Table of Contents

Git and GITHUB Tutorial

Free and open source version control system.
Repository: folder or place whrer the projects are kept.

Git and GitHUB:

Git is a tool which tracks the changes of tour code over the time.

GitHUB is a web site where you host all of your repositories online.

1. Git Commands:

Clone: Bring a repository that is hosted somewhere (like github) into a folder in your local machine.

Add: Track your files and changes in Git.

Commit: Save your files in Git.

Push: Upload Git commits to a remote repo like Github.

Pull: Download changes from remote repo to your local machine, the opposite of push.

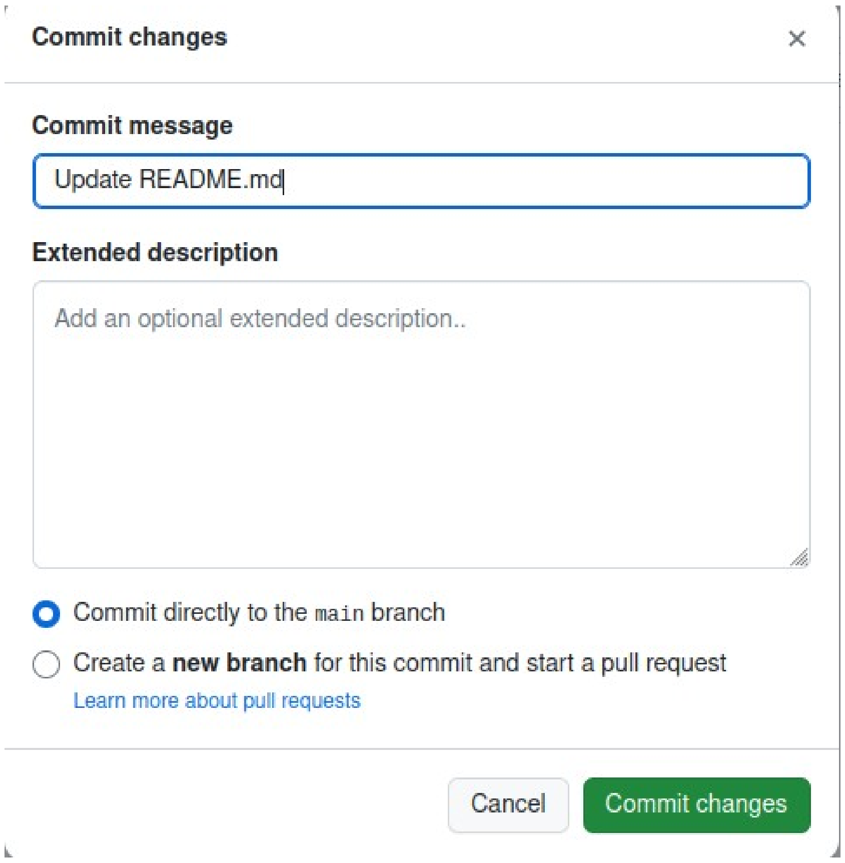
Readme: called as markdown file.

To create a repository:

go to top right corner > Click ‘your repository’ > New >

You can add a readme file.

After editing, do ‘commit changes’.


Commit changes

Commit message

Update README.md

Extended description

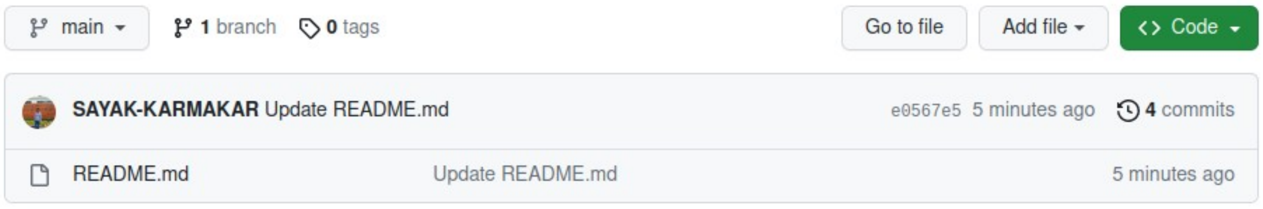
Add an optional extended description..

[✓ Commit directly to the main branch]

Create a **new branch** for this commit and start a pull request

YOU CAN CHANGE THE COMMIT MESSAGE also or add a description.

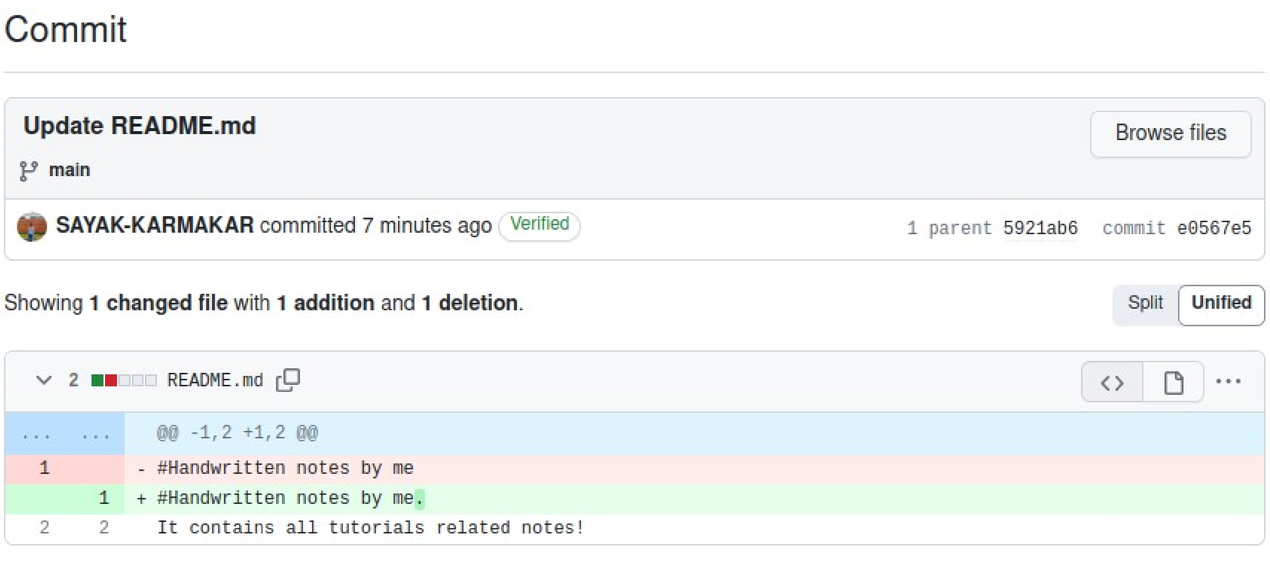
Click Update README.md beside README.md



SAYAK-KARMAKAR Update README.md e0567e5 5 minutes ago **4** commits

README.md Update README.md 5 minutes ago

Now you can see that what has been changed to the file in several times. Red colour mrans those lines were deleted or removed and green colour means those lines are added to the previous files. White colours means, it was kept as same.


Commit

Update README.md

main

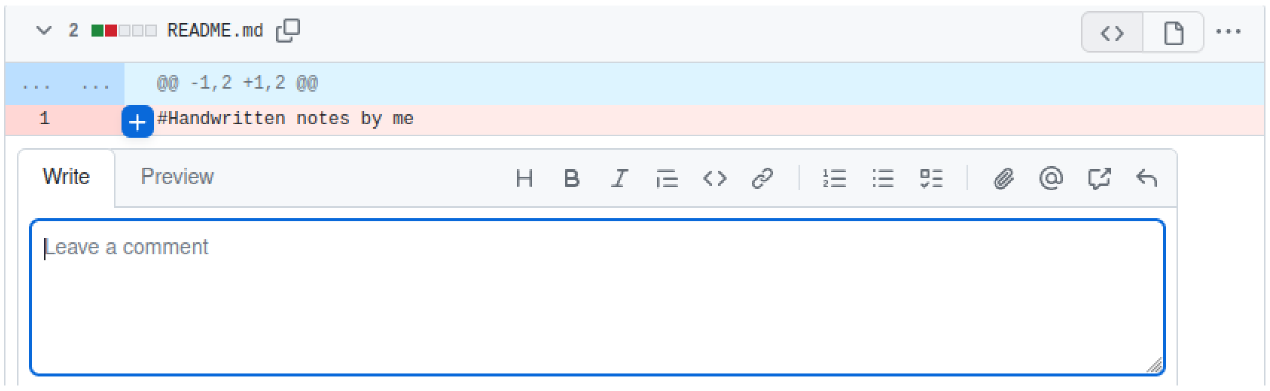
SAYAK-KARMAKAR committed 7 minutes ago Verified

1 parent 5921ab6 commit e0567e5

Showing 1 changed file with 1 addition and 1 deletion.

```
2 README.md
@@ -1,2 +1,2 @@
1     - #Handwritten notes by me
   1  + #Handwritten notes by me.
2  2    It contains all tutorials related notes!
```

If you click the+ or – sign, a blue + will come and you can add some message about that commit.


```
2 README.md
@@ -1,2 +1,2 @@
1  #Handwritten notes by me
```

Installing git!!!????

Here are some commands to install git, obtained from another tutorial. Details are not mentioned here. May be some parts are missing also. Chek it later.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git$
git config --global user.email
"sayakju97@gmail.com"
sayak@sayak-Precision-Tower-3620:~/Desktop/git$
git config --global user.name "SAYAK-KARMAKAR"
sayak@sayak-Precision-Tower-3620:~/Desktop/git$
git config user.name
```

