global information tracker

- Git is a DevOps tool used for source code management.
- It is a free and open-source version control system used to handle small to very large projects efficiently. Git is used to tracking changes in the source code, enabling multiple developers to work together on non-linear development, it is a cloud based remote repository and it is also known as tree history storage system.
- Git is created by Linus Torvald, creator of Linux in 2005
- Git Versioning: In windows – git bash
- In Linux – we can download git using yum, apt or snap to install it

LOCAL REPOSITORY SETUP

1. Set the name and email for the git to use when we commit
   - git config --global user.email "jnikshu@gmail.com"
   - git config --global user.name "nikita-123-cmyk"
2. Create a directory
3. Initialize the directory
   - git init
4. Create readme.md file
   - git add . (staging)
   - git commit (local commit)

REMOTE REPOSITORY SETUP

Create remote repository on GitHub, bitbucket or codecommit etc.
Now there are 2 types how we can use repo.

1. Local to remote integration –
   Here first we create a repo. locally on our system and then we are going to add the then we are going to add the remote repository location by running git remote add origin command and then we'll say git push so it will push our local repository content to remote repository. Then if there is any change in the remote repository, you can pull the changes by running git pull.
2. Clone repo to local- Here we can clone the code from remote repository to our local machine and then we can do changes whatever we want.

LAB – LOCAL REPOSITORY COMMIT

```
Nikita@DESKTOP-6DOKEUJ MINGW64 ~
$ cd /E

Nikita@DESKTOP-6DOKEUJ MINGW64 /E
$ mkdir primenumber/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E
$ cd primenumber/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber
$ mkdir numb

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber
$ git init

Initialized empty Git repository in E:/primenumber/.git/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber(master)
$ vim prime-number.py

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)

$ cat prime-number.py

for n in range(2, 10):
    for x in range (2 , n):
        if n%x==0:
            print(n,'equal',x,'*',n//x)
            break
        else:
            print(n,"is a prime no")


Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ ls
numb/ prime-number.py

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ ls -a
./  ../  .git/  numb/  prime-number.py
```

- **Note:**

  .git/ is a directory which maintains all the versions and history and configurations of the git repos. So here primenumber is a git repo. now

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ cd new/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/new (master)
$ touch hello.text

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/new (master)
$ ls
hello.text

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/new (master)
$ cd ..

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ cd numb/
```

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/numb (master)
$ touch numb.txt

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/numb (master)
$ vim numb.txt

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/numb (master)
$ ls
numb.txt

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/numb (master)
$ cat numb.txt
hello
q

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/numb (master)
$ git status

On branch master
No commits yet
Untracked files:
  (use "git add <file>..." to include in what will be committed
        ../new/
        ./
        ../prime-number.py

nothing added to commit but untracked files present (use "git add" to track)

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/numb (master)
$ git status

On branch master
No commits yet
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   numb.txt
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        ../new/
        ../prime-number.py


Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/numb (master)
$ cd ..

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ ls
new/  numb/  prime-number.py

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ git status
On branch master
No commits yet
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   numb/numb.txt
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        new/
        prime-number.py
```

- **Note:**

  Here by mistake I check the status in numb directory and added to the repository so now first we will come out of the directory and we will do commit .

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/numb (master)
$ cd ..
```

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ ls
new/  numb/  prime-number.py

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ git add .

warning: in the working copy of 'prime-number.py', LF will be replaced by CRLF the
next time Git touches it

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ git status

On branch master
No commits yet
Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   new/hello.text
        new file:   numb/numb.txt
        new file:   prime-number.py


Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ git commit -m "new files commiteed"

Author identity unknown
*** Please tell me who you are.
Run
  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"
to set your account's default identity.
Omit --global to set the identity only in this repository.
fatal: unable to auto-detect email address (got 'Nikita@DESKTOP-6DOKEUJ.(none)')
```

- **Here we get an error because we need to provide our email and username for the configuration of the files so we will give our email id and username so before that create git hub repo if not created.**

