**Deployment of Various Projects on Different Server**

**(Projects used are of Java, PHP & Dotnet)**

**Contents**

[**Installing the UDeploy Agent** 3](#_Toc454543635)

[**Installing the Jenkins agent (Master-Slave Configuration)** 6](#_Toc454543636)

[**Deploying jpetstore in the Tomcat with the help of the Jenkins** **Plugin** 9](#_Toc454543637)

[1. Check-in the Jpetstore project in a repository in the SVN. 9](#_Toc454543638)

[2. Committing the code in the subversion 9](#_Toc454543639)

[3. Running the project on the slave node: 10](#_Toc454543640)

[4. Pre Steps (SonarQube Scanner Configuration in Jenkins) 11](#_Toc454543641)

[5. Post Build Action (Deployment in the Tomcat Server) 12](#_Toc454543642)

[6. Build the Jenkins job. 13](#_Toc454543643)

[7. Tomcat Deployment 16](#_Toc454543644)

[**Deployment of the php project 19**](#_Toc454543645)

[1. Check-in php code in SVN 19](#_Toc454543646)

[2. Analyzing php project in Jenkins with Sonar integration 20](#_Toc454543647)

[3. Deployment 24](#_Toc454543648)

[**MS Deploy using UDeploy 25**](#_Toc454543649)

[1. Pre-requisites 25](#_Toc454543650)

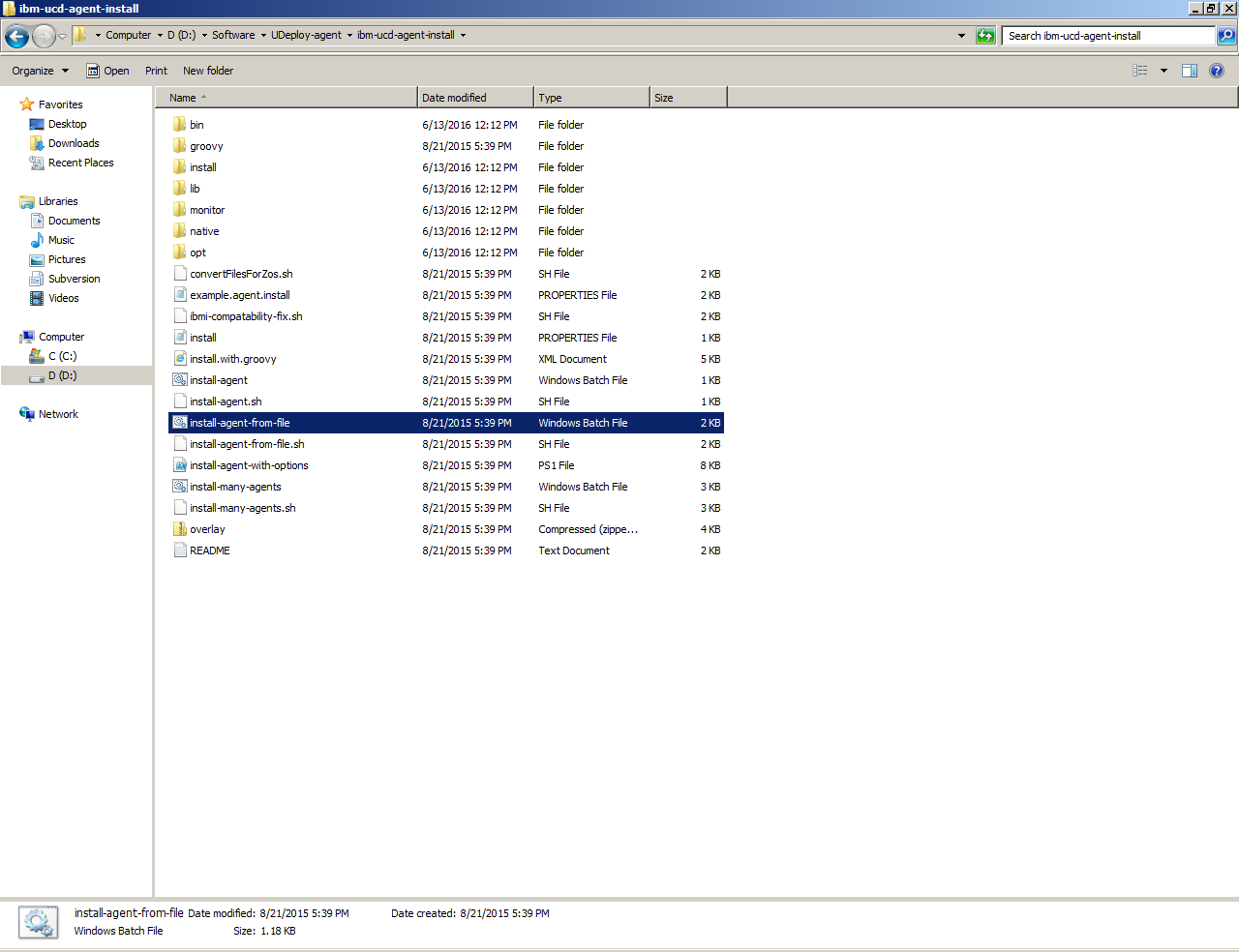
[2. Steps to be followed in IBM UDeploy to deploy the project 25](#_Toc454543651)

[**Deployment In IIS Server 39**](#_Toc454543653)

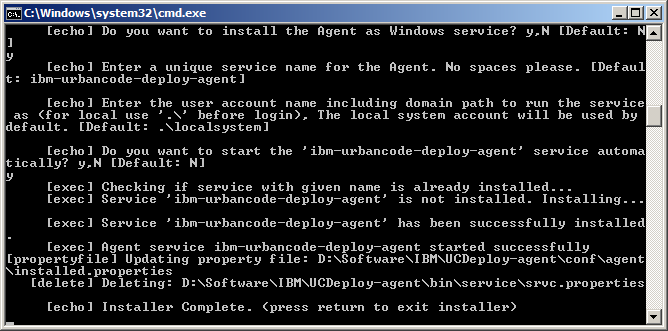
[WEB DEPLOYED **41**](#_Toc454543654)

# Installing the UDeploy Agent

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.

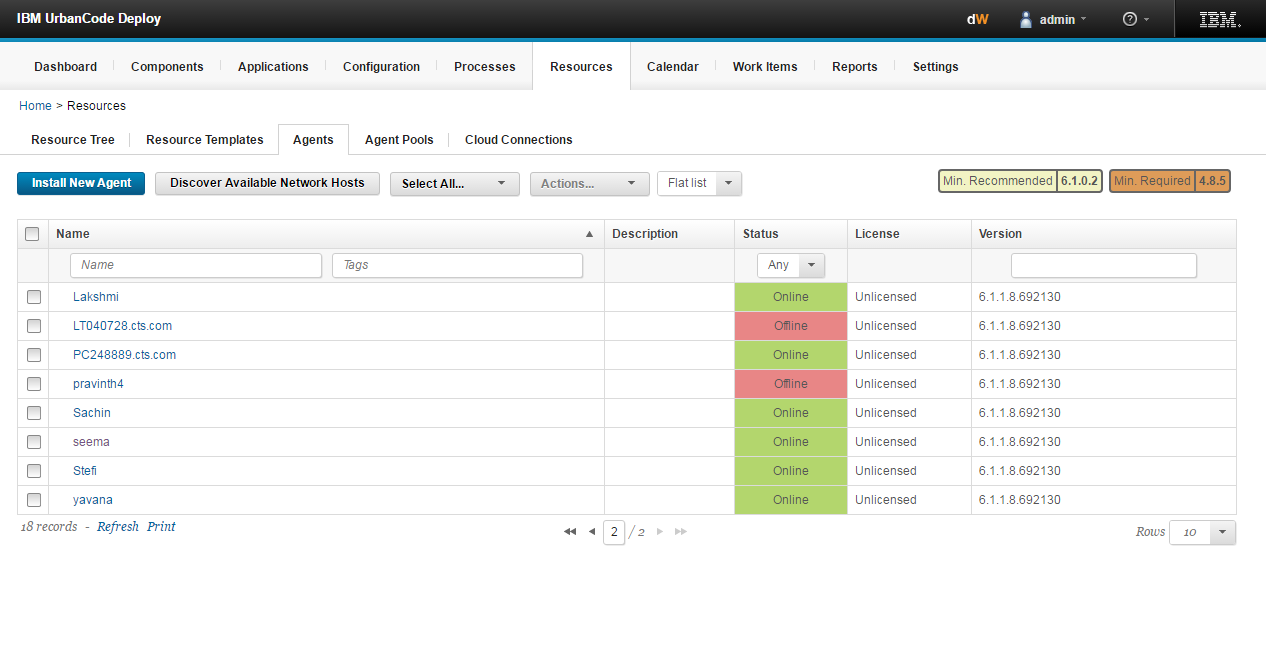


Just after this a command prompt will ask multiple question set all the answers there once its done the agent will be online on the udeploy server (http://10.242.181.79:9002).



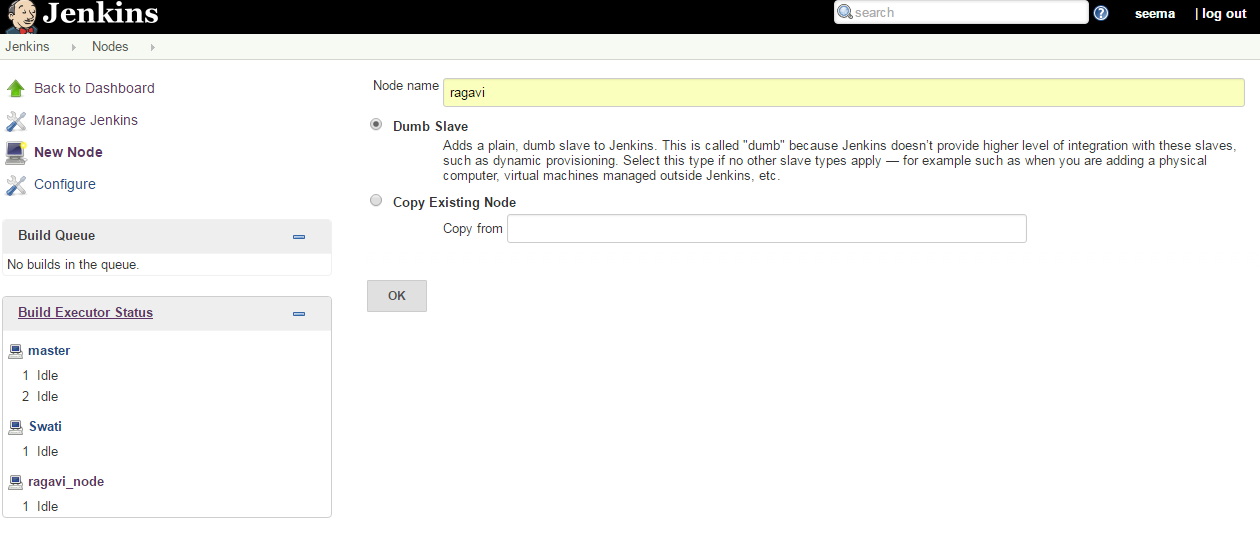
**UDeploy - agent installed**

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

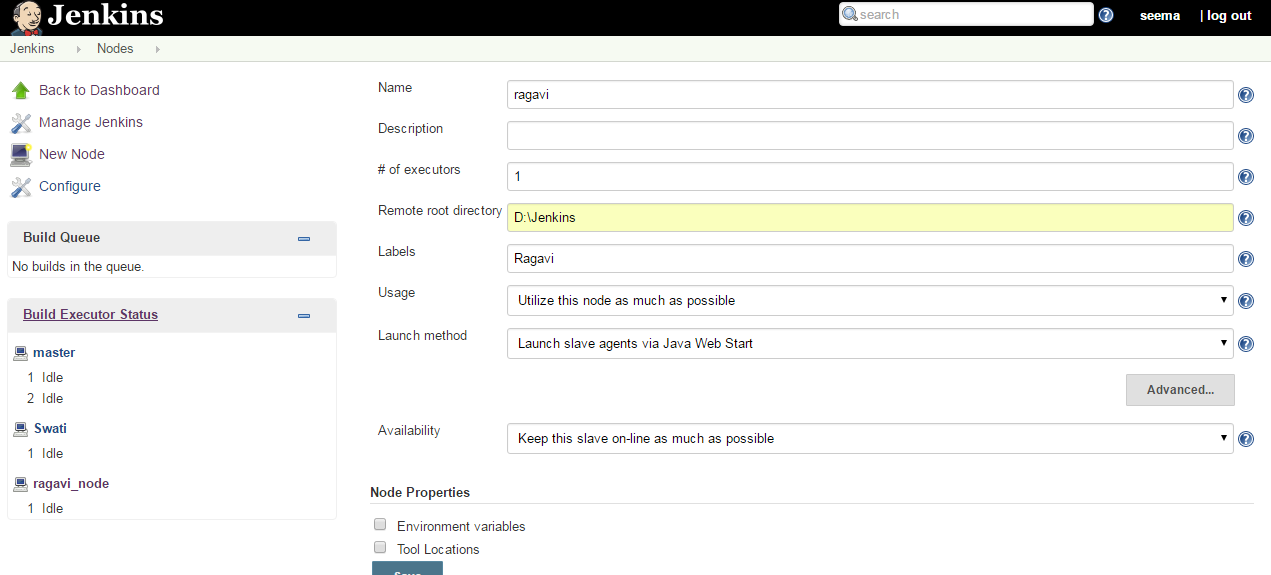


# Installing the Jenkins agent (Master-Slave Configuration)

Go to Manage Jenkins 🡺 manage node 🡺 click on New Node 🡺 Give the Node name and make it a dumb slave.



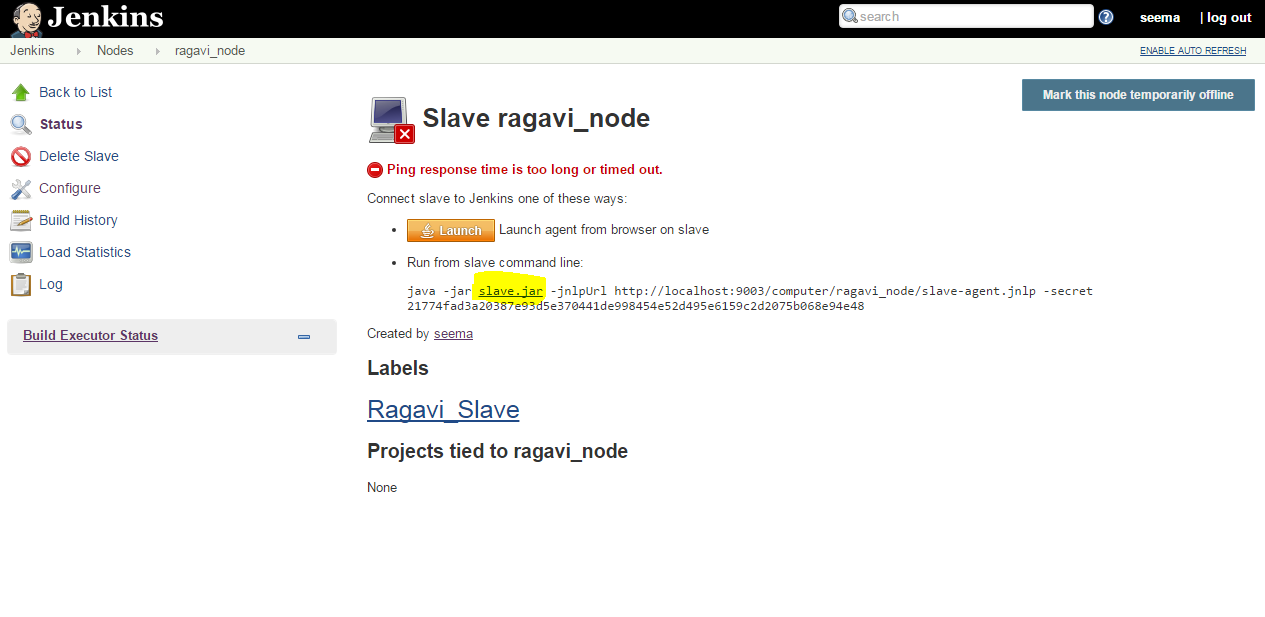
While making the slave configuration give the root directory name of the folder where we want to store our workspace in the slave’s machine.