```
SAYAK-KARMAKAR
sayak@sayak-Precision-Tower-3620:~/Desktop/git$
git config user.email
sayakju97@gmail.com
sayak@sayak-Precision-Tower-3620:~/Desktop/git$
git commit -m "first commit"
[master (root-commit) 2d5a61e] first commit
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
```

- **2. To check whether git is installed in our system or not**

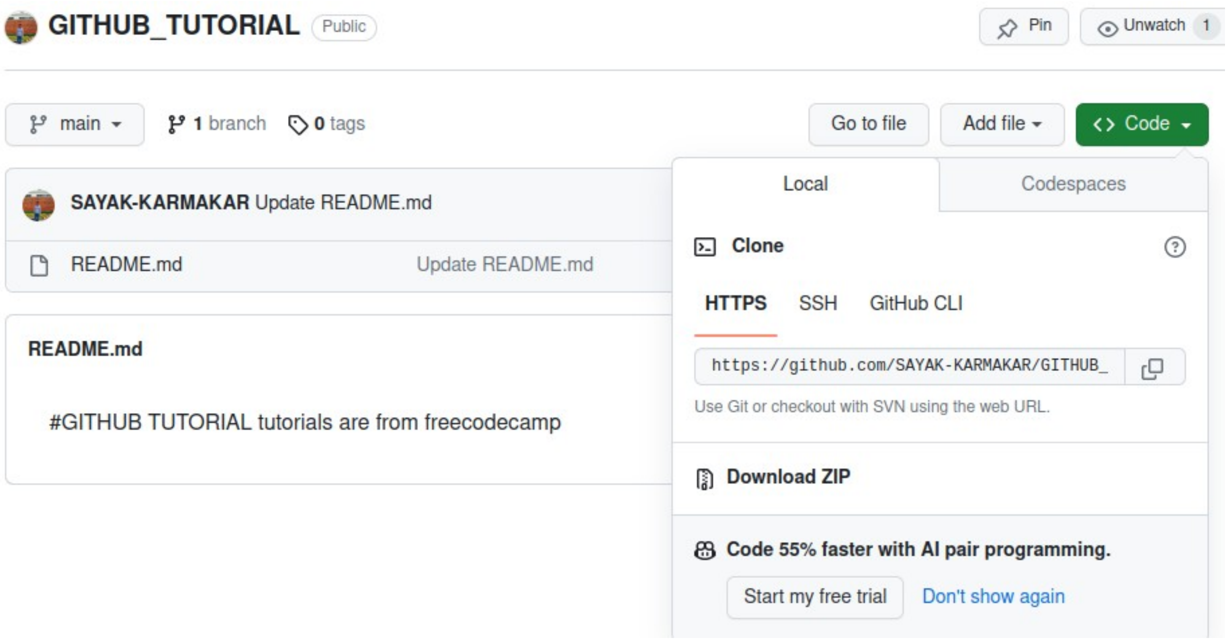
```
sayak@sayak-Precision-Tower-3620:~$ git –version
git version 2.34.1
```

- **3. Cloning repository in local system and to make changes locally and again push it to github**

First create a folder anywhere in the computer. In vs code, go to **file** > **open folder** and select that specific folder (folder name=’**git**’ here).

Then from **view** > **terminal**, open that folder in terminal within vs code environment.

Now, we will pull the repository created in Github into my local system using git. So, open that repository ‘GITHUB_TUTORIAL’ and under ‘**code**’ option, copy the address and write the following command. Then the repository will be cloned to the created folder in the computer.


GITHUB_TUTORIAL
Public
Pin
Unwatch 1
main
1 branch
0 tags
Go to file
Add file
Code
SAYAK-KARMAKAR Update README.md
README.md
Update README.md
README.md
#GITHUB TUTORIAL tutorials are from freecodecamp
Local
Codespaces
Clone
HTTPS
SSH
GitHub CLI
https://github.com/SAYAK-KARMAKAR/GITHUB_
Use Git or checkout with SVN using the web URL.
Download ZIP
Code 55% faster with AI pair programming.
Start my free trial
Don't show again

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git$
git clone
https://github.com/SAYAK-KARMAKAR/GITHUB_TUTORIAL.
git
```
#command for cloning

```
Cloning into 'GITHUB_TUTORIAL'...
remote: Enumerating objects: 15, done.
remote: Counting objects: 100% (15/15), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 15 (delta 1), reused 0 (delta 0),
pack-reused 0
Receiving objects: 100% (15/15), done.
Resolving deltas: 100% (1/1), done.
sayak@sayak-Precision-Tower-3620:~/Desktop/git$ cd
GITHUB_TUTORIAL/
```
#this is the cloned directory

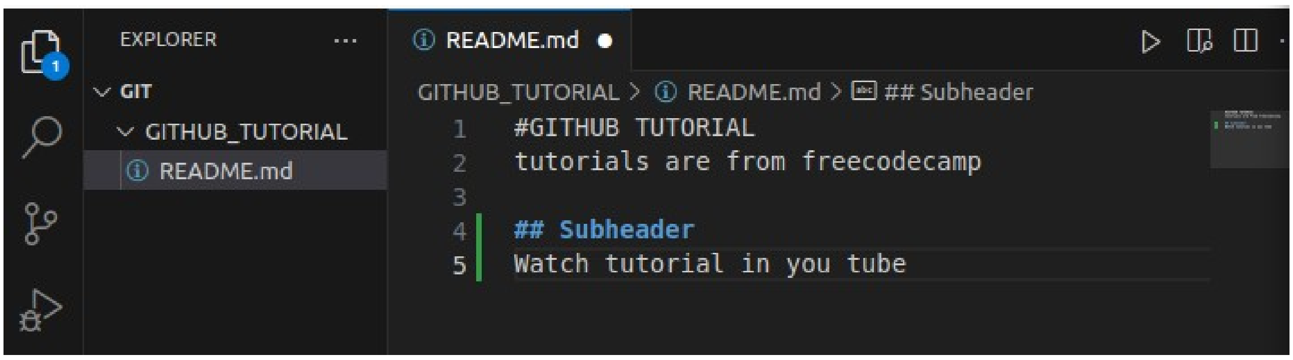
```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ ls -la
```
#this command will show the hidden files also in that directory.

```
total 16
drwxrwxr-x 3 sayak sayak 4096 Nov 25 17:48 .
drwxrwxr-x 3 sayak sayak 4096 Nov 25 17:48 ..
drwxrwxr-x 8 sayak sayak 4096 Nov 25 17:48 .git
```

#".git" in blue colour means it is a folder. This is a hidden folder. This folder contains all the changes recorded in the history of the repository, which we made in github.com.

```
-rw-rw-r-- 1 sayak sayak   49 Nov 25 17:48
README.md
```

Now, let us make some changes in the readme file locally. Add something in vs code.



```
#GITHUB TUTORIAL
tutorials are from freecodecamp

## Subheader
Watch tutorial in you tube
```

We need to save these changes to git.

First we will use 'git status' command.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git status
```

#this command shows all the files that were updated , created or deleted.But havenot been saved in a commit yet.

```
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be
committed)
  (use "git restore <file>..." to discard changes
in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or
"git commit -a")
```

If we create a new file in vs code named as ‘index.html’ and run ‘git status’ command:

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be
committed)
  (use "git restore <file>..." to discard changes
in working directory)
        modified:   README.md

Untracked files:
```

#untracked files means, git does not know about this file yet. So we need ‘git add <file>’ command.

```
  (use "git add <file>..." to include in what will
be committed)
        index.html

no changes added to commit (use "git add" and/or
"git commit -a")

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git add index.html
```

#this command permits gits to keep track of individual file.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git add .
```

#this command permits gits to keep track of everything added to this current folder

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git status
On branch main
```

```
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to
unstage)
        modified:   README.md
        new file:   index.html
```
#Now all the changes has been tracked and files are ready to be commited.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git commit -m "Added index.html
i.e a new file" -m "This thing goes to description
box"
```
#-m for message and you need to have a message in order to commit your files.This should ideally tell something about the changes you made. Second '-m' is for description box.

```
[main 789e78e] Added index.html i.e a new file
 2 files changed, 4 insertions(+)
 create mode 100644 index.html
```

Generating SSH key

Till now we have saved our code locally. C**ommit is not live on github yet**. So, we need command called '**git push**'. But in order to do that you need to prove github that you are the owner of the account. To connect the local machine to github account, you need ssh key.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ ssh-keygen -t rsa -b 4096 -C
"sayakju97@gmail.com"
```
#`-t rsa`: Specifies the type of key to create (RSA in this case.RSA, which stands for Rivest–Shamir–Adleman, is a widely used public-key cryptosystem that enables secure data transmission and digital signatures.). `-b 4096`: Specifies the number of bits in the key (4096 bits in this

case). `-C "sayakju97@gmail.com"`: Adds a comment to the key, typically an email address.

```
Generating public/private rsa key pair.
Enter file in which to save the key
(/home/sayak/.ssh/id_rsa): id_rsa
```

#'id_rsa' is the default file for ssh key. You may use another name also. But, better to use it to avoid further complications.

```
Enter passphrase (empty for no passphrase):
```

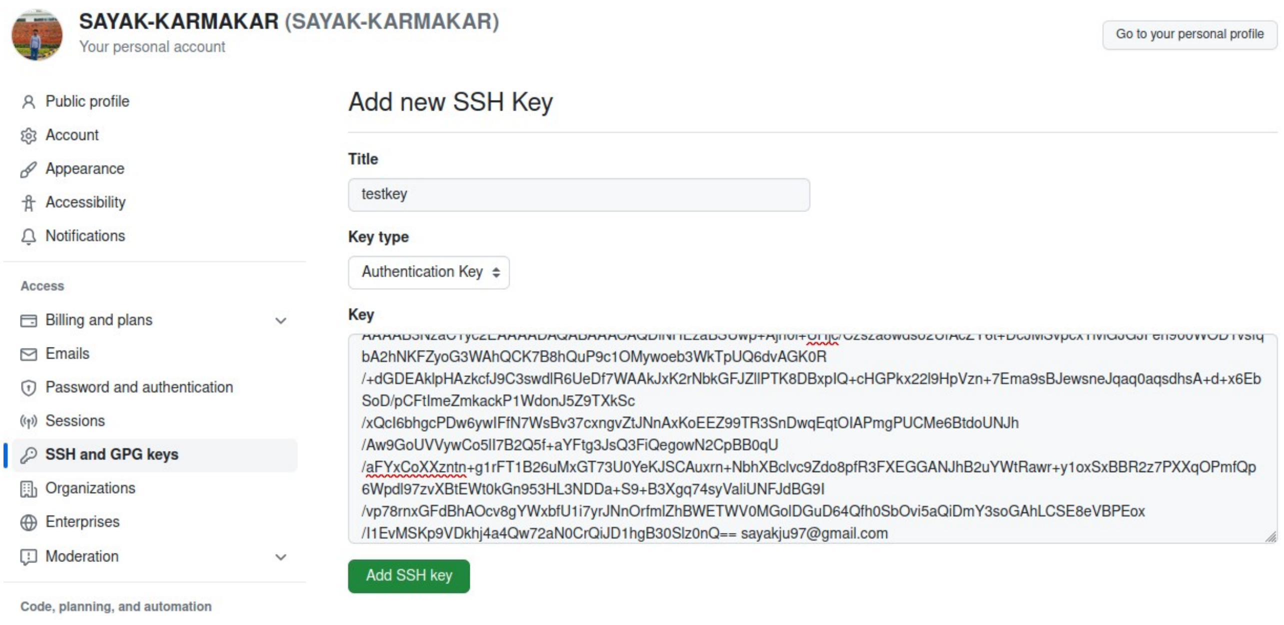
#entering passphrase is optional.

```
Your identification has been saved in id_rsa
Your public key has been saved in id_rsa.pub
The key fingerprint is:
SHA256:KUAucXkfGJWDqK5NbbWiLWSEIyAAVttgi3DhTSgueYw
sayakju97@gmail.com
The key's randomart image is:
+---[RSA 4096]----+
|*.=*+o.=..       |
|*o+BB + +        |
|+Bo++o . o       |
|E.=. .. ..       |
|o= . ...S        |
|  = + ..         |
| * + .           |
|. + .            |
|   .             |
+----[SHA256]-----+

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ ls | grep id_rsa
id_rsa
id_rsa.pub
```

