Documentation of Deployment of Projects

**Contents**

[**Installation of U-Deploy Agent-:** 3](#_Toc454967278)

[**CI/CD automation for Dotnet project with Jenkins, SonarQube and IIS server-:** 6](#_Toc454967279)

[1)Check-In sample Dot-Net project into SVN 6](#_Toc454967280)

[2) Configuring the project with Jenkins and scheduling a build with Sonar and MS-Build 7](#_Toc454967281)

[3) Deploy in IIS server 10](#_Toc454967282)

[**Creating master-Slave in Jenkins** 15](#_Toc454967283)

[**CI/CD automation for php project with Jenkins, SonarQube and Apache server** 19](#_Toc454967284)

[1) Check-In sample PHP project into SVN 19](#_Toc454967285)

[2) Configuring in Jenkins and scheduling a build with Sonar 20](#_Toc454967286)

[**CI/CD automation for Java project with Jenkins, SonarQube and Tomcat server & IBM Udeploy** 26](#_Toc454967287)

[1) Check-In sample Java project into SVN. 26](#_Toc454967288)

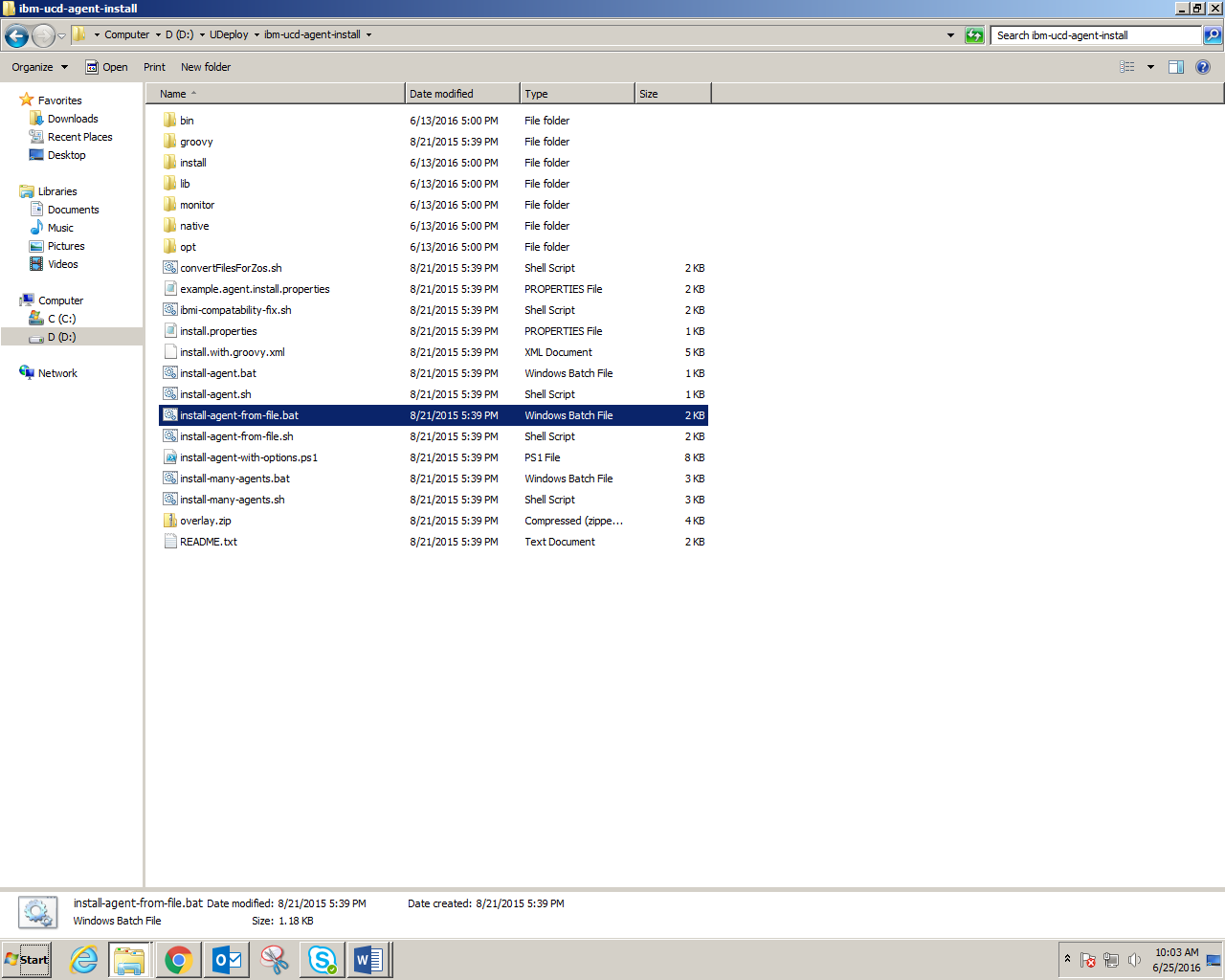
[2) Configuring in Jenkins and scheduling a build with Sonar 27](#_Toc454967289)

[3) IBM UrbanCode Deploy 31](#_Toc454967290)

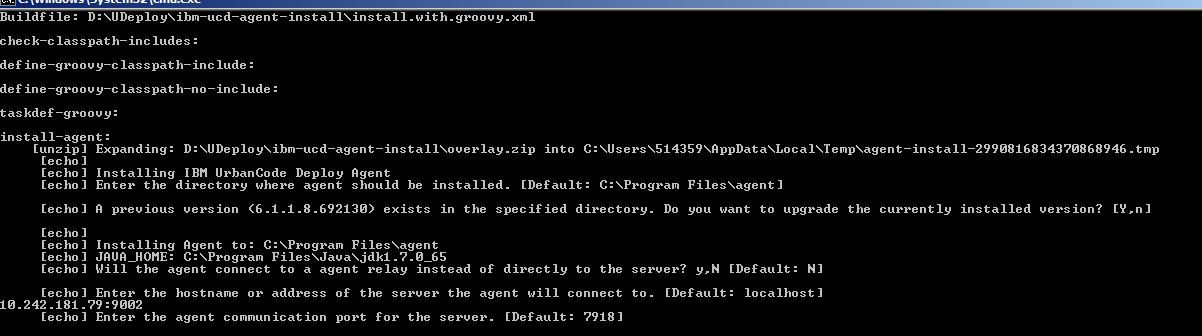
[4) Browse your application 46](#_Toc454967291)

# **Installation of U-Deploy Agent-:**

1. **Extract the files from the IBM UDeploy Agent folder 🡺 Go to Bin folder and run the install-agent-from-file.cmd command through command prompt.**

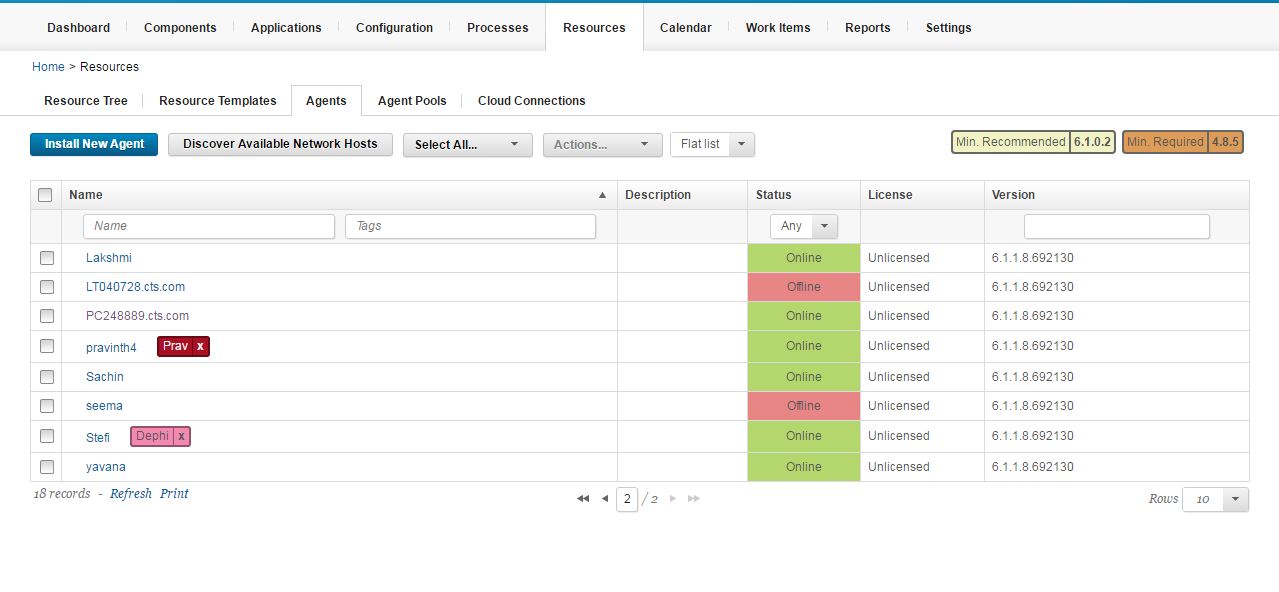


1. **In command prompt, set the values as shown, and agent is connected to UDeploy.**



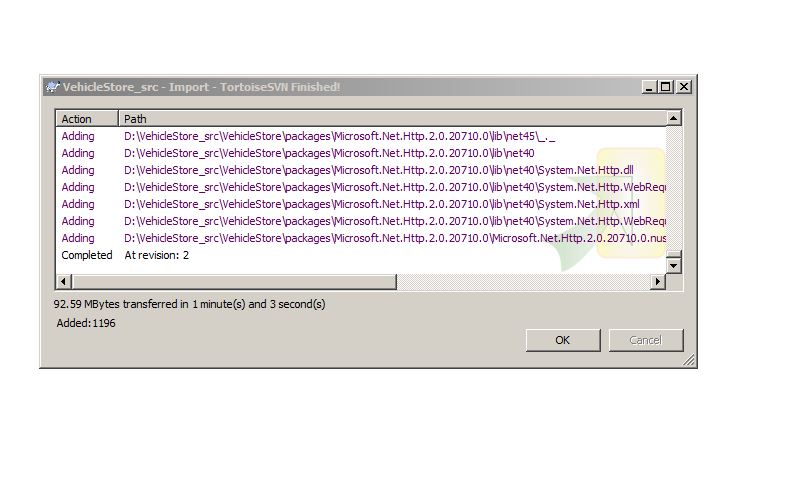


1. **You can check it on the web server of the IBM UDeploy (http://10.242.181.79:9002).**



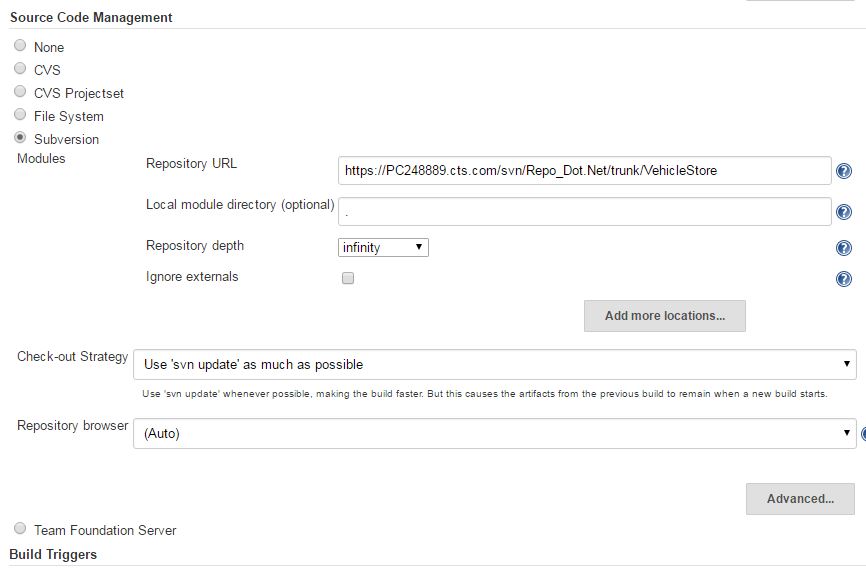
# **CI/CD automation for Dotnet project with Jenkins, SonarQube and IIS server-:**

## Check-In sample Dot-Net project into SVN



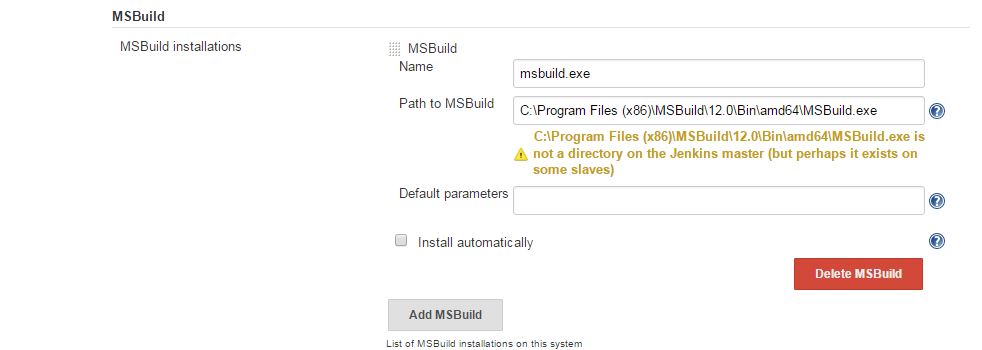
## 2) Configuring the project with Jenkins and scheduling a build with Sonar and MS-Build

2.1) Go to **Configure** option of the job, go to **Source Code Management**, and select **Subversion.** In Repository URL option, give the URL of Project you want to build.



