GitHub with salesforce Proof of Concept

(CP2369)

Information Technology Services Projects

Version 1.0

**TABLE OF CONTENTS**

[1.0 Objective 3](#_Toc473029173)

[2.0 Use Cases 3](#_Toc473029174)

[2.1 GitHub Introduction 3](#_Toc473029175)

[2.2 Install Eclipse 3](#_Toc473029176)

[2.3 Set Up proxy details for eclipse 3](#_Toc473029177)

[2.4 Install force.com IDE on eclipse 10](#_Toc473029178)

[2.5 Install eGIT 10](#_Toc473029179)

[2.6 Set Up Organization on GitHUB 11](#_Toc473029180)

[2.7 Set Up repository in GitHUB 11](#_Toc473029181)

[2.8 Creating a branch in GitHub 23](#_Toc473029182)

[2.9 Fetch branch from GitHUB 23](#_Toc473029183)

[2.10 Push Changes from Eclipse to GitHub Branch 28](#_Toc473029184)

[2.11 Roles and responsibilities 28](#_Toc473029185)

[Appendix A - Document Information 28](#_Toc473029186)

[Business Requirements Document Revision History 28](#_Toc473029187)

[Template Owner 29](#_Toc473029188)

[Information Classification 29](#_Toc473029189)

[Appendix B - Acceptance & Sign-off 29](#_Toc473029190)

[Appendix C - Attachments 31](#_Toc473029191)

# 

# Objective

The objective of this document is to provide detailed steps for setting up GitHub and integrating eclipse IDE with GitHub so that developers/Admin can enforce Version Control for salesforce projects within TFS

# Use Cases

## GitHub Introduction

Git allows groups of people to work on the same code at the same time, and without stepping on each other's toes. It's a distributed version control system.

## Install Eclipse

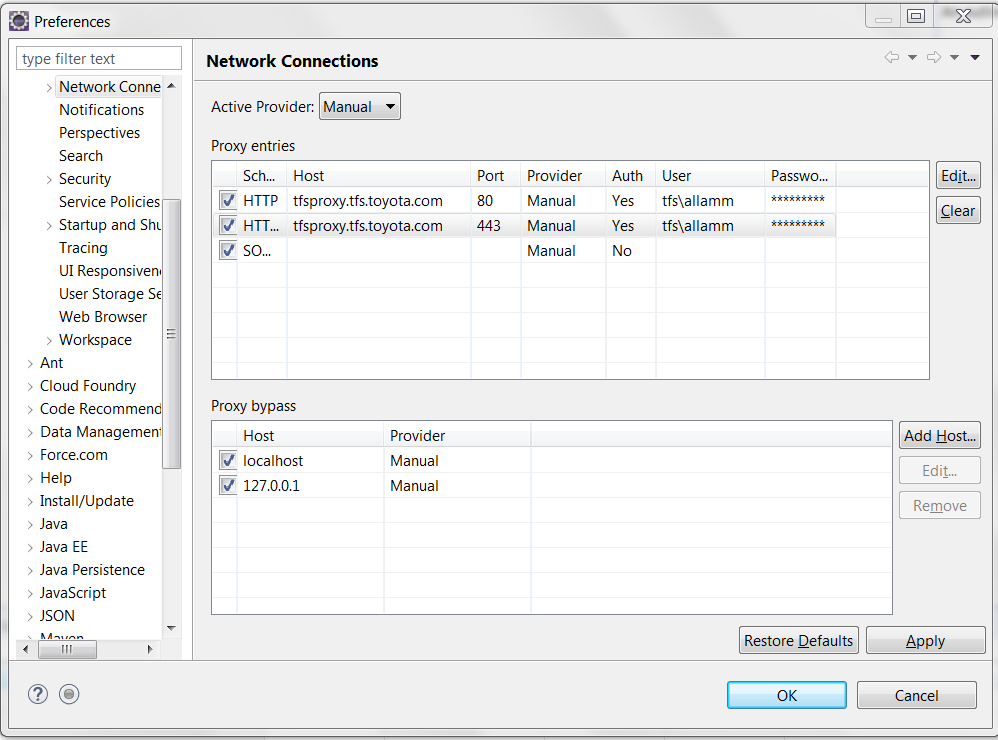
Note: Recommended Version: Neon.2 Release (4.6.2)

Download the installation file from <https://eclipse.org/downloads/>

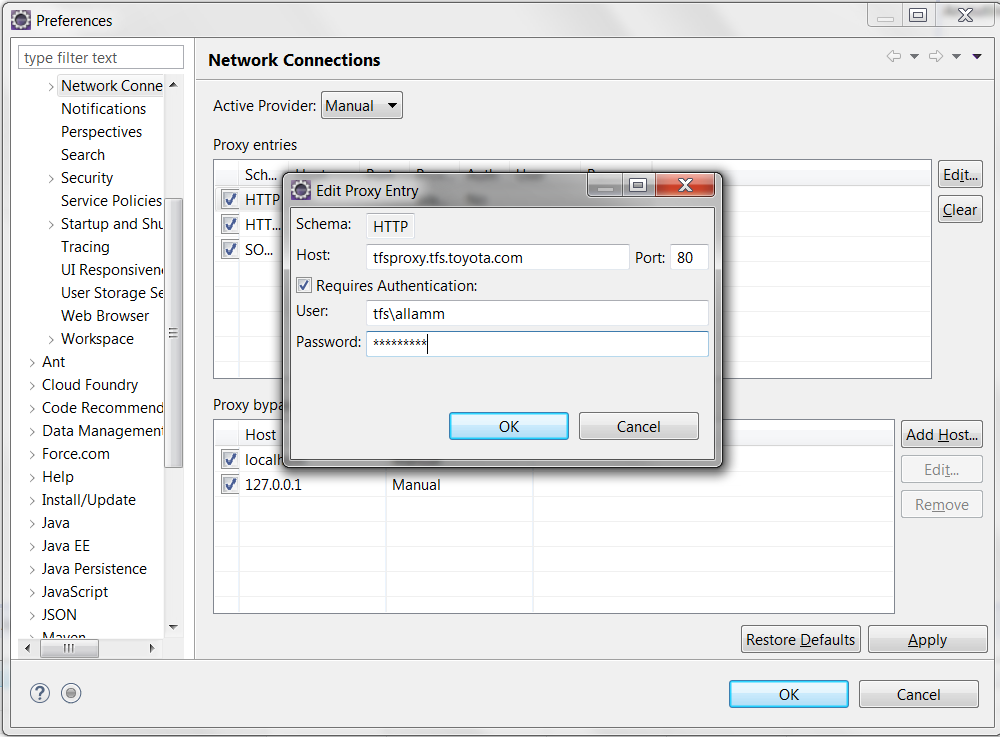
and install the software on to your computer. Eclipse Neon is the recommended version as it comes with eGIT installed by default.

## Set Up proxy details for eclipse

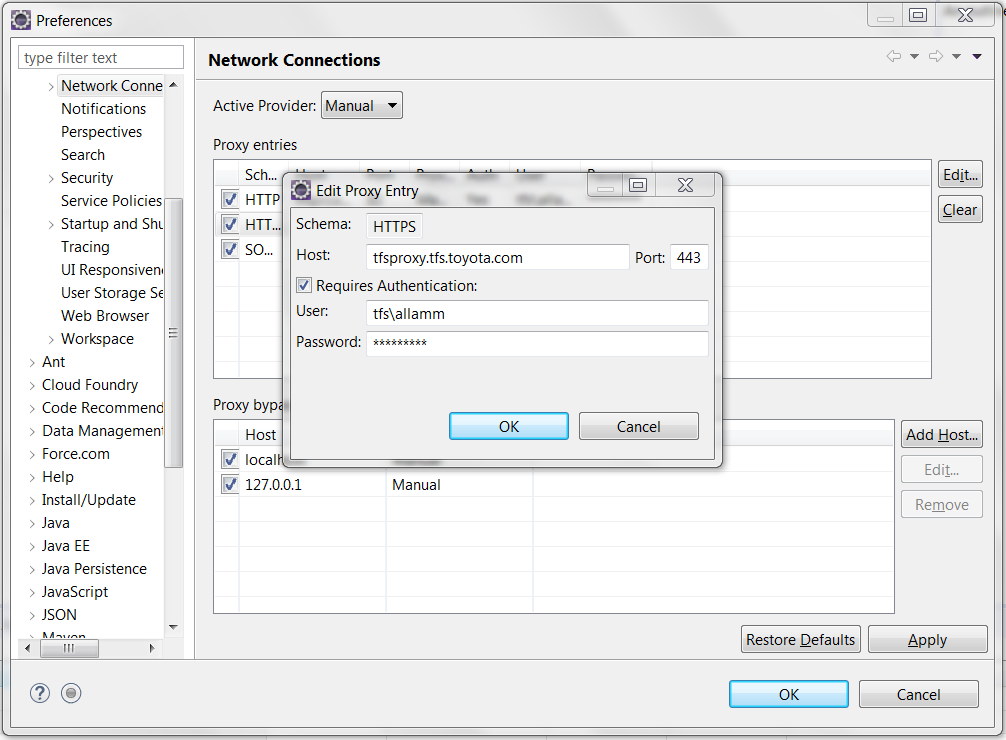
* Navigate to Windows -> Preferences ->General -> Network Connections



* Double click on the row which point to HTTP

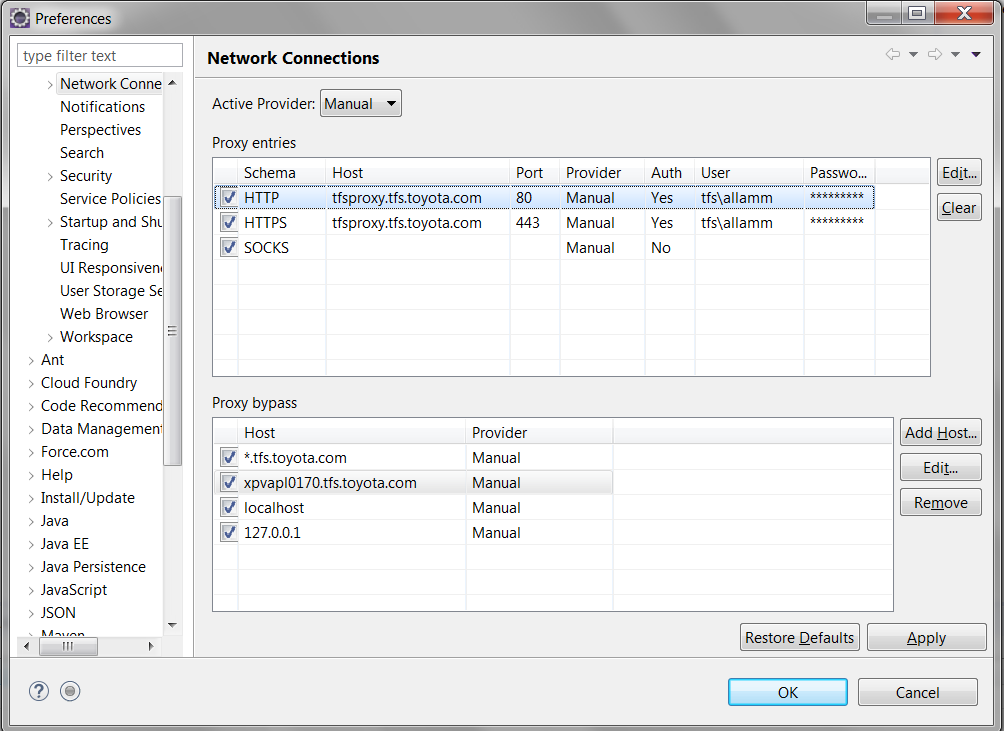


* Change the Host details to tfsproxy.tfs.toyota.com
* Port to 80
* Provide your TFS Lan ID and password
* Click OK to save the details
* Double click on the row which points to HTTPS and repeat the aforementioned steps



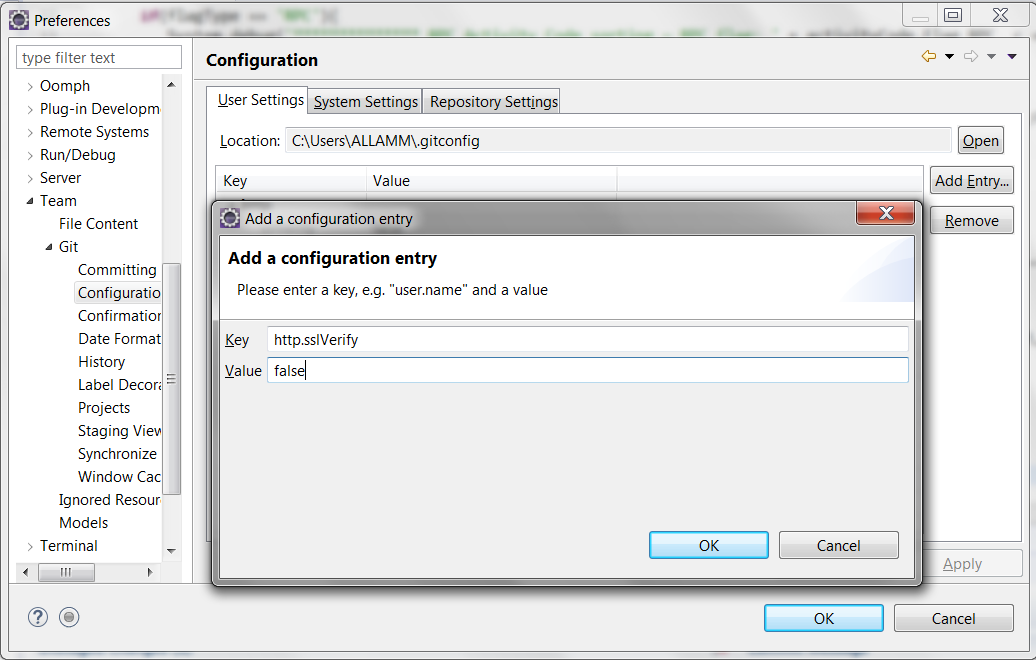
* In proxy bypass Click Add Host button and enter host names as below

