HEAVEN'S LIGHT IS OUR GUIDE

RAJSHAHI UNIVERSITY OF ENGINEERING AND TECHNOLOGY

COURSE TITLE: Software Engineering & Information System Design Sessional

COURSE NO: ECE 3118

Lab report Submission

Submission date: 20.10.2024

SUBMITTED TO:

OISHI JYOTI

ASSISTANT PROFESSOR

DEPARTMENT OF ECE,

RUET.

SUBMITTED BY:

PUJA ROY DIPA

2010040

Experiment No.: 03

Experiment Name: Study of different Git commands.

Theory:

Git is a distributed version control system that helps developers manage and track changes to their code. It allows multiple developers to work on the same project simultaneously, without overwriting each other's changes. Some of its key features include:

- **Version control:** Git tracks every change made to a project, making it easy to revert to earlier versions if needed.
- **Branching and merging:** Git allows developers to create multiple branches for different features or bug fixes, which can later be merged into the main codebase.
- **Distributed nature:** Every developer has a full copy of the project history, enabling offline work and better collaboration.
- **Collaboration:** Git enables team collaboration, supporting code reviews, issue tracking, and integration with platforms like GitHub, GitLab, and Bitbucket.

Git is widely used in software development due to its efficiency in managing code changes and collaboration in projects.

Git Commands:

1. **Git config command:** (user name & user email)
   The git config command in Git is used to set or get configuration options that control the behavior of Git.

```
20100@DESKTOP-1OJ31SQ MINGW64 ~
$ git config --global user.name "puja"

20100@DESKTOP-1OJ31SQ MINGW64 ~
$ git config --global user.email "pujaroydipa414@gmail.com"
```

2. **Git init command:**

   The git init command is used to initialize a new Git repository. When we run this command, Git creates a hidden .git directory in your project folder, which is where all the metadata, version history, and configuration for the repository are stored.

```
20100@DESKTOP-1OJ31SQ MINGW64 ~
$ git init
Initialized empty Git repository in C:/Users/20100/.git/
```

3. **Git status command:**

   The git status command is used to display the current state of the working directory and the staging area in Git. It shows which changes have been staged, which haven't, and which files are not being tracked by Git.

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ git status
```

```
No commits yet

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

4. **Touch command:**

   The touch command creates an empty file named index.html

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ touch index.html
```

```
index.html
```

5. **Git Add command:**

   The git add command in Git is used to stage changes (new files, modified files, or deleted files) to be included in the next commit. It moves changes from the working directory to the staging area (also called the index).

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ git add index.html
```

   Checking the status:

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ git status
```

```
No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   index.html

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

6. **Git Commit command:**

The git commit -m command is used to save changes to the local Git repository with a message describing the changes.

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ git commit -m "create a file index.html"
[master (root-commit) 77b7023] create a file index.html
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 index.html
```

7. **Git log command:**

The git log command is used in Git to display the commit history of a repository. It shows a list of commits along with their metadata, such as the author, date, and commit message.

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ git log
commit 77b70237e4ec3a3f3726cd82db73a2cb08b48646 (HEAD -> master)
Author: puja <pujaroydipa414@gmail.com>
Date:   Thu Oct 17 22:36:42 2024 -0700

    create a file index.html
```

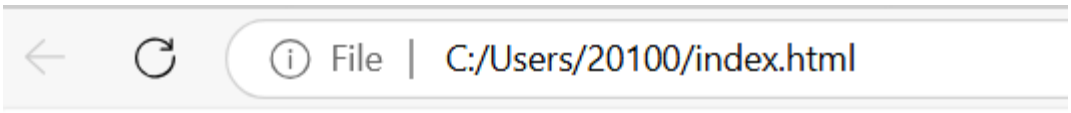
After that I make some change in the index.html file in vs code which is look like:

index.html

C: > Users > 20100 > index.html > ...

```
1   <!DOCTYPE html>
2   <html lang="en">
3   <head>
4       <meta charset="UTF-8">
5       <meta name="viewport" content="width=device-width, initail-scale=1.0">
6       <meta http-equiv="X-UA-Compatible" content="ie=edge">
7       <title>
8           Document
9       </title>
10  </head>
11  <body>
12      <h1>Various Git Command</h1>
13  </body>
14  </html>
```

After that browser is look like:



Various Git Command

Checking the status:

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ 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:   index.html
```

8. **Git ls-files command:**
   This command lists all the files in the current working directory that are tracked by Git. It can be useful to see which files are being tracked without looking through the entire working directory.

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ git ls-files
index.html
```

9. **Git checkout command:**
   The git checkout command is used in Git to switch between branches or to restore working tree files.

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ git checkout 77b70237e4ec3a3f3726cd82db73a2cb08b48646
Note: switching to '77b70237e4ec3a3f3726cd82db73a2cb08b48646'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at 77b7023 create a file index.html
M       index.html
```

10. **Git branch command:**

    The git branch command in Git is used to manage branches in a repository. Branches allow us to create separate lines of development, which is particularly useful for working on features, fixes, or experiments without affecting the main codebase.

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ ((77b7023...))
$ git branch
* (HEAD detached at 77b7023)
  master
```

Adding a new branch:

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ ((77b7023...))
$ git branch index2
```

Switch to branch master:

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ ((77b7023...))
$ git switch -
Switched to branch 'master'
M       index.html
```

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ git branch
  index2
* master
```

Switch to newly created branch (index2):

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (master)
$ git switch index2
Switched to branch 'index2'
M       index.html
```

Rename that branch:

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (index2)
$ git branch -m index2 index_report

20100@DESKTOP-1OJ31SQ MINGW64 ~ (index_report)
$
```

To show all the branches:

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (index_report)
$ git branch -a
  index40
* index_report
  master
```

To delete a branch:

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (index_report)
$ git branch -d index40
Deleted branch index40 (was 77b7023).
```

Now there is one branch with branch master:

```
20100@DESKTOP-1OJ31SQ MINGW64 ~ (index_report)
$ git branch -a
* index_report
  master
```

Reference:

https://videotutorials.notion.site/Introduction-to-Git-ac396a0697704709a12b6a0e545db049