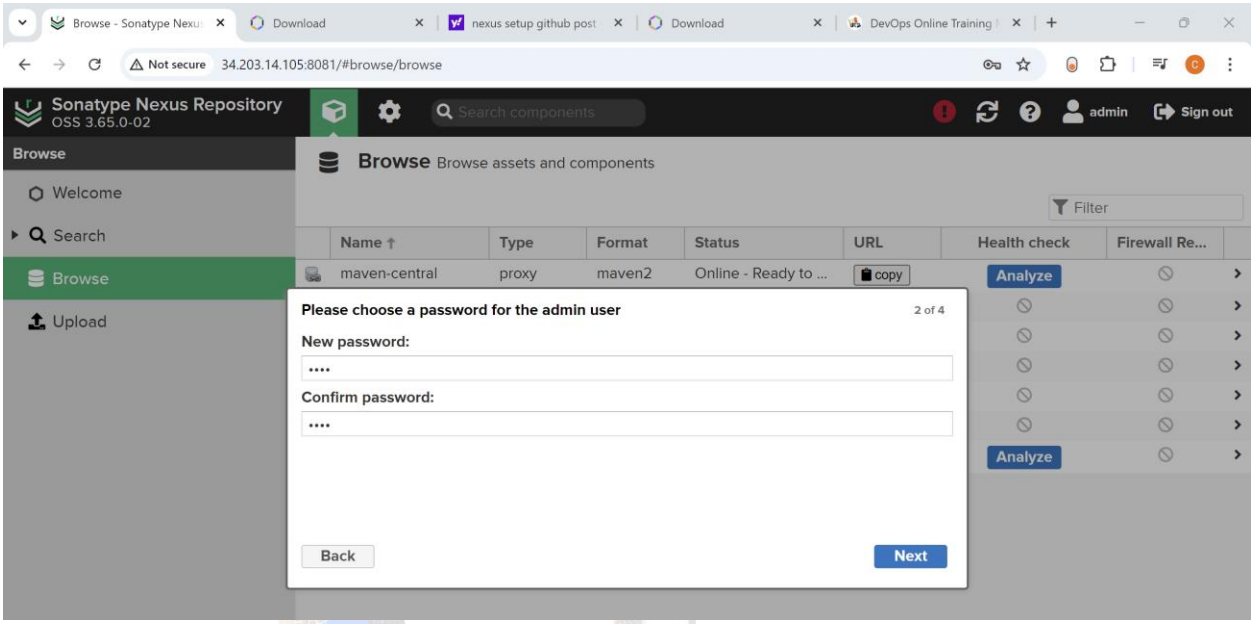


Step-14: Sign into nexus using credentials

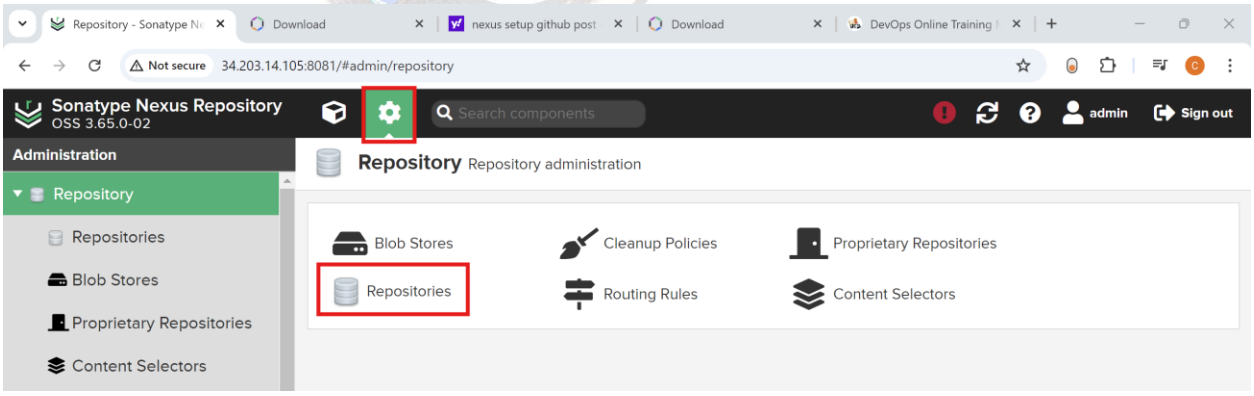
Username as **admin** and password you'll find in path (**/opt/sonatype-work/nexus3/admin.password)**



