As mentioned in the document, first we must perform the steps mentioned in the pre-requisites.

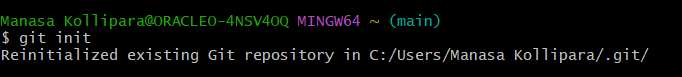
* Next, we can download the Git Bash and GUI from the below site and install it.

<https://git-scm.com/download/win>

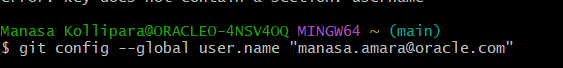
* We can perform many operations using GIT Bash and GUI which are explained in detail as below.

**GIT Commands step-by-step.**

1. **Git Init**: Git init command is used for creating an empty Git repository.  After the git init command is used, a ‘. git’ folder is created in the directory with some subdirectories.



1. **Git Config -** Git config command is used to set and get configuration variables that control various aspects of Git's behaviour. These configurations can be set at different levels, including system-wide, user-specific, or repository-specific.



1. **Git Remote -** Git remote command is used to create, view, and delete connections to other repositories.  The connections here are not like direct links into other repositories, but as bookmarks that serve as convenient names to be used as a reference.

Here we have to provide the link from the VBS or git hub which we have provisioned.





1. **Git Clone -** Git clone is a command for downloading existing source code from a remote repository (like Github, for example). In other words, Git clone basically makes an identical copy of the latest version of a project in a repository and saves it to your computer.



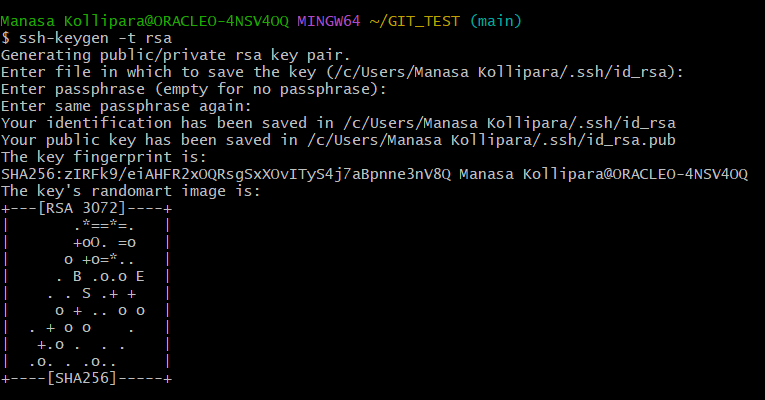
**Git cloning with SSH command and passphrase key:**

Type the command **ssh-keygen -t rsa.**

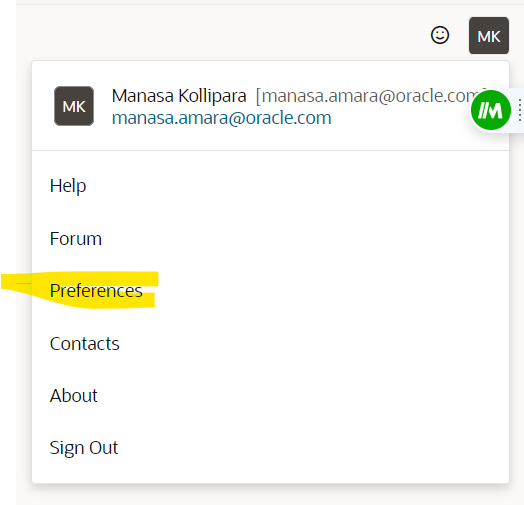
The **-t** flag allows you to specify the key type.

It will prompt us to provide the path of the file to save the key and then it will ask for the passphrase and then confirm passphrase.

Then once done successfully we can see .ssh folder created in the path mentioned and also, we can see the public key in the same path.



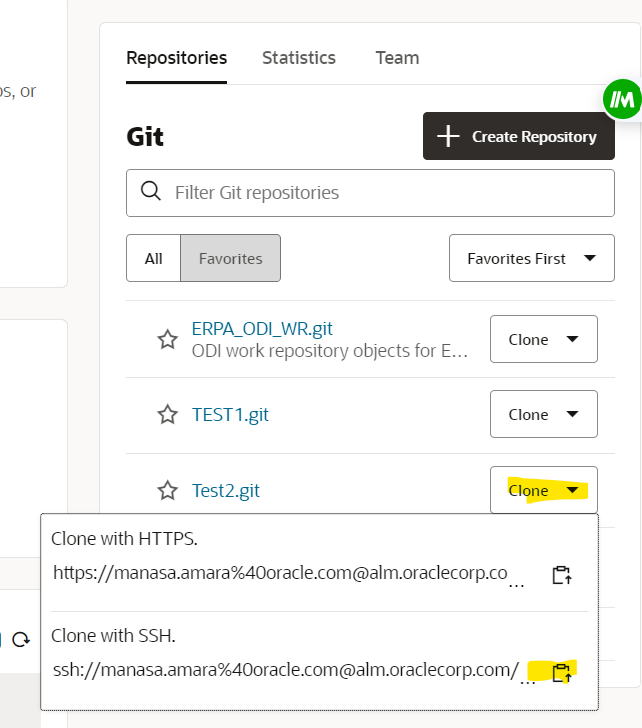
Once we create the ssh we have to go to the location : /c/Users/Manasa Kollipara/.ssh/id\_rsa.pub and copy the details present in **id\_rsa.pub** key and place it in the ALM VBS in the below path.





We can add a SSH key as per the above screenshot. Then we can clone any of the repository present in the VBS by using clone command.

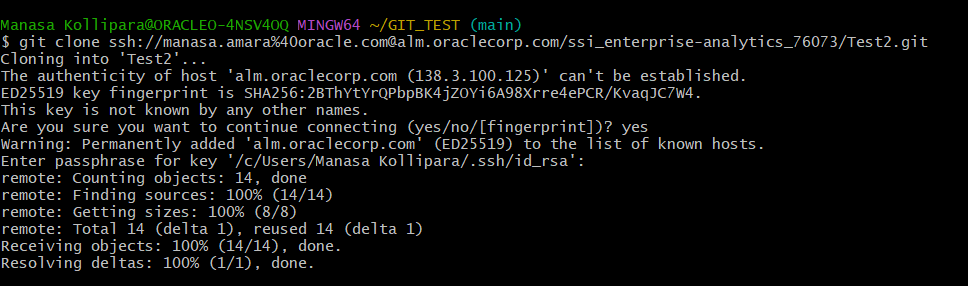
We can see the SSH link of repositories in the below location.



Now we can clone in the git bash as below:

Git clone <provide the copies SSH link>

Then it asks for the passphrase key which we have added earlier. Then we will be able to connect to the repository.



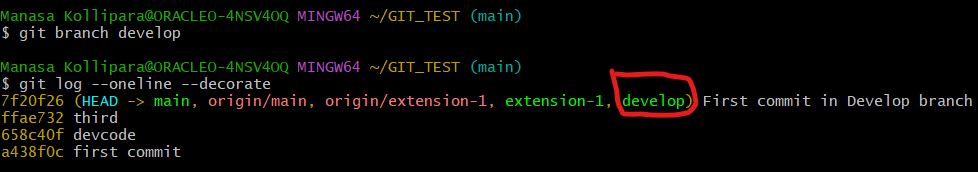
1. Git Branching: First we must create a branch with below command.

**Git branch develop.**

This will create a new branch: **develop** in local



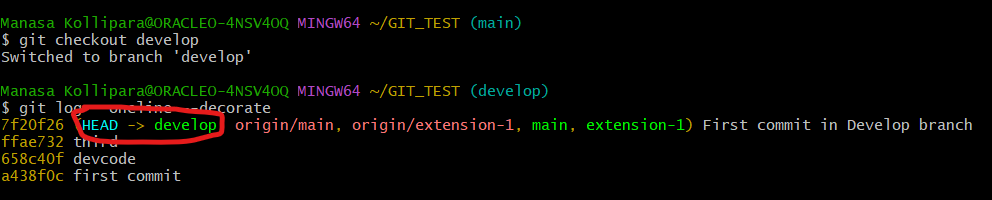
1. We can see the newly created branch when we check the log



We can see that as Head -> main, main is the actual branch and we have created a new branch develop.

1. Now in order to enter into the newly created branch: develop

**Git checkout develop**



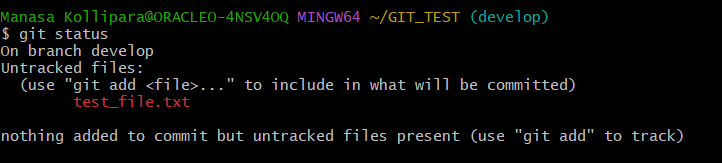
Now we can see the Head -> develop, Head is pointing to develop branch and so develop is our current branch.

1. Now we create a test file in develop branch as below:



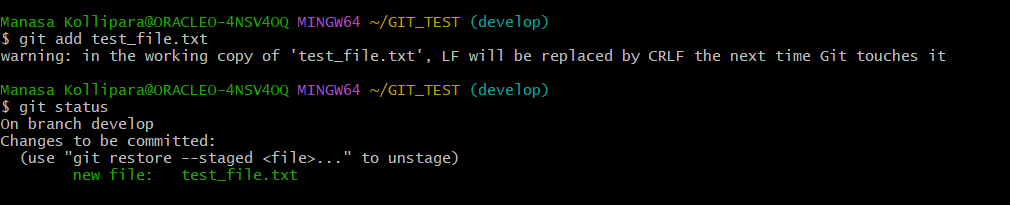
Once we create the test file, we can see the git status

**Git status**

****

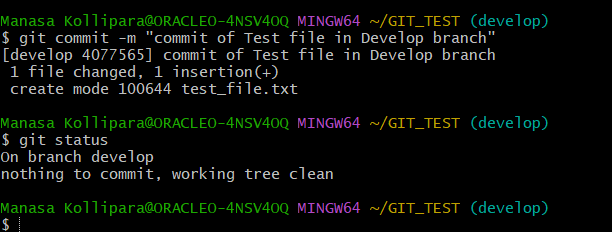
As we can see that this file has been created but it is not yet added or committed to the branch. So the next step is to add this file to the branch

**Git add test\_file.txt**

****

1. Now we have to commit the changes.

**Git commit -m “commit of Test file in Develop branch”**

****

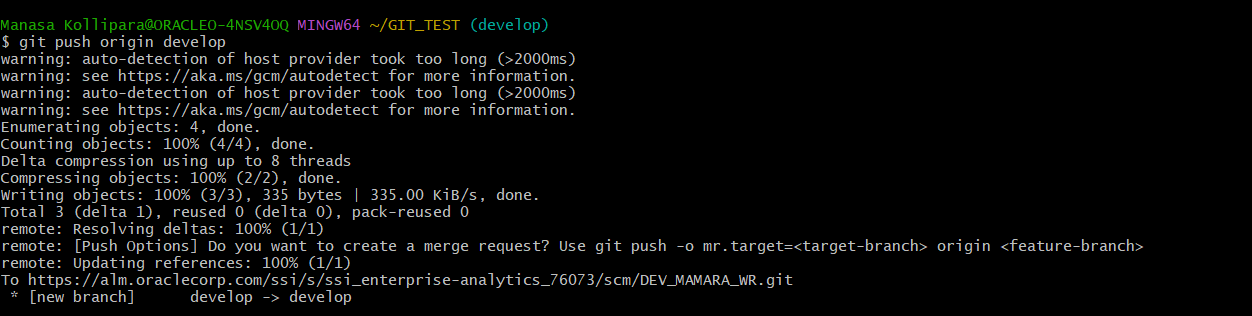
after committing it we can see the status as clean.

When we check the VBS, currently we have one main branch(main) and one extension with name (extension-1). As we did not push the newly developed branch ‘develop’ into the VBS, it is not yet visible.

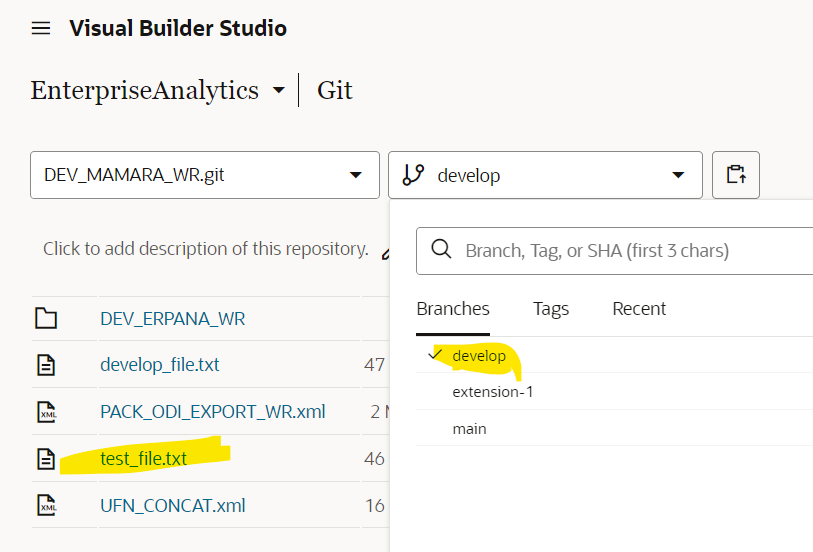


1. Till now all these changes are done in the local, now we must push these changes to the VBS (git hub). We can do this using a git push command.

**Git push origin develop**

****

1. Now if we check the VBS we should be able to see the newly created branch: develop which will contain all files present in main branch along with test\_file.txt which was created as part of the develop branch.



1. Now if we check the main branch, we will observe that test\_file will not be present in the main branch as we have not yet merged the changes.

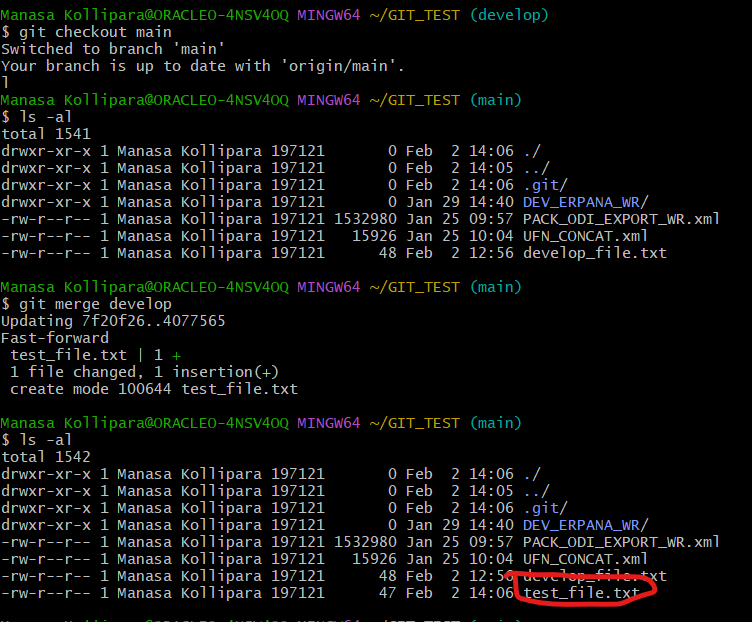


1. In order to merge a branch, we have to make sure that we are in the main branch: main and then do the merge to the main branch. So, to switch to the main branch,

**Git checkout main**

And then merge with the branch command as below:

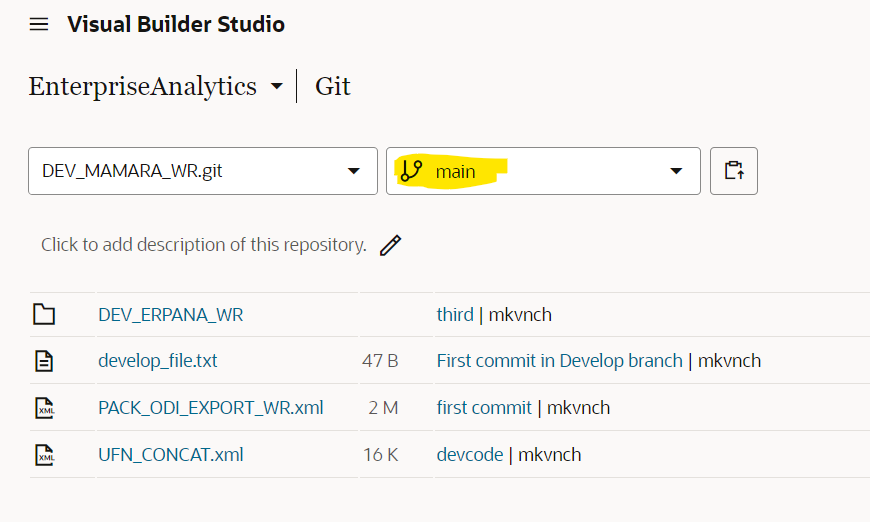
**Git merge develop**

****

If you observe, before merge, main does not have the test\_file and after merging the main with develop branch, we can see the test\_file.txt as per the above screenshot.

