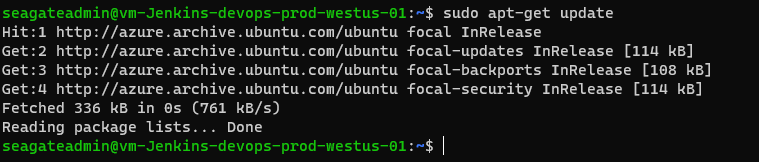
**PRE-REQUISITES INSTALLATION:**

1.**DOCKER-INSTALLATION**:

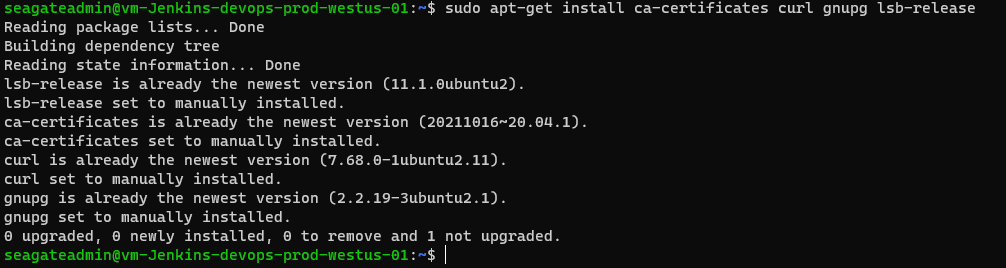
-----------------------------

steps:

sudo apt-get update



sudo apt-get install ca-certificates curl gnupg lsb-release



sudo mkdir -p /etc/apt/keyrings



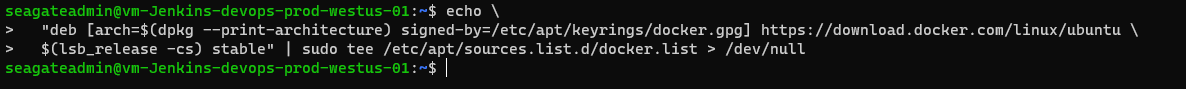
curl -fsSL <https://download.docker.com/linux/ubuntu/gpg> | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg



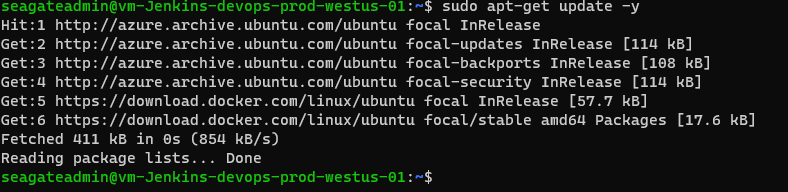
echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] <https://download.docker.com/linux/ubuntu> \

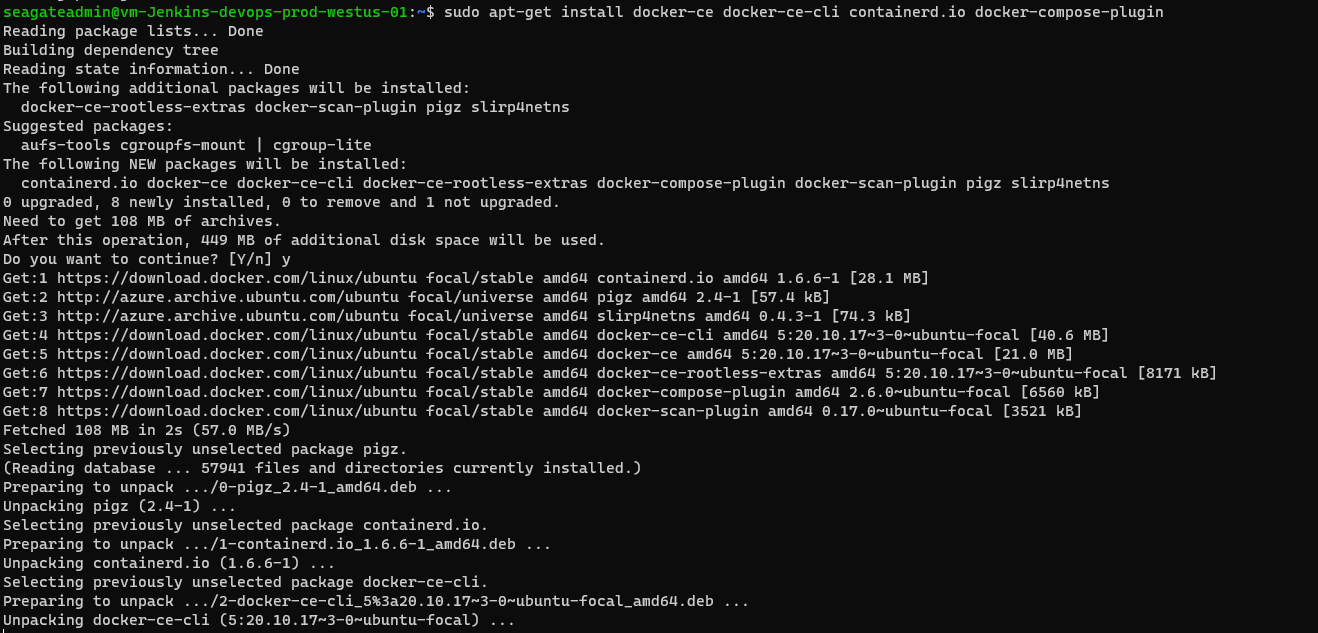
$(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null



