*Build Pipeline With Maven project & sonarqube analysis*





***Build-Pipeline with Maven Project & SonarQube Analysis***

*CONTENTS:*

1. *Jenkins ................................................................................................................................2*
2. *Pipeline ...............................................................................................................................2*
   1. *Benefit of Pipeline ........................................................................................................2*
   2. *Declarative vs Scripted Pipeline ...................................................................................2*
   3. *How to Create Jenkins Pipeline ....................................................................................2*
3. *Jenkins File .........................................................................................................................3*
   1. *Benefits using Jenkins File ...........................................................................................4*
4. *SonarQube .........................................................................................................................4*
   1. *Why we use SonarQube ..............................................................................................5*
   2. *How to Check Code Quality in SonarQube ..................................................................5*

*Jenkins:*

Jenkins is an open source continuous integration server that provides the ability to continuously perform automated builds and tests.

*Pipeline:*

* Jenkins Pipeline is a suite of plugin which support implementing and integrating continuous delivery pipelines in to Jenkins.
* A continuous delivery(CD) pipeline is an automated expression of our process for getting software from version control right through to the users and customers.  Every change to your software (committed in source control) goes through a complex process on its way to being released. This process involves building the software in a reliable and repeatable manner, as well as progressing the built software through multiple stages of testing and deployment.

*What are Benefits of Jenkins Pipeline:*

* Using Jenkins Pipeline to automate CI/CD Pipelines dramatically increased repeat-ability, reliability, efficiency and quality.

*Jenkins File:*

Jenkins Pipeline can be defined using a text file called **Jenkins File.** You can implement pipeline as code using Jenkins File, and this can be defined by using a domain specific language.

*Benefits using Jenkins File:*

* You can create Pipeline automatically for all branches and execute pull request with just one Jenkins File.
* You can review your code on the Pipeline.
* You can audit your Jenkins Pipeline.

*Declarative vs Scripted Pipeline Syntax:*

There are two types of syntax used for defining your Jenkins file.

1. Declarative
2. Scripted

*Declarative:*

Declarative Pipeline syntax offers an easy way to create Pipelines. It contains a predefined Hierarchy to create Jenkins Pipelines. It gives us the ability to control all aspect of a Pipeline execution in a simple and straight forward manner.

*Scripted:*

Scripted Jenkins Pipeline run on the Jenkins master with the help of a lightweight executer. It uses very few resource to translate the Pipeline into atomic commands. Both declarative and scripted syntax are different from each other and are defined totally differently.

*How to Create Jenkins Pipeline:*

Once you are logged in to your Jenkins dashboard.

Step-1: Click on **New Item**, then enter theItem Name (Ex- **Pipeline-maven -test**)

