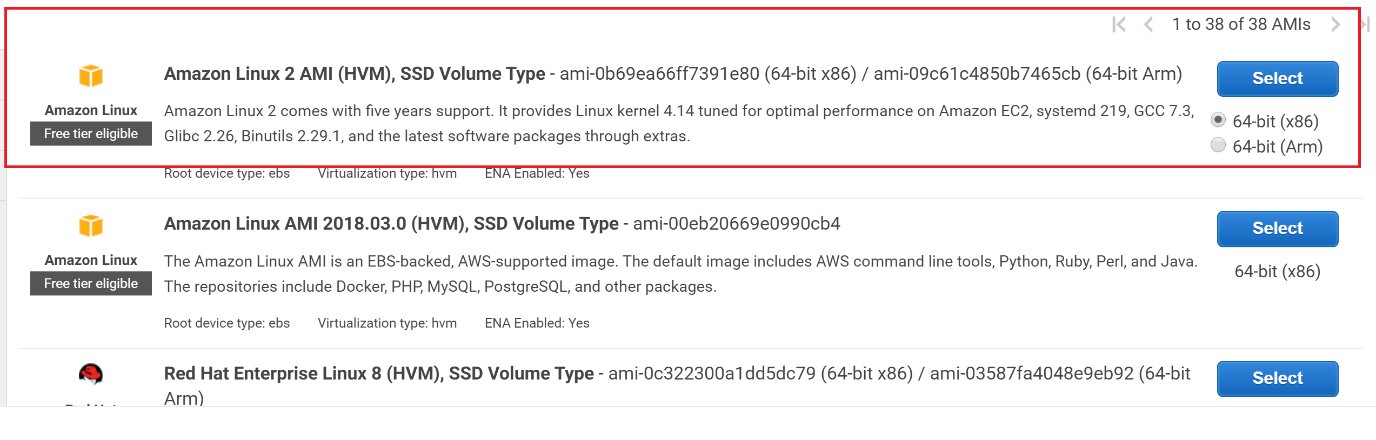
**Jenkins**: **Continuous Integration**/ Continuous deployment/ delivery tool

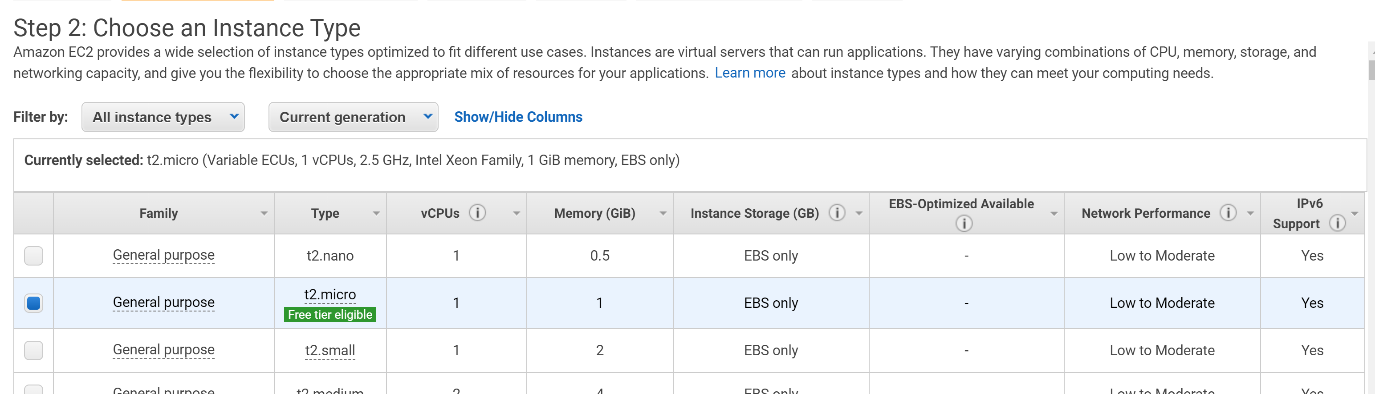
Configure a pipeline, it will pull up the latest from your github repository, it will try to do Maven build (compile, test cases, packaging (war file)) and if there is any failure, it will send mail to all the team members (manager+ scrum master+ whole team): build has failed. Either u revert your changes immediately, fix that.

