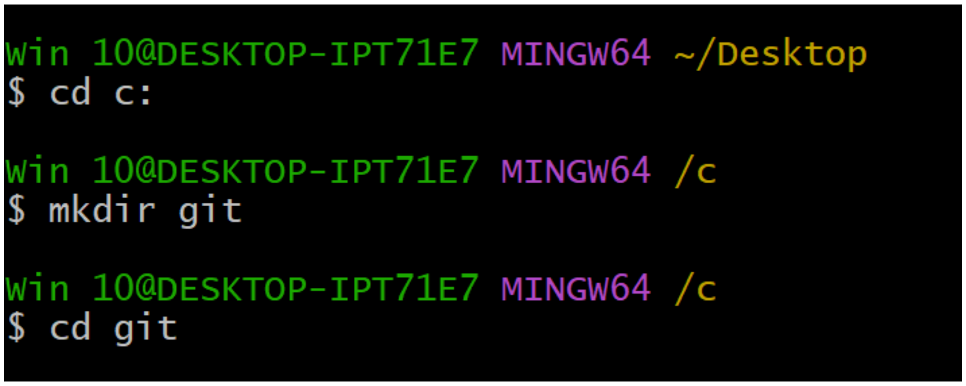
PROJECT MANAGEMENT WITH GIT

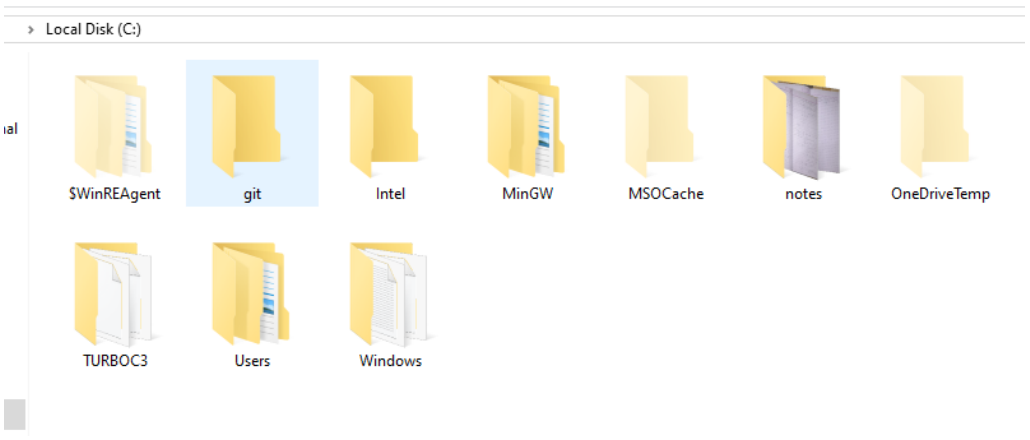
1)Setting Up and Basic Commands

- Initialize a new Git repository in a directory. Create a new file and add it to the staging area and commit the changes with an appropriate commit message.

Step1:

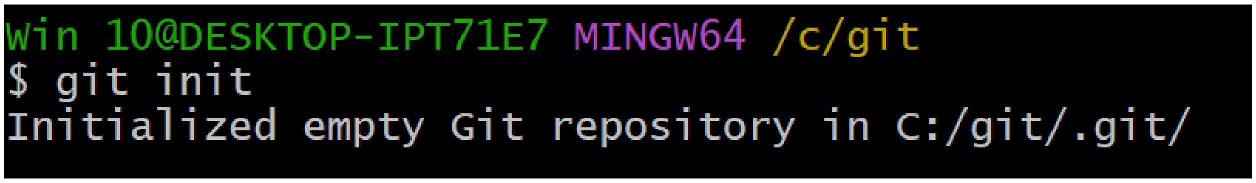


```
Win 10@DESKTOP-IPT71E7 MINGW64 ~/Desktop
$ cd c:

Win 10@DESKTOP-IPT71E7 MINGW64 /c
$ mkdir git

Win 10@DESKTOP-IPT71E7 MINGW64 /c
$ cd git
```

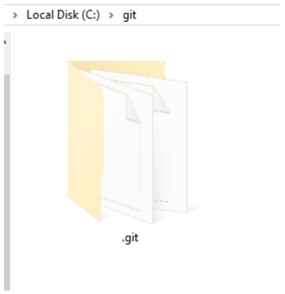


- cd c: → changes the current working directory into c disk
- mkdir git → creates a folder/directory in the present working directory.
- cd git → changes the directory to the git floder which was created.

Step2:

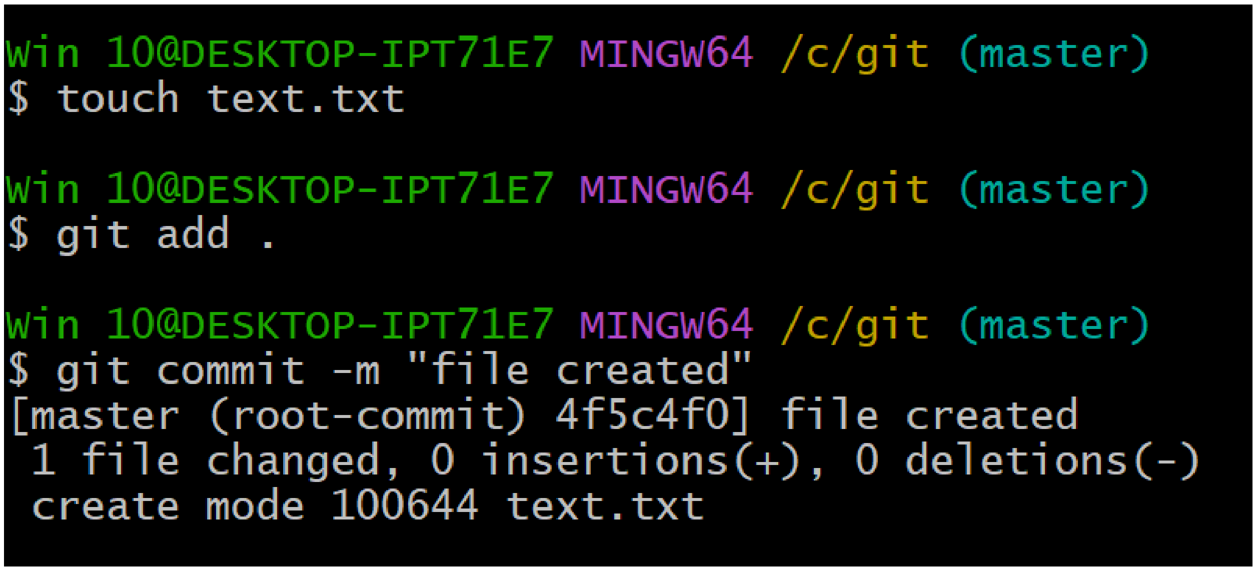


```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git
$ git init
Initialized empty Git repository in C:/git/.git/
```



- git init → initializes a empty git repository.
- We can see the .git folder created in the git folder ,in some cases the file is hidden and to see that hidden file we need to click on view the hidden files.

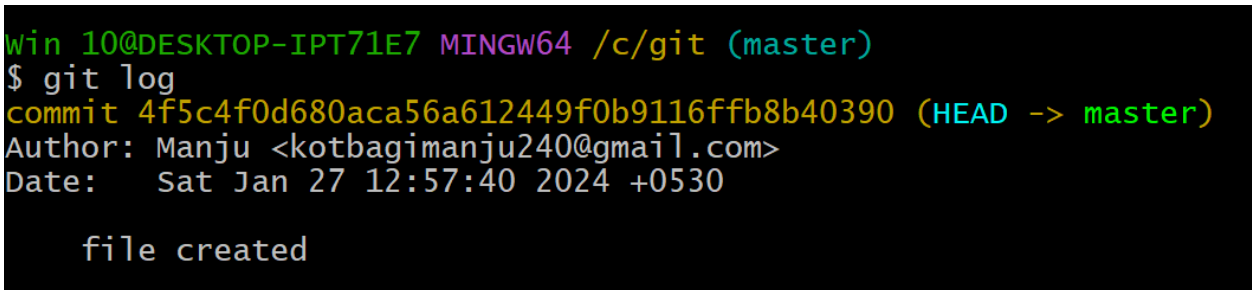
Step3:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ touch text.txt

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git add .

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git commit -m "file created"
[master (root-commit) 4f5c4f0] file created
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 text.txt
```

- touch text.txt →creates a empty file of txt extension in the current directory.
- git add . /git add text.txt → stages the file in the case of specific file add ,or add . will stage the whole files in the current directory and are ready to be commited.

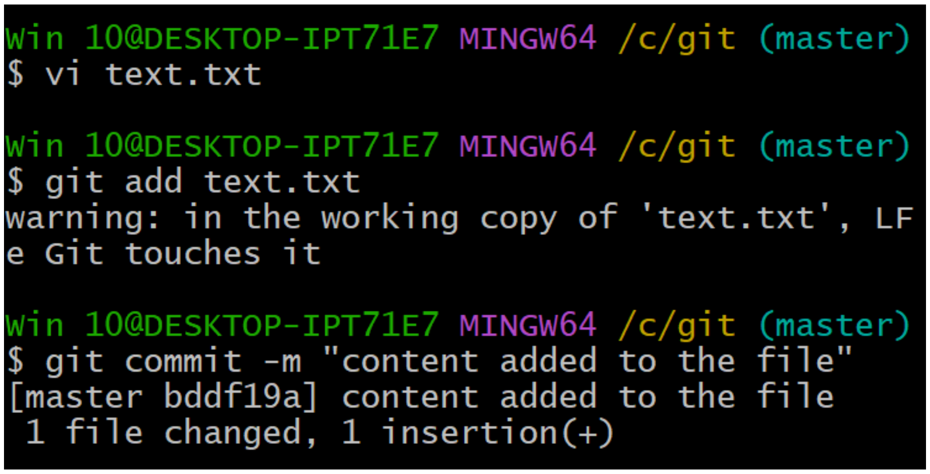
Step4:



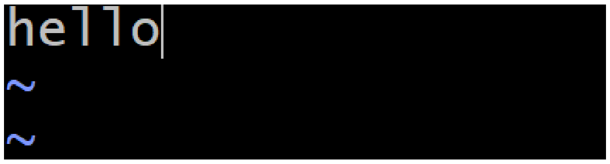
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git log
commit 4f5c4f0d680aca56a612449f0b9116ffb8b40390 (HEAD -> master)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 12:57:40 2024 +0530

    file created
```

- git log → displays all the history of commits with commit messages along with the author name and email.
- Every commits has a unique commit ID.

Step5:



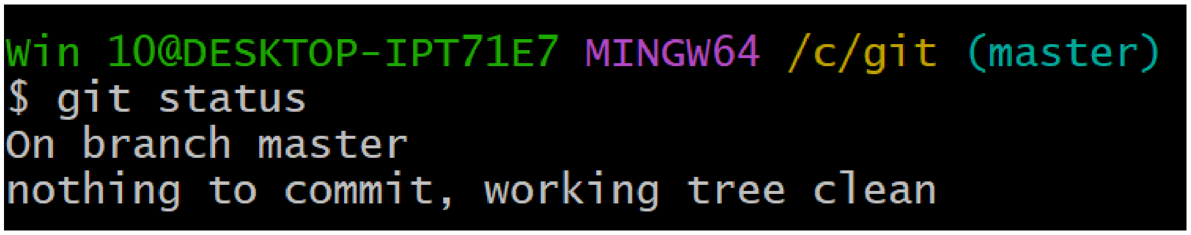
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ vi text.txt

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git add text.txt
warning: in the working copy of 'text.txt', LF
e Git touches it

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git commit -m "content added to the file"
[master bddf19a] content added to the file
 1 file changed, 1 insertion(+)
```



```
hello
~
~
```

- Here the file content is added using vi text.txt and then file is staged and commited with appropriate message.

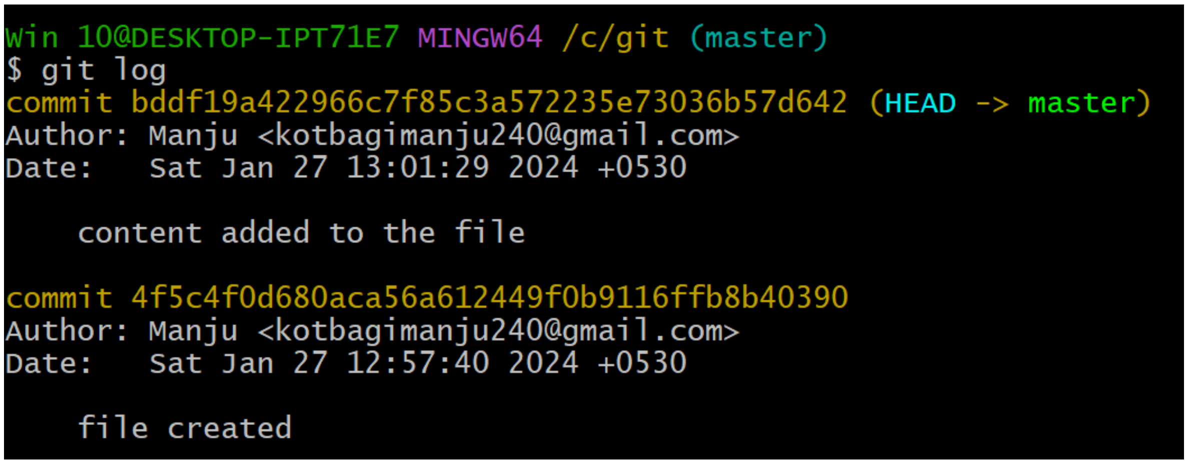
Step6:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git status
On branch master
nothing to commit, working tree clean
```

- git status → it checks the status ,like on which branch we are and is there any changes made which are not commited .
- if no any other changes has been made after recent commit the it displays the working tree is clean.

