What is DevOps: Patrick Debois, who’s often called “the father of DevOps”, coined the word “DevOps” in 2009. As the word depicts, it was formed by combining two words: “development” and “operations”. DevOps is a collaborative way of developing and deploying software. DevOps (a portmanteau of development and operations) is a software development method that stresses communication, collaboration and integration between software developers and information technology (IT) operation professionals

Father of DevOps: Patrick Debois

• DevOps is an approach based on agile and lean principles in which business owners, development, operations, and quality assurance team collaborate to deliver software in a continuous stable manner

• DevOps is an environment that promotes cross practicality, shared business tasks and belief

• DevOps is a movement that improves IT service delivery agility

• DevOps is a culture that promotes better working relationship within the company

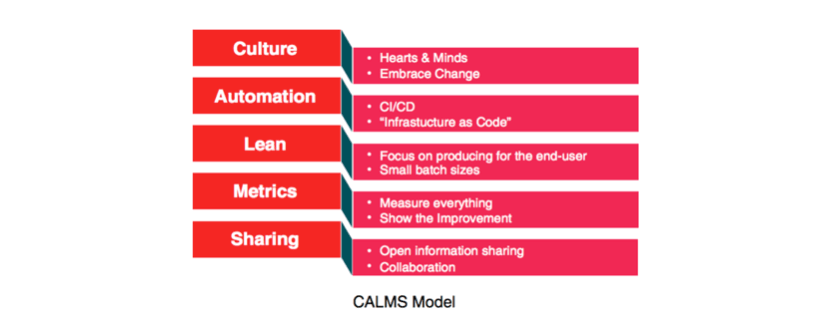
• DevOps is a set of practices that provides rapid, reliable software delivery

Do we really need DevOps: Developers always want changes as soon as possible and Operations want reliability and stability.

Lee Thomson describes this as a wall of confusion between development and operations. This wall of confusion not only exists between the mindsets of the two teams but also with the tools they use. Development uses some tools and operation uses some other tools to perform the same stuff.

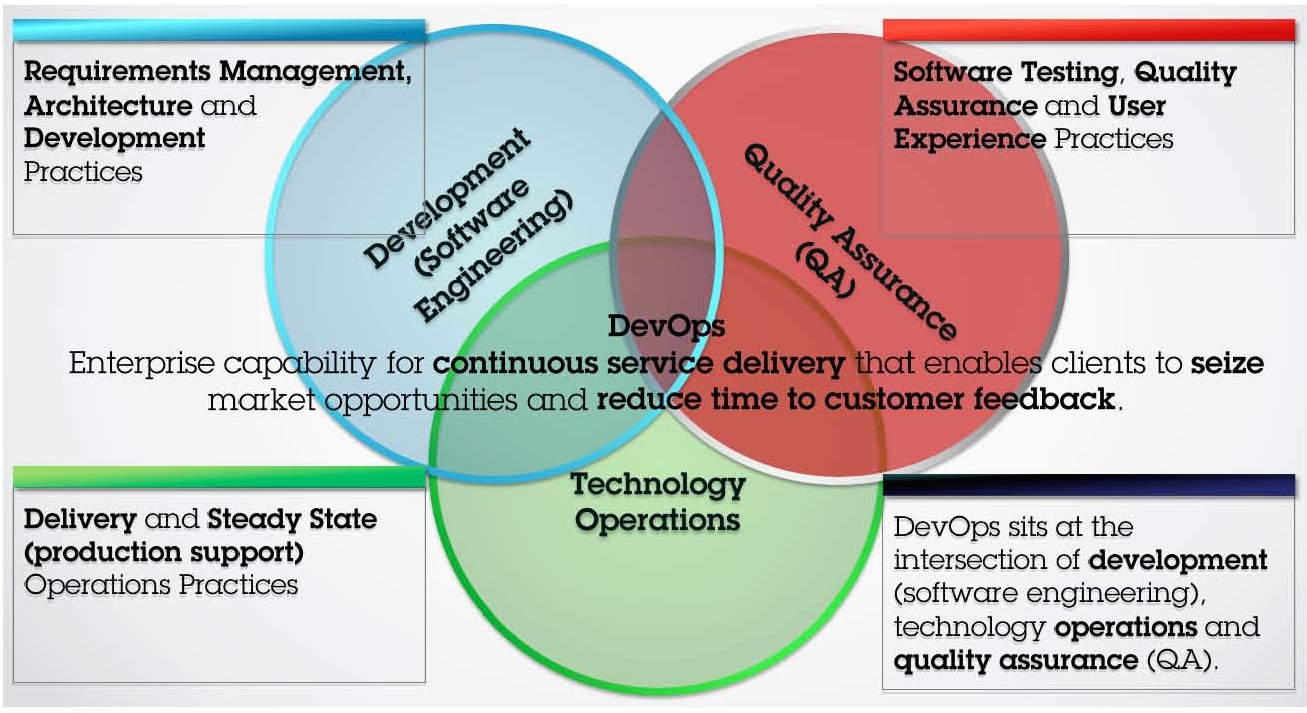
What drives the need for DevOps:

* The need for greater collaboration between development team and Operations teams
* A greater need for simultaneously deployment across different platforms.
* Pressures from the business to release apps more quickly to meet customer demand and enter new markets.
* Need to improve the end customer experience.
* The increasing need to develop or deploy cloud based applications.
* An increasingly complex IT infrastructure that is part physical, part virtualized, part cloud.
* Need to reduce IT costs.

DevOps is a way of thinking.

Five Basic Principles of DevOps:

* Eliminate the blame game, open post-mortems, Feedback, Rewarding Failures.
* Continuous delivery, Monitoring, Configuration management
* Business value for End user
* Performance Metrics(Monitoring tools like nagios), Logs, Business goal Metrics, People integration Metrics, KPI
* Ideas, Plans, Goals, Metrics, Compilations, Tools.



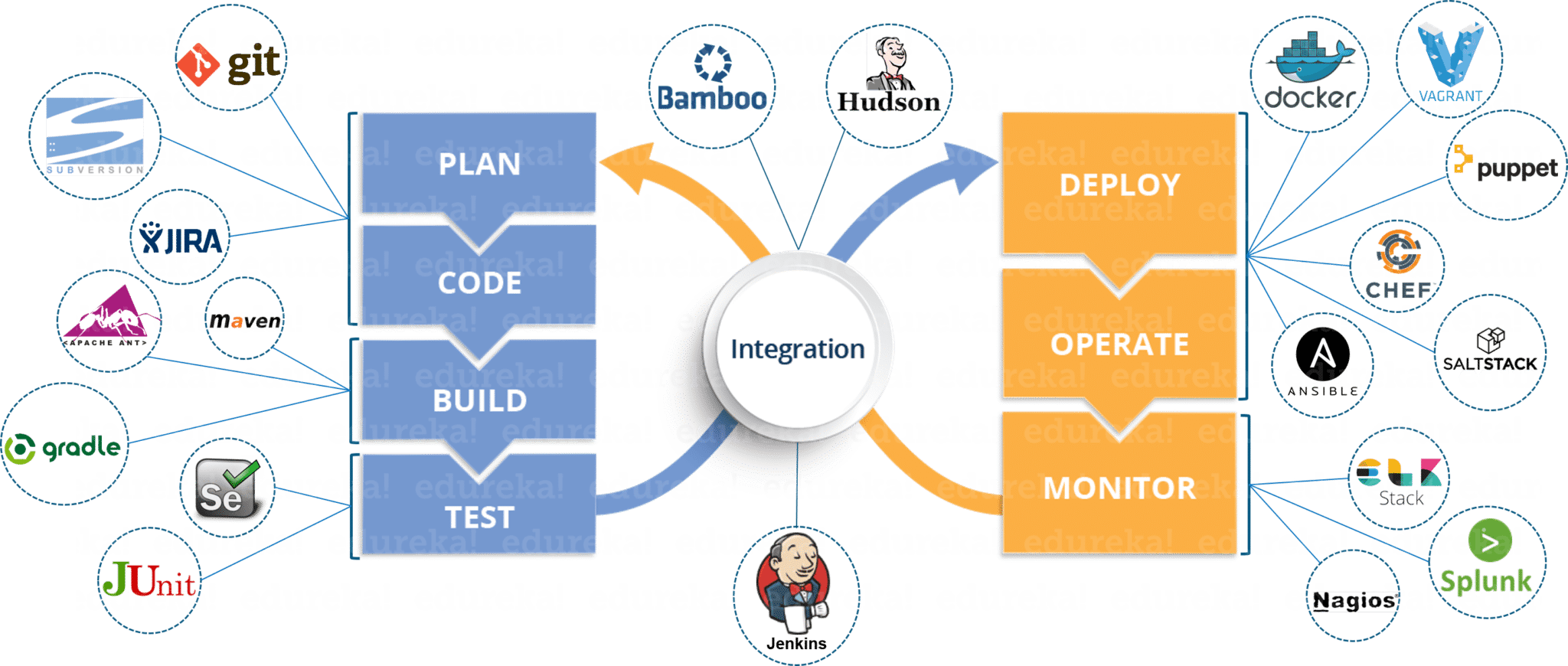
* DevOps combines the best of all the teams providing the following.
* Develops and verifies against production-like system.
* Reduces cost/time to deliver – Deploy often, deploy faster with repeatable, reliable process.
* Increase Quality – Automated testing, Reduce cost/time to test.
* Reduces Defect cycle time – Increase the ability to reproduce and fix defects.
* Increase Virtualized Environment utilization.
* Reduces deployment related downtime.
* Minimizes rollbacks

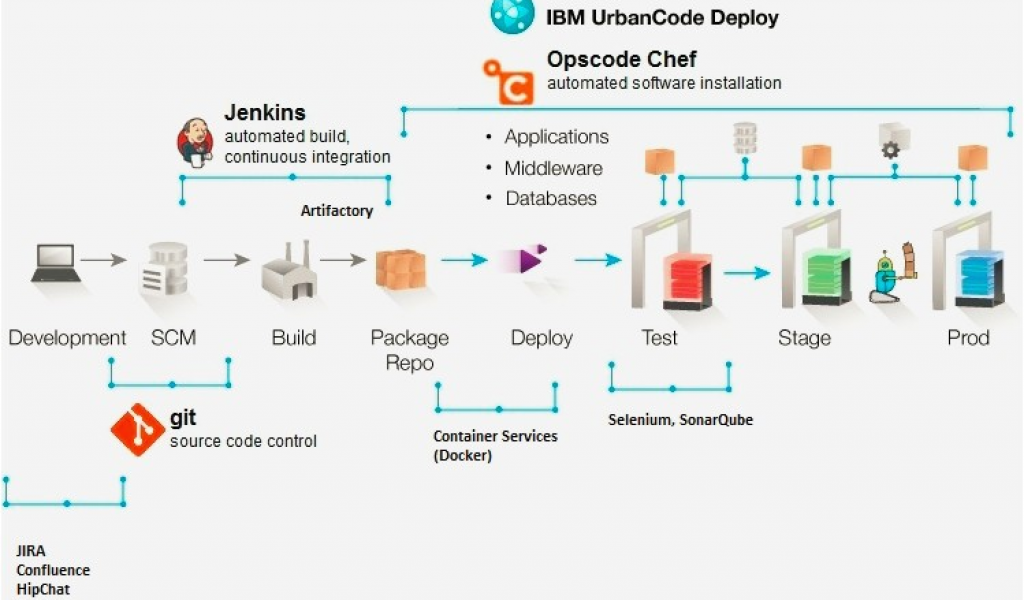


Waiter fall methodology: It’s a old process and time consuming process.

Agile Methodology: Most of the companies are following these process, here we will follow the process and Sprint (For particular time period) and will deliver the product on time with low cost/Safe/less bugs and risks.

DevOps Architecture:





Flow:

Developer will starts writing the code which they have stored in “SOURCE CODE MANAGENT” tool like ‘GIT/SVN’, all these tools called as a version control tools.

They used to compile the code by using build tools like “MAVEN”. Once the compiling the source code it will generates the packages like “WAR/JAR/EAR(Enterprise Architecture)”.

Then deploy the packages to the “**Docker**” containers.

We can automated the above tools like “SVN/GIT/MAVEN” by using continuous integration process like ‘JENKINS’.

We also automated the test environment by using the testing tools like ‘SELENIUM/JUNIT’ test cases.

Also we have to configured the code quality control tool like ‘SONARQUBE’ and we can packaging our application.

After creating a war file or packages we used to deploy to different environments like ‘DEV/STAGE/UAT(User acceptance test)/PROD’.

If you have different environments like above then by using cloud infrastructure like ‘AWS/AZURE/GOOGLE CLOUD’.

We used to install packages or services or softwares, for this we need to use automated tools i.e configuration management tools like “CHEFF/PUPPET/ANSIBLE” and also we can use these for deployment purpose.

“OPSCPODE TOOLS” is “CHEFF/PUPPET/ANSIBLE”

Some times we need to integrate database services also.

Usually we have to deploy packages like “WAR/JAR/EAR” In the application servers like “TOMCAT/JBOSS/WEBLOGIC”.

We need to track our bug fixes by using the tools like “JIIRA/BUGZILLA”.

Linux commands:

* Create new user: useradd username
* Assign Passwd: sudo passwd userpassword

User Permissions:

Go to /etc and do vim sudoers and change the permissions.

* root ALL=(ALL:ALL) ALL : All permissions
* siva ALL=(r:r) ALL : Only read permissions
* useradd sudo username : command to add the user to the sudo group
* su – username : command to switch to the new user account.
* sudo command\_to\_run :  verify that you can use sudo by prepending
* sudo ls -la /root : you can list the contents of the /root directory
* uname –a : gives you the name of your machine, the kernel version, the processor architecture
* uname –r : gives machine name
* ifcofig : IP address details
* cd ~ : to go home
* mv : to rename or move
* ls –ltr, ls -lh : list of files along with permissions and memory
* ls –R : list of files in recursive order
* free –h, free –g : memory details [Also cat/proc/meminfo or cat/proc/cpuinfo]
* df –h, du –h, du –k : Disk usage details
* tar -zcvf tar-archive-name.tar.gz source file : This will compress the contents of source-folder-name to a tar.gz archive named tar-archive-name.tar.gz
* tar zxvf : Untar
* wc –w filename, wc –l, wc –c: word count and lines and characters count in a file
* cd %userprofile%/.ssh : Check for the ssh keys
* ssh-keygen -t rsa -C siva27793@gmail.com : Create new ssh keys
* nslookup followed by the **domain name :** check the DNS name
* unit –o to unit -6 : Run l evels. Unit -0 for shutdown and Unit-6 for Reboot.
* head -4 : to get top 4 lines, it is same for tail(tail -4) to get bottom 4 lines
* kill alljava : It will kill all the running java files
* shutdown.sh & : & is for to run the process in background

Soft and Hard links in Unix/Linux:

A link in UNIX is a pointer to a file. Like pointers in any programming languages, links in UNIX are pointers pointing to a file or a directory. Creating links is a kind of shortcuts to access a file. Links allow more than one file name to refer to the same file, elsewhere.

There are two types of links :

1. Soft Link or Symbolic links
2. Hard Links

These links behave differently when the source of the link (what is being linked to) is moved or removed. Symbolic links are not updated (they merely contain a string which is the pathname of its target); hard links always refer to the source, even if moved or removed.

For example, if we have a file a.txt. If we create a hard link to the file and then delete the file, we can still access the file using hard link. But if we create a soft link of the file and then delete the file, we can’t access the file through soft link and soft link becomes dangling. Basically hard link increases reference count of a location while soft links work as a shortcut (like in Windows).

