CLM DEVOPS POC

Contents

[1. Force.com IDE Plugin Installation 1](#_Toc480373332)

[2. Salesforce Connection 3](#_Toc480373333)

[3. GIT Repository 6](#_Toc480373334)

[4.Project Sharing 9](#_Toc480373335)

[5. Code Commit and Push 11](#_Toc480373336)

[6. ANT Installation and Setup 15](#_Toc480373337)

[7. Force.com Migration Tool 17](#_Toc480373338)

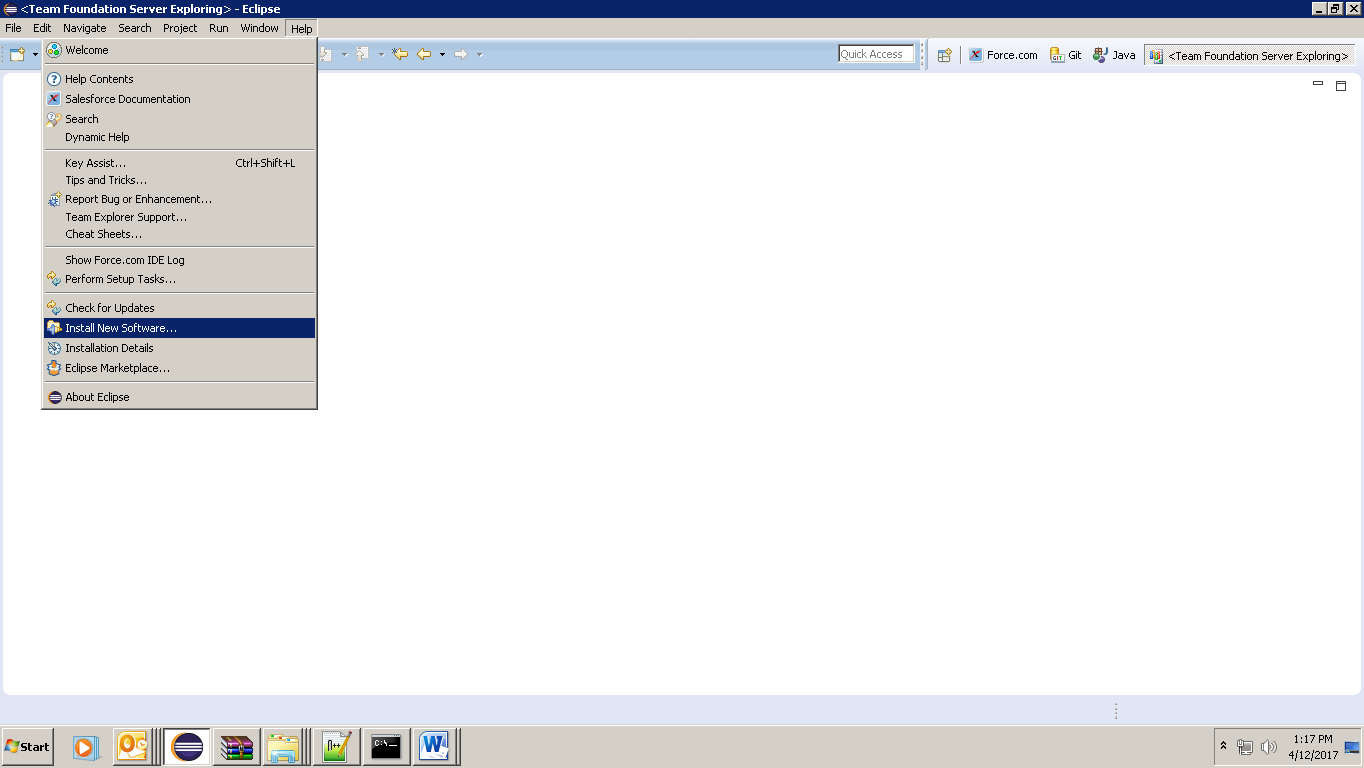
[8. Jenkins Setup 20](#_Toc480373339)

[9. Job Creation 25](#_Toc480373340)

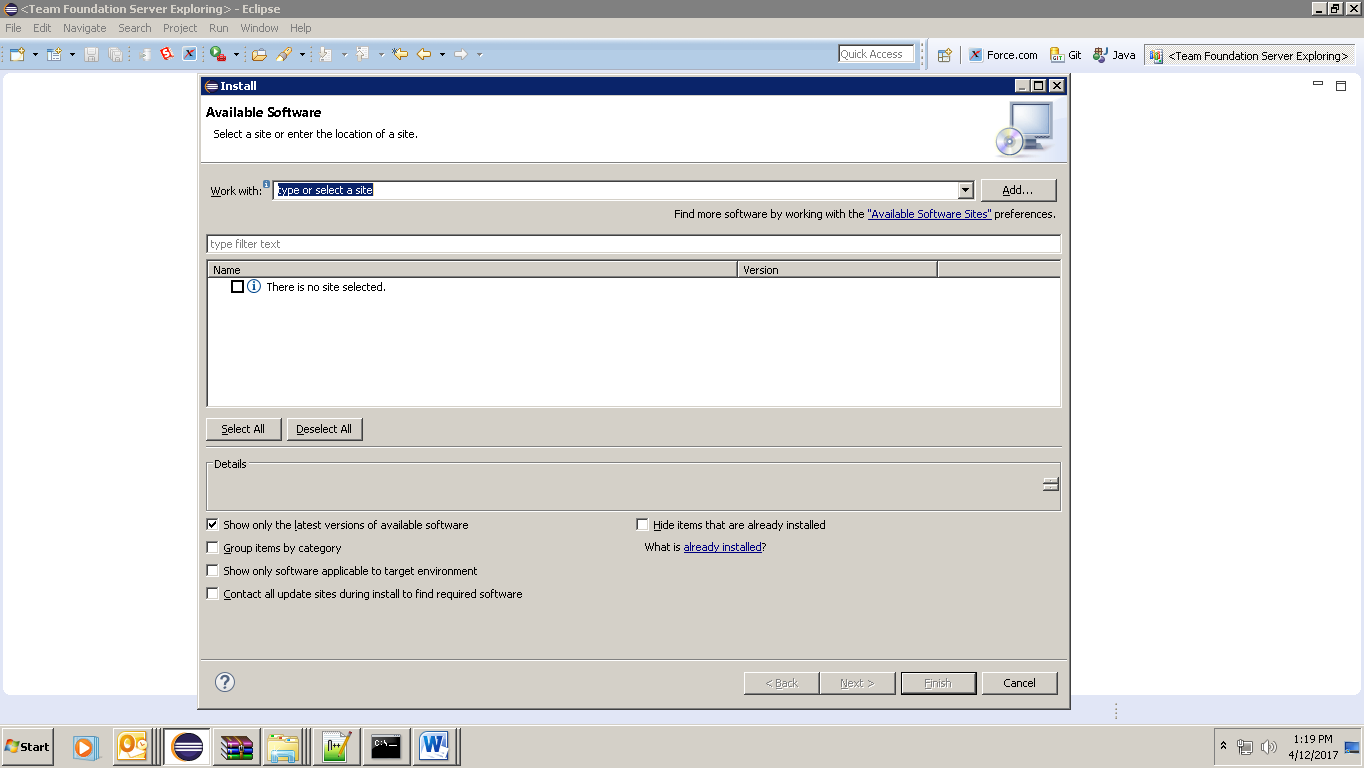
[10. Test 27](#_Toc480373341)

# 1. Force.com IDE Plugin Installation:

Step 1: Go to Eclipse>Help>Install New Software



Step 2: Click Add



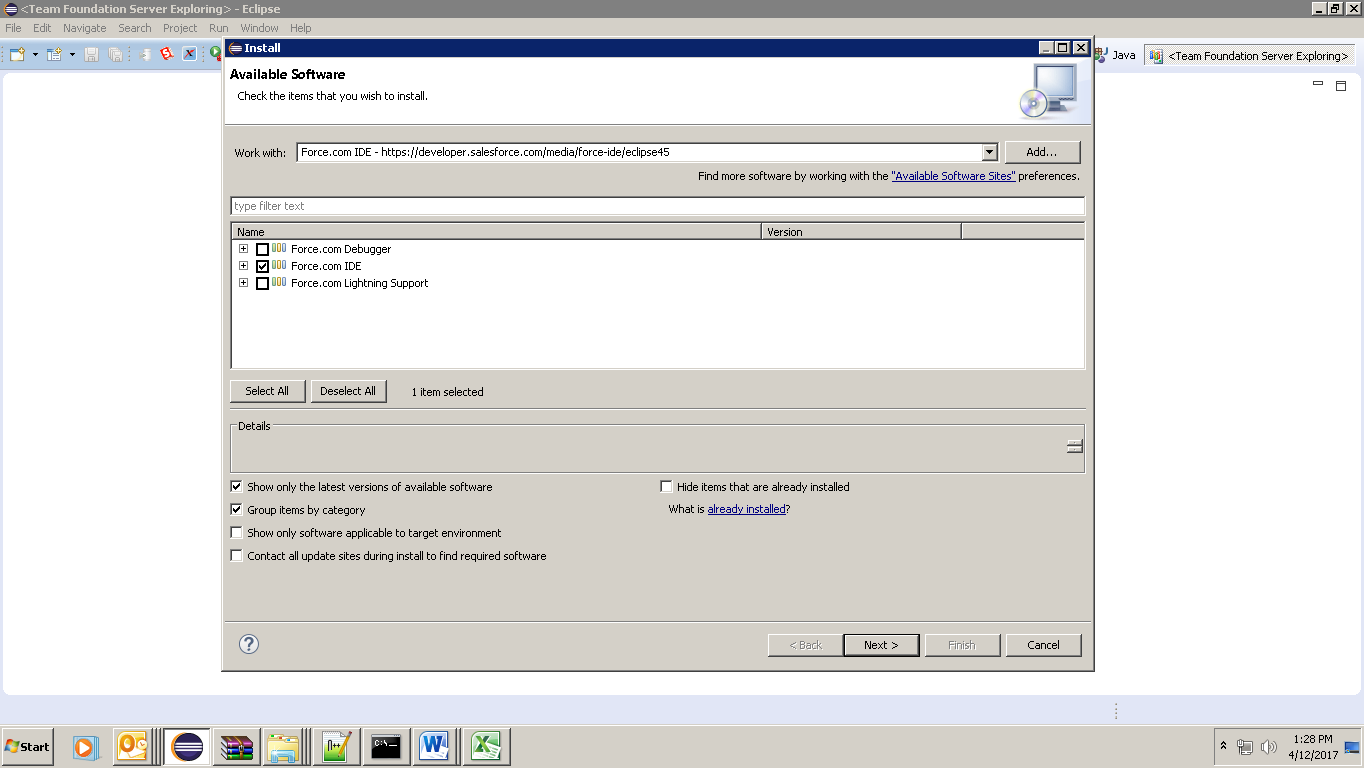
Step 3: Enter below details and click ok

Name: Force.com IDE Location

Location: <https://developer.salesforce.com/media/force-ide/eclipse45>

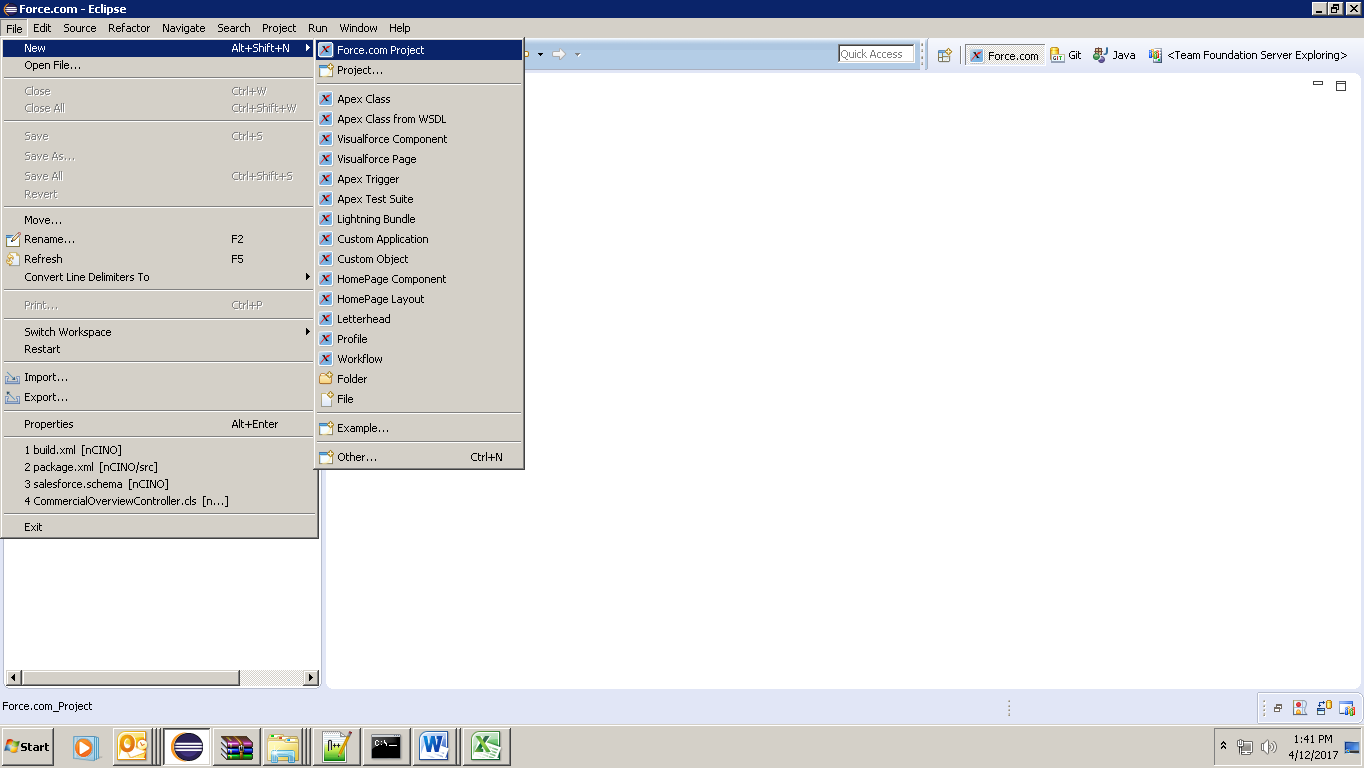


Step 4: select Force.com IDE and click Next and Finish Installation

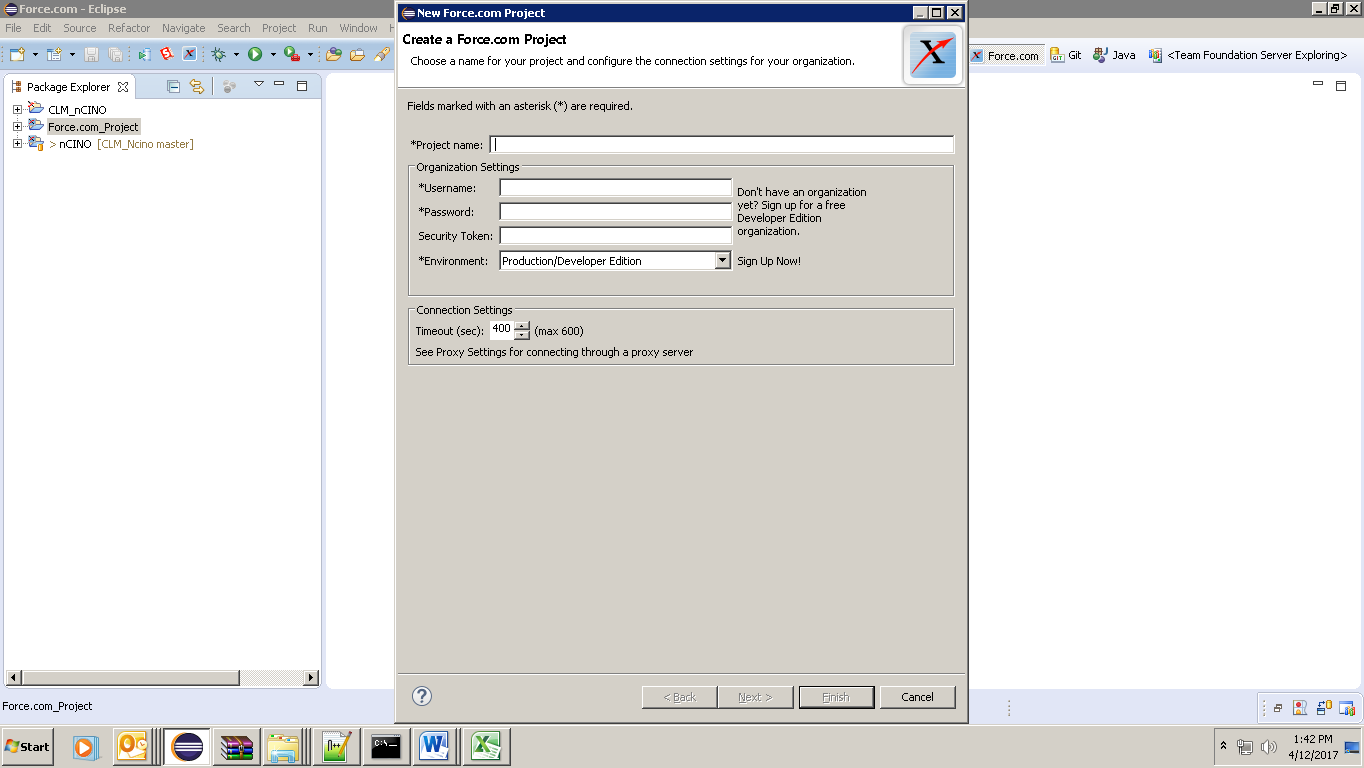


# 2. Salesforce Connection:

Step 1: Create a New Force.com Project in Eclipse



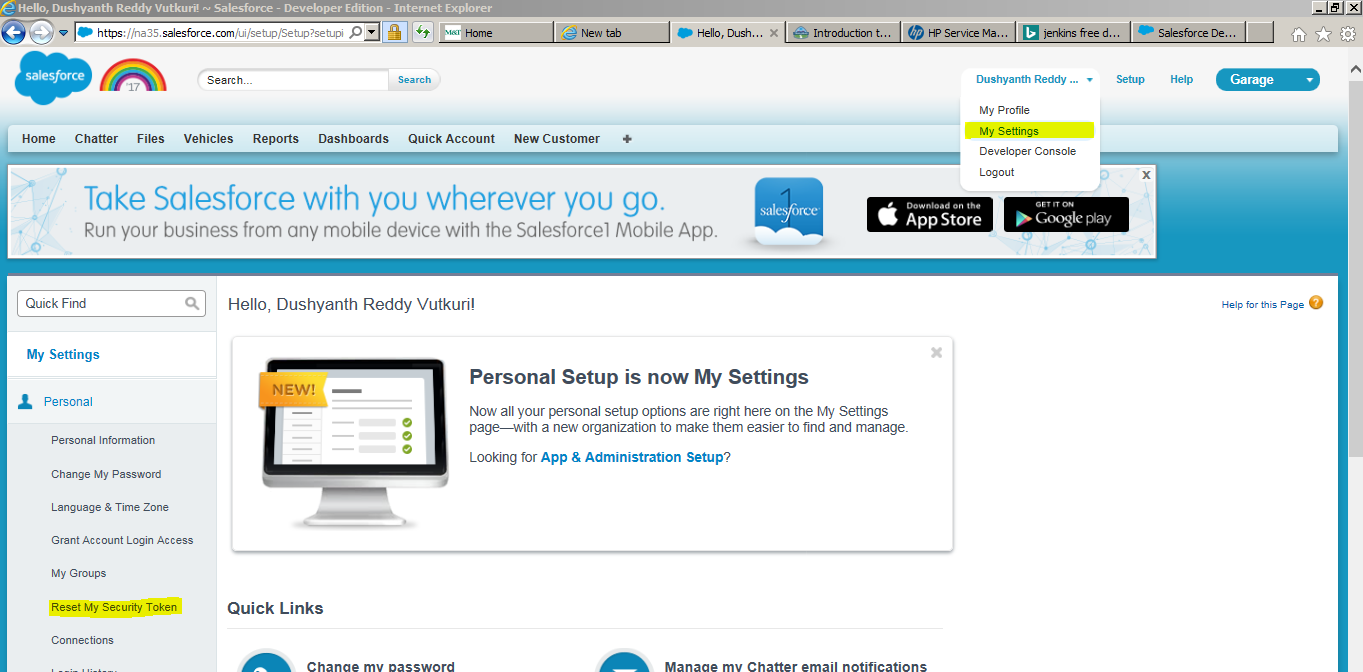
Step 2: Give required details



Username: Your salesforce login username

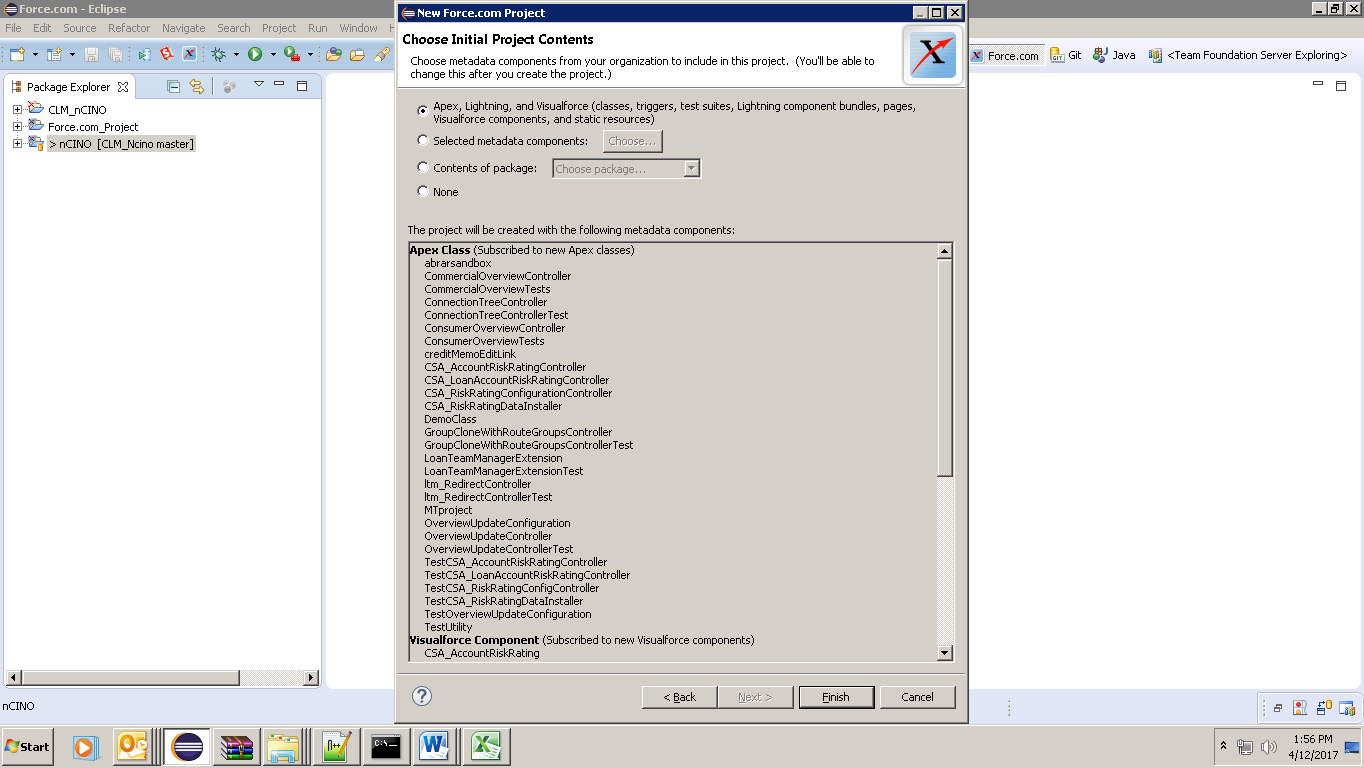
Password: Your salesforce login Password

Security Token: If you don’t know, you can reset your security key from salesforce. You will receive it to your registered email.