Step7:



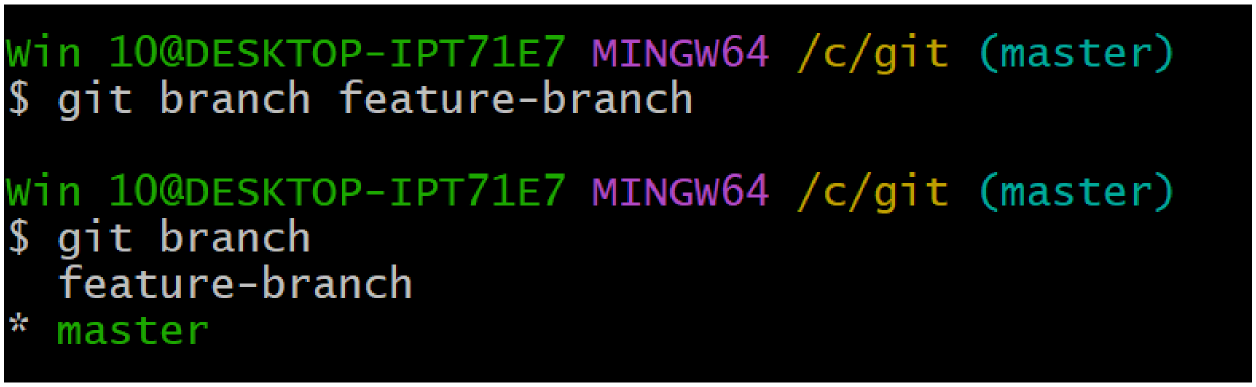
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git log
commit bddf19a422966c7f85c3a572235e73036b57d642 (HEAD -> master)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:01:29 2024 +0530

    content added to the file

commit 4f5c4f0d680aca56a612449f0b9116ffb8b40390
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 12:57:40 2024 +0530

    file created
```

- git log → this will display the history of the commits.

2)Creating and Managing Branches:

➢ Create a new branch named "feature-branch." Switch to the "master" branch. Merge the "feature-branch" into "master.

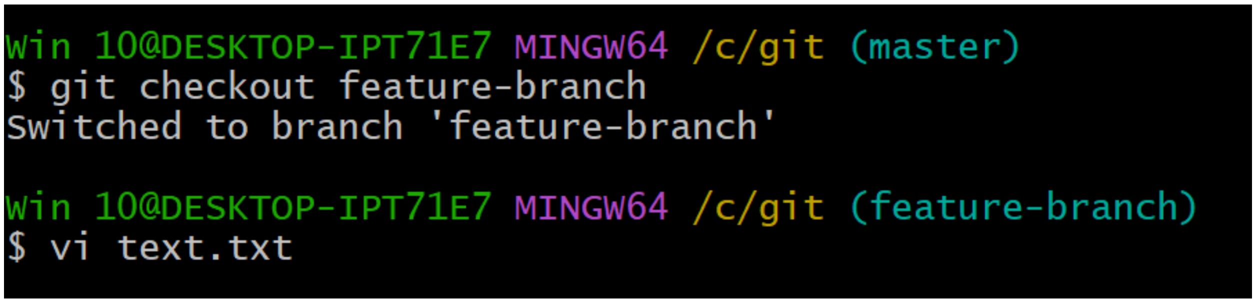
Step1:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git branch feature-branch

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git branch
  feature-branch
* master
```

- git branch feature-branch → this will create a new branch of the main master branch in which the contents and files are copied from the master branch .
- git branch → this command will show all the branches we have made and the current branch will be in green colour.

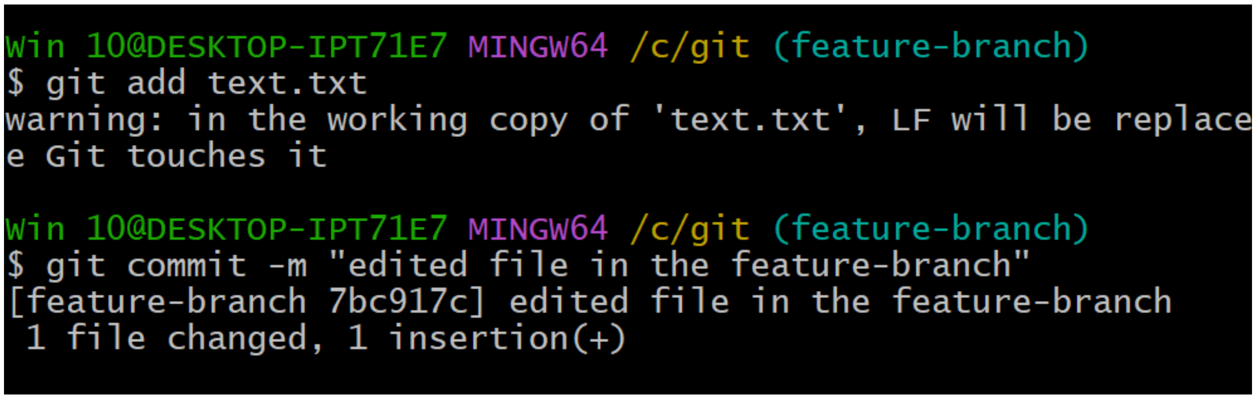
Step2:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git checkout feature-branch
Switched to branch 'feature-branch'

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (feature-branch)
$ vi text.txt
```

- git checkout → it is used to switch one branch to other branch ,here we are moving from branch master to the feature-branch.
- And also we have edited the file text.txt.

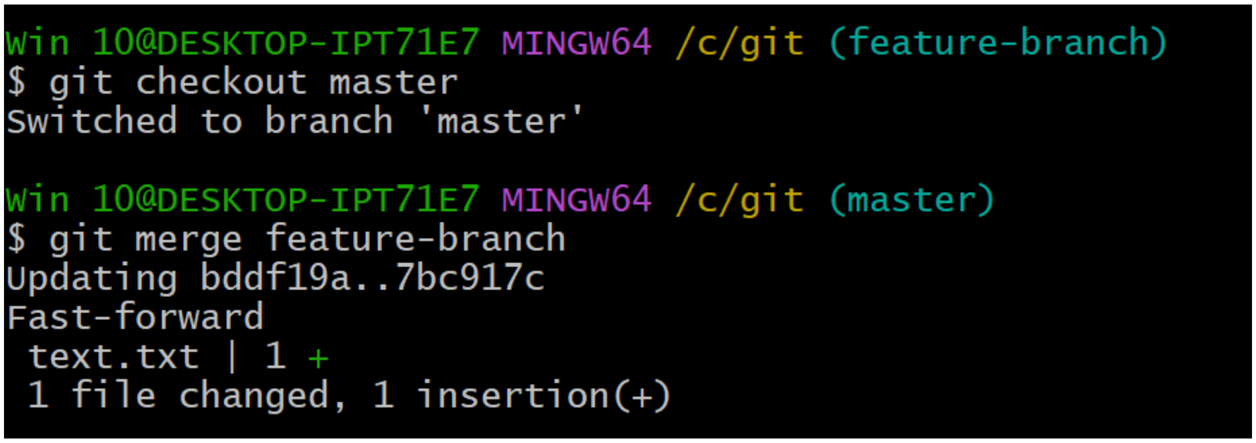
Step3:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (feature-branch)
$ git add text.txt
warning: in the working copy of 'text.txt', LF will be replace
e Git touches it

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (feature-branch)
$ git commit -m "edited file in the feature-branch"
[feature-branch 7bc917c] edited file in the feature-branch
 1 file changed, 1 insertion(+)
```

- Here in the feature-branch we staged the file and commited with message saying “edited file in the feature branch”.
- So here the file has been changed ,but in the master branch it will be as it is until we merge the feature-branch with the master branch.

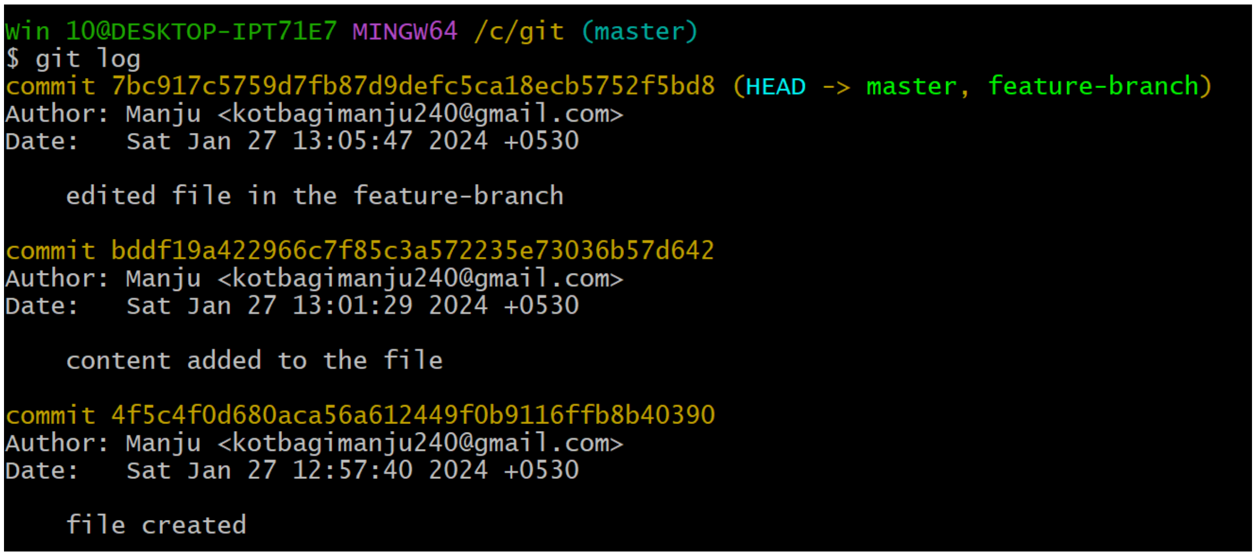
Step4:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (feature-branch)
$ git checkout master
Switched to branch 'master'

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git merge feature-branch
Updating bddf19a..7bc917c
Fast-forward
 text.txt | 1 +
 1 file changed, 1 insertion(+)
```

- For merging the file to the master branch we first need to move to the main/master branch using “git checkout master” command.
- Then we can merge the branch with “git merge feature-branch” command.
- So ,now the files will be meged ,the changes or the edits in the feature-branch will be merged.

Step5:



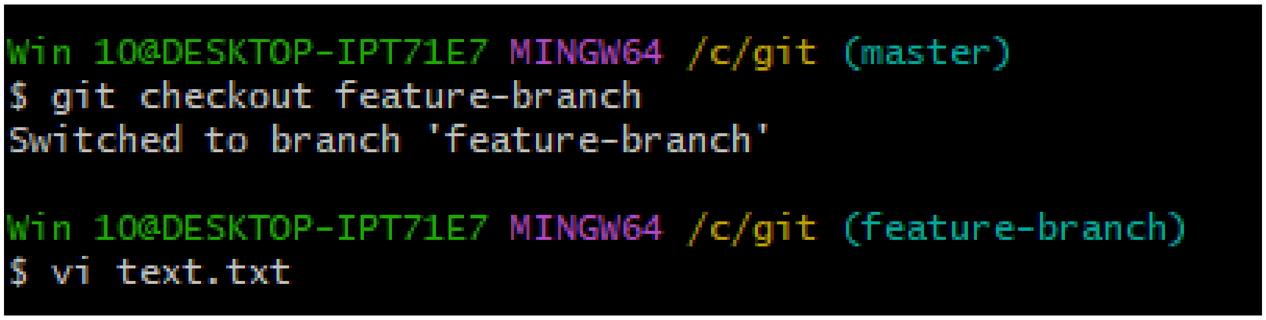
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git log
commit 7bc917c5759d7fb87d9defc5ca18ecb5752f5bd8 (HEAD -> master, feature-branch)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:05:47 2024 +0530

    edited file in the feature-branch

commit bddf19a422966c7f85c3a572235e73036b57d642
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:01:29 2024 +0530

    content added to the file

commit 4f5c4f0d680aca56a612449f0b9116ffb8b40390
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 12:57:40 2024 +0530

    file created
```

- git log → here the history of the commits will be visible.

3)Creating and Managing Branches

➢ Write the commands to stash your changes, switch branches, and then apply the stashed changes.

