

PROJECT MANAGEMENT WITH GIT

1. Setting Up and Basic Commands.

Program: Initialize a new git repository in a directory. Create a new file and add it to the staging area and commit the changes with the appropriate commit message.

Step 1:

- mkdir Hi -> creates a folder/directory in the present working directory.
- Cd Hi -> changes the directory to the git folder which was created.
- git init -> initializes an empty git repository.
- We can see the .git folder created in the git folder, in some cases the file is hidden and to see the hidden file we need to click on view the hidden files.

```
MINGW64:/d/Yashas_125/hi
vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git init
Reinitialized existing Git repository in D:/Yashas_125/hi/.git/
vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ touch dot.txt
vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git add .
vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git commit -m "file-created"
On branch master
nothing to commit, working tree clean
vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git log
commit cd384acd86f8d21db1a6284dfd929aff1cdbc525 (HEAD -> master)
Author: Ambresh <ambreshkumbar84@gmail.com>
Date: Tue Nov 4 14:34:11 2025 +0530

    File created
```

Step 2:

- touch dot.txt -> creates a empty file pf txt extension in the current directory.
- git add ./git add dot.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 the committed.
- git log -> displays all the history of commits with commit message along with the author name and email.
- Every commits has unique commit ID.

Step 3:

```
vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ nano pk.txt

vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git add .
warning: in the working copy of 'pk.txt', LF will be replaced by CRLF the next time Git touches it
```

- Here the file content is added using nano pk.txt and then file is staged and committed with appropriate message.

Step 4:

```
vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git commit -m "added a file"
[master 4c57442] added a file
1 file changed, 3 insertions(+)
create mode 100644 pk.txt

vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git branch
* master

vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git log
commit 4c57442801b79fe48123d3bcc3eb35ecf9c21e24 (HEAD -> master)
Author: Ambresh <ambreshkumbar84@gmail.com>
Date: Tue Nov 4 14:46:41 2025 +0530

    added a file

commit cd384acd86f8d21db1a6284dfd929aff1cdbd525
Author: Ambresh <ambreshkumbar84@gmail.com>
Date: Tue Nov 4 14:34:11 2025 +0530

    File created
```

- git status -> it checks the status, like on which branch we are and if there are any changes made which are not committed.
- If no other changes have been made after recent commit then it displays the working tree is clean.
- git log -> this will display the history of the commits.

2. Creating and Managing Branches:

Program: Create a new branch named “feature-branch.” Switch to “master” branch. Merge the “feature-branch” into “master”.

Step 1:

```
MINGW64/d/Yashas_125/hi
vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git branch new

vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git branch
* master
  new

vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git checkout new
Switched to branch 'new'

vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ nano pk.txt
```

- 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 branches we have made and the current branch will be in green colour.
- 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.

```
vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ git add .

vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ git commit -m"Edited a file"
On branch new
nothing to commit, working tree clean

vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ git commit -n "edited a file"
error: pathspec 'edited a file' did not match any file(s) known to git

vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ git checkout master
Switched to branch 'master'

vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git merge new
Already up to date.
```

- Here in the feature-branch we staged the file and committed with message saying “edited file in 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.

```
vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git merge new
Updating 4c57442..9fb0a5c
Fast-forward
 pk.txt | 4 ++--
 1 file changed, 2 insertions(+), 2 deletions(-)

vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git log
commit 9fb0a5cf24ef5771d0afa8bd576a494c27dbaa8b (HEAD -> master, new)
Author: Ambresh <ambreshkumbar84@gmail.com>
Date: Tue Nov 4 14:59:10 2025 +0530

    Edited a file

commit 4c57442801b79fe48123d3bcc3eb35ecf9c21e24
Author: Ambresh <ambreshkumbar84@gmail.com>
Date: Tue Nov 4 14:46:41 2025 +0530

    added a file

commit cd384acd86f8d21db1a6284dfd929aff1cddb525
Author: Ambresh <ambreshkumbar84@gmail.com>
Date: Tue Nov 4 14:34:11 2025 +0530

    File created
```

- 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 merged .the changes or the edits in the feature-branch will be merged
- git log -> here the history of the commits will be visible.