Installing Jenkins:

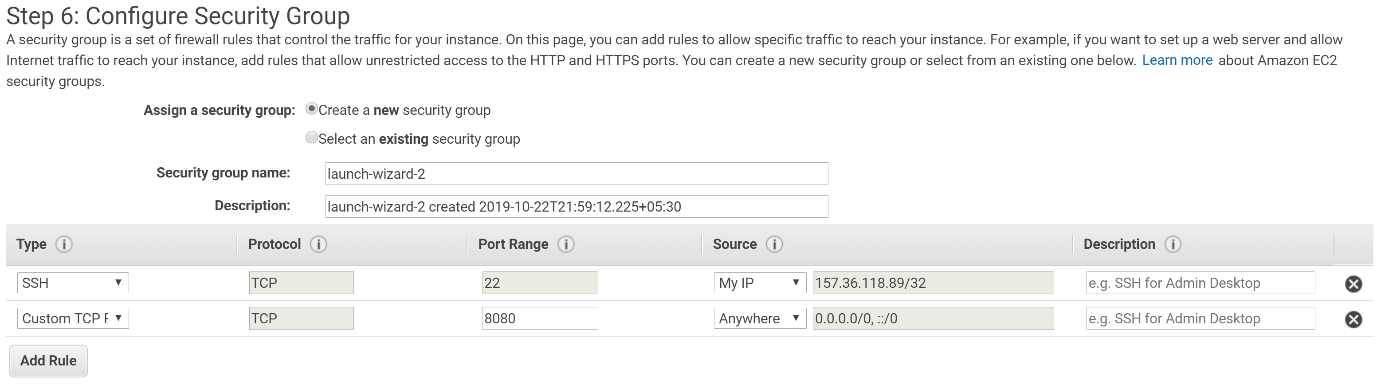
**EC2 service of AWS: buy a new machine**

Launch Instance



****

**VPC:** Virtual private cloud. By default, applications can on same cloud can communicate with each other. Else, if they are on diff clouds, lot of config is needed.



Login to your machine, it will ask for a private key. The one downloaded is a private key but in a diff format (pem file) . This needs to be converted to a ppk file. (different format)

Puttygen : to convert your pem file to a ppk file format.

Putty: to connect to ur machine and install softwares/ application on it (do ssh)

**AWS:**

1. You can scale on demand (8GB memory -> 32 GB-> 64GB-> 16GB). I pay only for what I use.
2. You don’t have to worry about managing the server, downtime, upgrading ..

**Install java & tomcat on linux machine**

yum install java-1.8.0-openjdk

java -version

sudo useradd -m -U -d /opt/tomcat -s /bin/false tomcat

cd /tmp

wget https://www-eu.apache.org/dist/tomcat/tomcat-9/v9.0.27/bin/apache-tomcat-9.0.27.tar.gz

tar -xf apache-tomcat-9.0.27.tar.gz

sudo mv apache-tomcat-9.0.27 /opt/tomcat/

sudo ln -s /opt/tomcat/apache-tomcat-9.0.27 /opt/tomcat/latest

sudo chown -R tomcat: /opt/tomcat

sudo sh -c 'chmod +x /opt/tomcat/latest/bin/\*.sh'

sudo nano /etc/systemd/system/tomcat.service

Paste the following content:

/etc/systemd/system/tomcat.service

**[Unit]**

Description=Tomcat 9 servlet container

After=network.target

**[Service]**

Type=forking

User=tomcat

Group=tomcat

Environment="JAVA\_HOME=/usr/lib/jvm/jre"

Environment="JAVA\_OPTS=-Djava.security.egd=file:///dev/urandom"

Environment="CATALINA\_BASE=/opt/tomcat/latest"

Environment="CATALINA\_HOME=/opt/tomcat/latest"

Environment="CATALINA\_PID=/opt/tomcat/latest/temp/tomcat.pid"

