Experiment -3.2

Student Name: Km Ayushi **UID:22BDO10055**
Branch: CSE(DevOps) **Section/Group-22BCD-1(B)**
Semester: 4th **Date of Performance:5/04/24**
Subject Name- Git and GitHub **Subject Code:22CSH-293**

1. Aim/Overview of the practical: Understanding the various reset modes in GitHub.
2. Software used: Git bash and GitHub
3. Steps for experiment/practical:

- First open GitHub, and clone any repository.
- Open a file named f1,f2,f3 and then add and commit it.
- Check the status and then check the git log history.
- Now use the git reset –soft command and check the status of the file.

```
ayush@LAPTOP-14JQ3IMU MINGW64 ~ (main)
$ git clone https://github.com/Ayushi121104/exp9.git
Cloning into 'exp9'...
warning: You appear to have cloned an empty repository.

ayush@LAPTOP-14JQ3IMU MINGW64 ~ (main)
$ git clone https://github.com/Ayushi121104/exp9.git
fatal: destination path 'exp9' already exists and is not an empty directory.

ayush@LAPTOP-14JQ3IMU MINGW64 ~ (main)
$ cd exp9

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git status
On branch main

No commits yet

nothing to commit (create/copy files and use "git add" to track)

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ vi f1

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ gid add f1
bash: gid: command not found

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git add f1
warning: in the working copy of 'f1', LF will be replaced by CRLF the next time Git touches it

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git commit -m"f1 created"
[main (root-commit) 666bcec] f1 created
 1 file changed, 1 insertion(+)
 create mode 100644 f1
```

```
ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ vi f2

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git add f2
warning: in the working copy of 'f2', LF will be replaced by CRLF the next time Git to

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git commit -m"f2 created"
[main bac41b3] f2 created
 1 file changed, 1 insertion(+)
 create mode 100644 f2

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git status
On branch main
Your branch is based on 'origin/main', but the upstream is gone.
  (use "git branch --unset-upstream" to fixup)

nothing to commit, working tree clean

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git log
commit bac41b325fc5d507ee7e611b9254d403c9dffaa3 (HEAD -> main)
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:57 2024 +0530

    f2 created

commit 666bcec8e13fad28651ae231a704cc6a5edff383
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:22 2024 +0530

    f1 created
```

MINGW64:/c/Users/Ayush/exp9

```
ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ vi f3

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git add f3
warning: in the working copy of 'f3', LF will be replaced by CRLF the

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git commit -m"f3 created"
[main b9adc93] f3 created
 1 file changed, 1 insertion(+)
 create mode 100644 f3

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git log
commit b9adc93df6e9026a1fd1fb5bfcbaf098d095999c (HEAD -> main)
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:21:39 2024 +0530

    f3 created

commit bac41b325fc5d507ee7e611b9254d403c9dffaa3
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:57 2024 +0530

    f2 created

commit 666bcec8e13fad28651ae231a704cc6a5edff383
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:22 2024 +0530

    f1 created
```

- The last commit has been reversed based on the current status check.

```
ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git reset --soft HEAD~1

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git log
commit bac41b325fc5d507ee7e611b9254d403c9dffaa3 (HEAD -> main)
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:57 2024 +0530

    f2 created

commit 666bcec8e13fad28651ae231a704cc6a5edff383
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:22 2024 +0530

    f1 created

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ vi f1

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git add f1
warning: in the working copy of 'f1', LF will be replaced by CRL

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git commit -m"f1 modified"
[main 182f1f9] f1 modified
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 f3

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git log
commit 182f1f9f6d672f4cb41de23be3518caa5b00008f (HEAD -> main)
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:22:57 2024 +0530

    f1 modified
```

- But the file is still present in the staging area.
- Now use the git reset –mixed command and check the status of the file.
- F3 is upstaged from the staging area.
- We can easily check the status by using the git log command.
- Now use the git reset –hard command and check the status of the file.
- The f2 is removed from the staging area and is deleted.
- We can easily check the status by using the git log command.

MINGW64:/c/Users/Ayush/exp9

```
ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git reset --mixed HEAD~1
Unstaged changes after reset:
M       f1

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ ls
f1  f2  f3

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git status
On branch main
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:   f1

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        f3

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

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git log
commit bac41b325fc5d507ee7e611b9254d403c9dffaa3 (HEAD -> main)
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:57 2024 +0530

    f2 created

commit 666bcec8e13fad28651ae231a704cc6a5edff383
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:22 2024 +0530

    f1 created
```

```
ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git log
commit bac41b325fc5d507ee7e611b9254d403c9dffaa3 (HEAD -> main)
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:57 2024 +0530

    f2 created

commit 666bcec8e13fad28651ae231a704cc6a5edff383
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:22 2024 +0530

    f1 created

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git reset --hard HEAD~1
HEAD is now at 666bcec f1 created

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ ls
f1  f3

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git status
On branch main
Untracked files:
  (use "git add <file>..." to include in what will be committed)
        f3

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

ayush@LAPTOP-14JQ3IMU MINGW64 ~/exp9 (main)
$ git log
commit 666bcec8e13fad28651ae231a704cc6a5edff383 (HEAD -> main)
Author: Ayushi <ayushi121104@gmail.com>
Date:   Thu Apr 11 15:19:22 2024 +0530

    f1 created
```

4. Result/Output/Writing Summary:

In this experiment, we have learnt about the different types of reset commands used in git, how they impact the files that have already been added, and how they help us to undo the changes that we have already made.

Learning outcomes (What I have learnt):

1. Learn about the reset command and its types
2. Learn about the effect of the soft reset command.
3. Learn about the effect of mixed and hard reset commands.
4. Learn how to check git history.
5. Learn how to undo previous changes that we have made.

Evaluation Grid (To be created as per the SOP and Assessment guidelines by the faculty):

Sr. No.	Parameters	Marks Obtained	Maximum Marks
1.			
2.			
3.			