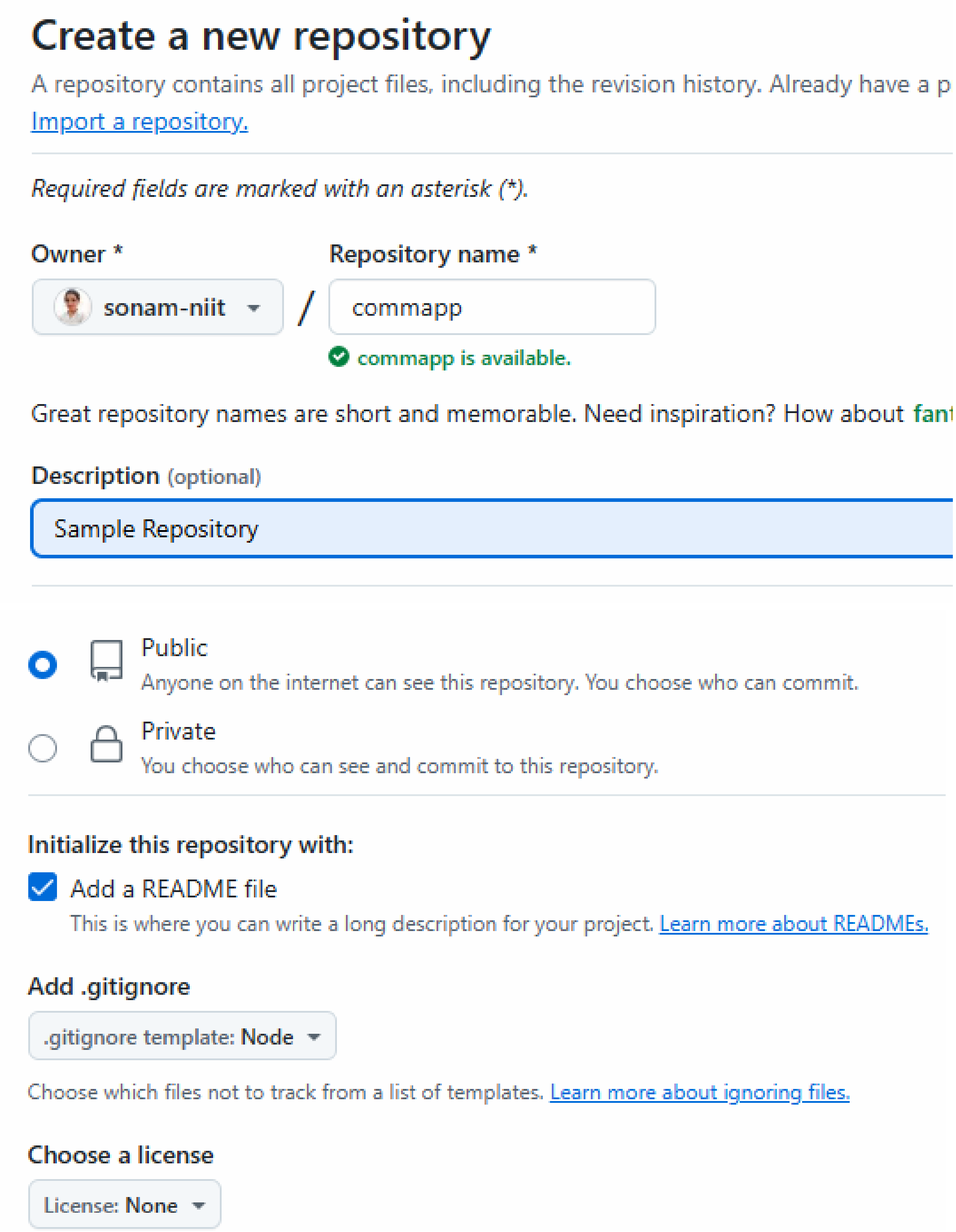
GIT and GITHUB

Clone Repository
To copy the entire repository to your system you can use git clone command
git clone repo-link
Go to Github.com and create new repo.



Create a new repository

A repository contains all project files, including the revision history. Already have a p

Import a repository.

Required fields are marked with an asterisk (*).

Owner * sonam-niit / Repository name * commapp

✓ commapp is available.

Great repository names are short and memorable. Need inspiration? How about fan

Description (optional)

Sample Repository

● Public
Anyone on the internet can see this repository. You choose who can commit.

○ Private
You choose who can see and commit to this repository.

Initialize this repository with:

☑ Add a README file
This is where you can write a long description for your project. Learn more about READMEs.