Now for the changes to be visible in the VBS, we must again push the main branch.

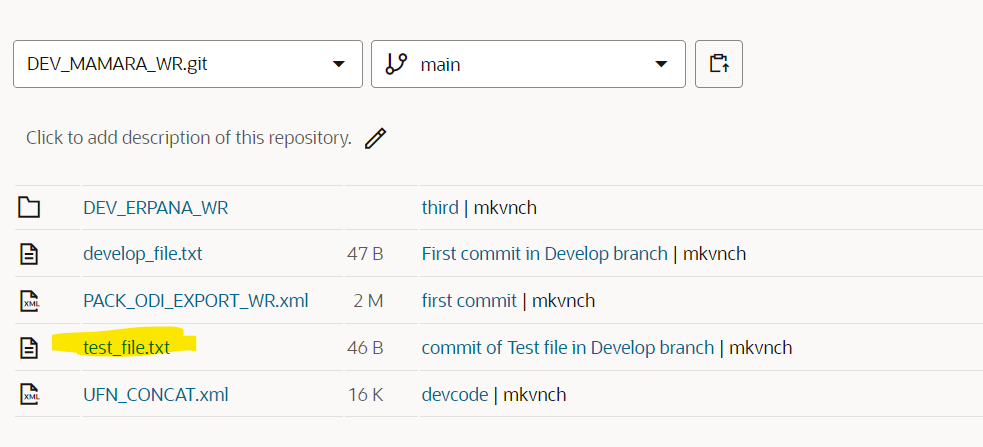
VBS before pushing:



**Git push origin main**

****

Now when we refresh the VBS, we can see that the new file is present in the main branch as well.



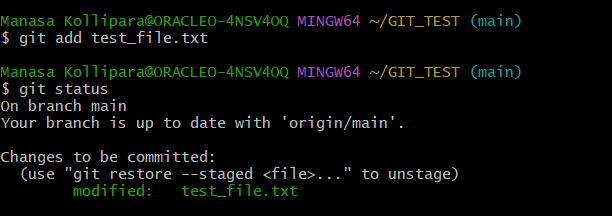
**GIT Restore commands:**

Suppose we have a scenario to modify the existing file, then we can modify the file and then add the file.

Modified the test\_file and when we check the git status then we can see as below.

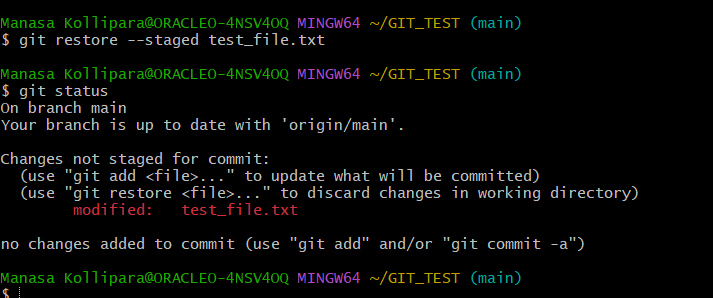


We have to add the file using: git add test\_file.txt



So now the file is in stage state. At this state, if we want to restore the changes done in the file then we can use restore command.

Git restore –staged test\_file.txt



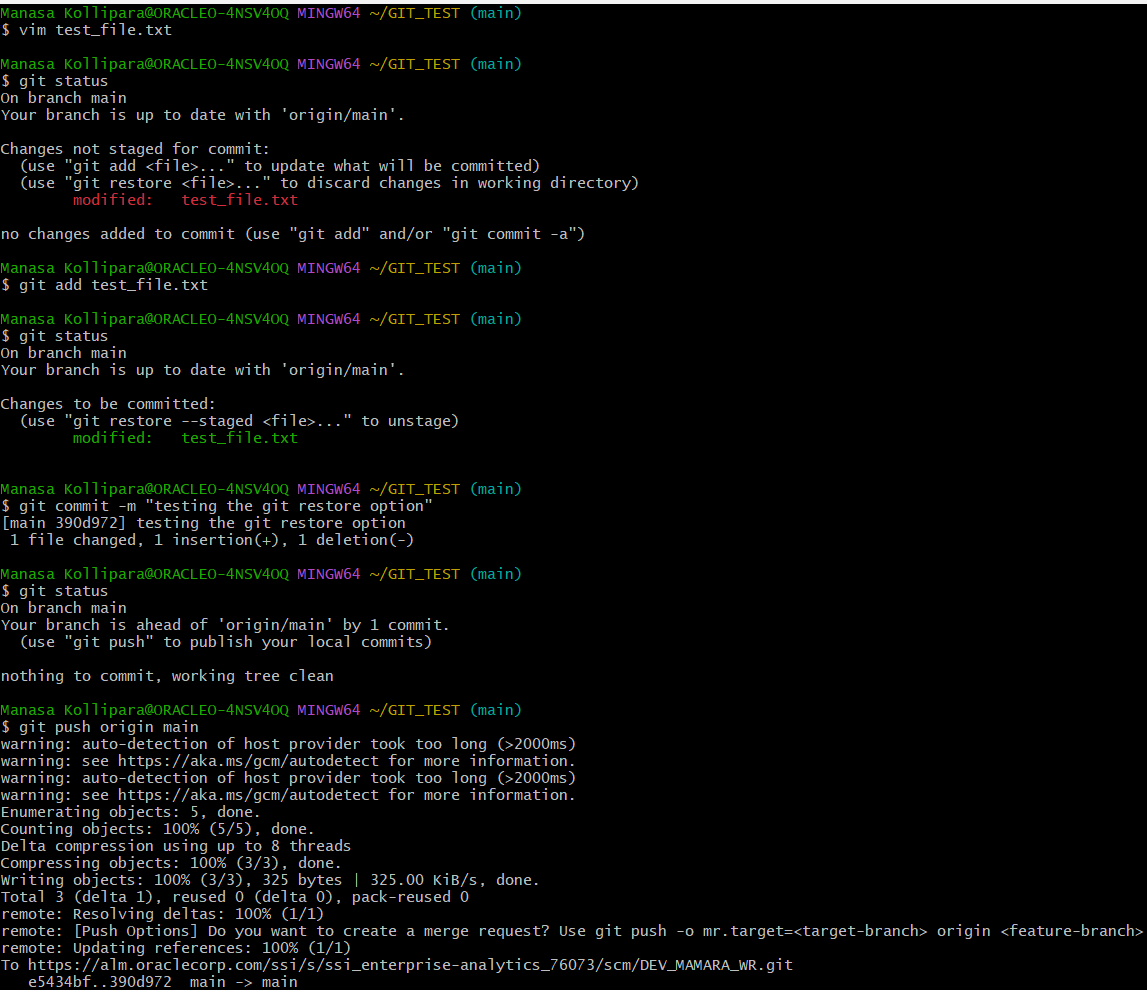
So now we can observe that the file is moved from stage to change not saved state.

If we further want to restore the file, then we can further restore it as below.

**Git restore test\_file.txt**



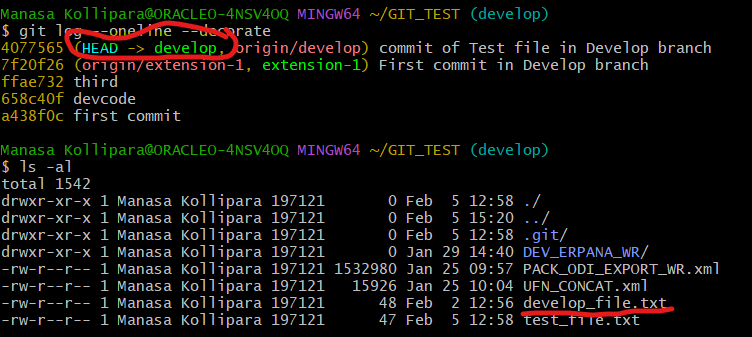
So now we can modify the file and then we can add the file, move to stage, commit the file, and then push the file to the main origin as per the below commands.



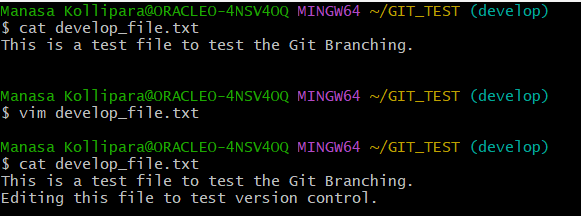
**Action Items:**

**Change existing code in a new branch and merge. See if version history is visible in main branch:**

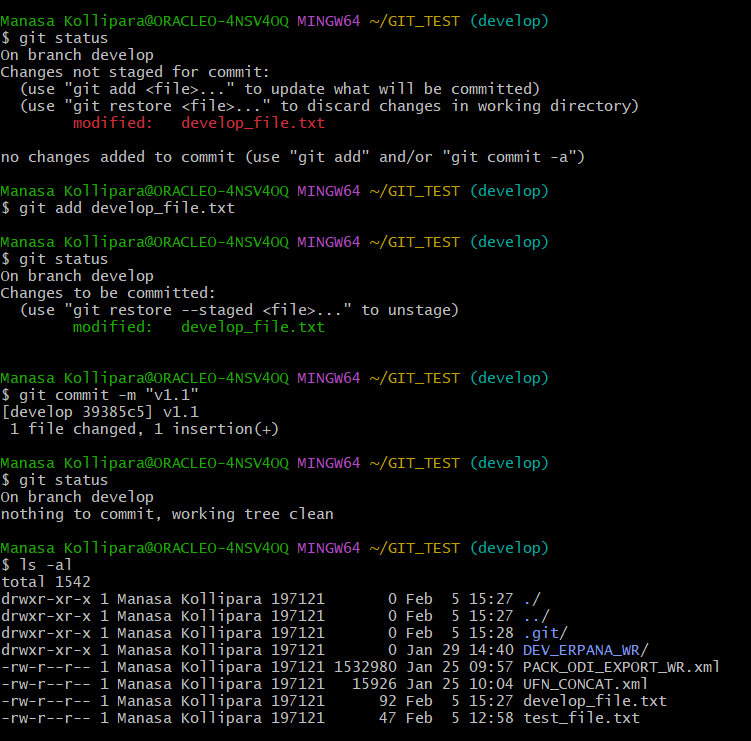
First, we enter one of the branches which is **develop.**  Now we are going to edit one of the existing files ‘develop\_file.txt’.

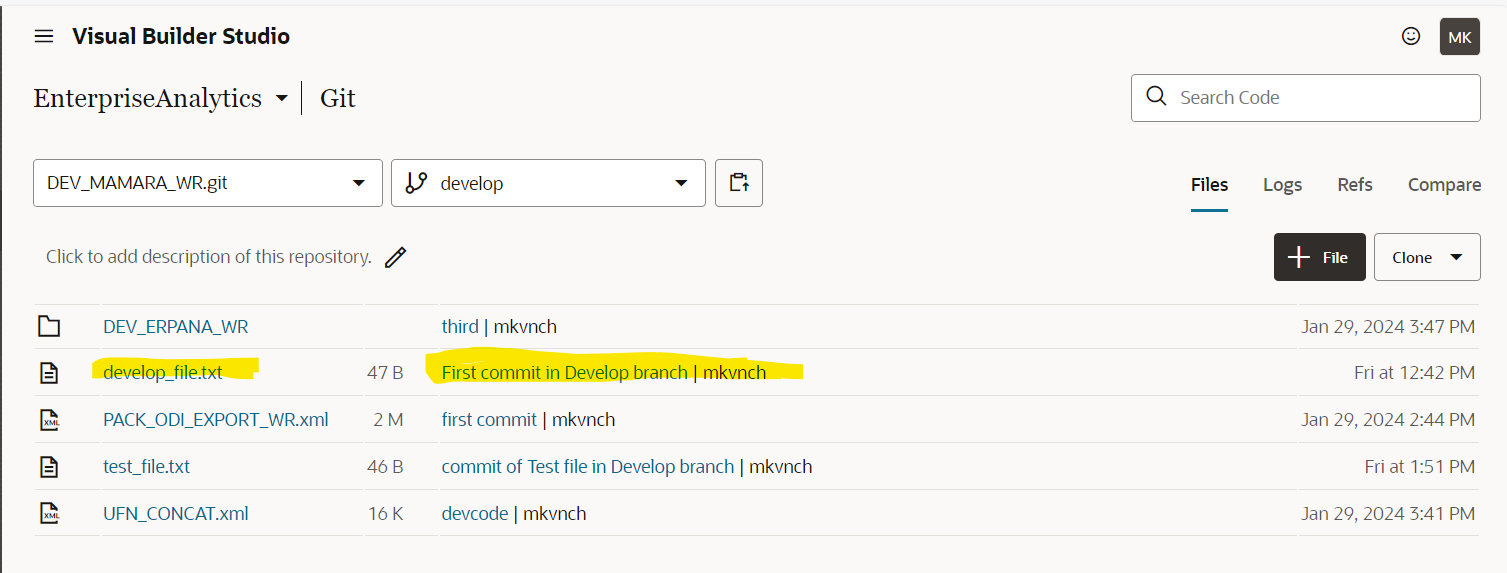


Below screen we can see the existing data in the file and then we have edited the file.

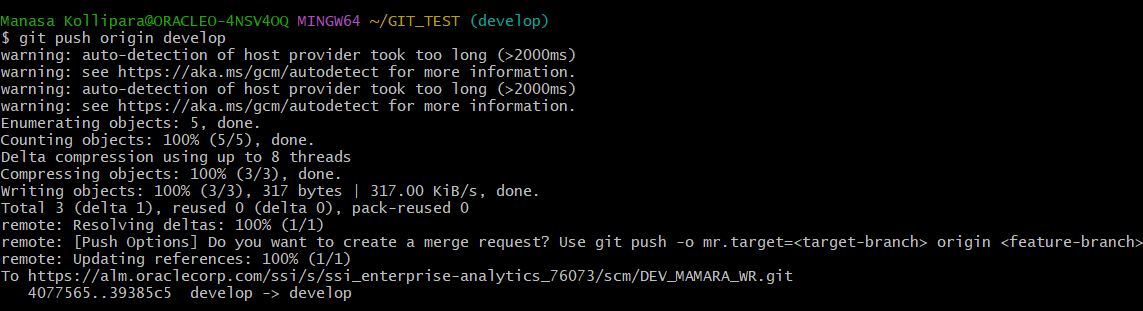


Now we have to add and commit the file by using the below steps.





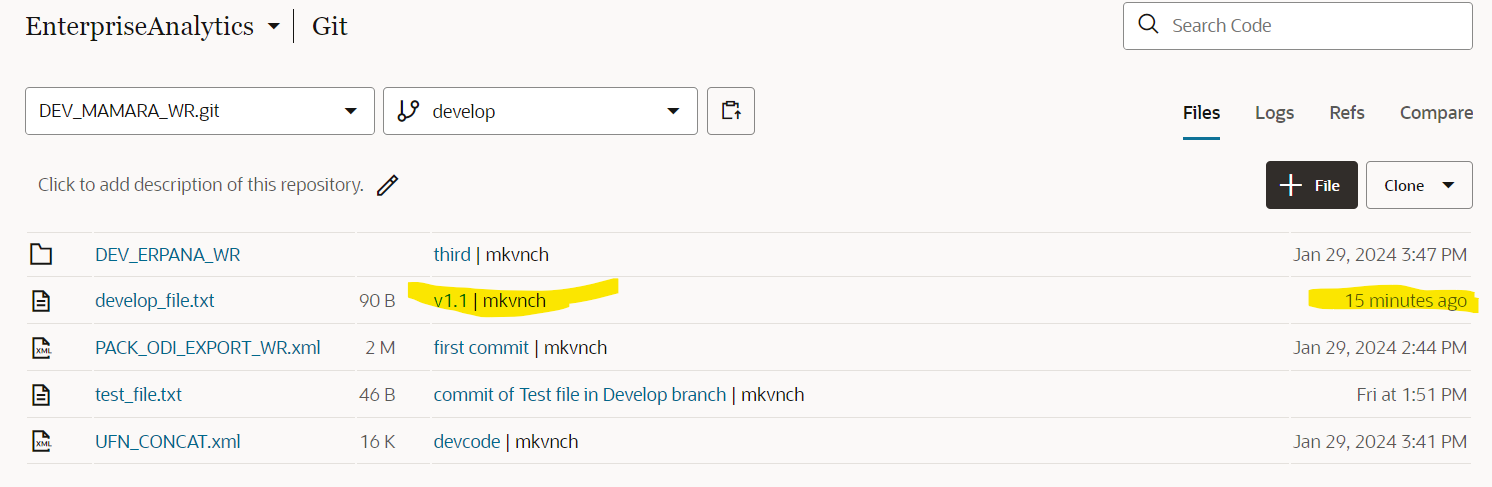
We observe that the changes are not applied yet to VBS. Now we must push the changes to VBS by using below commands.



We can see that develop\_file.txt data is updated as below.



We can observe that in VBS new version is updated.



We have modified the existing file in the develop branch and version has been updated in the develop branch, but these changes are not merged in the main branch. So now we will merge these changes in the main branch.