**1. Hard Links:**

* Each hard linked file is assigned the same Inode value as the original, therefore they reference the same physical file location. Hard links more flexible and remain linked even if the original or linked files are moved throughout the file system, although hard links are unable to cross different file systems.
* ls -l command shows all the links with the link column shows number of links.
* Links have actual file contents
* Removing any link, just reduces the link count, but doesn’t affect other links.
* We cannot create a hard link for a directory to avoid recursive loops.
* If original file is removed then the link will still show the content of the file.
* **ln [original filename] [link name] : Command to create a hardlink.**
* **ls –i : to know the inode value**
* **find . -type f -exec ls -s {} \; | sort -n -r | head -1 : To find the largest file in the current directory**
* **grep -v "^$" file.txt : To remove the blank lines using the grep command.**

**Softlink:**

* A soft link is similar to the file shortcut feature which is used in Windows Operating systems. Each soft linked file contains a separate Inode value that points to the original file. As similar to hard links, any changes to the data in either file is reflected in the other. Soft links can be linked across different file systems, although if the original file is deleted or moved, the soft linked file will not work correctly (called hanging link).
* ls -l command shows all links with first column value 1? and the link points to original file.
* Soft Link contains the path for original file and not the contents.
* Removing soft link doesn’t affect anything but removing original file, the link becomes “dangling” link which points to nonexistent file.
* A soft link can link to a directory.
* Link across filesystems: If you want to link files across the filesystems, you can only use symlinks/soft links.
* **ln -s [original filename] [link name] : Command to create a softlink**

**Sort:**

* **sort filename :** When we have a mix file with both uppercase and lowercase letters then first the lower case letters would be sorted following with the upper case letters .
* **sort -nr filename.txt :** To sort a file with**numeric data in reverse order** we can use the combination of two options as stated below.
* **finger :** To know the logged in user details

**Crontab:** The crontab (short for "cron table") is a list of commands that are scheduled to run at regular time intervals on your computer system. The crontab command opens the crontab for editing, and lets you add, remove, or modify scheduled tasks.

The daemon which reads the crontab and executes the commands at the right time is called cron. It's named after Kronos, the Greek god of time.

Default crontab details located on ‘/var/spool/cron’

* **crontab [-u user] file or crontab file**
* **crontab [-u user] [-l | -r | -e] [-i] [-s]**

**SED:**

SED is a powerful text stream editor. Can do insertion, deletion, search and replace(substitution).

SED command in unix supports regular expression which allows it perform complex pattern matching.

**$CAT > GEEKFILE.TXT**

unix is great os. unix is opensource. unix is free os.

learn operating system.

unix linux which one you choose.

unix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

1.REPLACING OR SUBSTITUTING STRING :

**$SED 'S/UNIX/LINUX/' GEEKFILE.TXT**

Output: linux is great os. unix is opensource. unix is free os.

learn operating system.

linux linux which one you choose.

linux is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.

2. REPLACING THE NTH OCCURRENCE OF A PATTERN IN A LINE : Use the /1, /2 etc flags to replace the first, second occurrence of a pattern in a line. The below command replaces the second occurrence of the word “unix” with “linux” in a line.

**$SED 'S/UNIX/LINUX/2' GEEKFILE.TXT**

**$SED 'S/UNIX/LINUX/G' GEEKFILE.TXT : G is for to replace all occurances.**

Permissions to access the files:

Chmod 755 filename

Here 7 is userpermisson, 5 is group permission and another 5 is others.

For read=4, write=2 and execute=1, i.e 4+2+1=7, so if you give 7 then the file has complete permissions

SVN:

Step1: Install apache2 or httpd

sudo apt-get update

sudo apt-get install apache2

Step2: Install SVN Server

sudo apt-get install subversion libapache2-mod-svn libapache2-svn libsvn-dev

Step3: After Installation, need to enable below Apache modules and then restart the apache service.

sudo a2enmod dav

sudo a2enmod dav\_svn

Restarting Apache service: sudo service apache2 restart

Step4: Cofiguring Apache with Subversion.

Subversion Apache module package creates an configuration file /etc/apache2/mods-enabled/dav\_svn.conf. You just need to make necessary changes to it.

Alias /svn /var/lib/svn

<Location /svn>

DAV svn

SVNParentPath /var/lib/svn

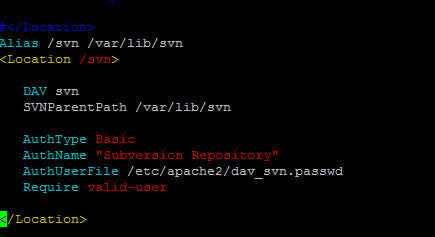
AuthType Basic

AuthName "Subversion Repository"

AuthUserFile /etc/apache2/dav\_svn.passwd

Require valid-user

</Location>

/

Step 5: Create First SVN Repository:

sudo mkdir -p /var/lib/svn/

sudo svnadmin create /var/lib/svn/myrepo

sudo chown -R www-data:www-data /var/lib/svn

sudo chmod -R 775 /var/lib/svn

Step 6: Create Users for Subversion:

Now create first svn user in /etc/apache2/dav\_svn.passwd file. These users will use for authentication of svn repositories for checkout, commit processes.

sudo htpasswd -cm /etc/apache2/dav\_svn.passwd admin

Then it will prompts for the password.

SVN Credentials:

User: admin

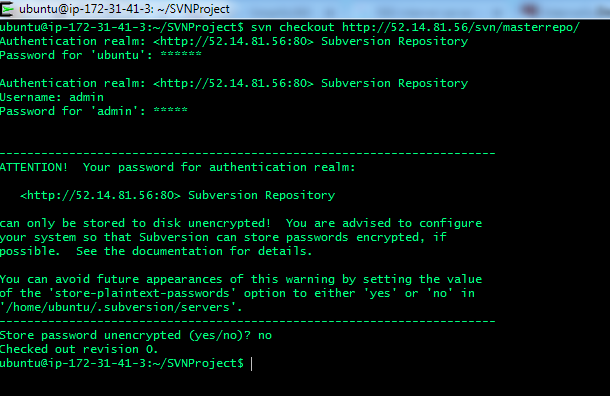
Passwd: admin@123

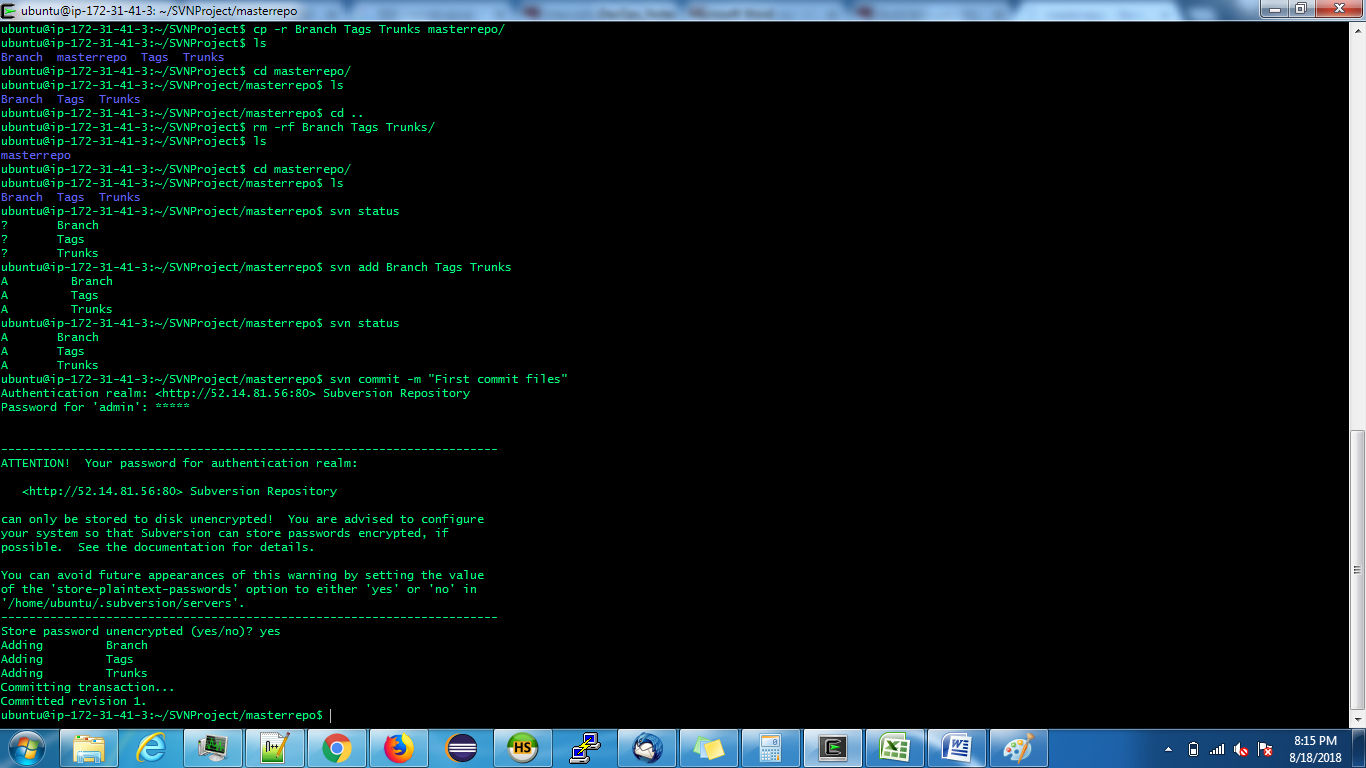
To add additional users:

sudo htpasswd -m /etc/apache2/dav\_svn.passwd user1

sudo htpasswd -m /etc/apache2/dav\_svn.passwd user2

Once we checkout the screen is





Branching Stratagies:

* 1. Stable trunk : Fllow wil be Branch=>Tags=>Trunks
  2. Un-stable trunk : Fllow will be Trunks=>Tags=>Branch

SVN Common Commands:

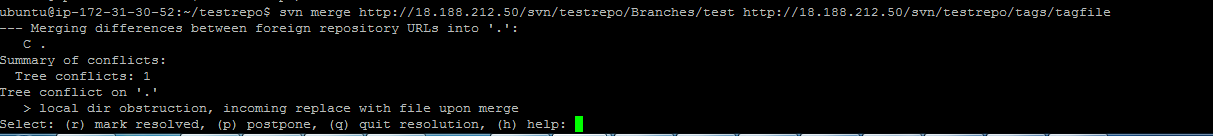
svn checkout=svn export

svn commit=svn import

SVN Hooks: Here we have to give the conditional statements

Merging: Here we get two types of conflicts.

1. Tree conflict: It will come while merging between two branches.



1. File conflict: It will come when two developers works on same directory, same file and same line.

GIT:

Difference between GIT and SVN,

 Git is a distributed VCS; SVN is a non-distributed VCS.

 Git has a centralized server and repository; SVN does not have a centralized server or repository.

 The content in Git is stored as metadata; SVN stores files of content.

 Git branches are easier to work with than SVN branches.

 Git does not have the global revision number feature like SVN has.

 Git has better content protection than SVN.

 Git was developed for Linux kernel by Linus Torvalds; SVN was developed by CollabNet, Inc.

 Git is distributed under GNU, and its maintenance overseen by Junio Hamano; Apache Subversion, or SVN, is distributed under the open source license.

* Git has staging where SVN don’t have

Read more: [Difference Between Git and SVN | Difference Between](http://www.differencebetween.net/technology/software-technology/difference-between-git-and-svn/#ixzz5OniefUAV) <http://www.differencebetween.net/technology/software-technology/difference-between-git-and-svn/#ixzz5OniefUAV>

Git commands:

git clone <https://github.com/Sivakumar27793/Master_siva.git>

git config –list: To display user details list

To cofig user details.

git config --global user.name "Sivakumar27793" Password: Repo@482

git config --global user.email [sivakumar93277@gmail.com](mailto:sivakumar93277@gmail.com)

To create a new branch:

$ git branch <feature\_branch>

$ git checkout <feature\_branch>

$ git add .   
$ git commit -m "adding a change from the feature branch"

$ git checkout master

$ git push origin <feature\_branch>

Git merge:

Git stash: To save the files in your working directory

Git stash pop: Popping your stash removes the changes from your stash and reapplies them to your working copy.

Git stash apply: To apply the same stashed changes to multiple branches.

MAVEN:

First we need to create a empty repostarty by using

Mkdir mvn\_repo

Then create a maven project by using

Mvn archetype:generate

Then move to pom.xml folder and do the maven phases

Mvn clean package

Then come to .git folder and add commit and push the code then it will go to the git.

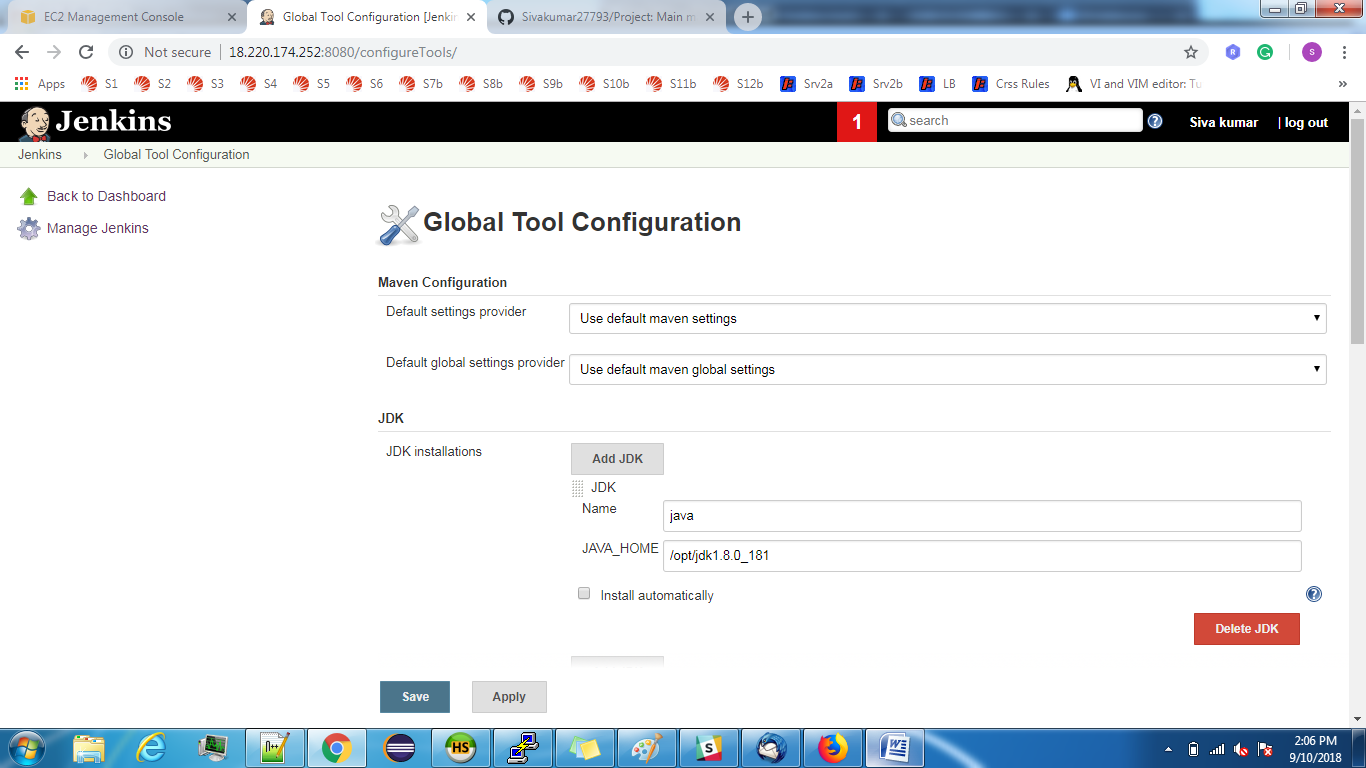
JAVA:

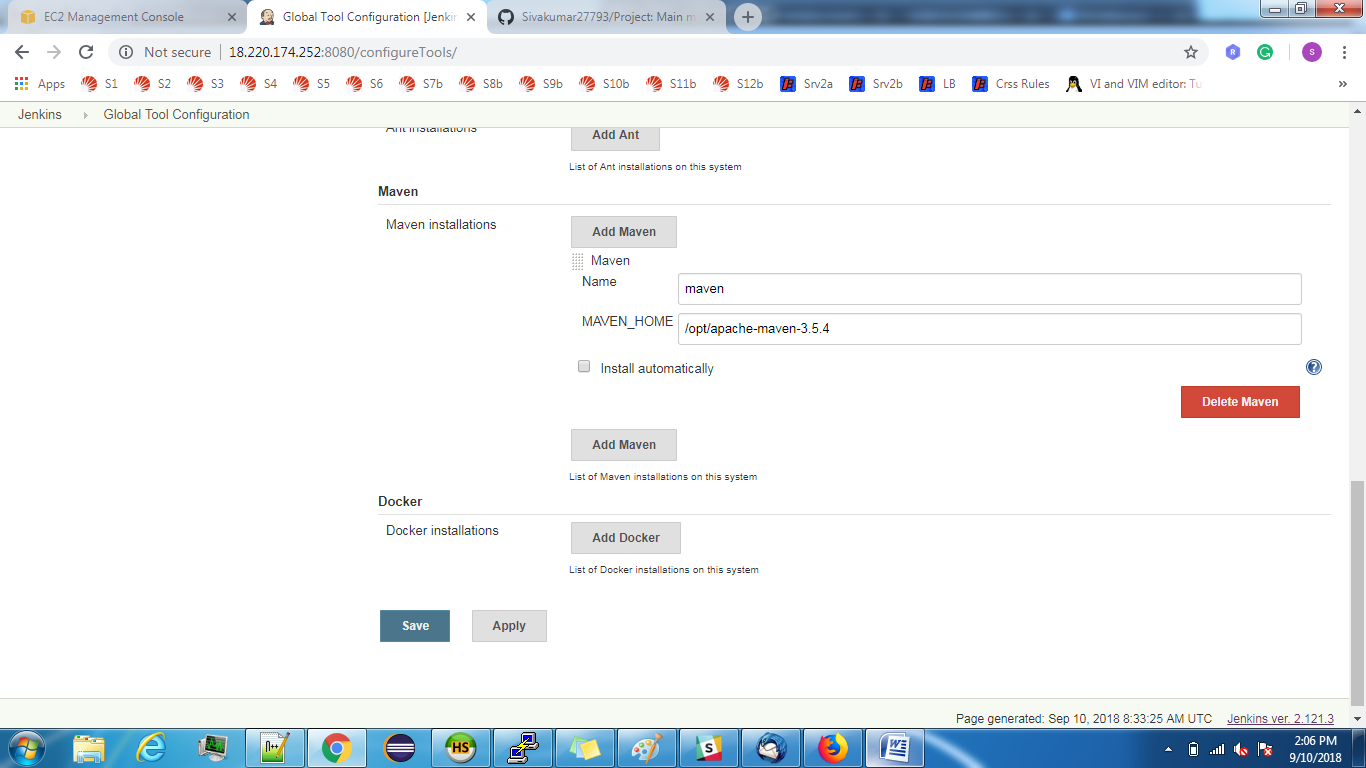
wget --continue --no-check-certificate -O jdk-8u191-linux-x64.tar.gz --header "Cookie: oraclelicense=a" <http://download.oracle.com/otn-pub/java/jdk/8u191-b12/2787e4a523244c269598db4e85c51e0c/jdk-8u191-linux-x64.tar.gz>

Jenkins: Continuous Integration tool or Build automation tool

Steps to set a path for JAVA and MAVEN

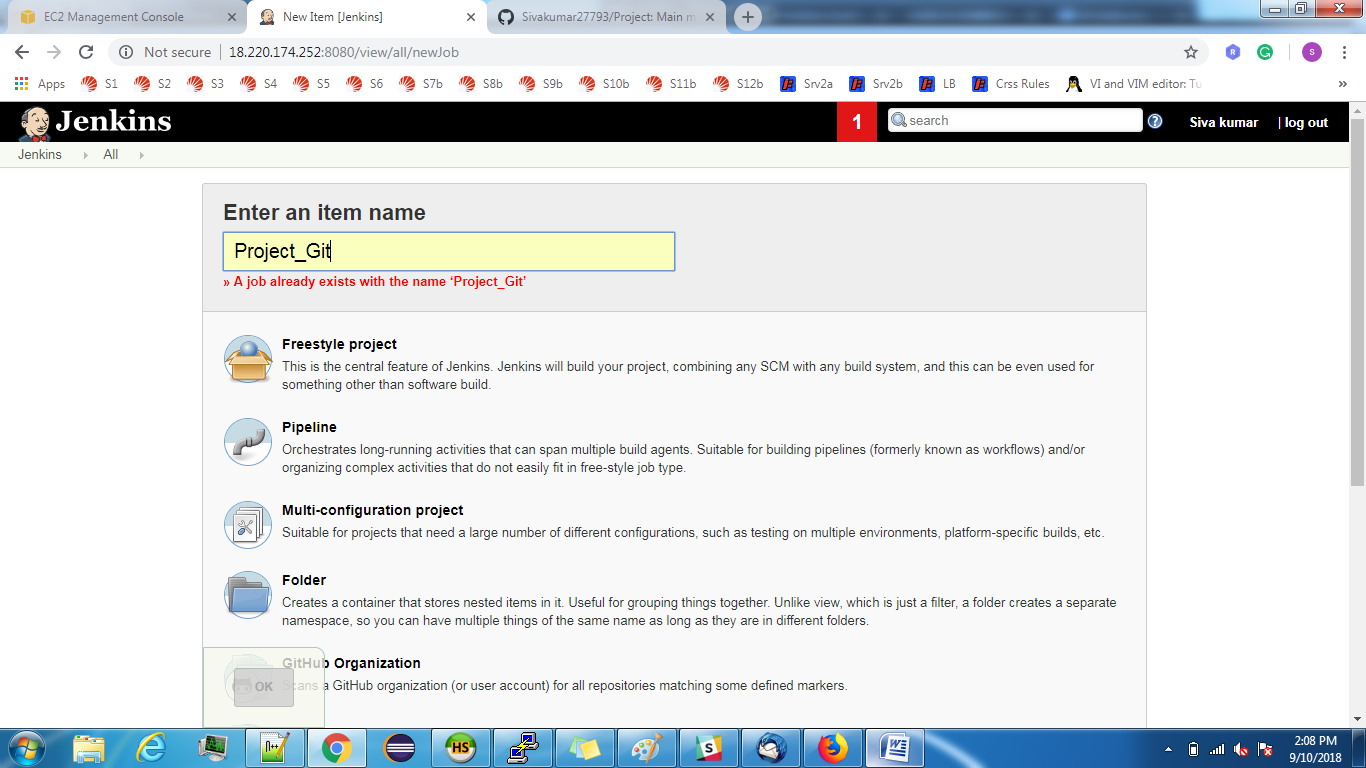
Manage Jenkins=>Global tool configuration





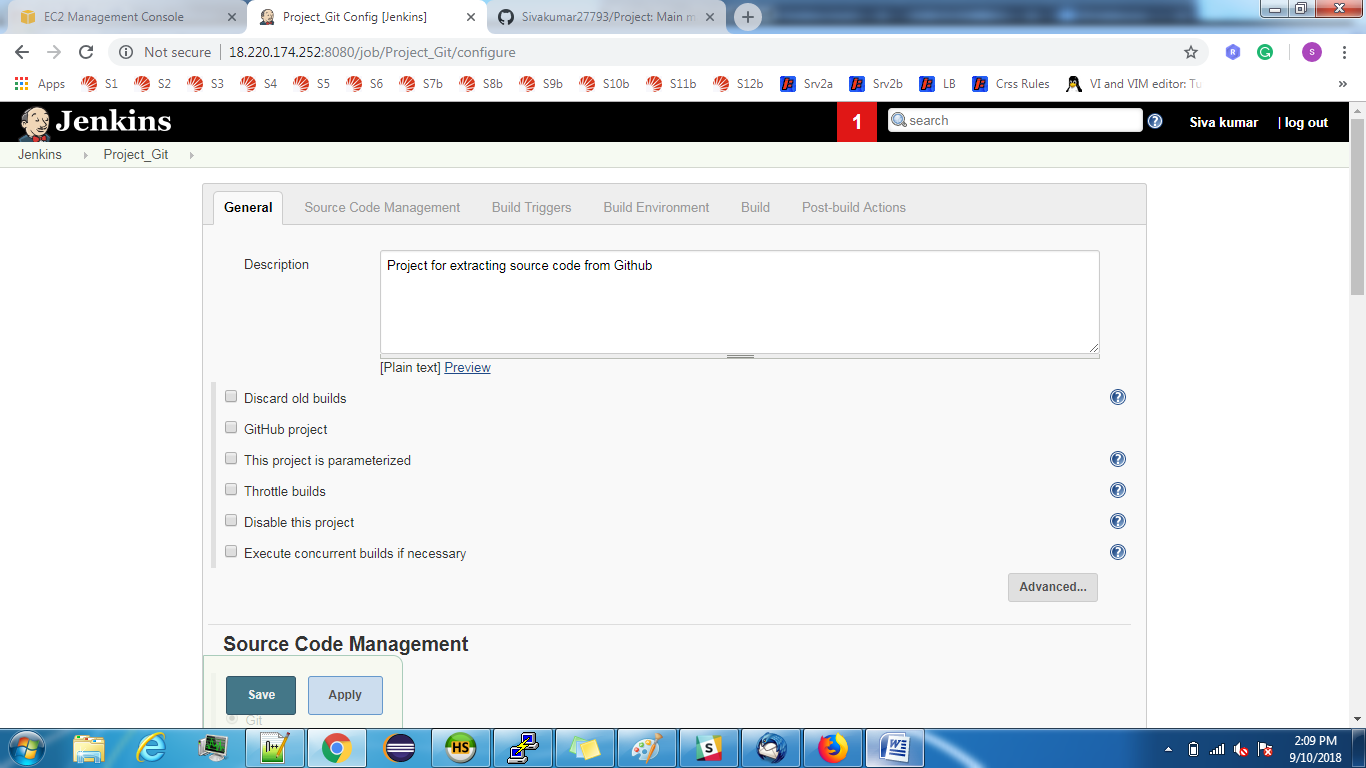
Steps to create a project:

Login into Jenkins and **Create a New job** or select **New Item**.

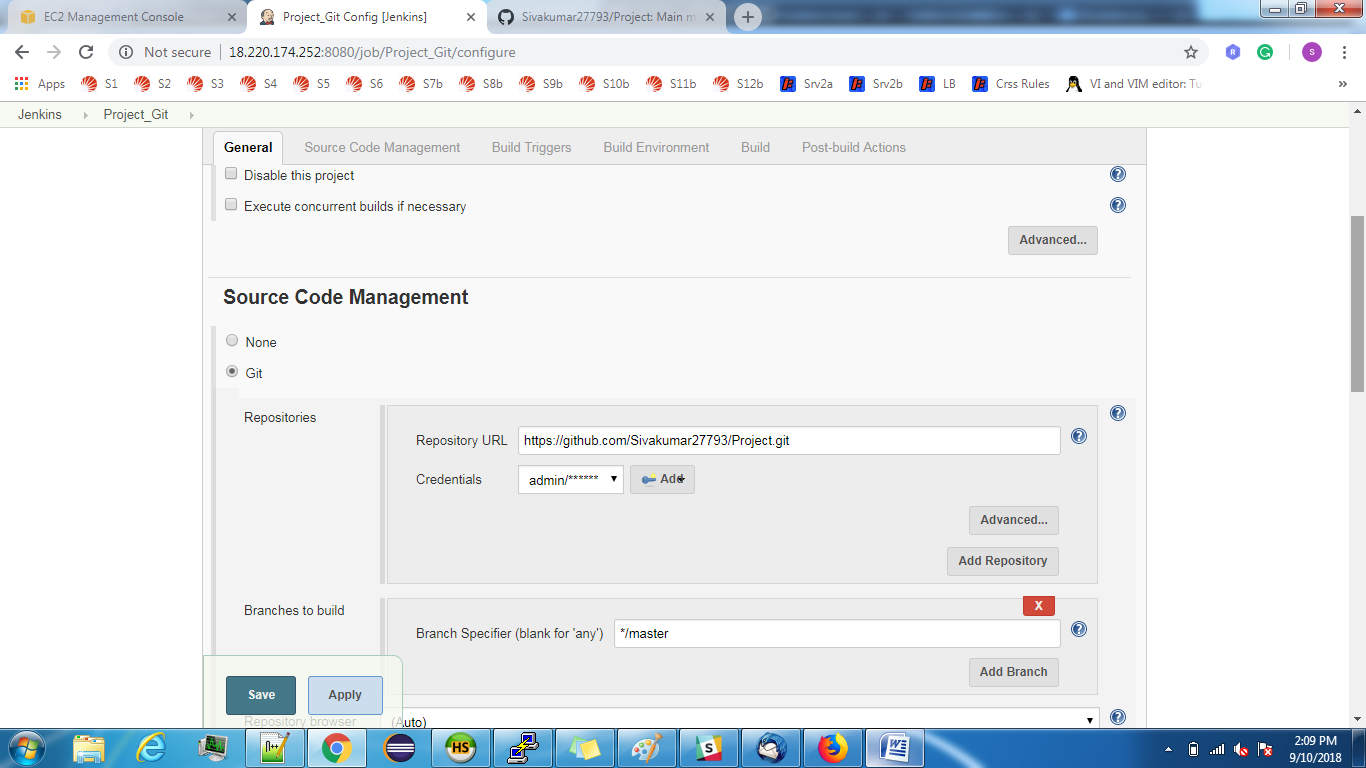


And Select Free style project which will support for all kinds of source codes.

General Information about Project.



SCM Management:

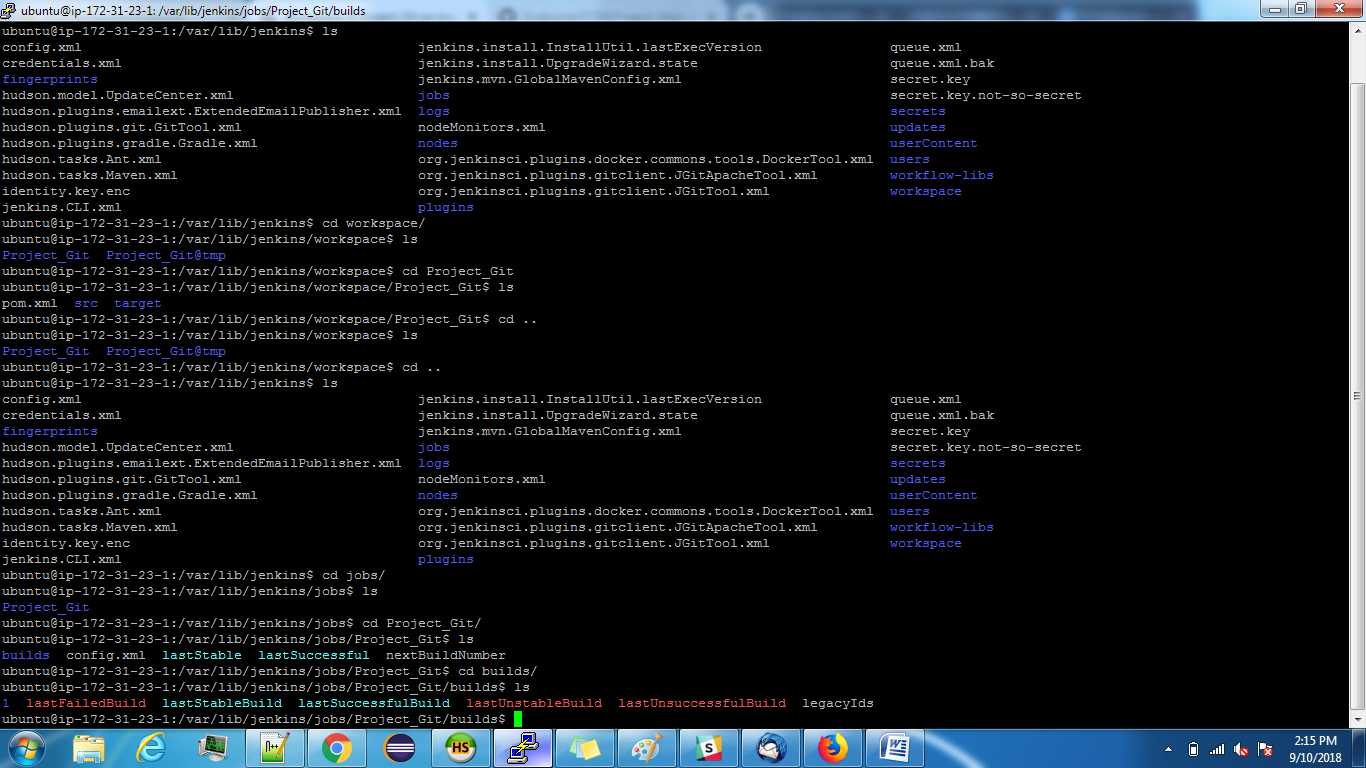


And then come to project home page and click on Build now to build and you can see the console out put.