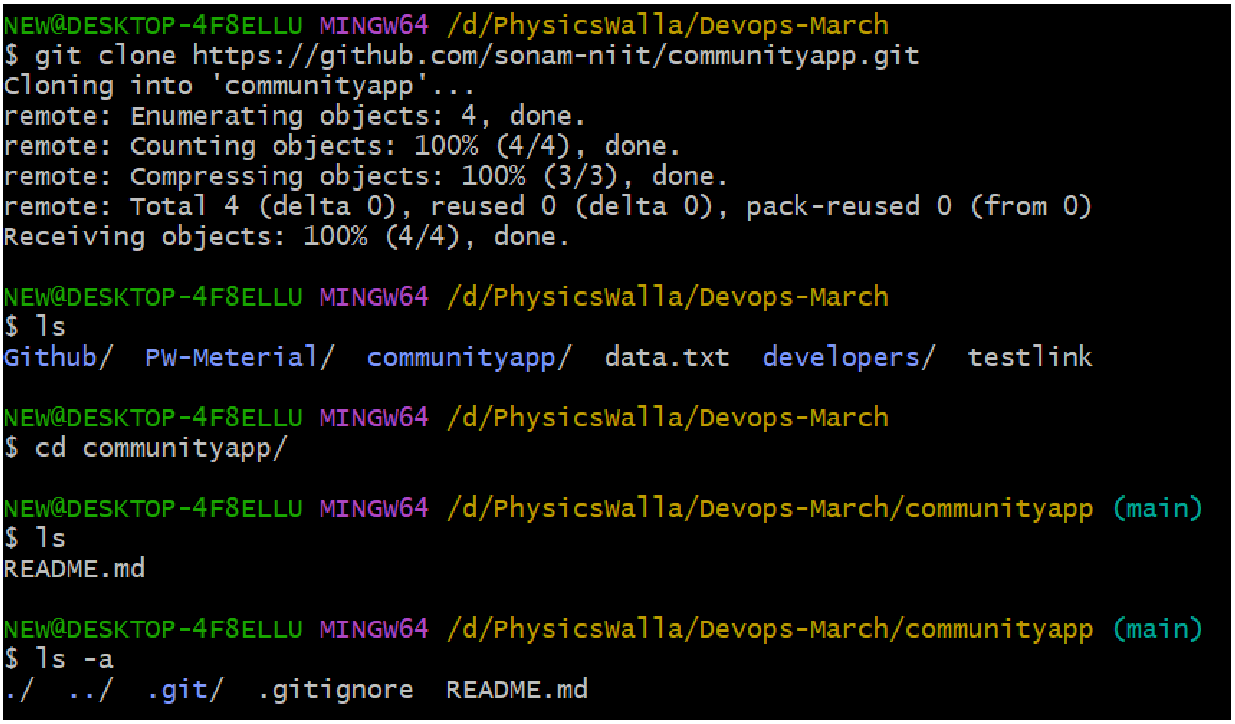
Add .gitignore

.gitignore template: Node

Choose which files not to track from a list of templates. Learn more about ignoring files.

Choose a license

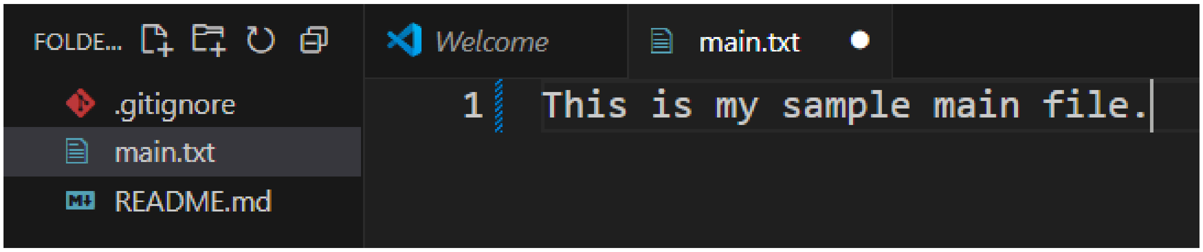
License: None



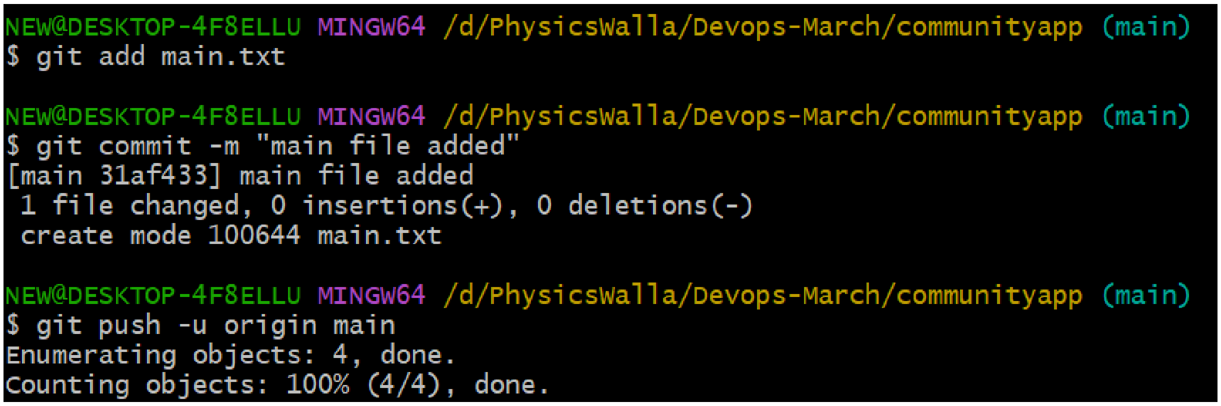
```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March
$ git clone https://github.com/sonam-niit/communityapp.git
Cloning into 'communityapp'...
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 4 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (4/4), done.

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March
$ ls
Github/  PW-Meterial/  communityapp/  data.txt  developers/  testlink

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March
$ cd communityapp/

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/communityapp (main)
$ ls
README.md

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/communityapp (main)
$ ls -a
./  ../  .git/  .gitignore  README.md
```

Inside this repo if you want to do anything just add, commit and push.

create one file main.txt add some content, save it.



```
This is my sample main file.
```