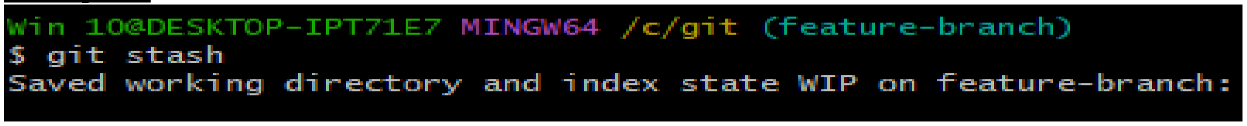
Step1:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git checkout feature-branch
Switched to branch 'feature-branch'

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (feature-branch)
$ vi text.txt
```

- Moving to the feature branch and have made changes in the text.txt file using the commands git checkout feature-branch and vi text.txt for changing the branch and editing the file respectively.
- Here we have not staged and commited the changes in the text.txt file.

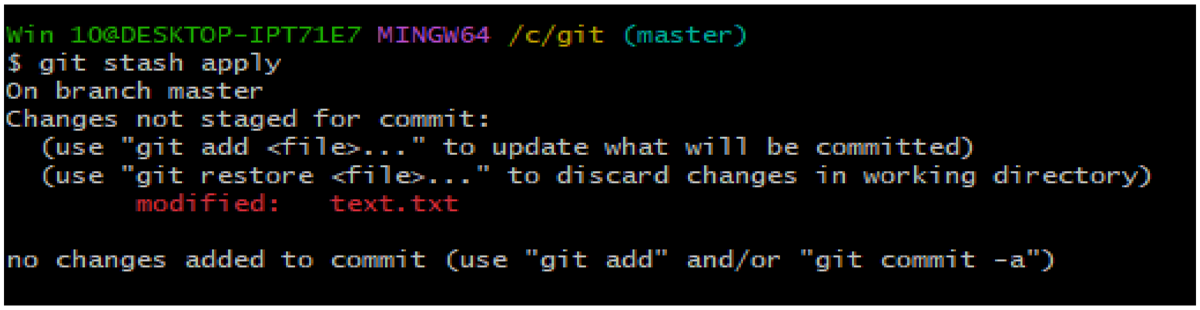
Step2:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (feature-branch)
$ git stash
Saved working directory and index state WIP on feature-branch:
```

- In this step we have stashed the changes which have been made in the text.txt file in the feature-branch.
- Here we have not added/staged the file and commited the changes.
- The changes will be saved in the branch without the commiting the changes.
- The command used fir stashing the changes →git stash

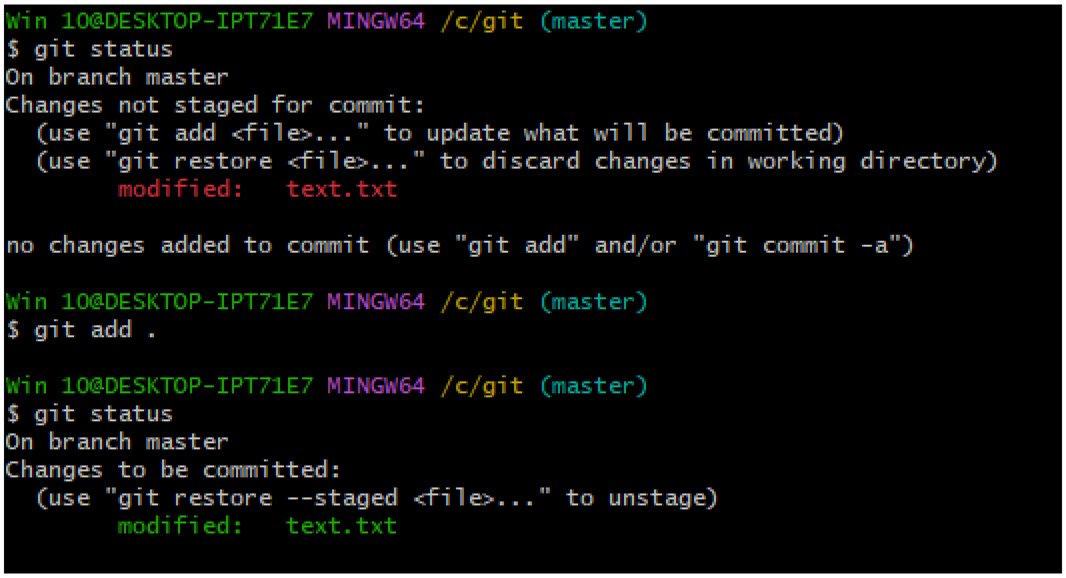
Step3:



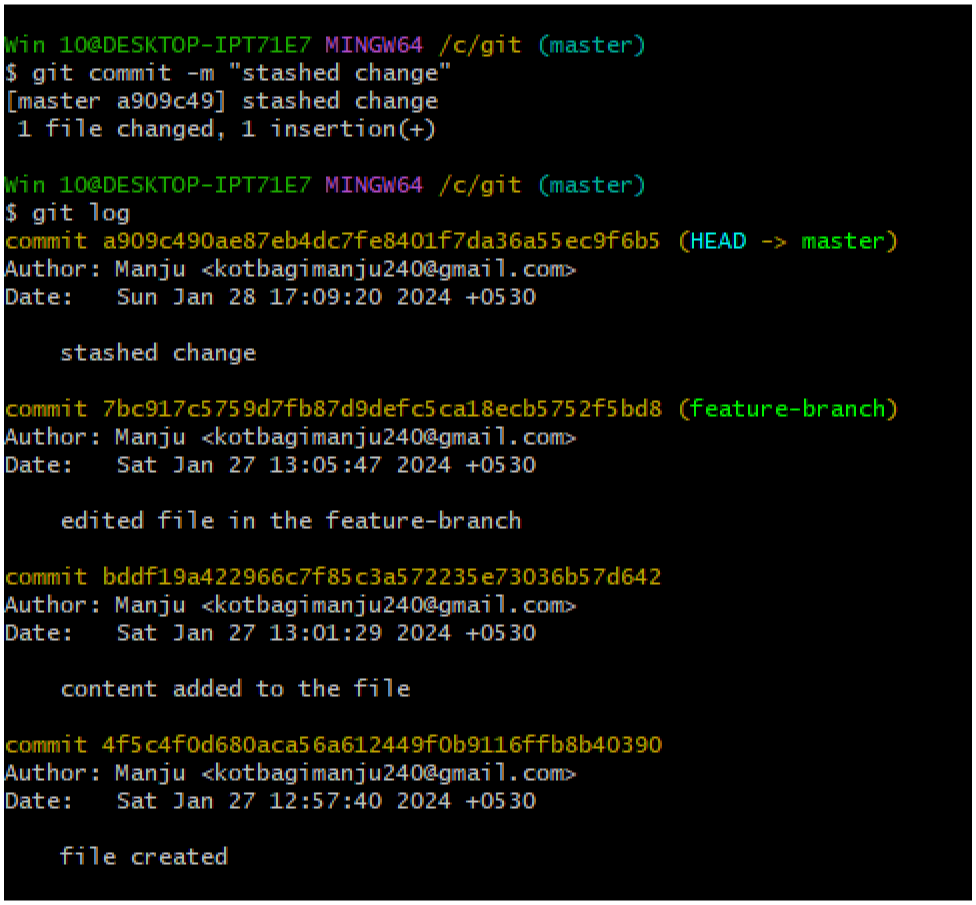
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git stash apply
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   text.txt

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

- Before applying the changes made in the feature-branch first we moved to the master branch.
- Then ,in the master branch we applied the stashed changes made in the feature-branch.
- After applying the stash to the master. It will give us a message saying that the applied stash is not staged and commited in the master branch.

Step4:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git status
On branch master
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:   text.txt

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

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git add .

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   text.txt
```



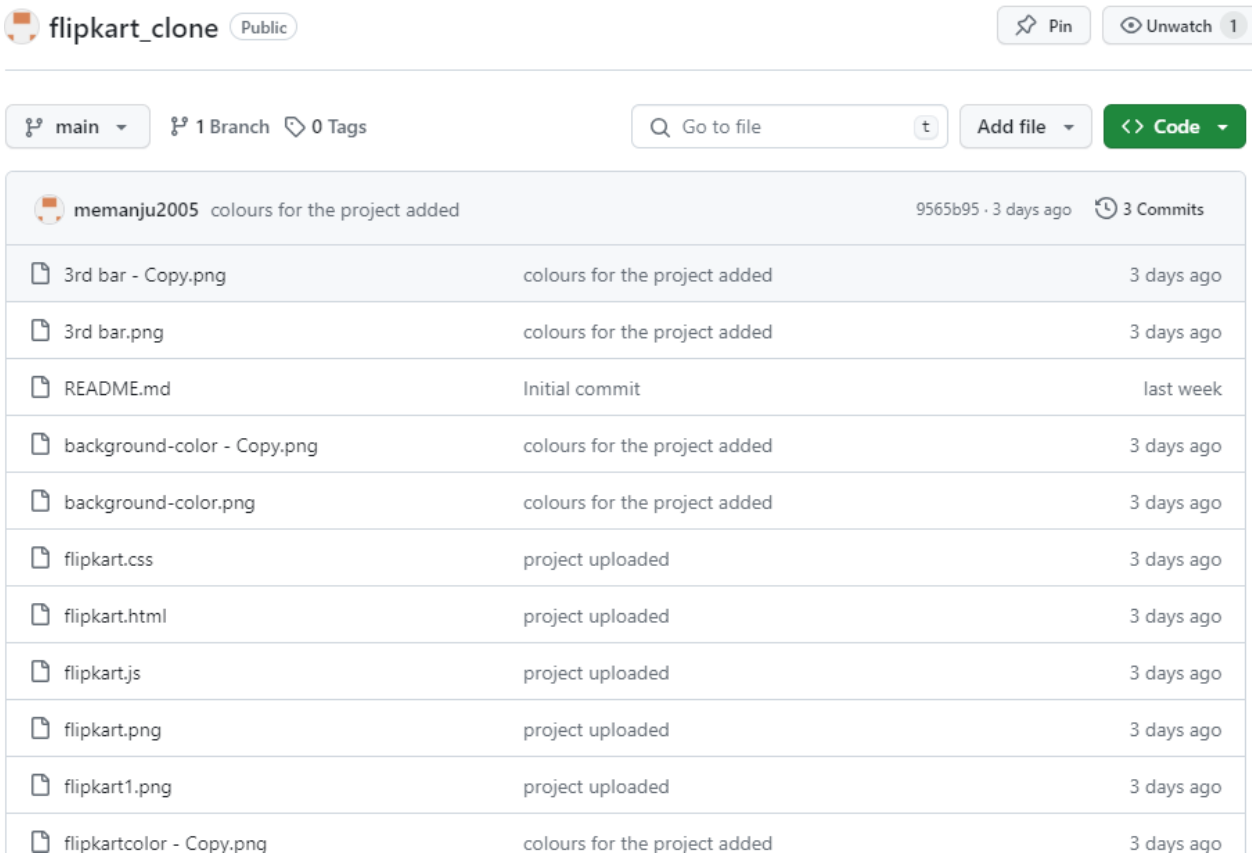
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git commit -m "stashed change"
[master a909c49] stashed change
 1 file changed, 1 insertion(+)

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git log
commit a909c490ae87eb4dc7fe8401f7da36a55ec9f6b5 (HEAD -> master)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sun Jan 28 17:09:20 2024 +0530

    stashed change

commit 7bc917c5759d7fb87d9defc5ca18ecb5752f5bd8 (feature-branch)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:05:47 2024 +0530

    edited file in the feature-branch

commit bddf19a422966c7f85c3a572235e73036b57d642
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:01:29 2024 +0530

    content added to the file

commit 4f5c4f0d680aca56a612449f0b9116ffb8b40390
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 12:57:40 2024 +0530

    file created
```

- After stashing we can see the staus ,here it showed that the file is modified but not yet committed.
- If we watch git status after adding on the stage but not commiting , then it will "say changes to be commited."
- After commiting the changes we can see the log of the repository.

4)Collaboration and Remote Repositories

➢ Clone a remote Git repository to your local machine.

Step1:

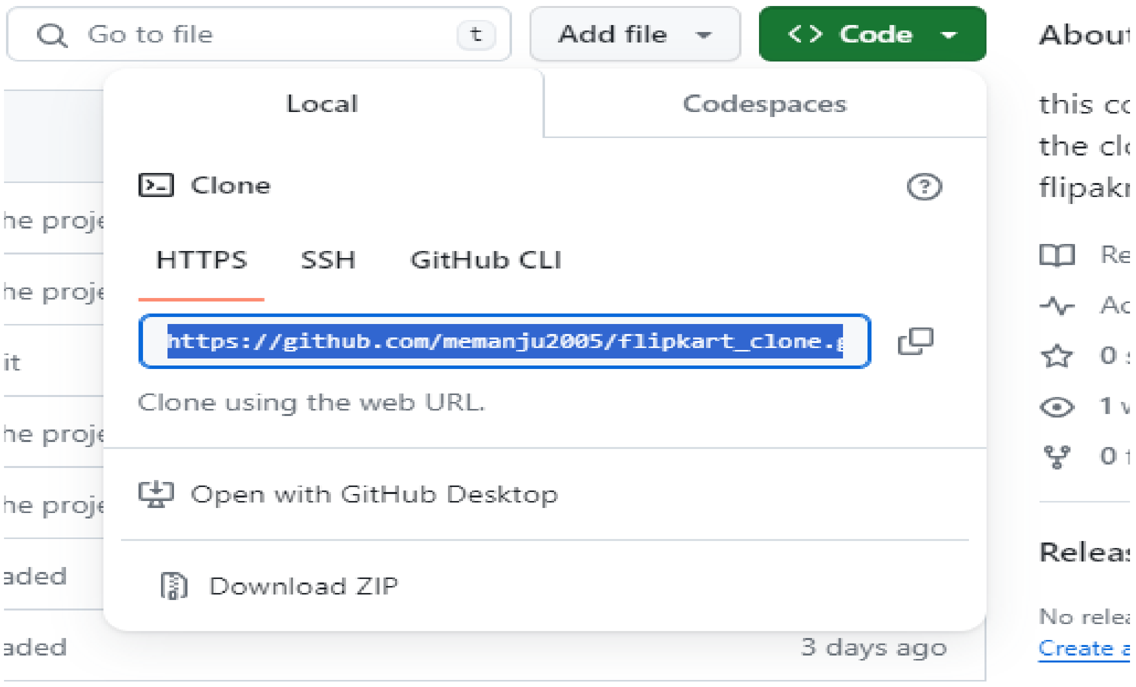

flipkart_clone Public

Pin | Unwatch 1

main | 1 Branch | 0 Tags | Go to file | Add file | Code

memanju2005 colours for the project added — 9565b95 · 3 days ago — 3 Commits

3rd bar - Copy.png	colours for the project added	3 days ago
3rd bar.png	colours for the project added	3 days ago
README.md	Initial commit	last week
background-color - Copy.png	colours for the project added	3 days ago
background-color.png	colours for the project added	3 days ago
flipkart.css	project uploaded	3 days ago
flipkart.html	project uploaded	3 days ago
flipkart.js	project uploaded	3 days ago
flipkart.png	project uploaded	3 days ago
flipkart1.png	project uploaded	3 days ago
flipkartcolor - Copy.png	colours for the project added	3 days ago

