Name: Raj Singh

T22-105

EXPERIMENT – 3

AIM - To Perform various GIT operations on local and Remote repositories using GIT Cheat-Sheet

Theory:

Git is a distributed version control system that allows developers to track changes, collaborate, and manage source code efficiently. Git provides numerous commands to handle local and remote repositories.

1. Setting Up Git

Before performing Git operations, configure Git with your details:

git config --global user.name "Your Name"

git config --global user.email "your.email@example.com" Verify the configuration:

git config --list

2. Initializing a Git Repository To create a new Git repository: git init

This initializes a new repository in the current directory.

3. Cloning a Repository To clone a remote repository: git clone <repository_url> Example:

git clone https://github.com/your-username/repository.git

4. Staging and Committing Changes
   - To check the status of the working directory:
   - git status
   - To add files to the staging area:
   - git add <file_name> or to add all changes:

git add .

- To commit changes with a message:
- git commit -m "Your commit message" 5. Viewing Commit History To view commit logs:

git log

For a compact version: git
log --oneline

6. Branching in Git

- To create a new branch:
- git branch <branch_name>
- To switch to another branch: · git checkout <branch_name>
- To create and switch to a new branch simultaneously:
- git checkout -b <branch_name> · To view all branches:
- git branch

7. Merging Branches

- First, switch to the main branch:
- git checkout main
- Merge a branch into the main branch:
- git merge <branch_name> 8. Pushing Changes to Remote Repository · To push changes to GitHub: · git push origin <branch_name> · If pushing for the first time:
- git push --set-upstream origin <branch_name>

9. Pulling Changes from Remote Repository To fetch
and merge changes from a remote repository:

git pull origin <branch_name> 10.
Handling Merge Conflicts If a
merge conflict occurs:

1. Open conflicting files and resolve issues manually.
2. Add resolved files to the staging area:
3. git add <file_name>
4. Commit the resolved changes:
5. git commit -m "Resolved merge conflict"

11. Undoing Changes

- To undo changes before staging:
- git checkout -- <file_name> · To unstage a file:
- git reset HEAD <file_name>

- To revert the last commit:
- git revert HEAD

12. Deleting a Branch

- To delete a local branch: · git branch -d <branch_name> · To delete a remote branch:
- git push origin --delete <branch_name>

13. Creating and Using a .gitignore File

A .gitignore file is used to ignore specific files or directories:

```
echo "node_modules/" >> .gitignore
git add .gitignore git commit -m
"Added .gitignore file"
```

14. Checking Differences in Files

- To compare working directory changes:
- git diff
- To compare staged changes:
- git diff --staged

15. Stashing Changes

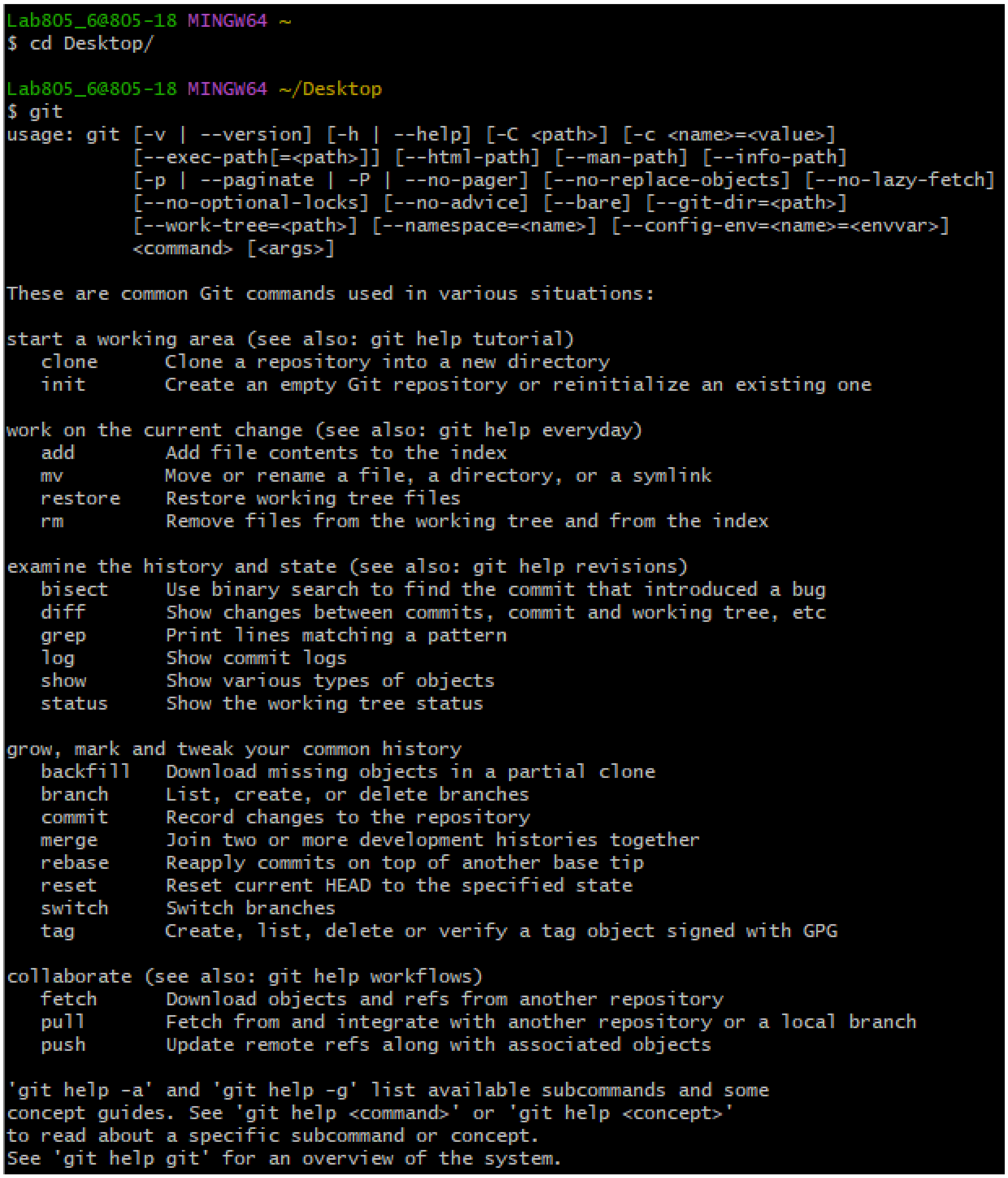
To temporarily save uncommitted changes:

```
git stash
```