sudo apt-get update



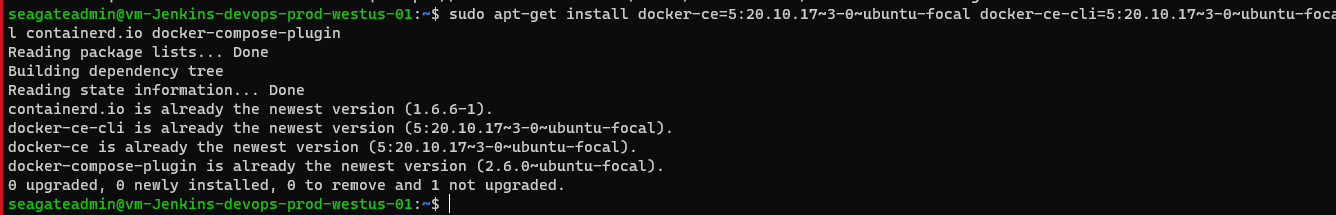
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin

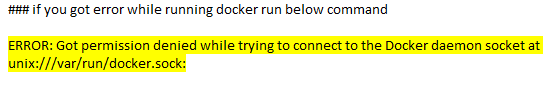


sudo chmod a+r /etc/apt/keyrings/docker.gpg



sudo apt-get install docker-ce=5:20.10.17~3-0~ubuntu-focal docker-ce-cli=5:20.10.17~3-0~ubuntu-focal containerd.io docker-compose-plugin