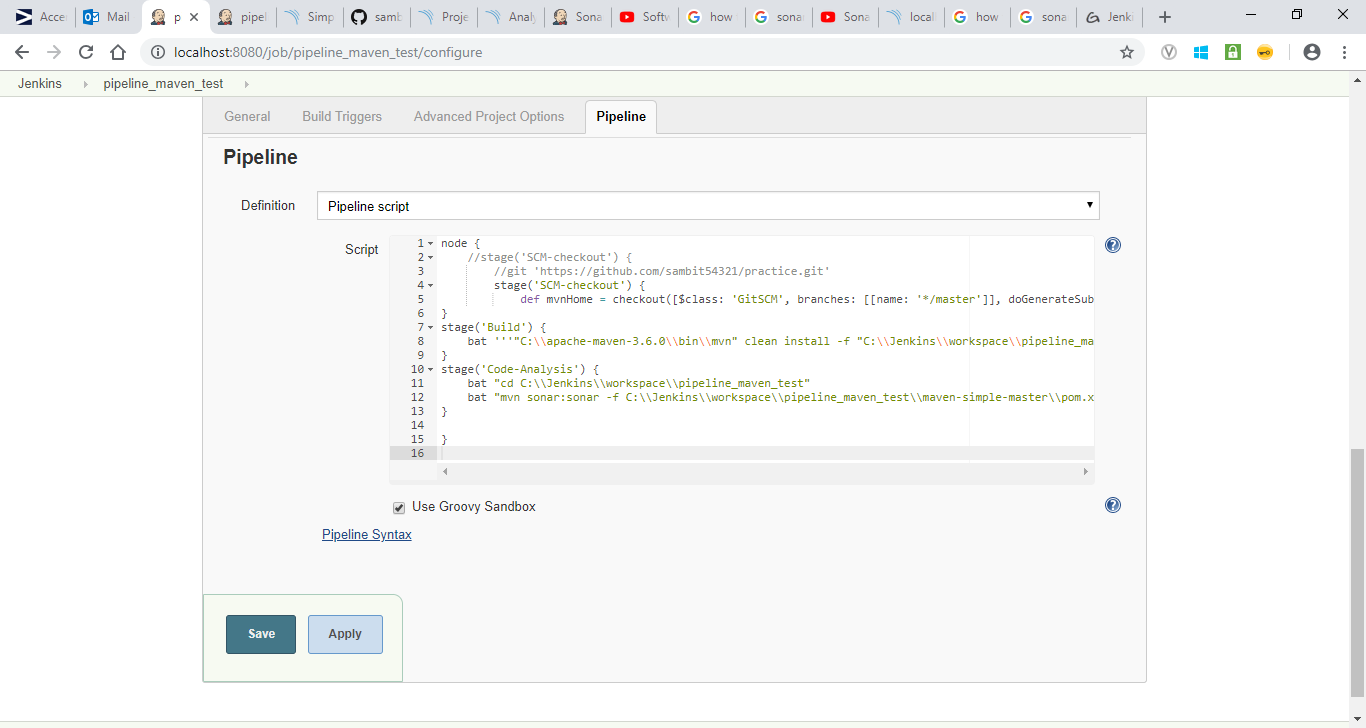
Step-2: Select Build a **pipeline view** under options.

Step-3: Click **Ok**

Step-4: Now go to pipeline section and select Definition as “**pipeline script”** & and in script chose the option “**try sample pipeline**”.

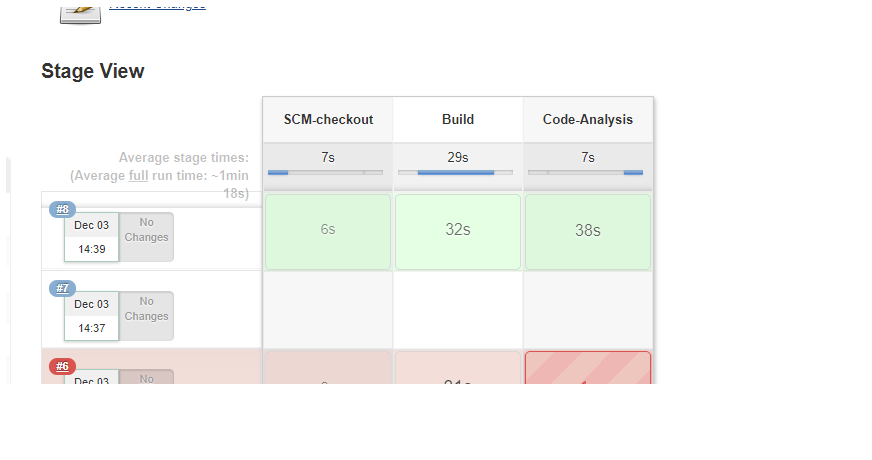
Step-5: Finally, you can start working your **pipeline script** in script page.