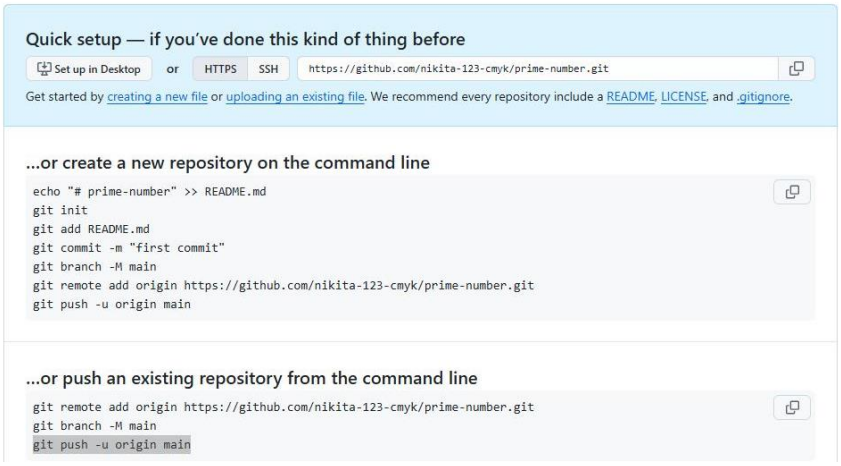
```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ git config --global user.email "jnikshu@gmail.com"

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ git config --global user.name "nikita-123-cmyk"

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ git commit -m "new files committed"

[master (root-commit) b1a8215] new files committed
 3 files changed, 21 insertions(+)
 create mode 100644 new/hello.text
 create mode 100644 numb/numb.txt
 create mode 100644 prime-number.py
```

- **so here we successfully committed the files and directories.**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
$ git status
On branch master
nothing to commit, working tree clean
```

REMOTE REPOSITORY COMMIT-

1. **Login into github account. if not created then create the account and then create a repository**
   I have created a repository by name prime-number which is public repo. github provide us 2 type of rep- public and private.
2. **Now we can login to remote repository through ssh or http. Here I used httpd link**



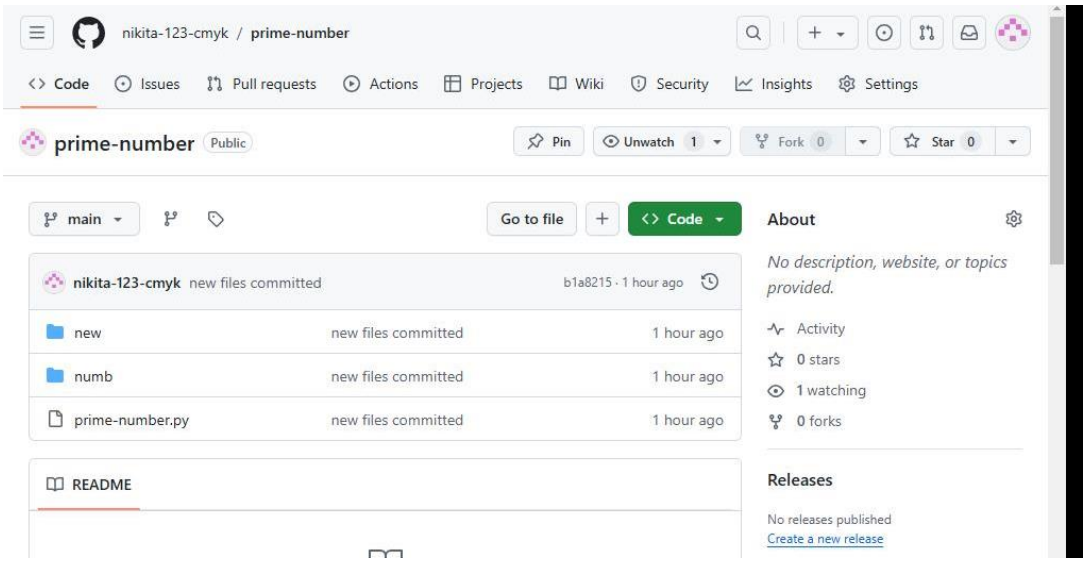
3. **We can use the commands to get login it will ask to authorize the account so after doing that push of the code will get done..**
4. **As the account is added to our local repo. it will change to main branch.**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber
(master)
$ git remote add origin https://github.com/nikita-
123-cmyk/prime-number.git

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ cat .git/config
[core]
        repositoryformatversion = 0
        filemode = false
        bare = false
        logallrefupdates = true
        symlinks = false
        ignorecase = true
[remote "origin"]
        url = https://github.com/nikita-123-cmyk/prime-
number.git
        fetch = +refs/heads/*:refs/remotes/origin/*
[branch "main"]
        remote = origin
        merge = refs/heads/main
```