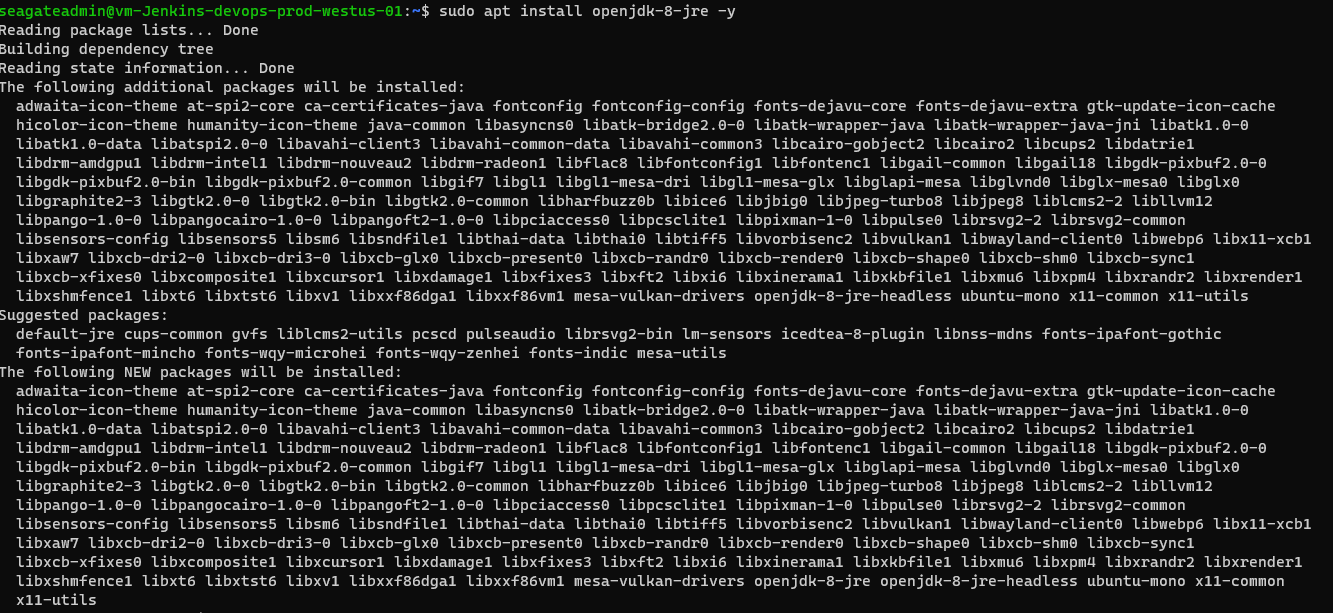
sudo chmod 777 /var/run/docker.sock



2. **JAVA INSTALLATION**:

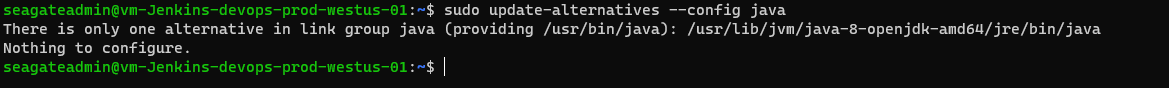
--------------------------

sudo apt install openjdk-8-jre -y



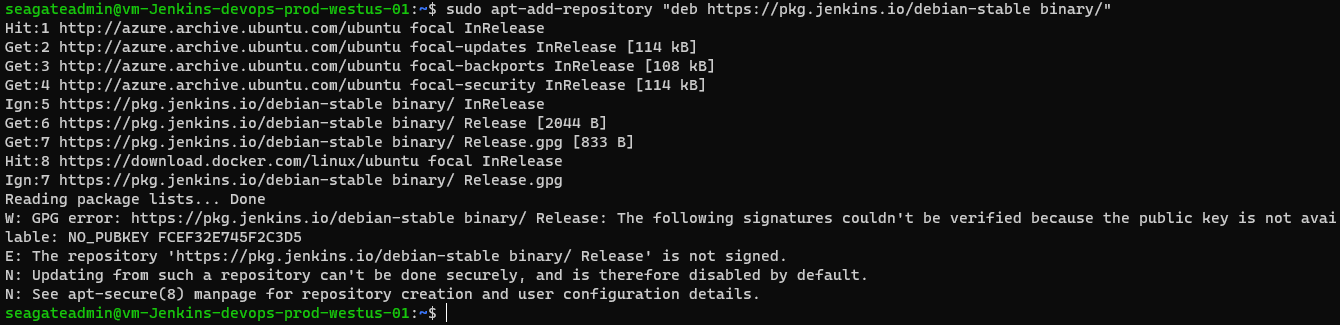
# Set default java version as java 8

sudo update-alternatives --config java



# Add repository

sudo apt-add-repository "deb [https://pkg.jenkins.io/debian-stable binary/](https://pkg.jenkins.io/debian-stable%20binary/)"



**3. JENKINS INSTALLATION:**

-----------------------------

* curl -fsSL <https://pkg.jenkins.io/debian-stable/jenkins.io.key> | sudo tee \

/usr/share/keyrings/jenkins-keyring.asc > /dev/null

* echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \

[https://pkg.jenkins.io/debian-stable binary/](https://pkg.jenkins.io/debian-stable) | sudo tee \ /etc/apt/sources.list.d/jenkins.list > /dev/null

* sudo apt-get update
* sudo apt-get install jenkins

After Jenkins Installation we got Crumb Error.