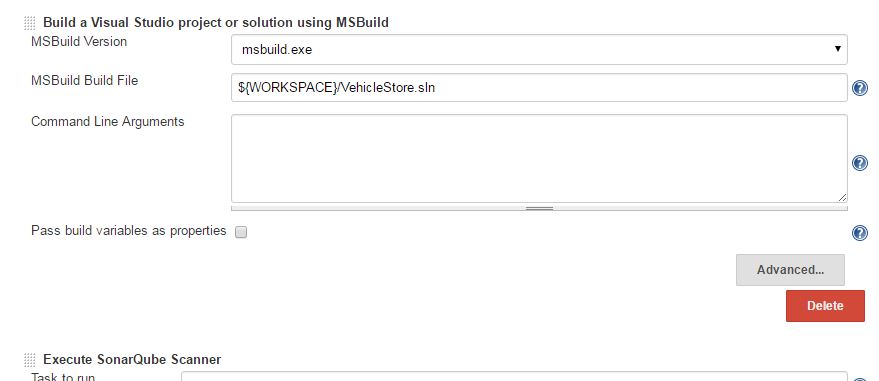
2.2) Scheduling a build with MS Build and Sonar Analysis.

To perform a MS Build, download the MS Build Plugin from Jenkins and then **Manage Jenkins** 🡺 **Configure System** 🡺 **MS Build Installation** 🡺 Name-: give\_name (.exe);   
Path to MSBuild-: give path of MS Build.



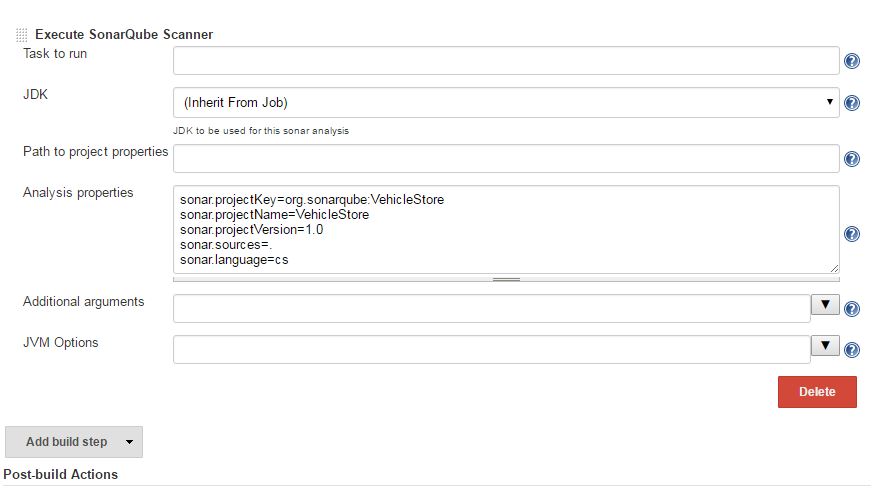
Using **Configure** option of your job, Scroll down to bottom of page to **Add build Step** and select **build a Visual Studio project or solution with MS build**. In the MS Build version select the MSBuild.exe file from dropdown.

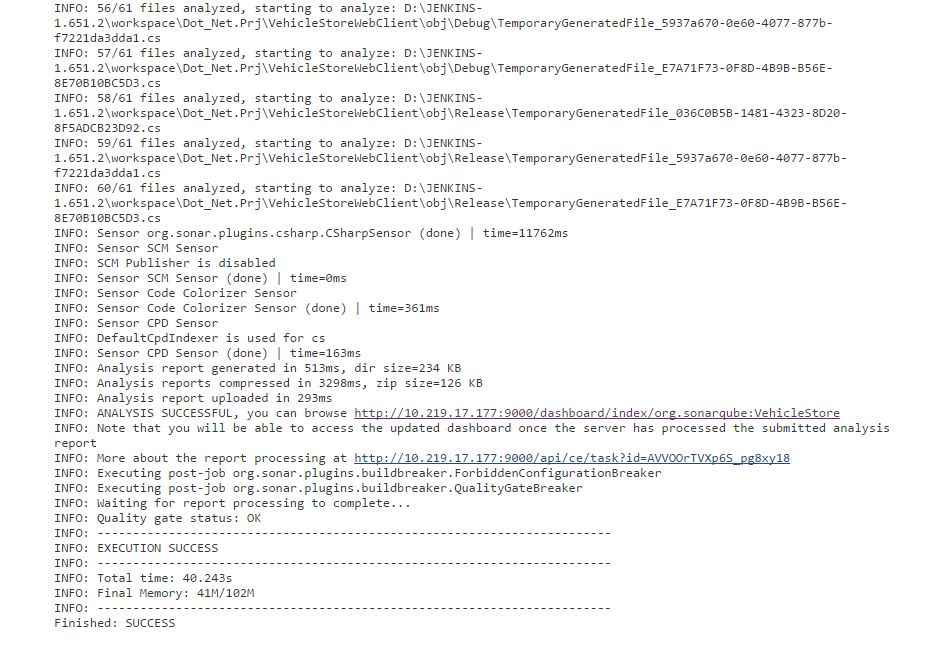
In MS Build file, select the .sln file from workspace.



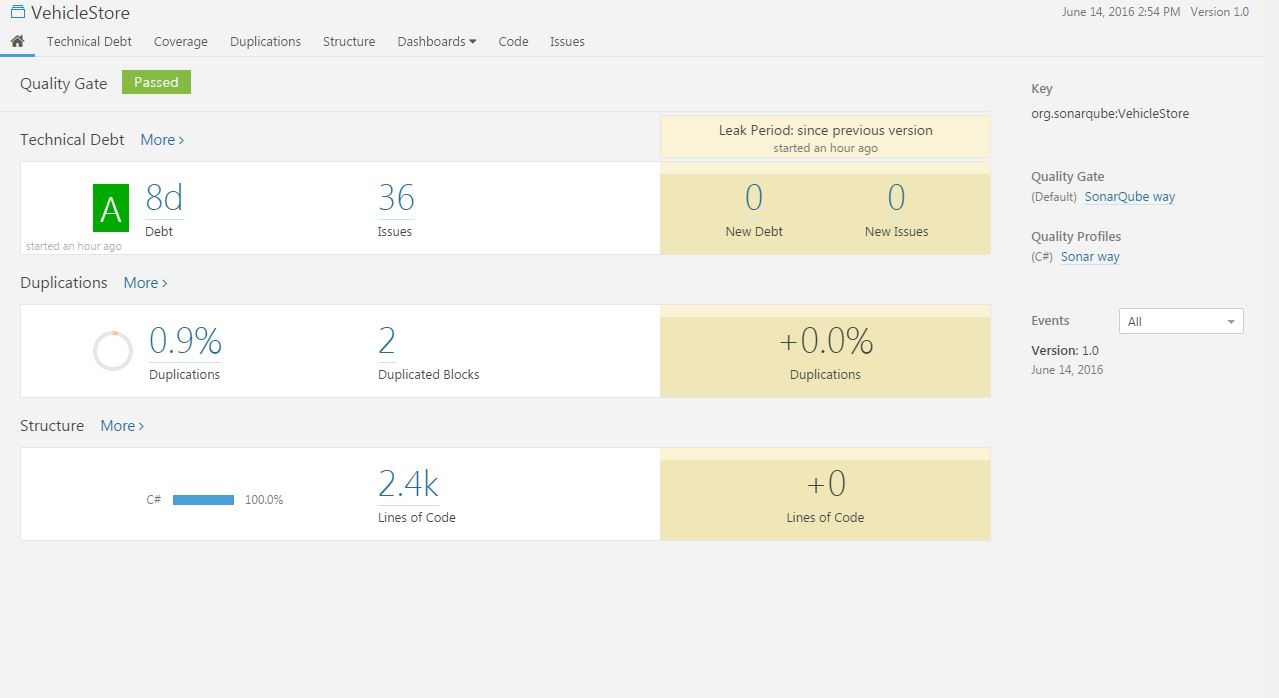
2.3) Setting sonar properties for sonar analysis

In **Execute SonarQube Scanner**, in Analysis properties give the property of sonar (or you can include the sonar-project. properties file in the root folder of project)



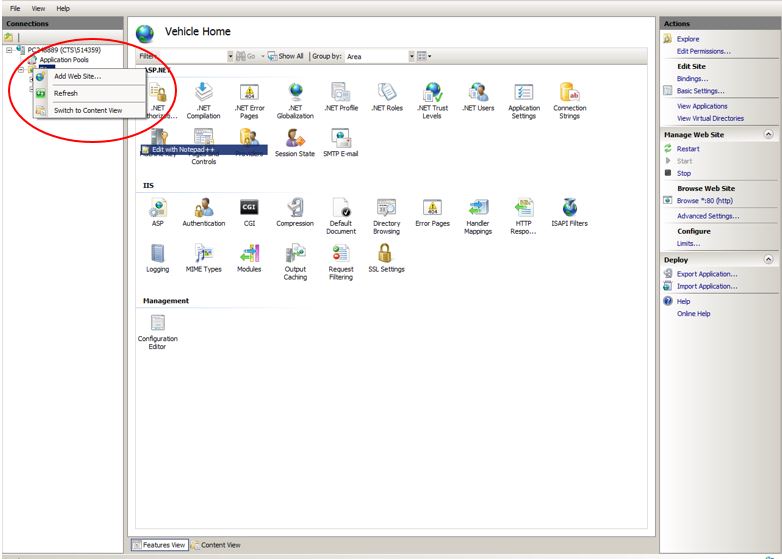
2.4) Schedule a build for your job in Jenkins from **Build Now** option. In the console output you will find analysis successful. 

2.5) Browse the URL of sonar, Quality Gate Passed in SonarQube.

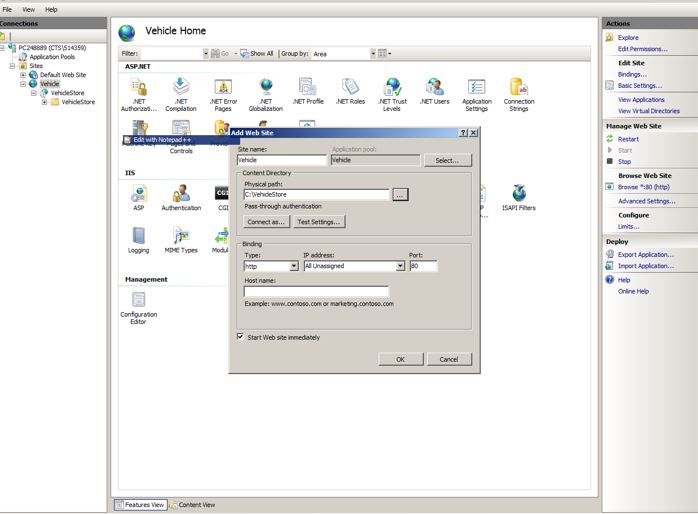


## 3) Deploy in IIS server

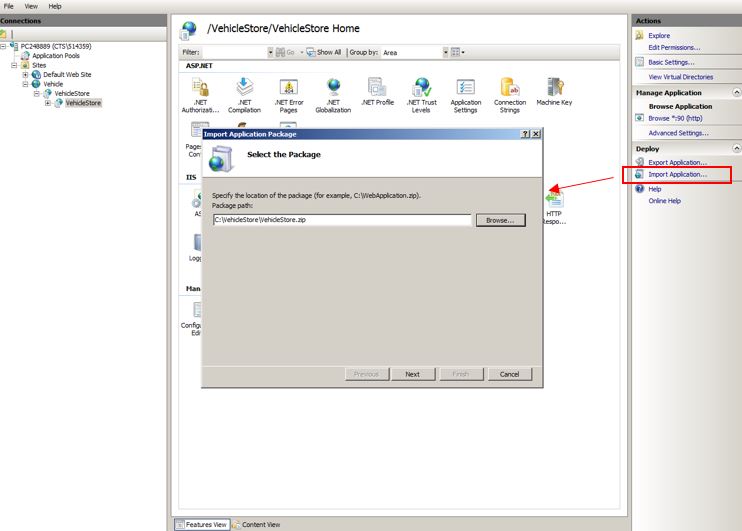
3.1) Expand the local Computer. Right Click on the **Sites**. Click **Add New Site** as shown below.