\*.tfs.toyota.com, xpvapl0170.tfs.toyota.com

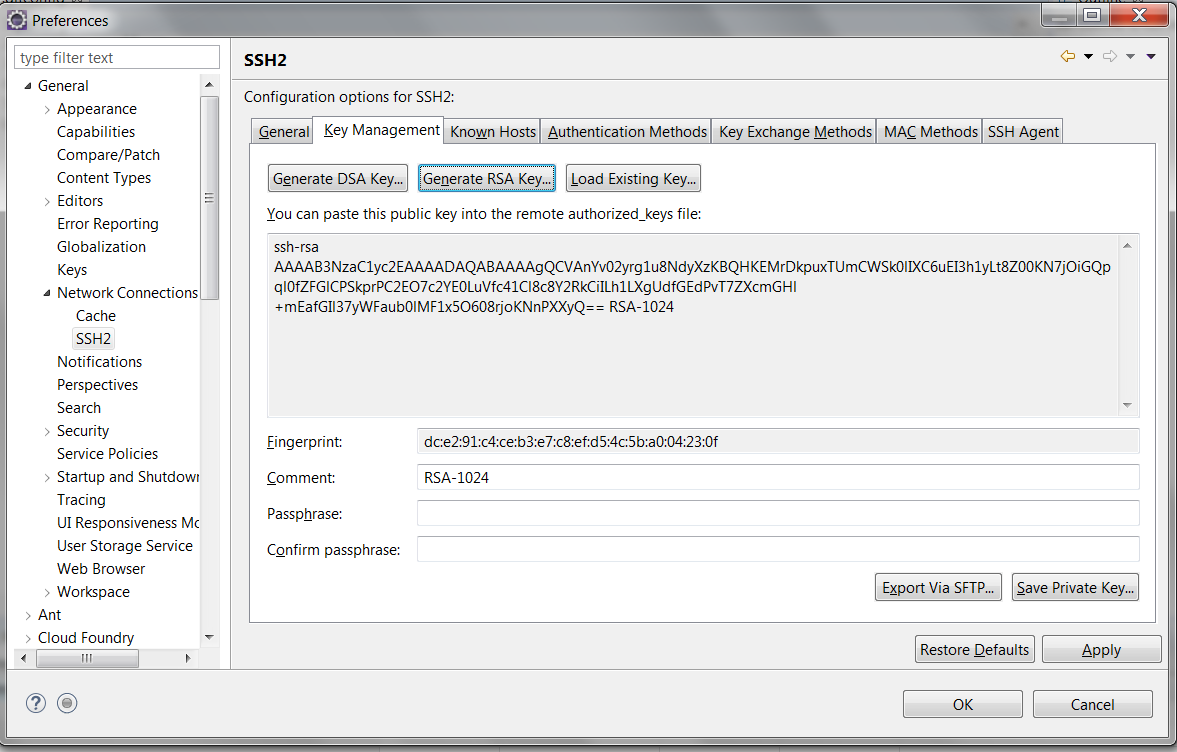


* Click Apply and OK
* Navigate to Windows -> Preferences ->Team -> Git -> Configuration

Under the User settings tab click Add Entry button and enter the Key (http.sslVerify) and value (false)



* Click OK
* To configure SSH, Go to Windows -> Preferences -> General -> Network connections -> SSH2 -> Key management -> Click on Generate RSA key



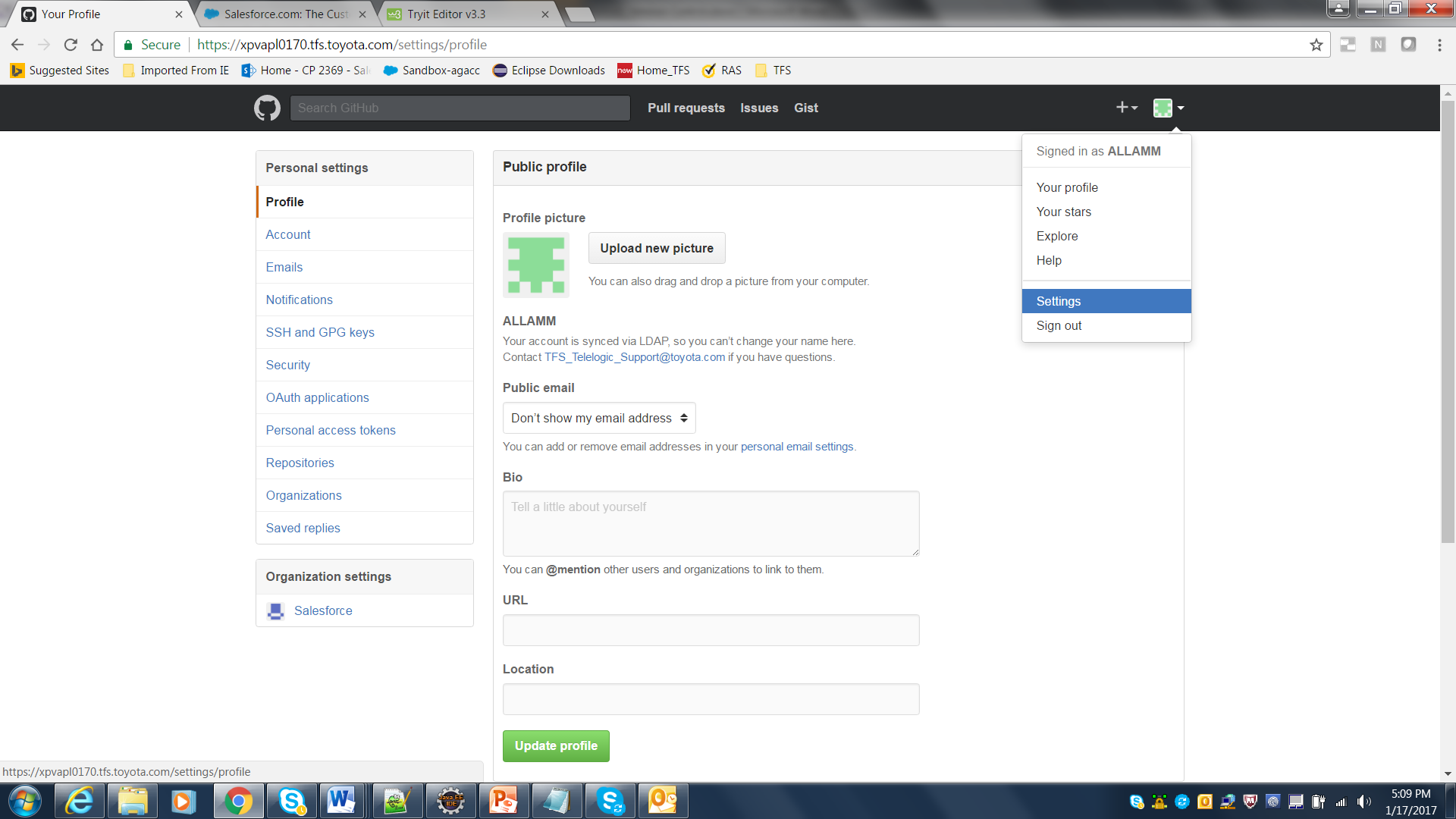
* Click OK
* Copy the generated key from eclipse and paste it in the GitHub Key section.
* Follow the below stpes for Add SSH key for Eclipse

Note: Use google chrome for GitHub login

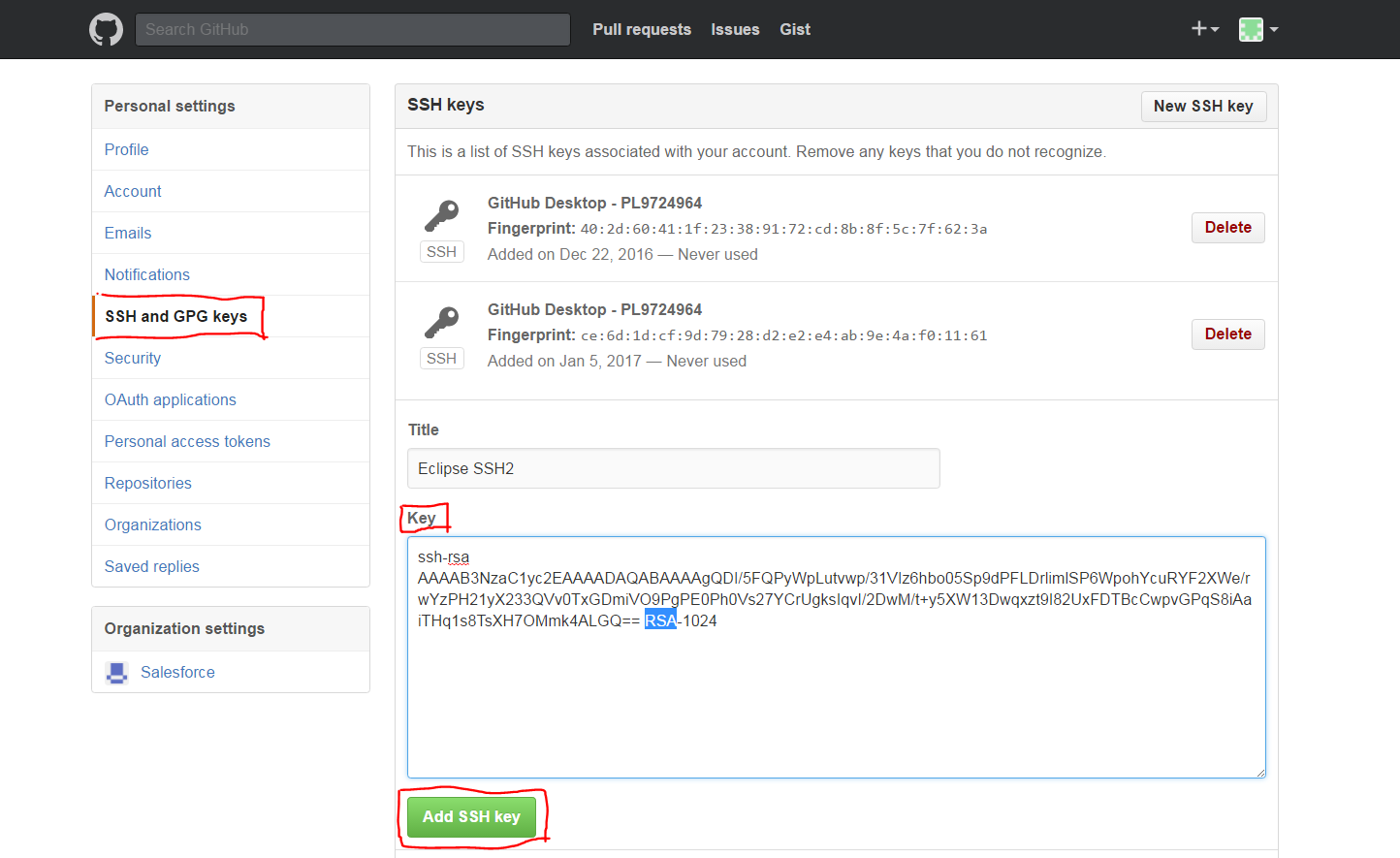
* + - Click the below link and enter the online TFS GitHub login credentials

<https://xpvapl0170.tfs.toyota.com>

* + - Click on View profile and more then find the Settings option



* Copy the generated key from eclipse and paste it in the GitHub Key section.
* Click Add SSH key button