#id_rsa.pub is the key that you are going to upload to your github interface. It is a public key,this key is visible to other public.Other one is private key, that should be kept secure in your local machine, don't share it with anybody. Public code is generated using this private key.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ cat id_rsa.pub
ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAACAQDTzkJ6piDNSZs0GC84GH
3YSzmqGly5z+/8SsqMZizAec5Qk+Y/
cqKNB6RtG23s5alHT5Aur1yhtElVkMGvfgi+p6s6dmUH1aswJ5
f4NjqHgqbyE6Q5FDxiytkZlXPZgh8leZrEvM2asr/
Ks1qONZvZqfNgiGy3CJSLOGOCNzjJTjHgC7uyaJJRVQoPD+Qck
HY8zc8sks1eekgoLGJvKKodL5g3n6RAksSrEL6w0reHjwmJNL3
Um0dElPd4eWrTel45nvfpQwYp+lbyrB4yWjJtuLrb3KH2DjRR7
KcDxncHDuCeNvRCjAmw0iPgotpG4m3UXIGv/
63LoUaxnGCIhp1Ur+hjyBu6sOhyqwP0vTUsn1LZrlXzCx1vwG1
FDaABz+P4/
wBcoff6xnJRN35btV32FhC2vx7rfqL2eI5fFDMEb04mv15rYd5
HcWjUa5a6Hp8FQTh6EJV2Zka+up7dLcVMd2wXMO14H4WnBWrIU
Lrt4FIj9Xv1esbyYIFRLXC30fOyBWl1l7iUDvD/
4TSIajr6OkiSpEHjAyy5uUvmwlV5LcjPTL3C4TR3QjYm7p4SaU
FOG0u3LGwtFsInb+mLsdqm7k56S9xwtyRzzX6PSyr/
Rahn43lryOU0qzaq+exdwDcw5nuPQTZPRSASDom+ejo7qKQHxg
Miv8v9D4KeUfr64w== sayakju97@gmail.com
#this is the public key that has been generated.
```

Now copy this key url. Go to github website > settings > SSH and GPG keys> click ‘new ssh key’ > Give title of the key and paste the key url. > click ‘Add SSH key’.


SAYAK-KARMAKAR (SAYAK-KARMAKAR)
Your personal account

Go to your personal profile

Add new SSH Key

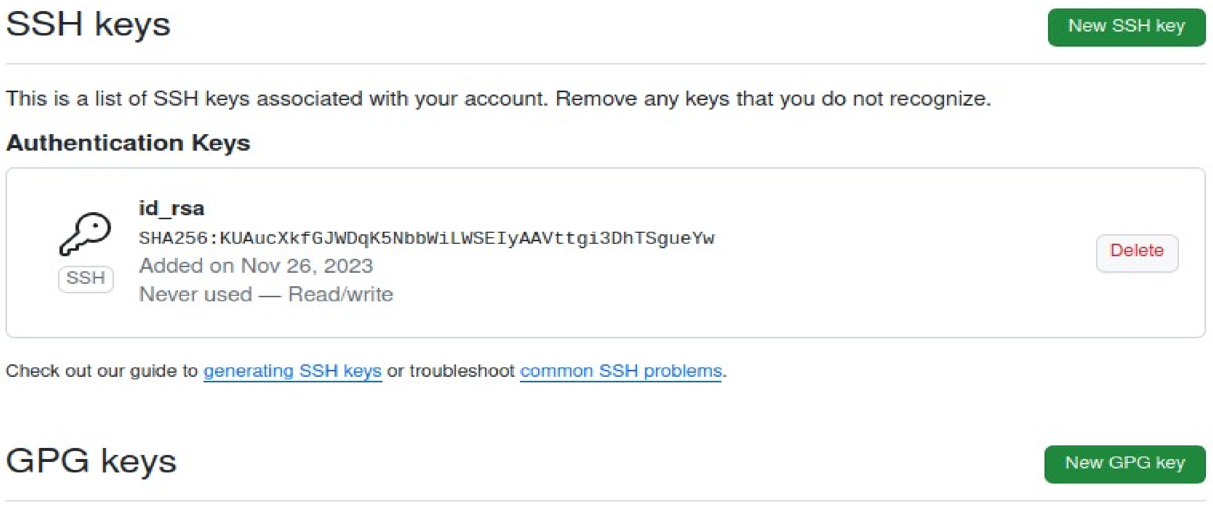
Title: testkey

Key type: Authentication Key

Key

```
[illegible]
bA2hNKFZyoG3WAhQCK7B8hQuP9c1OMywoeb3WkTpUQ6dvAGK0R
/+dGDEAklpHAzkcfJ9C3swdlR6UeDf7WAAkJxK2rNbkGFJZlIPTK8DBxplQ+cHGPkx22l9HpVzn+7Ema9sBJewsneJqaq0aqsdhsA+d+x6Eb
SoD/pCFtImeZmkackP1WdonJ5Z9TXkSc
/xQcl6bhgcPDw6ywIFfN7WsBv37cxngvZtJNnAxKoEEZ99TR3SnDwqEqtOIAPmgPUCMe6BtdoUNJh
/Aw9GoUVVywCo5lI7B2Q5f+aYFtg3JsQ3FiQegowN2CpBB0qU
/aFYxCoXXzntn+g1rFT1B26uMxGT73U0YeKJSCAuxrn+NbhXBclvc9Zdo8pfR3FXEGGANJhB2uYWtRawr+y1oxSxBBR2z7PXXqOPmfQp
6Wpdl97zvXBtEWt0kGn953HL3NDDa+S9+B3Xgq74syValiUNFJdBG9I
/vp78rnxGFdBhAOcv8gYWxbfU1i7yrJNnOrfmlZhBWETWV0MGolDGuD64Qfh0SbOvi5aQiDmY3soGAhLCSE8eVBPEox
/I1EvMSKp9VDkhj4a4Qw72aN0CrQiJD1hgB30Slz0nQ== sayakju97@gmail.com
```

Now, provide the password of you github account (i.e 74....s). Now key is successfully added. See below.


SSH keys

This is a list of SSH keys associated with your account. Remove any keys that you do not recognize.

Authentication Keys

id_rsa
SHA256:KUAucXkfGJWDqK5NbbWiLWSEIyAAVttgi3DhTSgueYw
Added on Nov 26, 2023
Never used — Read/write

Check out our guide to generating SSH keys or troubleshoot common SSH problems.

GPG keys

From here, “Check out our guide to generating SSH keys orc troubleshoot common SSH problems” click generating SSH keys > click Generating a new SSH key and adding it to the ssh-agent > now you can see the guidelines for mac, linux and windows regarding generating ssh keys.

Adding your SSH key to the ssh-agent

Now, make sure that your local git command line interface knows about the key you just generated. Adding your SSH key to the `ssh-agent` is a way to securely store and manage your private SSH keys on your local machine. The `ssh-agent` is a background process that manages authentication credentials, particularly private keys, for you. It helps you avoid entering your passphrase every time you use your private key for authentication. Use the following commands:

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ eval "$(ssh-agent -s)"
Agent pid 22694
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ ssh-add ~/.ssh/id_rsa
Identity added: /home/sayak/.ssh/id_rsa
(sayakju97@gmail.com)
```

Git push

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git branch
```

#this is the command to verify the name of the local branch you are currently on.

```
* main
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git push origin main
```

# 'origin' keyword stands for the location of the git repository. 'main' is the branch that we want to push to.

```
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 432 bytes | 432.00
KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused
0
```

```
To
https://github.com/SAYAK-KARMAKAR/GITHUB_TUTORIAL.
git
   ac1f3d8..789e78e  main -> main
```

Now you can see all the changes that are made locally in the github website also.

4. Push Locally made repo to Github:

Now let’s create a complete new repo locally (Here, demo-repo3). Its not a git repo like the previous one. Lets go to that folder from terminal.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/demo-
repo3$ git init
#initialized git repository in this folder.
hint: Using 'master' as the name for the initial
branch. This default branch name
hint: is subject to change. To configure the
initial branch name to use in all
hint: of your new repositories, which will
suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch
<name>
hint:
hint: Names commonly chosen instead of 'master'
are 'main', 'trunk' and
hint: 'development'. The just-created branch can
be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in
/home/sayak/Desktop/demo-repo3/.git/
```

```
sayak@sayak-Precision-Tower-3620:~/Desktop/demo-
repo3$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will
be committed)
        README.md

nothing added to commit but untracked files
present (use "git add" to track)
```
#untracked README.md file should be tracked. So, ‘add’ command is needed.

```
sayak@sayak-Precision-Tower-3620:~/
Desktosayak@sayak-Precision-Tower-3620:~/Desktop/
demo-repo3$ git add README.md
sayak@sayak-Precision-Tower-3620:~/Desktop/demo-
repo3$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   README.md
```
#Now file is ready to be commited.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/demo-
repo3$ git commit -m "message is created README" -
m "some description"
[master (root-commit) b92f760] message is created
README
 1 file changed, 2 insertions(+)
 create mode 100644 README.md
```
#Now the file is committed.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/demo-repo3$ git branch
* master
sayak@sayak-Precision-Tower-3620:~/Desktop/demo-repo3$ git push origin master
fatal: 'origin' does not appear to be a git repository
fatal: Could not read from remote repository.

Please make sure you have the correct access rights
and the repository exists.
```

*#This repository was completely created locally, We have not cloned it down from any git repository like the previous one. To put it live on github, "**git push origin master**" command will not work, because it is not connected to anything in GITHUB. We need to make the connection.*

Now, create an empty repository (with the same name as local repo)in github and copy the link of that repository. We will connect our local repository to it.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/demo-repo3$ git remote add origin https://github.com/SAYAK-KARMAKAR/demo-repo3.git
```

#This command is used to add a remote repository named "origin" to your local Git repository. This remote repository points empty repository on GitHub that is created just now.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/demo-repo3$ git remote -v
```

#this command shows the remote repository connected to this local repository

```
origin  https://github.com/SAYAK-KARMAKAR/demo-repo3.git (fetch)
origin  https://github.com/SAYAK-KARMAKAR/demo-repo3.git (push)
sayak@sayak-Precision-Tower-3620:~/Desktop/demo-repo3$ git push origin master
```

```
#It pushes the contents of local repo to that empty repo of github.
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 292 bytes | 292.00
KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused
0
To https://github.com/SAYAK-KARMAKAR/demo-
repo3.git
 * [new branch]      master -> master
sayak@sayak-Precision-Tower-3620:~/Desktop/demo-
repo3$ git push -u origin master
#This is done because later only "git push" command will do the same
work of "git push origin master".
Branch 'master' set up to track remote branch
'master' from 'origin'.
Everything up-to-date
```

"git push -u origin master" explained:

- git push: This is the basic command for pushing changes to a remote repository.
- `-u` or `--set-upstream`: This flag establishes a tracking relationship between your local branch and the remote branch. After setting the upstream branch, *you can simply use `git push` or `git pull` without specifying the remote and branch names.*
- `origin`: This is the name of the remote repository.
- `master`: This is the local branch you want to push.