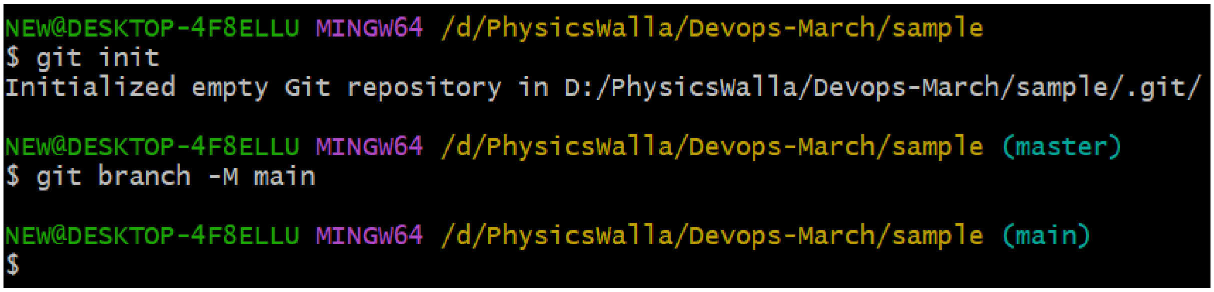


```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/communityapp (main)
$ git add main.txt

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/communityapp (main)
$ git commit -m "main file added"
[main 31af433] main file added
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 main.txt

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/communityapp (main)
$ git push -u origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
```

Git Diff command:

it is used to find differences between working directory and commit, staging and commit, between 2 heads, 2 branches

Let's create a new folder named sample and open git bash, init git repo.



```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample
$ git init
Initialized empty Git repository in D:/PhysicsWalla/Devops-March/sample/.git/

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (master)
$ git branch -M main

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$
```

open this folder in vs code.

create file file1.txt and add some content then stage it and commit it.

make changes to this file1.txt

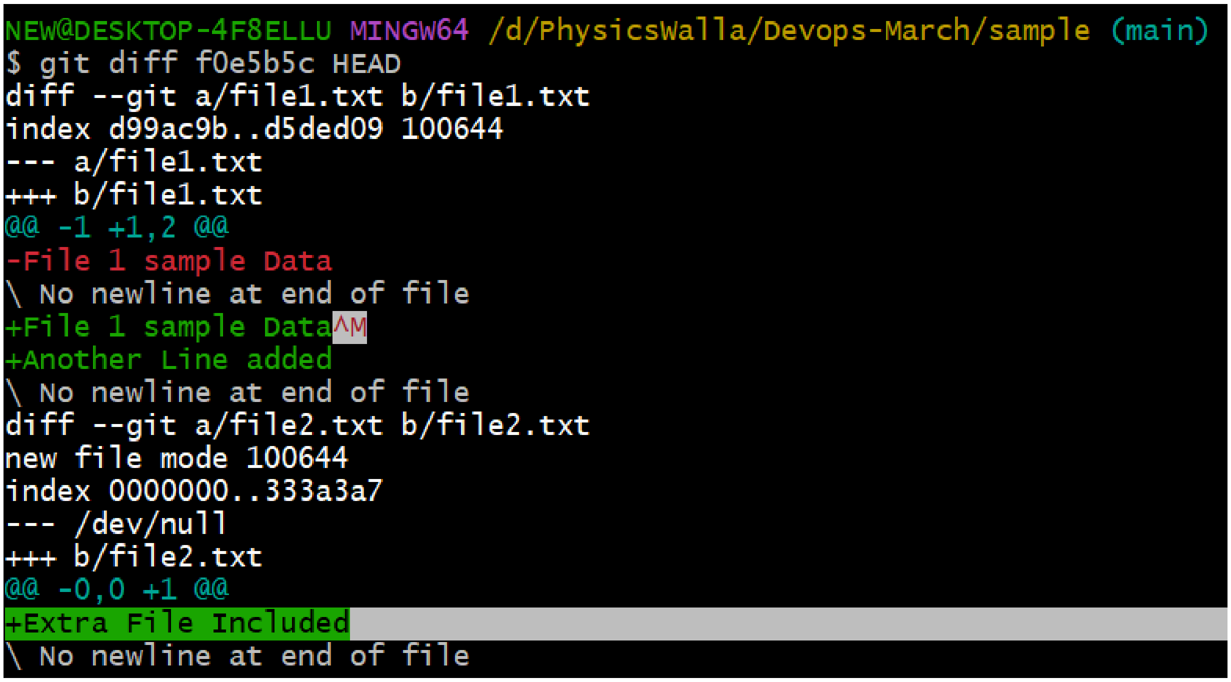
now to see the difference between you commit and current changes you can use git diff command.

Now create one more file file2.txt then stage it and commit the same.