* Click OK and restart eclipse

## 

## Install force.com IDE on eclipse

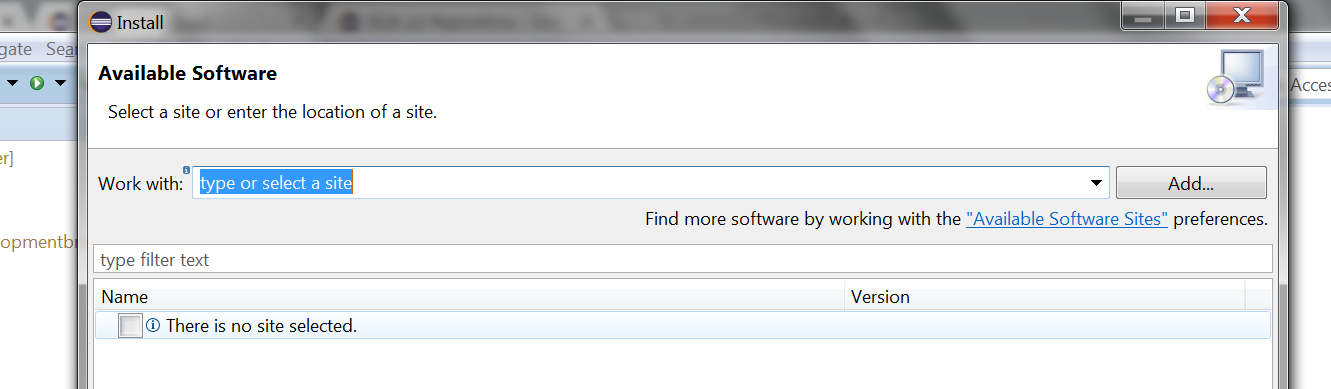
Follow the steps provided in <https://developer.salesforce.com/page/Force.com_IDE_Installation> to install force.com IDE

## Install eGIT

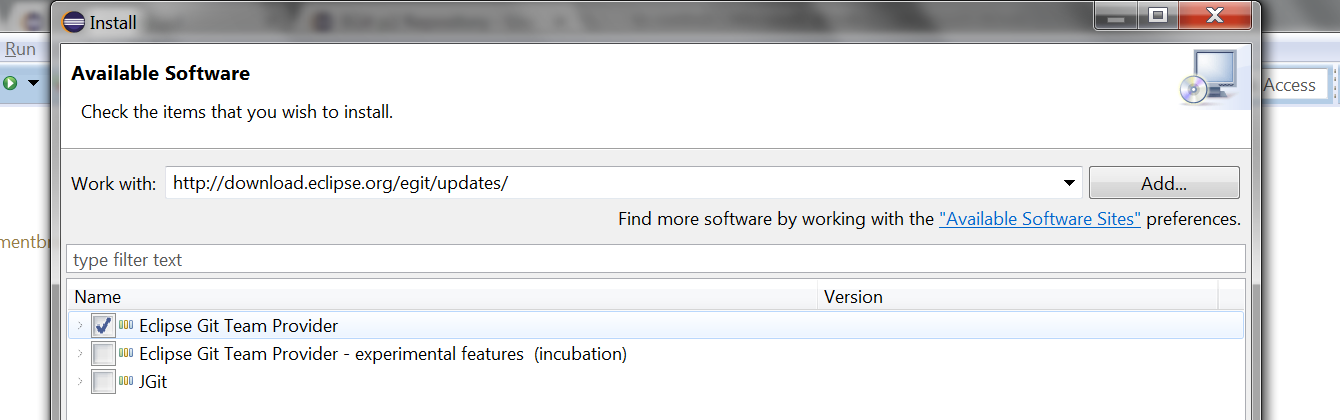
Note: Eclipse Neon comes with eGIt installed by default.

For eclipse versions which do not have eGit installed, follow the steps provided below.

* Navigate to Help -> Install New software



* Enter http://download.eclipse.org/egit/updates/ in the Work with field and press the Enter key



* Select the Eclipse Git Team Provider version
* Click next to install eGit.

## Set Up Organization on GitHUB

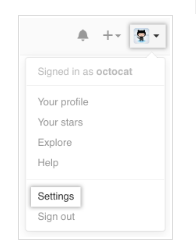
In order to set up version control for salesforce with GitHub an “Organization” has to be created in GitHub. Any number of repositories can be created under an organization.

As part of the POC an organization called salesforce has been created on GitHUB.

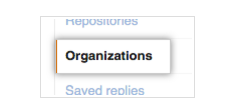
The GitHub admin will have to perform the steps mentioned below in order to set up an organization.

NOTE: When you create a new organization from scratch, it doesn't have any repositories associated with it. At any time, [all members of an organization can add new repositories](https://help.github.com/articles/repository-permission-levels-for-an-organization/#creating-repositories), and members with owner or admin permissions can transfer existing repositories.

* Login To GitHub
* In the top right corner of any page, click your profile photo, then click **Settings**



* In your user settings sidebar, click Organizations.



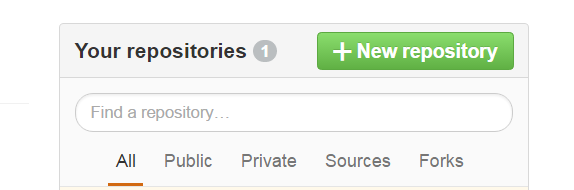
* In the "Organizations" section, click New Organization.
* Under "Organization name", give the organization a name.
* Click Create organization.
* Once the Organization is created the admin can add team member to the organization from the Teams Tab

## Set Up repository in GitHUB

**A repository in GitHub represents the salesforce code base which will be used for the POC.**

**To create a repository in GitHub:**

* Navigate to the Organization under which the repository needs to be created
* Click on the New Repository Button



* Enter the name of the repository, select public if the repository has to be visible to everyone, select private if the repository has to be visible to only a set of users



* Click on the “Create Repository” button.

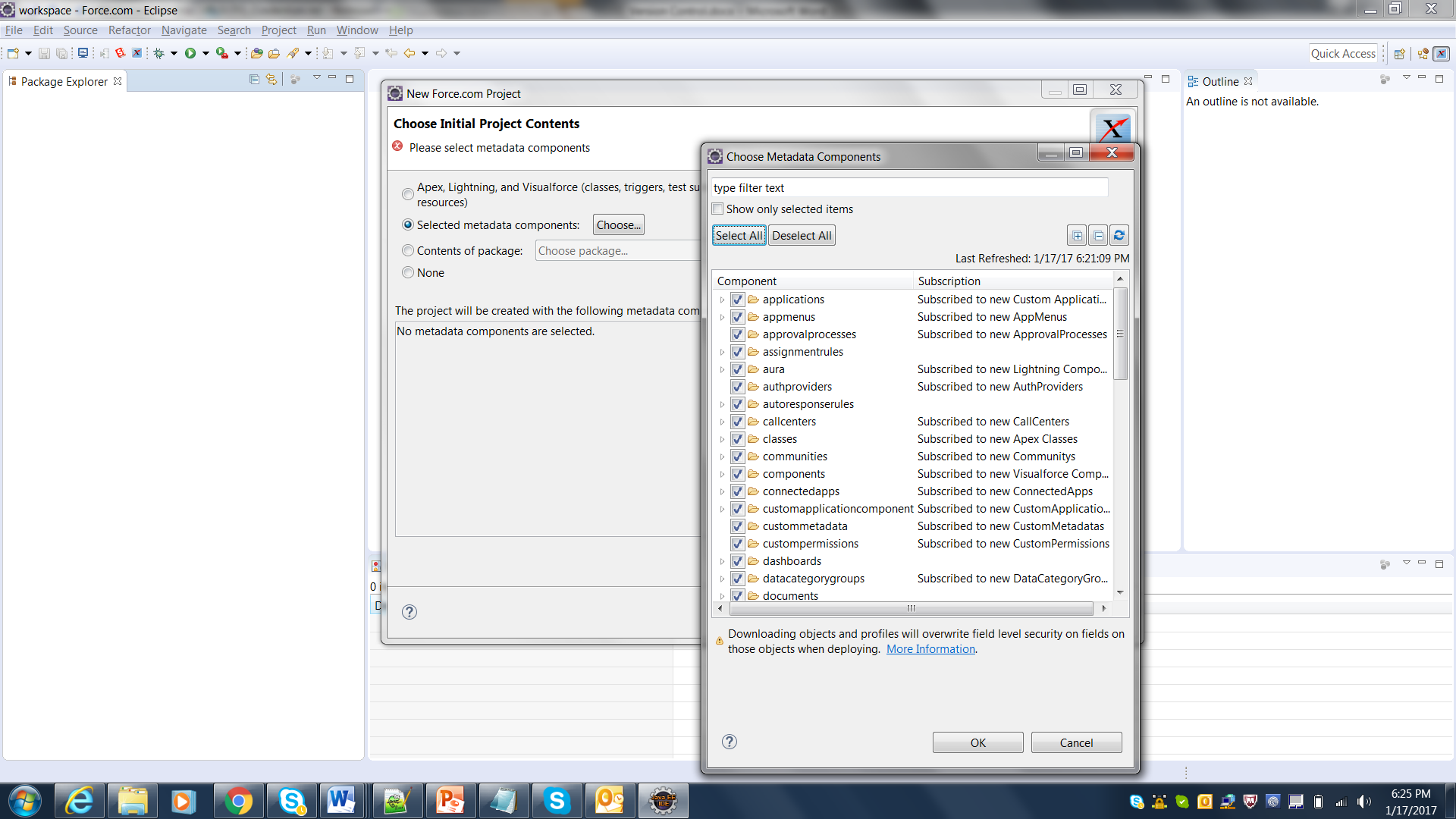
Note: The repository that has been created is a placeholder/folder for the salesforce code base.

Push salesforce repository to GitHub as part of initial set up-

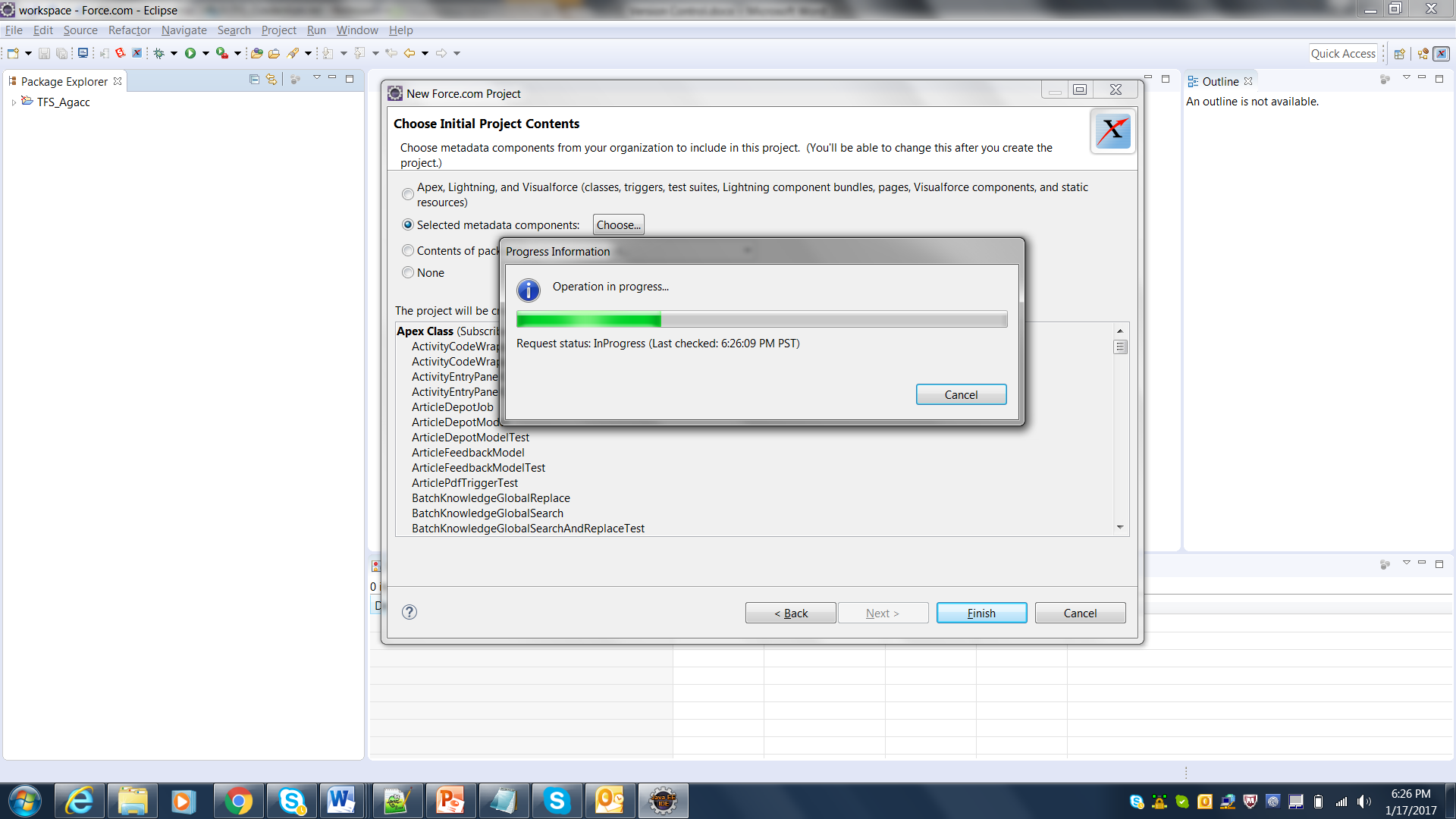
As part of the initial set up for a salesforce repository to be set up on GitHub, the admin must pull the salesforce metadata on to eclipse.

