**What is GIT** : It is a version control system.

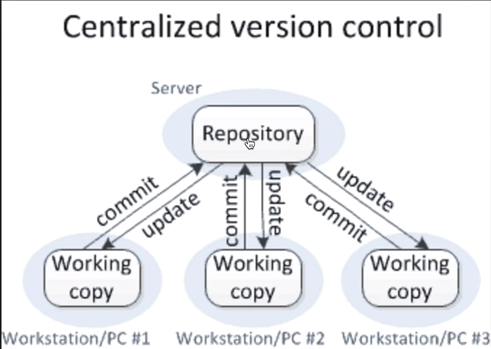
What is Version control system : If I have a file and I want to track all changes made in the file then version control system can help us to maintain a record of all the changes made OR it maintain version history. Version control is used to track changes in files and folders. We use GIT because it also helps us to collaborate in a team.

Version control system is of two types :

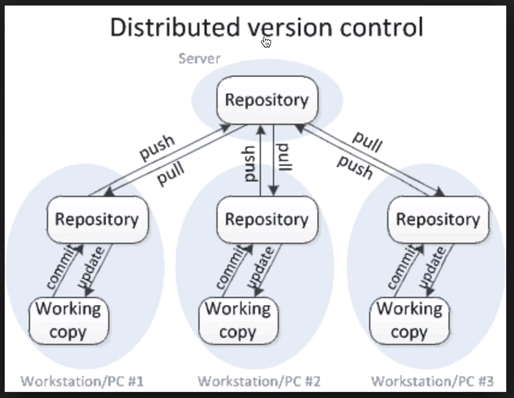
1. Centralized

2. Distributed (GIT is distributed VCS)

Centralized :



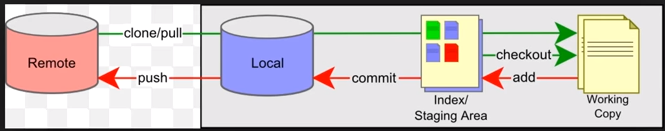
Distributed :



What is GIT-HUB ?

: It is a web site to upload your repositories online.

Workflow of GIT :



**Steps to add your project to GIT :**

1. Install GIT

2. cmd > git --version : will give you your current git version available on your machine.

3. Close that cmd

4. Go to the project folder of your machine > in address bar type CMD and enter. it will open a CMD at that location OR Right click > git bash here (use this way)

4. add a folder / file here (in your project then) cmd >  git status > enter

5. cmd > git add abc.txt ( lets assume that in project folder I have abc.txt and I want to add it to GIT. git add \*/git add . will add everything available in that folder)

6. cmd > git commit -m "added a file abc.txt" (" " in double commas, we can add a comment while adding a file.)

7. Now make some changes in text1.txt + add one more file (we will add changes and new file to git now)

8. cmd > git status (it will show that changes done in text1.txt file + a new file is added)

9. We have to add multiple files here now we can use : git add **.**    This command will add everything that is not added to git

\*\*\* till this point we have added everything to our LOCAL repository \*\*\* check the above diagram.

Now we will PUSH it to the remote repository.

IMP : As you probably know, git is a distributed version control system. Most operations are done locally. To communicate with the outside world, git uses what are called **remote**.

1. Log in to GIT HUB

2. Create a new repository by using '+' symbol in top left corner

3. cmd > git remote add origin [https://github.com/Pratik0705/TestApril2020](https://eur02.safelinks.protection.outlook.com/?url=https%3A%2F%2Fgithub.com%2FPratik0705%2FTestApril2020&data=04%7C01%7Cpratik.toke%40fisglobal.com%7C60722f5c01364ffbec7208d980ccb40a%7Ce3ff91d834c84b15a0b418910a6ac575%7C0%7C0%7C637682438982324023%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=ykXEN98NFij2NR9JGjsxsufd4wULfNSlUOX%2B2yQSiYc%3D&reserved=0) (the URL is of the newly created repository in step#2 ) (We are adding a URL as a Origin. And we will refer it as Origin in next commands instead of mentioning the URL again and again. So, it creates a new remote called origin located at git hub website)

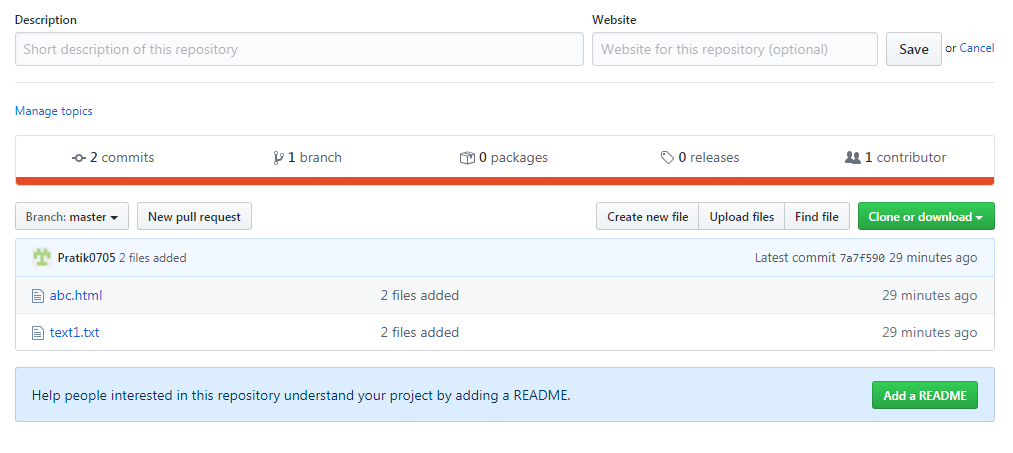
4. cmd > git push -u origin master (means push to master branch of Origin (i.e. above URL))

5. it will ask GIT HUB uname and pwd. enter that (we can also enter uname and pwd with the help of following commands)

    git config --global user.email "[abc@gmail.com](mailto:abc@gmail.com)"

    git config --global [user.name](https://eur02.safelinks.protection.outlook.com/?url=http%3A%2F%2Fuser.name%2F&data=04%7C01%7Cpratik.toke%40fisglobal.com%7C60722f5c01364ffbec7208d980ccb40a%7Ce3ff91d834c84b15a0b418910a6ac575%7C0%7C0%7C637682438982324023%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=PnVkOeOm5dmEtQcAW3ZE8Aa58QbBUNDVxjgajQbkGWY%3D&reserved=0) "username"

6. Now go to Git Hub and log in OR if already logged in then refresh the page. We can see the newly added project / files.



