SHRI VILEPARLE KELAVANI MANDAL'S
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC ACCREDITED with "A" GRADE (CGPA : 3.18)

A.Y. 2022 – 23

Software Engineering

Experiment-10

Div: B **Batch:** B1

Team Members:
Meet Patel - 60004200104
Kartik Jolapara - 60004200107

Aim: Study of Configuration Management using GitHub

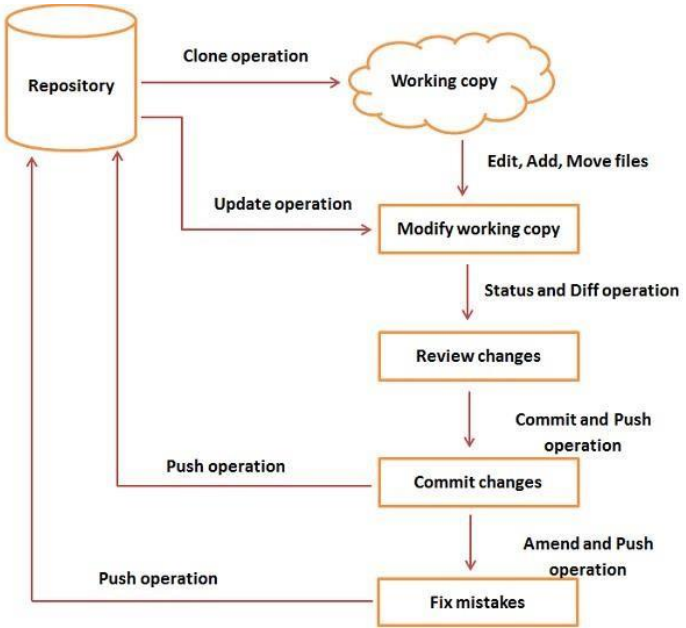
Theory:

Git is a distributed revision control and source code management system with an emphasis on speed. Git was initially designed and developed by Linus Torvalds for Linux kernel development. Git is a free software distributed under the terms of the GNU General Public License version 2.

Git Life Cycle

General workflow is as follows −

1. Clone the Git repository as a working copy.
2. Modify the working copy by adding/editing files.
3. If necessary, update the working copy by taking other developer's changes.
4. Review the changes before commit.
5. Commit changes. If everything is fine, then push the changes to the repository.
6. After committing, if something is wrong, then correct the last commit and push



A.Y. 2022 – 23
Git Life Cycle

1. Creating Git Repository

Initialize a new repository by using **init** command followed by **--bare** option. It initializes the repository without a working directory. By convention, the bare repository must be named as **.git**.

```
[gituser@CentOS ~]$ pwd
/home/gituser

[gituser@CentOS ~]$ mkdir project.git

[gituser@CentOS ~]$ cd project.git/

[gituser@CentOS project.git]$ ls

[gituser@CentOS project.git]$ git --bare init Initialized
empty Git repository in /home/gituserm/project.git/

[gituser@CentOS project.git]$ ls
branches config description HEAD hooks info objects refs
```

```
DJSCE.Student@MUMO922CPUO710 MINGW64 ~
$ pwd
/c/Users/djsce.student

DJSCE.Student@MUMO922CPUO710 MINGW64 ~
$ mkdir project.git

DJSCE.Student@MUMO922CPUO710 MINGW64 ~
$ cd project.git/

DJSCE.Student@MUMO922CPUO710 MINGW64 ~/project.git
$ ls

DJSCE.Student@MUMO922CPUO710 MINGW64 ~/project.git
$ git -- bare init
unknown option: --
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] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--config-env=<name>=<envvar>] <command> [<args>]

DJSCE.Student@MUMO922CPUO710 MINGW64 ~/project.git
$ git --bare init
Initialized empty Git repository in C:/Users/djsce.student/project.git/

DJSCE.Student@MUMO922CPUO710 MINGW64 ~/project.git (BARE:master)
$ ls
HEAD  config  description  hooks/  info/  objects/  refs/

DJSCE.Student@MUMO922CPUO710 MINGW64 ~/project.git (BARE:master)
$ :
```

2. Generate Public-Private RSA Key Pair

```
User1@CentOS ~]$ pwd
/home/user1

[user1@CentOS ~]$ ssh-keygen
```

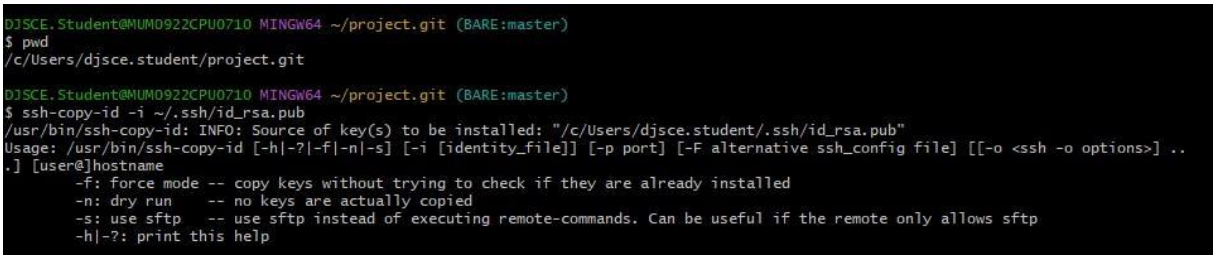
```
DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git (BARE:master)
$ pwd
/c/Users/djsce.student/project.git

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git (BARE:master)
$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/djsce.student/.ssh/id_rsa):
Created directory '/c/Users/djsce.student/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /c/Users/djsce.student/.ssh/id_rsa
Your public key has been saved in /c/Users/djsce.student/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:siF8i6a2HRpCT+lpQR3XMg/E7v49crM76cmBZkDz47E DJSCE.Student@MUM0922CPU0710
The key's randomart image is:
+---[RSA 3072]----+
|     .oo.        |
|    . o= .       |
|   . ..o=        |
|  ... ..o.       |
| . +o =.S+       |
|. + o+ *o =      |
|. .=+ +  E ..    |
| .o* . .o.o*o    |
| .+..    ..+B*   |
+----[SHA256]-----+
```

3. Adding keys to authorized keys

Suppose there are two developers working on a project. Both users have generated public keys.