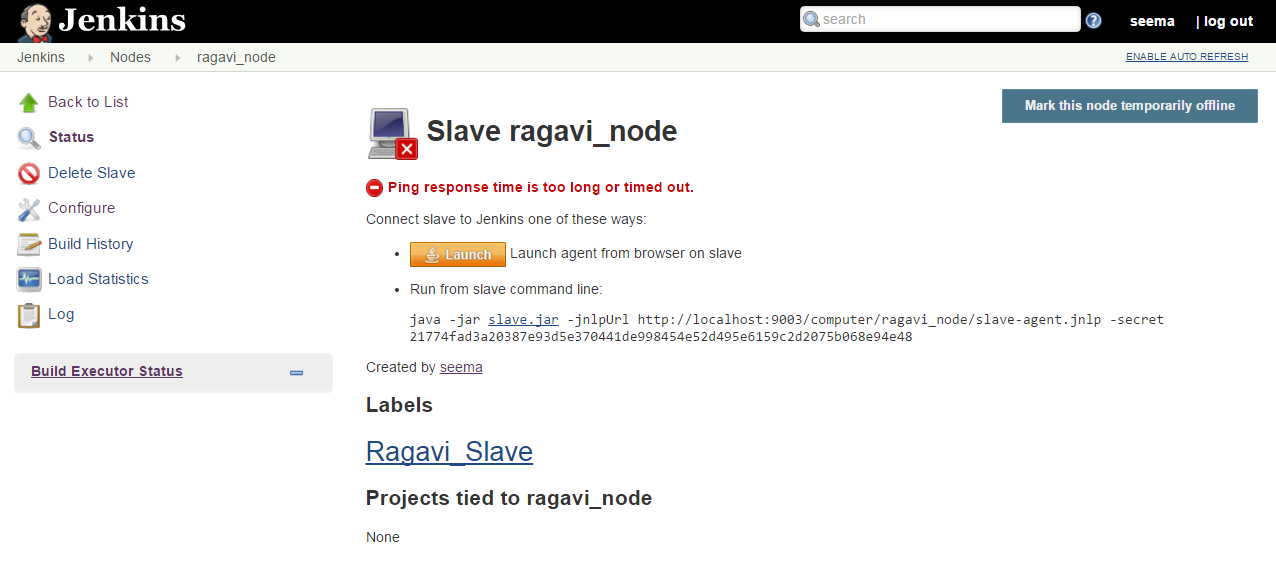
 Now you need to connect your slave machine to the master using the following steps.

1. Open a browser on the slave machine and go to the Jenkins master server URL.
2. Go to **Manage Jenkins, Manage Nodes**, Click on the newly created slave machine.

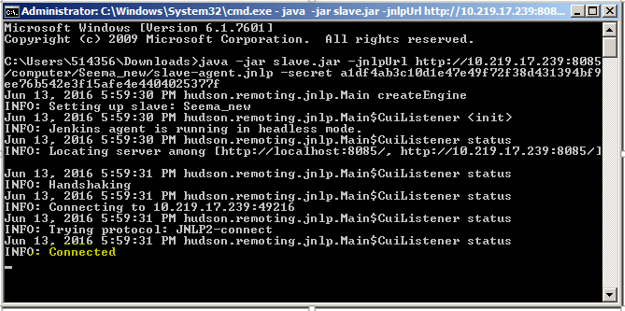
3. Download the slave.jar in the slave machine.



4. Browse the slave.jar just downloaded 🡺 go to command prompt 🡺 copy and paste the command given in the window for running the slave from command line.

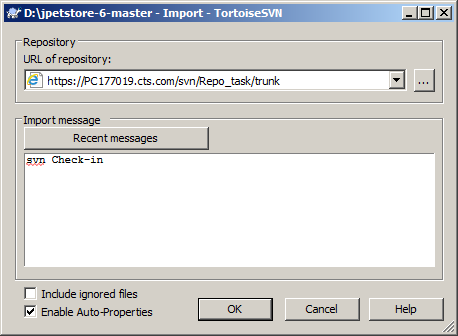


5. Jenkins Slave connected to the master machine.

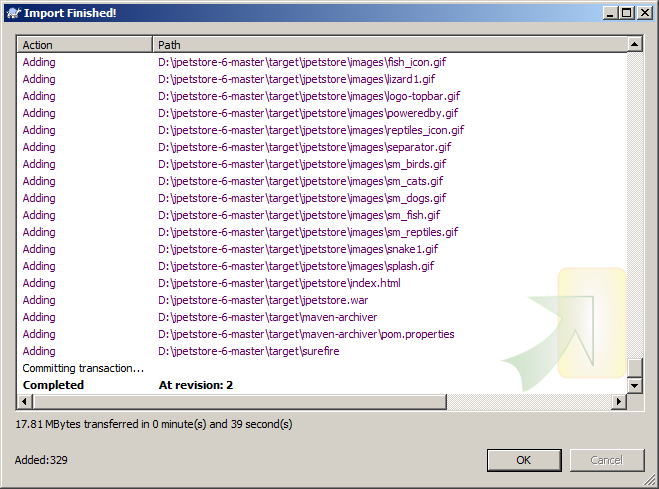


# Deploying jpetstore in the Tomcat with the help of the Jenkins Plugin

## **Check-in the Jpetstore project in a repository in the SVN.**

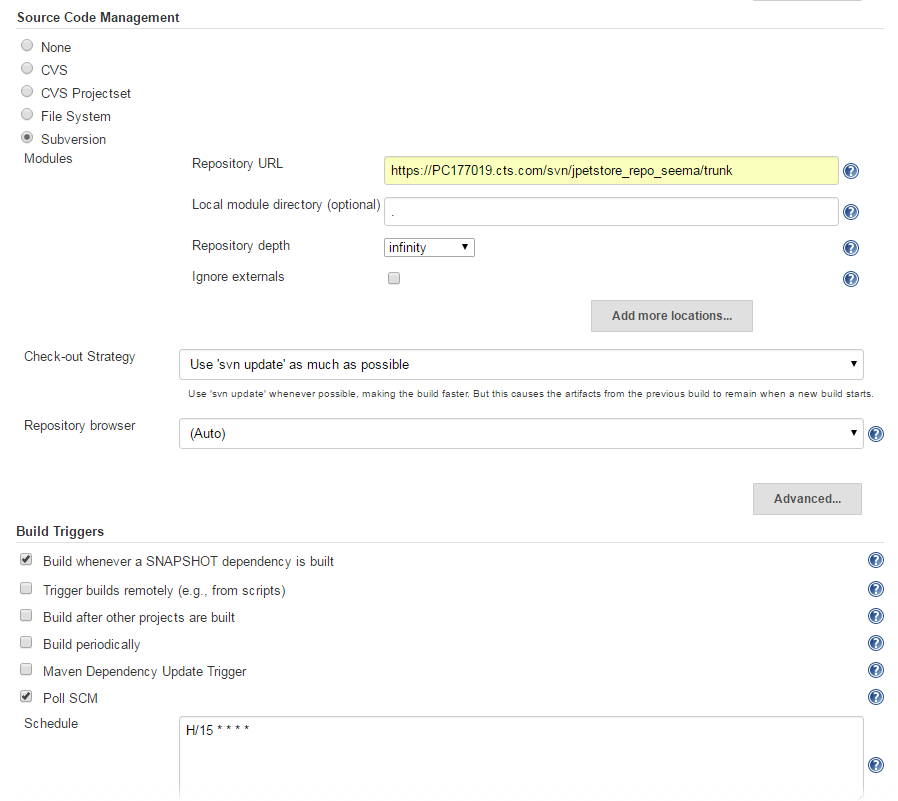


## **Committing the code in the subversion**



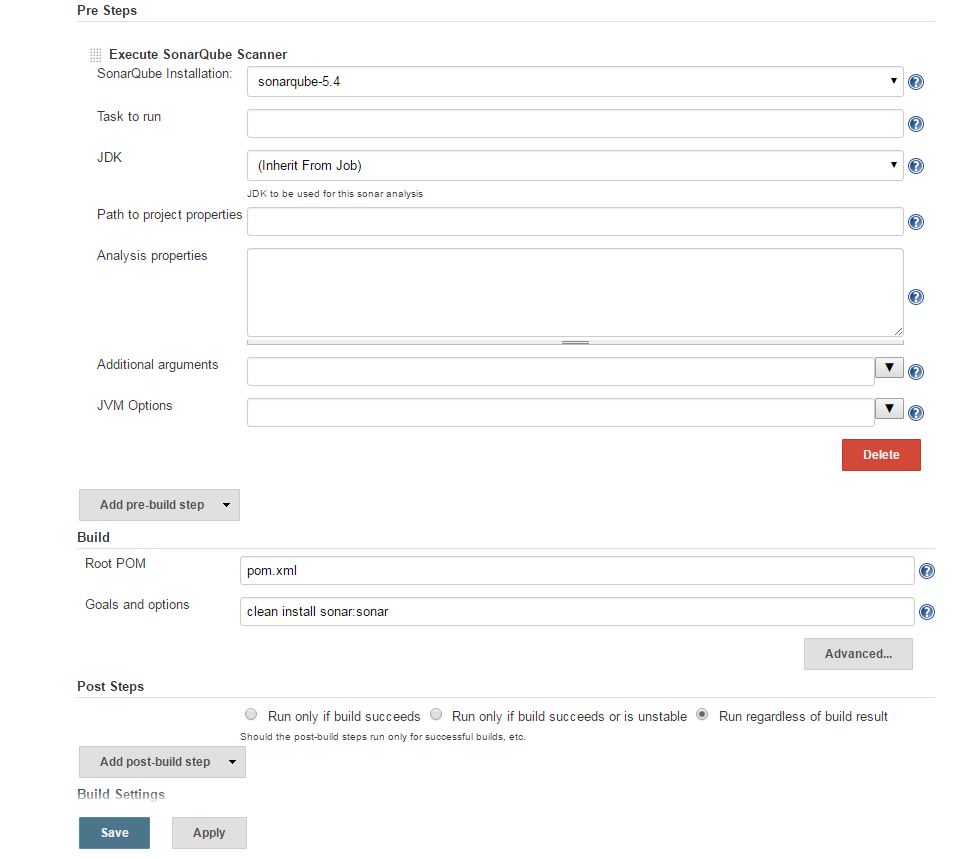
## **Running the project on the slave node:**

* Restricting the project running on the specified slave node will make the user to create the workspace in the slave’s machine.
* Polling the subversion Repository every 15 minutes will allow Jenkins to poll the code change in every 15 minutes of time and build the code changes.



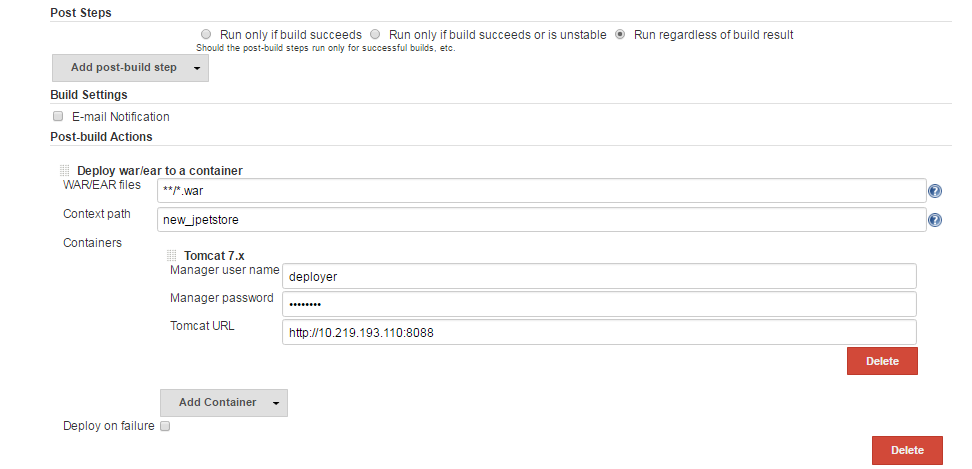
## **Pre Steps (SonarQube Scanner Configuration in Jenkins)**

Add Execute SonarQube Scanner in Pre Step and as here maven project is used the pom.xml will require the Goals. Set the goal as clean install sonar: sonar so that it will be analyzed in the sonar automatically.



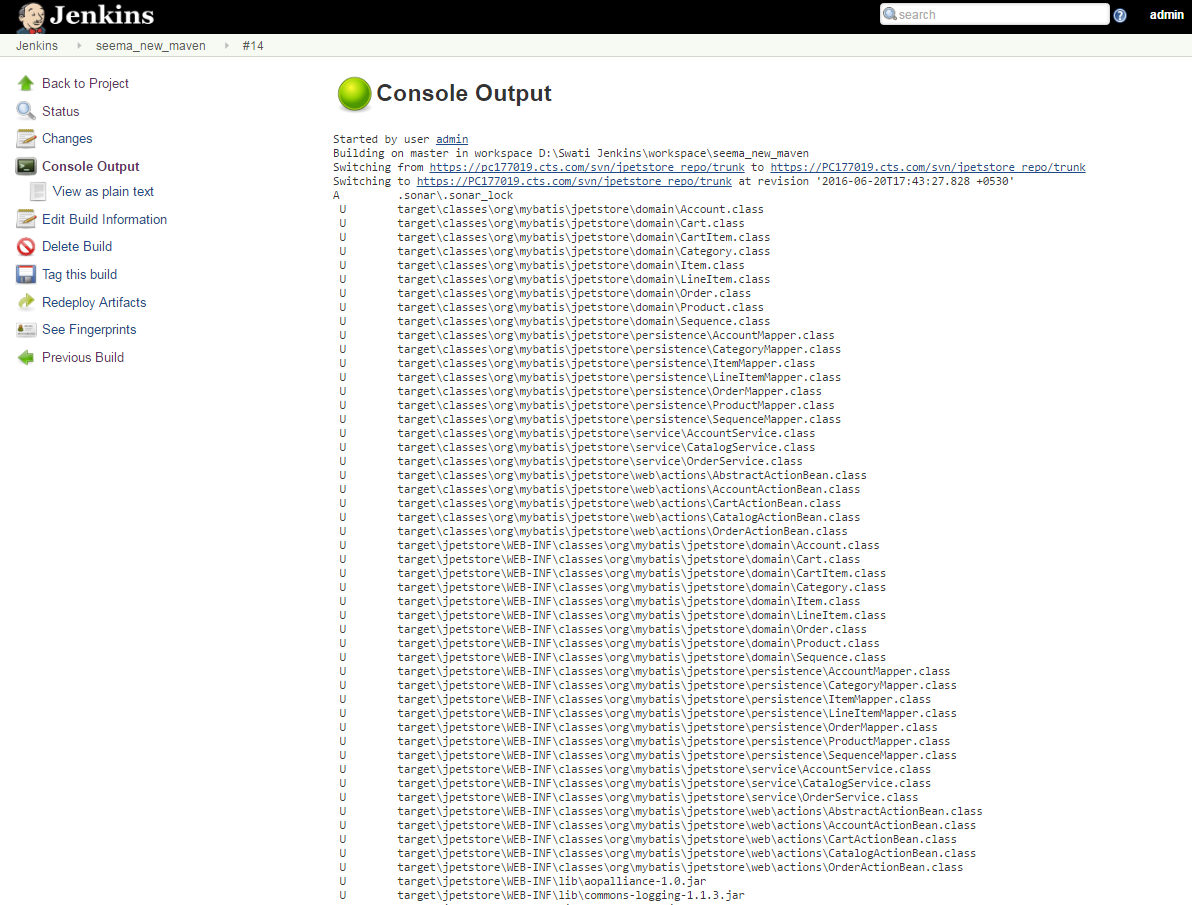
## **Post Build Action (Deployment in the Tomcat Server)**

