

Lesson 03 Demo 04

Stashing in Git

Objective: To use Git stashing and temporarily save uncommitted changes, allowing switch between branches without losing progress

Tools required: Git and Ubuntu

Prerequisites: None

Steps to be followed:

1. Set up the environment
2. Initialize Git and make initial commits
3. Manage branch transitions and changes
4. Stash changes and verify

Step 1: Set up the environment

- 1.1 Open the Ubuntu terminal from your practice lab and create a directory using the following command:

mkdir stashdemo

```
poojahksimplile@ip-172-31-79-37:~$ mkdir stashdemo
```

- 1.2 Run the following command to navigate to **stashdemo** :

cd stashdemo/

```
poojahksimplile@ip-172-31-79-37:~$ cd stashdemo/  
poojahksimplile@ip-172-31-79-37:~/stashdemo$ █
```

- 1.3 Configure Git using the following commands:

git config --global user.email "user_email_ID"

git config --global user.name "user_name"

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git config --global user.email "poojahksimplile@gmail.com"  
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git config --global user.name "poojahksimplile"
```

Note: Replace **user_email_ID** with your Git email ID and **user_name** with your GitHub username

Step 2: Initialize Git and make initial commits

2.1 Initialize the Git using the following command:

git init

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git init
Reinitialized existing Git repository in /home/poojahksimplile/stashdemo/.git/
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

2.2 Create a file using the following command:

echo "">index.txt

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ echo "">index.txt
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

2.3 Run the following command to stage the changes in the Git repository:

git add .

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git add .
poojahksimplile@ip-172-31-79-37:~/stashdemo$ █
```

2.4 Commit the staged changes using the following command:

git commit -m "created index.txt"

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git commit -m "created index.txt"
[master (root-commit) c4c267c] created index.txt
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 index.txt
poojahksimplile@ip-172-31-79-37:~/stashdemo$ █
```

2.5 Run the following command to create and switch to the **feature** branch:

git checkout -b feature

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git checkout -b feature
Switched to a new branch 'feature'
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

Step 3: Manage branch transitions and changes

3.1 Run the following command to list the files:

ls

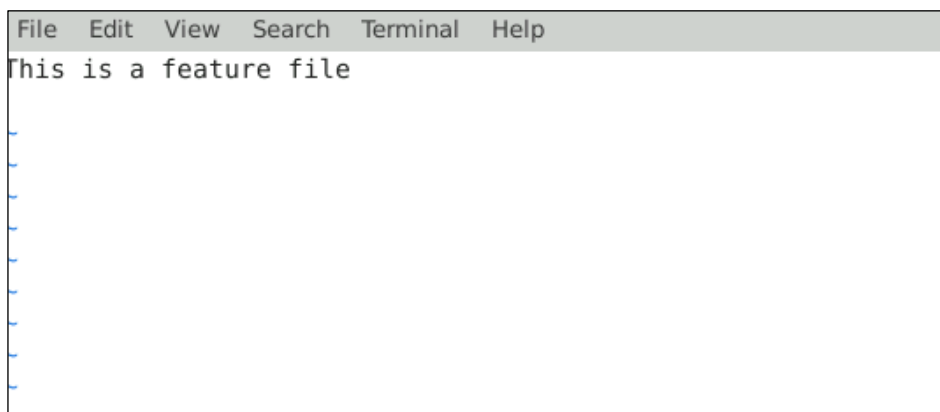
```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ ls
index.txt
poojahksimplile@ip-172-31-79-37:~/stashdemo$ █
```

3.2 Run the following command to create a new file:

vi feature.txt

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ vi feature.txt
poojahksimplile@ip-172-31-79-37:~/stashdemo$ █
```

