GIT: git is a version control system , git helps you keep track code of changes , git is used to collaborate the code.

Git is used to create a local repository.To push the code to the local repository first we have to create a file which we want to push to local repository.

How to add the code to local repository:

**Step 1**: open vs using command “code .”

**Step 2**: initialize the project using “git init” (some git file is created)

**git init** => one git file is created as a repository

**Step 3**: create a file and write code in vs code which we want to push to the local repo(ex:index.html)

**Step 4:**now we have to add this to the staging area i.e when we add a file to staging area it means is ready to commit.for add it to the staging area the command is

**git add .=>** To add all the files (or)

**git add index.html** => To add single files or one flle , required file

**Step 5**: now we have to commit the file i.e we are adding the staging area files to local repository

**git commit -m :** or we can write this command using message also

**git commit -m “index.html file is created”** (to increase readability or to understand we use this)

for every commit a id will be created

**What is a repository in GitHub?**

A repository contains all of your project's files and each file's revision history.

**other commands:**

**git status** : which shows status of the local repository i.e if we have files but which are not added to staging area then this git status shows one file is untracked. If all the files are added to repository it shows nothing to commit.

**git log**: which shows the history of commits .i.e how many commits have done , and the autor name and date , messages(commit messages)

**git config --global list**: To view your global Git configuration settings:

**git config --list:** To view the configuration settings for a specific Git repository (local configuration):

**git config global user.name=”your name”** :To set your name globally

**git config global user.email=”your email”** :To set your emailId globally

GitHub: is used to create a remote repository or a central repository.

we are adding local repositories to central repository(git hub)

to add local repo to central repo there are three commands:

**git remote add origin** [**https://github.com/bhavya123111/git-first.git**](https://github.com/bhavya123111/git-first.git) **:** to create remote repo with name of origin

**description:**

[**https://github.com/bhavya123111/git-first.git: link**](https://github.com/bhavya123111/git-first.git:%20link) **of github**

**origin : remote name**

**git branch -M** **Main** (changing the branch name master to main)(initially in git , master branch was created now we just rename it to main)

**git push -u origin main:** To push the code to git hub

if we have any decline issues:

**git remote set-url origin** [**https://bhavya123111@github.com/bhavya123111/git-first.git**](https://bhavya123111@github.com/bhavya123111/git-first.git)

