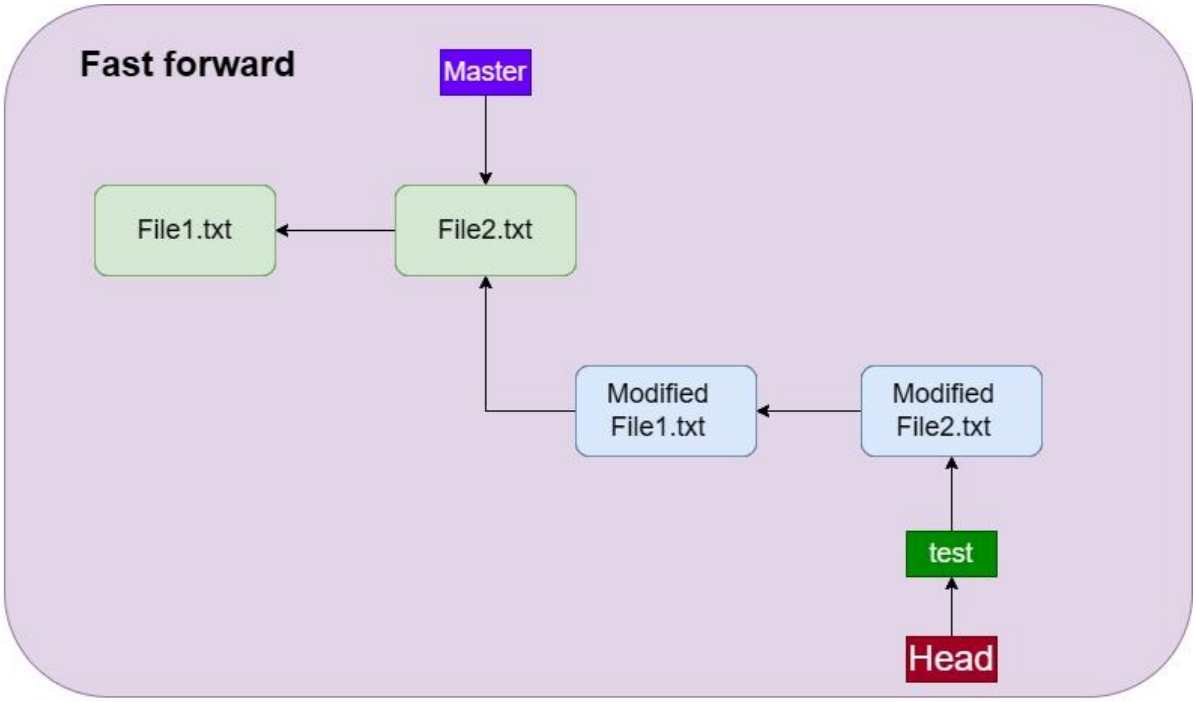
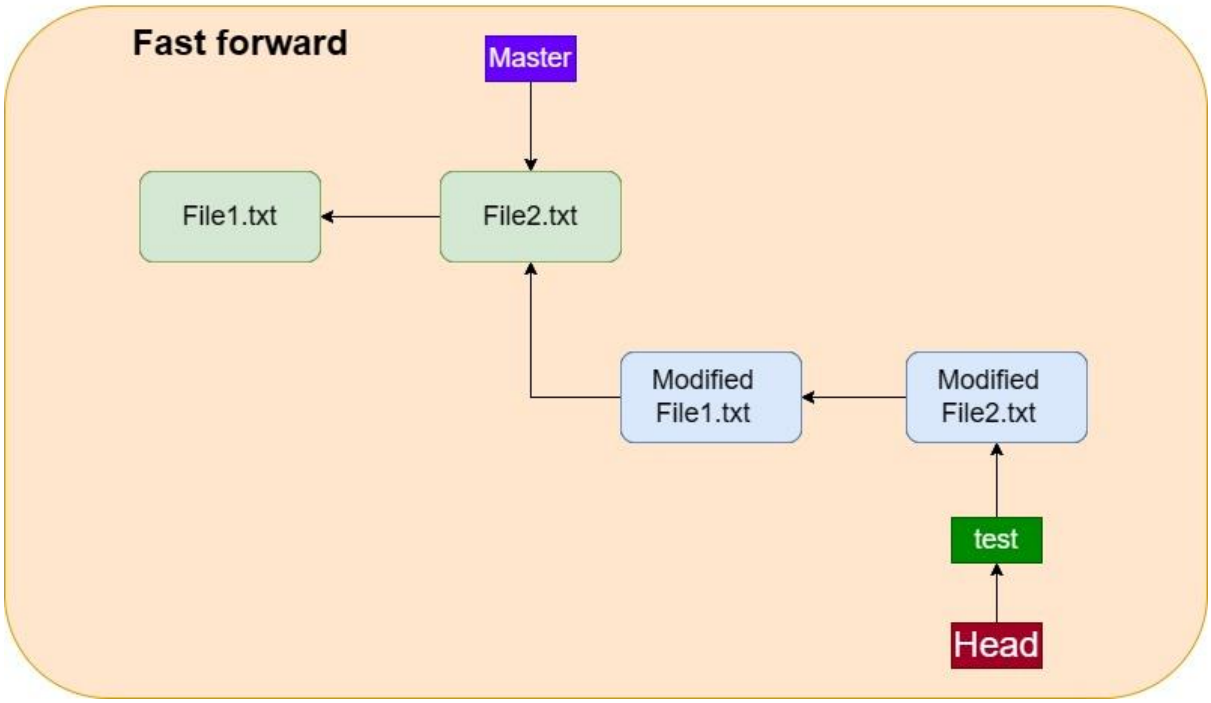
Assignment-3
Fast Forward Merge

