**JENKINS MASTER AND SLAVE CONFIGURATIONS**

**JENKINS: -** Jenkins is open-source automation server. It is developed by java language. Jenkins provides hundreds of plugins to support building, deploying and automating any project.

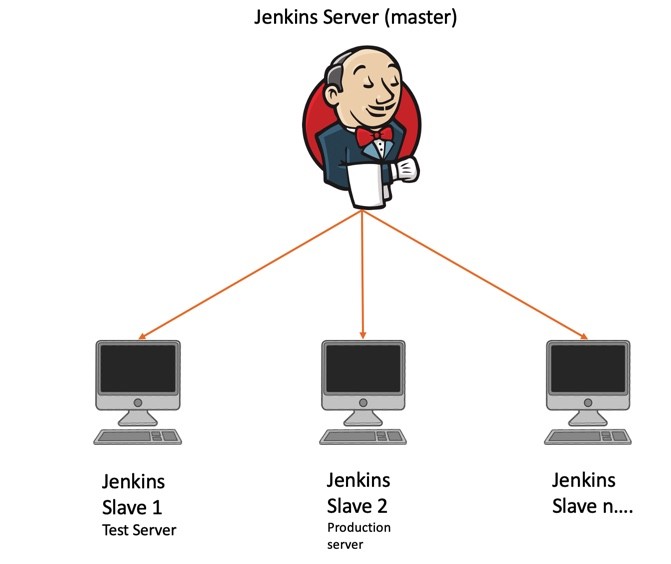
* Path of Jenkins: /var/lib/Jenkins
* Default port: 8080
* Jenkins is used for automate Build and Deployment process. Using Jenkins we can implement CI & CD (Continuous Integration and Continuous Deployment).
* CI & CD is used to build and test software project or application continuously making it easier to integrate changes in the project.

**CONTINUOUS INTEGRATION: -** Continuous Integration is development practice where, when any new code is committed or any code change is happened it will build and test automatically.

**CONTINUOUS DEPLOYMENT: -** Continuous Deployment is a process where any new build is happened it will automatically deploy into production servers (live server).

**JENKINS MASTER AND SLAVE CONFIGURATION: -** The Jenkins master acts to schedule the jobs, assign slaves, and send builds to slaves to execute the jobs. In order to perform testing in different environments, Jenkins uses various Slaves.

**JENKINS MASTER AND SLAVE ARCHITECTURE**



* Create EC2 Instance for master server (kernal version + t2.medium + 16gb storage) and another EC2 Instance for Slave server(kernal + t2.medium). Connect to the instances.
* Install Jenkins in master server by executing Jenkins installations commands.

**SLAVE NODE CONFIGURATIONS**

**Step-1: -** Update the instance

* Command: sudo yum update -y

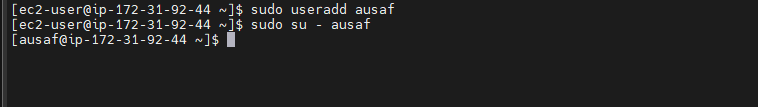
**Step-2: -** Install java in machine

* Command: sudo yum intsall java -y

**Step-3: -** Create a new user and add password to that user.

* Commands: sudo useradd slave3

Sudo passwd slave3

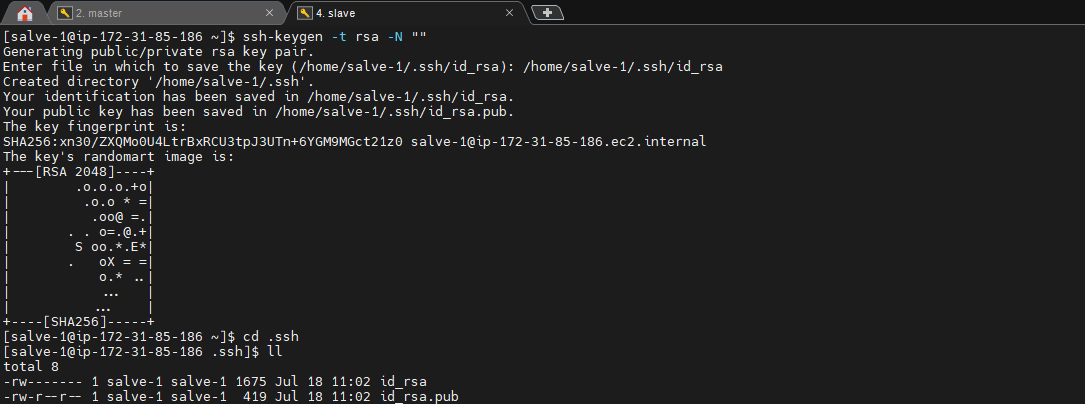


**Step-4: -** Switch to the New user (slave3)

* Command: sudo su - <user name>

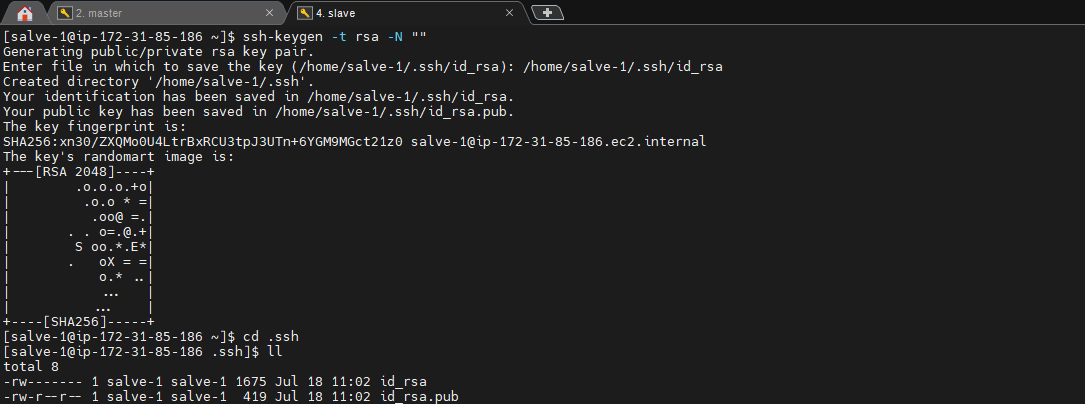
**Step-5: -** Generate a new key in the user.