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (main)
$ git branch -M main

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (main)
$ git push -u origin main
info: please complete authentication in your browser...
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (7/7), 606 bytes | 20.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/nikita-123-cmyk/prime-number.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (main)
$ git push -u origin main
info: please complete authentication in your browser...
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (7/7), 606 bytes | 20.00 KiB/s, done.
Total 7 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/nikita-123-cmyk/prime-number.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

❖ **NOTE:**

Here it asked me to first authorize the GitHub either through password and username or through browser. select accor… and authorize it .



❖ **Now we can see the files and folders are pushed to GitHub account .**

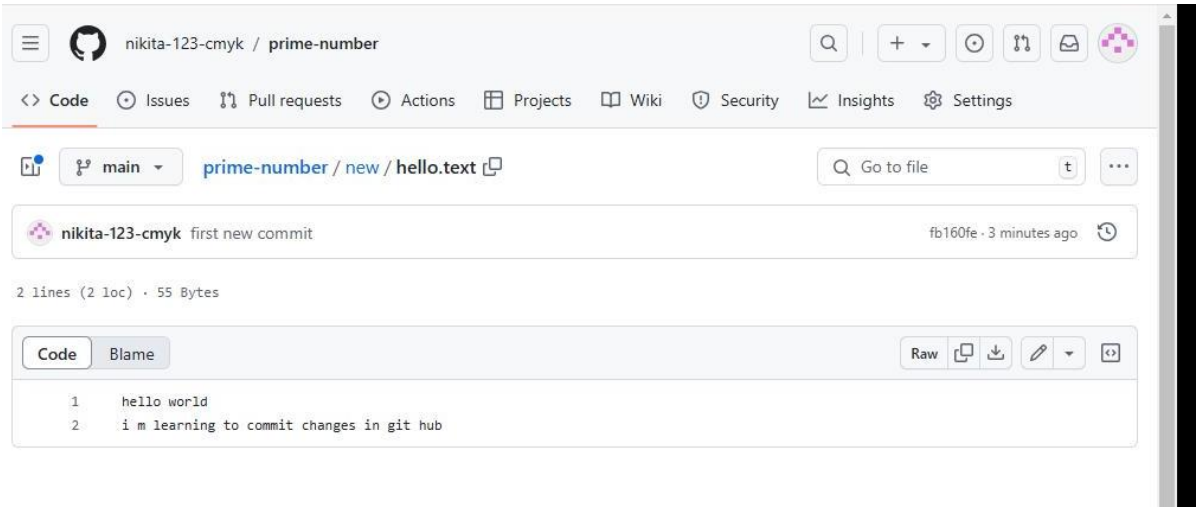
MORE COMMITS IN REPO. AFTER ADDITION OF CONTENT.