**command:**

**git remote set-url origin "https://github username@github.com/github username/git repo username.git"**

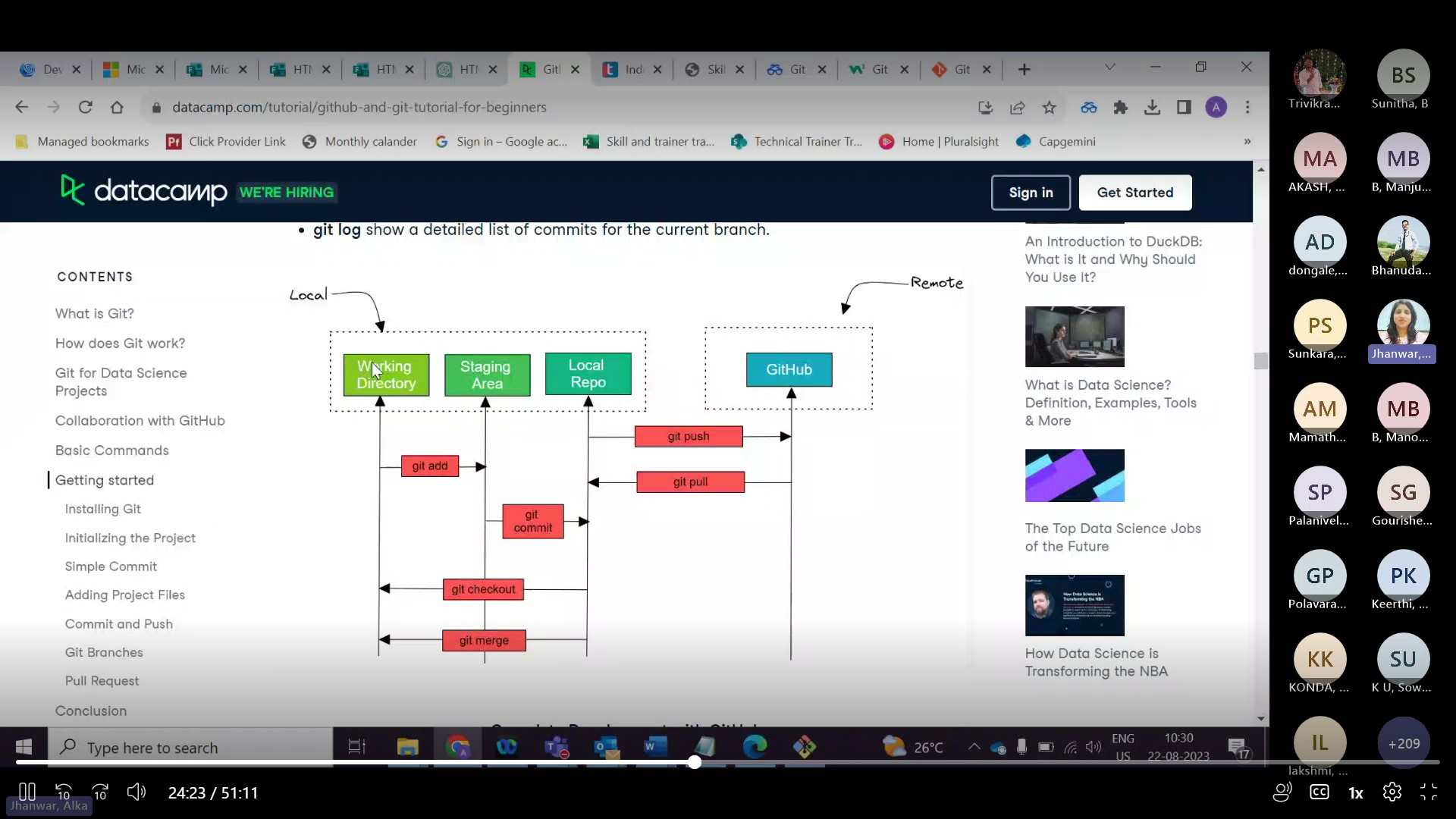
To know username command is : git config -l:

Then it shows one line like

Username username reponame

remote.origin.url=https://bhavya123111@github.com/bhavya123111/git-first.git

**To Understand git and github easily:**



**Branches:** when a developer wants to make some changes on local repository and he wants to add to remote or central repository but he was not sure that it was correct so that he creates a branch for code and move to that branch from local repository.if he feels that it was correct then he merge into main or matster.

Through branches multiple people can work on multiple branches if that was correct then they will move their code to main branch and merge it.

**git commit –amend -m “new message”:** if we want to change the commit message then we use this command. this will only change the last commit message not all previous commits.

**git commit -a -m “message”:** if we are make any changes to the file(in code) after commit it to repository , to make those changes to commit we use this command. Here there is no need to add like no need to use “git add .” , we can directly commit the change because we already done with add by first commit now we are just commit the changes of file

to commit changes of a file:simplest way

we can commit the changes by **“git add .”** and **“git commit -m “message “”**but which is already added to staging area so that we can directly commit it by **“ git commit -a -m “message””**

**git diff:** it shows the difference between working directory and staged area (ex: file is in vs code we haven’t done any add command then it shows the difference that one file is not added otherwise nothing will be showed)

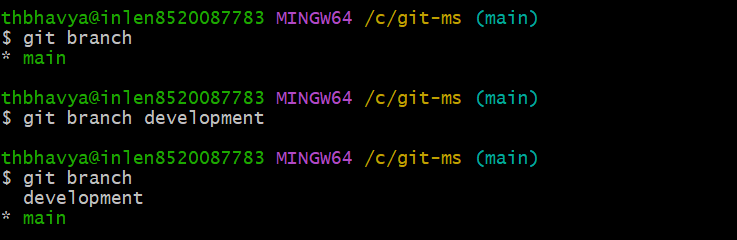
**git diff -staged**: it shows the difference between staging area and local repository(ex: we have added a file to staging area(used add .) but we haven’t commit(havents done commit command) it to local repository then it shows that difference otherwise nothing will be showed)

**git branch**: it will gives you how many branches are there in your local repository

and if it gives star(\*) before branch name that we are presently in that branch only.

