VIT Vidyalankar Institute of Technology ACCREDITED A+ BY NAAC	**Department of Information Technology**

Semester	B.E. Semester VIII – INFT
Subject	DevOps Lab
Subject Professor In -charge	Prof. Rohit Barve

Student Name	Nupur Sawarkar	
Roll Number	18101B0030	
Grade and Subject Teacher's Signature		

Experiment Number	01	
Experiment Title	Version Control on software/website using GIT version control tool.	
Resources / Apparatus Required	Hardware: ● Intel Core i3/i5/i7 Processor with Intel VT-X support. ● 4 GB RAM ● 500 GB Hard disk	Software: Operating systems: Windows
Theory	**Version Control:** Version control, also known as source control, is the practice of tracking and managing changes to software code. Version control systems are software tools that help software teams manage changes to source code over time. As development environments have accelerated, version control systems help software teams work faster and smarter. They are especially useful for DevOps teams since they help them to reduce development time and increase successful deployments.	

GIT:

Git is an open-source distributed version control system. It is designed to handle minor to major projects with high speed and efficiency. It is developed to coordinate the work among the developers. The version control allows us to track and work together with our team members at the same workspace.

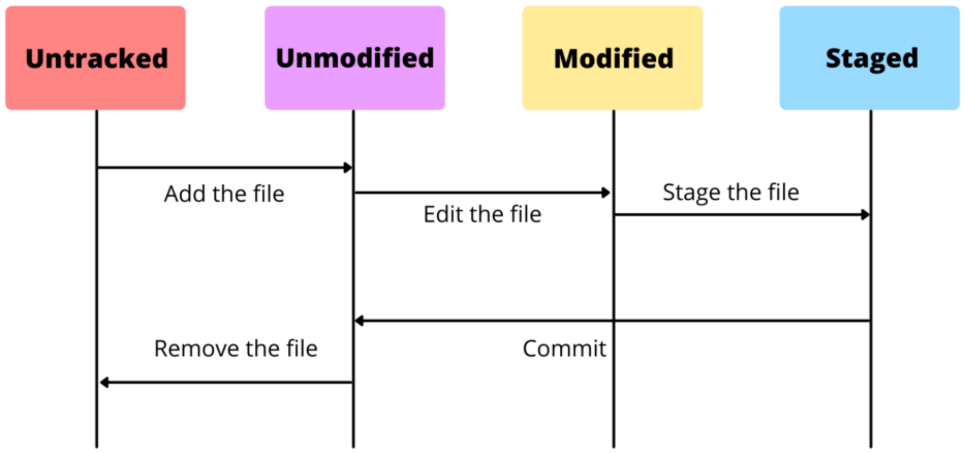
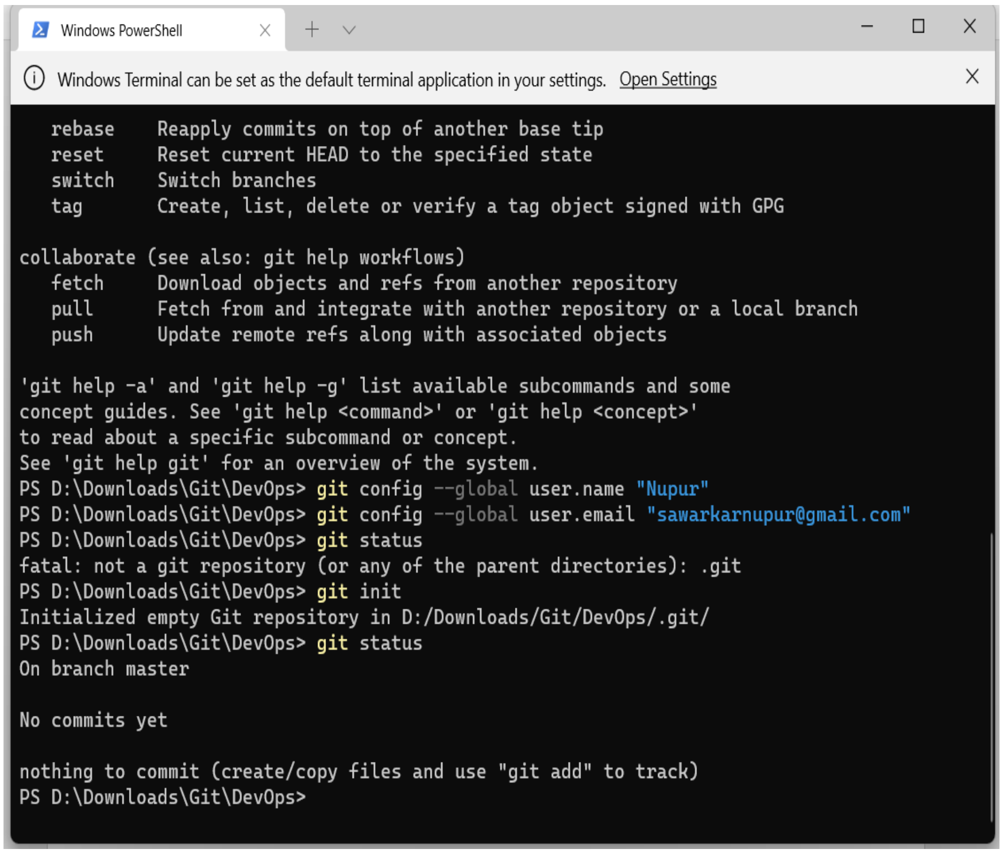


