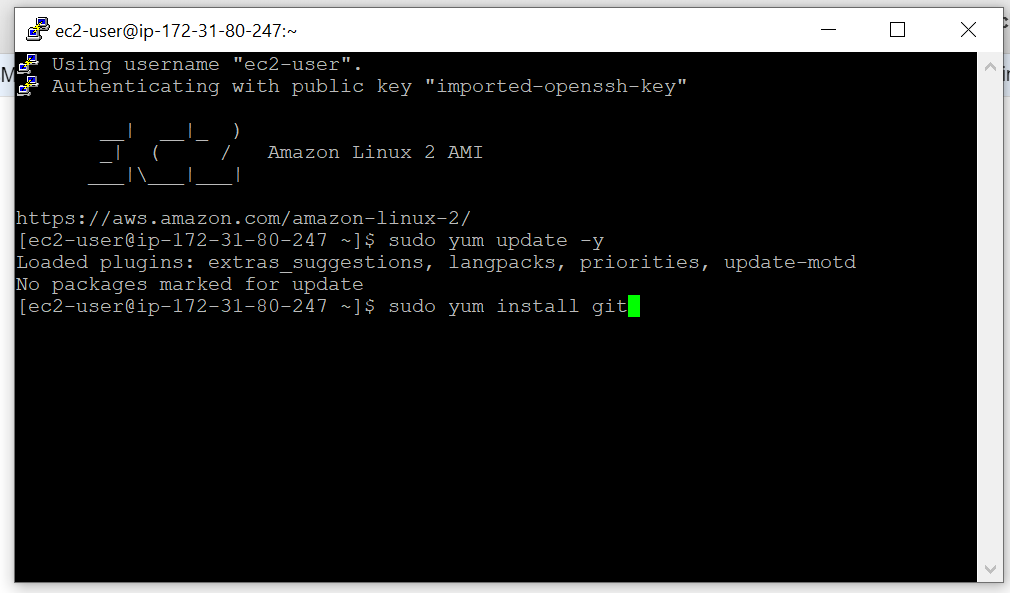
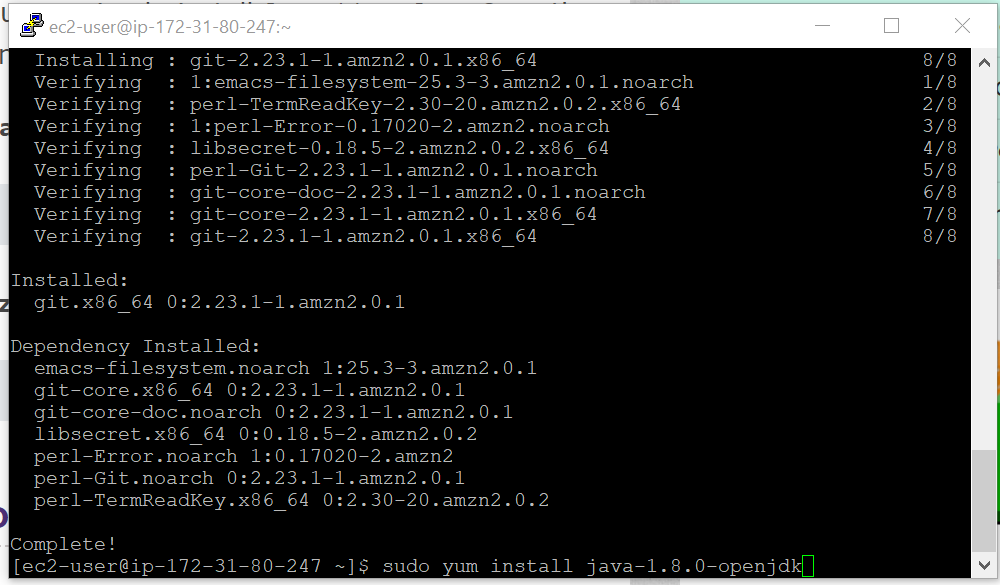
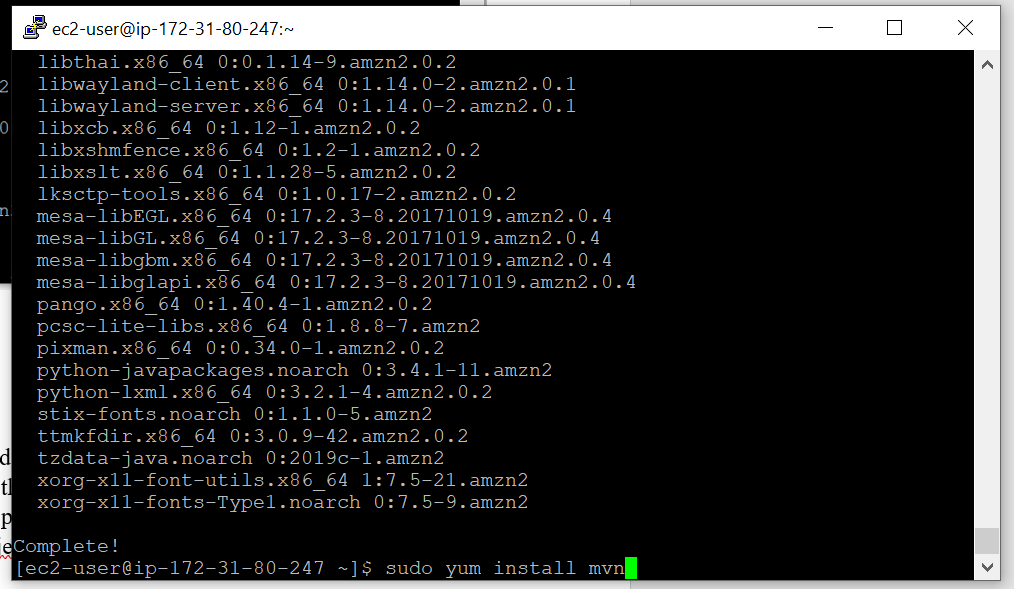
Assignment 2: Building a Jenkins Pipeline that builds a sample java application and hosts in multiple tomcat servers.