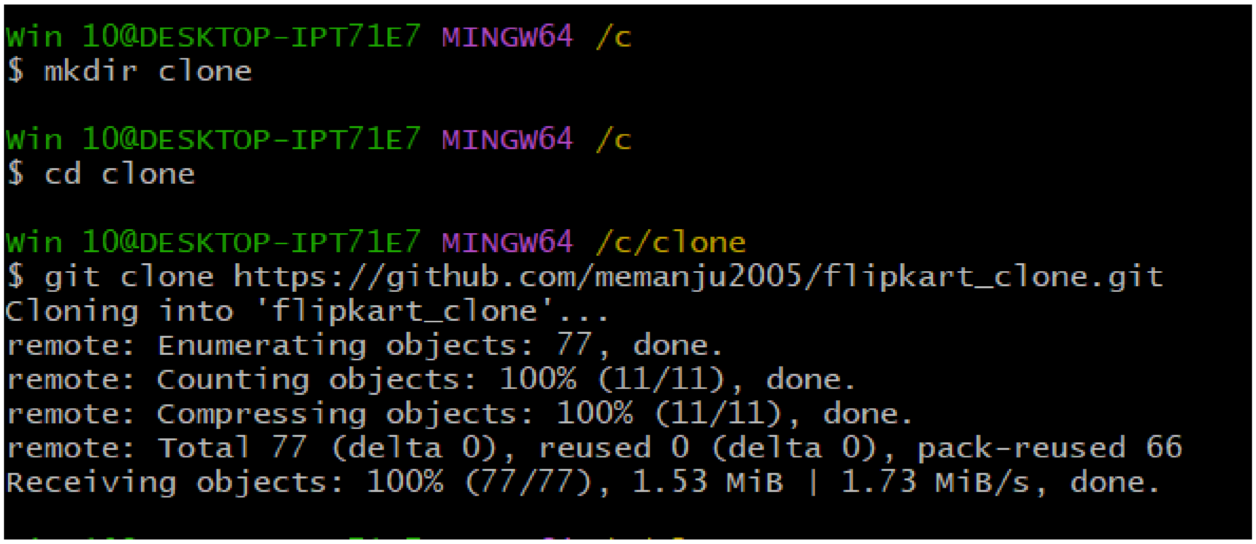


```
https://github.com/memanju2005/flipkart_clone.g
```

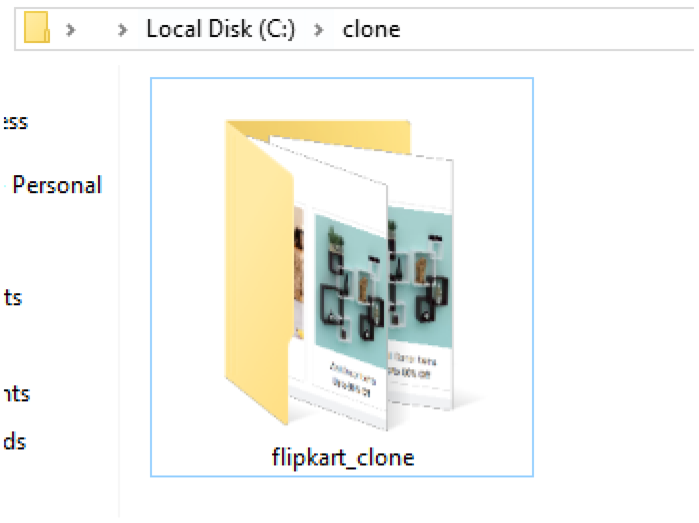
Clone using the web URL.

- To clone a remote git repository ,first we need to open the github.com and open any of the account on the github.com
- After that we have chosen the repository which we want to clone into our local machine.
- After choosing the repository , in that repository we clicked on the green button "code" ,which open a dropdown list of links in that ,we copied the "HTTPS" link from that. →https://github.com/memanju2005/flipkart_clone.git

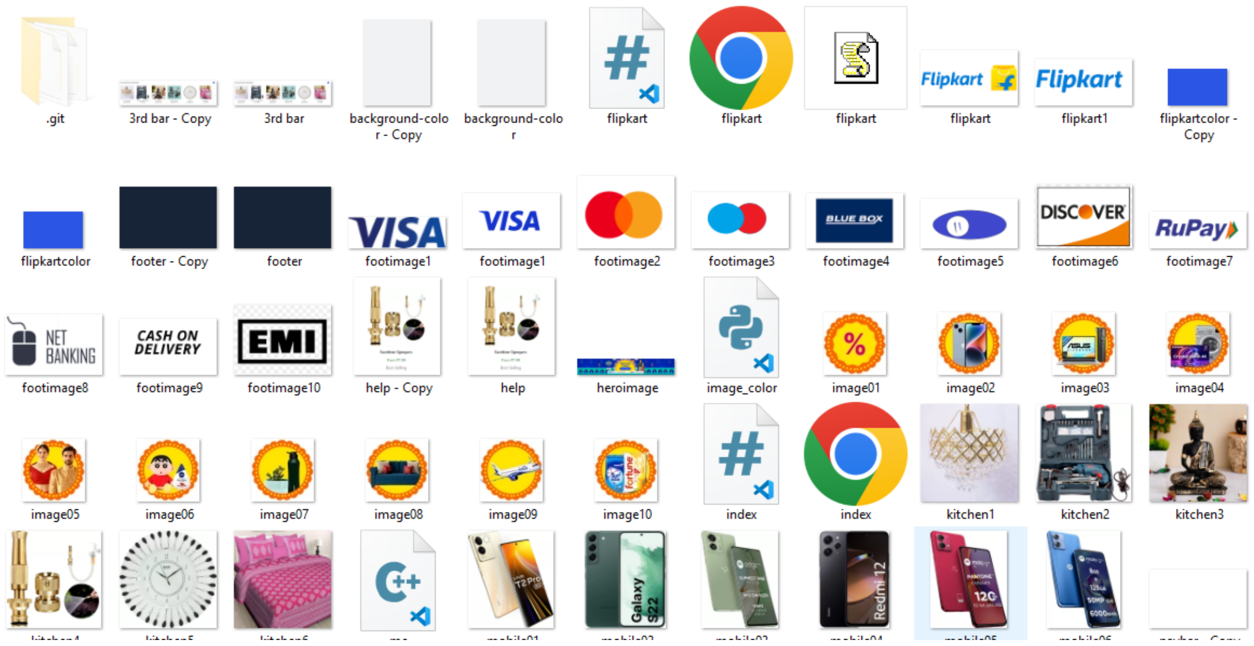
Step2:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c
$ mkdir clone

Win 10@DESKTOP-IPT71E7 MINGW64 /c
$ cd clone

Win 10@DESKTOP-IPT71E7 MINGW64 /c/clone
$ git clone https://github.com/memanju2005/flipkart_clone.git
Cloning into 'flipkart_clone'...
remote: Enumerating objects: 77, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 77 (delta 0), reused 0 (delta 0), pack-reused 66
Receiving objects: 100% (77/77), 1.53 MiB | 1.73 MiB/s, done.
```



- In the second step we need to open our git bash and in some directory where you need to clone the remote repository we need to move to that location using "cd <path> " command.
- Here we want to copy the repository to the folder clone in the c so we moved to that location
- git clone https://github.com/memanju2005/flipkart_clone.git → this command will copy the repository from remote to the local machine in the working directory
- you can see above the "flipkart_clone" repository is succesfully copied in the clone directory/folder.

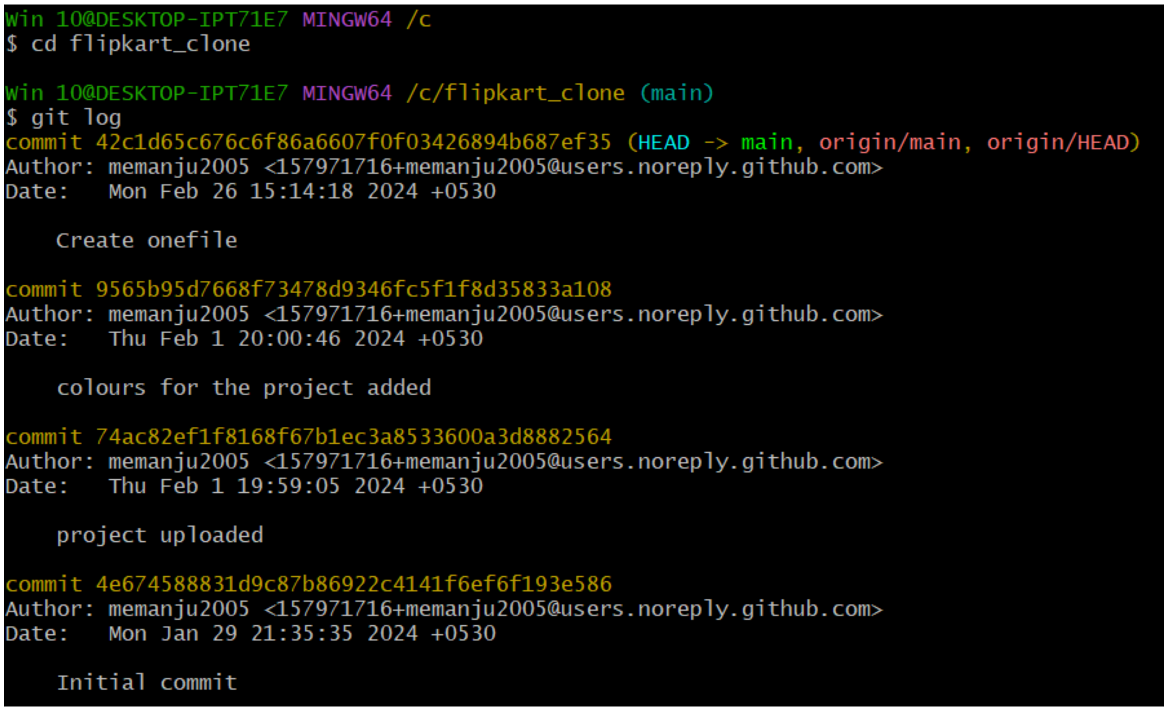


- We know that whole repository have been cloned to the local machine so the files and ".png" files can be seen in that repo.

5)Collaboration and Remote Repositories

➢ Fetch the latest changes from a remote repository and rebase your local branch onto the updated remote branch.

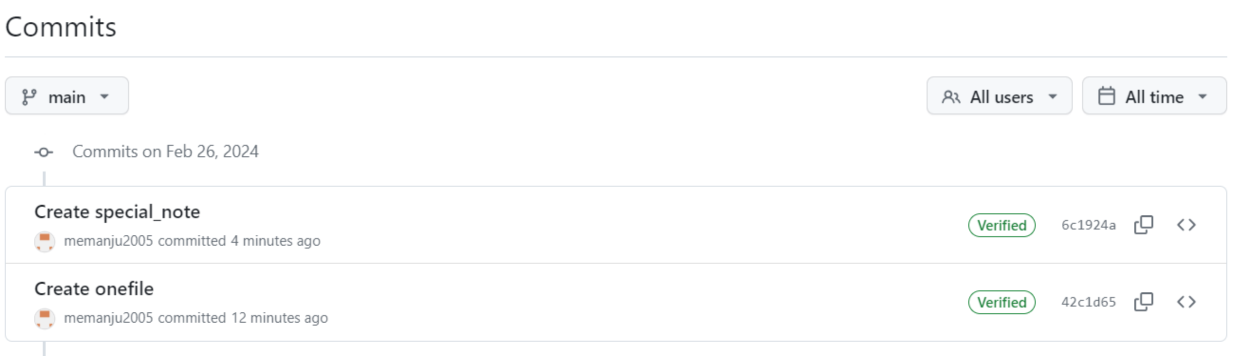
Step1)



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c
$ cd flipkart_clone

Win 10@DESKTOP-IPT71E7 MINGW64 /c/flipkart_clone (main)
$ git log
commit 42c1d65c676c6f86a6607f0f03426894b687ef35 (HEAD -> main, origin/main, origin/HEAD)
Author: memanju2005 <157971716+memanju2005@users.noreply.github.com>
Date:   Mon Feb 26 15:14:18 2024 +0530

    Create onefile

commit 9565b95d7668f73478d9346fc5f1f8d35833a108
Author: memanju2005 <157971716+memanju2005@users.noreply.github.com>
Date:   Thu Feb 1 20:00:46 2024 +0530

    colours for the project added

commit 74ac82ef1f8168f67b1ec3a8533600a3d8882564
Author: memanju2005 <157971716+memanju2005@users.noreply.github.com>
Date:   Thu Feb 1 19:59:05 2024 +0530

    project uploaded

commit 4e674588831d9c87b86922c4141f6ef6f193e586
Author: memanju2005 <157971716+memanju2005@users.noreply.github.com>
Date:   Mon Jan 29 21:35:35 2024 +0530

    Initial commit
```

- To fetch and rebase the remote repository to local repository ,we will move to the already cloned repo.
- Initially before fetching the changes from the remote repo the last commit was "created onefile" after logging the commits

Step2:


Commits

main · All users · All time

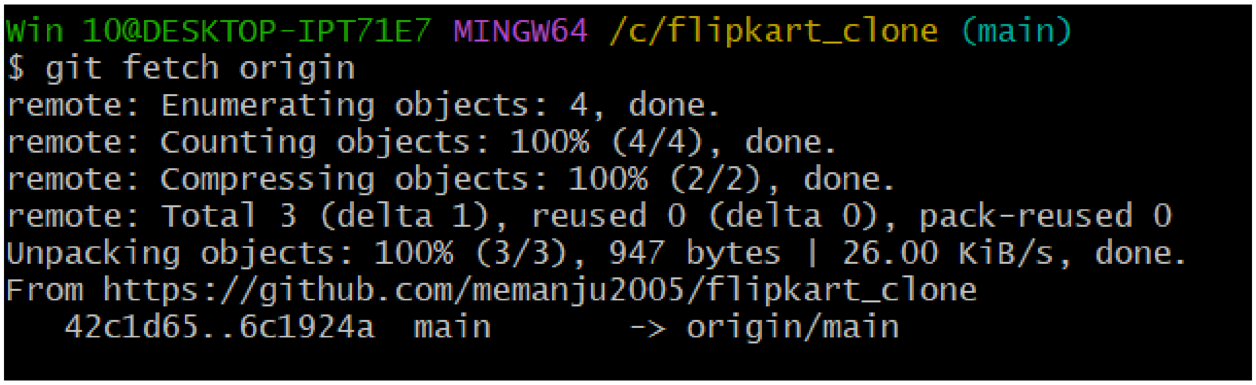
Commits on Feb 26, 2024

Create special_note
memanju2005 committed 4 minutes ago · Verified · 6c1924a

Create onefile
memanju2005 committed 12 minutes ago · Verified · 42c1d65

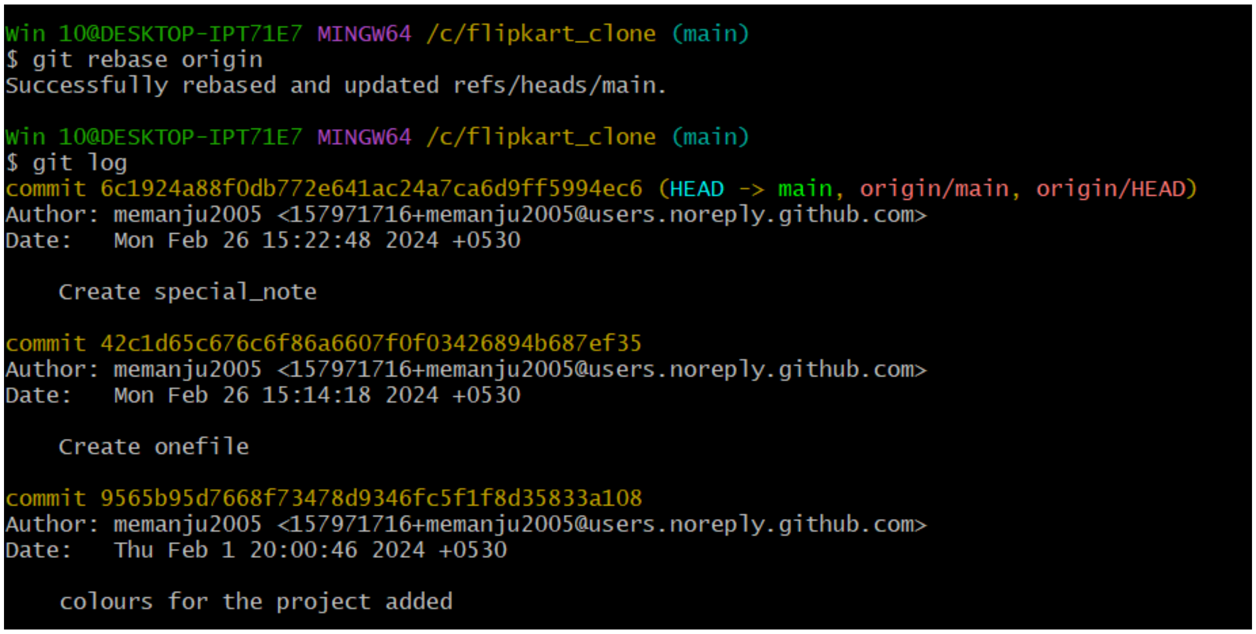
- Move to the remote repo and make some changes/add new file and commit it.

Step3:


```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/flipkart_clone (main)
$ git fetch origin
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), 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), 947 bytes | 26.00 KiB/s, done.
From https://github.com/memanju2005/flipkart_clone
   42c1d65..6c1924a  main       -> origin/main
```

- git fetch origin → this will fetch the latest changes from the remote repo that is the file named “special_note” which was created and committed.
- These changes after fetching will not be be available in the working directory.

Step4:



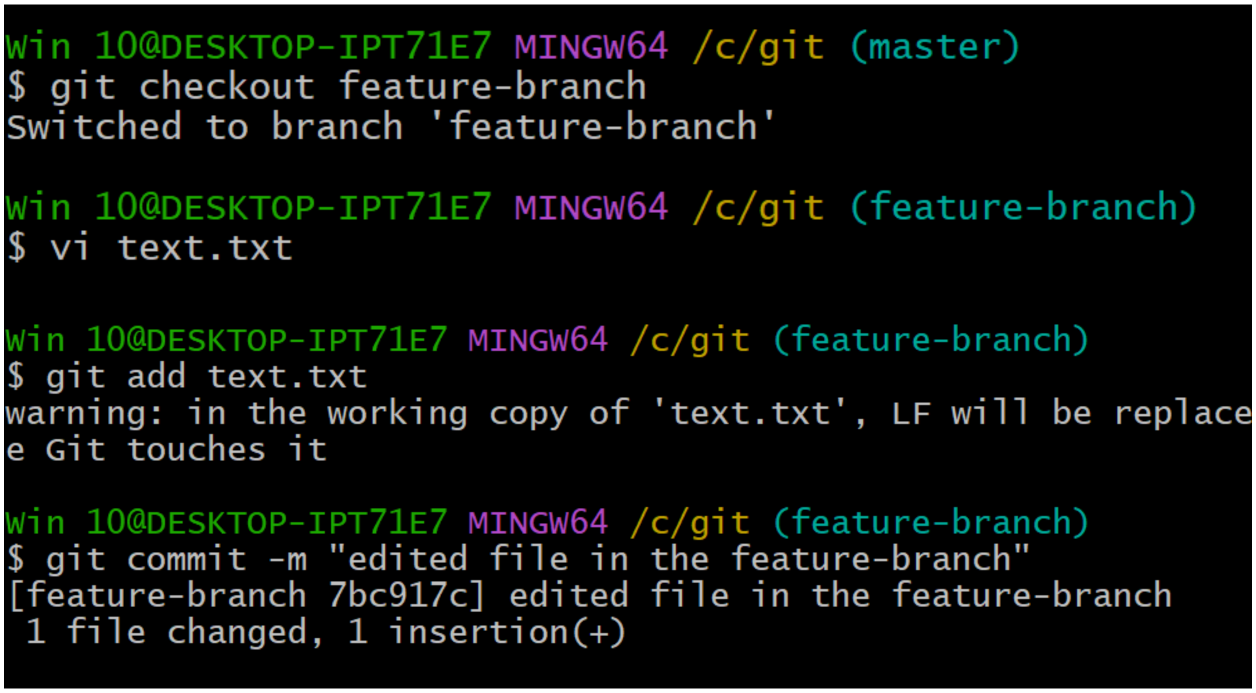
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/flipkart_clone (main)
$ git rebase origin
Successfully rebased and updated refs/heads/main.

Win 10@DESKTOP-IPT71E7 MINGW64 /c/flipkart_clone (main)
$ git log
commit 6c1924a88f0db772e641ac24a7ca6d9ff5994ec6 (HEAD -> main, origin/main, origin/HEAD)
Author: memanju2005 <157971716+memanju2005@users.noreply.github.com>
Date:   Mon Feb 26 15:22:48 2024 +0530

    Create special_note

commit 42c1d65c676c6f86a6607f0f03426894b687ef35
Author: memanju2005 <157971716+memanju2005@users.noreply.github.com>
Date:   Mon Feb 26 15:14:18 2024 +0530

    Create onefile

commit 9565b95d7668f73478d9346fc5f1f8d35833a108
Author: memanju2005 <157971716+memanju2005@users.noreply.github.com>
Date:   Thu Feb 1 20:00:46 2024 +0530

    colours for the project added
```

- git rebase origin →this command is used to bring the changes which are fetched and present in the local repo to the working directory
- after rebasing the remote branch to local branch ,the commit which are made in remote repo that will added to local branch.

6)Collaboration and Remote Repositories

➢ Write the command to merge "feature-branch" into "master" while providing a custom commit message for the merge.

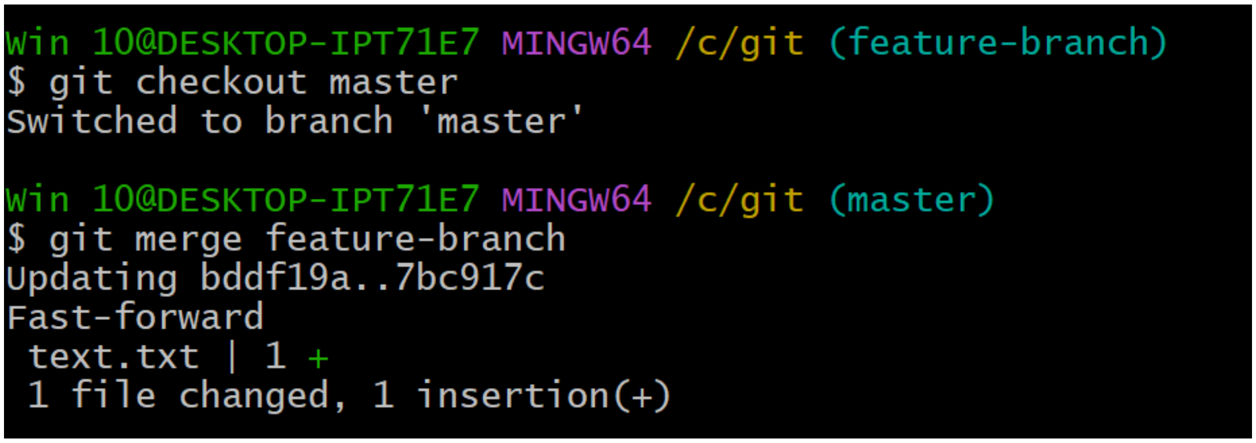
Step1:



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git checkout feature-branch
Switched to branch 'feature-branch'

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (feature-branch)
$ vi text.txt

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (feature-branch)
$ git add text.txt
warning: in the working copy of 'text.txt', LF will be replace
e Git touches it

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (feature-branch)
$ git commit -m "edited file in the feature-branch"
[feature-branch 7bc917c] edited file in the feature-branch
 1 file changed, 1 insertion(+)
```

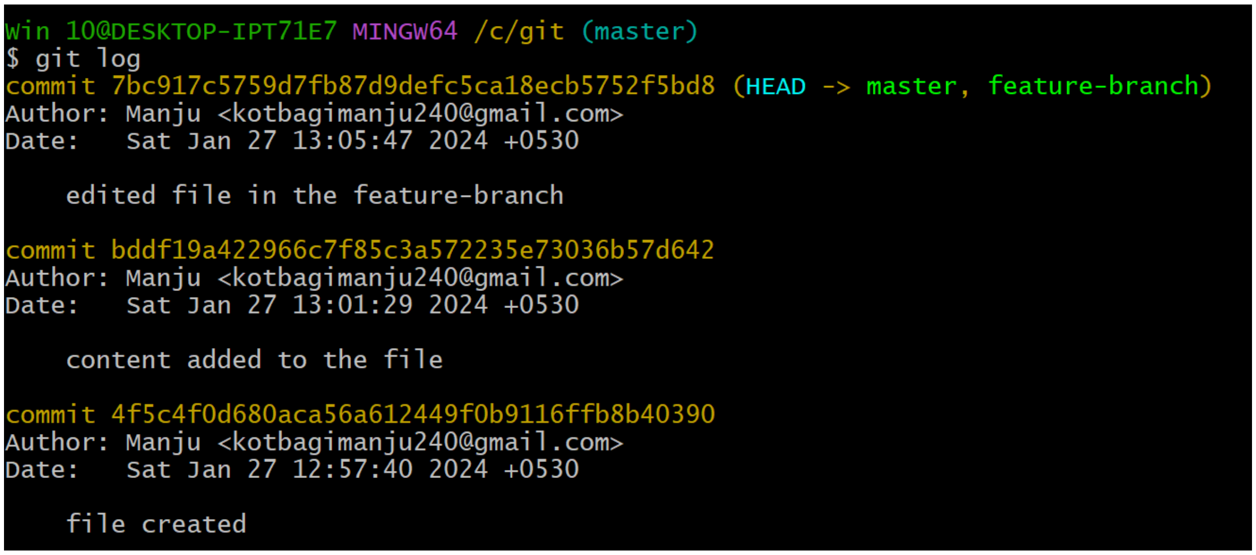
- Move to the feature branch and make some changes in the that branch ,stage and commit the changes.
- For committing we can use additional/optional → -m “message” , will commit the state with appropriate message.