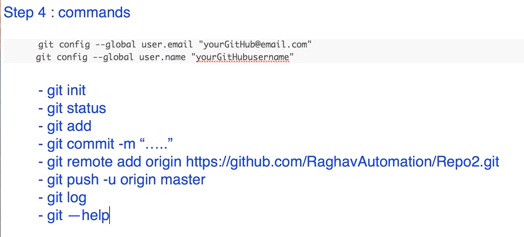
we can use

1. cmd > git log

this command shows us all log records

2. cmd > git --help

it will show all commands with its use



Branches :

It is not recommended to perform changes directly in the master branch. We always should make a new branch, checkout that branch and then make all changes / addition in that branch and after validating and testing we should merge those changes to the master branch.

1. Create branch :

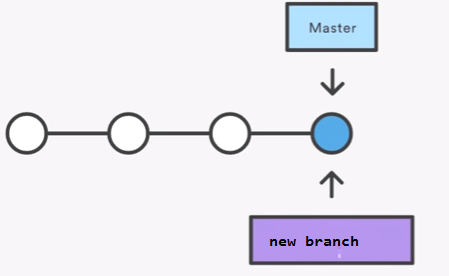
cmd > Git branch "branch name"

2. To work on the newly created branch, we have to checkout it. (checkout means switch to new branch)

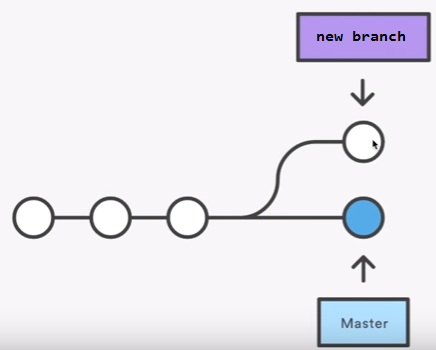
cmd > Git checkout "new branch name"

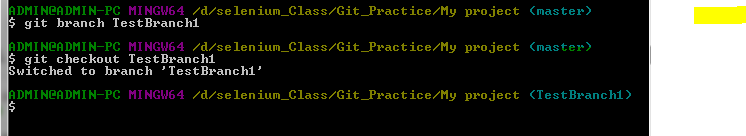
after the above command, Git shows a result as : Switched to branch " new branch name "

after creating a branch but before checkout :



After checkout (switch to new branch) :





In the above Image, it changed the branch after checkout from Master to testBranch1

After checkout, my local machine's changes will go to the New Branch and then we can merge the new branch to the master.

3. Now make some changes on local machine in your project (add / update a file)

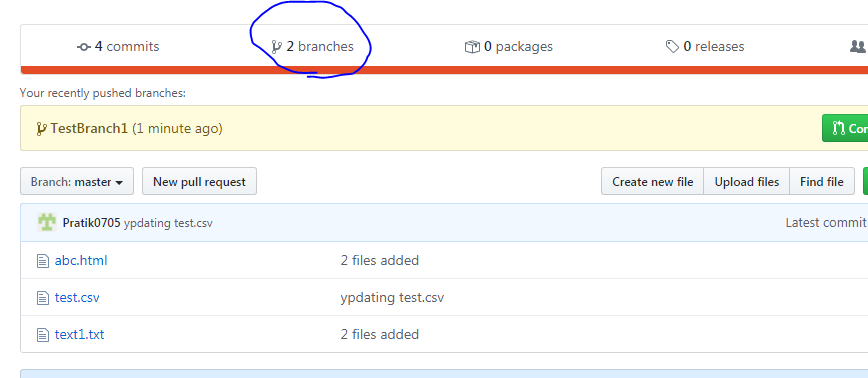
4. cmd > git add **.**

5. cmd > git commit -m "adding comment"

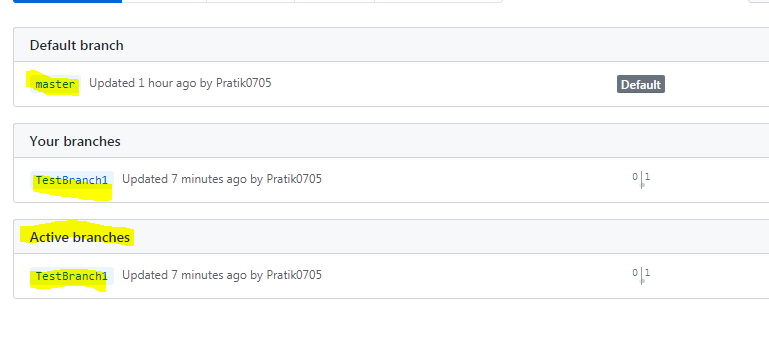
6. cmd > git push -u origin **TestBranch**-->( TestBranch  is the name of my New Branch )

In the above command now we will **not** write git push -u origin **Master** since we are now on the newly created branch and we cant push the changes directly to the Master. We have to push it to the new branch and then we will merge the new branch to Master

7. Now go to GitHub and refresh, the new branch will be visible.



click on this link



on cmd, if I execute :cmd > git checkout master. then I will be switched to master branch and in the folder(window folder explorer ) I will not be able to see the newly added file because that is on New branch not on the master. Now, if I execute : cmd > git checkout TestBranch. then I will be switched to my new branch and now in windows folder explorer I can see the newly added file as its available on this branch.

8. Merge the new branch to the Master : (NOTE : we have to checkout i.e. switch to Master before executing this command)

cmd > git checkout master

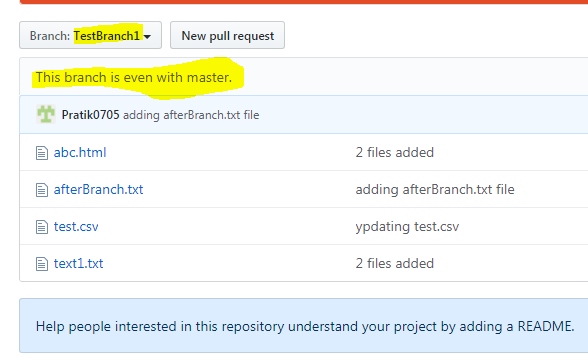
cmd > git merge "New branch name"

9. Now push to master

cmd > git push -u origin master

Now we can see the changes in Master branch on Git Hub and in folder explorer as well.

