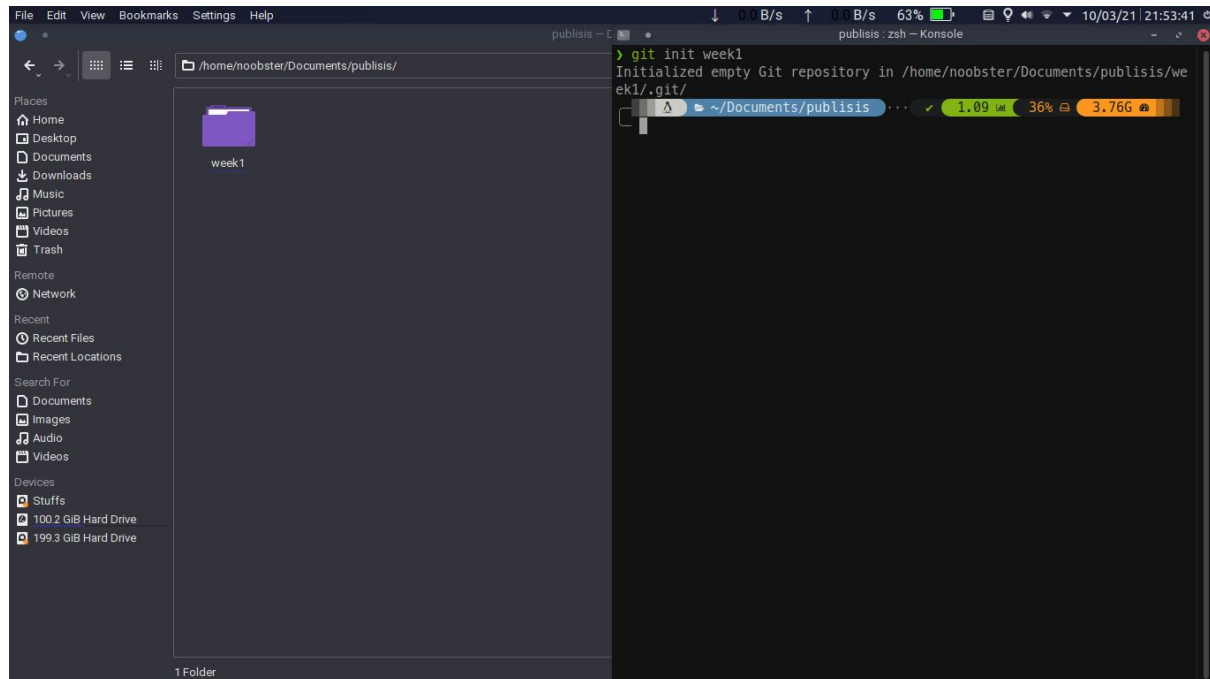


Shivam Sahu([shivam14sahu@gmail.com](mailto:shivam14sahu@gmail.com))