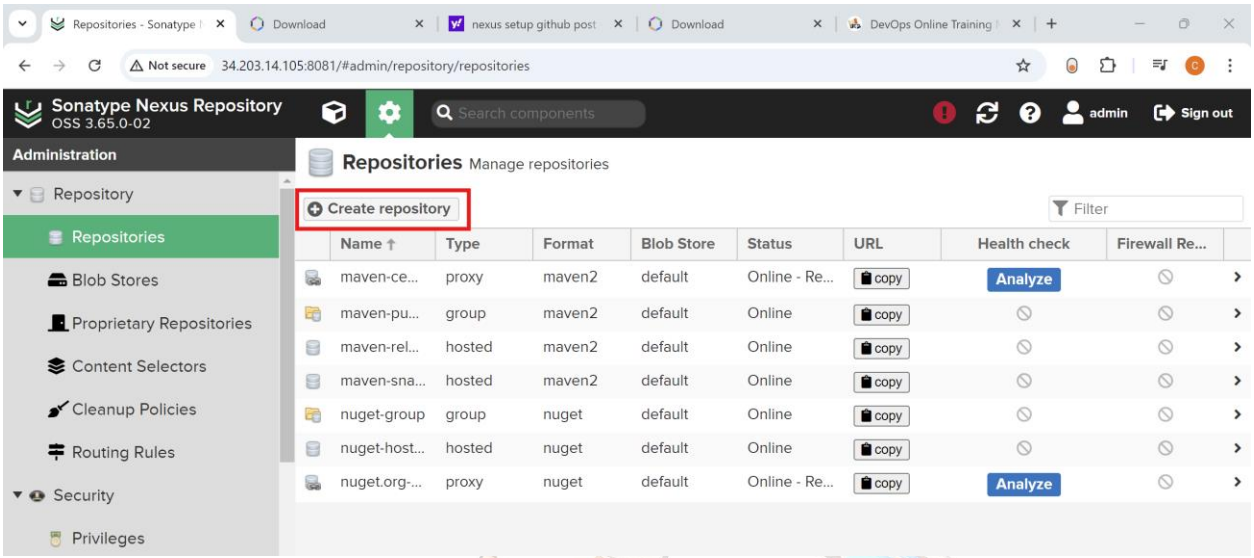
Step-15: Create a new password

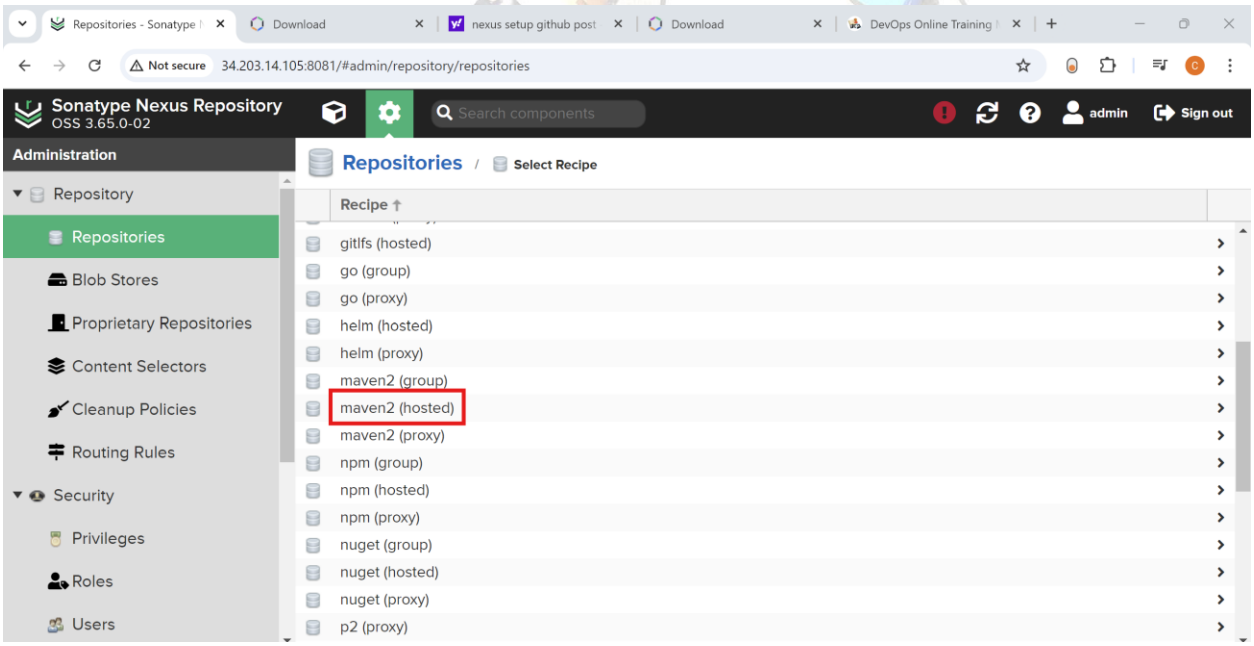


Step-16: Go to setting and create a repository.