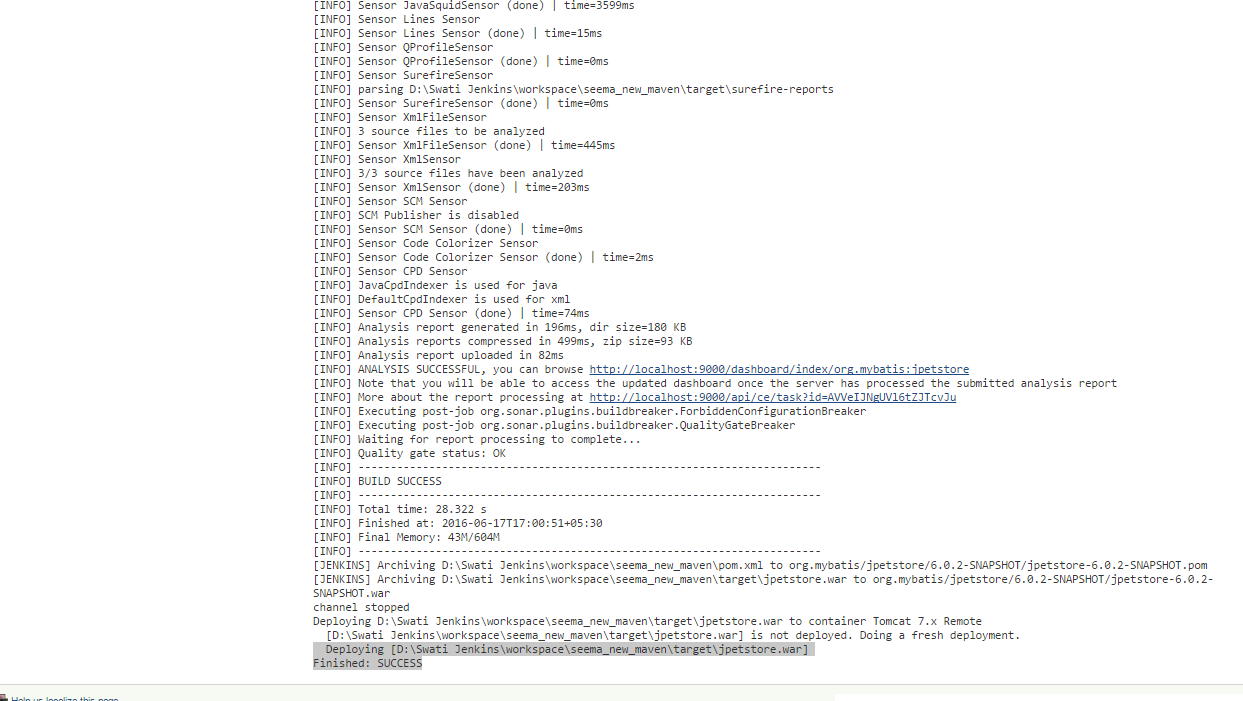
* Scroll down to bottom of page to “**Post Build Actions**”. Select **“Deploy war/ear to container**” from “**Add Post Build Actions**”.
* For WAR file you can use wild cards, e.g. \*\*/\*.war. The context path is the path of the URL under which your application will be published in Tomcat.
* Select the appropriate Tomcat version and for Manager User name and password just copy the name and password from Tomcat-users.xml file.
* The tomcat URL is the base URL under which your application can be reached.



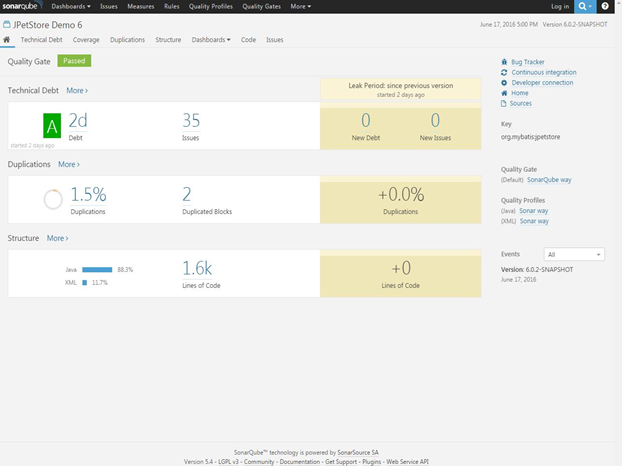
Here, the Deploy war/ear to a container plugin is configured in Jenkins, this plugin will automatically deploy war file to the Tomcat server.

## **Build the Jenkins job.**



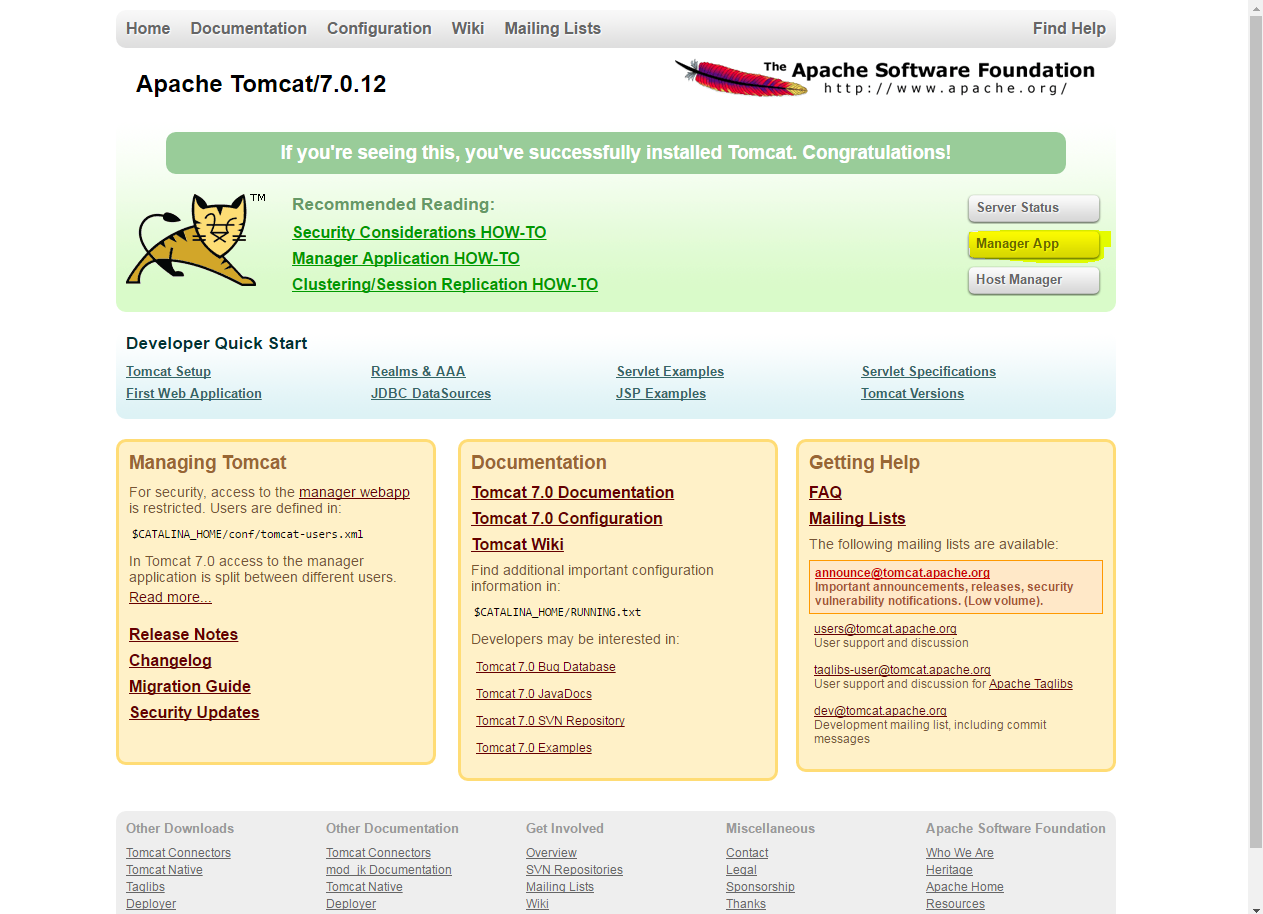


The sonar URL given in the Jenkins build (Console Output) will redirect you directly to the sonar analysis of the project as shown below:

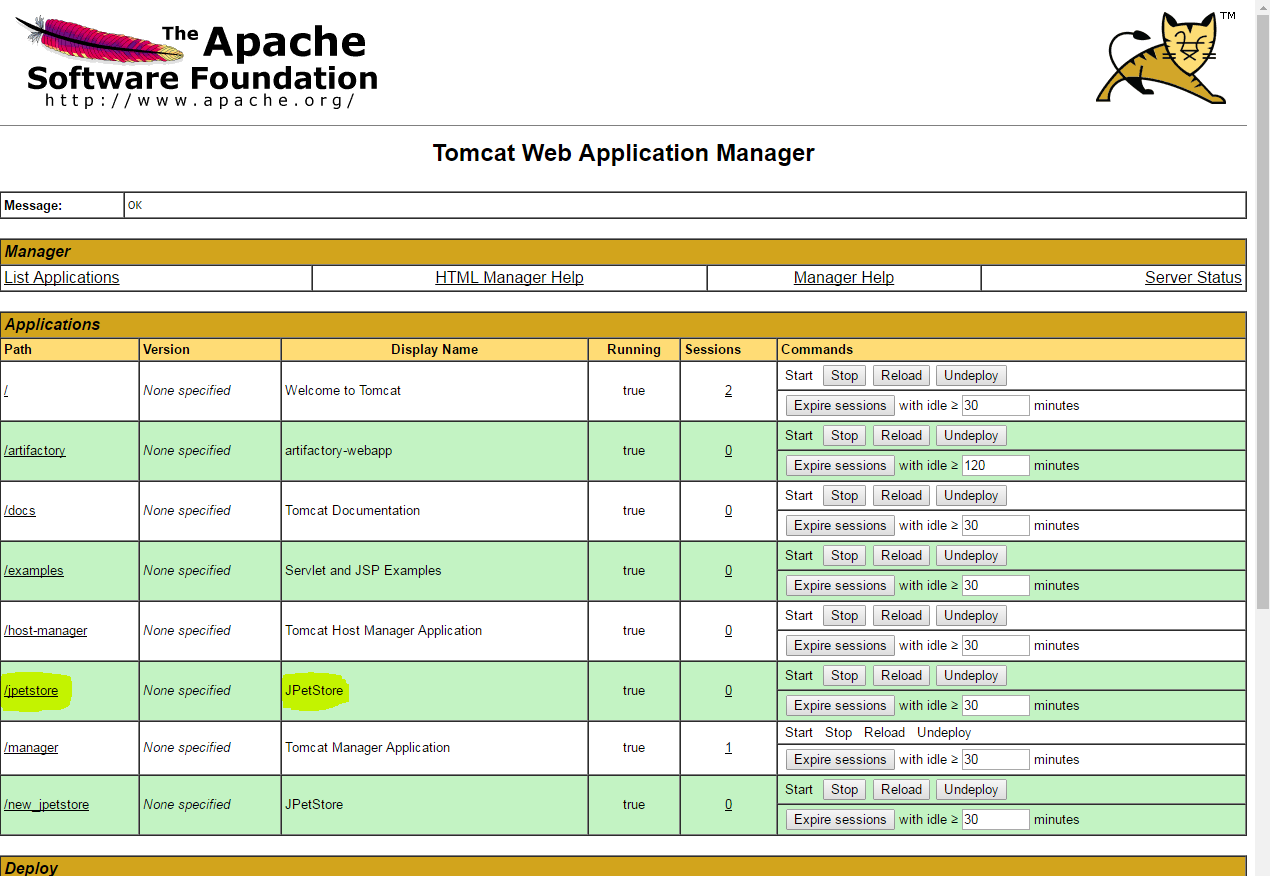


## **Tomcat Deployment**

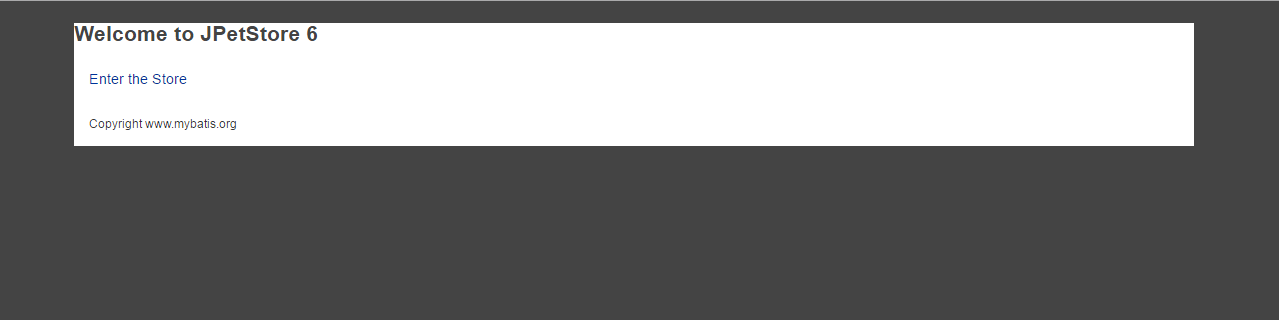
For viewing the deployment browse the **Tomcat URL** (the URL given in the Jenkins in Deploy to container plugin configuration). 🡺 Click on Manager App.



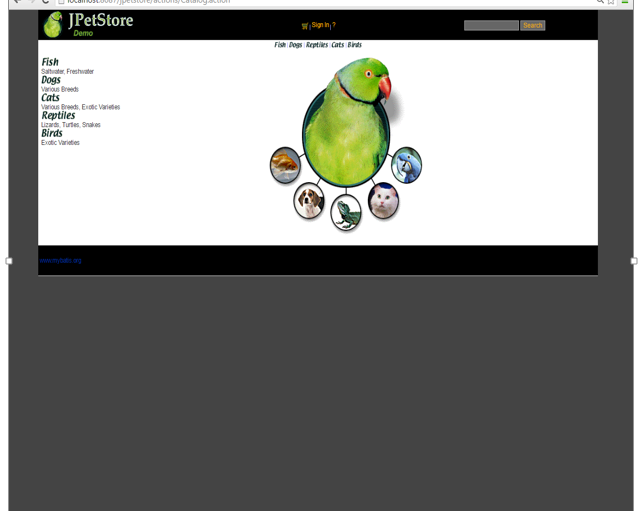
The deployment will be visible over here.



Click on the jpetstore 🡺 it will redirect to the web deployed jpetstore.