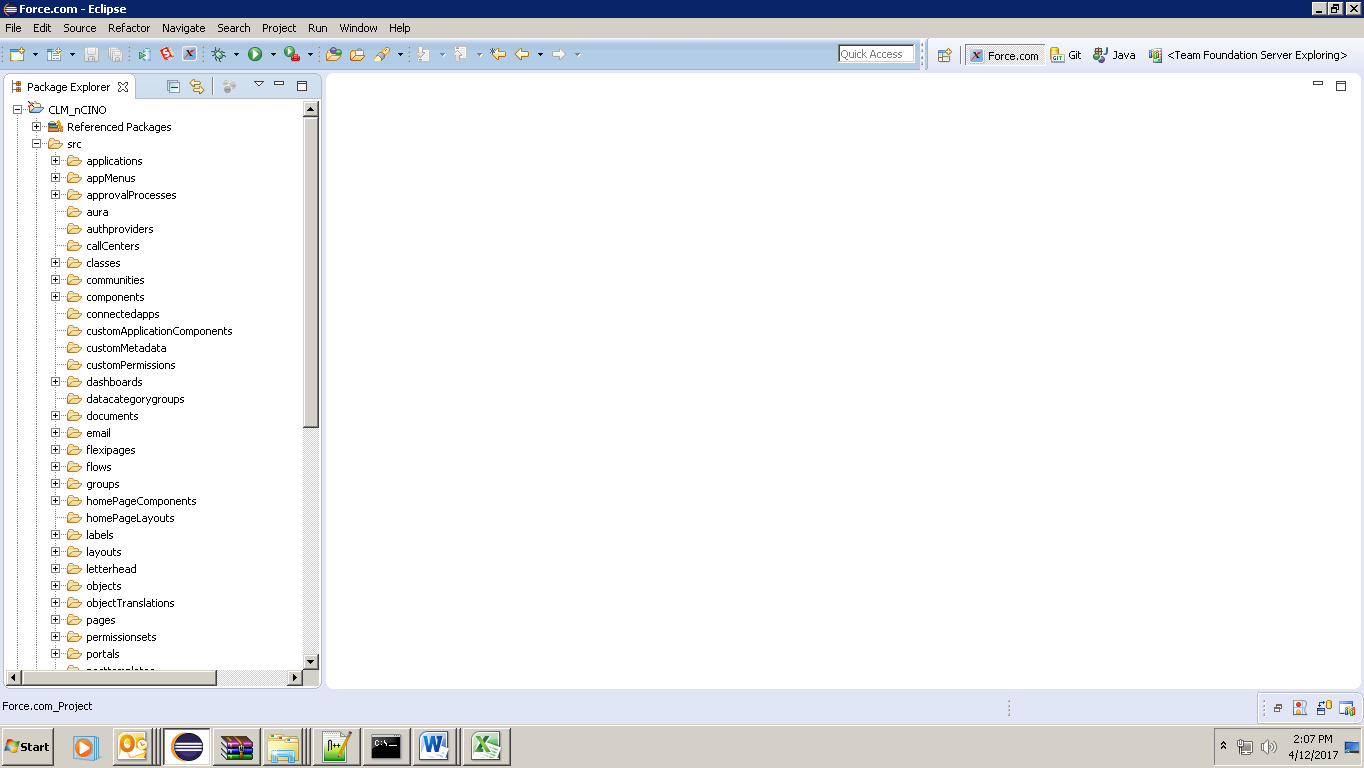
Environment: If you are connecting to a sandbox, select sandbox.

Step 3: click next – You would see something like this



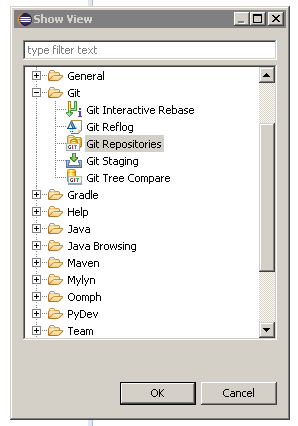
You can simply click on finish to create a new salesforce project or you can also browse the required metadata components from your org/sandbox.

You would see all the packages in your eclipse which are available in salesforce.

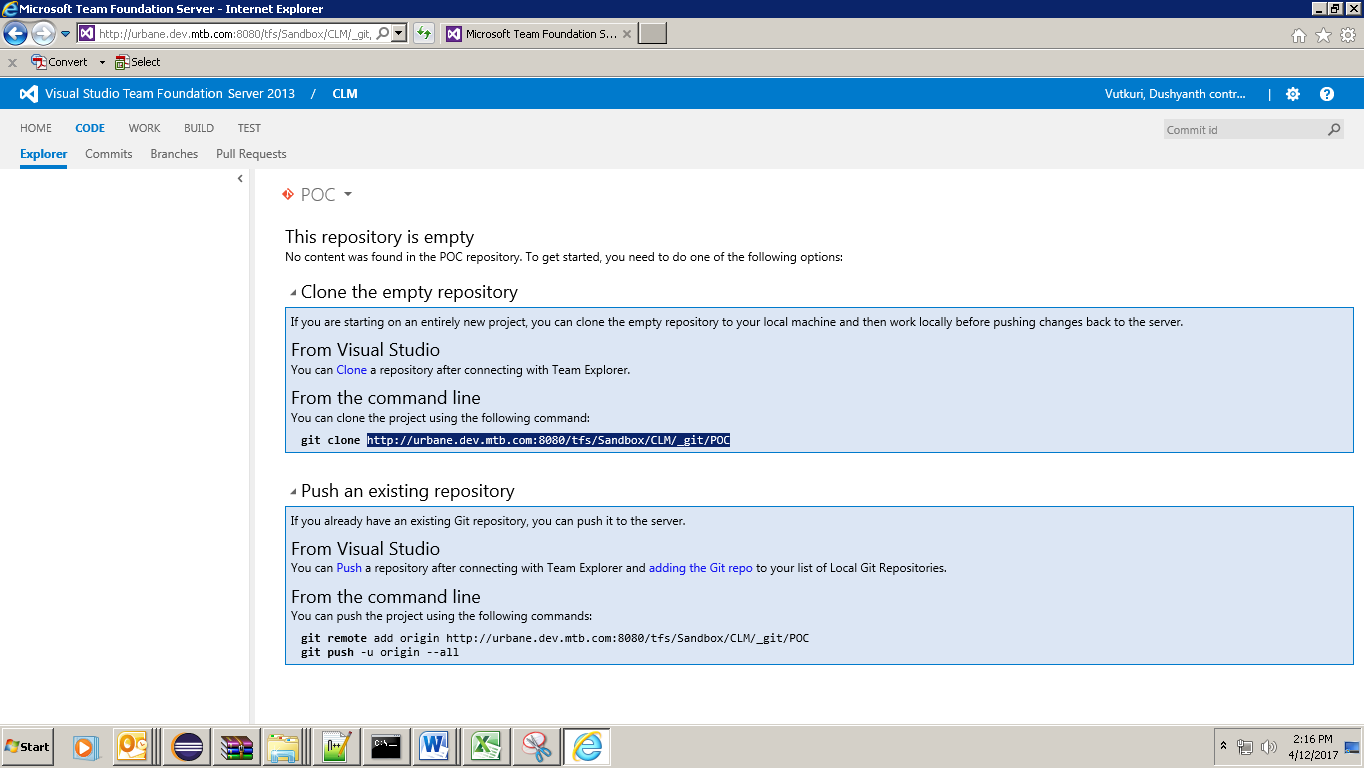


# 3. GIT Repository:

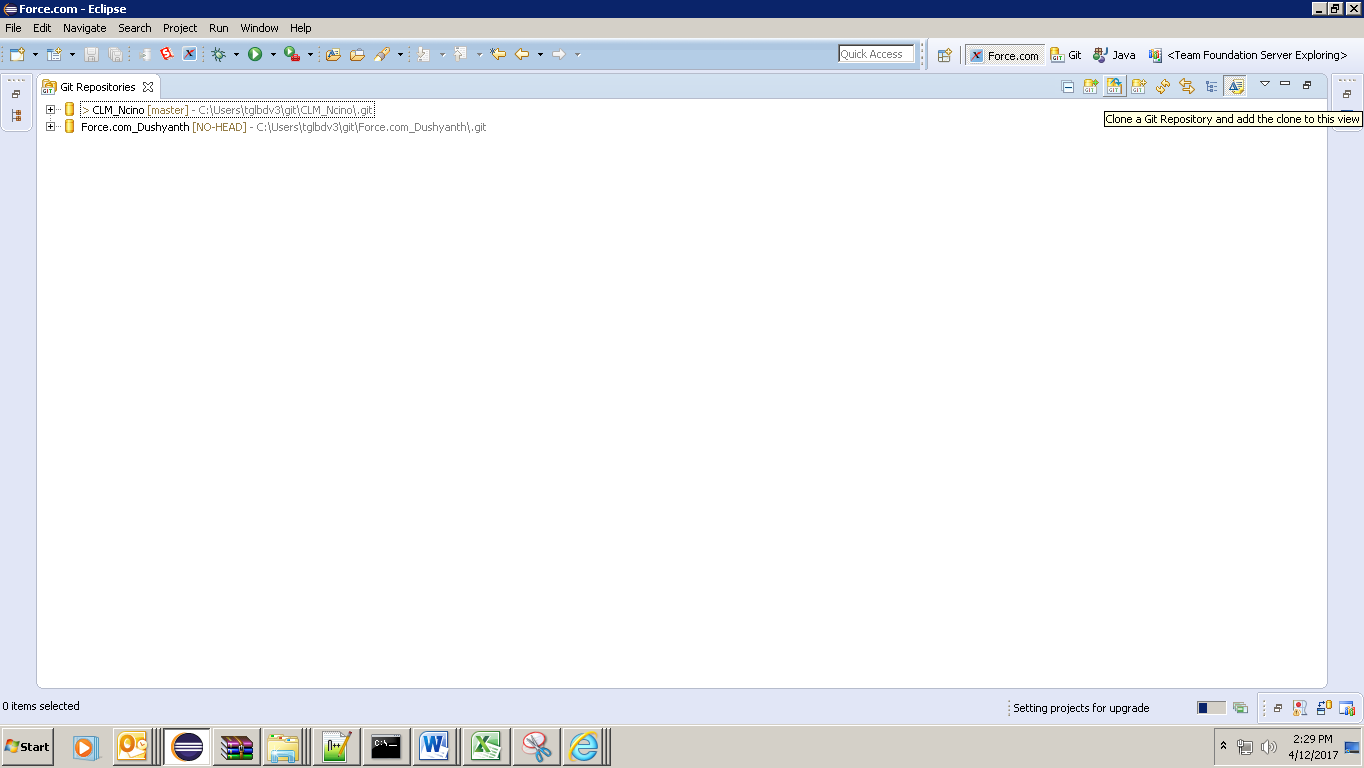
If you Installed Eclipse from M&T Software Catalog, GIT Plugin is already packaged in Eclipse.

Step 1: Go to GIT Repositories - Window>Showview>Other>Git>Git Repositories 

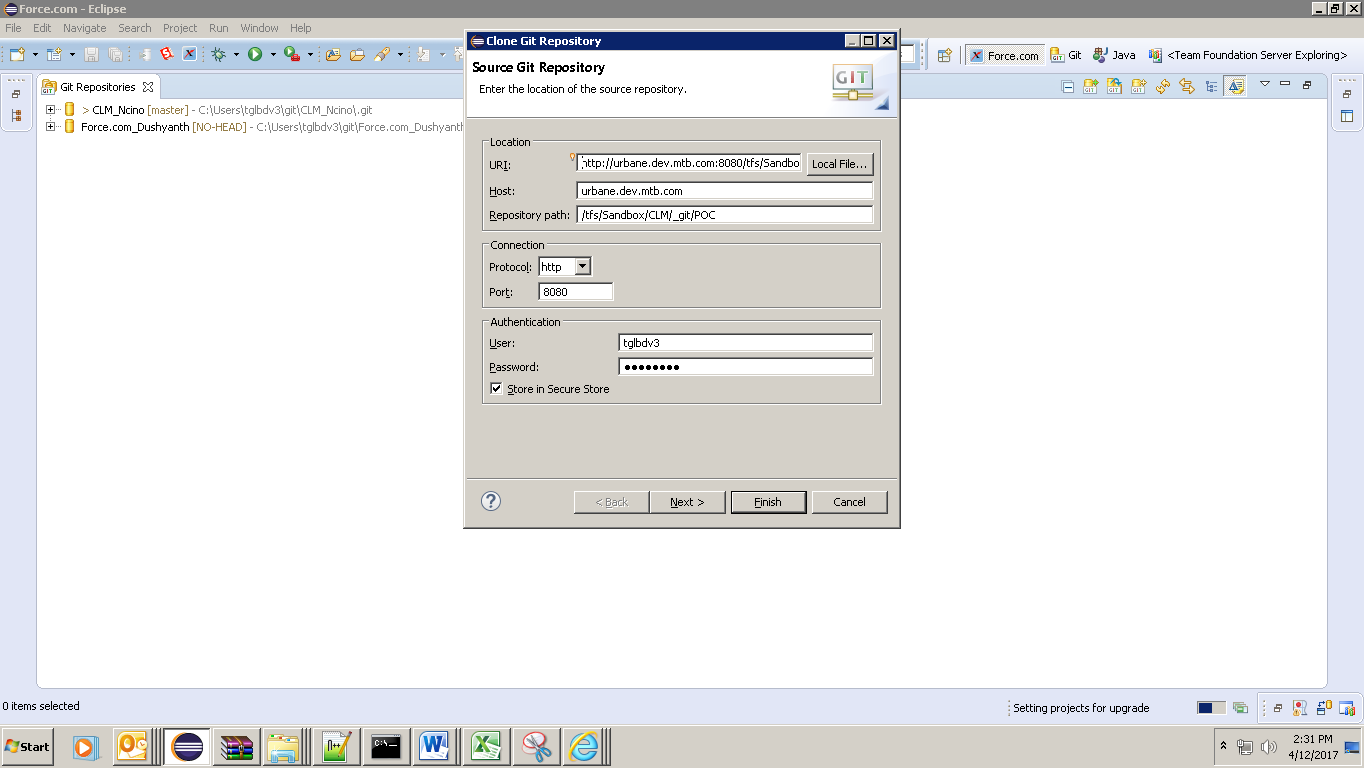
Step 2: Copy URI from your Remote GIT source code repository (Github/ TFS Git Repository Project)

Here I have a TFS Project GIT Repository

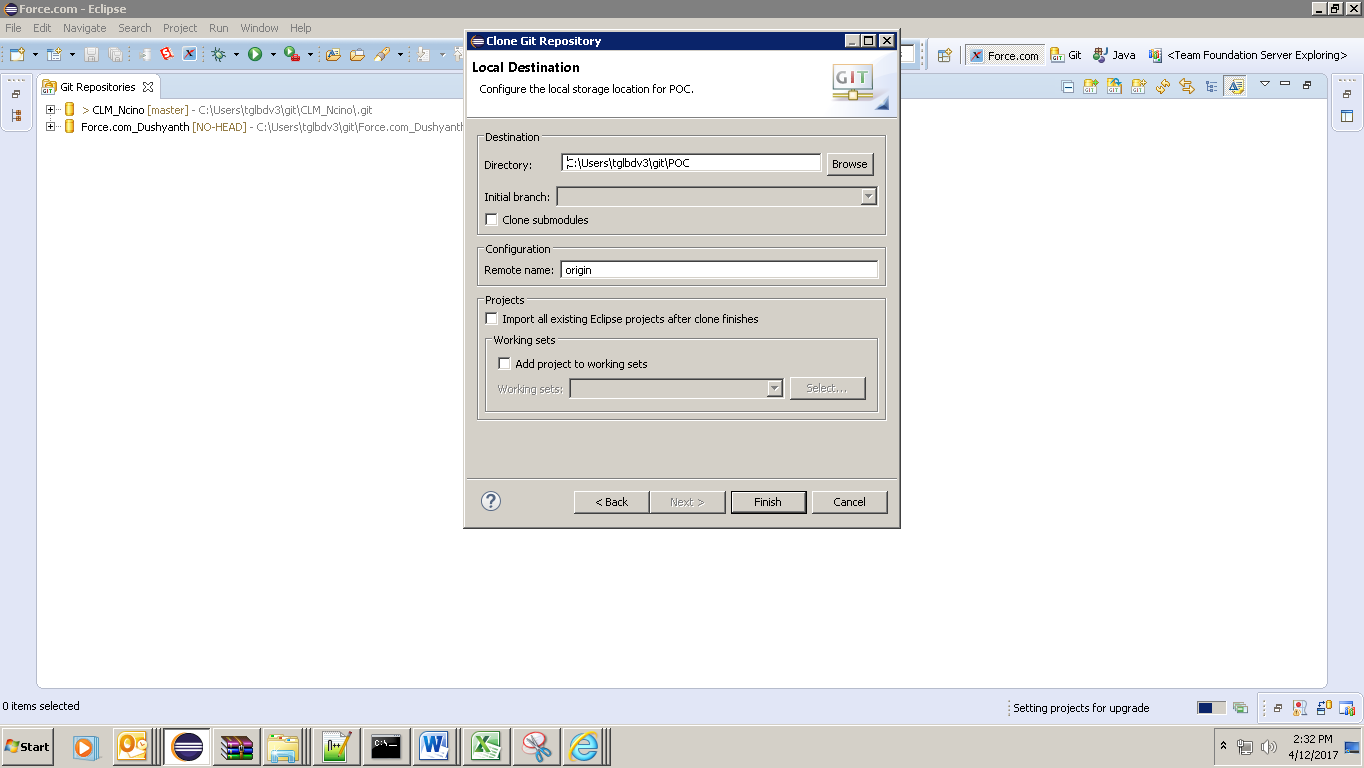
Step 3: Click on Clone a GIT Repository



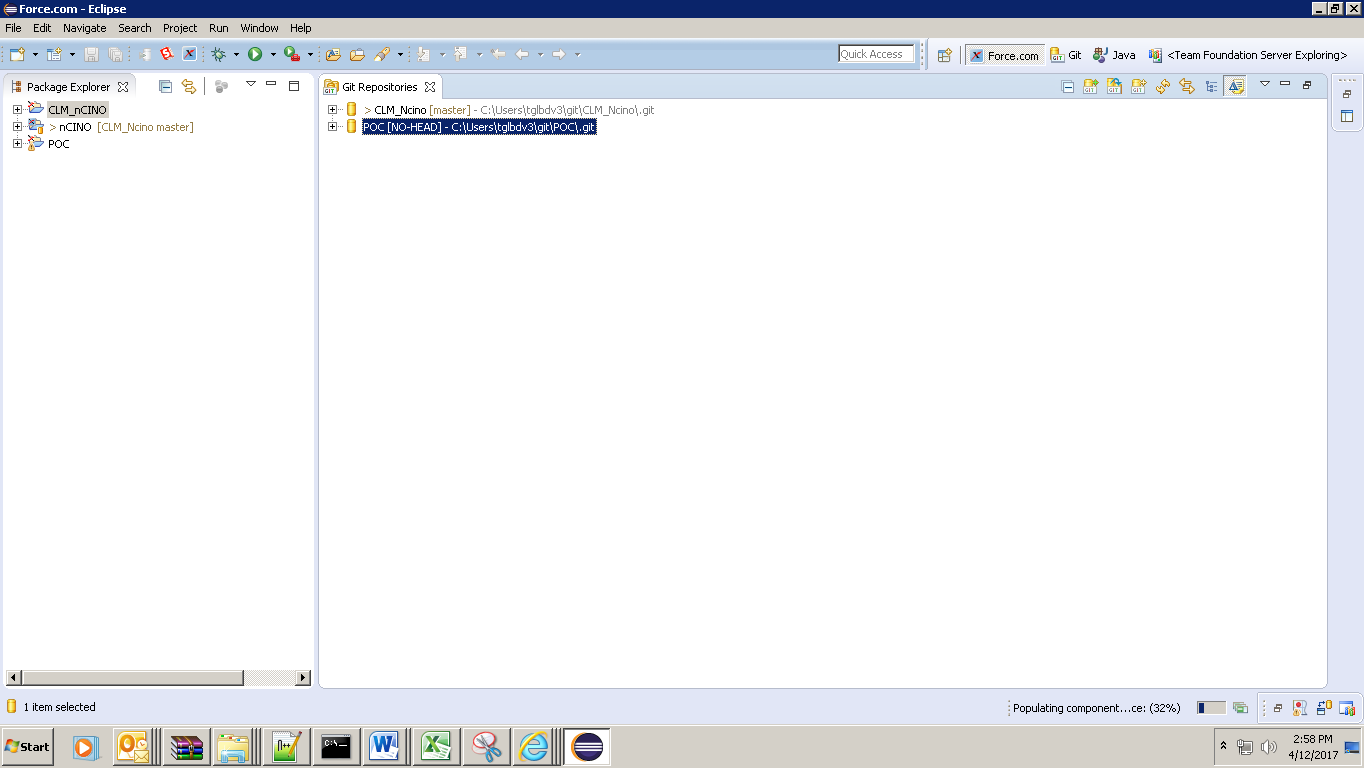
Step 4: Paste the URI, it will auto fill the details.