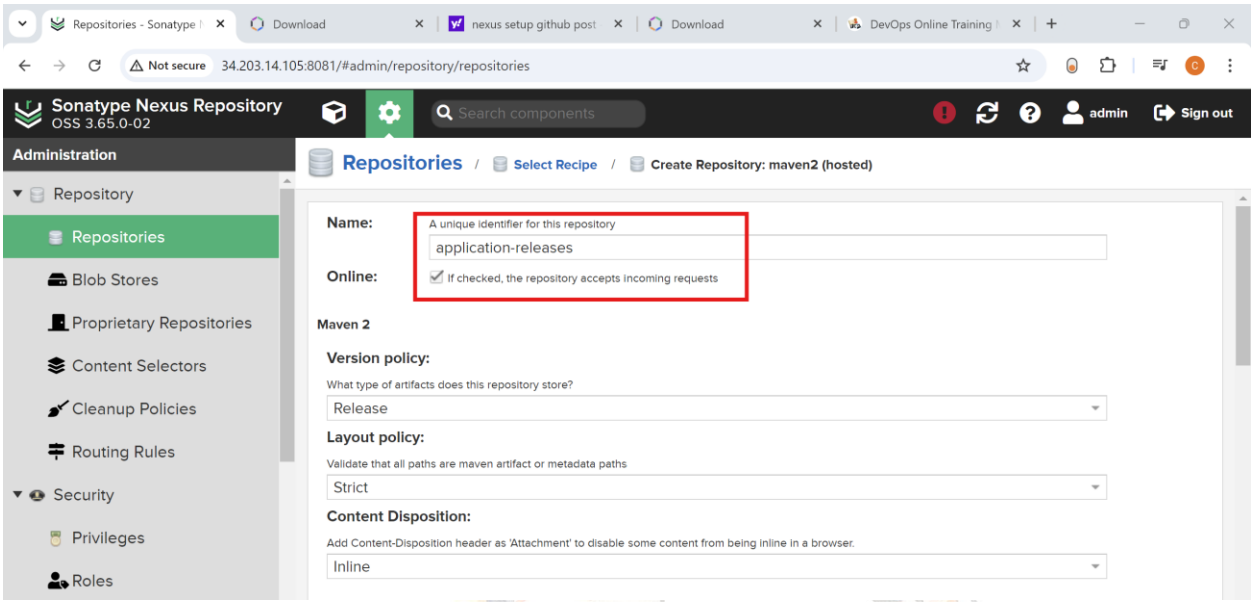


Step-17: Create a repo

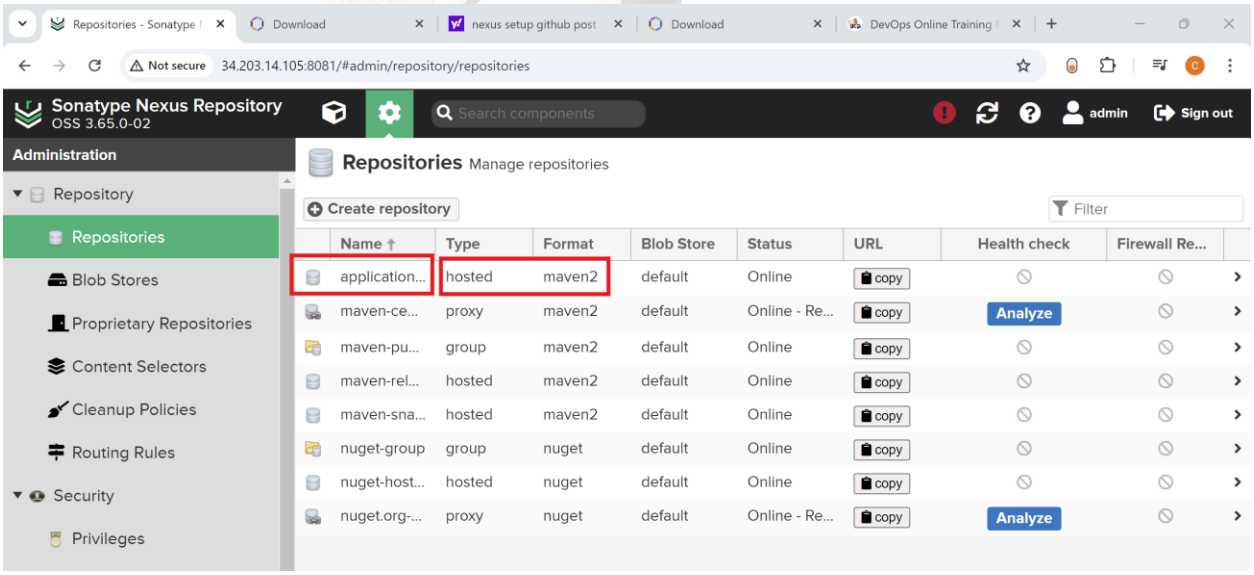




Step-18: Create a repository name



Under repositories you'll find **application-releases**



Step-19: Integrating Nexus with Jenkins.

Install **Nexus antifactory uploader**