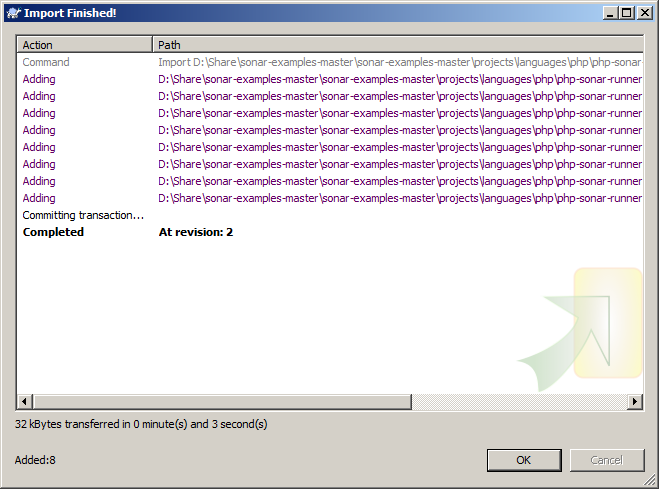


Click on the Enter the store 🡺 the below shown page will be displayed for jpetstore



# Deployment of the php project

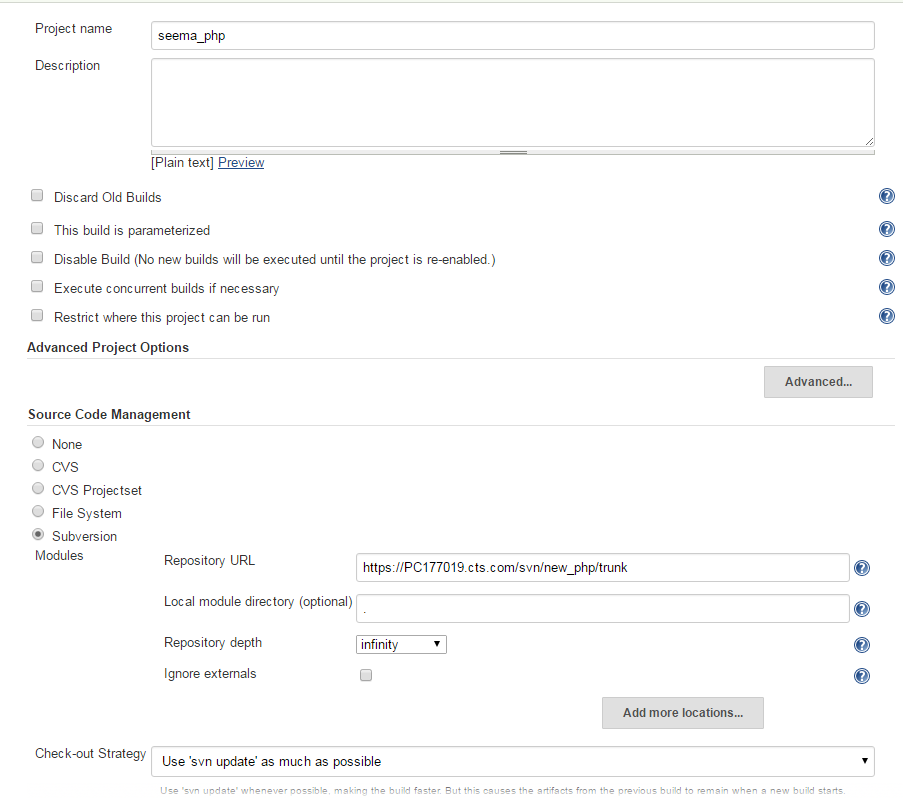
## **Check-in php code in SVN**



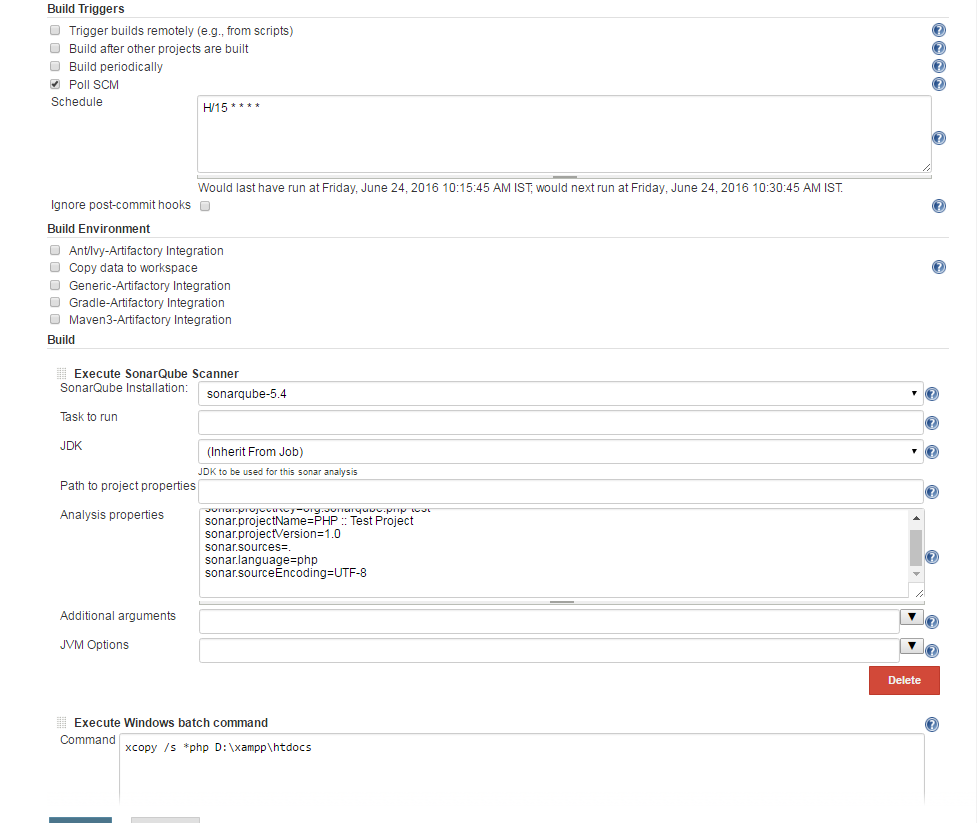
## **Analyzing php project in Jenkins with Sonar integration**

Set the following configuration for building Jenkins job with sonar analysis.

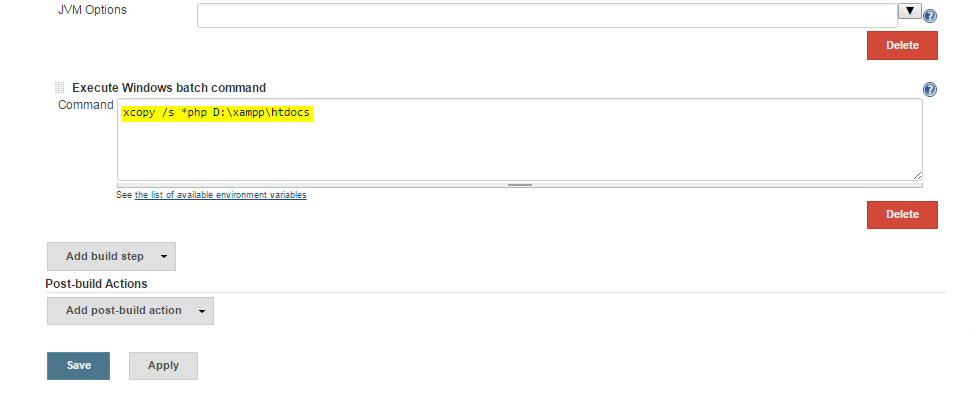
* Take the project from the subversion repository (give the appropriate repository URL), you can even poll the SCM by setting the schedule for it.



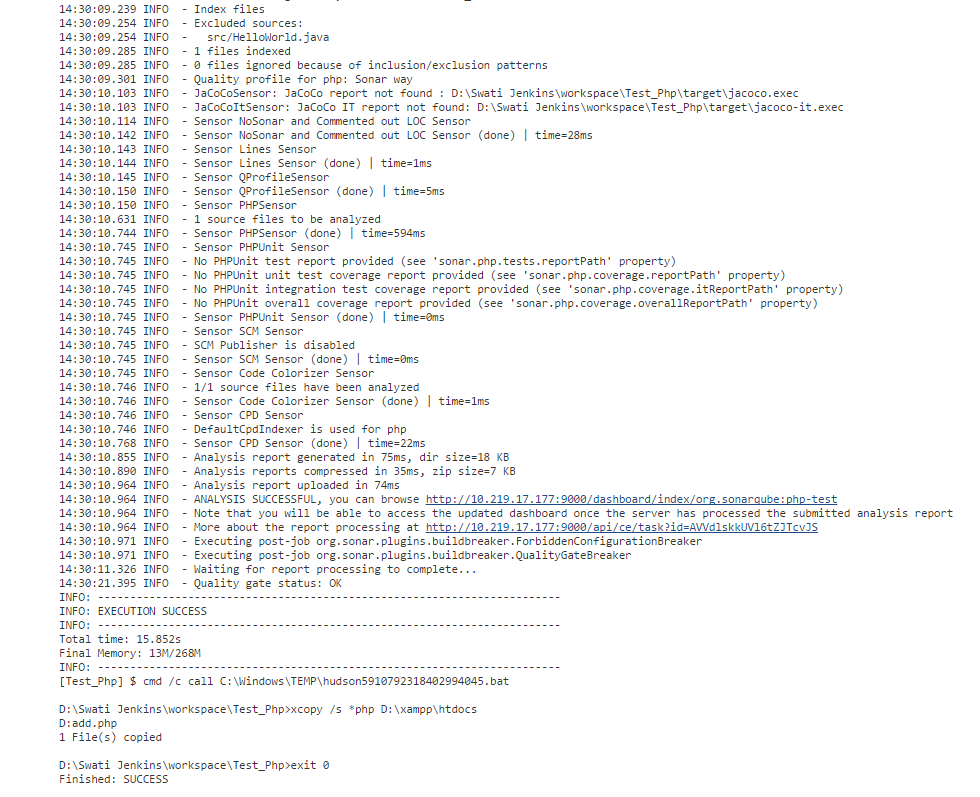
* Poll the SVN repository by scheduling the time. (Polling the code from the SVN every 15 minute through Build trigger poll SCM).



* Configure the sonar Scanner and the property file, then after building the php project in Jenkins the below shown is the output which we will get.
* Add the windows batch command for copying the php file to the htdocs folder of the XAMPP for direct deployment of the web application (Here, xcopy /s \*.php C:\xampp\htdocs).

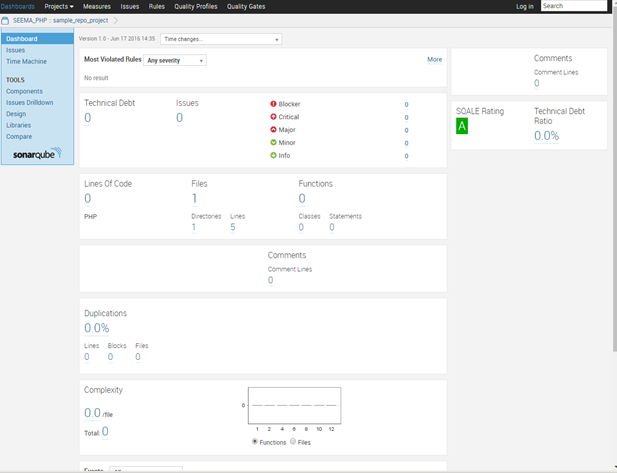


* After all these configuration Save and apply the configuration and build the Jenkins job.



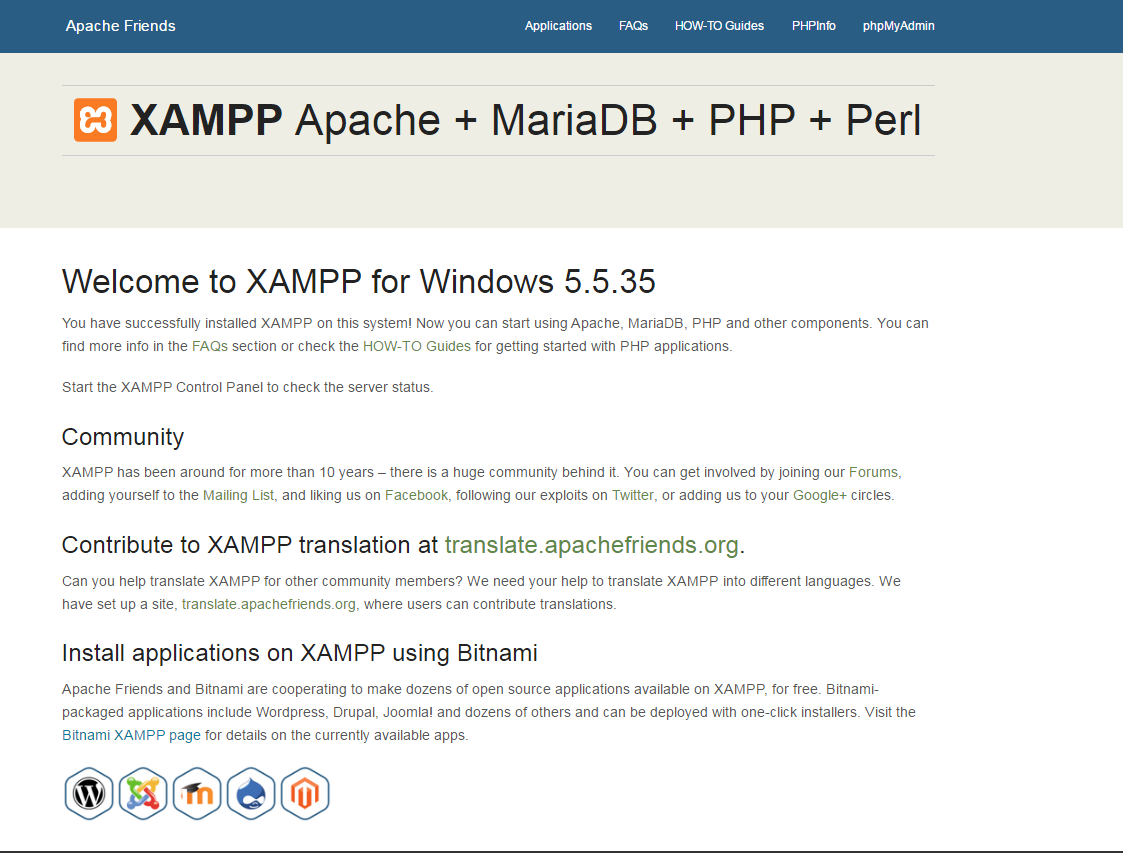
This means the project is successfully build in the Jenkins agent .As the Jenkins job is successfully build and copied the .php file from Jenkins workspace to the htdocs folder in XAMPP so we can now easily see the .php file on web.

The link given at the bottom of the above screenshot will redirect us to the sonar dashboard where we can see the sonar analysis for this project.

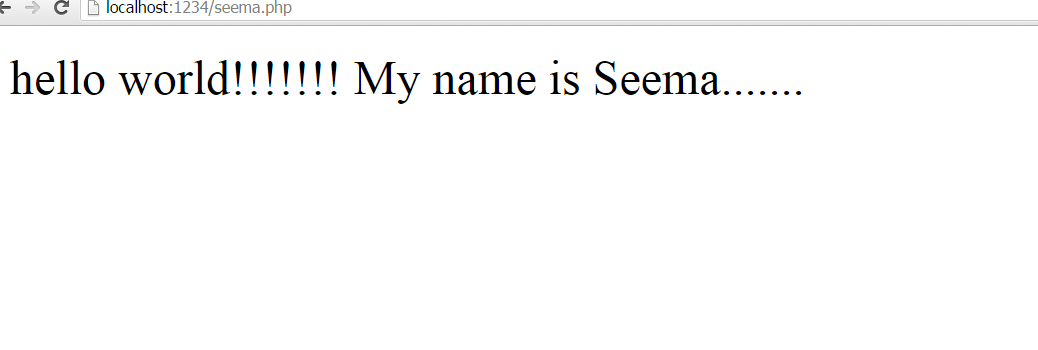


## **Deployment**

After this for viewing the web page, open the XAMPP website and just write the php file name on the website after the host.



This will redirect you to the made web page in this case it is simple php file.



# MS Deploy using UDeploy

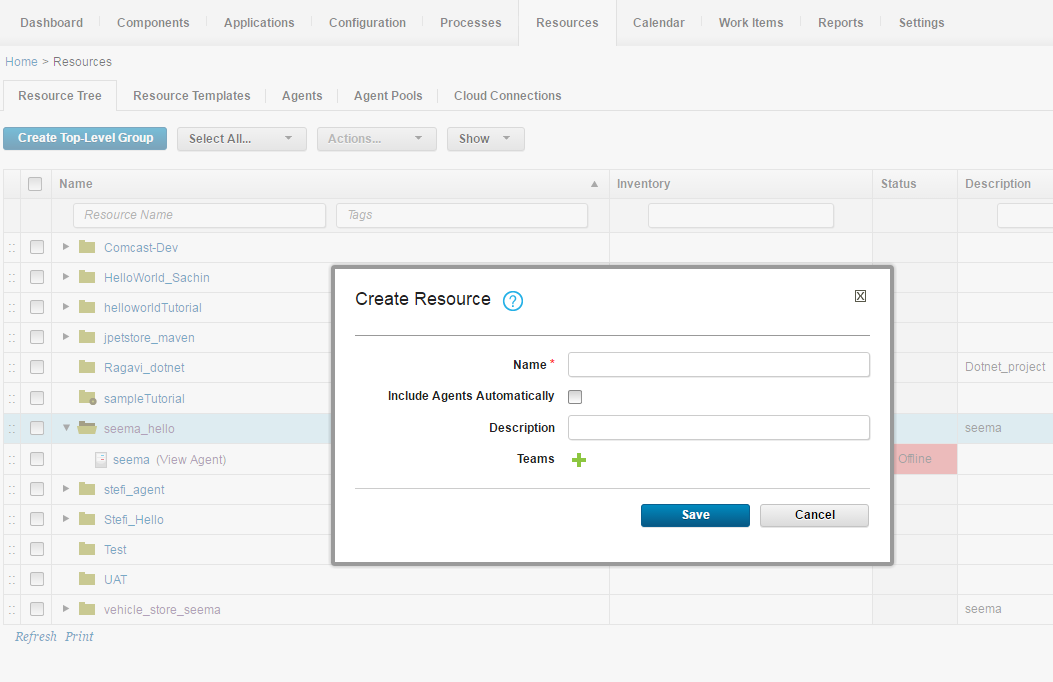
## **Pre-requisites**

* Before starting with Udeploy one should have IBM UDeploy agent installed on the system and,
* The one should also install the Jenkins plug-in for IBM Urban Code Deploy on the Jenkins server.
* Download the latest version of the plug-in from the following link: <https://developer.ibm.com/urbancode/plugin/jenkins/>.