Fig: The Git Lifecycle

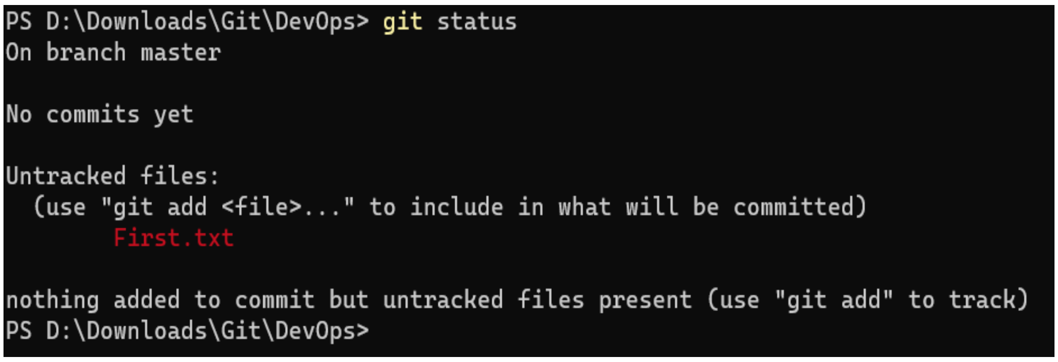
Branching and merging are the great features of Git, which makes it different from the other SCM tools. Git allows the creation of multiple branches without affecting each other. We can perform tasks like creation, deletion, and merging on branches, and these tasks take a few seconds only.

A branch is essentially a 'new' directory, on which you can work on a specific part or feature of a project, before you merge those changes to the main branch that contains all of your source code. The default branch that you always start with is always called the master branch. The master branch contains the most updated, available source code. Always assume that the master branch is ready to be deployed.

Steps	Create a repository and type command **git init** in command to make an existing project as a Git project. The git init command creates a .git subdirectory in the current working directory. It also initializes a HEAD pointer for the master branch of the repository. Then type **git status**. It allows us to see the tracked, untracked files and changes.  Create a text document and then again type git status. We can see the file is untracked. So type **git add filename** to add file contents to the Index (Staging Area).