<u>Step2:</u>



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (feature-branch)
$ git checkout master
Switched to branch 'master'

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git merge feature-branch
Updating bddf19a..7bc917c
Fast-forward
 text.txt | 1 +
 1 file changed, 1 insertion(+)
```

- Then checkout to the master branch and merge the branch

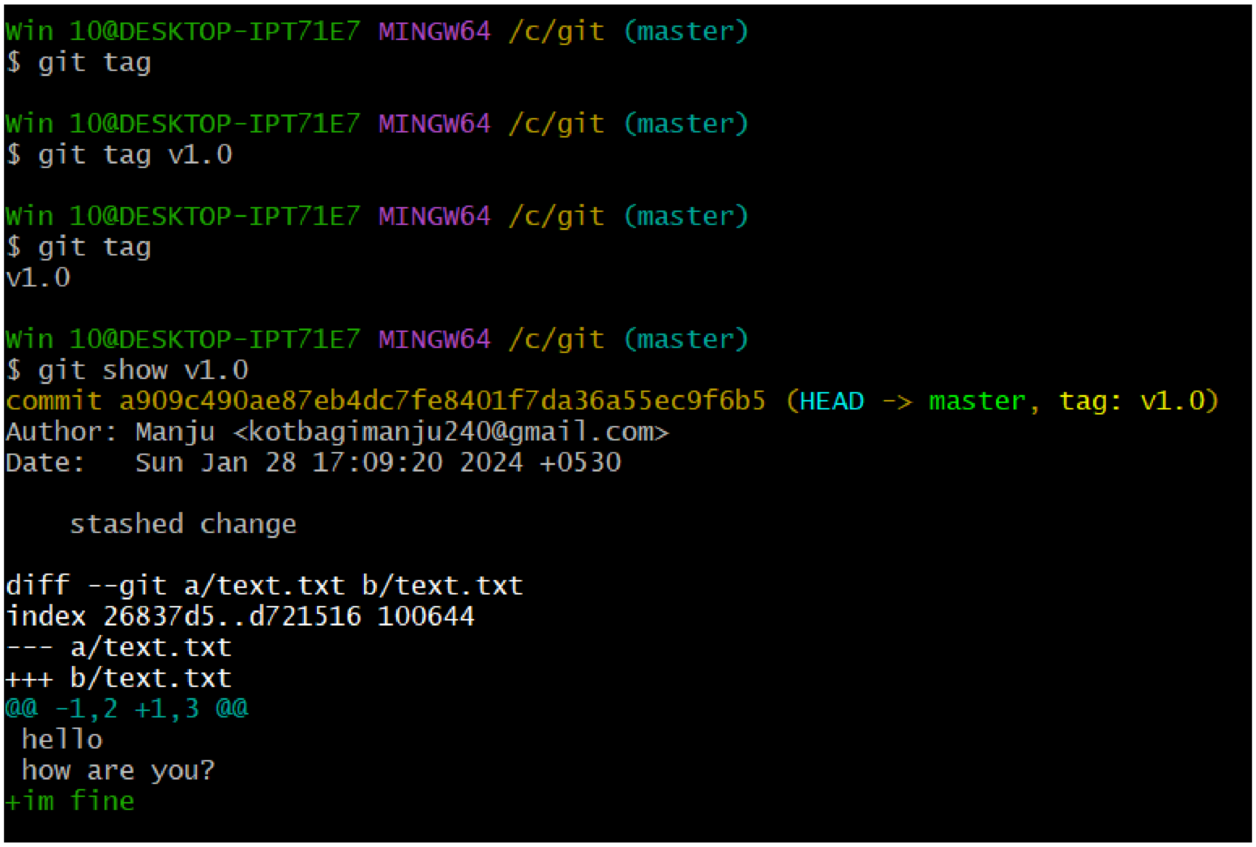


```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git log
commit 7bc917c5759d7fb87d9defc5ca18ecb5752f5bd8 (HEAD -> master, feature-branch)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:05:47 2024 +0530

    edited file in the feature-branch

commit bddf19a422966c7f85c3a572235e73036b57d642
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:01:29 2024 +0530

    content added to the file

commit 4f5c4f0d680aca56a612449f0b9116ffb8b40390
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 12:57:40 2024 +0530

    file created
```

7)Git Tags and Releases

➢ Write the command to create a lightweight Git tag named "v1.0" for a commit in your local repository.

Step1:



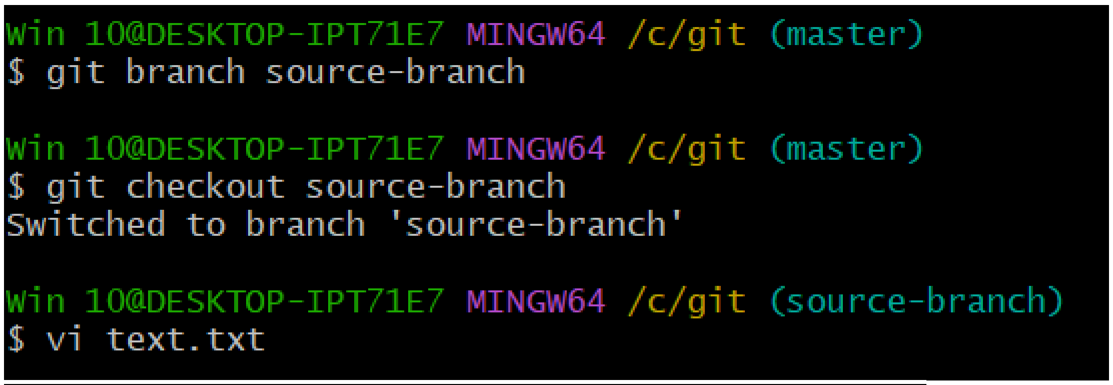
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git tag

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git tag v1.0

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git tag
v1.0

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git show v1.0
commit a909c490ae87eb4dc7fe8401f7da36a55ec9f6b5 (HEAD -> master, tag: v1.0)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sun Jan 28 17:09:20 2024 +0530

    stashed change

diff --git a/text.txt b/text.txt
index 26837d5..d721516 100644
--- a/text.txt
+++ b/text.txt
@@ -1,2 +1,3 @@
 hello
 how are you?
+im fine
```

- git tag v1.0 → this will create a tag of the latest commit(or we can specify the particular commit with commit ID)   or we can also add a tag message using → -m "message".
- git tag→ this command will show the all tags made i.e, v1.0 created.
- git show v1.0 → this will show details in that tag(v1.0) with full description.

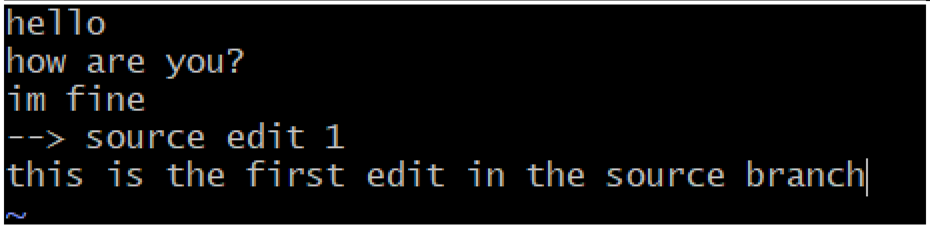
8) Advanced Git Operations

➢ Write the command to cherry-pick a range of commits from "source-branch" to the current branch.

Step1:

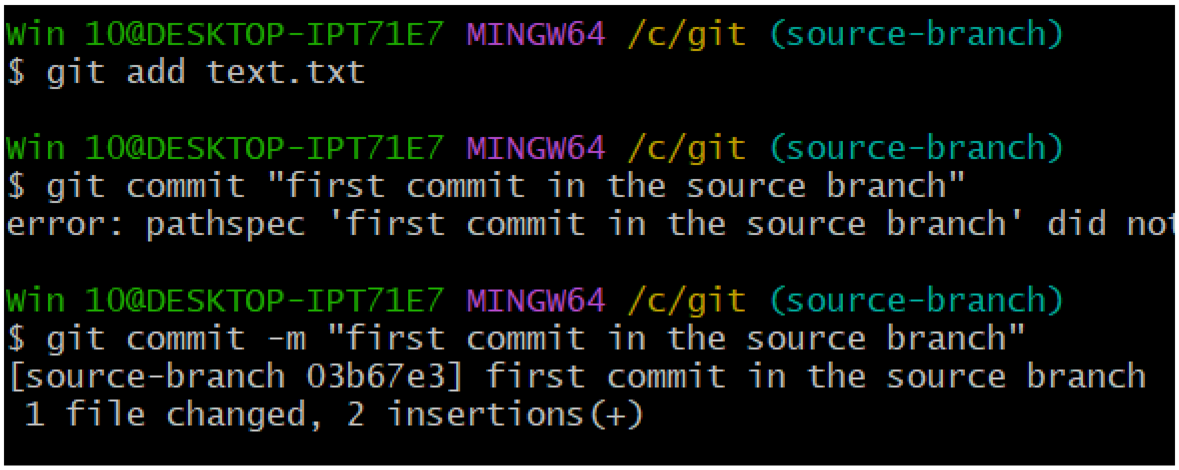


```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git branch source-branch

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git checkout source-branch
Switched to branch 'source-branch'

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (source-branch)
$ vi text.txt
```



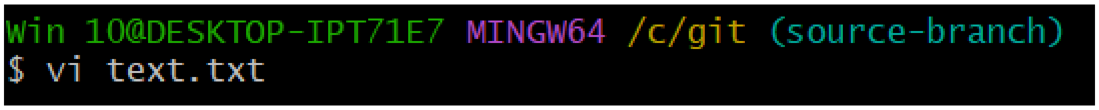
```
hello
how are you?
im fine
--> source edit 1
this is the first edit in the source branch
~
```

- Create a branch named source branch and check out to the source branch.
- And make the first some change in the text.txt file.

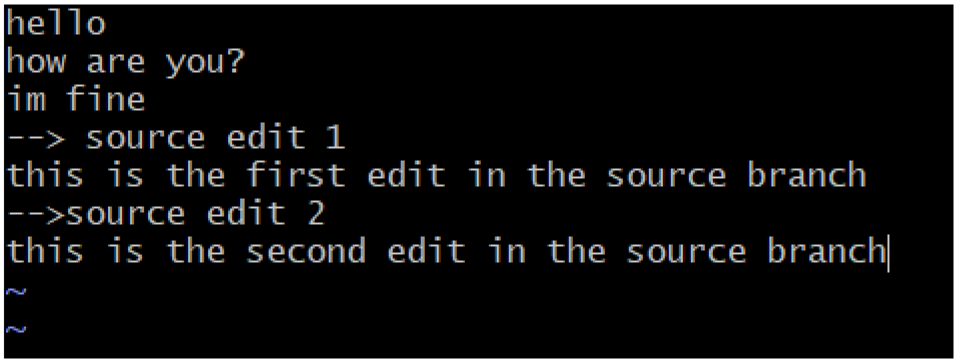


```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (source-branch)
$ git add text.txt

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (source-branch)
$ git commit "first commit in the source branch"
error: pathspec 'first commit in the source branch' did not

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (source-branch)
$ git commit -m "first commit in the source branch"
[source-branch 03b67e3] first commit in the source branch
 1 file changed, 2 insertions(+)
```

- Stage and commit the changes with commit message saying “first commit in the source branch.

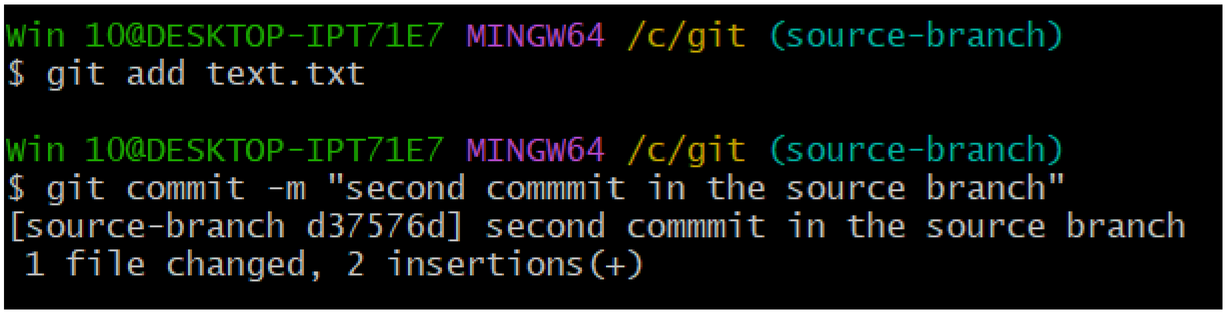


```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (source-branch)
$ vi text.txt
```



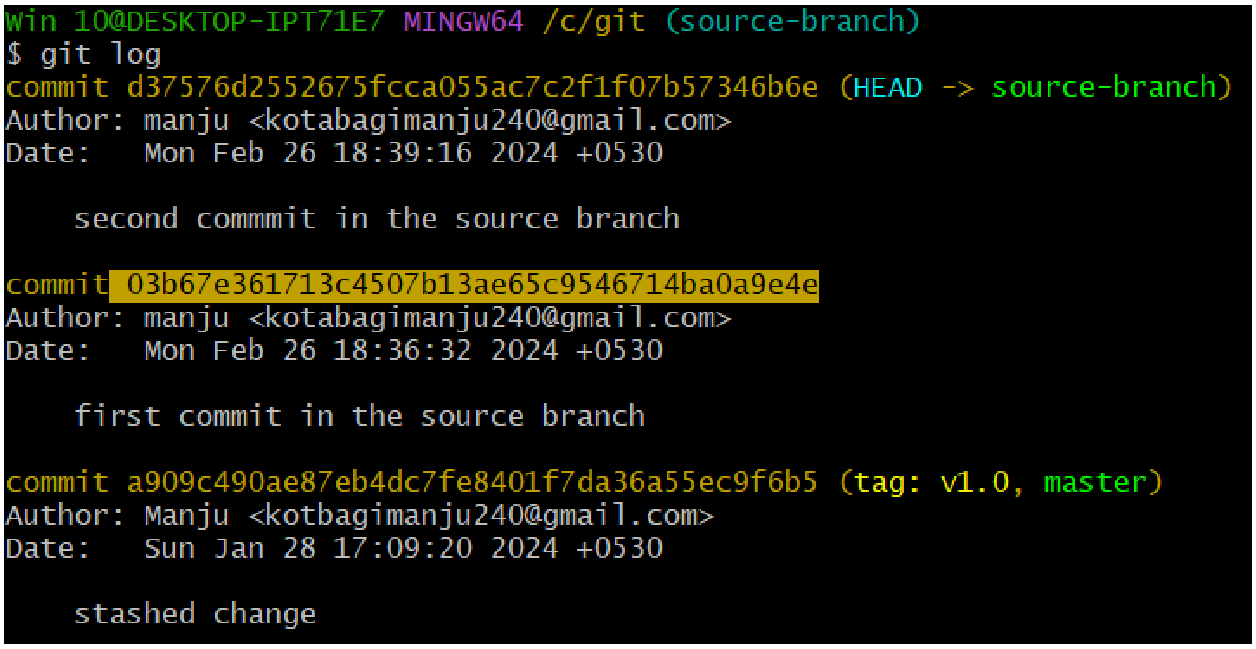
```
hello
how are you?
im fine
--> source edit 1
this is the first edit in the source branch
-->source edit 2
this is the second edit in the source branch
~
~
```

- Again make some changes in the file or add few more line in the text.txt file.