**Submitted By: Tushar Das**

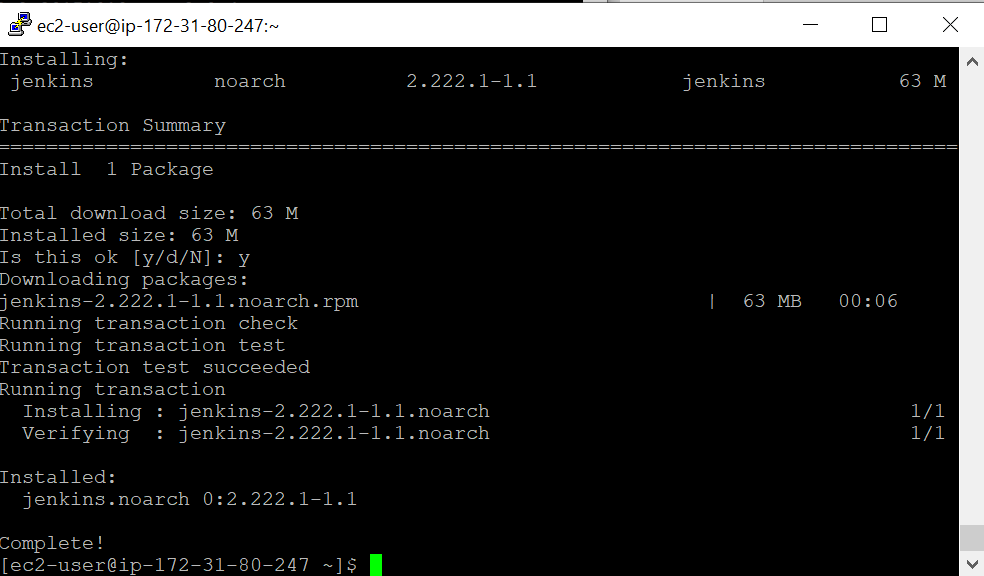
The following servers are used:   
a) JenkinsMaster -- To host the Jenkins service,  
b) AppServer1 and AppServer2 -- To host war files on Apache Tomcat server  
  
We need the following installations on JenkinsMaster server:  
1) Git   
cmd: sudo yum install git