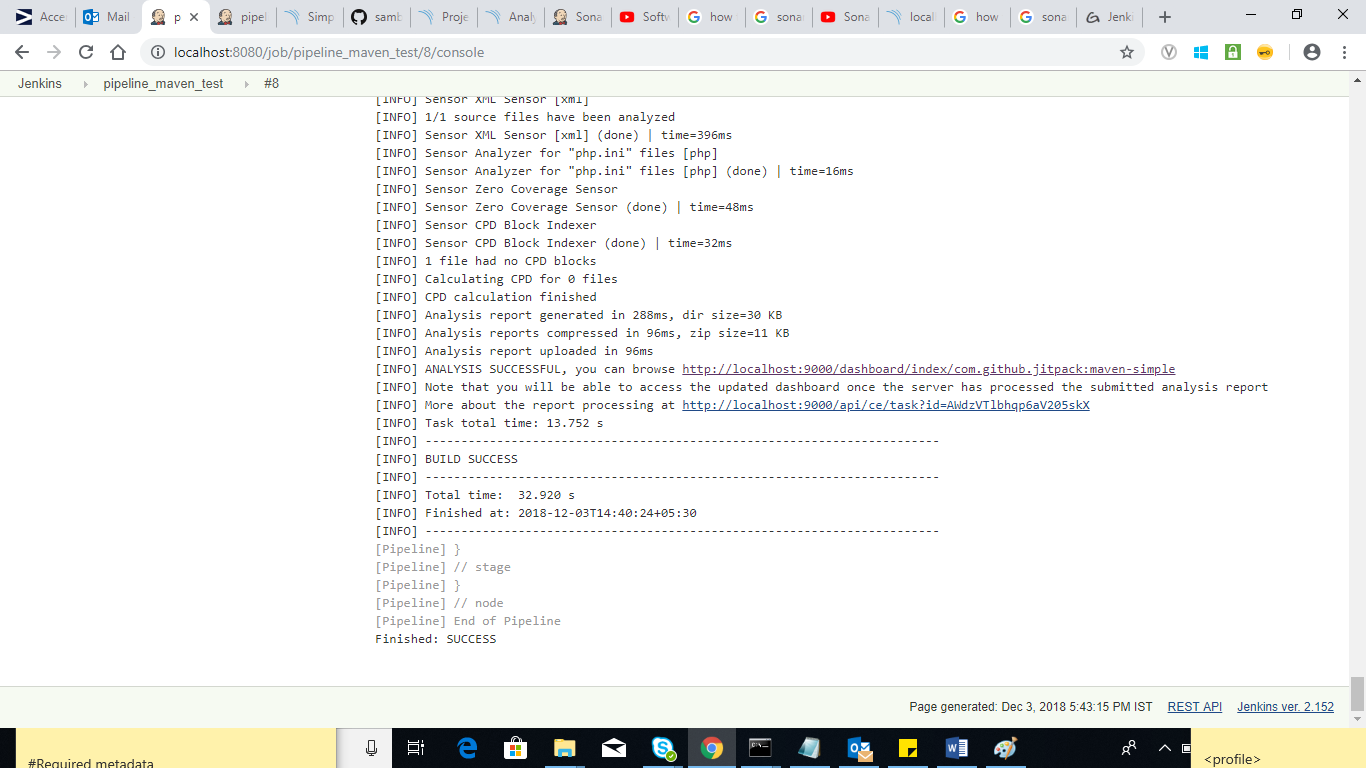
***Script Page:***



Step-6: After writing the script, then click on “**Apply”** and “**Save”.**

Step-7: And finally, you Have to build your pipeline, so for that click on “**Build Now”** option.





Then, after that you Have to go through SonarQube for code quality.

*SonarQube:*

Sonar is a web-based code quality analysis toll for Maven based java projects.

*Why we use SonarQube:*

Sonar covers some section of code quality like.

* Architecture and design
* Unit tests
* Duplicated code
* Potential bugs
* Complex code
* Coding standards

*How to check code quality in SonarQube:*

Step-1: First install the SonarQube

Step-2: Then open the browser and type <http://localhost:9000> (localhost:9000 is the default host name of SonarQube)

Step-3: Then you Have to integrate to SonarQube with Jenkins. So that you Have to go to **manage** **Jenkins**-> **configure system**-> **sonar server**->then you Have to specify **Name**, **Server URL** and **Server authentication token**. Then click on **apply** and **save**.

Then again you Have to go to Global tool configuration-> SonarQube scanner-> then you have to mansion **Name** and **Sonar Runner Home**. -> then **apply** and **save.**

Step-4: After build the Pipeline You Have to define one more **stage** to define the code analysis. Then you have click on **Apply** and **Save** option.

Step-5: Then finally click on Build Now option for building the pipeline.

Step-6: Then go to SonarQube dash board getting the output.

