NAME:_____ STD.:_____ DIV.:_____ DATE : PAGE :

Roll No: 67
NAME: HERAMB R.PAWAR
SUB: SEPM

PRACTICAL 2&3

AIM:
a) To understand Version Control System / Source Code Management Install git and Create a Github account.
b) To perform various GIT operations on local and Remote repositories using GIT Cheat-Sheet.

Problem Statement:
To obtain complete knowledge of the "version Control System" to effectively track changes augmented with GIT and GITHUB.

Theory:

Version Control System:

VCS stands for Version Control System, which is a software tool that helps manage and keep track of changes made to the code base of a software project. It allows multiple developers to work on same code, keep track of changes made to the code, and easily revert back to previous version if needed. Popular example of VCS include Git, Mercurial and Subversions.

NAME:_____ STD.:_____ DIV.:_____

DATE :
PAGE :

CHARACTERISTICS OF VCS:

The main characteristics of Version Control System include:

a) Version Tracking:
Vcs Track changes made to the code base and Store different versions of the code, allowing developers to easily access and compare previous version

b) Collabration:
Vcs allow multiple developers to work on the same code base simultaneously and merge their changes into a single code repositary.

c) Branching:
Vcs enables developers to create separate branches of the code base allowing them to work on new feature or bug fixes without affecting the main code

D) Merging:
Vcs makes it possible to merge changes made in different branches into the main code base.

e) Rollback:
Vcs allows developers to revert back to previous version of code if necessary.

NAME:_____ STD.:_____ DIV.:_____ DATE : PAGE :

F) History:
Vcs provides a history of all changes made to the code, making it possible to see who made that changes and when

G) Conflict Resolution:
Vcs provides tools for resolving conflicts that arise when developers make multiple changes to the same part of the code simultaneously

H) Security:
Vcs provides access control mechanism to ensure that only authorized user can make changes to the code base.

GIT:

Git is popular and widely-use distributed version control system (VCS) for Software development. It was created by Linus Torvalds in 2005 to manage the development of the Linux kernel, but has since been adopted by a large number of Software development projects

Git allows multiple developers to work on the same code base simultaneously, keep track of changes made to the code, and easily revert back to previous versions if necessary. Git is known for its speed reliability and efficient handling of large project

With many files.

Git is a distributed VCS, meaning that each developer has a complete copy of code repository on their local machine allowing them to work offline and keep track of changes locally. When they are ready to share their changes they can push them to be central repository where other developer can access and merge them into the main code base.

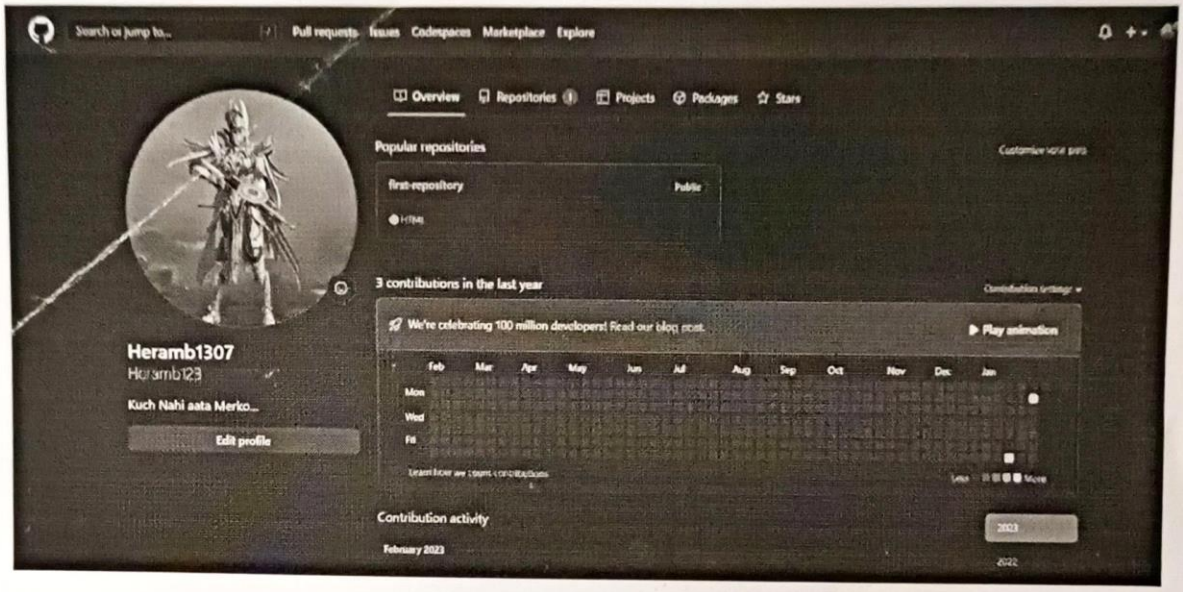
Git also provides features such as branching, merging and conflict resolution, making it a powerful tool for managing complexity of software development project.

