

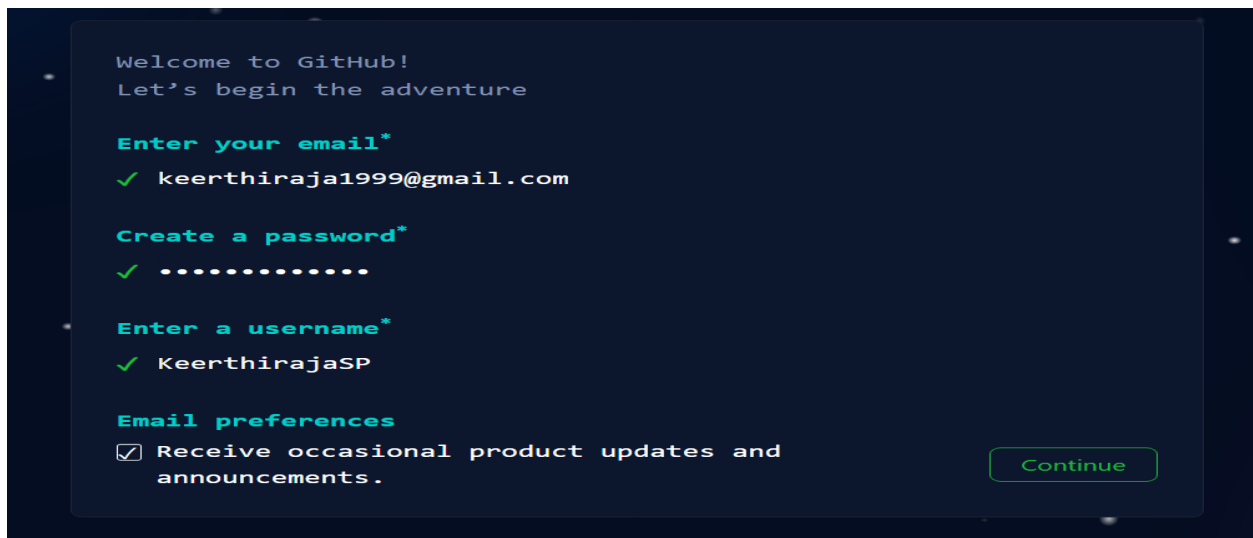
Hands-On 1: Creating Git and GitHub account:

It is a continuous integration and configuration management tool.

- 1) GIT → local repository (configuration management tool)
- 2) GITHUB → remote repository
- 3) Continuous integration

Git and GitHub:

Step 1: <https://github.com/> → click **Sign Up** → enter **mail id** → click **continue** → enter **new password** → Click **continue** → Enter **username** → click **continue** → click **email performance** → click **continue**

A screenshot of the GitHub sign-up form. The form is dark-themed with light blue text. It includes fields for email, password, and username, each with a green checkmark indicating successful input. There is a checkbox for email preferences and a 'Continue' button at the bottom right.

Welcome to GitHub!
Let's begin the adventure

Enter your email*

✓ keerthiraja1999@gmail.com

Create a password*

✓

Enter a username*

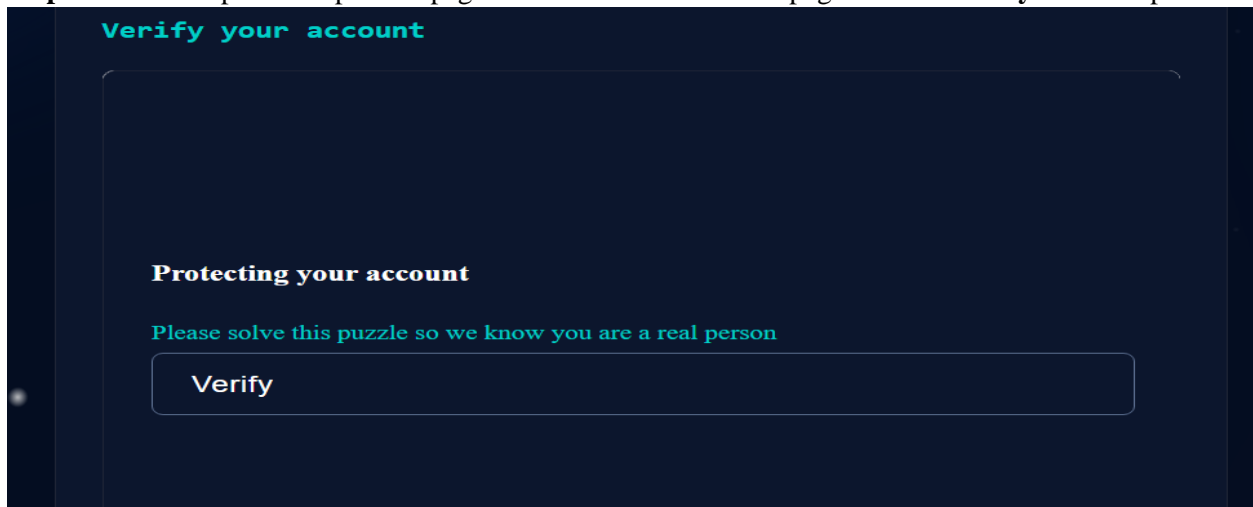
✓ KeerthirajaSP

Email preferences

☒ Receive occasional product updates and announcements.

Continue

Step 2: After complete this process page is redirect to verification page → Click **verify** → Solve puzzle

A screenshot of the GitHub account verification page. The page is dark-themed with light blue text. It features a 'Verify your account' heading, a 'Protecting your account' sub-heading, and a message asking the user to solve a puzzle. A 'Verify' button is located at the bottom.