a) Design Schema





b) Explanation

Fast forward merge can be performed when there is a direct linear path from the source branch to the target branch. In fast-forward merge, git simply moves the source branch pointer to the target branch pointer without creating an extra merge commit. Let us look at an example implementing fast-forward merge.

We have a master branch with 2 commits. Next, we create a branch called tes branch. In git a branch is nothing but a pointer to a commit. At this point both test and master are pointing to the same commit.

Now let us switch to the test branch and do a couple of commits. Now we need to bring the changes to the master branch. There is a linear path from test to master.

In order to merge the changes to the master branch, all git has to do is to change the pointer of master forward. This is what we call fast-forward merge.

Points to Consider before Fast Forward Merge

1) Extra branch should be created from where we will try to merge the code to receiving branch.
2) Immidiate parent of the first commit one new branch should be the last parent of the receiving branch.

Solution :-

Master branch pointer will be changed from the last commit of the receiving branch to last commit of new branch.

c) User Case

1) Make repo having name fastforward and initialize it.

```
bhush@Bhushan MINGW64 /c/DevOps-Practice
$ mkdir fastforward

bhush@Bhushan MINGW64 /c/DevOps-Practice
$ cd fastforward/

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward
$ git init
Initialized empty Git repository in C:/DevOps-Practice/fastforward/.git/

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ ls -la
total 4
drwxr-xr-x 1 bhush 197609 0 Mar 15 01:02 ./
drwxr-xr-x 1 bhush 197609 0 Mar 15 01:02 ../
drwxr-xr-x 1 bhush 197609 0 Mar 15 01:02 .git/
```

2) Now create file File1.txt and File2.txt and check the status of it. It is untracked files. So add it in staging area and commit.

```
bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ vi File1.txt

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        File1.txt

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

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git add File1.txt
warning: in the working copy of 'File1.txt', LF will be replaced by CRLF the next time Git touc
hes it

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git commit -m "File1.txt is committed!"
[master (root-commit) 40b1e9f] File1.txt is committed!
 1 file changed, 1 insertion(+)
 create mode 100644 File1.txt
```

Like wise add File2.txt and check the status of it. It is untracked files. So add it in staging area and commit.

3) Check the graph to show the all the commits.

```
bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git log --graph
* commit efbc2ba4d5c9e9c84f9984b62bcdc1322550c72a (HEAD -> master)
| Author: rutujatopre <rutujamm27@gmail.com>
| Date:   Wed Mar 15 01:11:51 2023 +0530
|
|     File2.txt is committed!
|
* commit 40b1e9fa018c9f2128c150c41a91de0fb03abefb
  Author: rutujatopre <rutujamm27@gmail.com>
  Date:   Wed Mar 15 01:07:57 2023 +0530

      File1.txt is committed!
```

4) Now create test branch. After that check branches. Switch to test branch * is in-front of test so it means that we are in test branch.