So, `git push -u origin master` pushes the changes from your local "master" branch to the "origin" remote's "master" branch and sets up tracking for future pushes and pulls.

5. Comparison of git and GITHUB workflow:

Github Workflow :

Write code --> commit changes --> Make a pull request

`Local git workflow:`

`Write code --->stage changes (git add)--> Commit changes (git commit) --> push changes (git push)--> make a pull request`

- **Write Code:**

  This is the initial step where you write or modify code in your local project. You work on implementing new features, fixing bugs, or making improvements.

- **Stage Changes (git add):**

  After making changes to your code, you need to tell Git which changes you want to include in the next commit. This is done using the `git add` command.
  For example:

  ```
  bash
  ```

  ```
  git add file1.txt file2.cpp
  ```

  This command stages specific files (`file1.txt` and `file2.cpp` in this case) for the next commit.

- **Commit Changes (git commit):**

  Once your changes are staged, you commit them to the local repository. A commit is a snapshot of your changes with a commit message describing what you did.
  For example:

  ```
  bash
  ```

  ```
  git commit -m "Implement new feature XYZ"
  ```

This commits the staged changes with the given commit message.

- **Push Changes (git push):**

  After committing changes locally, you may want to share them with a remote repository. The `git push` command is used to push your local commits to a remote repository.
  For example:

  ```
  bash
  ```

  ```
  git push origin master
  ```

  This pushes the local commits from your "master" branch to the remote repository named "origin."

- **Make a Pull Request:**

  If you are working on a collaborative project hosted on a platform like GitHub, GitLab, or Bitbucket, you typically contribute changes through pull requests (PRs) or merge requests.
  After pushing your changes, you navigate to the repository on the platform, create a new pull request, and propose your changes for review.
  Your team members can review the changes, provide feedback, and eventually merge the pull request if everything looks good.

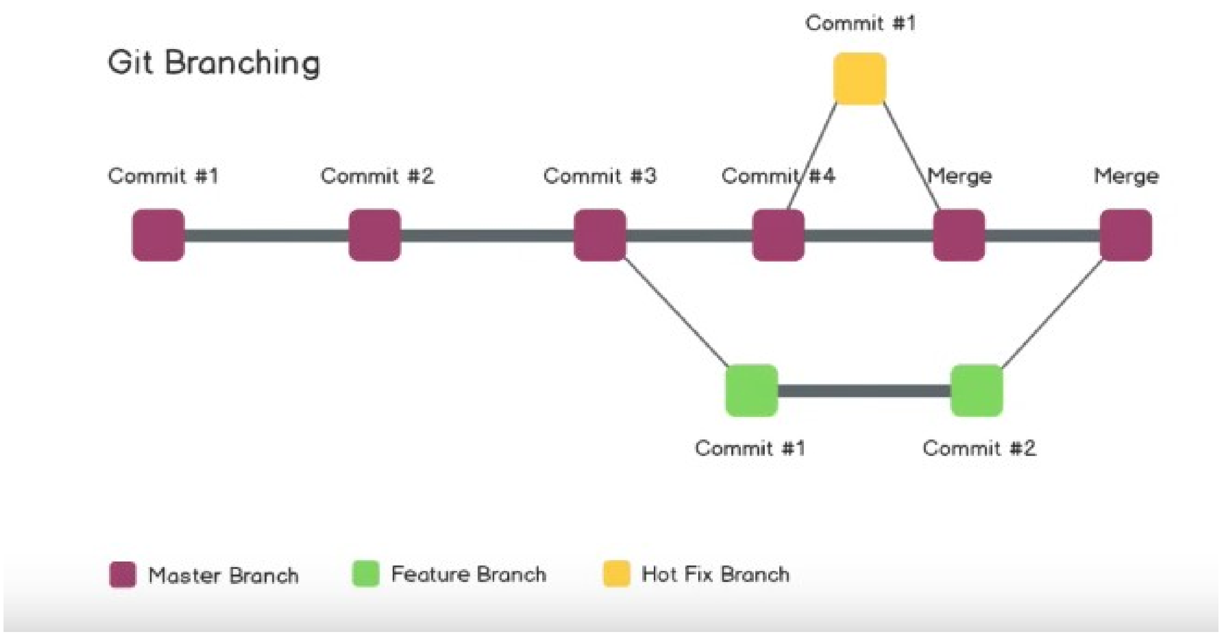
6. Git Branching

We know, ‘master’ is the naming convention of the main or default branch in a repository. Previously we had ‘master’ branch and we were commiting in the master branch only. Now we will make another branch called as feature branch.

<u>Each individual branch has no way to of knowing what commits or what changes have been made to any other branches.</u>

This is useful because you may add new features to you code, that may damage your works and you do not want to save them in the main master branch. When you feel that the works of feature bunch is okay, later you can merge it with the main branch.

There is another type, called as hotfix brunch. The purpose of a hotfix branch is to allow developers to work on a fix for a critical problem in the production code without disrupting the normal development workflow. The idea is to make a quick fix to the production code and then merge those changes back into both the main development branch and any active release branches.



```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git branch
* main
```

#Here, '' means, we are currenty in that branch.*

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git checkout -b feature-readme-
instructions
```

#Creating a new branch named as 'feature-readme-instructions'.

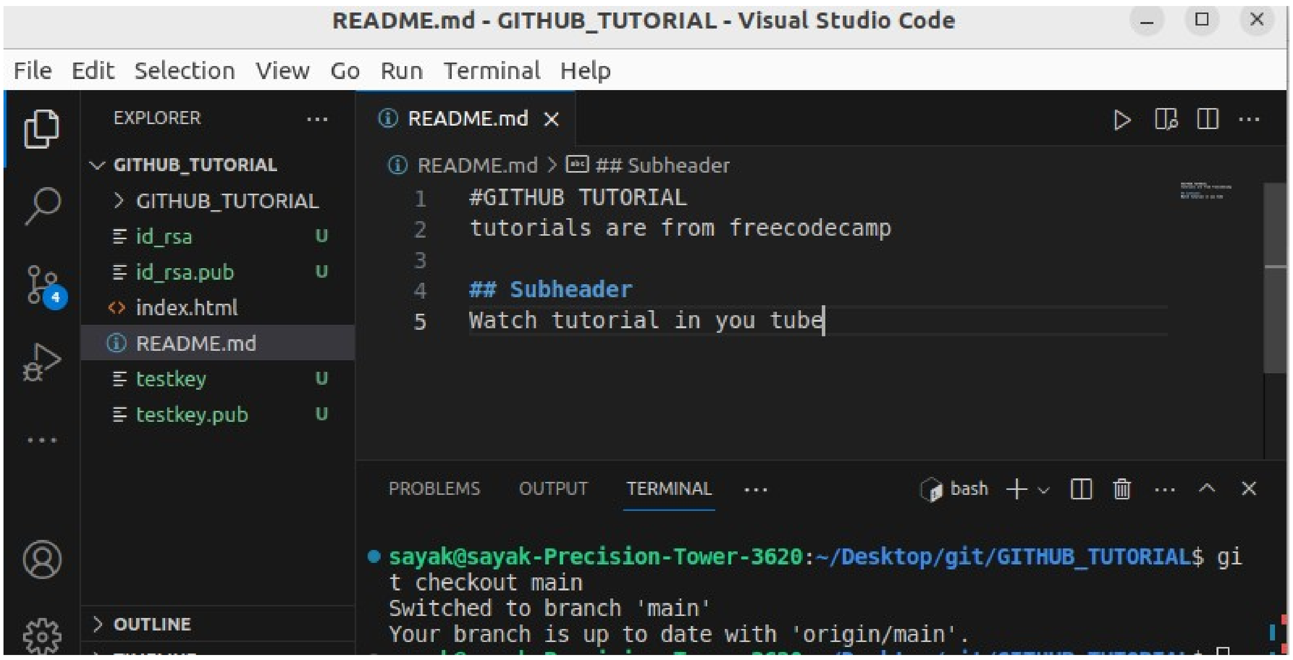
```
Switched to a new branch 'feature-readme-
instructions'
```

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git branch
* feature-readme-instructions
  main
#Now, we have two branches in this repository and we are currently in
'feature-readme-instructions' branch.

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git checkout main
#using this command, we can switch between the branches.

Switched to branch 'main'
Your branch is up to date with 'origin/main'.

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git checkout feature-readme-
instructions
Switched to branch 'feature-readme-instructions'
#Going back to feature branch
```

Now, GITHUB_TUTORIAL folder has a README.md file. Let's make some change into that.

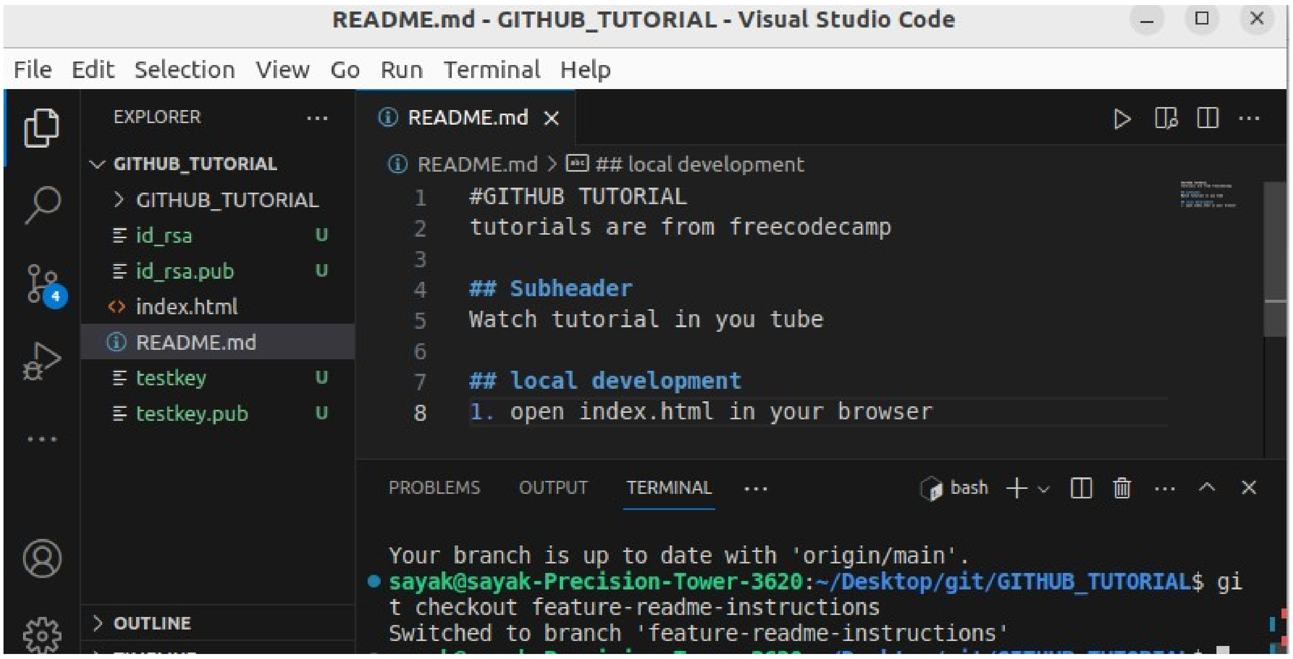
```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git status
#checking the status of the files after making the changes.
On branch feature-readme-instructions
Changes not staged for commit:
  (use "git add <file>..." to update what will be
committed)
  (use "git restore <file>..." to discard changes
in working directory)
        modified:   README.md
Untracked files:
  (use "git add <file>..." to include in what will
be committed)
        GITHUB_TUTORIAL/
        id_rsa
        id_rsa.pub
        testkey
        testkey.pub
no changes added to commit (use "git add" and/or
"git commit -a")

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git add README.md
#just added readme.md file. You may use “git add .” command to add all
files. But we are now interseted in README.md only.

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git commit -m "just updated
readme, not other files"
[feature-readme-instructions 14fa555] just updated
readme, not other files
 1 file changed, 4 insertions(+), 1 deletion(-)
```