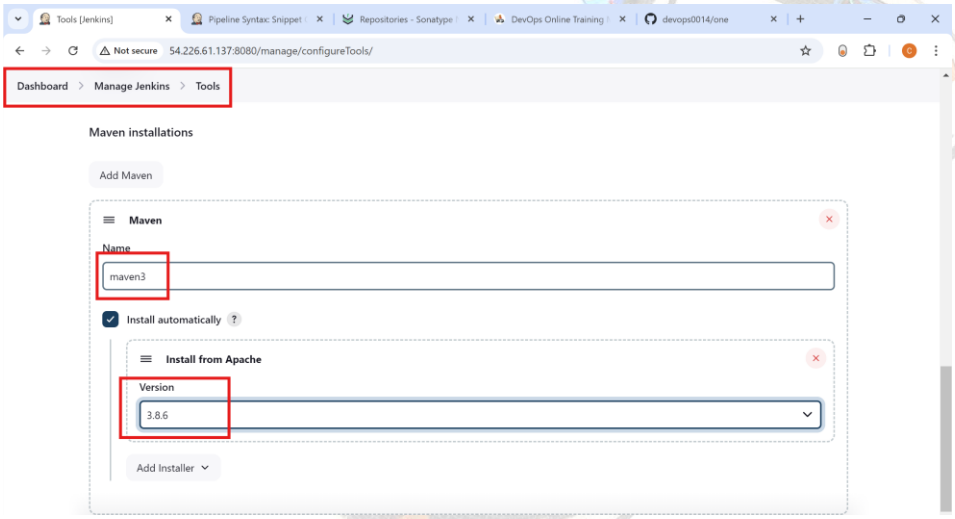
Step 20: To get the code in the artifact repository, you need to fetch the code from GitHub, build it into a packaged format, and then store it in the artifact repository (such as Nexus).

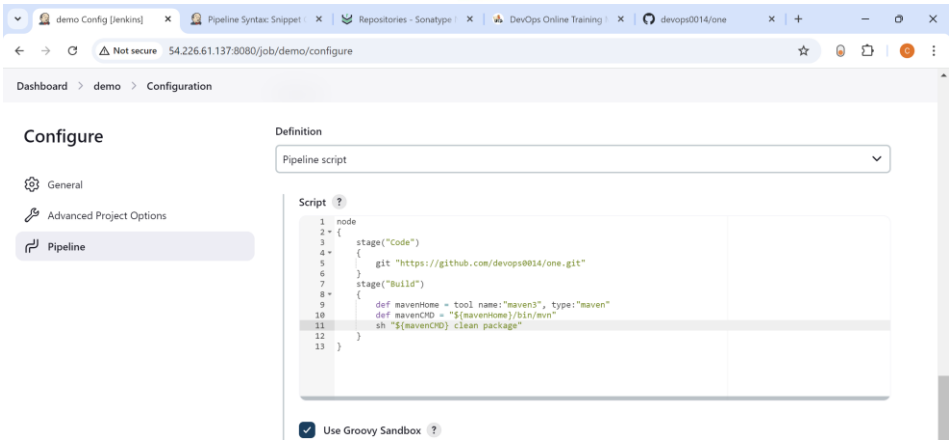
Write a pipeline for this:

You should git and maven on Jenkins server, instead you can simply go to

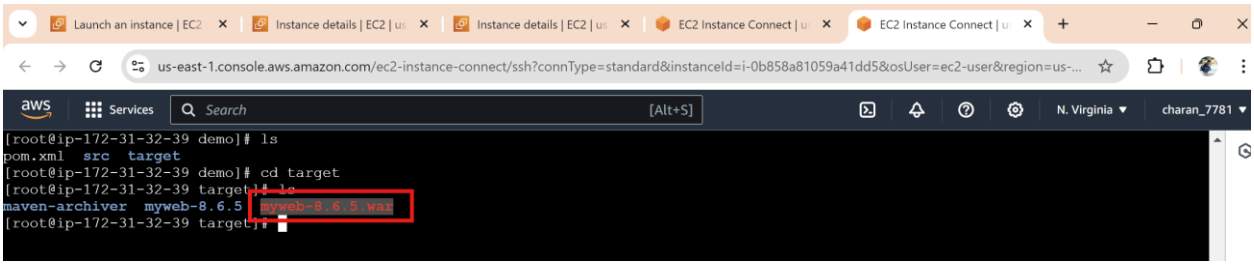
Manage Jenkins > global tool configuration>> maven installation

Installing Maven on a Jenkins server can result in two different Java versions: one for Jenkins and another for Maven. If not managed properly, these versions could override each other, leading to potential conflicts during builds. To avoid issues, ensure both tools use the same Java version or configure Maven to use the correct one explicitly.





Once build is finished, you'll get war file.

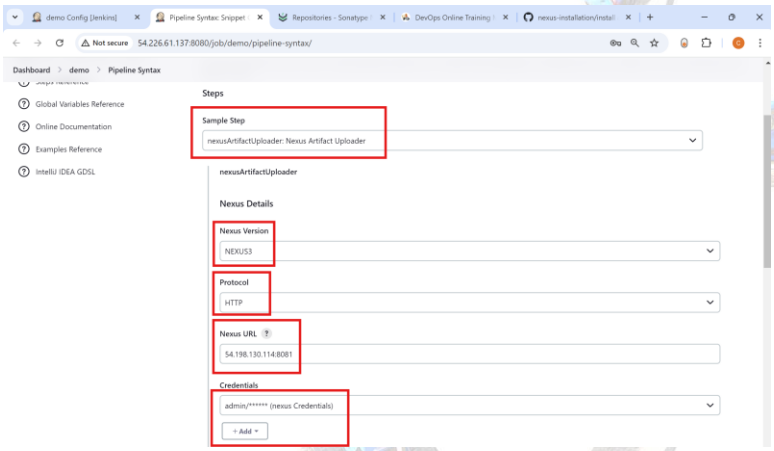


Here you should generate syntax for artifact stage

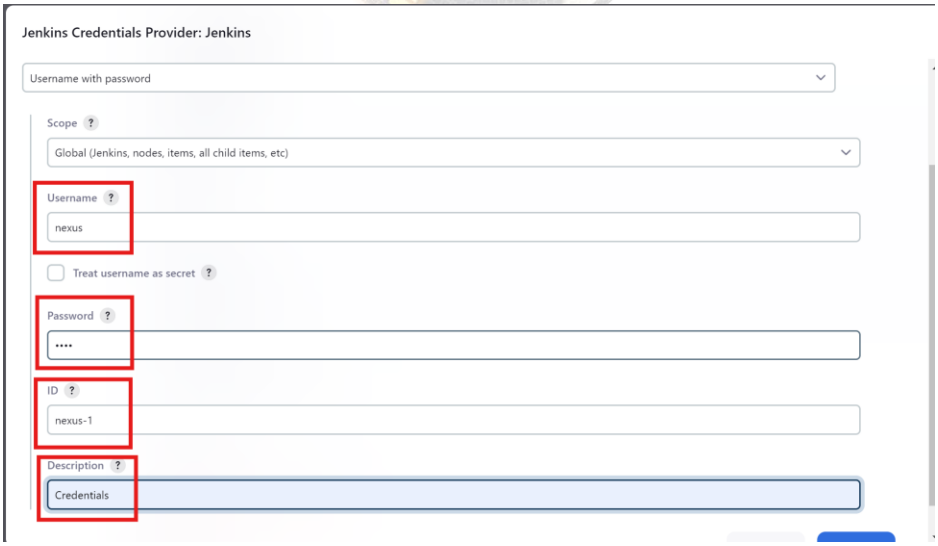
Select **nexusArtifacctUploader: Nexus Artifact Uploader**