Environment="CATALINA\_OPTS=-Xms512M -Xmx1024M -server -XX:+UseParallelGC"

ExecStart=/opt/tomcat/latest/bin/startup.sh

ExecStop=/opt/tomcat/latest/bin/shutdown.sh

**[Install]**

WantedBy=multi-user.target

Copy

Save and close the file.

cat /etc/systemd/system/tomcat.service

sudo systemctl daemon-reload

sudo systemctl enable tomcat

sudo systemctl start tomcat

sudo systemctl status tomcat

sudo firewall-cmd --zone=public --permanent --add-port=8080/tcp

sudo yum install firewalld

sudo systemctl start firewalld

sudo systemctl enable firewalld

sudo systemctl status firewalld

sudo firewall-cmd --zone=public --permanent --add-port=8080/tcp

sudo firewall-cmd –reload

[**Configure Tomcat Web Management Interface**](https://linuxize.com/post/how-to-install-tomcat-9-on-centos-7/#configure-tomcat-web-management-interface)

sudo nano /opt/tomcat/latest/conf/tomcat-users.xml

/opt/tomcat/latest/conf/tomcat-users.xml

**<tomcat-users>**

*<!--*

*Comments*

*-->*

**<role** rolename="admin-gui"**/>**

**<role** rolename="manager-gui"**/>**

**<user** username="admin" password="admin\_password" roles="admin-gui,manager-gui"**/>**

**</tomcat-users>**

By default Tomcat web management interface is configured to allow access only from the localhost. If you want to be able to access the web interface from a remote IP or from anywhere which is not recommended because it is a security risk you can open the following files and make the following changes. If you need to access the web interface from anywhere open the following files and comment or remove the lines highlighted in yellow:

sudo nano /opt/tomcat/latest/webapps/manager/META-INF/context.xml

/opt/tomcat/latest/webapps/manager/META-INF/context.xml

**<Context** antiResourceLocking="false" privileged="true" **>**

*<!--*

*<Valve className="org.apache.catalina.valves.RemoteAddrValve"*

*allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />*

*-->*

**</Context>**

sudo nano /opt/tomcat/latest/webapps/host-manager/META-INF/context.xml

/opt/tomcat/latest/webapps/host-manager/META-INF/context.xml

**<Context** antiResourceLocking="false" privileged="true" **>**

*<!--*

*<Valve className="org.apache.catalina.valves.RemoteAddrValve"*

*allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" />*

*-->*

**</Context>**

If you need to access the web interface only from a specific IP, instead of commenting the blocks add your public IP to the list. Let’s say your public IP is 41.41.41.41 and you want to allow access only from that IP:

/opt/tomcat/latest/webapps/manager/META-INF/context.xml

**<Context** antiResourceLocking="false" privileged="true" **>**

**<Valve** className="org.apache.catalina.valves.RemoteAddrValve"

allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1|41.41.41.41" **/>**

**</Context>**

Copy

/opt/tomcat/latest/webapps/host-manager/META-INF/context.xml

**<Context** antiResourceLocking="false" privileged="true" **>**

**<Valve** className="org.apache.catalina.valves.RemoteAddrValve"

allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1|41.41.41.41" **/>**

**</Context>**

Copy

The list of allowed IP addresses is a list separated with vertical bar |. You can add single IP addresses or use a regular expressions.

sudo systemctl restart tomcat

cat /etc/systemd/system/tomcat.service

# **Install Jenkins:**

curl --silent --location http://pkg.jenkins-ci.org/redhat-stable/jenkins.repo | sudo tee /etc/yum.repos.d/jenkins.repo

sudo rpm --import <https://jenkins-ci.org/redhat/jenkins-ci.org.key>

sudo yum install jenkins

**Before running the Jenkins, make sure your 8080 port is available or else you could run Jenkins on any other available port by simply changing the port inside the configuration file of CentOS rpm based linux i.e. /etc/sysconfig/jenkins file(The location in debian based linux is /var/default/jenkins), change the port as JENKINS\_PORT=”8081”, Jenkins\_args**