- **Now if we want to commit some changes in our file suppose in new folder, we created a file and given text so let's see how we did that**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ ls
new/  numb/  prime-number.py

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ cd new/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/new (main)
$ ls
hello.text

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/new (main)
$ cat hello.text

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/new (main)
$ cat >> hello.text
hello world
i m learning to commit changes in git hub
```

- **Here we added the content to the file and we will commit it**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/new (main)
$ ls
hello.text

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber/new (main)
$ cd ..

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ 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:   new/hello.text

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

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ git add .

warning: in the working copy of 'new/hello.text', LF will be replaced by CRLF the
next time Git touches it

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ git commit -m "first new commit"

[main fb160fe] first new commit
 1 file changed, 2 insertions(+)

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ git push origin main

Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 408 bytes | 408.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/nikita-123-cmyk/prime-number.git b1a8215..fb160fe  main -> main
```



❖ To check the commits from the command line (GIT)

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ git log

commit fb160fe87b5e2c053fa7ee52346ce41d8050f15a (HEAD -> main,
origin/main)
Author: nikita-123-cmyk <jnikshu@gmail.com>
Date:   Fri Nov 15 18:31:56 2024 +0530

    first new commit

commit b1a8215810b80e4fc9112237ddba6b771da51bcf
Author: nikita-123-cmyk <jnikshu@gmail.com>
Date:   Fri Nov 15 16:49:26 2024 +0530

    new files committed
```

❖ To check the status of commit with small ids and the content we commit

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ git log -oneline

fb160fe (HEAD -> main, origin/main) first new commit
b1a8215 new files committed

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ git show fb160fe

Author: nikita-123-cmyk <jnikshu@gmail.com>
Date:   Fri Nov 15 18:31:56 2024 +0530

    first new commit
diff --git a/new/hello.text b/new/hello.text
index e69de29..49c3759 100644
--- a/new/hello.text
+++ b/new/hello.text

@@ -0,0 +1,2 @@
+hello world
+i m learning to commit changes in git hub
```

❖ Now we pull the commit changes done in repository and shown through command line what we pull :

Through GitHub account and folder numb I deleted the content written there and then I changed the content of the file numb.text then I pulled the content from cmd and check the log and status of the file deletion and addition is shown through + and - sign

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ git pull

remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (4/4), 1.01 KiB | 3.00 KiB/s, done.
From https://github.com/nikita-123-cmyk/prime-number
   fb160fe..52c619f  main       -> origin/main
Updating fb160fe..52c619f
Fast-forward
 numb/numb.txt | 15 +--------------
 1 file changed, 1 insertion(+), 14 deletions(-)

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ git log

commit 52c619f7ecc1f256f7dabd232e7a72ba783ee921 (HEAD -> main, origin/main)
Author: nikita-123-cmyk <jnikshu@gmail.com>
Date:   Fri Nov 15 20:48:37 2024 +0530

    update numb.txt

commit fb160fe87b5e2c053fa7ee52346ce41d8050f15a
Author: nikita-123-cmyk <jnikshu@gmail.com>
Date:   Fri Nov 15 18:31:56 2024 +0530

    first new commit

commit b1a8215810b80e4fc9112237ddba6b771da51bcf
Author: nikita-123-cmyk <jnikshu@gmail.com>
Date:   Fri Nov 15 16:49:26 2024 +0530

    new files committed

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ git log -oneline

52c619f (HEAD -> main, origin/main) update numb.txt
fb160fe first new commit
b1a8215 new files committed

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (main)
$ git show 52c619f

commit 52c619f7ecc1f256f7dabd232e7a72ba783ee921 (HEAD -> main, origin/main)
Author: nikita-123-cmyk <jnikshu@gmail.com>
Date:   Fri Nov 15 20:48:37 2024 +0530