* Command: ssh-keygen (in /home/slave1/.ssh/id\_rsa path)



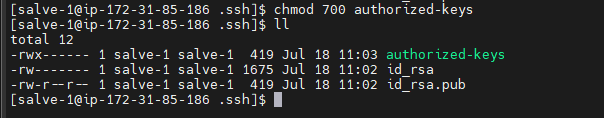
**Step-6: -** Copy public key id\_rsa.pub in new file with name authorized\_keys.

* Command: cat id\_rsa.pub > authorized\_keys



**Step-7: -** Change the permission authorized\_keys file i.e., give all permissions to the user.

* Command: chmod 700 authorized\_keys

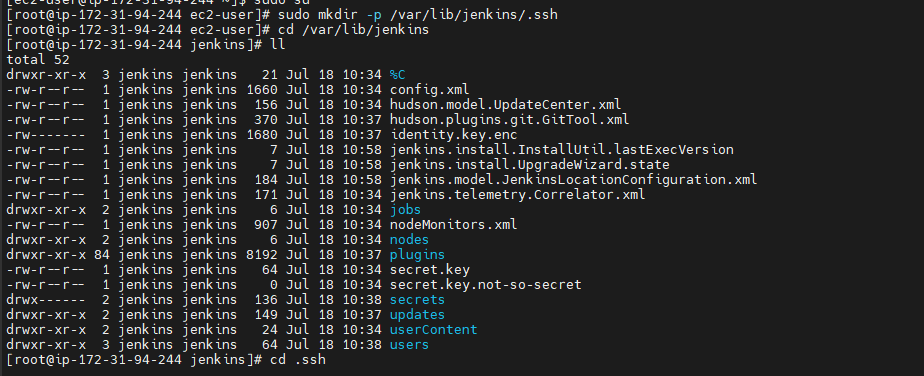


**MASTER SERVER CONFIGURATION**

Copy the slave node public key to master node’s known\_hosts.

**Step-1: -** Create a .ssh directory in path /var/lib/Jenkins/.ssh.

* Command: sudo mkdir -p /var/lib/Jenkins/.ssh



**Step-2: -** Go to the .ssh directory.

* Command: cd /var/lib/Jenkins/.ssh

Cd .. (Come out of .ssh directory)

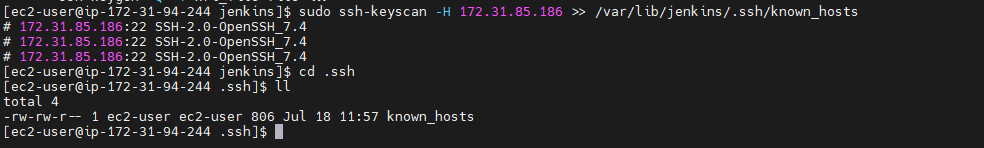
**Step-3: -** Change the permissions of .ssh directory i.e., give all permissions to users, group and others.

* Command: sudo chmod 777 .ssh

Go inside the .ssh directory with cd command.

**Step-4: -** copy the public key(id\_rsa.pub) of slave1 node to known\_hosts.

* Command: sudo ssh-keyscan -H Slave\_Node\_private\_Ip >>/var/lib/Jenkins/.ssh/known\_hosts



**Step-5: -** Go to known\_host directory to check public key of slave1 is copied.

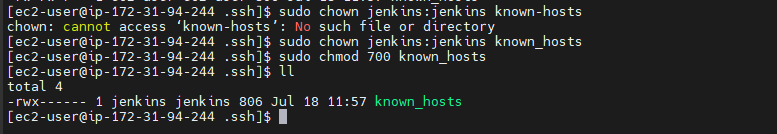
* Command: cd /var/lib/Jenkins/.ssh/known\_hosts

**Step-6: -** Change the ownership known\_hosts from ec2-user to Jenkins.

* Command: sudo chown Jenkins:Jenkins Known\_hosts

**Step-7: -** Change the permissions of known\_hosts i.e., give all permissions to user only.

* Command: sudo chmod 700 known\_hosts



**ATTACHING SLAVE NODE TO MASTER SERVER**

Now attach slave server to master server

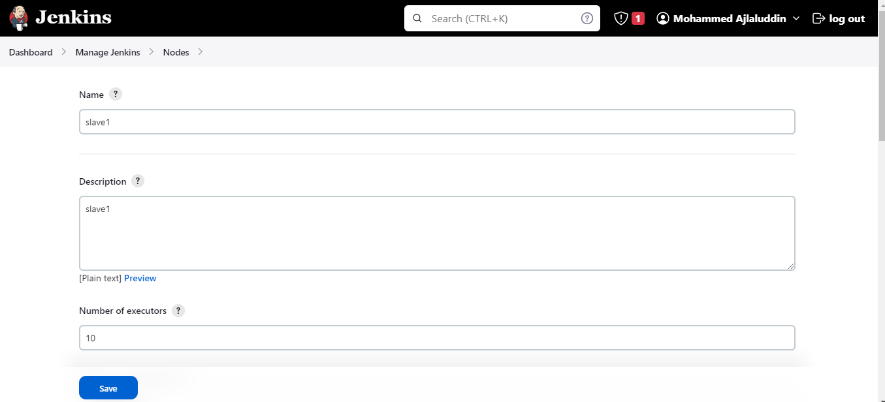
**Step-1: -** Go to Jenkins server.