So you have 2 commits in your repo: git log --oneline

To compare 2 commits: git diff hashcode-of-first-commit HEAD

(above will show you comparison between 2 commits



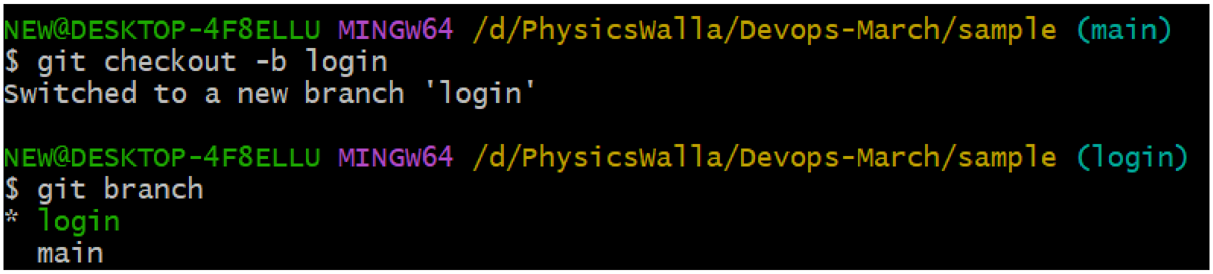
```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git diff f0e5b5c HEAD
diff --git a/file1.txt b/file1.txt
index d99ac9b..d5ded09 100644
--- a/file1.txt
+++ b/file1.txt
@@ -1 +1,2 @@
-File 1 sample Data
\ No newline at end of file
+File 1 sample Data^M
+Another Line added
\ No newline at end of file
diff --git a/file2.txt b/file2.txt
new file mode 100644
index 0000000..333a3a7
--- /dev/null
+++ b/file2.txt
@@ -0,0 +1 @@
+Extra File Included
\ No newline at end of file
```

Let's Understand Branching and Merging

I want to make experiment like implement Login Feature, to do it I want to create Login Branch

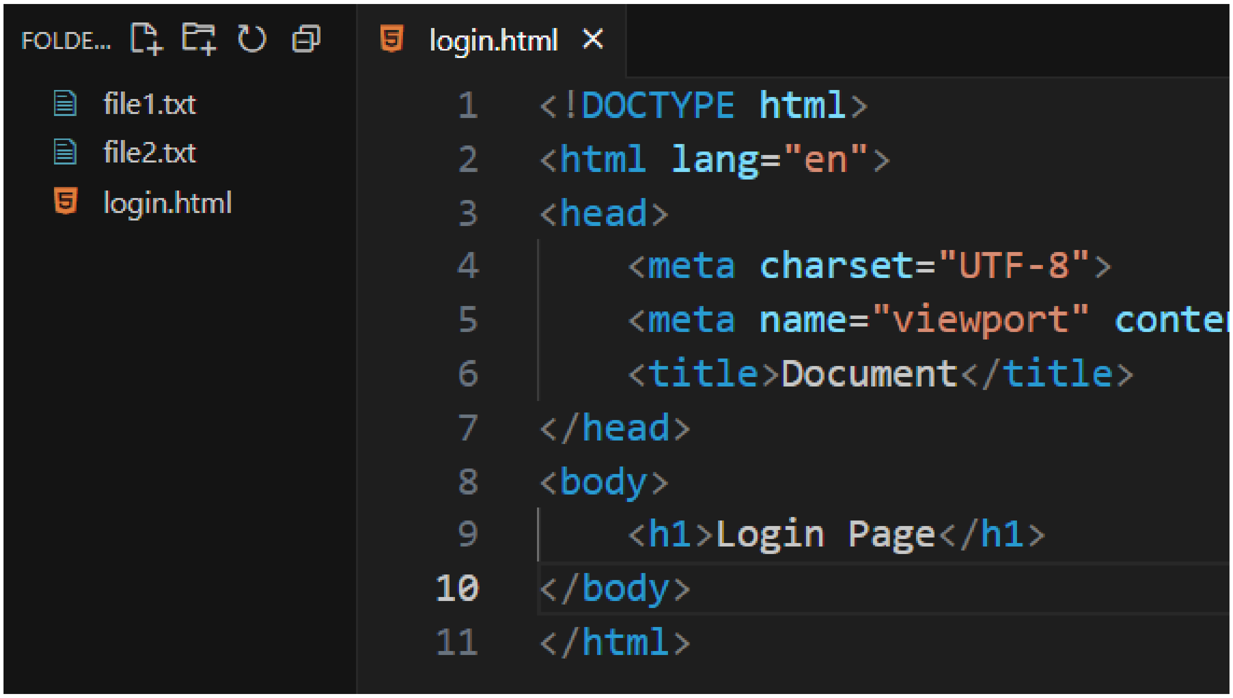
implement feature, check it, if all good merge to main and delete login branch.