3.3 Enter the desired data inside the **feature.txt** file as shown in the screenshot below:



3.4 Run the following command to display the current state (changed, staged, and untracked) of the working directory files:

git status

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git status
On branch feature
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    feature.txt

nothing added to commit but untracked files present (use "git add" to track)
poojahksimplile@ip-172-31-79-37:~/stashdemo$ █
```

3.5 Run the following command to stage the changes in the Git repository:

git add .

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git add .
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

3.6 Commit the staged changes using the following command:

git commit -m "feature file WIP"

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git commit -m "feature file WIP"
[feature d83c6b9] feature file WIP
1 file changed, 2 insertions(+)
create mode 100644 feature.txt
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

3.7 Switch to the master branch using the following command:

git checkout master

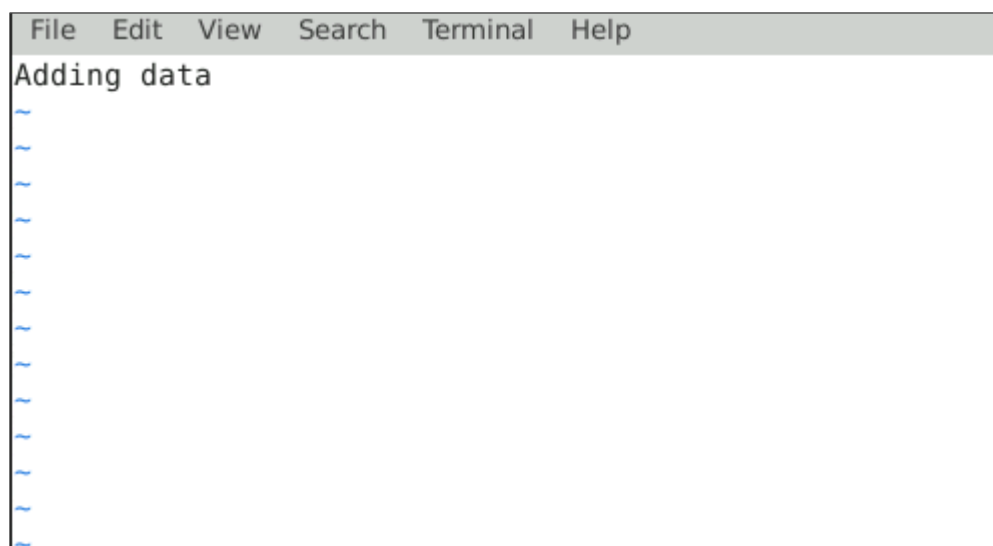
```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git checkout master
Switched to branch 'master'
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

3.8 Navigate inside the index.txt using the following command:

vi index.txt

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ vi index.txt
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

3.9 Add the desired data inside the index.txt file as shown in the screenshot below:



3.10 Run the following command to stage the changes in the Git repository:

git add .

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git add .
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

3.11 Commit the staged changes using the following command:

git commit -m "index file add"

```
poojahksimplile@ip-172-31-29-173:~/stashdemo$ git commit -m "index file add"
[master cacacc2] index file add
1 file changed, 1 insertion(+), 1 deletion(-)
```

3.12 Switch to the feature branch using the following command:

git checkout feature

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git checkout feature
Switched to branch 'feature'
poojahksimplile@ip-172-31-79-37:~/stashdemo$ █
```

3.13 Run the following command to display the commit history of the current branch showing a list of commits with details:

git log

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git log
commit d83c6b92faa9ac03e38a21c1b21eb63c45ec1a72 (HEAD -> feature)
Author: poojahkgowda10 <poojahkgowda101299@gmail.com>
Date: Mon May 6 07:31:03 2024 +0000

    feature file WIP

commit c4c267c2e5559ee11fac3b835aff4be82906cc40
Author: poojahkgowda10 <poojahkgowda101299@gmail.com>
Date: Mon May 6 07:23:57 2024 +0000

    created index.txt
poojahksimplile@ip-172-31-79-37:~/stashdemo$ █
```