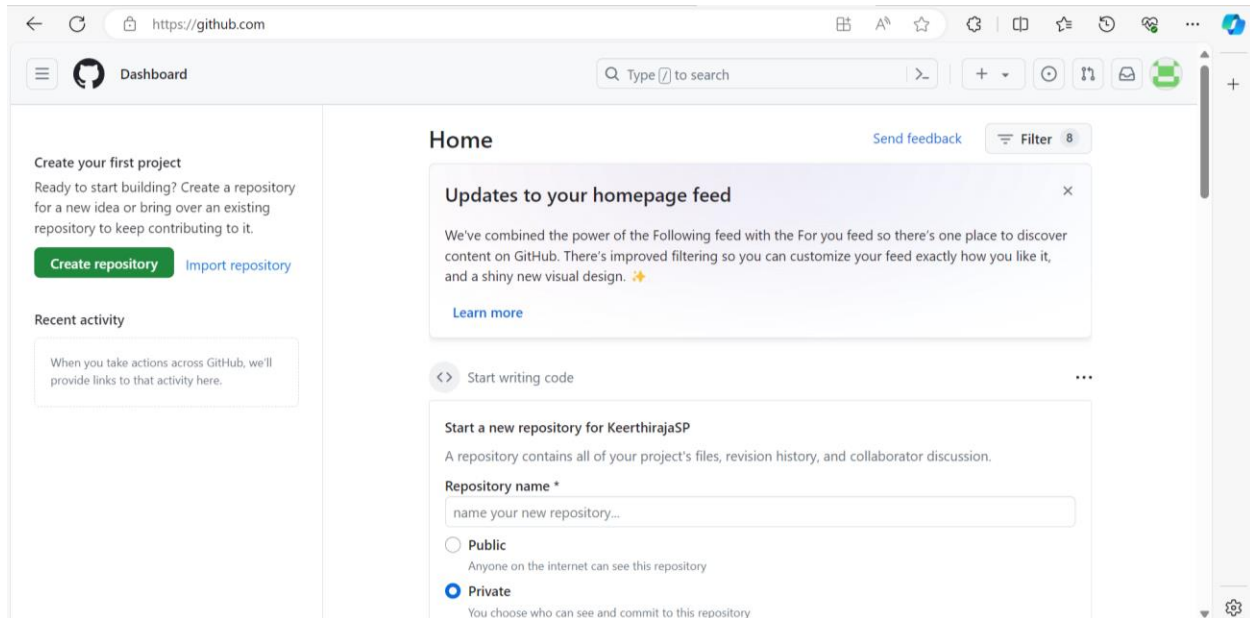
Verify your account

Protecting your account

Please solve this puzzle so we know you are a real person

Verify

Step 3: then enter verification code received to given mail id → then conform your **personal profession** → select **what specific features are you interested in using** → click **continue** → then select **Free trial or enterprise version** → click **continue**. Then you will have redirected to **Dashboard**.



Creating Repository in GitHub:

Step 1: Create new repository in GitHub → click create repository → give Repository name → select public or private → click create repository.

<> Start writing code ...

Start a new repository for KeerthirajaSP

A repository contains all of your project's files, revision history, and collaborator discussion.

Repository name *

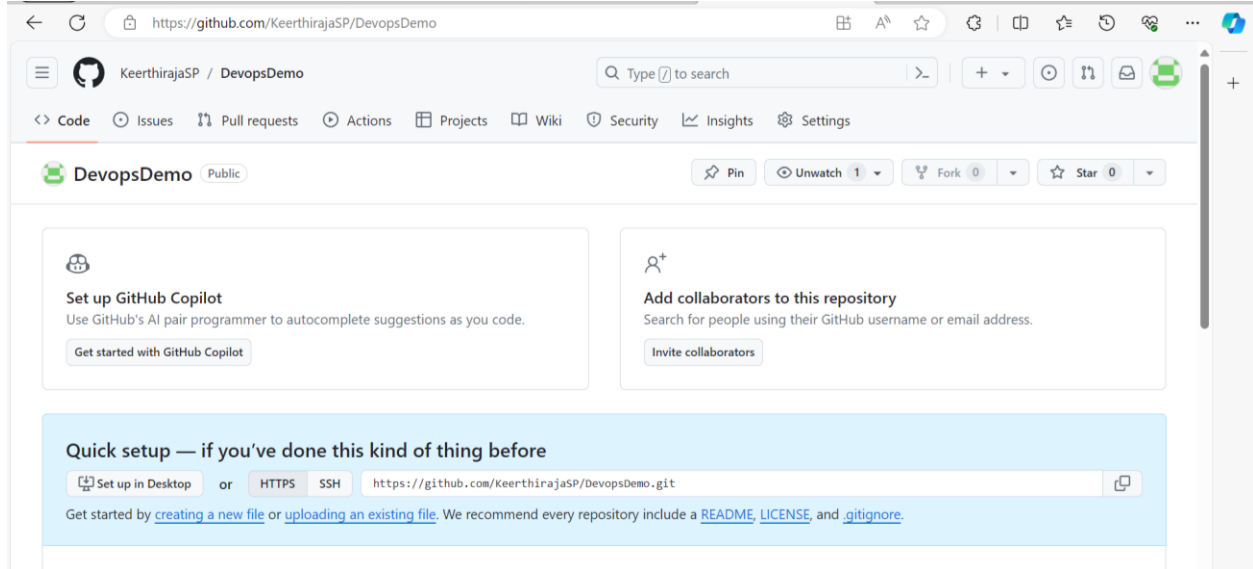
✓ DevopsDemo is available.

☒ Public
Anyone on the internet can see this repository

☐ Private
You choose who can see and commit to this repository

Create a new repository

Step 2: After clicking the create repository it will generate repository URL. (copy the url after creating the repository for future purpose)



Git commands for Local Repository:

- **git init** - go to gitbash and enter the command '**git init**' it will create a repository in local repository

```
MINGW64:/c:/Users/keerthiraja.sp
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~
$ git init
Initialized empty Git repository in C:/Users/keerthiraja.sp/.git/
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~ (master)
$
```

- **git clone** - you can create the copy of original repo which is in GitHub (go to GitHub click on code in that click on HTTPS then you will see a url copy that and paste in the command) by giving the command '**git clone "url of the repo"**

```
MINGW64:/c:/Users/keerthiraja.sp
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~ (master)
$ git clone "https://github.com/KeerthirajaSP/DevopsDemo.git"
Cloning into 'DevopsDemo'...
warning: You appear to have cloned an empty repository.
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~ (master)
$
```

- **ls** – it will display the list of files and directories

```
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~ (master)
$ ls
bash: '$'\302\223ls': command not found

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~ (master)
$ |
```

- **cd (change directory)** – to change into a specified directory enter '**cd filename**' it will take you to the filename you entered

```
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~ (master)
$ cd DevopsDemo

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$
```

- you can check what are all the files are there in the repo Devops7thMarch24 by giving '**ls**' command

to create file in local work station following 3 commands are used:-

- **cat**: you can create a file in your local work station using the command '**cat > filename**'
- **vi**: you can create a new file using the command '**vi filename**' it will show you an editor you can update, delete and add data in between
- **touch**: you can create an empty file using the command '**touch filename**'
- create a file named File1 using command '**cat > File1**' and enter the data in it after data entry is complete press **ctrl+D** to save and exit from the file.

```
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ ls

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ cat > File1
This is line 1
This is line 2
This is line 3

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$
```

- create a file you can check whether the file created or not by giving the command '**cat filename**'(cat File1)

```
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ cat File1
ThiThis is line 2
This is line 3

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ |
```

- **git add**: We created the file in local work station now we need to push it into local repository by giving the command '**git add .**'

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git add .
warning: in the working copy of 'File1', LF will be replaced by CRLF the next time Git touches it

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ |

```

- **git status:** we can see the information of directory by giving the command ‘git status’

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git status
On branch main
Your branch is based on 'origin/main', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

nothing to commit, working tree clean

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ |

```

- **git commit:** it will save/record the changes into local repository, the commit is done by using the command ‘git commit –m “provide the info of the commit”’

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git commit -m "File1 is commit and pushed to main in central repository by keerthiraja"
[main (root-commit) 6aee120] File1 is commit and pushed to main in central repository by keerthiraja
Committer: Keerthiraja S P <keerthiraja.sp@hcl.com>
Your name and email address were configured automatically based on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the following command and follow the instructions in your editor to edit your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 6 insertions(+)
create mode 100644 File1