For resolving this error, we used following below command

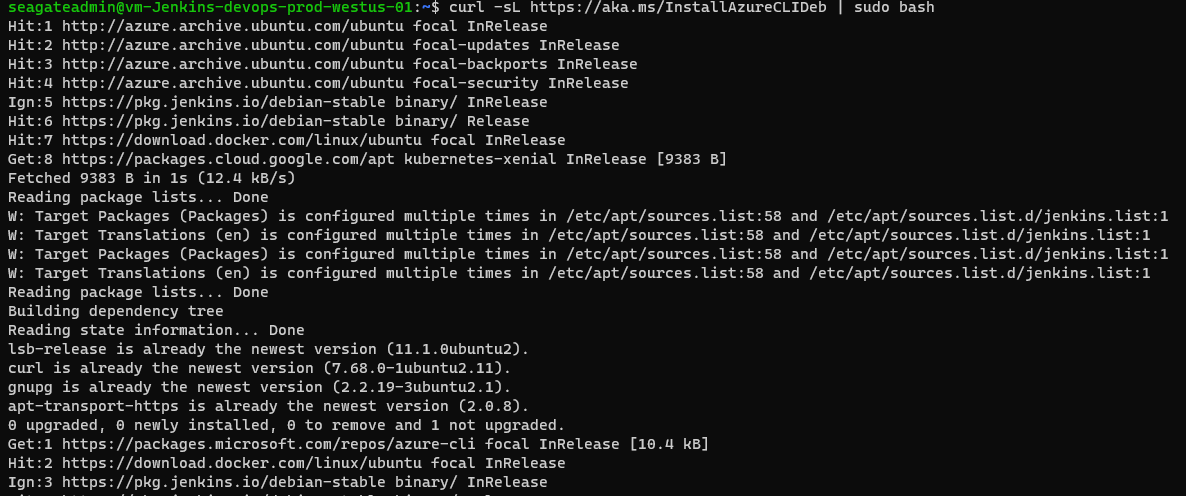
1. Create a file in this path :- /var/lib/jenkins/
2. File name :- startup-properties.groovy
3. Add the given content inside the file as

import jenkins.model.Jenkins  
def instance = Jenkins.instance  
instance.setCrumbIssuer(null)

4. **INSTALL AZURE CLI:**

**------------------------**

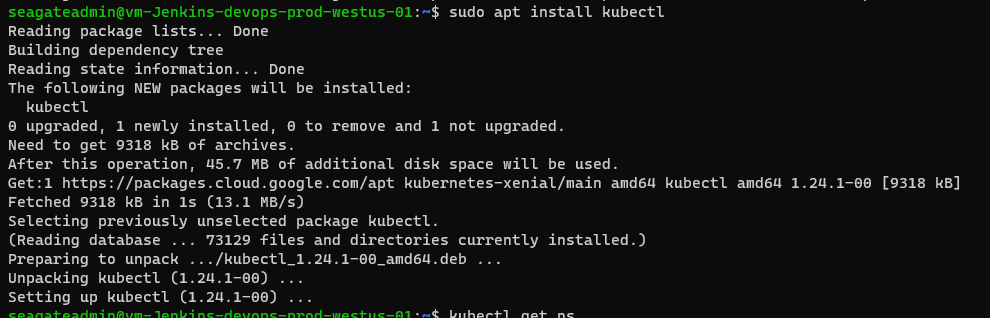
curl -sL <https://aka.ms/InstallAzureCLIDeb> | sudo bash



