Name of Student: Pushkar Sane		
Roll Number: 45	**Lab Assignment Number:3**	
Title of Lab Assignment: To implement version control using GITHUB to sync local GIT repositories and perform various related operations using GIT Cheat-Sheet.		
DOP: 03-02-2024	**DOS: 05-02-2024**	
CO Mapped: **CO2**	**PO Mapped:** **PO1, PO5, PSO1, PSO2**	**Signature:**

Practical No. 3

Aim: To implement version control using GITHUB to sync local GIT repositories and perform various related operations using GIT Cheat-Sheet.

What is Version Control?

Version control is a system that records changes to a file or set of files over time. It allows you to track and manage modifications to your project, facilitating collaboration among multiple contributors. Version control systems help in maintaining a chronological history of changes, enabling you to revert to previous states, track the evolution of the codebase, and work seamlessly in a team environment.

Introduction to Git:

Git is a distributed version control system designed for tracking changes in source code during software development. Developed by Linus Torvalds, the creator of Linux, Git offers a powerful and flexible platform for managing project versions. It allows developers to work collaboratively, keeping track of code modifications, merging changes from multiple contributors, and maintaining a consistent and reliable project history.

Understanding GitHub:

GitHub is a web-based platform that provides hosting for software development using Git. It acts as a centralized repository where developers can store their projects, collaborate with others, and manage version control. GitHub enhances the collaborative nature of Git by offering features such as issue tracking, pull requests, and a graphical interface for easier navigation and interaction with repositories.

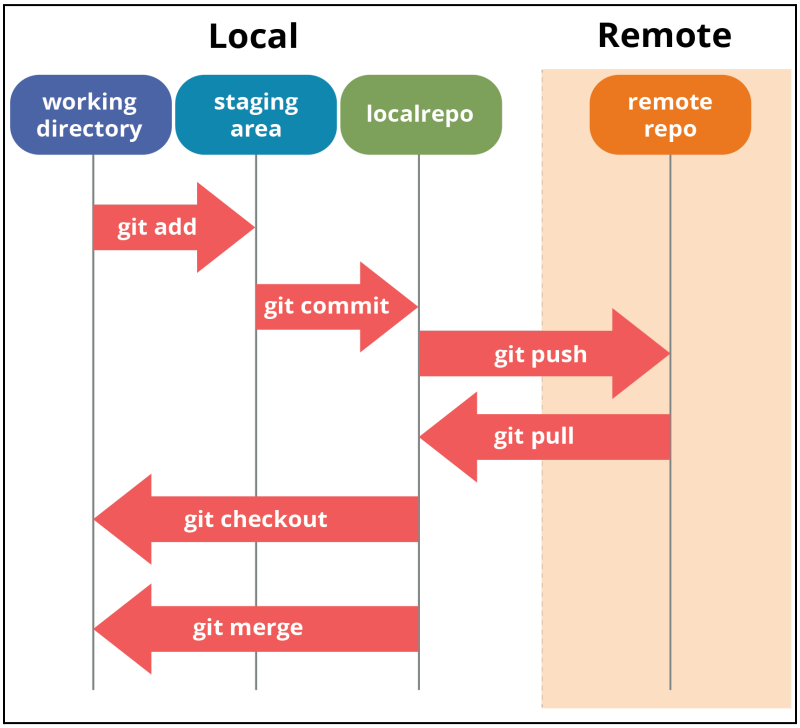
How Version Control Helps:

- Collaboration: Version control allows multiple developers to work on a project simultaneously. It helps in merging changes seamlessly, avoiding conflicts and ensuring a smooth collaborative workflow.

- History Tracking: Every change made to the project is recorded, providing a comprehensive history. This enables developers to understand how the codebase evolved over time, who made specific changes, and why.
- Branching and Merging: Version control systems like Git facilitate branching, allowing developers to work on separate features or bug fixes without affecting the main codebase. Merging branches brings these changes back together, preserving the integrity of the project.
- Reverting Changes: If a mistake is made or an undesirable change is introduced, version control allows developers to revert to a previous state, mitigating the impact of errors.
- Backup and Recovery: The central repository in version control serves as a backup. In case of data loss or system failures, developers can recover their work from the repository.