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$

```

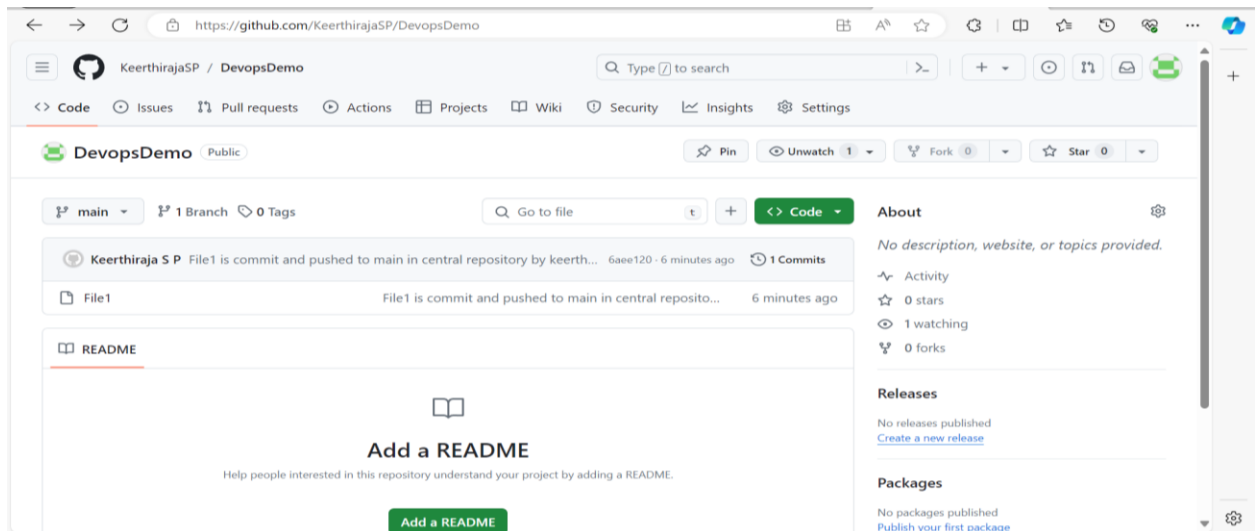
- **git push:** it will push the data from local repo to centralized repo, the push is done by the command ‘git push’

```

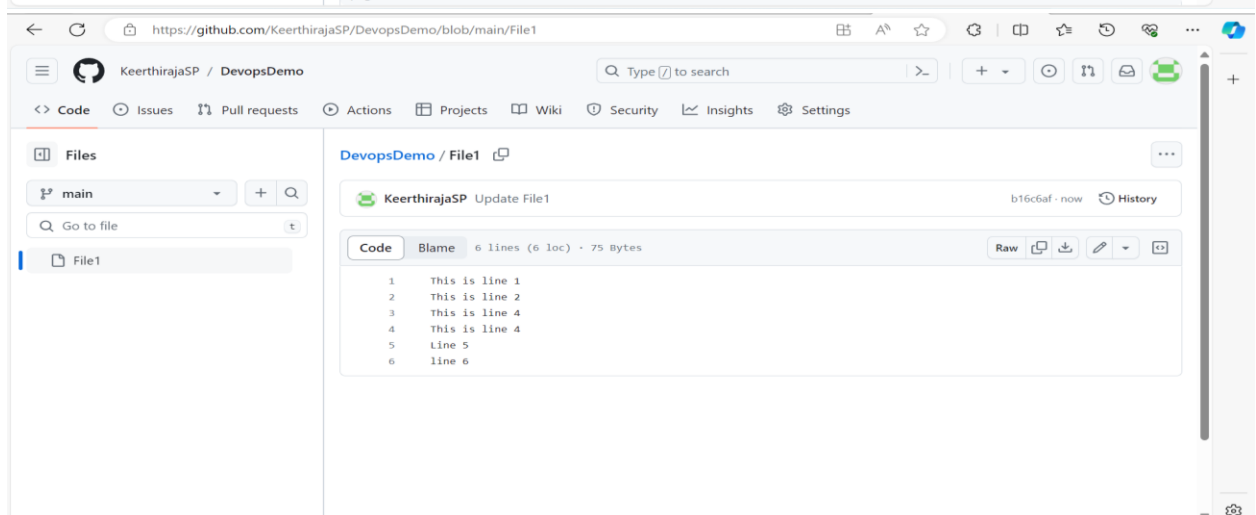
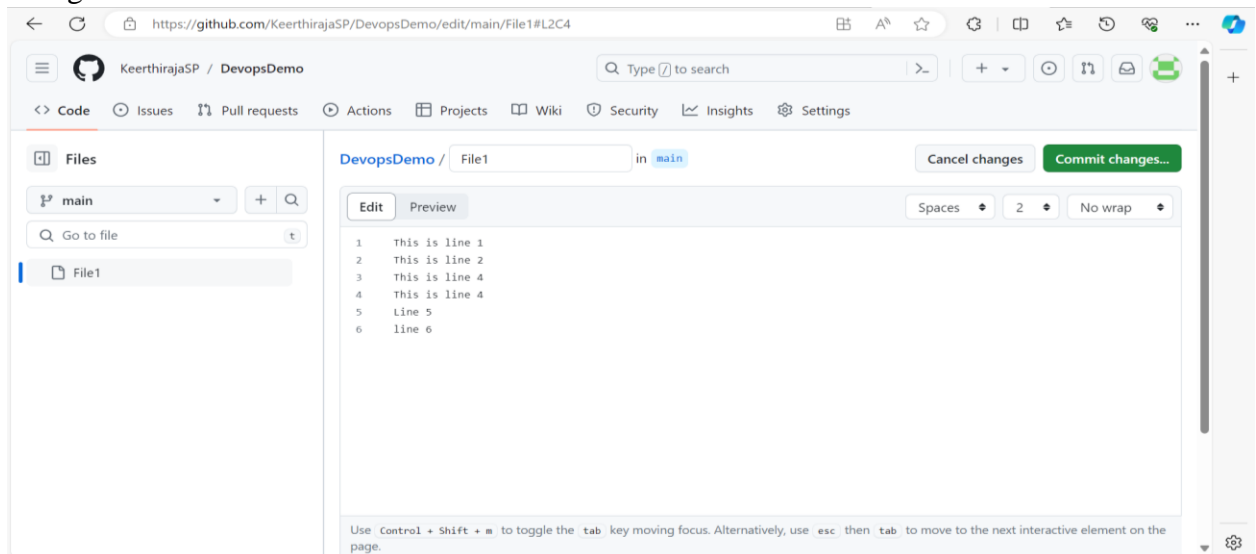
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 283 bytes | 23.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/KeerthirajaSP/DevopsDemo.git
 * [new branch]      main -> main

```

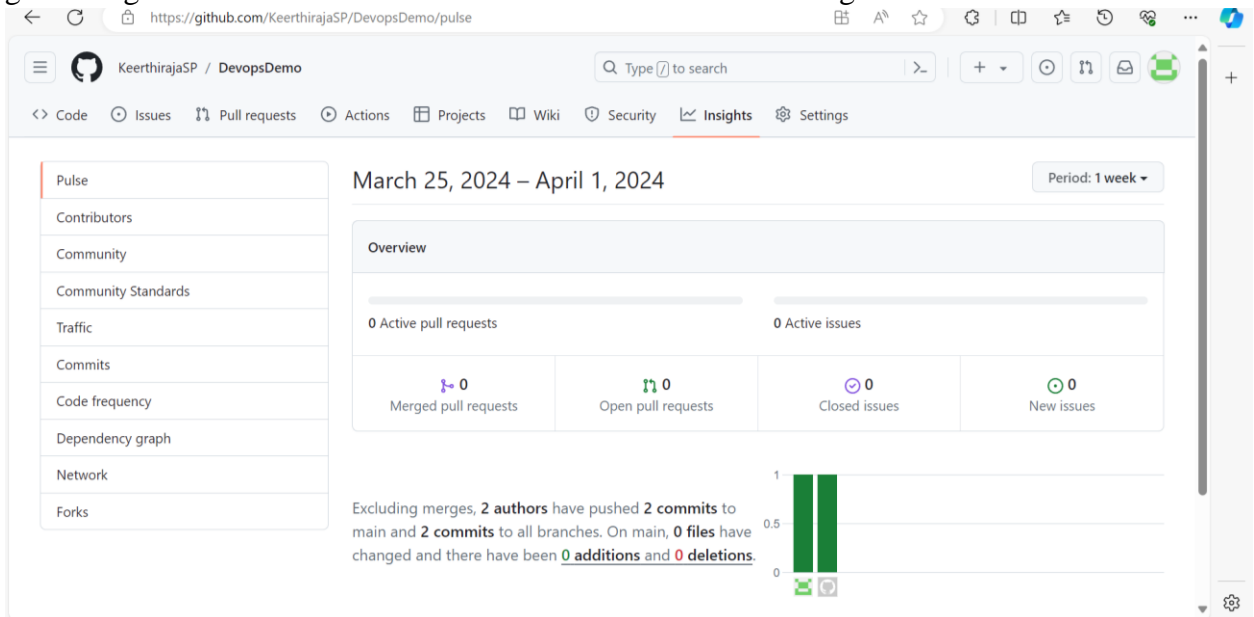
- now go to GitHub and check whether the file is pushed or not



- now if any person from the stake holders changes the data directly form GitHub or from their local workstation it will be auto merged with the file we created and display the changes in the GitHub



- go to insights and click on network to check how the data is changed



- git pull:** now if we want to make changes to the code we have to pull it from central repo to local repo, the pulling of data from central repo to local repo is done by giving the command **'git pull'**

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 917 bytes | 11.00 KiB/s, done.
From https://github.com/KeerthirajaSP/DevopsDemo
 6aee120..b16c6af main      -> origin/main
Updating 6aee120..b16c6af
Fast-forward
 File1 | 10 ++++++-----
 1 file changed, 5 insertions(+), 5 deletions(-)