5**. Install Kubectl:**

**------------------**

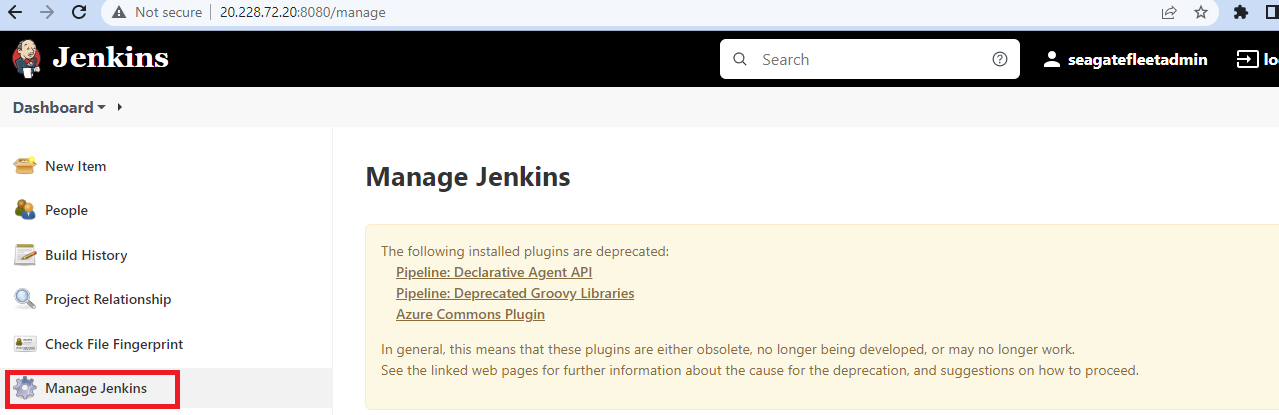
sudo apt install kubectl

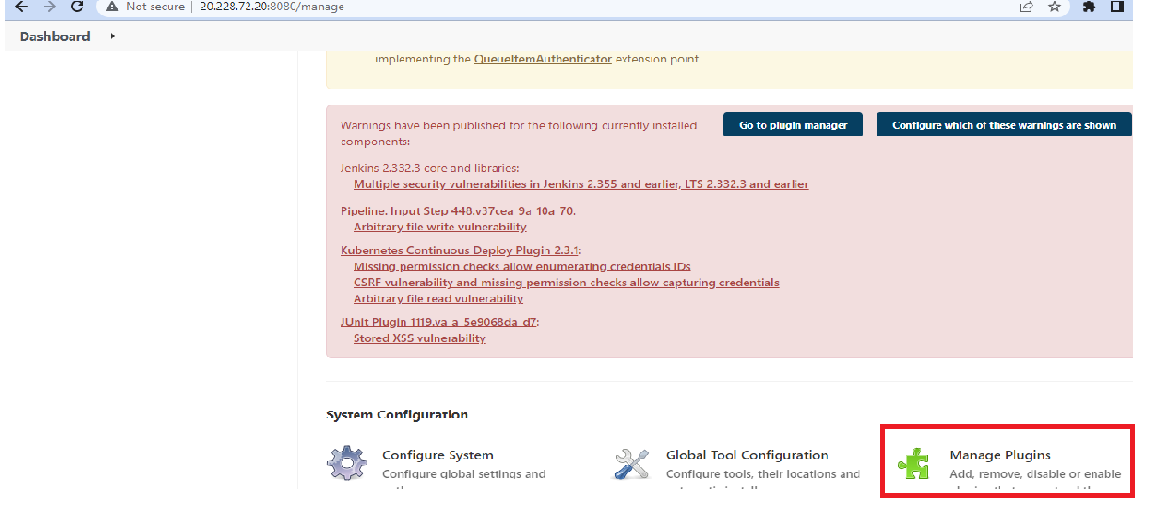


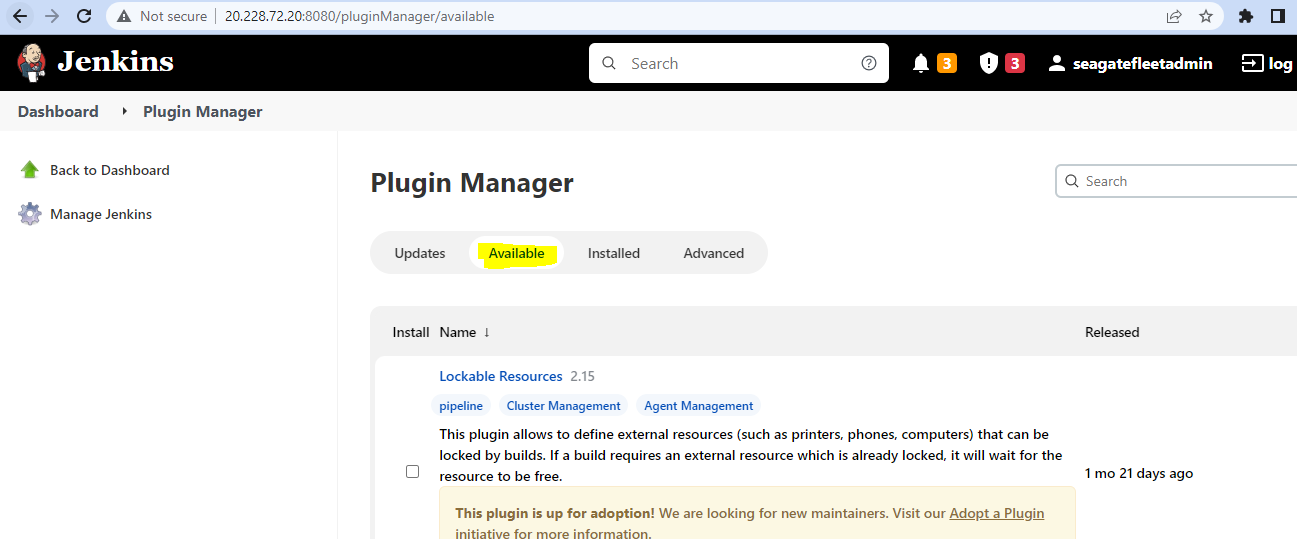
6. **PLUGIN INSTALLATION:**