By adopting version control with Git and platforms like GitHub, developers enhance collaboration, improve project management, and build a robust foundation for software development. The ability to manage changes effectively contributes to a more efficient and organized development process.

The following diagram depict the all supported operations in GIT



Step 1: Install Git

- Download the Git installer from the official website: Git for Windows.
- Run the installer and follow the on-screen instructions.
- During the installation, choose the default options unless you have specific preferences.
- Open the Git Bash terminal to verify the installation. You can find it in the Start menu.

Step 2: Configure Git

- Open a terminal (Git Bash on Windows, Terminal on macOS/Linux) and set your name and email:
- Code:
  git config --global user.name "Your Name"
  git config --global user.email "email@example.com"

Step 3: Create a GitHub Account

- Visit GitHub and click on "Sign up" to create a new account.
- Follow the prompts to set up your account, including choosing a username and verifying your email.

Step 4: Create a New Repository on GitHub

- Log in to your GitHub account.
- Click on the '+' sign in the top right corner and select "New repository."
- Fill in the repository name, add a description, and choose to initialize it with a README if you want. Click "Create repository."

Step 5: Initialize a Git Repository Locally

- Navigate to the directory where you want to initiate version control using Git:
  cd /path/to/your/project
- // Initialize a new Git repository:
  git init

Step 6: Link Local Repository to GitHub

- Copy the HTTPS or SSH URL of your GitHub repository. You can find it on the repository page.
  git remote add origin https://github.com/username/repository.git

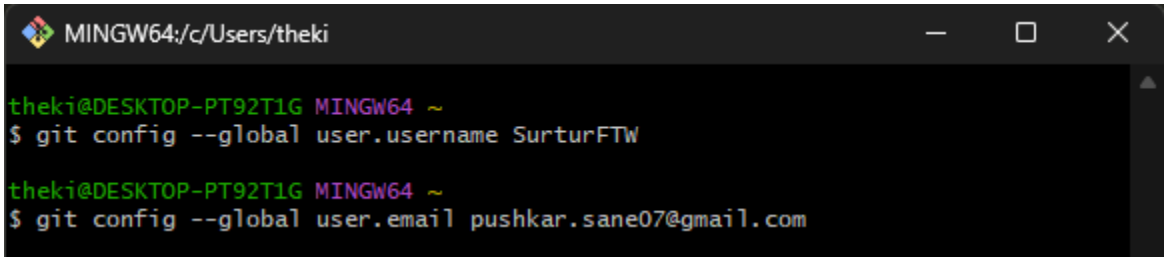
Step 7: Add and Commit Files

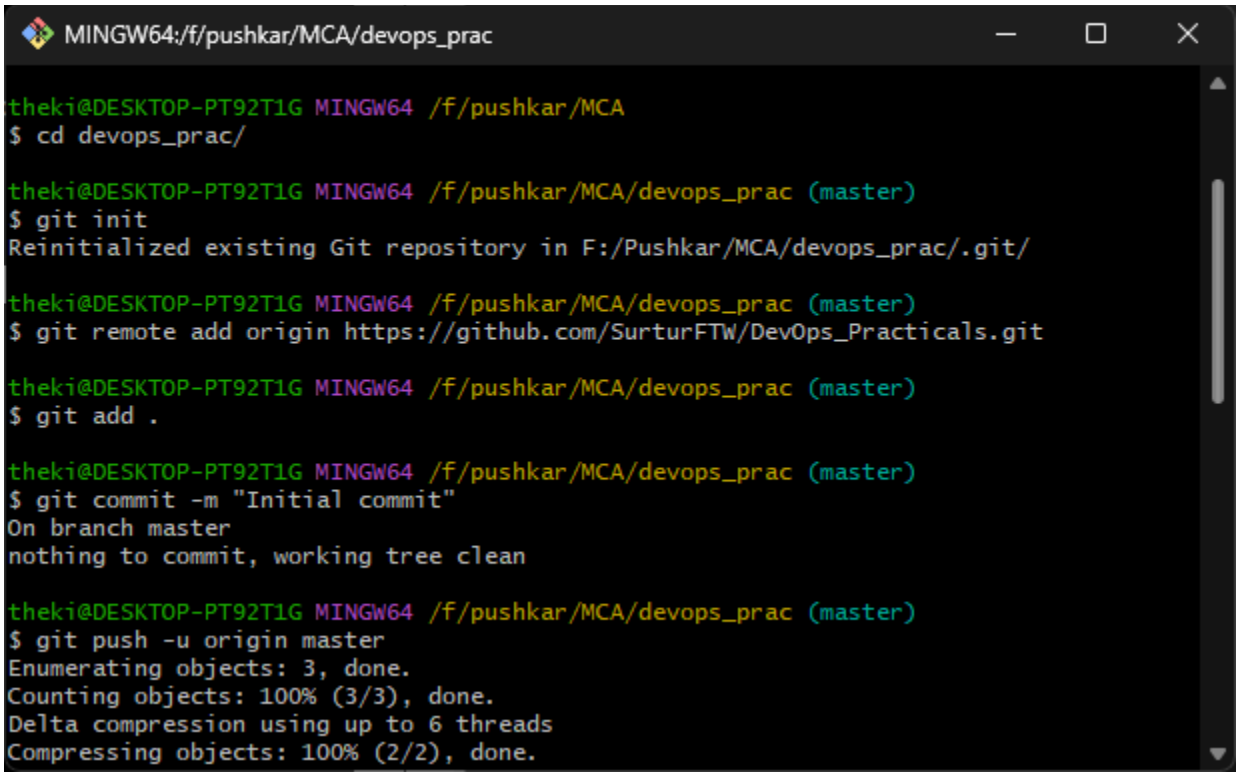
- //Add files to your local repository:
  git add .
  //Commit the changes:
  git commit -m "Initial commit"