Both add their public key to the server by using ssh-copy-id command as given below

```
[user1@CentOS ~]$ pwd
/home/user1

[user2@CentOS ~]$ ssh-copy-id -i ~/.ssh/id_rsa.pub
gituser@git.server.com
```



4. Push changes to the repository

We have created a bare repository on the server and allowed access for two users. Both users can push their changes to the repository by adding it as a remote.

Git init command creates **.git** directory to store metadata about the repository every time it reads the configuration from the .**git/config** file.

User1 creates a new directory, adds README file, and commits his change as initial commit. After commit, he verifies the commit message by running the **git log** command.

```
[user1@CentOS ~]$ pwd
/home/user1

[user1@CentOS ~]$ mkdir user1_repo

[user1@CentOS ~]$ cd user1_repo/

[user1@CentOS user1_repo]$ git init Initialized
empty Git repository in
/home/user1/user1_repo/.git/

[user1@CentOS user1_repo]$ echo 'TODO: Add contents for
README' > README

[user1@CentOS user1_repo]$ git status -s
?? README

[user1@CentOS user1_repo]$ git add .

[user1@CentOS user1_repo]$ git status -s
A README

[user1@CentOS user1_repo]$ git commit -m 'Initial commit'
```

5. Checking log message by executing the git log command.

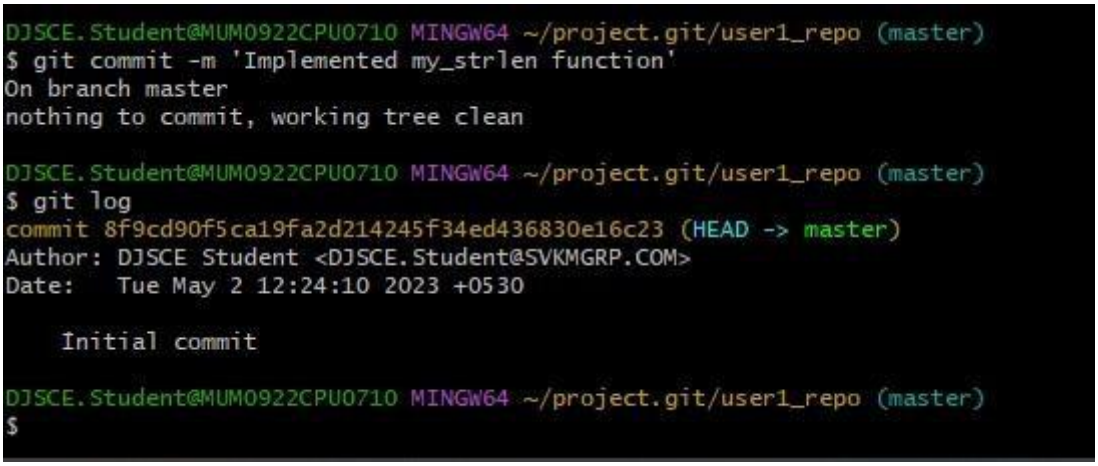
```
[user1@CentOS user1_repo]$ git log
```

```
DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git (BARE:master)
$ pwd
/c/Users/djsce.student/project.git

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git (BARE:master)
$ mkdir user1_repo

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git (BARE:master)
$ cd user1_repo/

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git/user1_repo (BARE:master)
$ git init
Initialized empty Git repository in C:/Users/djsce.student/project.git/user1_repo/.git/

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git/user1_repo (master)
$ echo 'TODO: Add Contents for README'> README

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git/user1_repo (master)
$ git status -s
?? README

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git/user1_repo (master)
$ git add.
git: 'add.' is not a git command. See 'git --help'.

The most similar command is
        add

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git/user1_repo (master)
$ git add .
warning: in the working copy of 'README', LF will be replaced by CRLF the next time Git touches it

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git/user1_repo (master)
$ git status -s
A  README

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git/user1_repo (master)
$ git commit -m 'Initial commit'
[master (root-commit) 8f9cd90] Initial commit
 Committer: DJSCE Student <DJSCE.Student@SVKMGRP.COM>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly. Run the
following command and follow the instructions in your editor to edit
your configuration file:

    git config --global --edit

After doing this, you may fix the identity used for this commit with:

    git commit --amend --reset-author

 1 file changed, 1 insertion(+)
 create mode 100644 README

DJSCE.Student@MUM0922CPU0710 MINGW64 ~/project.git/user1_repo (master)
$ git log
commit 8f9cd90f5ca19fa2d214245f34ed436830e16c23 (HEAD -> master)
Author: DJSCE Student <DJSCE.Student@SVKMGRP.COM>
Date:   Tue May 2 12:24:10 2023 +0530

    Initial commit
```

6. Commit changes

To commit the changes, he used the git commit command followed by –m option. If we omit –m option. Git will open a text editor where we can write multiline commit message

```
[user2@CentOS project]$ git commit -m 'Implemented
my_strlen function'
```



Conclusion:

Git is a version control software that can store different versions on a local machine or can be integrated with remote file management system. We used Git bash cmd to run commands and store the local files on github files management server. We also seen the git log command which stores the the activity log happens on git.