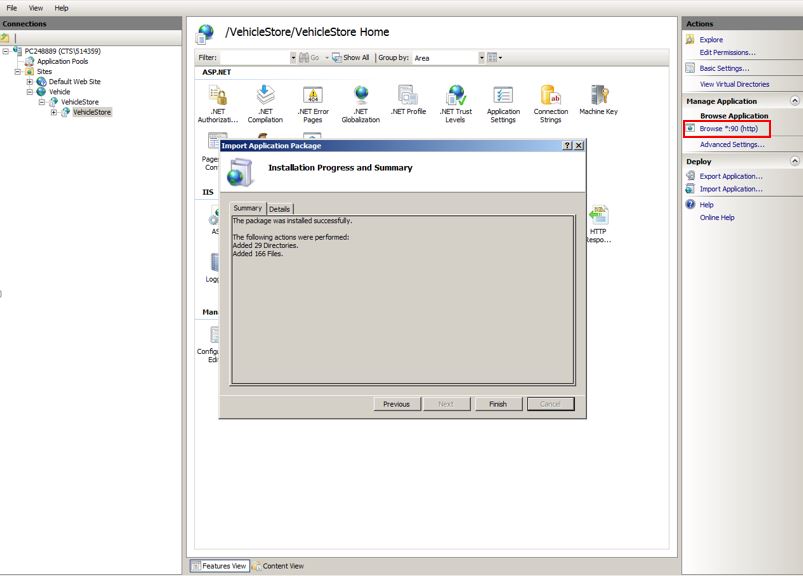


3.2) Type the name of the web site, Type the location of the web application or site. Give a port number and then press **OK**.

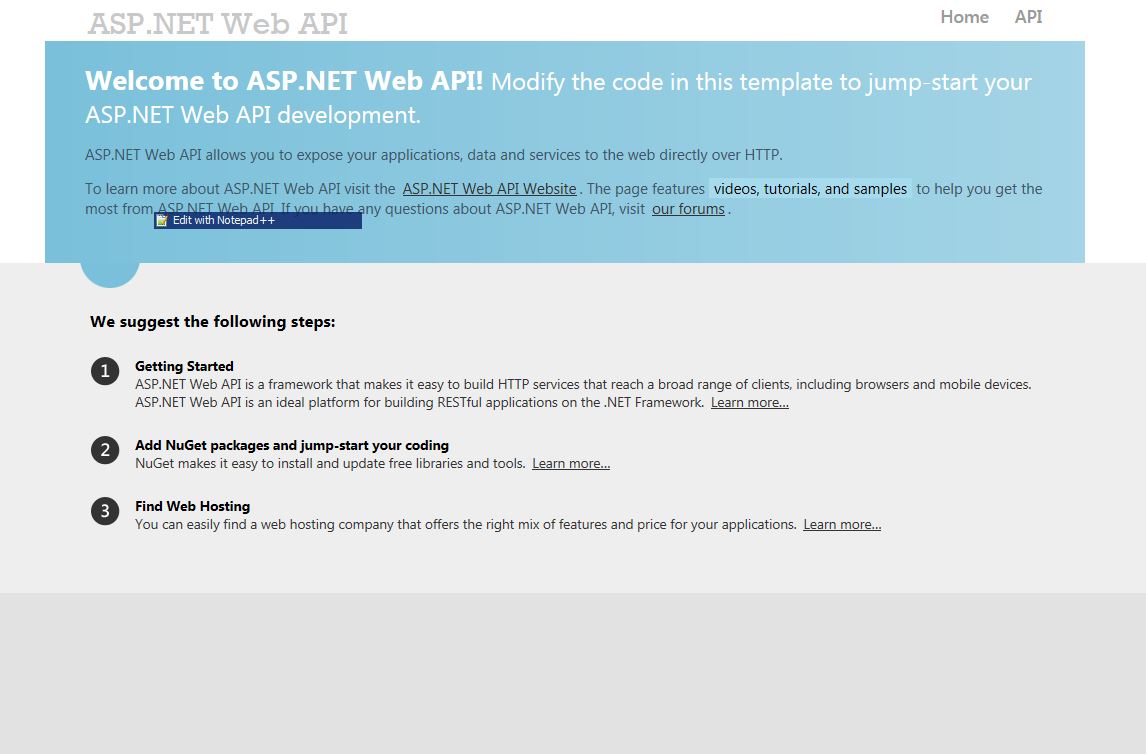


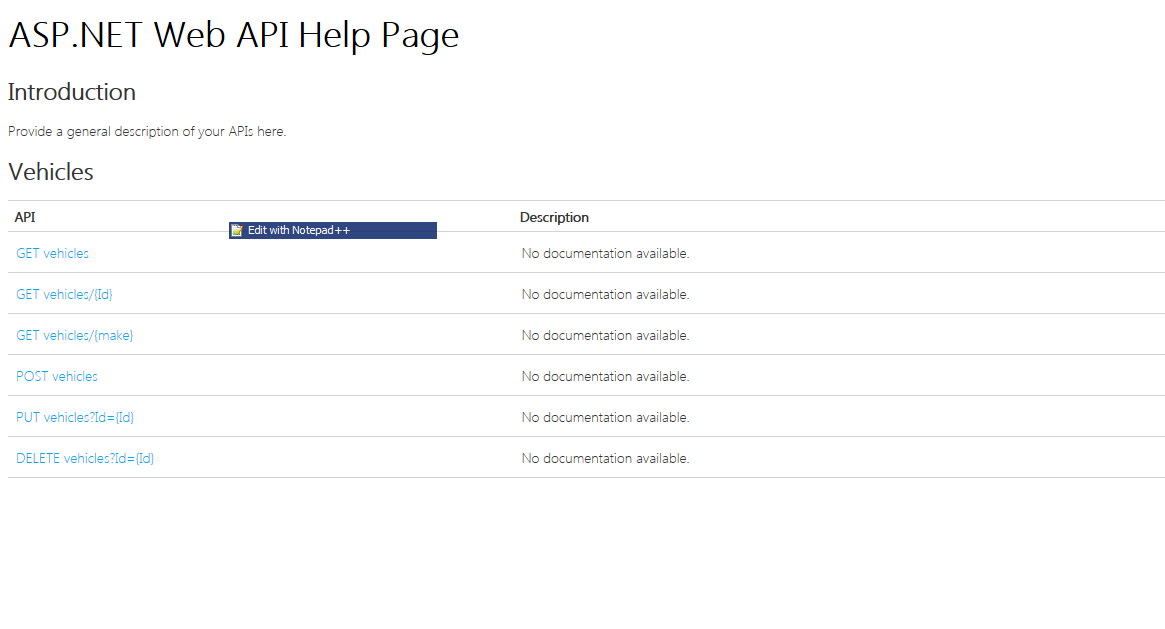
3.3) Click on **Import Application**; browse the path of Zip folder (zip folder of the project you want to deploy in IIS). Click 🡺 **Next**





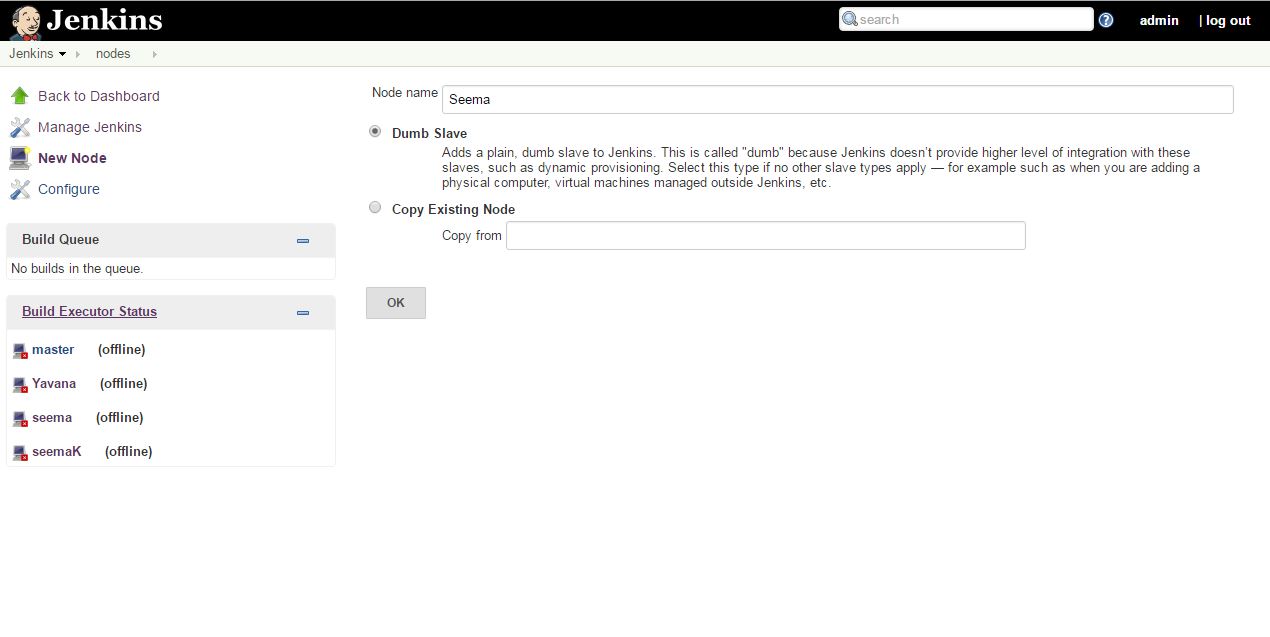
3.4) Browse the port \*. (90).http, you will find your application deployed-:



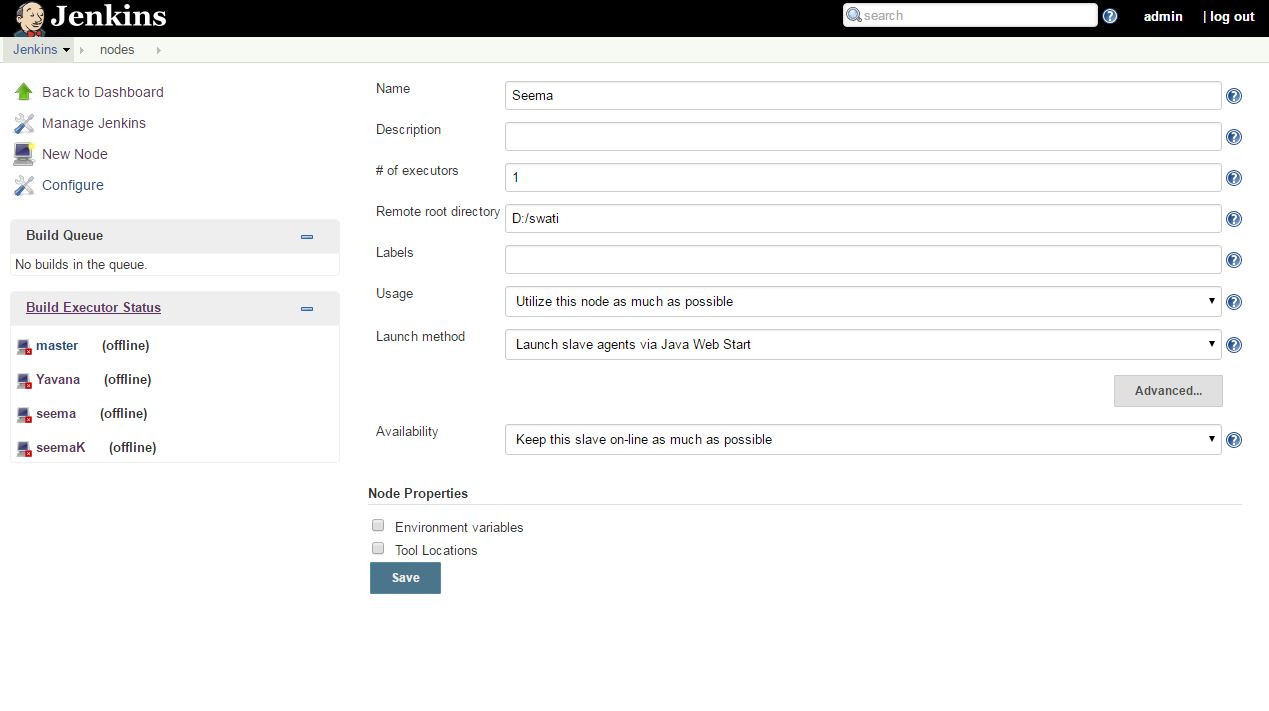


# **Creating master-Slave in Jenkins**

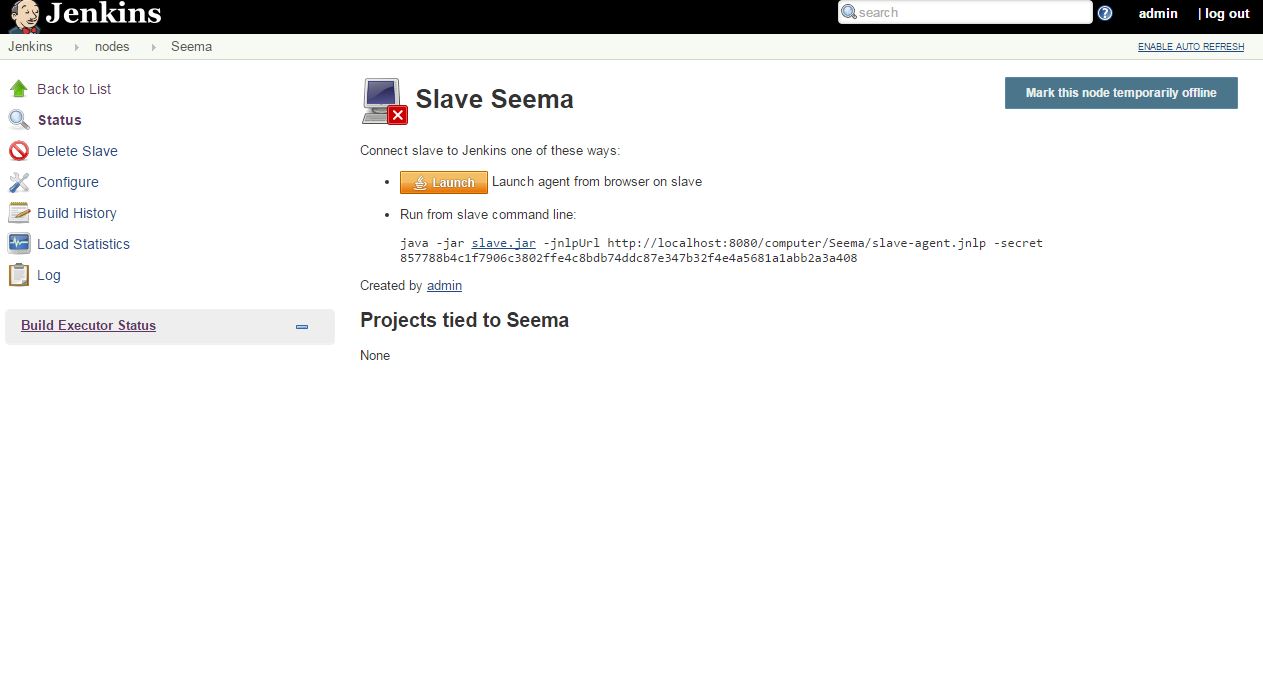
1. Creating a Node
2. In Jenkins, **Manage Jenkins** 🡺 **Manage Nodes**
3. Create a new node
4. **New Node** 🡺 **Enter Node Name;**
5. Select **Dumb Slave** 🡺Click **OK.**



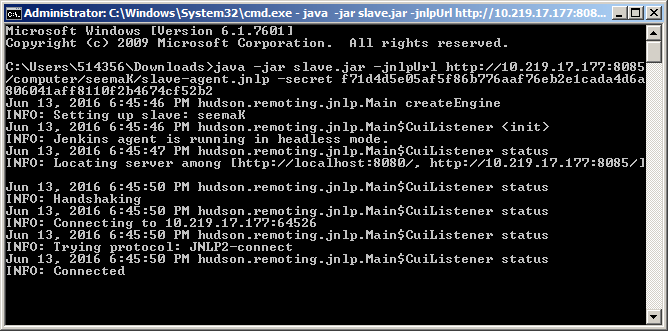
1. Set the following details-:
2. **# Of executors-:** This controls the number of concurrent builds the node can run.
3. **Remote FS root-:** This is the directory on your slave machine where Jenkins will install files necessary to run projects. Something like C:\Jenkins.
4. **Usage-:** Accept the default of Utilize this slave as much as possible.
5. **Launch method-:** For Windows, select Launch slave agents via Java Web Start.
6. **Availability-:** Accept the default of Keep this slave on-line as much as possible.



1. Now you need to connect your slave machine to the master using the following steps.
2. Open a browser on the slave machine and go to the Jenkins master server URL.
3. Go to **Manage Jenkins 🡺 Manage Nodes**, Click on the newly created slave machine.
4. Click on the **Launch** button to launch agent from browser on slave, slave.jar file is downloaded.
5. Go to slave.jar folder, run it using command prompt giving command as highlighted.

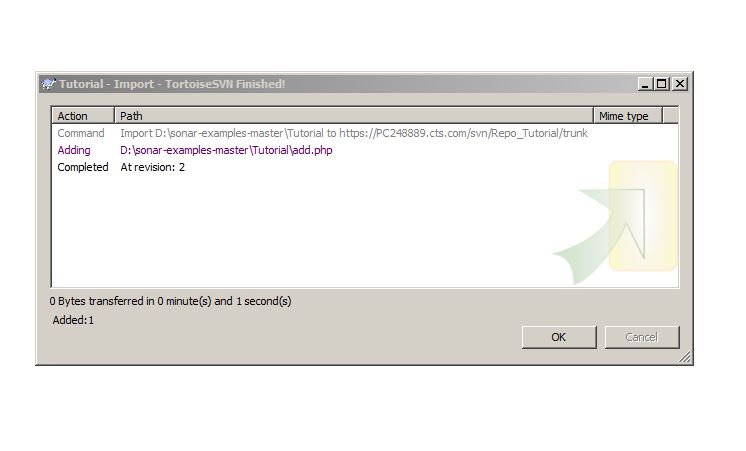


1. Running the program, you should see the Slave machine connected under Nodes.



# **CI/CD automation for php project with Jenkins, SonarQube and Apache server**

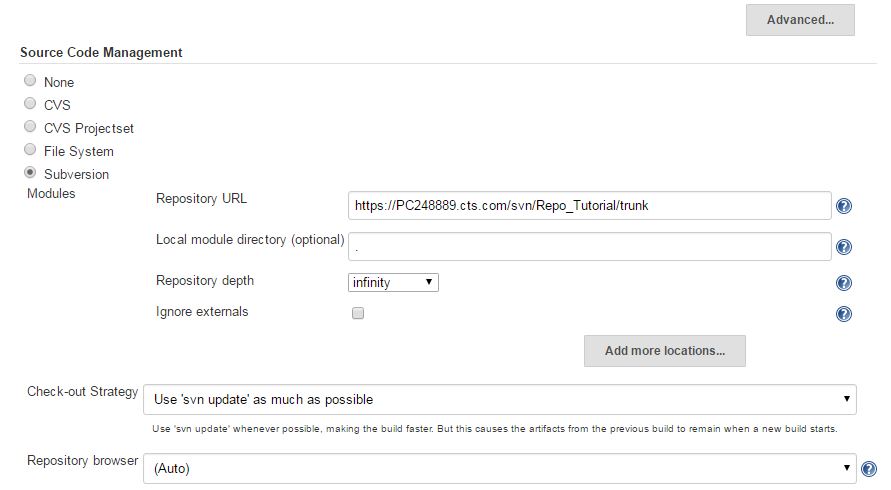
## 1) Check-In sample PHP project into SVN



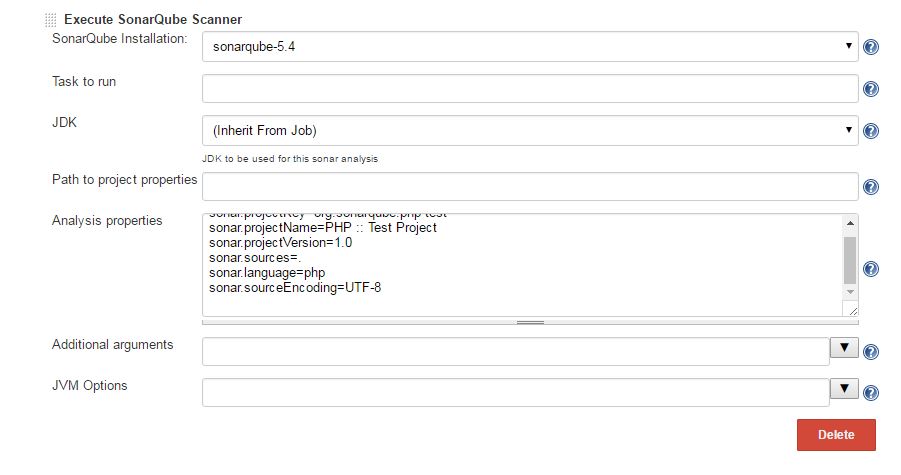
## 2) Configuring in Jenkins and scheduling a build with Sonar

2.1) Importing project from SVN repository.

Go to **Configure** option of the job, go to **Source Code Management**, and select **Subversion**. In Repository URL option, give the URL of Project you want to build.



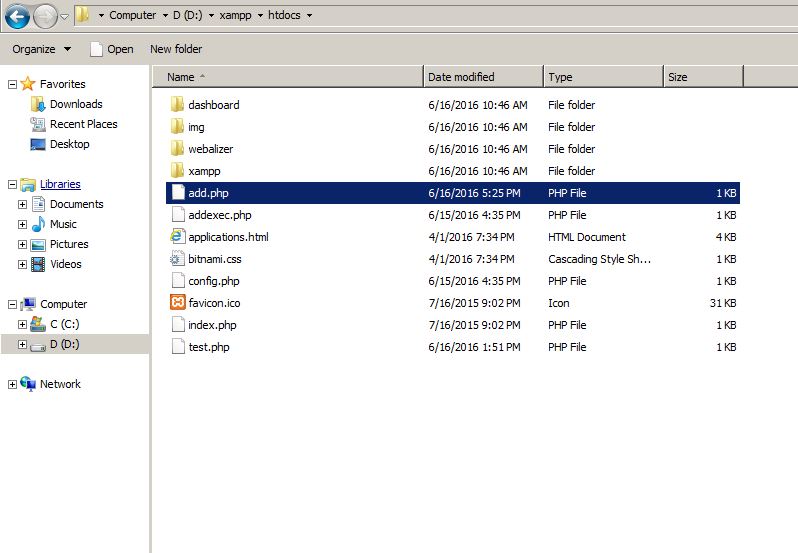
2.2) In **Execute SonarQube Scanner**, in Analysis properties give the property of sonar (or you can include the sonar-project. properties file in the root folder of project)