Step 1: create login Branch and switch to that branch

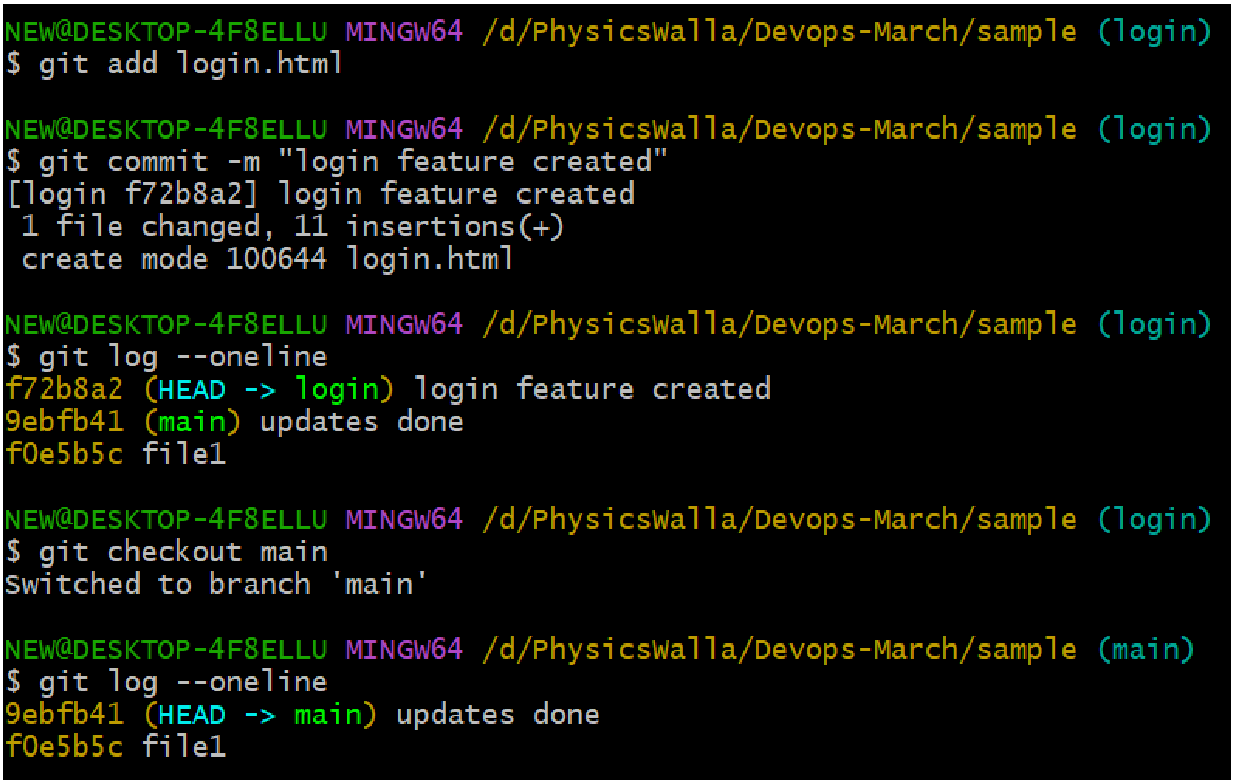


```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git checkout -b login
Switched to a new branch 'login'

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (login)
$ git branch
* login
  main
```

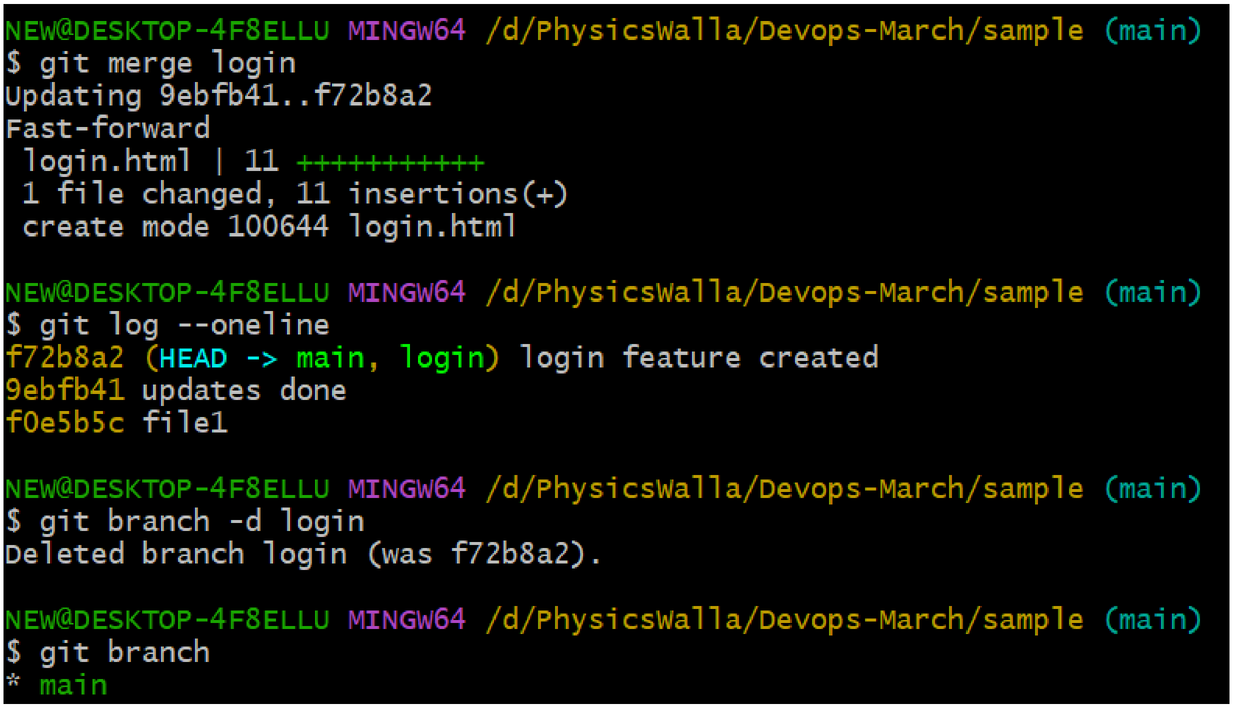
Let's Implement Login page.



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" conte
    <title>Document</title>
</head>
<body>
    <h1>Login Page</h1>
</body>
</html>
```



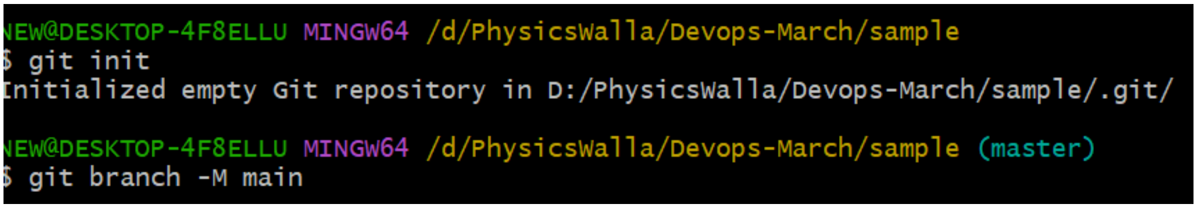
```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (login)
$ git add login.html