Step 5: Click Next and Next and Finish. The directory path is your local repository.

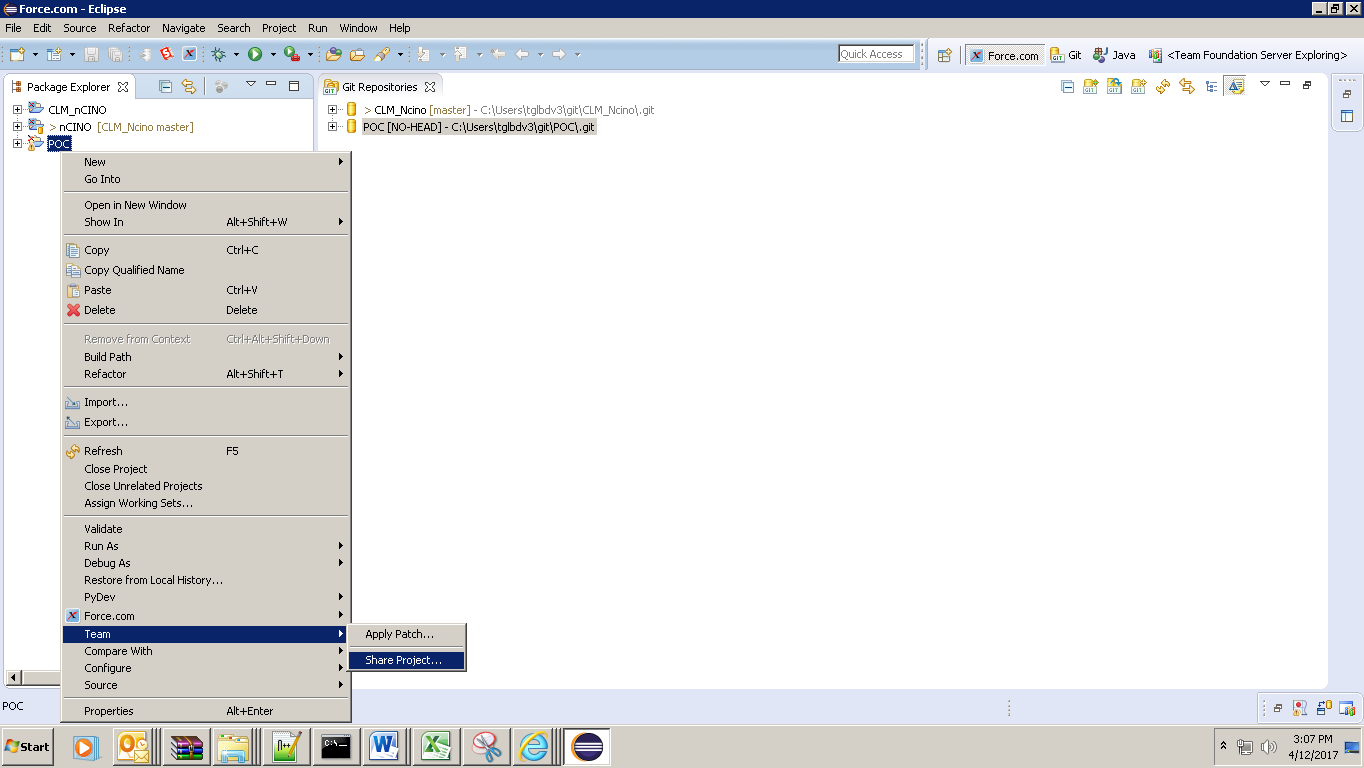


Step6: Repository is created in local C:\Users\tglbdv3\git\POC

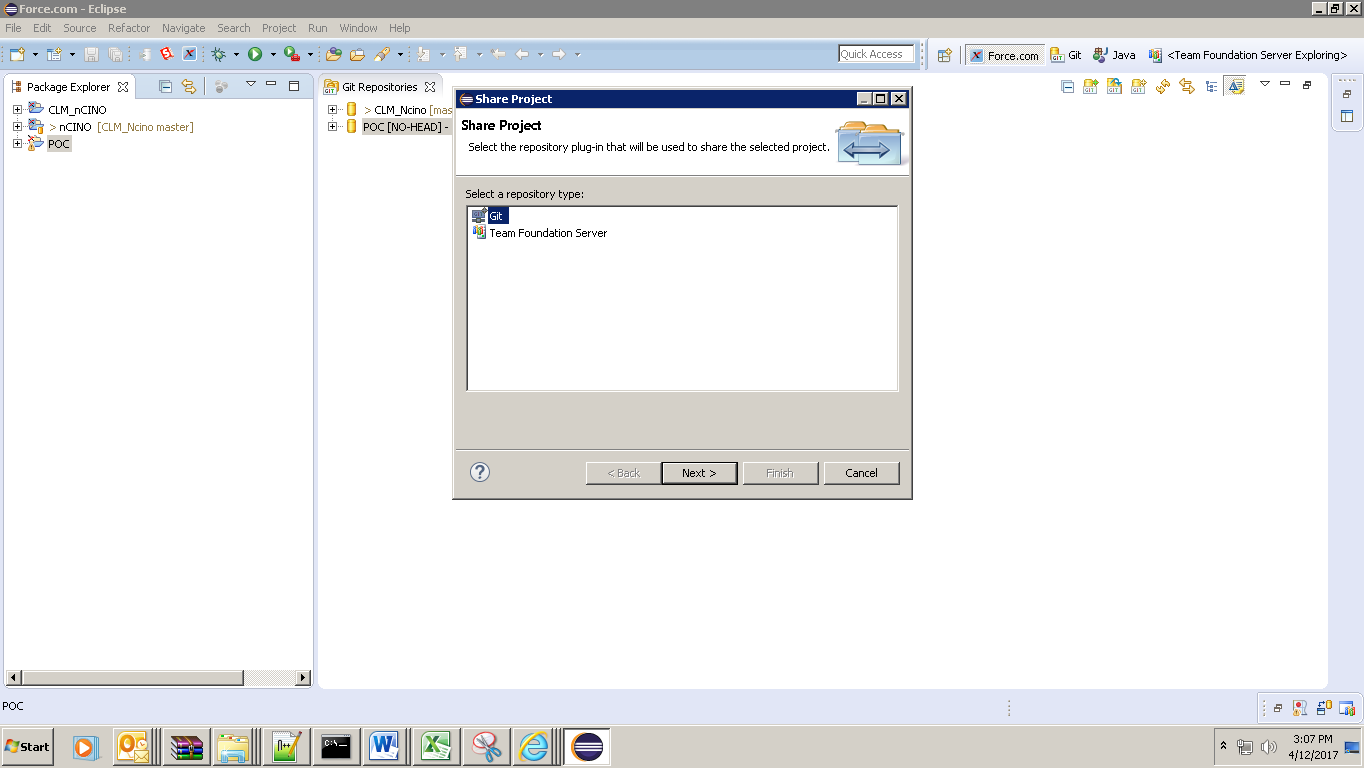


# 4. Project Sharing:

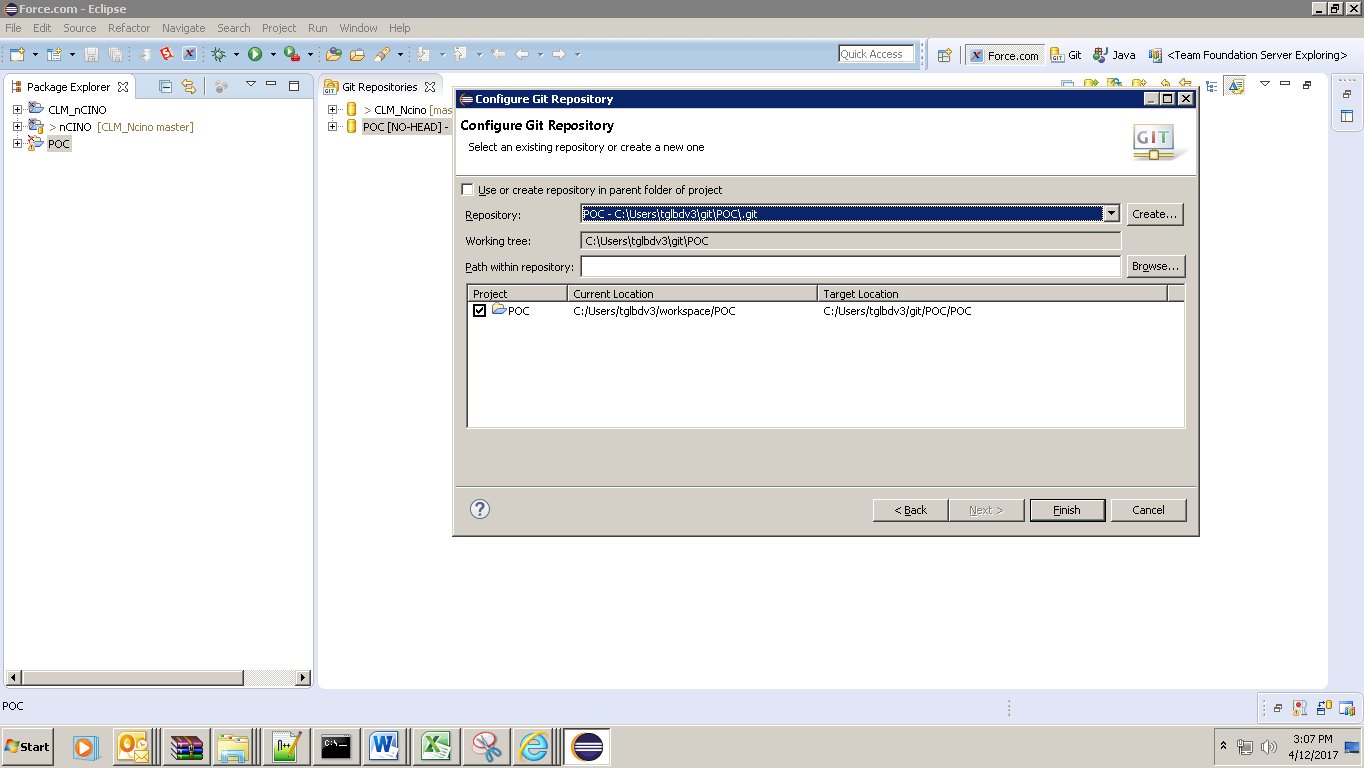
Step 1: Right Click on project>Team>share project



Step 2: Select GIT

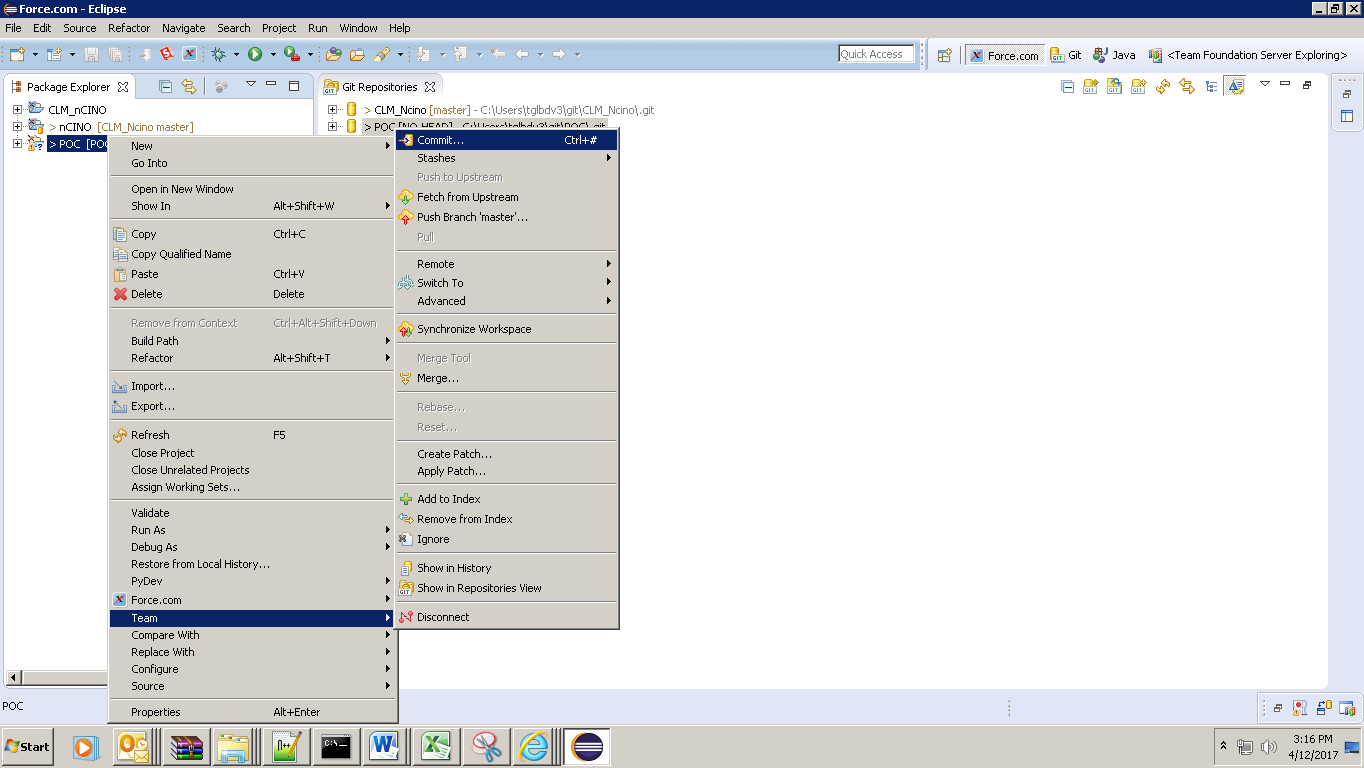


Step 3: Choose the Repository to which you want to share the project from dropdown and click Finish. This will connect the Eclipse project to GIT repository.

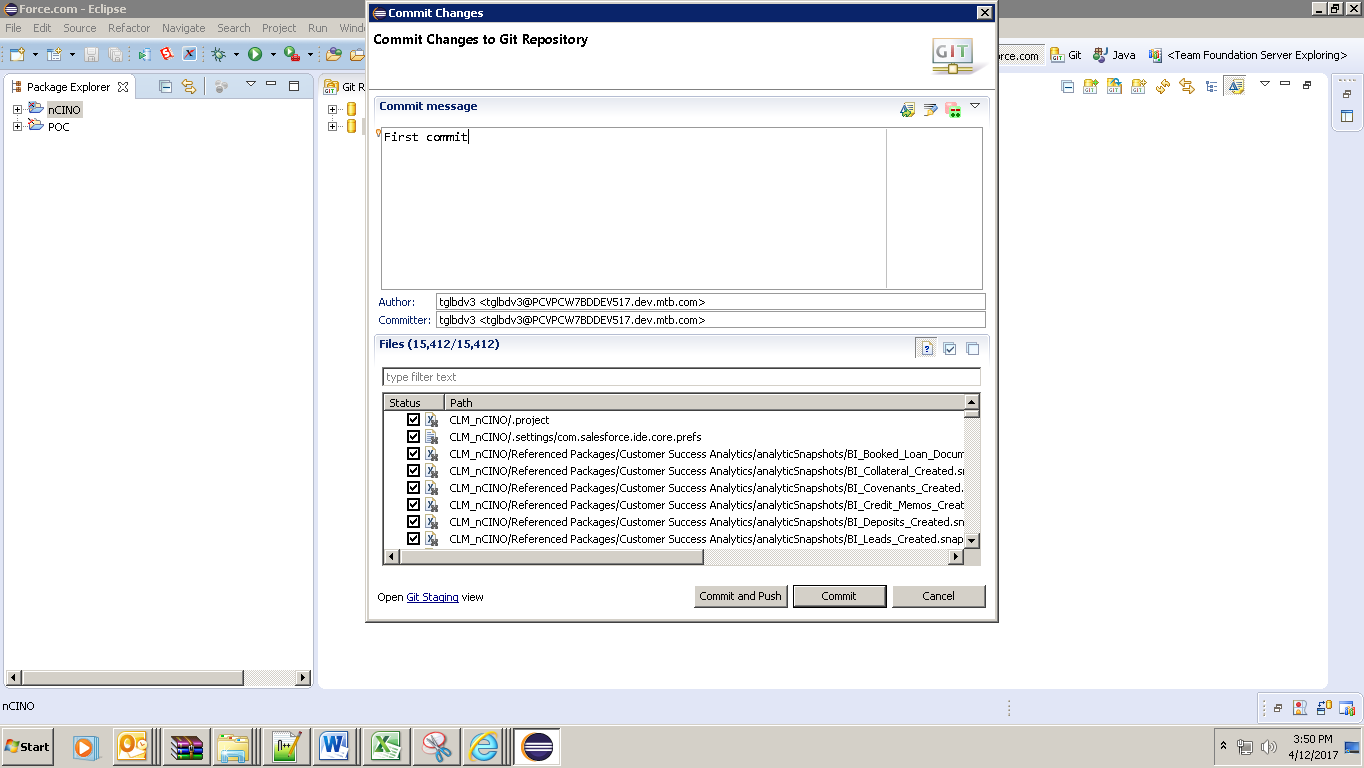


# 5. Code Commit and Push:

Step 1: Right click on project Team>Commit

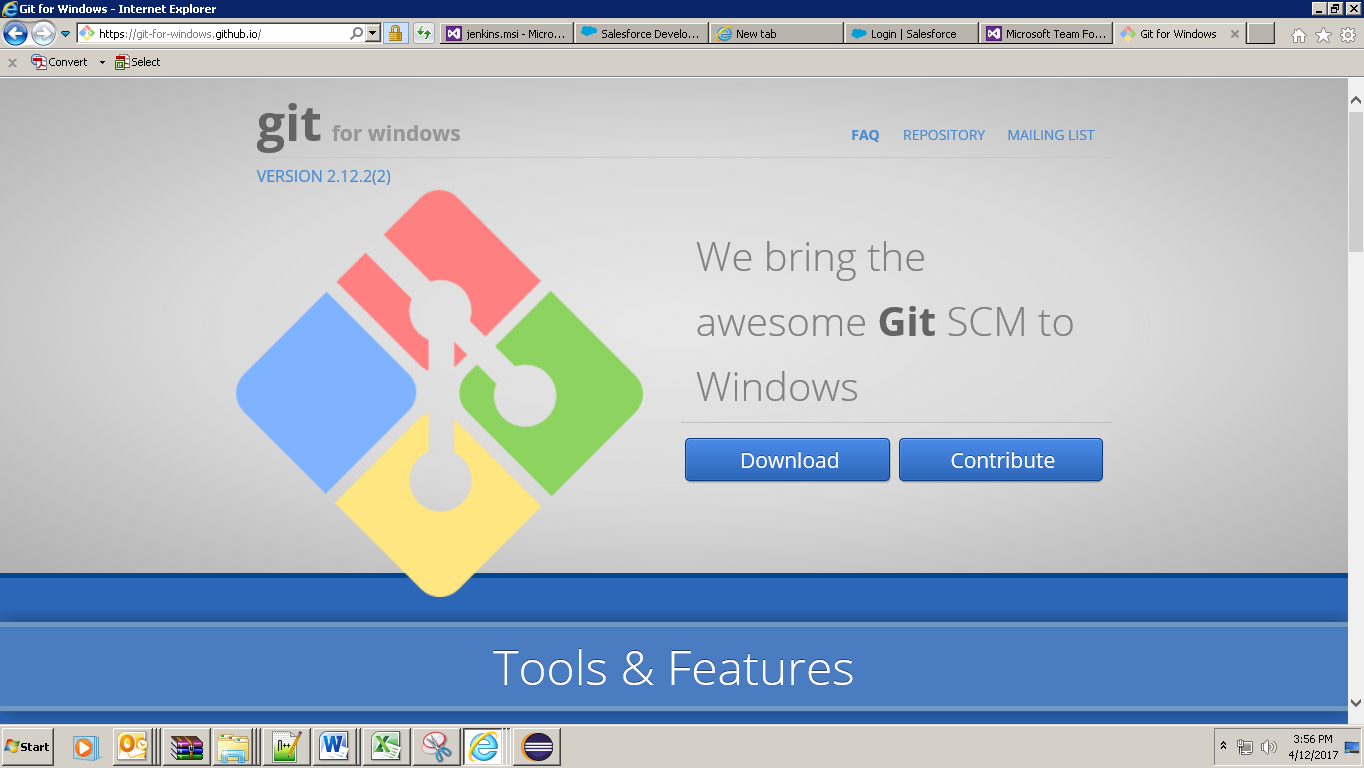


Step2: Select all and enter commit message. Code will be committed in local.