Now, if you switch between the branches using “git checkout” command, you can see the changes are applied to feature branch only, not in the main branch.



```
#GITHUB TUTORIAL
tutorials are from freecodecamp

## Subheader
Watch tutorial in you tube
```

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$ gi
t checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```



```
#GITHUB TUTORIAL
tutorials are from freecodecamp

## Subheader
Watch tutorial in you tube

## local development
1. open index.html in your browser
```

```
Your branch is up to date with 'origin/main'.
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$ gi
t checkout feature-readme-instructions
Switched to branch 'feature-readme-instructions'
```

Now, we are in the feature branch. If we want to see the difference with main branch. See will use the following command:

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git diff main
diff --git a/README.md b/README.md
index 2c86248..f96a953 100644
--- a/README.md
+++ b/README.md
:...skipping...
diff --git a/README.md b/README.md
index 2c86248..f96a953 100644
--- a/README.md
+++ b/README.md
@@ -2,4 +2,7 @@
 tutorials are from freecodecamp

 ## Subheader
-Watch tutorial in you tube
\ No newline at end of file
+Watch tutorial in you tube
+
+## local development
+1. open index.html in your browser
:...skipping...
```

```
diff --git a/README.md b/README.md
index 2c86248..f96a953 100644
--- a/README.md
+++ b/README.md
@@ -2,4 +2,7 @@
 tutorials are from freecodecamp

 ## Subheader
-Watch tutorial in you tube
\ No newline at end of file
+Watch tutorial in you tube
+
+## local development
+1. open index.html in your browser
\ No newline at end of file
~
~
~
~
~
~
~
~
(END)...skipping...
diff --git a/README.md b/README.md
index 2c86248..f96a953 100644
--- a/README.md
+++ b/README.md
@@ -2,4 +2,7 @@
 tutorials are from freecodecamp

 ## Subheader
-Watch tutorial in you tube
\ No newline at end of file
+Watch tutorial in you tube
+
+## local development
+1. open index.html in your browser
\ No newline at end of file
~
```

```
~
~
~
~
~
~
~
~
(END)
```

We have already commited the change in README.md file locally. Now, we will push that changed file to github from ‘feature’ branch. Let’s try it.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git branch
* feature-readme-instructions
  main
#just checking , in which branch we are currently in.
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git push
fatal: The current branch feature-readme-
instructions has no upstream branch.
To push the current branch and set the remote as
upstream, use

    git push --set-upstream origin feature-readme-
instructions
#this command is not working because, we have not created upstream for feature branch. We just created it for ‘main’ branch.(”--set-upstream” and “-u” are same thing)

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$  git push -u  origin feature-
readme-instructions
#pushing changes to github from feature branch.
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
```

```
Writing objects: 100% (3/3), 425 bytes | 425.00
KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused
0
remote:
remote: Create a pull request for 'feature-readme-
instructions' on GitHub by visiting:
remote:
https://github.com/SAYAK-KARMAKAR/GITHUB_TUTORIAL/
pull/new/feature-readme-instructions
remote:
To
https://github.com/SAYAK-KARMAKAR/GITHUB_TUTORIAL.
git
 * [new branch]      feature-readme-instructions -
> feature-readme-instructions
Branch 'feature-readme-instructions' set up to
track remote branch 'feature-readme-instructions'
from 'origin'.
```

Pull request or PR is basically a request to have your code pulled into another branch. Here, We have a feature branch and we want to have our code pulled into the master branch. So we make a pull request from feature branch to master branch.

Pull Request Creation:

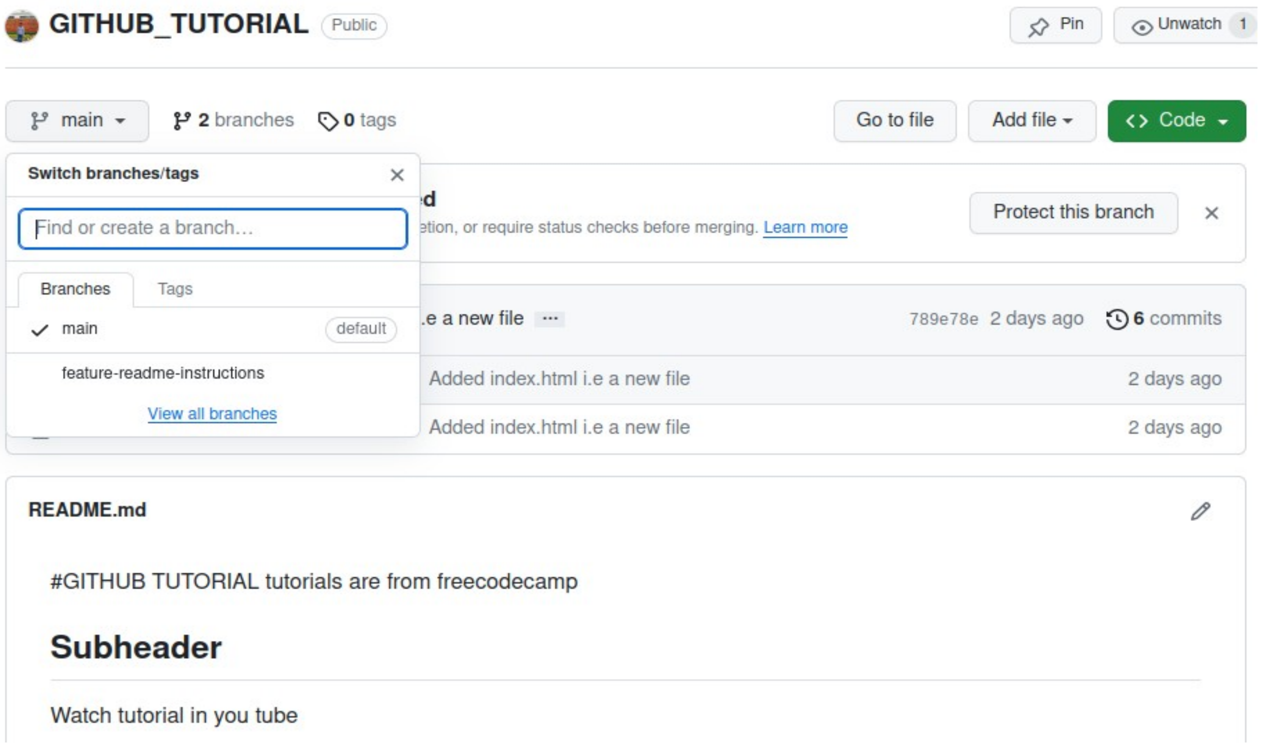
- A pull request is a formal way to propose changes from one branch to another. It's a request to merge the changes in the feature branch into the target branch (e.g., `main`).
- The developer goes to the repository hosting platform (e.g., GitHub, GitLab, Bitbucket), opens a new pull request, and selects the feature branch as the source and the main branch as the target.

Code Review:

- Team members, including other developers or leads, review the changes in the pull request. They may leave comments, ask questions, or suggest modifications.

Once the PR is merged, generally you delete your feature or source branch and switch back to your master branch.

To make apull request, go to that repo in github and click 'view all branches'