Now we create a branch like development for that we use

**git branch development**: creates a branch named with development.



**git checkout development**: now we are moving to development branch from main

now if we check the branches count in our localrepo i.e

**git branch:(executing command)**

result :

main

development(it means now we are in development branch)



=>All the files in main branch are also in development branch.but if we make any changes to development branch it wont affect the main branch i.e changes not added to main branch

=> if we want to add the changes in main branch from development branch then we have to merge those files.

**Syntax: git merge “branch\_name**” (write the branch name which we want to merge to main )

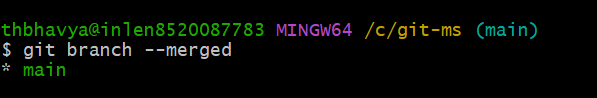
=>i.e **git merge development (**now main and development branches are merged and changes made on development branch will be added to main branch).

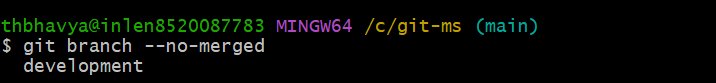
Some more commands on merging:

**git branch --merged**: it shows which branches are merged

**git branch --no-merged:** it shows the branches name which are not merged with main branch

before merging:



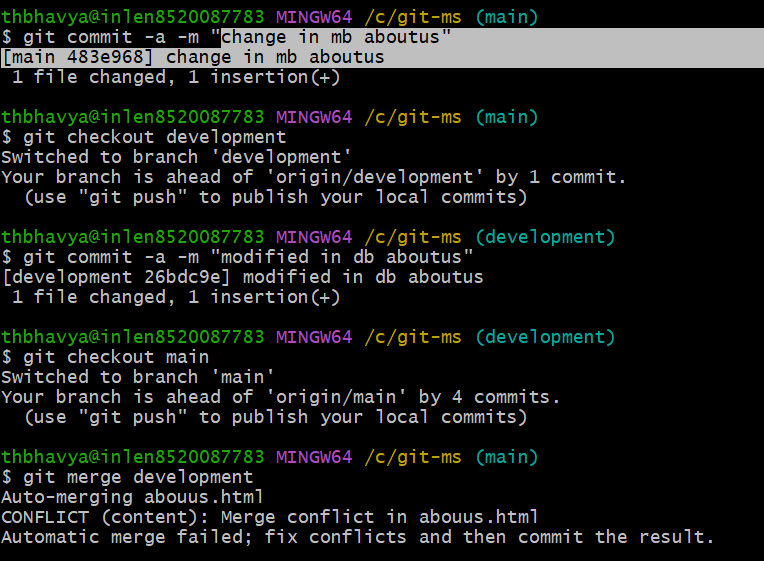


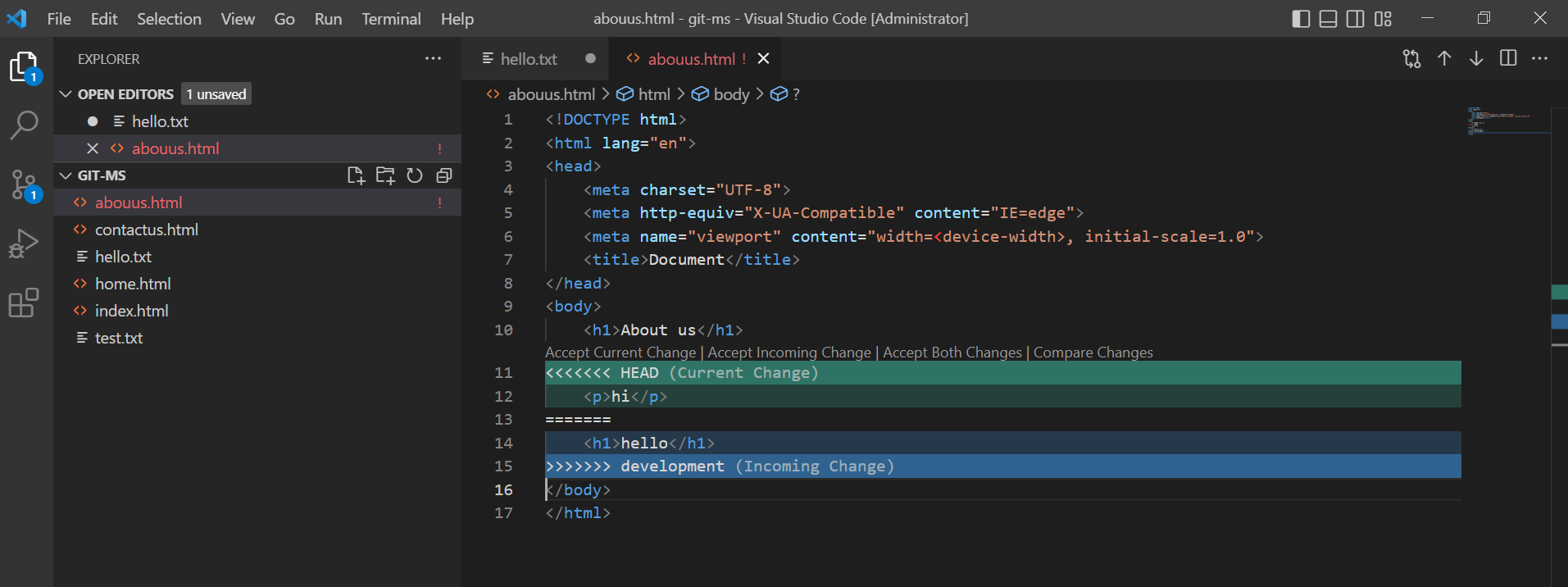
After merging development branch with main:



**Git checkout -b newbranch:** it will directly create a new branch with “newbranch” name and we will move to that branch.

If we make changes in both development branch and main branch , while merge those files it raises a merge conflict issue , for that we can manually solve the issue in vs code.



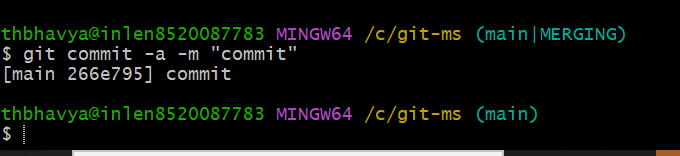


I just modify th code in two branches in main, I add **hi** text and in development I add **“hello”**

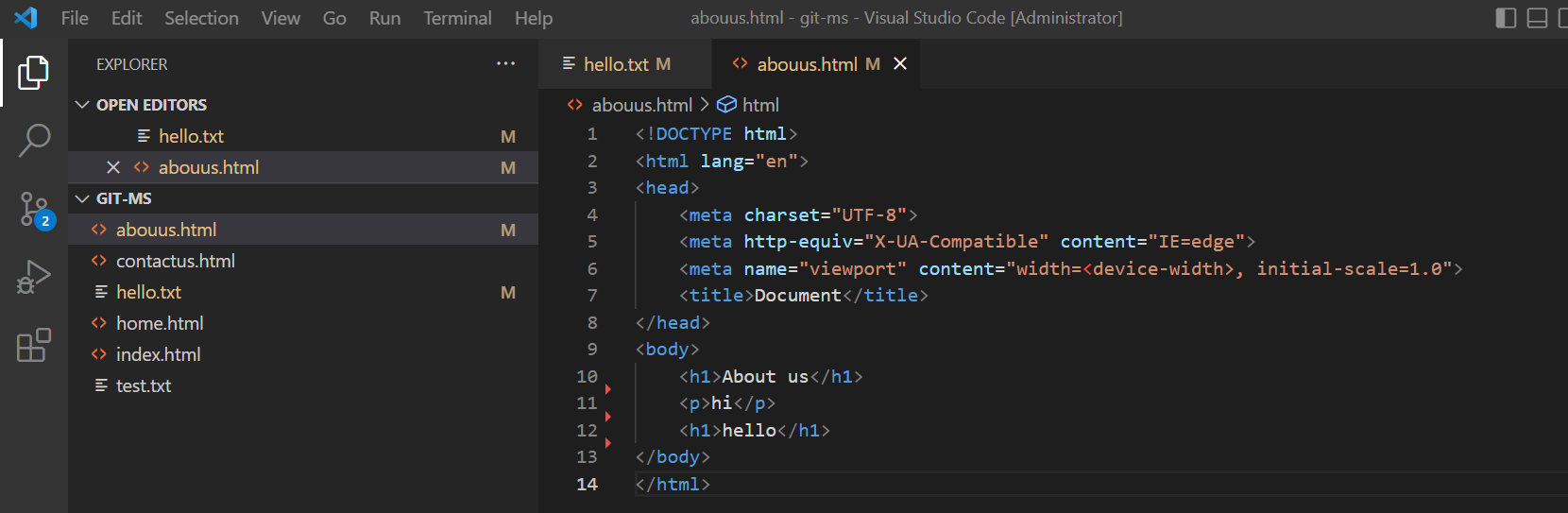
We have to select one ,

If we select **Accept current change** main branch change will be added(hi is added) or if we select **Accept incoming change** only development change will be added (hello is added) or if we click **Accept both changes** then development and main changes will be added.(hi and hello added)