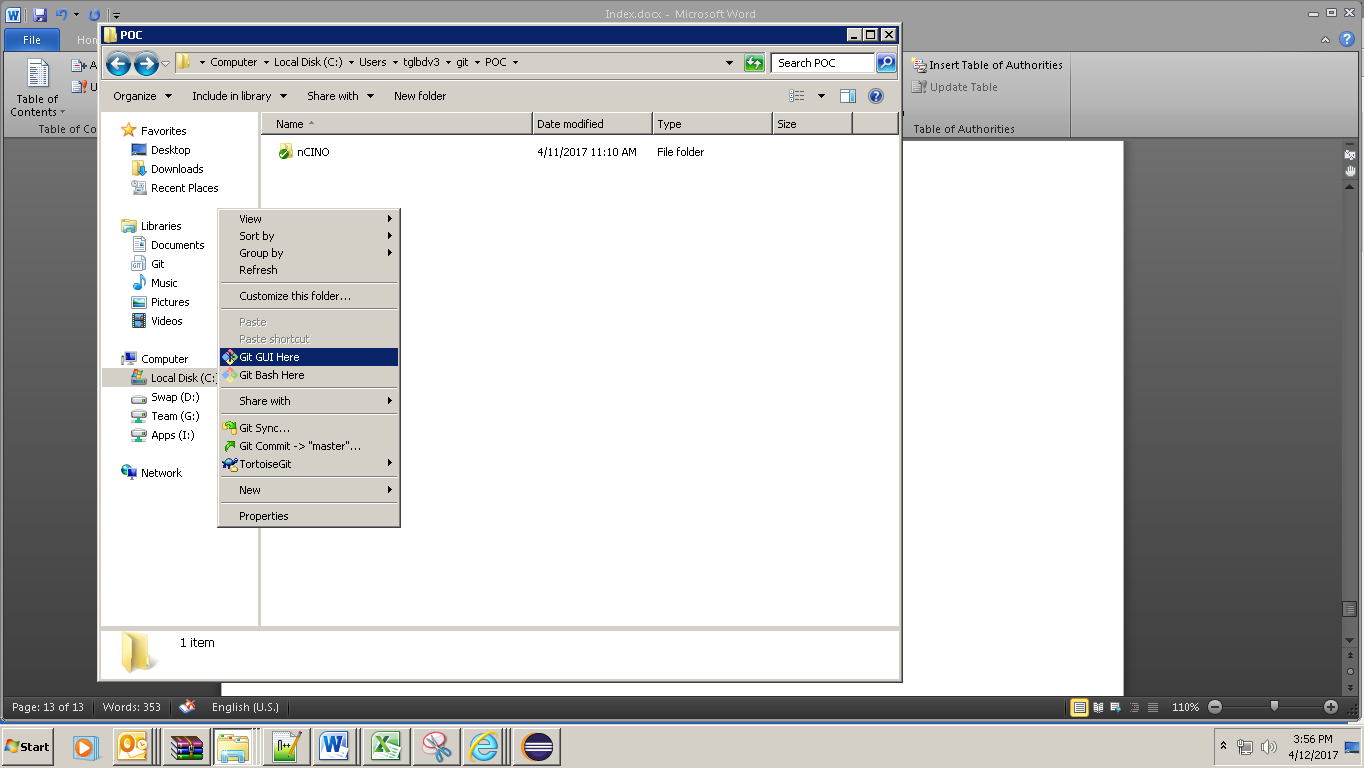


Step 3: Push – IF you are having troubles to push the code from eclipse, use GIT for windows <https://git-for-windows.github.io/>

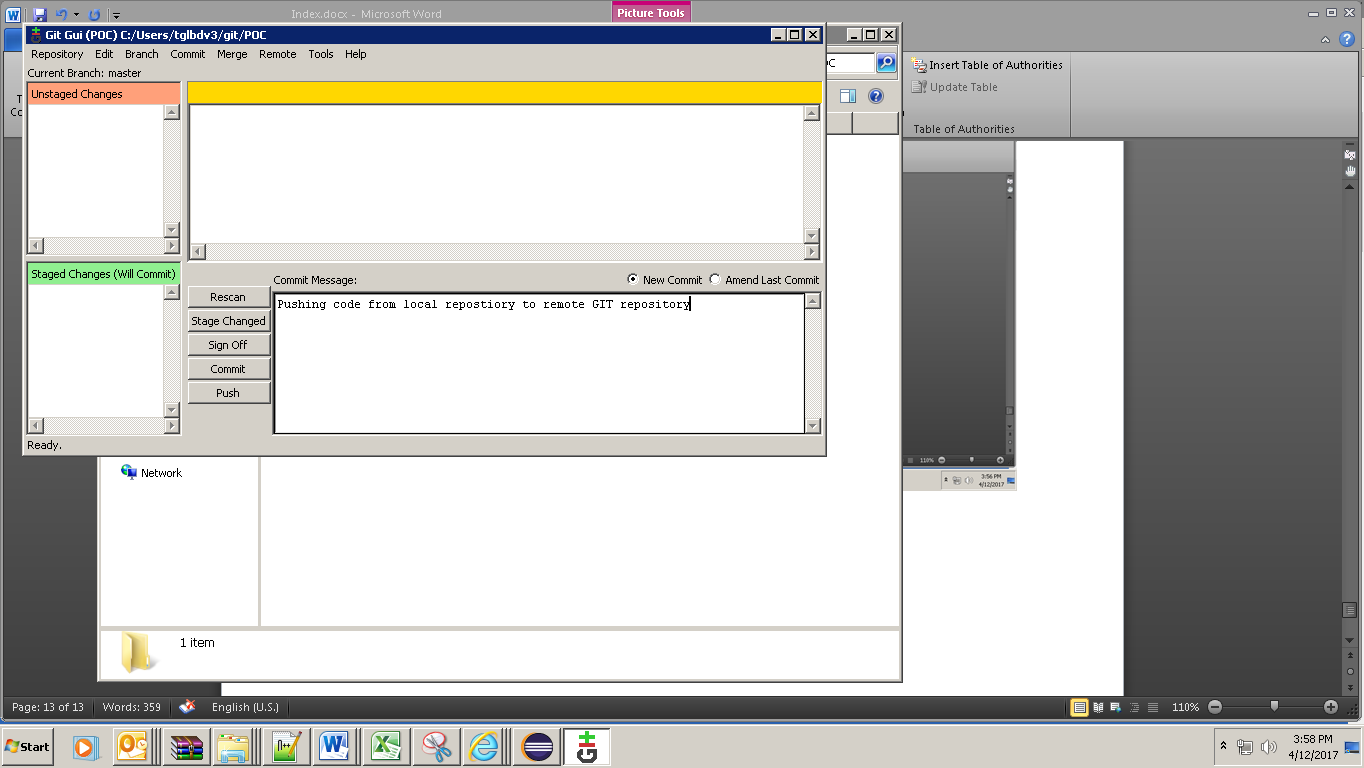
Download GIT for windows



Step 4: Go to Local Code Repository – Right click > GIT GUI here

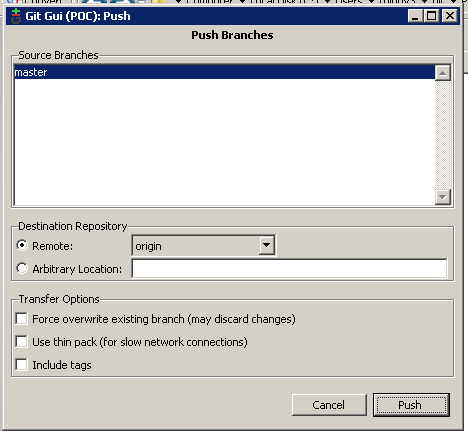


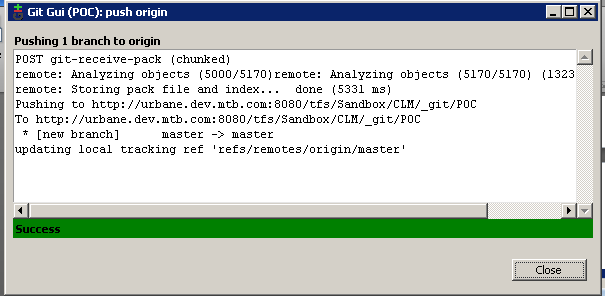
Step 5: If it asks for Database compression permission – Click yes then type Commit message and click PUSH



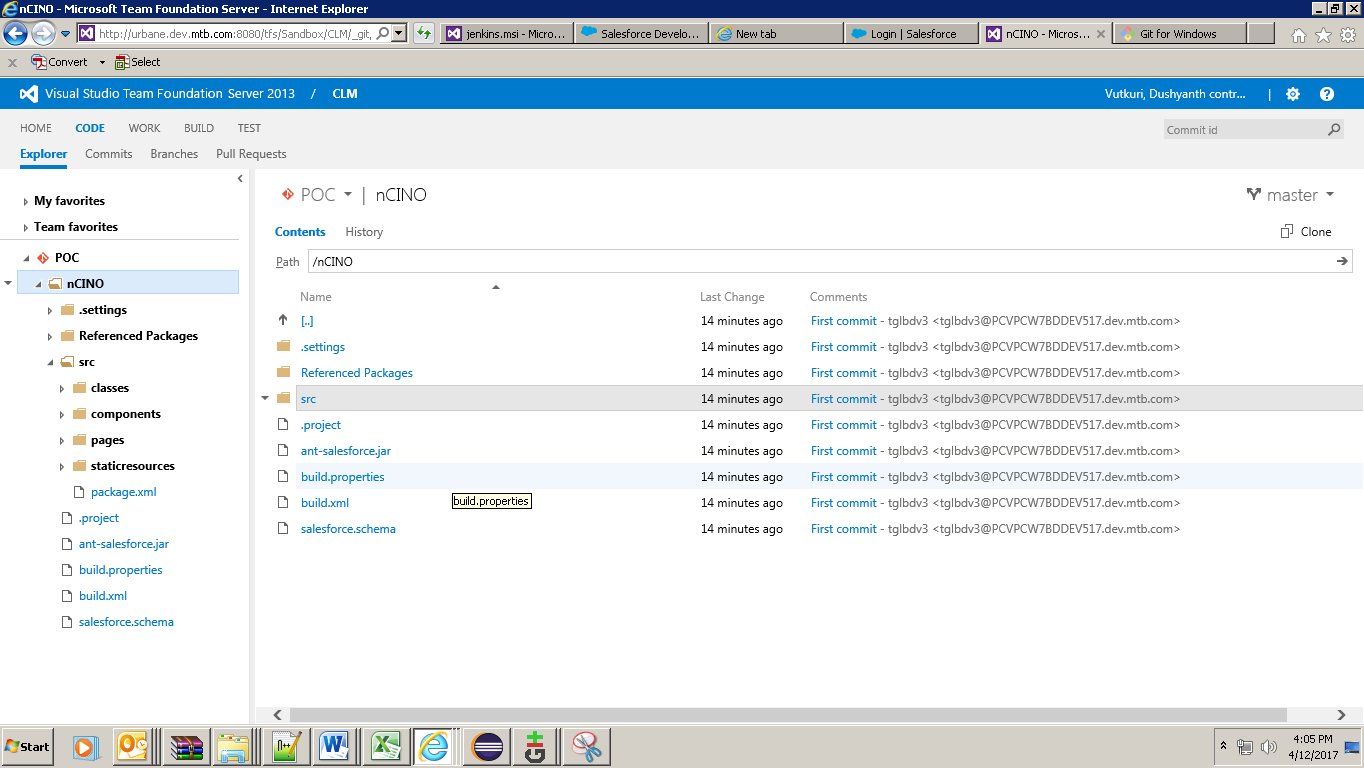
Step 6: If it asks for credentials, give the same credentials what were used while creating GIT repository. (These credentials would be your domain credentials)

Select Master and Click Push



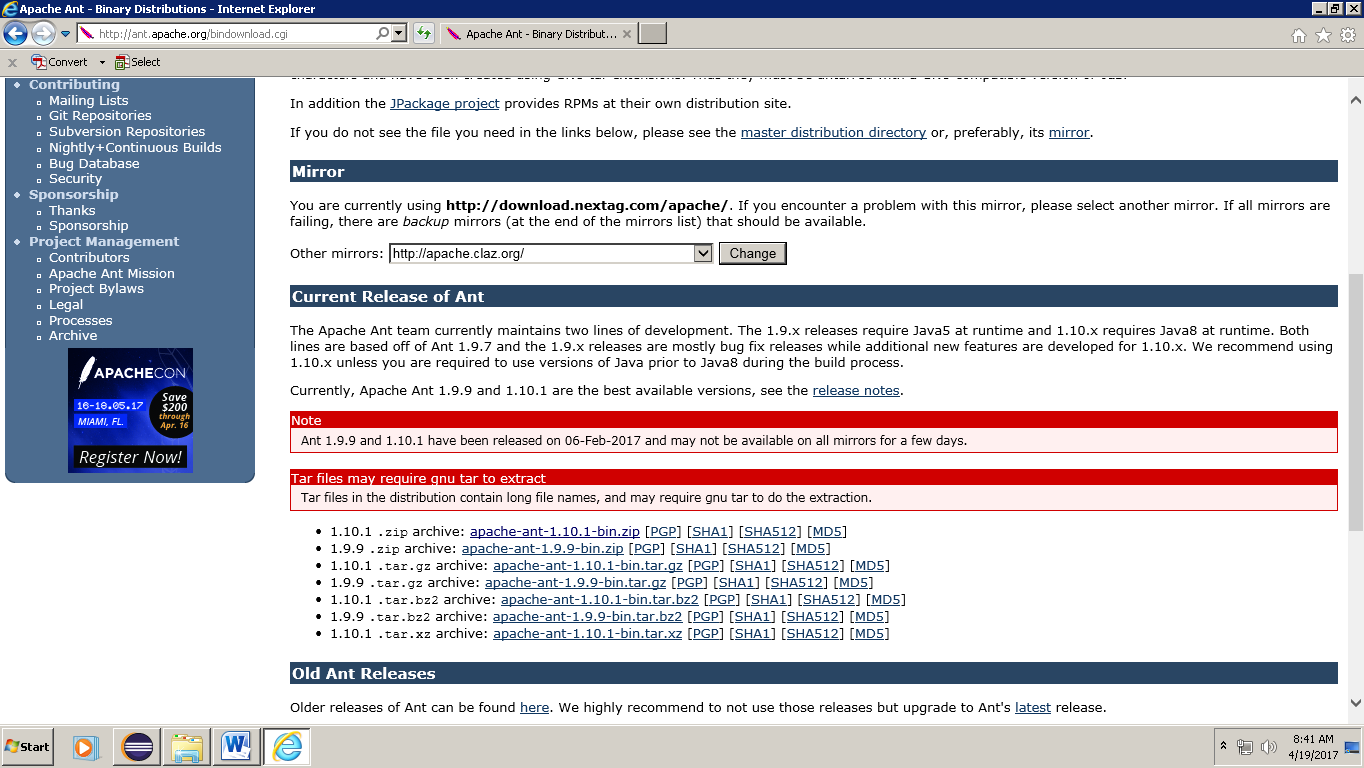
You would see a success message

Step 7: Check the Remote Repository for checked in code

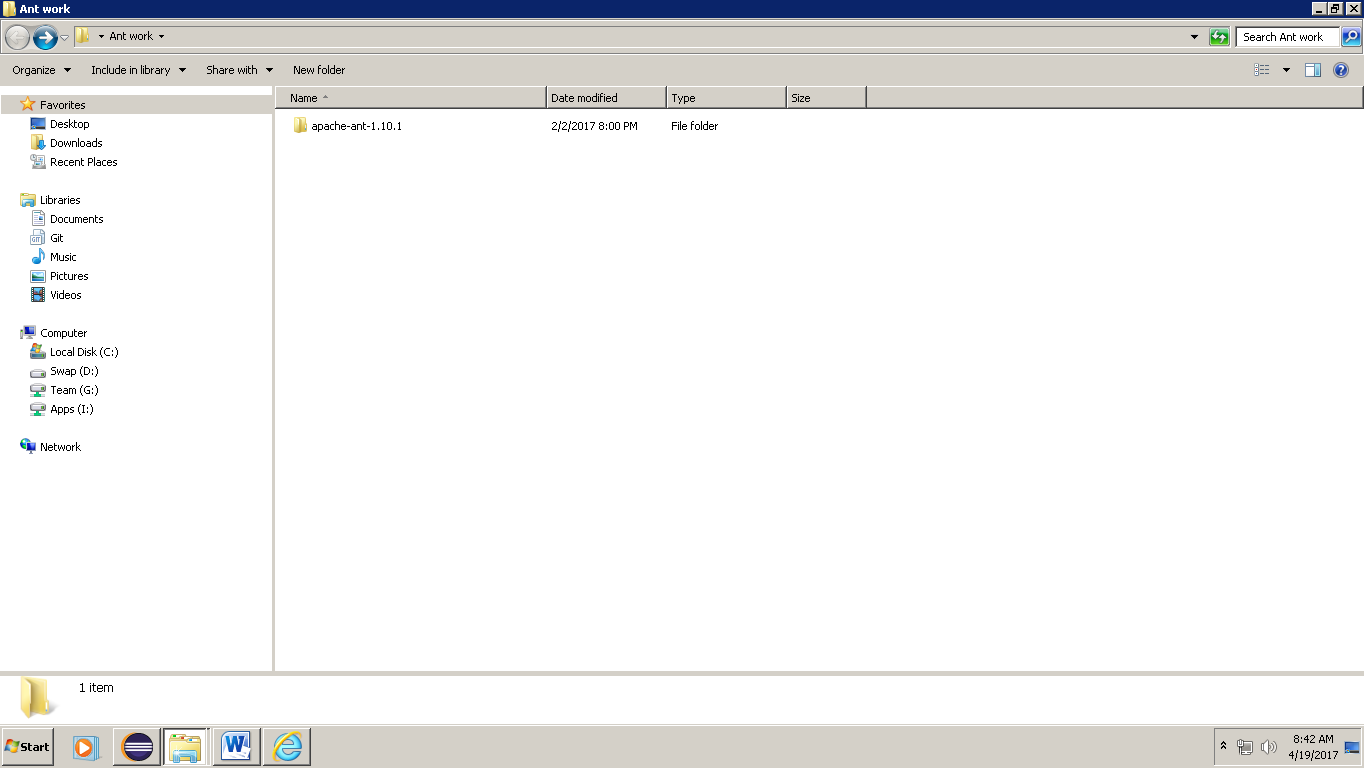
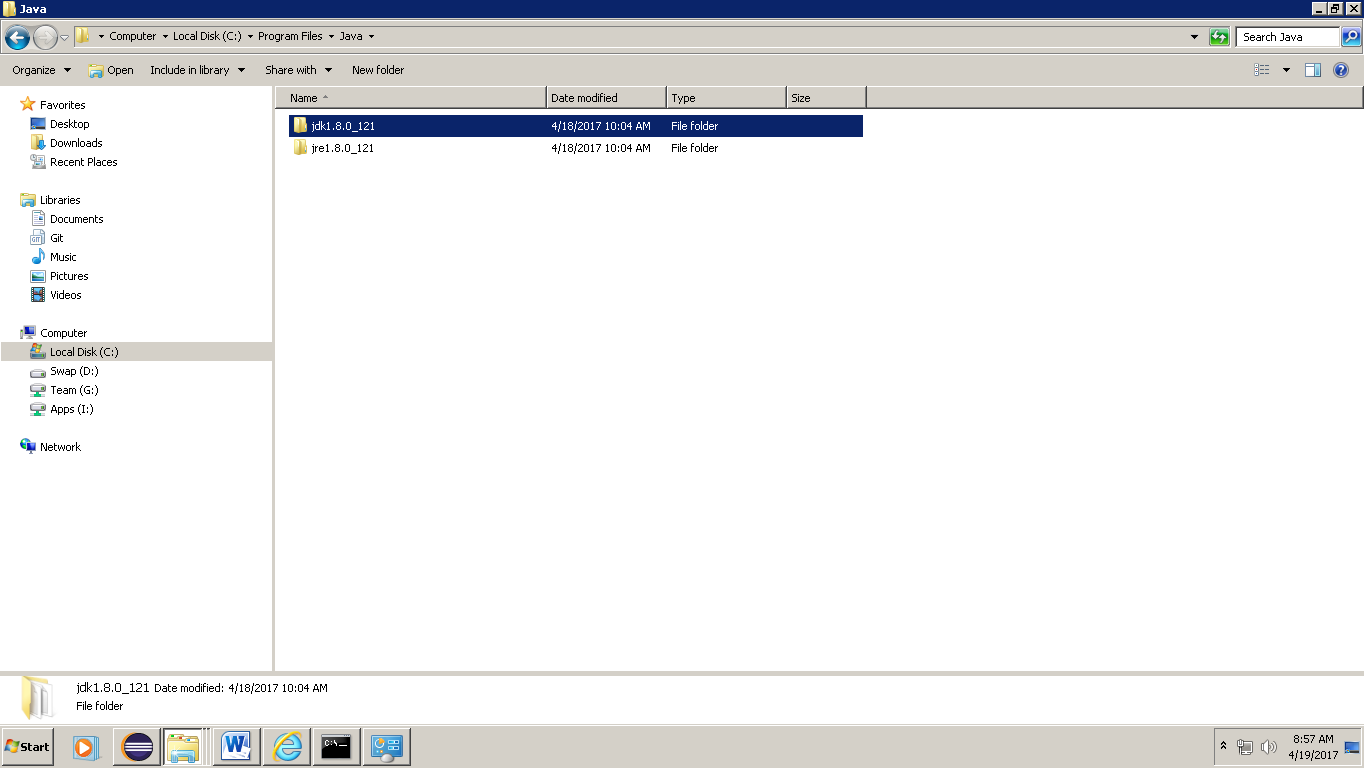


# 6. ANT Installation and Setup:

Step 1: Download the latest version of ant from the link <http://ant.apache.org/bindownload.cgi>



Step 2: Extract it to a folder and also make sure you have installed JDK. Java version 1.7.x or later is recommended.

r

Step 3: Set Environment Variables using your own folder paths of ANT and JAVA as below,

ANT\_HOME should be mapped to the folder path where it is installed.