Now I can see a line in the below ss. It is my new branch and it says that new branch is in sync with the master branch



3. how to **DELETE** a newly created branch :

cmd > git branch -d "name of my new branch". It will delete the branch from local but not from Git Hub.

cmd > git push origin -- delete "name of new branch". this will delete from remote as well.

Continue with : [https://www.youtube.com/watch?v=govmXpDGLpo&list=PLhW3qG5bs-L8OlICbNX9u4MZ3rAt5c5GG&index=7](https://eur02.safelinks.protection.outlook.com/?url=https%3A%2F%2Fwww.youtube.com%2Fwatch%3Fv%3DgovmXpDGLpo%26list%3DPLhW3qG5bs-L8OlICbNX9u4MZ3rAt5c5GG%26index%3D7&data=04%7C01%7Cpratik.toke%40fisglobal.com%7C60722f5c01364ffbec7208d980ccb40a%7Ce3ff91d834c84b15a0b418910a6ac575%7C0%7C0%7C637682438982333976%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=1%2FF0EHEXL6ruxnumwAZHxSU5TZLoxBfVkMuEntzzuk0%3D&reserved=0)

**All Commands step by step :**

git bash here

1. git add. / git add abc.java

2. git commit -m "added a command" (comit to local repo)

3. git status (wil show the changes)

== till here code reaches to local repo now wil add to github

4. git remote add origin https://github.com/Pratik0705/TestApril2020

5. git push -u origin TestBranch (branch other than master)

6. > Enter credentials here

7. git checkout master (switch to master branch)

8. git merge TestBranch (merge TestBranch to master)

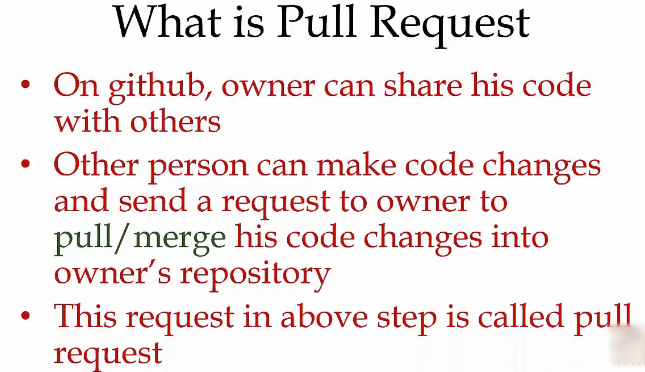
9. git push -u origin master (merged)

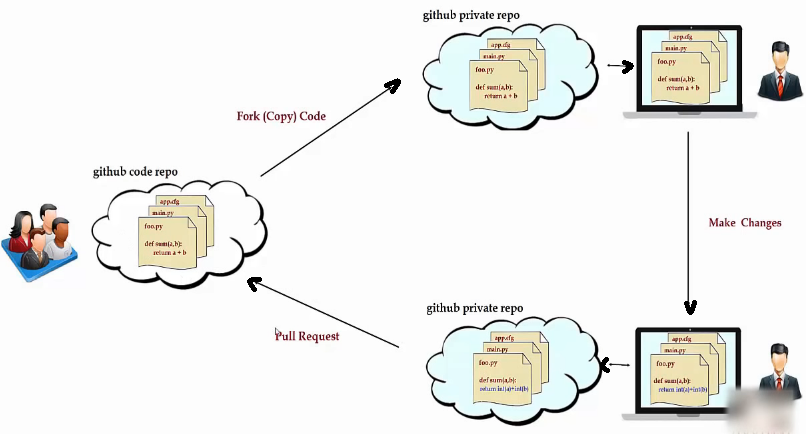
To DEKETE BRANCH

git branch -d "name of my new branch". It will delete the branch from local but not from Git Hub.

cmd > git push origin -- delete "name of new branch". Delete from github as well.

**GIT PULL REQUEST** :





My GIT repository#2 :

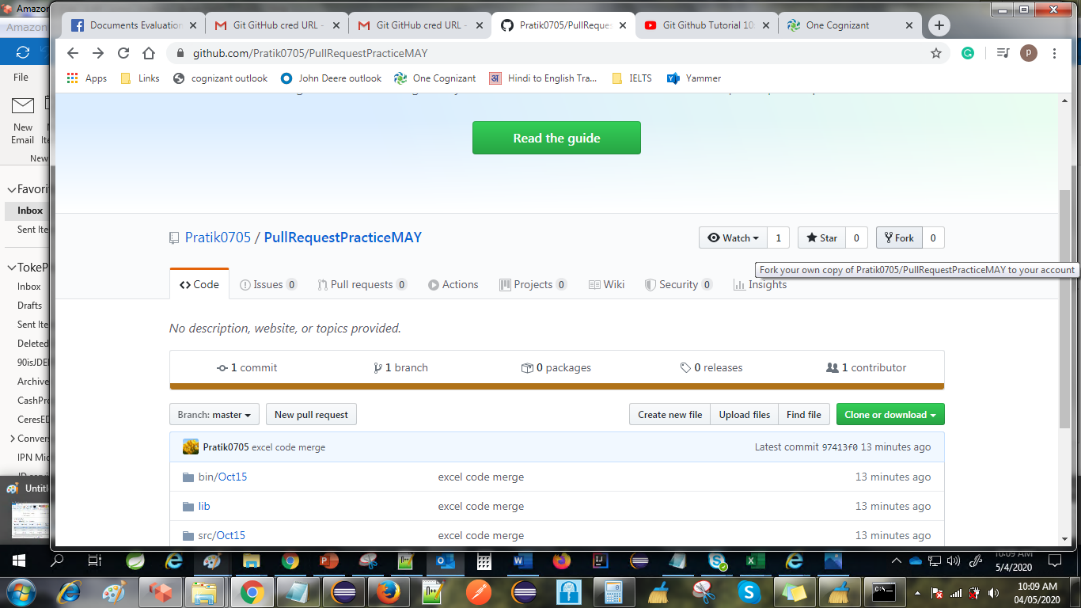
Uname : Pratik0707

email : [pratik.toke91@gmail.com](mailto:pratik.toke07@gmail.com)

