**Version Control**

* Version control is a system for tracking and managing changes to files over time, allowing developers and teams to collaborate, revert to previous versions, and maintain a clear history of a project's evolution.
* It uses repositories to store project files and their history, with tools like Git enabling features such as parallel development, conflict resolution, and easy access to any past iteration of the project.

**Difference Between Git and GitHub**

|  |  |
| --- | --- |
| **Git** | **GitHub** |
| Git is a distributed version control system. | GitHub is a web-based platform that provides hosting for Git repositories. |
| It helps developers track changes in their code, revert to earlier versions, and collaborate with others. | It lets you store your Git projects online, so you can share them, collaborate with others, and use features like pull requests, issues, and CI/CD. |
| It works locally on your computer. | Requires internet to sync your local Git repo with the remote repo hosted on GitHub. |

**Local Repository**

* This is the copy of your project on your own computer.
* It has:
* Your project files.
* The Git history (commits, branches, etc.).
* You can work offline: commit, branch, merge, etc. all happen locally.

**Remote Repository**

* This is a copy of your project stored on the internet or a server (e.g., GitHub, GitLab, Bitbucket).
* It allows multiple people to work on the same project and share changes.
* To connect the two, you use commands:
* git push → send your local changes to the remote.
* git pull → bring others’ changes from the remote to your local copy.
* **git remote -v:** Lists the current Repo that I am working on.
* **git remote remove origin :** This will remove the repo that is unlink that repo.
* **git remote add origin** [**https://github.com/Surendra1204/Git.git**](https://github.com/Surendra1204/Git.git) **:** This will connect to the new repo.

**Make Some Changes to the files present in the repo.**

* **git add Git\_hub.docx:** It moves Git\_hub.docx into the staging area.
  + **Staging Area (Index) –** a “waiting room” where you put changes you want in the next commit (git add).
* **git commit -m "added some changes”:** This is like taking a snapshot.
  + **A** commit **in Git is like a** snapshot of your project’s files at a specific point in time**.**
    - **It records** what changed **(additions, deletions, modifications).**
    - **It includes a** commit message **you write to describe the change.**
    - **It has a unique** ID (hash) **so Git can always find it later.**
* **git push:** This will push all the things in the remote repo.

**Make Some Changes to the files present in the repo.**

* **git checkout -b feature-branch:** This will create a new branch.
* **copy Git\_hub.docx Git\_hub\_copy.docx:** Copying a file just to check.
* **git add Git\_hub\_copy.docx:** Add the files to the branch repo.
* **git commit -m " New branch added”:** Commit
* **git push -u origin feature-branch:** Push the branch to the repo, there you will see two branch and feature-branch

**git fetch:** Downloads new commits, branches, tags from the remote (origin) → but does NOT merge them into your working code.

**git pull:** Does two steps in one:

* git fetch (download new commits)
* git merge (merge those commits into your current branch)

**Git Status**

**What is HEAD?**

* HEAD = a pointer that tells Git “Which commit am I currently looking at?”