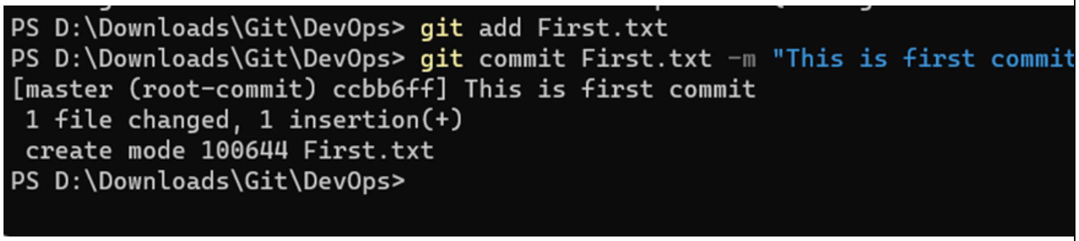
Windows PowerShell
Windows Terminal can be set as the default terminal application in your settings. Open Settings

```
   rebase     Reapply commits on top of another base tip
   reset      Reset current HEAD to the specified state
   switch     Switch branches
   tag        Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
   fetch      Download objects and refs from another repository
   pull       Fetch from and integrate with another repository or a local branch
   push       Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
PS D:\Downloads\Git\DevOps> git config --global user.name "Nupur"
PS D:\Downloads\Git\DevOps> git config --global user.email "sawarkarnupur@gmail.com"
PS D:\Downloads\Git\DevOps> git status
fatal: not a git repository (or any of the parent directories): .git
PS D:\Downloads\Git\DevOps> git init
Initialized empty Git repository in D:/Downloads/Git/DevOps/.git/
PS D:\Downloads\Git\DevOps> git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
PS D:\Downloads\Git\DevOps>
```

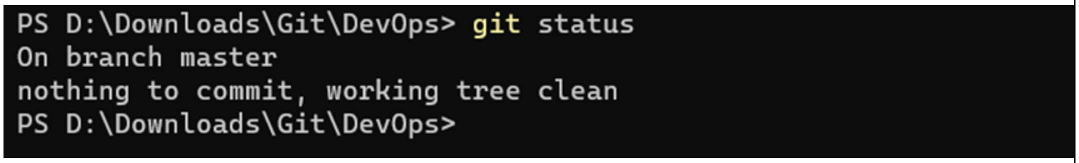


```
PS D:\Downloads\Git\DevOps> git status
On branch master

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
PS D:\Downloads\Git\DevOps>
```

After that type **git commit** to record the changes in the repository. Every commit contains the index data and the commit message.

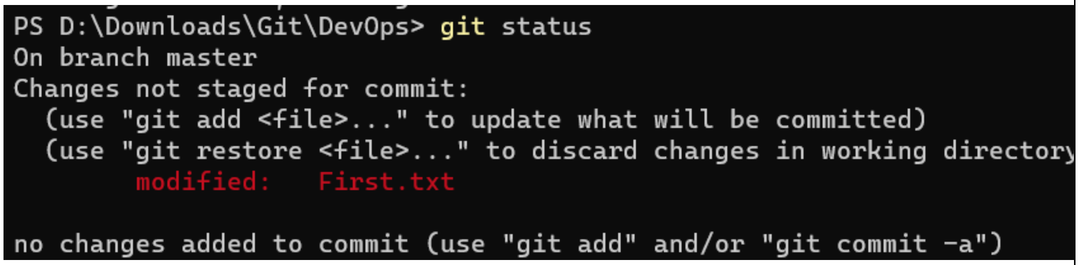


```
PS D:\Downloads\Git\DevOps> git add First.txt
PS D:\Downloads\Git\DevOps> git commit First.txt -m "This is first commit
[master (root-commit) ccbb6ff] This is first commit
 1 file changed, 1 insertion(+)
 create mode 100644 First.txt
PS D:\Downloads\Git\DevOps>
```