    update numb.txt

@@ -1,14 +1 @@
-
-iikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (master)
-$ ls -a
-./  ../  .git/  numb/  prime-number.py
-hello
-q
-[m
-
-
-
-
-
-Q
-
\ No newline at end of file
+print the prime number and take the changes
```

BRANCHES

- **Now we have a team and another team member wants to write same code with different version then he makes a new branch as we don't want to disturb this stable copy of the code.**
- **So, let's say we have a branch called as Main or Master, that is your main branch and that we will keep a stable copy over here.**
- **If there is any new change that is going to be done by a developer or for a period of time. Like in an agile environment, you will have sprints, different iteration, in iteration one or Sprint one.**
- **You may be developing some set of features and then in sprint two, you'll be developing some other features. For that period of time, we can create a new branch which will be copy of this existing branch, make the changes.**
- **we can keep making regular changes in that branch, but the main copy will be intact. So, the main code will be having the stable copy and once all the changes are done or everything is stable on the other sprint branch, you can then merge it with the main branch.**

We'll see how that works.

So, for that, we need to create some branches. We can do it graphically from GitHub or we can do it locally also. So, I do it locally.

LAB

- **Let's create a branch name prim_numb by command git branch -c <branchname>.**

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (main)
$ git branch -c prim_num
```

- **List of all branches**

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (main)
$ git branch -a

* main
  prim_num
  remotes/origin/main
```

- **LIST OF MAIN BRANCHES**

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (main)
$ ls
new/  numb/  prime-number.py
```

- **Now switch to branch we created so we checkout from main branch and if we check the list, it is same as main branch**

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (main)
$ git checkout prim_num
```

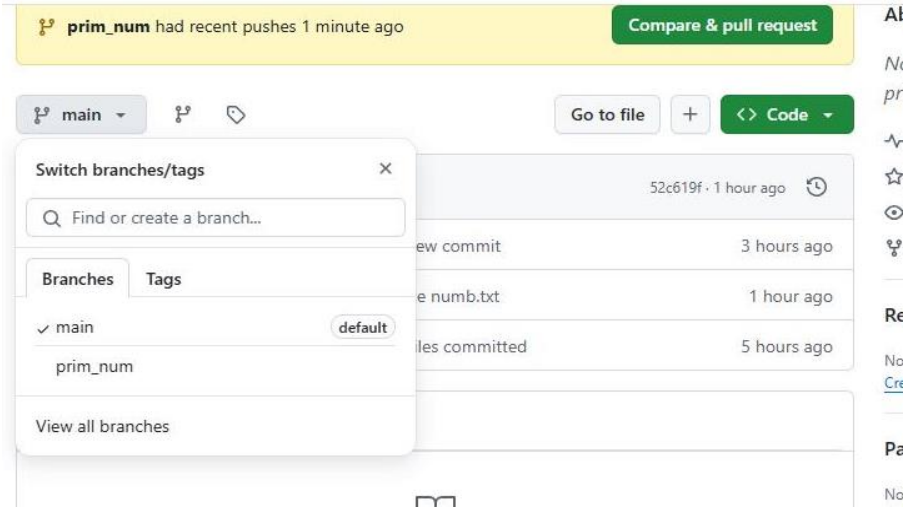
```
Switched to branch 'prim_num'
Your branch is up to date with 'origin/main'.

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (prim_num)
$ ls
new/ numb/  prime-number.py
```

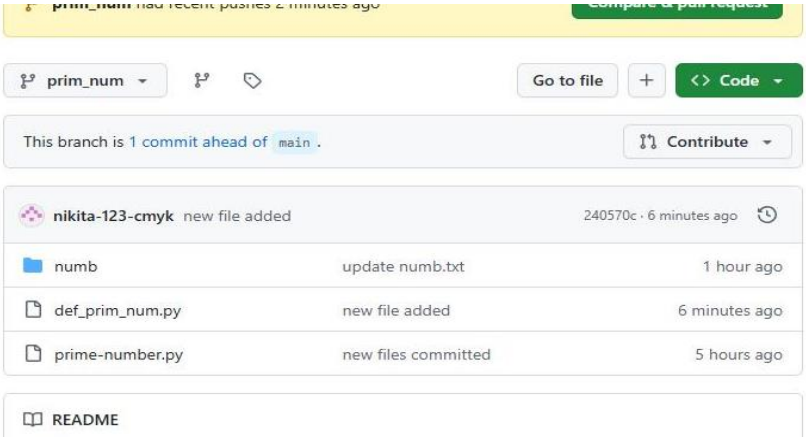
- **we Remove the folder new/ to see the change in branch and create a new file def_prim_num.py**

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (prim_num)
$ git rm -r new/
rm 'new/hello.text'

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (prim_num)
$ ls
numb/  prime-number.py

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (prim_num)
$ vim def_prim_num.py

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (prim_num)
$ ls
def_prim_num.py  numb/  prime-number.py
```