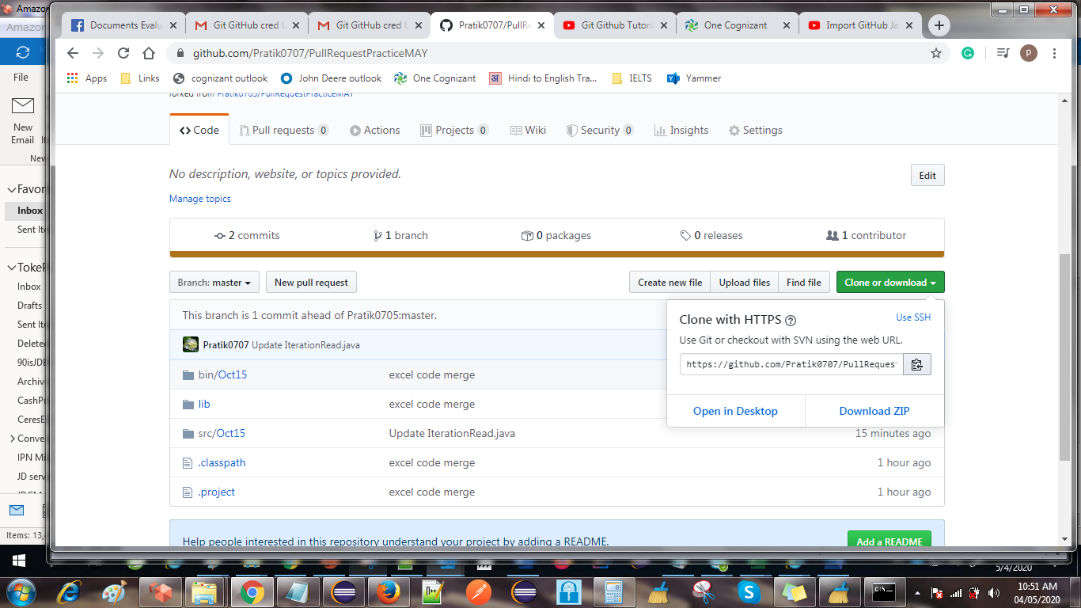
pwd : Admin@12345#\_!

\*\*\*\*\* STEPS OF PULL - MERGE request \*\*\*\*\*

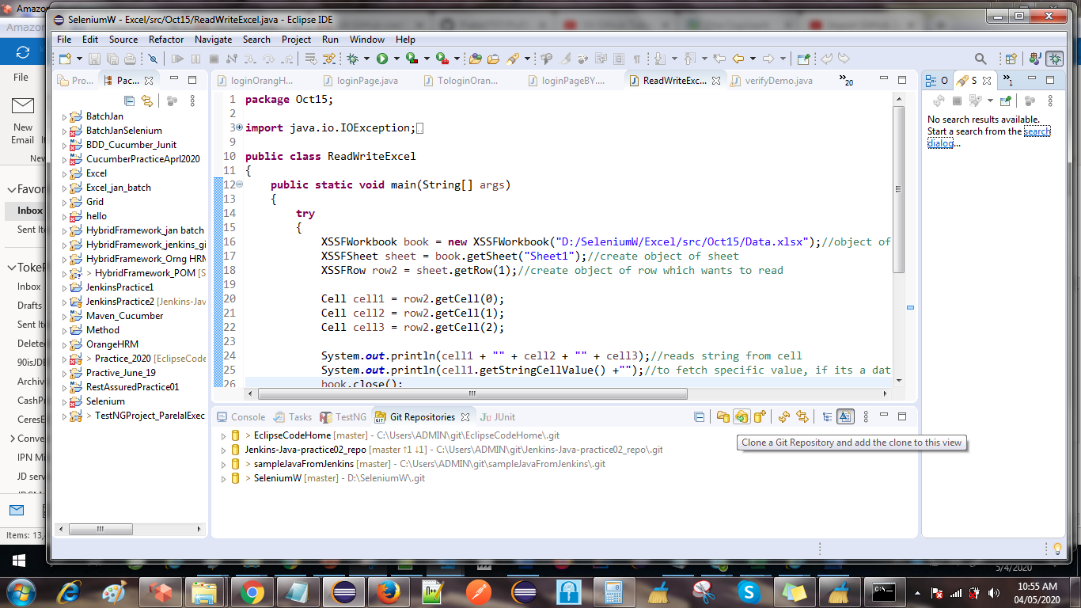
1. User1 has a repository with java code in it and user2 opens the user1s repository. Now, user2 clicks on **FORK**, and then the code from the User1s repository will be copied in the User2s repository.