The admin is required to perform the steps provided below to pull the salesforce metadata on to eclipse

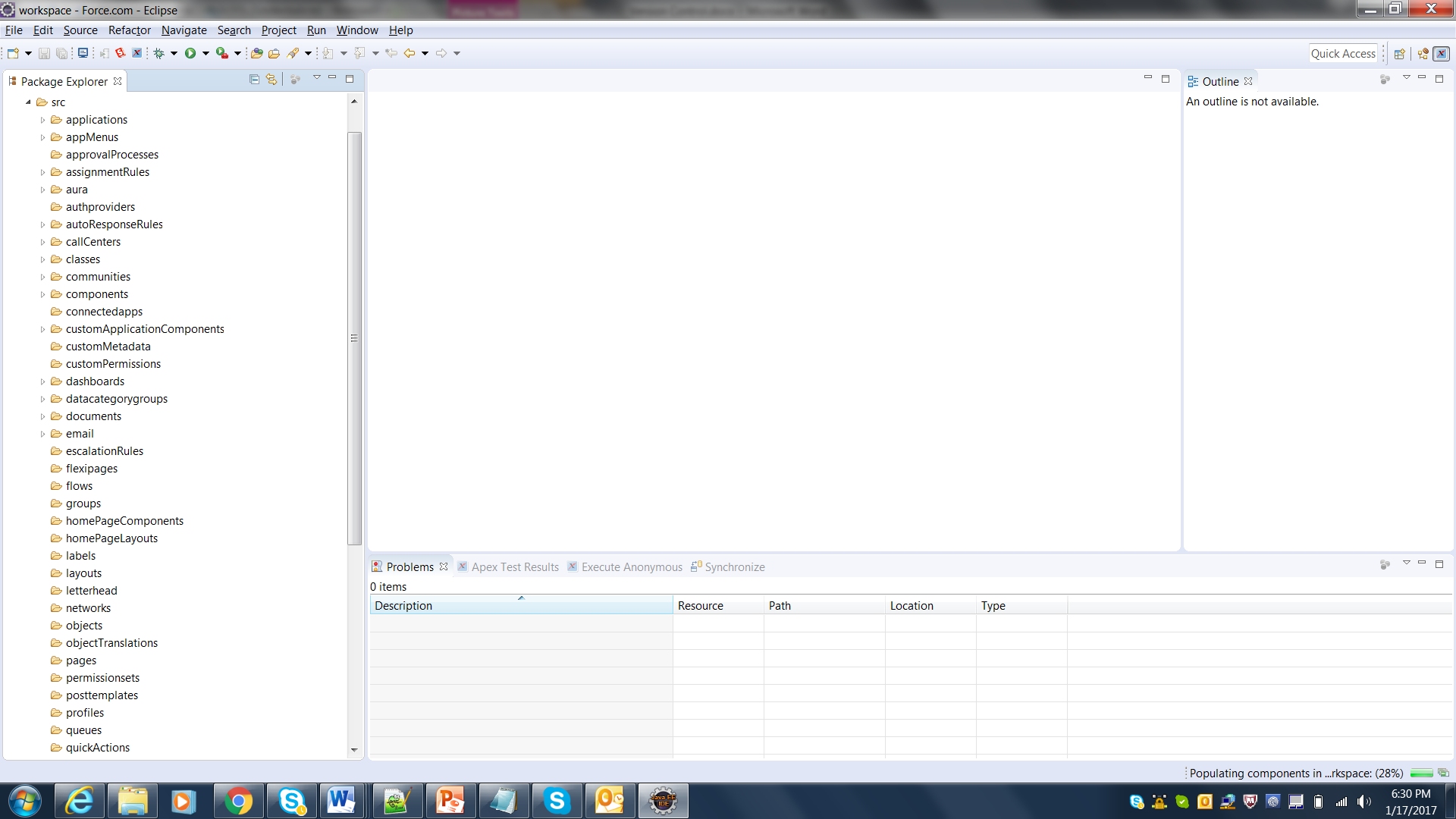
* Using eclipse force.com ide, create the local project from your production instance.
* Create New->Force.com Project
* Connect to your org
* Select all the components from the org:



* Click Finish.

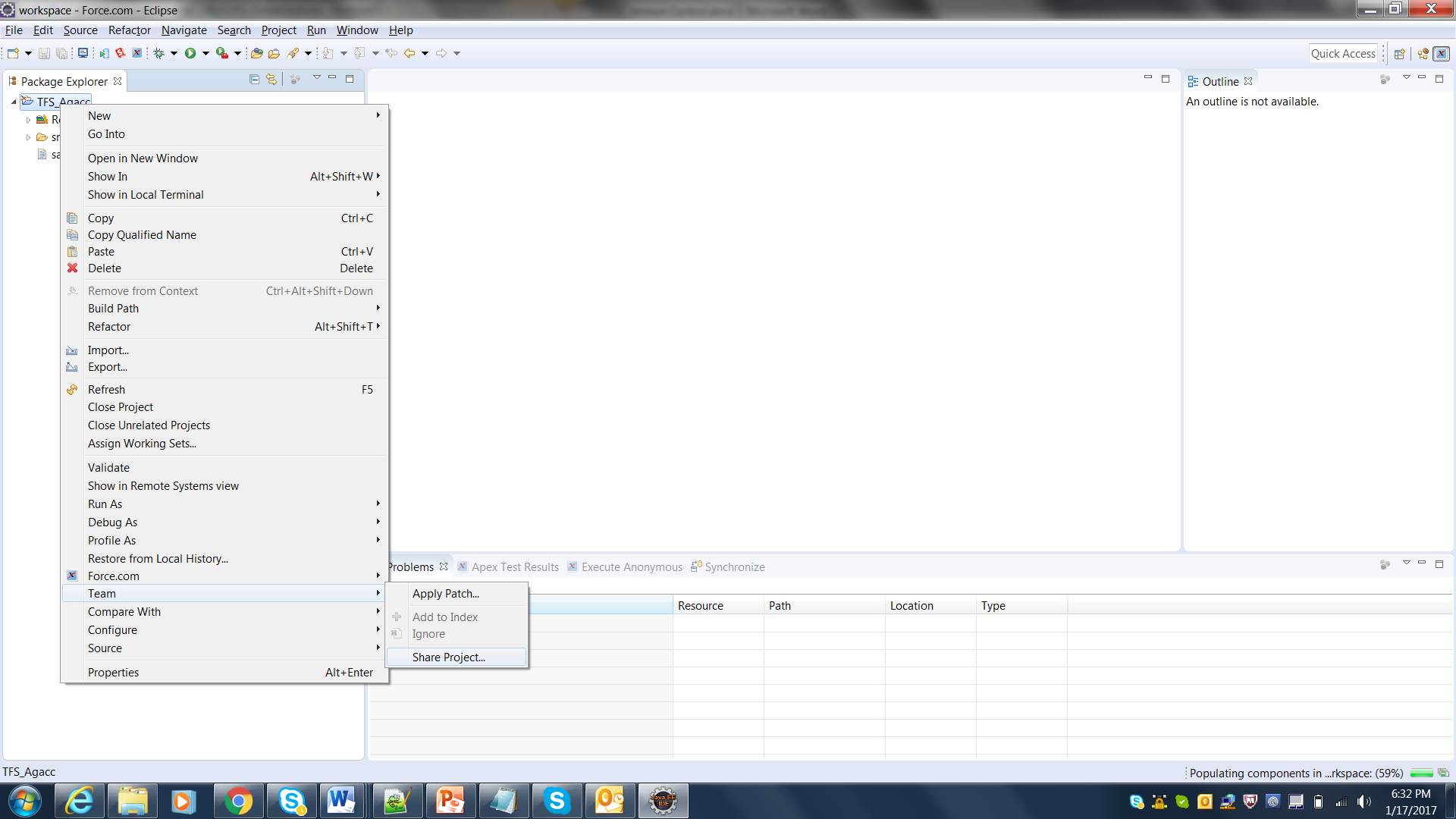


* On Click of finish all the metadata from salesforce will be pulled on to eclipse.

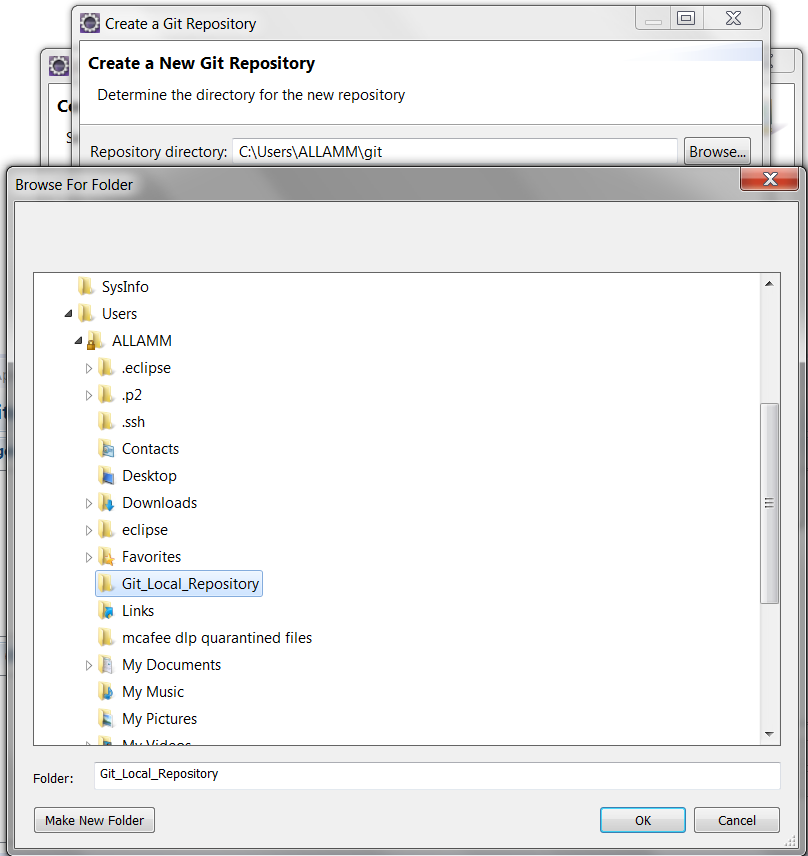


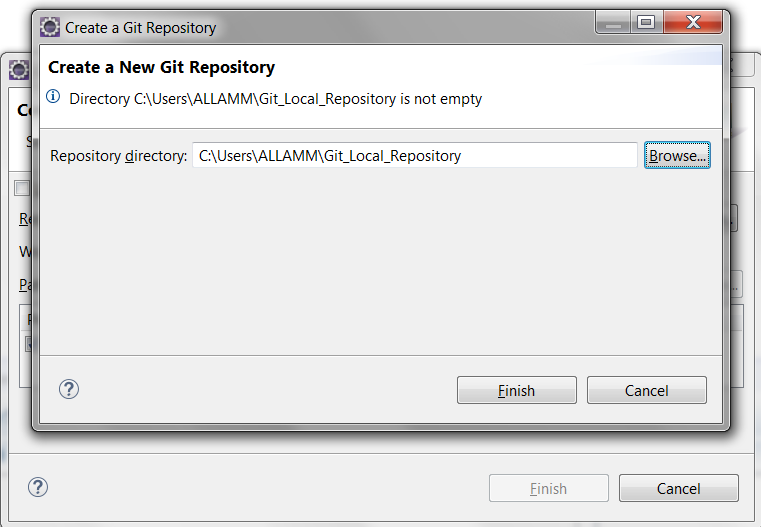
**The next step is to push the metadata to the repository that was created on GitHub. The admin needs to perform the following steps in order to push the salesforce metadata to GitHub**

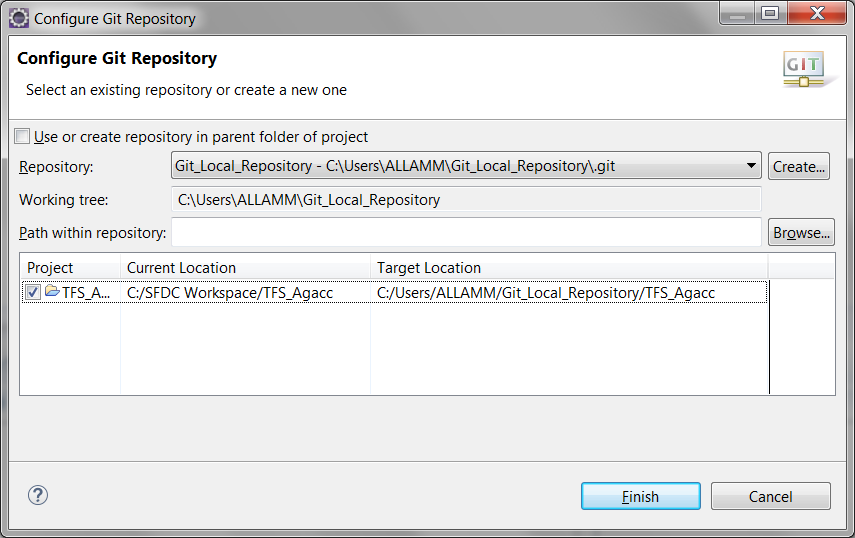
* **Right click on the force.com project that needs to be pushed and select Team->Share Project**