Enter Nexus version as **Nexus3**

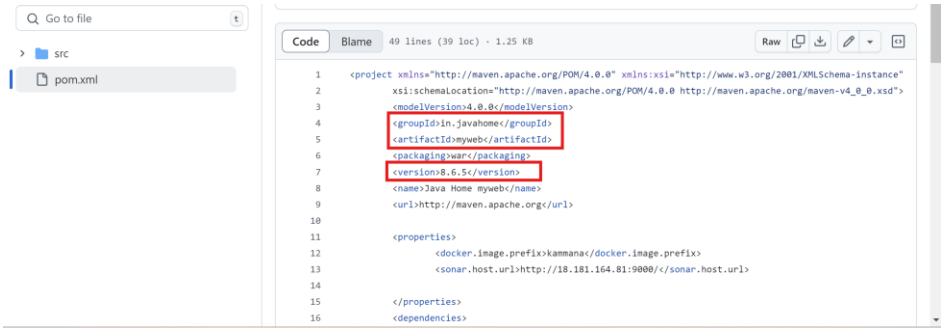
Protocol as **HTTP** and give **Nexus URL.**

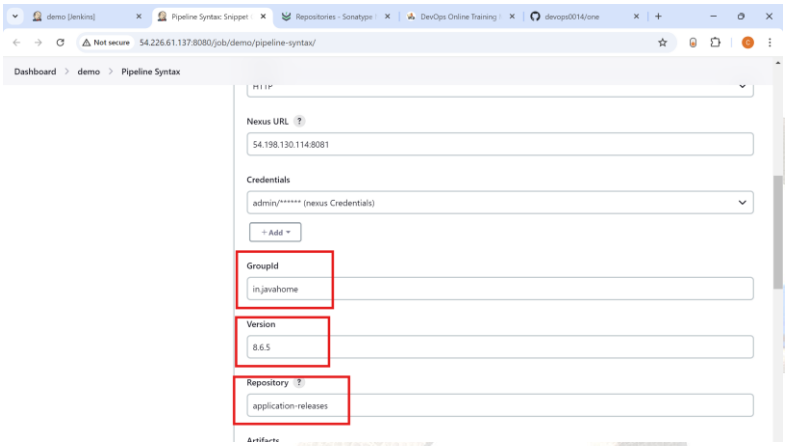


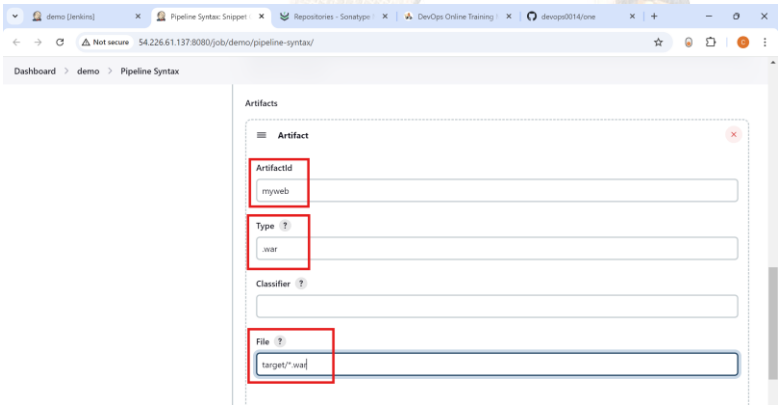
In the credentials section add username (nexus), nexus_password and give id and description.