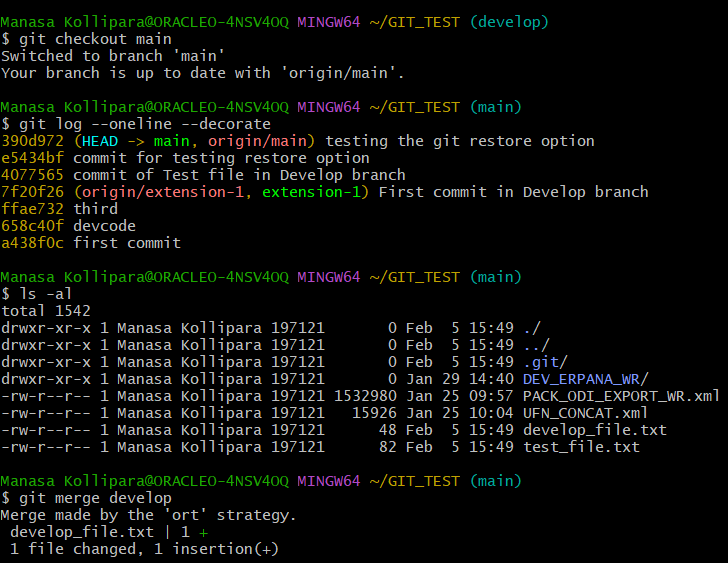
First, we have to move to the main branch by using checkout command.

In order to merge a branch , we have to make sure that we are in the main branch :main and then do the merge to the main branch. So, in order to switch to the main branch,

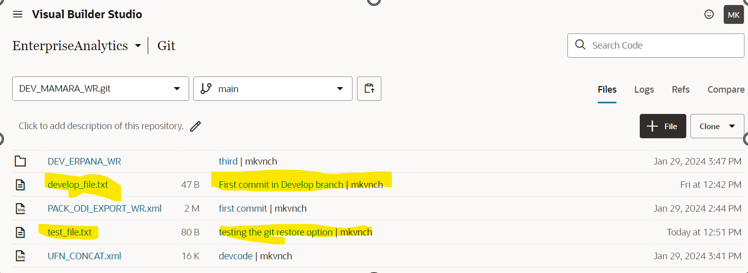
**Git checkout main**

And then merge with the branch command as below:

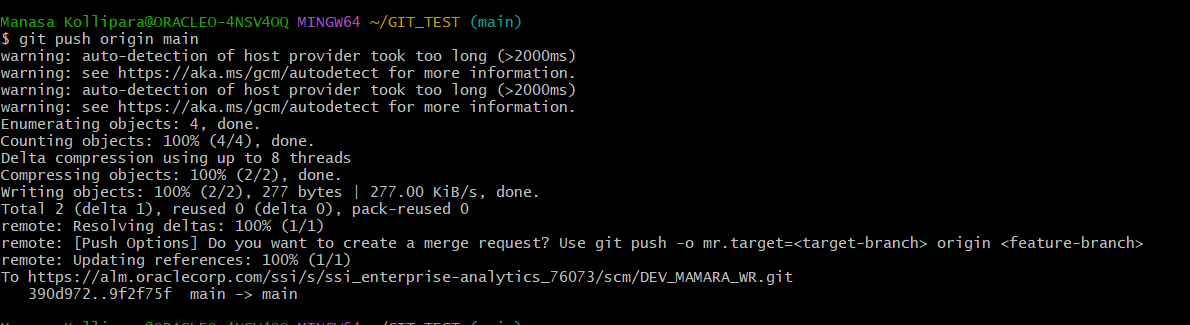
**Git merge develop**

****

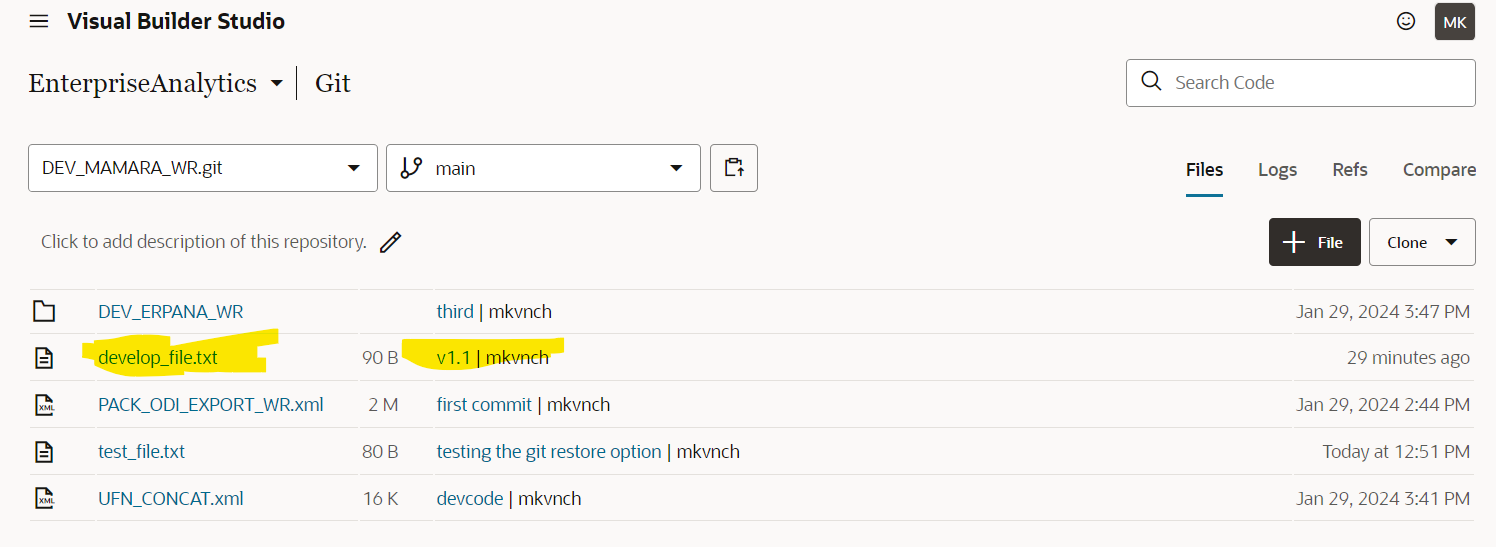
In VBS, we can see that updated version is not yet moved to the main branch. We must push the changes to the main branch.



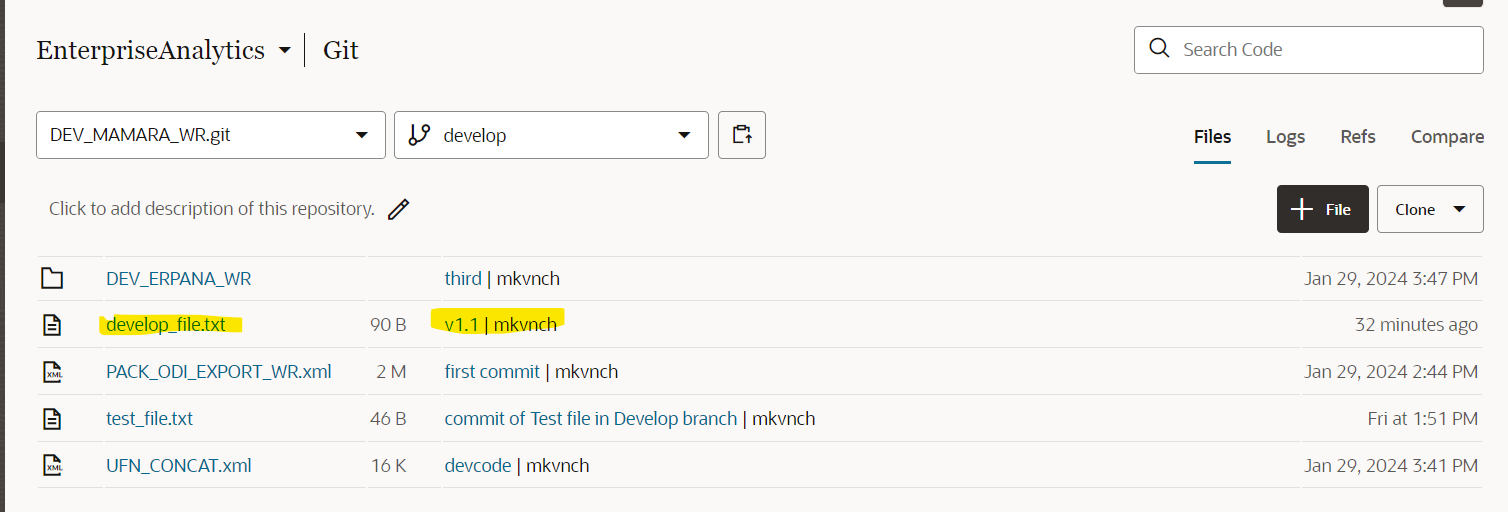
Command to push the changes to main branch.



Once the changes are pushed to main branch, and if we refresh the VBS we can observe the latest version is merged in VBS.

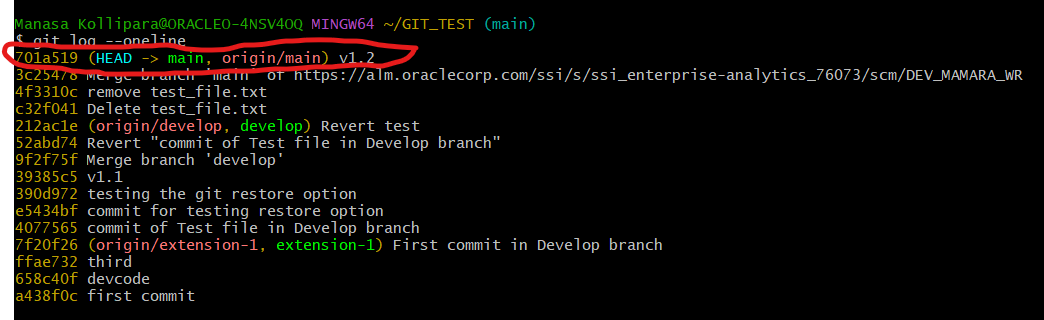


So now develop and main branch are in sync. We can check the develop branch as below.

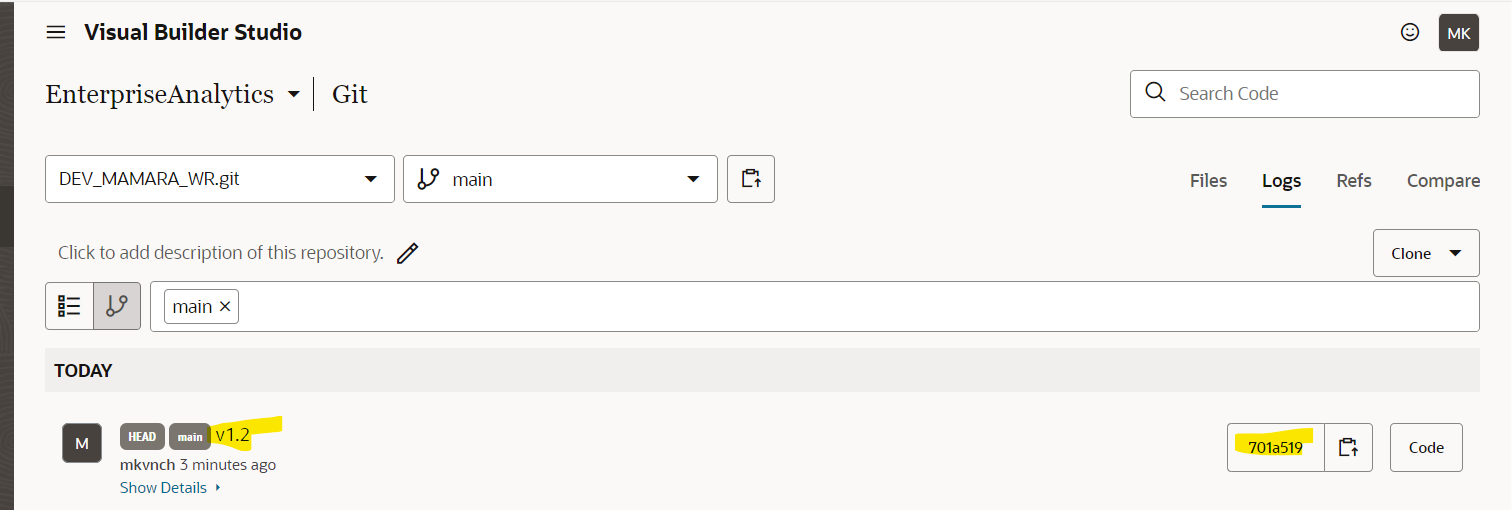


**Feasibility of version rollback after committing to main branch**

we can roll back the version even after committing to the main branch by using the git revert option. For this to work we need the ‘revision id’ which we can get from the log.

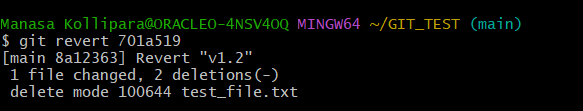


So here now we have recently committed the new version which is v1.2 and this changes have been pushed to the main branch as well

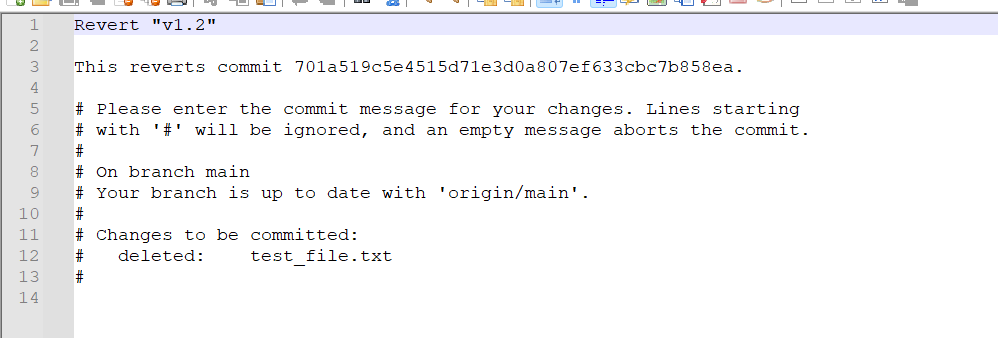


Now we can revert it back to previous commit by using git revert option

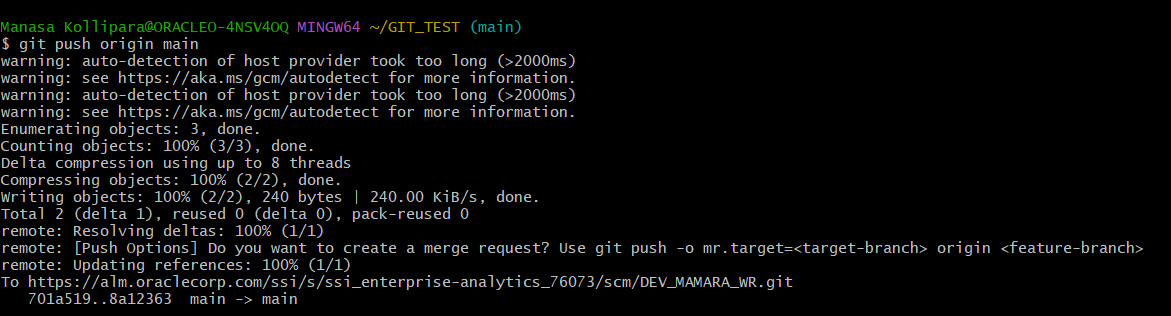
Git revert revision id(701a519)

