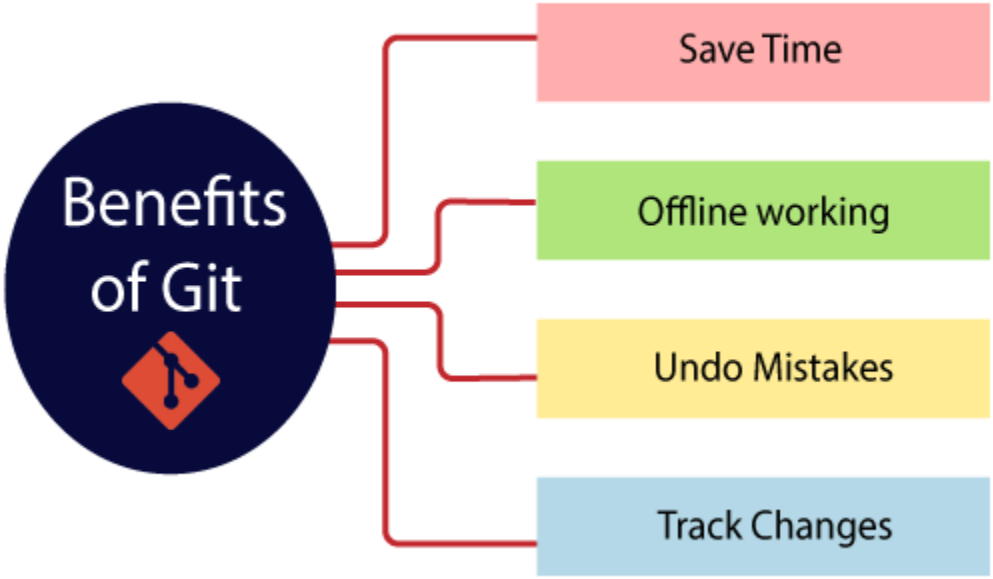
GIT(Global Information Tracker)

Git is a modern and widely used **distributed version control** system in the world. It is developed to manage projects with high speed and efficiency. The version control system allows us to monitor and work together with our team members at the same workspace.

Benefits of Git

A version control application allows us to **keep track** of all the changes that we make in the files of our project. Every time we make changes in files of an existing project, we can push those changes to a repository. Other developers are allowed to pull your changes from the repository and continue to work with the updates that you added to the project files.

Some **significant benefits** of using Git are as follows:



GitHub is a Git repository hosting service. GitHub also facilitates with many of its features, such as access control and collaboration. It provides a Web-based graphical interface.

GitHub is an American company. It hosts source code of your project in the form of different programming languages and keeps track of the various changes made by programmers.

It offers both **distributed version control and source code management (SCM)** functionality of Git. It also facilitates with some collaboration features such as bug tracking, feature requests, task management for every project.

Why is git needed?
When a team works on real-life projects, git helps ensure no code conflicts between the developers. Furthermore, the project requirements change often. So a git manages all the versions. If needed, we can also go back to the original code. The concept of branching allows several projects to run in the same codebase.

GitHub

By the name, we can visualize that it is a Hub, projects, communities, etc. GitHub is a Git repository hosting service that provides a web-based graphical interface. It is the largest community in the world. Whenever a project is open-source, that particular repository gains exposure to the public and invites several people to contribute.
The source code of several projects is available on github which developers can use in any means.
Using github, many developers can work on a single project remotely because it facilitates collaboration.

Git Install and Setup steps

- Download git from https://git-scm.com/download/win
- Install git and complete the setup
- Set environment variables
  - **Path C:\Program Files\Git\bin**
  - **Path C:\Program Files\Git\cmd**
- Create account in GitHub
- **Create any folder in d drive and add files in it. Right click on folder and click on "open git bash here"**

Below Are the git commands which are to be performed to pull and push files.

1. **git init**

The git init command creates a new Git repository. It can be used to convert an existing, unversioned project to a Git repository or initialize a new, empty repository. Most other Git commands are not available outside of an initialized repository, so this is usually the first command you'll run in a new project.

2. **git add .**

Common usages and options for git add

- git add <path>: Stage a specific directory or file
- git add .: Stage all files (that are not listed in the .gitignore) in the entire repository
- git add -A: stages all files, including new, modified, and deleted files, including files in the current directory *and* in higher directories that still belong to the same git repository
- git add -u: stages modified and deleted files only, NOT new files

3. **Make your first commit using below command**

git commit -m "New added(Any Message)"

4. **git remote add origin "https://github.com/lju/mern"**

When GitHub creates your repository, it presents an HTTP link, which is required as part of the git remote add origin command. The URL provided that uniquely identifies the GitHub repository I created is: **https://github.com/lju/mern**

mern is the repository name that has been created on **github**.

Open a terminal window and run the following git remote add origin command:

git remote add origin "**https://github.com/lju/mern**"

This command will execute, but the system won't provide any feedback to the terminal window. To verify that the remote repo was added to your configuration, use the git remote –v command. This command will show that GitHub is the fetch and push targets of the local repository.