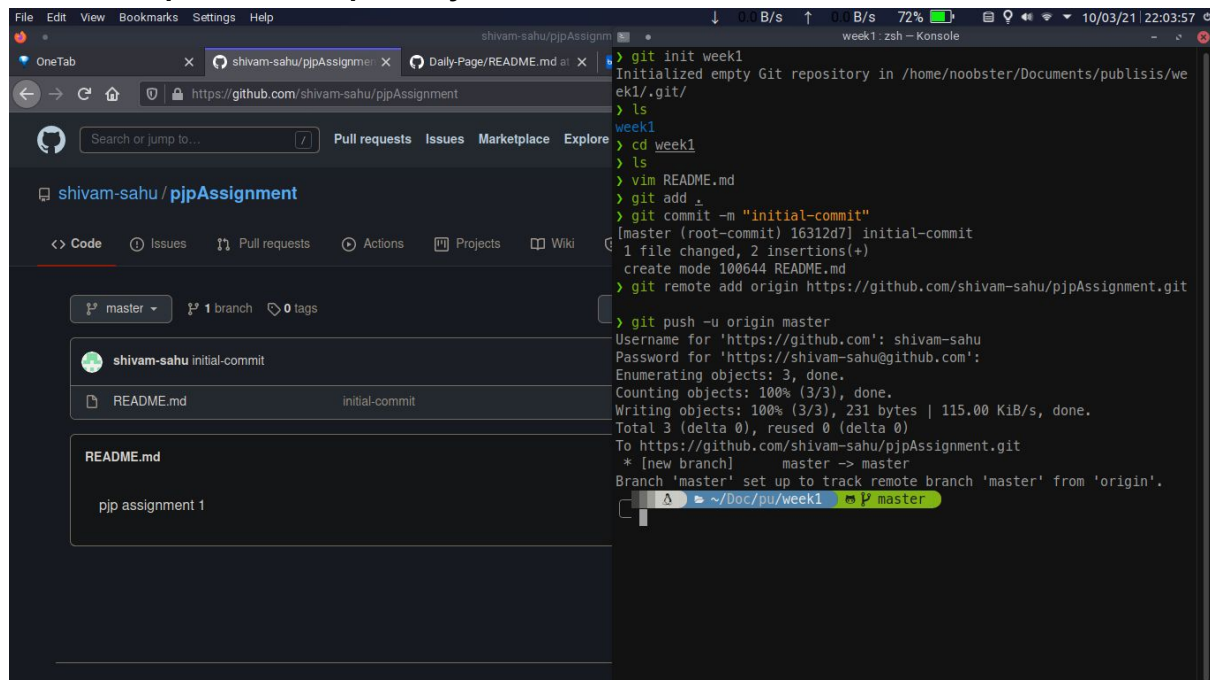
## Github Assignment 1

### 1. Basic usage of cli

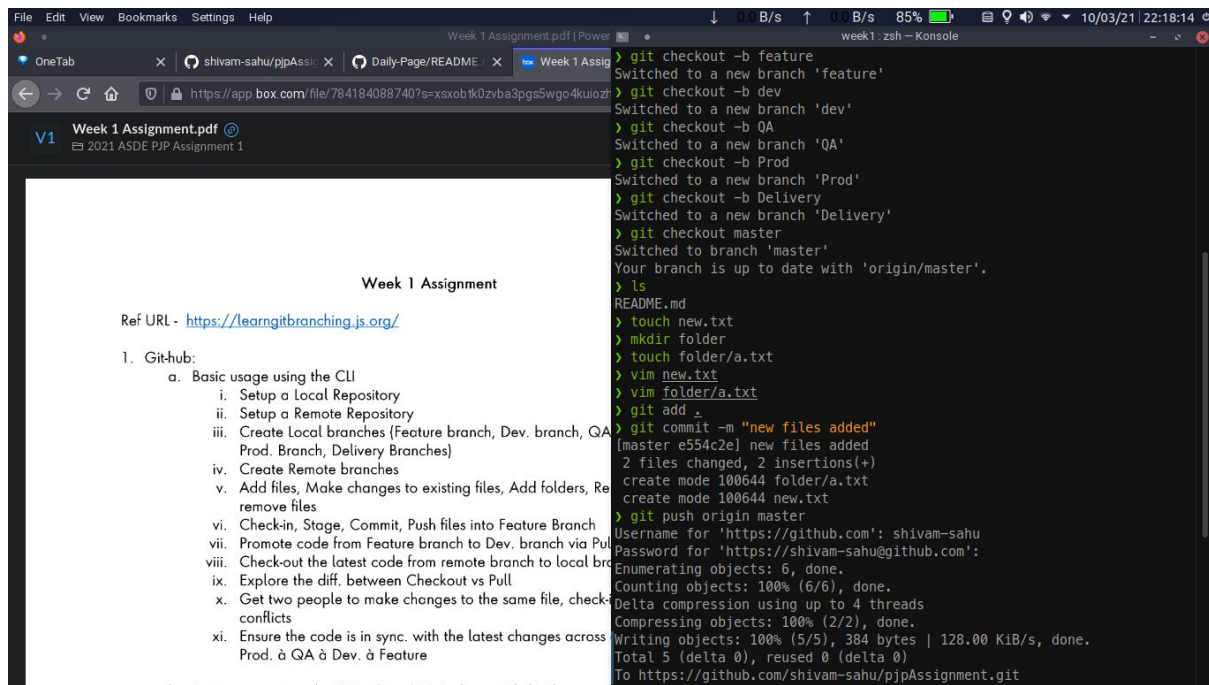
#### a. Setup a Local Repository



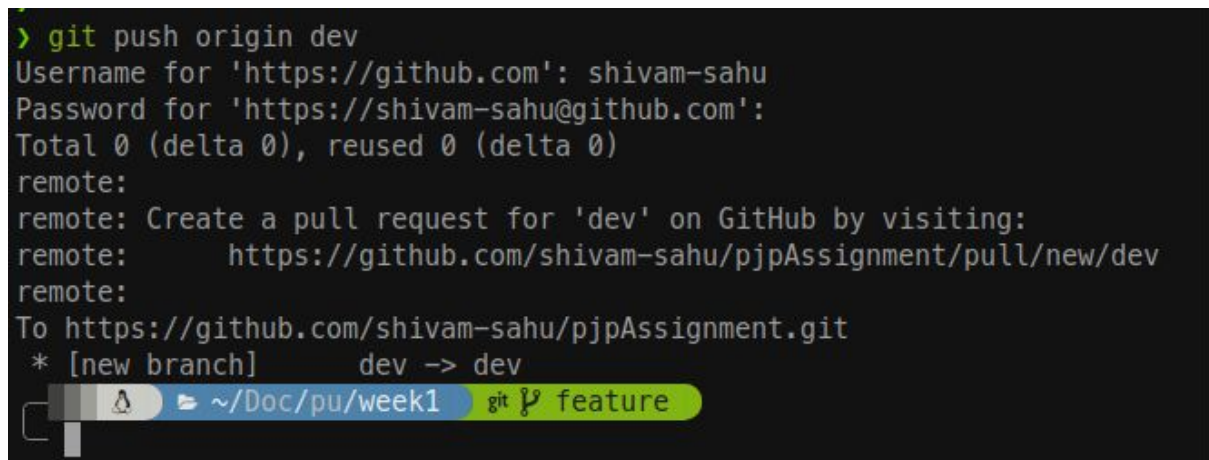
#### b. Setup a remote repository



### C. Create local branches



## D. Create remote branches

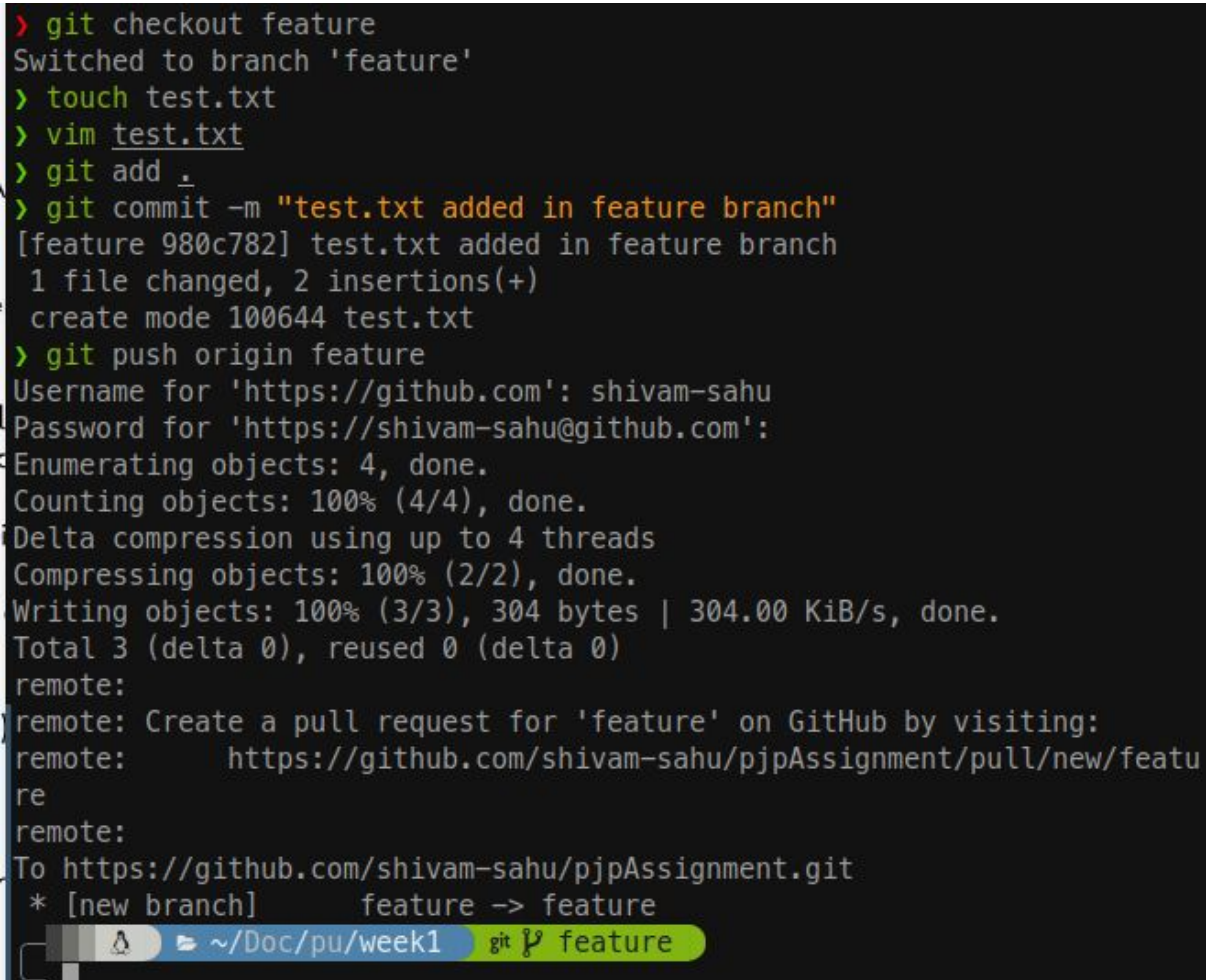


## E. Add files/folders, make changes to files, remove files/folders

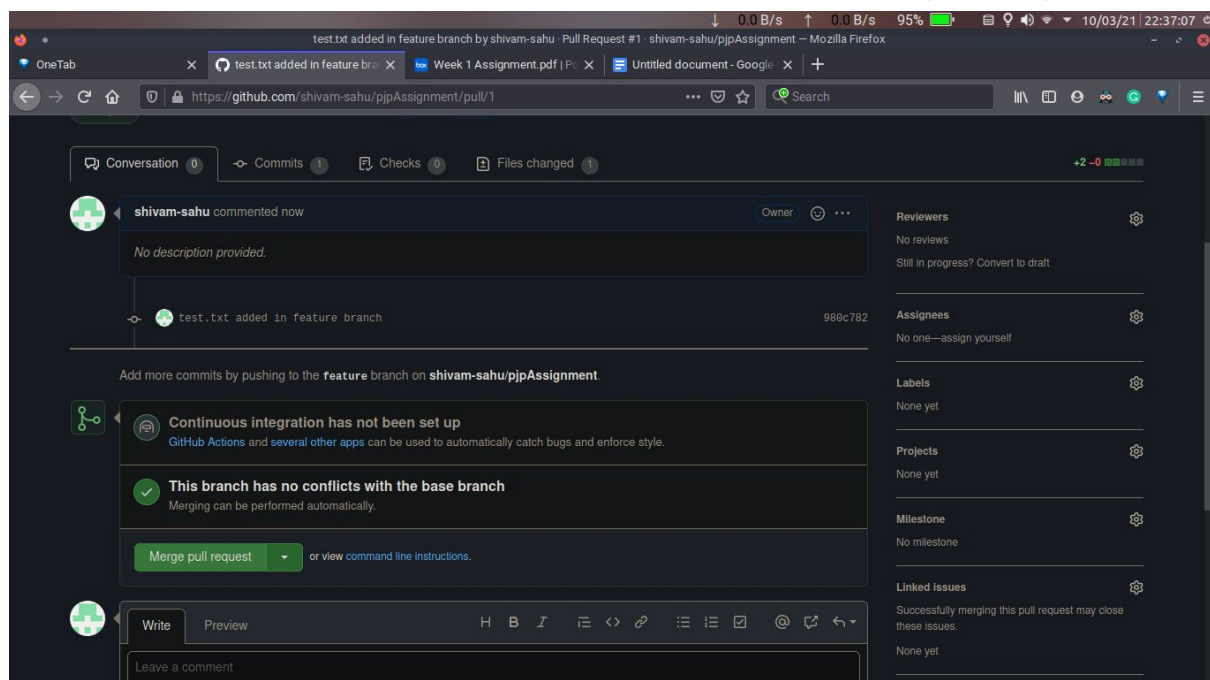
```
File Edit View Bookmarks Settings Help
Untitled document - Google
OneTab x shivam-sahu/pjpAssi... x Daily-Page/README x Week 1 Assig
https://docs.google.com/document/d/1up6wRAYOSq-dgl3ZHDdCjBuBqGIWJ
Untitled document ☆ ⓘ
File Edit View Insert Format Tools Add-ons Help Last edit was seconds ago
Normal text Arial 11 B I U A
Headings that you add to the document will appear here.
Week 1 Assignment
Ref URL: https://learningbranching.io/
1. Goals:
    a. Basic usage using the CLI
    b. Setup a Local Repository
    c. Create Local Branches: Feature Branch, Dev. branch, Prod. Branch, Delivery Branches
    d. Create Remote Branches
    e. Add files, make changes to existing files, Add files, remove files
    f. Checkin, Stage, Commit, Push files into Feature Branch
    g. Promote code from Feature branch to Dev. branch
    h. Checkout the latest code from remote branch to local
    i. Explore the diff. between Checkout vs Pull
    j. Get two people to make changes to the same file, < conflict
    k. Ensure the code is in sync with the latest changes
    l. Prod. & QA & Dev. & Feature
> vim new.txt
> vim folder/a.txt
> git add .
> git commit -m "new files added"
[master e554c2e] new files added
2 files changed, 2 insertions(+)
create mode 100644 folder/a.txt
create mode 100644 new.txt
> git push origin master
Username for 'https://github.com': shivam-sahu
Password for 'https://shivam-sahu@github.com':
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (5/5), 384 bytes | 128.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0)
To https://github.com/shivam-sahu/pjpAssignment.git
16312d7..e554c2e master -> master
> ls
folder new.txt README.md
> rmdir -r folder
rmdir: invalid option -- 'r'
Try 'rmdir --help' for more information.
> rm -r folder
> ls
new.txt README.md
> rm -r new.txt
> git add .
> git commit -m "files deleted"
[master c9bae3a] files deleted
2 files changed, 2 deletions(-)
delete mode 100644 folder/a.txt
delete mode 100644 new.txt
~/Doc/pu/week1 master +1
```