- **NOW we add the file and then commit the changes and finally push the code to remote repo to prim_num branch**

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (prim_num)
$ git add .

warning: in the working copy of 'def_prim_num.py', LF will be
replaced by CRLF the next time Git touches it

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (prim_num)
$ git commit -m "new file added"

[prim_num 240570c] new file added
 2 files changed, 15 insertions(+), 2 deletions(-)
 create mode 100644 def_prim_num.py
 delete mode 100644 new/hello.text

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (prim_num)
$ git push origin prim_num

Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 489 bytes | 163.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'prim_num' on GitHub by visiting:
remote:      https://github.com/nikita-123-cmyk/prime-
number/pull/new/prim_num
remote:
To https://github.com/nikita-123-cmyk/prime-number.git
 * [new branch]      prim_num -> prim_num
```





- **Now if we want to merge prim_num branch to main branch**
- **First we will go to main branch then we will merge the prim_num branch with main so the files present in prim_num branch will be in main branch**

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (prim_num)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (main)
$ git merge prim_num

Updating 52c619f..240570c
Fast-forward
 def_prim_num.py | 15 +++++++++++++++
 new/hello.text  |  2 --
 2 files changed, 15 insertions(+), 2 deletions(-)
 create mode 100644 def_prim_num.py
 delete mode 100644 new/hello.text

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/primenumber (main)
$ ls
def_prim_num.py  numb/  prime-number.py
```

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (prim_num)
$ git push --all origin
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/nikita-123-cmyk/prime-number.git
   52c619f..240570c  main -> main

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/primenumber (prim_num)
$ cd ..
```

- **Here we removed the repository from local machine**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E
$ rm -rf primenumber
```

- **Now we don't have any repository on local machine but it is present in remote repository**

  Now we clone the repository from remote repository so here I create a gitrepos folder in E drive and clone the repos. in that.

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E
$ mkdir gitrepos/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E
$ cd gitrepos/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos
$ git clone https://github.com/nikita-123-cmyk/prime-
number.git
Cloning into 'prime-number'...
remote: Enumerating objects: 18, done.
remote: Counting objects: 100% (18/18), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 18 (delta 0), reused 14 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (18/18), done.

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos
$ ls
prime-number/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos
$ cd prime-number/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ ls
def_prim_num.py  numb/  prime-number.py

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git branch
* main

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git branch -a
* main
  remotes/origin/HEAD -> origin/main
  remotes/origin/main
  remotes/origin/prim_num
```

GIT ROLLBACK COMMANDS

These commands are used to undoing the changes.

1. **Git revert <commit>-**
   Create new commit that undoes all of the changes made in commit, then apply it to the current branch
2. **Git reset <file>-**
   Remove files from the staging area, but leave the working directory unchanged. This unstages a file without overwriting any changes.
3. **Git clean -n –**
   Shows which files would be removed from working directory. Use the -f flag in place of the -n flag to execute the clean
4. **Git diff –**
   to see the changes made in the file before and after rollback.
   Show unstaged changes between your index and working directory.
5. **git diff –cached -**
   Show difference between staged changes and last commit.
6. **git reset --hard <commit id to which we want to rollback>**
   Reset staging area and working directory to match most recent commit and overwrites all changes in the working directory

Now we are going to see how to rollback and there are different, different levels where we can rollback.

1. **Before staging area, we rollback using command**
   git checkout
2. **After staging the file, we rollback using the Command:**
   git restore, git restore –staged <filename>
3. **After commit the file, we rollback using the Command:**
   git revert HEAD
4. **To reset every change done i.e... hard reset command used**
   git reset –hard <commit id of origin/head>

LAB

First let's edit any file and I'll just put some content over here.

- So, for that first I get into my repo. /E/gitrepos/prime-number (main)
- Then I have chosen numb/ directory and make changes in numb.txt file