Now I will select **Accept both changes then ,** we have to commit that **i.e**



**So here hi and hello will be added**

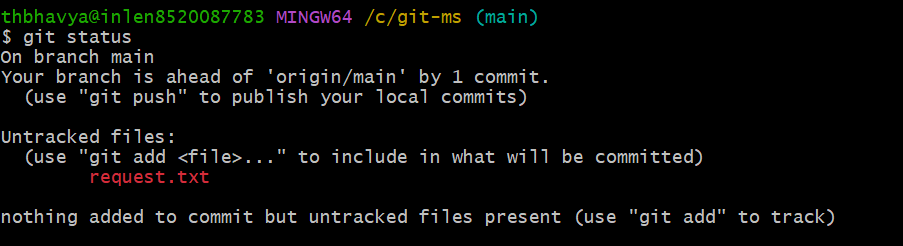


**git push -u origin -all**: we are adding all the branches to central repo.

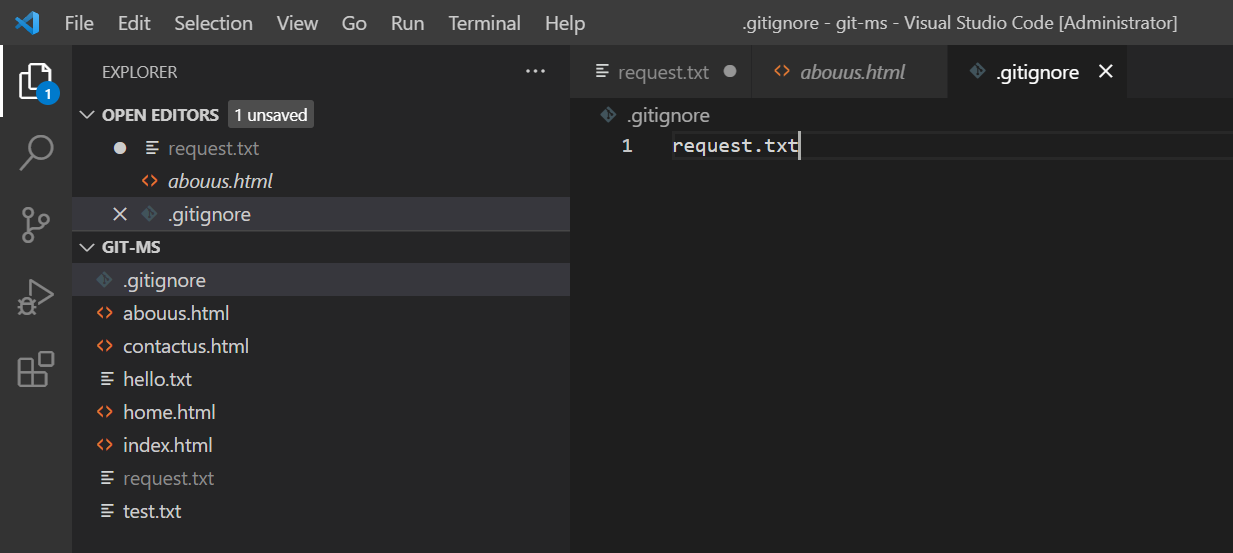
**git push -u origin development**: only development branch is added to central repo.

.gitignore:

If we want to ignore any file that we don’t want to add to local repo theh that file name will be write in .gitignore file so that file will be ignored