5. **How you push your first commit see below command** :

○ **git push origin master**

Here master is the branch of the created repository on GITHUB.

Suppose, I made any changes locally in file test.txt and I want to push only that file to my repository mern

- git add test.txt
- git commit -m "updated file"
- git push origin master

```
Nidhi@ITICT406-9 MINGW64 /d/NAS/FSD2/mongodb/ZG
$ git init
Initialized empty Git repository in D:/NAS/FSD2/mongodb/ZG/.git/

Nidhi@ITICT406-9 MINGW64 /d/NAS/FSD2/mongodb/ZG (master)
$ git add -A

Nidhi@ITICT406-9 MINGW64 /d/NAS/FSD2/mongodb/ZG (master)
$ git remote add origin "https://github.com/nidhi2808/mern1"

Nidhi@ITICT406-9 MINGW64 /d/NAS/FSD2/mongodb/ZG (master)
$ git commit -m "New added"
[master (root-commit) 84b131d] New added
 7 files changed, 227 insertions(+)
 create mode 100644 Commands 5-7-2023.docx
 create mode 100644 m1.js
 create mode 100644 one.js
 create mode 100644 task.js
 create mode 100644 then & catch.txt
 create mode 100644 two.html
 create mode 100644 two.js

Nidhi@ITICT406-9 MINGW64 /d/NAS/FSD2/mongodb/ZG (master)
$ git push origin master
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (9/9), 21.31 KiB | 992.00 KiB/s, done.
Total 9 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/nidhi2808/mern1
 * [new branch]      master -> master

Nidhi@ITICT406-9 MINGW64 /d/NAS/FSD2/mongodb/ZG (master)
$ git branch -M branch1

Nidhi@ITICT406-9 MINGW64 /d/NAS/FSD2/mongodb/ZG (branch1)
$ git push origin branch1
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'branch1' on GitHub by visiting:
remote:      https://github.com/nidhi2808/mern1/pull/new/branch1
remote:
To https://github.com/nidhi2808/mern1
 * [new branch]      branch1 -> branch1

Nidhi@ITICT406-9 MINGW64 /d/NAS/FSD2/mongodb/ZG (branch1)
$ git pull https://github.com/nidhi2808/mern1 branch1
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 673 bytes | 1024 bytes/s, done.
From https://github.com/nidhi2808/mern1
 * branch            branch1    -> FETCH_HEAD
Updating 84b131d..6b1cdb6
Fast-forward
 m1.js | 7 ++-----
 1 file changed, 2 insertions(+), 5 deletions(-)
```

git status

The git status command displays the state of the working directory and the staging area. It lets you see which changes have been staged, which haven't, and which files aren't being tracked by Git.

Branch in git

- **Listing current branches**

Use the following command to view current branches in the code repository:

```
git branch
```

- **Making a new branch**

The following command makes a new branch in the code repository:

```
git branch <branch-name>
```

- **Switching to a branch**

The following command allows you to switch to a branch:

```
git checkout <branch-name>
```

- **Pushing the branch to Git**

To push the new branch to Git, we can use the following command:

```
git push origin <branch-name>
```

Assuming the repo you're working in contains pre-existing branches, you can switch between these branches using git checkout. To find out what branches are available and what the current branch name is, execute git branch.

- **Rename your local branch**

If you are on the branch you want to rename:

```
git branch -m new-name
```

If you are on a different branch:

```
git branch -m old-name new-name
```

git pull

The git pull command is used to fetch and download content from a remote repository and immediately update the local repository to match that content.
We can pull the repository by using the **git pull** command. The syntax is given below:

```
git pull origin master
```

- In the above syntax, the term **origin** stands for the repository location where the remote repository situated.
- **Master** is considered as the main branch of the project.
- It will overwrite the existing data of the local repository with a remote repository.

You can check the remote location of your repository. To check the remote location of the repository, use the below command:

```
git remote -v
```

If origin already exist and you want to add/move to new origin:

- First remove origin

```
git remote remove origin
```

- Add New origin

```
git remote add origin "https://github.com/lju/Trial"
```

React App upload (commands to be written in vs code terminal > in your react app)

```
git init
git add .
git commit -m "Committed save"
git remote add origin "https://github.com/lju/FSD-2"
git push origin master
```

Additional commands

- **Configure the proxy**

You may need to configure a proxy server if you're having trouble cloning or fetching from a remote repository or getting an error like unable to access '...' Couldn't resolve host '...'.

You can configure these globally in your user ~/.gitconfig file using the --global switch, or local to a repository in its .git/config file.

Setting a global proxy

Configure a global proxy if all access to all repos require this proxy

```
git config --global http.proxy http://192.168.10.252:808
```

- **Setting Global Git Username and Email**

Git allows you to set a global and per-project username and email address. You can set or change your git identity using the git config command. Changes only affect future commits. The name and email associated with the commits you made prior to the change are not affected.

The global git username and email address are associated with commits on all repositories on your system that don’t have repository-specific values.

To set your global commit name and email address run the git config command with the --global option:

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

git config --global user.email "youremail@yourdomain.com"
```