## F. Check in, stage, commit, push changes into feature branch

```
> git checkout feature
Switched to branch 'feature'
> touch test.txt
> vim test.txt
> git add .
> git commit -m "test.txt added in feature branch"
[feature 980c782] test.txt added in feature branch
1 file changed, 2 insertions(+)
create mode 100644 test.txt
> git push origin feature
Username for 'https://github.com': shivam-sahu
Password for 'https://shivam-sahu@github.com':
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 304 bytes | 304.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'feature' on GitHub by visiting:
remote:   https://github.com/shivam-sahu/pjpAssignment/pull/new/feature
remote:
To https://github.com/shivam-sahu/pjpAssignment.git
* [new branch]      feature -> feature
```

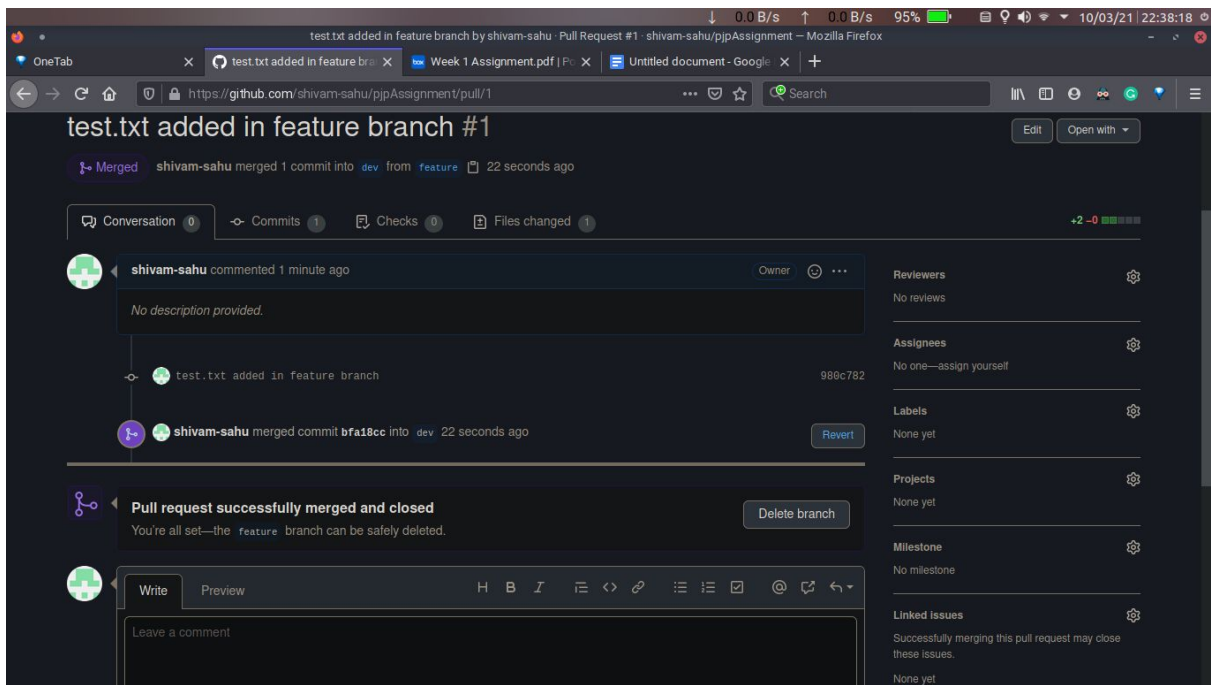


## G. Promote code from feature branch to dev branch via pull requests.



The screenshot shows a GitHub Pull Request (PR) page for the repository 'shivam-sahu/pjpAssignment'. The PR is titled 'test.txt added in feature branch' and is associated with commit '980c782'. The interface includes a 'Conversation' tab with a comment from the owner, 'shivam-sahu', stating 'No description provided.' Below the comment, a message indicates that 'Continuous integration has not been set up' and that 'This branch has no conflicts with the base branch', allowing for automatic merging. A 'Merge pull request' button is visible. On the right side, there are sections for 'Reviewers', 'Assignees', 'Labels', 'Projects', 'Milestone', and 'Linked issues', all of which are currently empty or set to 'None yet'. The bottom of the page shows a 'Write' tab for adding a comment.