```

- we should not always push the code/data from local repo to central repo (main branch) because everybody will think that it is the final one so do your job in a separate branch after finalizing only push it to the main branch
 - we can create another branch in git and GitHub
- creating another branch in git: use the command **'git branch branchname'** (git branch feature_branch)

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git branch future_branch

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$

```

- git checkout:** it will switch the branch from main branch to feature branch by giving the command **'git checkout branchname'** (git checkout feature_branch)

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git checkout future_branch
Switched to branch 'future_branch'

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (future_branch)
$

```

- let's add some data to our file1 for this use vi command '**vi File1**' and click enter. you will see an editor format like shown below

```
This is line 1  
This is line 2  
This is line 4  
This is line 4  
Line 5  
line 6
```

File1 [dos] (15:51 01/04/2024) 1,1 All

- now press 'I' on the keyboard to go into insert mode where we can do changes, you will see insert in the bottom of the screen

```
MINGW64:/c/Users/keerthiraja.sp/DevopsDemo  
This is line 1  
This is line 2  
This is line 4  
This is line 4  
Line 5  
line 6  
~  
~  
~  
~  
~  
~  
~  
~  
~  
~  
File1 [dos] (15:51 01/04/2024) 1,1 All  
-- INSERT --
```

- do the changes as per your requirement.
- after the completion of changes press Esc button ‘:wq!’ on the keyboard to save and exit from the vi editor

```
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (future_branch)
$ cat File1
This is line 1
This is line 2
This is line 4
This is line 4
Line 5
line 6
line 7
```


- now push the file to local repository by giving the command '**git add .**' and then commit the changes using the command '**git commit -m "info about commit"**'

```

MINGW64:/c/Users/keerthiraja.sp/DevopsDemo
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (future_branch)
$ git add .

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (future_branch)
$ git commit -m "future branch is created for working code"
[future_branch 5ea610c] future branch is created for working code
Committer: Keerthiraja S P <keerthiraja.sp@hcl.com>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

1 file changed, 1 insertion(+)
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (future_branch)
$

```

- here the feature branch is not available in the centralized repo as it is created in local repo so to push along with the branch '**git push --set-upstream origin branchname**' command is used

```

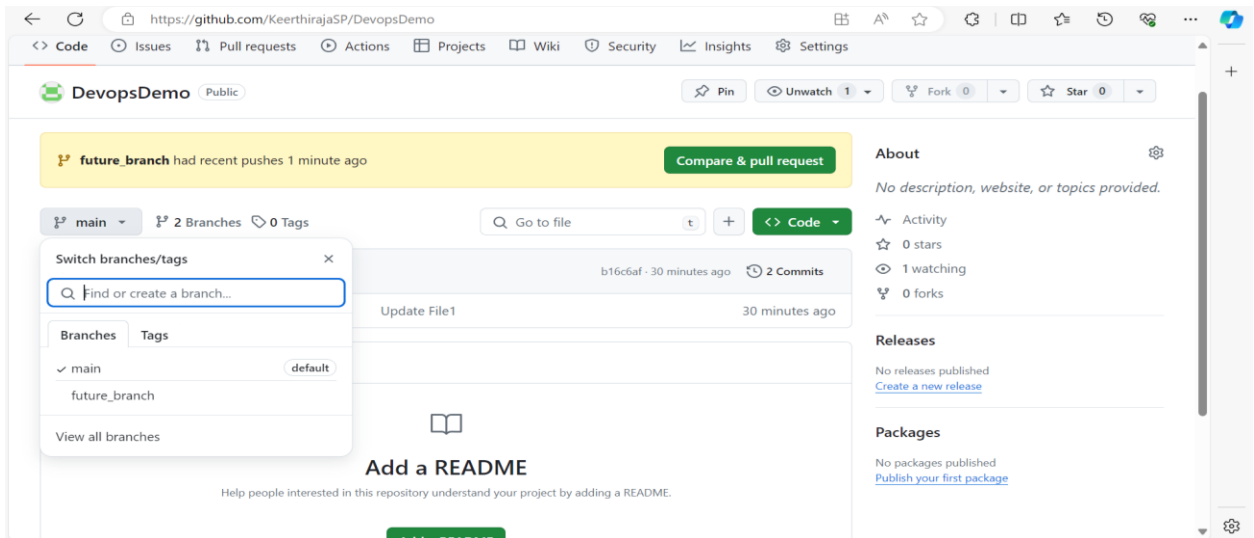
MINGW64:/c/Users/keerthiraja.sp/DevopsDemo
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (future_branch)
$ git commit -m "future branch is created for working code"
On branch future_branch
nothing to commit, working tree clean

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (future_branch)
$ git push --set-upstream origin future_branch
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 316 bytes | 31.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'future_branch' on GitHub by visiting:
remote:   https://github.com/KeerthirajaSP/DevopsDemo/pull/new/future_branch
remote:
To https://github.com/KeerthirajaSP/DevopsDemo.git
 * [new branch]      future_branch -> future_branch
branch 'future_branch' set up to track 'origin/future_branch'.

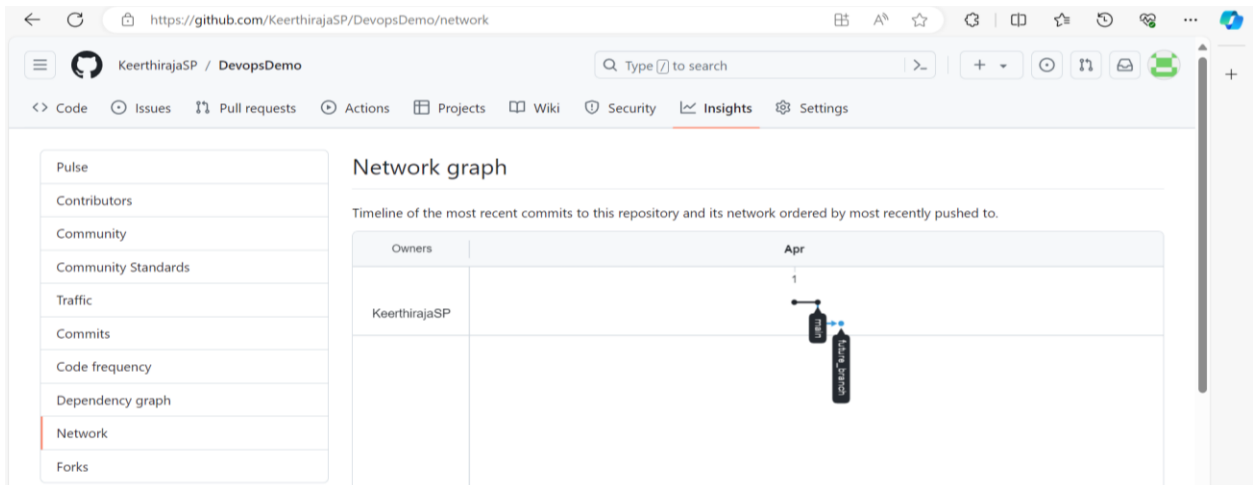
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (future_branch)
$ |