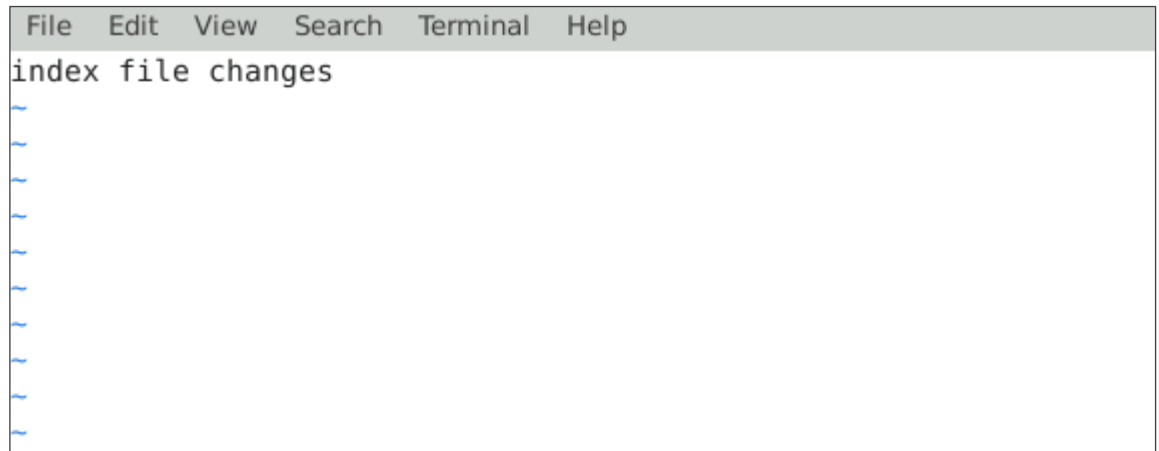
In the above output, observe that the commit of the feature.txt and index.txt files are successful.

3.14 Open index.txt from the **feature** branch (current branch) using the following command:

vi index.txt

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ vi index.txt
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

3.15 Enter the desired data inside the **index.txt** file as shown in the screenshot below:

A screenshot of a text editor window. The title bar shows 'File Edit View Search Terminal Help'. The text area contains the text 'index file changes' followed by ten blue tilde (~) characters on separate lines.

```
File Edit View Search Terminal Help
index file changes
~
~
~
~
~
~
~
~
~
~
```

3.16 Run the following command to display the current state of the working directory files:

git status

A screenshot of a terminal window. The prompt is 'poojahksimplile@ip-172-31-79-37:~/stashdemo\$'. The command 'git status' has been executed. The output shows the current branch is 'feature', that there are changes not staged for commit, and that the file 'index.txt' has been modified. It also provides instructions on how to stage changes or discard them. The prompt returns to the shell.

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git status
On branch feature
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:   index.txt

no changes added to commit (use "git add" and/or "git commit -a")
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

In the above screenshot, observe that the **index.txt** file has been modified from the **feature** branch.

Step 4: Stash changes and verify

4.1 Run the following command to switch to the master branch:

git checkout master

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git checkout master
error: Your local changes to the following files would be overwritten by checkout:
      index.txt
Please commit your changes or stash them before you switch branches.
Aborting
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

Note: The error message indicates that switching to the master branch would overwrite local changes made to index.txt. Thus, you must either commit or stash these changes before changing branches.

4.2 Run the following command to temporarily save changes:

git stash

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git stash
Saved working directory and index state WIP on feature: d83c6b9 feature file WIP
poojahksimplile@ip-172-31-79-37:~/stashdemo$
```

4.3 Run the following command to list all stashed changes:

git stash list

```
poojahksimplile@ip-172-31-79-37:~/stashdemo$ git stash list
stash@{0}: WIP on feature: d83c6b9 feature file WIP
poojahksimplile@ip-172-31-79-37:~/stashdemo$ █
```

4.4 Switch back to the **master** branch using the below command:

git checkout master

```
poojahksimplile@ip-172-31-29-173:~/stashdemo$ git checkout master
Switched to branch 'master'
```

4.5 Apply the stashed changes to the master branch using the following command:

git stash apply

```
poojahksimplile@ip-172-31-29-173:~/stashdemo$ git stash apply
Auto-merging index.txt
CONFLICT (content): Merge conflict in index.txt
On branch master
Unmerged paths:
  (use "git restore --staged <file>..." to unstage)
  (use "git add <file>..." to mark resolution)
      both modified:   index.txt
no changes added to commit (use "git add" and/or "git commit -a")
```

4.6 Execute the following command to check the status of the applied changes:

git status