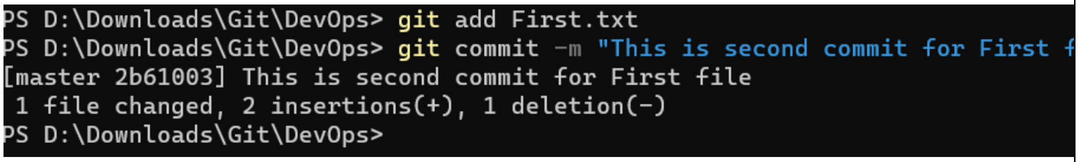


```
PS D:\Downloads\Git\DevOps> git status
On branch master
nothing to commit, working tree clean
PS D:\Downloads\Git\DevOps>
```

Now modify the text file and save it and type git status. We come to know that something is modified.

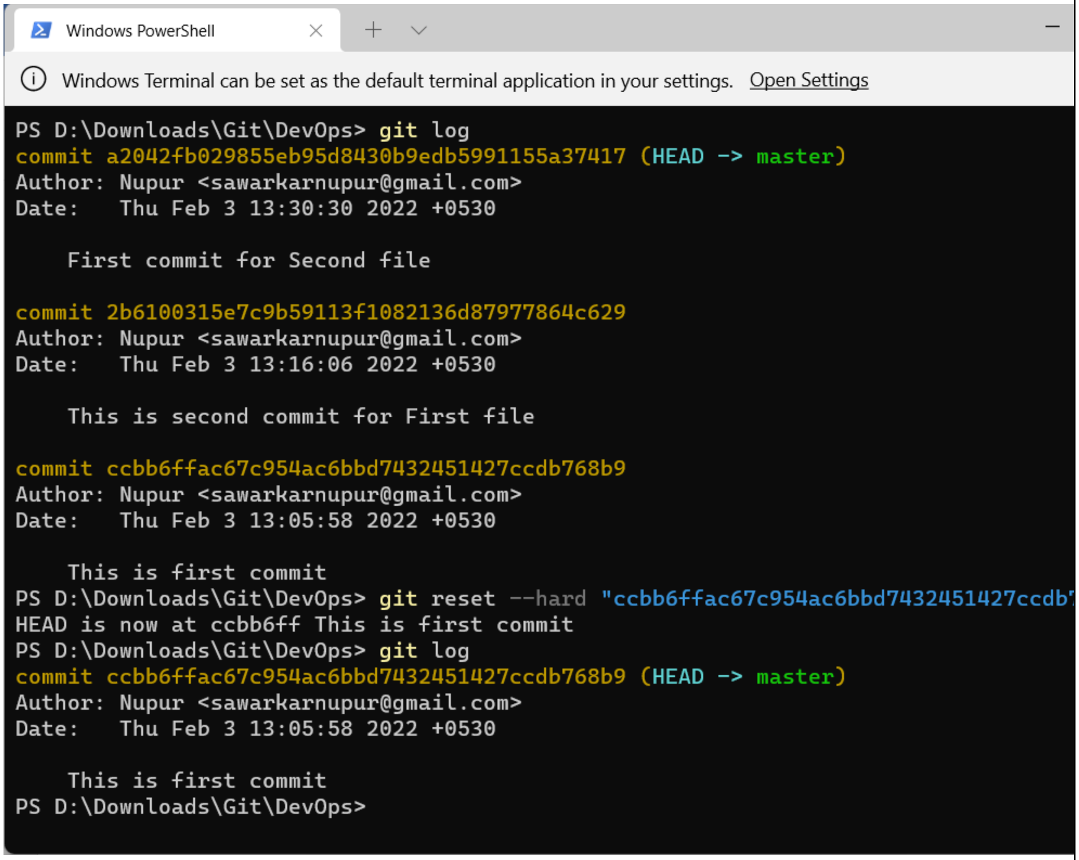


```
PS D:\Downloads\Git\DevOps> git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory
        modified:   First.txt

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



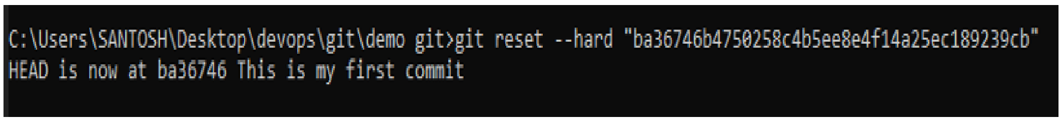
```
PS D:\Downloads\Git\DevOps> git add First.txt
