**Complete Project Setup**

**====================**

**Git**

**===**

# sudo yum -y install git

# git --version

**Java**

**====**

# sudo yum -y install java-1.8.0

# sudo yum -y install java-1.8.0-devel

# java -version

**Maven**

**======**

# sudo yum -y install maven

# mvn --version

**Nexus**

**======**

# wget <nexus-2-link.tar>

**#** wget**<http://www.sonatype.org/downloads/nexus-latest-bundle.tar.gz>**

# cd nexus/nexus-2.x/bin

# ./nexus start

**{** if any problem change ownership to devops to both dirs of nexus**}**

# netstat -ntpl | grep 8081

{ be patient it takes some time to start }

#<http://ip-addr:8081/nexus>

{ allow port 8081 through firewall }

**# login and give default creds admin & admin123**

**ERROR**: **Change the permissions of two nexus directories with logged in username**

# sudo ln -s ~/nexus-dir/bin/nexus /etc/init.d/nexus

# sudo chkconfig --add nexus

# sudo chkconfig --levels 345 nexus on { start on bootup }

# service nexus start { **Don’t use sudo** }

**Sonarqube [Requires java 1.8+]**

**=========**

**Install MySql 5.6+**

# wget http://repo.mysql.com/mysql-community-release-el7-5.noarch.rpm

# sudo rpm -ivh mysql-community-release-el7-5.noarch.rpm

# sudo yum -y install mysql-server

# sudo systemctl start mysqld

. sudo grep 'temporary password' /var/log/mysqld.log  
   temp passwd genetrated : VbRsrn<r1VTE

# sudo mysql\_secure\_installation  
# sudo mysql -u root -p

# CREATE DATABASE sonar;

# CREATE USER 'sonar' IDENTIFIED BY 'sonar';

# GRANT ALL ON sonar.\* TO 'sonar'@'localhost' IDENTIFIED BY 'sonar';

# FLUSH PRIVILEGES;

exit

# sudo wget -O /etc/yum.repos.d/sonar.repo http://downloads.sourceforge.net/project/sonar-pkg/rpm/sonar.repo

# sudo yum -y install sonar

# sudo vim /opt/sonar/conf/sonar.properties

Changes to make in **sonar.properties**

**==============================**

# sonar.jdbc.username=sonar

# sonar.jdbc.password=sonar

# uncomment sonar.jdbc.url of MySQL 5.6

# uncomment sonar.web.host=0.0.0.0

# uncomment sonar.web.port=9000

# sudo systemctl start sonar

# sudo systemctl enable sonar

# allow port 9000 through firewall

# firewall-cmd --zone=public --add-port=9000/tcp --permanent

# firewall-cmd --reload

# [http://serverAddress:9000](http://serveraddress:9000)

# username and password is admin & admin

**Jenkins**

**=======**

# sudo wget -O /etc/yum.repos.d/jenkins.repo<https://pkg.jenkins.io/redhat-stable/jenkins.repo>

# sudo rpm --import<https://pkg.jenkins.io/redhat-stable/jenkins.io.key>

# sudo yum -y install jenkins

# sudo systemctl start jenkins

# sudo systemctl enable jenkins

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

# sudo firewall-cmd --reload

**Tomcat**

**=======**

Grab the binary archive for tomcat 7 from maven tomcat

**# Change the PORT number 7777**

**# sudo vi tomcat-users.xml { gui tomcat manager }**

Copy this in **<tomcat-users>** tags

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

*<user username="tomcat" password="tomcat" roles="manager-gui"/>*

*<user username="tomcat1" p./assword="tomcat1" roles="manager-script"/>*

**# change the port number of tomcat**

**# sudo vi server.xml { change port in** Connector **}**

Allow port through **firewall**

**# sudo systemctl start tomcat**

**# Download tar of tomcat7 and extract → cd tomcat/bin → ./startup.sh**

**Starting the Project  
================**

# Go to github.com and check the project Info  
# Copy project URL  
[https:/ravi2krishna/github.com//spring-hibernate-maven-webapp.git](https://github.com/ravi2krishna/spring-hibernate-maven-webapp.git)

**JOB - 1**

Freestyle Job [ **01 - Validate and Compile** ]

→ Discard Old builds

Max no of builds 3

→ Source code management

GitHub url

→ Build

**Invoke top-level Maven targets**

**Goals : validate compile**

**Make Sure to change the settings in pom.xml → Remove Nexus Elements**

**JOB - 2**

Freestyle Job [ **02 - Unit Testing** ]

→ Discard Old builds

Max no of builds 3

→ Source code management

GitHub url  
→ Build

**Invoke top-level Maven targets**

**Goals : test  
  
But we can’t see how many test cases ran and how many were failed/succeeded**

**→** Go to workspace → target → surefire-reports → index.html, testng-results.xml

**→ At the end Post-build Actions  
 Publish JUnit test result report**

**Locations : \*\*/target/surefire-reports/\*.xml**

→ Run the build again for 2-3 times and you can see the test graph  
→ Click on **Latest Test Result** ⇒ On left side click on **History**