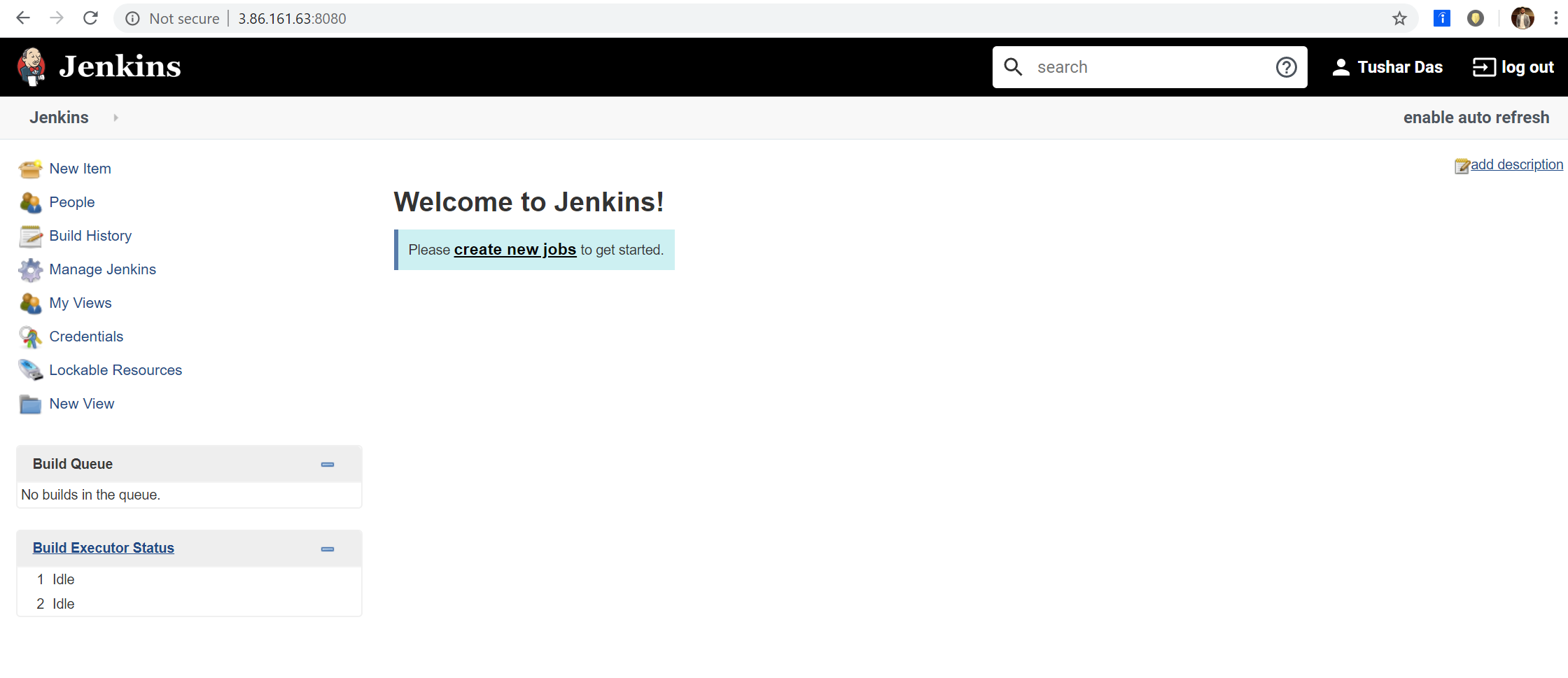


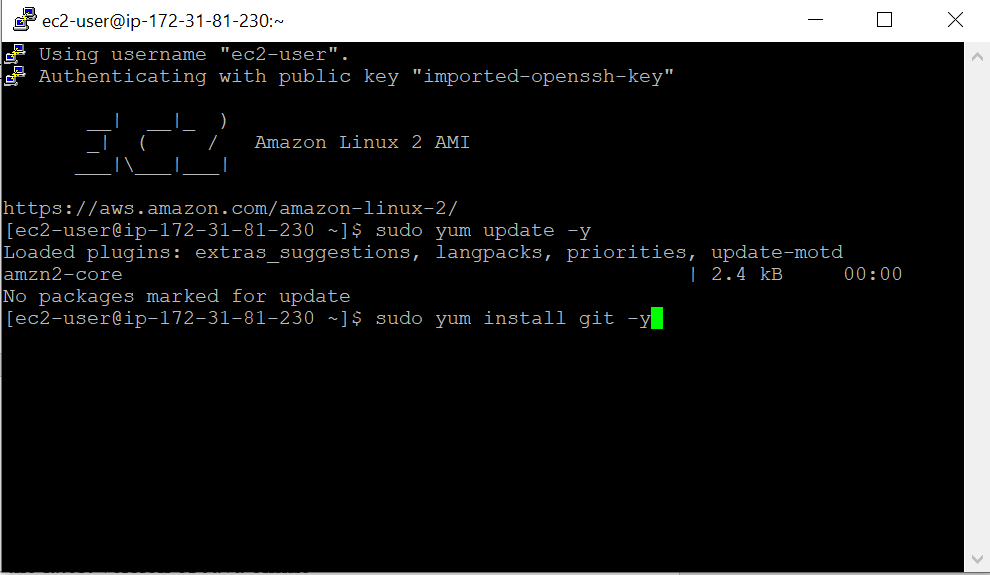
2) Java  
cmd: sudo yum install java-1.8.0-openjdk

  
  
3) Maven   
cmd: sudo yum install maven  


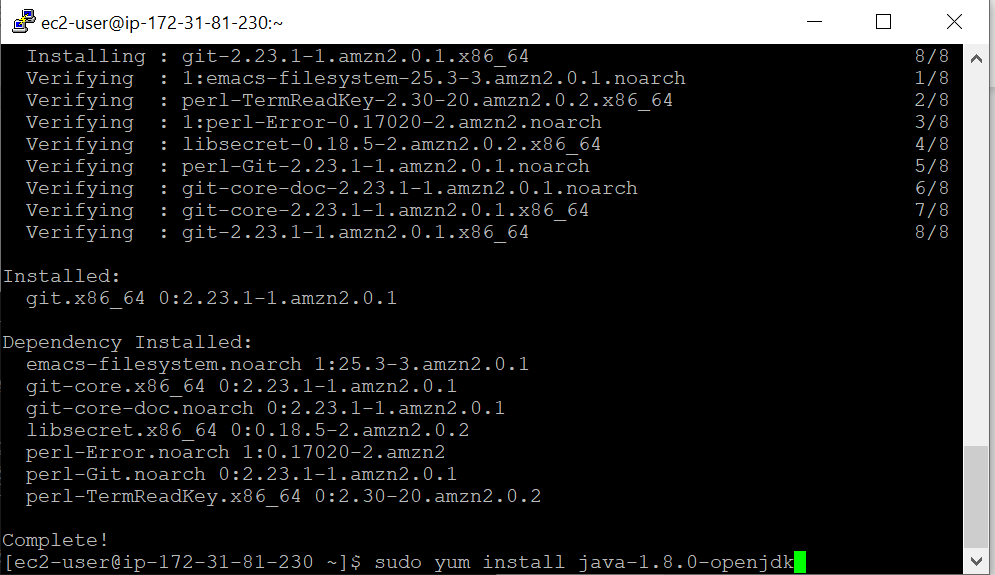
4) Jenkins   
First, the jenkins repos should be downloaded from the Jenkins website. This will save the repository in the /etc/yum.repos.d. After which the key provided along with the repo should be downloaded as well. This key will provide the password when Jenkins is accessed later.   
cmd to install Jenkins: sudo yum install Jenkins. Then we must use the cmd: “sudo service jenkins start” to start the Jenkins service on our server. By default, it runs on port 8080.