```

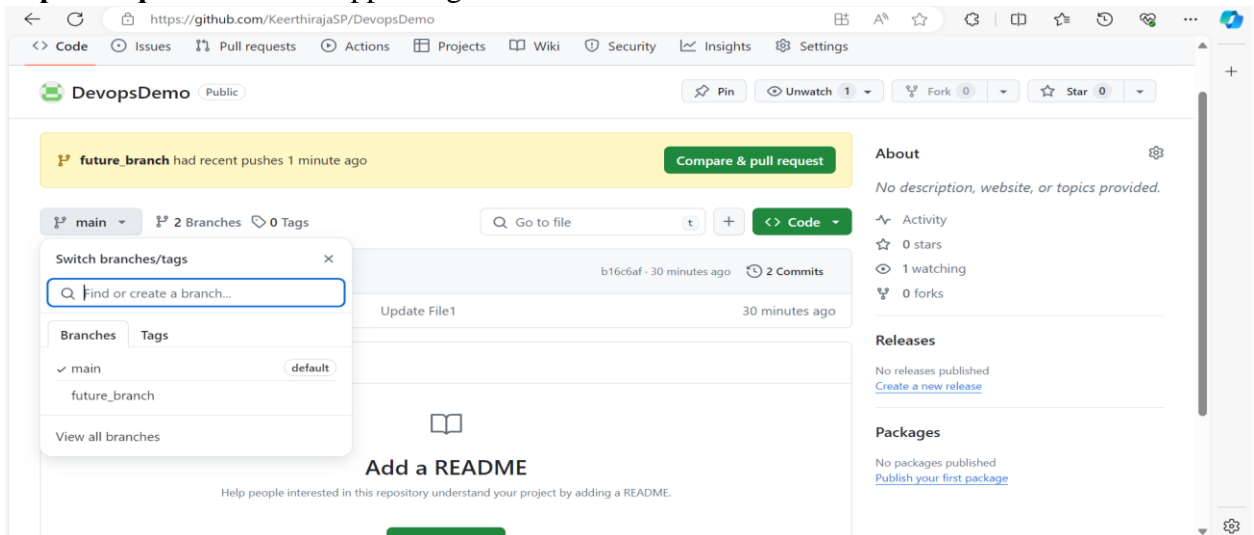
- now go to centralized repository (GitHub) you will see the feature branch is added over there



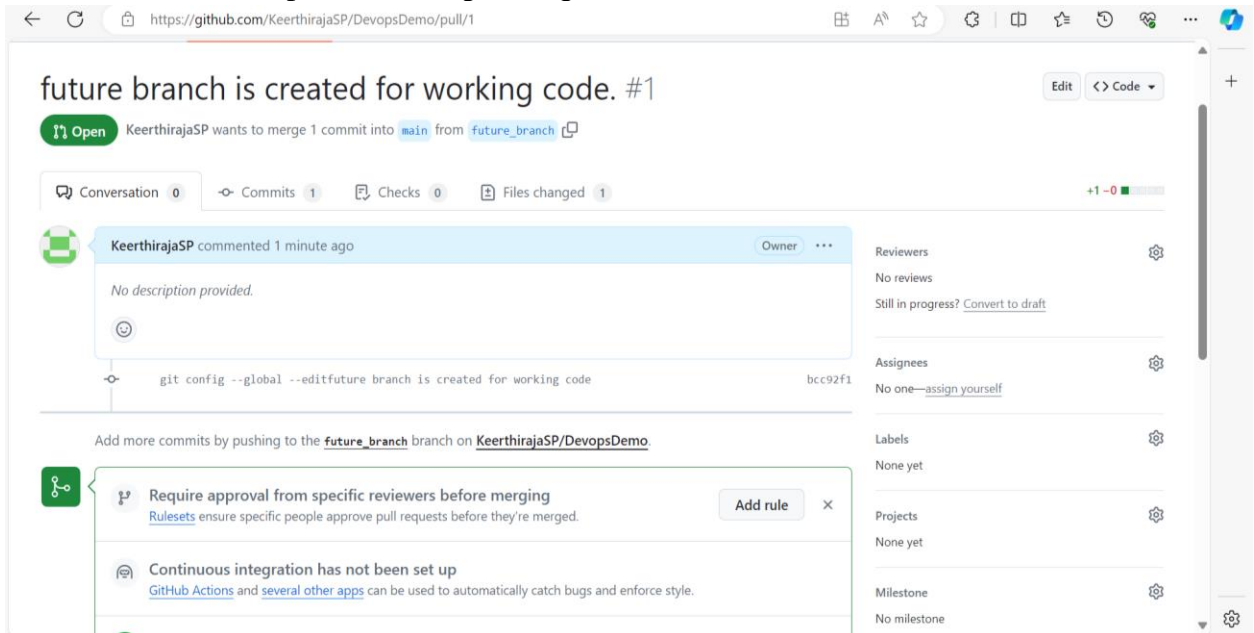
- now go to insights and click on network you will see the following changes that the copy of main branch is made into feature branch