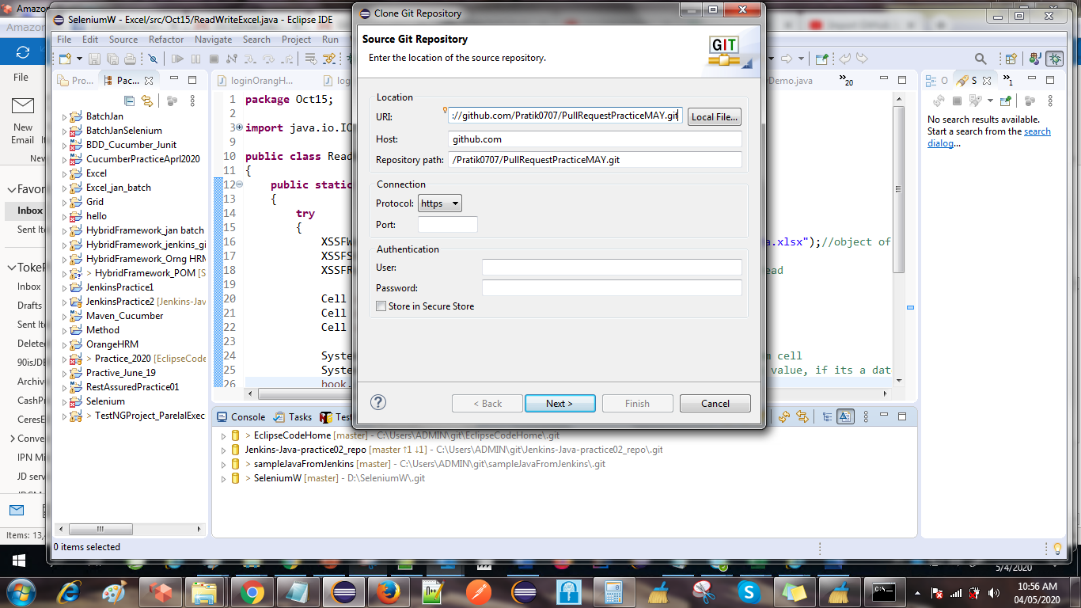
2. Now, user2 gets the code from User1, now User2 will open the project in eclipse and make some changes in the code. Copy the URL as shown below



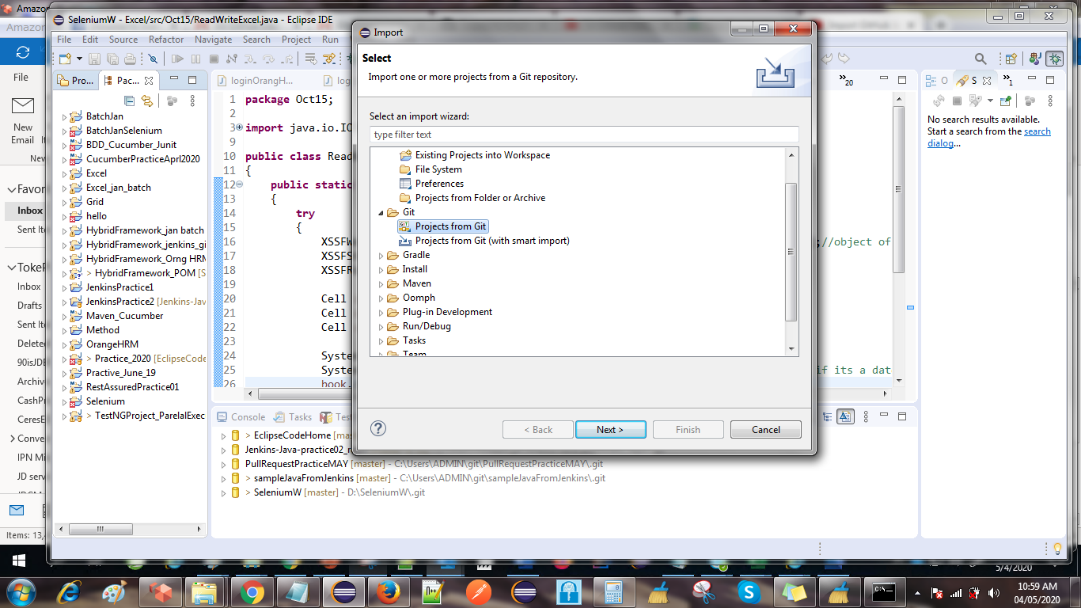
3. click this option in eclipse to clone repository in eclipse



4. pest the clipboard URL. next > next > finish.