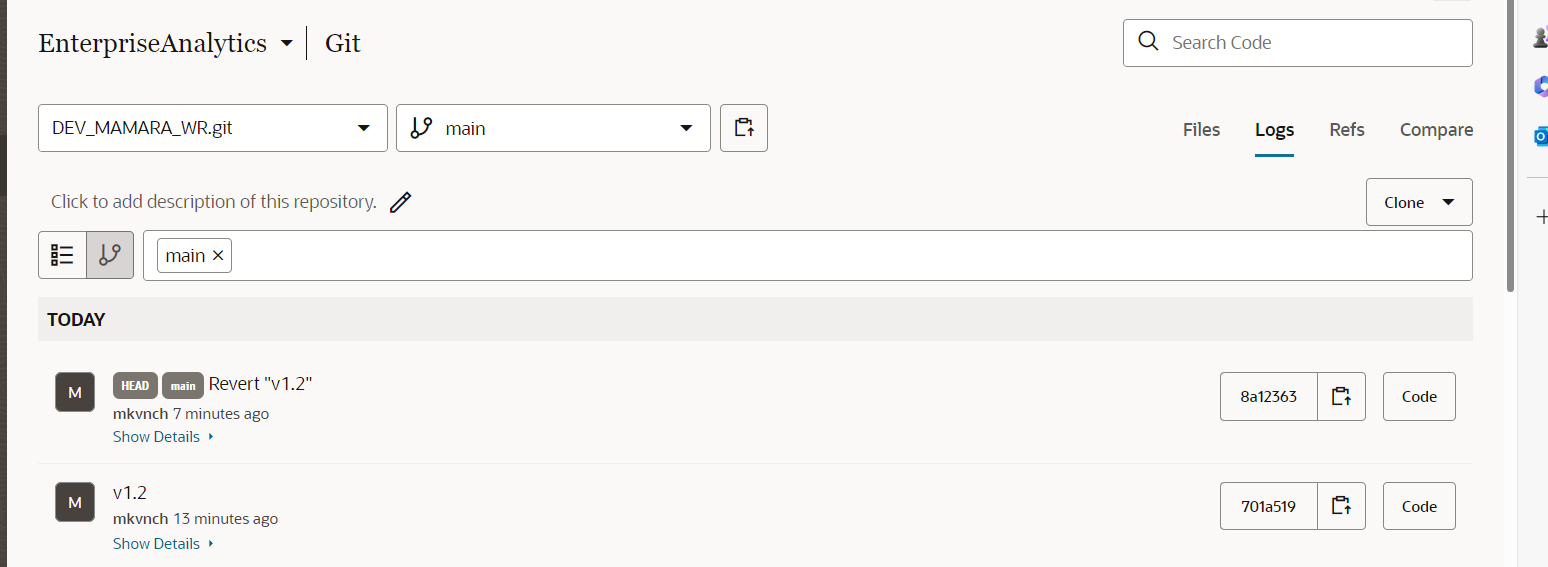


This will automatically move to the editor screen as below:

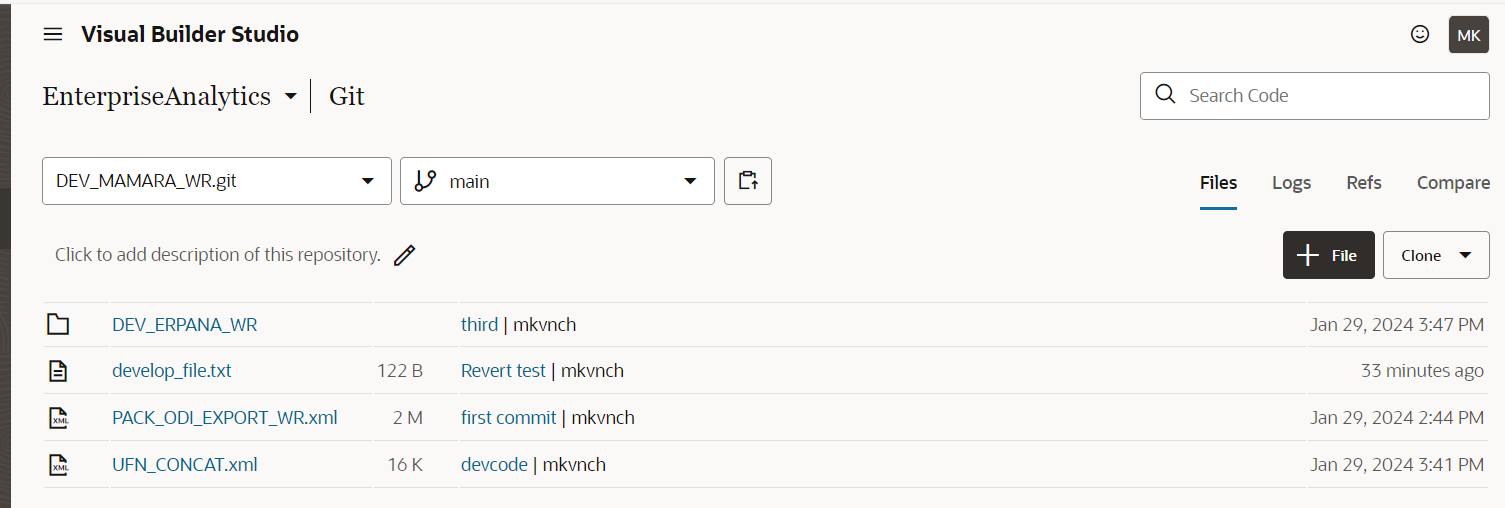


We must push these changes to git hub using git push

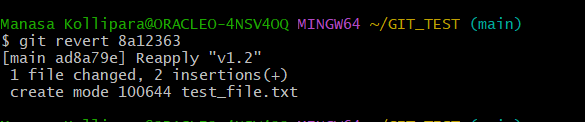




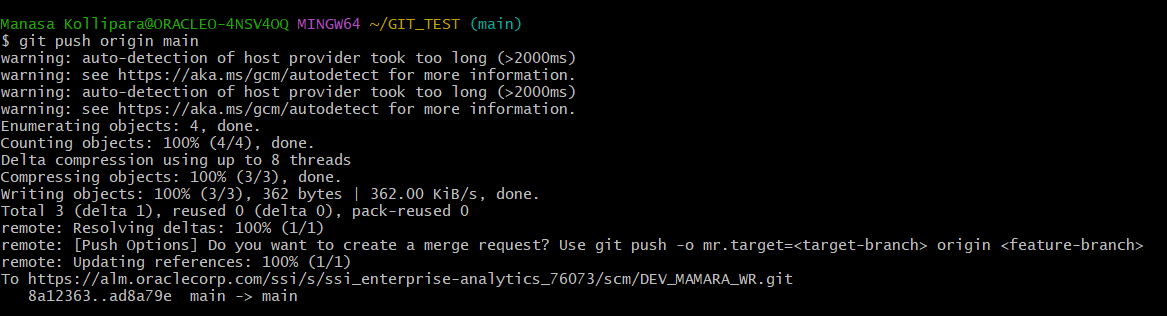
Now as we have reverted the changes, we can see that test\_file is not present in main branch.



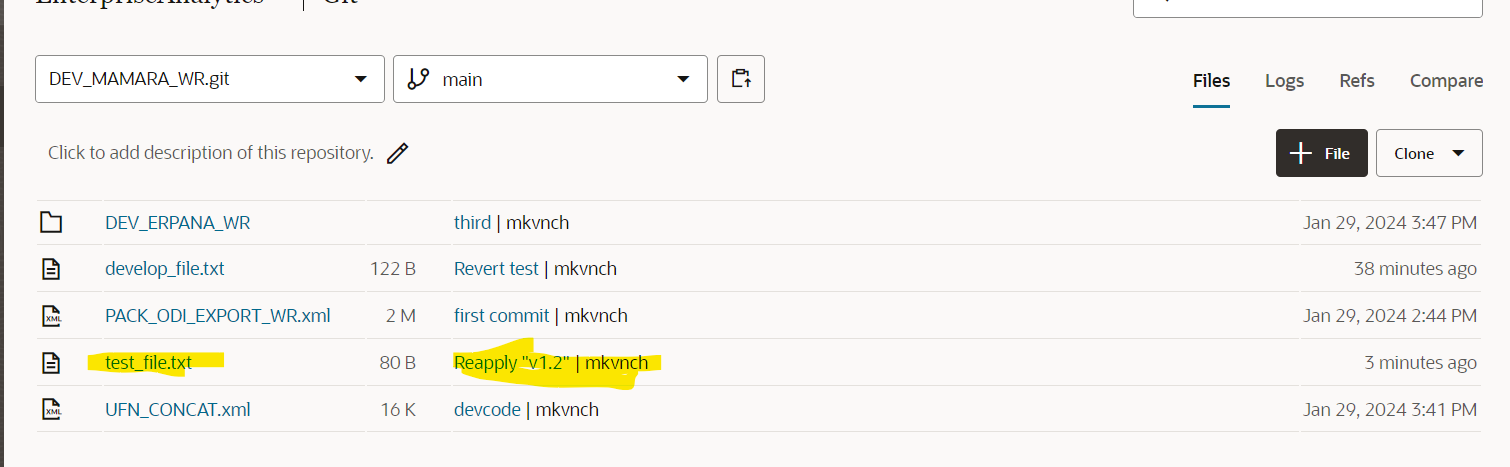
Now if we need the test\_file again, we can again revert the changes as below.



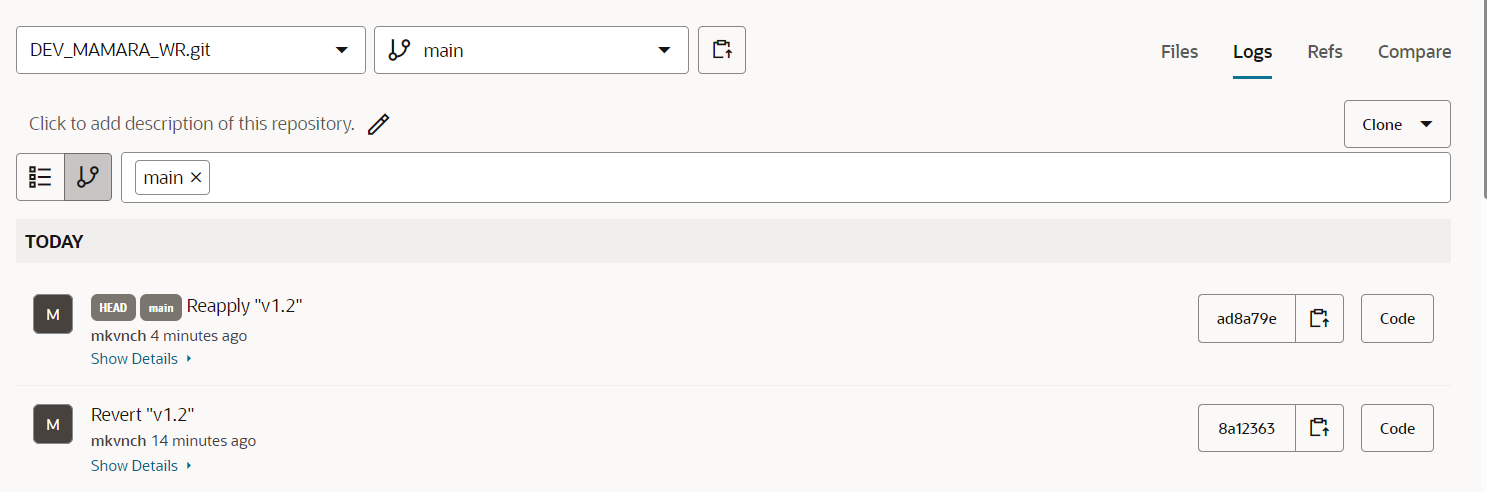




Now we can check in VBS if test\_file is available.



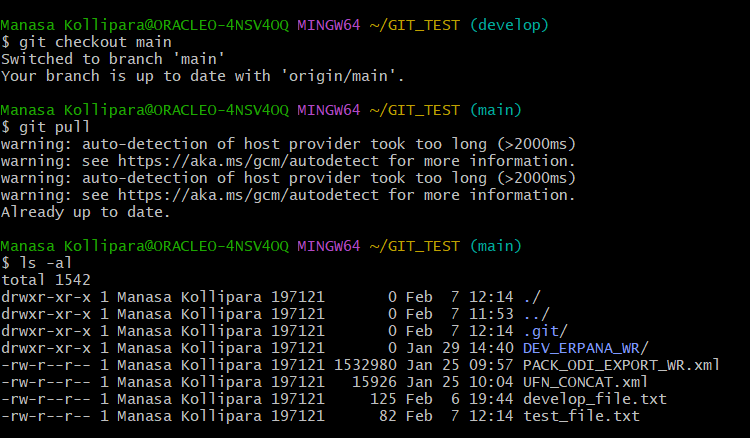
Logs screenshot.



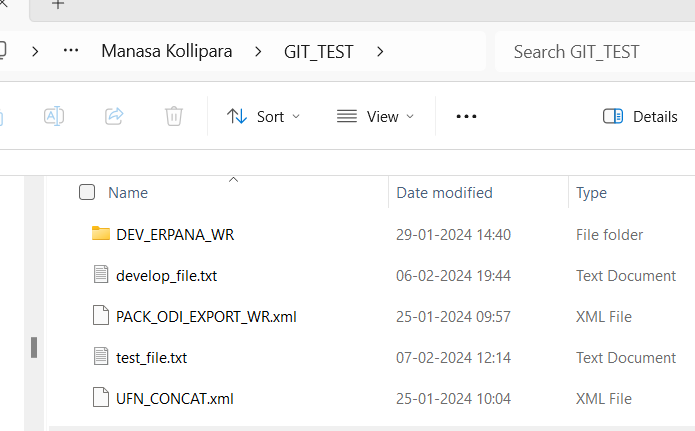
**Feasibility of file/folder deletion:**

**File Deletion from a Branch:**

Before deletion from the main branch, we can go a git pull so that we have same files into our local.

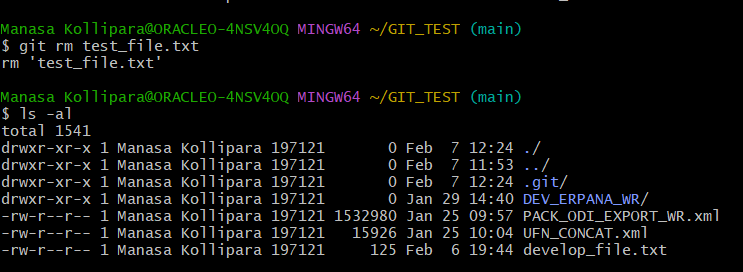
****

After using git pull, we can see that we have same data in our local.

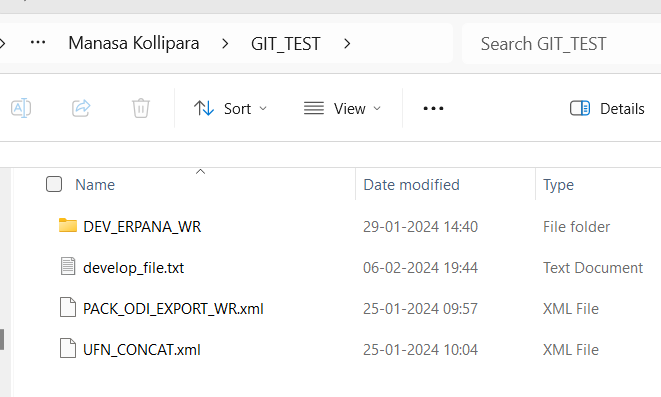


Now we can try to delete the file, test\_file.txt using git rm command.

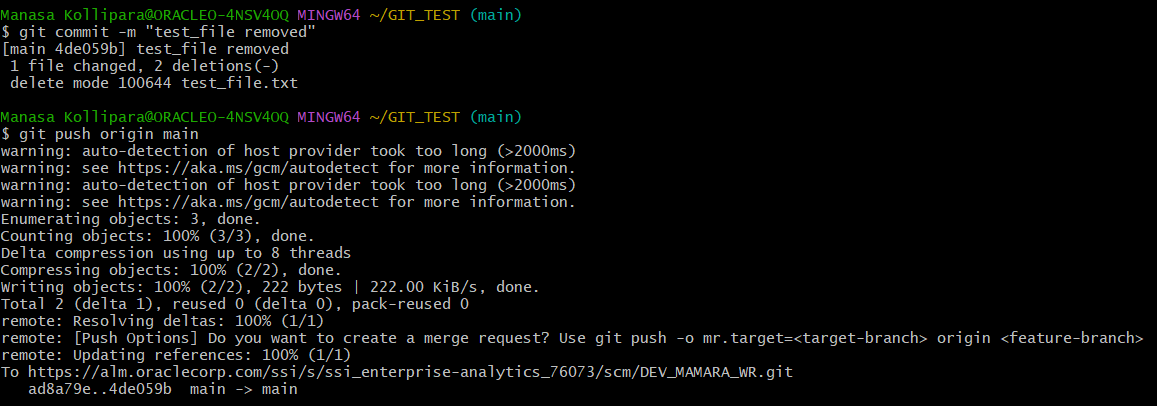
Git rm test\_file.txt

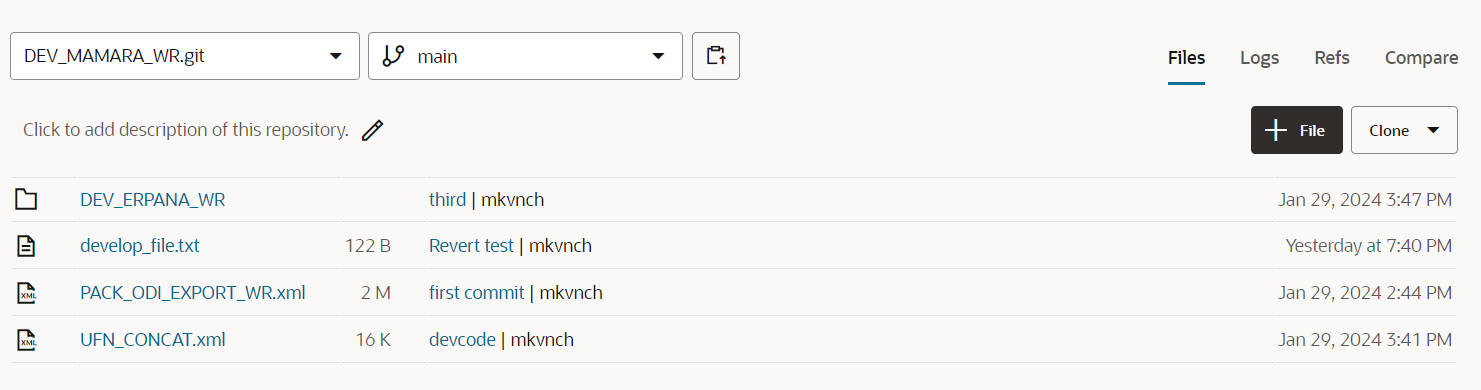


We observe that it is removed from local as well



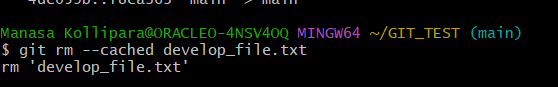
We must commit and push these changes to VBS using below commands



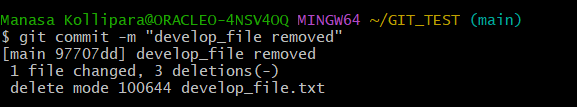


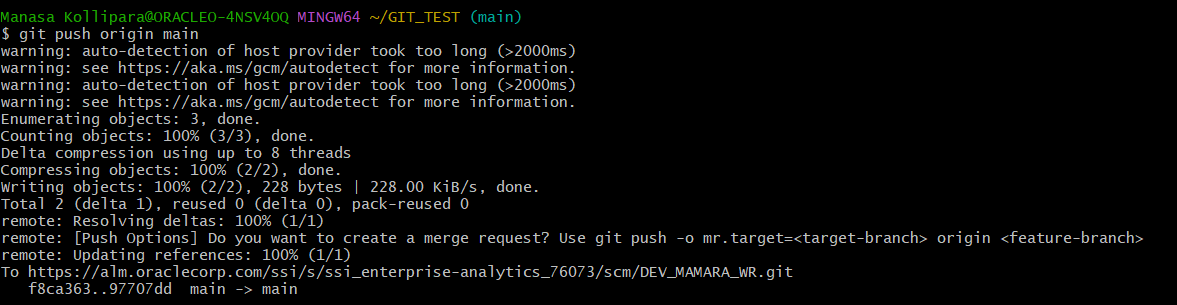
Suppose we want to delete the file in the branch, but we want to keep it in our local, then we can use below git command.