## I. Checkout latest code from remote to local

```
> git fetch origin master
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), 617 bytes | 617.00 KiB/s, done.
From https://github.com/shivam-sahu/pjpAssignment
 * branch                master      -> FETCH_HEAD
   c9bae3a..f665861      master      -> origin/master
> git checkout master
Already on 'master'
Your branch is behind 'origin/master' by 3 commits, and can be fast-forwarded.
(use "git pull" to update your local branch)
```

## H. Explore difference between checkout and pull

Ans.

**checkout**-> **Fetches** the latest changes. You should already have this repo downloaded. It **does not merge** those new changes but makes your working directory reflect them. You can merge them at your leisure later.

**pull**-> fetches the latest changes from remote and merges them into the local branch of the same name.

```
> git fetch origin master
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), 617 bytes | 617.00 KiB/s, done.
From https://github.com/shivam-sahu/pjpAssignment
 * branch            master       -> FETCH_HEAD
   c9bae3a..f665861  master       -> origin/master
> git checkout master
Already on 'master'
Your branch is behind 'origin/master' by 3 commits, and can be fast-forwarded.
  (use "git pull" to update your local branch)
> ls
README.md
> git pull origin master
From https://github.com/shivam-sahu/pjpAssignment
 * branch            master       -> FETCH_HEAD
Updating c9bae3a..f665861
Fast-forward
 test.txt | 2 ++
 1 file changed, 2 insertions(+)
 create mode 100644 test.txt
```

- I. **Get two people make changes to same file, checkin & handle merge conflicts**
  - a. Let's first create 2 new branches and make changes to the test.txt file.

```

> git branch person1
> git branch person2
> git checkout person1
Switched to branch 'person1'
> ls
README.md  test.txt
> cat test.txt
random text file

> echo "added by person1" >>test.txt
> git add .
> git commit -m "text added by person1"
[person1 d8c81dc] text added by person1
1 file changed, 1 insertion(+)
> git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
> cat test.txt
random text file

> git checkout person2
Switched to branch 'person2'
> cat test.txt
random text file

> echo "added by person2">>test.txt
> git add .
> git commit -m "text added by person2"
[person2 005e2a2] text added by person2
1 file changed, 1 insertion(+)

```

Now let's try to merge changes from both the branches to feature branch

```

> git checkout feature
Switched to branch 'feature'
> git merge person1
Updating bfa18cc..d8c81dc
Fast-forward
 test.txt | 1 +
1 file changed, 1 insertion(+)
> git merge person2
Auto-merging test.txt
CONFLICT (content): Merge conflict in test.txt
Automatic merge failed; fix conflicts and then commit the result.

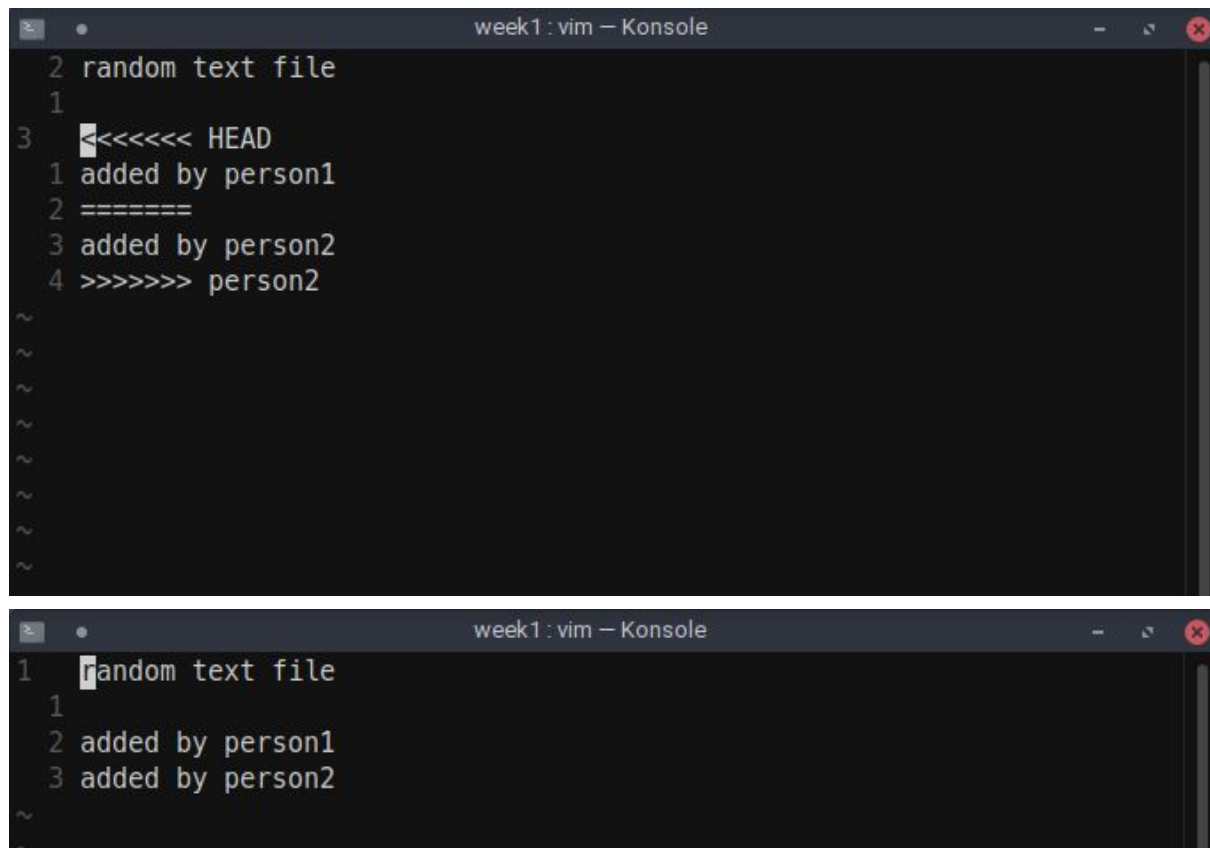
```