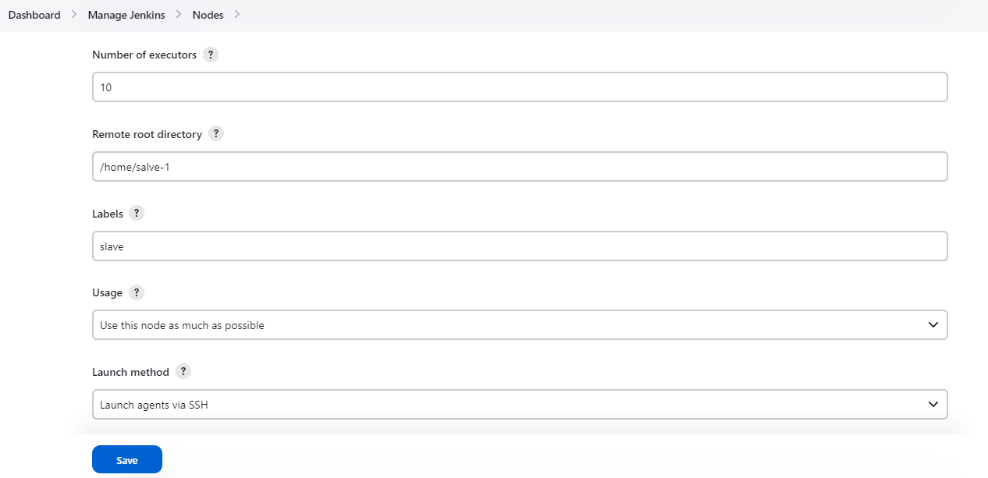
**Step-2: -** Click Manage Jenkins option in Dashboard.

**Step-3: -** Click on Manage Nodes and Cloud and then click on New Node.

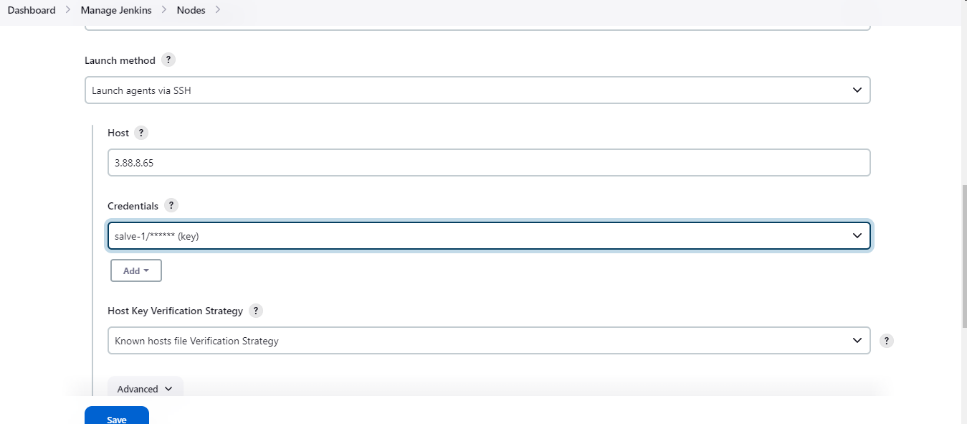
**Step-4: -** Enter the node name and select permanent agent and click ok.

**Step-5: -** Enter the Node configuration such description, label name, number of executors, credentials, remote root directory etc.