```
poojahksimplile@ip-172-31-29-173:~/stashdemo$ git status
On branch master
Unmerged paths:
  (use "git restore --staged <file>..." to unstage)
  (use "git add <file>..." to mark resolution)
        both modified:   index.txt

no changes added to commit (use "git add" and/or "git commit -a")
poojahksimplile@ip-172-31-29-173:~/stashdemo$
```

This shows the status of the Git repository on the master branch with an unmerged path for the file **index.txt**, which is both modified and needs resolution.

4.7 Open the **index.txt** file in any text editor

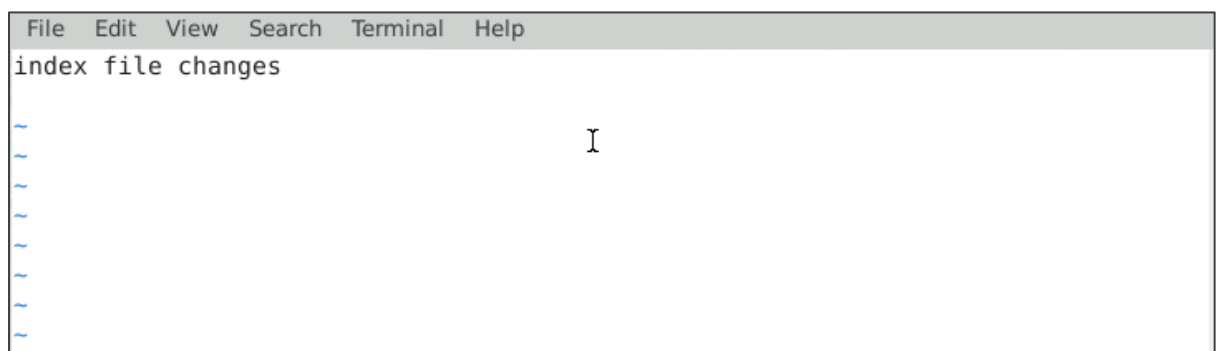


The screenshot shows a text editor window with a menu bar (File, Edit, View, Search, Terminal, Help). The content of the file is as follows:

```
<<<<<<< Updated upstream
Adding data
=====
index file changes
>>>>>>> Stashed changes
~
~
~
~
~
~
~
```

The cursor is positioned on the line ">>>>>>> Stashed changes".

4.8 Remove the conflict markers (<<<<<<<, =====, >>>>>>>) and decide which changes to keep or how to merge them



The screenshot shows the same text editor window after removing the conflict markers. The content is now:

```
index file changes
~
~
~
~
~
~
~
```

The cursor is positioned on the second tilde (~) line.

4.9 Again, execute the **git status** to check the state of the applied stash changes

```
poojahksimplile@ip-172-31-29-173:~/stashdemo$ git status
On branch master
Unmerged paths:
  (use "git restore --staged <file>..." to unstage)
  (use "git add <file>..." to mark resolution)
        both modified:   index.txt

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

4.10 Execute the following commands to complete the merge process:

git add index.txt

git commit -m "Resolved merge conflict in index.txt"

```
poojahksimplile@ip-172-31-29-173:~/stashdemo$ git add index.txt
poojahksimplile@ip-172-31-29-173:~/stashdemo$ git commit -m "Resolved merge conflict in index.txt"
[master 1694098] Resolved merge conflict in index.txt
1 file changed, 2 insertions(+), 1 deletion(-)
```

4.11 To remove the stashed changes after applying, use the following pop command:

git stash pop

```
poojahksimplile@ip-172-31-29-173:~/stashdemo$ git stash pop
On branch master
nothing to commit, working tree clean
Dropped refs/stash@{0} (222cf73d9e42f59cec2336dd7b2a508003eae94f)
```

4.12 Execute the **git stash list** to verify that the stash changes are successfully removed

```
poojahksimplile@ip-172-31-29-173:~/stashdemo$ git stash list
poojahksimplile@ip-172-31-29-173:~/stashdemo$
```

The empty stash list represents that the applied stash is successfully popped.

By following these steps, you have successfully performed stashing in Git to save uncommitted changes temporarily.