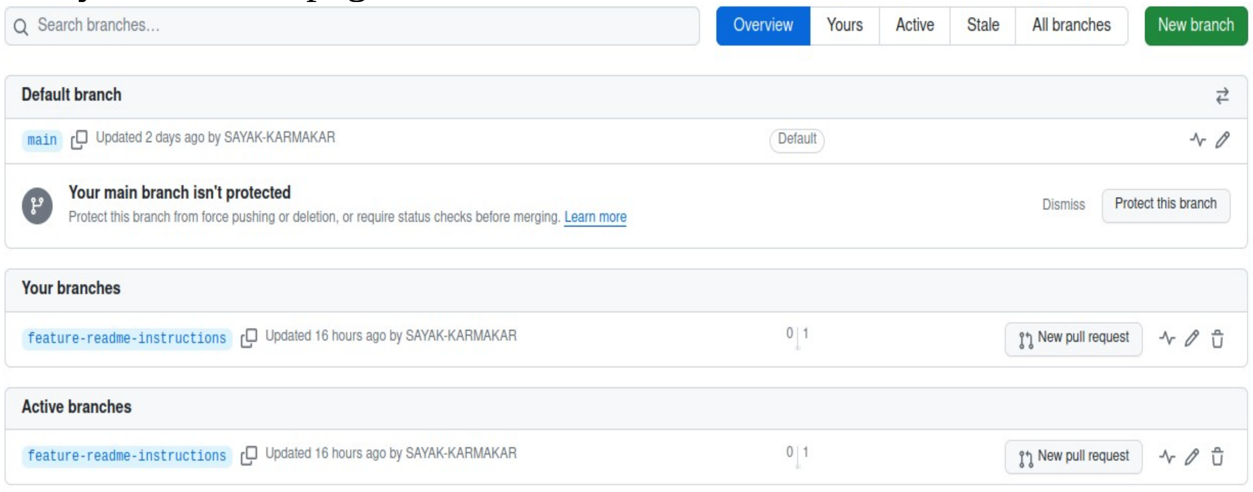


#GITHUB TUTORIAL tutorials are from freecodecamp

Subheader

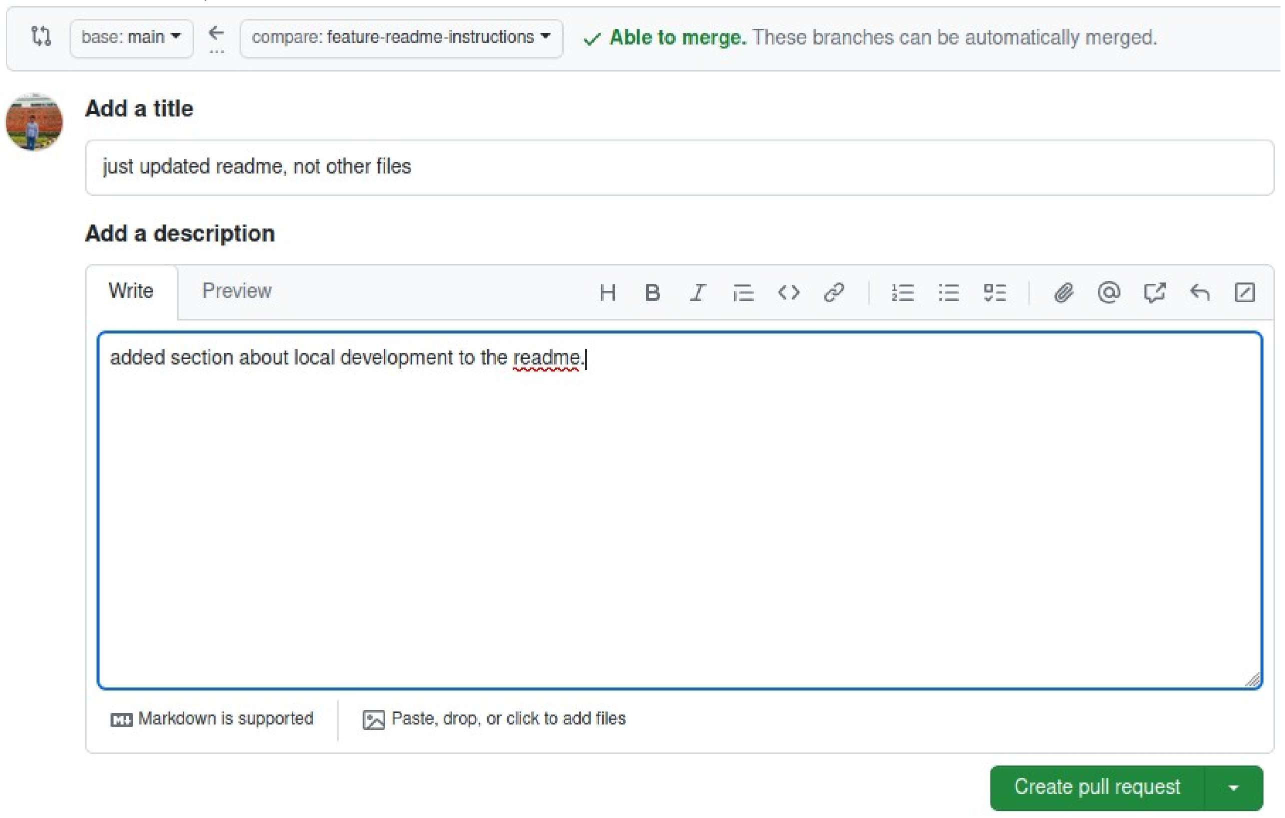
Watch tutorial in you tube

Then you can see apage like below:



Search branches...
Overview
Yours
Active
Stale
All branches
New branch
Default branch
main
Updated 2 days ago by SAYAK-KARMAKAR
Default
Your main branch isn't protected
Protect this branch from force pushing or deletion, or require status checks before merging. Learn more
Dismiss
Protect this branch
Your branches
feature-readme-instructions
Updated 16 hours ago by SAYAK-KARMAKAR
0 | 1
New pull request
Active branches
feature-readme-instructions
Updated 16 hours ago by SAYAK-KARMAKAR
0 | 1
New pull request

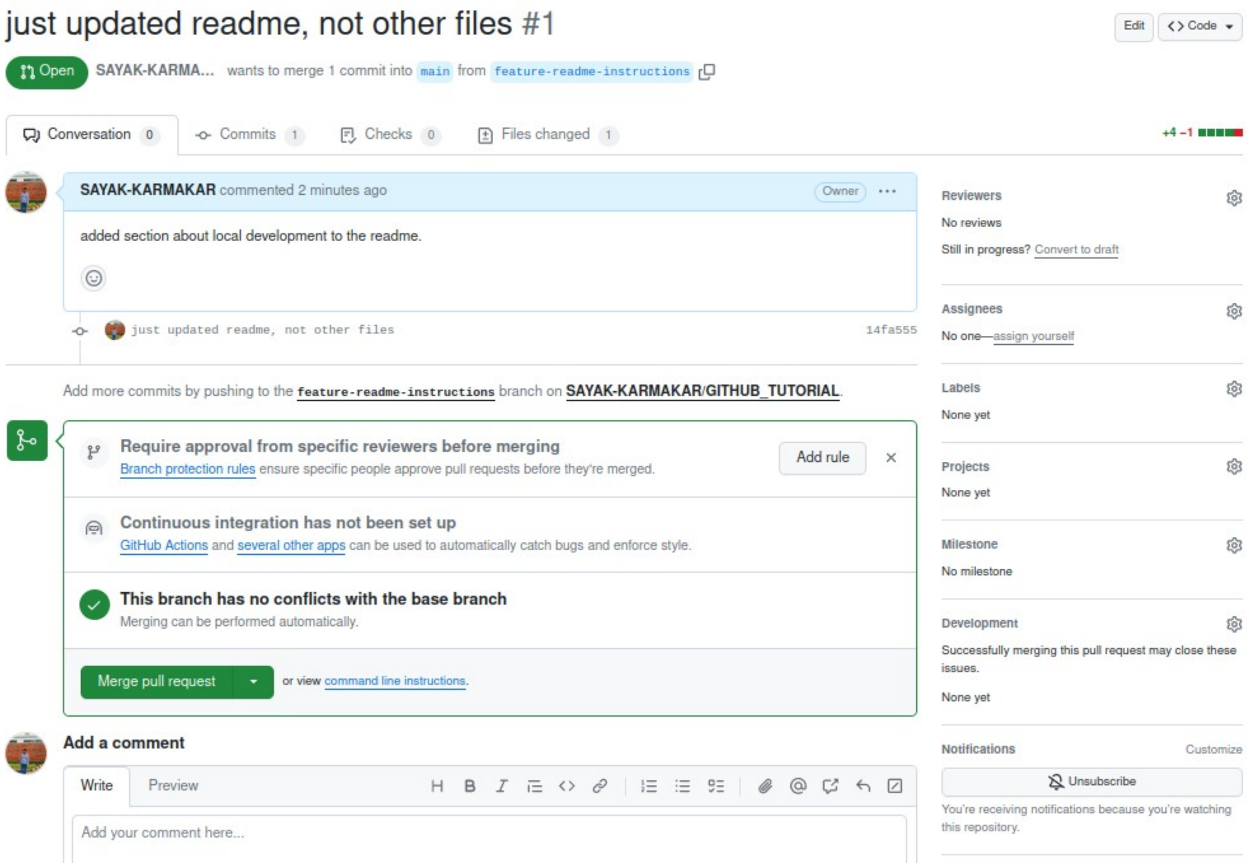
From here, create a ‘new pull request’. Then, in the next page, you can see that changes are going to merge into main branch (by seeing the direction of the arrow).


base: main
compare: feature-readme-instructions
Able to merge. These branches can be automatically merged.
Add a title
Add a description
Write
Preview
Markdown is supported
Paste, drop, or click to add files
Create pull request

just updated readme, not other files

added section about local development to the readme.

Then after creating pull request, you can merge your pull request in the next page. Also you have several options like conservations, commitschecks etc. Now click ‘merge pull request’ option.


just updated readme, not other files #1

Open SAYAK-KARMA... wants to merge 1 commit into `main` from `feature-readme-instructions`

Conversation 0 · Commits 1 · Checks 0 · Files changed 1 · +4 −1

SAYAK-KARMAKAR commented 2 minutes ago (Owner)

added section about local development to the readme.

just updated readme, not other files 14fa555

Add more commits by pushing to the `feature-readme-instructions` branch on SAYAK-KARMAKAR/GITHUB_TUTORIAL.

Require approval from specific reviewers before merging
Branch protection rules ensure specific people approve pull requests before they're merged.

Continuous integration has not been set up
GitHub Actions and several other apps can be used to automatically catch bugs and enforce style.

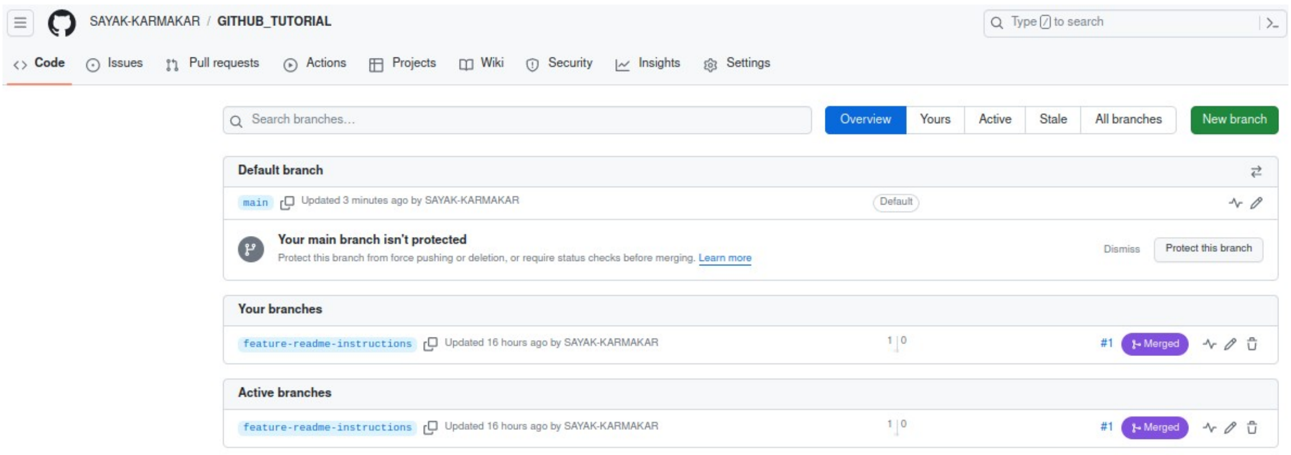
This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request — or view command line instructions.

Add a comment

Reviewers: No reviews. Still in progress? Convert to draft
Assignees: No one—assign yourself
Labels: None yet
Projects: None yet
Milestone: No milestone
Development: Successfully merging this pull request may close these issues. None yet
Notifications: Unsubscribe. You're receiving notifications because you're watching this repository.

Now, if you go to code section and check for branch, you can see only main branch is present now and feature branch is already merged.