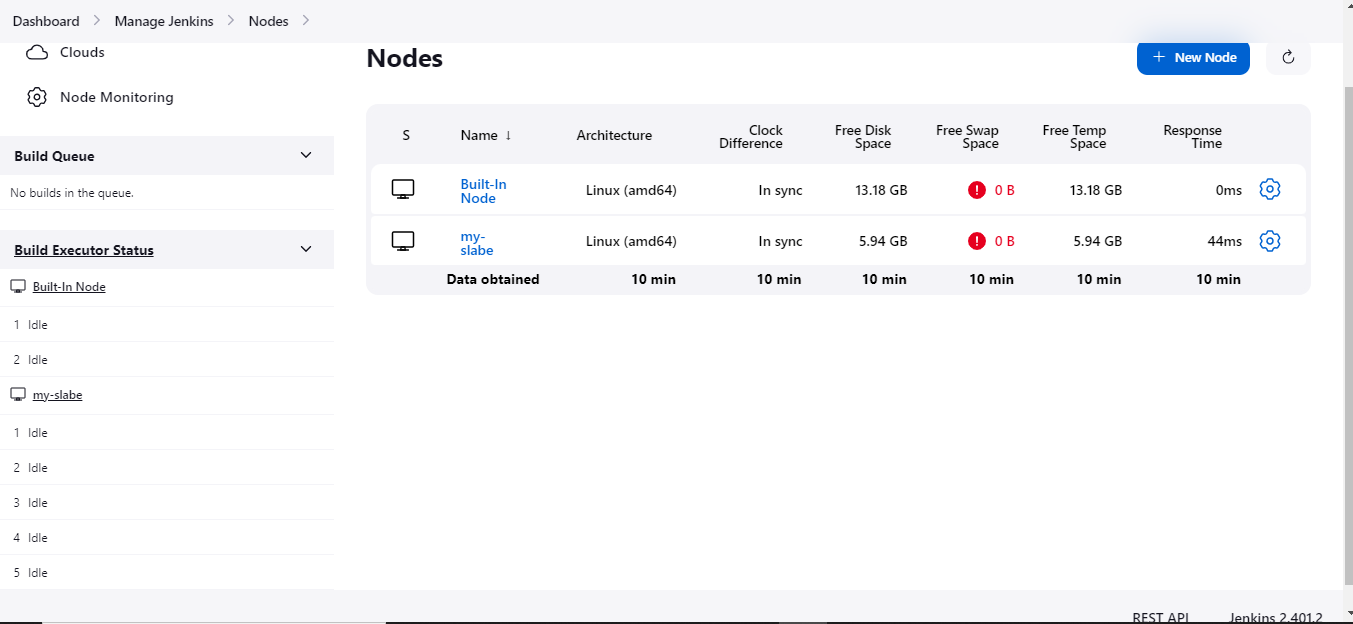




**Step-6: -** While adding credentials choose kind as SSH username with private key and provide user name (slave1) and private key which is generated in slave1 user.



**Step-7: -** Click on save option and check in the log weather the slave server is online or offline.



**PART-2**

**BUILDING MAVEN PROJECT USING JENKINS**

**What is Maven?**

Maven is a popular open-source build tool developed by the Apache Group to build, publish, and deploy several projects at once for better [project management](https://www.simplilearn.com/project-management-skills-article).

* Apache Maven is a software project management and comprehension tool. Based on the concept of a project object model (POM), Maven can manage a project's build, test and package.

**What pom.xml file?**

POM stands for “Project Object Model “. The pom. xml file contains information of project and configuration information for the maven to build the project such as dependencies, build directory, source directory, test source directory, plugin, goals etc.

**What is Apache Tomcat?**

Apache Tomcat is a web container. It allows the users to run Servlet and JAVA Server Pages that are based on the web-applications.

* Apache Tomcat is used to deploy project (java project). It can be used only for hosting JAVA based code.

**TOMCAT INSTALLTION THROUGH LINUX:**

**Step-1:** - Launch a EC2 instance by adding 8080,8081,8090, etc port in security group.

**Step-2: -** Connect to the instance and update instance.

**Step-3: -** Create a new user and go inside the user.

Command: Sudo useradd Ajlal

Sudo su – Ajlal

**Step-4: -** Install java in instance.

* Command: Sudo yum install java 1.8\*

**Step-5: -** Install Tomcat 9 in instance.

* Command: sudo wget <https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.78/bin/apache-tomcat-9.0.78.tar.gz>

**Step-6: -** Once the download is complete, extract tomcat archive files using the following command.

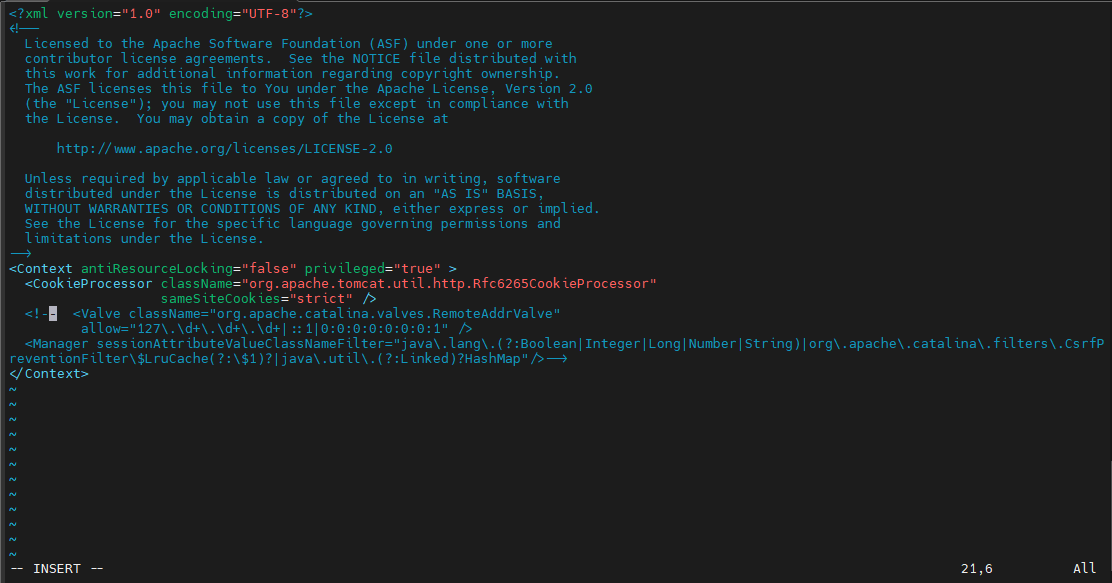
* Command: tar -xzvf apache-tomcat-9.0.78.tar.gz

**Step-7: -** Go to apache-tomcat-9.078 directory with cd command.

**Step-8: -** Edit context.xml file with vi editor i.e., we have to comment last four lines such that we have to (<! —) at begninig of 4th line and ( --> )at last line.