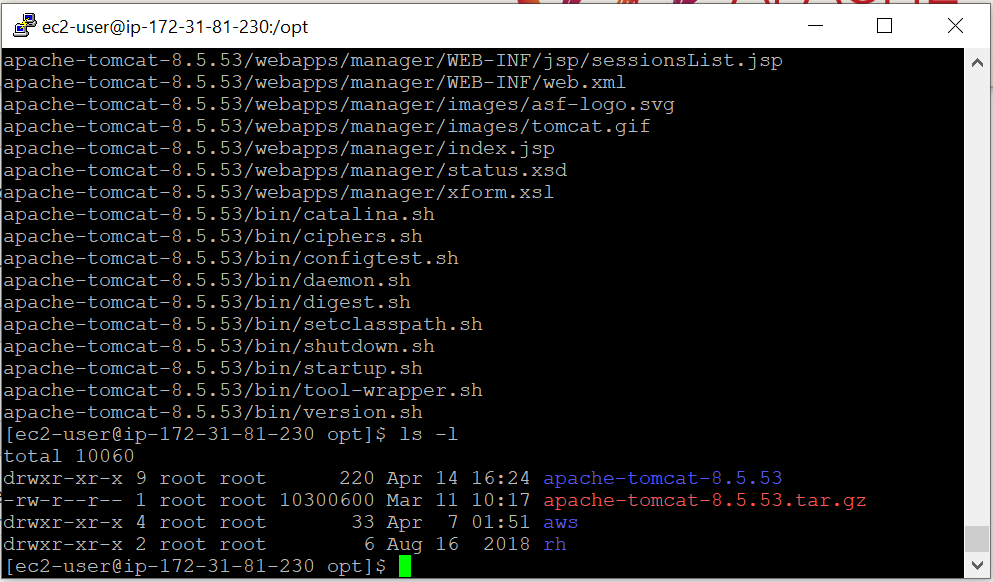


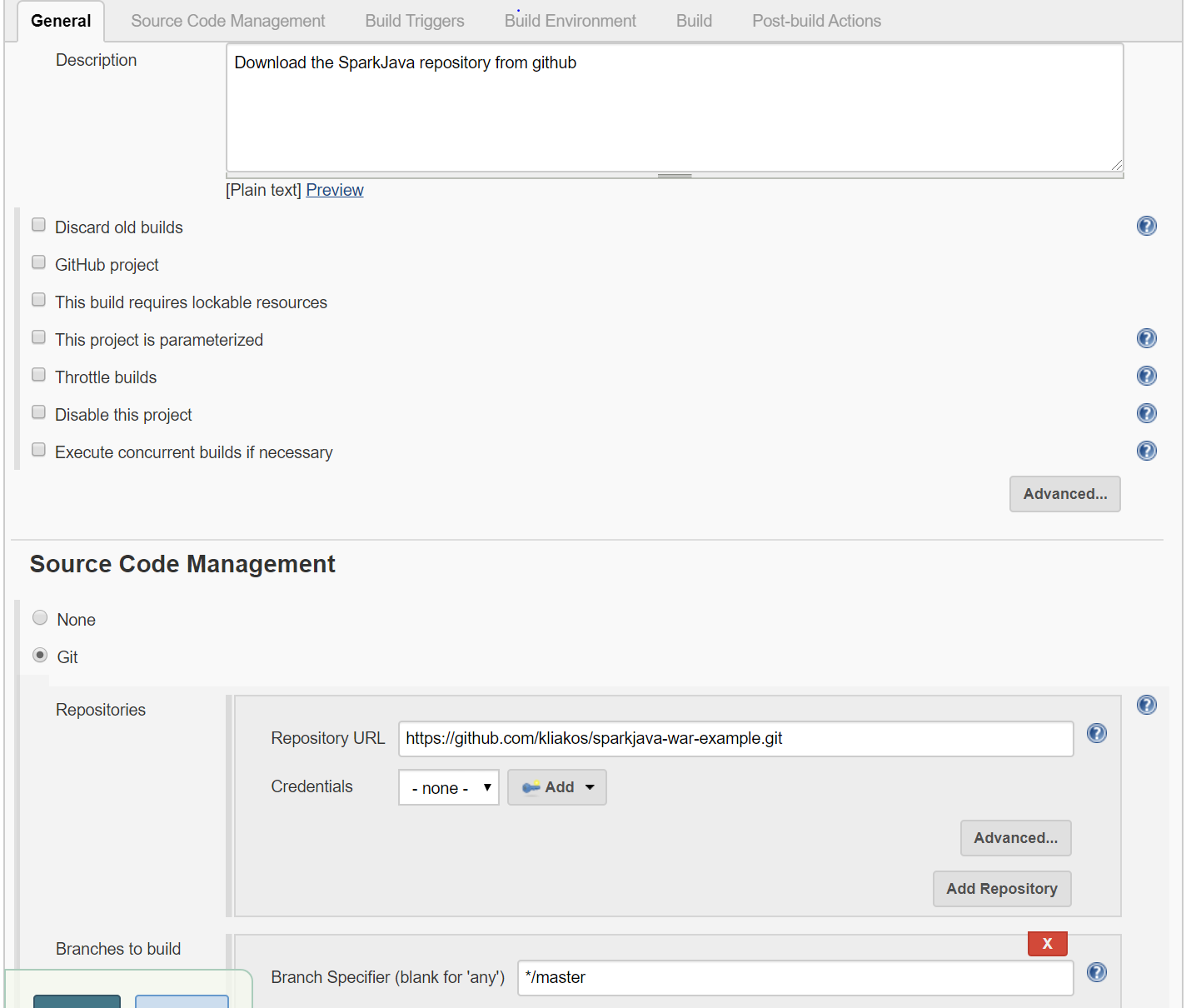
We need the following installations on AppServer1 and AppServer2:  
  