[ec2-user ~]$ sudo service jenkins start

To start the jenkins service at boot-up, you can run,

[ec2-user ~]$ chkconfig jenkins on

or

[ec2-user ~]$ systemctl start jenkins.service

[ec2-user ~]$ systemctl enable jenkins.service

Then, finally you could be able to visit your server IP or public DNS at the above port to get the jenkins dashboard by making sure that the port is open i.e. you added the inbould rule for that port on your AMI server. As, in my case the port is 8081, I’ll visit the following address, <http://server-ip-address:8081/>

Find Jenkins running on which port: Default is 8080

sudo yum install net-tools

netstat -pnltu

To make Jenkins run on port 80:

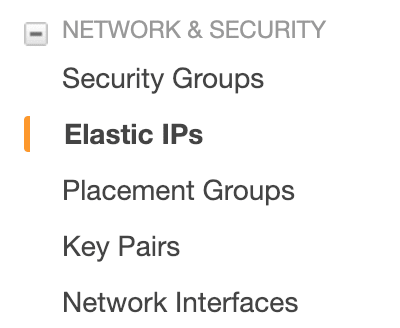
|  |  |
| --- | --- |
|  | sudo iptables -A PREROUTING -t nat -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8081 |

sudo cat /var/lib/jenkins/secrets/initialAdminPassword

The default workspace directory of jenkins in CentOS is “**/var/lib/jenkins**”

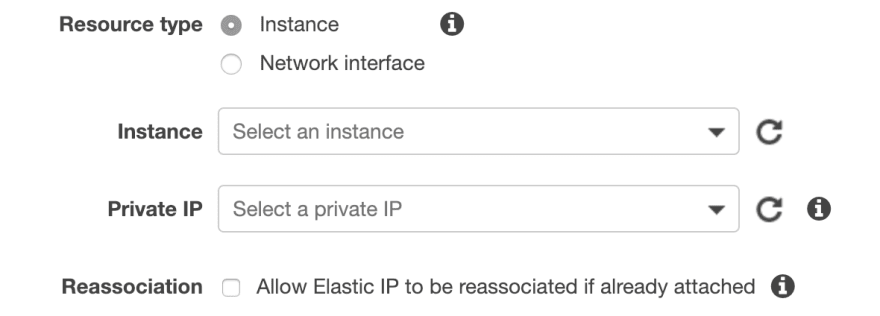
## Assigning an Elastic IP address to the instance and setting a domain.

In the EC2 service, select Elastic IPs option from the nav bar, under "Network & Security" option:

[](https://res.cloudinary.com/practicaldev/image/fetch/s--pS-Fxf5p--/c_limit%2Cf_auto%2Cfl_progressive%2Cq_auto%2Cw_880/https:/thepracticaldev.s3.amazonaws.com/i/3qvxuixc8jy4vojmbibh.png)

Then click in "Allocate new address" and Allocate a new address, I use "Amazon pool" option. Here you receive the new IP, for example 127.0.0.1. Then select the IP and go to the option "Associate address".

In the next form you need to select the Instance and then click in Associate.

[](https://res.cloudinary.com/practicaldev/image/fetch/s--98YvOn5V--/c_limit%2Cf_auto%2Cfl_progressive%2Cq_auto%2Cw_880/https:/thepracticaldev.s3.amazonaws.com/i/gry7kq396bts8pmkm9lu.png)