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (login)
$ git commit -m "login feature created"
[login f72b8a2] login feature created
 1 file changed, 11 insertions(+)
 create mode 100644 login.html

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (login)
$ git log --oneline
f72b8a2 (HEAD -> login) login feature created
9ebfb41 (main) updates done
f0e5b5c file1

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (login)
$ git checkout main
Switched to branch 'main'

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git log --oneline
9ebfb41 (HEAD -> main) updates done
f0e5b5c file1
```



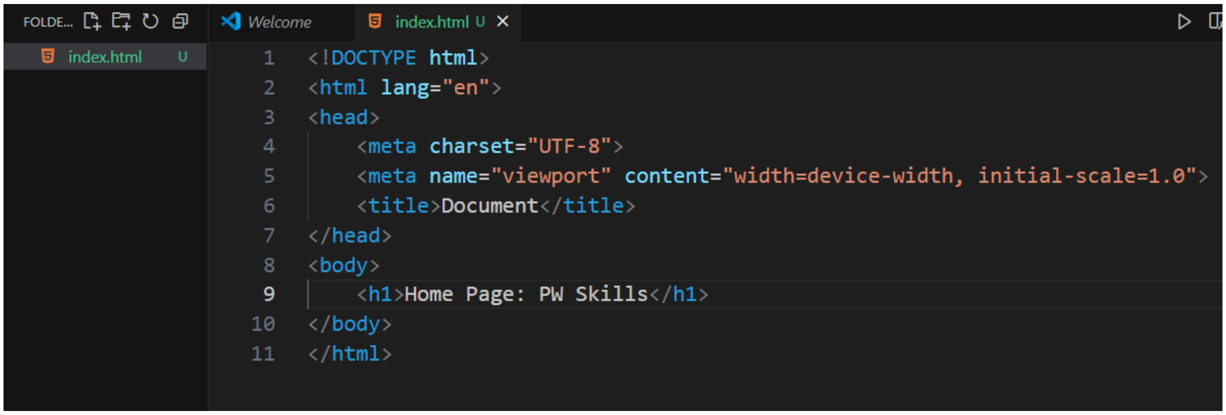
```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git merge login
Updating 9ebfb41..f72b8a2
Fast-forward
 login.html | 11 +++++++++++
 1 file changed, 11 insertions(+)
 create mode 100644 login.html

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git log --oneline
f72b8a2 (HEAD -> main, login) login feature created
9ebfb41 updates done
f0e5b5c file1

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git branch -d login
Deleted branch login (was f72b8a2).

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git branch
* main
```

Let's Understand Git Conflict.

Create Folder named sample and open it in VS Code and git bash.

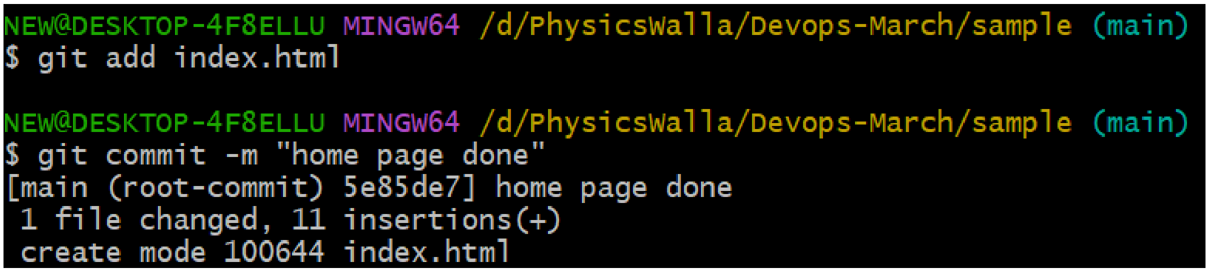


```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample
$ git init
Initialized empty Git repository in D:/PhysicsWalla/Devops-March/sample/.git/

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (master)
$ git branch -M main
```



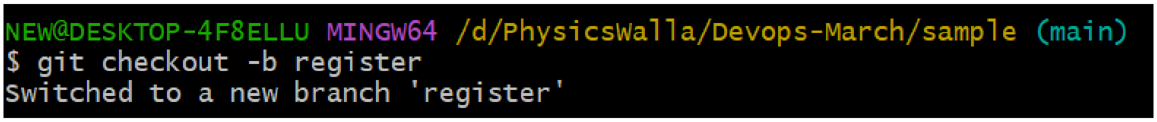
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1>Home Page: PW Skills</h1>
</body>
</html>
```

create index.html file inside main branch, stage it and commit it.