Git rm –cached develop\_file.txt

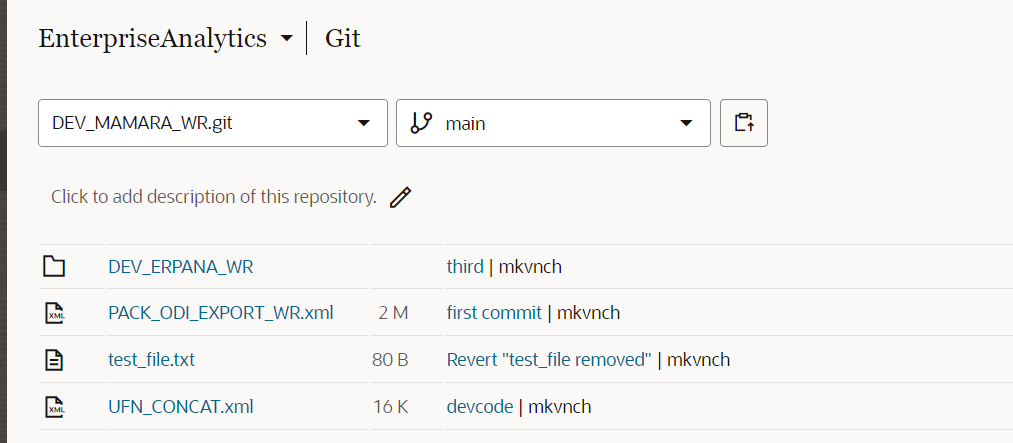


Then we commit the changes and push it into VBS.

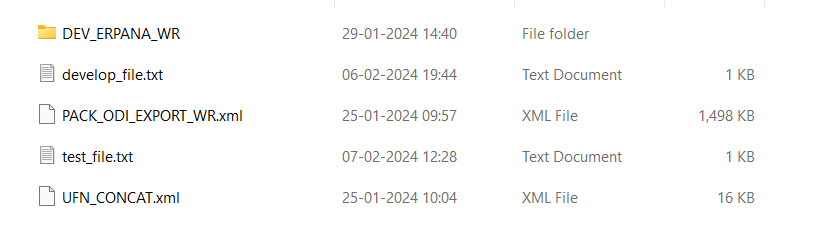


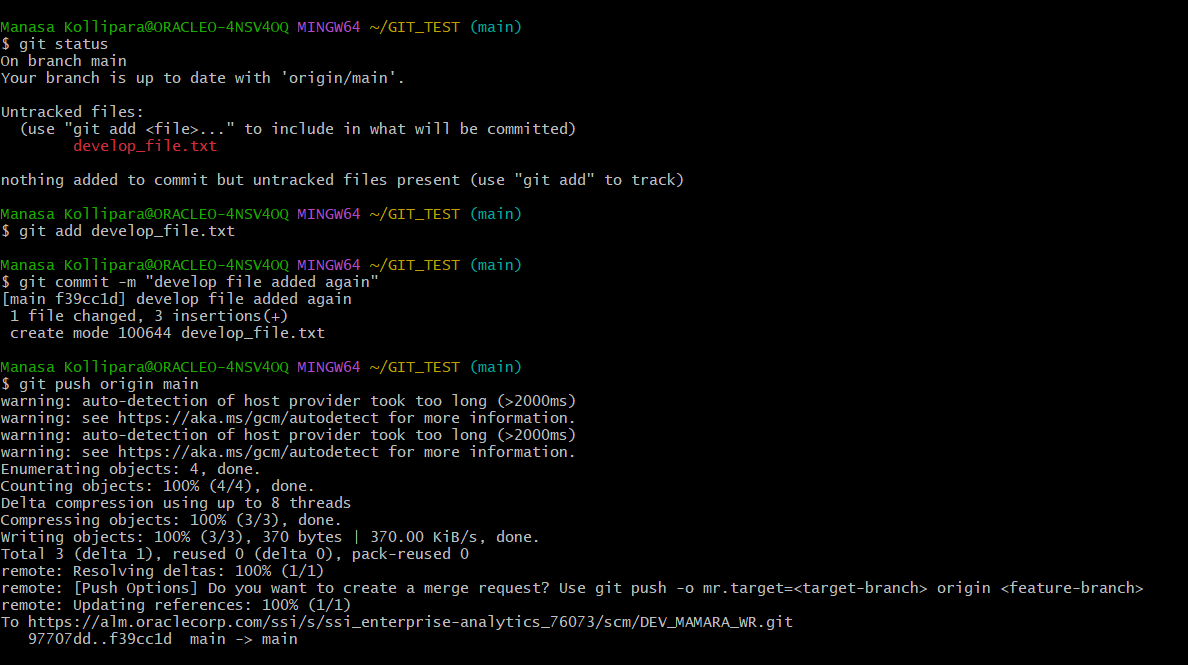


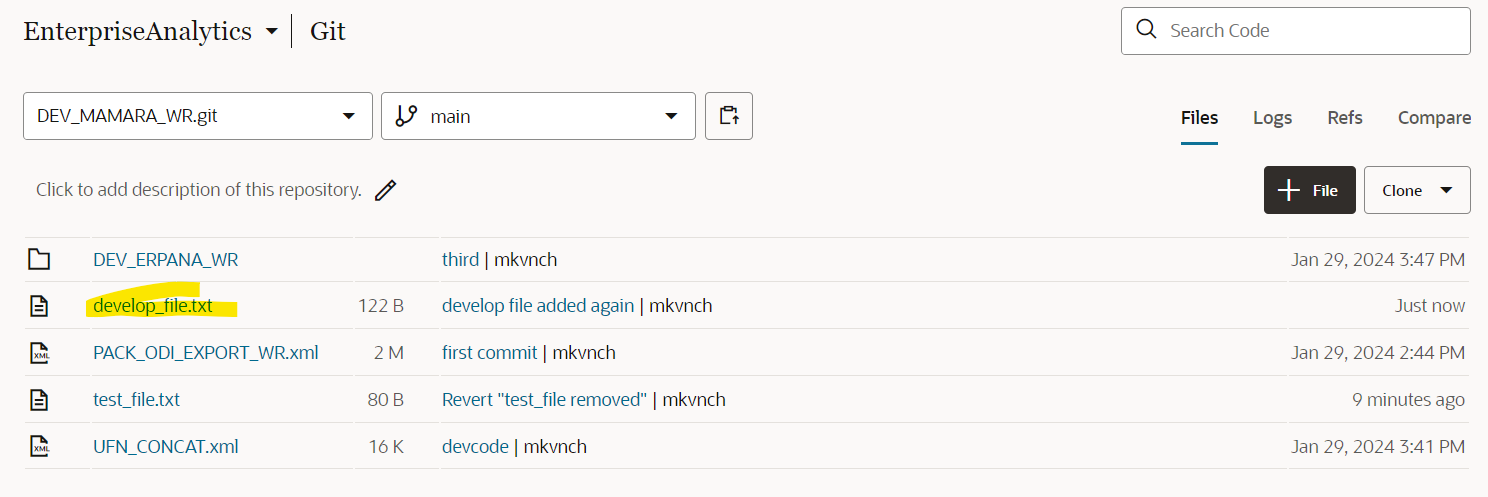
Once we refresh VBS, we can observe that we do not have the develop file in the main branch



But when we check out local, we can still see the develop\_file here. So, we will be able to add the file again and push it back to the main branch.

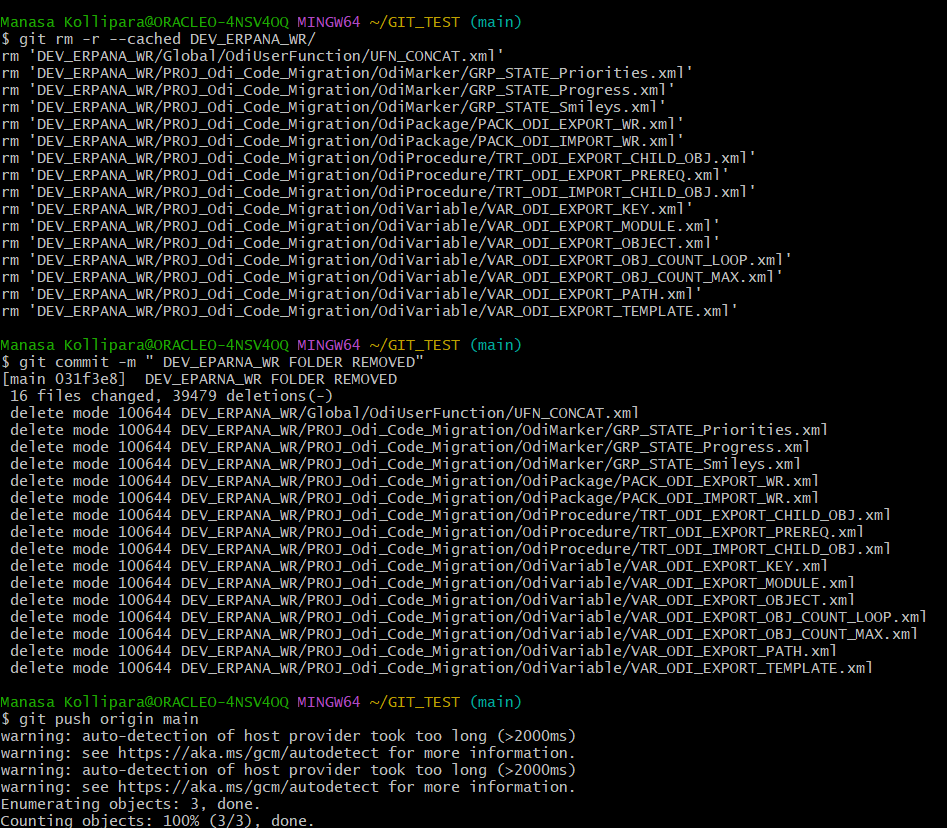




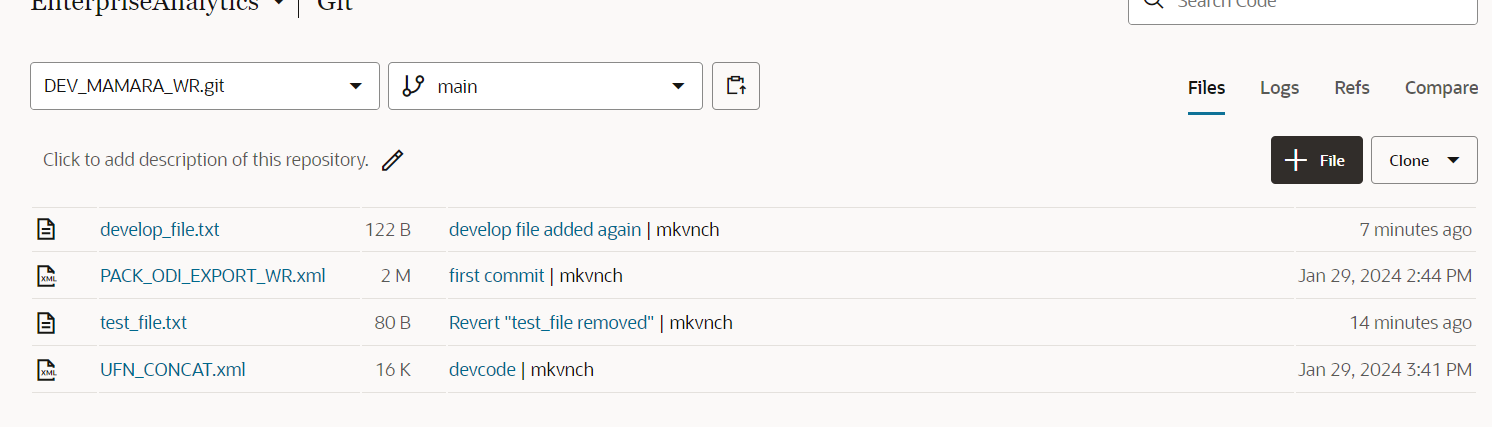
**Remove folder :**

To remove a folder, we have to use -r (recursive) option.

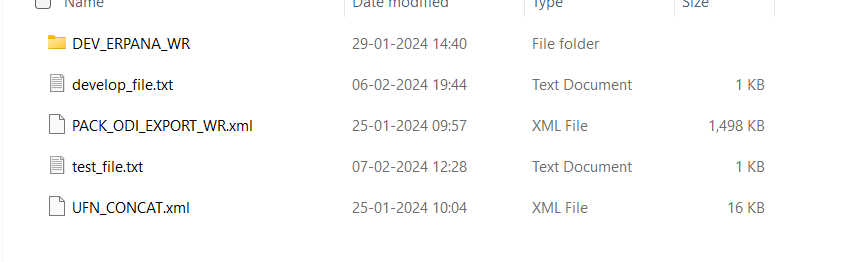
Git rm -r –cached <file\_name>



After removing the folder, commit and push it to the main branch. Then we can observe that the changes are applied to the VBS as well.



As we used –cached option, we can still see the folder in the local folder.



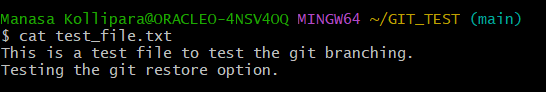
**Feasibility to get any previous committed version of a file:**

We can get any previous committed version of a file. For that we need the ‘sha’ value from the git log.

**Git log –oneline**

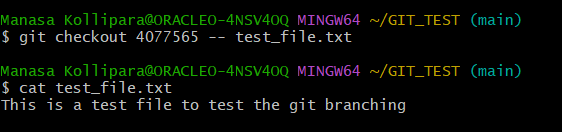
****

We will check the data present in test\_file before moving to the previous version



So now to get the previous version, we can use below command.