PS D:\Downloads\Git\DevOps> git commit -m "This is second commit for First f
[master 2b61003] This is second commit for First file
 1 file changed, 2 insertions(+), 1 deletion(-)
PS D:\Downloads\Git\DevOps>
```

To find out what is modified and when we use **git log**.


Windows PowerShell
Windows Terminal can be set as the default terminal application in your settings.
Open Settings

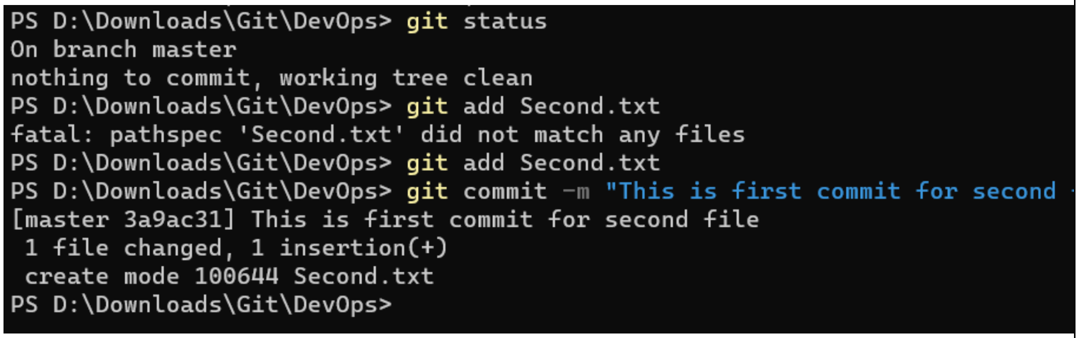
```
PS D:\Downloads\Git\DevOps> git log
commit a2042fb029855eb95d8430b9edb5991155a37417 (HEAD -> master)
Author: Nupur <sawarkarnupur@gmail.com>
Date:   Thu Feb 3 13:30:30 2022 +0530