JAVA\_HOM should be mapped to the folder path where it is installed.

Path should be mapped to ant bin folder and java folder with a ; in between them.

ANT\_OPTS should be mapped to the proxy settings which you would use.

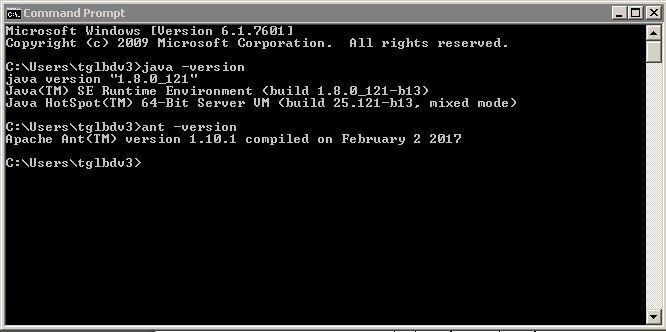
ANT\_HOME = C:\Users\tglbdv3\Desktop\Ant work\apache-ant-1.10.1

JAVA\_HOME= C:\Program Files\Java\jdk1.8.0\_121

Path = C:\Program Files\Java\jdk1.8.0\_121\bin;C:\Users\tglbdv3\Desktop\Ant work\apache-ant-1.10.1\bin

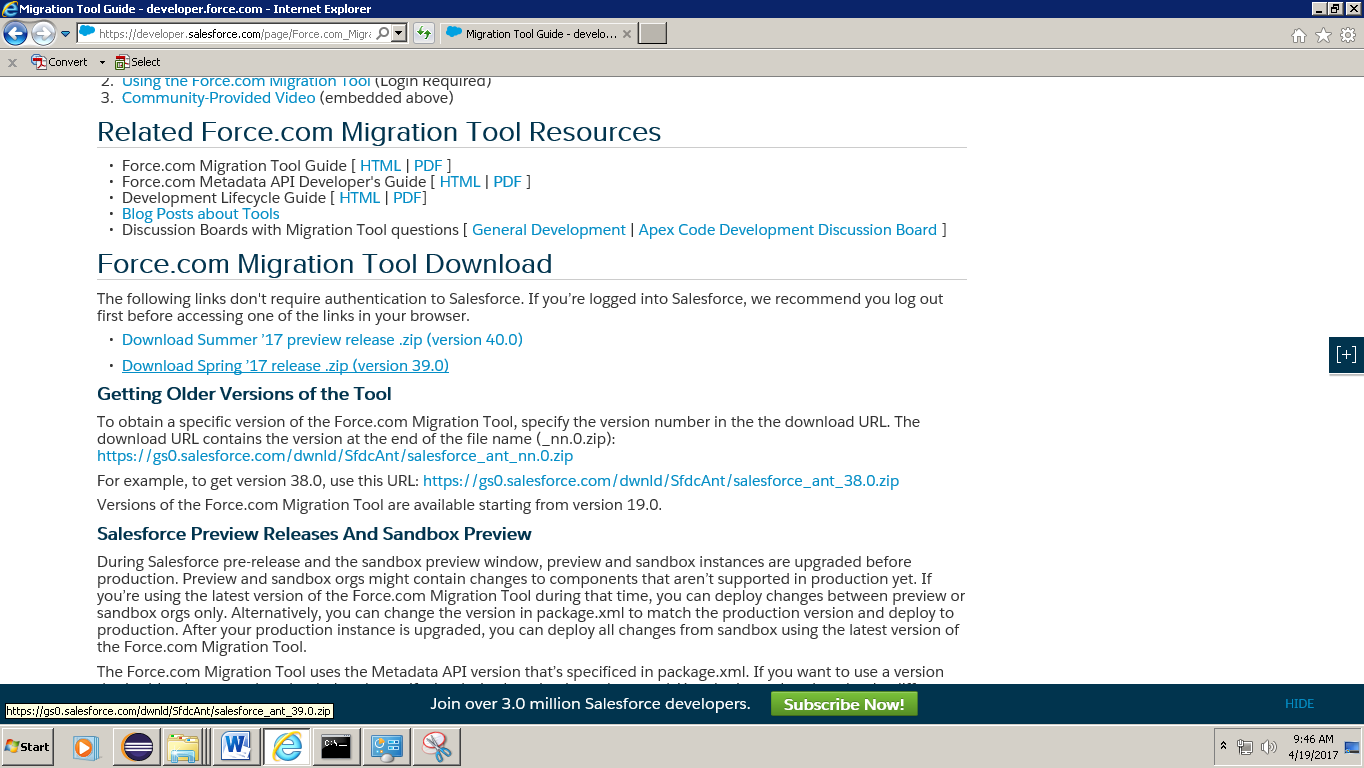
ANT\_OPTS = -Dhttps.proxyHost=proxy.dev.mtb.com -Dhttps.proxyPort=8082

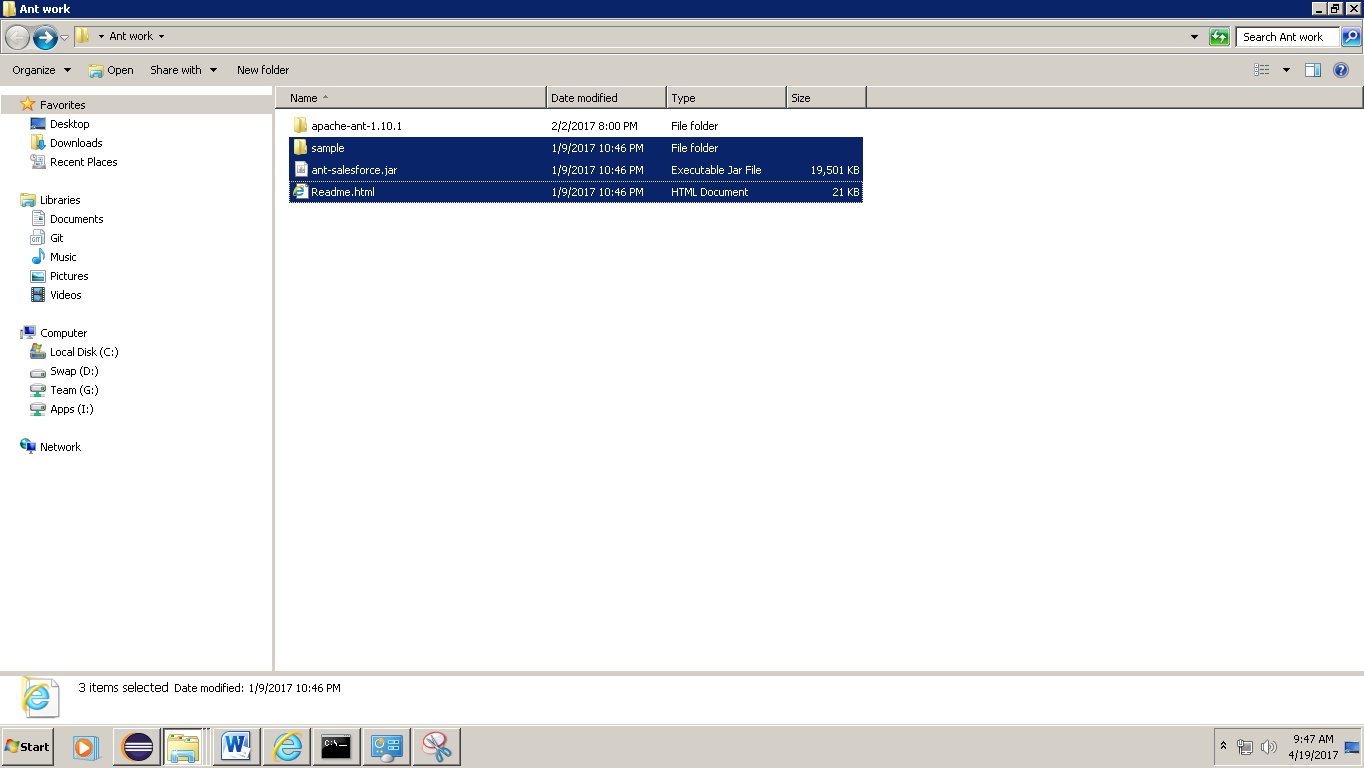
Step 4: Verify Java and ANT from CMD, if you are seeing any error which means your environment variables are not set properly.



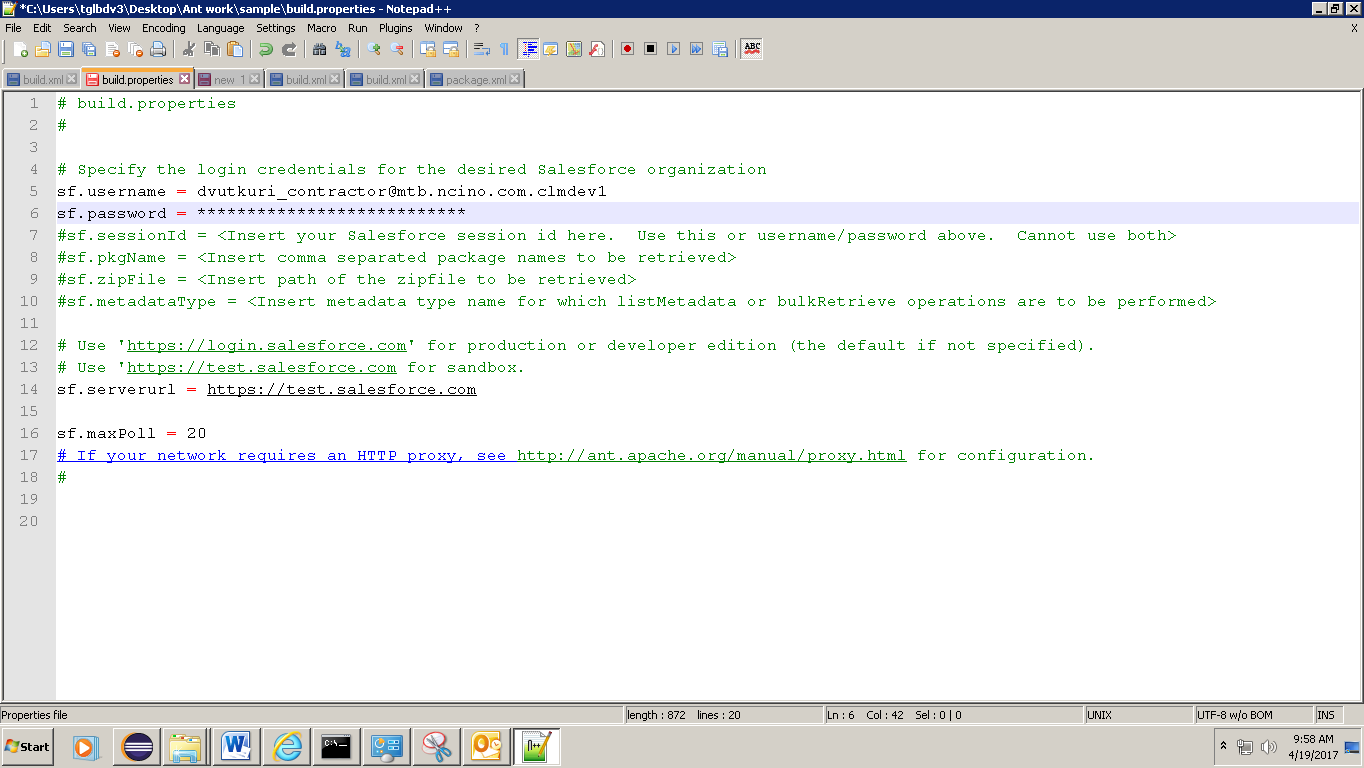
# 7. Force.com Migration Tool:

Step 1: Download Force.com Migration tool from the link <https://developer.salesforce.com/page/Force.com_Migration_Tool>

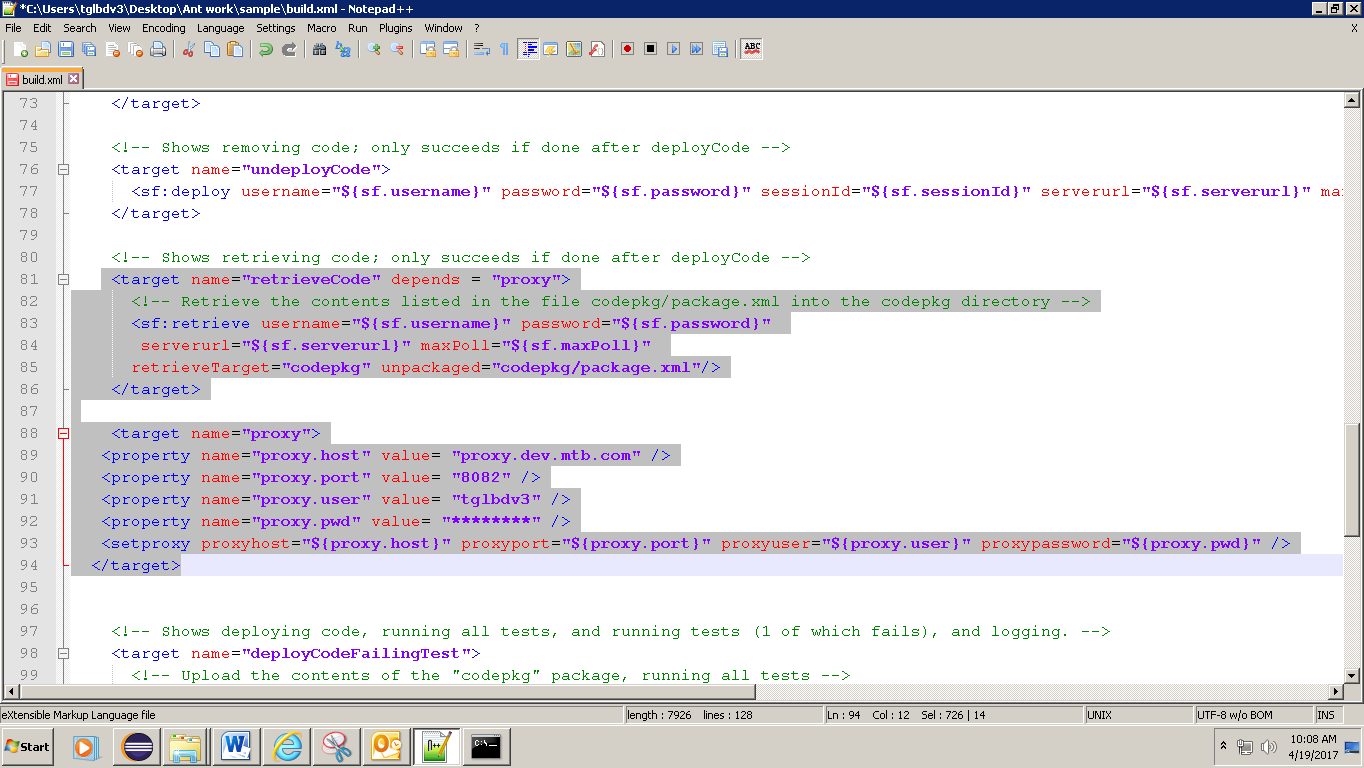


Step 2: Extract it to a folder and then copy ant-salesforce.jar and paste it in apache-ant-1.10.1\lib folder 

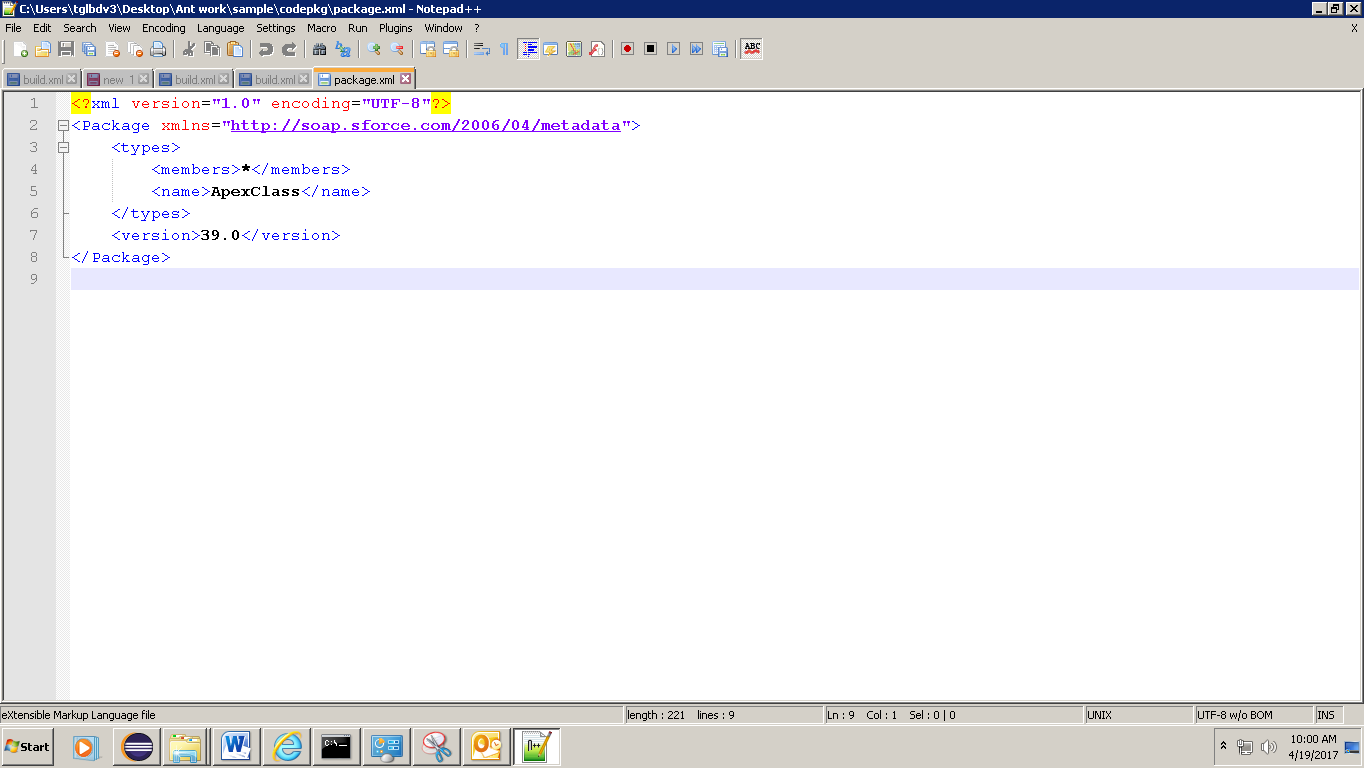
Step 3: Update Build.properties and Build.xml in \sample folder