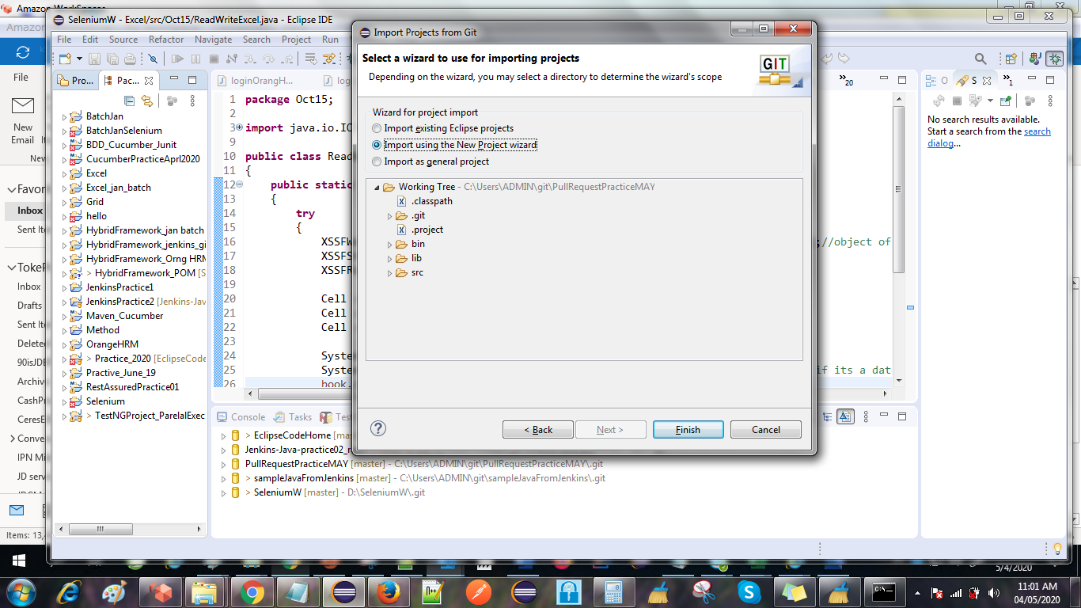


5. right click on eclipse project explorer > select Import  >

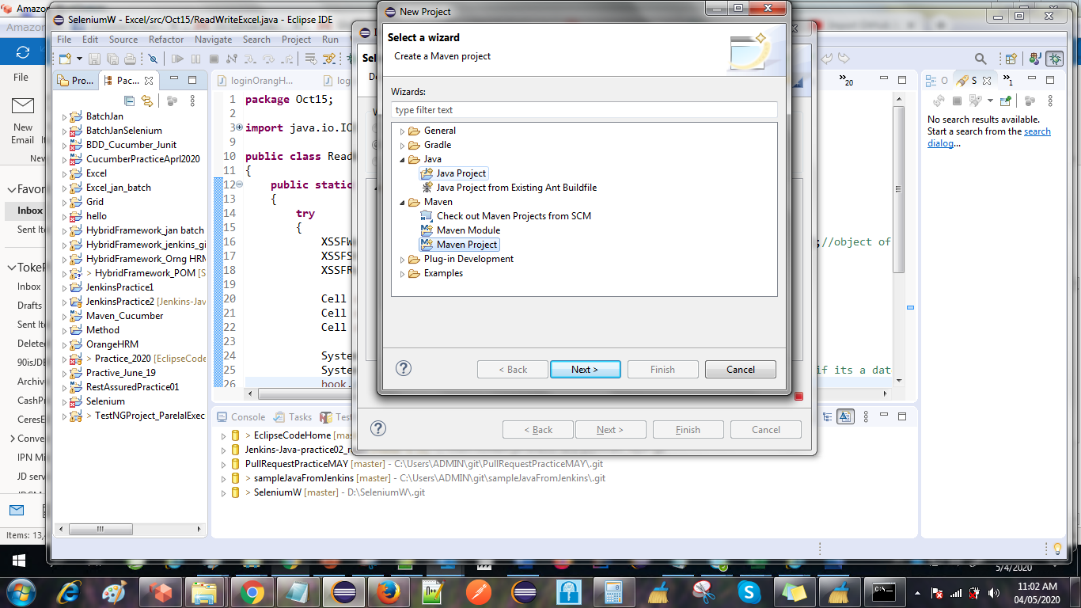


6. click next and select the project (git repo name will be shown here)

7. Select this option



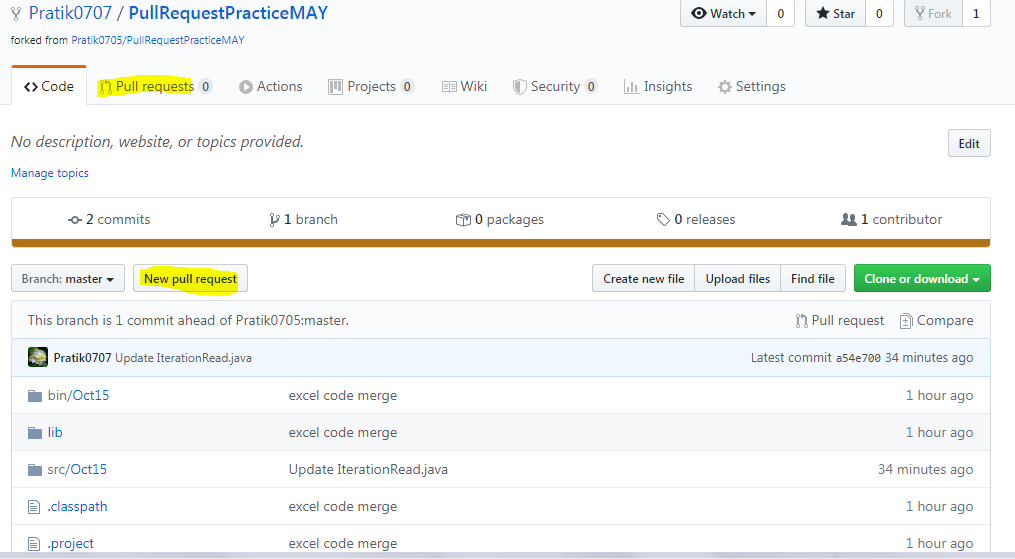
8. click next and select whether its java / maven project.



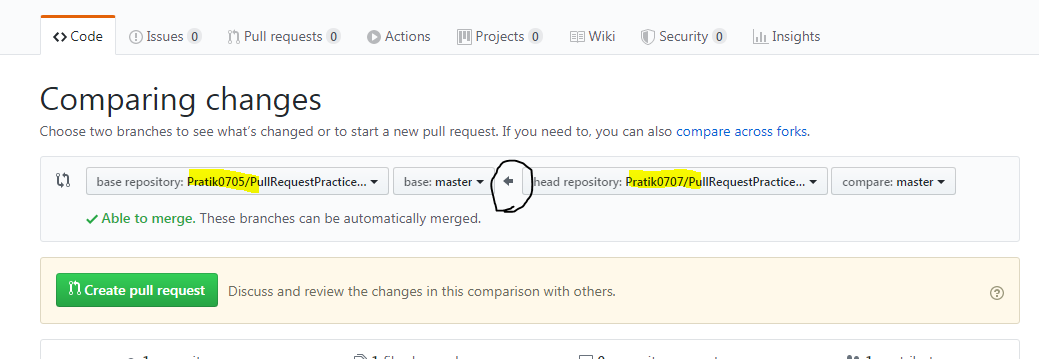
9. Now give the project name, open it, and make some changes, then push those changes back to git so that we can raise a pull / merge request.

10. Now, User2 completed the code change and willing to merge his changes to User1s original code.

11. click on any one pull request button.



12. it also shows from which repo we are merging in which repository. We can also see the code changes on this page ( scroll down). Click on the create pull request button and add some comment.



13. Now the pull request is raised.

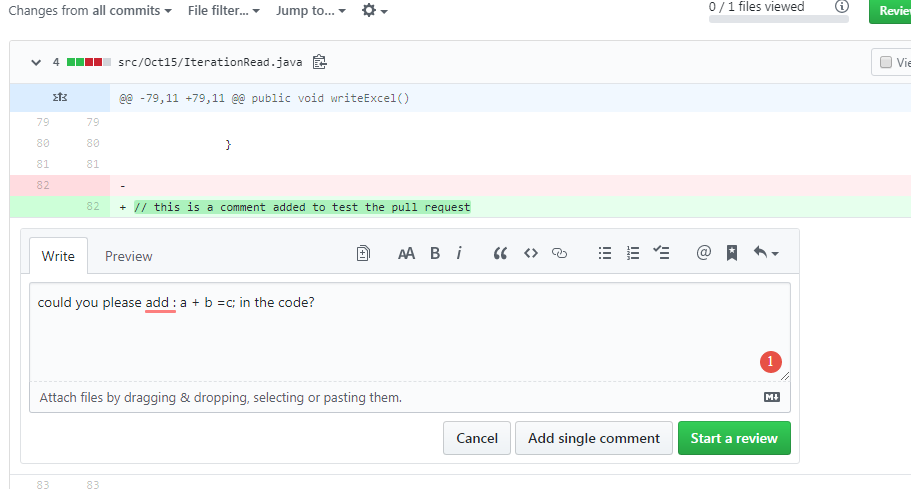
14. Now go to the User1s git account. He will see a notification of a pull request.