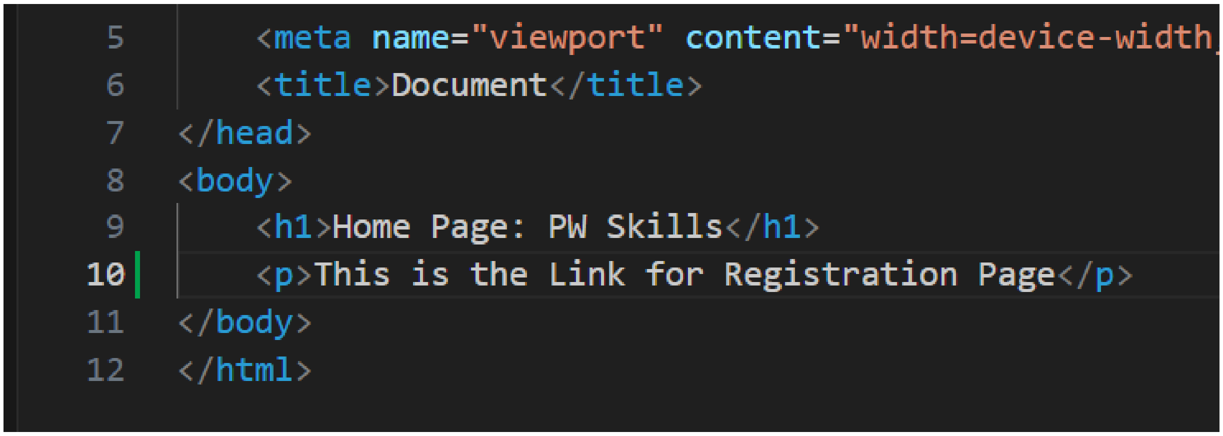
```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git add index.html

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git commit -m "home page done"
[main (root-commit) 5e85de7] home page done
 1 file changed, 11 insertions(+)
 create mode 100644 index.html
```

Now Switch to register branch and edit code for index.html to add link for Registration.



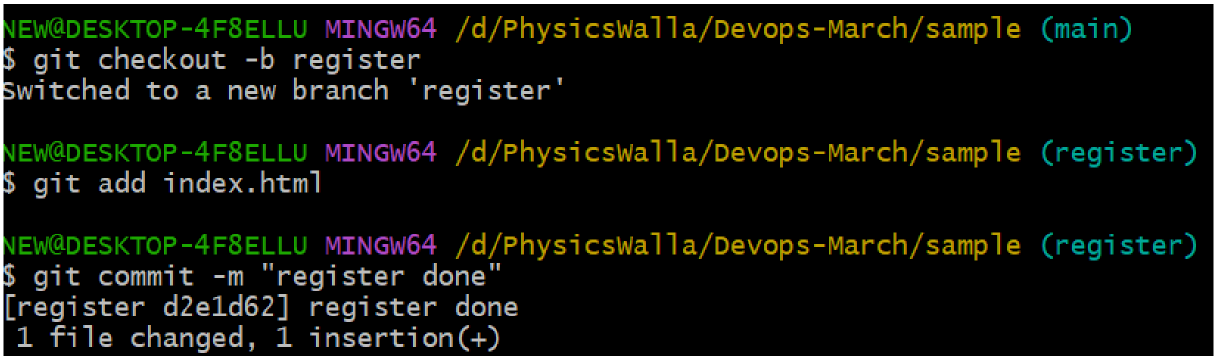
```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git checkout -b register
Switched to a new branch 'register'
```

Edit index.html as shown below.



```html
    <meta name="viewport" content="width=device-width
    <title>Document</title>
</head>
<body>
    <h1>Home Page: PW Skills</h1>
    <p>This is the Link for Registration Page</p>
</body>
</html>
```

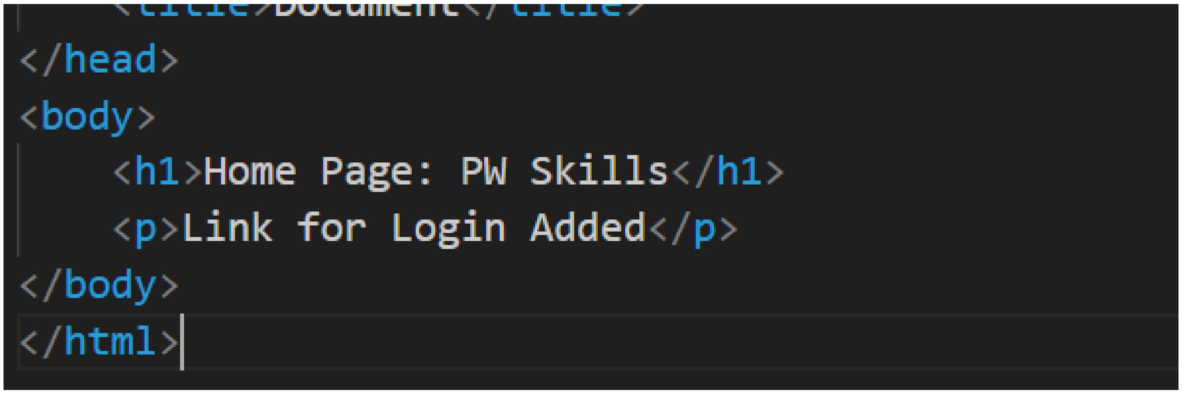
Stage changes and Commit



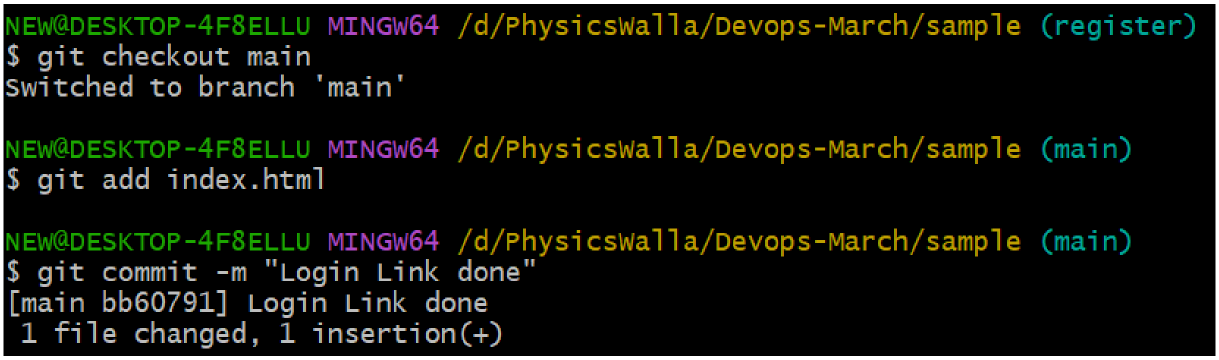
```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git checkout -b register
Switched to a new branch 'register'

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (register)
$ git add index.html

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (register)
$ git commit -m "register done"
[register d2e1d62] register done
 1 file changed, 1 insertion(+)
```