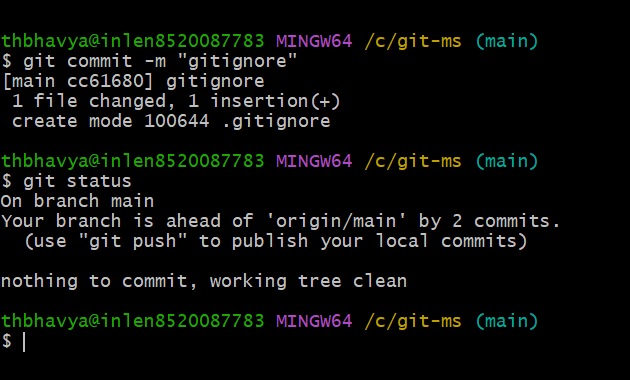


After adding .gitignore file,



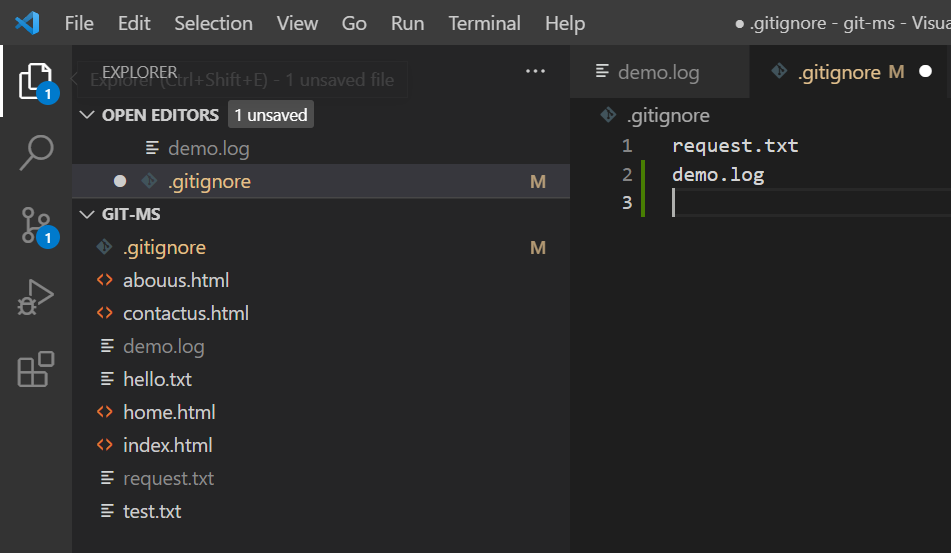
Then

git add .

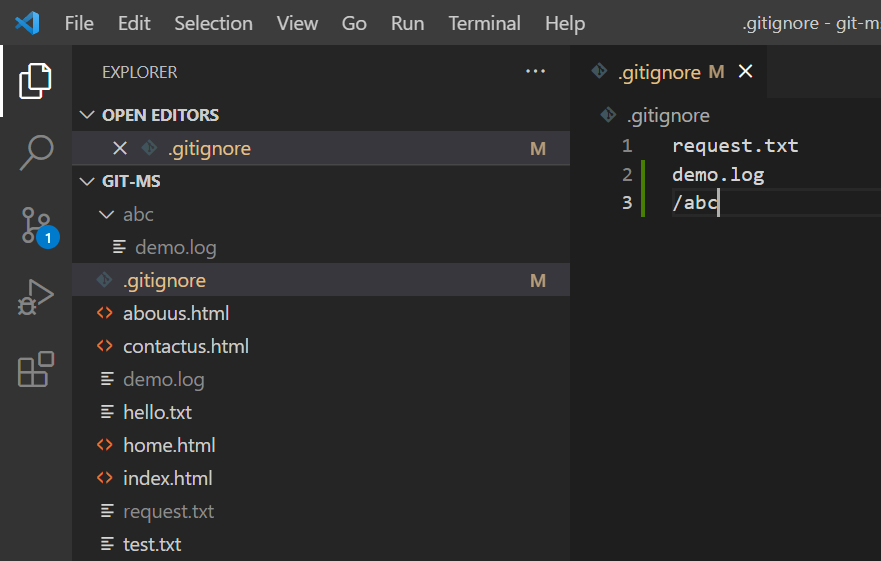


Request.txt file is ignored.

If we want to ignore more than one file then,



To ignore a folder:



To ignore files where those files are extended with .html or etc

Then write “\*.html” in .gitignore file.

**git clone:** we can clone any repository i.e we can add all the files in repo by using repo link in github and command is

**git clone repolink.**

if we made any changes in centralrepo and we want to save those in localrepo then we use these commands:

**git fetch**: pulls the files from central repo to local repo.

**git merge**: pull the files from localrepo to working directory/workspace.

Or else we can use only one command is

**git pull**: adding files or pull files from central repo to current workspace