## **Steps to be followed in IBM UDeploy to deploy the project**

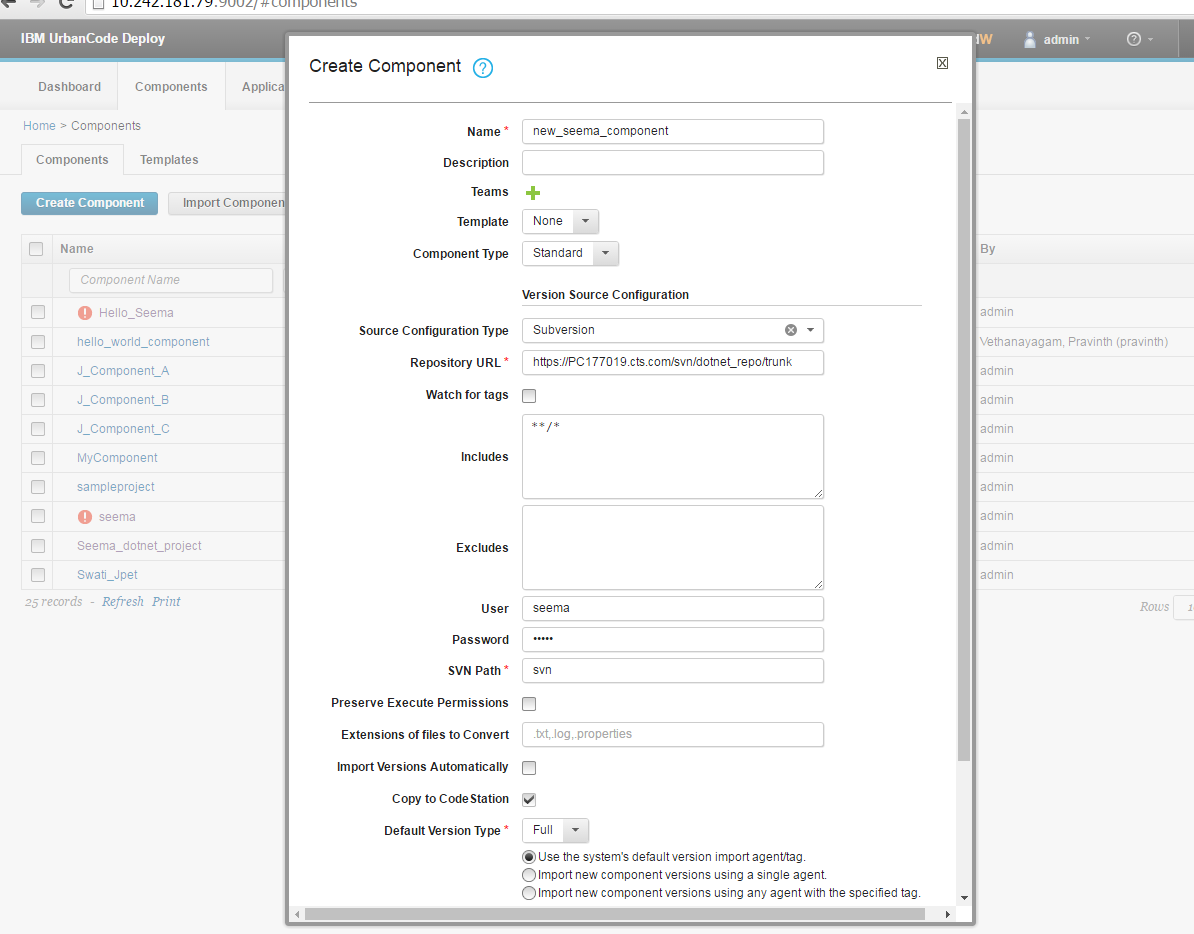
The following scenario shows how to use the Jenkins plug-in to publish Jenkins build output to versioned artifacts in IBM UrbanCode Deploy and to deploy the application.

1. Create a resource group, and then add the IBM UrbanCode Deploy agent to the resource group, as shown in the following screen capture(click top level group)

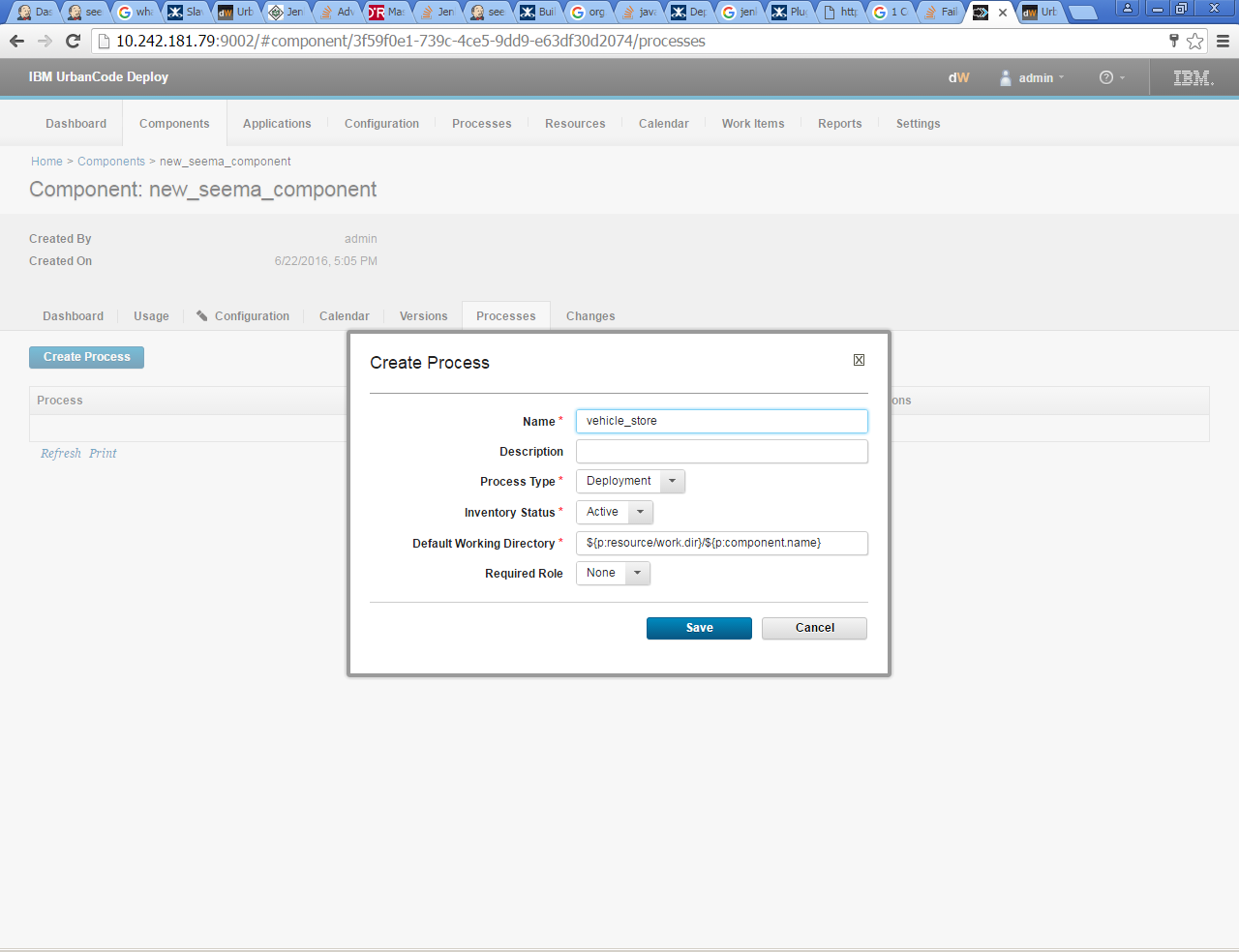


1. Create components in IBM UrbanCode Deploy

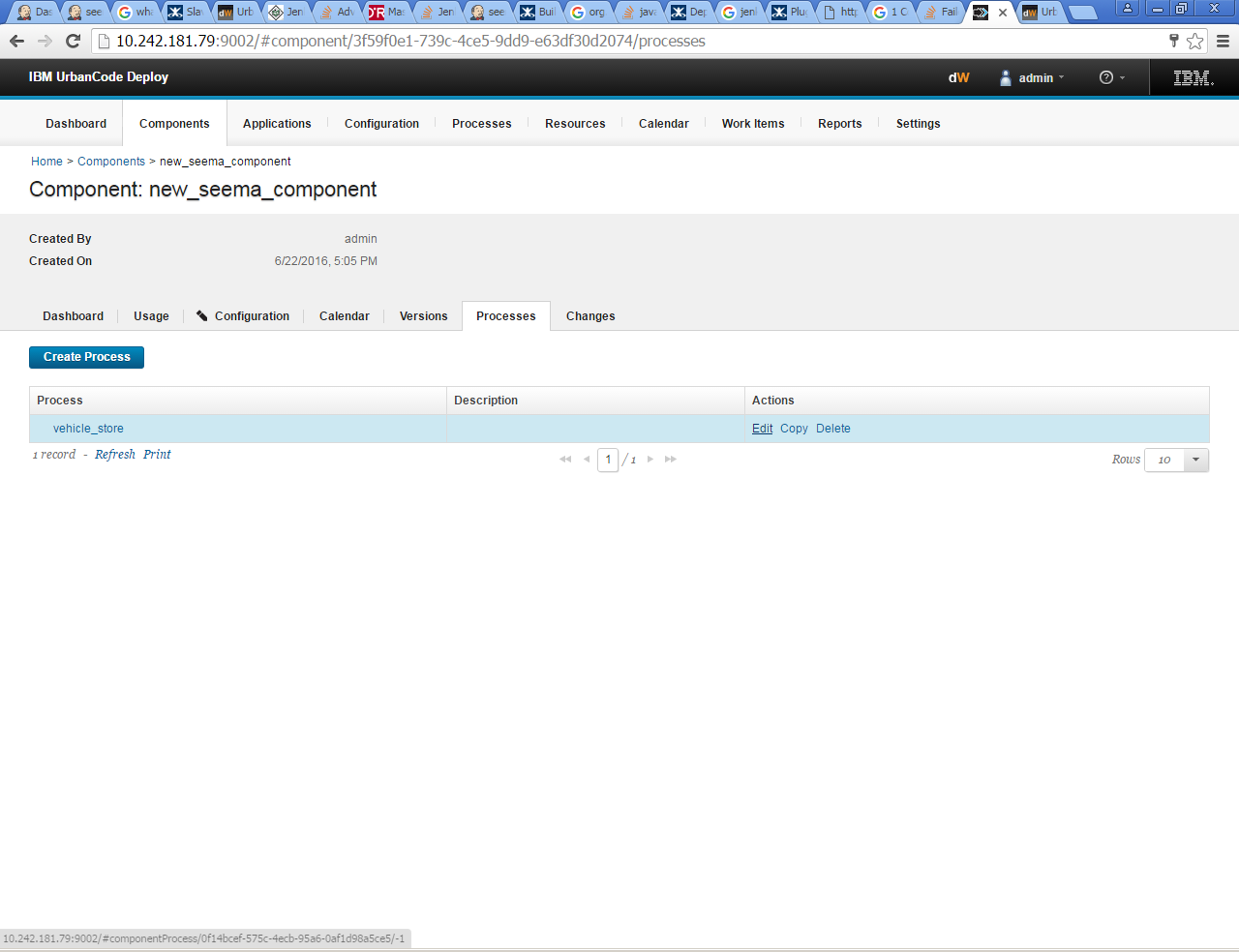
* Click the **Components** tab and then click **Create Component**. In this window, you specify where the artifacts for the component are.
* In the **Name** field, give your component name.
* In the **Source Configuration Type** list, select **Subversion** and pass the required repository URL.
* Click **Save**.



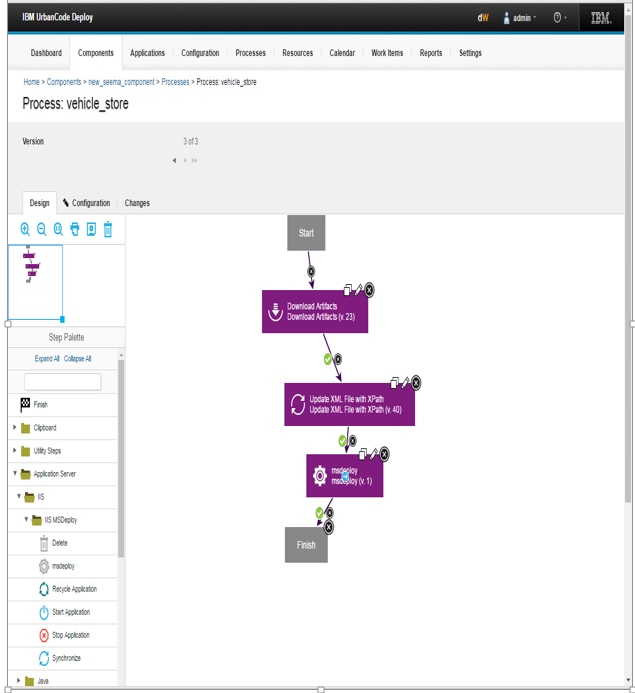
1. In the component go to the process and click on the create process



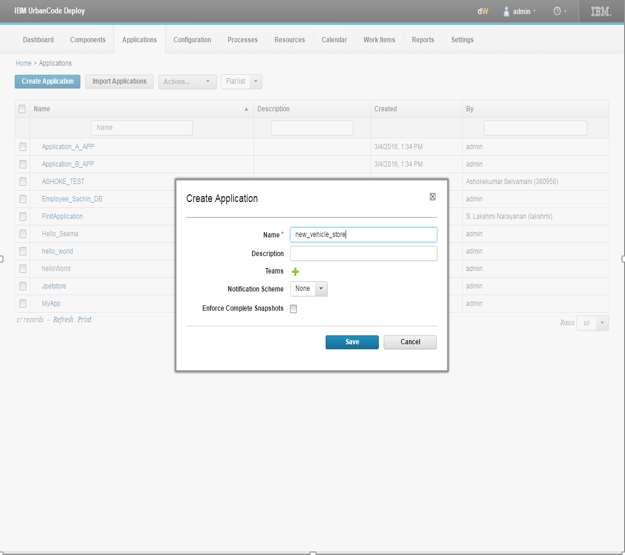
1. Go to process created and edit the process.