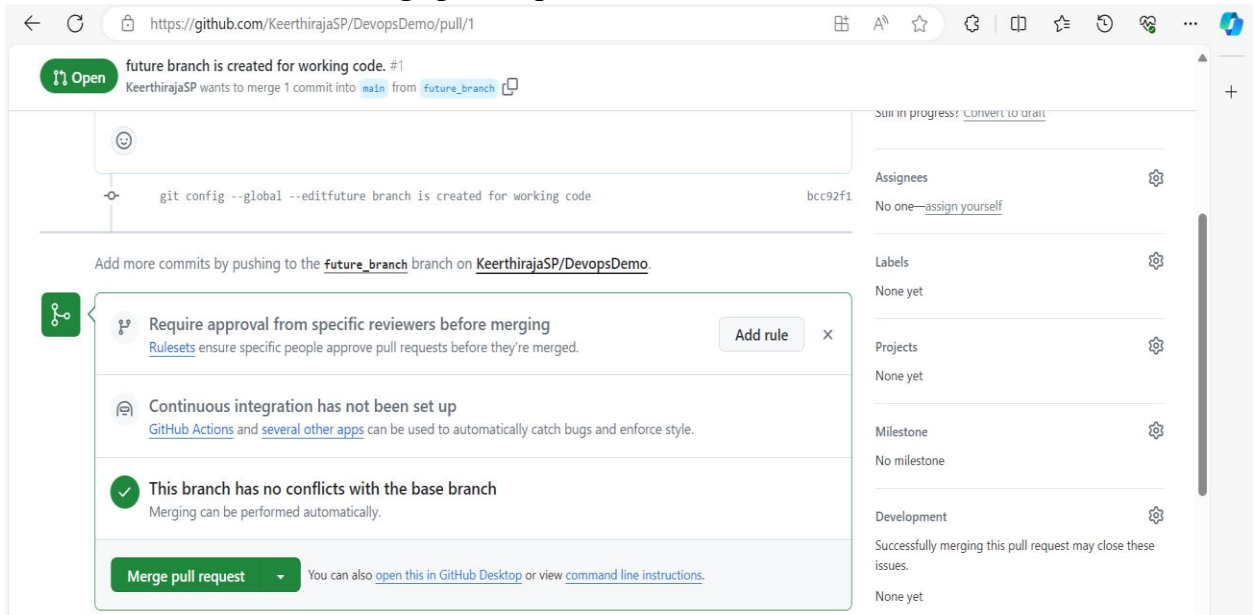
- Now the data is still in the feature branch to push it to the main branch click on '**compare & pull request**' which is appearing on the screen as shown below



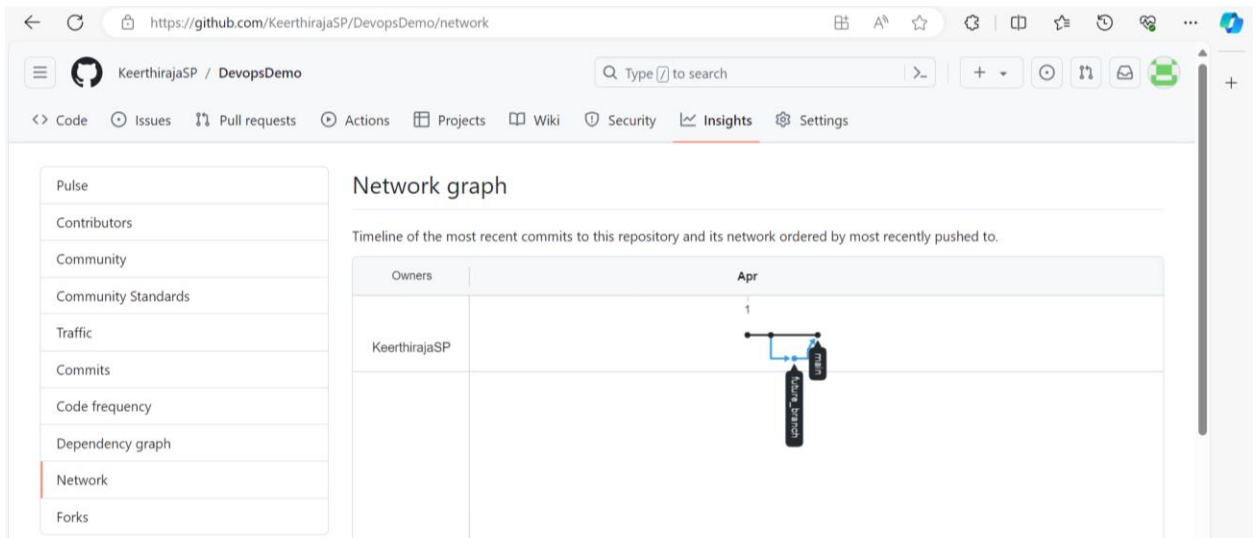
- Now you can see the following page, add a title in the title box
- Now scroll down and press **‘create pull request’**



- Scroll down and click on **‘merge pull request’**



- Click on **‘confirm merge’**
- Now go to insights and click on networks you can see that the data in the feature branch is moved to main branch



- Now give the command '**cat filename**' and see which data is there in the feature branch

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (future_branch)
$ cat File1
This is line 1
This is line 2
This is line 4
This is line 4
Line 5
line 6
line 7

```

- Switch the branch to main branch by giving command '**git checkout**' and check which data is there in it by giving the command '**cat filename**'

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (future_branch)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

```

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ ls
File1

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ cat File1
This is line 1
This is line 2
This is line 4
This is line 4
Line 5
line 6

```

- By observing the above image, we can say that, in local repository the merge of feature branch to main branch is not done but is has been merged in central repository so to get the updated code in local repository pull the code by giving the command '**git pull**'

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git pull
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), 920 bytes | 21.00 KiB/s, done.
From https://github.com/KeerthirajaSP/DevopsDemo
   b16c6af..ec3a969  main       -> origin/main
Updating b16c6af..ec3a969
Fast-forward
 File1 | 1 +
 1 file changed, 1 insertion(+)