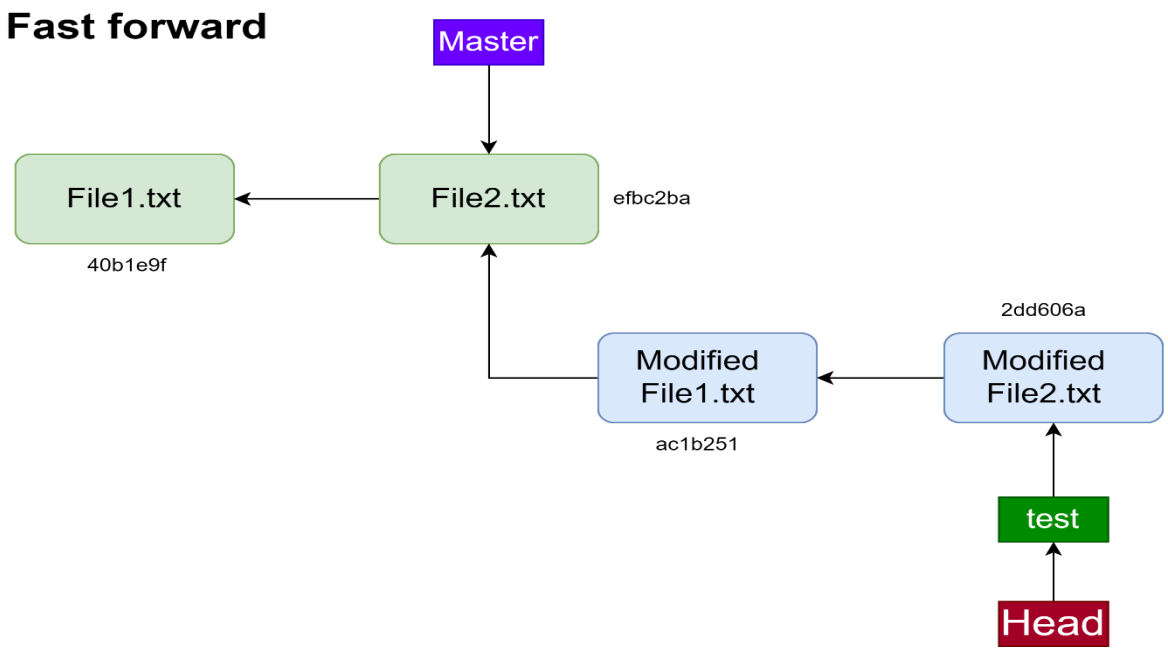
```
bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git branch
* master

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git branch test

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git branch
* master
  test

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git checkout test
Switched to branch 'test'

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ git branch
  master
* test
```

5) Now modify file File1.txt and File2.txt and check the status of it. It is modified files. So add it in staging area and commit.

```
bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ vi File1.txt

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ git status
On branch test
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:   File1.txt

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

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ git add File1.txt
warning: in the working copy of 'File1.txt', LF will be replaced by CRLF the next time Git touc
hes it

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ git commit -m "File1.txt is modifed and committed!"
[test ac1b251] File1.txt is modifed and committed!
 1 file changed, 1 insertion(+)
```

```
bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ vi File2.txt

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ git status
On branch test
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:   File2.txt

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

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ git add File2.txt
warning: in the working copy of 'File2.txt', LF will be replaced by CRLF the next time Git touc
hes it

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ git commit -m "File2.txt is modified and committed!"
[test 2dd606a] File2.txt is modified and committed!
 1 file changed, 1 insertion(+)
```

6) Now check where test and master branch pointing towards which commit.



```
bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ git log --graph --pretty=oneline
* 2dd606a04b1b9378eb4a043d8e3946b7adb55790 (HEAD -> test) File2.txt is modified and committed!
* ac1b251e989cbc7aeef088e0d962e69241d2fb5d File1.txt is modifed and committed!
* efbc2ba4d5c9e9c84f9984b62bcdc1322550c72a (master) File2.txt is committed!
* 40b1e9fa018c9f2128c150c41a91de0fb03abefb File1.txt is committed!
```