1) Git   
cmd: sudo yum install git -y  


2) Java  
cmd: sudo yum install java-1.8.0-openjdk

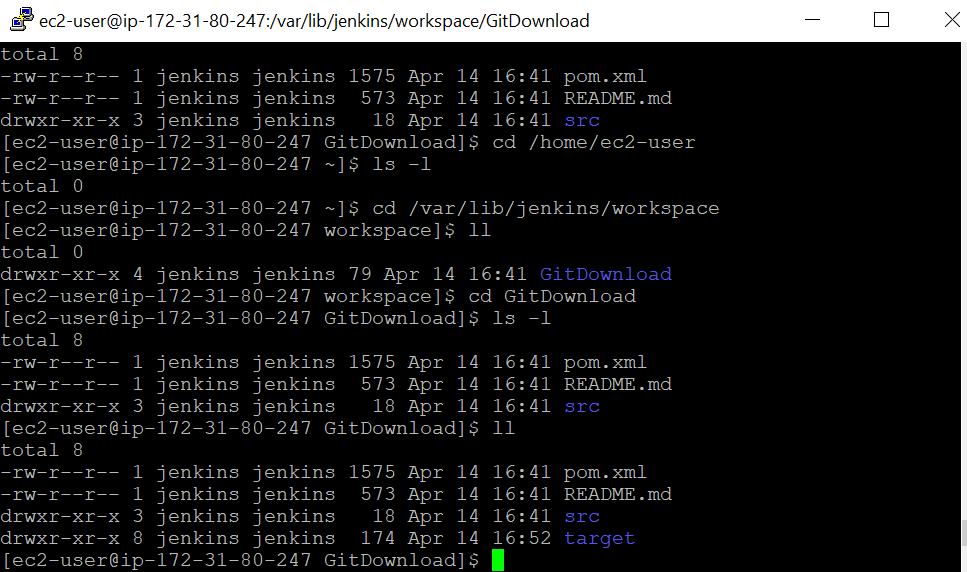
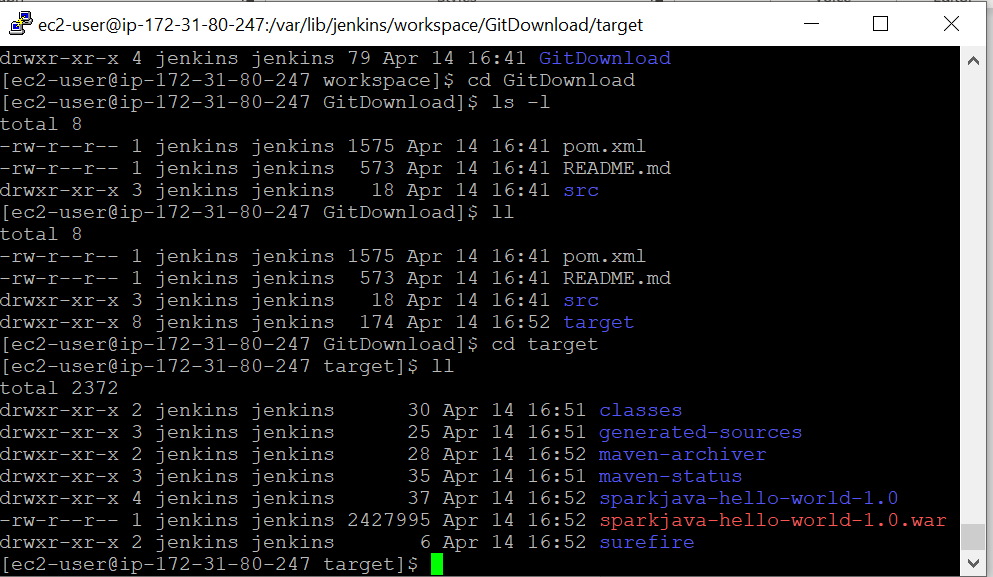
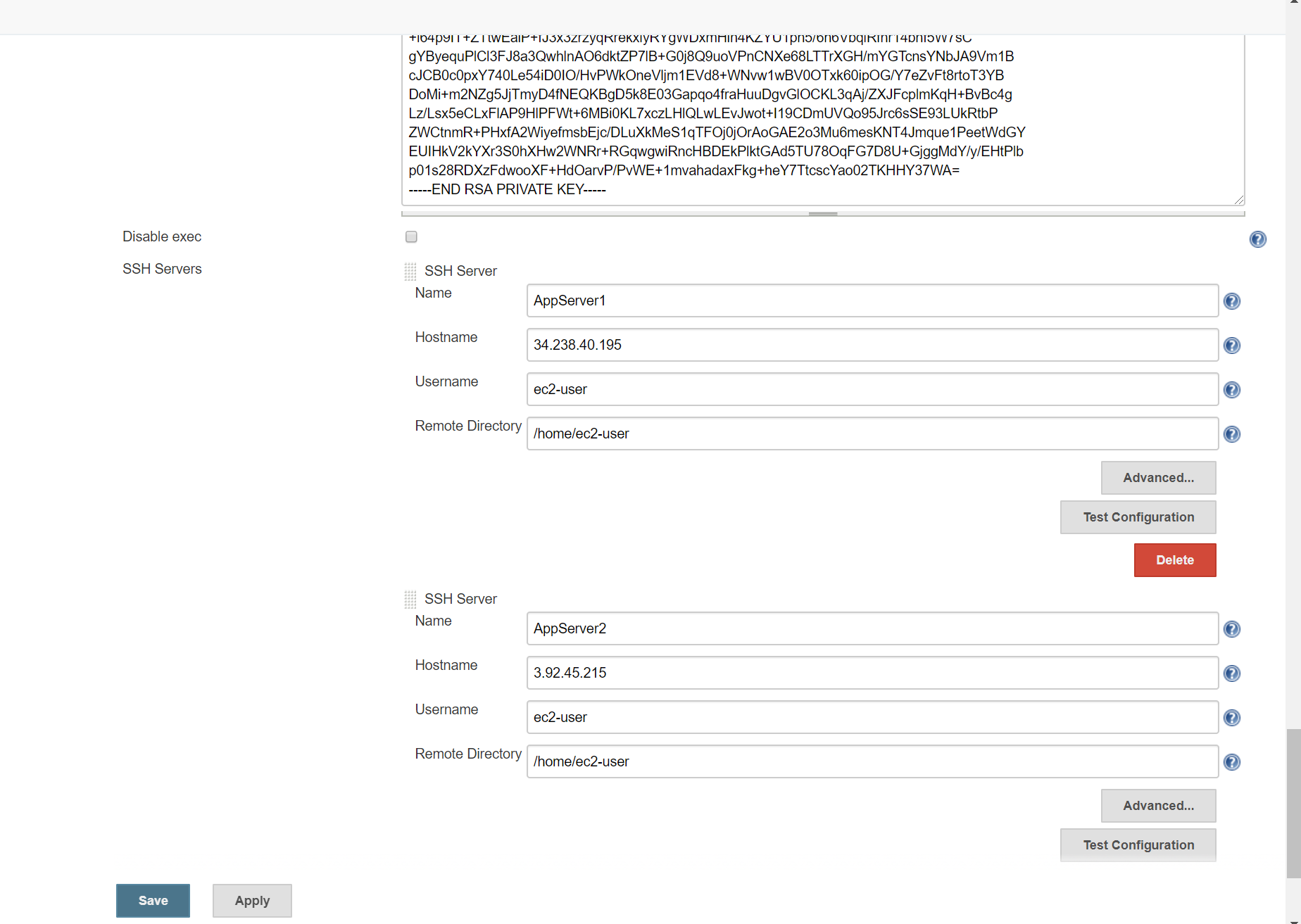