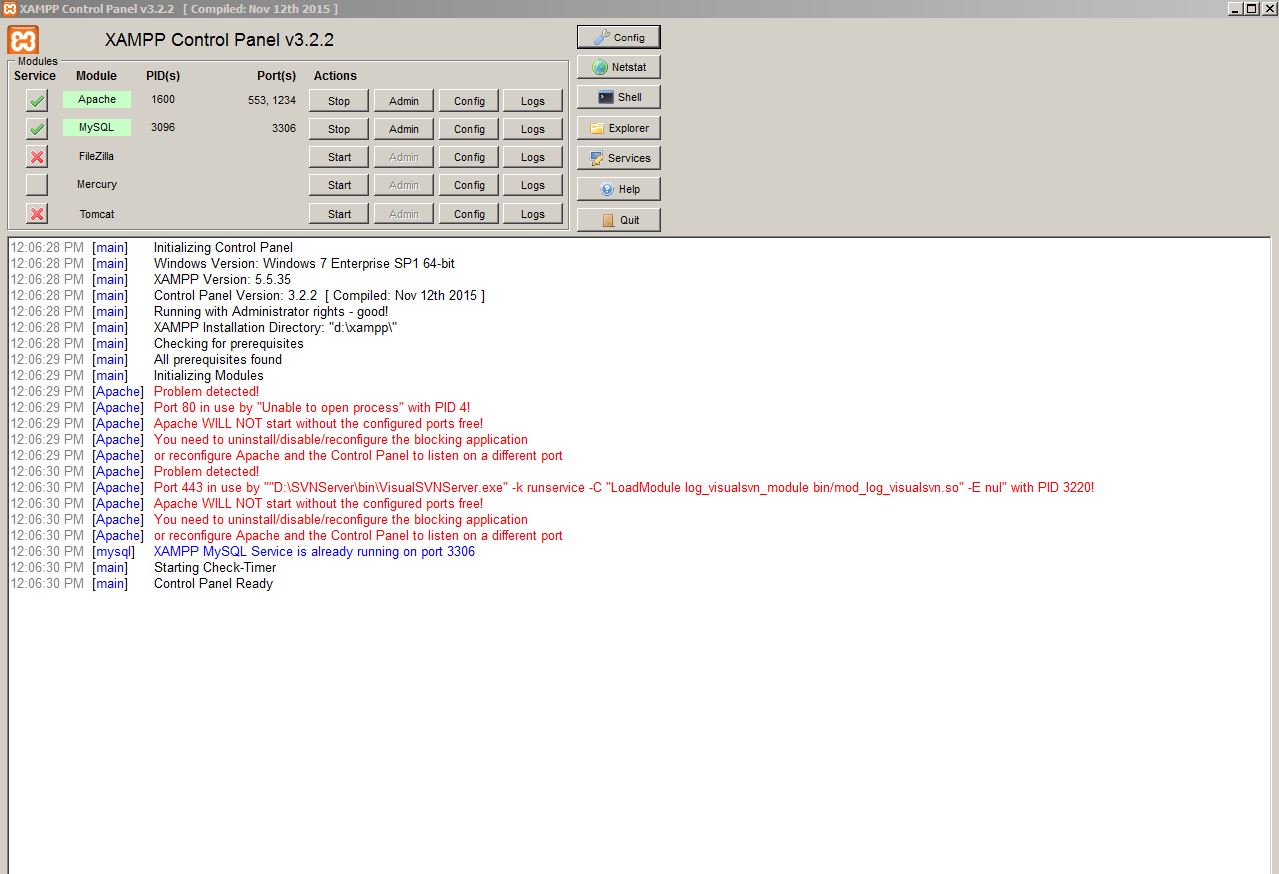
2.3) In **Add Build Steps** 🡺 **Execute Windows batch command**

Command-: Here pass the pass the command that will copy the .php file to **Xamps htdocs** folder.





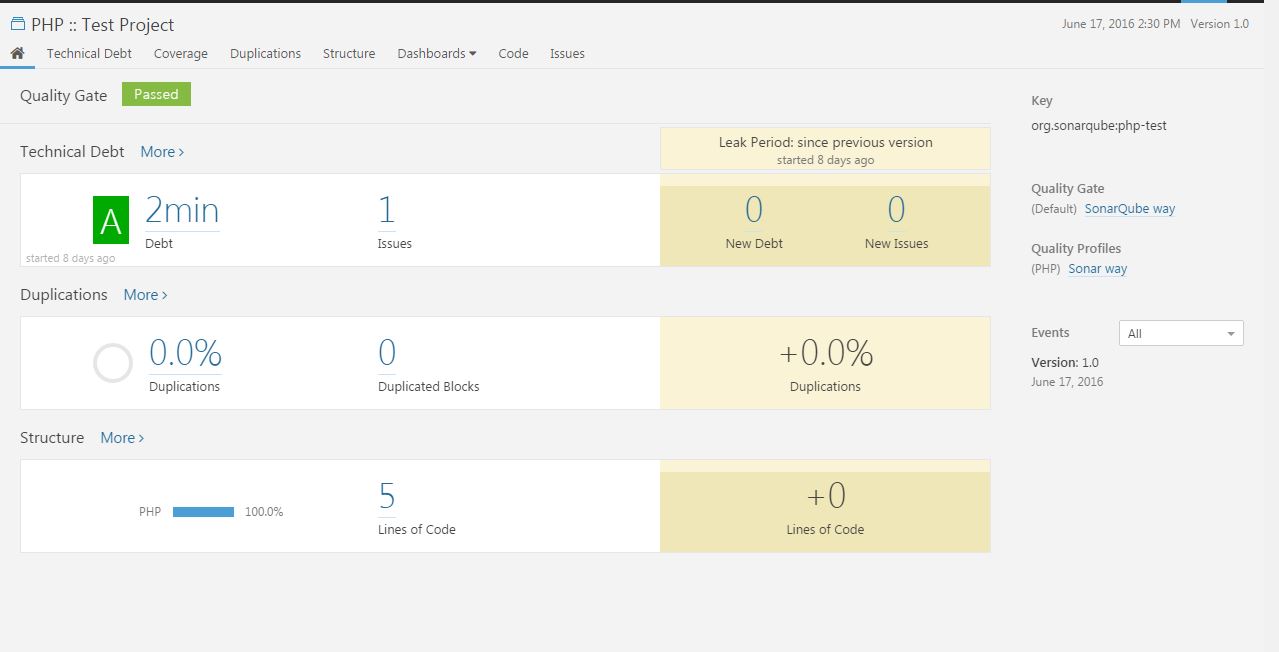
[XAMPP](http://www.apachefriends.org/en/xampp.html) is an easy to install Apache distribution containing MySQL, PHP and Perl. The **XAMPP Control Panel** allows you to manually start and stop Apache and MySQL, or install them as services.



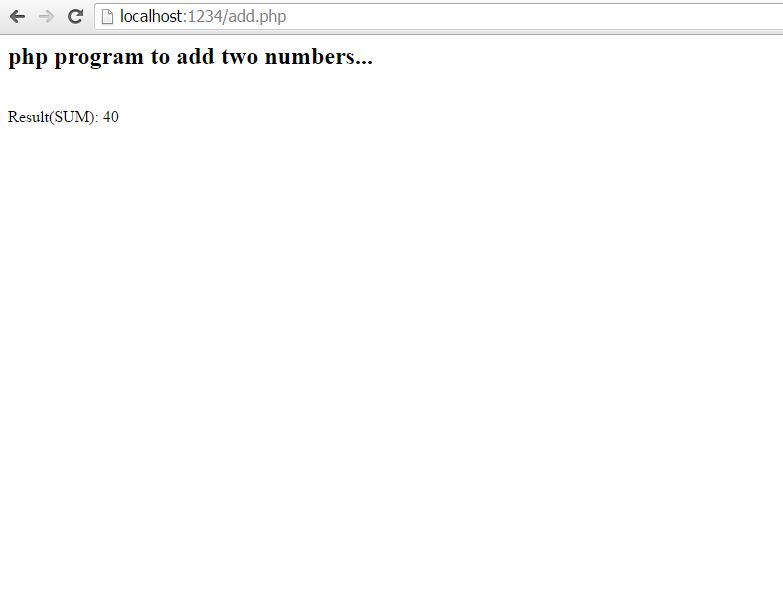
2.4) Schedule a build for your job in Jenkins from **Build Now** option. In the console output you will find analysis successful.



2.5) browse the URL of sonar, Quality Gate Passed in SonarQube-:

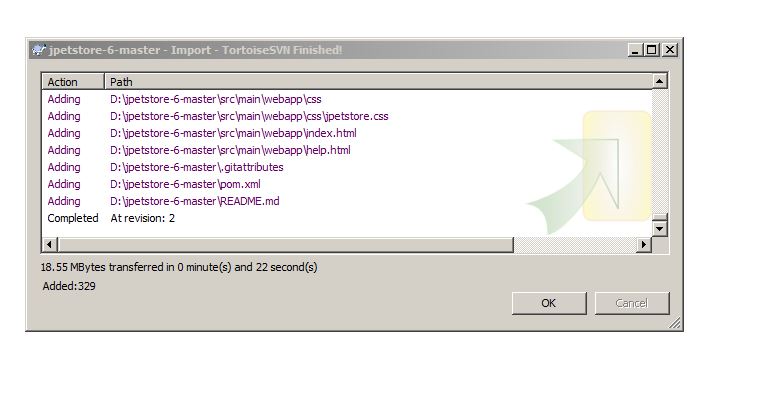


2.6) Browse localhost-:



# **CI/CD automation for Java project with Jenkins, SonarQube and Tomcat server & IBM UDeploy**

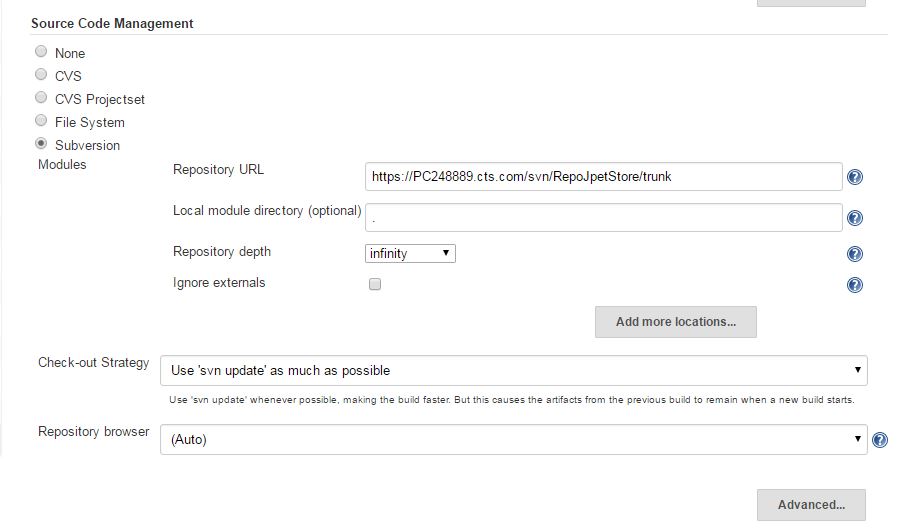
## 1) Check-In sample Java project into SVN.

****

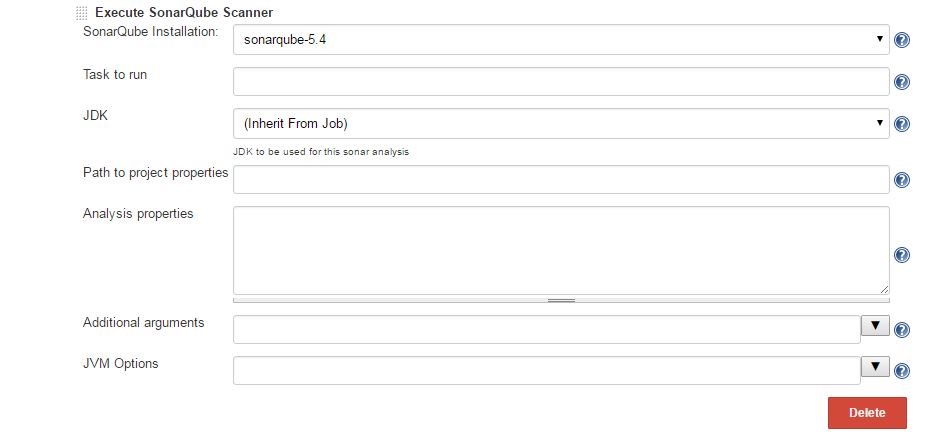
## 2) Configuring in Jenkins and scheduling a build with Sonar

2.1) In the **Configure** option of the job, go to **Source Code Management**, select **Subversion**.