* **Create a new local folder for the repository (Local Git repository). Click on the create button**



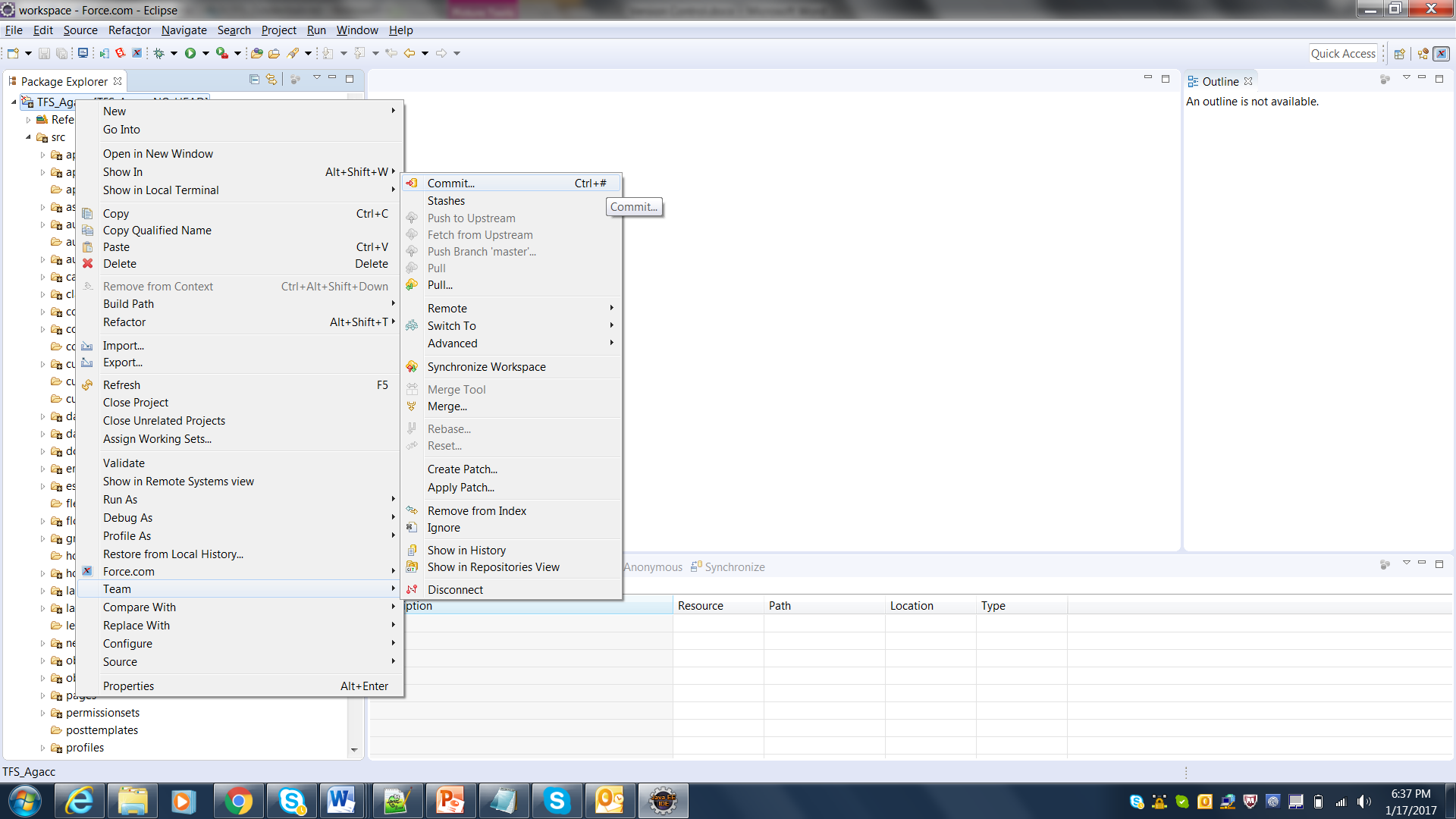




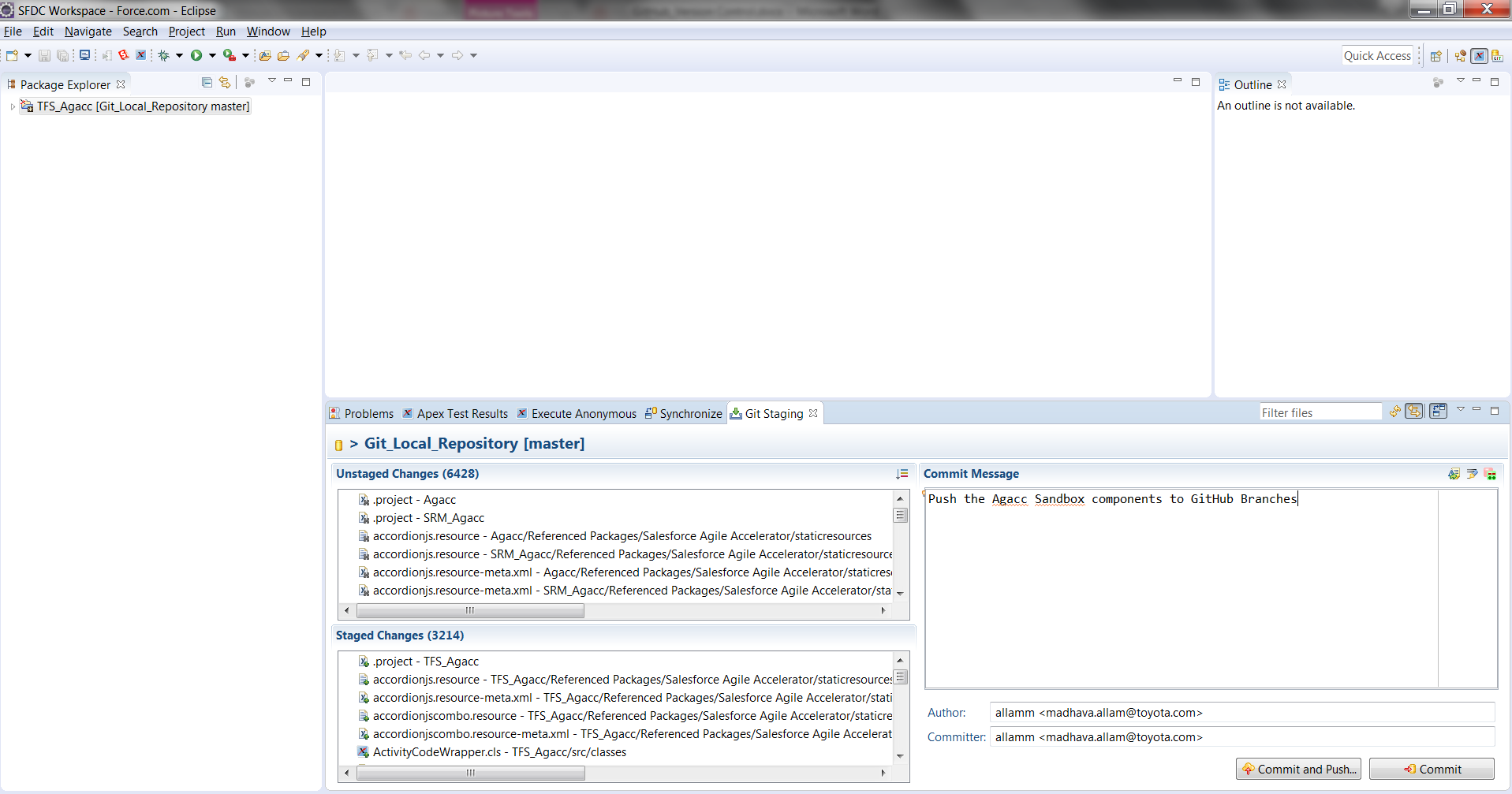
* **On click of finish the salesforce metadata will be copied over to the newly created local git repository folder.**
* Select all the components of the project in left pane and right click -> Team->Add To Index



* Right click on the force.com Project that needs to be pushed to GitHib, Select team-> Share Project -> Commit

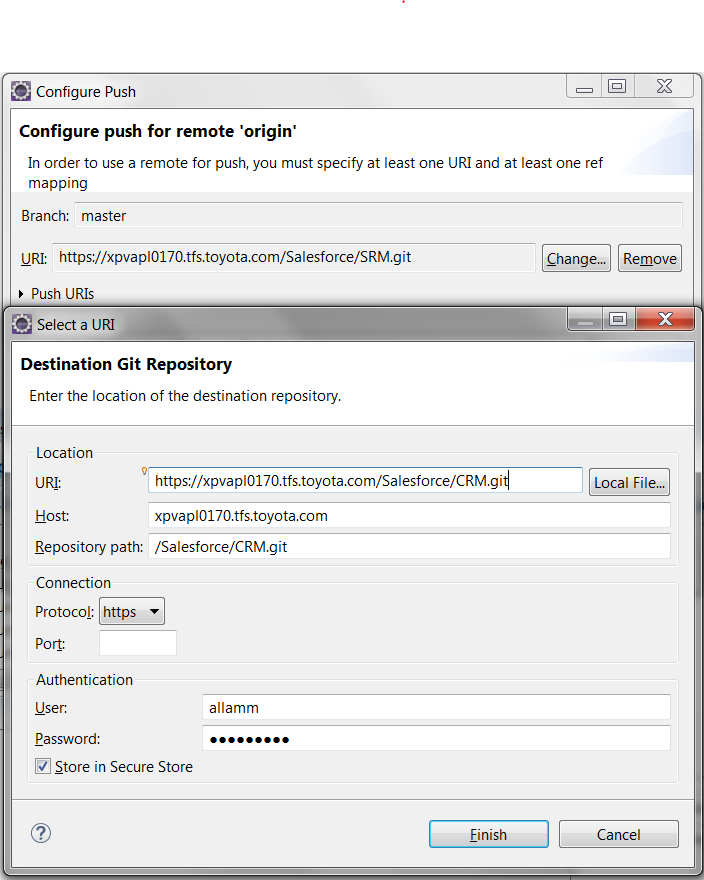


* It moves all the selected components to staged area in “git Repo “ view

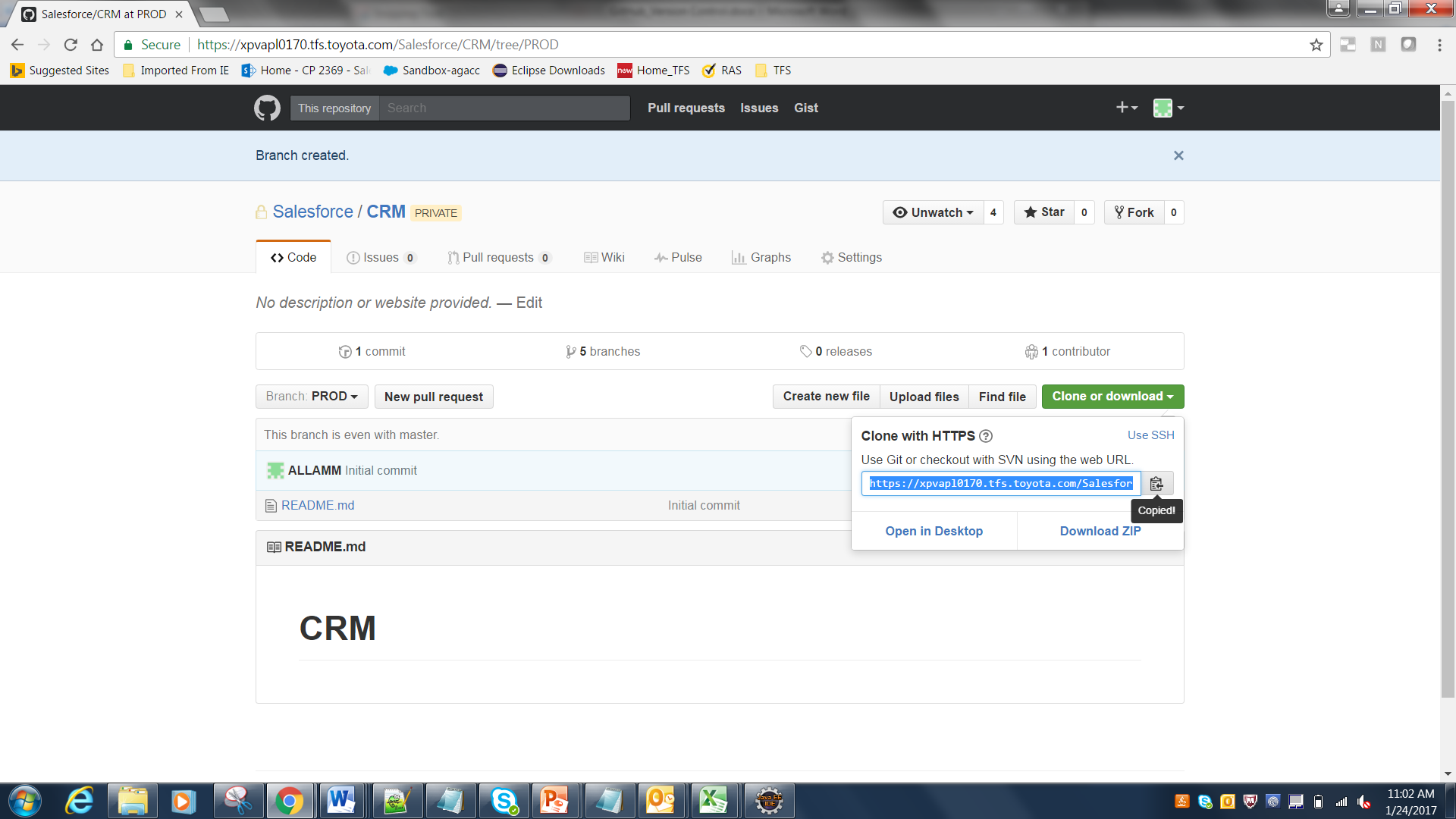


All the components which were added to the index will appear in the staged changes