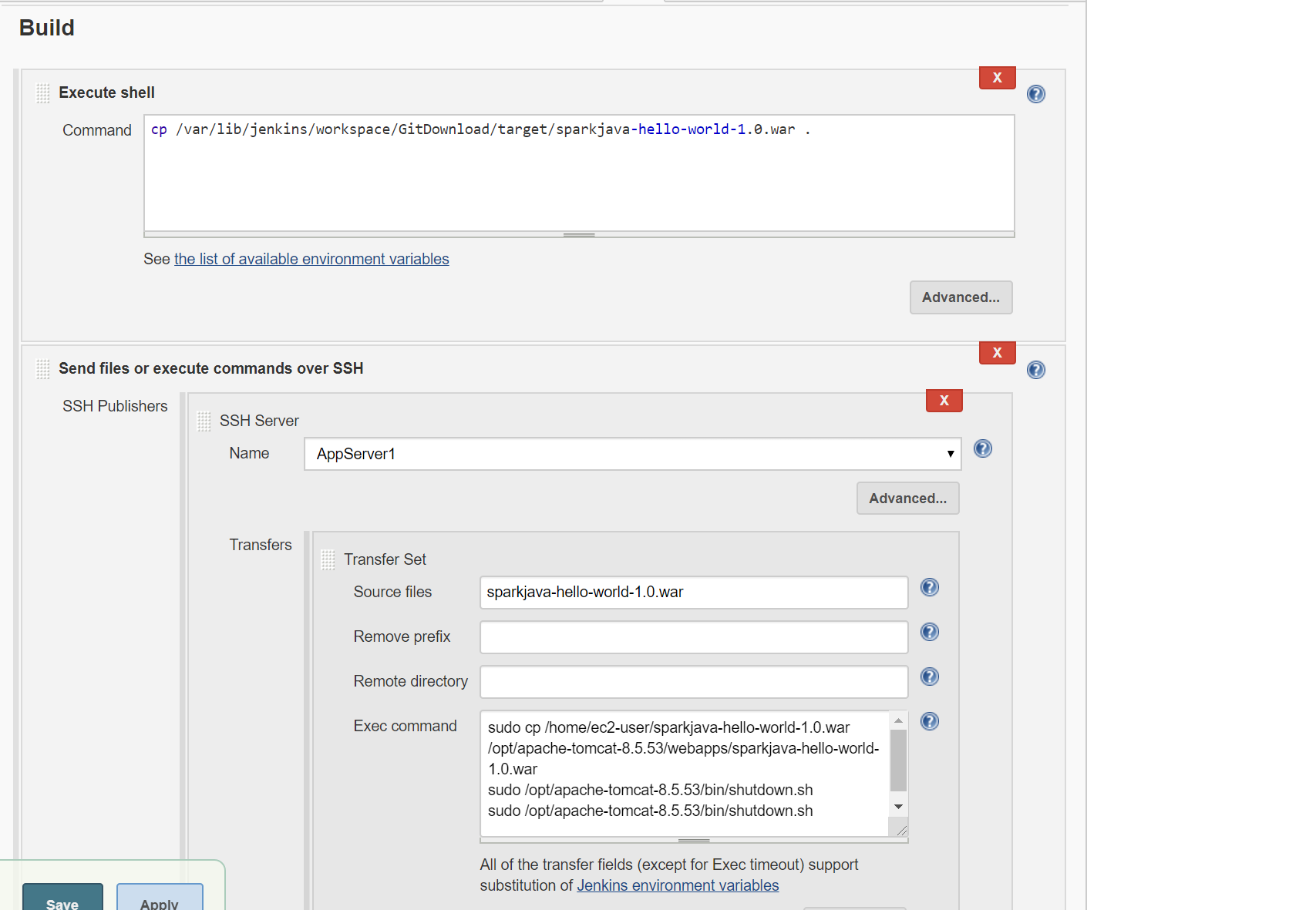
3) Apache Tomcat   
The .tar file for Apache was downloaded from the Apache Tomcat website into the /opt directory. The tar -xvzf cmd gives us access to the Apache Tomcat folder.