```
Nikita@DESKTOP-6DOKEUJ MINGW64 ~
$ cd /E/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E
$ ls
'$RECYCLE.BIN'/                     hkhcoder/
'Imran Teli.docx'                   scrpits/
'Newproject AWS lift and shift'/    study/
'System Volume Information'/       'tarro readings'/
 Untitled-2.sh*                    'udemy devops'/
'drivers nd windows setup'/         vagrant-vms/
 git/                              'version control system.docx'
 gitrepos/                         ''$'\360\237\233\240'' Project
Name.docx'

Nikita@DESKTOP-6DOKEUJ MINGW64 /E
$ cd gitrepos/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos
$ ls
prime-number/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos
$ cd prime-number/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ ls
def_prim_num.py  numb/  prime-number.py

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ cd numb/

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number/numb (main)
$ ls
numb.txt

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number/numb (main)
$ vim numb.txt

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number/numb (main)
$ cat numb.txt

print the prime number and take the changes
hi i m doing the rollback lets checkout how we do it ..
well we will use revert command to rollback to previes stage
```

- NOTE: - Content I have inserted in my file line 2 and line 3
- Now I want to roll back the content I have inserted before getting into the staging area. So, for that we will use git checkout command here
- Before staging area we use command git checkout to rollback

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number/numb (main)
$ git checkout numb.txt
Updated 1 path from the index

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number/numb (main)
$ cat numb.txt
print the prime number and take the changes
```

- **So here we can see changes we done are gone and it is back to the previous work done.**
- **Now we will make some other changes in the same file but mistakenly I have created a new file with same name in main branch and will stage the file and then we will rollback from there but before that we will check the status**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number/numb (main)
$ cd ..

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ vim numb.text

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ vim numb.text

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ cat numb.text

hello world i have created a new file in main branch
now we stage it
and will rll back the content of the file
```

- **Content of file I want to rollback but before let's check the status of the file.**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

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

nothing added to commit but untracked files present (use "git add" to track)
```

- **This command is used to see the difference between the changes done befor and after but here mistakenly we created the new file as emply and then given some content so we cannot see any difference.**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git diff
```

- **Now we will staged our file and check the status of the file it is ready to commit but when we we used command git diff it showed us nothing because after staging, we cannot see the difference so to see the changes we made we used the command git diff –cached**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git add.

warning: in the working copy of 'numb.text', LF will be replaced by CRLF the next
time Git touches it

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ 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)
        new file:   numb.text


Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git diff

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git diff --cached



@@ -0,0 +1,3 @@
+hello world i have created a new file in main branch
+now we stage it
+and will rll back the content of the file
```

- **Now we will restore the staged commit changes and see the status of the file. We can see file is in modified stage and if we check the diff there is no changes applied shown as previously**
- **So here we rollbacked to our unstaged stage. So we used**
- **Command:**
- **Git restore , git restore –staged <filename>**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git restore --staged numb.text

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git status

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

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

nothing added to commit but untracked files present (use "git add" to track)

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git diff
```

- **Now we will commit the file and then we will rollback and see the diff and diff –cached there will be nothing to see the difference**

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git add .

warning: in the working copy of 'numb.text', LF will be replaced by CRLF the next
time Git touches it


Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git commit -m "rollback changes"

[main 507b316] rollback changes
 1 file changed, 3 insertions(+)
 create mode 100644 numb.text

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git status

On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git diff

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git diff –cached
```

- **Now we see in command diff and diff -–cached ,there is no difference shown as file is already committed, so , we will use the command git log –oneline to get the id of all the commits done and will see diff between previous commit and the current commit done**

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git log --oneline

507b316 (HEAD -> main) rollback changes
240570c (origin/prim_num, origin/main, origin/HEAD) new file added
52c619f update numb.txt
fb160fe first new commit
b1a8215 new files committed

