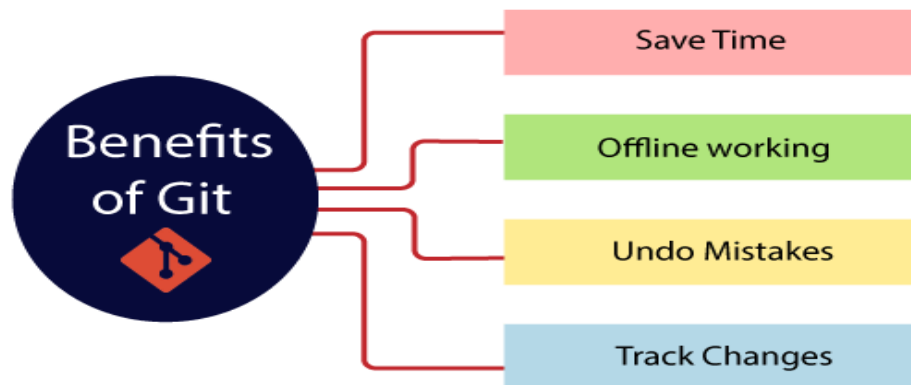


## Git and GitHub

### Git (Global Information Tracker)

Git is a **distributed version control system** designed for **speed, efficiency, and team collaboration**. It helps developers track file changes, manage multiple versions, and merge contributions from multiple developers.



### Why Git?

- ✓ Tracks all changes made to files in a project
- ✓ Supports **branching** and **merging** for isolated feature development
- ✓ Enables independent work and safe integration of code
- ✓ Allows rollback to previous versions if needed

### GitHub

GitHub is a **cloud-based platform** for hosting Git repositories. It provides a **web interface** and powerful tools for **collaboration, project tracking, and remote teamwork**.

### Key Features:

- ✓ Web interface to manage repositories
- ✓ Pull requests and code review tools
- ✓ Bug tracking, feature requests, and task management
- ✓ Supports open-source contributions
- ✓ Enables teams to work together remotely

## Chapter 8

Tool	Purpose
Git	Version control (local changes tracking)
GitHub	Online collaboration and code hosting

---

### Git Installation & Setup

1. **Download Git:** <https://git-scm.com/download/win>
2. **Install Git** (default options are usually fine)
3. **Set Environment Variables** (if needed):
  - C:\Program Files\Git\bin
  - C:\Program Files\Git\cmd
4. **Create a GitHub account:** <https://github.com>
5. **Create a local folder** (e.g., D:\MyProject)
6. **Open Git Bash** in that folder: Right-click → *"Open Git Bash here"*

## Basic Git Commands (Pull/Push Workflow)

### 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>**: Stages a specific directory or file.
- **git add .**: Stages new, modified, and deleted files in the current directory and its subdirectories.
- **git add -A**: Stages all new, modified, and deleted files across the entire repository.
- **git add -u**: Stages only modified and deleted files, excluding new files.

Consider below structure:

```
/project
├── src/
│   └── app.js (modified)
└── README.md (deleted)
```

- ✓ If you're in `/project/src` and run **git add .**  
→ It will stage `app.js`, **but not README.md** (since it's outside `src`)
- ✓ If you run **git add -A** from anywhere  
→ It will stage **both app.js and the deleted README.md**

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

```
git commit -m "Initial commit(Any Message)"
```

- ✓ **git commit:**  
Creates a **commit** — a recorded version of the current project state (like saving a checkpoint).
- ✓ **-m "Initial commit":**  
Adds a **commit message** inline (`-m` stands for message).  
"Initial commit" is the message describing **what this commit does**.

### 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**.

## Chapter 8

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

```
git remote add origin "https://github.com/liu/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 below command.
  - **git remote -v**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 init
- 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 ml.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), 21.21 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..6b1cbb6
Fast-forward
 ml.js | 7 +++++
 1 file changed, 2 insertions(+), 5 deletions(-)
```

## Chapter 8

### git remote add origin <URL>

Use **only once per local project** to link it to a GitHub repo.

This tells Git where (which GitHub repository) to push your code.

Example:

```
git remote add origin https://github.com/your-username/project-name
```

**Only needed again** if you:

- Rename your remote
- Move the project to a new GitHub repo
- Delete and re-add the remote

### git push origin master (or main)

Use this command **every time you want to upload commits** to GitHub.

Example

```
git add .  
git commit -m "Feature added"  
git push origin master
```

**Remember:**

- origin = the GitHub repo you connected
- master (or main) = the branch you're pushing to

### git remote remove origin

Remove the existing remote. This deletes the current origin (removes the old GitHub link).

### Pull Command

**Pull latest changes from the remote master (or main) branch**

```
git pull origin master
```

This fetches and merges the latest changes from the remote master branch into your current branch.

You can use git pull with **any branch name**, not just master. Just replace master with the branch you want to pull from.

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

### git status

The git status command shows the current **state** of your working directory and **staging area**. It tells you:

## Chapter 8

- Which files are **modified** but not yet staged
- Which files are **staged** and ready to be committed
- Which files are **untracked** (new files not added to Git)

```
git status
```

## Branching Commands

### List local branches

```
git branch
```

This shows all branches in your local repo. The current branch will be marked with an asterisk \*.

### Create a new branch

```
git branch feature-login
```

This creates a new branch named feature-login.

### Switch to an existing branch

```
git checkout feature-login
```

This changes your working directory to the feature-login branch.

### Push the branch to GitHub

```
git push origin feature-login
```

This uploads the new branch to your GitHub repository.

### Rename the current branch

```
git branch -m dev
```

This renames the current branch to dev.

### Rename a different branch

```
git branch -m old-feature new-feature
```

This renames old-feature branch to new-feature.

### 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"
```

## Chapter 8

No.	Command	Description	Example Syntax
<b>Git Basics &amp; Setup</b>			
1	git init	Initialize a new Git repository	git init
2	git status	Shows staged, unstaged, untracked files	git status
3	git add .	Stages all changes in current directory	git add .
4	git add -A	Stages changes in entire repository	git add -A
5	git commit -m "msg"	Saves staged changes with a message	git commit -m "Initial commit"
<b>Remote Repository (GitHub)</b>			
6	git remote add origin <url>	Connects local repo to GitHub	git remote add origin https://github.com/user/repo.git
7	git remote -v	Lists configured remotes	git remote -v
8	git push origin master	Pushes local commits to master branch on GitHub	git push origin master
9	git remote remove origin	Removes the linked GitHub remote	git remote remove origin
<b>Branching</b>			
10	git branch	Lists local branches	git branch
11	git branch <branch-name>	Creates a new branch	git branch feature-login
12	git checkout <branch-name>	Switches to an existing branch	git checkout feature-login
13	git push origin <branch-name>	Pushes branch to remote	git push origin feature-login
14	git branch -m new-name	Renames current branch	git branch -m dev
15	git branch -m old new	Renames a different branch	git branch -m old-feature new-feature
<b>Pulling Changes</b>			
16	git pull origin master	Pulls latest master from GitHub	git pull origin master
<b>Git Configuration</b>			
22	git config --global user.name "Your Name"	Sets global username	git config --global user.name "John Doe"



## Chapter 8

No.	Command	Description	Example Syntax
23	git config --global user.email "you@example.com"	Sets global email	git config --global user.email "john@example.com"
24	git config --global http.proxy <proxy-url>	Set global proxy	git config --global http.proxy http://192.168.10.252:808