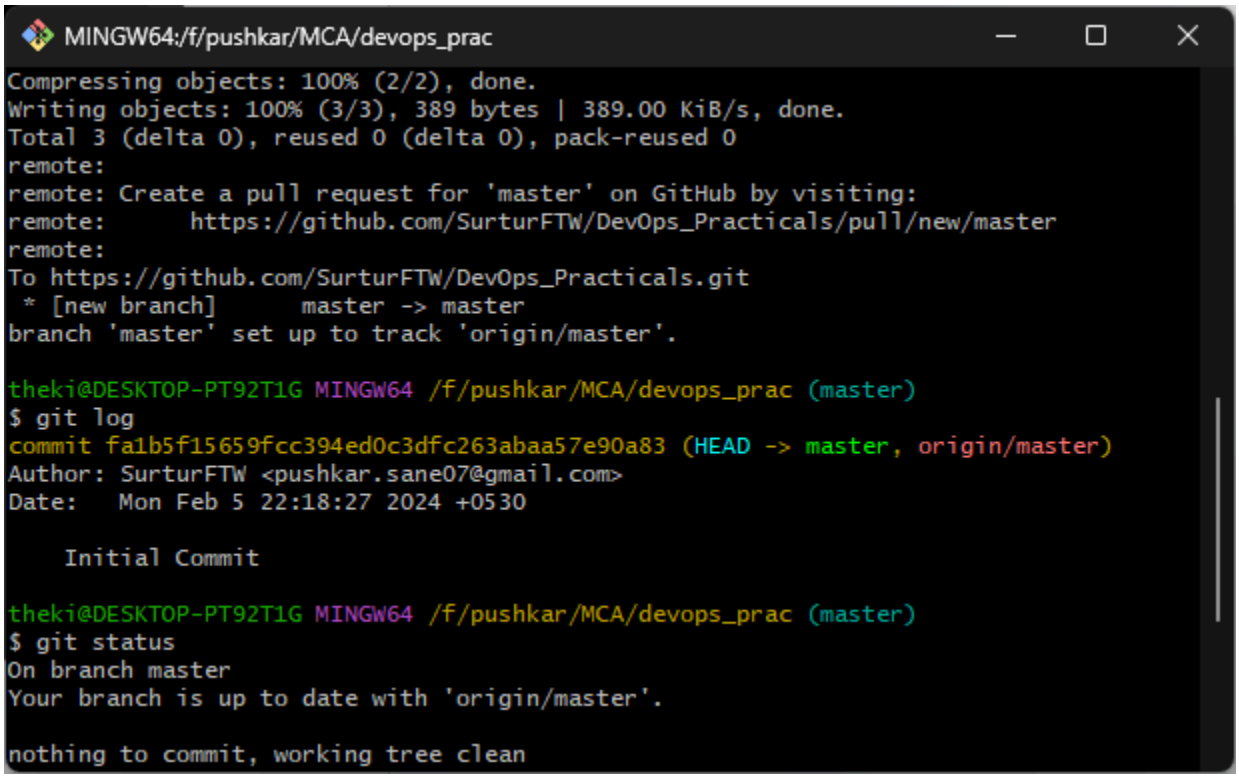
Step 8: Push to GitHub

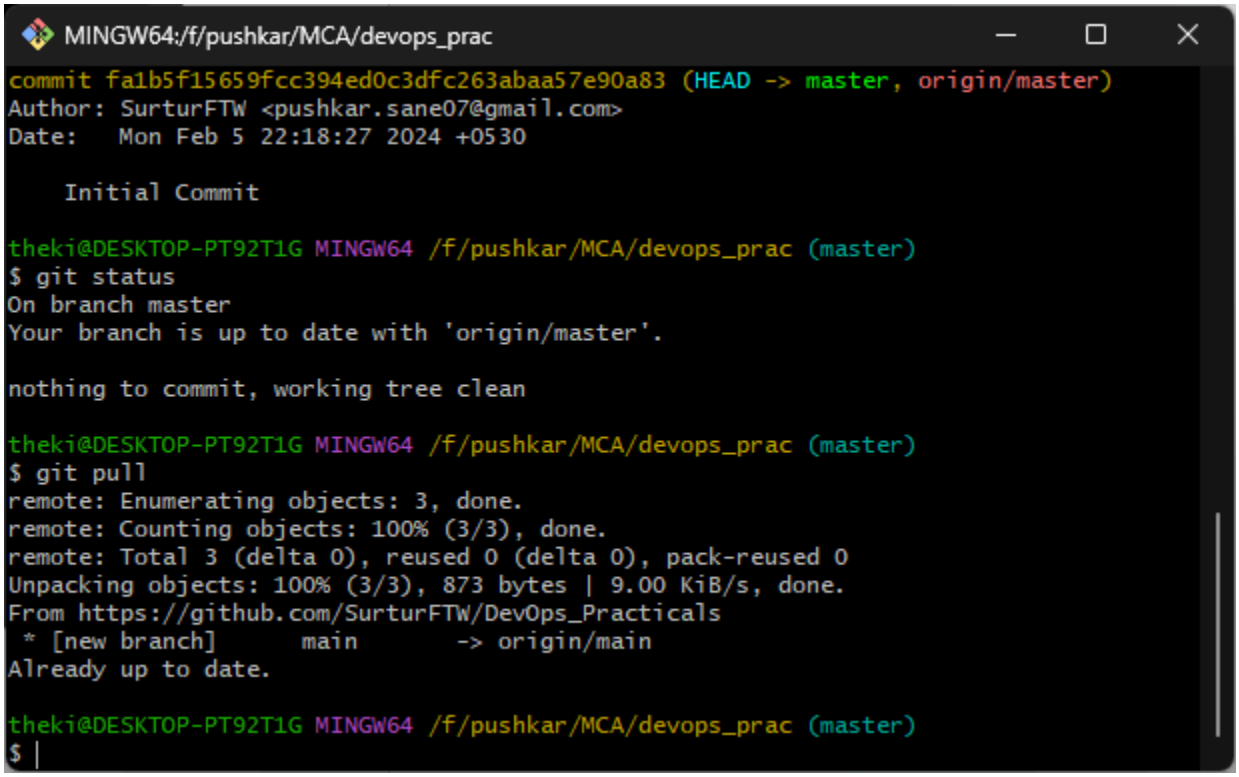
- //Push your changes to GitHub:
  git push -u origin master

Code:






MINGW64:/f/pushkar/MCA/devops_prac
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 389 bytes | 389.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote: https://github.com/SurturFTW/DevOps_Practicals/pull/new/master
remote:
To https://github.com/SurturFTW/DevOps_Practicals.git
* [new branch] master -> master
branch 'master' set up to track 'origin/master'.
theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (master)
$ git log
commit fa1b5f15659fcc394ed0c3dfc263abaa57e90a83 (HEAD -> master, origin/master)
Author: SurturFTW <pushkar.sane07@gmail.com>
Date: Mon Feb 5 22:18:27 2024 +0530
Initial Commit
theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.
nothing to commit, working tree clean