Switch Branch to main and then add link for login

Update Index.html code

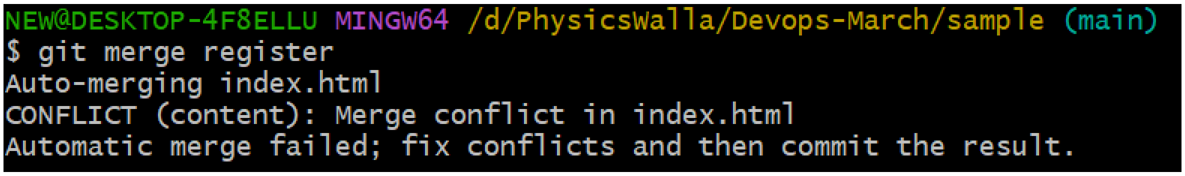


```
    <title>Document</title>
</head>
<body>
    <h1>Home Page: PW Skills</h1>
    <p>Link for Login Added</p>
</body>
</html>
```



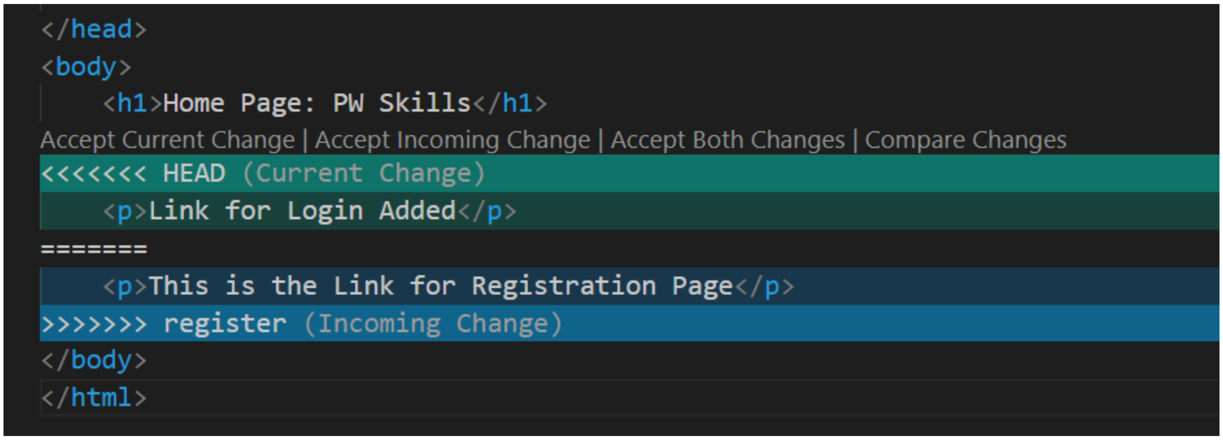
```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (register)
$ git checkout main
Switched to branch 'main'

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git add index.html

NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git commit -m "Login Link done"
[main bb60791] Login Link done
 1 file changed, 1 insertion(+)
```

Now Let's Merge.



```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git merge register
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
```

You can resolve this manually or use VS Code Editor to see the conflicts.

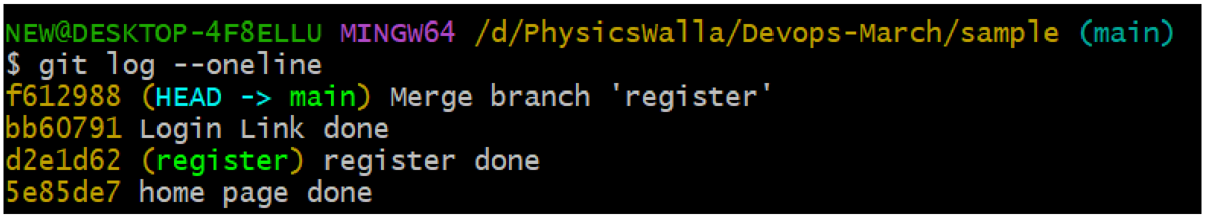


Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes

```
</head>
<body>
    <h1>Home Page: PW Skills</h1>
<<<<<<< HEAD (Current Change)
    <p>Link for Login Added</p>
=======
    <p>This is the Link for Registration Page</p>
>>>>>>> register (Incoming Change)
</body>
</html>
```

Which Change you want to take accept it. Like I am accepting both changes.

you can open Source Control panel and stage changes and commit.



```
NEW@DESKTOP-4F8ELLU MINGW64 /d/PhysicsWalla/Devops-March/sample (main)
$ git log --oneline
f612988 (HEAD -> main) Merge branch 'register'
bb60791 Login Link done
d2e1d62 (register) register done
5e85de7 home page done
```

As Discussed in live session please try to create repo and send the collaborator invitation and try to work on same repo

with different branched and practice with pull request and merge request.