* Command: cd apache-tomcat-9.0.78/webapps/manager/META-IN

Vi context.xml



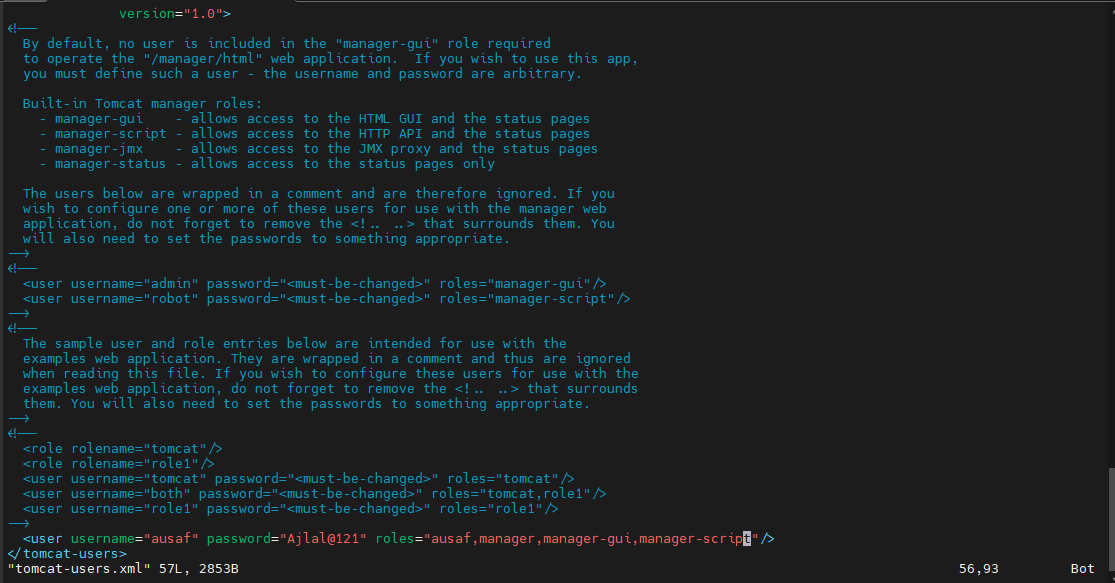
**Step-9: -** Add user and password at end of the tomcat-user.xml file.

* Command: cd apache-tomcat-9.0.78/conf

vi tomcat-user.xml

Add user, password and roles in tomcat-user.xml

<user username= “Ajlal” password= “ajlal121” roles=”Ajlal,manager,manager-gui,manager-script”/>