----------------------------

Go to manage jenkins then manage plugin, here you need to search available plugin in available tab and install required plugin:







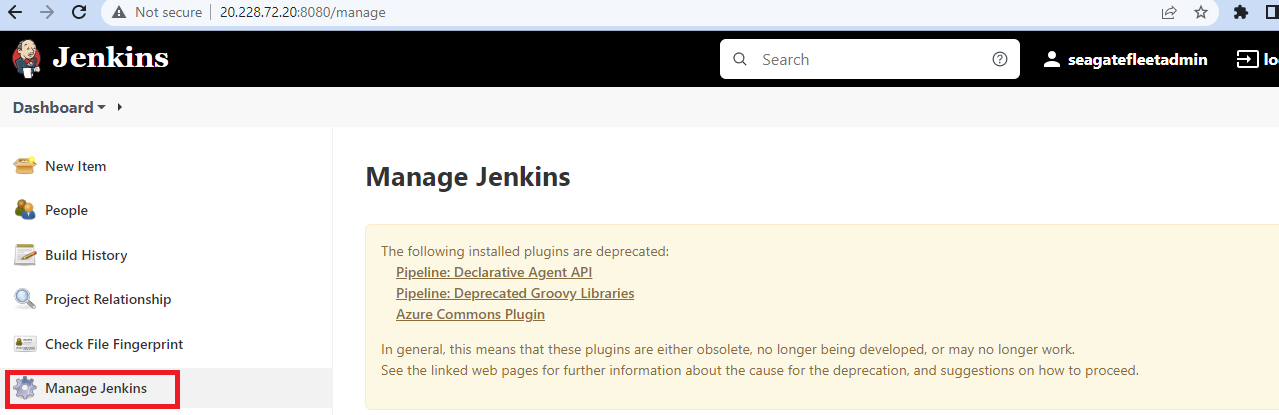
Below is the list of installed plugin:

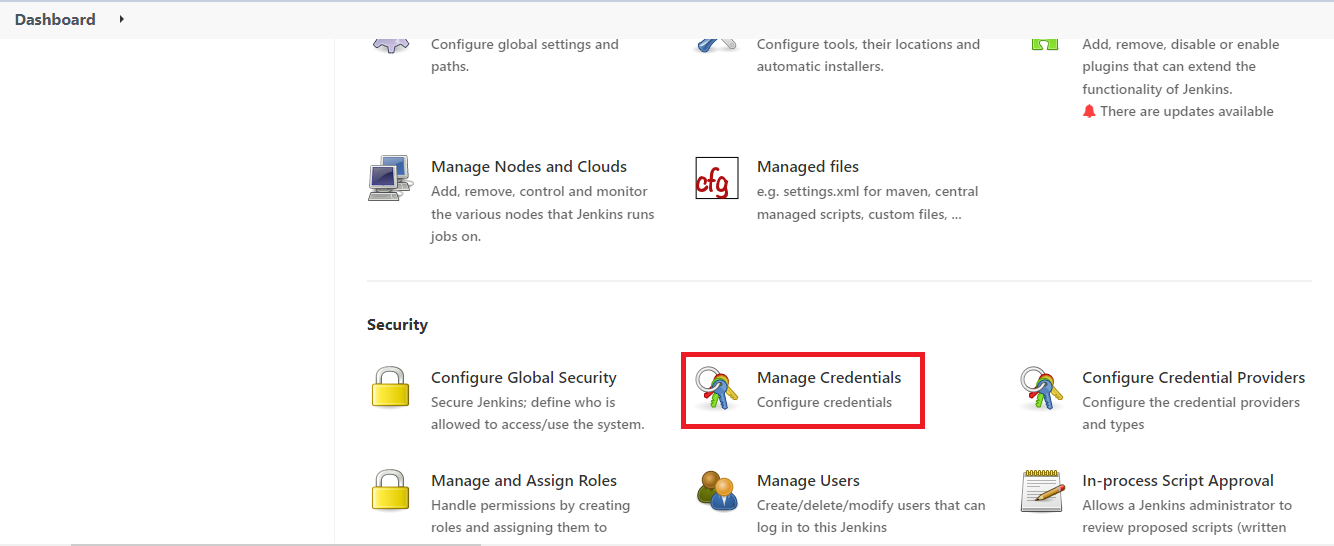
* **Azure Container Service Plugin** :- Manage Azure credentials using Jenkins Credentials API
* **Azure Storage plugin** :-Azure Blob Storage Plugin enables uploading build artifacts and downloading blob files from Azure Blob storage.
* **Docker Pipeline** :-Build and use Docker containers from pipelines.
* **Docker plugin** :-This plugin integrates Jenkins with Docker
* **Email Extension** :-This plugin is a replacement for Jenkins's email publisher. It allows to configure every aspect of email notifications: when an email is sent, who should receive it and what the email says
* **Git Parameter Plug-In :**- Adds ability to choose branches, tags or revisions from git repositories configured in project.
* **GitHub Authentication plugin** :-Authentication plugin using GitHub OAuth to provide authentication and authorization capabilities for GitHub and GitHub Enterprise.
* **GitHub Integration Plugin** :-GitHub Integration Plugin for Jenkins
* **Green Balls** :-Because green is better than blue! For color blind support configure user property.
* **Kubernetes plugin**:- This plugin integrates Jenkins with Kubernetes
* **NPM and Yarn Wrapper and Steps** :-A convenience plugin for using npm and yarn in Jenkins pipelines and freestyle builds
* **Pipeline NPM Integration Plugin**:-This plugin provides integration with Pipeline, configures npm environment to use within a pipeline job by calling sh npm or bat npm. The select npmrc will be configured and placed in the workspace prior to the build in order to take precedence over any user or global npmrc.
* **Pipeline: Declarative Agent API**:-Replaced by Pipeline: Declarative Extension Points API plugin.
* **Pipeline: GitHub Groovy Libraries**:- Allows Pipeline Groovy libraries to be loaded on the fly from GitHub.
* **Role-based Authorization Strategy**:-Enables user authorization using a Role-Based strategy. Roles can be defined globally or for particular jobs or nodes selected by regular expressions.
* **SonarQube Scanner for Jenkins**:-This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.
* **Workspace Cleanup Plugin**:-This plugin deletes the project workspace when invoked.