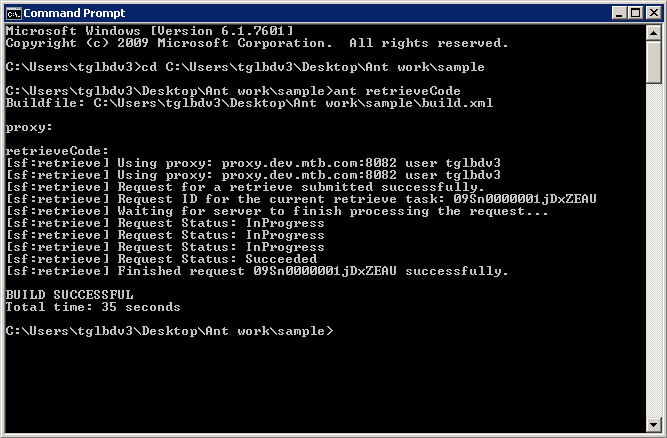
Added proxy settings



Step 4: Update Package.xml from sample\codepkg folder



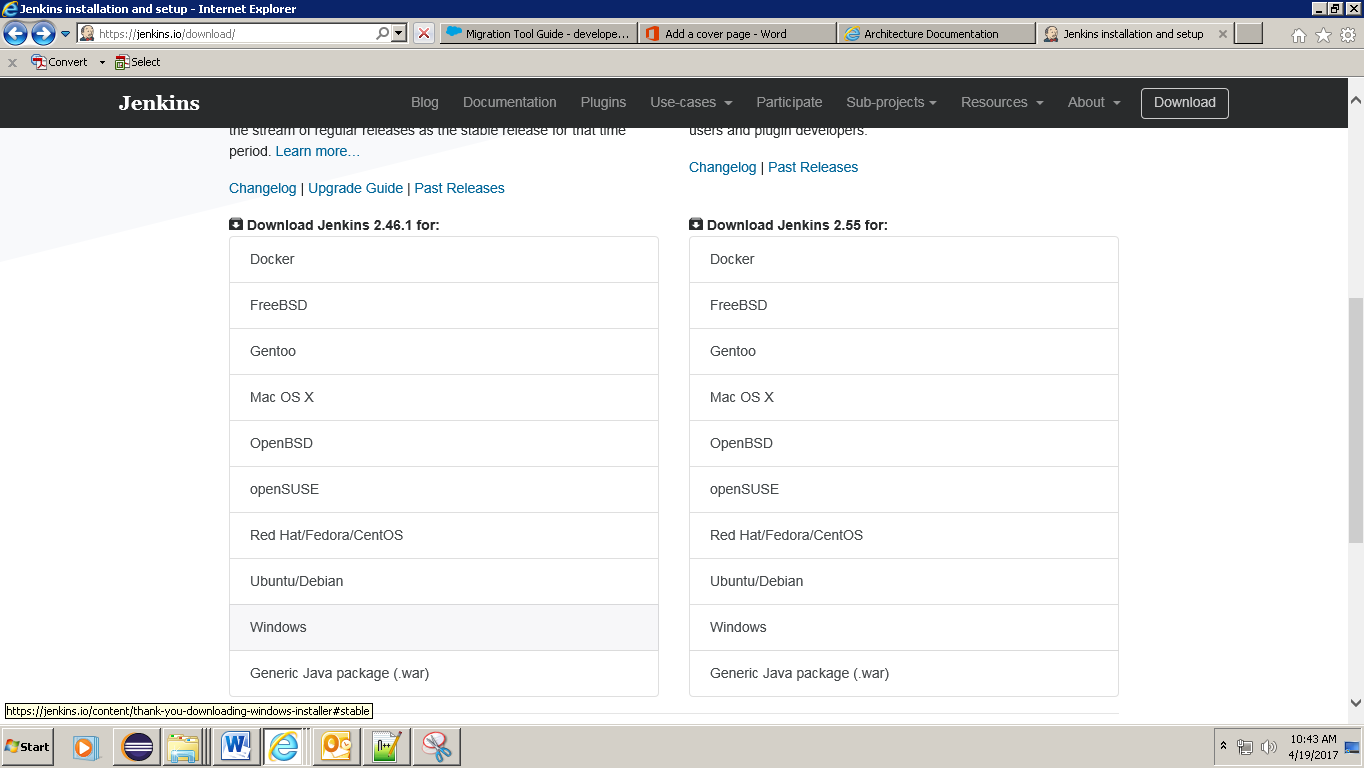
Step 5: Verify if ANT and force.com migration tools are working properly from CMD



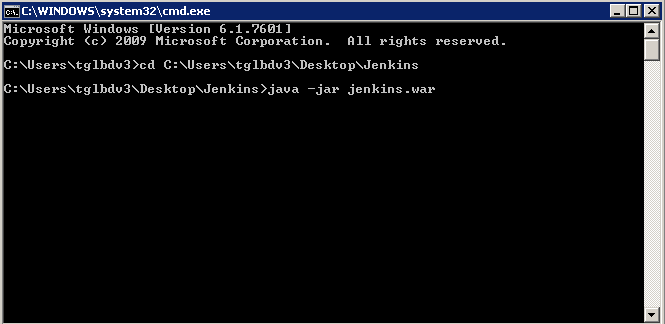
All the classes from sandbox will be loaded into \sample\codepkg\classes

# 8. Jenkins Setup:

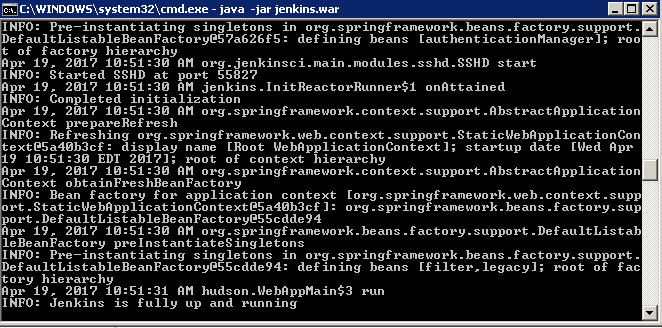
Step1: Download Jenkins for windows from the link <https://jenkins.io/download/> and then extract and Install



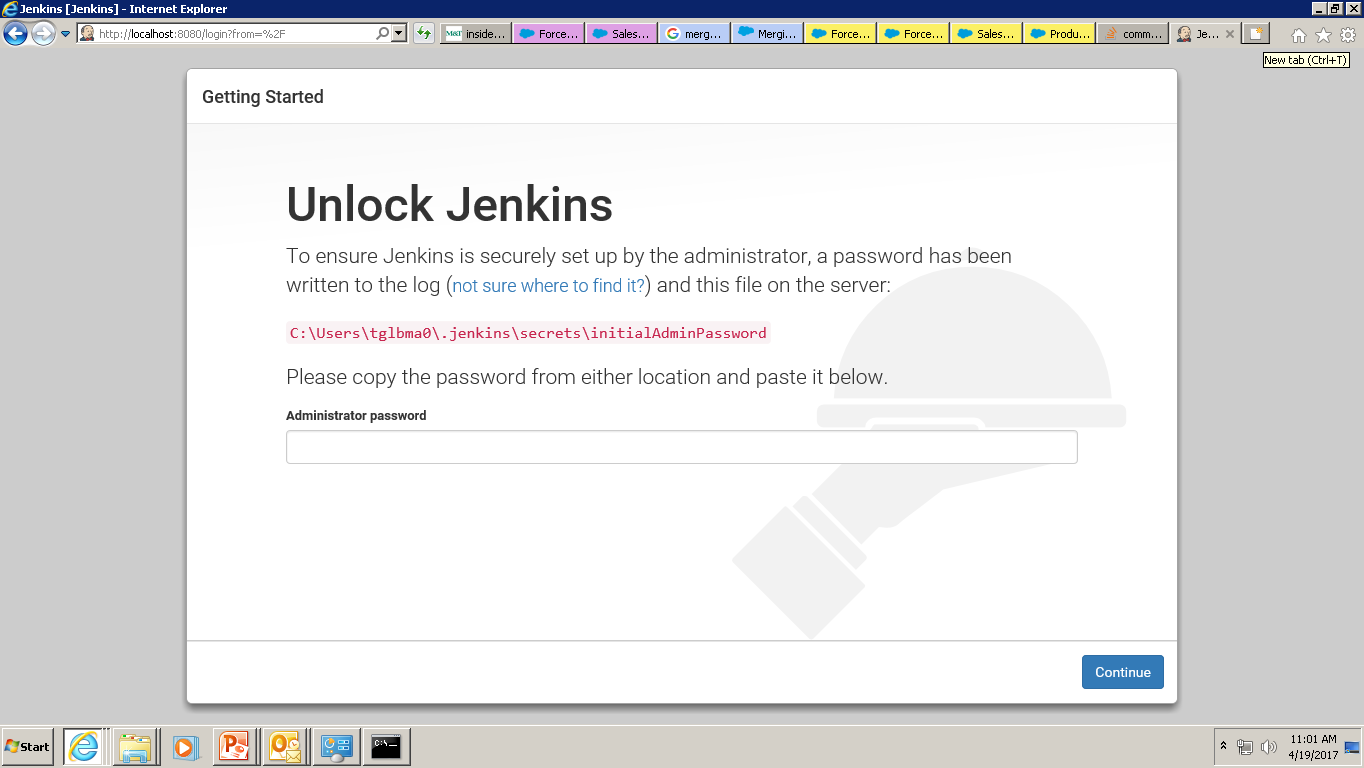
Step 2: Run the Jenkins.war file from CMD



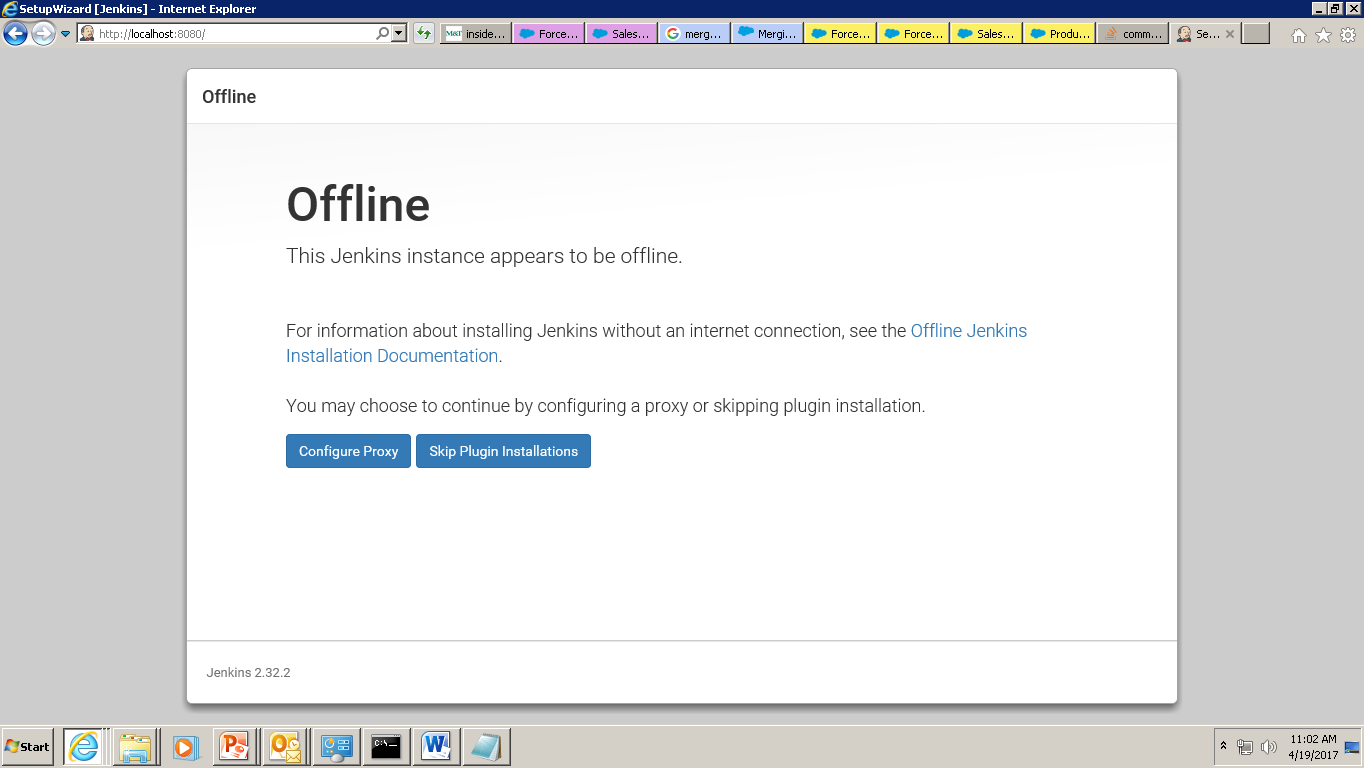
You will see a message – Jenkins is fully up and running



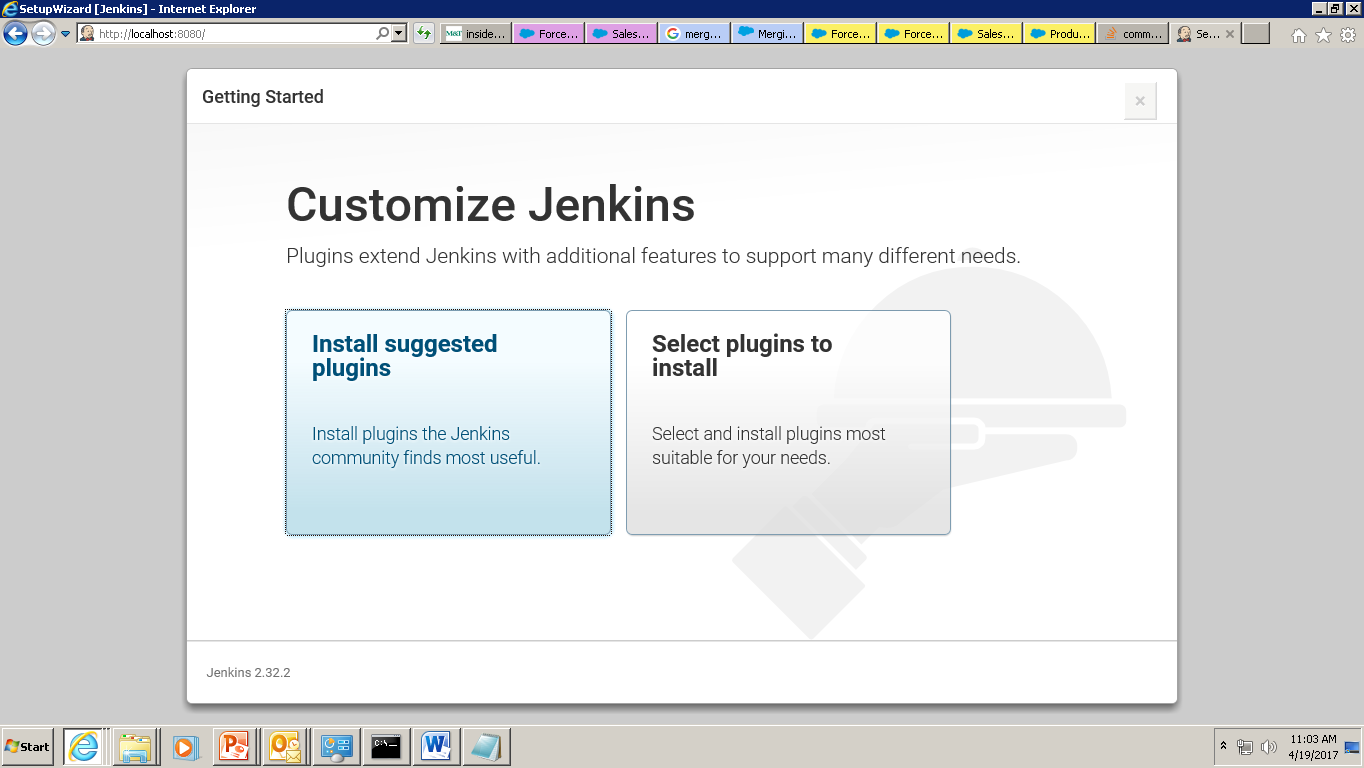
Step3: Open http://localhost:8080 that’s where Jenkins will run. Copy and Paste the password.