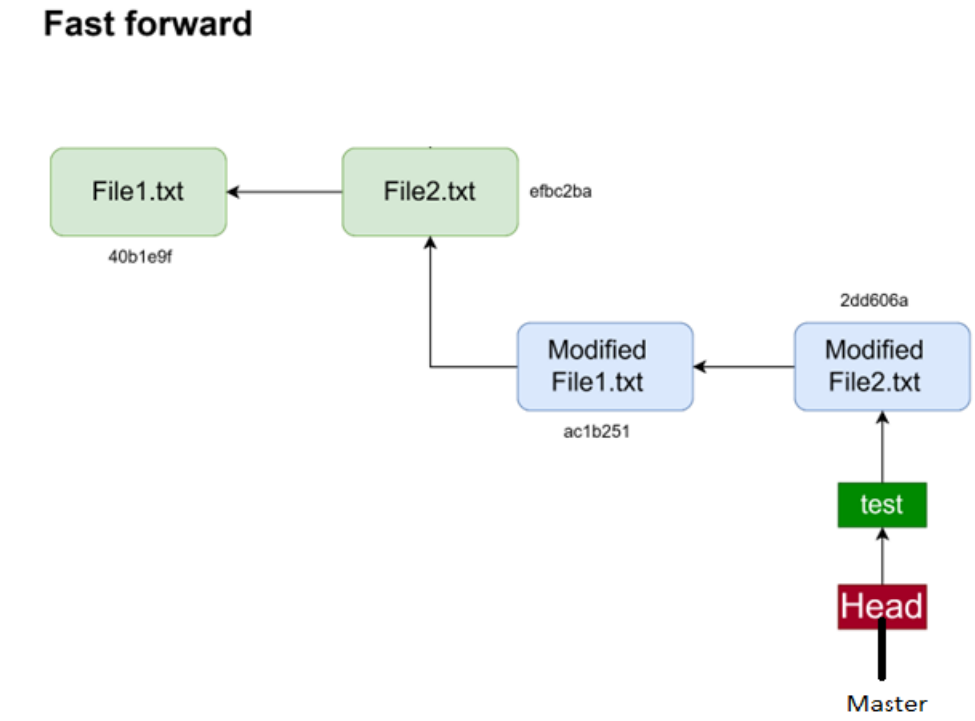
7) Now switch to master branch. Check branches * is in-front of master it means that we are on master branch.

```
bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (test)
$ git checkout master
Switched to branch 'master'

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git branch
* master
  test
```

8) If we want to change the test to master then merge it. It will show it's a fast forward merge and it is updating from efbc2ba..2dd606a this commit.

```
bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git merge test
Updating efbc2ba..2dd606a
Fast-forward
 File1.txt | 1 +
 File2.txt | 1 +
 2 files changed, 2 insertions(+)
```

9) Now after merging master is pointing towards the last commit of new branch as shown in following diagram.



```
bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward (master)
$ git log --graph --pretty=oneline
* 2dd606a04b1b9378eb4a043d8e3946b7adb55790 (HEAD -> master, test) File2.txt is modified and com
mitted!
* ac1b251e989cbc7aeef088e0d962e69241d2fb5d File1.txt is modifed and committed!
* efbc2ba4d5c9e9c84f9984b62bcdc1322550c72a File2.txt is committed!
* 40b1e9fa018c9f2128c150c41a91de0fb03abefb File1.txt is committed!
```

```
bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward/.git (GIT_DIR!)
$ cat HEAD
ref: refs/heads/master

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward/.git (GIT_DIR!)
$ cat /refs/heads/master
cat: /refs/heads/master: No such file or directory

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward/.git (GIT_DIR!)
$ cd /refs/heads/master
bash: cd: /refs/heads/master: No such file or directory

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward/.git (GIT_DIR!)
$ cd /refs/heads
bash: cd: /refs/heads: No such file or directory

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward/.git (GIT_DIR!)
$ cd refs/heads

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward/.git/refs/heads (GIT_DIR!)
$ cat master
2dd606a04b1b9378eb4a043d8e3946b7adb55790

bhush@Bhushan MINGW64 /c/DevOps-Practice/fastforward/.git/refs/heads (GIT_DIR!)
$ cat test
2dd606a04b1b9378eb4a043d8e3946b7adb55790
```