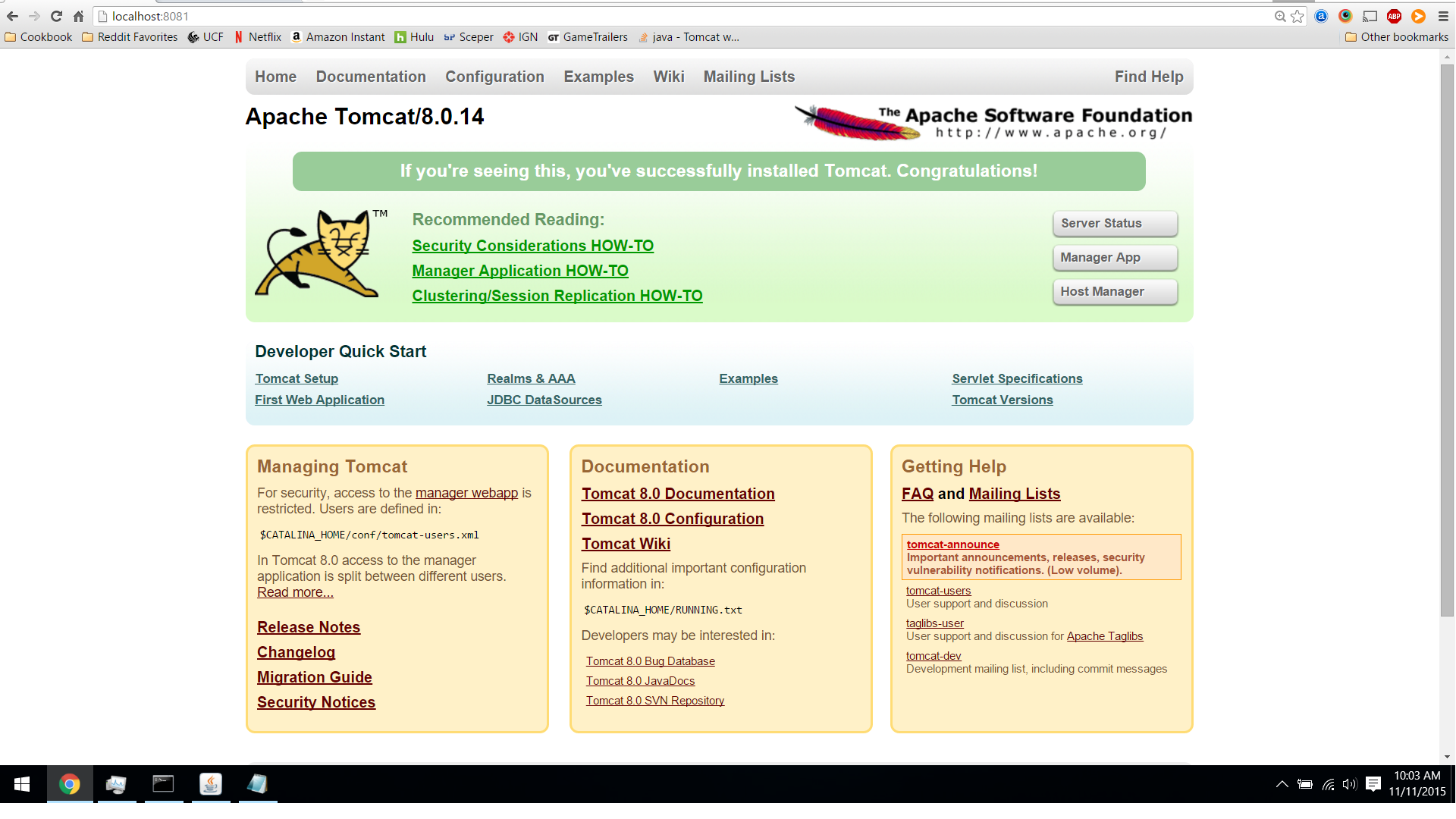


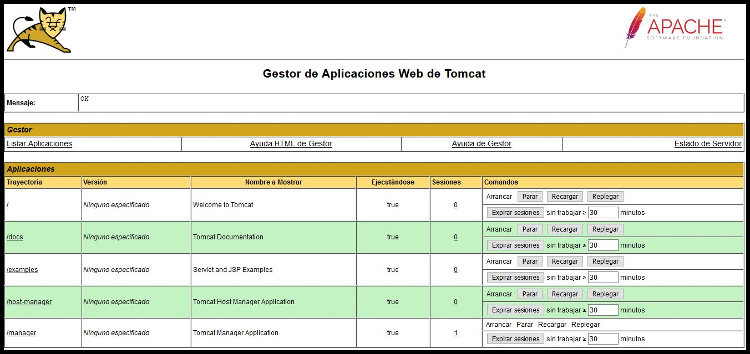
**Step-10: -** Go to bin directory

* Command: cd apache-tomcat-9.0.78/bin

**Step-11: -** Run ./startup.sh command to start tomcat server.

* Browse with public Ip of Instance with 8080 port to use Tomcat Server.





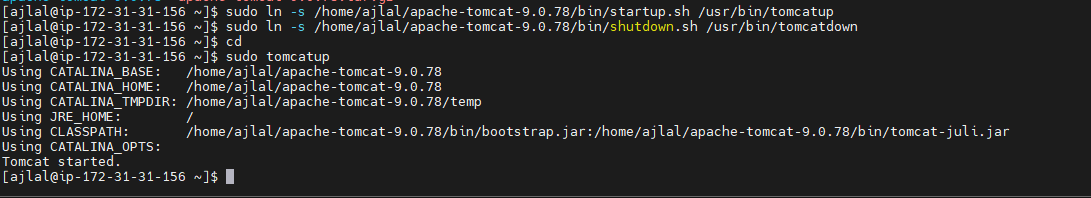
**Step-12: -** Run ./shutdown.sh command to stop tomcat server.

**Note:** To start tomcat server from any directory execute following command.

Command: sudo ln -s /home/ajlal/apache-tomcat-9.0.78/bin/startup.sh /usr/bin/tomcatup

* To stop tomcat server from any directory execute following command.

Command: sudo ln -s /home/ajlal/apache-tomcat-9.0.78/bin/shutdown.sh usr/bin/tomcatdown