**→ Refresh the Job page to view updated changes**

**JOB - 3**

Freestyle Job [ **03 - Cobertura Code Coverage** ]

→ Discard Old builds

Max no of builds 3

→ Source code management

GitHub url  
→ Build

**Invoke top-level Maven targets**

**Goals : cobertura:cobertura -Dcobertura.report.format=xml**

**→ Now if you see console output :  
 → Cobertura: Loaded information on 10 classes.  
 → Cobertura Report generation was successful.**

**→ But we can’t see any reports, so let’s go and work with plugin now  
 → Even if we see in Post-build actions nothing is found  
 → Install Cobertura plugin**

**→ At the end Post-build Actions  
 Publish Cobertura Coverage Report**

**Locations : \*\*/target/site/cobertura/coverage.xml**

**→ Refresh the Job page to view updated changes**

**JOB - 4**

Freestyle Job [ **04 - Documentation** ]

→ Discard Old builds

Max no of builds 3

→ Source code management

GitHub url  
→ Build

**Invoke top-level Maven targets**

**Goals : javadoc:javadoc**

**→ Now if you see console output :  
 → All the documentation is being generated in target/site/apidocs**

**→ Even if we see in Post-build actions nothing is found  
 → Install Javadoc Plugin**

**→ At the end Post-build Actions**

**Publish Javadoc**

**Locations : target/site/apidocs**

**→ Refresh the Job page to view updated changes**

**JOB - 5**

Freestyle Job [ **05 - Sonar Analysis - Code Analysis** ]

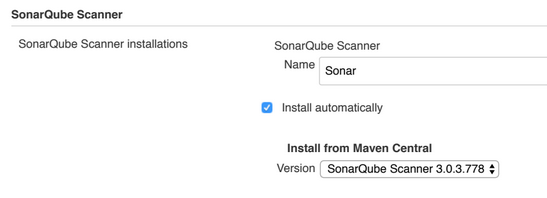
→ Discard Old builds

Max no of builds 3

→ **Source code management**

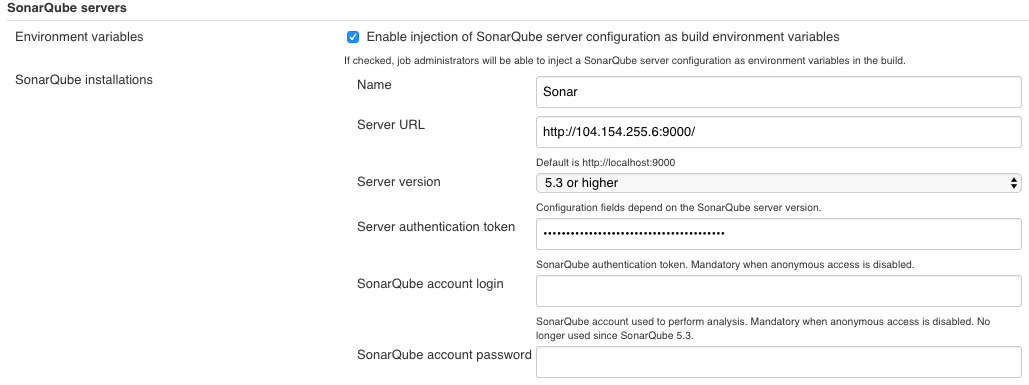
GitHub url  
→ **Build Environment →** There is no sonar here

→ **Manage Jenkins → Global Tool Config**

****

**→ Install Sonarqube Scanner for jenkins**

**→ Now go to the Manage Jenkins → Configure System → SonarQube servers**

****

**→ Job Configure** → **Build Environment**✅**Prepare SonarQube Scanner env**

→ **Build**

**Invoke top-level Maven targets**

**Goals : clean compile sonar:sonar**

**JOB - 6**

Freestyle Job [ **06 - Packaging and Nexus Release** ]

→ **Source code management**

GitHub url

→ **Build**

**Invoke top-level Maven targets  →  Advanced**

**Goals : clean package deploy  
 Settings File  
 Settings file in filesystem  
 File path - settings.xml**

**→ Post Build Actions** : **Archive the artifacts** - **\*\*/\*.war**

**JOB - 7**

Freestyle Job [ **07 - Tomcat Deployment** ]

→ Discard Old builds

Max no of builds 3

→ Source code management

GitHub url

→ **Build**

**Invoke top-level Maven targets**

**Goals : clean package**

**→ Install Deploy to container plugin**

**→ At the end Post-build Actions**

**Deploy war/ear to a container**

WAR/EAR files - \*\*/\*.war

Context path    - /SpringProject

**Containers**

Tomcat 7.x

Credentials tomcat1/\*\*\*\*\*\* (Tomcat) Add

Tomcat URL<http://104.197.89.223:7070>

**Install Build Pipeline Plugin → New View → CI-CD and check Build Pipeline View**

**Tasks**

======

* Implement Jenkins, Nexus & Sonarqube on Docker
* Implement Master & Slave Configuration
* Create three branches dev, qa and prod in code and deploy to three different environments.
* The three different environments should be configured with chef.
* Deploy the different branches to different tomcats which were configured with tomcat.
* Run the prod tomcat with Nginx reverse proxy.