SAYAK-KARMAKAR / GITHUB_TUTORIAL

Default branch
`main` Updated 3 minutes ago by SAYAK-KARMAKAR (Default)

Your main branch isn't protected
Protect this branch from force pushing or deletion, or require status checks before merging. Learn more

Your branches
`feature-readme-instructions` Updated 16 hours ago by SAYAK-KARMAKAR 1 | 0 #1 Merged

Active branches
`feature-readme-instructions` Updated 16 hours ago by SAYAK-KARMAKAR 1 | 0 #1 Merged

6. Merging locally:

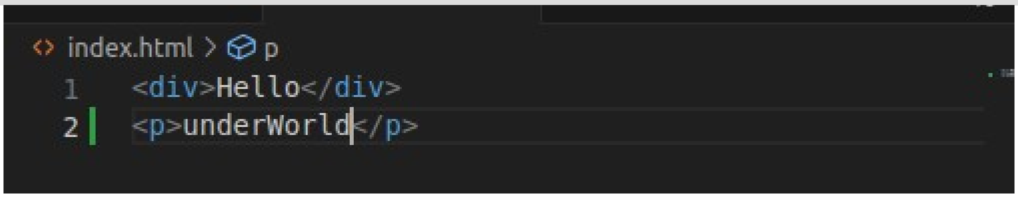
How merging has been done in github environment, not locally. If you write ‘git checkout main’ in terminal you can see that.Now, go to main branch and pull the changes.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git pull
# main brach takes all changes that are made to the feature branch.
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0),
pack-reused 0
Unpacking objects: 100% (1/1), 658 bytes | 658.00
KiB/s, done.
From
https://github.com/SAYAK-KARMAKAR/GITHUB_TUTORIAL
   789e78e..76d2ffd  main       -> origin/main
Updating 789e78e..76d2ffd
Fast-forward
 README.md | 5 ++++-
 1 file changed, 4 insertions(+), 1 deletion(-)

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git branch -d feature-readme-
instructions
Deleted branch feature-readme-instructions (was
14fa555).
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git branch
* main
#Now, feature branch got deleted. Only main branch is there.
```

7. Merge Conflict

If multiple people works on different branches for a file, while merging, git does not know which branch is to keep or which one to ignore. You need to do the manually.

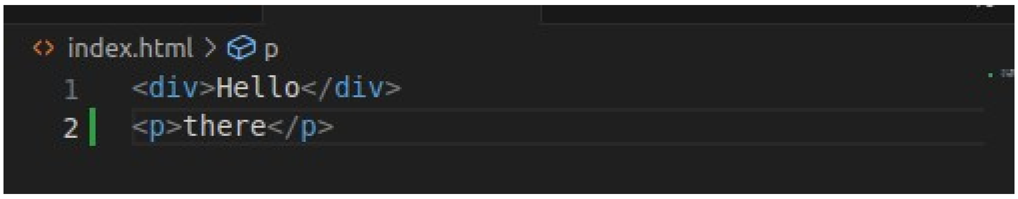
```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git checkout -b quick-test
Switched to a new branch 'quick-test'
```
# created a new branch 'quick-test' and modified the “index.html” file in that branch.



```
index.html > p
<div>Hello</div>
<p>underWorld</p>
```

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git status
On branch quick-test
Changes not staged for commit:
  (use "git add <file>..." to update what will be
committed)
  (use "git restore <file>..." to discard changes
in working directory)
        modified:   index.html
no changes added to commit (use "git add" and/or
"git commit -a")

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git commit -am "added underworld"
[quick-test 8dd0726] added underworld
 1 file changed, 1 insertion(+), 1 deletion(-)
```
# ”-am” both adds and commits at the same time. So, both operations are done only using one command line. But, remember it only works for modified files not for newly created files.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git checkout main
warning: unable to rmdir 'GITHUB_TUTORIAL':
Directory not empty
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```
#switching to the main branch and made some changes to index.html.



```
index.html > p
<div>Hello</div>
<p>there</p>
```

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git checkout quick-test
error: Your local changes to the following files
would be overwritten by checkout:
        index.html
Please commit your changes or stash them before
you switch branches.
Aborting
```

#Now see, you can not switch between the branches. Because, you have after creating the feature branch named 'quick-test', you also changed the 'main' branch. You should do commit first.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git branch
* main
  quick-test
```

#checking the current branch.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git commit -am "added there"
[main 1b4cbef] added there
 1 file changed, 2 insertions(+), 1 deletion(-)
```

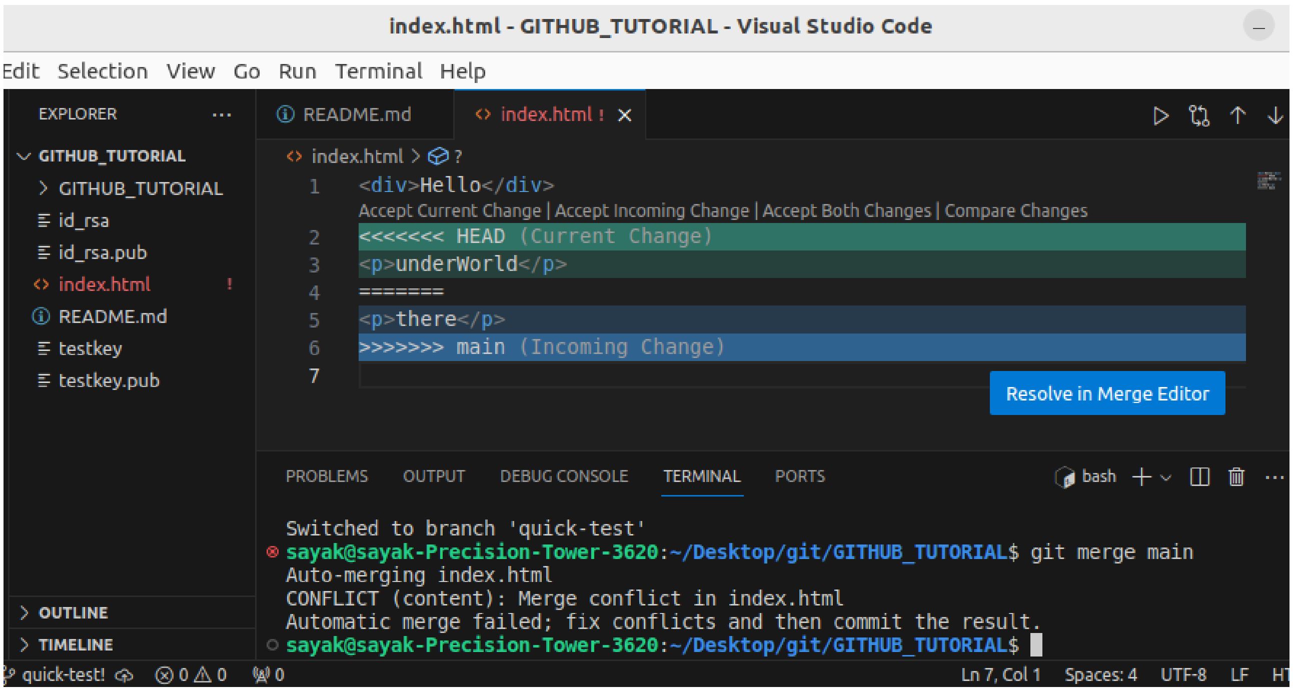
#So, you commited. Now, you can change the branch.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git checkout quick-test
Switched to branch 'quick-test'
```

#Changed to feature branch.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git merge main
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
```

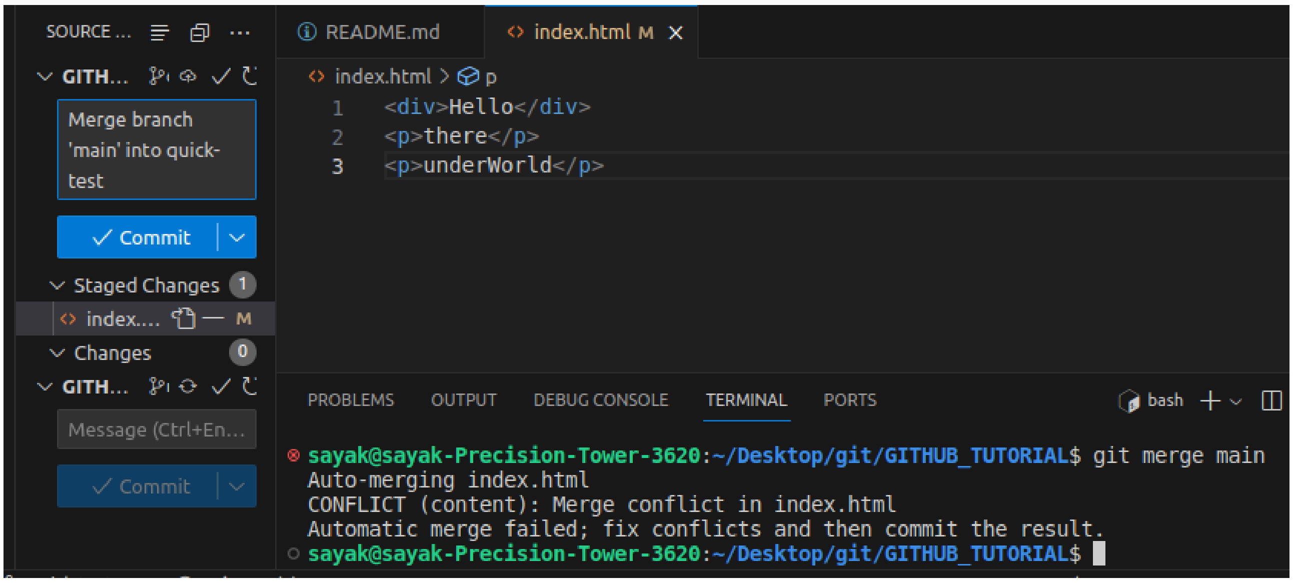
```
Automatic merge failed; fix conflicts and then
commit the result.
```
#So there is merge conflict. See the result below:


index.html - GITHUB_TUTORIAL - Visual Studio Code
Edit Selection View Go Run Terminal Help
EXPLORER
README.md
index.html !
GITHUB_TUTORIAL
GITHUB_TUTORIAL
id_rsa
id_rsa.pub
index.html
README.md
testkey
testkey.pub
OUTLINE
TIMELINE
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
Resolve in Merge Editor
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
bash
quick-test!
Ln 7, Col 1 Spaces: 4 UTF-8 LF

```
<div>Hello</div>
<<<<<<< HEAD (Current Change)
<p>underWorld</p>
=======
<p>there</p>
>>>>>>> main (Incoming Change)
```

```
Switched to branch 'quick-test'
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$ git merge main
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$
```

Now you can keep some lines or delete some as your wish from vs code editor as your own decision. You can accept current changes (cuurent branch: quick-test) or incoming changes (from main branch). Here , i accepted both. So file looks like:



```
<div>Hello</div>
<p>there</p>
<p>underWorld</p>
```

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$ git merge main
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$
```

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git commit -am "updated with
master"
[quick-test 1bed0e2] updated with master
# Made a commit after merging the changes. (I.T=> You may make a
commit from vs code also, that also gives an option)
```

8. Undoing in Git

We can undo our stages or commits.

Let’s makke some modifications in readme.md file and see how to undo the staging of a modified file. After modifications, check git status:

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git status
On branch quick-test
Changes not staged for commit:
  (use "git add <file>..." to update what will be
committed)
  (use "git restore <file>..." to discard changes
in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or
"git commit -a")
#git status shows that file is modified but not staged.

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git add README.md
```

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$ git status
On branch quick-test
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   README.md
```

#The file is staged and is ready to be committed.

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$ git reset
```

#This command undo the last action.

```
Unstaged changes after reset:
M       README.md
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$ git status
On branch quick-test
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   README.md

no changes added to commit (use "git add" and/or "git commit -a")
```

#After the "git reset" command, again the file becomes as 'not staged'.