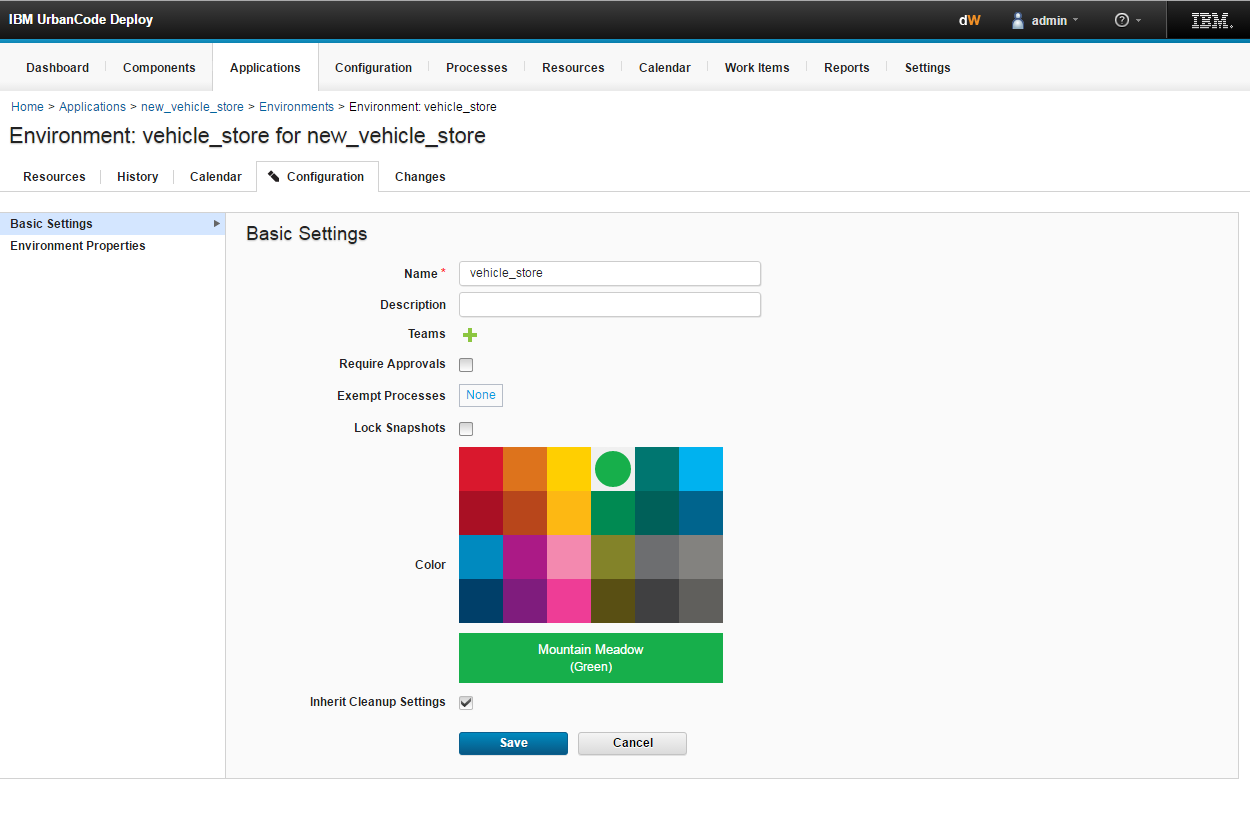
1. Click on edit and make the proper flow chart of the whole process.



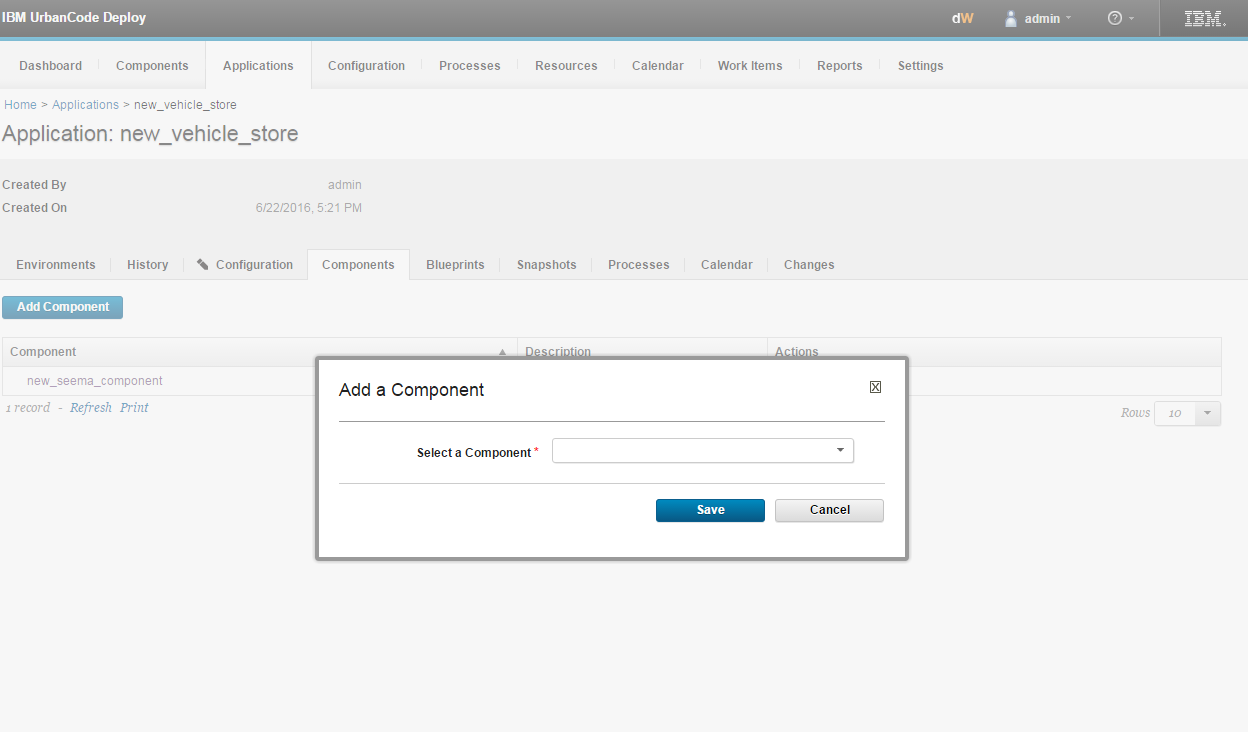
1. Now go to the Application and create a new application



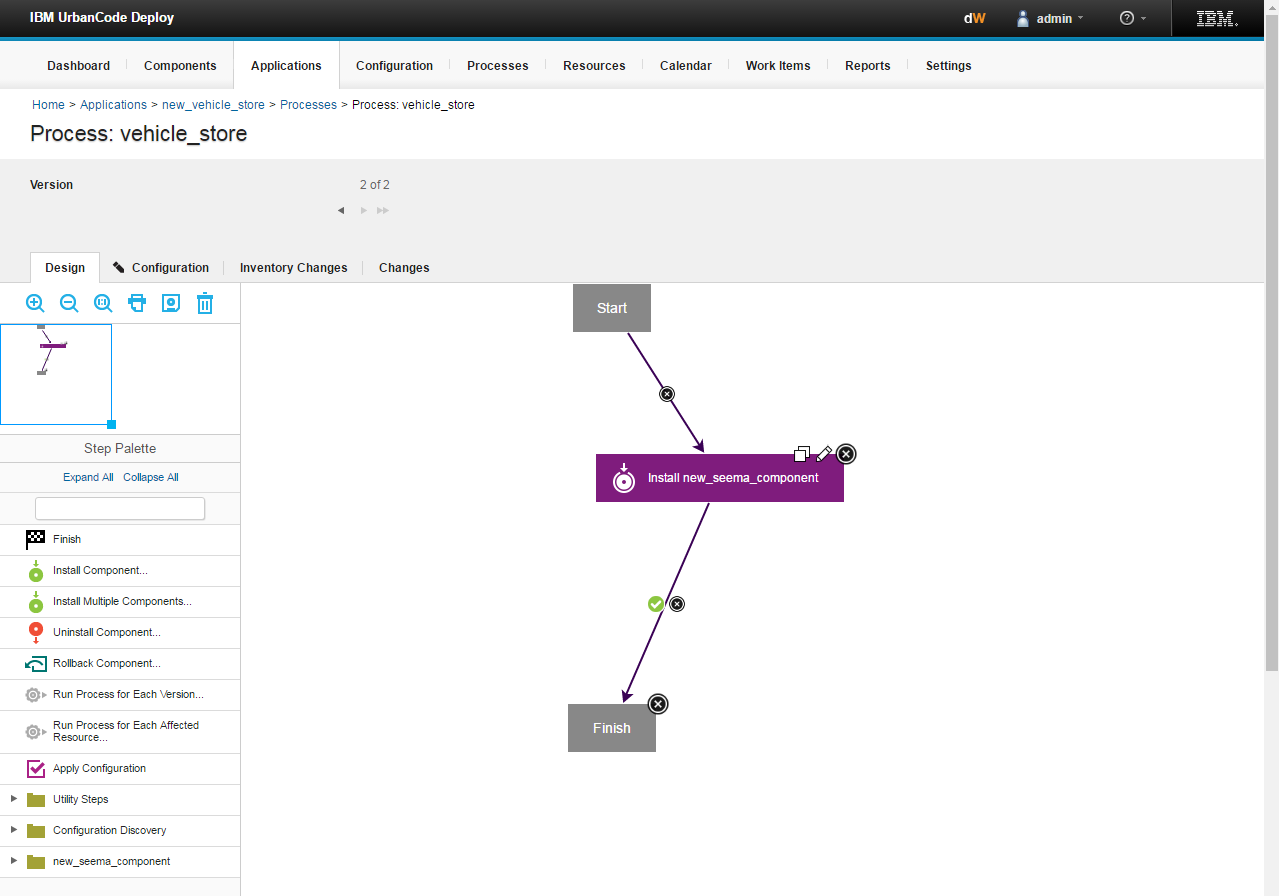
1. In the application click on create Environment to create a new environment.



1. Go to the environment created and in that go to the Component then add your component to the created environment.



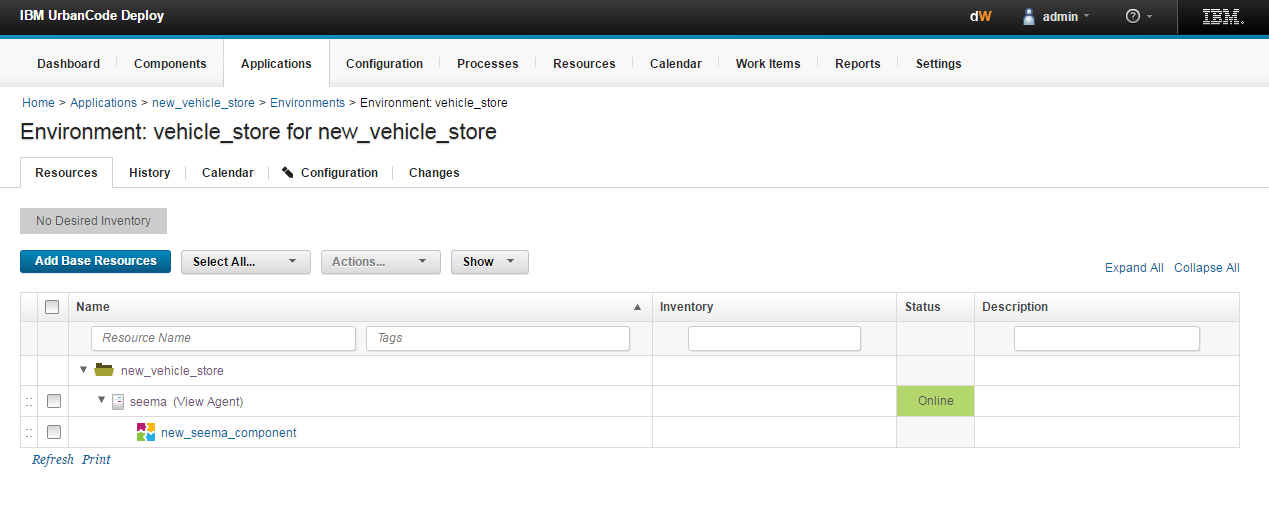
1. In the component Create a new process 🡺 in that process create the flow chart to install the component (Drag the install component and connect with start and finish).



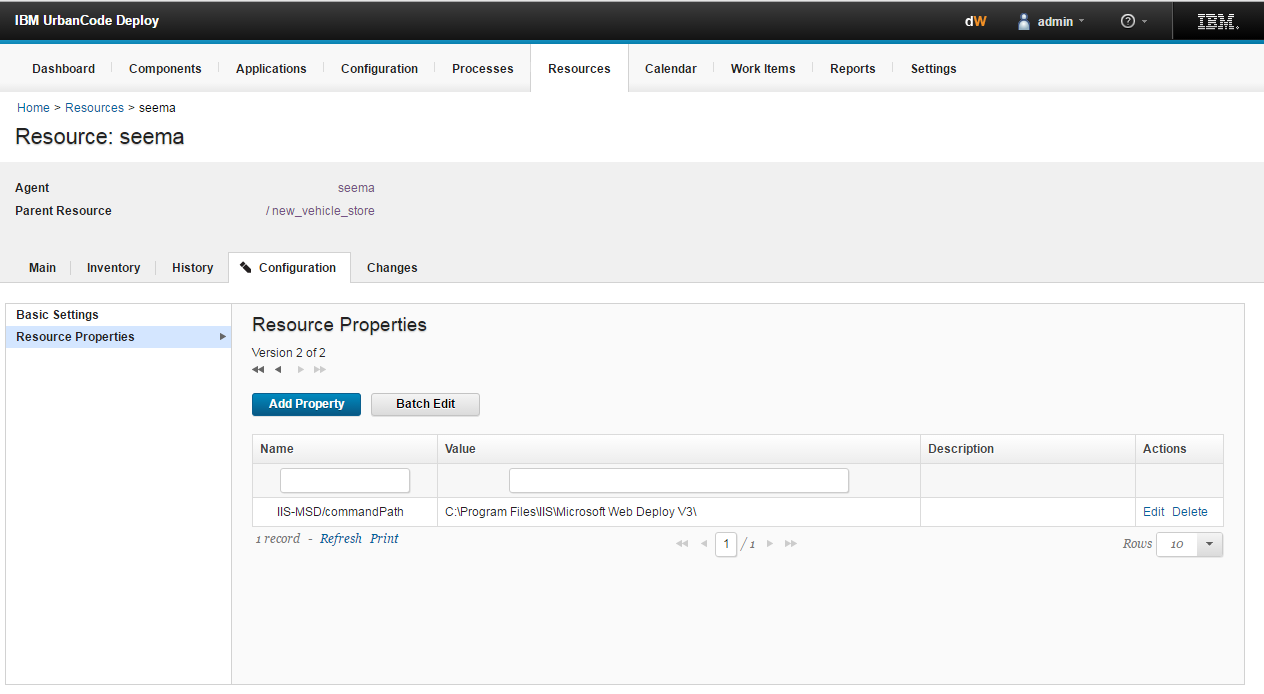
1. Then add the base resource to the application.

In the Applications map the resource to the agent and then map the agent to the component for this

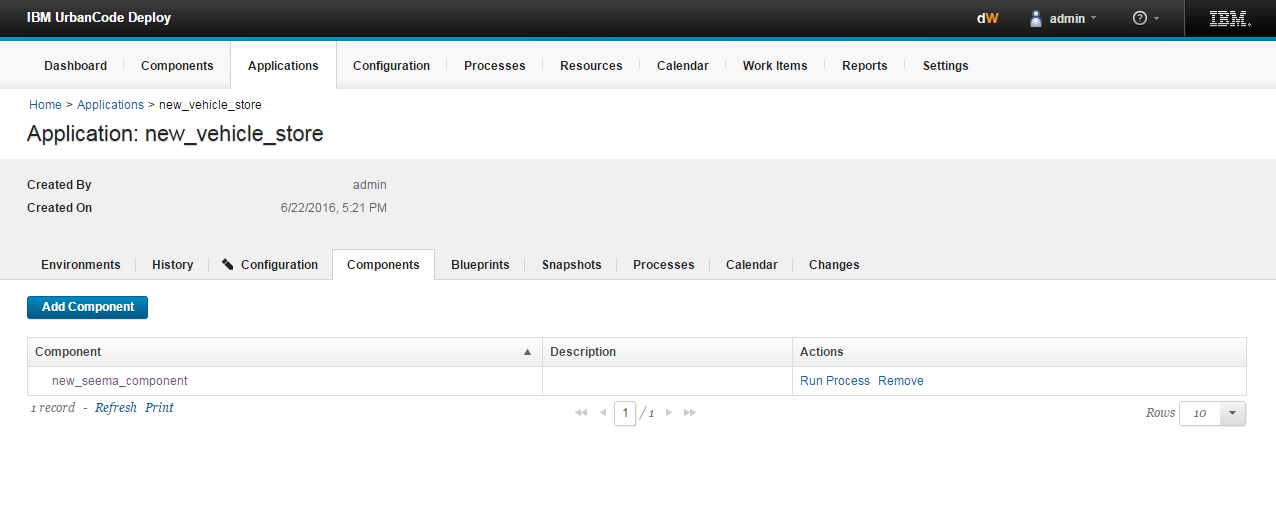
Go to the Applications 🡺 go to the environment created by you 🡺 click on the Add Base Resources and select your resource.