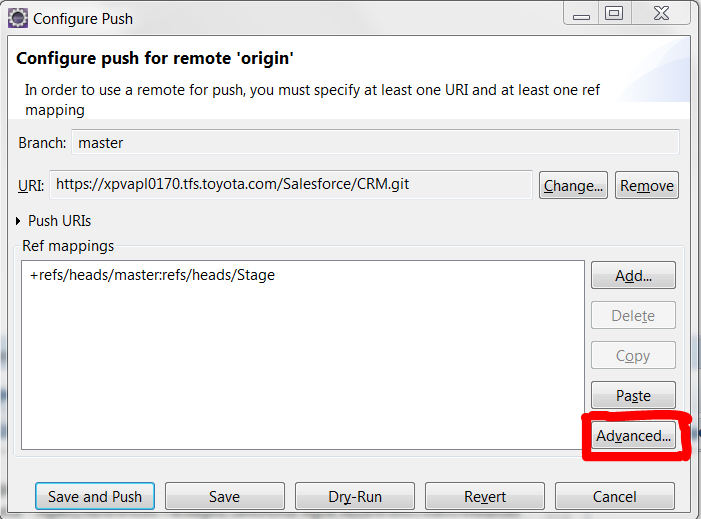
* Enter a Commit message and click on Commit and Push
* After hit the Commit and Push button configure the URI click on the Change button Enter the URI



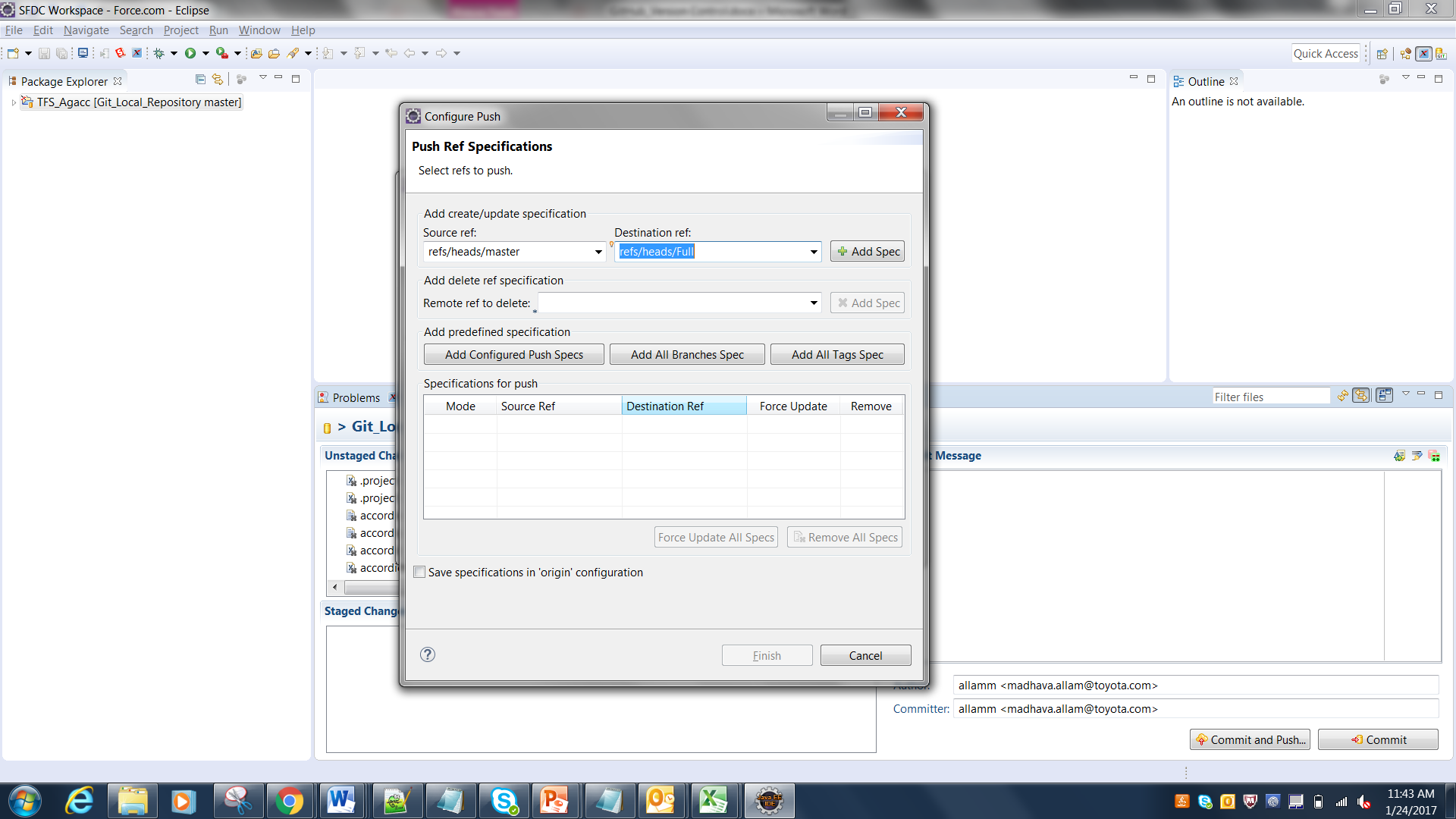
* In the URI field enter the URL of the GitHub repository. The URL can be found on GitHub



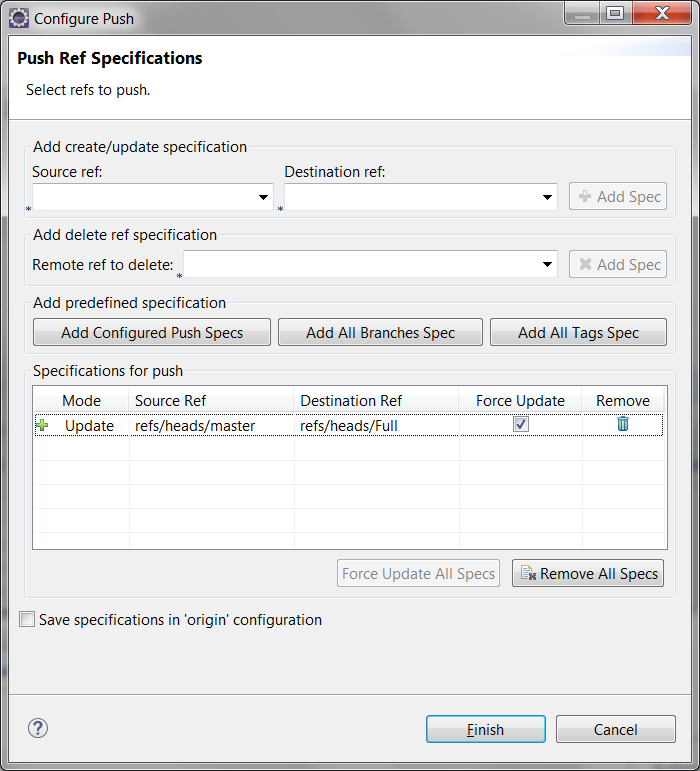
* Click on Finish button and select the option Advanced button as provided in the screenshot below



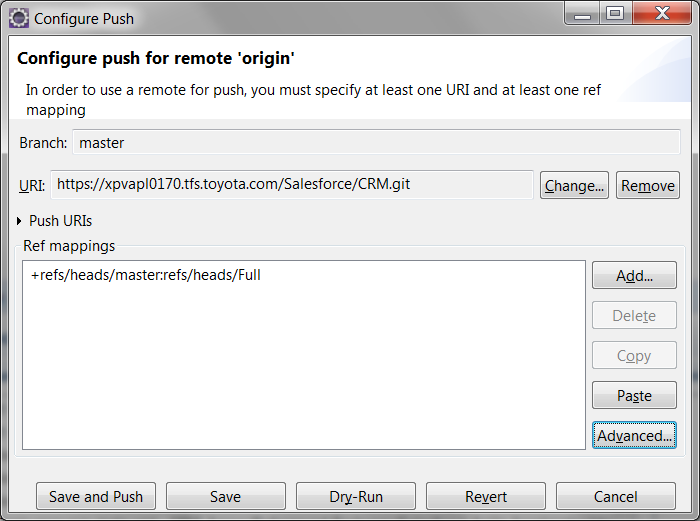
* After click the Advanced button select the source ref and Destination ref as provided in the screenshot below



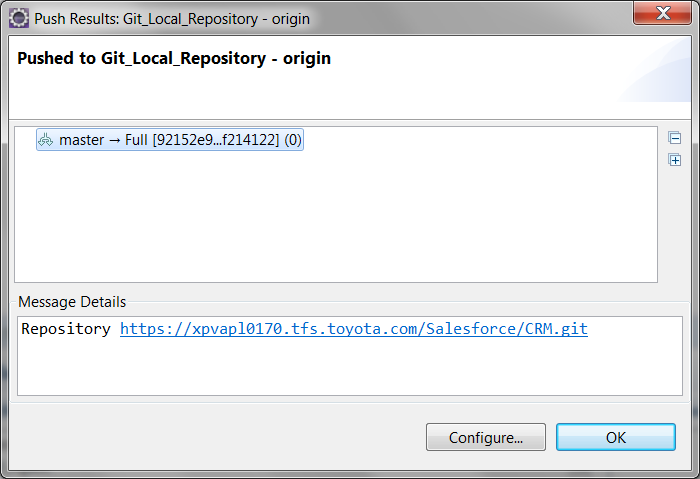
* After select Source and destination click on +Add Spec button, then select the Force Update box under the specification for push section



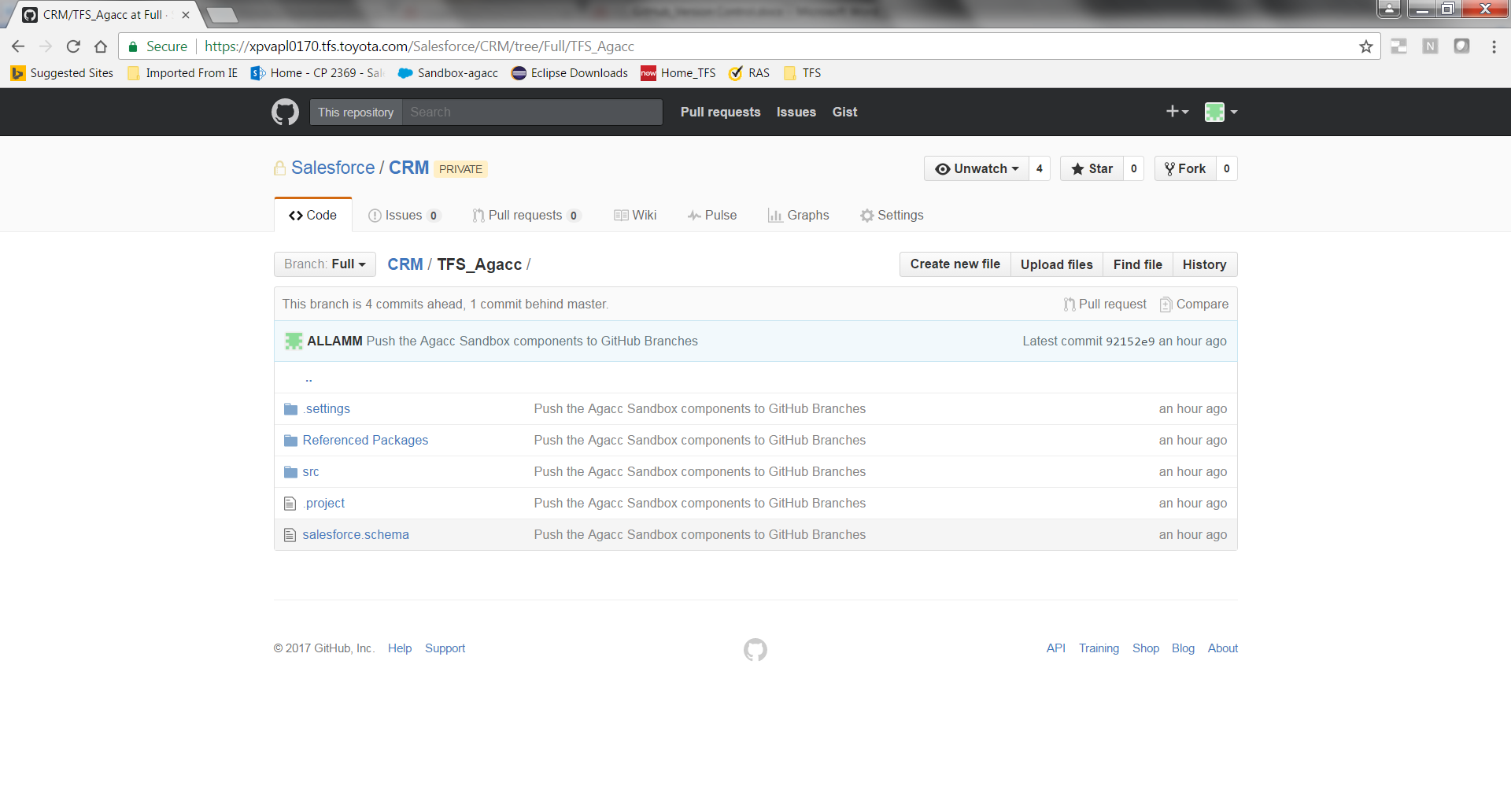
* Then click on Finish button.
* Click on Save and push button as provided in the screenshot below