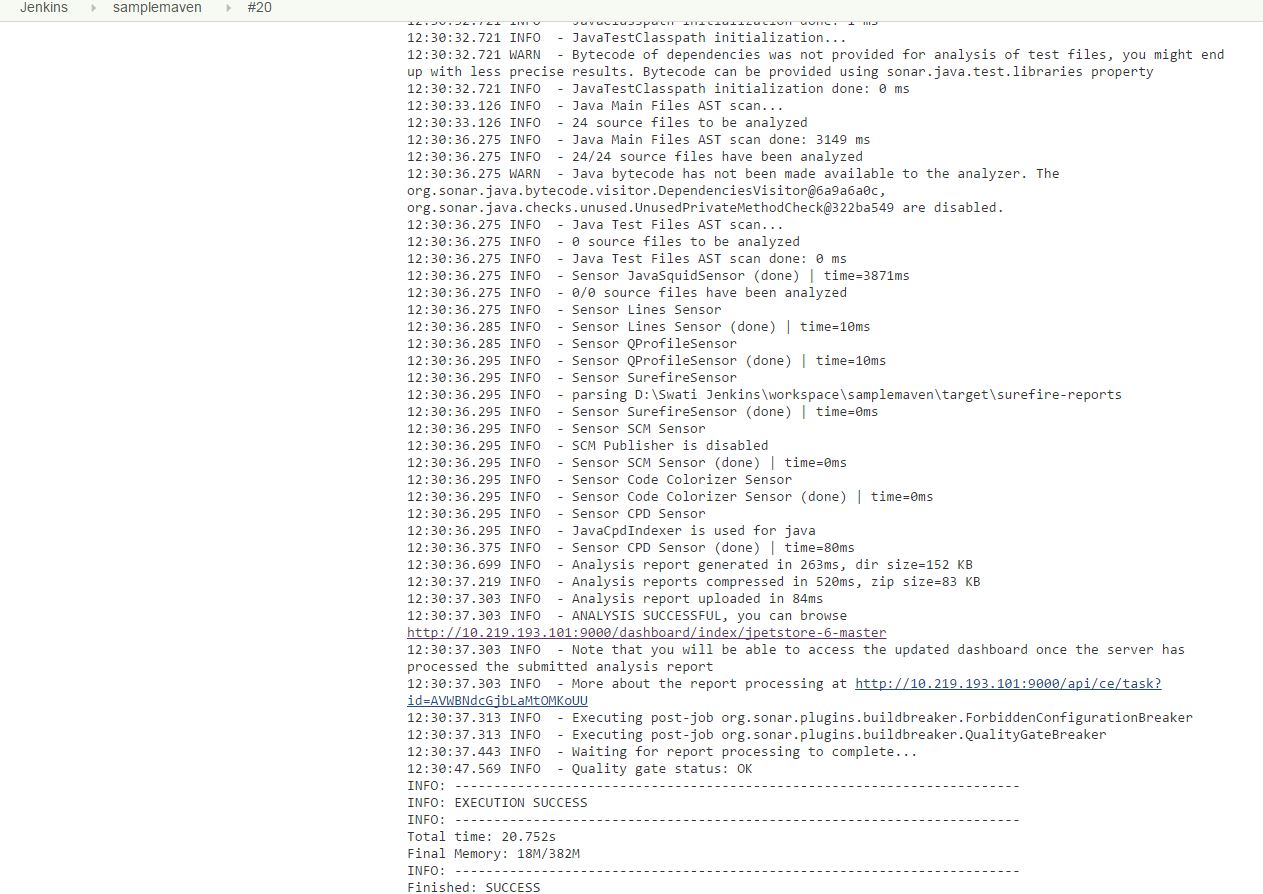
In Repository URL option, give the URL of Project you want to build.



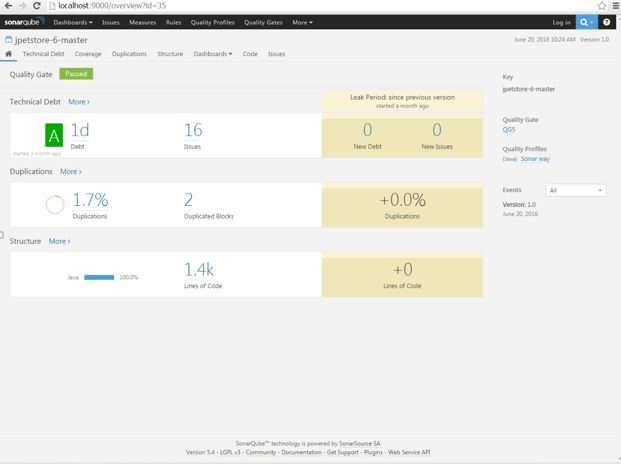
2.2) To invoke sonarqube for sonar analysis, select the Sonar you want to use from **SonarQube Installation** dropdown.



2.3) Schedule the **Build Now** option for your job in Jenkins. In the console output you will find analysis successful, indicating war file has been deployed.



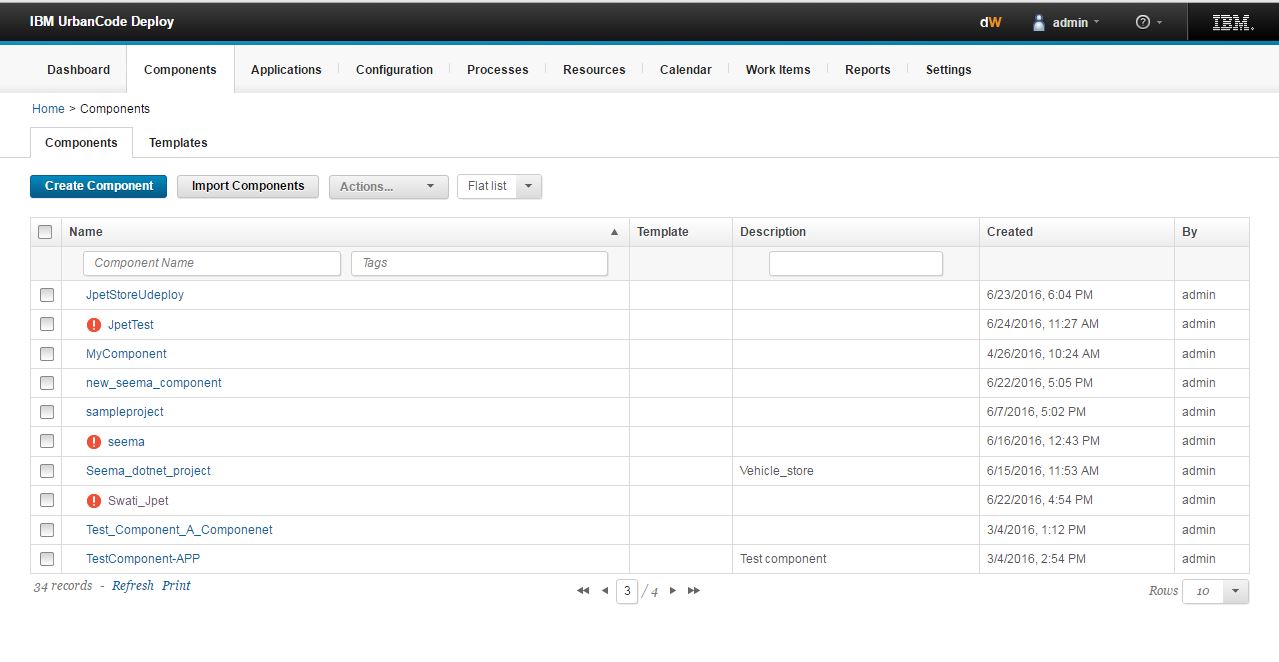
2.4) To see the sonar analysis of the job, browse the sonar URL from Jenkins console Output-



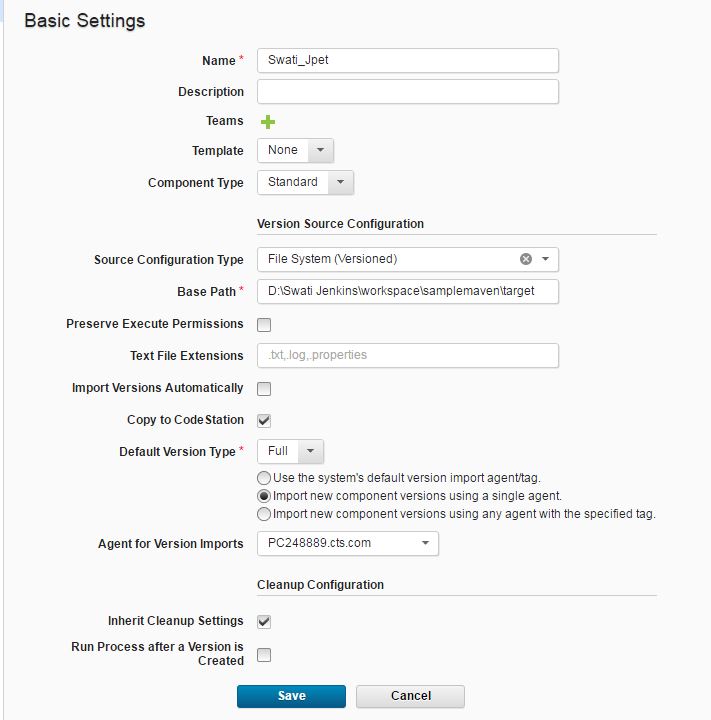
## 3) IBM UrbanCode Deploy

3.1) create the web component:

1. Click the **Components** tab and then click **Create Component**.

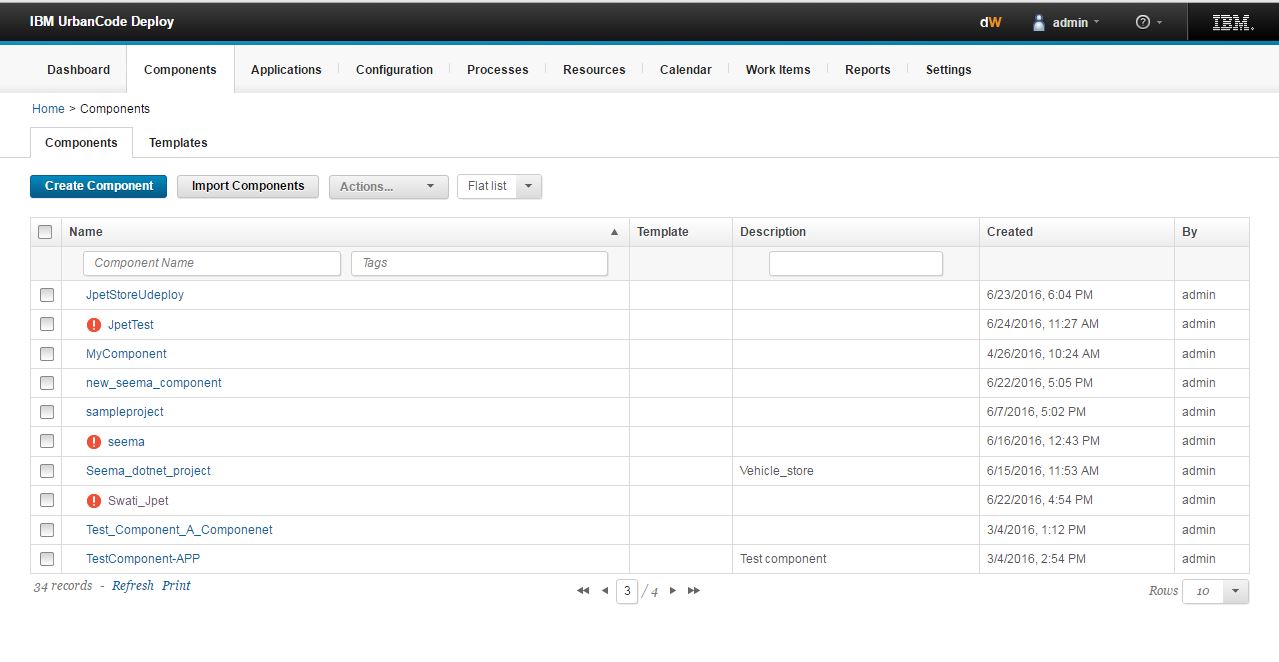


1. In the **Name** field, type (component\_name), Swati\_Jpet.
2. In the **Source Configuration Type** list, select **File System (Versioned)**.
3. In the **Base Path** field, specify the location of the app folder on the server, such as D:\Swati Jenkins\workspace\samplemaven.
4. Under **Default Version Type**, click **Import new component versions using a single agent** and then select your agent in the **Agent for Version Imports** list
5. Accept the default values for the other fields on the page.