Conclusion:
Thus we created our Git Account and we created First GIT Repository and Perfomed various operations on it

Name: Heramb Pawar **Sub:** SEPM **Roll:** 67

Output:

1. GitHub Account:



2. Creating/cloning repository:

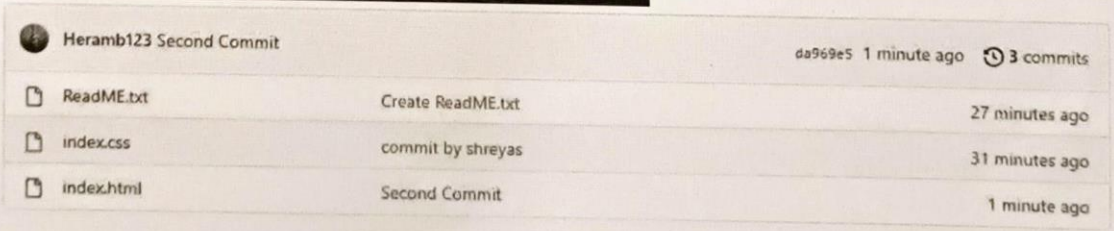
```
user@DESKTOP-Q7AO9SN MINGW64 ~/Desktop
$ git clone https://github.com/Shreyassatre/GitPrac.git
Cloning into 'GitPrac'...
remote: Enumerating objects: 16, done.
remote: Counting objects: 100% (16/16), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 16 (delta 5), reused 8 (delta 2), pack-reused 0
Receiving objects: 100% (16/16), done.
Resolving deltas: 100% (5/5), done.
```

3. Making changes in project and pushing it back:

```
user@DESKTOP-Q7AO9SN MINGW64 ~/Desktop/GitPrac (main)
$ git add .

user@DESKTOP-Q7AO9SN MINGW64 ~/Desktop/GitPrac (main)
$ git commit -m 'added status of complition'
[main 3ac21c4] added status of complition
 2 files changed, 2 insertions(+)
```

```
user@DESKTOP-Q7AO9SN MINGW64 ~/Desktop/GitPrac (main)
$ git push origin main
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 368 bytes | 368.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/Shreyassatre/GitPrac.git
   a67f933..3ac21c4  main -> main
```



4 ■■■■ index.html

```
@@ -7,6 +7,8 @@
    <title>Git Practical</title>
</head>
<body>
-   <h1>Hey teammates</h1>
+   <h1>Hello</h1>
+   <h2>This Is SEPM Practical 1</h2>
+
</body>
</html>
```

4. **Creating new Branch:**

```
user@DESKTOP-Q7AO9SN MINGW64 ~/Desktop/GitPrac (main)
$ git branch test
```

```
user@DESKTOP-Q7AO9SN MINGW64 ~/Desktop/GitPrac (main)
$ git checkout test
Switched to branch 'test'

user@DESKTOP-Q7AO9SN MINGW64 ~/Desktop/GitPrac (test)
$ git branch
  main
* test
```

```
user@DESKTOP-Q7AO9SN MINGW64 ~/Desktop/GitPrac (test)
$ git push origin test
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 299 bytes | 299.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/Shreyassatre/GitPrac.git
   3ac21c4..d5c2418  test -> test
```

5. **Merging created branch with main branch:**

```
user@DESKTOP-Q7AO9SN MINGW64 ~/Desktop/GitPrac (main)
$ git merge test
Updating 3ac21c4..ea24b40
Fast-forward
 index.html | 2 +-
 1 file changed, 1 insertion(+), 1 deletion(-)
```

6. **Deleting a branch:**

```
user@DESKTOP-Q7AO9SN MINGW64 ~/Desktop/GitPrac (main)
$ git branch -d test
Deleted branch test (was ea24b40).
```

7. **Contributors:**