    First commit for Second file

commit 2b6100315e7c9b59113f1082136d87977864c629
Author: Nupur <sawarkarnupur@gmail.com>
Date:   Thu Feb 3 13:16:06 2022 +0530

    This is second commit for First file

commit ccbb6ffac67c954ac6bbd7432451427ccdb768b9
Author: Nupur <sawarkarnupur@gmail.com>
Date:   Thu Feb 3 13:05:58 2022 +0530

    This is first commit
PS D:\Downloads\Git\DevOps> git reset --hard "ccbb6ffac67c954ac6bbd7432451427ccdb
HEAD is now at ccbb6ff This is first commit
PS D:\Downloads\Git\DevOps> git log
commit ccbb6ffac67c954ac6bbd7432451427ccdb768b9 (HEAD -> master)
Author: Nupur <sawarkarnupur@gmail.com>
Date:   Thu Feb 3 13:05:58 2022 +0530

    This is first commit
PS D:\Downloads\Git\DevOps>
```

Git hard reset is used for undoing changes. The --hard option changes the Commit History, and ref pointers are updated to the specific commit. It means any awaiting work will be lost.



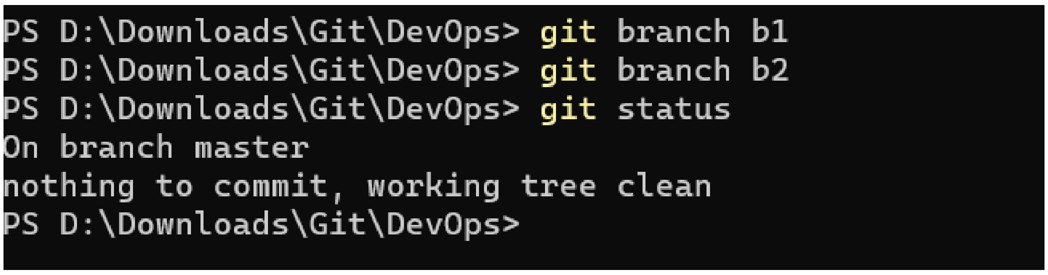
```
C:\Users\SANTOSH\Desktop\devops\git\demo git>git reset --hard "ba36746b4750258c4b5ee8e4f14a25ec189239cb"
HEAD is now at ba36746 This is my first commit
```

Create another file Second.txt.



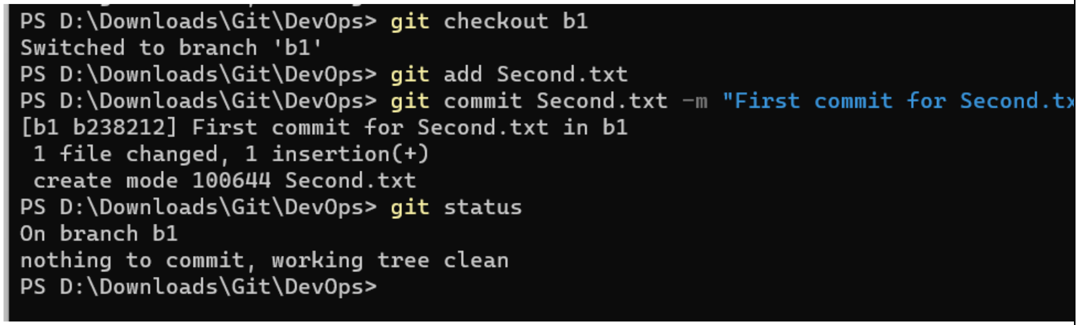
```
PS D:\Downloads\Git\DevOps> git status
On branch master
nothing to commit, working tree clean
PS D:\Downloads\Git\DevOps> git add Second.txt
fatal: pathspec 'Second.txt' did not match any files
PS D:\Downloads\Git\DevOps> git add Second.txt
PS D:\Downloads\Git\DevOps> git commit -m "This is first commit for second
[master 3a9ac31] This is first commit for second file
 1 file changed, 1 insertion(+)
 create mode 100644 Second.txt
PS D:\Downloads\Git\DevOps>
```

Now create two branches b1 and b2 using **git branch** command. And then type **git checkout** command which is used to switch between branches in a repository and go to b1 and add first file in b1.



```
PS D:\Downloads\Git\DevOps> git branch b1
PS D:\Downloads\Git\DevOps> git branch b2
PS D:\Downloads\Git\DevOps> git status
On branch master
nothing to commit, working tree clean
PS D:\Downloads\Git\DevOps>
```

Now go to branch b2 and then add the modified Second.txt and Third.txt in b2.



```
PS D:\Downloads\Git\DevOps> git checkout b1
Switched to branch 'b1'
PS D:\Downloads\Git\DevOps> git add Second.txt
PS D:\Downloads\Git\DevOps> git commit Second.txt -m "First commit for Second.tx
[b1 b238212] First commit for Second.txt in b1
 1 file changed, 1 insertion(+)
 create mode 100644 Second.txt
PS D:\Downloads\Git\DevOps> git status
On branch b1
nothing to commit, working tree clean
PS D:\Downloads\Git\DevOps>
```

Conclusion:	We came to know about how to use different commands of GIT version-control tool. Version control is "a system that records changes to a file or set of files over time so that we can recall specific versions later."