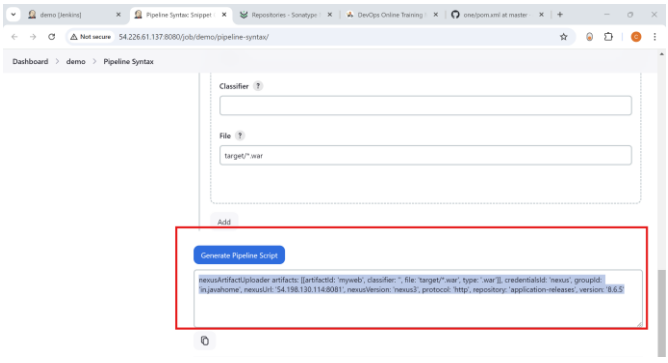
From.xml file enter the details



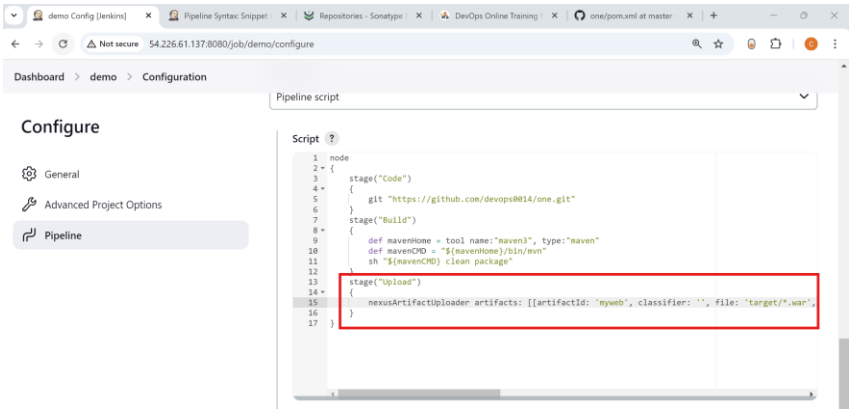




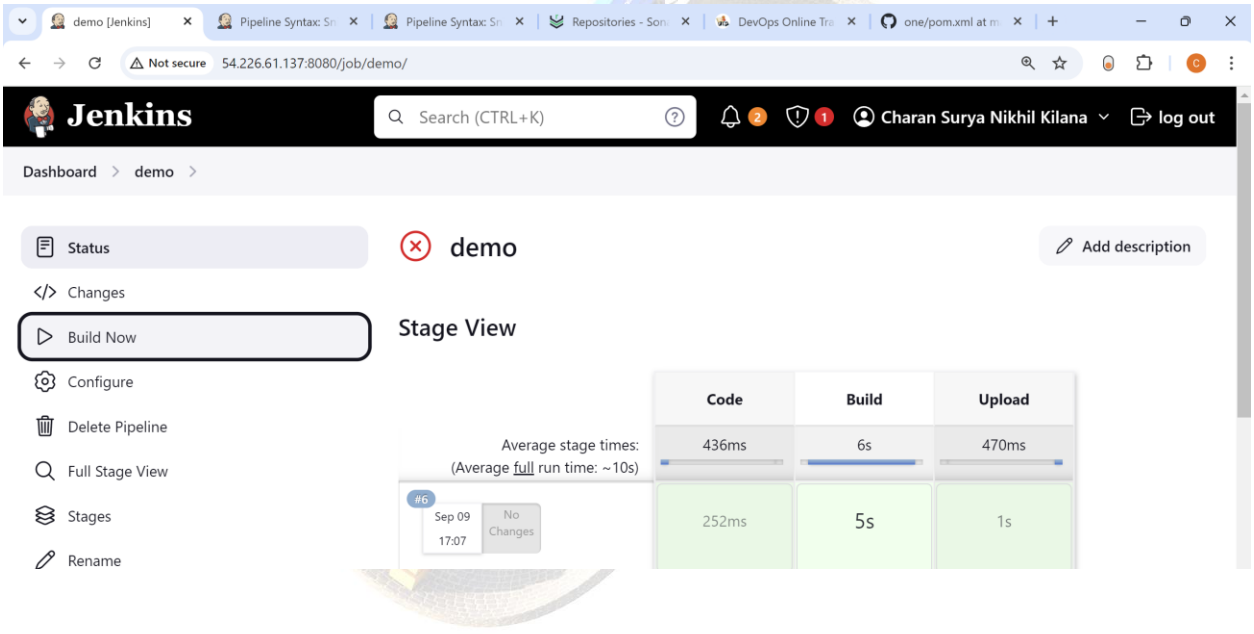
Then generate syntax



Step-20: Enter the syntax in artifact block



After build:



Once Build is done, you'll get the war file in Artifact.