15. click on the pull request tab and open that pull request. In this section we can see the code changes as well

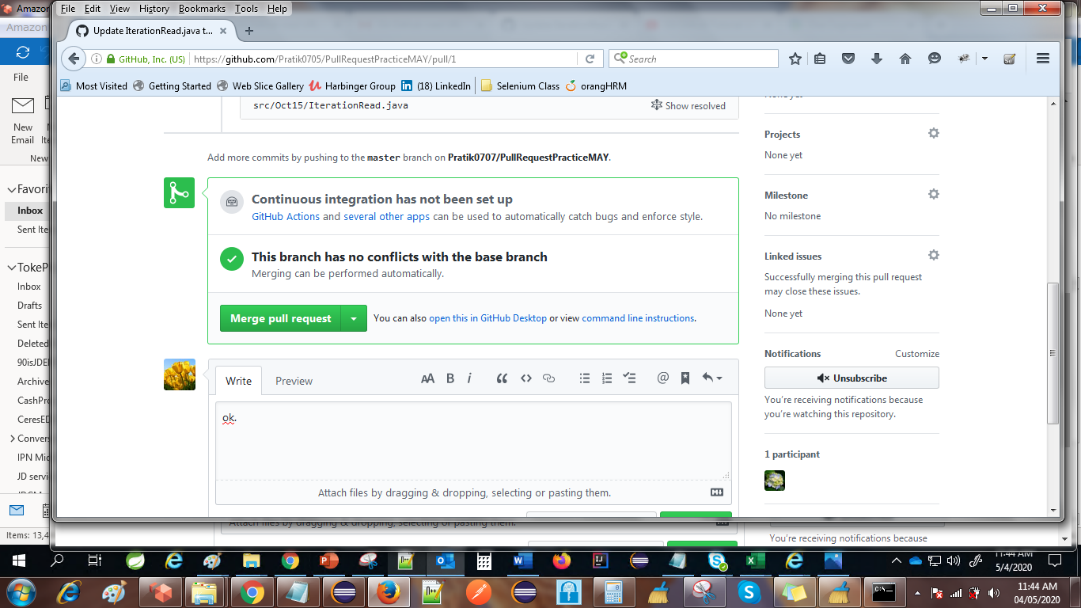


16. as a reviewer I can add a comment (mouse hover on new code lines). and ask for some code change as a reviewer.



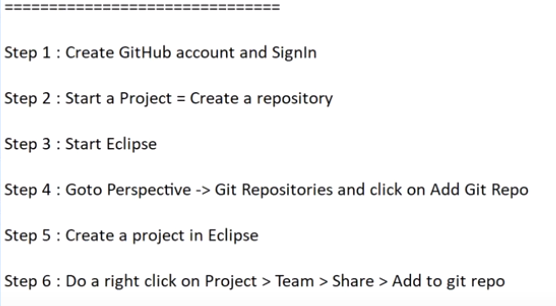
17. Now the user2 can see that in his comment section and he will act on it (edit the code commit.) and comment again.

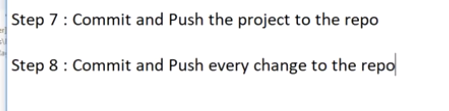
18. Now User1 can verify the code changes and can click on Merge pull request button under pull request > conversation tab.



and now user1 can see the code changes of user2 in his own repository.

Steps to configure eclipse and GIT





C:\Users\ADMIN\git\EclipseCodeHome

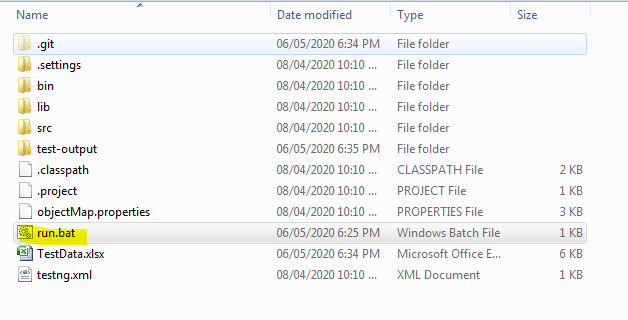
Running a selenium project from Jenkins with the help of BAT file

1. Go to your selenium project and create a .txt file and add below command in the file.

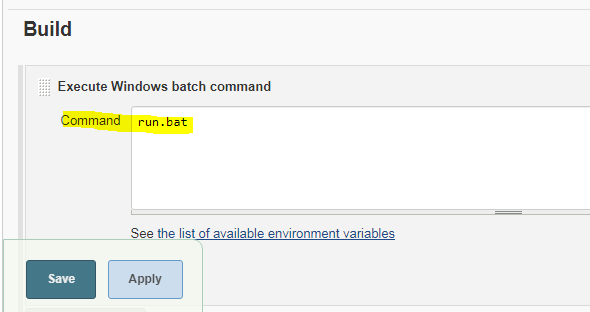
java -cp ".\bin;.\lib\\*" org.testng.TestNG testng.xml

2. saveas that .txt file as : "run.bat" (insert " " in file name while saving )

3. then we get run.bat file created  as below :



4. Now push that project to GIT and create a jenkins job for that repository and add below settings :



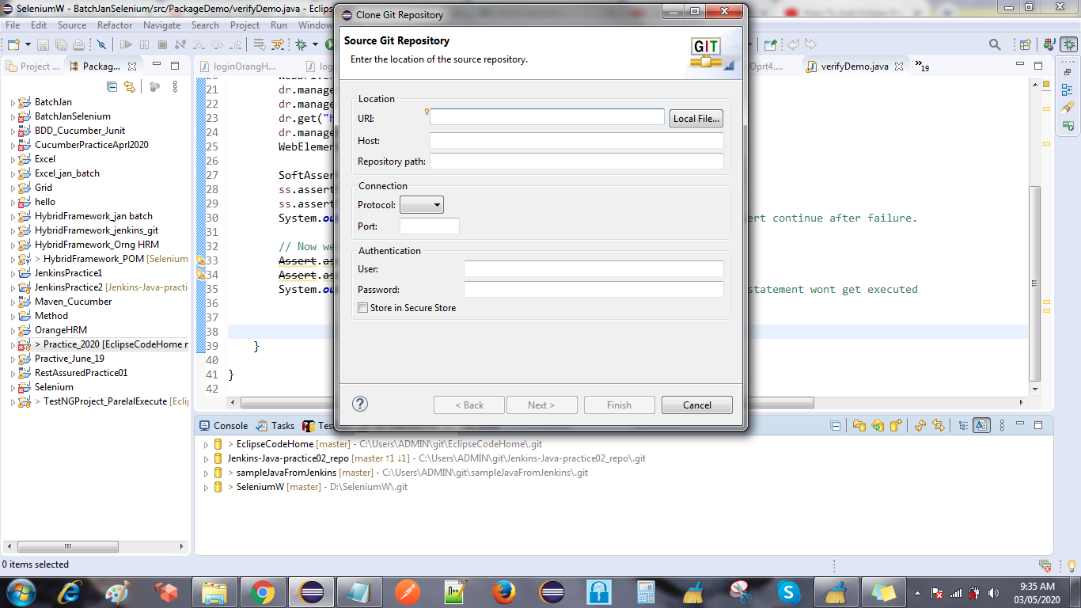
so now onwards the selenium project will be executed by running this bat file. (Instead of creating a bat file we can write this command in the above Command text box. : java -cp ".\bin;.\lib\\*" org.testng.TestNG testng.xml. this also works in the same way. )

NOTE : we can click on this bat file in folder explorer to start the execution of our selenium project.

On Mon, May 4, 2020 at 11:46 AM Pratik Toke <[pratik.toke07@gmail.com](mailto:pratik.toke07@gmail.com)> wrote:

How to operate git from eclipse :

clone your repository :

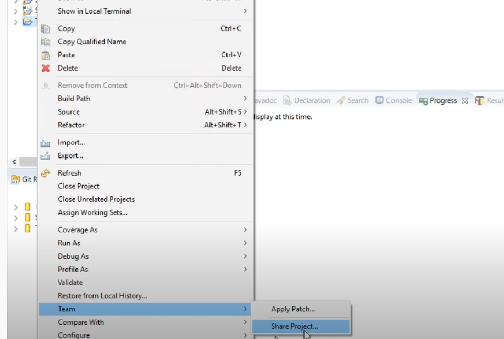


enter details and finish.

Then we can see our repository in eclipse.

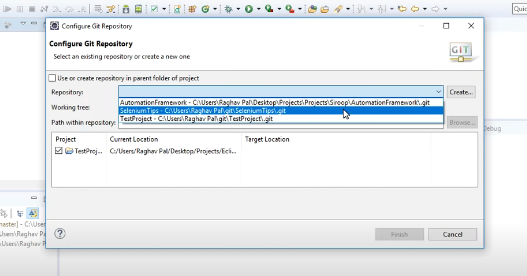
Now create a new project in eclipse which we can add to git from eclipse.

right click on new project and

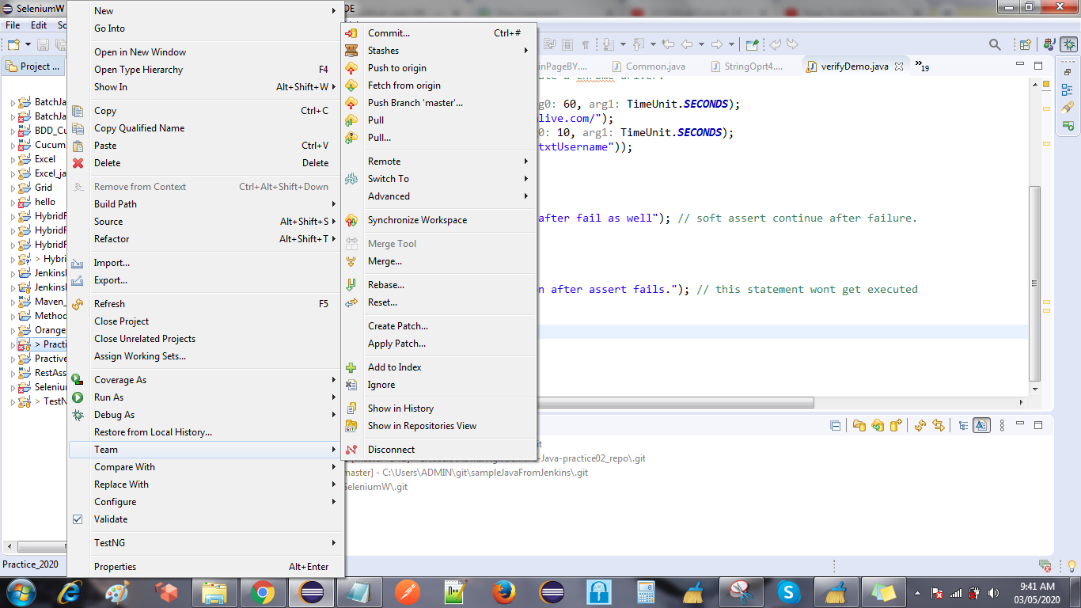




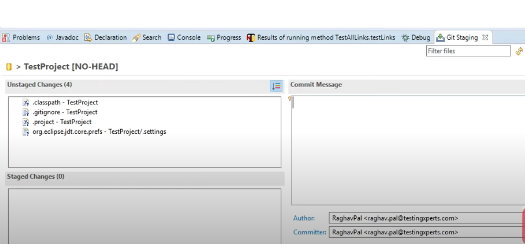
select the added repository



now again right click on the project  > team



now we can see these option. click commit



select all 4 files shown above and drag and drop it in Stage area. click commit and push. enter credentials.