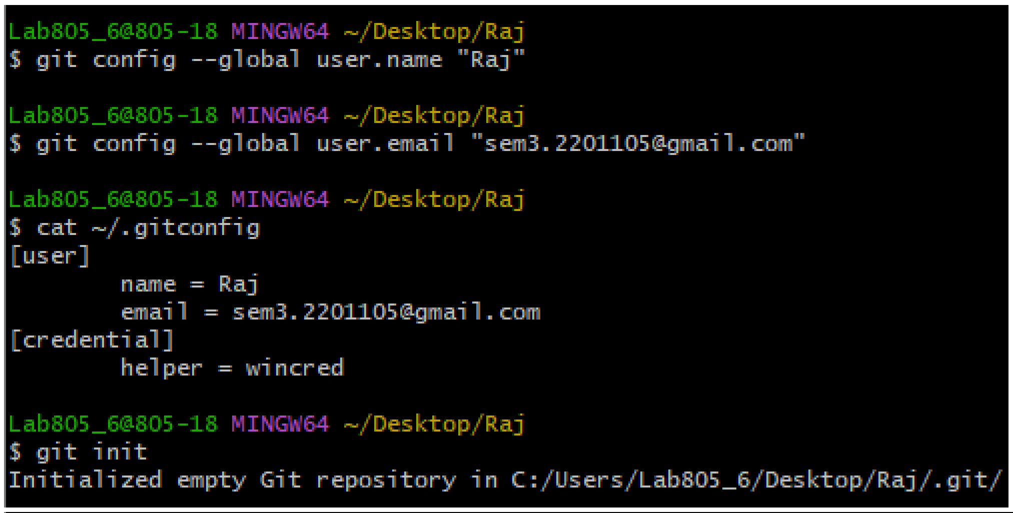
To apply the stashed changes:

```
git stash apply
```