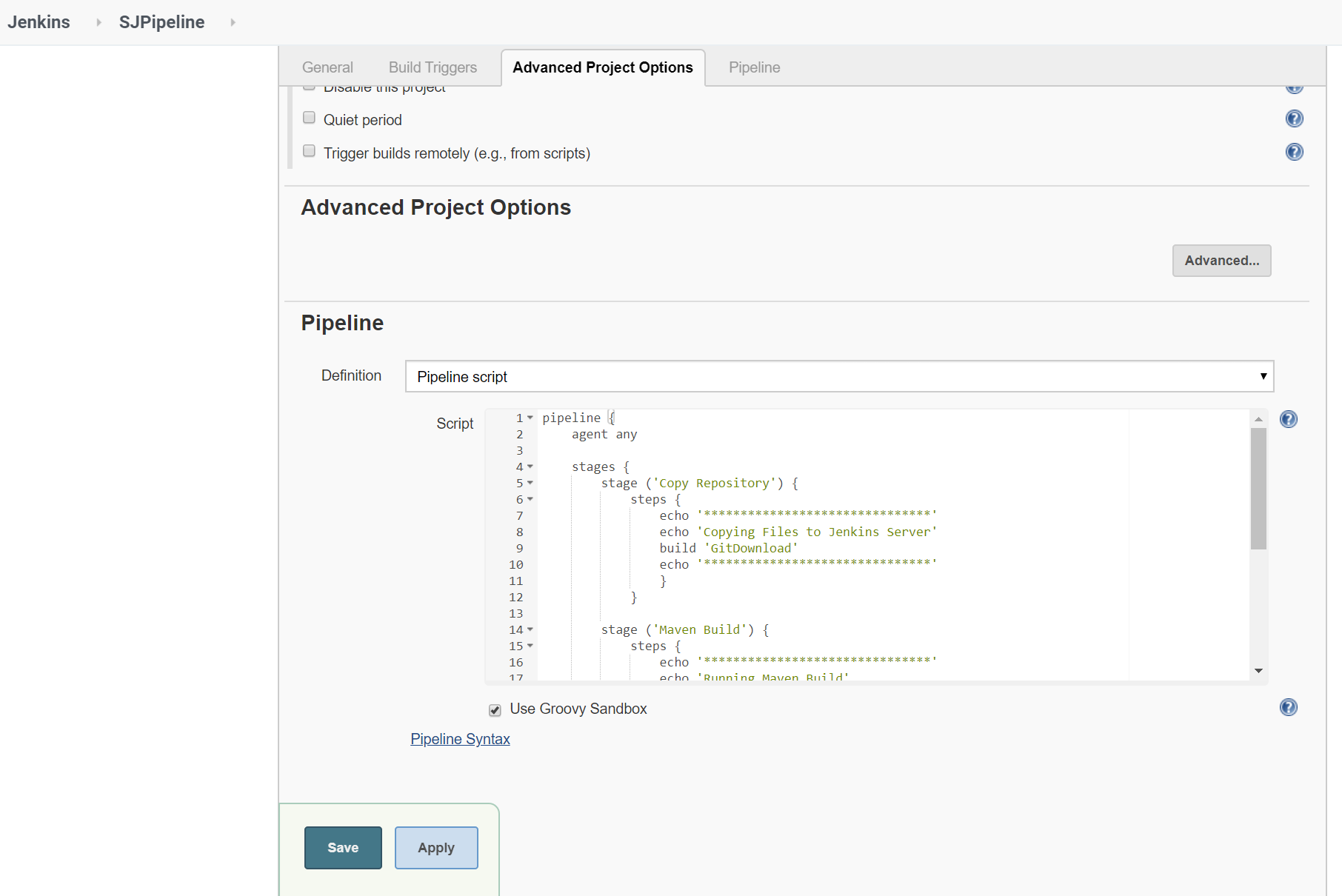
Then we must go back and create three separate Jenkins jobs:   
  
1) GitDownload: This job is used to download the SparkJava GitHub repository into the /var/lib/jenkins/workspace/GitDownload directory of JenkinsMaster.   
  
  
  
As seen above, the link to the GitHub repo was provided.

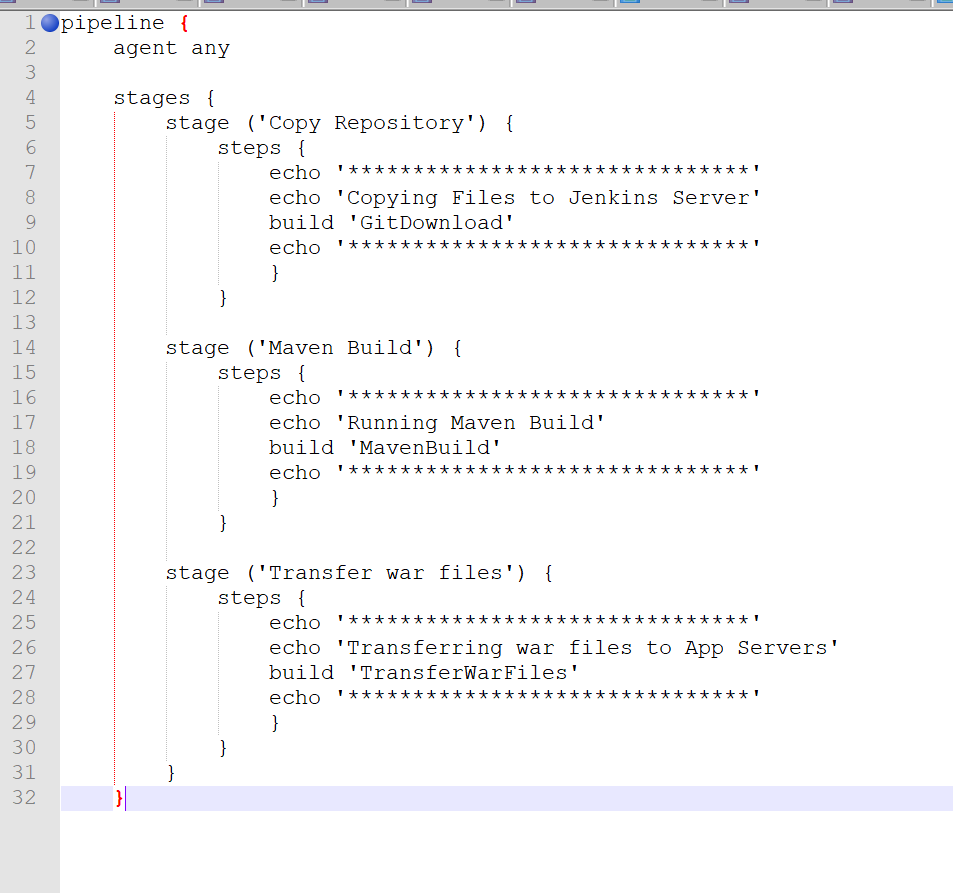
As seen below, the files were successfully downloaded into JenkinsMaster. The target directory is a result of running MavenBuild.