Now try to fix changes manually

```
> git status
On branch feature
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
        both modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")
> vim test.txt
```



The top screenshot shows a vim editor window titled "week1: vim - Konsole". The content of the file is as follows:

```
2 random text file
1
3 <<<<<<< HEAD
1 added by person1
2 =====
3 added by person2
4 >>>>>>> person2
~
~
~
~
~
~
~
```

The bottom screenshot shows the same vim editor window after the conflict has been resolved. The content is now:

```
1 random text file
1
2 added by person1
3 added by person2
~
~
```

Now commit changes and merge them



```

> git status
On branch feature
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
    both modified:   test.txt

no changes added to commit (use "git add" and/or "git commit -a")
> git add .
> git commit -m "merging files"
[feature 9f76483] merging files
> cat test.txt
random text file

added by person1
added by person2

```

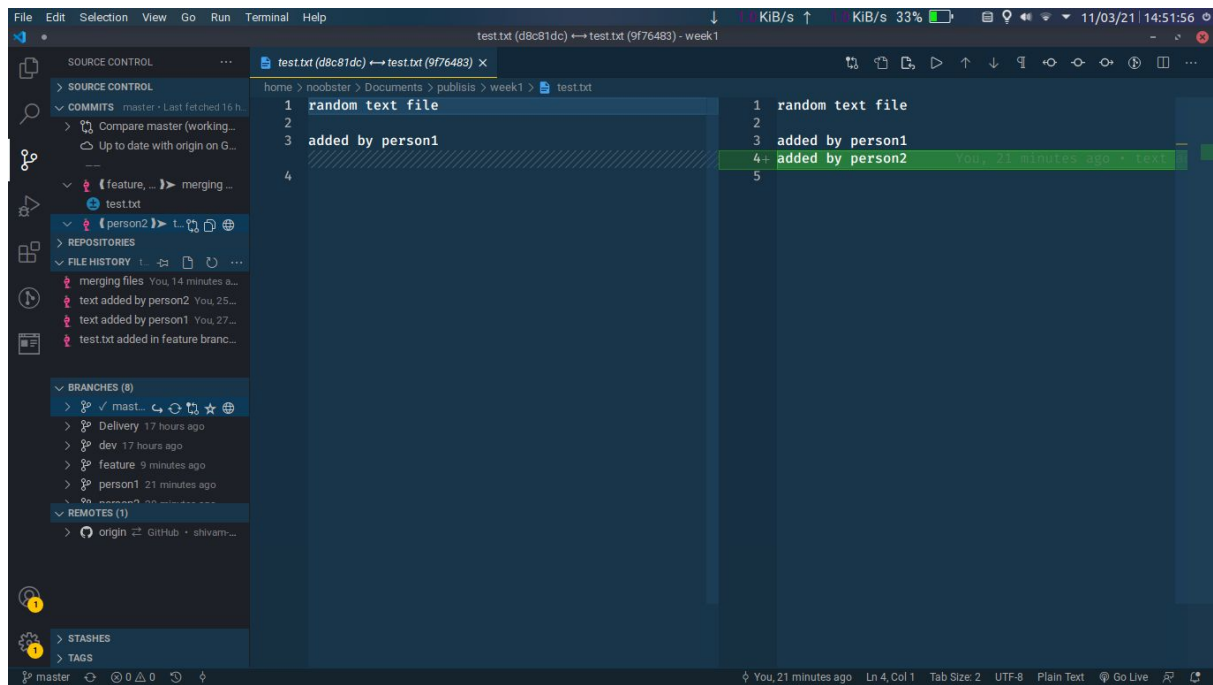
## J. Ensure code is sync across all branches

```

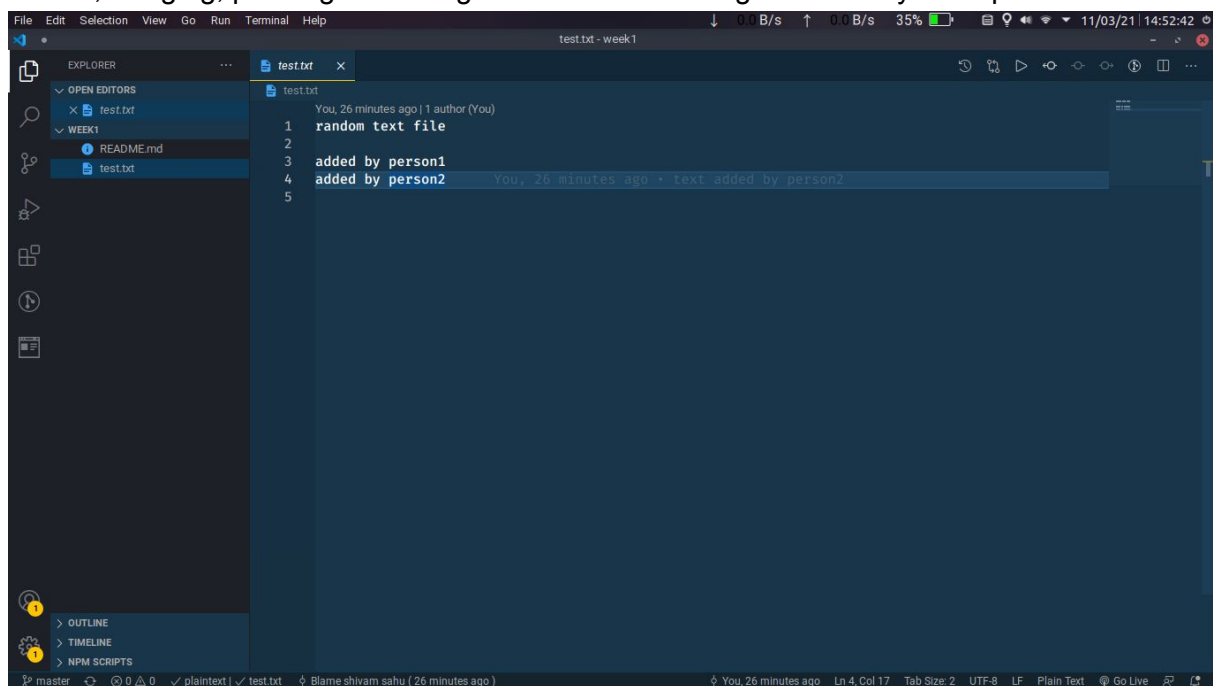
> git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
> git merge feature
Updating f665861..9f76483
Fast-forward
 test.txt | 2 ++
 1 file changed, 2 insertions(+)
> git push origin master
Username for 'https://github.com': shivam-sahu
Password for 'https://shivam-sahu@github.com':
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (9/9), 853 bytes | 213.00 KiB/s, done.
Total 9 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/shivam-sahu/pjpAssignment.git
 f665861..9f76483 master -> master