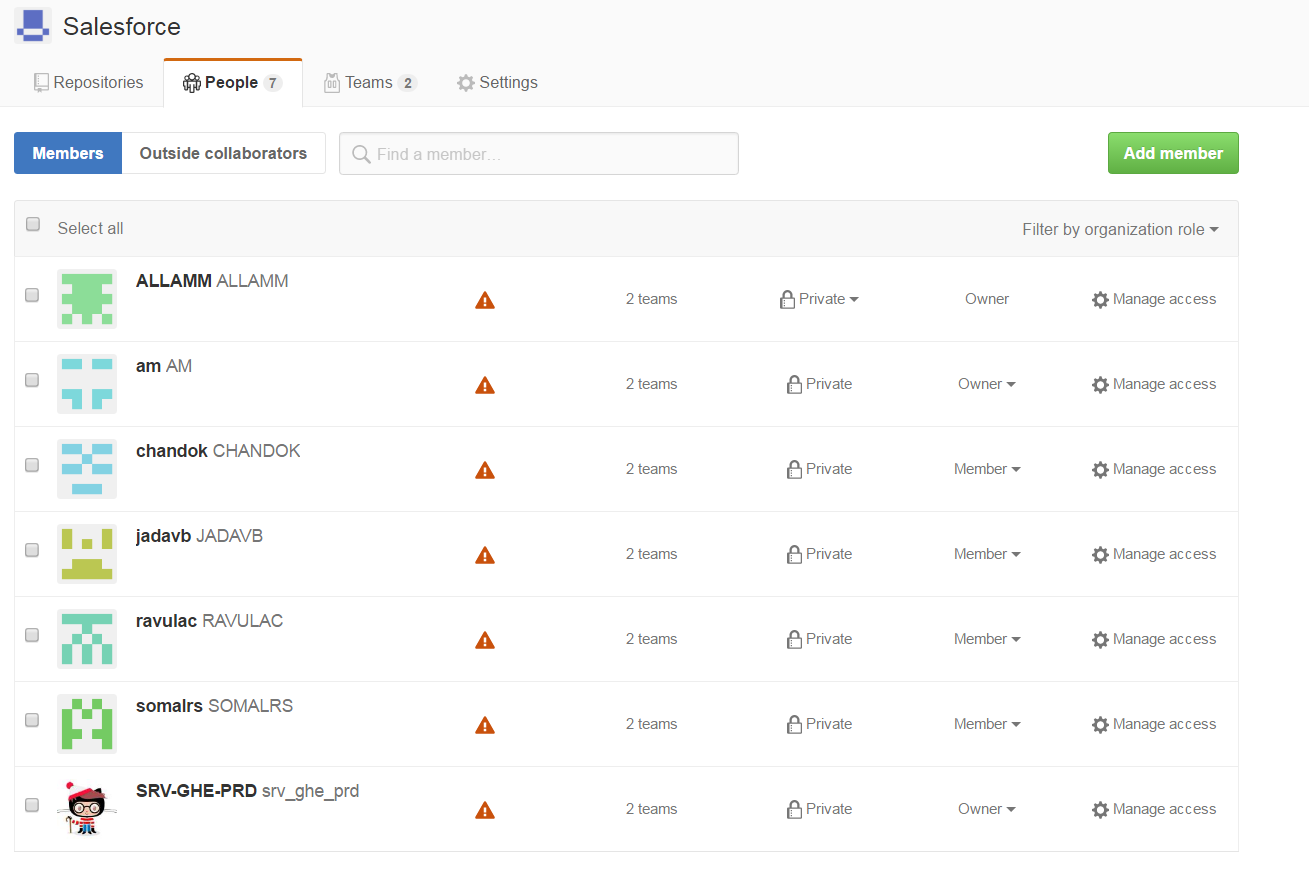


* Click OK button.



* The salesforce metadata can be verified in the repository on GitHub





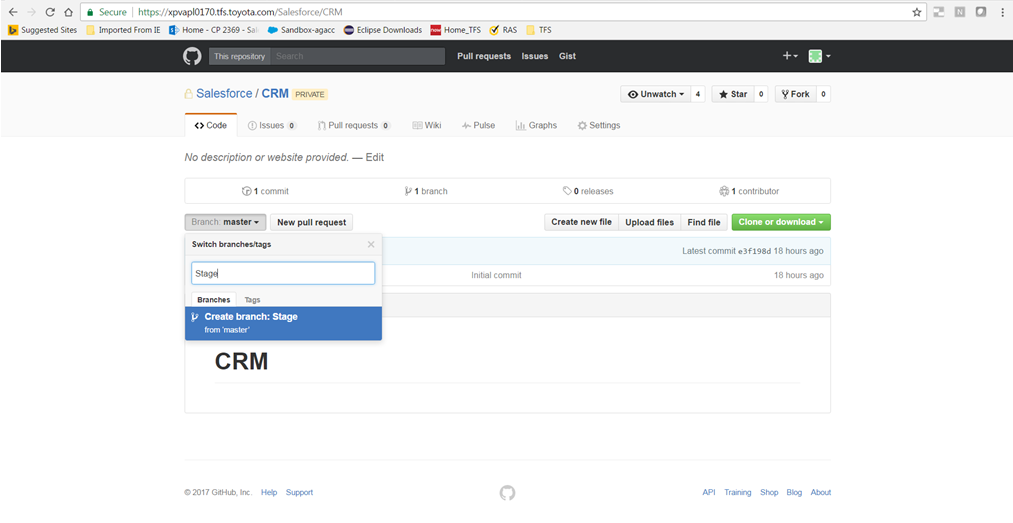
**Note: As part of the POC an organization called salesforce is created.**

## Creating a branch in GitHub

Once the master branch is created on GitHub any number of branches can be created on the Master. Each Branch can potentially represent a release. The Branches can be “merged” to the master branch and pushed into Production.

To create a branch, the GitHub Admin will have to perform the following Steps.

* Navigate to the repository for which a branch needs to be created.
* Click on the dropdown/button named “Stage:**Master”**



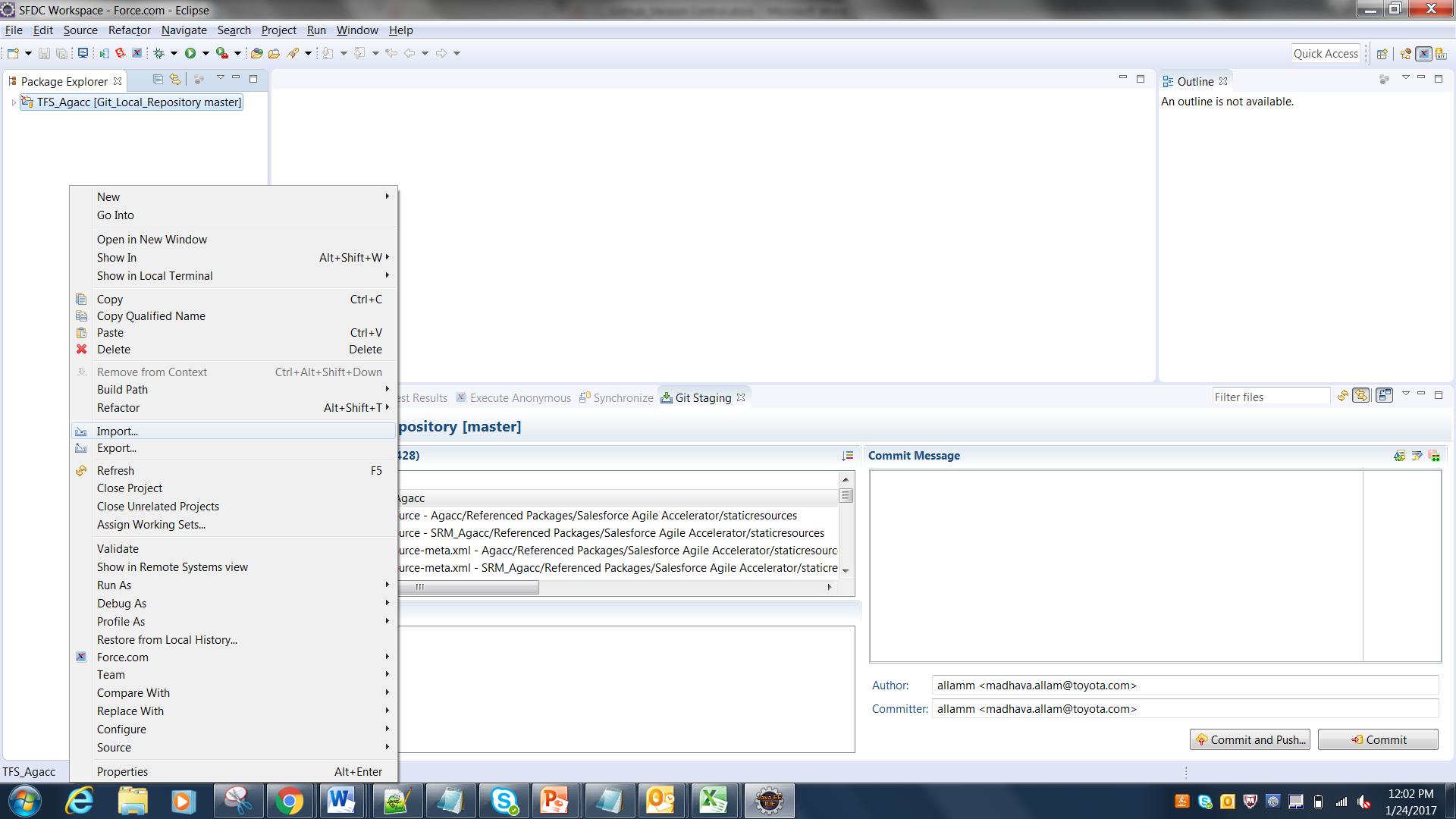
* Enter the name of the branch which needs to be created and press Enter
* A new Branch for the repository will be created

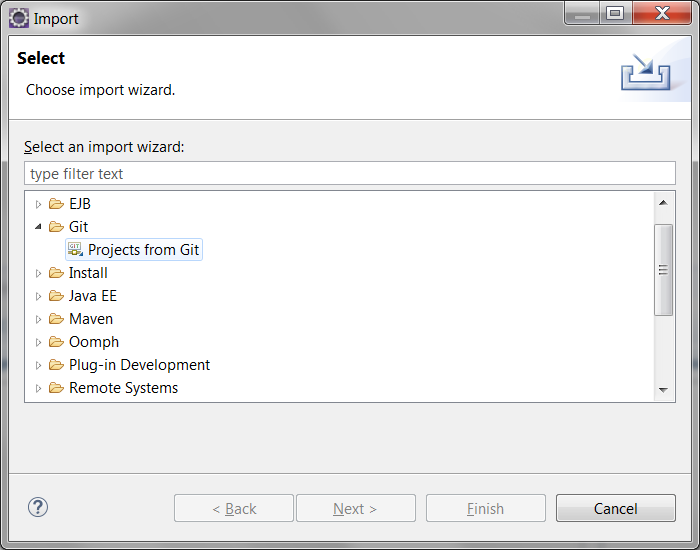
## Fetch branch from GitHUB

Once the Branch is created, each of the developers working on a specific release will have to pull the branch on to their local Git Repositories using eclipse.