1. Click **Save**.

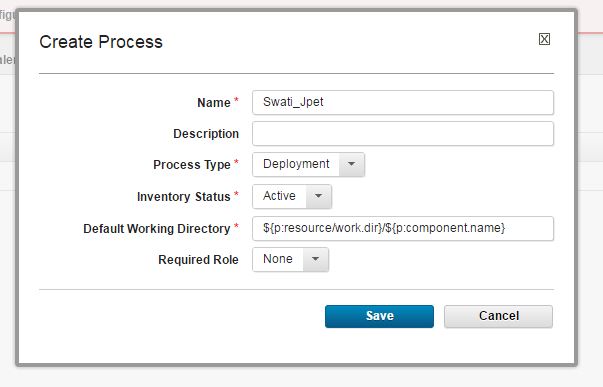
Component is created as shown.



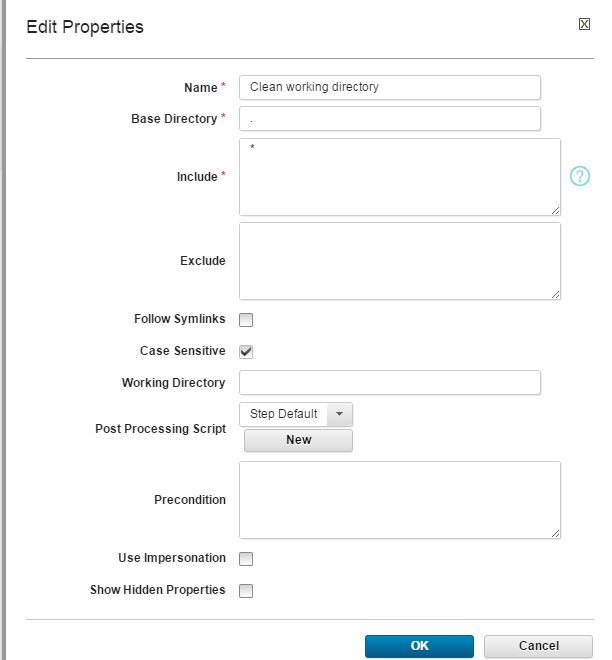
3.2) From the Components tab, click the Swati\_Jpet

Click **Processes** and then click **Create Process**.

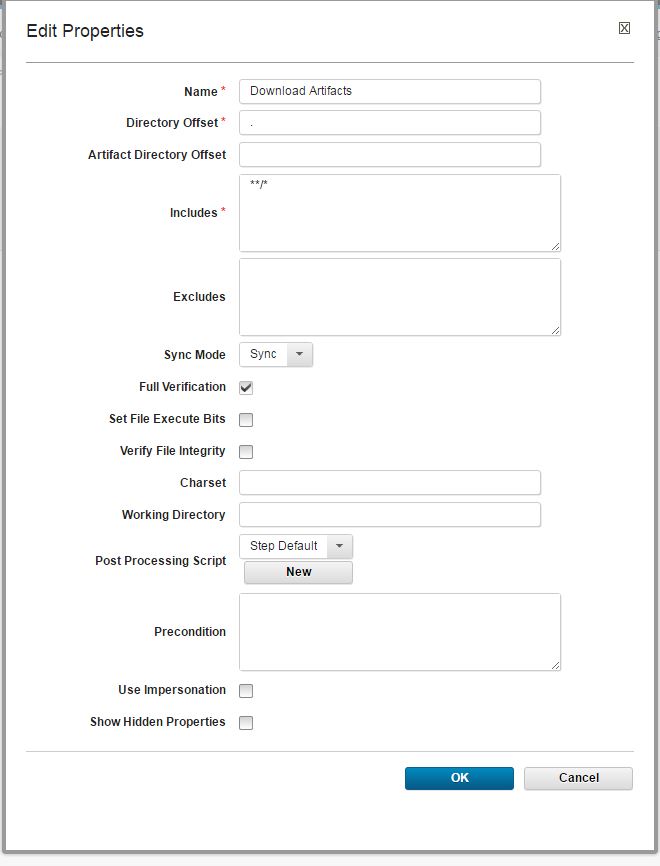
1. In the Create Process window, specify the name.
2. In the **Process Type** list, select **Deployment**. This list has other options for processes that uninstall or configure components.
3. Accept the default values for the other properties and click **Save**.



1. From the list of processes, click the process you created (Swati\_Jpet). The process opens in the process editor. This editor shows the steps in the process in a graphical form, as a flowchart. The **Start** and **Finish** boxes represent the beginning and the end of the process. From here, you add steps to the process and link them between the **Start** and **Finish** steps to show the order of the steps.
2. Add a step to clean any files out of the working directory. The process runs within a working directory. To ensure that you are using the most recent version of the files in the web component, run a command to clean the working directory:
   1. At the left of the process editor, under **Step Palette**, expand **Utilities** > **FileUtils**. The **Step Palette** menu shows the available steps. You can explore the trees to find steps, or you can type into the search box at the top of the window.
   2. Under the **FileUtils** tree item, click and drag the **Delete Files and Directories** step to the process editor. The Edit Properties window opens, showing the properties for the step. Some of these properties are unique to the step and others are the same for all steps.
   3. In the **Name** field, specify the name to be clean working directory.
   4. In the **Base Directory** field, specify a single period (.).
   5. In the **Include** field, specify an asterisk (\*).
   6. Accept the defaults for the other properties and then click **OK**.



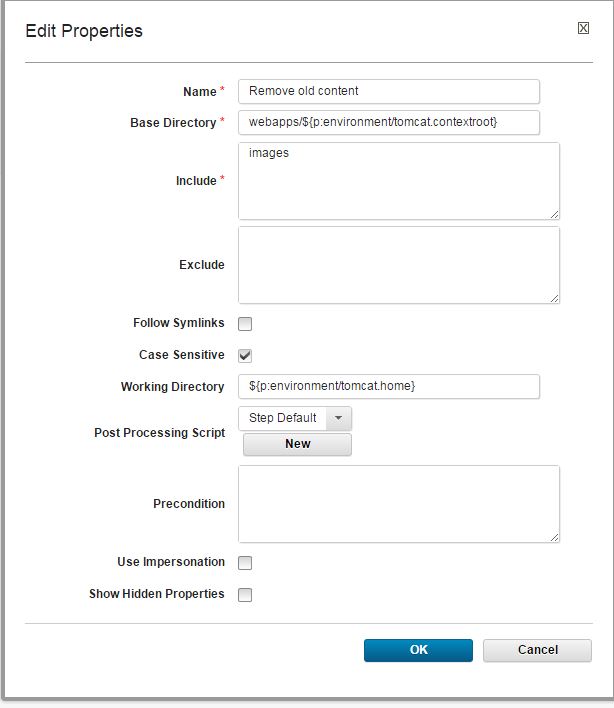
1. Add a step to download the latest version of the component artifacts. The **Download Artifacts**step is used in most deployment processes. This step downloads the specified version of the component artifacts to the target computer. When you run the process, you will specify whether to use the most recent version of the component artifacts or a specific version.
2. Under **Step Palette**, expand **Repositories** 🡺 **Artifact** 🡺 **IBM UrbanCode Deploy** and drag the **Download Artifacts** step to the process editor.
3. In the Edit Properties window, accept the default values and then click **OK**.



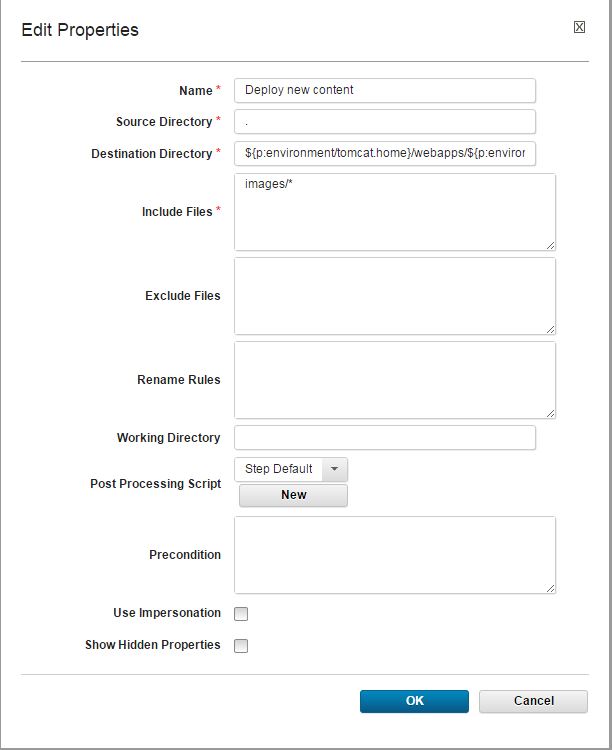
1. Add a step to **remove the old web content** from the server:
2. Expand **Utilities** > **FileUtils** and drag another **Delete Files and Directories** step to the process editor.
3. In the Edit Properties window, specify the name to be Remove old content.
4. In the **Base Directory** field, specify the following directory:

**Webapps:/${p:envioronment/tomcat.contextroot}**

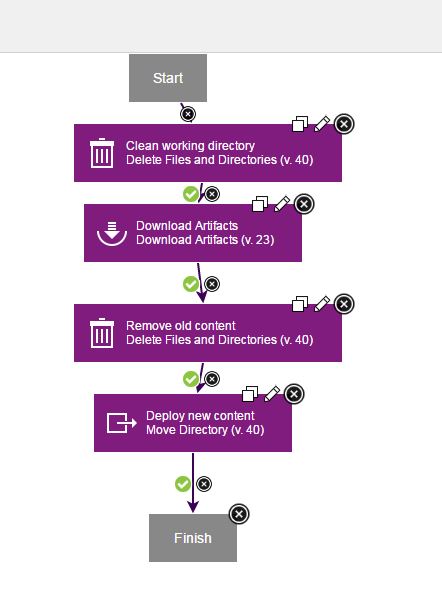
1. The variable represents the location where Tomcat stores your deployed files. You specify a value for this variable later.
2. In the **Include** field, specify the following directory: **images**
3. In the **Working Directory** field, specify the following variable :**${ p: environment/tomcat.home}.** This variable represents the main folder of the Tomcat web server. You will specify a value for this variable later.
4. Accept the default values in the other fields and click **OK**.



1. Add a step to deploy the new content to the server. This step copies the component artifacts to the application server.
2. Expand **Utilities** > **FileUtils** and drag a **Move Directory** step to the process editor.
3. In the Edit Properties window, specify the name to be Deploy new content.
4. In the **Source Directory** field, specify a single period (.).
5. In the **Destination Directory** field, specify the following directory: ${p:environment/tomcat.home}/webapps/${p:environment/tomcat.contextroot}
6. In the **Include Files** field, specify the following code: images/\*
7. Accept the default values in the other fields and click **OK**.



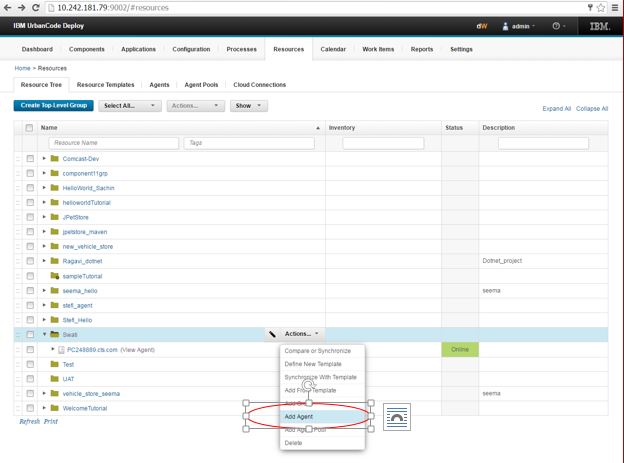
1. Now the process has four steps, as in the following figure:



1. Save the process by clicking the **Save** http://www.ibm.com/support/knowledgecenter/en/SS4GSP_6.2.1/com.ibm.udeploy.tutorial.doc/images/save_button.gif icon under **Tools**.