```

- Now you can see the updated code as shown below

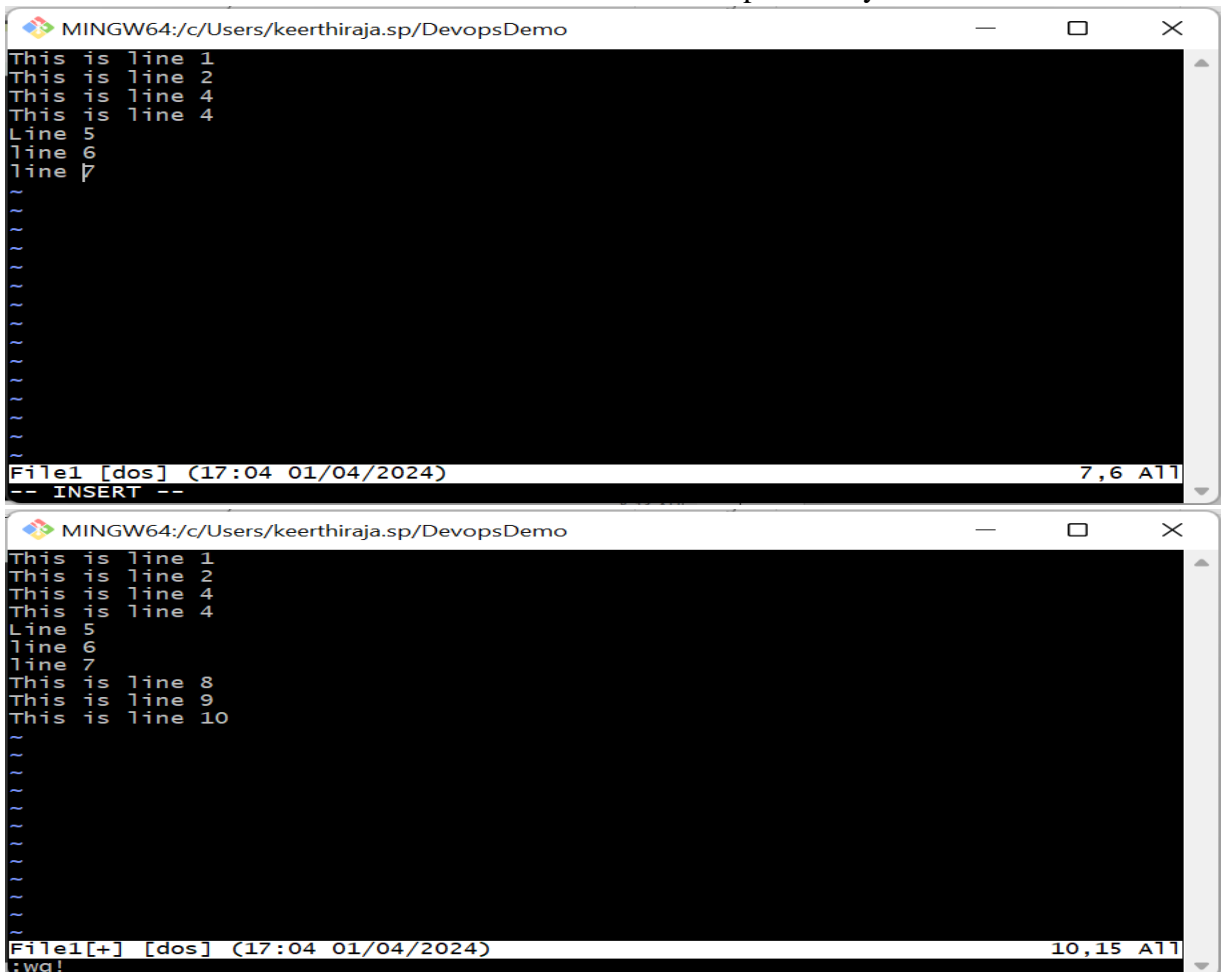
```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ cat File1
This is line 1
This is line 2
This is line 4
This is line 4
Line 5
line 6
line 7

```

MERGE CONFLICT

- Now here say we are modifying the file1 by entering some data for this use '**vi filename**' command and enter the data and come out of it as we did previously



The first screenshot shows a terminal window titled 'MINGW64/c/Users/keerthiraja.sp/DevopsDemo'. The content of 'File1' is displayed as follows:

```

This is line 1
This is line 2
This is line 4
This is line 4
Line 5
line 6
line 7

```

The status bar at the bottom indicates 'File1 [dos] (17:04 01/04/2024)' and '7,6 All'. Below the terminal content, the text '-- INSERT --' is visible.

The second screenshot shows the same terminal window after editing. The content of 'File1' is now:

```

This is line 1
This is line 2
This is line 4
This is line 4
Line 5
line 6
line 7
This is line 8
This is line 9
This is line 10

```

The status bar at the bottom indicates 'File1[+] [dos] (17:04 01/04/2024)' and '10,15 All'. Below the terminal content, the text ':wq!' is visible.

- Now push the code to the local repository by giving the command **'git add .'** and commit the change by giving the command **'git commit -m "info about commit"'**

```
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git add .

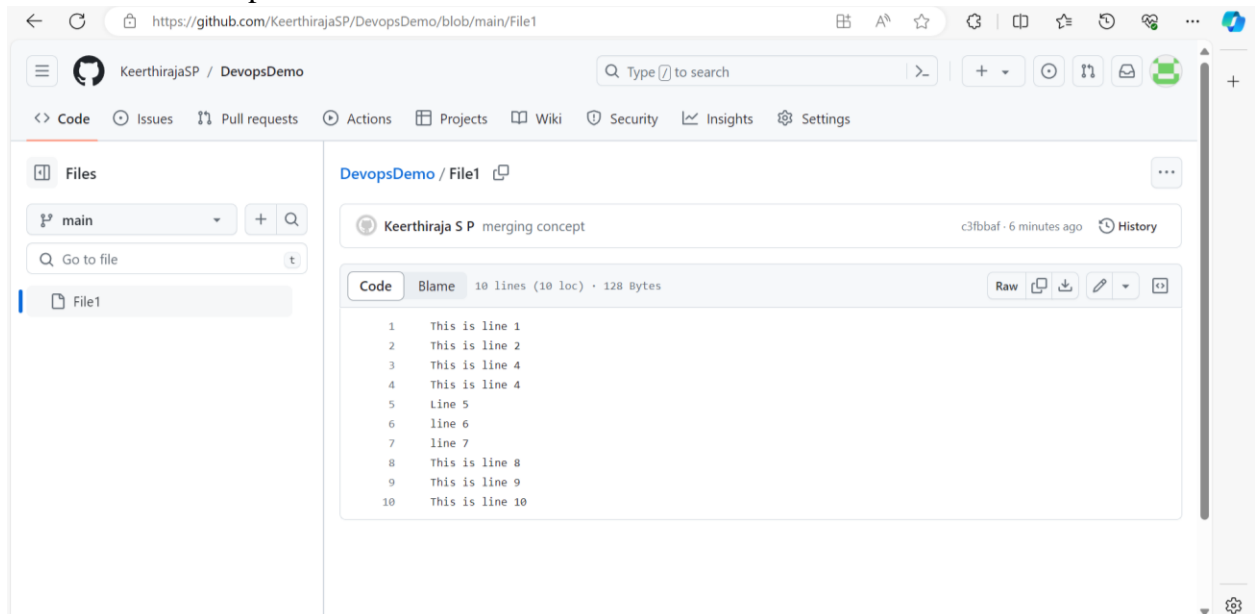
keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git push --set-upstream origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 288 bytes | 28.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/KeerthirajaSP/DevopsDemo.git
   ec3a969..c3fbbaf  main -> main
branch 'main' set up to track 'origin/main'.

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git commit -m "merging concept"
On branch main
Your branch is up to date with 'origin/main'.

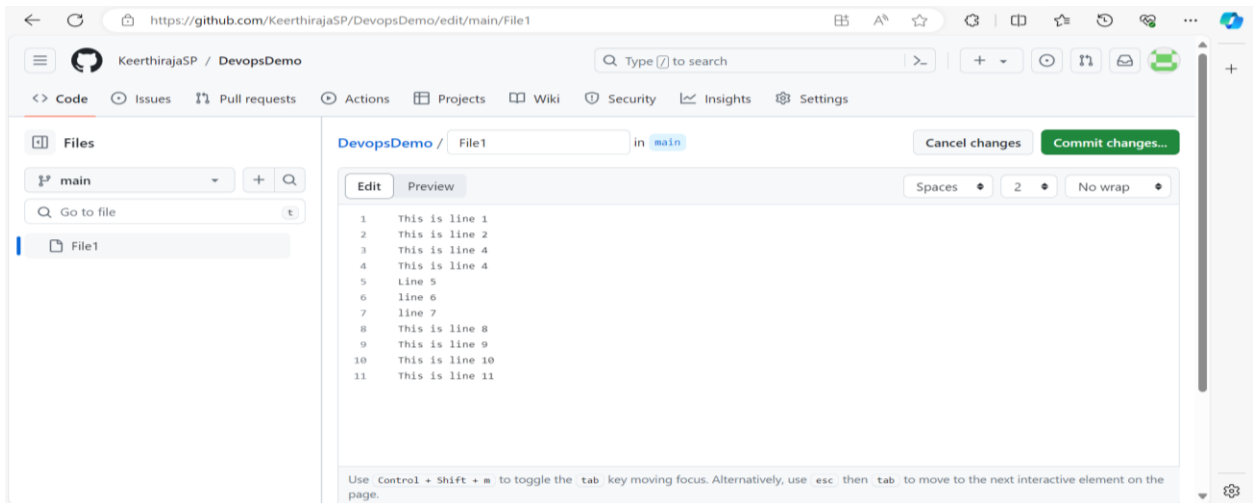
nothing to commit, working tree clean

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$
```