```

## 2. Perform above actions using vscode/git gui



In vs code we can use different extensions like git lens for action like staging, commit, rebase, merging, pushing etc. and git blame to see changes added by each person.



### 3. On ground day to day scenarios

#### a. Git reset/revert changes (soft/hard reset)

##### --soft reset

Does not touch the index file or the working tree at all (but resets the head to <commit>, just like all modes do). This leaves all your changed files "Changes to be committed", as git status would put it.

```

> git log --oneline
> ls
README.md  test.txt
> echo "one more change" >>test.txt
> cat test.txt
random text file

added by person1
added by person2
one more change
> git log --oneline
> git reset --soft f665861
> git status
On branch feature
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   test.txt

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:   test.txt

> cat test.txt
random text file

added by person1
added by person2
one more change

```

Git log before soft reset

```

9f76483 (HEAD -> feature, origin/master, master) merging files
005e2a2 (person2) text added by person2
d8c81dc (person1) text added by person1
f665861 Merge pull request #2 from shivam-sahu/dev
bfa18cc (origin/dev) Merge pull request #1 from shivam-sahu/feature
980c782 (origin/feature) test.txt added in feature branch
c9bae3a files deleted
e554c2e new files added
16312d7 (dev, QA, Prod, Delivery) initial-commit
(END)

```

Git log after soft reset

```

f665861 (HEAD -> feature) Merge pull request #2 from shivam-sahu/dev
bfa18cc (origin/dev) Merge pull request #1 from shivam-sahu/feature
980c782 (origin/feature) test.txt added in feature branch
c9bae3a files deleted
e554c2e new files added
16312d7 (dev, QA, Prod, Delivery) initial-commit
(END)

```

### --hard reset example

Making changes to test.txt

```
> ls
README.md  test.txt
> cat test.txt
random text file

added by person1
added by person2
> echo "hard reset example" >>test.txt
> cat test.txt
random text file

added by person1
added by person2
hard reset example
> git log --oneline
```

Git log before hard reset

```
9f76483 (HEAD -> feature, origin/master, master) merging files
005e2a2 (person2) text added by person2
d8c81dc (person1) text added by person1
f665861 Merge pull request #2 from shivam-sahu/dev
bfa18cc (origin/dev) Merge pull request #1 from shivam-sahu/feature
980c782 (origin/feature) test.txt added in feature branch
c9bae3a files deleted
e554c2e new files added
16312d7 (dev, QA, Prod, Delivery) initial-commit
(END)
```

Changes in file is also gone in hard reset

```
> cat test.txt
random text file

added by person1
added by person2
hard reset example
> git log --oneline
> git reset --hard f665861
HEAD is now at f665861 Merge pull request #2 from shivam-sahu/dev
> git log --oneline
> cat test.txt
random text file
```

Git log after hard reset



```
f665861 (HEAD -> feature) Merge pull request #2 from shivam-sahu/dev
bfa18cc (origin/dev) Merge pull request #1 from shivam-sahu/feature
980c782 (origin/feature) test.txt added in feature branch
c9bae3a files deleted
e554c2e new files added
16312d7 (dev, QA, Prod, Delivery) initial-commit
```

## b. Git stash

Let person 1 make changes to test.txt file and commit it

```
> ls
README.md  test.txt
Week 1 Assignment.pdf | Powered by Box
random text file

added by person1
added by person2
> echo "stash example added by person1" >>test.txt
> git add .
> git commit -m "stash example person1"
[person1 3fdaa71] stash example person1
1 file changed, 1 insertion(+)
```

Now let master make changes and it tries to merge person after making changes.

```
> git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
> cat test.txt
random text file

added by person1
added by person2
> echo "stash example added by master">>test.txt
> git merge person1
Updating 9f76483..3fdaa71
error: Your local changes to the following files would be overwritten by
merge:
    test.txt
Please commit your changes or stash them before you merge.
Aborting
```

Performing git stash



```

> git stash
Saved working directory and index state WIP on master: 9f76483 merging f
iles
> git merge person1
Updating 9f76483..3fdaa71
Fast-forward
 test.txt | 1 +
 1 file changed, 1 insertion(+)
> git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean

```

### 3. Git rebase

Adding new.txt file by master

```

> touch new.txt
> echo "new txt added by master" >> new.txt
> git add .
> git commit -m "new.txt added"
[master 15740fb] new.txt added
 1 file changed, 1 insertion(+)
 create mode 100644 new.txt