Step 4: Configure Proxy settings

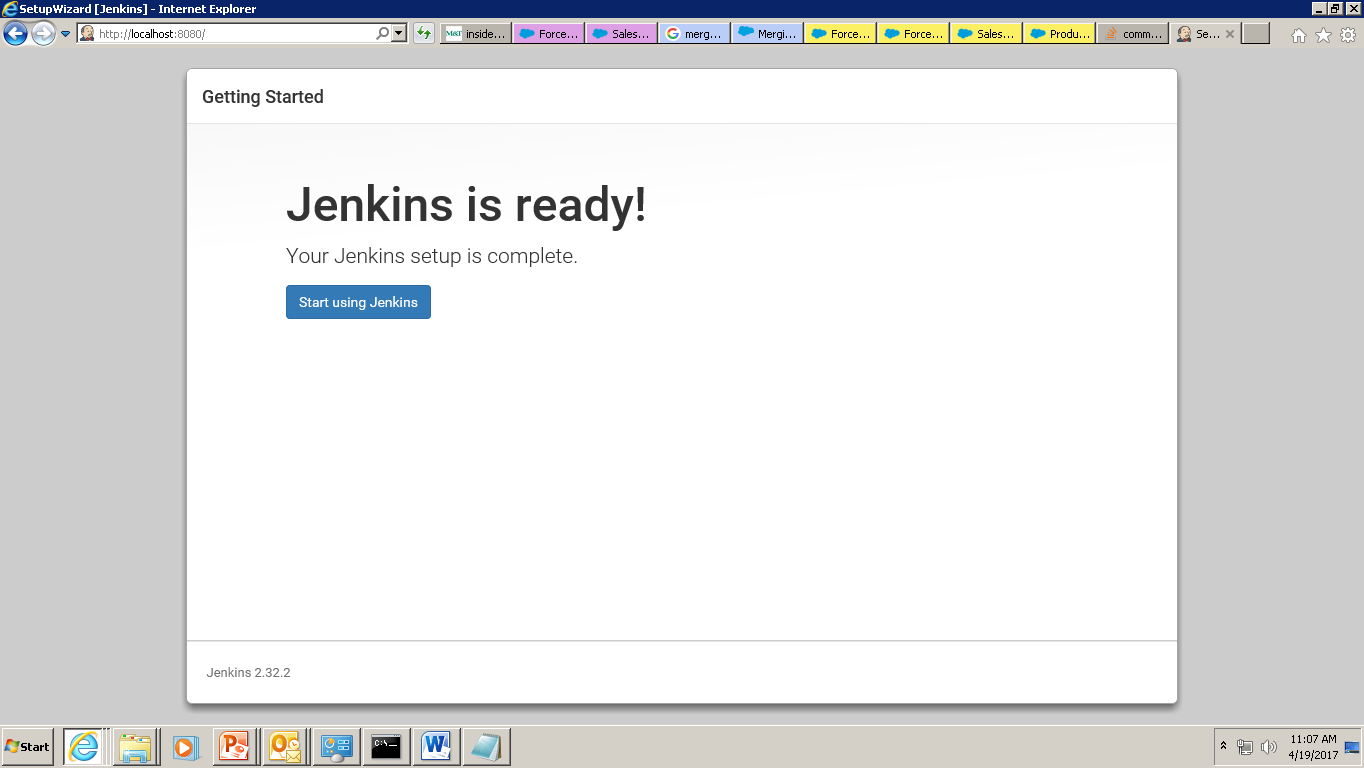


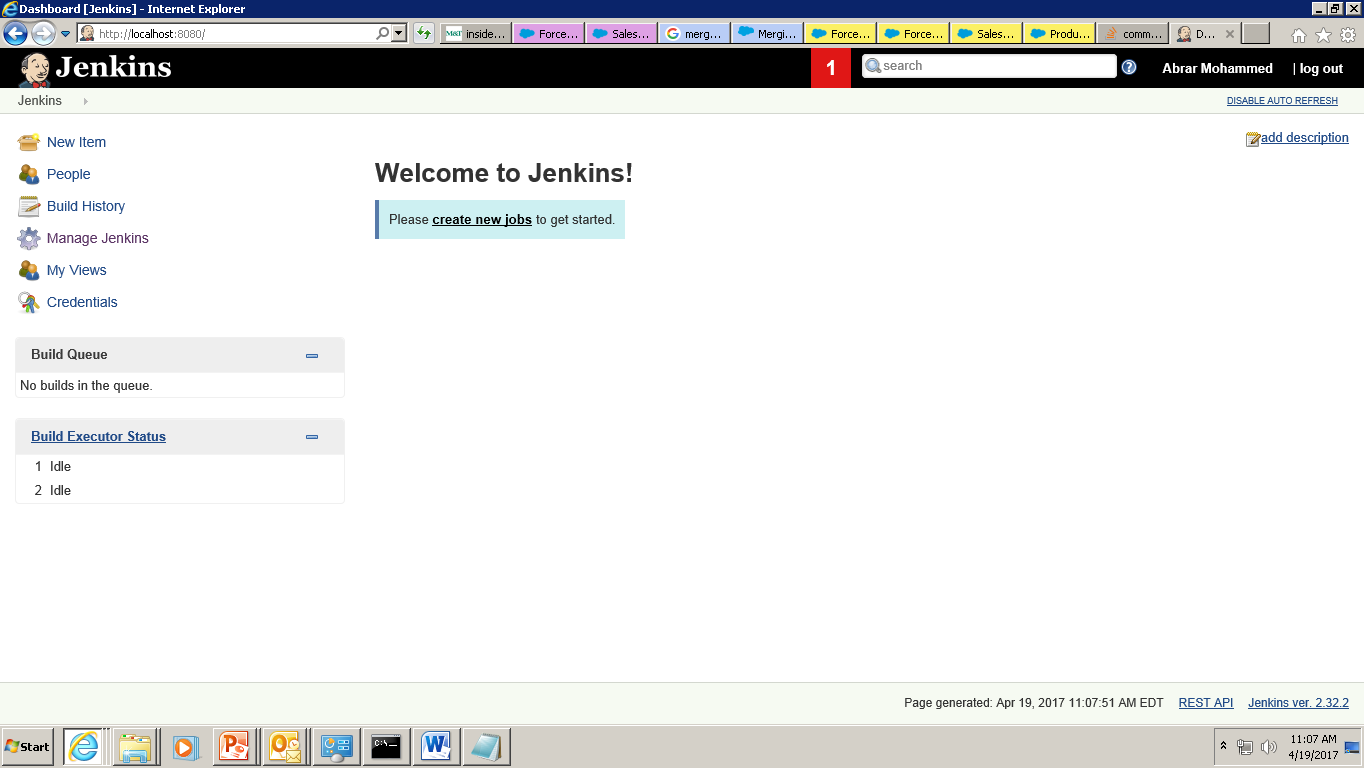
Step 5: Install Suggested Plugins





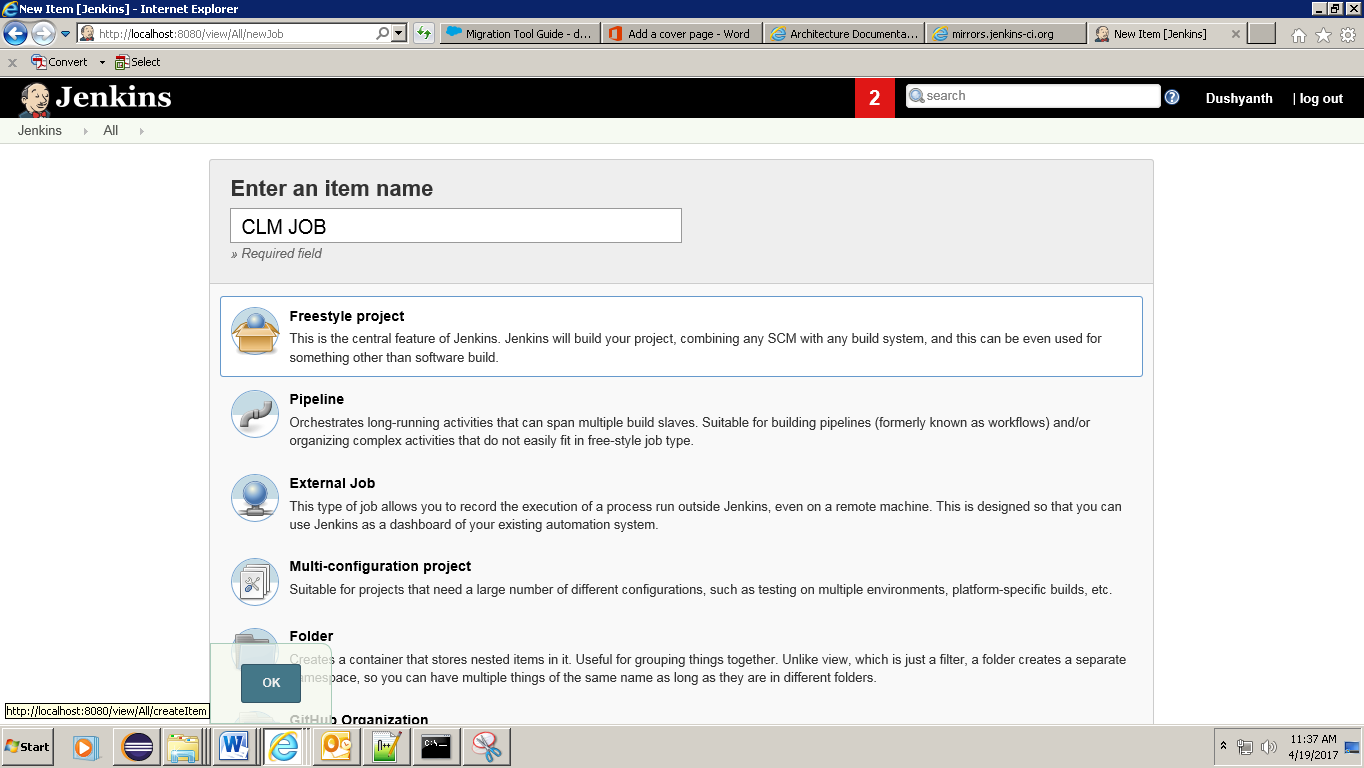
Step 6: Create your first admin user so Jenkins would be ready



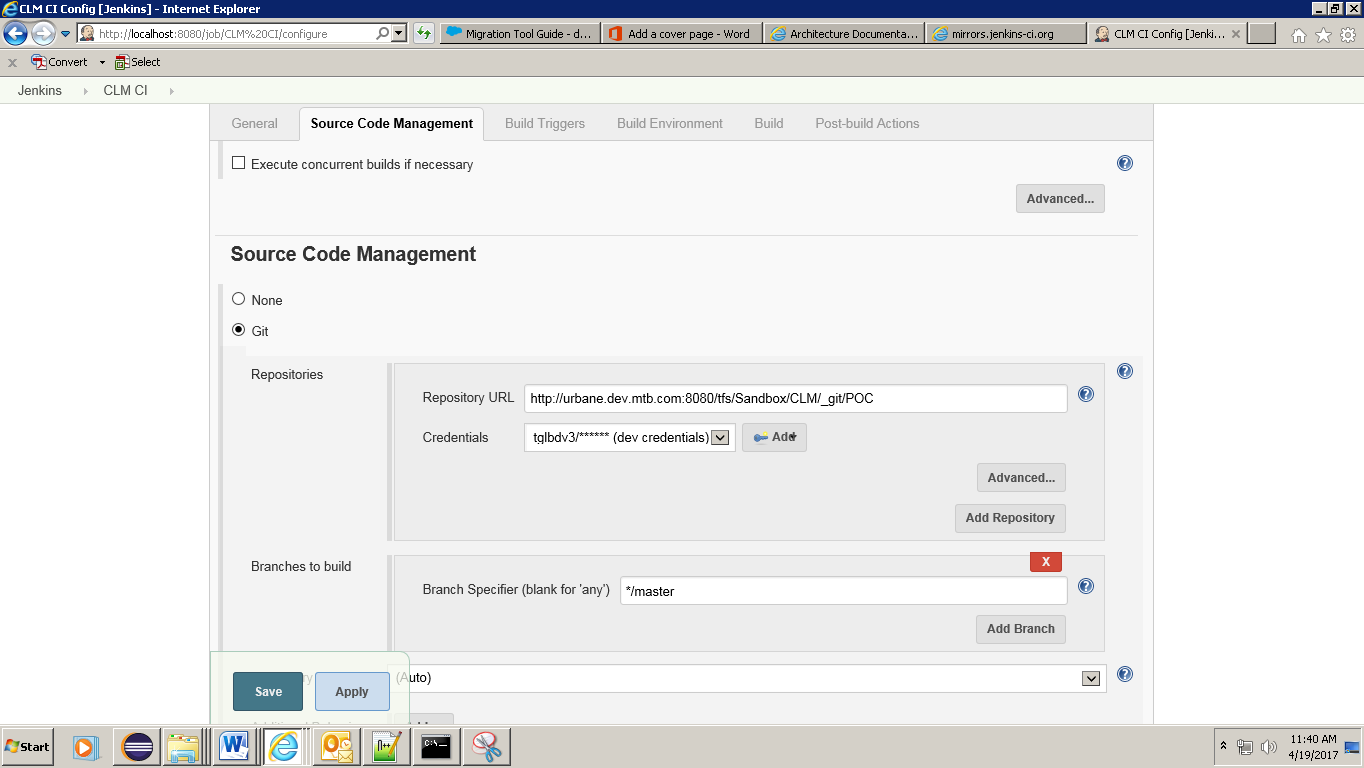


# 9. Job Creation:

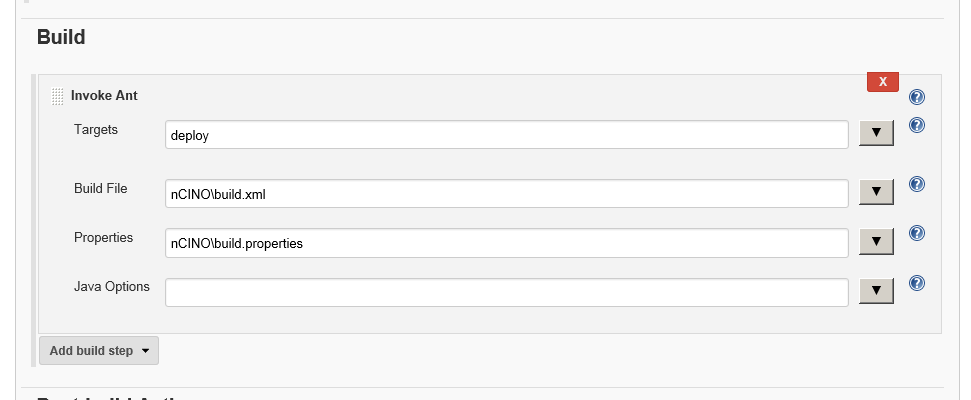
Step 1: Create a new job to get started. Select a free style project and click OK.



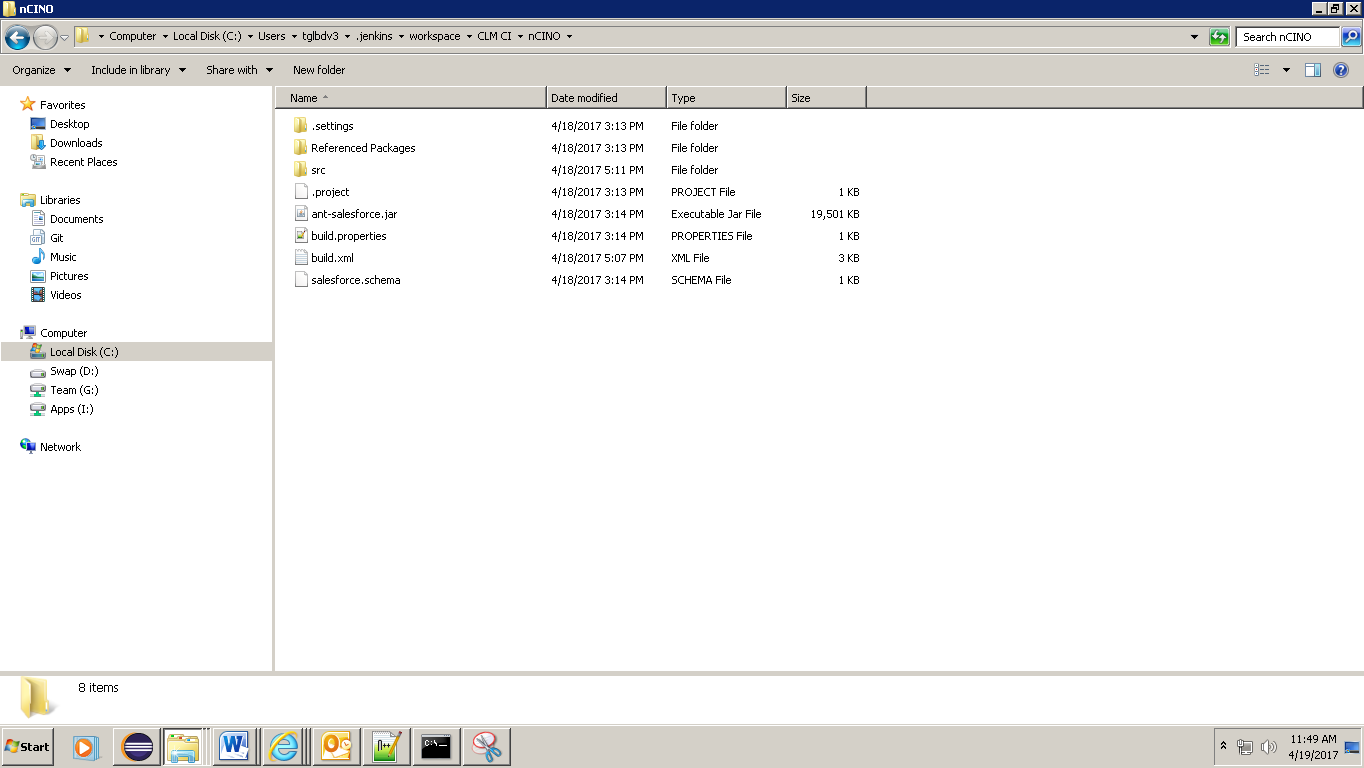
Step 2: Source Code Management – Select GIT and paste the project’s GIT repository URI and add credentials to Jenkins, which should be same as your repository credentials.



Step 3: Build – select Invoke Ant. In Targets space pass the actions which you would like to perform from Build.xml. You can pass multiple actions with space in between them. Click advance to map the build.xml and build.properties files

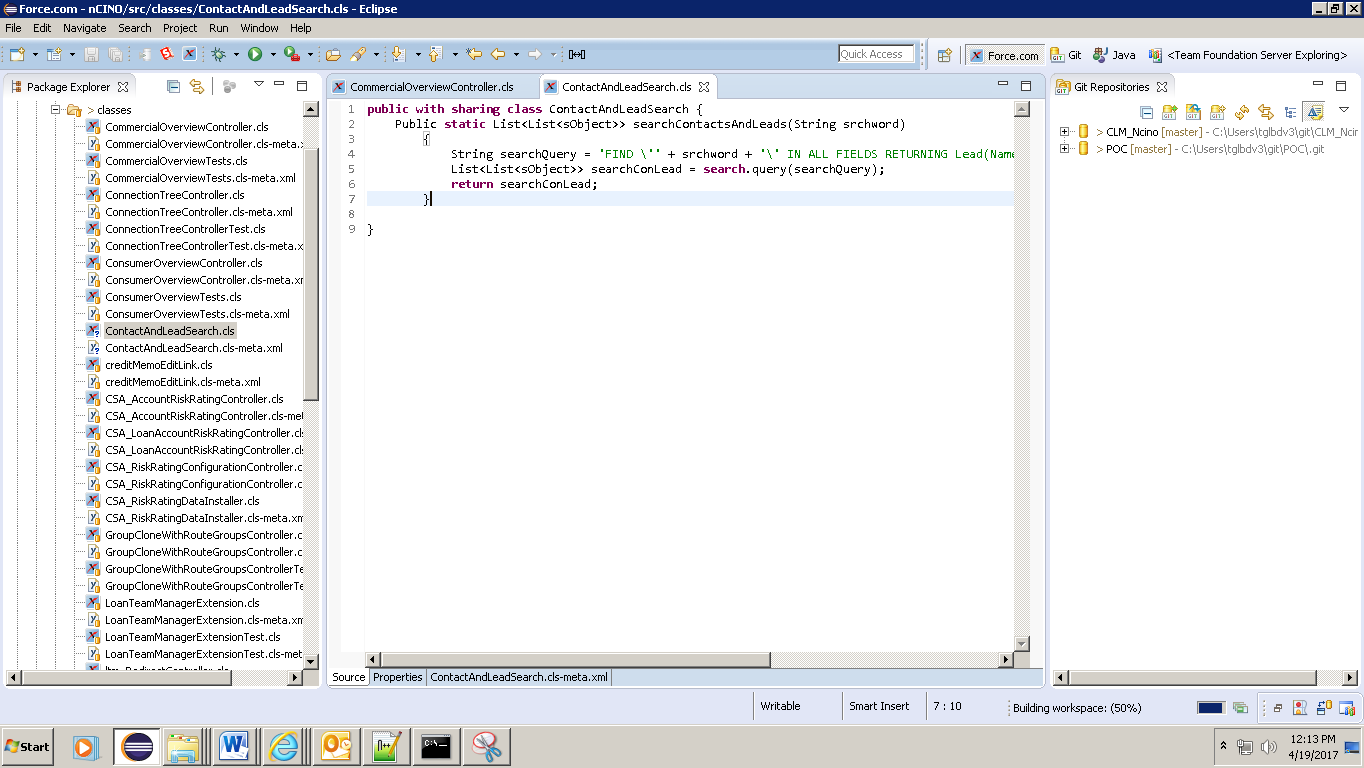


These targets are taken from .jenkins workspace. It downloads the project from github into workspace\job folder. Ex: C:\Users\tglbdv3\.jenkins\workspace\CLM CI\nCINO

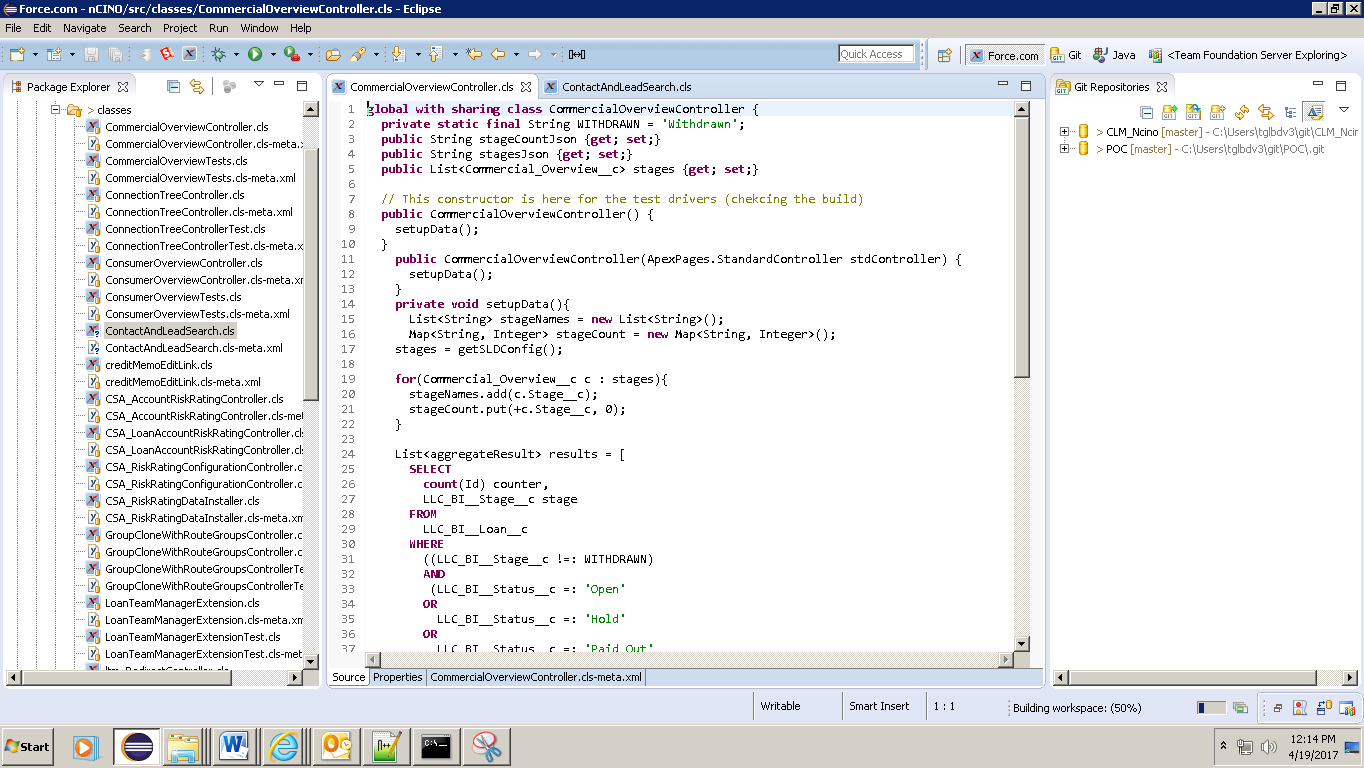


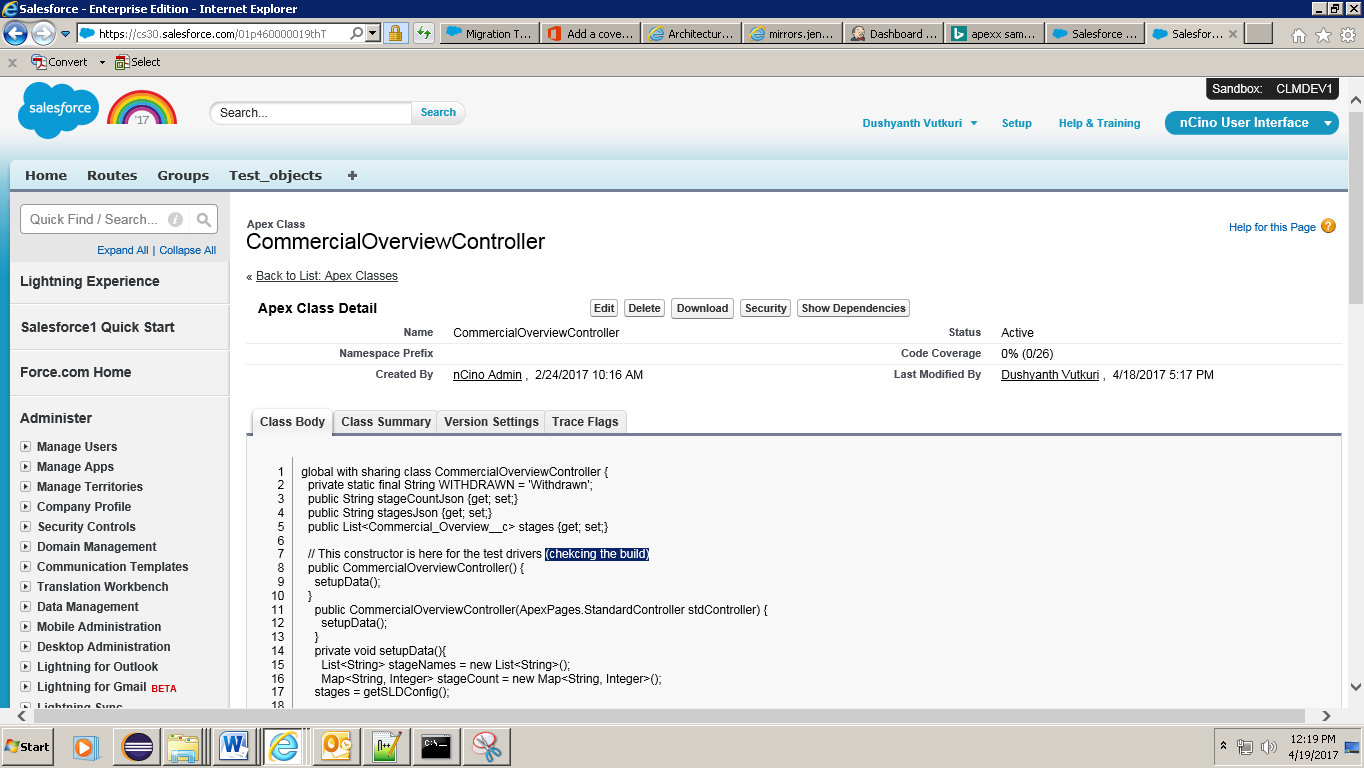
# 10. Test:

Step1: Add a class in Eclipse project and also make a change in already existing class



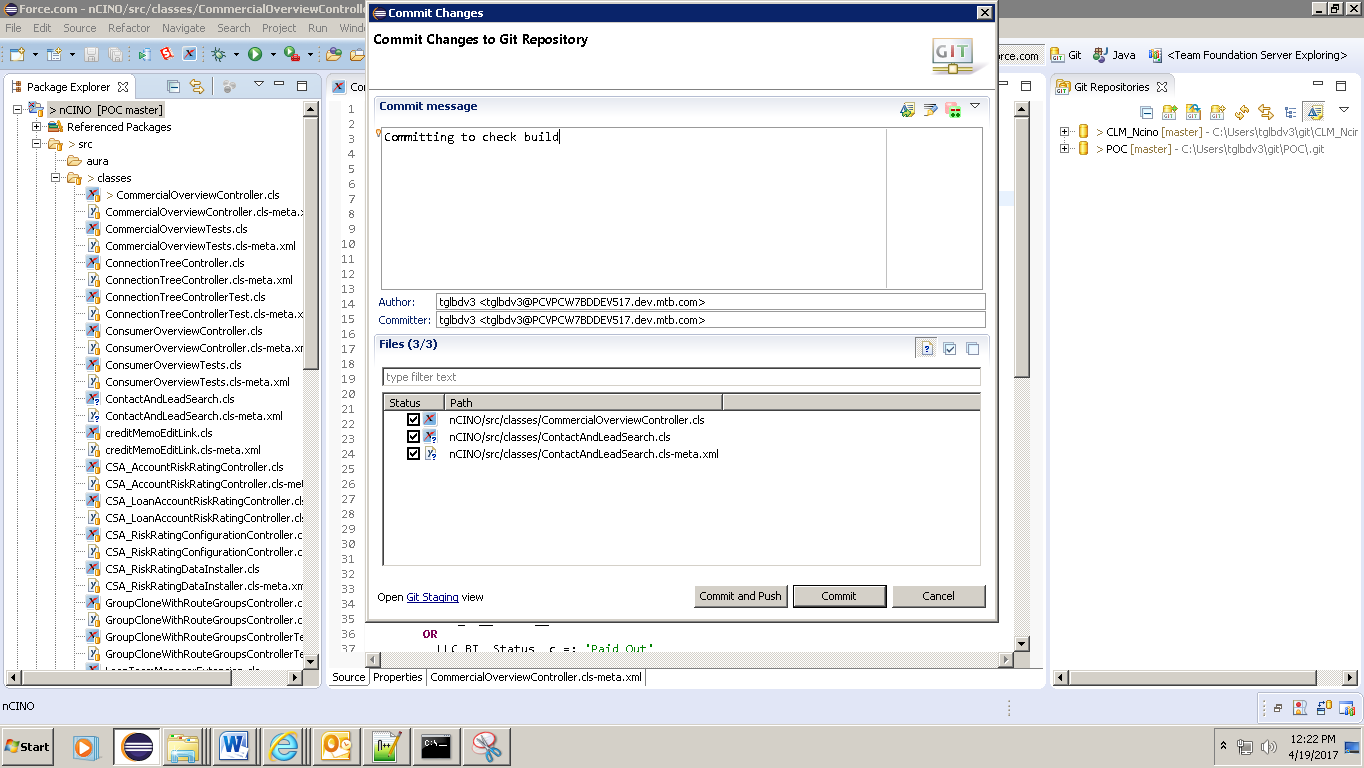
Removing **(checking the build)** from existing class.



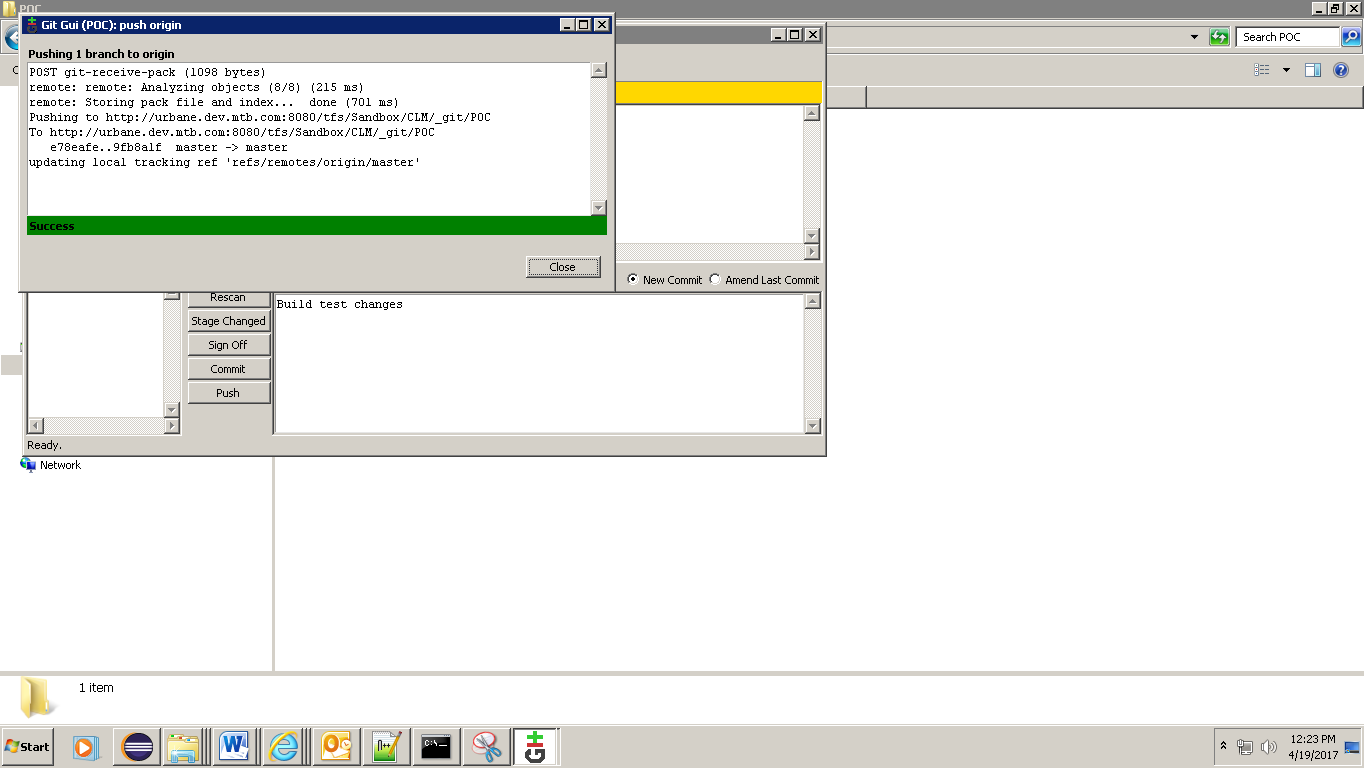


Step 2: Commit to local repository and Push to Remote Repository as mentioned above.

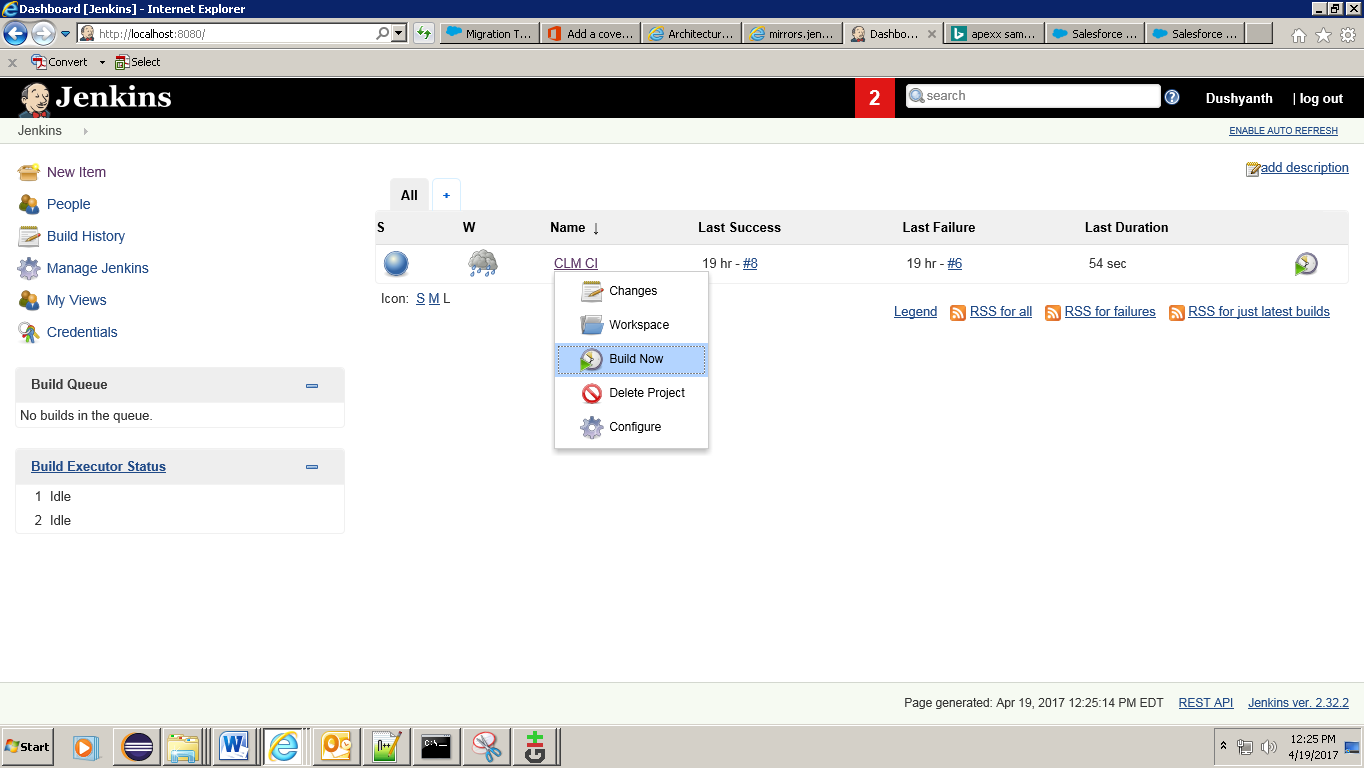
Commit



Push



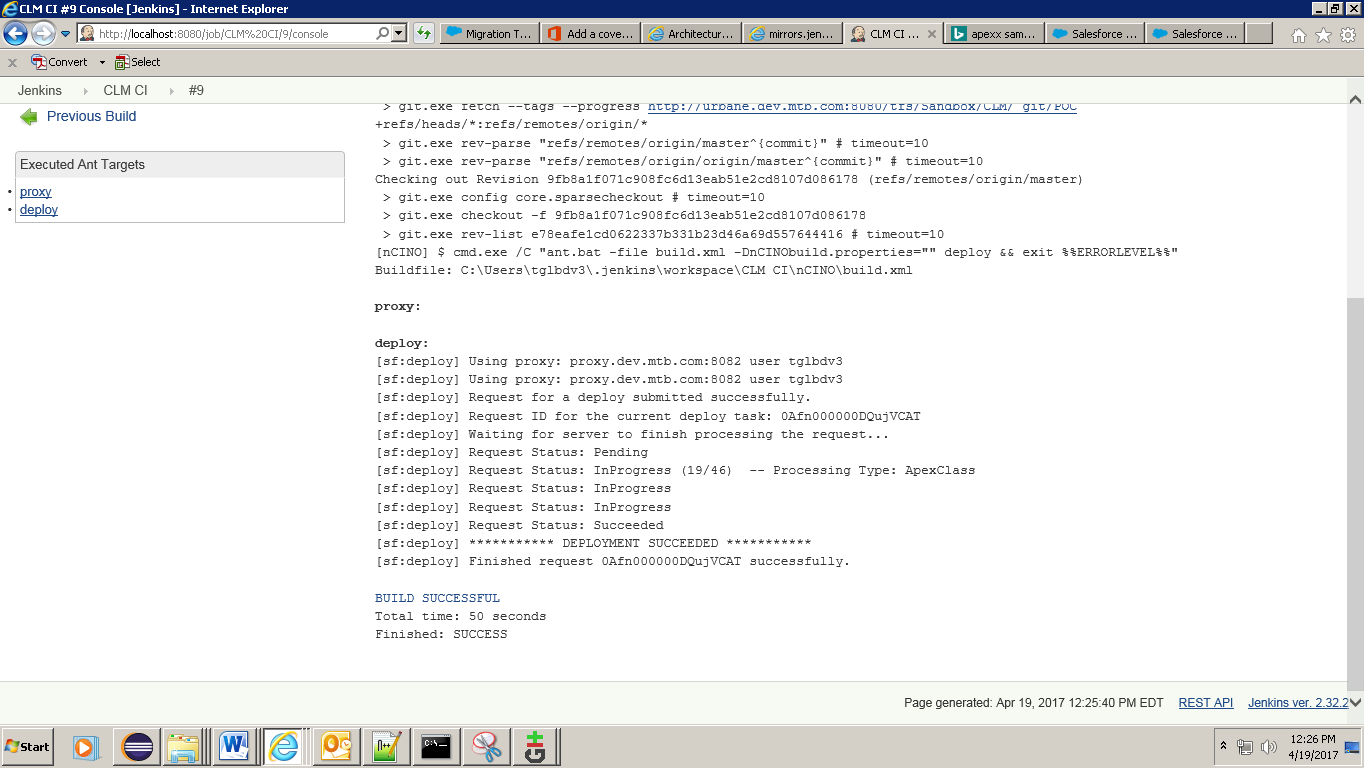
Step 3: Run the Jenkins Build



Build In progress

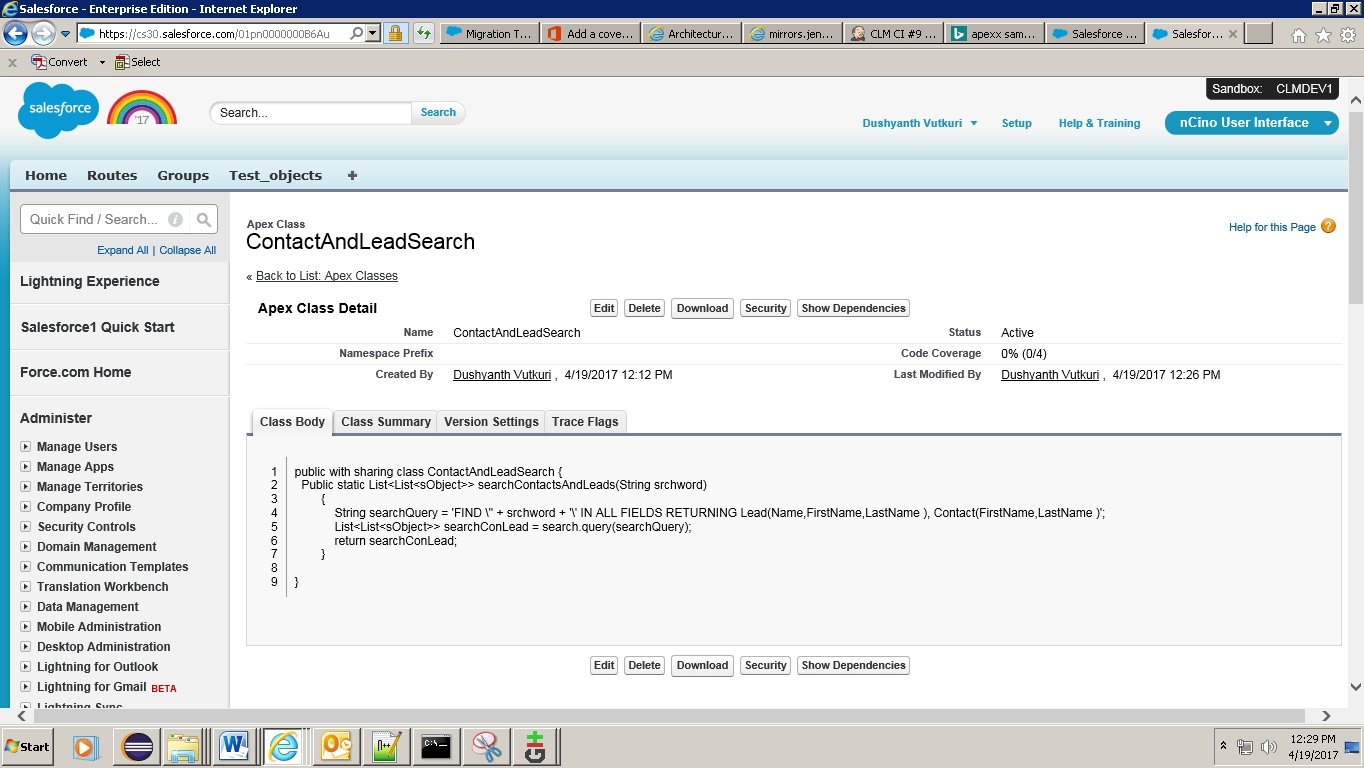


Build Successful

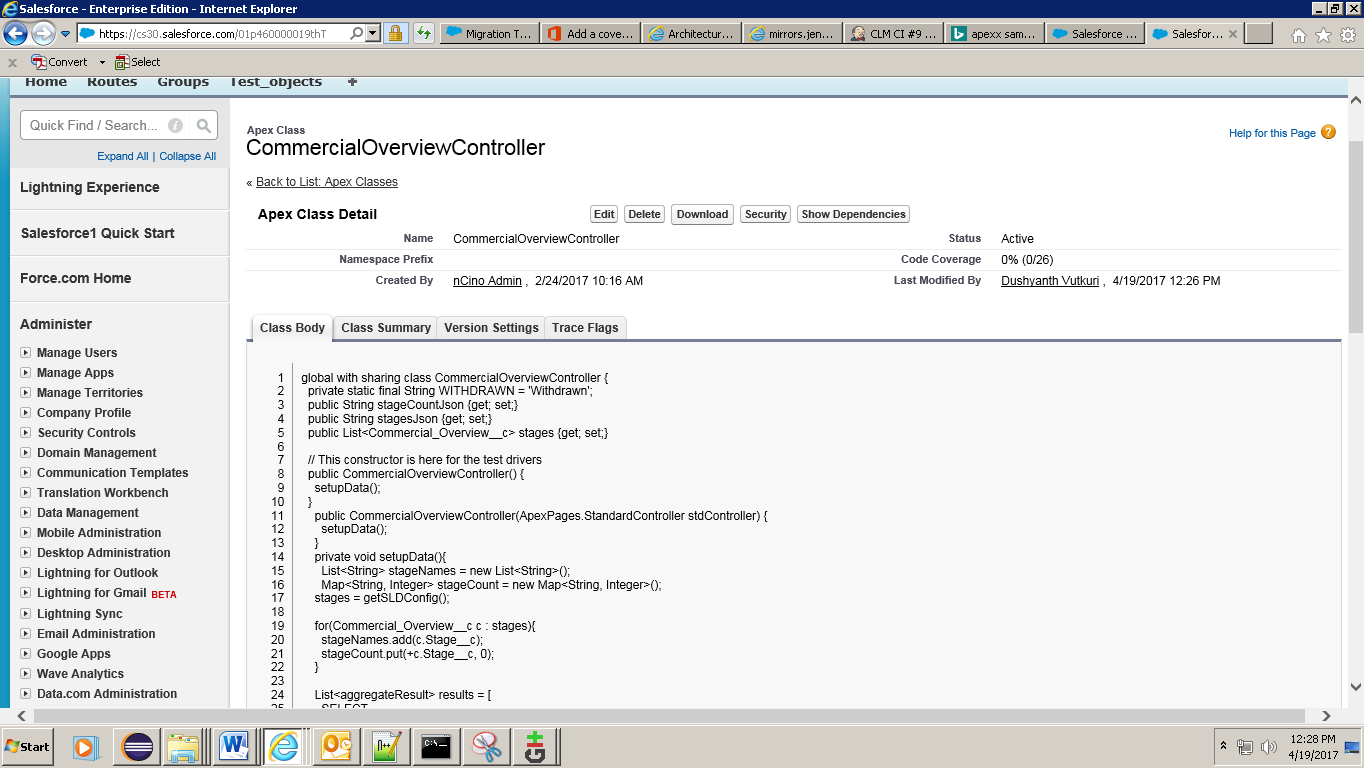


Step 4: Log into sandbox and validate the changes

New Class



Existing class



Using Jenkins it’s possible to deploy changes from one sandbox to another sandbox or from SRC to a sandbox. It’s also possible to create triggers for Build.