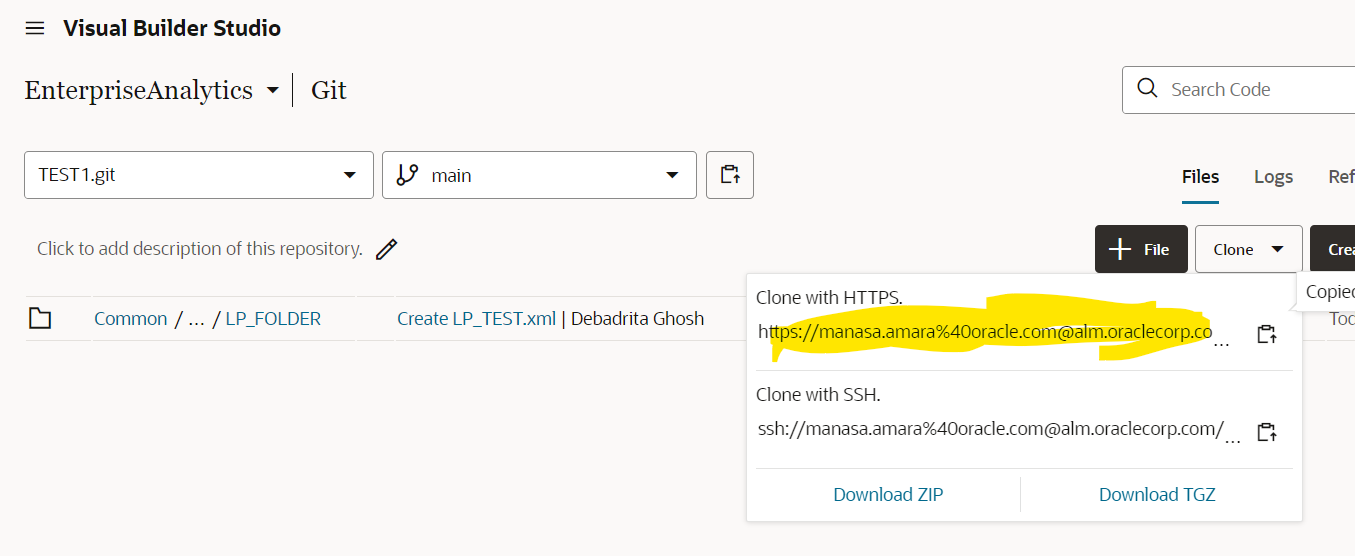
Git checkout <sha value> -- <file name>



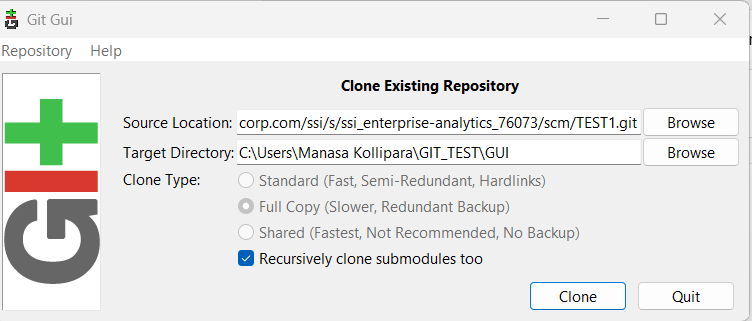
Now we can see that we got the previous version of the file.

**GIT GUI**

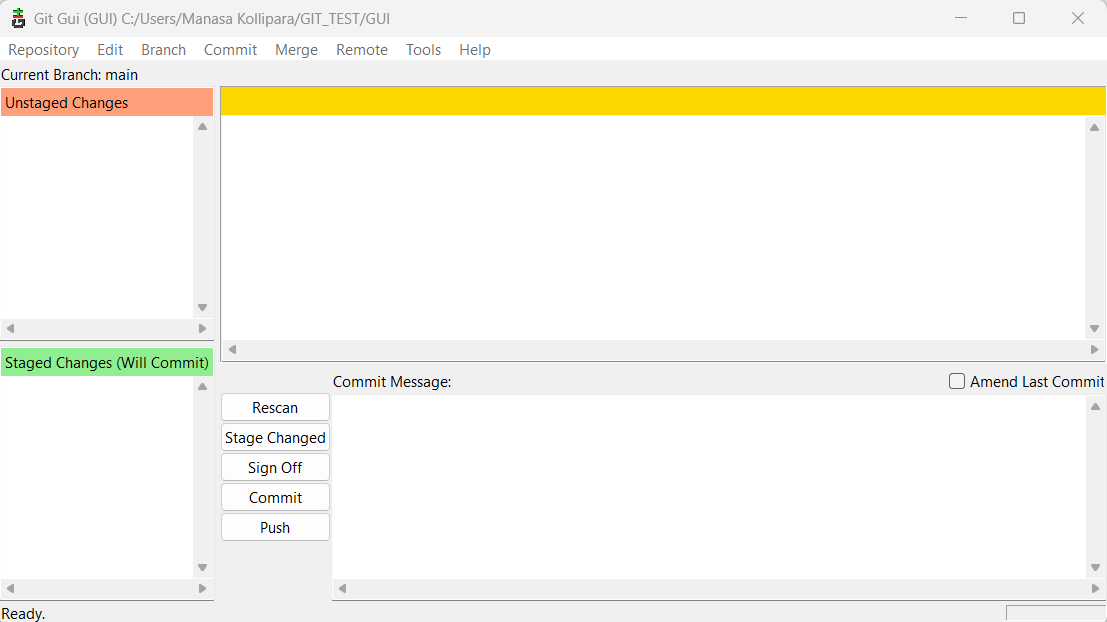
1. First, we can open GIT GUI and then clone to the existing repository. We can get the source location from the VBS.



And target location can be given some local location in our system.

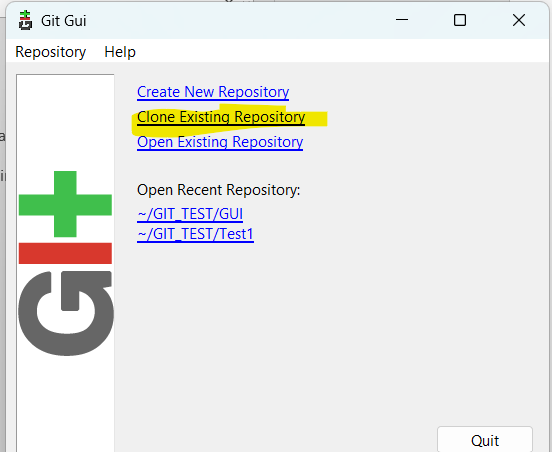


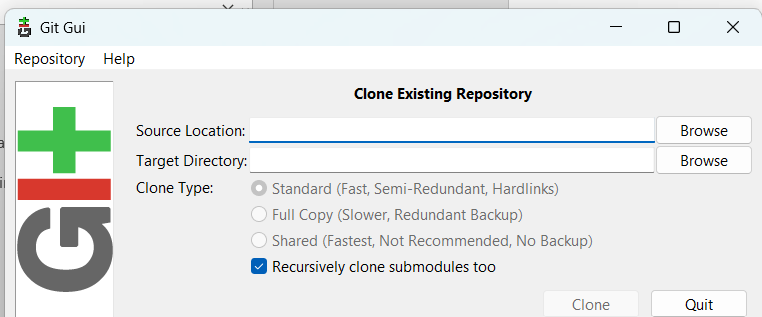
Then we can see the below screen:



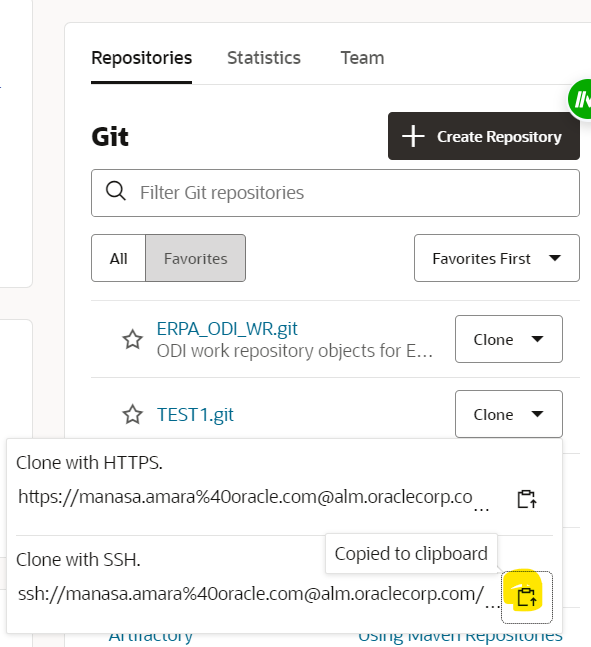
Cloning using SSH key:

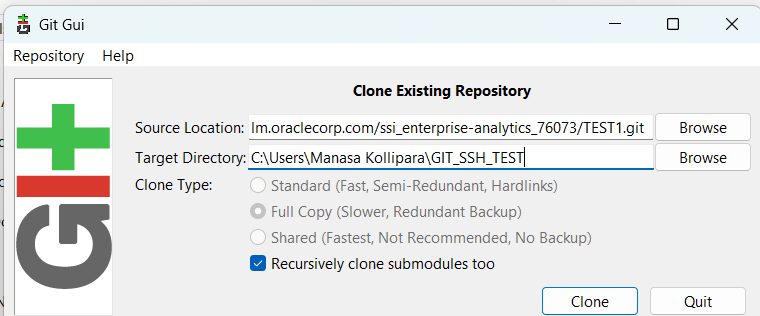
Open GUI and then select ‘clone Existing Repository’ .



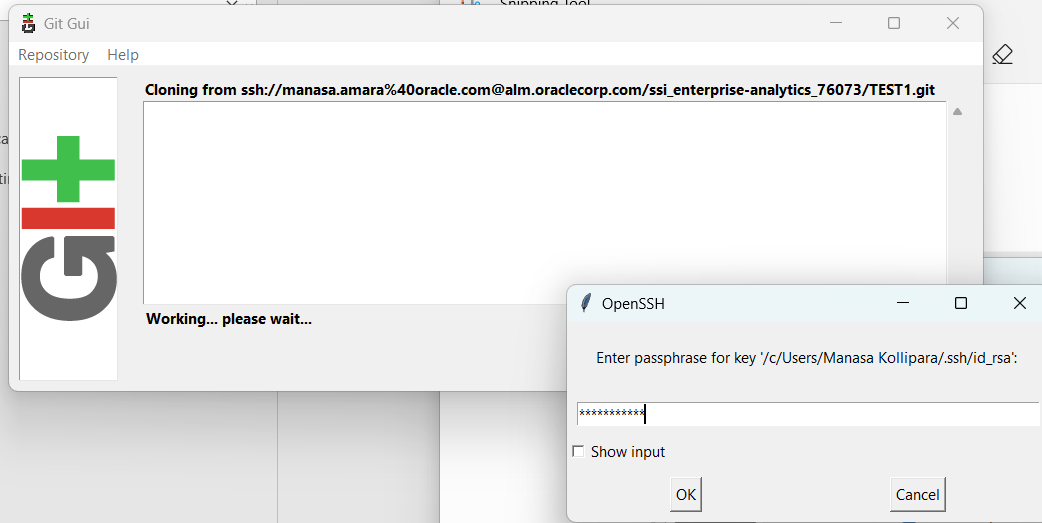


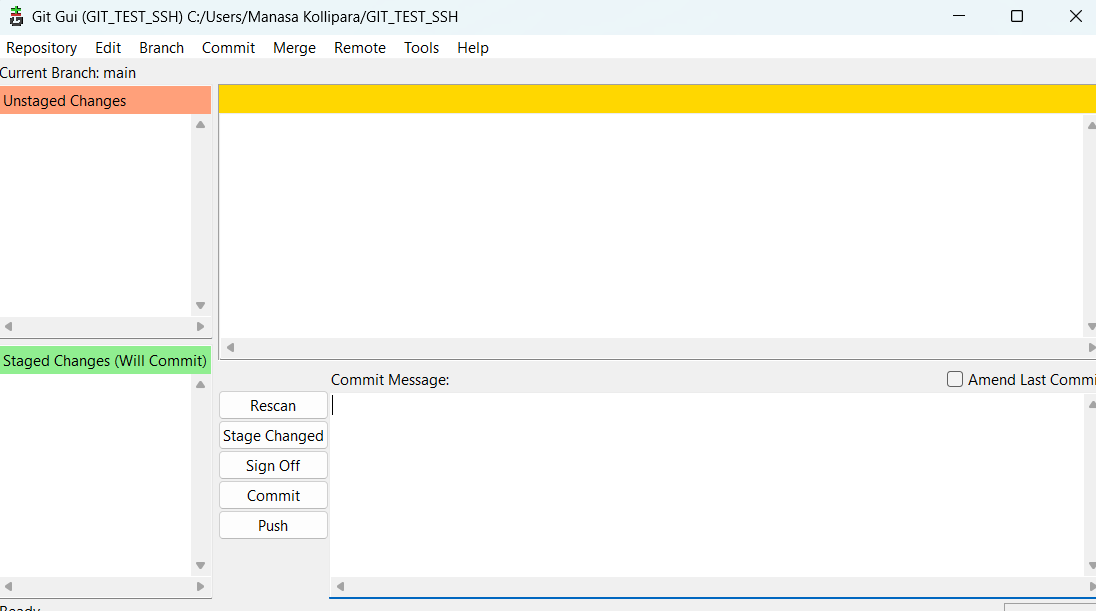
In the source location we must provide the SSH link from the repositories. And then for the target directory we can provide any location in our system.

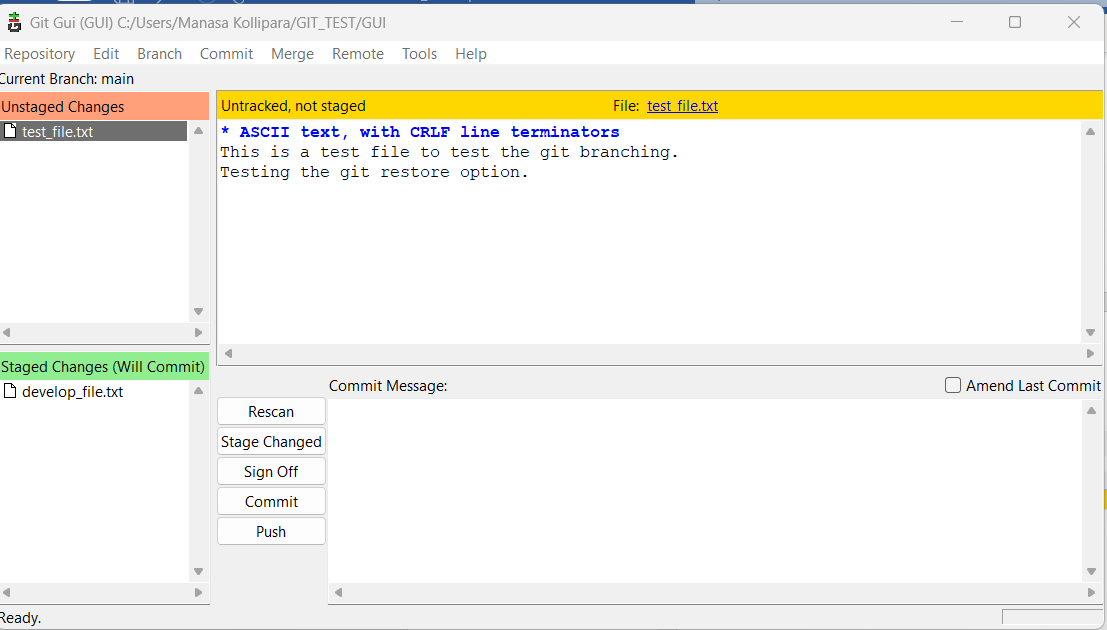




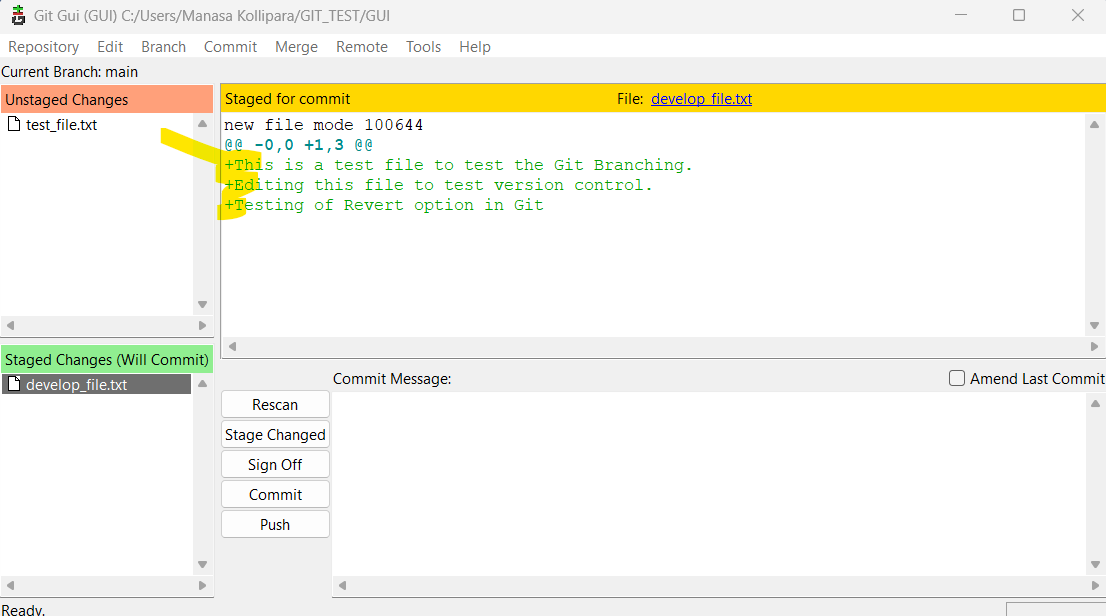
Click on clone. It will prompt for the passphrase around 3 times which we have created earlier. Then we will be able to clone to the repository.