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git diff 240570..507b316


@@ -0,0 +1,3 @@
+hello world i have created a new file in main branch
+now we stage it
+and will rll back the content of the file
```

- Now I want to roll back to the previous commit so we wil use the command git revert HEAD to make the revert

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git revert HEAD

Revert "rollback changes"

[main 76c79d9] Revert "rollback changes"
 1 file changed, 3 deletions(-)
 delete mode 100644 numb.text

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ cat numb.text
cat: numb.text: No such file or directory
```

- when I did the rollback after commit so it asked me whether I want to rollback the commit so I said okey and then it reverted the commit ..so the file I created was deleted.

- If we use the command git log –oneline we can see the file is deleted but the commit we did is seen in log and the revert commit also.

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git log -oneline

76c79d9 (HEAD -> main) Revert "rollback changes"
507b316 rollback changes
240570c (origin/prim_num, origin/main, origin/HEAD) new file added
52c619f update numb.txt
fb160fe first new commit
b1a8215 new files committed
```

- Now if we want to reset the changes made after rollback and want wo rollback from where we started the we can use the command git reset –hard <commit id of origin/head>

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git reset --hard 240570c

HEAD is now at 240570c new file added

Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ git log -oneline

240570c (HEAD -> main, origin/prim_num, origin/main, origin/HEAD)
new file added
52c619f update numb.txt
fb160fe first new commit
b1a8215 new files committed
```

- so, we can our rollback commits are deleted as it is hard rollback of doing every reset

GIT SSH LOGIN

- Till Now we have used the http URL to clone or connect with the remote repository so in this case we need to remember username and password for authentication of every login to GitHub account which is difficult and not a good way to use
- so, we will know understand the concept of ssh login with private and public ssh key.
- If you do cat to this file .git/config file, you will see there is an https URL.

```
Nikita@DESKTOP-6D0KEUJ MINGW64 /E/gitrepos/prime-number (main)
$ cat .git/config

[core]
        repositoryformatversion = 0
        filemode = false
        bare = false
        logallrefupdates = true
        symlinks = false
        ignorecase = true
[remote "origin"]
        url = https://github.com/nikita-123-cmyk/prime-number.git
        fetch = +refs/heads/*:refs/remotes/origin/*
[branch "main"]
        remote = origin
        merge = refs/heads/main
```

- Now first we need to create ssh keys to login but how?

How to generate ssh keys

- To generate SSH keys we use command ssh-keygen.exec
- It will create public and private key

```
Nikita@DESKTOP-6D0KEUJ MINGW64 ~
$ ssh-keygen.exe

Generating public/private ed25519 key pair.
Enter file in which to save the key (/c/Users/Nikita/.ssh/id_ed25519):
Created directory '/c/Users/Nikita/.ssh'.
Enter passphrase for "/c/Users/Nikita/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/Nikita/.ssh/id_ed25519
Your public key has been saved in /c/Users/Nikita/.ssh/id_ed25519.pub

Nikita@DESKTOP-6D0KEUJ MINGW64 ~
$ ls .ssh/

id_ed25519  id_ed25519.pub
```

- we will copy the public key and go to GitHub account setting and in setting go to ssh and GPG key and paste the public key over there and click generate.

```
Nikita@DESKTOP-6D0KEUJ MINGW64 ~
$ cat .ssh/id_ed25519.pub
```

Command given me my public key but I deleted it from here

- **Now we will clone the repository.**
- **Before cloning I have deleted my prime-number repo from local machine and then I cloned it which is successful if our remote repo is private then it will ask for authentication otherwise not .**

```
Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos
$ git clone git@github.com:nikita-123-cmyk/prime-number.git

Cloning into 'prime-number'...
The authenticity of host 'github.com (20.207.73.82)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 18, done.
remote: Counting objects: 100% (18/18), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 18 (delta 0), reused 14 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (18/18), done.

Nikita@DESKTOP-6DOKEUJ MINGW64 /E/gitrepos
$ ls
prime-number/
```