MINGW64:/f/pushkar/MCA/devops_prac
commit fa1b5f15659fcc394ed0c3dfc263abaa57e90a83 (HEAD -> master, origin/master)
Author: SurturFTW <pushkar.sane07@gmail.com>
Date: Mon Feb 5 22:18:27 2024 +0530
Initial Commit
theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.
nothing to commit, working tree clean
theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (master)
$ git pull
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 873 bytes | 9.00 KiB/s, done.
From https://github.com/SurturFTW/DevOps_Practicals
* [new branch] main -> origin/main
Already up to date.
theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (master)
$

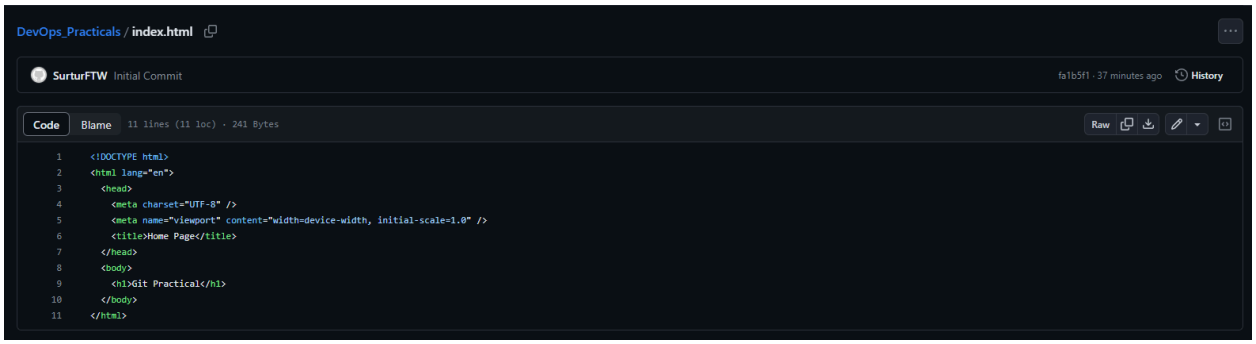
File on Local Repository:

index.html

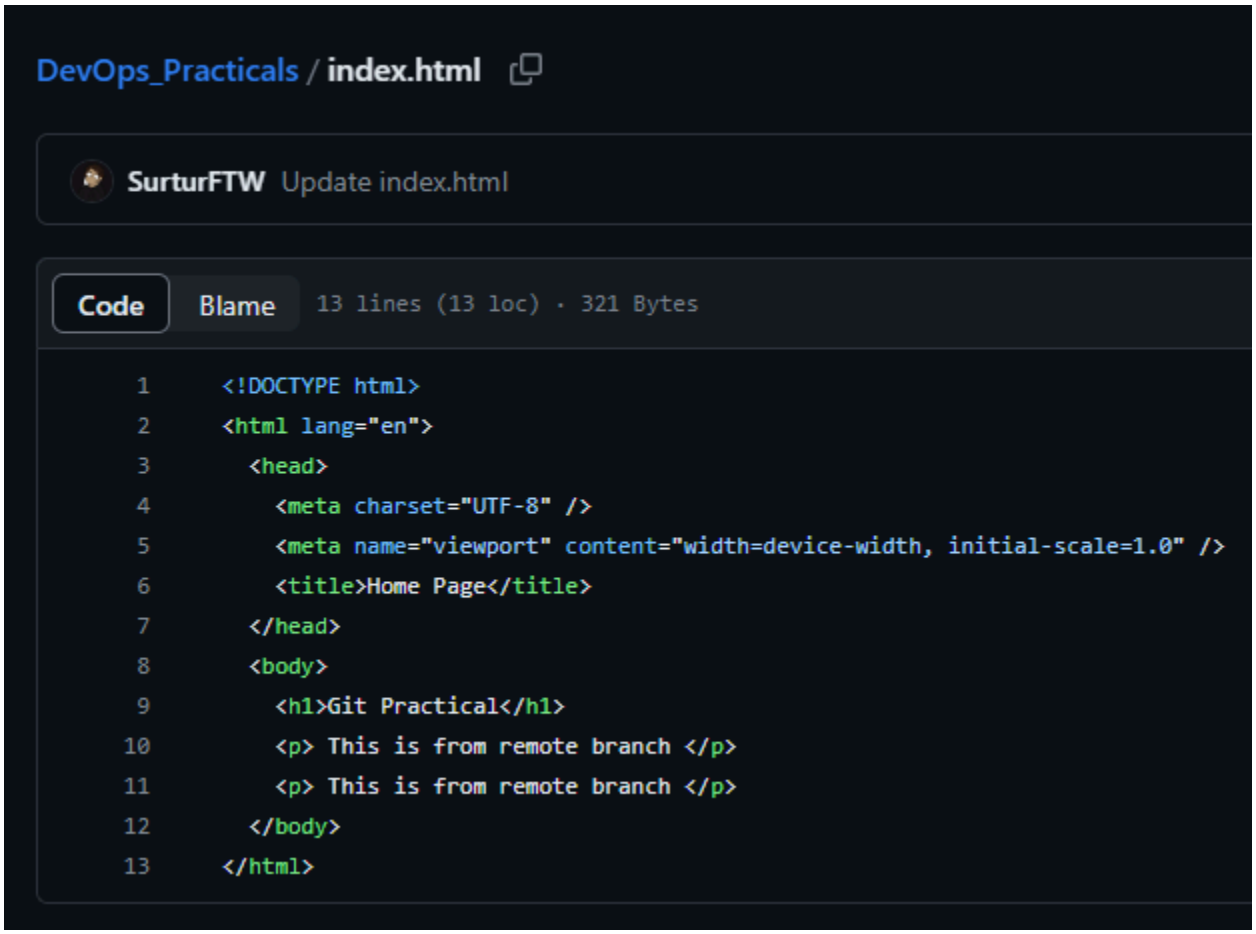
```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Home Page</title>
  </head>
  <body>
    <h1>Git Practical</h1>
  </body>
</html>
```

```
MINGW64:/f/pushkar/MCA/devops_prac
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 873 bytes | 9.00 KiB/s, done.
From https://github.com/SurturFTW/DevOps_Practicals
 * [new branch]      main       -> origin/main
Already up to date.

theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (master)
$ git checkout master
Already on 'master'
Your branch is up to date with 'origin/master'.

theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (master)
$ git branch main master -f

theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (master)
$ git checkout main
Switched to branch 'main'

theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (main)
$ git push origin main -f
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/SurturFTW/DevOps_Practicals.git
 + 8d7acab...fa1b5f1 main -> main (forced update)

theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (main)
$
```

File on Remote Repository:



File on Remote Repository with changes committed:



Pull command is used to download the remote updated repository into local one.

```
MINGW64:/f/pushkar/MCA/devops_prac
$ git remore -v
git: 'remore' is not a git command. See 'git --help'.

The most similar command is
        remote

theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (main)
$ git remote -v
origin  https://github.com/SurturFTW/DevOps_Practicals.git (fetch)
origin  https://github.com/SurturFTW/DevOps_Practicals.git (push)
orign   https://github.com/SurturFTW/DevOps_Practicals.git (fetch)
orign   https://github.com/SurturFTW/DevOps_Practicals.git (push)

theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (main)
$ git pull origin master
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 984 bytes | 9.00 KiB/s, done.
From https://github.com/SurturFTW/DevOps_Practicals
 * branch            master     -> FETCH_HEAD
   fa1b5f1..6fcbbf5  master     -> origin/master
Updating fa1b5f1..6fcbbf5
Fast-forward
 index.html | 2 ++
 1 file changed, 2 insertions(+)

theki@DESKTOP-PT92T1G MINGW64 /f/pushkar/MCA/devops_prac (main)
$
```

Now you can see the changes in local repository

index.html

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Home Page</title>
  </head>
  <body>
    <h1>Git Practical</h1>
    <p> This is from remote branch </p>
    <p> This is from remote branch </p>
  </body>
</html>
```