Now, let us see how to undo a commit. Let’s first add and commit README.md file:

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git add README.md

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git commit -m "added to tell to
have fun"
[quick-test fe2b8cb] added to tell to have fun
 1 file changed, 2 insertions(+), 1 deletion(-)

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git status
On branch quick-test
nothing to commit, working tree clean

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git reset HEAD~1
Unstaged changes after reset:
M       README.md
# ’HEAD’ means the pointer to the last commit. “~1” tells git to go
further one step back from the last commit. So it will be uncommitted as
well as unstaged.

sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git diff
#The output of this command indicates changes made to the README.md
file after the last commit.
diff --git a/README.md b/README.md
index f96a953..a7d3bd9 100644
--- a/README.md
+++ b/README.md
@@ -5,4 +5,5 @@ tutorials are from freecodecamp
 Watch tutorial in you tube

 ## local development
-1. open index.html in your browser
# “-” indicates the line that was removed.
\ No newline at end of file
```

```
+1. open index.html in your browser
+2.Have fun
# "+" indicate the lines that were added.
\ No newline at end of file
```

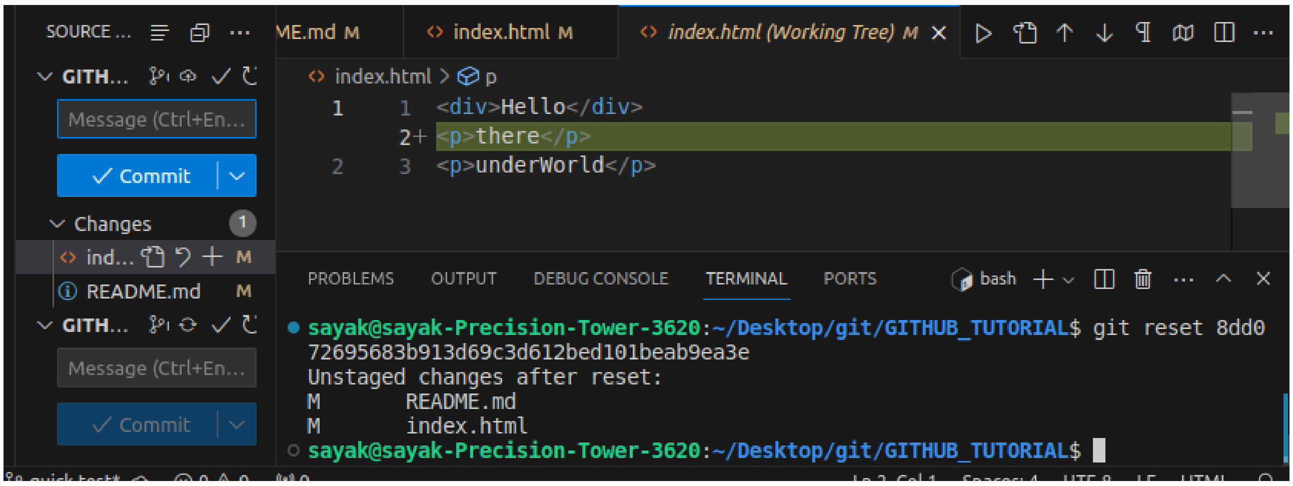
To see the logs of all of your commits use this following command:

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git log
commit 1bed0e2a19d6edd2400d18046575fa35f938e10c
#This is the unique identifier (hash) of the commit. It uniquely identifies
this specific commit in the Git history.
 (HEAD -> quick-test)
Merge: 8dd0726 1b4cbef
#This line shows that this commit is a merge commit. It merged changes
from two parent commits with the hash 8dd0726 and 1b4cbef.
Author: SAYAK-KARMAKAR <sayakju97@gmail.com>
Date:   Mon Nov 27 13:09:26 2023 +0530

    updated with master

commit 1b4cbefe81efbb8c09f8fcb0fe2875798a489afb
(main)
Author: SAYAK-KARMAKAR <sayakju97@gmail.com>
Date:   Mon Nov 27 12:54:00 2023 +0530

    added there

commit 8dd072695683b913d69c3d612bed101beab9ea3e
Author: SAYAK-KARMAKAR <sayakju97@gmail.com>
Date:   Mon Nov 27 12:33:33 2023 +0530

    added underworld

commit 92a7a35dd93405973ad8dfeaaa65e48b7c1f9313
Author: SAYAK-KARMAKAR <sayakju97@gmail.com>
Date:   Mon Nov 27 12:24:51 2023 +0530

    test commit
```

```
commit 76d2ffd4a8cfd471442caca1b1e4882145932ce9
(origin/main, orig
```

If you want to unstage any change to a file after a specific commit, you may use ‘hash’ of that commit with the following command:

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git reset
8dd072695683b913d69c3d612bed101beab9ea3e
#’hash’ of the commit message ‘added underworld’.
Unstaged changes after reset:
M       README.md
M       index.html
#unstaged or not saved in git, but changes are
still visible in the file. See below:
```

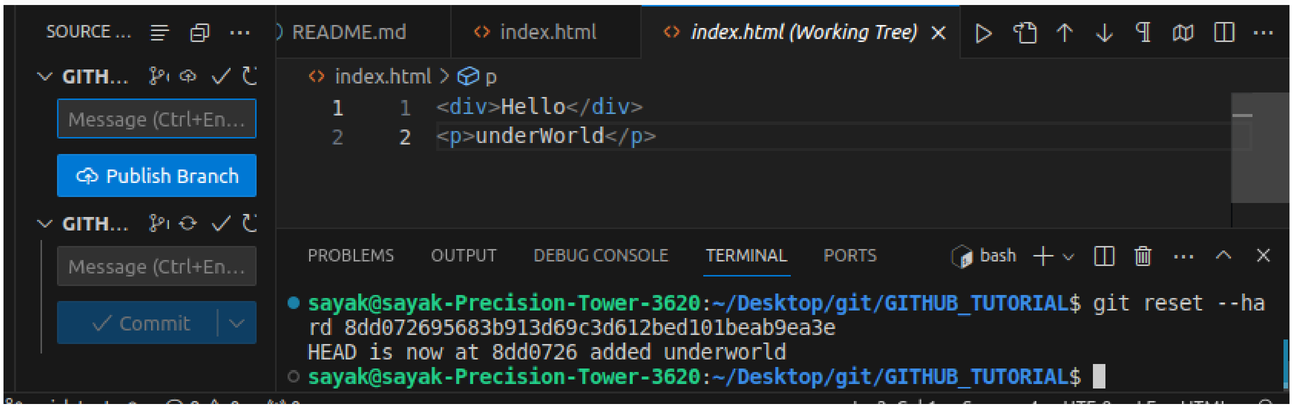


```
<div>Hello</div>
<p>there</p>
<p>underWorld</p>
```

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$ git reset 8dd0
72695683b913d69c3d612bed101beab9ea3e
Unstaged changes after reset:
M       README.md
M       index.html
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$ 
```

But, if you want to remove it completely then use the following command:

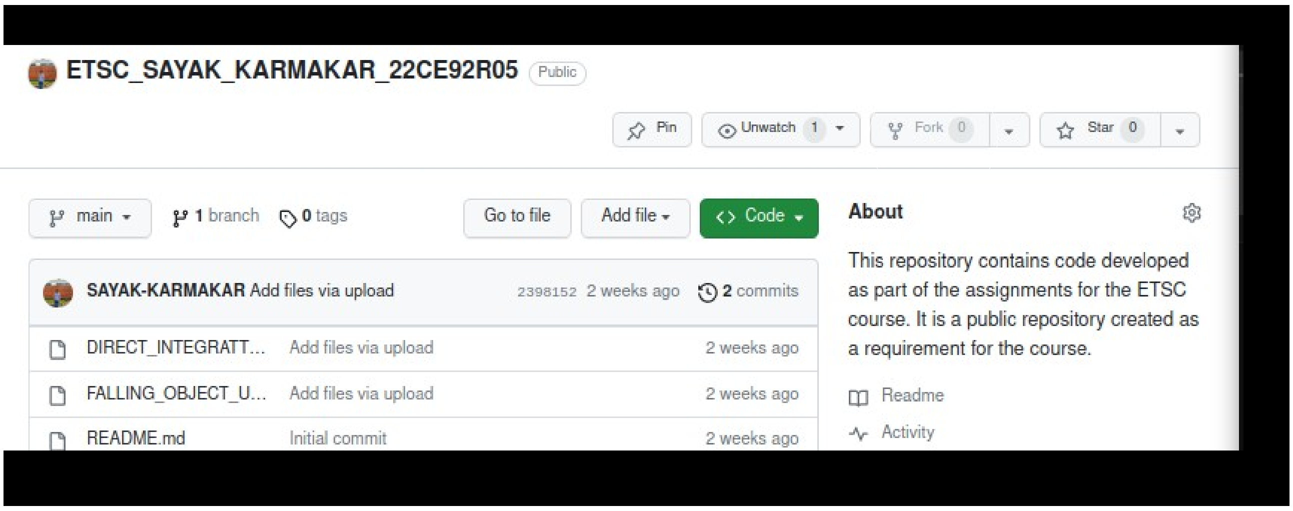
```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/
GITHUB_TUTORIAL$ git reset --hard
8dd072695683b913d69c3d612bed101beab9ea3e
HEAD is now at 8dd0726 added underworld

#Now, you can not see the word ‘there’ anymore.
Because it was added after
“8dd072695683b913d69c3d612bed101beab9ea3e” commit.
```



```
<div>Hello</div>
<p>underWorld</p>
```

```
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$ git reset --ha
rd 8dd072695683b913d69c3d612bed101beab9ea3e
HEAD is now at 8dd0726 added underworld
sayak@sayak-Precision-Tower-3620:~/Desktop/git/GITHUB_TUTORIAL$
```

9. Forking



SAYAK-KARMAKAR Add files via upload 2398152 2 weeks ago 2 commits

DIRECT_INTEGRATT...	Add files via upload	2 weeks ago
FALLING_OBJECT_U...	Add files via upload	2 weeks ago
README.md	Initial commit	2 weeks ago

This repository contains code developed as part of the assignments for the ETSC course. It is a public repository created as a requirement for the course.

Readme
Activity

In this above figure there is ‘fork’ option in top right corner. It makes a complete copy of the repository. If you are a member of a github organization (currently, I am not), you may ‘fork’ other people’s repo in your github account and then you can apply some changes to that copied repo and can make a pull request to send to the fellow organization members for review.

Your commits will go to the ‘dev’ branch; not in the ‘master’ or ‘main’ branch.