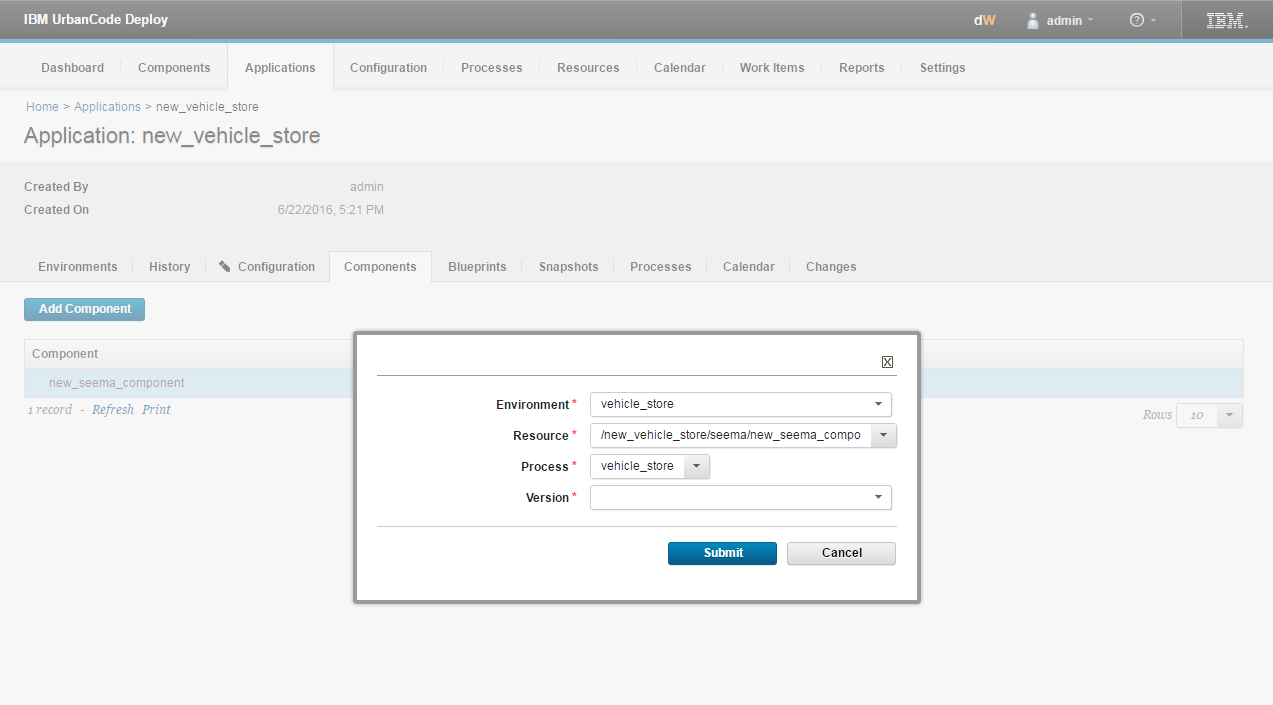
1. Then go to the resources 🡺go to your agent 🡺 go to configuration, Add the required property in the property you have to give the name of the property.(like, here IIS-MSD/commandPath) and the value in value you have to specify the physical location of the Microsoft Web deploy V3.



xii. Then go to the application 🡺 click on the component tab 🡺 Go to your component.

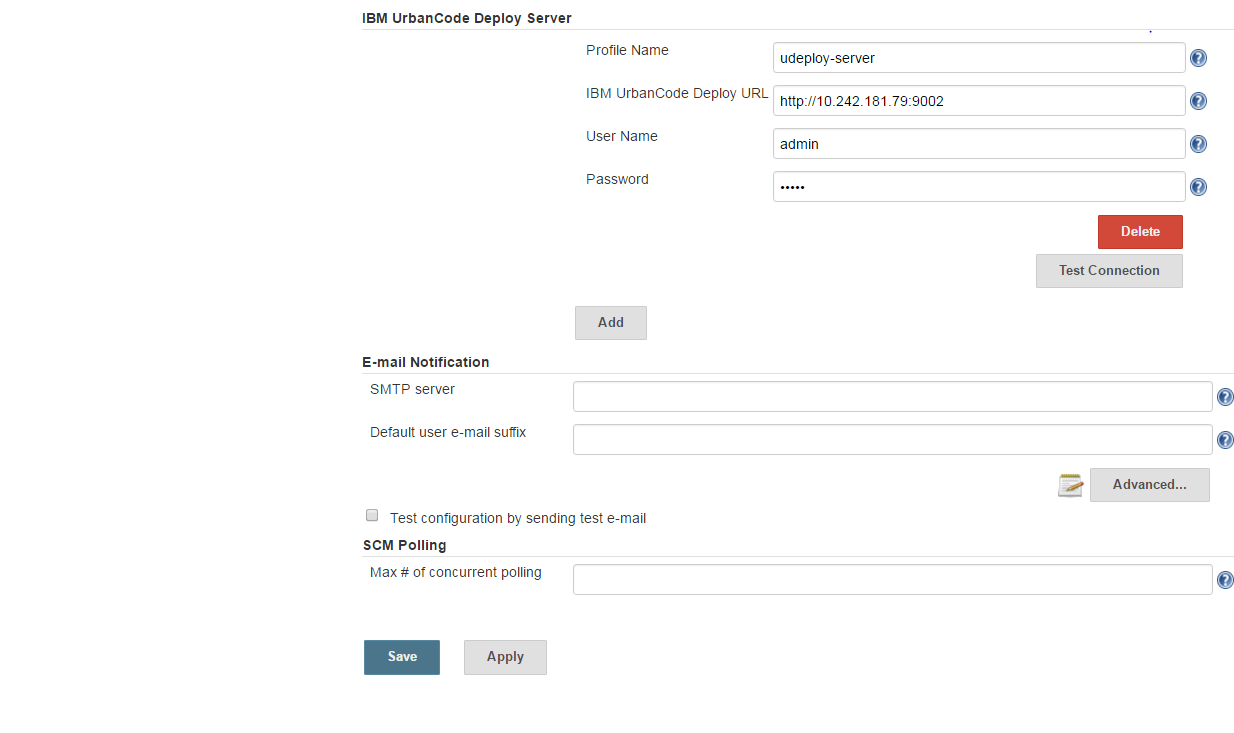


xiii. Click on the RUN PROCESS, it will show this window. Here we need version this version we need to generate from the Jenkins.

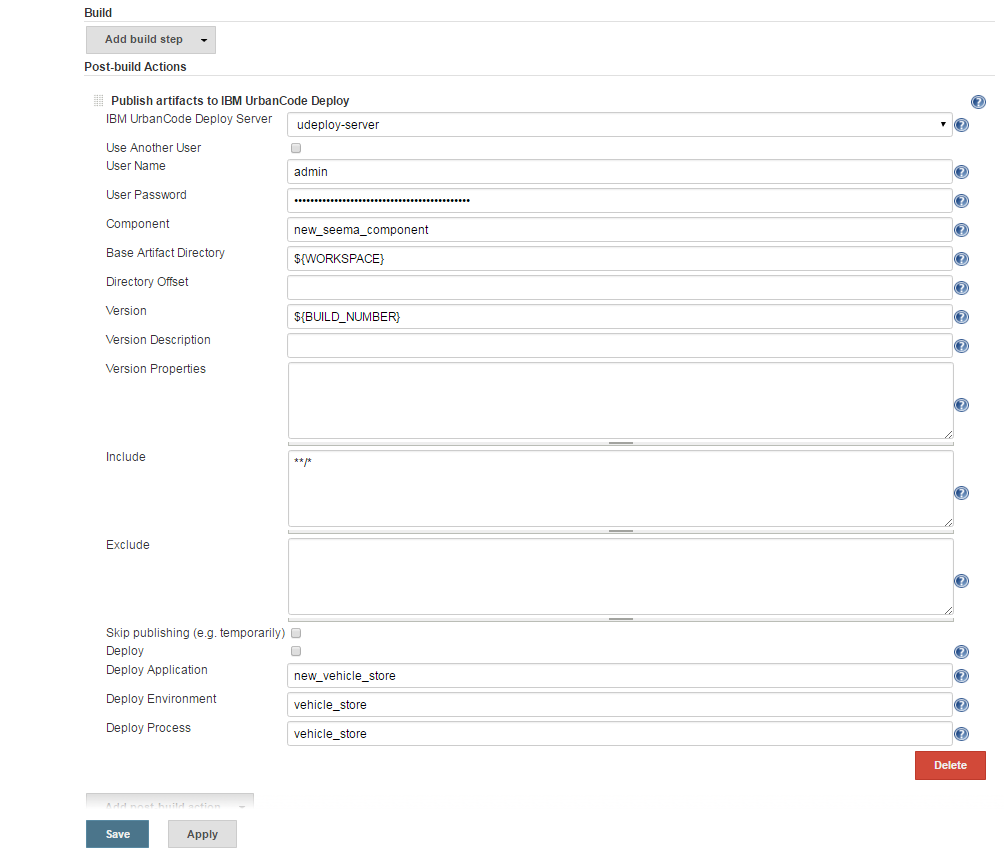


xiv. Jenkins Configuration with the UDeploy server

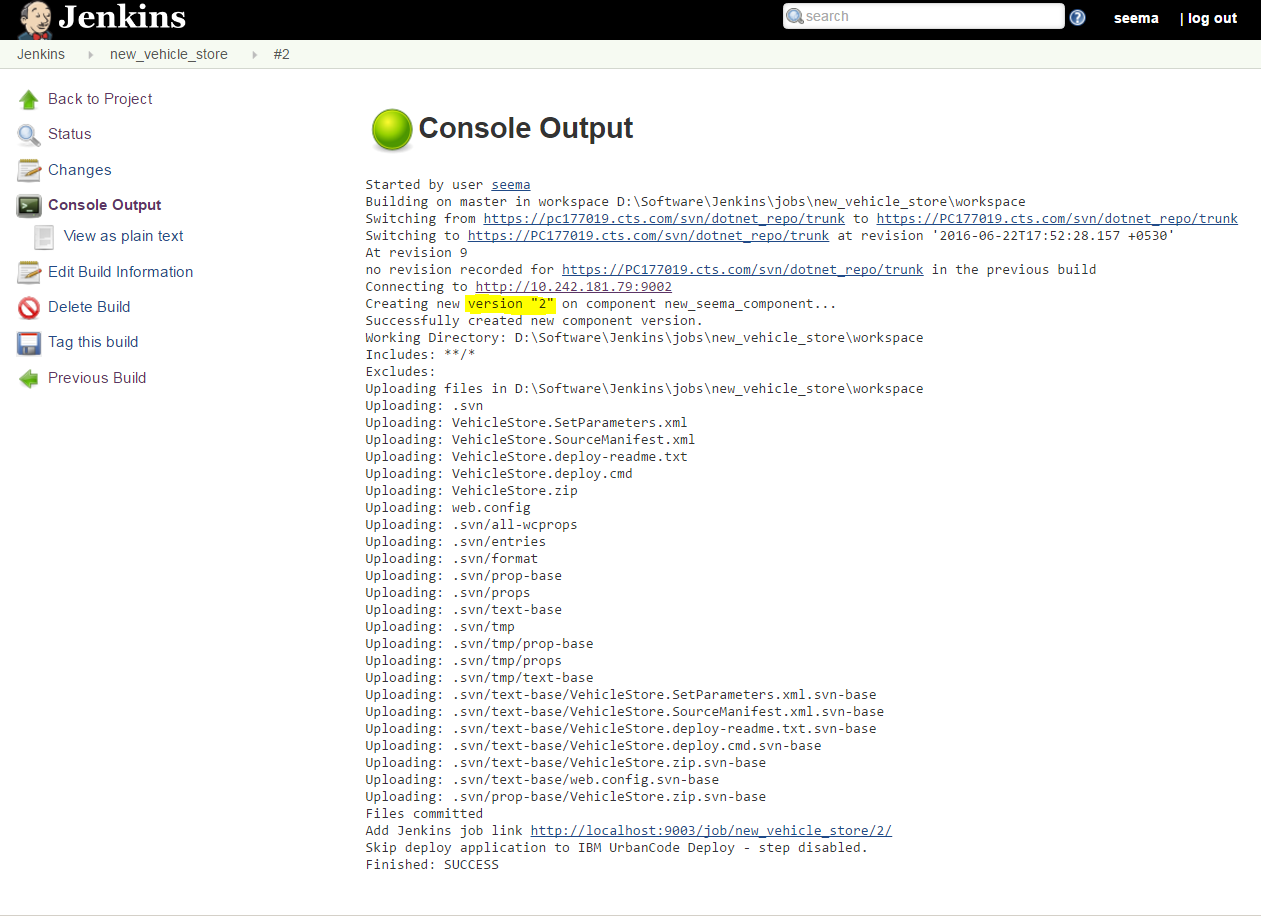
Firstly, download the IBM UrbanCode Deploy plugin 🡺 go to Manage Jenkins 🡺 go to Configure System then configure it as:

****

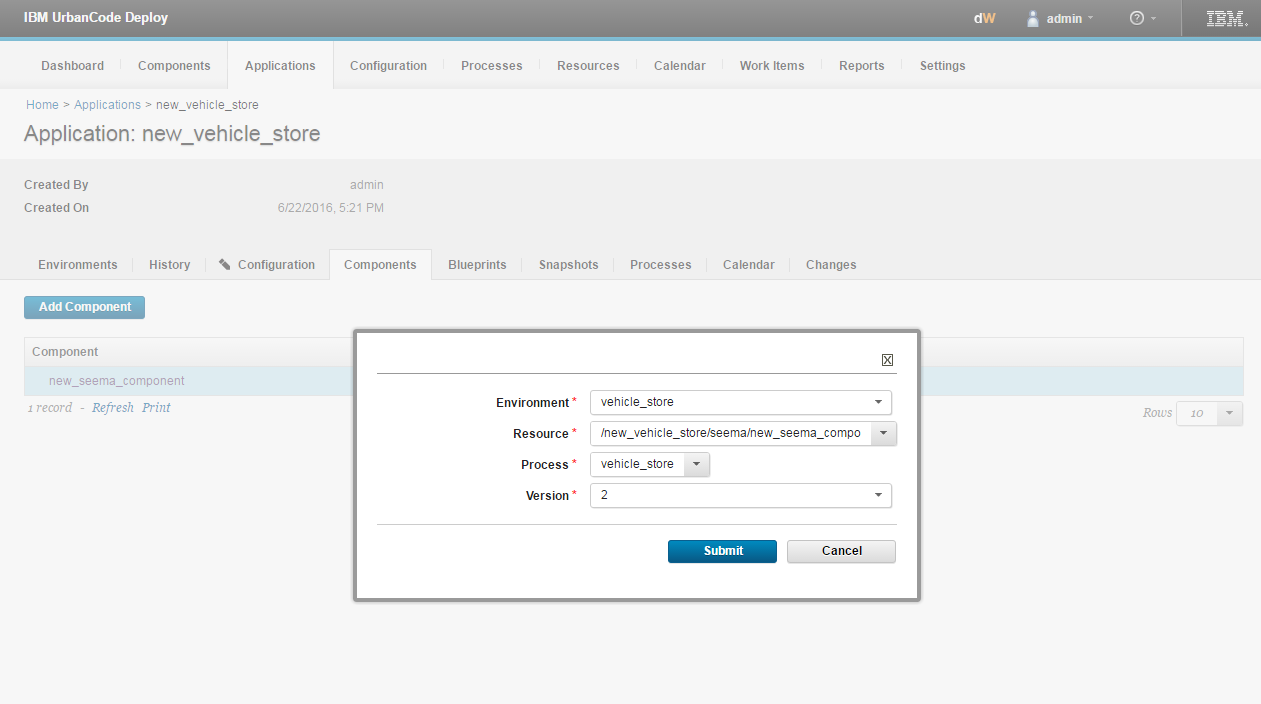
Xv. Create a new job 🡺 add Publish artifacts to IBM UrbanCode Deploy in post build step with in Jenkins.