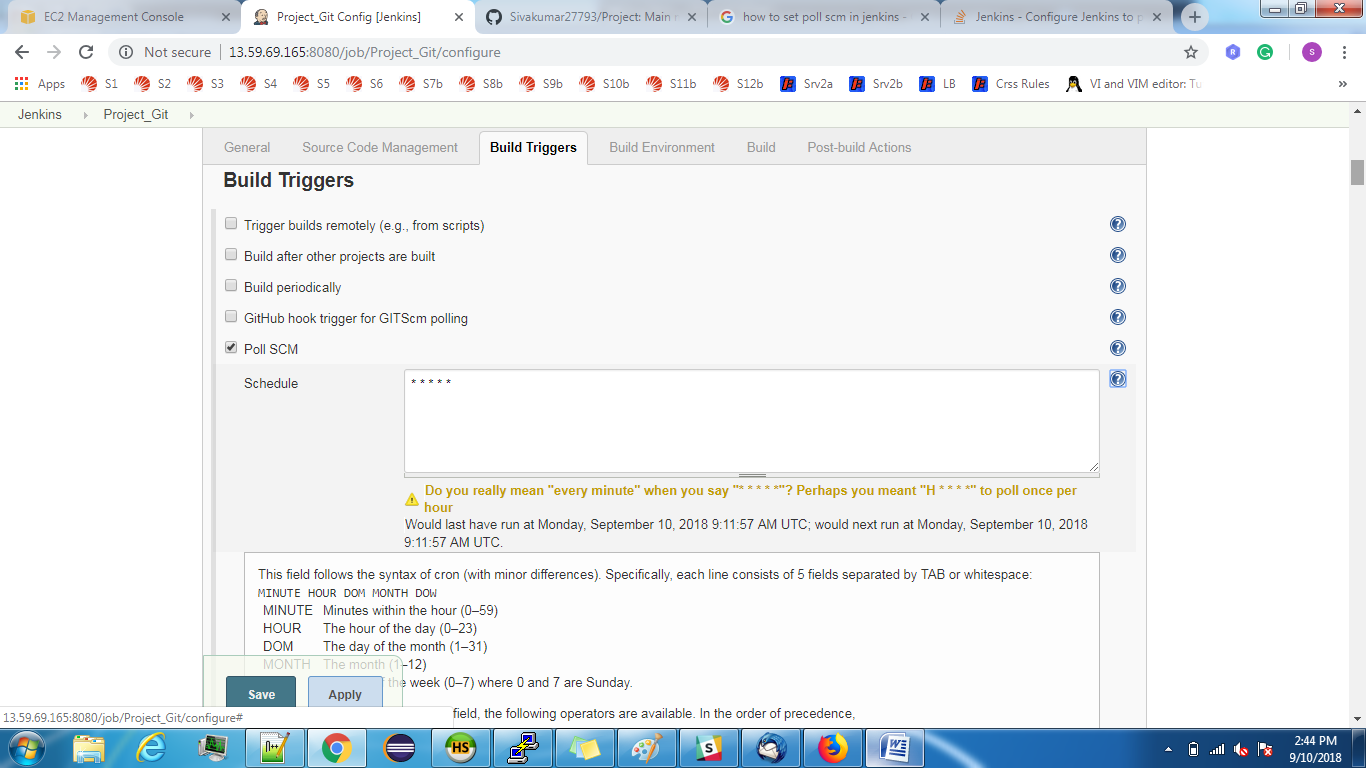
And then come Project **workspace** will be created and under that you can notice **Target** folder.

In AWS server, we can find these details in /var/lib/Jenkins

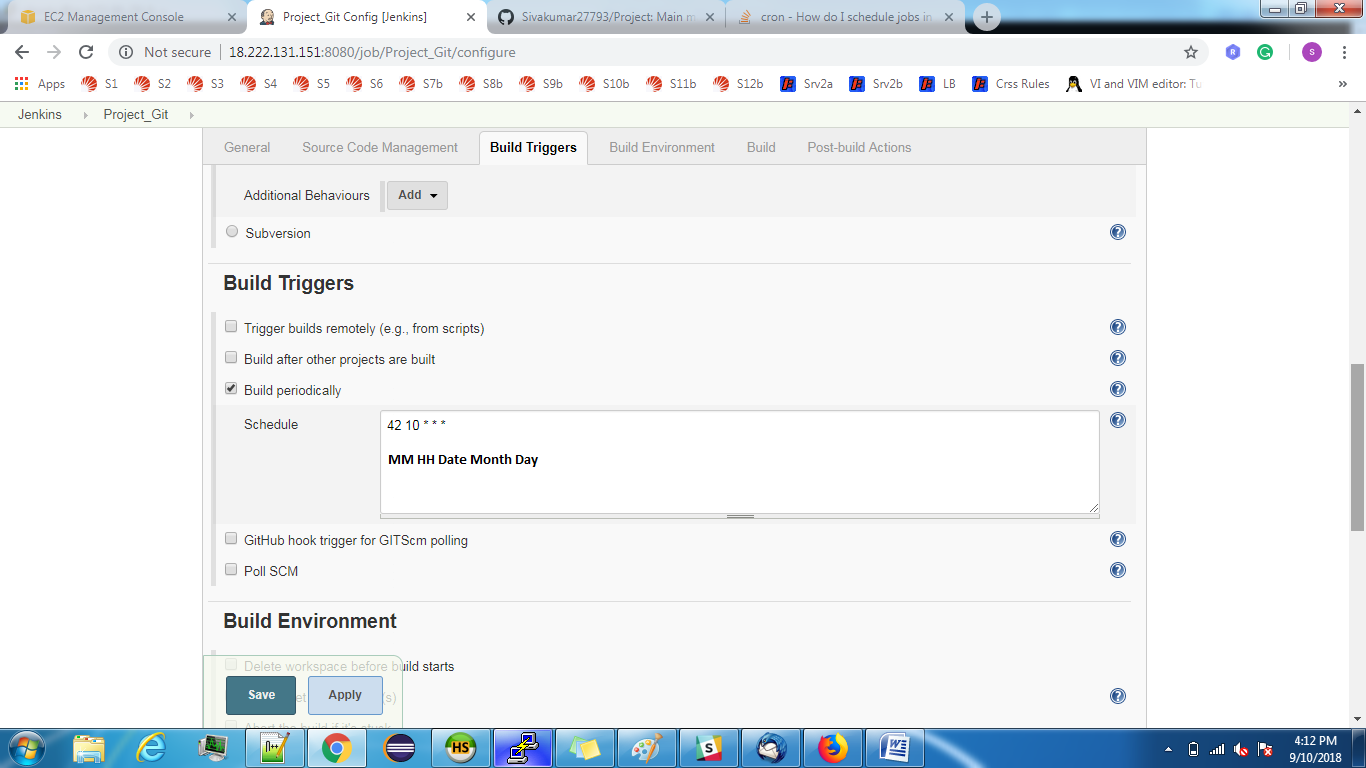


Build Triggers:

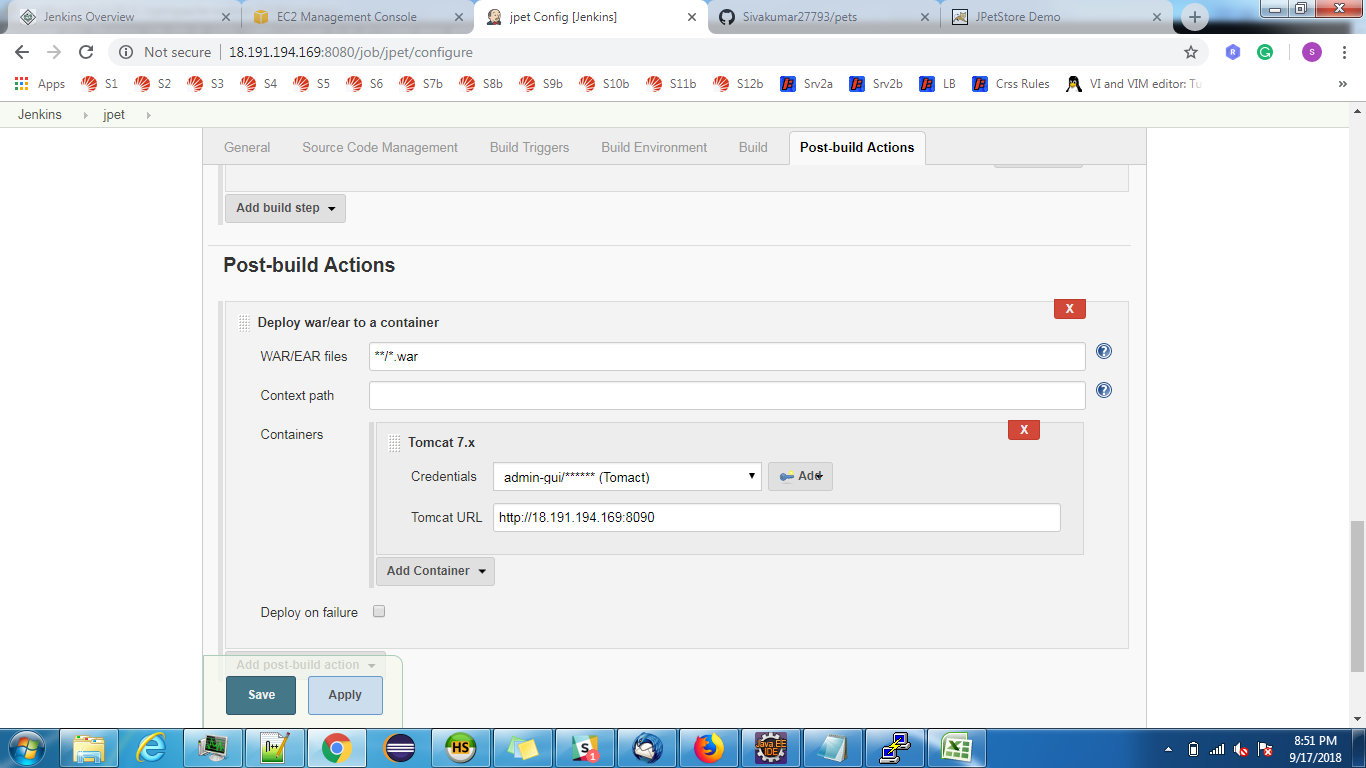
* 1. Poll SCM: If we select this option it will trigger a build when ever the changes happens on Git.



* 1. Build periodically: If you want to schedule a build for certain time we need to do this.

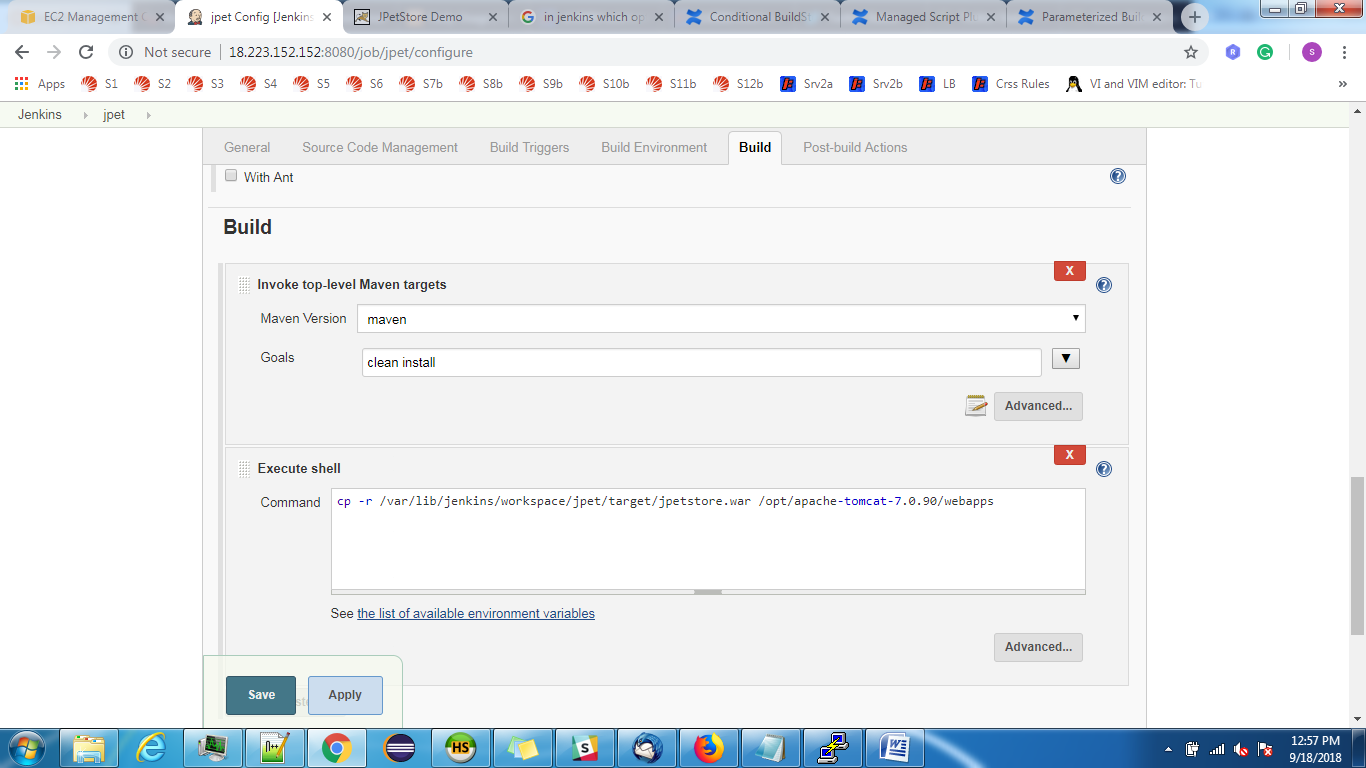


To deploy automatically.



To deploy War file using script:

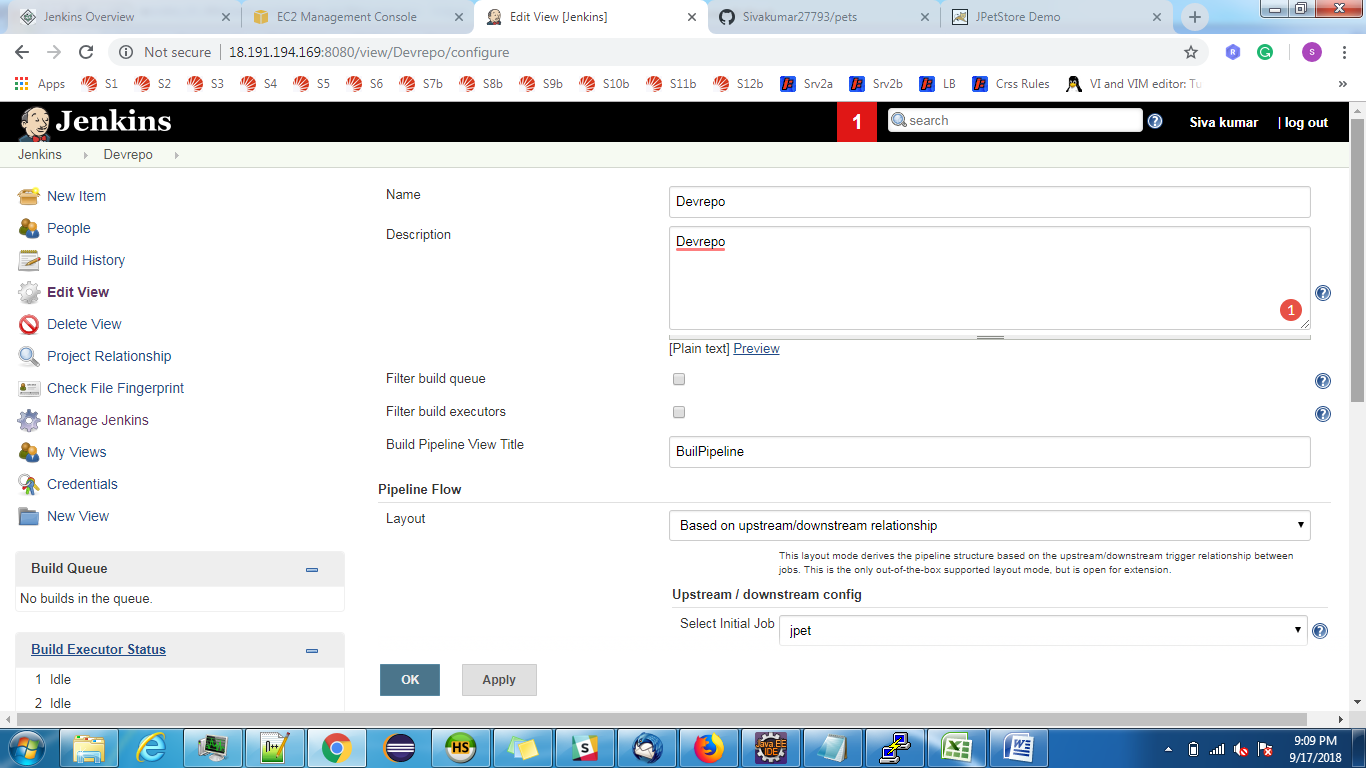
For this you have to select “Excecute shell” option in “Add build step”.

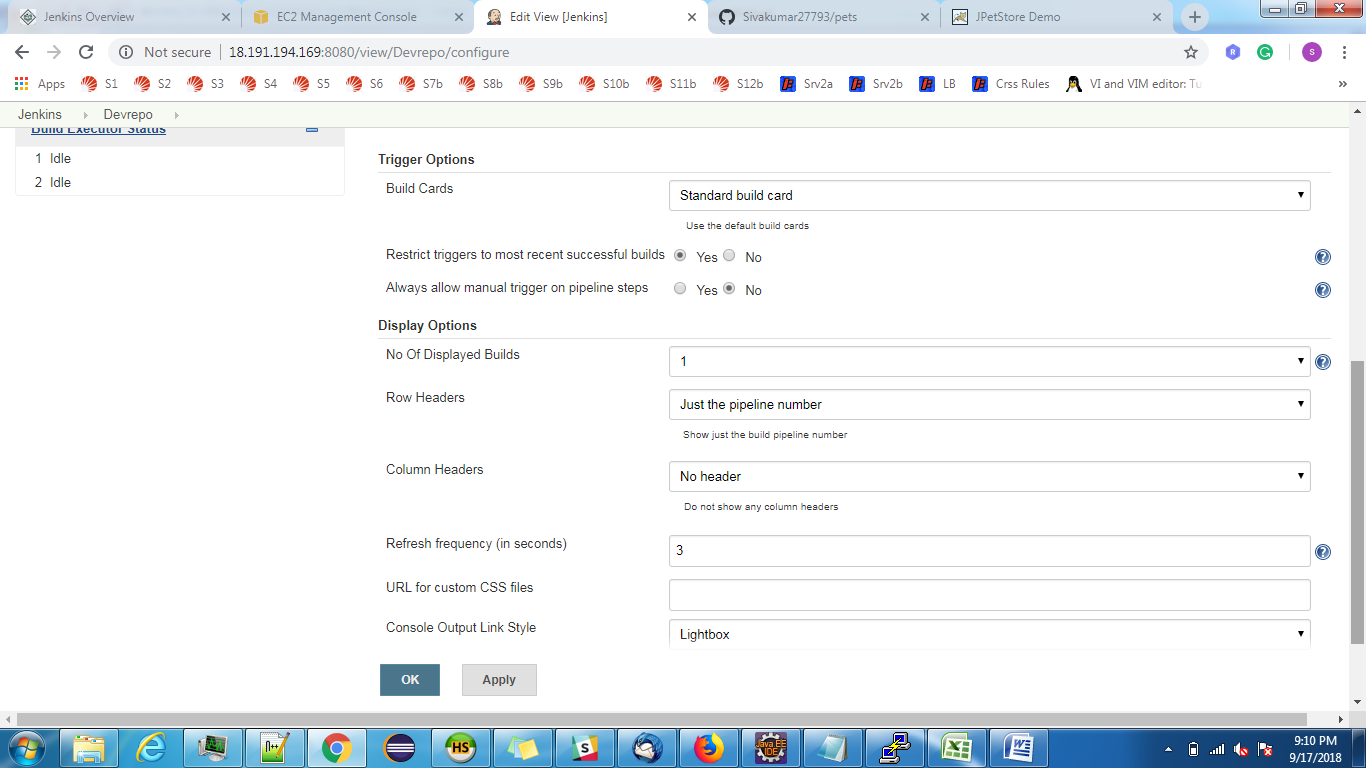


Pipeline:

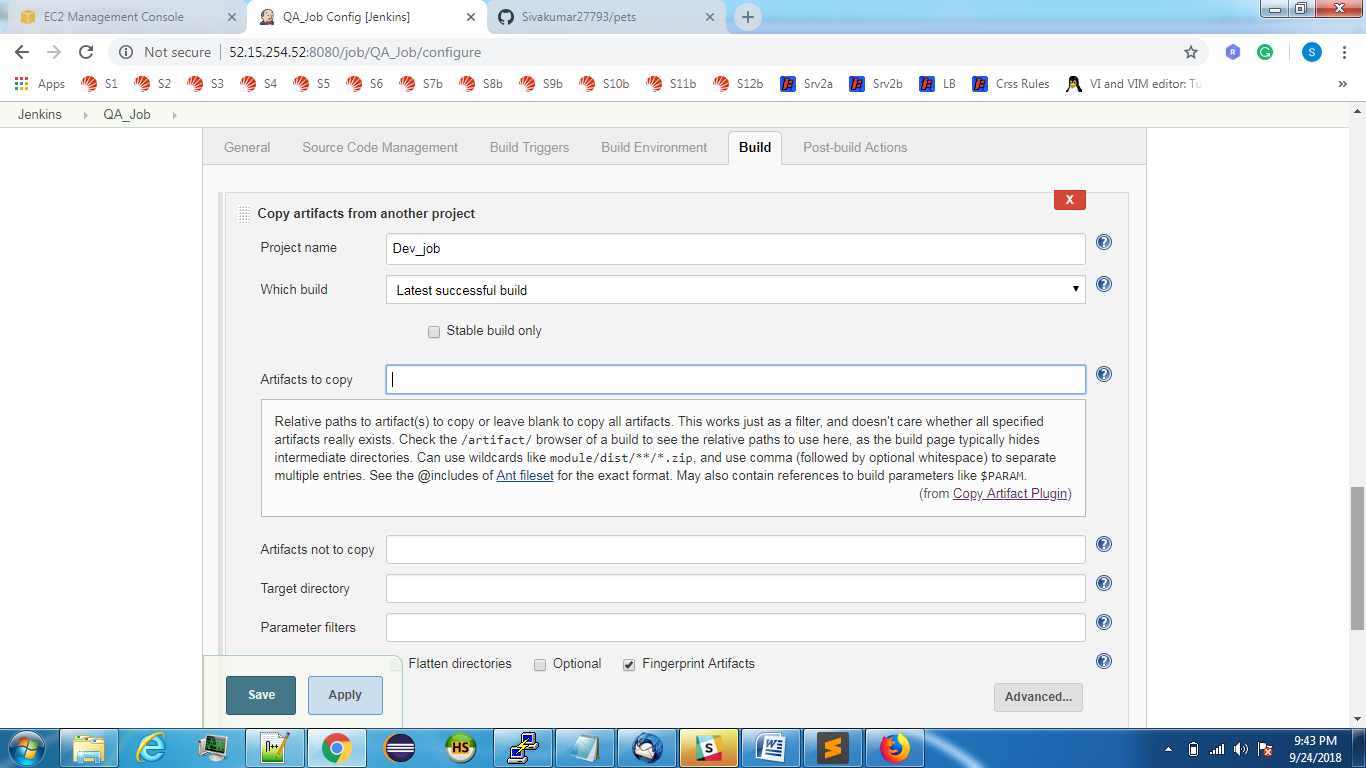
Plugine: Build Pipeline





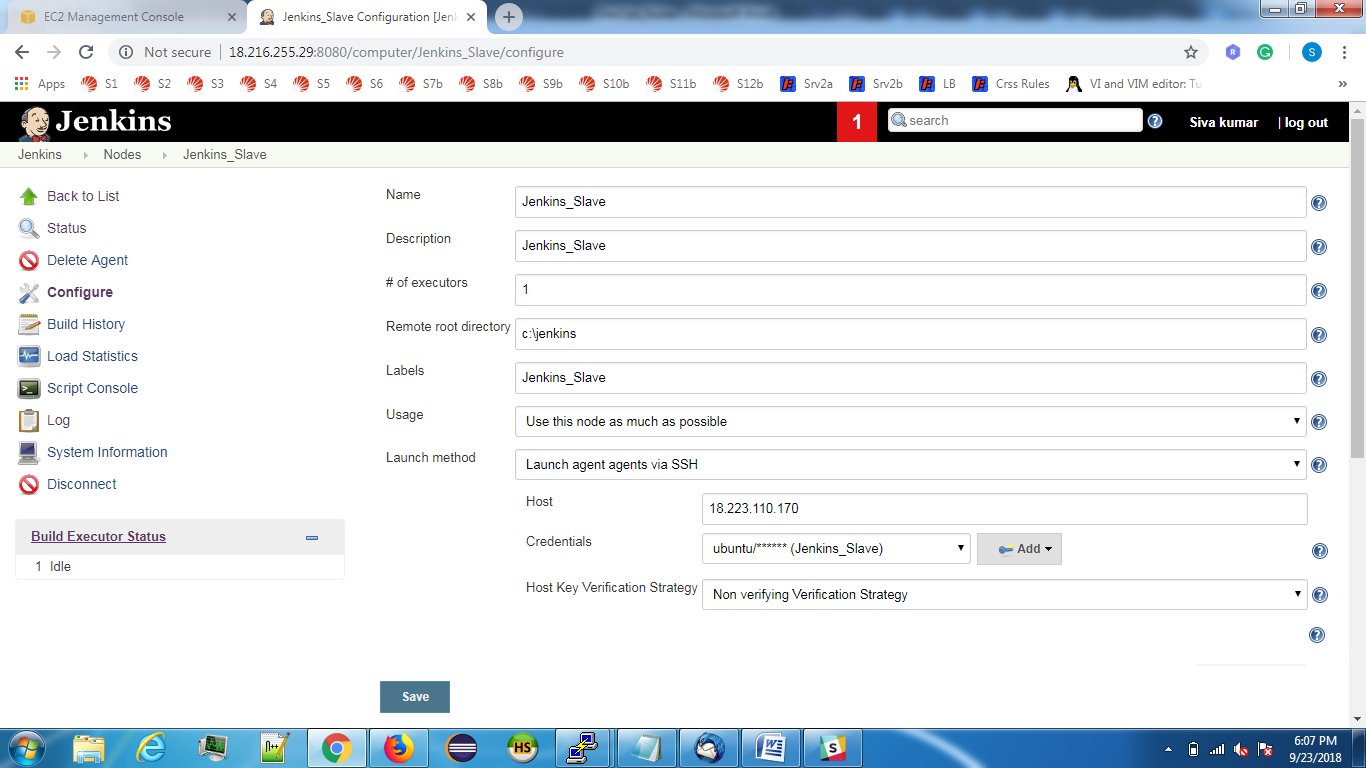
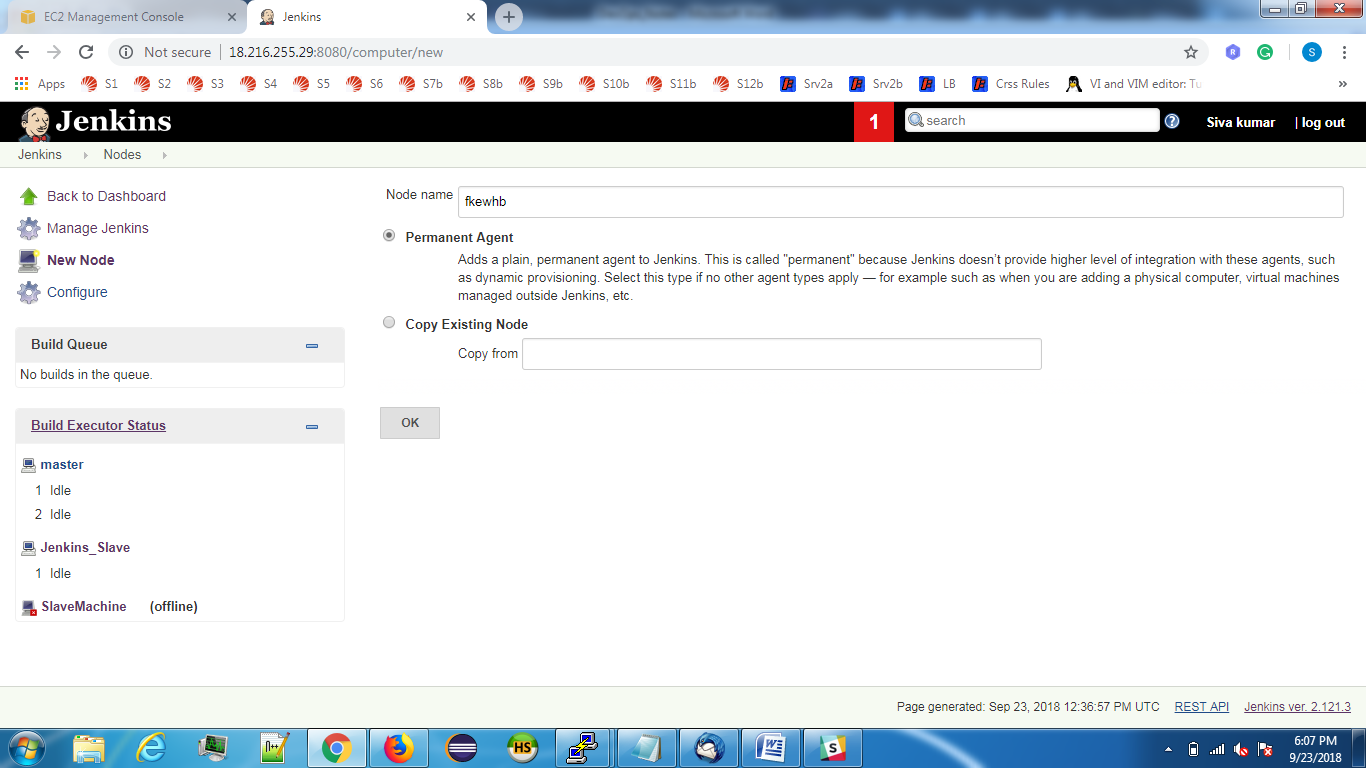


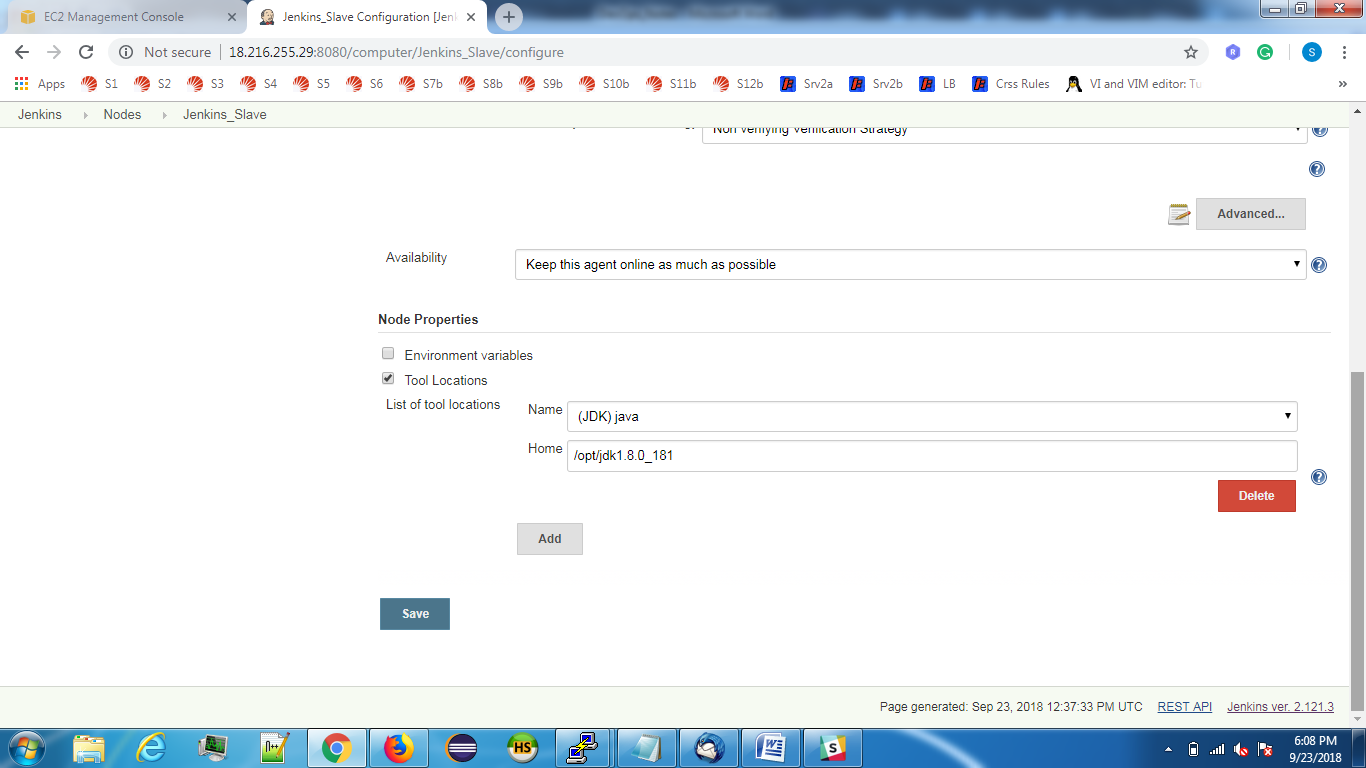
Copy artifacts from One Job/Project to another Job/Project:

s

How to create Jenkins Slave machine:

Manage Jenkins=>Manage Nodes=>New Node





In slave machine we have to install JAVA and Maven path as like as Master.

In slave machine once the build is successful you will get following files.

c:

'c:\jenkins' In Slave machine I have selected Jenkins home directory is c:\jenkins

Then war file will be on the below path.

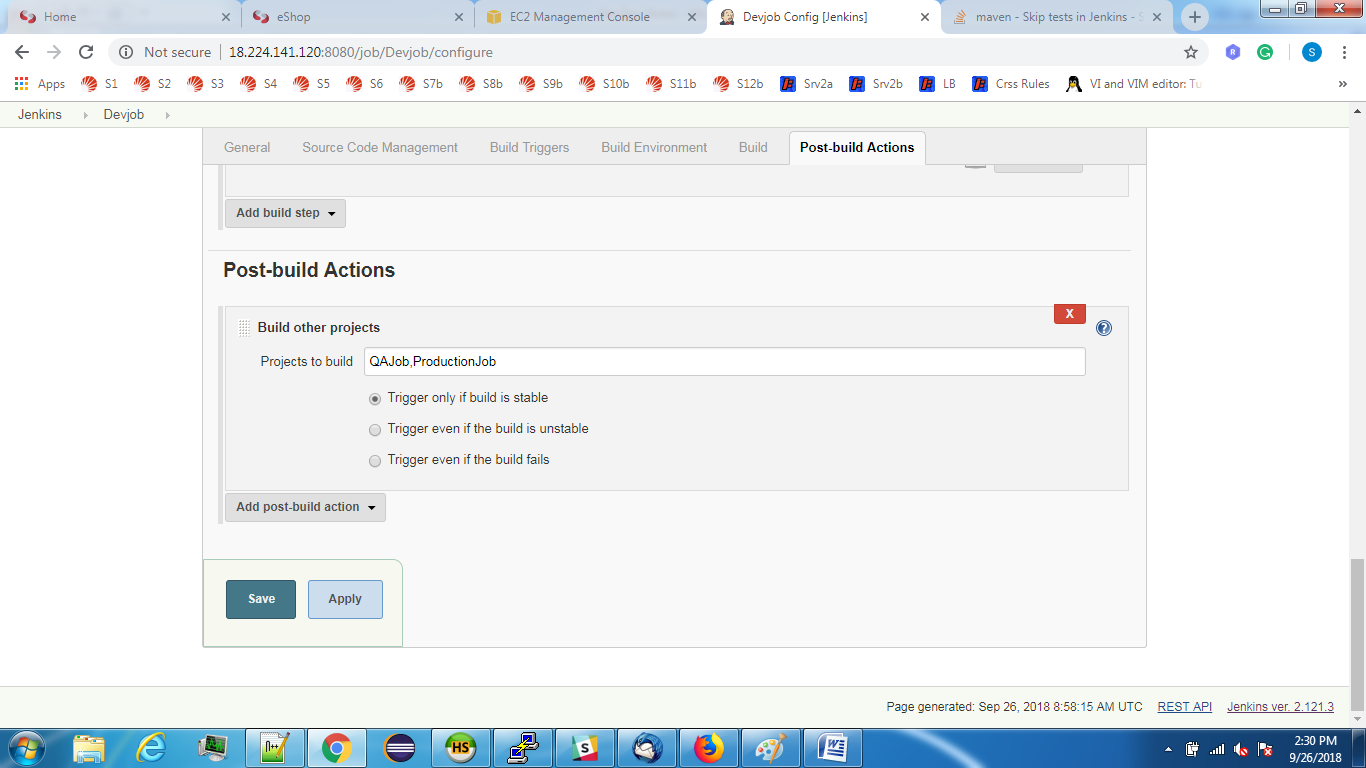
/home/ubuntu/c:/jenkins/c:/jenkins/workspace/jpet/target

Hints: If you want to restatart the Jenkins, **http://18.220.80.191:8080** **/restart**

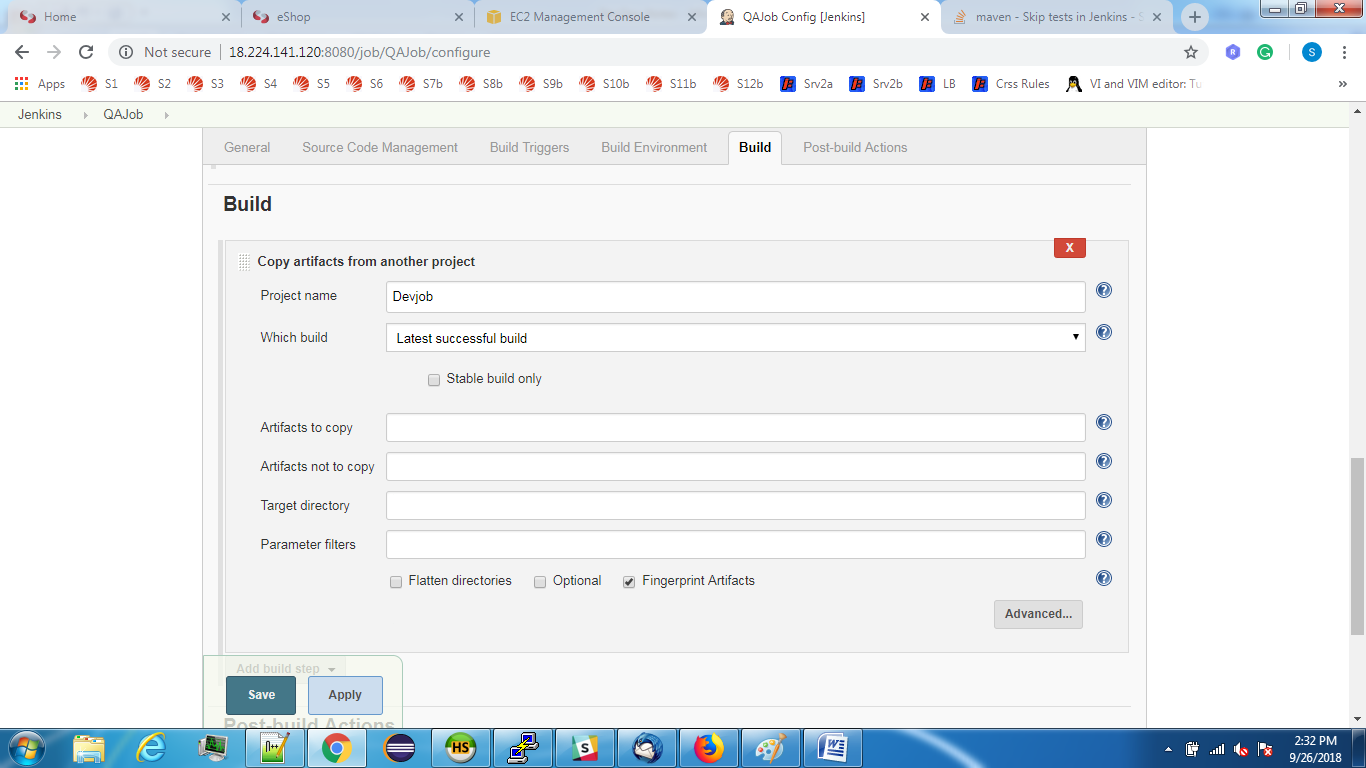
**To create a pipeline project to build the code in Dev and copy the artifacts to QA and Production.**

Plugin: Copy artifacts from another project.

1. Provide the Git and Maven details in Devjob.
2. Add the QA and Production as Downstream jobs in Dev job.

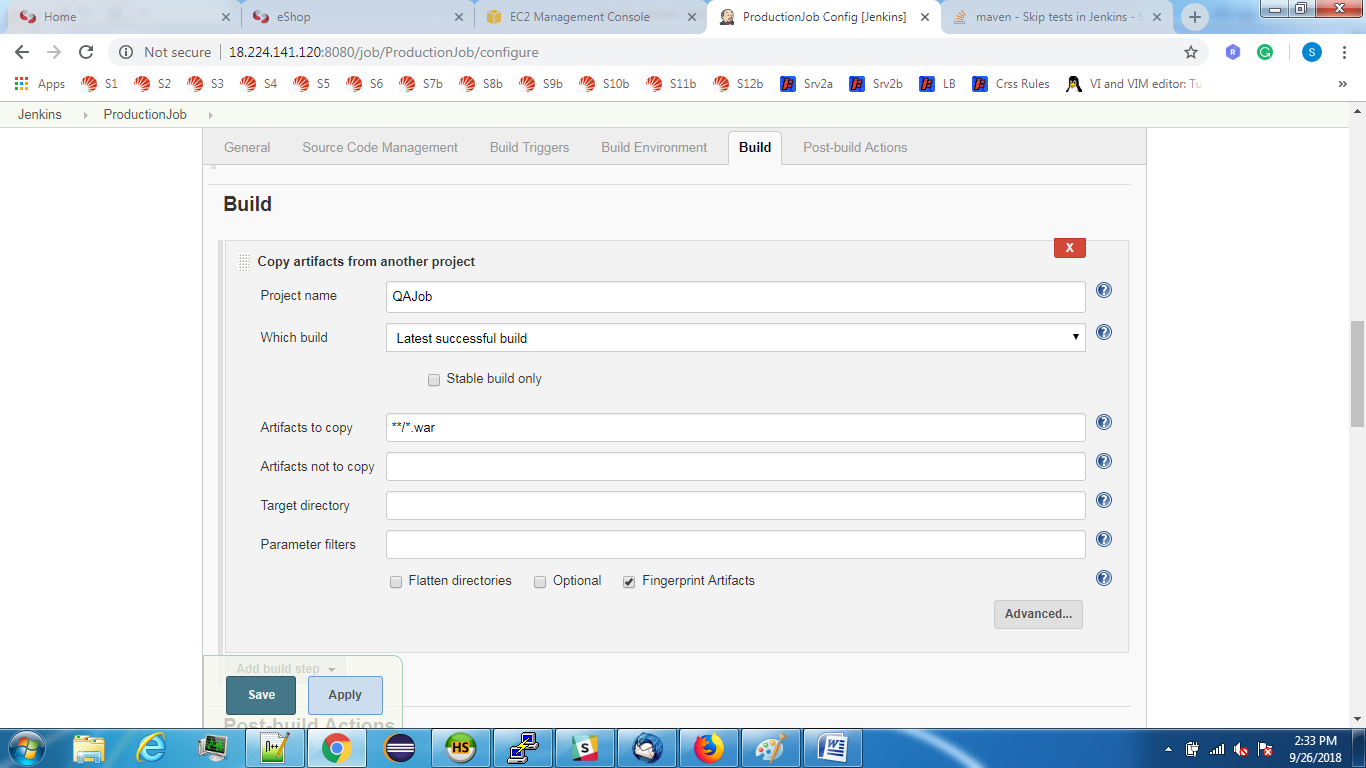


1. In QA job don’t give Git and Maven detsils, just copy the artifacts from Dev job.

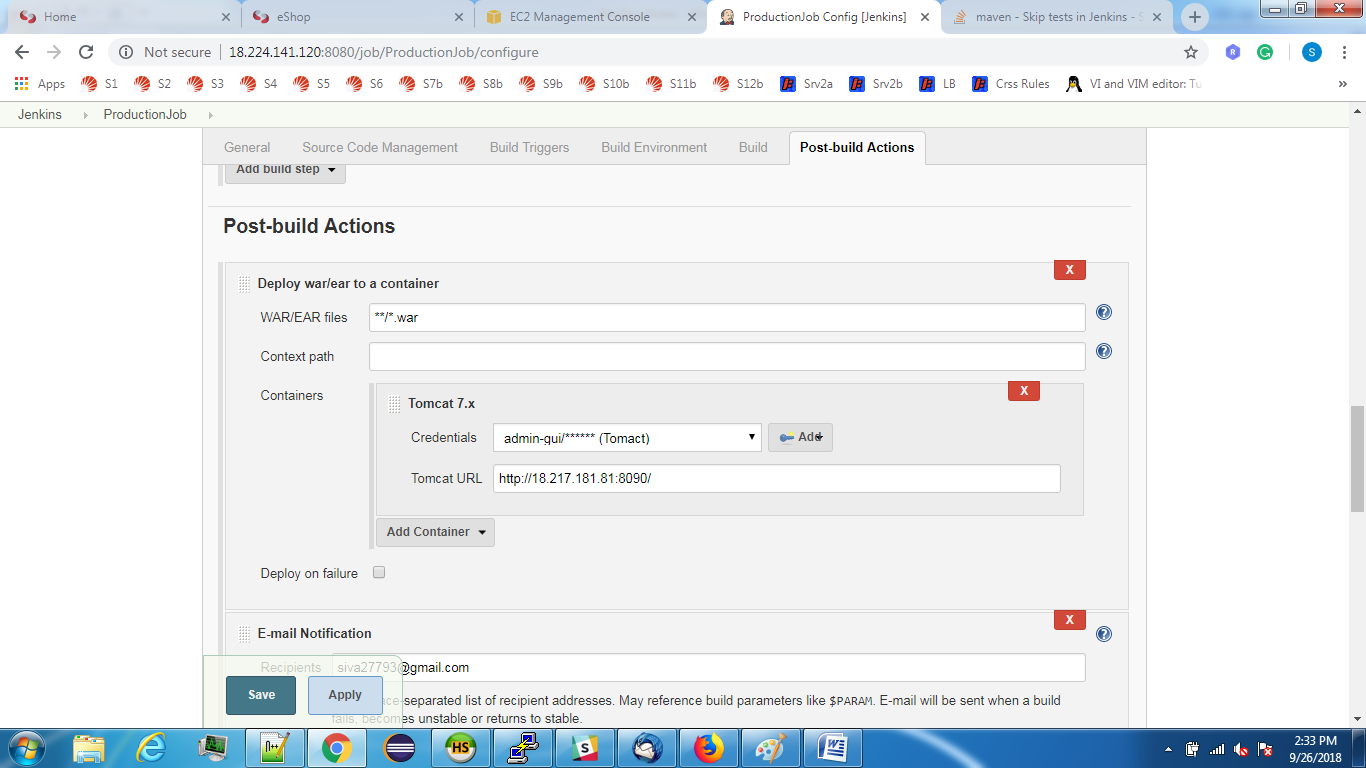


Note: If its failed to copy we have to add \*\*/\*.war in Artifact to copy filed.

1. Then In Production Job copy the artifacts from QA job and deploy it in tomcat.



If the build is failed then select “**Copy from WORKSPACE of latest complete build**” instead of “**Latest Successful build**”



Then the artifacts which were created on Dev job has been deployed in Production Job.

Nexus:

### [1.1. What is a Repository Manager](http://www.vogella.com/tutorials/Nexus/article.html#what-is-a-repository-manager)

An repository manager allows to store and retrieve build artifacts. The most popular examples for repository manager are [Maven Central Repository](http://search.maven.org/) and [jcenter at Bintray](https://bintray.com/bintray/jcenter), which you can use to retrieve your dependencies for a Maven build.

### [1.2. What is Nexus?](http://www.vogella.com/tutorials/Nexus/article.html#what-is-nexus)

A Nexus installation brings you such a repository for your company. So you can host your own repositories, but also use Nexus as a proxy for public repositories. With such a proxy the time to receive an artifact is reduced and it saves ﻿bandwidth. Nexus allows you to host your private build artifacts. Nexus is avaialable as commercial and Open Source distribution.

Steps: Login to Nexus with below credentials.

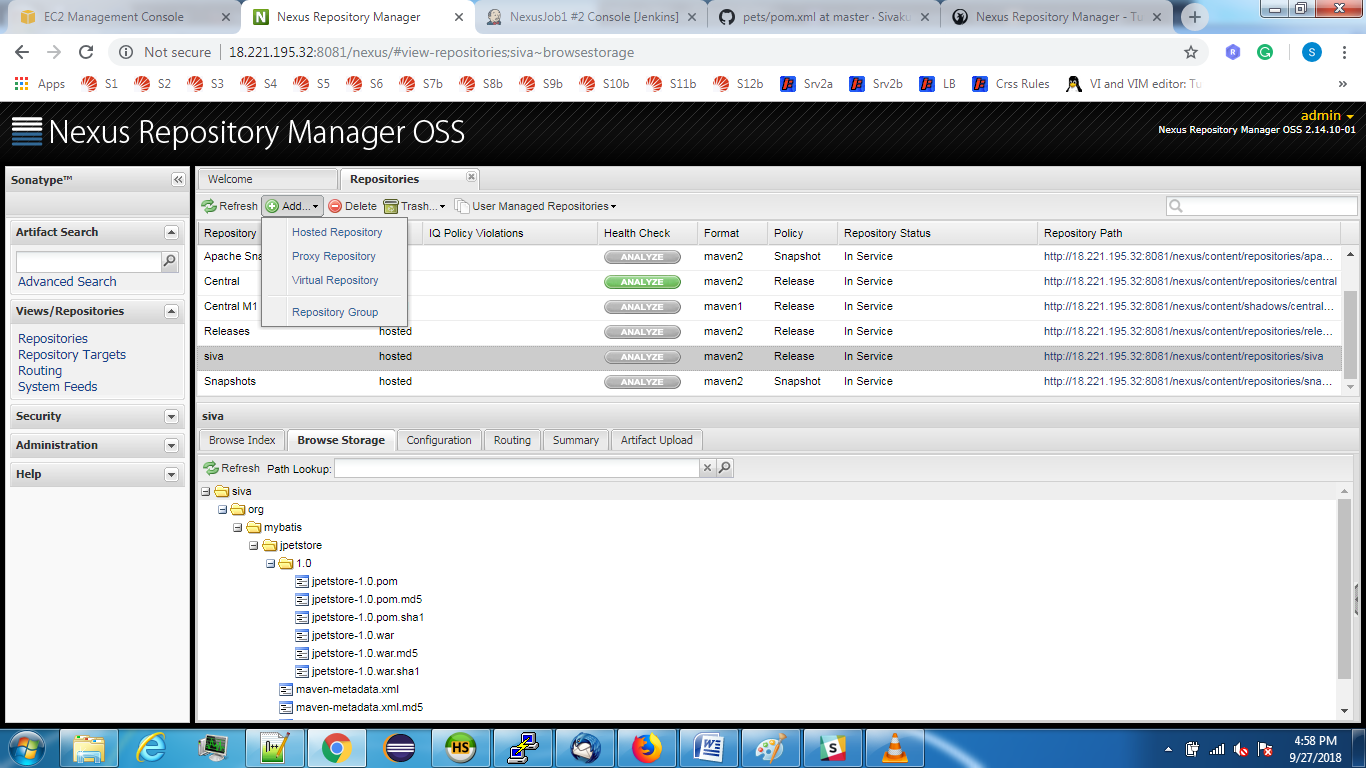
User: admin

Passwd: admin123

To start nexus: Goto /opt=>Nexus folder=>bin then ./nexus start

We have to mention the path as export RUN\_AS\_USER=root

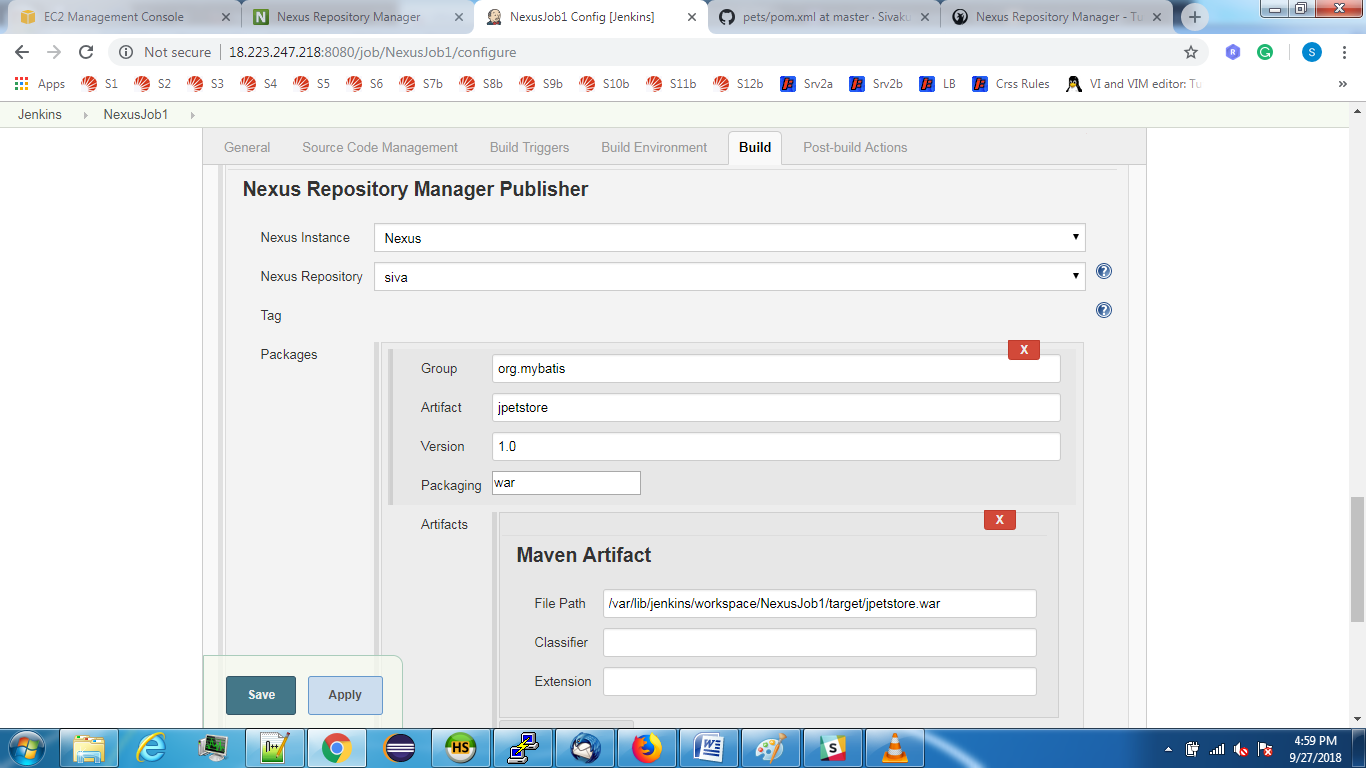
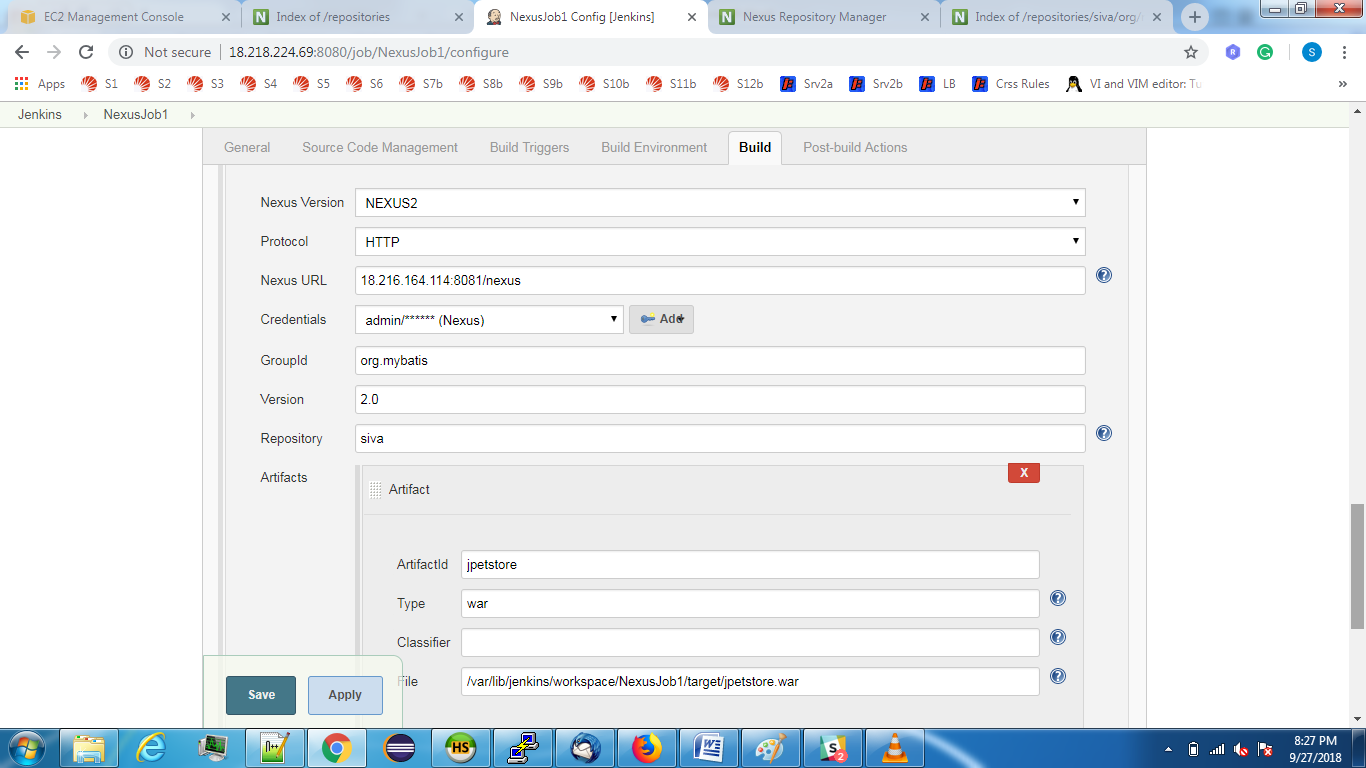
Nexus Url: <http://18.221.195.32:8081/nexus>



Configuring Nexus to Jenkins:

Plugin: **Nexus Repository Manager Publisher**

Add this in Build Step.

Plugin2: 

Here Group Id, Artifact Id taken from pom.xml file

File Path: Path of war file.

**Sonar Qube:**

It is nothing but a Code quality checker.

Installation of Sonar Qube:

Download sonar qube zip from internet and Unzip it.

Once you have installed it then follow the steps.

We have to start Sonar In path: /opt/sonarqube-6.2/bin/linux-x86-64

./sonar.sh start

We have to configure Port and host Ip in the path: /opt/sonarqube-6.2/conf

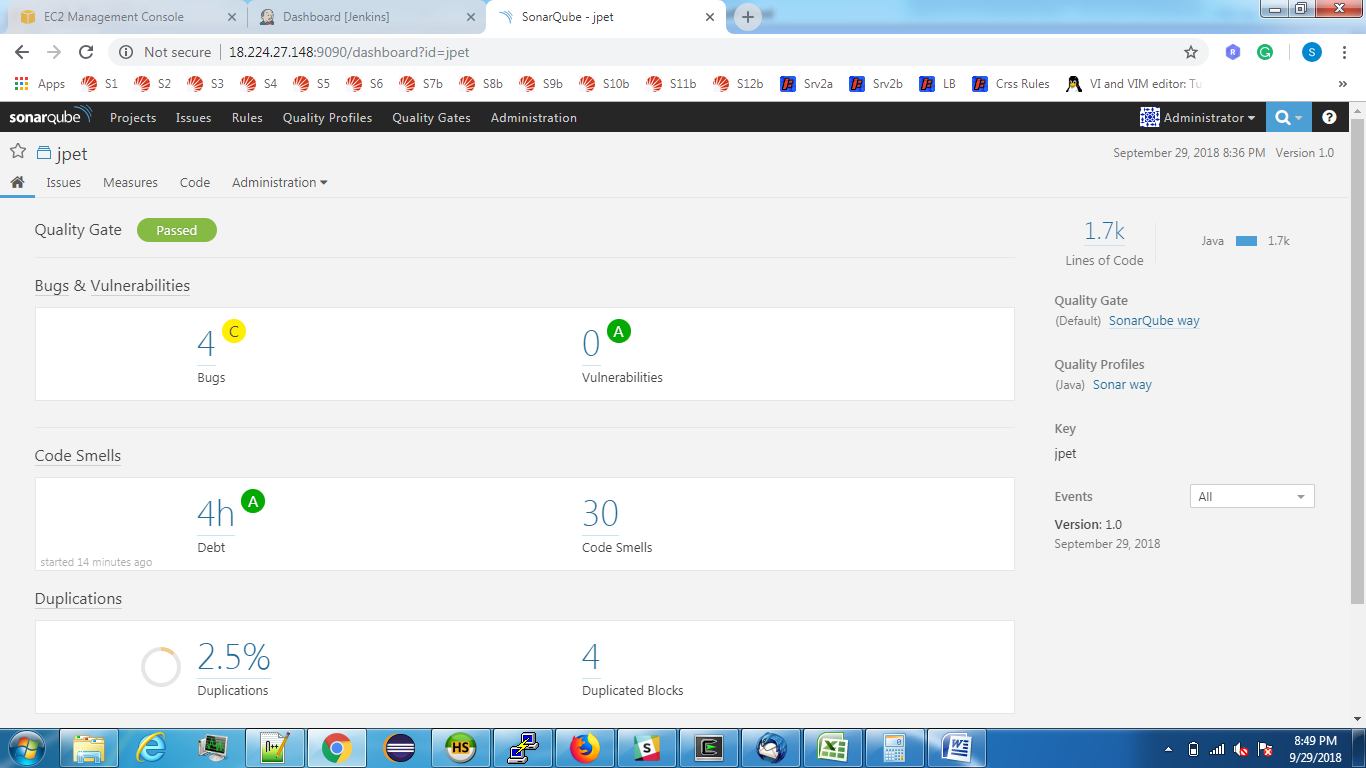


Url for browser: <http://18.224.27.148:9090>

Credentials: Username: admin

Password: admin

Sonar display will be like below.



Inegrating Sonar with Jenkins:

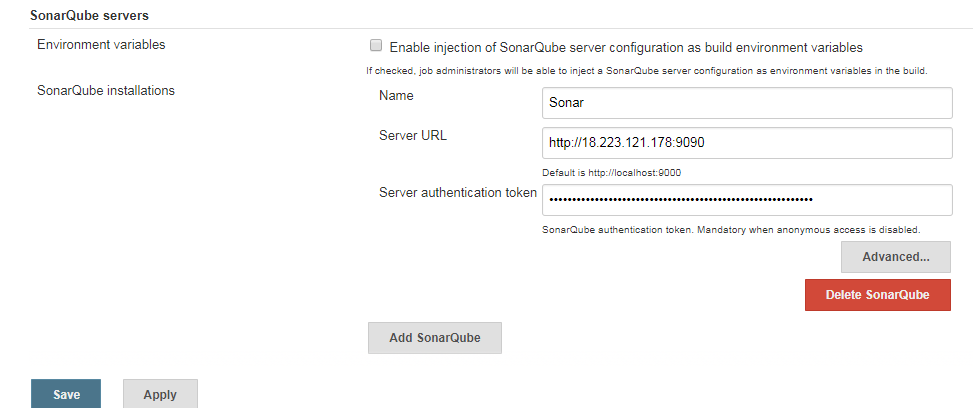
Plugins: [CodeSonar Plugin](https://wiki.jenkins-ci.org/display/JENKINS/CodeSonar+Plugin)

[Sonar Quality Gates Plugin](https://github.com/jenkinsci/sonar-quality-gates-plugin/blob/master/README.md)

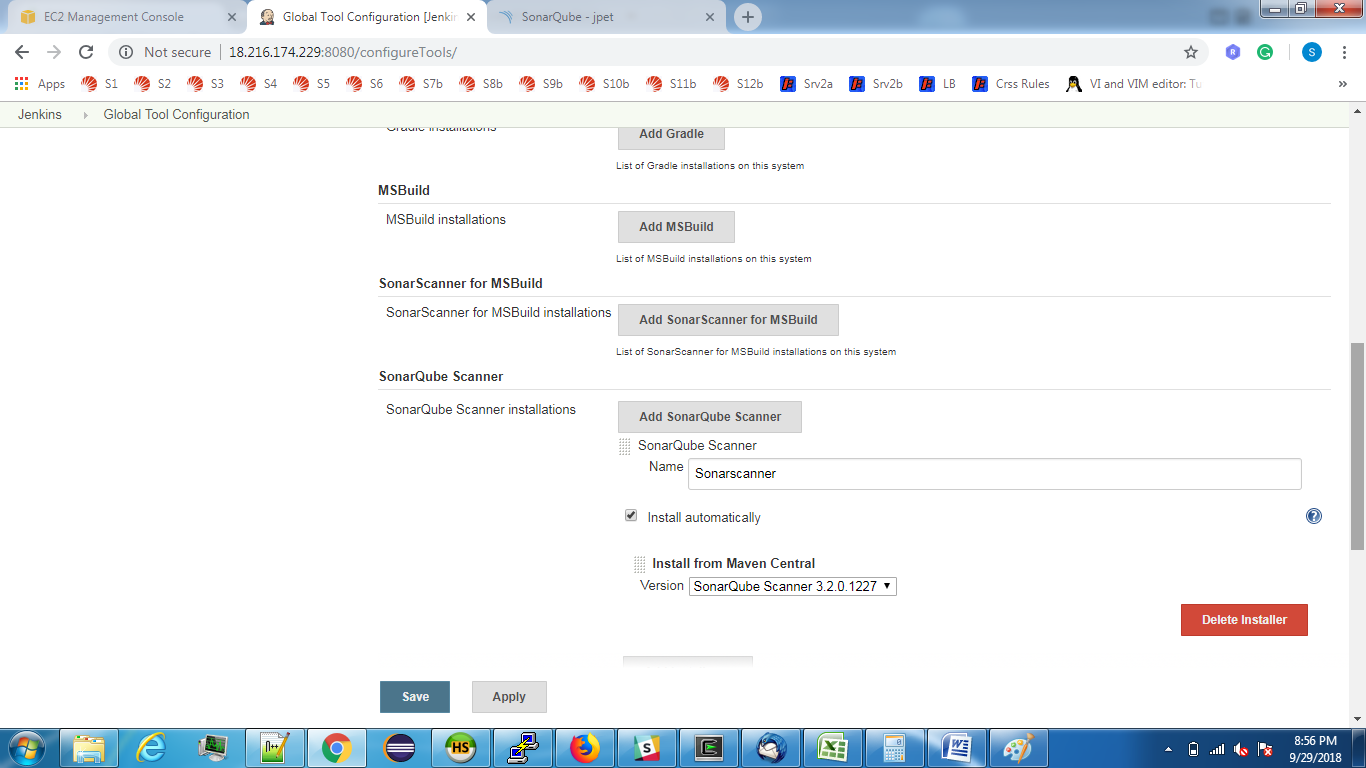
[Sonargraph Plugin](https://wiki.jenkins-ci.org/display/JENKINS/Sonargraph+Plugin)

[SonarQube Scanner for Jenkins](http://redirect.sonarsource.com/plugins/jenkins.html)

Configarations in Jenkins:

In Configure System: 

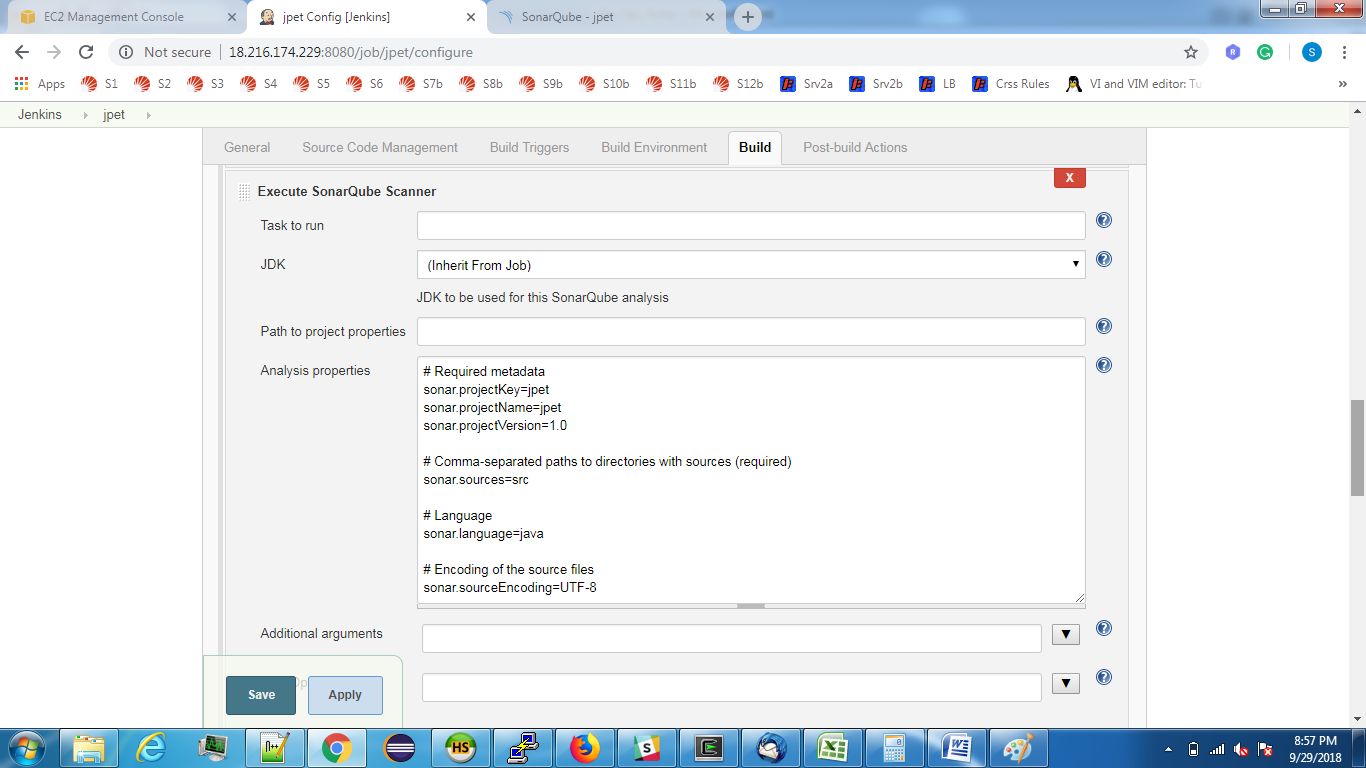
In Global Tool Configaration:



**In Jenkins adding sonar qube details:**

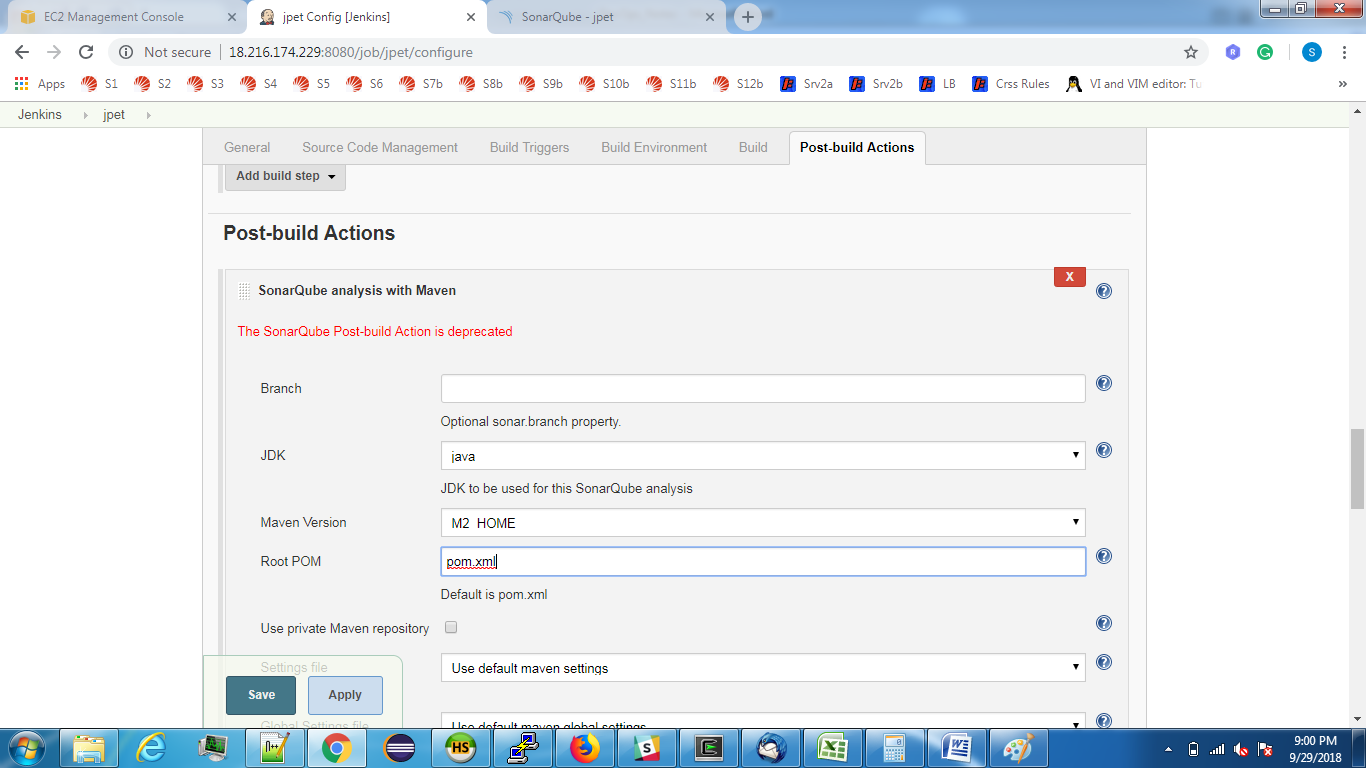
**Plugin**: **Execute SonarQube Scanner**

**In Build Step:**

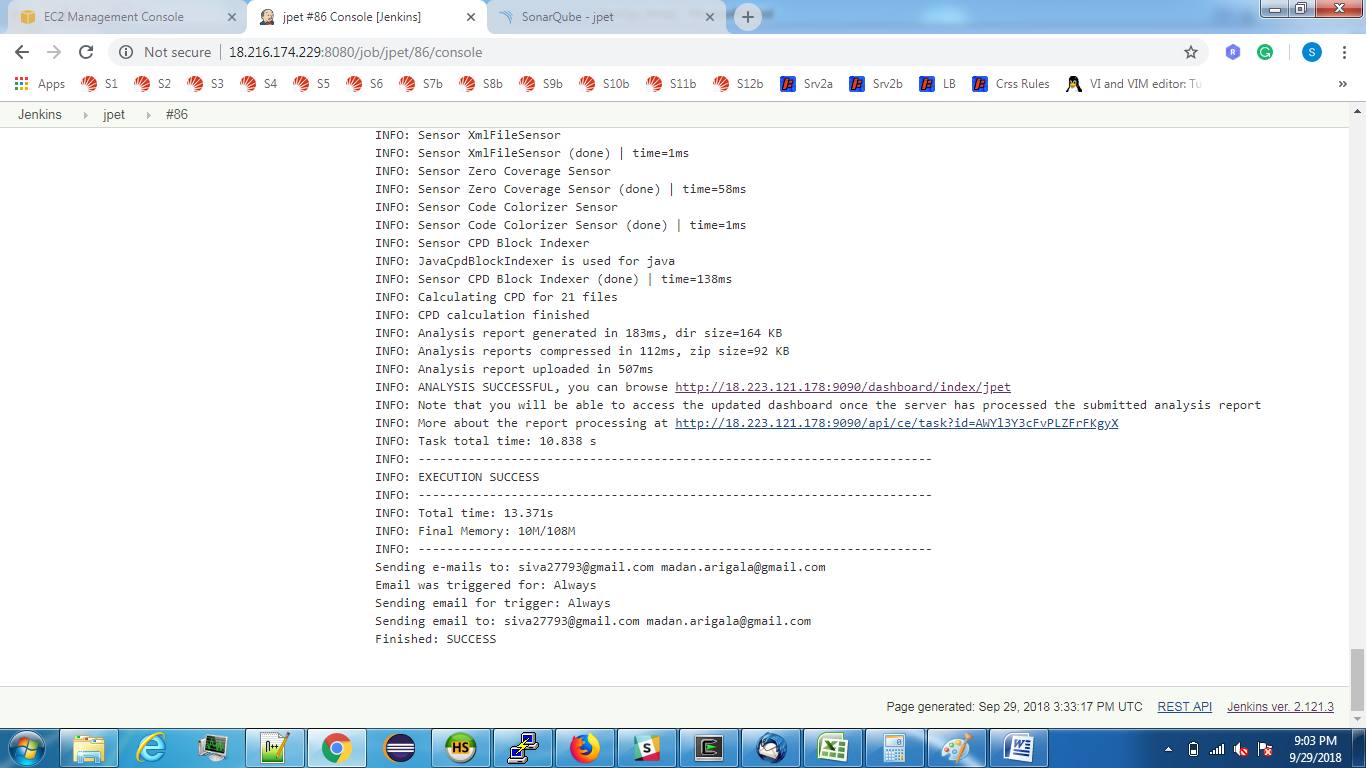


**In Post Build actions:**

**Plugin: SonarQube analysis with Maven**



Console out when we run at Build step:



Ansible Setup In Jenkins,