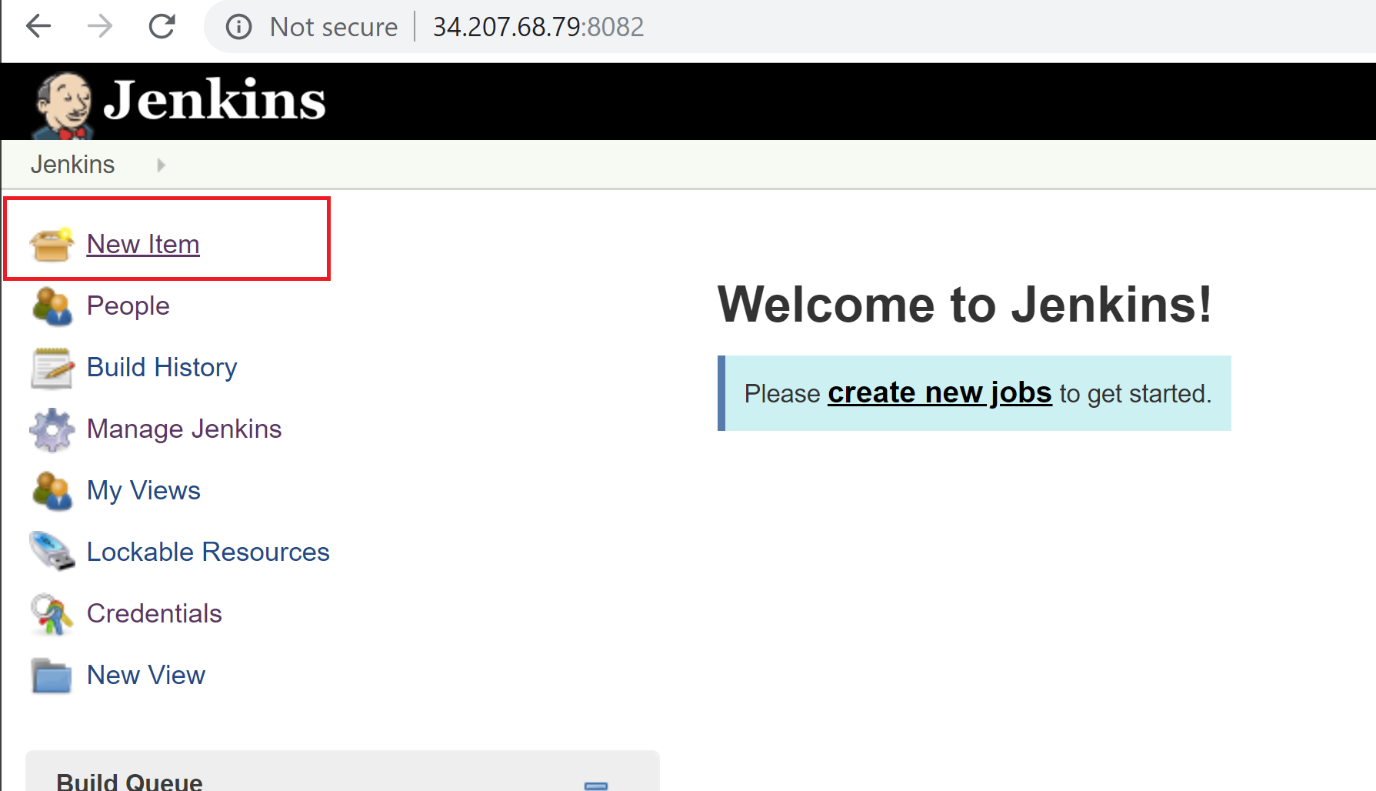
With this now you have the new Instance allocated to the new Elastic IP, so the IP of your instance has changed.

Now we are going to set-up a subdomain, if you have any registered in aws is registered in Route 53. Go to Route 53 service and select "Create Record Set".

Complete the subdomain that you want and:

* Type: A - IPv4 Address
* Alias: NO
* Value: Put the IP that your instance have
* Routing Policy: Simple

Then click on "Save Record Set" and the subdomain it's ready.