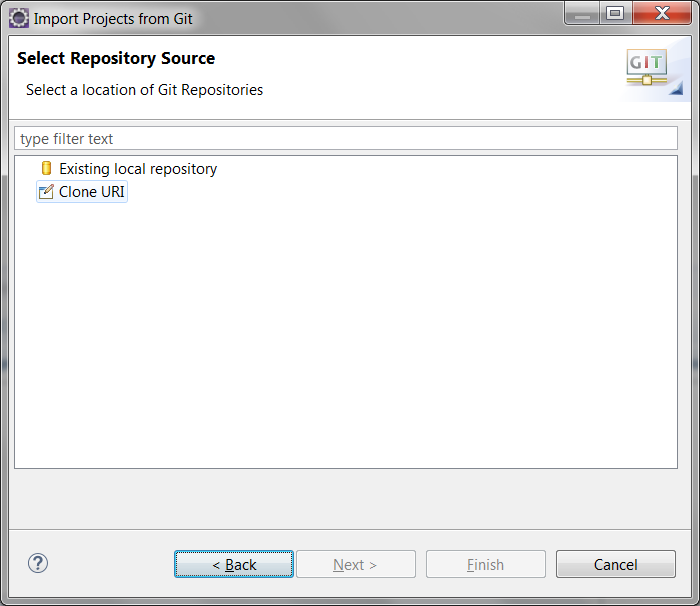
The steps bellow illustrate the process of pulling the branch on to local Git repository

* In the project explorer pane in eclipse, right click and select Import
* In the dialog box which opens on click of Import, select “Projects from Git” and click Next

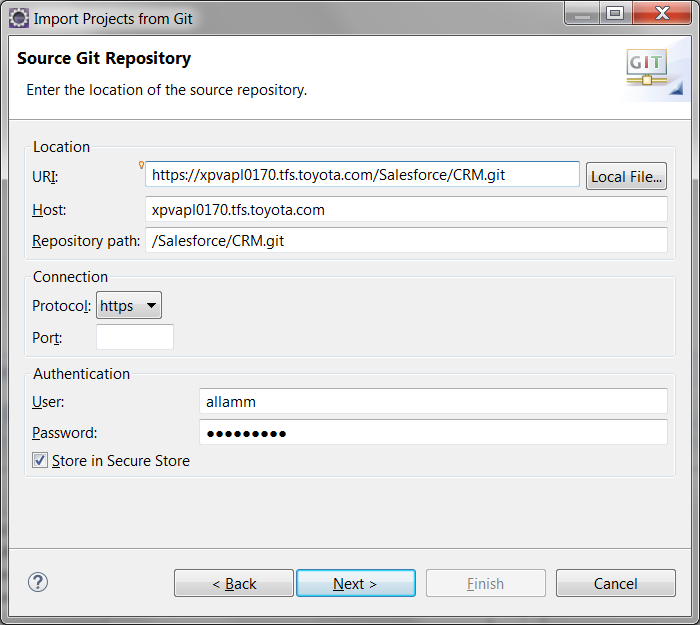




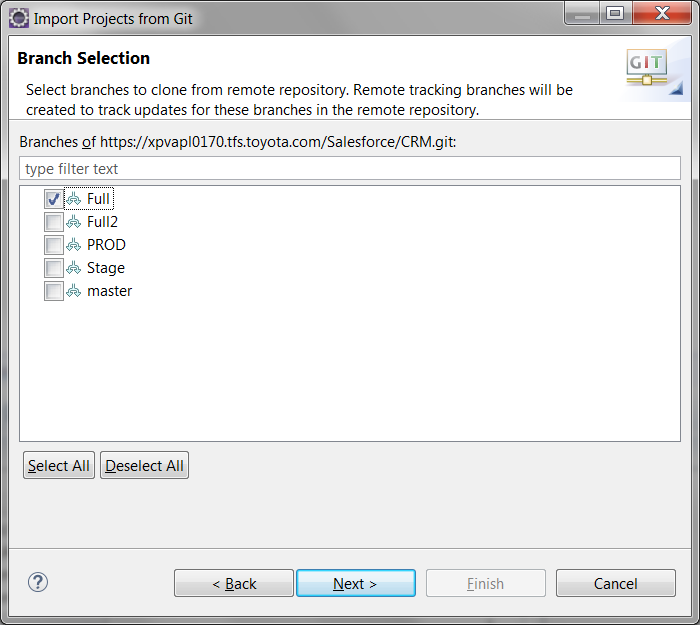
* Select Clone URI



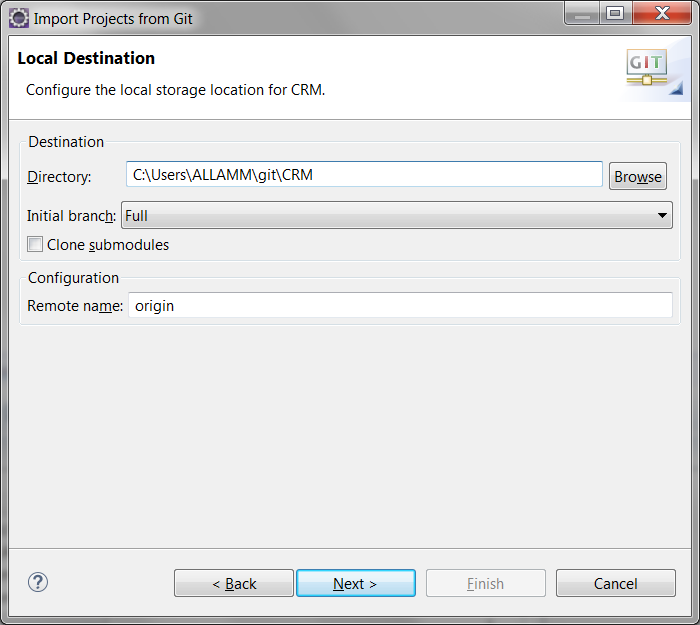
* Enter the URI of the repository that needs to be pulled and click Next



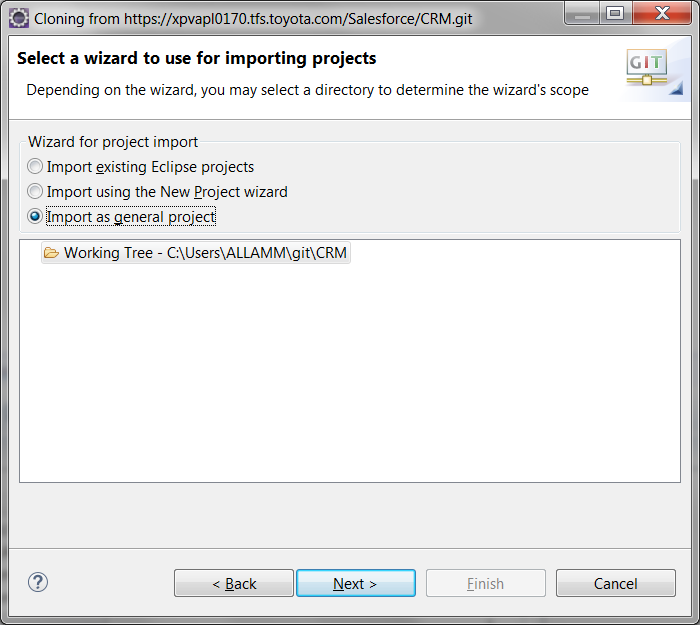
* Select branch which needs to be pulled and click Next



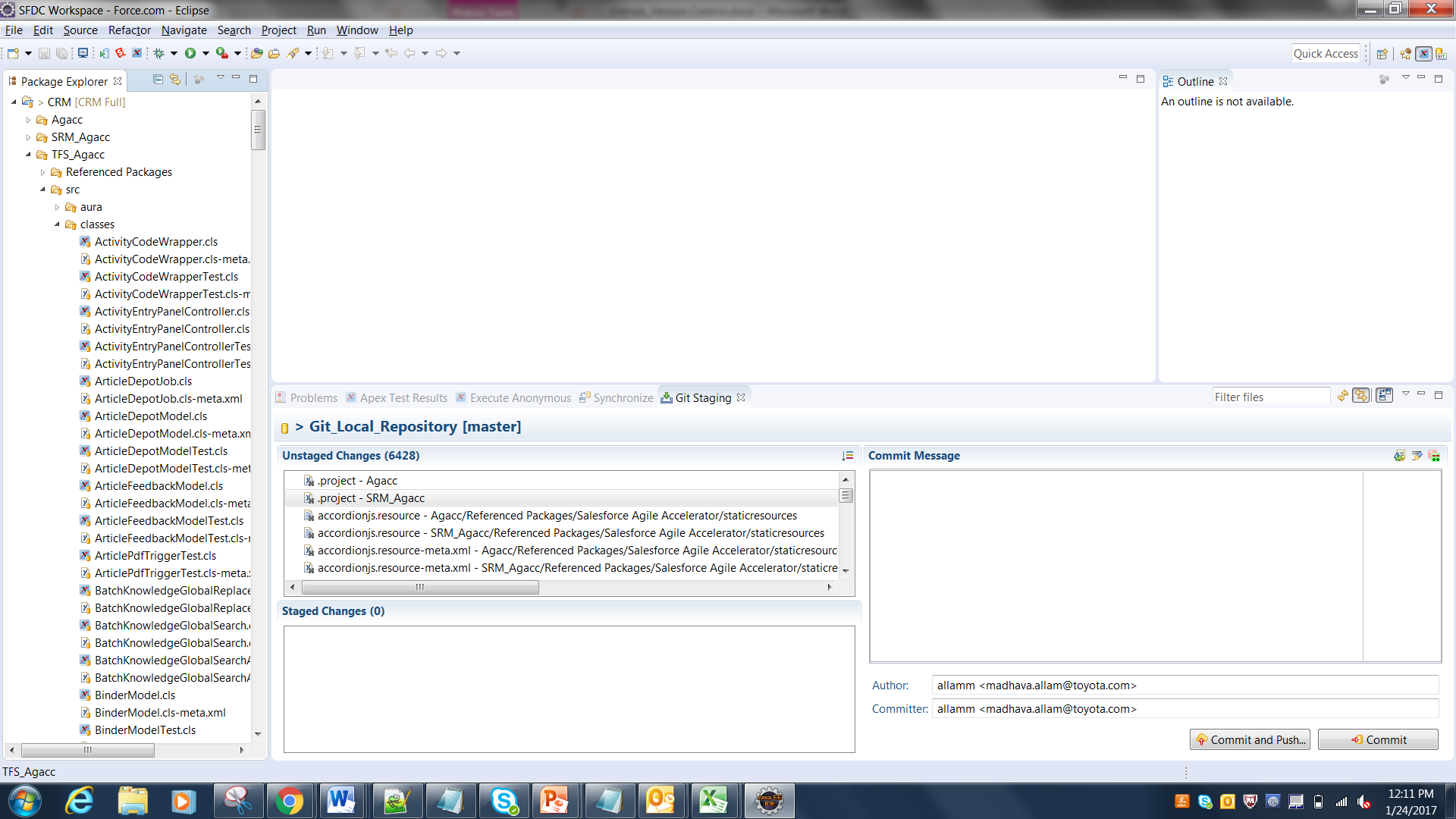
* Check the Clone Submodules Checkbox in the subsequent dialog box and click



* On click on Next all the salesforce metadata from the branch on GitHub will be pulled on to the local Git Repository.



* The new development branch will appear on Eclipse



# 

# Appendix A - Document Information

## Business Requirements Document Revision History

Below is a record of all modifications made to this document.

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Name** | **Details** |
| 1.0 | 01/113/2017 | Madhava Allam | Initial Draft. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## Template Owner

Please contact the template owner to provide feedback or suggest improvements.

\* Note: Used the TFS PMO BRD Template as a base and created this document based on that original BRD template.

|  |  |
| --- | --- |
| TFS PMO | |
| Company: | TFS |
| Dept: | TFS Business Technology Solutions |
| Email: | [TFS\_PMO@Toyota.com](mailto:TFS_PMO@Toyota.com) |

## Information Classification

The default document classification is “Protected Information.” Please update the footer with the appropriate classification (i.e. Confidential, Public or Protected).

# Appendix B - Acceptance & Sign-off

The following stakeholders have reviewed the document and have indicated:

|  |  |
| --- | --- |
| **Lead** | **Sundar Movva** |

**STAKEHOLDER APPROVALS:**

containing Email Aprvls

**Document Control**

**Document Revisions**

| **Version** | **Date** | **Name** | **Details** |
| --- | --- | --- | --- |
| 1.0 | 01/13/2017 | Madhava Allam | Initial Draft. |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Document Approvals**

| **Version** | **Date** | **Name** | **Approvals**  **------------**  **Y; N; Conditional;**  **No Involvement** | **Comments / Suggestions** |
| --- | --- | --- | --- | --- |
| 1.0 |  |  |  |  |
| 1.1 |  |  |  |  |
| 1.2 |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**Document Reviewers**

| **Version** | **Date** | **Name** | **Approvals**  **------------**  **Y; N; Conditional;**  **No Involvement** | **Comments / Suggestions** |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# Appendix C - Attachments

**APNDX-C1**:

N/A