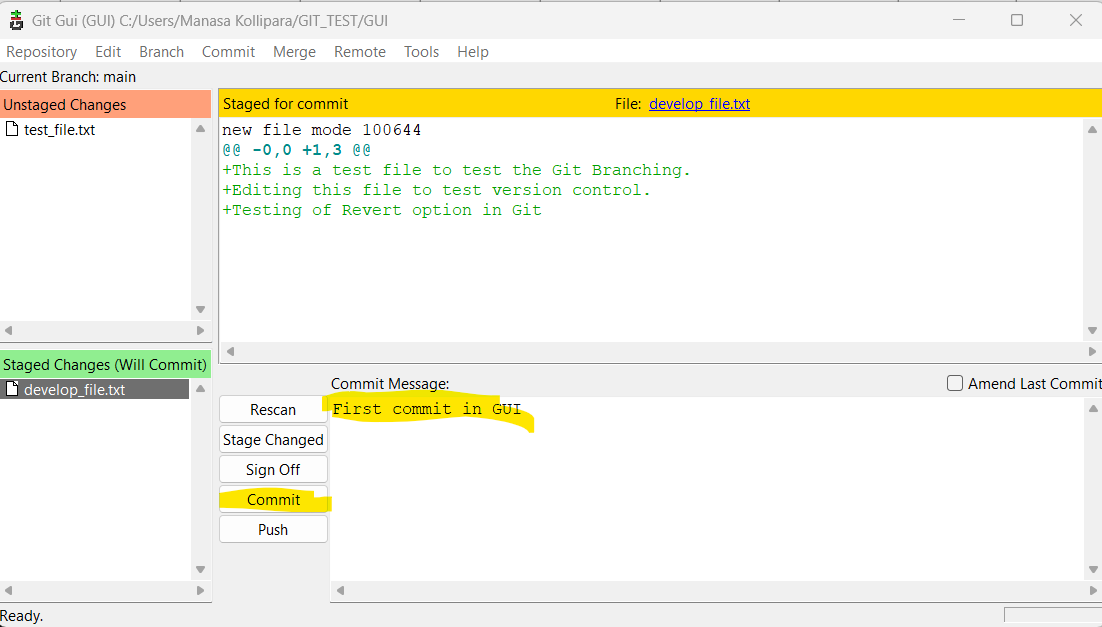




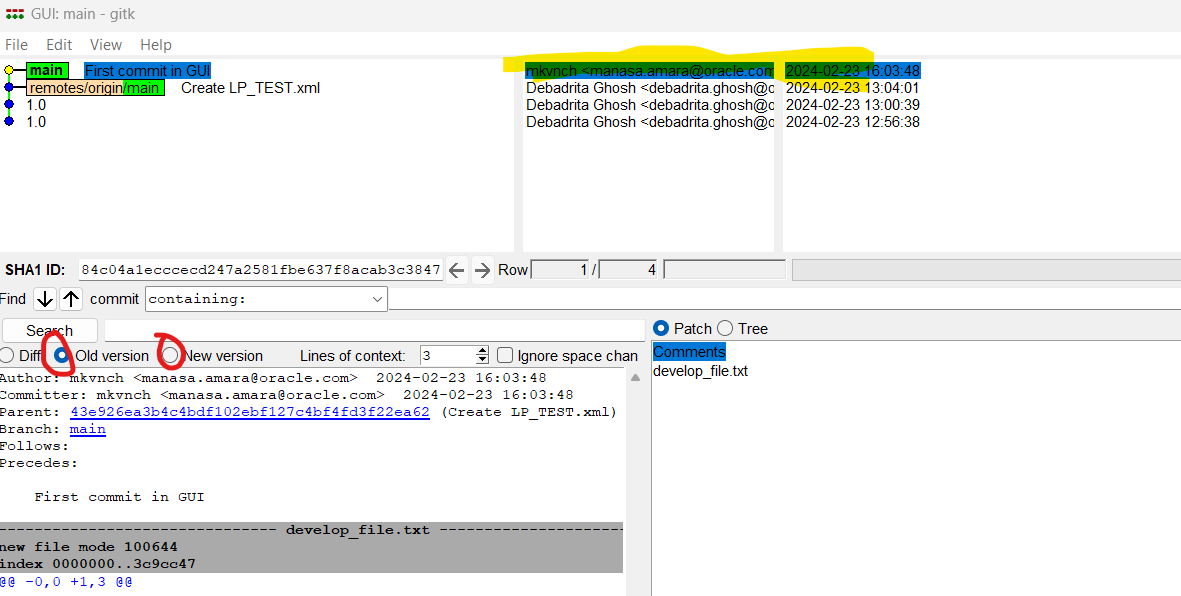
In the ‘Unstaged changes’ section, we can see a file test\_file.txt which means that this file is not yet moved to stage and committed. We can click on the file to view what is present in the file. We can see in the yellow colour that it is **untracked, not staged**. When we click on stage changes button below, then it will be moved to **staged changes** section. In the above screen we can see develop\_file.txt under staged changes section. When we click on this file, we can see there is a ‘+’ sign before every line, + indicates new addition to the file. As you are entering the file for the first time, we can + sign before every line.



We can click on commit and provide the commit Message as below:

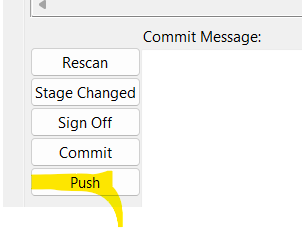


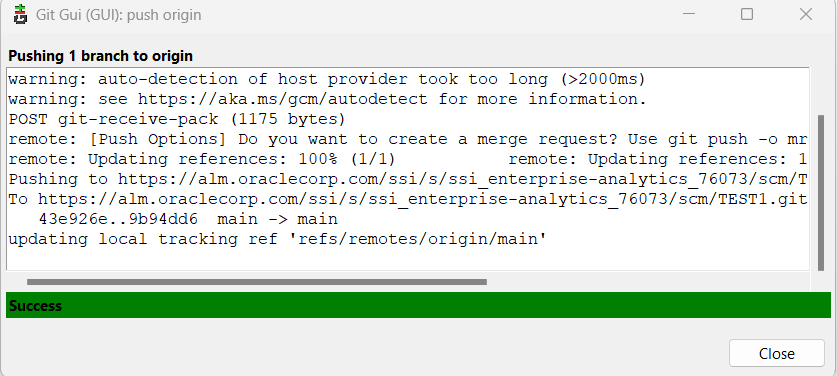
When we click on Repository -> visualize mains history, we can see the log of all the commits as below. We can also see the old version and new version of the files.



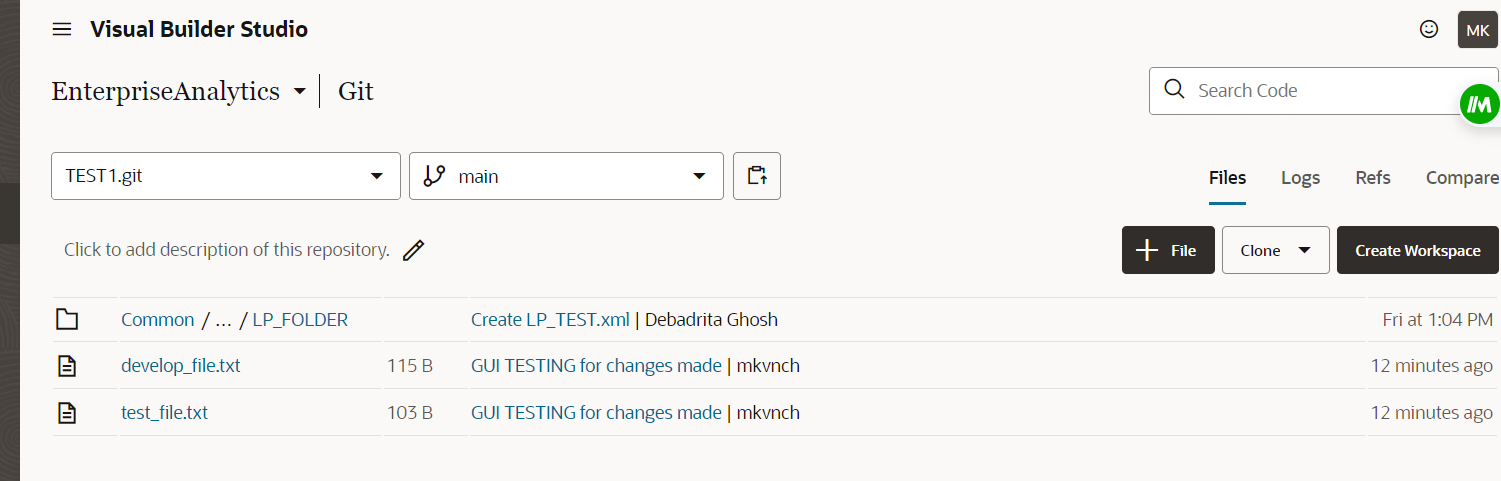
**GIT PUSH:**

Once the changes are committed, we can commit these changes to the main branch by clicking on the Push option.



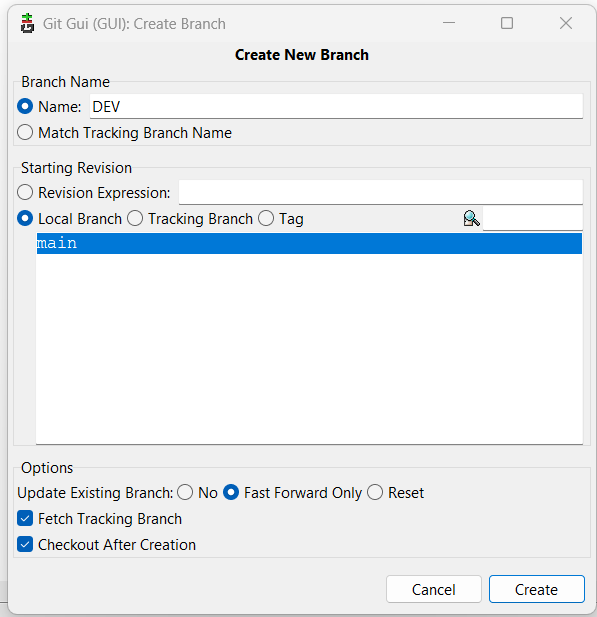


Now if we check in the VBS, we can see that the files are pushed to VBS.

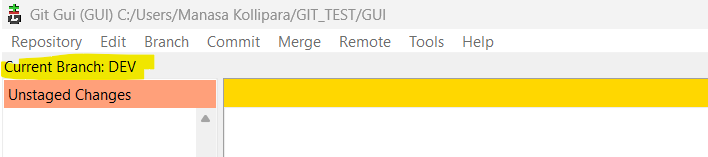


**GIT Branching:**

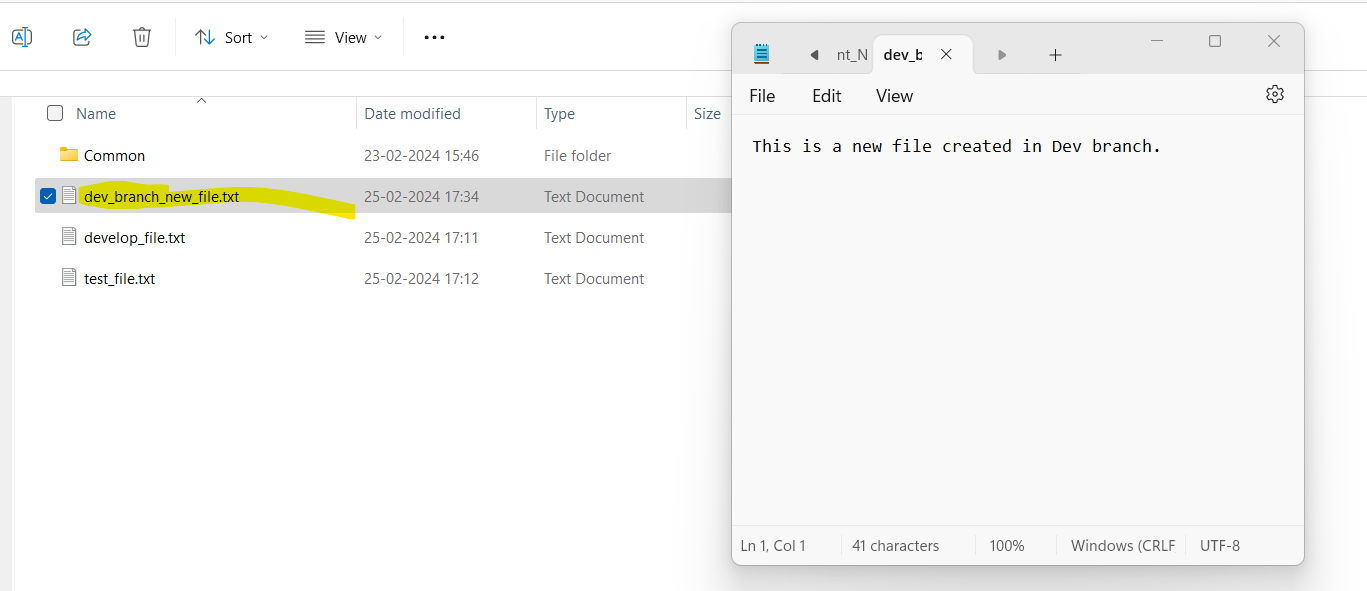
We can create branches from GUI by using the option Branch -> create. Then we can give any name for the branch like Dev.



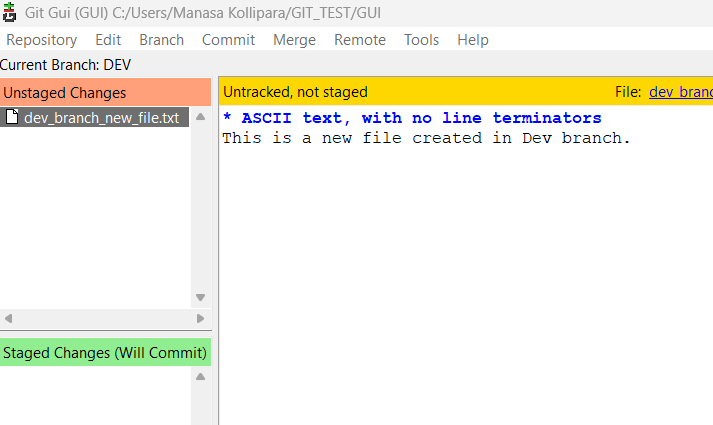
After creation, we can see that it shows that current branch is DEV as below.



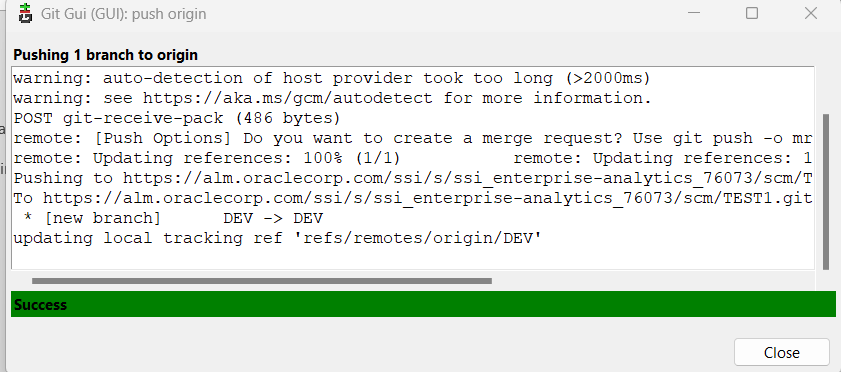
Now I have created a new file in Dev branch with name dev\_branch\_new\_file.txt



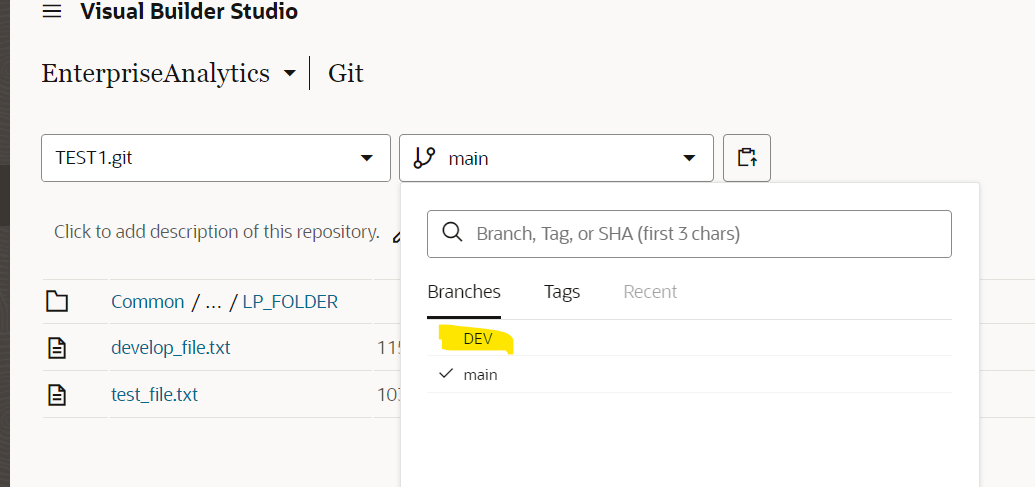
Now when we do Rescan, we can see the file under Unstaged changes section



Before pushing the changes, we can see that in VBS we have only main branch. Now we will perform the push operation.