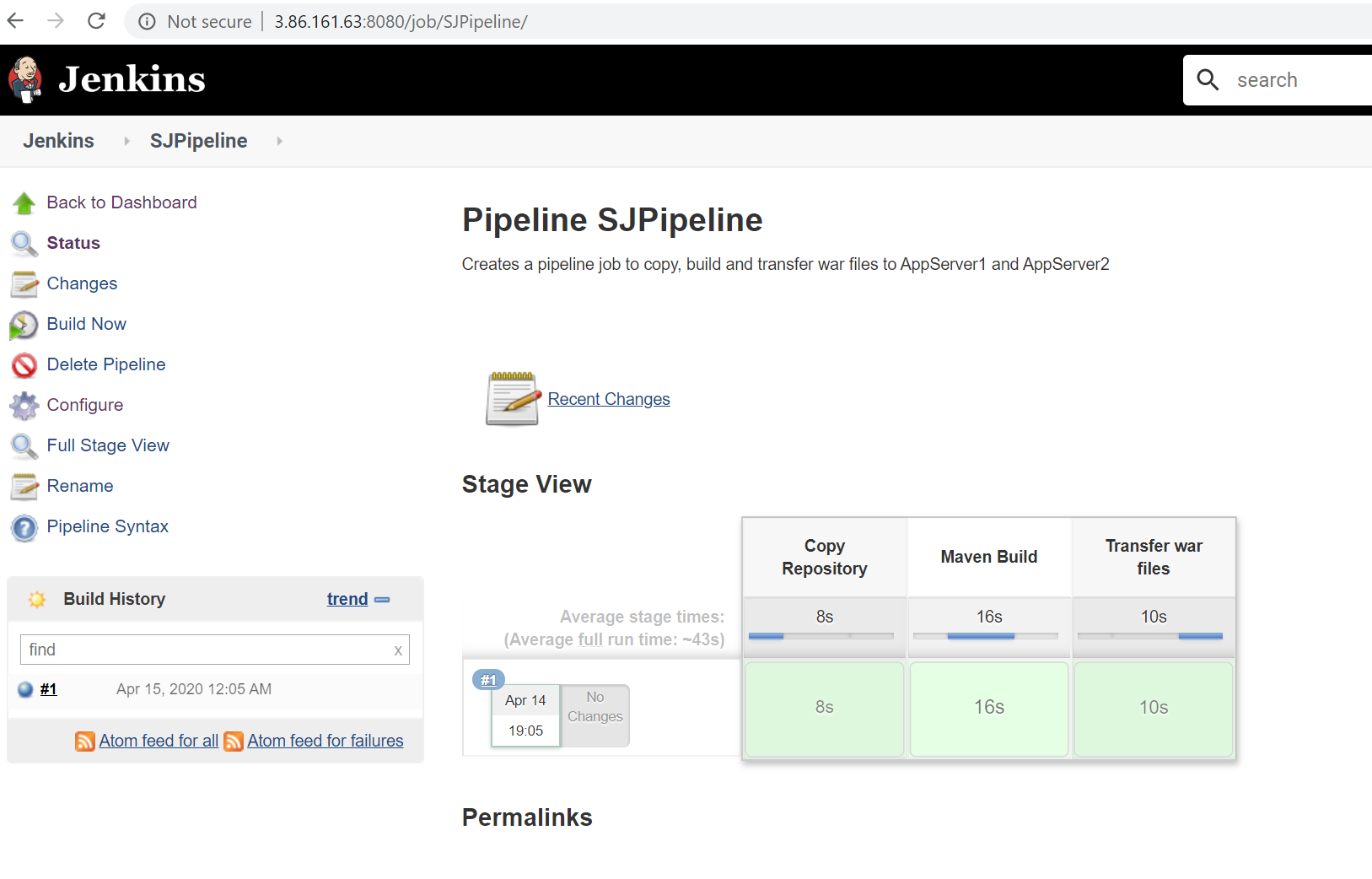
  
  
  
2) MavenBuild: MavenBuild was used to resolve dependencies and to get the sparkjava-hello-world-1.0.war which would be hosted in the Tomcat servers in AppServer1 and AppServer2.   
  
  
  
  
As shown above, MavenBuild’s configuration had a build step which executed shell commands. These commands would go to the GitDownload directory, list the contents of the directory for the console log and then executed mvn clean install which would make the target folder in which the sparkjava-hello-world-1.0.war file would be located.   
  
  
  
As shown above, the target folder which the sparkjava-hello-world-1.0.war would be made upon running mvn clean install in the build step.   
  
  
3) TransferWarFiles: This job was used to transfer the sparkjava-hello-world-1.0.war from JenkinsMaster to the /opt/apache-tomcat-9.0.34/webapps directory of AppServer1 and AppServer2. This job was also used to restart the Tomcat server on AppServer1 and AppServer2.   
  
Before this job was configured, the Plugin Manager was used to download the Publish over SSH plugin. This would be used to send build artifacts over SSH.   
After this step, Jenkins was configured with the RSA key, server name, server username and server hostname of AppServer1 and AppServer2.   
  


Mentioned below are commands to copy the war file into present working directory and then send files or execute commands over SSH configuration for the TransferWarFiles job.   
  
  
  
As shown above, the sparkjava-hello-world-1.0.war file from JenkinsPrime was being sent over SSH to AppServer1 and AppServer2. Exec commands were added to move the file from /home/ec2-user directories of both AppServer1 and AppServer2 to their webapps directory in Apache Tomcat to as to host java program.

4) SJPipeline: Finally, a pipeline job named SJPipeline was made to orchestrate the jobs: GitDownload, MavenBuild and TransferWarFiles.



The Groovy script shown above was used in the configuration of SJPipeline.  


Upon running the job, the following stage View was observed.   
  


Finally, you can type the IP address of the app server in the browser to access the web application.