7. **CREATING SECRETS:**

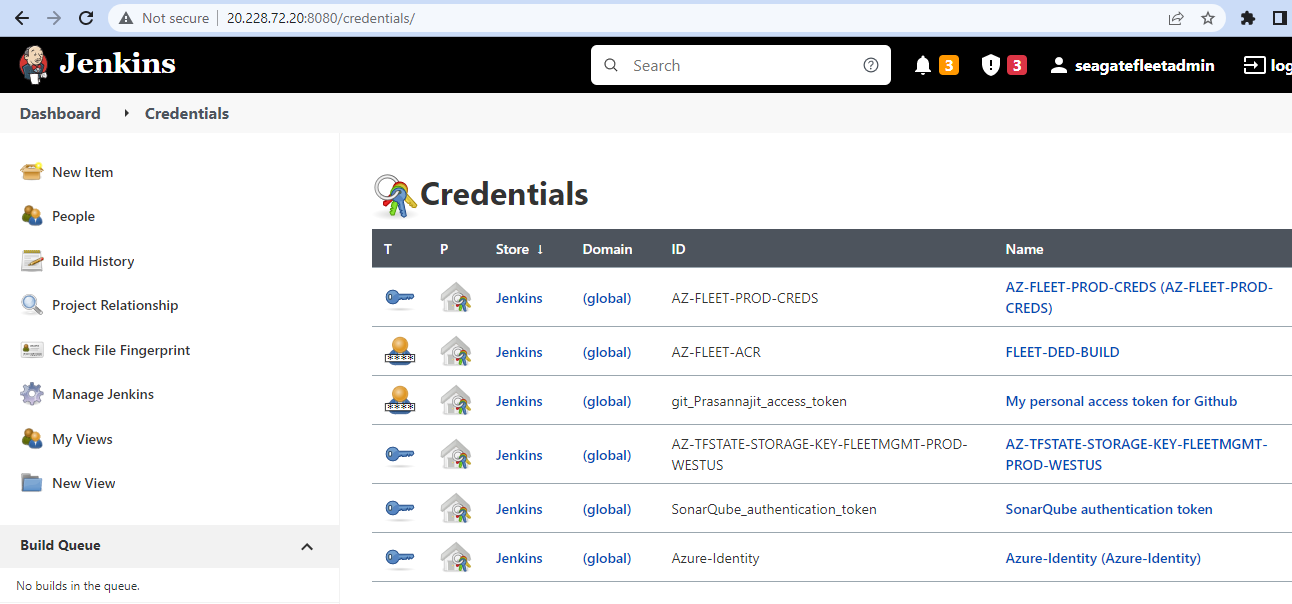
**-----------------------**

**G**o to Mangage Jenkins-->manage Credentials



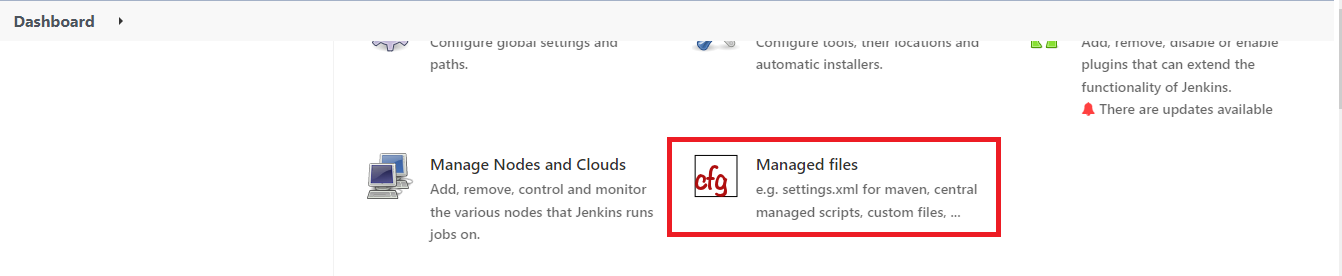


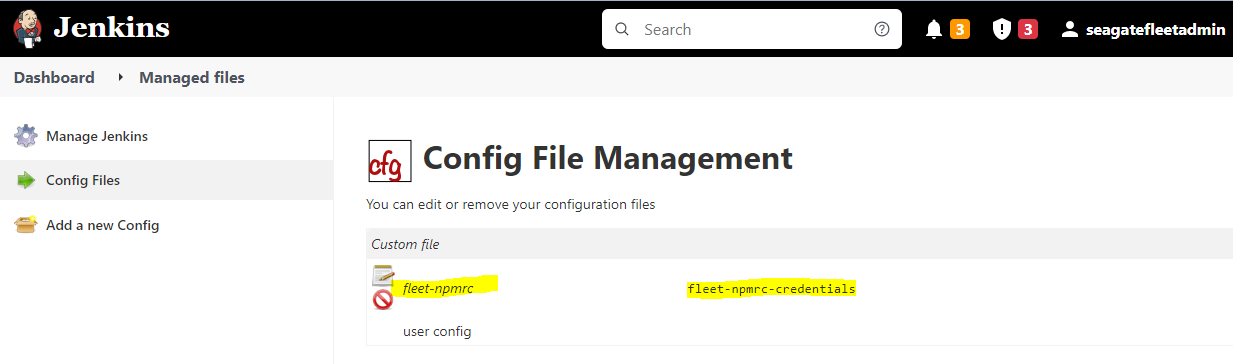
Here we have created below list of secrets:



For NPMRC secrets we have created below path:

Manage Jenkins--->Managed file





8. **CLAIR SCANNER INSTALLATION**

-----------------------------------

1. docker run -p 5432:5432 -d --name db arminc/clair-db:latest

2. docker run -p 6060:6060 --link db:postgres -d --name clair arminc/clair-local- scan:latest