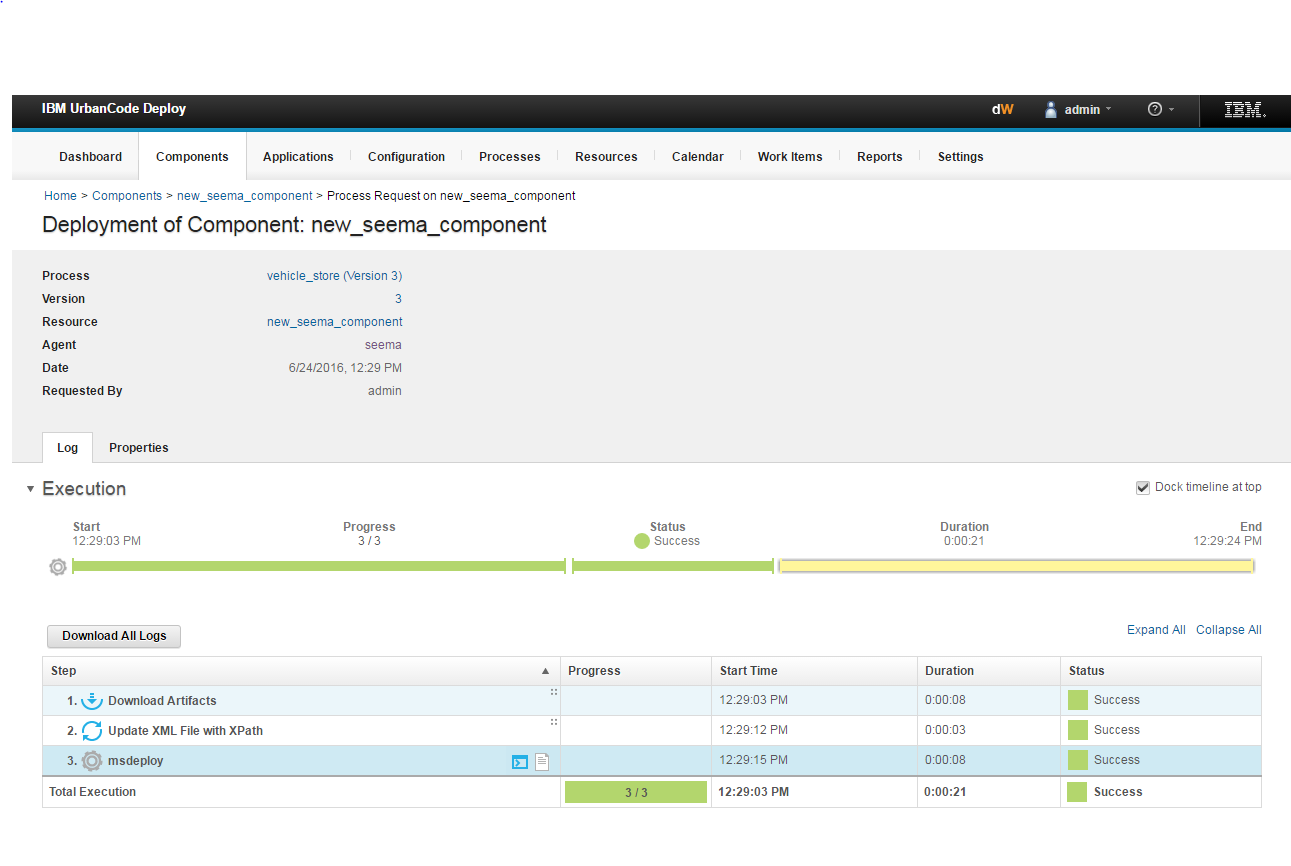
Once it is done build the job it will generate the build number which will be the version required in the UDeploy.



xvi. Then click on Run Process in UDeploy and select your environment and process and select the version from the dropdown 🡺 Click on Submit 🡺 it will deploy the application in the IIS Server.

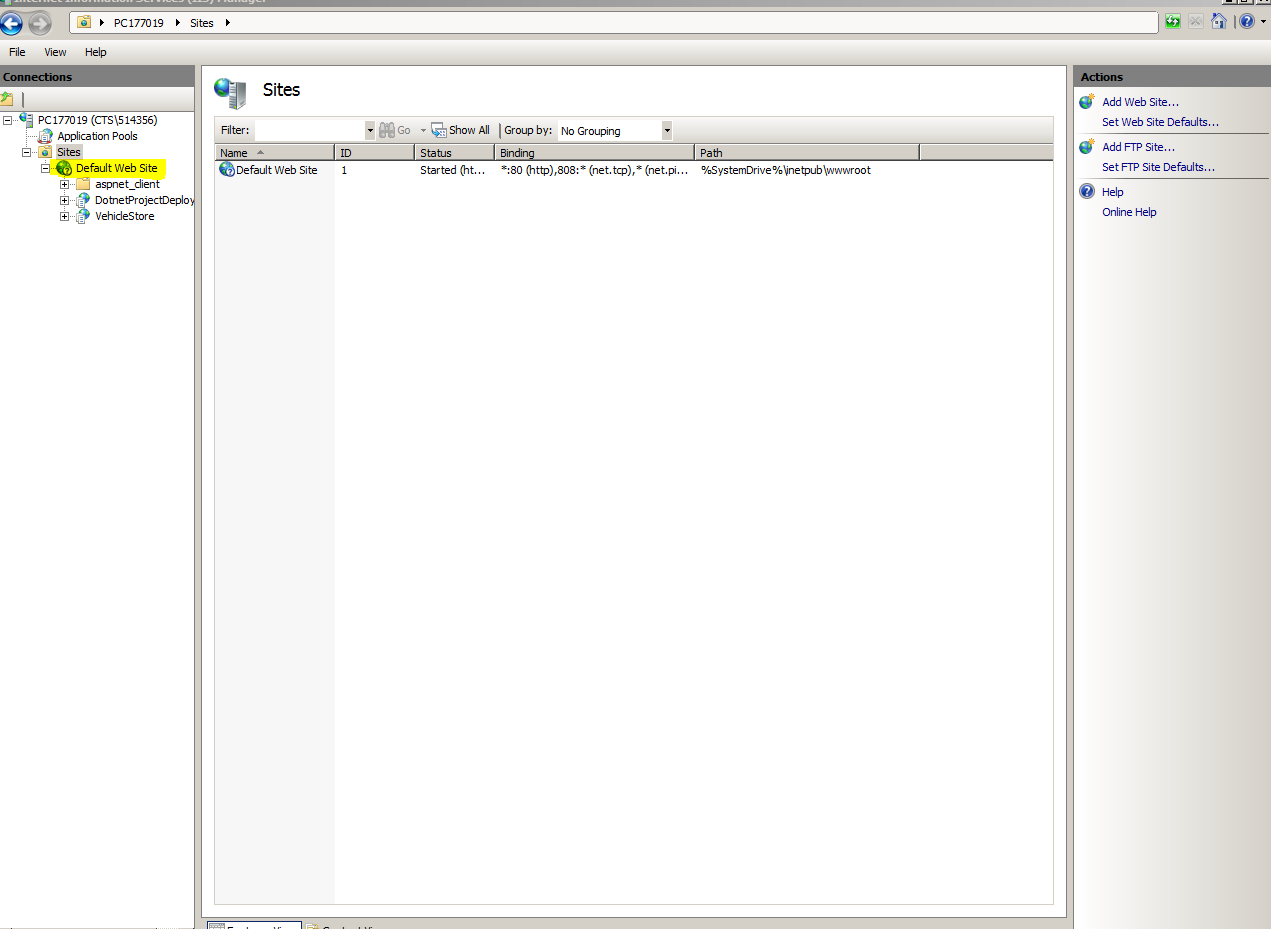


Click on Submit and RUN the process again by clicking on the RUN PROCESS.

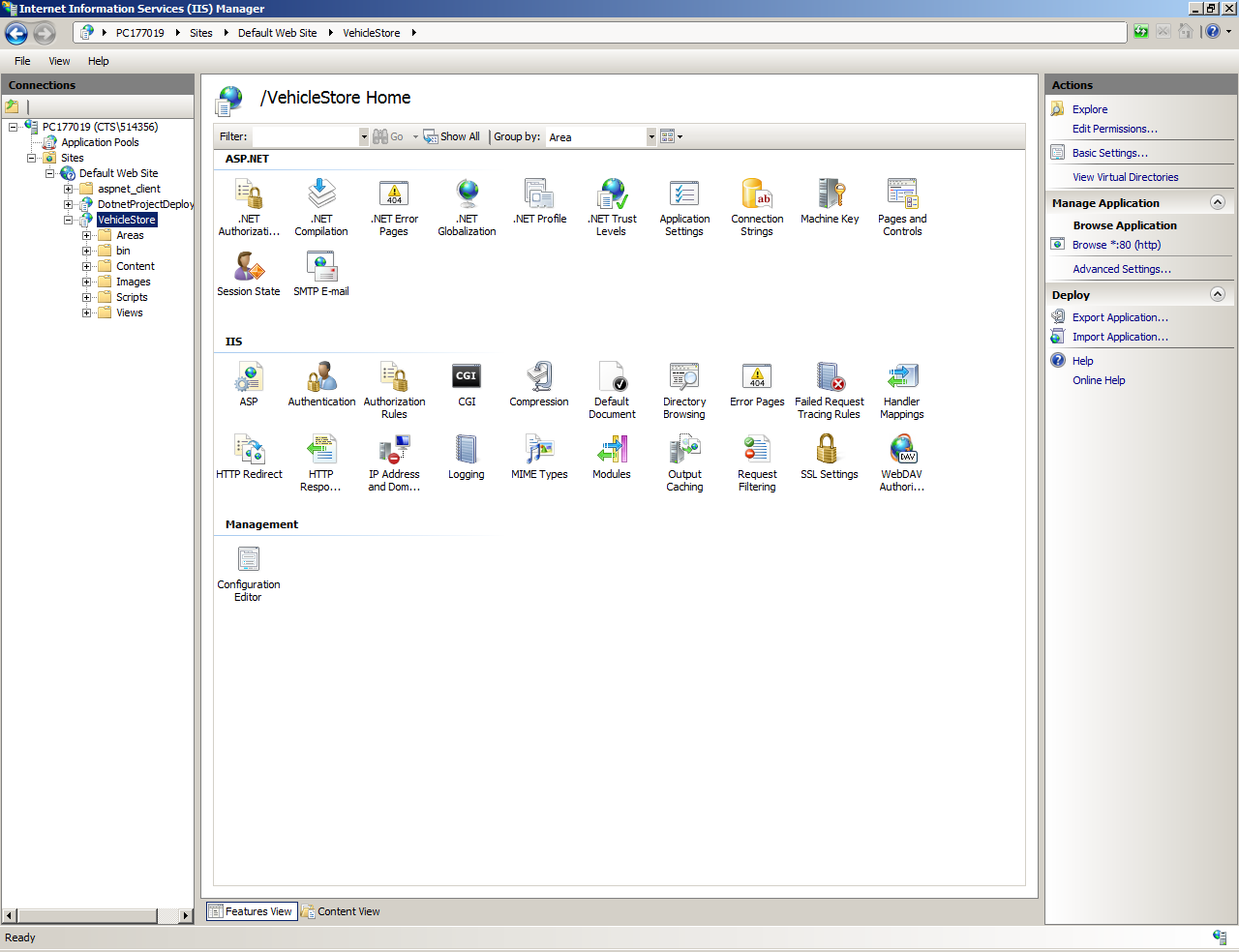


# Deployment In IIS Server

1. Open Internet Information Service (IIS) Manager 🡺 click on the sites 🡺click on the default web site.
2. The deployed website will be in the Default web Site.

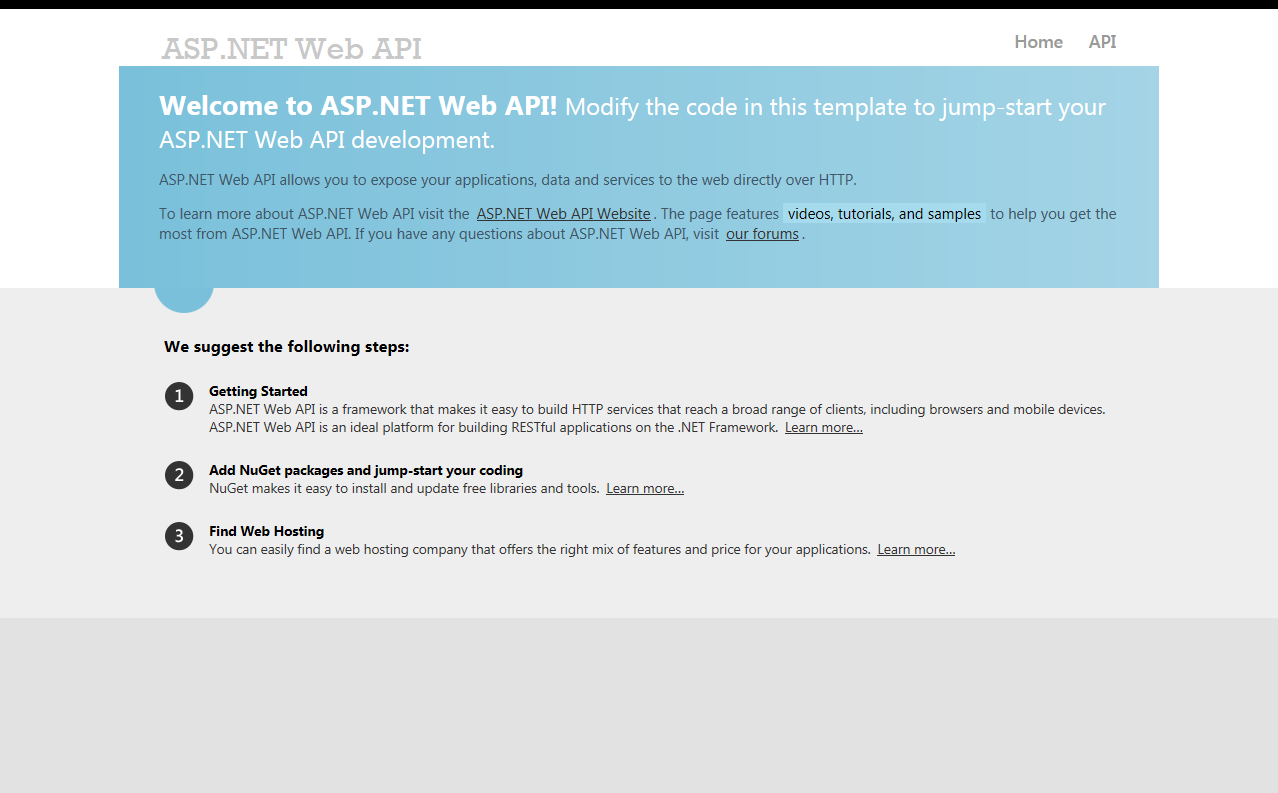


1. Expand default web site, the application will be visible over here. Now, to view the web page click on the deployed project name and then Browse \*:80(http) on the right side of IIS server.



## **WEB DEPLOYED**

The web page will be displayed like this (Your web app will be deployed over here):

****

Click on the API it will redirect you automatically to the ASP.NET Web API Page.