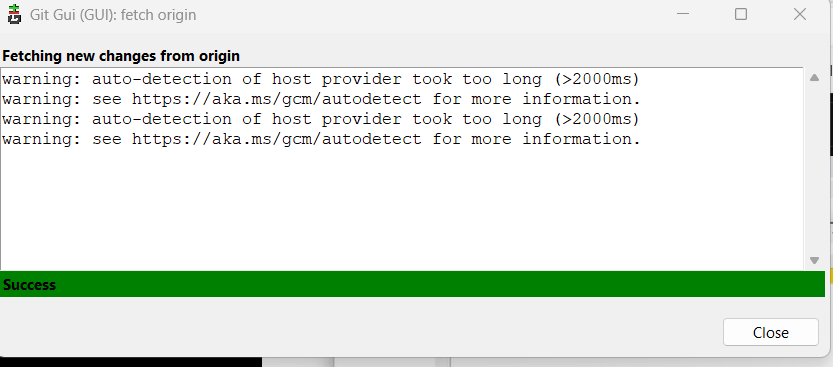
Then when we refresh VBS, we can see that new branch is displayed here.



And we can see the old files along with the new file we have created as below.

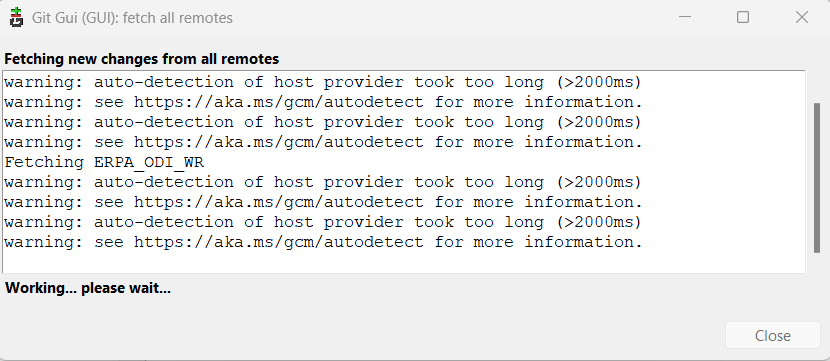
**GIT PULL:**

In order to pull the changes made by another person, we can fetch those changes from VBS. For this go to GIT GUI -> Remote -> Fetch from -> origin. Then the below screen appears.



We can get the new changes from any repository that we wish to. We have this option in the path: Git GUI -> Remote -> Fetch. We can select any repository, or we can select **ALL** which shows the status of each repository.

Here in the below screen , I have selected the option to Fetch All of them, so it is trying to get new changes one after another for each repository.

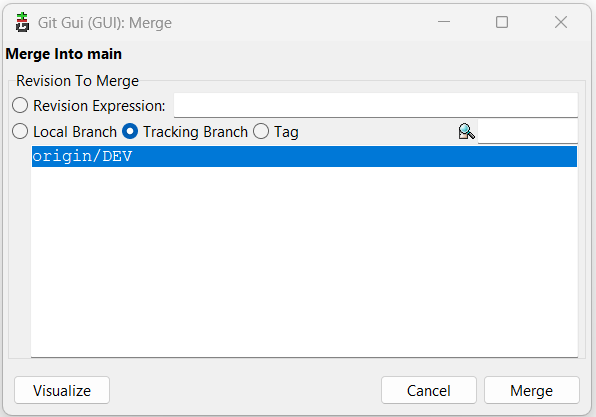


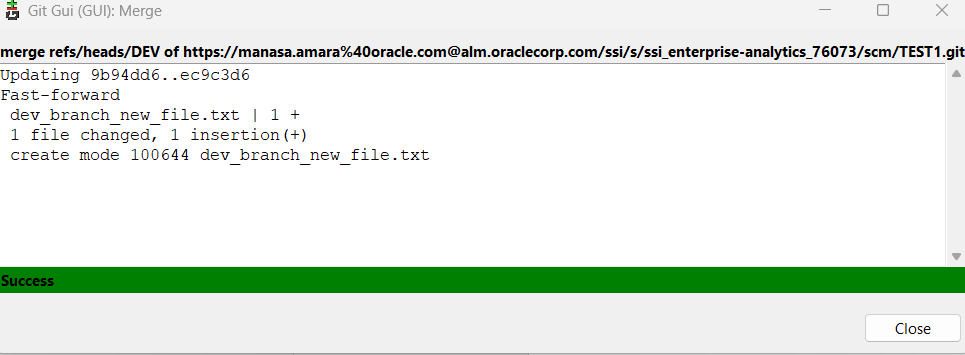
**GIT Merging:**

Now that we have created two branches and now, I want to move to my ‘Main’ branch, then we can to GUI screen and then navigate to Branch -> checkout -> main (select the branch you need). Then we can see that we have moved to the main branch.



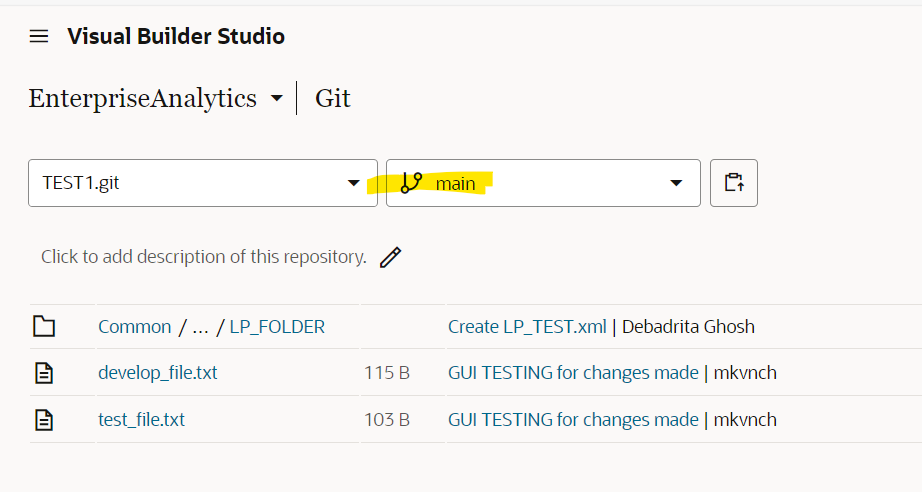
Then go to Merge -> Local Merge



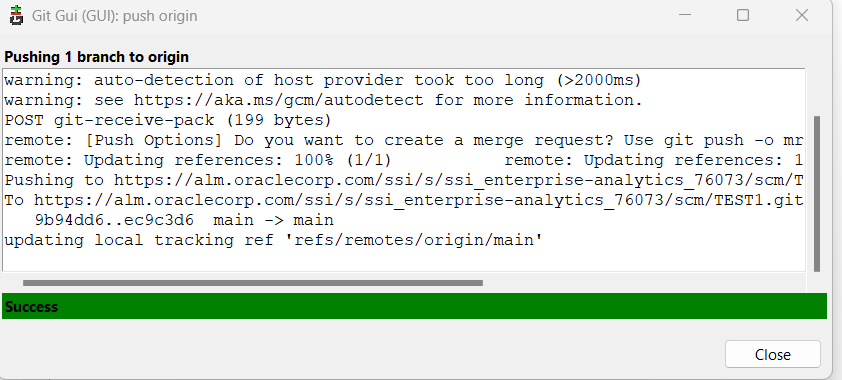


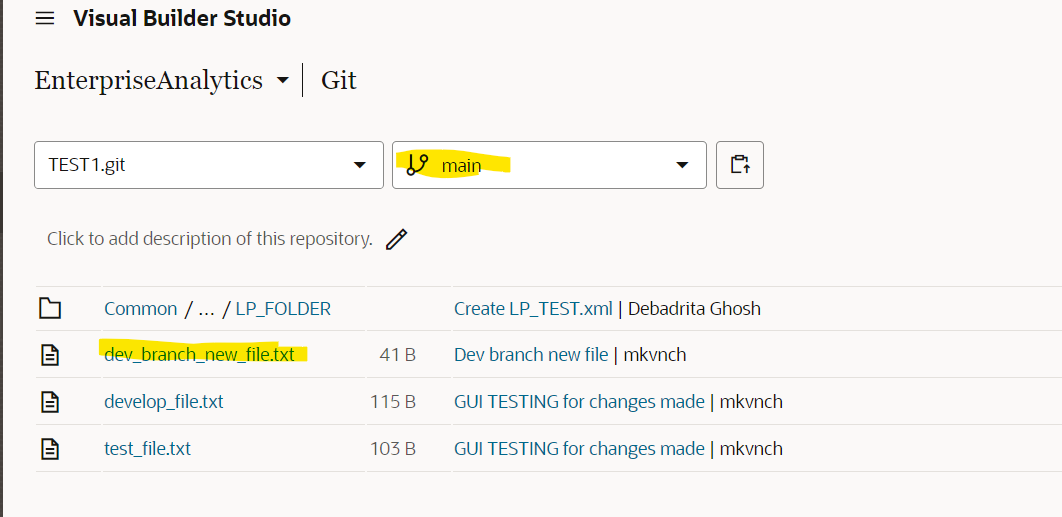
We can see that the new file which we created in the Dev branch is merged here.

As we have not yet pushed these changes to VBS, we do not see the newly created file in VBS main branch.



So lets push these changes and then check in VBS.

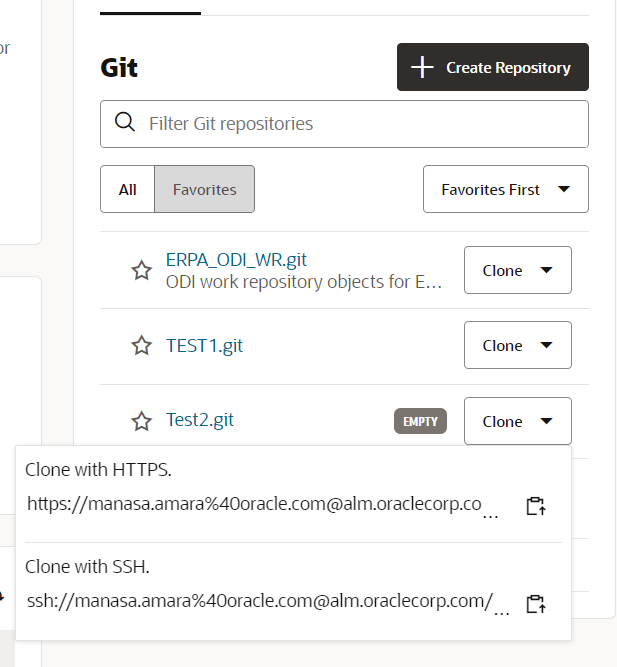


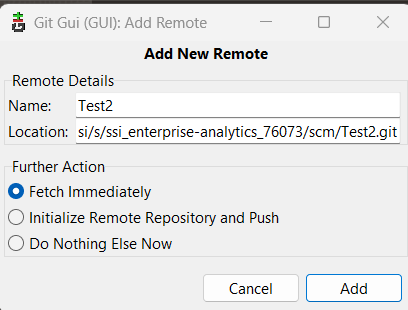


**GIT GUI ADD REMOTE:**

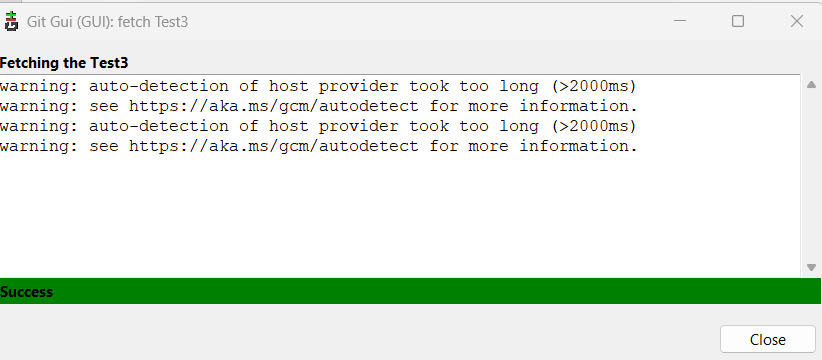
we can add new repository details using ADD Remote feature. Path is Git GUI -> Remote -> ADD

Here I am trying to add new repository which is Test2. So, we have to copy the Https link and paste it.



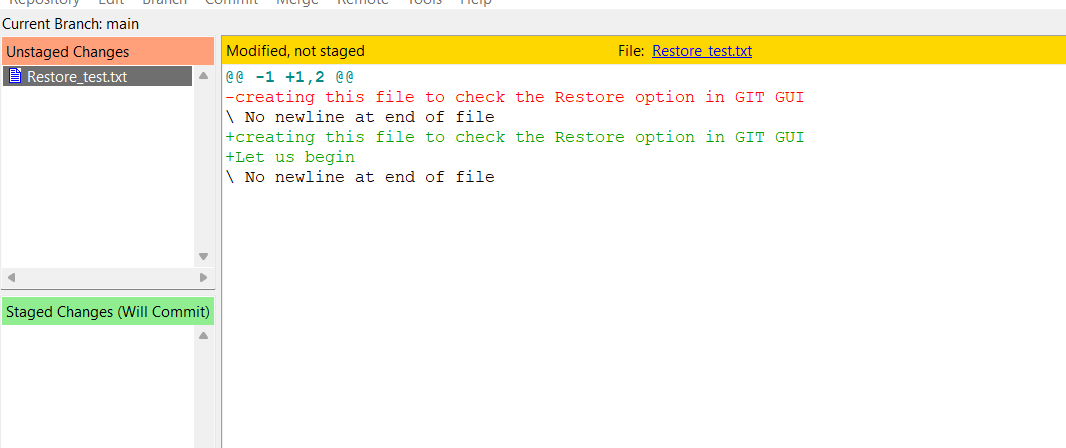


Once it is added we get the success msg as below.

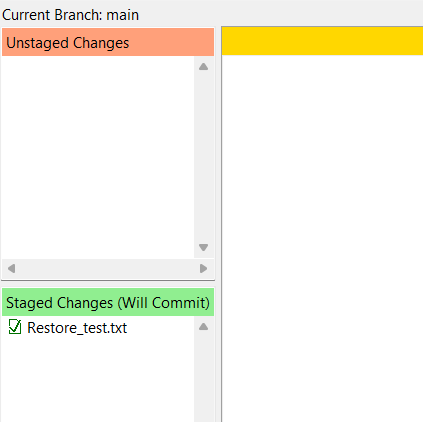


**GIT RESTORE:**

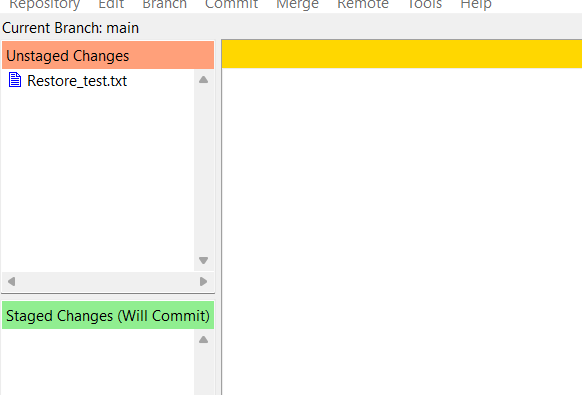
If I have edited a file in main branch, then when we do the Re-scan option, we can see the file that is updated in the ‘Unstaged changes’ section.



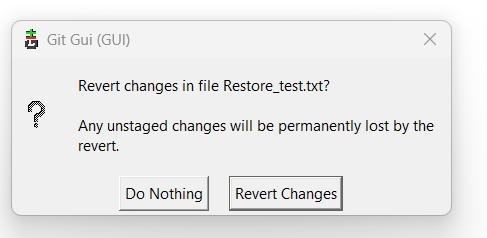
Once we stage the changes to commit then we can see it in ‘Staged changes’ section as below:



Suppose if we want to revert the staged change to unstaged then go to GUI -> commit -> unstage from commit. Then we can see that the file is moved back to unstaged changes.



Now if we permanently want to revert the changes made to the file then we click on gui->commit -> revert changes. Then the below dialog box appears and then we can select ‘Revert changes’ option.



Thus, if we modify a file by mistake, we can revert back the changes using this option.