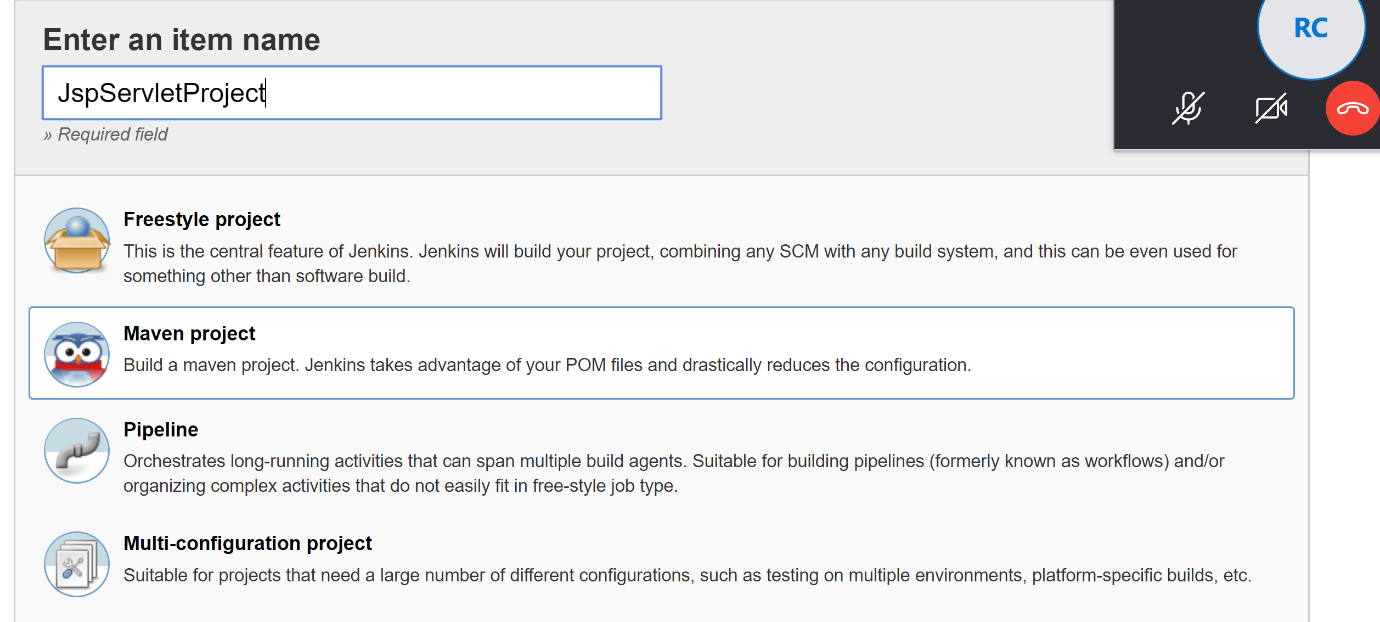
Maven Integration plugin

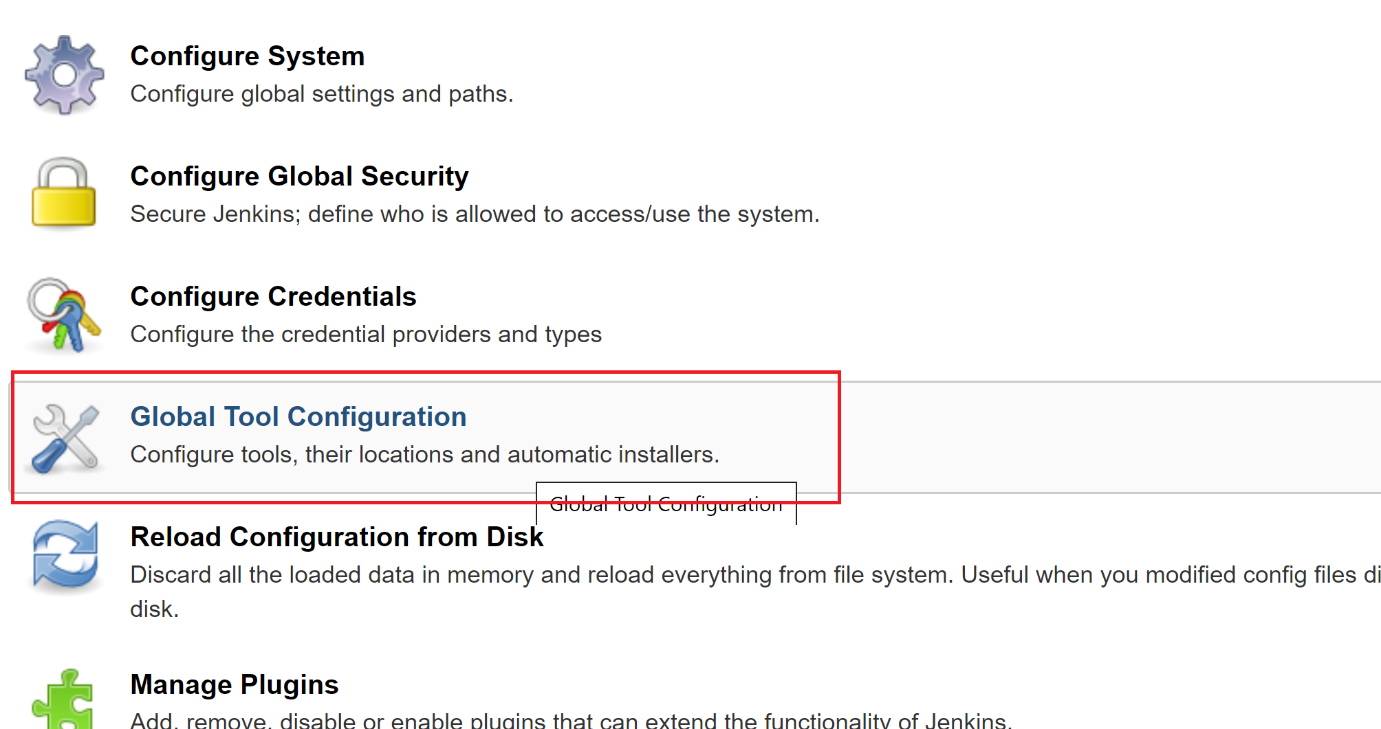