```

Adding a.txt file by person1

```

> git checkout person1
Switched to branch 'person1'
> ls
README.md  test.txt
> touch a.txt
> echo "adding text to a.txt" >> a.txt
> git add .
> git commit -m "a.txt added "
[person1 605e410] a.txt added
 1 file changed, 1 insertion(+)
 create mode 100644 a.txt
> git log
> git log --oneline

```

Git log of person1 branch before rebase

```

605e410 (HEAD -> person1) a.txt added
b4d778b (origin/master) changes made
3fdaa71 stash example person1
9f76483 (feature) merging files
005e2a2 (person2) text added by person2
d8c81dc text added by person1
f665861 Merge pull request #2 from shivam-sahu/dev
bfa18cc (origin/dev) Merge pull request #1 from shivam-sahu/feature
980c782 (origin/feature) test.txt added in feature branch
c9bae3a files deleted
e554c2e new files added
16312d7 (dev, QA, Prod, Delivery) initial-commit
(END)

```

Applying git rebase

```

> git rebase master
First, rewinding head to replay your work on top of it...
Applying: a.txt added
> git log
> git log --oneline

```

Git log after rebase

```

a9d8465 (HEAD -> person1) a.txt added
15740fb (master) new.txt added
b4d778b (origin/master) changes made
3fdaa71 stash example person1
9f76483 (feature) merging files
005e2a2 (person2) text added by person2
d8c81dc text added by person1
f665861 Merge pull request #2 from shivam-sahu/dev
bfa18cc (origin/dev) Merge pull request #1 from shivam-sahu/feature
980c782 (origin/feature) test.txt added in feature branch
c9bae3a files deleted
e554c2e new files added
16312d7 (dev, QA, Prod, Delivery) initial-commit
(END)

```

#### 4. Git log ,reflog, status

## Git log

```
commit b4d778ba076f44206bd8d0dde8ae05fcbc178ebe (HEAD -> master, origin/  
master, person1)  
Author: shivam sahu <shivam14sahu@gmail.com>  
Date: Thu Mar 11 15:59:21 2021 +0530  
  
    changes made  
  
commit 3fdaa7126958e3a25cb9edf8c24461aef0b9f79a  
Author: shivam sahu <shivam14sahu@gmail.com>  
Date: Thu Mar 11 15:47:48 2021 +0530  
  
    stash example person1  
  
commit 9f764831c3dbadee7a4f2a34d8bc6e170a5d802f (feature)  
Merge: d8c81dc 005e2a2  
Author: shivam sahu <shivam14sahu@gmail.com>  
Date: Thu Mar 11 14:37:25 2021 +0530  
  
    merging files  
  
commit 005e2a27fc40f591c64af447245aa045a11318a2 (person2)  
Author: shivam sahu <shivam14sahu@gmail.com>  
Date: Thu Mar 11 14:26:37 2021 +0530  
  
    text added by person2  
  
commit d8c81dc1a333c81f063f3ebec561d001010b9caa  
Author: shivam sahu <shivam14sahu@gmail.com>  
Date: Thu Mar 11 14:25:20 2021 +0530  
  
    text added by person1  
  
commit f6658610d99ba4589d6e219fb5bd1d0bcf7ab8c9  
Merge: c9bae3a bfa18cc  
Author: shivam sahu <33172300+shivam-sahu@users.noreply.github.com>  
:
```

## Git status

```
> git status  
On branch master  
Your branch is up to date with 'origin/master'.  
  
nothing to commit, working tree clean
```

## Git reflog



```
b4d778b (HEAD -> master, origin/master, person1) HEAD@{0}: checkout: moving from person1 to master
b4d778b (HEAD -> master, origin/master, person1) HEAD@{1}: merge master: Fast-forward
3fdaa71 HEAD@{2}: checkout: moving from master to person1
b4d778b (HEAD -> master, origin/master, person1) HEAD@{3}: reset: moving to b4d778ba076f44206bd8d0dde8ae05fcbc178ebe
b4d778b (HEAD -> master, origin/master, person1) HEAD@{4}: commit: changes made
3fdaa71 HEAD@{5}: merge person1: Fast-forward
9f76483 (feature) HEAD@{6}: reset: moving to HEAD
9f76483 (feature) HEAD@{7}: checkout: moving from person1 to master
3fdaa71 HEAD@{8}: commit: stash example person1
9f76483 (feature) HEAD@{9}: merge master: Fast-forward
d8c81dc HEAD@{10}: reset: moving to d8c81dc1a333c81f063f3ebec561d001010b9caa
d8c81dc HEAD@{11}: checkout: moving from feature to person1
9f76483 (feature) HEAD@{12}: merge master: Fast-forward
f665861 HEAD@{13}: reset: moving to f665861
9f76483 (feature) HEAD@{14}: merge master: Fast-forward
f665861 HEAD@{15}: checkout: moving from master to feature
9f76483 (feature) HEAD@{16}: checkout: moving from feature to master
f665861 HEAD@{17}: checkout: moving from master to feature
9f76483 (feature) HEAD@{18}: reset: moving to 9f764831c3dbadee7a4f2a34d8bc6e170a5d802f
9f76483 (feature) HEAD@{19}: checkout: moving from feature to master
f665861 HEAD@{20}: reset: moving to f665861
9f76483 (feature) HEAD@{21}: checkout: moving from master to feature
9f76483 (feature) HEAD@{22}: reset: moving to 9f764831c3dbadee7a4f2a34d8bc6e170a5d802f
9f76483 (feature) HEAD@{23}: checkout: moving from feature to master
9f76483 (feature) HEAD@{24}: checkout: moving from master to feature
9f76483 (feature) HEAD@{25}: reset: moving to 9f764831c3dbadee7a4f2a34d8bc6e170a5d802f
9f76483 (feature) HEAD@{26}: reset: moving to 9f76483
:
```