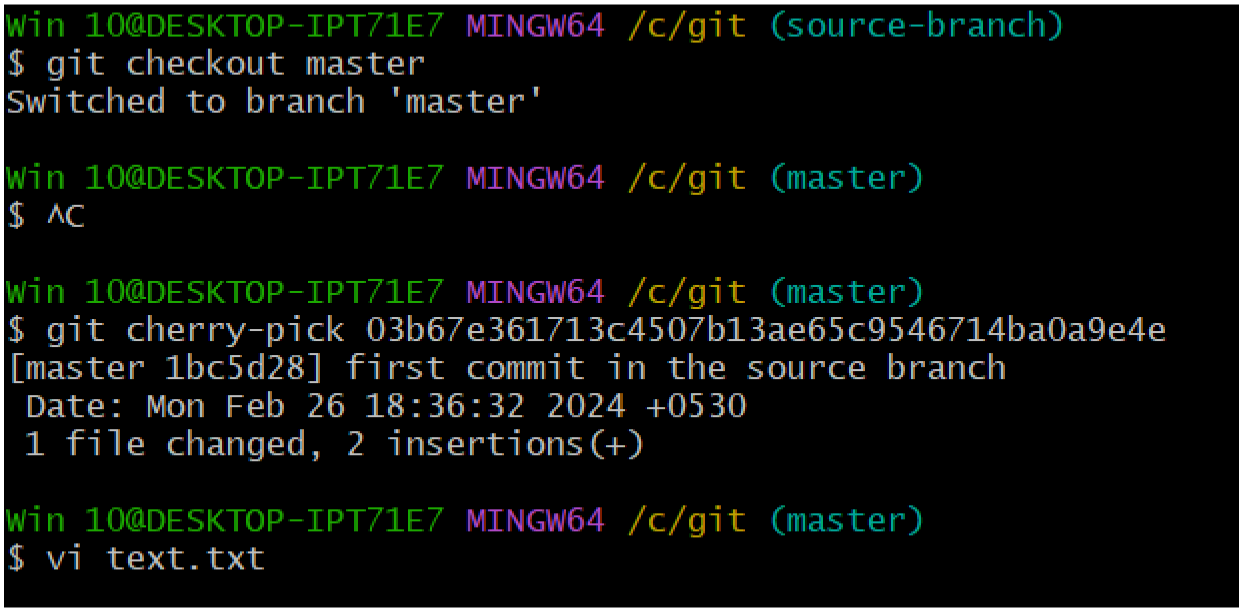


```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (source-branch)
$ git add text.txt

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (source-branch)
$ git commit -m "second commmit in the source branch"
[source-branch d37576d] second commmit in the source branch
 1 file changed, 2 insertions(+)
```

- Stage and commit the changes with message saying “second commit in the source branch”.

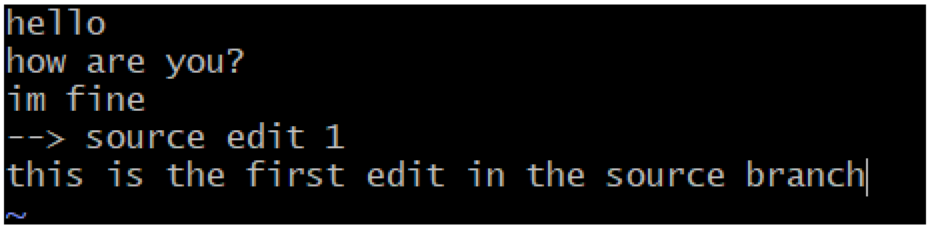

```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (source-branch)
$ git log
commit d37576d2552675fcca055ac7c2f1f07b57346b6e (HEAD -> source-branch)
Author: manju <kotabagimanju240@gmail.com>
Date:   Mon Feb 26 18:39:16 2024 +0530

    second commmit in the source branch

commit 03b67e361713c4507b13ae65c9546714ba0a9e4e
Author: manju <kotabagimanju240@gmail.com>
Date:   Mon Feb 26 18:36:32 2024 +0530

    first commit in the source branch

commit a909c490ae87eb4dc7fe8401f7da36a55ec9f6b5 (tag: v1.0, master)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sun Jan 28 17:09:20 2024 +0530

    stashed change
```

- Here we want copy the commit ID to cherry-pick the specific state from the git log


```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (source-branch)
$ git checkout master
Switched to branch 'master'

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ ^C

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git cherry-pick 03b67e361713c4507b13ae65c9546714ba0a9e4e
[master 1bc5d28] first commit in the source branch
 Date: Mon Feb 26 18:36:32 2024 +0530
 1 file changed, 2 insertions(+)

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ vi text.txt
```

- Now move to the master branch.
- Git cherry-pick <commit ID>→ this will take the mentioned commit ID stage and merge to the master branch.

- Main advantage of using cherry pick is we can pick the required snapshot from the branches and add to the master branch.



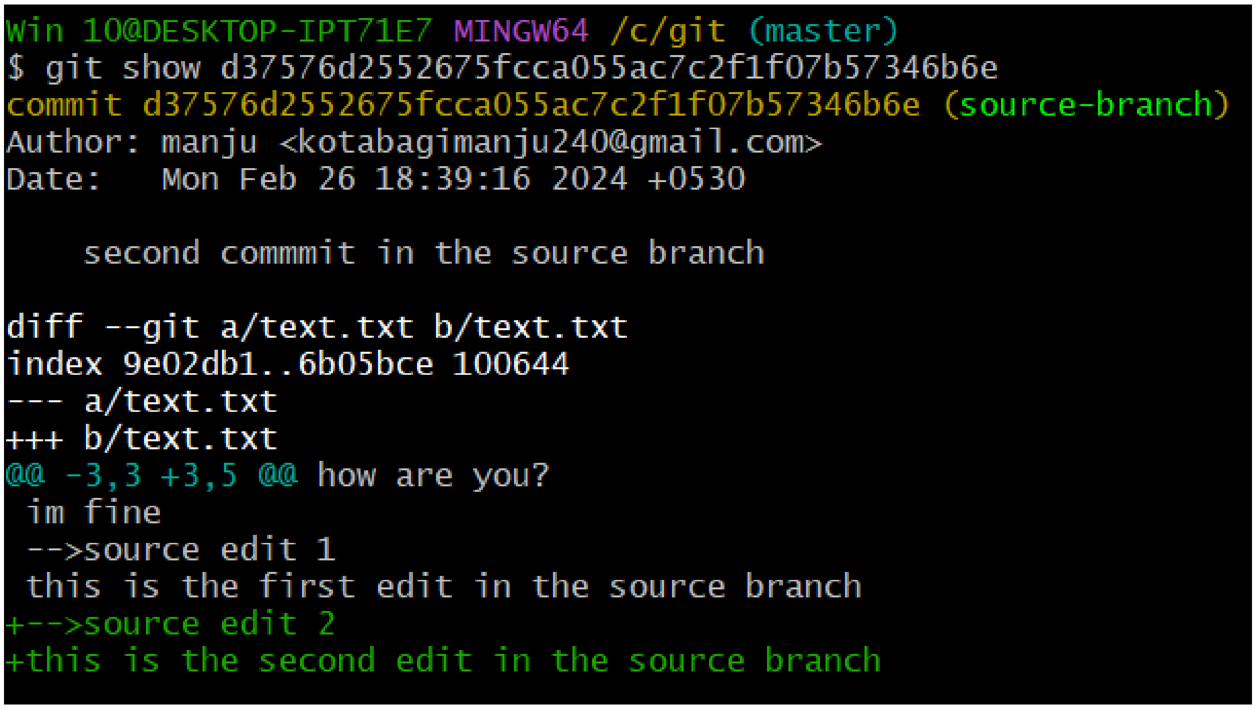
```
hello
how are you?
im fine
--> source edit 1
this is the first edit in the source branch
~
```

- Here we can see the content of the text.txt file at that snapshot is added to the master.

9)Analysing and Changing Git History

➢ Given a commit ID, how would you use Git to view the details of that specific commit, including the author, date, and commit message?

<u>Step1:</u>

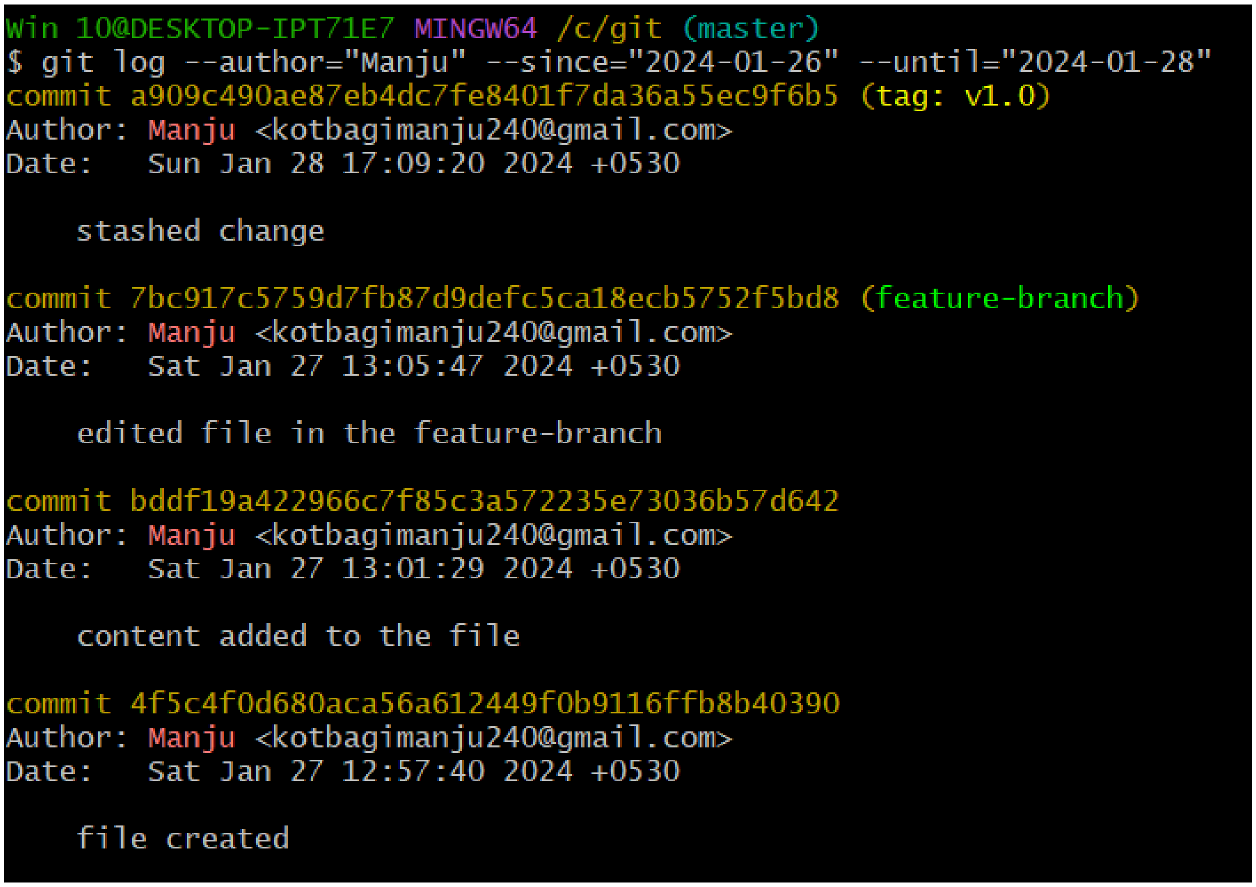


```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git show d37576d2552675fcca055ac7c2f1f07b57346b6e
commit d37576d2552675fcca055ac7c2f1f07b57346b6e (source-branch)
Author: manju <kotabagimanju240@gmail.com>
Date:   Mon Feb 26 18:39:16 2024 +0530

    second commmit in the source branch

diff --git a/text.txt b/text.txt
index 9e02db1..6b05bce 100644
--- a/text.txt
+++ b/text.txt
@@ -3,3 +3,5 @@ how are you?
 im fine
 -->source edit 1
 this is the first edit in the source branch
+-->source edit 2
+this is the second edit in the source branch
```

- To view the details of the specific commit including author, date and commit message we should copy the specific commit which you want to view in detail.
- git show <commit ID> → this will show the full detail of the commit ID mentioned ,added changes will be shown in green colour and deleted changes will be shown in red colour.

10)Analysing and Changing Git History

- ➢ Write the command to list all commits made by the author "JohnDoe" between "2024-01-27" and "2023-01-28."