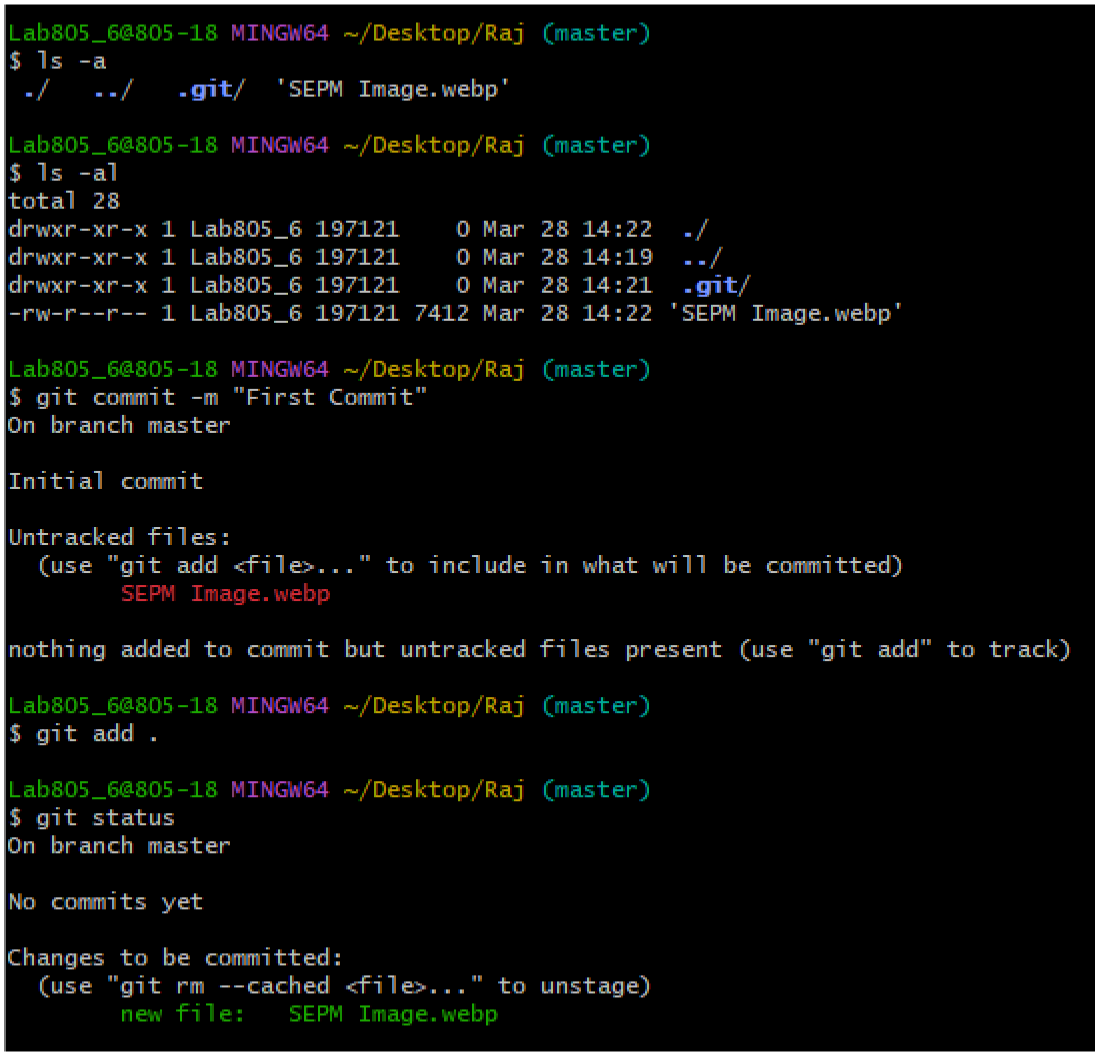
Output:



```
Lab805_6@805-18 MINGW64 ~
$ cd Desktop/

Lab805_6@805-18 MINGW64 ~/Desktop
$ git
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--no-lazy-fetch]
           [--no-optional-locks] [--no-advice] [--bare] [--git-dir=<path>]
           [--work-tree=<path>] [--namespace=<name>] [--config-env=<name>=<envvar>]
           <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone      Clone a repository into a new directory
   init       Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add        Add file contents to the index
   mv         Move or rename a file, a directory, or a symlink
   restore    Restore working tree files
   rm         Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect     Use binary search to find the commit that introduced a bug
   diff       Show changes between commits, commit and working tree, etc
   grep       Print lines matching a pattern
   log        Show commit logs
   show       Show various types of objects
   status     Show the working tree status

grow, mark and tweak your common history
   backfill   Download missing objects in a partial clone
   branch     List, create, or delete branches
   commit     Record changes to the repository
   merge      Join two or more development histories together
   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.
```



```
Lab805_6@805-18 MINGW64 ~/Desktop/Raj
$ git config --global user.name "Raj"

Lab805_6@805-18 MINGW64 ~/Desktop/Raj
$ git config --global user.email "sem3.2201105@gmail.com"

Lab805_6@805-18 MINGW64 ~/Desktop/Raj
$ cat ~/.gitconfig
[user]
        name = Raj
        email = sem3.2201105@gmail.com
[credential]
        helper = wincred

Lab805_6@805-18 MINGW64 ~/Desktop/Raj
$ git init
Initialized empty Git repository in C:/Users/Lab805_6/Desktop/Raj/.git/
```



```
Lab805_6@805-18 MINGW64 ~/Desktop/Raj (master)
$ ls -a
 ./   ../   .git/  'SEPM Image.webp'

Lab805_6@805-18 MINGW64 ~/Desktop/Raj (master)
$ ls -al
total 28
drwxr-xr-x 1 Lab805_6 197121    0 Mar 28 14:22  ./
drwxr-xr-x 1 Lab805_6 197121    0 Mar 28 14:19  ../
drwxr-xr-x 1 Lab805_6 197121    0 Mar 28 14:21  .git/
-rw-r--r-- 1 Lab805_6 197121 7412 Mar 28 14:22 'SEPM Image.webp'

Lab805_6@805-18 MINGW64 ~/Desktop/Raj (master)
$ git commit -m "First Commit"
On branch master

Initial commit

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

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

Lab805_6@805-18 MINGW64 ~/Desktop/Raj (master)
$ git add .

Lab805_6@805-18 MINGW64 ~/Desktop/Raj (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   SEPM Image.webp
```

CONCLUSION:

Thus, we have successfully studied and performed various GIT operations on local and Remote repositories using GIT Cheat-Sheet.