Pipeline

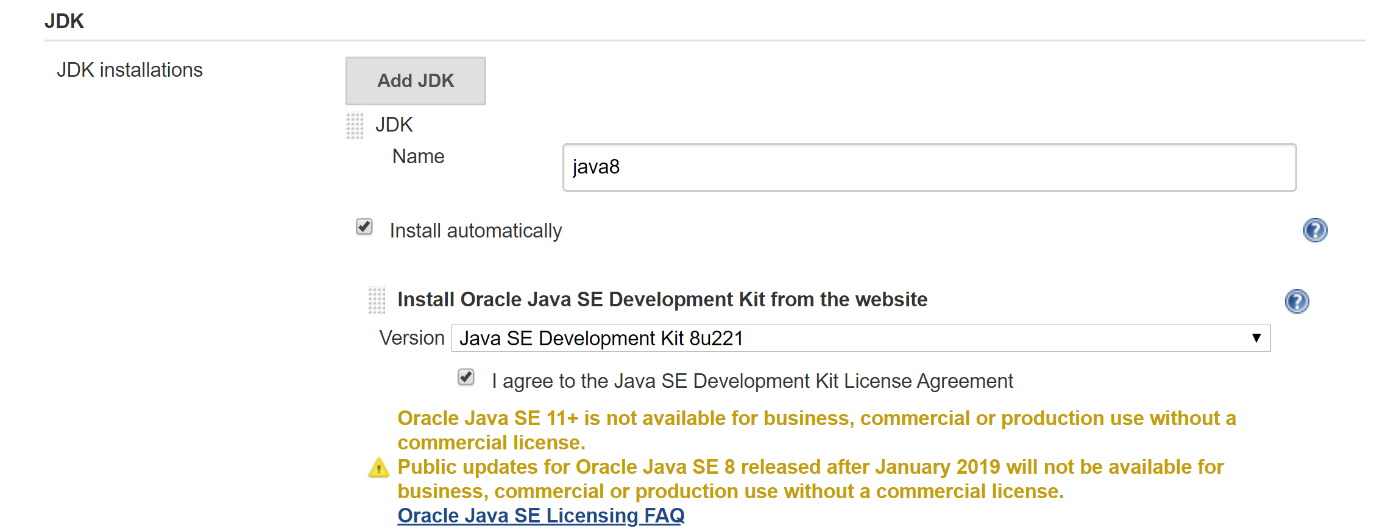
Docker Pipeline

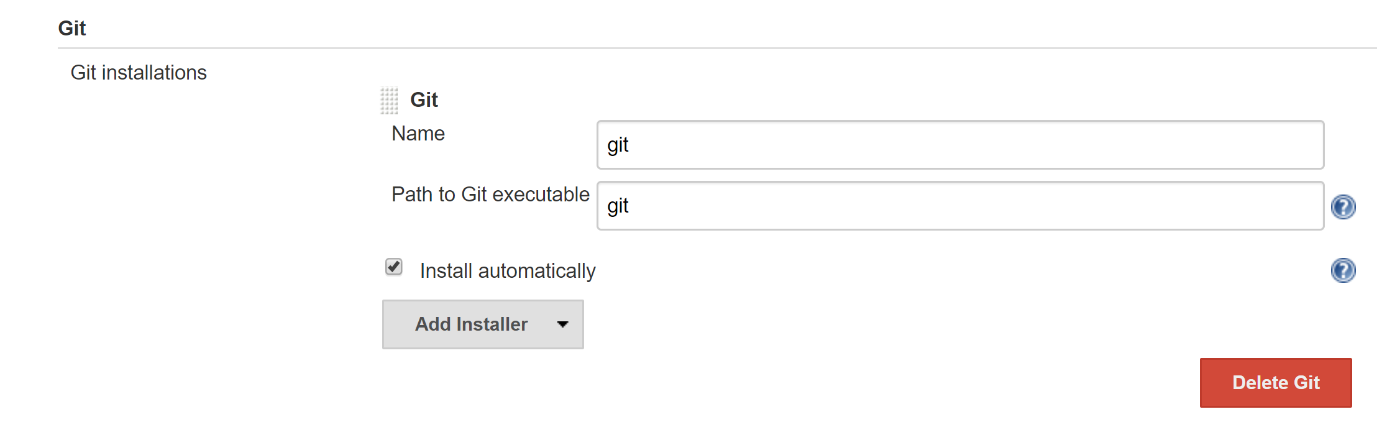