3.Creating and Managing Branches:

Program: write the commands to stash your changes, switch branches, and then apply the stashed changes commands to stash your changes, switch branches, and then apply the stashed changes

Step 1:

```
MINGW64/d/Yashas_125/hi  
vtula@VTU MINGW64 /d/Yashas_125/hi (master)  
$ git checkout new  
Switched to branch 'new'
```

- 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 committed the changes in the text.txt file.

```
vtula@VTU MINGW64 /d/Yashas_125/hi (new)  
$ git stash  
Saved working directory and index state WIP on new: 9fb0a5c Edited a file  
vtula@VTU MINGW64 /d/Yashas_125/hi (new)  
$ git stash apply  
On branch new  
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:   pk.txt  
  
no changes added to commit (use "git add" and/or "git commit -a")
```

Step 2:

- 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 committed the changes.
- The changes will be saved in the branch without the committing the changes.
- The command used for stashing the changes -> git stash
- 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 the applied stash is not staged and committed in the master branch.

```
vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ git status
On branch new
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:   pk.txt

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

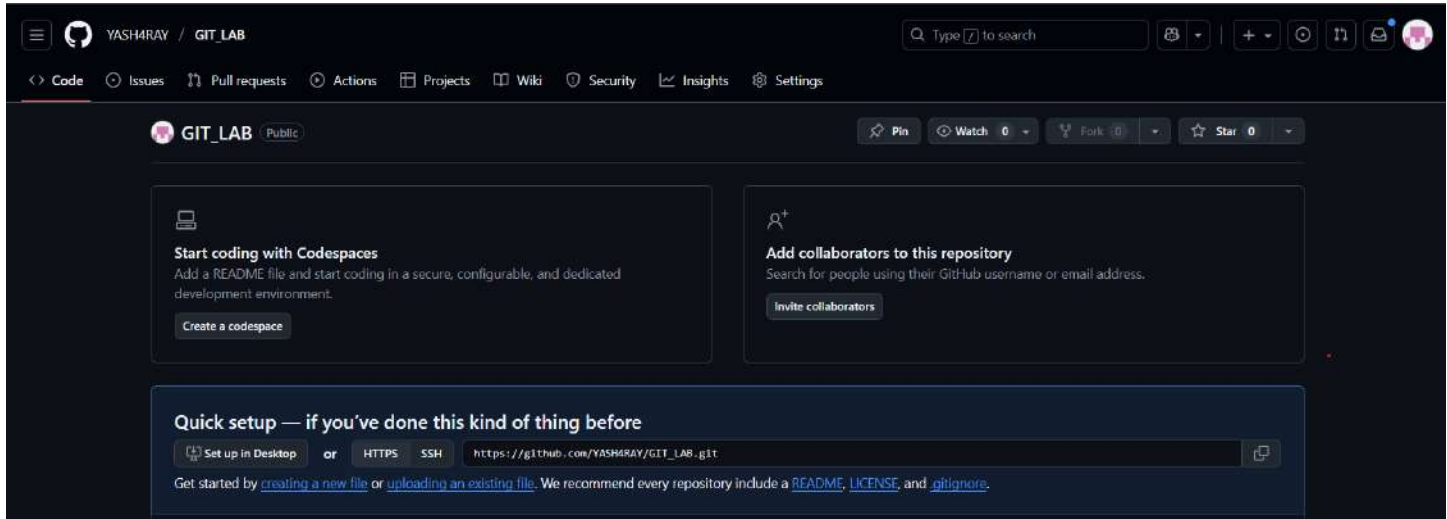
vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ git add .

vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ git status
On branch new
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   pk.txt
```

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

4. Collaboration and Remote Repositories:

Program: Clone a remote Git repository to your local machine.



- To clone a remote 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 lists links in that , we copied the “HTTPS” link from that.

```
vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ mkdir clone

vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ cd clone

vtula@VTU MINGW64 /d/Yashas_125/hi/clone (new)
$ git clone https://github.com/YASH4RAY/GIT_LAB.git
fatal: repository '' does not exist
```

Step 2:

- In the second step we need to open our git bash in some directory where you need clone the remote repository, we need to move to that location using “cd ” 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/YASH4RAY/GIT_LAB.git -> this command will copy the repository from remote to the local machine in the working directory
- you can see above the “GIT_LAB” repository is successfully copied in the clone directory/folder.

5.Collabration and Remote Repositories:

Program: Fetch the latest changes from a remote repository and rebase your local branch onto the updated remote branch

```
vtula@VTU MINGW64 /d/Yashas_125/hi/clone (new)
$ cd GIT_LAB

vtula@VTU MINGW64 /d/Yashas_125/hi/clone/GIT_LAB (main)
$ git fetch origin

vtula@VTU MINGW64 /d/Yashas_125/hi/clone/GIT_LAB (main)
$ git rebase origin
fatal: invalid upstream 'origin'
```