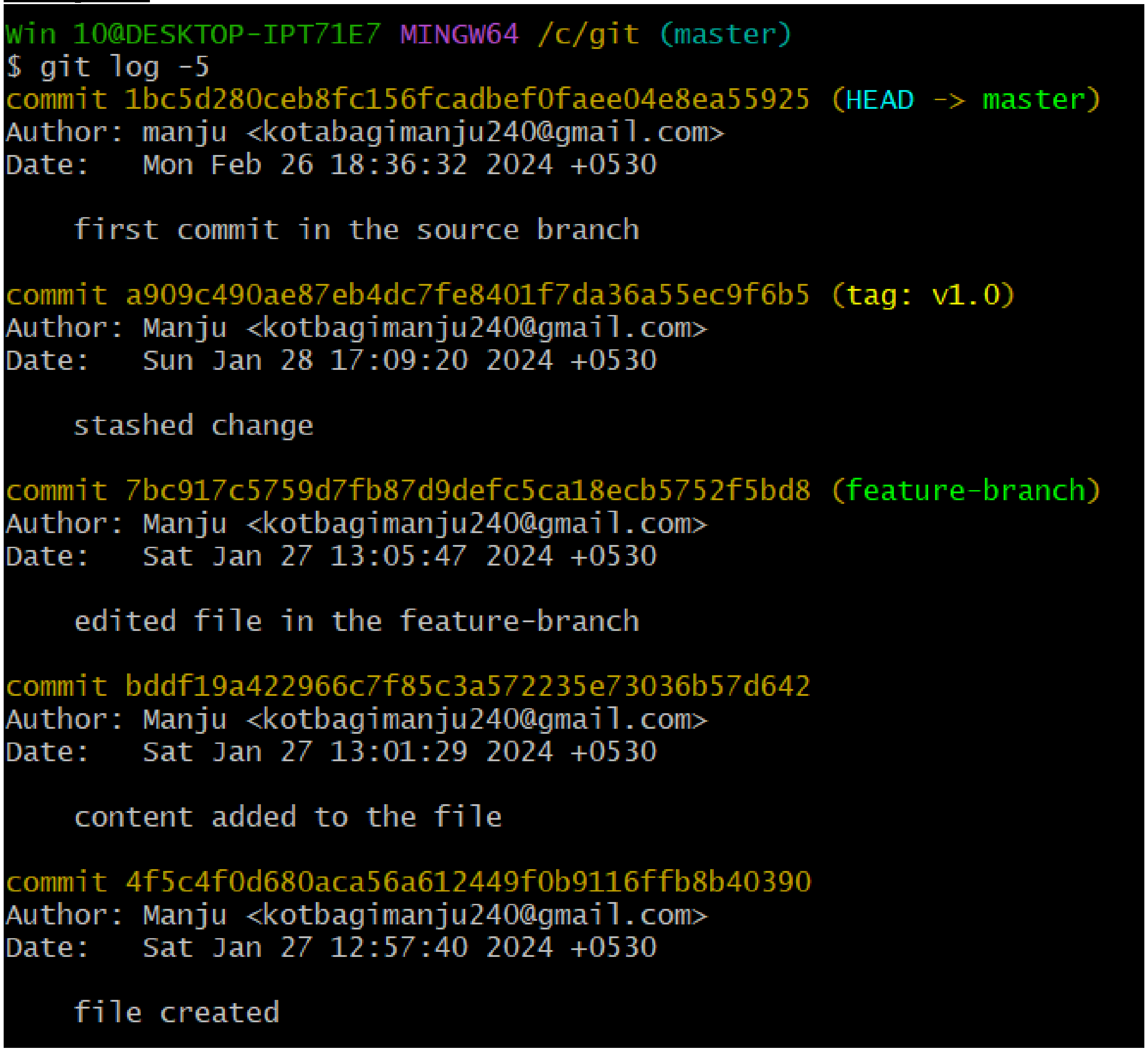
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git log --author="Manju" --since="2024-01-26" --until="2024-01-28"
commit a909c490ae87eb4dc7fe8401f7da36a55ec9f6b5 (tag: v1.0)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sun Jan 28 17:09:20 2024 +0530

    stashed change

commit 7bc917c5759d7fb87d9defc5ca18ecb5752f5bd8 (feature-branch)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:05:47 2024 +0530

    edited file in the feature-branch

commit bddf19a422966c7f85c3a572235e73036b57d642
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:01:29 2024 +0530

    content added to the file

commit 4f5c4f0d680aca56a612449f0b9116ffb8b40390
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 12:57:40 2024 +0530

    file created
```

- git log --author="Manju" --since="2024-01-26" --until="2024-01-28" →this will show all the commits made by the author “Manju” b/w dated “2024-01-26" and "2024-01-28".

11)Analysing and Changing Git History

➢ Write the command to display the last five commits in the repository's history.

Step 1:



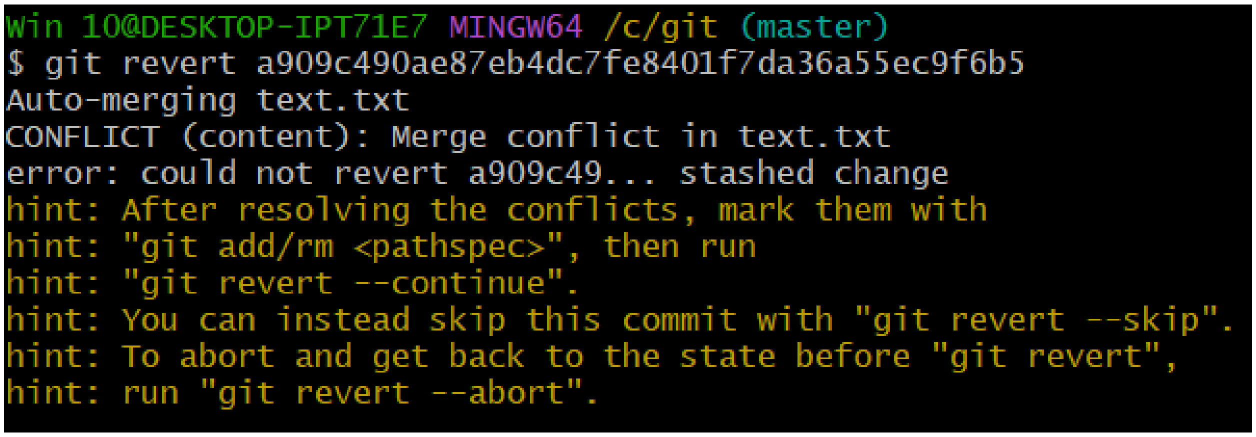
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git log -5
commit 1bc5d280ceb8fc156fcadbef0faee04e8ea55925 (HEAD -> master)
Author: manju <kotabagimanju240@gmail.com>
Date:   Mon Feb 26 18:36:32 2024 +0530

    first commit in the source branch

commit a909c490ae87eb4dc7fe8401f7da36a55ec9f6b5 (tag: v1.0)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sun Jan 28 17:09:20 2024 +0530

    stashed change

commit 7bc917c5759d7fb87d9defc5ca18ecb5752f5bd8 (feature-branch)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:05:47 2024 +0530

    edited file in the feature-branch

commit bddf19a422966c7f85c3a572235e73036b57d642
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 13:01:29 2024 +0530

    content added to the file

commit 4f5c4f0d680aca56a612449f0b9116ffb8b40390
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sat Jan 27 12:57:40 2024 +0530

    file created
```

- git log –n →this will display last n no.of commits. Here n is 5.

12)Analysing and Changing Git History

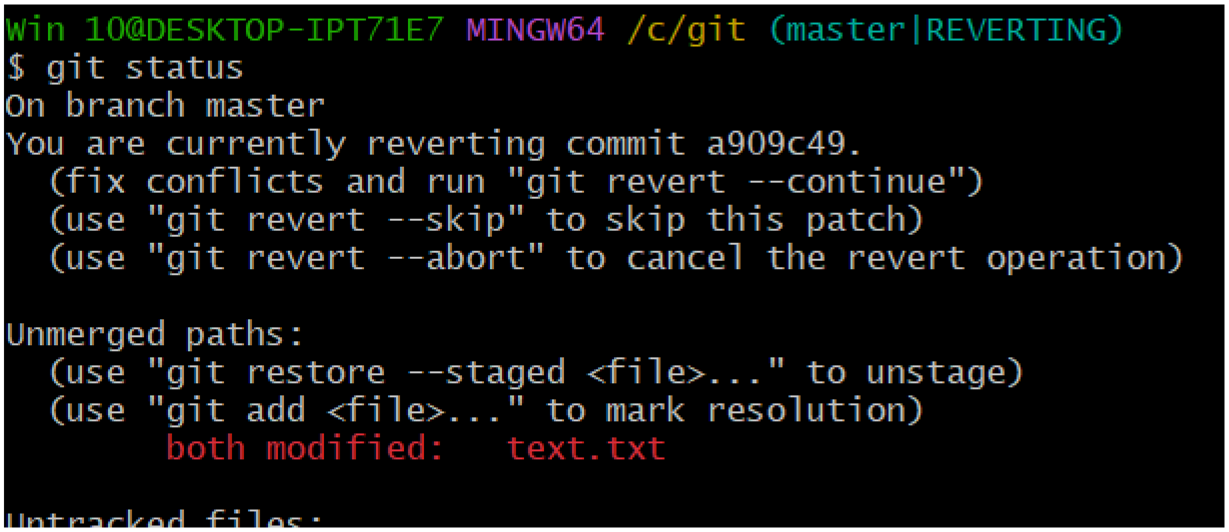
➢ Write the command to undo the changes introduced by the commit with the ID "abc123".

Step 1:



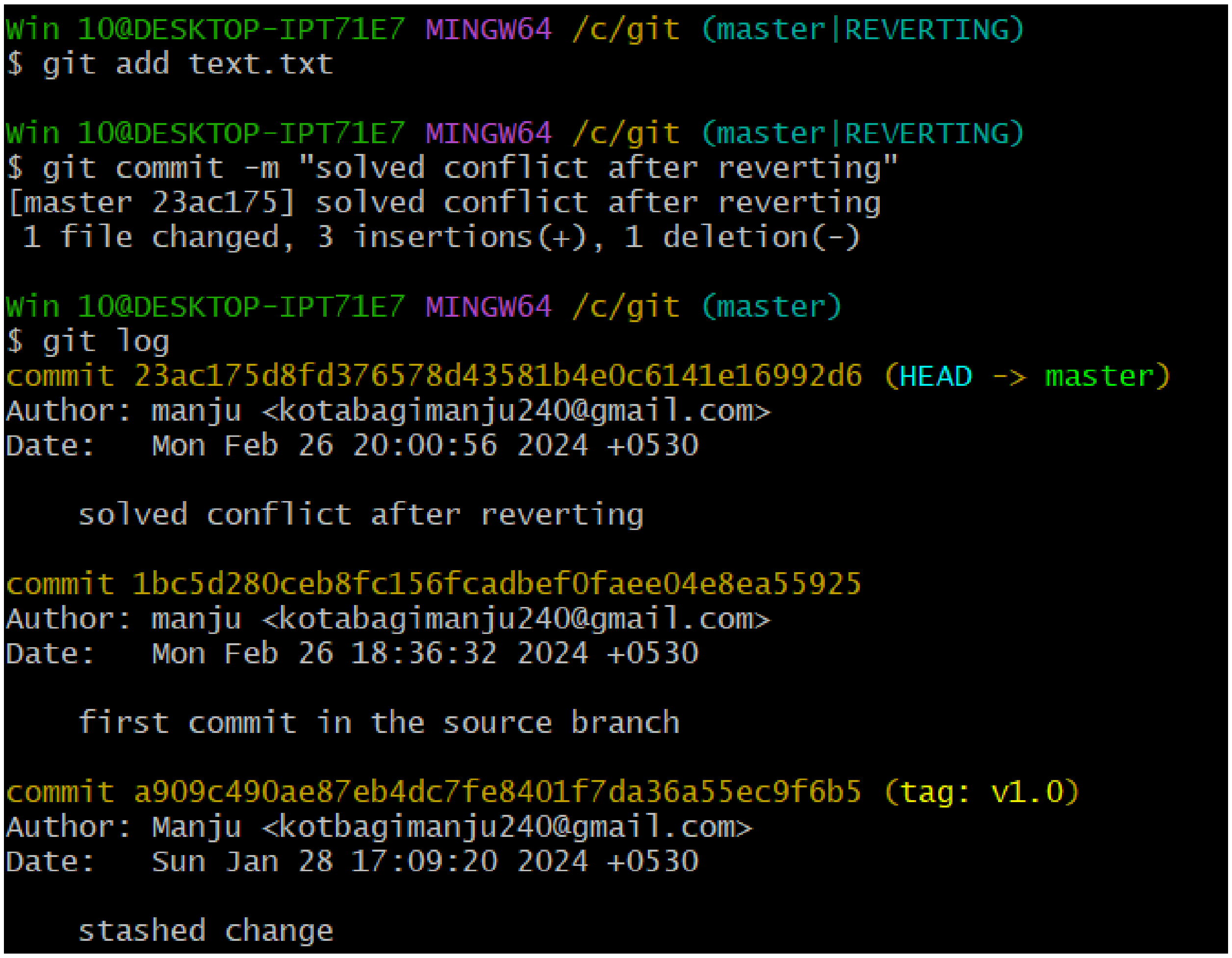
```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git revert a909c490ae87eb4dc7fe8401f7da36a55ec9f6b5
Auto-merging text.txt
CONFLICT (content): Merge conflict in text.txt
error: could not revert a909c49... stashed change
hint: After resolving the conflicts, mark them with
hint: "git add/rm <pathspec>", then run
hint: "git revert --continue".
hint: You can instead skip this commit with "git revert --skip".
hint: To abort and get back to the state before "git revert",
hint: run "git revert --abort".
```

- git revert <commit ID > →this will revert to the that stage of commit
- In case of failed of auto conflict , conflict will arise and we should solve conflict.



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master|REVERTING)
$ git status
On branch master
You are currently reverting commit a909c49.
  (fix conflicts and run "git revert --continue")
  (use "git revert --skip" to skip this patch)
  (use "git revert --abort" to cancel the revert operation)

Unmerged paths:
  (use "git restore --staged <file>..." to unstage)
  (use "git add <file>..." to mark resolution)
        both modified:   text.txt

Untracked files:
```

- After git status we can see the modified file “text.txt” which we had modified the file to solved the conflict.



```
Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master|REVERTING)
$ git add text.txt

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master|REVERTING)
$ git commit -m "solved conflict after reverting"
[master 23ac175] solved conflict after reverting
 1 file changed, 3 insertions(+), 1 deletion(-)

Win 10@DESKTOP-IPT71E7 MINGW64 /c/git (master)
$ git log
commit 23ac175d8fd376578d43581b4e0c6141e16992d6 (HEAD -> master)
Author: manju <kotabagimanju240@gmail.com>
Date:   Mon Feb 26 20:00:56 2024 +0530

    solved conflict after reverting

commit 1bc5d280ceb8fc156fcadbef0faee04e8ea55925
Author: manju <kotabagimanju240@gmail.com>
Date:   Mon Feb 26 18:36:32 2024 +0530

    first commit in the source branch

commit a909c490ae87eb4dc7fe8401f7da36a55ec9f6b5 (tag: v1.0)
Author: Manju <kotbagimanju240@gmail.com>
Date:   Sun Jan 28 17:09:20 2024 +0530

    stashed change
```

- We should stage the file and commit with the appropriate message