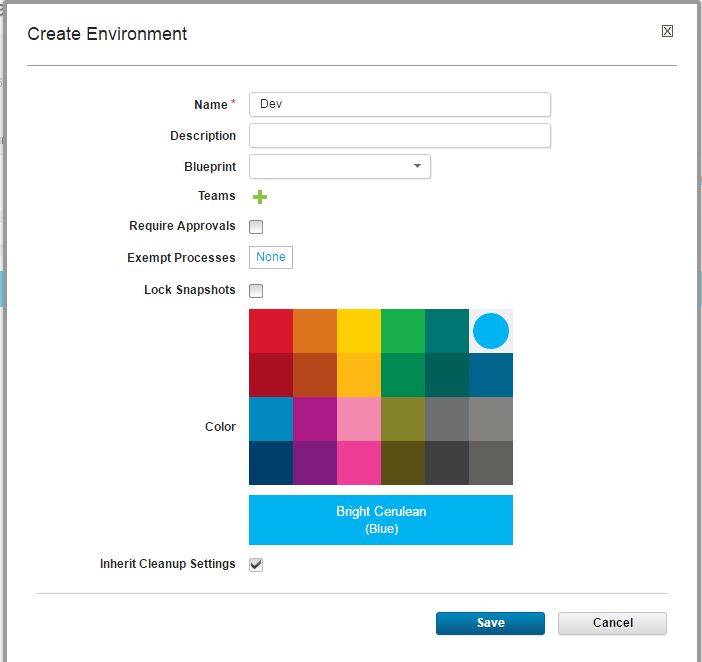
3.3) Click **Resources** 🡺 **Create Top level Group**

**Add Agent 🡺 Add Component**

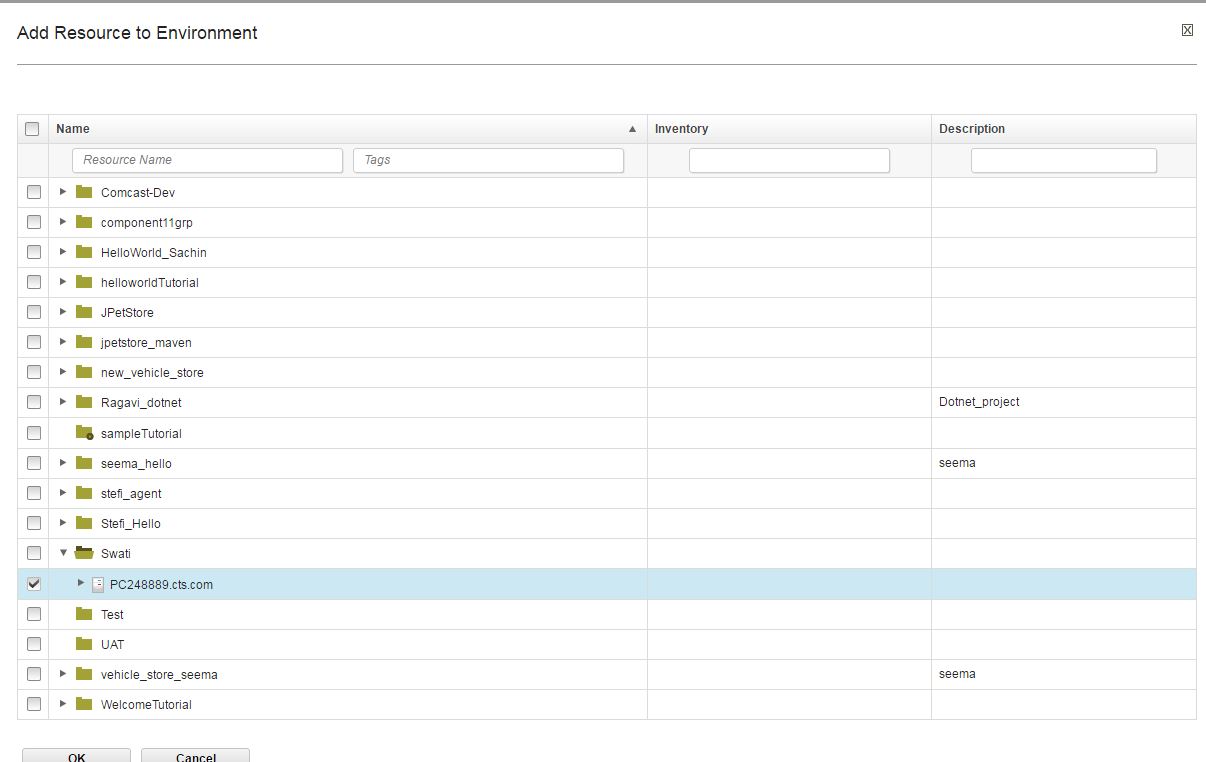


3.4) Click **Application 🡺 JpetStore\_Swati 🡺 Create Enviornment**

Give Name-: Dev 🡺 **Save**

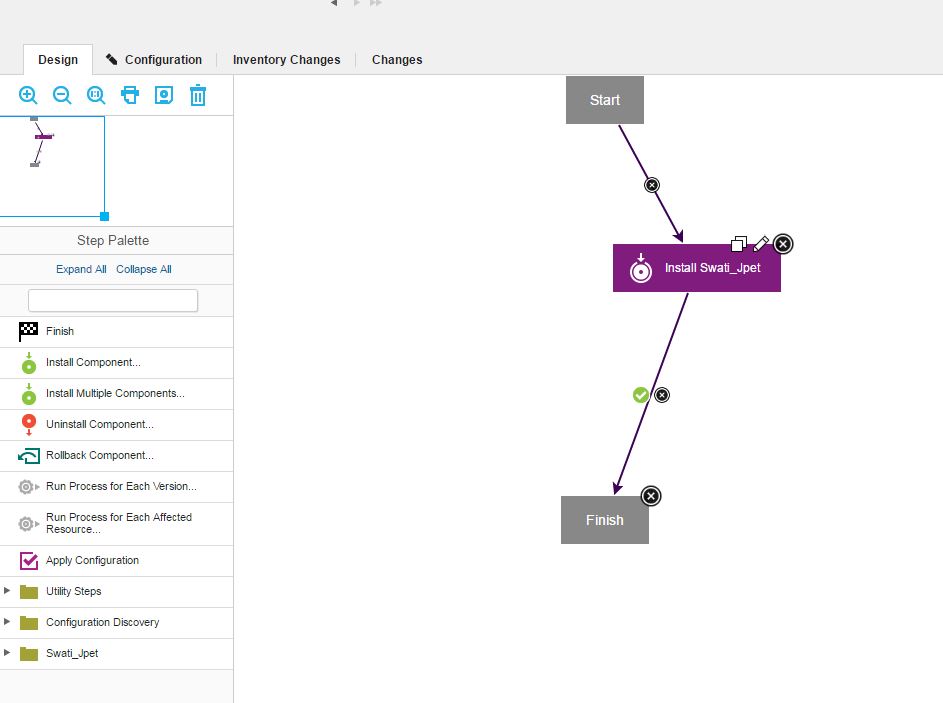
****

Click **Dev** 🡺 Select your agent



3.5) Click **Application 🡺 JpetStore\_Swati 🡺 Component🡺 Add Component**(select your component)

🡺**Processes🡺 Create Processes 🡺 Edit**

****

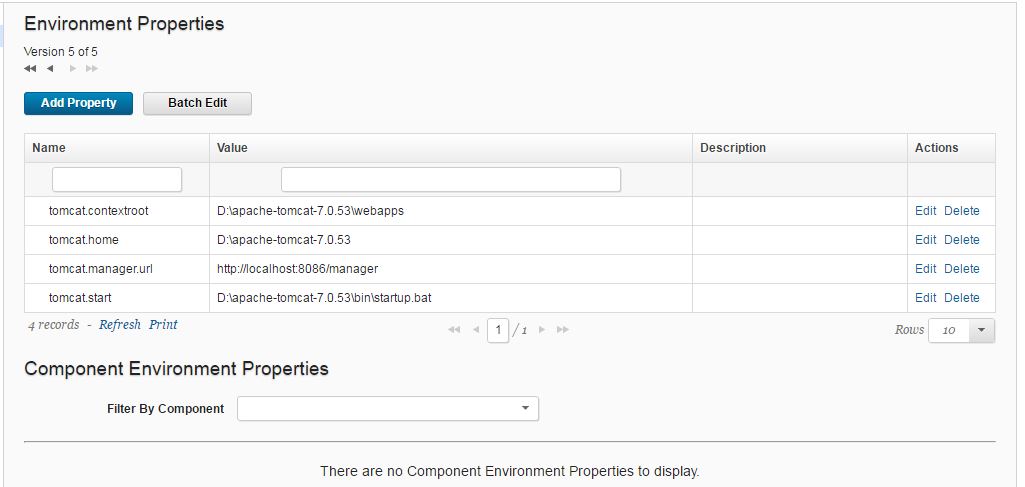
3.6) Click **Application 🡺 JpetStore\_Swati** (select your application) 🡺 **Configuration🡺 Enviornment Properties 🡺** set properties as shown-:

Tomcat.home-: The Tomcat home folder on the target computer

tomcat.contextroot -: The Tomcat webapp folder on the target computer, which is an identifier in the application URL

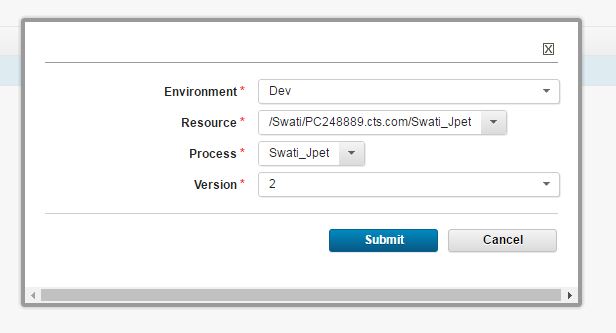
tomcat.manager.url -: The location of the Tomcat manager application

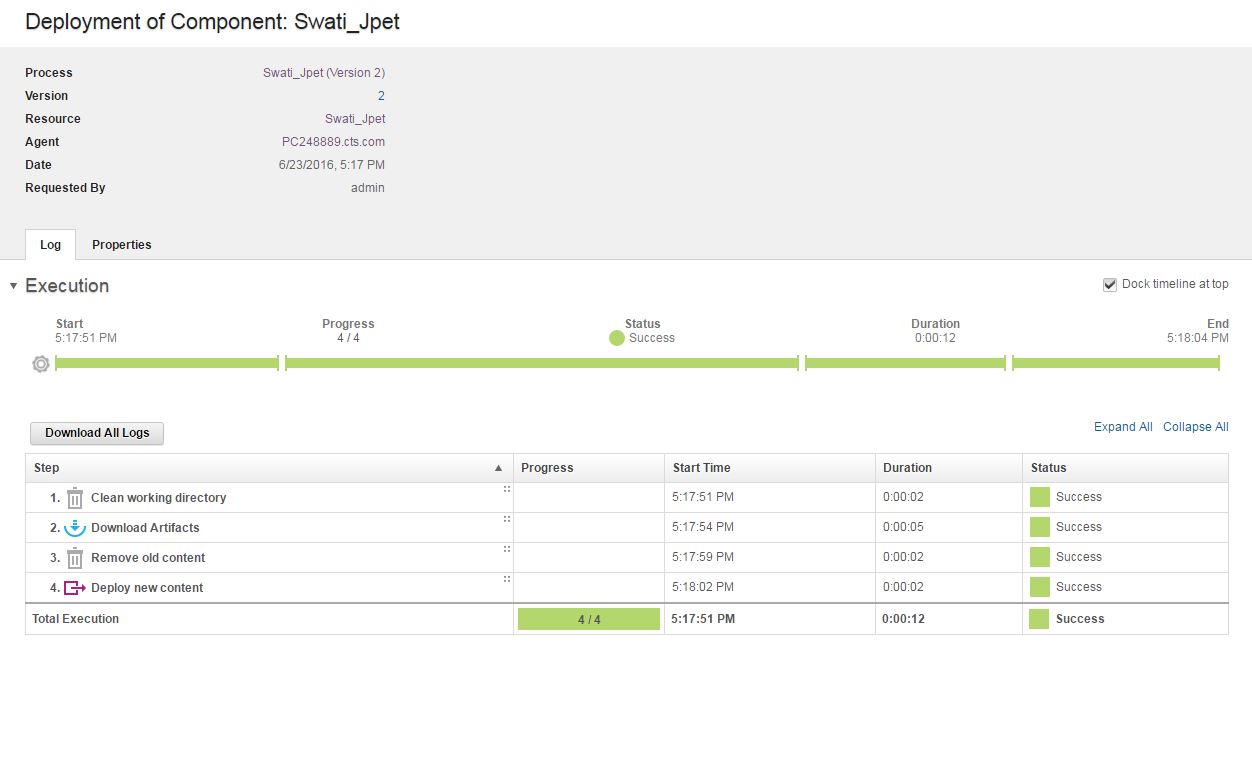
tomcat.start -: The location of the startup script for Tomcat

****

3.7) Click **Application** 🡺 **Components** 🡺 **Run Process**

Select the Environment and Version.

****

****

## 4) Browse your application

4.1)If you browse the URL of Tomcat, you will find your deployed application-:

