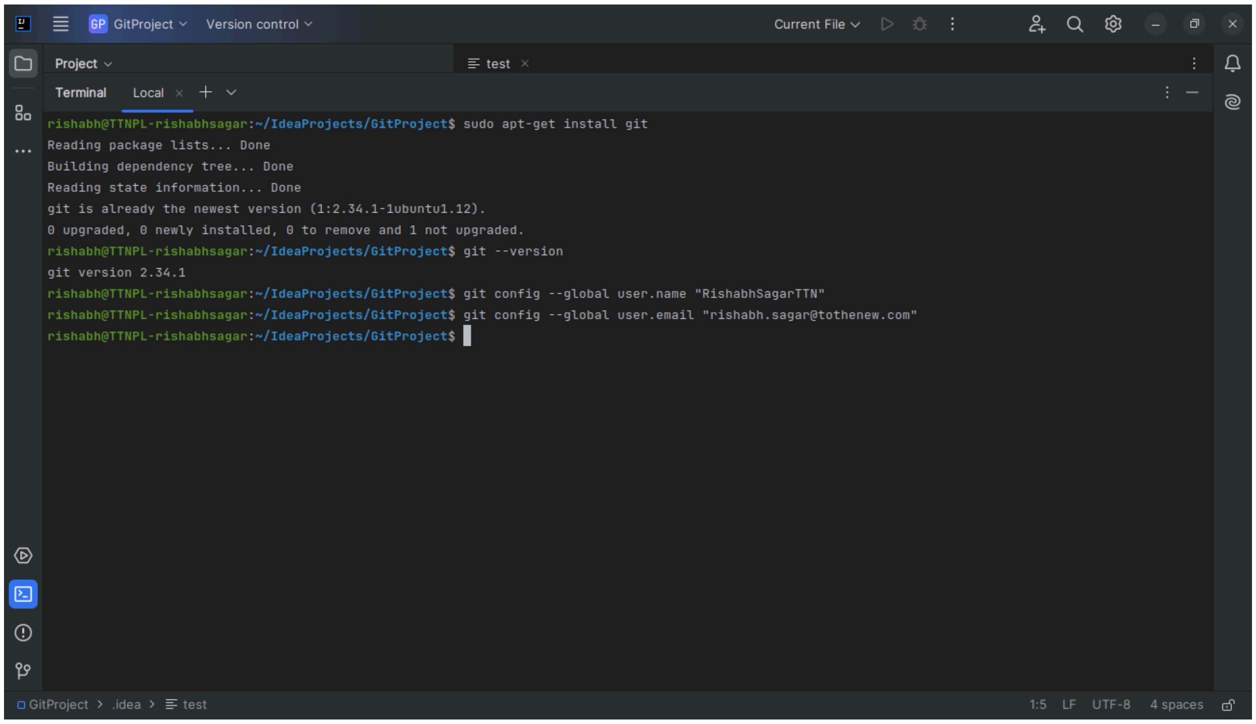
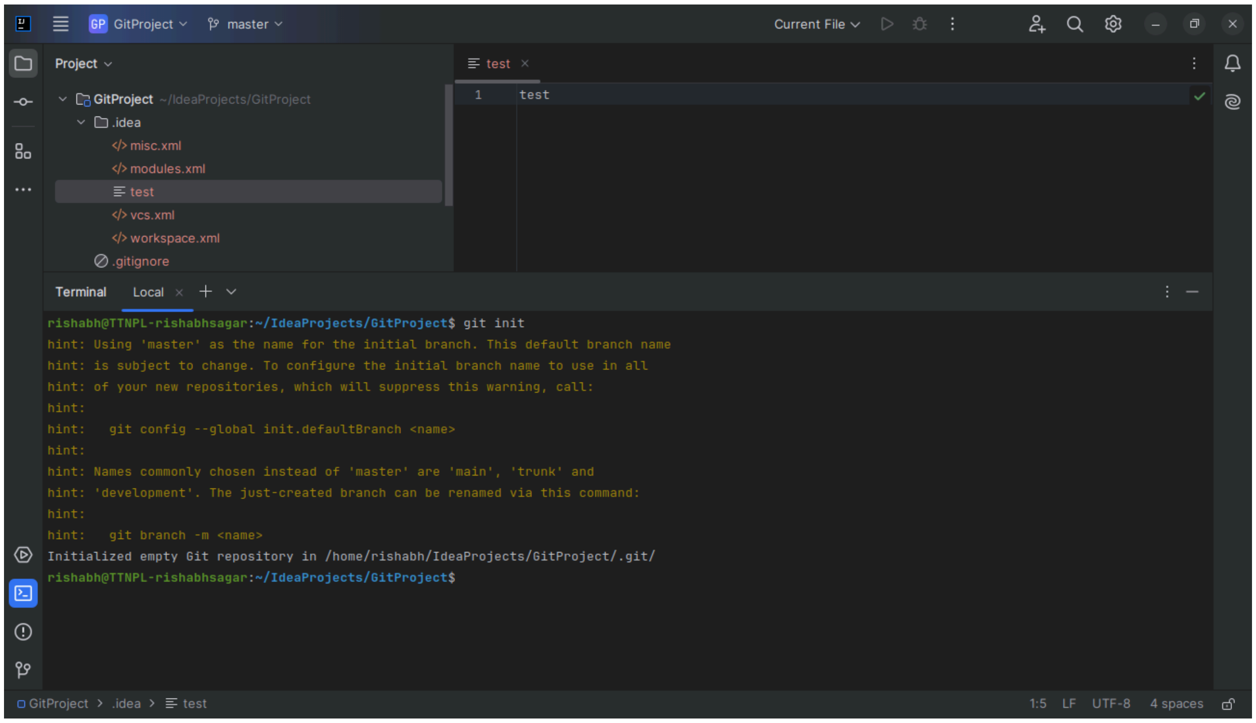
Introduction to Version Control

Q1)Git Setup https://confluence.atlassian.com/bitbucket/set-up-git-744723531.html
Ans. sudo apt-get install git is used to download and install the git in the linux to veriy weather the git is installed or not git - - version if installed in the system the git version is shown in the terminal. After git is installed in the system we config the git so that git know the user who is committing or adding the file in the git.



```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProject$ sudo apt-get install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.34.1-1ubuntu1.12).
0 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProject$ git --version
git version 2.34.1
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProject$ git config --global user.name "RishabhSagarTTN"
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProject$ git config --global user.email "rishabh.sagar@tothenew.com"
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProject$
```

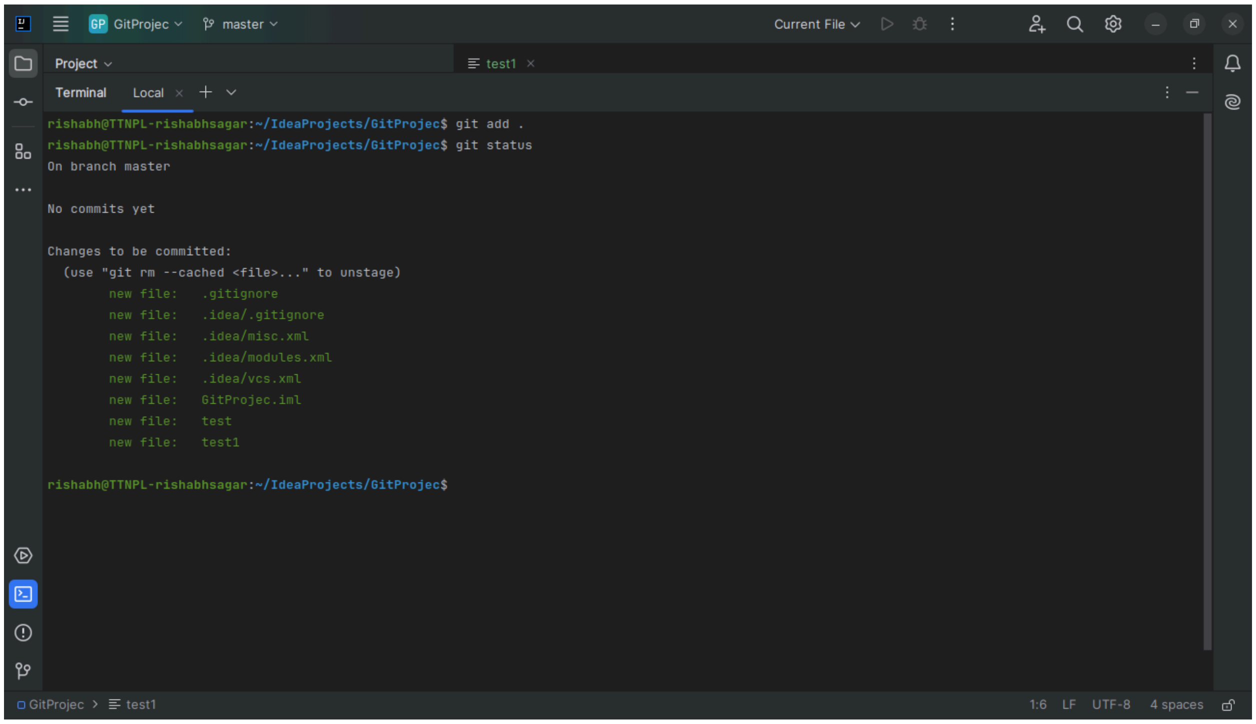
Q2)Initialize a Git Repository
Ans. To initialize the the git repository we first make the repository to whom we want to track, we then go to that repository and execute git init command in the terminal, this command will initialize the git in the repository and git will start tracking the repository.



```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProject$ git init
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/rishabh/IdeaProjects/GitProject/.git/
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProject$
```

Q3)Add files to the repository

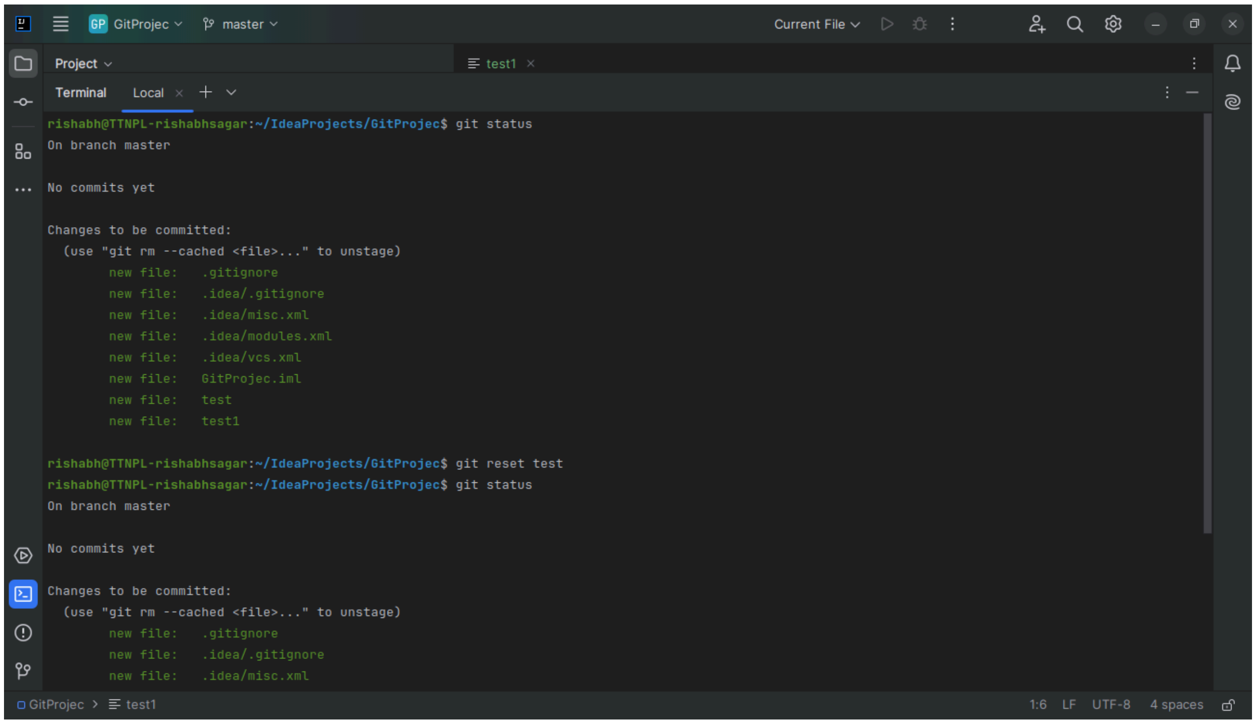
Ans. to add a file in the repository first create the file in the repository, then execute the git add . command in the terminal, this will add all the file in the repository to the git staged area where it is ready to commit.



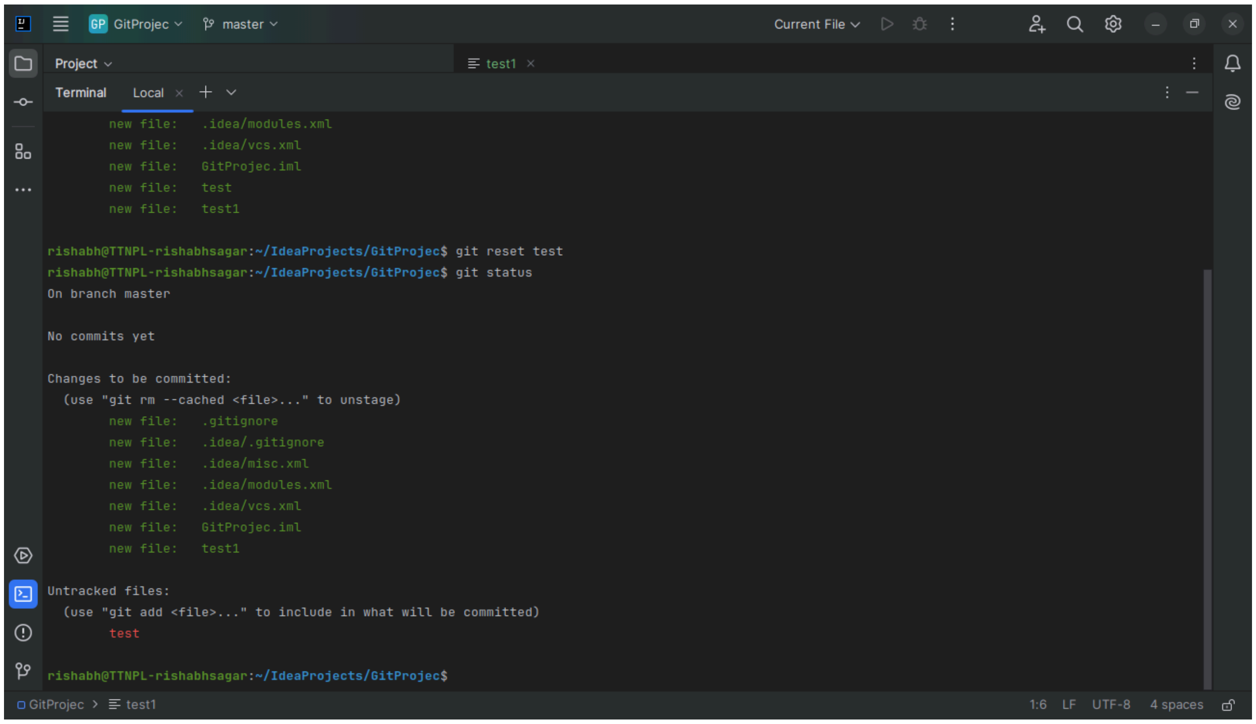
```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git add .
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .gitignore
        new file:   .idea/.gitignore
        new file:   .idea/misc.xml
        new file:   .idea/modules.xml
        new file:   .idea/vcs.xml
        new file:   GitProjec.iml
        new file:   test
        new file:   test1

rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

Q4)Unstage 1 file

Ans. git status – check the status of the file weather it is in staged, unstaged etc.
To bring back the file to unstaged area execute git reset <file name in staged area>,



```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .gitignore
        new file:   .idea/.gitignore
        new file:   .idea/misc.xml
        new file:   .idea/modules.xml
        new file:   .idea/vcs.xml
        new file:   GitProjec.iml
        new file:   test
        new file:   test1

rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git reset test
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .gitignore
        new file:   .idea/.gitignore
        new file:   .idea/misc.xml
```

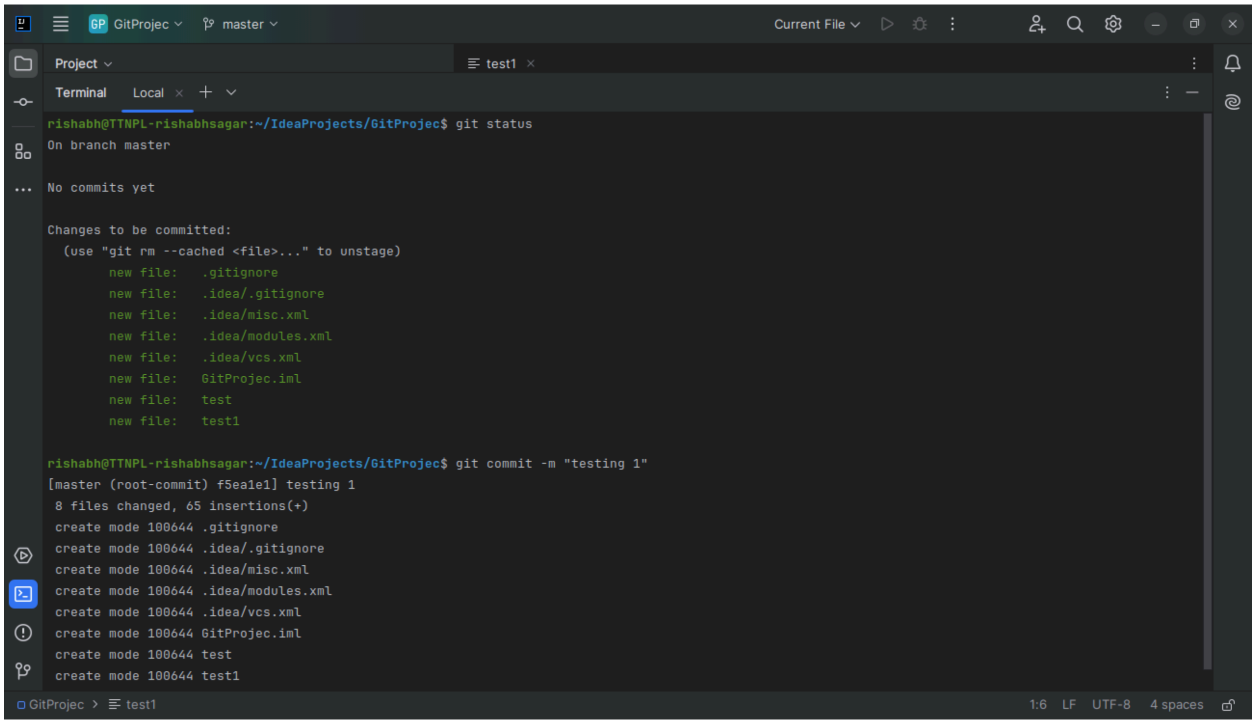


```
        new file:   .idea/modules.xml
        new file:   .idea/vcs.xml
        new file:   GitProjec.iml
        new file:   test
        new file:   test1

rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git reset test
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .gitignore
        new file:   .idea/.gitignore
        new file:   .idea/misc.xml
        new file:   .idea/modules.xml
        new file:   .idea/vcs.xml
        new file:   GitProjec.iml
        new file:   test1

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

rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

Q5)Commit the file

Ans.Git commit is used to save changes in staging area to the local git repository so that we can rollback to any time of the fike if something went wrong.

Command-git commit -m(message or comment) " xyz "



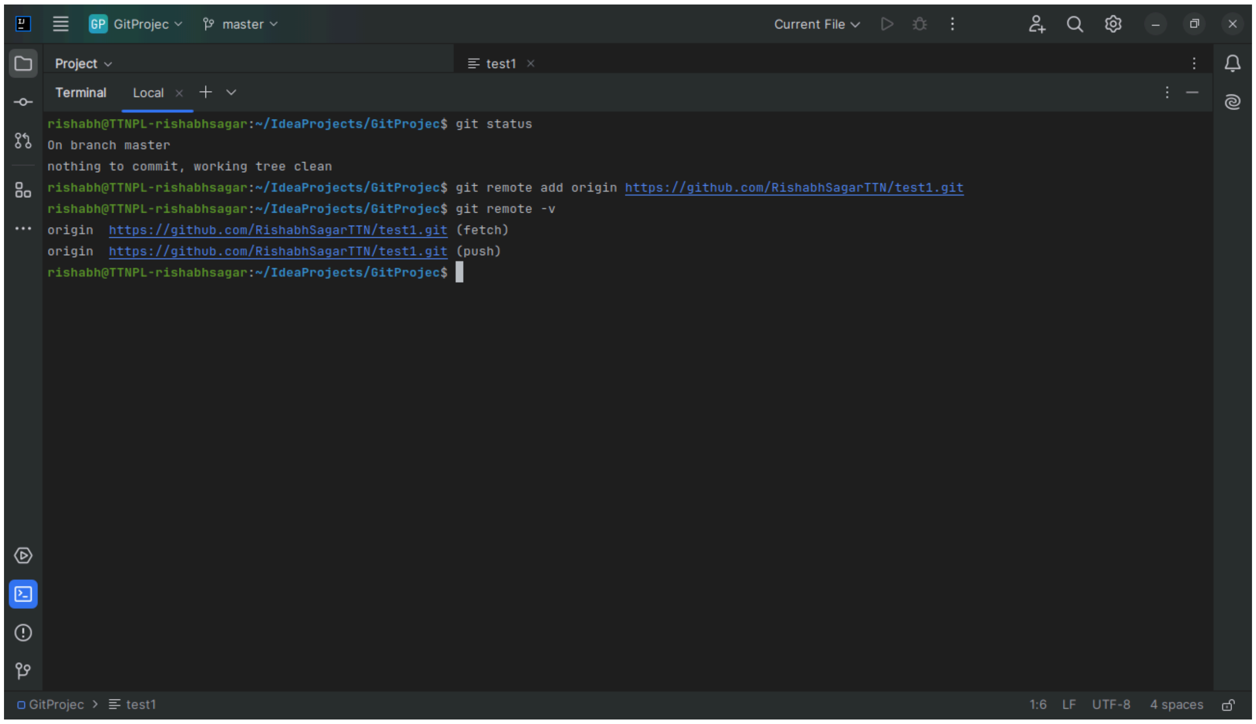
```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .gitignore
        new file:   .idea/.gitignore
        new file:   .idea/misc.xml
        new file:   .idea/modules.xml
        new file:   .idea/vcs.xml
        new file:   GitProjec.iml
        new file:   test
        new file:   test1

rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git commit -m "testing 1"
[master (root-commit) f5ea1e1] testing 1
 8 files changed, 65 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 .idea/.gitignore
 create mode 100644 .idea/misc.xml
 create mode 100644 .idea/modules.xml
 create mode 100644 .idea/vcs.xml
 create mode 100644 GitProjec.iml
 create mode 100644 test
 create mode 100644 test1
```

Q6)Add a remote

Ans.git remote(remote repository) add(to add the repository) origin(default name of the remote repository) <link of the repository>

command – git remote add origin <link>

To see all the added remote repository use git remote -v

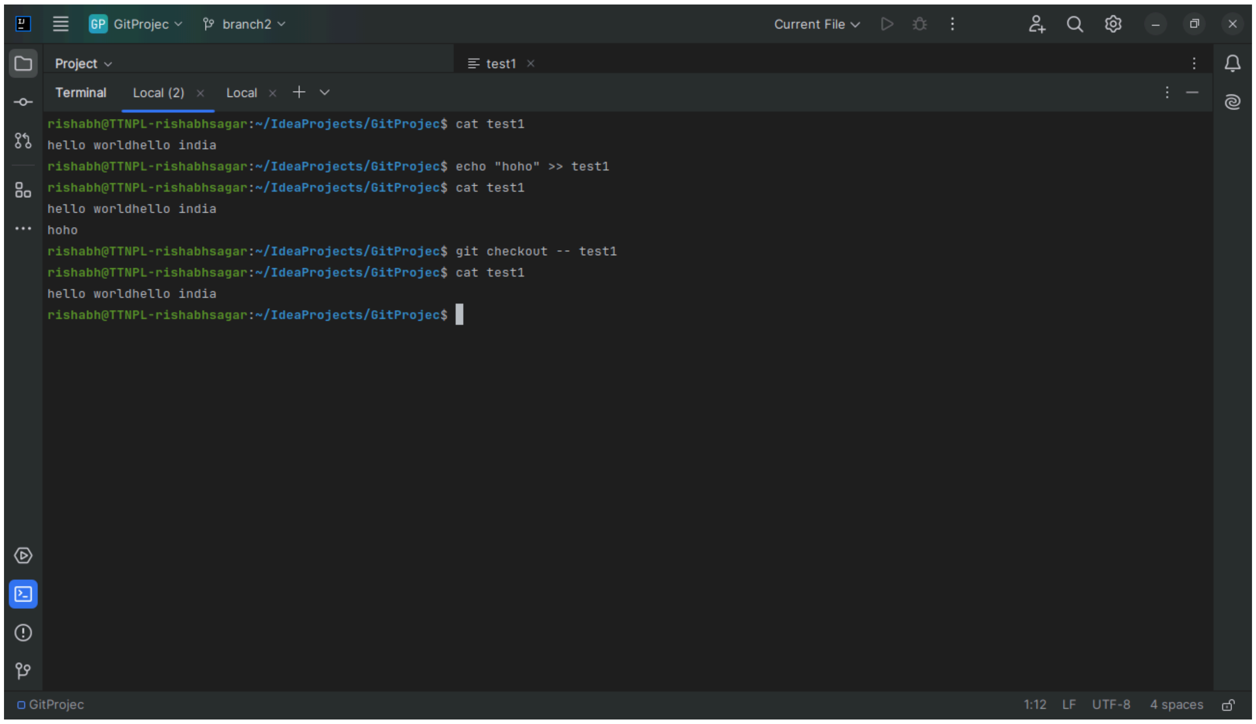


```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git status
On branch master
nothing to commit, working tree clean
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git remote add origin https://github.com/RishabhSagarTTN/test1.git
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git remote -v
origin  https://github.com/RishabhSagarTTN/test1.git (fetch)
origin  https://github.com/RishabhSagarTTN/test1.git (push)
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

Q7)Undo changes to a particular file

Ans.Command - git checkout – <file name>

This will undo the unstaged file

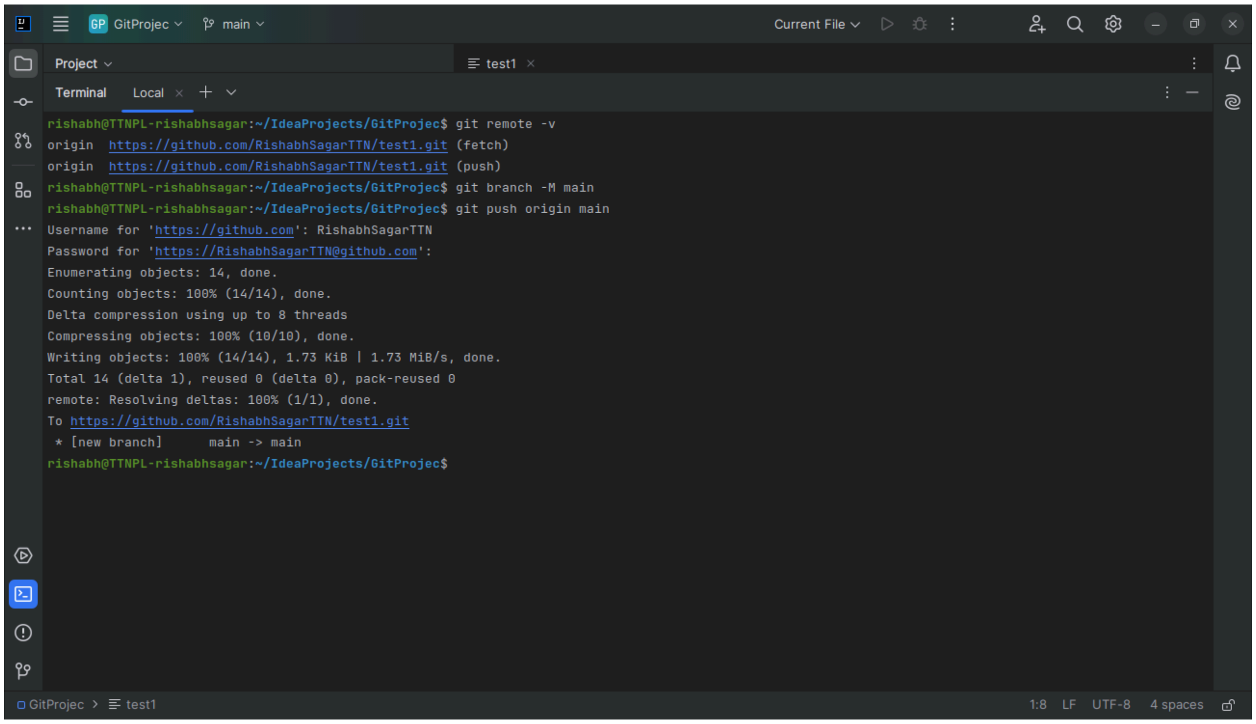


```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ cat test1
hello worldhello india
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ echo "hoho" >> test1
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ cat test1
hello worldhello india
hoho
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout -- test1
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ cat test1
hello worldhello india
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

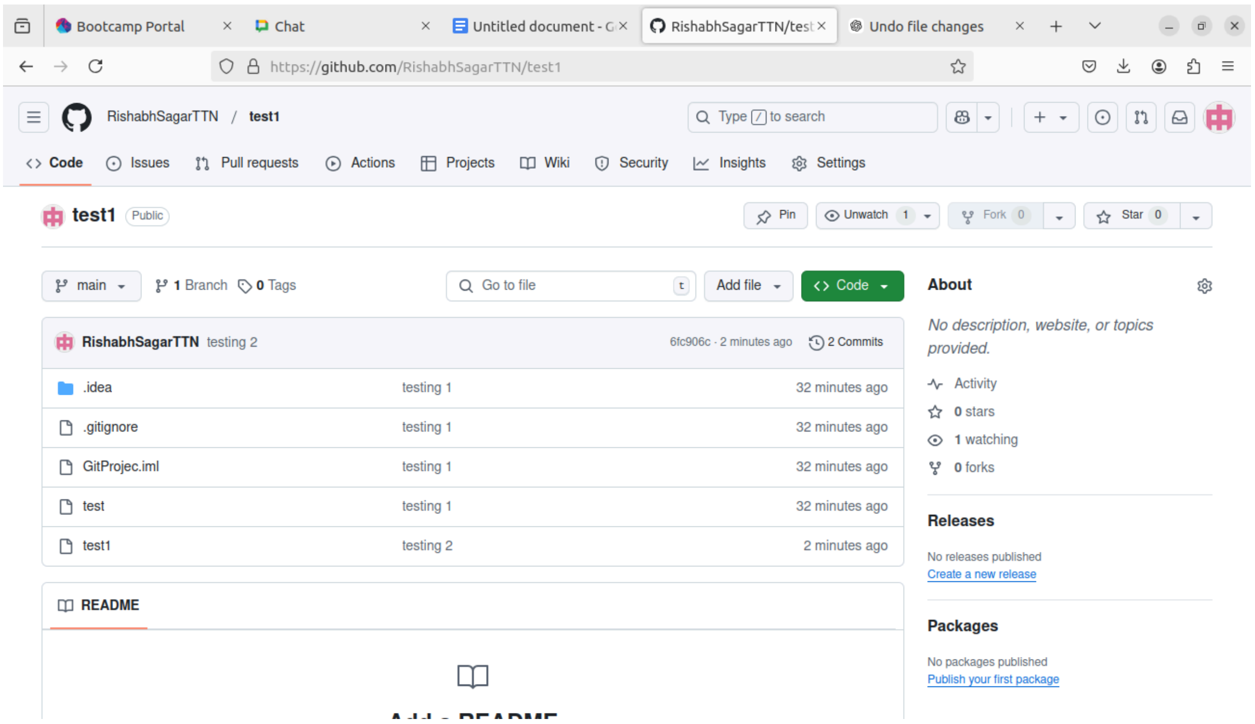
Q8)Push changes to Github

Ans.Command- git push origin <branch name to be pushed>

To use that command you have to first add the remote repository as in Q6.



```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git remote -v
origin  https://github.com/RishabhSagarTTN/test1.git (fetch)
origin  https://github.com/RishabhSagarTTN/test1.git (push)
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git branch -M main
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git push origin main
Username for 'https://github.com': RishabhSagarTTN
Password for 'https://RishabhSagarTTN@github.com':
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 8 threads
Compressing objects: 100% (10/10), done.
Writing objects: 100% (14/14), 1.73 KiB | 1.73 MiB/s, done.
Total 14 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/RishabhSagarTTN/test1.git
 * [new branch]      main -> main
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```



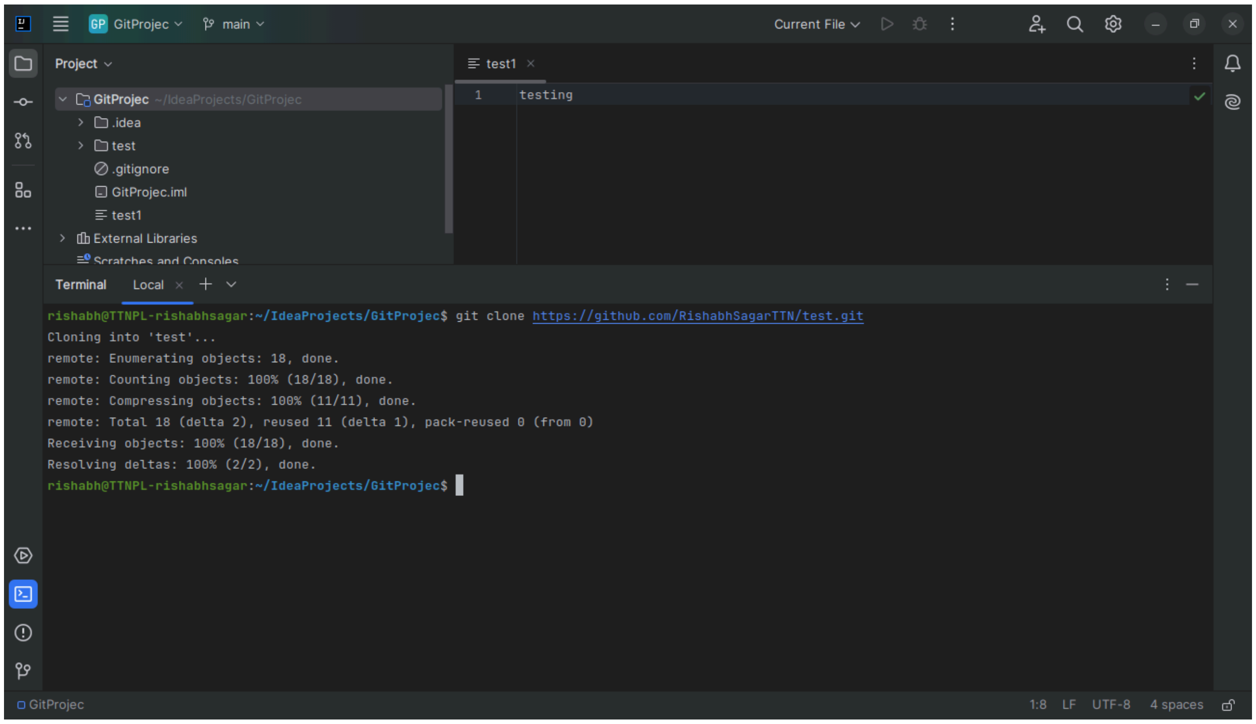
No description, website, or topics provided.

RishabhSagarTTN testing 2 — 6fc906c · 2 minutes ago — 2 Commits

.idea	testing 1	32 minutes ago
.gitignore	testing 1	32 minutes ago
GitProjec.iml	testing 1	32 minutes ago
test	testing 1	32 minutes ago
test1	testing 2	2 minutes ago

Q9)Clone the repository

Ans.Command- git clone <git repository link to be cloned>
Used to create the copy of the git repository in the local system



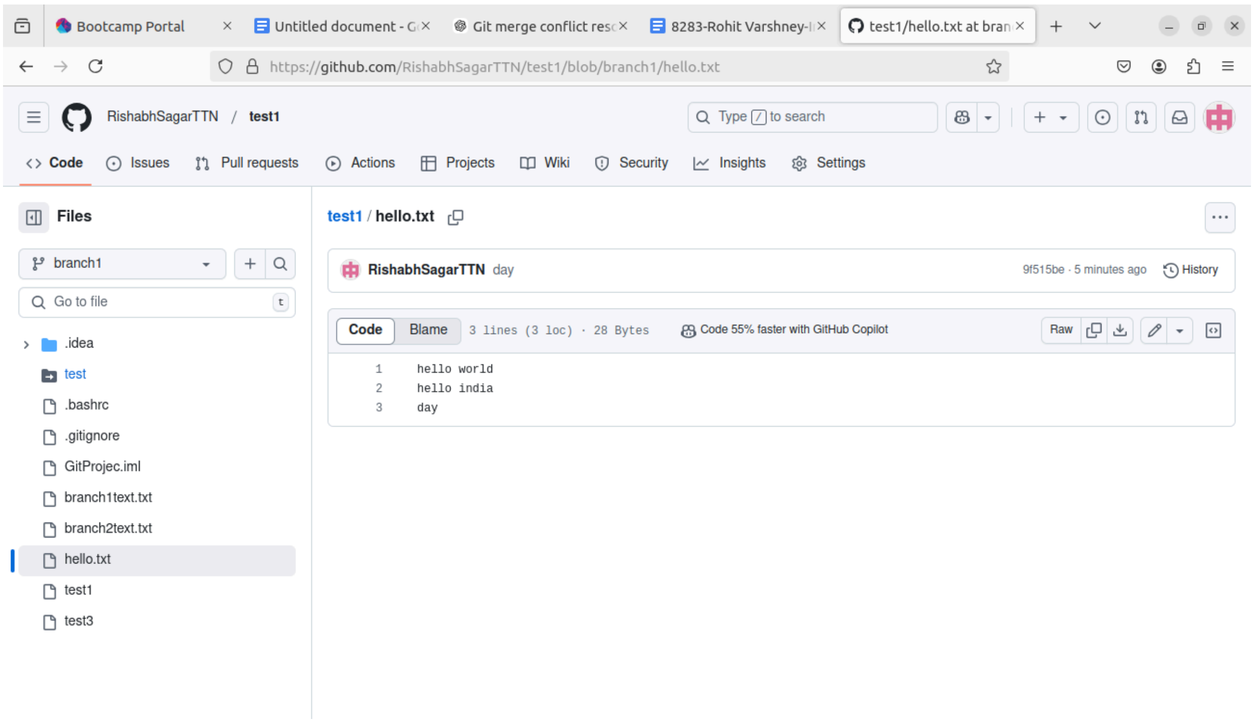
```
testing
```

```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git clone https://github.com/RishabhSagarTTN/test.git
Cloning into 'test'...
remote: Enumerating objects: 18, done.
remote: Counting objects: 100% (18/18), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 18 (delta 2), reused 11 (delta 1), pack-reused 0 (from 0)
Receiving objects: 100% (18/18), done.
Resolving deltas: 100% (2/2), done.
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

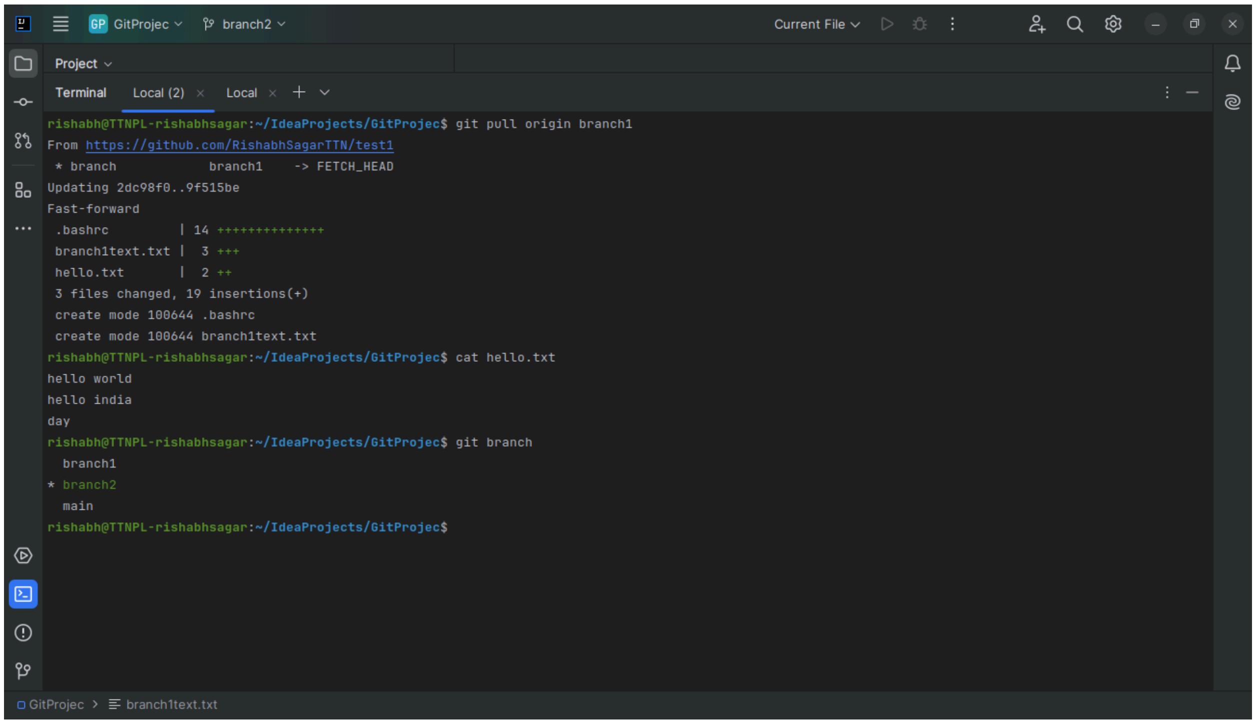
Q10)Add changes to one of the copies and pull the changes in the other.

Ans.We have the file name hello.txt in remote repository in the branch name ‘branch1’ and the same name file is in the local repository.

When we pull that branch in the local repository, the git we merge the branch with local branch hence hello.txt from remote and local will be merged.
Command- git pull origin <branch name to be pulled>

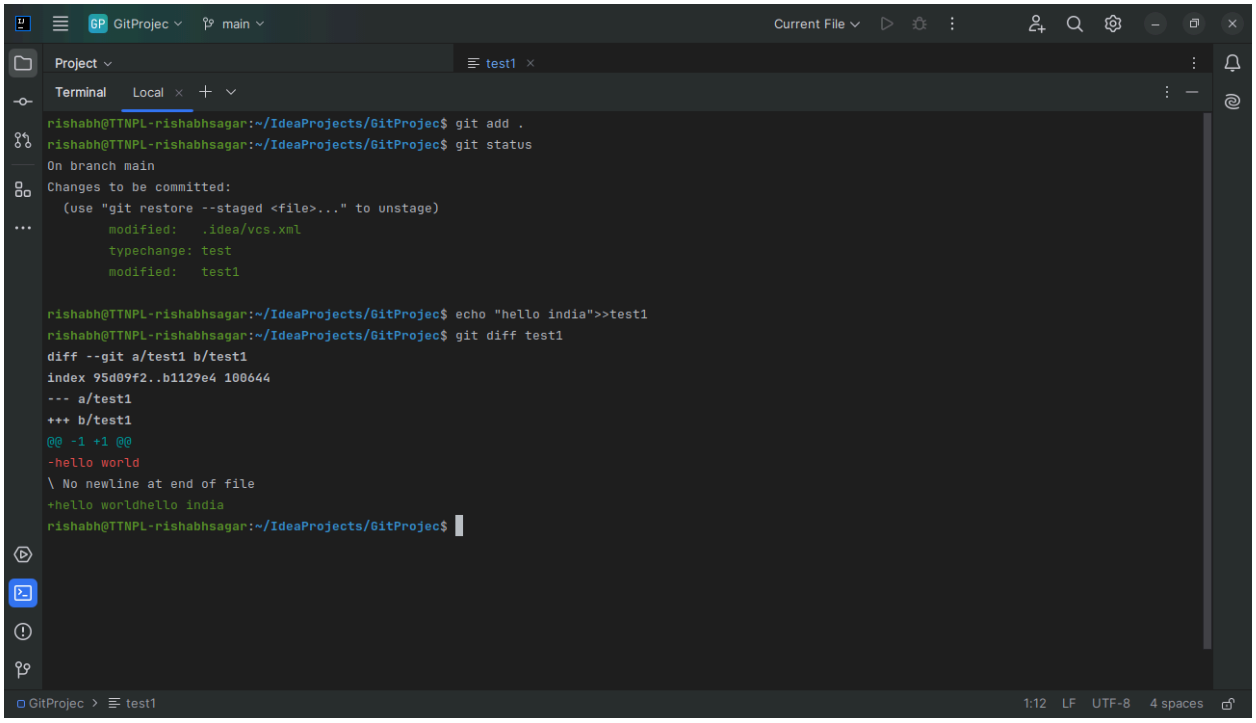


```
hello world
hello india
day
```



```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git pull origin branch1
From https://github.com/RishabhSagarTTN/test1
 * branch            branch1    -> FETCH_HEAD
Updating 2dc98f0..9f515be
Fast-forward
 .bashrc         | 14 ++++++++++++++
 branch1text.txt |  3 +++
 hello.txt       |  2 ++
 3 files changed, 19 insertions(+)
 create mode 100644 .bashrc
 create mode 100644 branch1text.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ cat hello.txt
hello world
hello india
day
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git branch
  branch1
* branch2
  main
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

Q11)Check differences between a file and its staged version
Ans. command- git diff <file name>
This will compare the file which is in the staged and in the unstaged area with the same name.

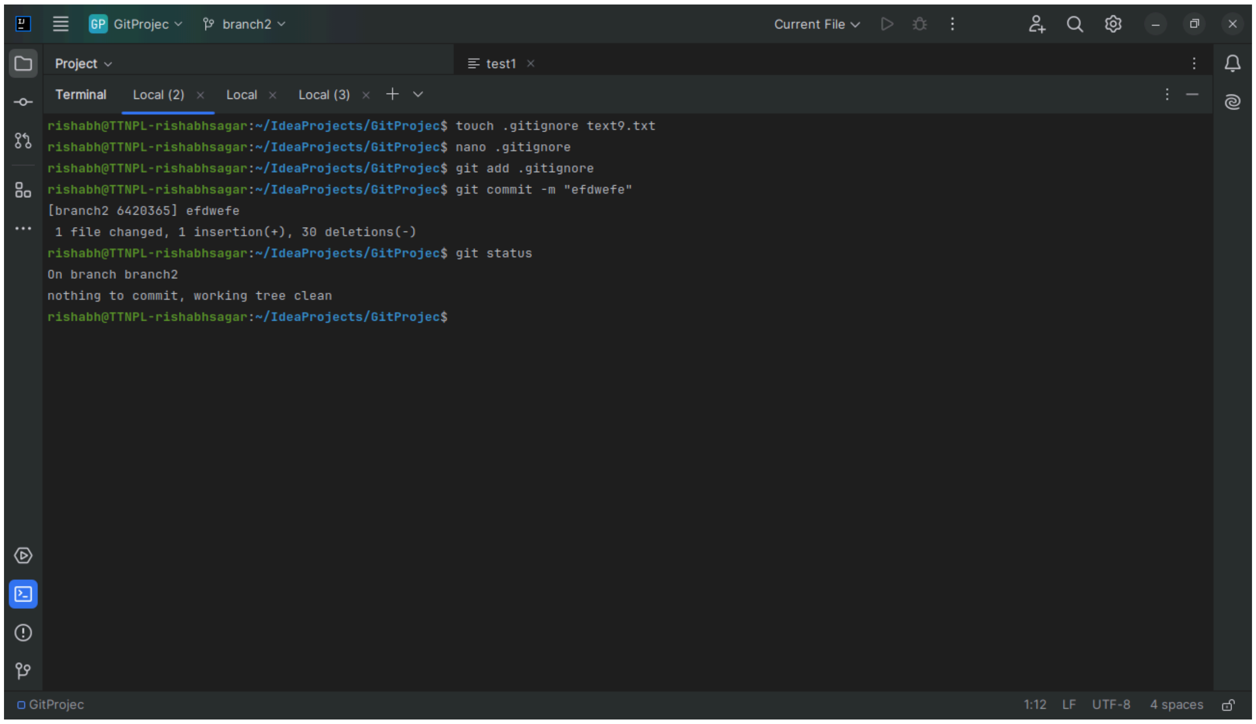


```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git add .
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git status
On branch main
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   .idea/vcs.xml
        typechange: test
        modified:   test1

rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ echo "hello india">>test1
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git diff test1
diff --git a/test1 b/test1
index 95d09f2..b1129e4 100644
--- a/test1
+++ b/test1
@@ -1 +1 @@
-hello world
\ No newline at end of file
+hello worldhello india
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

Q12)Ignore a few files to be checked in

Ans. to ignore the file or directory to be checked in we create .gitignore file. git will see this file and ignore all the file,directory that was name in this file.

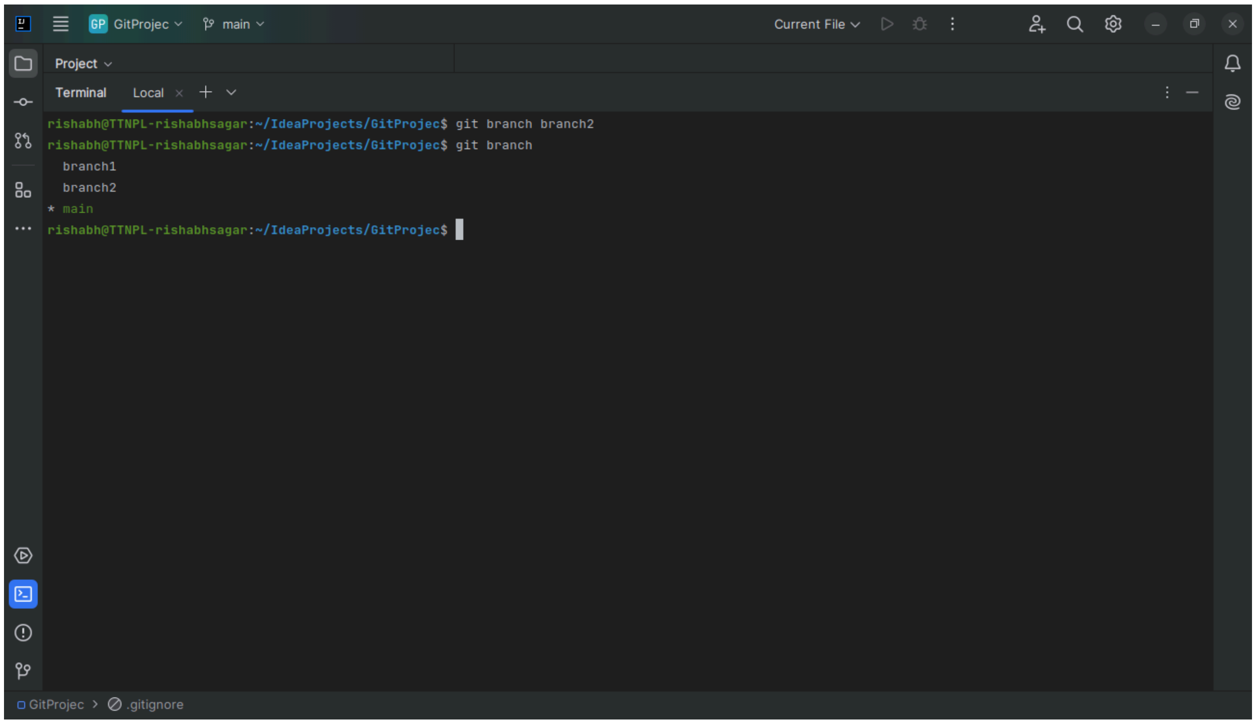
Result—test9.txt is ignored



```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ touch .gitignore text9.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ nano .gitignore
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git add .gitignore
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git commit -m "efdwefe"
[branch2 6420365] efdwefe
 1 file changed, 1 insertion(+), 30 deletions(-)
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git status
On branch branch2
nothing to commit, working tree clean
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

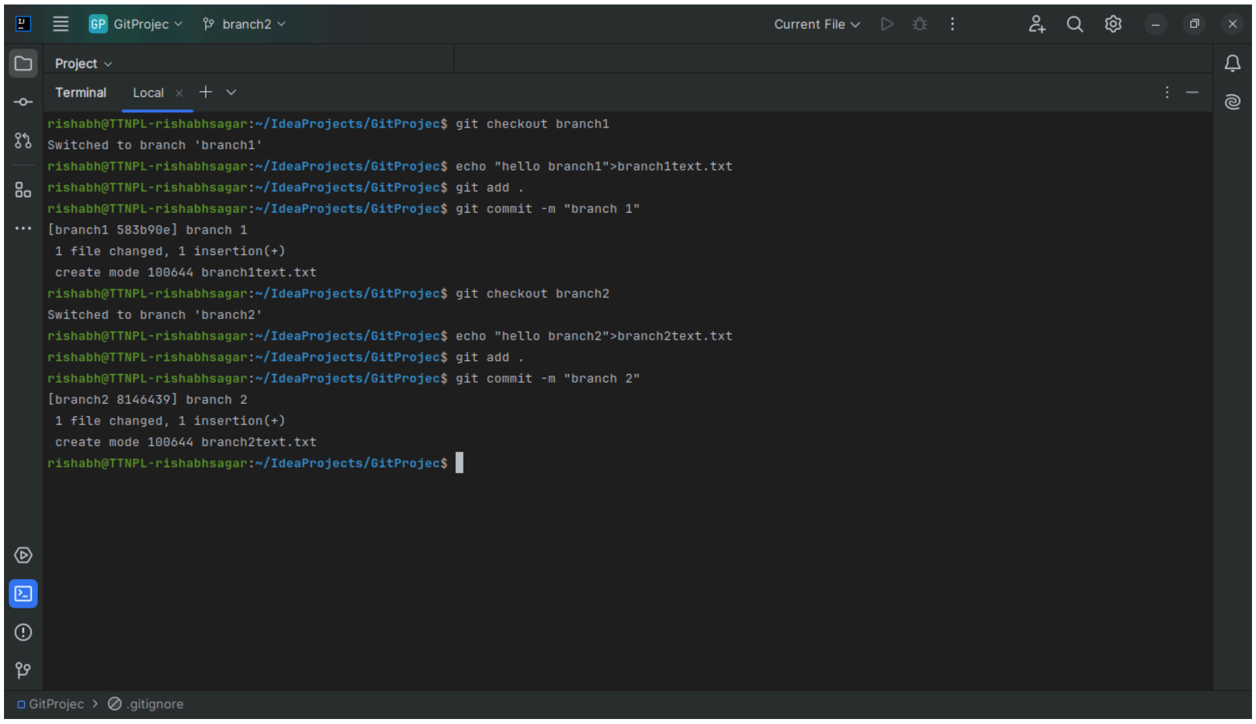
Q13)Create a new branch

Ans. command- git branch <branch name>
This will create the branch which is just the different path from that point generally used in collaboration within the same project



```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git branch branch2
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git branch
  branch1
  branch2
* main
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

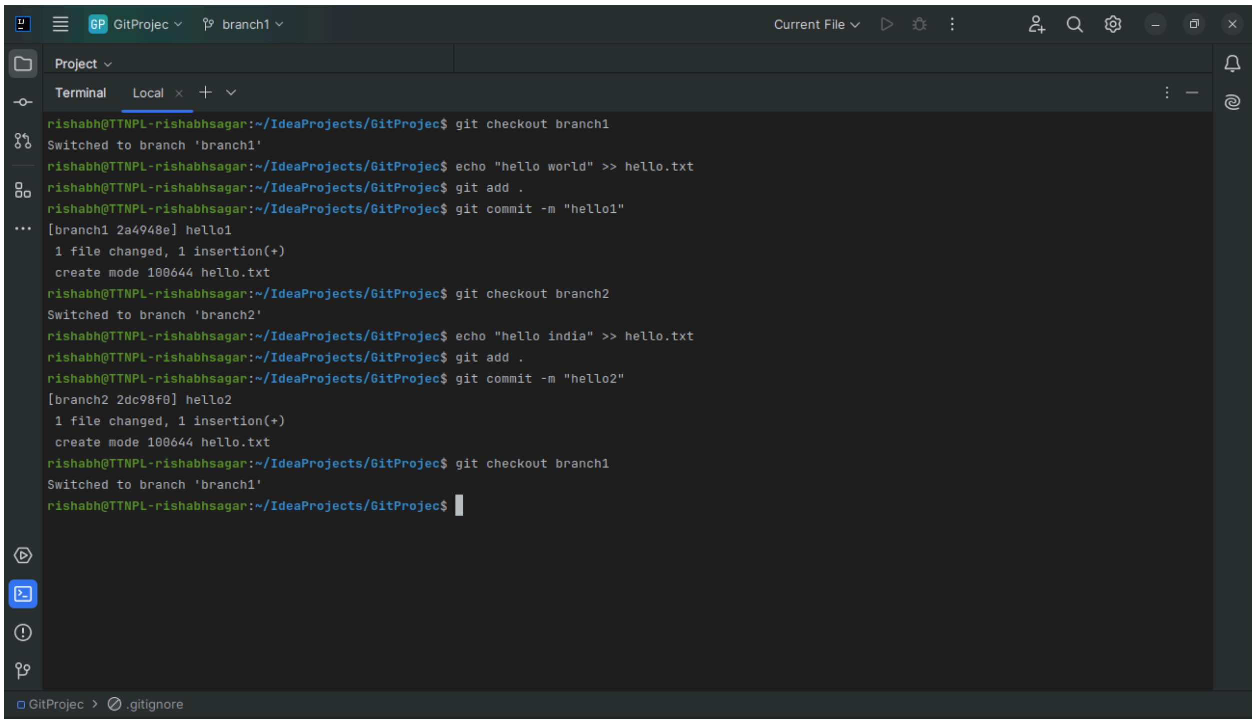
Q14)Diverge them with commits
Ans.We make the two branch from the same point and in both the branch we make the file and commit it , this will diverge point in two branches with different version/feature of it



```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch1
Switched to branch 'branch1'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ echo "hello branch1">branch1text.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git add .
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git commit -m "branch 1"
[branch1 583b90e] branch 1
 1 file changed, 1 insertion(+)
 create mode 100644 branch1text.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch2
Switched to branch 'branch2'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ echo "hello branch2">branch2text.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git add .
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git commit -m "branch 2"
[branch2 8146439] branch 2
 1 file changed, 1 insertion(+)
 create mode 100644 branch2text.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

Q15)Edit the same file at the same line on both branches and commit

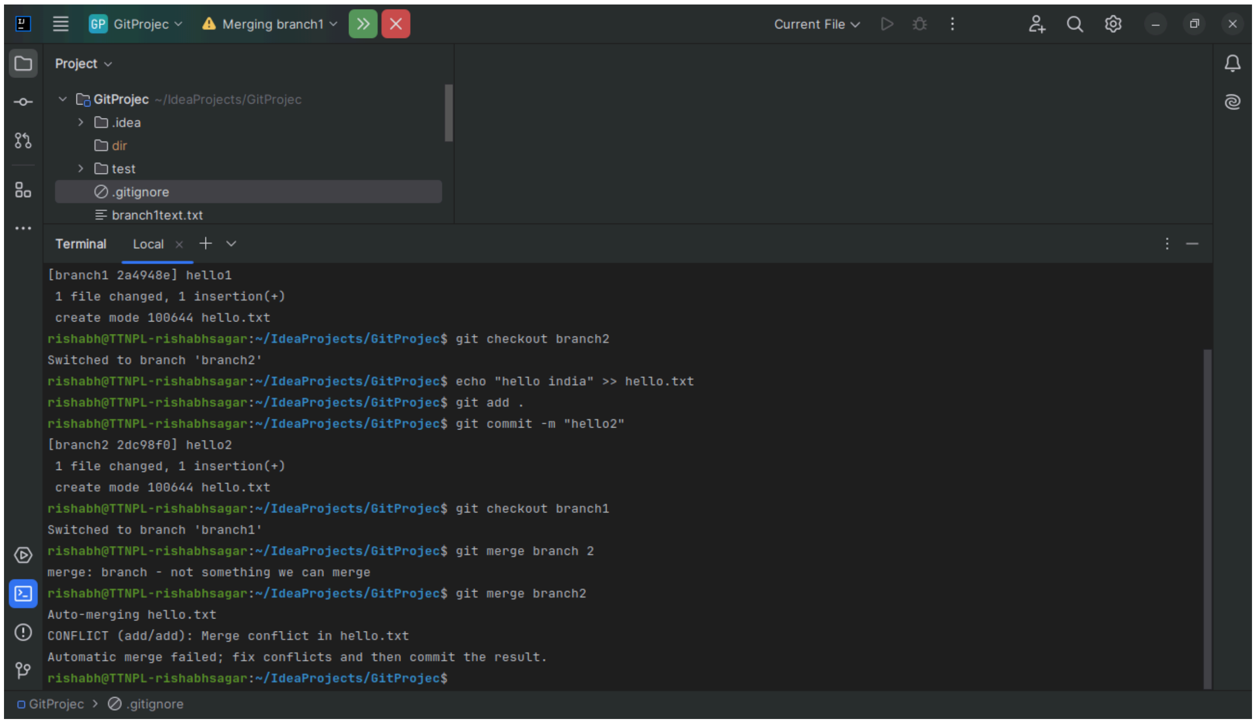
Ans.We edit the same file(hello.tt) at the same line on both branches and commit



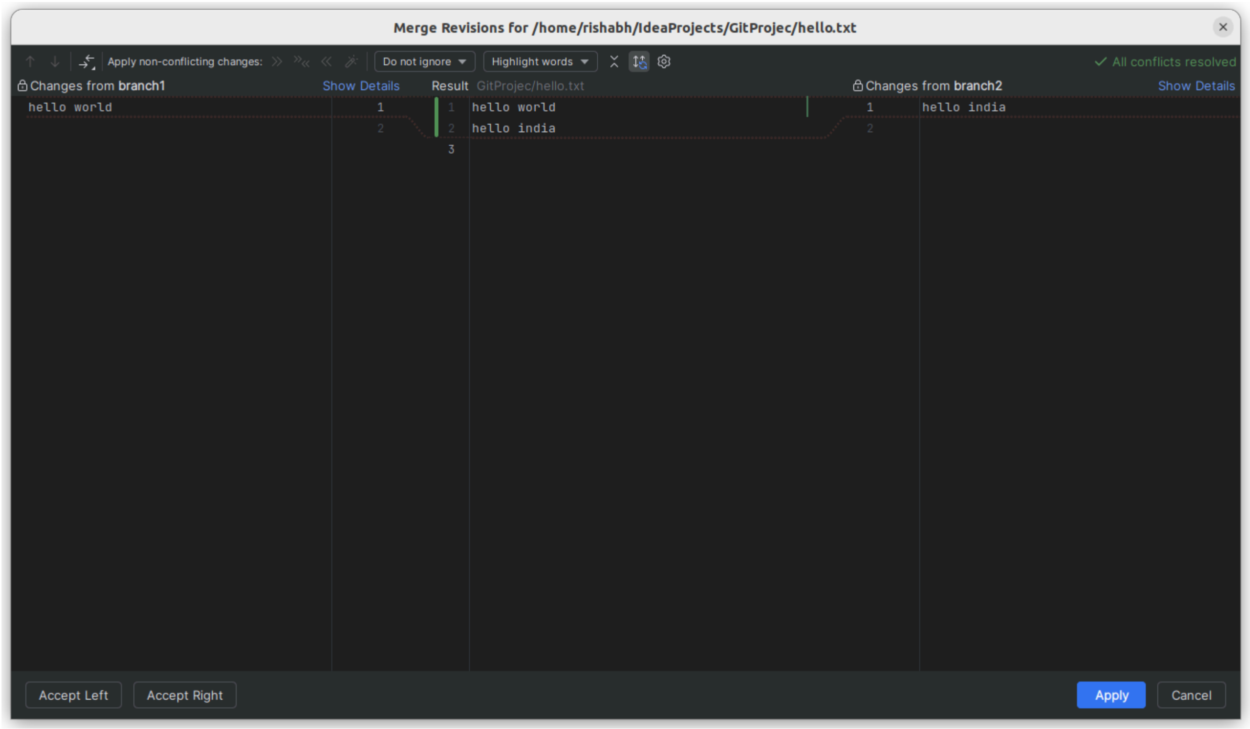
```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch1
Switched to branch 'branch1'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ echo "hello world" >> hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git add .
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git commit -m "hello1"
[branch1 2a4948e] hello1
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch2
Switched to branch 'branch2'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ echo "hello india" >> hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git add .
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git commit -m "hello2"
[branch2 2dc98f0] hello2
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch1
Switched to branch 'branch1'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

Q16)Try merging and resolve merge conflicts

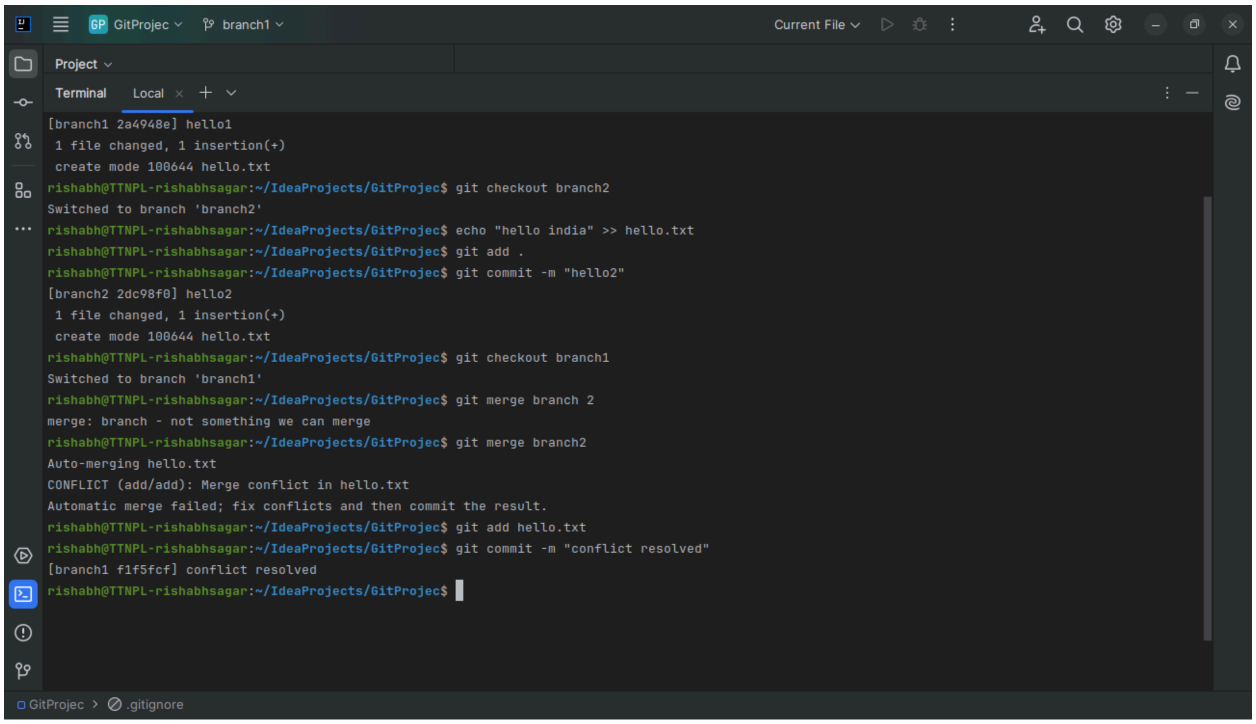
Ans.after merging the branch using git merge <branch name>, some conflict arise as two branches has the same file name and in that file, there are different line in same location and hence git doesnt know which line from which file branch it should take and hence we have to resolve it manually and then add and commit that file again after resolving.



```
[branch1 2a4948e] hello1
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch2
Switched to branch 'branch2'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ echo "hello india" >> hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git add .
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git commit -m "hello2"
[branch2 2dc98f0] hello2
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch1
Switched to branch 'branch1'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git merge branch 2
merge: branch - not something we can merge
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git merge branch2
Auto-merging hello.txt
CONFLICT (add/add): Merge conflict in hello.txt
Automatic merge failed; fix conflicts and then commit the result.
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

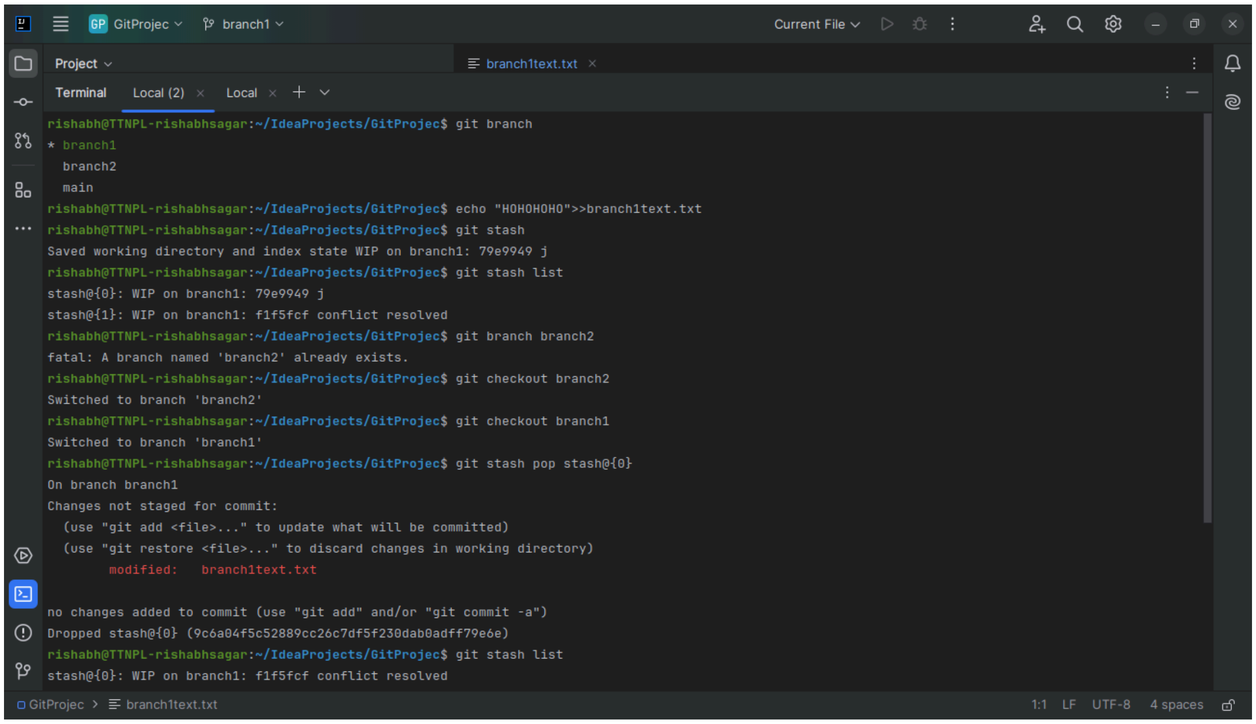

Merge Revisions for /home/rishabh/IdeaProjects/GitProjec/hello.txt

```
hello world            1  hello world          hello india
                       2  hello india
                       3
```



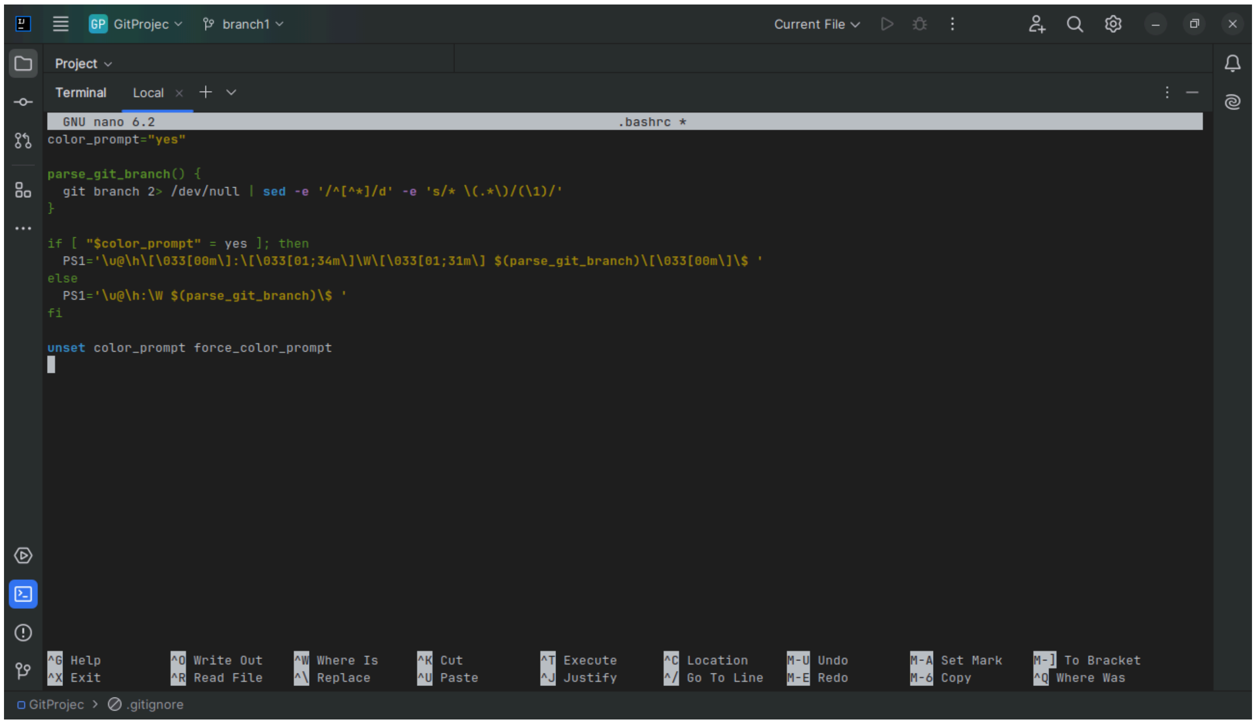
```
[branch1 2a4948e] hello1
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch2
Switched to branch 'branch2'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ echo "hello india" >> hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git add .
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git commit -m "hello2"
[branch2 2dc98f0] hello2
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch1
Switched to branch 'branch1'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git merge branch 2
merge: branch - not something we can merge
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git merge branch2
Auto-merging hello.txt
CONFLICT (add/add): Merge conflict in hello.txt
Automatic merge failed; fix conflicts and then commit the result.
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git add hello.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git commit -m "conflict resolved"
[branch1 f1f5fcf] conflict resolved
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$
```

Q17)Stash the changes and pop them
Ans. command- git stash pop/drop/apply/list/show
Used to save the staged file in the temporary directory so that when we change the branch before commit and switch back to the same branch we can bring back the file that was in the staged area.

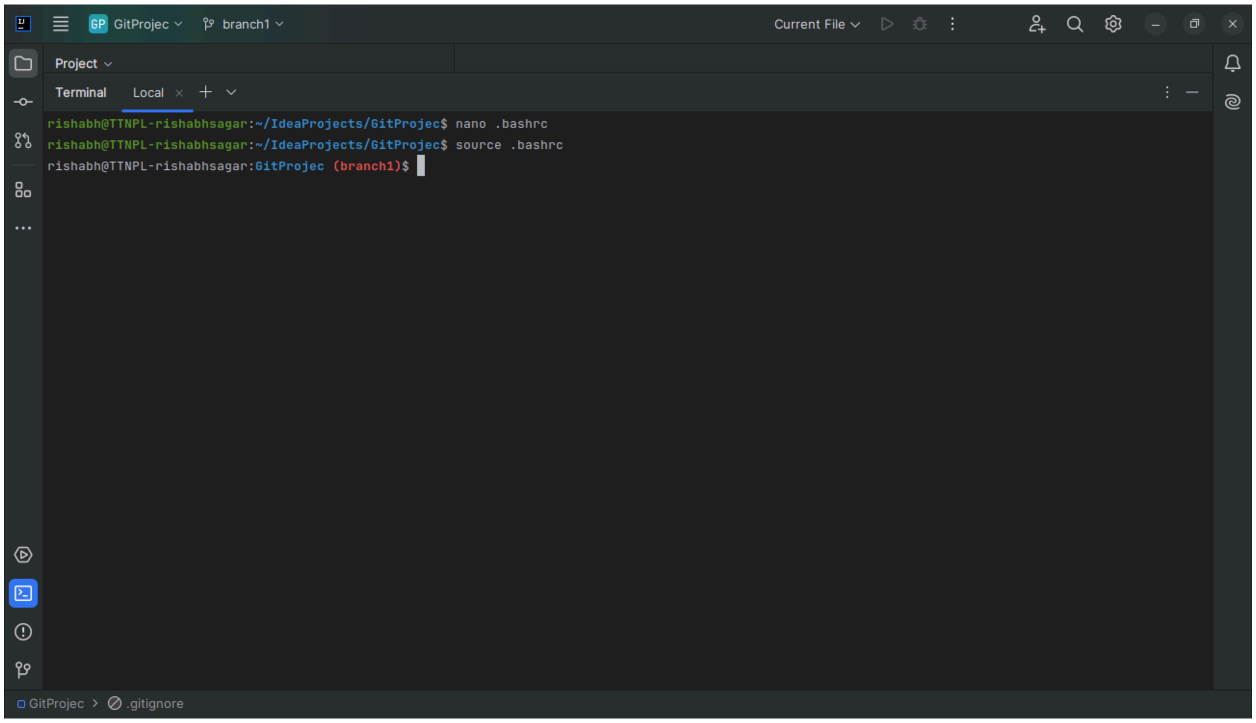


```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git branch
* branch1
  branch2
  main
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ echo "HOHOHOHO">>branch1text.txt
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git stash
Saved working directory and index state WIP on branch1: 79e9949 j
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git stash list
stash@{0}: WIP on branch1: 79e9949 j
stash@{1}: WIP on branch1: f1f5fcf conflict resolved
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git branch branch2
fatal: A branch named 'branch2' already exists.
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch2
Switched to branch 'branch2'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git checkout branch1
Switched to branch 'branch1'
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git stash pop stash@{0}
On branch branch1
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:   branch1text.txt

no changes added to commit (use "git add" and/or "git commit -a")
Dropped stash@{0} (9c6a04f5c52889cc26c7df5f230dab0adff79e6e)
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ git stash list
stash@{0}: WIP on branch1: f1f5fcf conflict resolved
```

Q18)Add the following code to your .bashrc file : color_prompt="yes" parse_git_branch() { git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*\)/(\1)/' } if ["$color_prompt" = yes]; then PS1='\u@\h\[\033[00m\]:\[\033[01;34m\]\W\[\033[01;31m\] $(parse_git_branch)\[\033[00m\]\$ ' else PS1='\u@\h:\W $(parse_git_branch)\$ ' fi unset color_prompt force_color_prompt



```
  GNU nano 6.2                                         .bashrc *
color_prompt="yes"

parse_git_branch() {
  git branch 2> /dev/null | sed -e '/^[^*]/d' -e 's/* \(.*\)/(\1)/'
}

if [ "$color_prompt" = yes ]; then
  PS1='\u@\h\[\033[00m\]:\[\033[01;34m\]\W\[\033[01;31m\] $(parse_git_branch)\[\033[00m\]\$ '
else
  PS1='\u@\h:\W $(parse_git_branch)\$ '
fi

unset color_prompt force_color_prompt

^G Help      ^O Write Out  ^W Where Is  ^K Cut      ^T Execute  ^C Location  M-U Undo  M-A Set Mark  M-] To Bracket
^X Exit      ^R Read File  ^\ Replace   ^U Paste    ^J Justify  ^/ Go To Line M-E Redo  M-6 Copy      ^Q Where Was
```



```
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ nano .bashrc
rishabh@TTNPL-rishabhsagar:~/IdeaProjects/GitProjec$ source .bashrc
rishabh@TTNPL-rishabhsagar:GitProjec (branch1)$
```