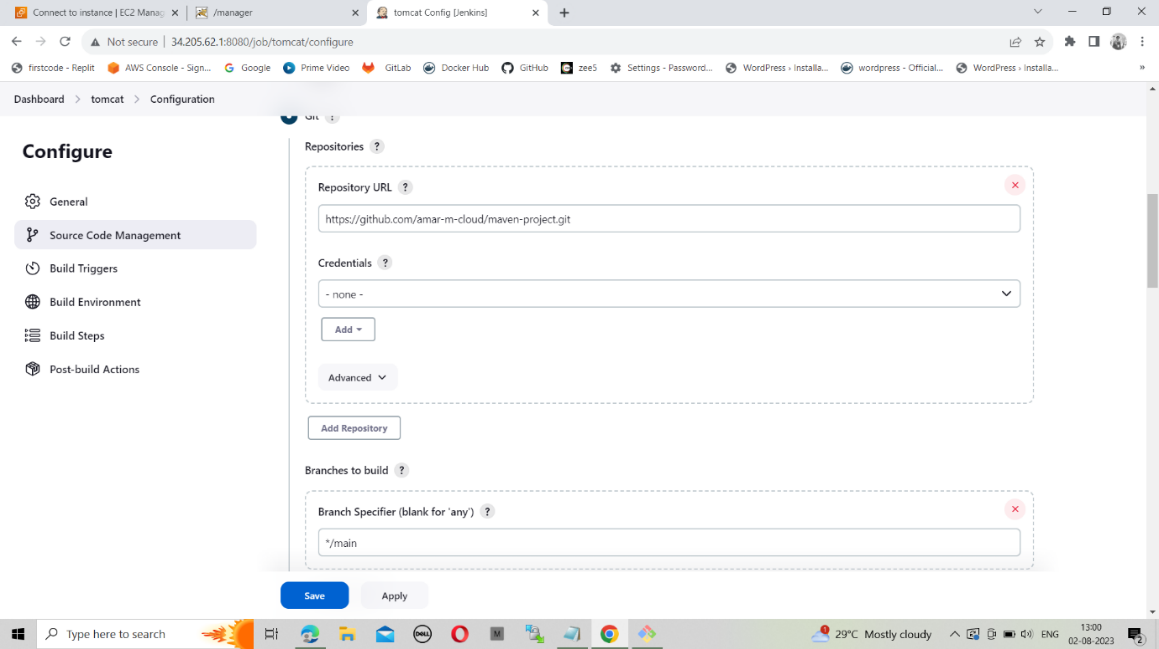


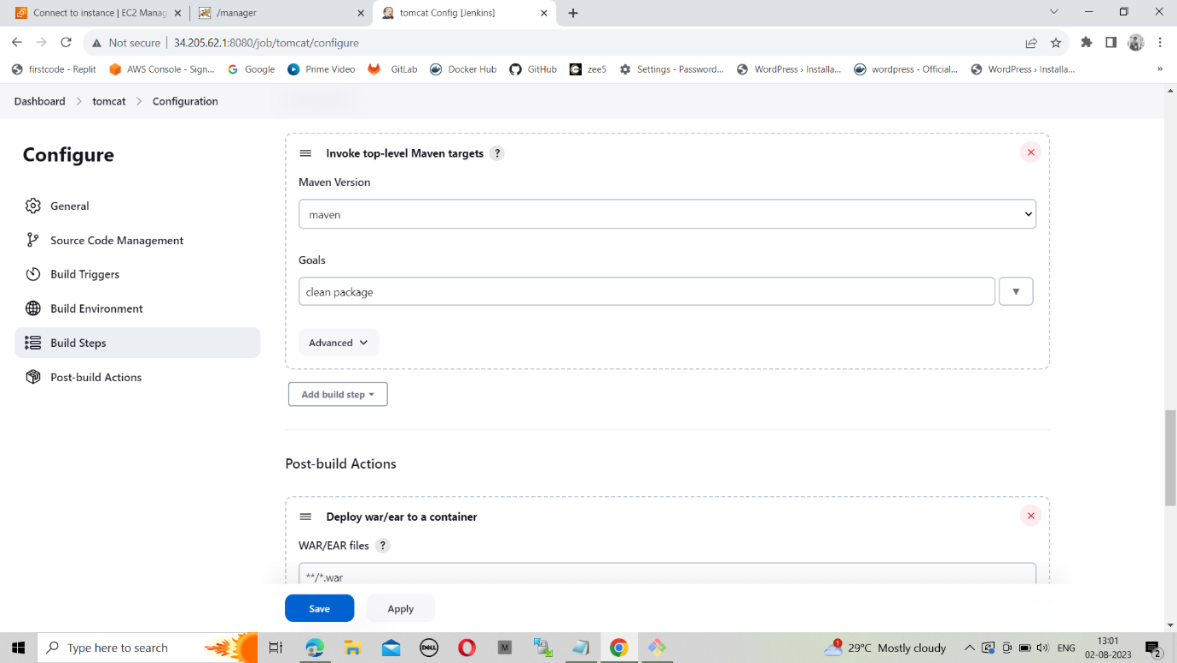
**BUILDING AND DEPLOYING MAVEN PROJECT**

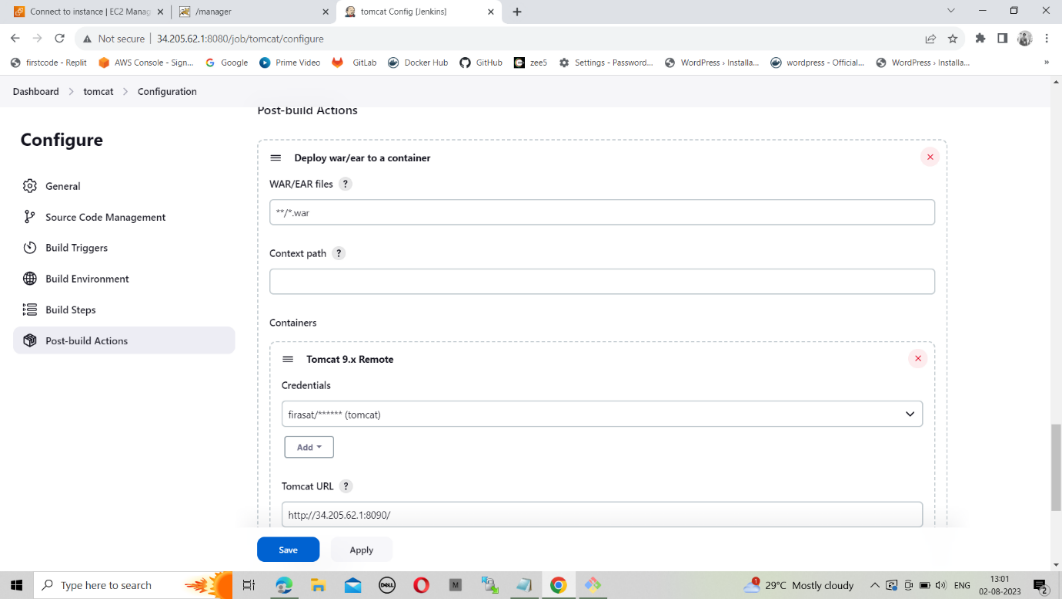
**Step-1: -** login into the Jenkins server with an 8080 port and click on manage Jenkins to configure the system with plugins of maven integration & deploy to container and also add the maven installation tool in the system Tool configuration.