Pipeline Maven Integration

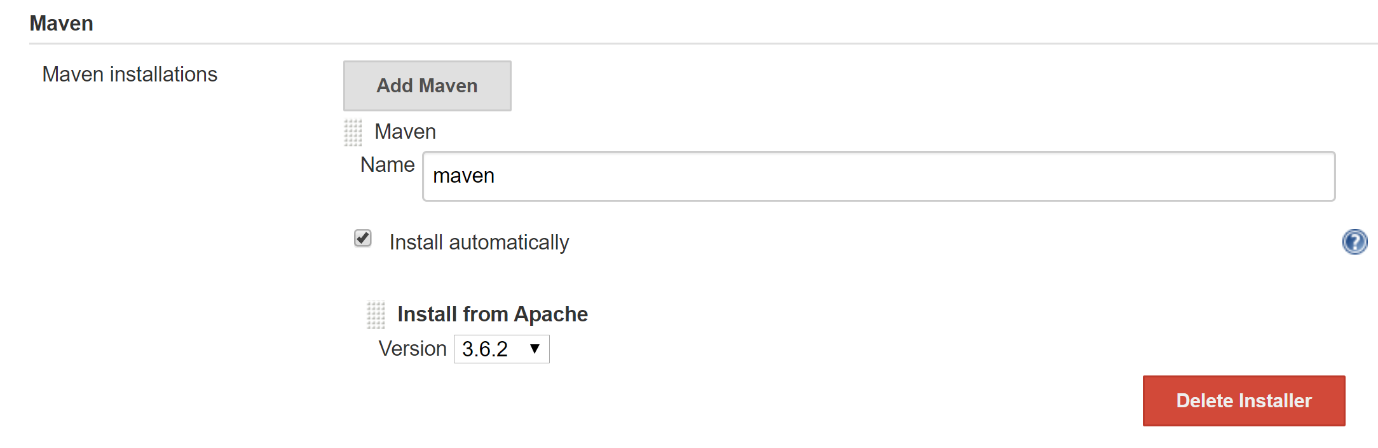
Github Integration











New Maven Project