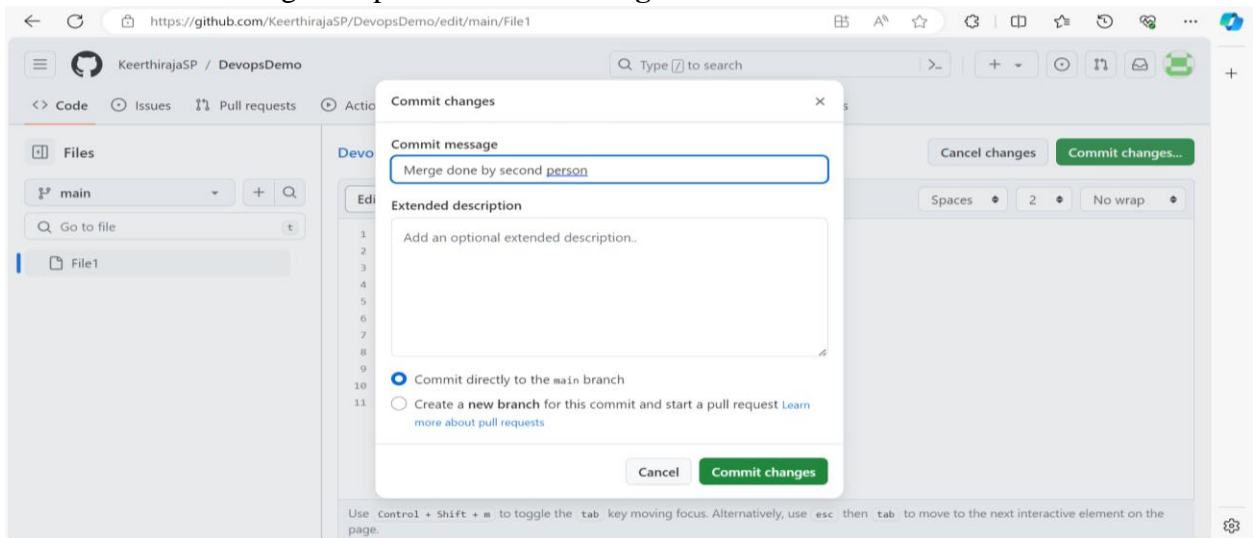
- Meanwhile some other person also modifying the main code from the central repository or from their local pc as shown below



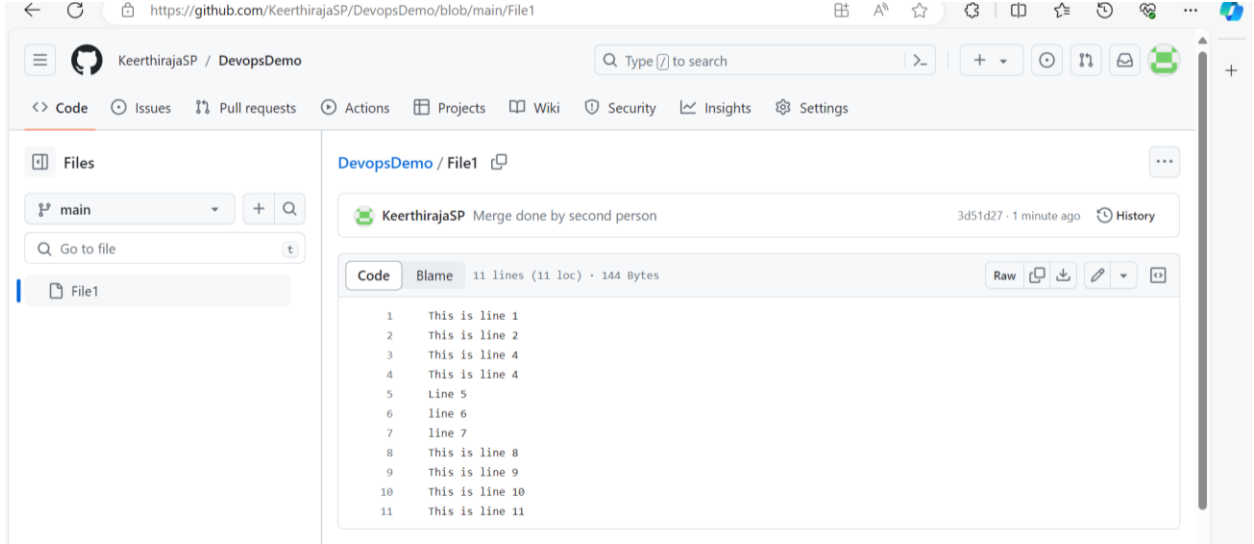
- Press **'commit changes'** as shown below



- Give a commit message and press ‘commit changes’



- Now the data is committed by some other person



- Now if you try to push your data/code into central repo you will get a message as rejected like shown below
 - This is happened because you are trying to update something but there is some other updating already available which is not your current code

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git push
To https://github.com/KeerthirajaSP/DevopsDemo.git
! [rejected]        main -> main (fetch first)
error: failed to push some refs to 'https://github.com/KeerthirajaSP/DevopsDemo.git'
hint: Updates were rejected because the remote contains work that you do not
hint: have locally. This is usually caused by another repository pushing to
hint: the same ref. If you want to integrate the remote changes, use
hint: 'git pull' before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$

```

- So now we have to pull the updated code and then push, but when we pull the code we can see merge conflict in its status telling that automatic merge failed fix the conflict and then commit the result

```

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 941 bytes | 10.00 KiB/s, done.
From https://github.com/KeerthirajaSP/DevopsDemo
 c3fbbaf..3d51d27  main       -> origin/main
Updating c3fbbaf..3d51d27
Fast-forward
 File1 | 1 +
 1 file changed, 1 insertion(+)

keerthiraja.sp@LP-5CG2261HYG MINGW64 ~/DevopsDemo (main)
$

```

- We can see in the image that there is (main|MERGING) showing that there is a conflict in order to fix the conflict we have to open the file by giving the command **'vi filename'**
- now we can see that we ahead of some code and behind of some code so we need to fix this



```

MINGW64 ~/Users/ambatu/Devops7thMarch24
this is line1
this is line2
this is line4
this is line5
this is line6
this is line7
<<<<<<< HEAD
this is line8
this is line9
=====
this is line10
this is line11
>>>>>>> a5f13fd00ded1298404a54b7e003aa60a89a7b6b

File1 [dos] (22:41 17/03/2024)
"File1" [dos] 13L, 225B

```


- now do the modifications as per your requirement, after fixing the conflict save and exit from the file
- Now push this data into local repository by giving the command '**git add .**' and commit the changes by using the command '**git commit -m "info about commit"**'
- Now push the code into central repo using the command '**git push**' we can see that the data get pushed without any conflict
- Now go to central repo here we can clearly see the updated code