**Step-2: -** Create a job for the Tomcat web application naming tomcat.

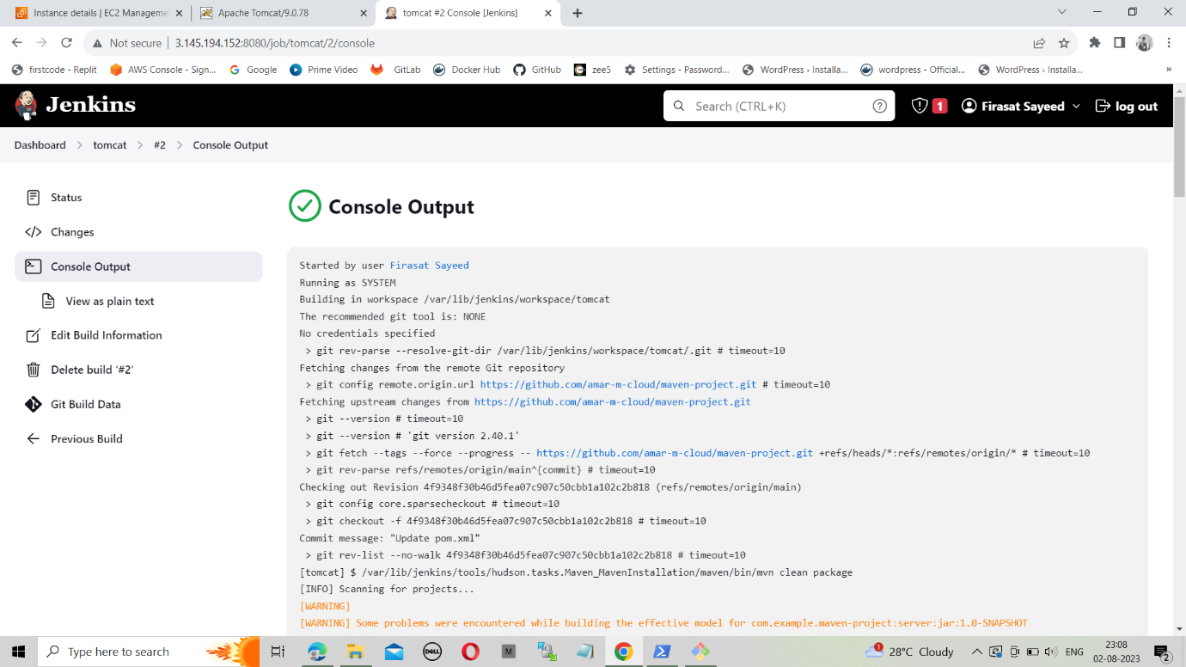
**Step-3: -** Configuring the job in source code management with Git repository, Invoking top-level maven targets in build steps with the goal and post build Actions with Deploy War file to a container.



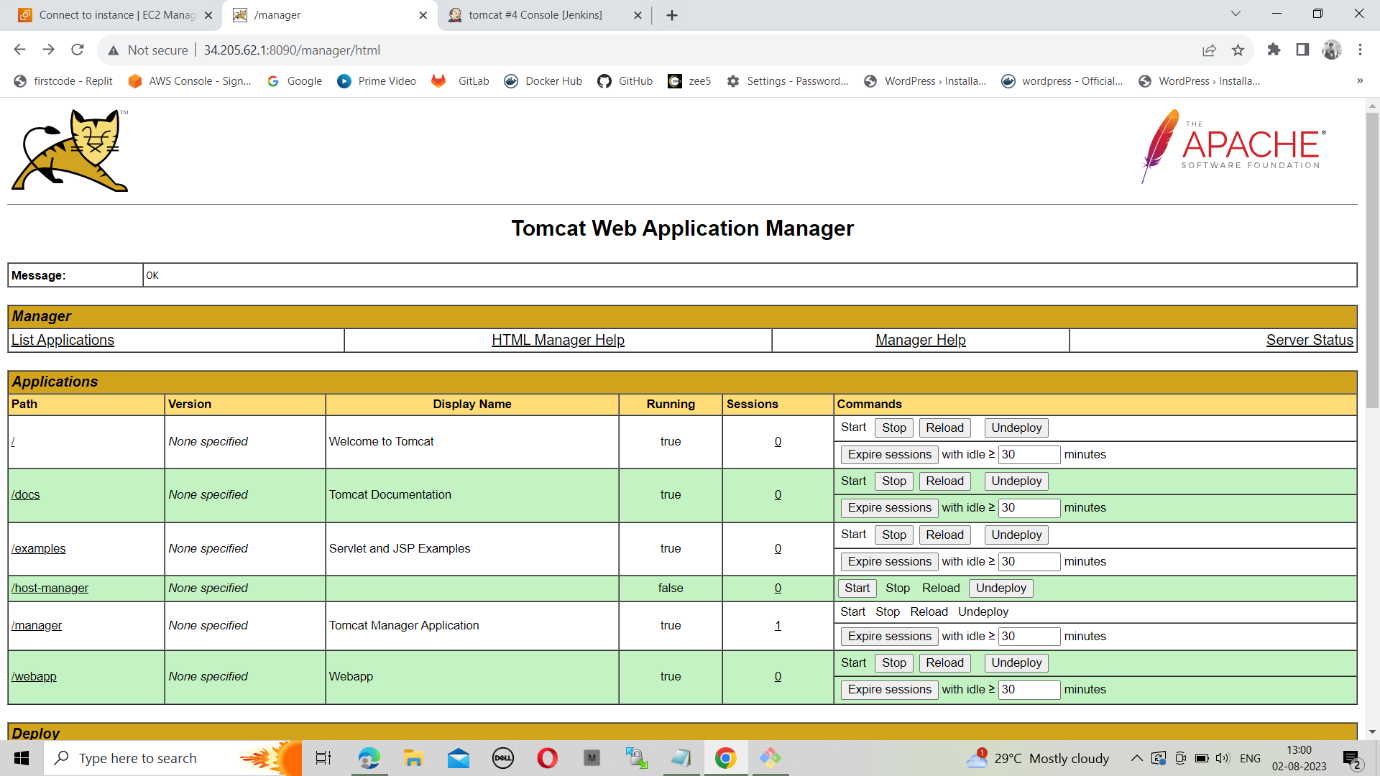




**Step-4: -** Apply and Save the configuration to Build the job assigned in Dashboard.



**Step-5: -** Check the Web application output in the Apache Tomcat server.



---------------------------------------------------------x----------------------------------------------------------------