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

```
vtula@VTU MINGW64 /d/Yashas_125/hi/clone/GIT_LAB (main)
$ git log
fatal: your current branch 'main' does not have any commits yet

vtula@VTU MINGW64 /d/Yashas_125/hi/clone/GIT_LAB (main)
$ git add ,
fatal: pathspec ',' did not match any files

vtula@VTU MINGW64 /d/Yashas_125/hi/clone/GIT_LAB (main)
$ git add .
warning: in the working copy of 'readme.txt', LF will be replaced by CRLF

vtula@VTU MINGW64 /d/Yashas_125/hi/clone/GIT_LAB (main)
$ git commit -m "Added a new file"
[main (root-commit) 4120e09] Added a new file
1 file changed, 1 insertion(+)
create mode 100644 readme.txt

vtula@VTU MINGW64 /d/Yashas_125/hi/clone/GIT_LAB (main)
$ git log
commit 4120e095e361697d2d2409f4daf9606e2deb16a2 (HEAD -> main)
Author: Ambresh <ambreshkumbar84@gmail.com>
Date: Tue Nov 4 15:47:32 2025 +0530

    Added a new file
```

- 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:

Program: Write the command to merge “feature-branch” into “master” while providing a custom commit message for the merge

Step 1:

```

MINGW64:/d/Yashas_125/hi
vtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git checkout new
Switched to branch 'new'

vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ nano pk.txt

vtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ git add .
warning: adding embedded git repository: clone/GIT_LAB
hint: You've added another git repository inside your current repository.
hint: Clones of the outer repository will not contain the contents of
hint: the embedded repository and will not know how to obtain it.
hint: If you meant to add a submodule, use:
hint:
hint:   git submodule add <url> clone/GIT_LAB
hint:
hint: If you added this path by mistake, you can remove it from the
hint: index with:
hint:
hint:   git rm --cached clone/GIT_LAB
hint:
hint: See "git help submodule" for more information.
hint: Disable this message with "git config set advice.addEmbeddedRepo false"

```

- 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.

Step 2:

- Then checkout to the master branch and merge the branch

```
rtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git merge new
Updating 9fb0a5c..4ae4009
Fast-forward
 clone/GIT_LAB | 1 +
 pk.txt       | 3 +++
 2 files changed, 4 insertions(+)
 create mode 160000 clone/GIT_LAB

rtula@VTU MINGW64 /d/Yashas_125/hi (master)
$ git log
commit 4ae4009c8fe4a73a456aa70caa12f5edbbc24d88 (HEAD -> master, new)
Author: Ambresh <ambreshkumbar84@gmail.com>
Date: Tue Nov 4 15:58:29 2025 +0530

    edited a file

commit 7c6b33bbbd317065370cd417bf7e3040699585c6
Author: Ambresh <ambreshkumbar84@gmail.com>
Date: Tue Nov 4 15:22:19 2025 +0530

    stash changed
```

```
rtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ git commit -m "edited a file"
[new 4ae4009] edited a file
 2 files changed, 3 insertions(+)
 create mode 160000 clone/GIT_LAB

rtula@VTU MINGW64 /d/Yashas_125/hi (new)
$ git checkout master
warning: unable to rmdir 'clone/GIT_LAB': Directory not empty
Switched to branch 'master'
```

7. Git Tags and Releases:

Program: Write the command to create a lightweight Git tag named “v1.0” for a commit in your local repository.

Step 1:

```
YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git tag v1.0

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git tag v1.0 f9b5f495640236e1edbc38f678e1ac616a8f7ab8
fatal: tag 'v1.0' already exists

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git show v1.0
commit f9b5f495640236e1edbc38f678e1ac616a8f7ab8 (HEAD -> master, tag: v1.0)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 18:59:44 2025 +0530

    Added a new file

diff --git a/pk.txt b/pk.txt
new file mode 100644
index 0000000..8676a5a
--- /dev/null
+++ b/pk.txt
@@ -0,0 +1,2 @@
+Name: Yashas.R
+USN: 2VX24AD125

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$
```

- 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.

Step 1:

```
YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git branch
* master
  new
  source-branch

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git checkout source-branch
Switched to branch 'source-branch'

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (source-branch)
$ nano pk.txt
```

- Create a branch named source branch and check out to the source branch.
- And make the first some changes in the text.txt file.
- Stage and commit the changes with commit message saying first commit in the source branch.

Step 2:

- Again make some changes in the file or add few more line in the text.txt file.
- Stage and commit the changes with message saying “second commit in the source branch”

```

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (source-branch)
$ git add .

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (source-branch)
$ git commit -m "This is edited in source branch"
[source-branch c3d8b2b] This is edited in source branch
1 file changed, 1 insertion(+), 1 deletion(-)

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (source-branch)
$ nano pk.txt

```

Step 3:

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

Step 4:

- Now move to the master branch.
- Git cheery-pick -> 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.

```

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (source-branch)
$ git add .

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (source-branch)
$ git commit -m "this is edited in source branch"
[source-branch 466e397] this is edited in source branch
1 file changed, 1 insertion(+)

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (source-branch)
$ git log
commit 466e397c6a9a81eadd3c8a9be8d405b963c04ac1 (HEAD -> source-branch)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 19:20:44 2025 +0530

    this is edited in source branch

commit c3d8b2bf6caa1e7b6304d41e79902c7a3d250420
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 19:14:47 2025 +0530

    This is edited in source branch

commit e8cb8571fea89337bf600a4eb8d2402a82997c84
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 19:12:06 2025 +0530

    commit for P8

commit f9b5f495640236e1edbc38f678e1ac616a8f7ab8 (tag: v1.0, new, master)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 18:59:44 2025 +0530

    Added a new file

```

```

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (source-branch)
$ git checkout master
Switched to branch 'master'

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git cherry-pick e8cb8571fea89337bf600a4eb8d2402a82997c84
[master d28883d] commit for P8
Date: Tue Nov 4 19:12:06 2025 +0530
1 file changed, 2 insertions(+)

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ |

```

9. Analysing and Changing Git History:

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

Step 1:

```

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git show --c3d8b2bf6caa1e7b630hd41e79902c7a3d250420
fatal: unrecognized argument: --c3d8b2bf6caa1e7b630hd41e79902c7a3d250420

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git show f9b5f495640236e1edbc38f678e1ac616a8f7ab8
commit f9b5f495640236e1edbc38f678e1ac616a8f7ab8 (tag: v1.0, new)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 18:59:44 2025 +0530

    Added a new file

diff --git a/pk.txt b/pk.txt
new file mode 100644
index 0000000..8676a5a
--- /dev/null
+++ b/pk.txt
@@ -0,0 +1,2 @@
+Name: Yashas.R
+USN: 2VX24AD125

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$

```


- 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 ->` 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:

Program:Write the command to list all commits made by the author “JohnDoe” between “2024-11-17” and “2024-11-19.”

Step 1:

```
YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git log
commit d28883d23ad28338353a0b5798d335bc844ac7dd (HEAD -> master)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 19:12:06 2025 +0530

    commit for P8

commit f9b5f495640236e1edbc38f678e1ac616a8f7ab8 (tag: v1.0, new)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 18:59:44 2025 +0530

    Added a new file
```

- `git log --author="Yashas" --since="2024-04-" --until="2024-11-19" ->`this will show all the commits made by the author “Hemanth” b/w dated “2024-11-17” and “2024-11-19”

11. Analysing and Changing Git History:

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

Step 1:

```

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git log
commit d28883d23ad28338353a0b5798d335bc844ac7dd (HEAD -> master)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 19:12:06 2025 +0530

    commit for P8

commit f9b5f495640236e1edbc38f678e1ac616a8f7ab8 (tag: v1.0, new)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 18:59:44 2025 +0530

    Added a new file

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git log -s
commit d28883d23ad28338353a0b5798d335bc844ac7dd (HEAD -> master)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 19:12:06 2025 +0530

    commit for P8

commit f9b5f495640236e1edbc38f678e1ac616a8f7ab8 (tag: v1.0, new)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 18:59:44 2025 +0530

    Added a new file

YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ |

```

- `git log -n` -> this will display last n no. of commits . Here n is 5.

12. Analysing and Changing Git History:

Program: Write the command to undo the changes introduced by the commit with the ID “abc123”.

Step 1:

```
File Edit View

Name: Yashas.R
USN: 2VX24AD125

A commit for program 8
```

- The above image is before reverting

```
YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git log
commit d465c5ca867082da97ef9a00819c8f6ba1a9759b (HEAD -> master)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 21:45:27 2025 +0530

    Revert "abc123"

This reverts commit 3385b8ef0fff539b1438e0e9c857b3abc1f61a62.
```

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

```
Revert "abc123"

This reverts commit 3385b8ef0fff539b1438e0e9c857b3abc1f61a62.

# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
#
# On branch master
# Changes to be committed:
#   modified:   pk.txt
#
```

- The above image is after reverting

```
YASH@Yashas MINGW64 ~/Downloads/GIT/GIT_lab/hi (master)
$ git log
commit d465c5ca867082da97ef9a00819c8f6ba1a9759b (HEAD -> master)
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 21:45:27 2025 +0530

    Revert "abc123"

    This reverts commit 3385b8ef0fff539b1438e0e9c857b3abc1f61a62.

commit 3385b8ef0fff539b1438e0e9c857b3abc1f61a62
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 21:42:50 2025 +0530

    abc123

commit 3f6b1492321ce9b609b08ecd7abd871400555b45
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 21:30:16 2025 +0530

    abc123
    Revert "Added a line for program 12"

    This reverts commit f4291c9ce229bcf3ee44ceeab334d474962475b5.

commit f4291c9ce229bcf3ee44ceeab334d474962475b5
Author: YASH4RAY <yashas.4628@gmail.com>
Date: Tue Nov 4 21:27:20 2025 +0530

    Added a line for program 12
```