**Git and GitHub**

**What is Git?**

Git is a **version control system (VCS)** that helps track changes in your code. It allows multiple people to work on a project simultaneously and keeps a history of all changes.

**What is GitHub?**

GitHub is a **cloud-based hosting service** for Git repositories. It allows developers to collaborate, review code, and manage projects efficiently.

**Git Basics: Commands**

**Initializing a Repository: -** This initializes an empty Git repository.

git init

**Cloning an Existing Repository: - This** copies a repository from GitHub to your local machine.

git clone <repo\_url>

**Checking Repository Status: -** Shows the current state of your repository (new, modified, staged files).

git status

**Adding Changes to Staging Area: -**

git add <filename> # Add a specific file

git add . # Add all changes

**Committing Changes: -**

git commit -m "Your commit message"

**Saves your changes with a message describing them: -** Viewing Commit History

git log

Shows the list of commits.

**4. Working with Branches**

Branches allow working on different features without affecting the main codebase.

**Creating a New Branch**

git branch new-feature

**Switching to Another Branch**

git checkout new-feature

**or (in newer Git versions)**

**git switch new-feature**

**Merging Branches**

**git checkout main # Switch to the main branch**

**git merge new-feature # Merge the branch into main**

**5. Pushing Code to GitHub**

1. Link your local repository to GitHub:

**git remote add origin <repo\_url>**

1. Push your code:

**git push -u origin main**

**6. Pulling and Fetching Changes**

Pulling Updates from GitHub

**git pull origin main**

Fetches and merges changes from the remote repository.

Fetching Changes Without Merging

**git fetch origin**

Gets updates from GitHub but doesn’t merge them automatically.

**7. Resolving Merge Conflicts**

When multiple people edit the same file, conflicts can occur.

1. Git will show the conflict in the file.
2. Manually edit the file to resolve the conflict.
3. Add the resolved file and commit it:

git add <conflicted\_file>

git commit -m "Resolved merge conflict"

**8. Undoing Changes**

Undo Last Commit (But Keep Changes)

**git reset --soft HEAD~1**

Undo Last Commit (And Remove Changes)

**git reset --hard HEAD~1**

**What is a Pull Request (PR)?**

A **Pull Request (PR)** is a feature in GitHub (and other Git platforms like GitLab, Bitbucket) that allows developers to propose changes to a project and request that the changes be reviewed and merged into the main branch.

Pull Requests are commonly used in **collaborative development** when working in teams, especially in open-source projects.

1. Steps to Create a Pull Request on GitHub

**Step 1: Fork and Clone the Repository**

If you're contributing to an open-source project, first **fork** the repository (creates a copy in your GitHub account), then clone it:

|  |
| --- |
| git clone https://github.com/your-username/repository-name.git  cd repository-name |

If you're working on your own repository, just clone it:

|  |
| --- |
| git clone https://github.com/original-owner/repository-name.git |

Step 2: Create a New Branch

|  |
| --- |
| git checkout -b new-feature  git checkout new-feature  or  git switch new-feature |

Before making changes, create a separate branch:

Step 3: Make Changes and Commit

Modify the code, then stage and commit your changes:

|  |
| --- |
| git add .  git commit -m "Added a new feature" |

Step 4: Push Your Changes

Push the new branch to your GitHub repository:

|  |
| --- |
| git push origin new-feature |

Step 5: Create a Pull Request on GitHub

1. Go to your repository on GitHub.
2. Click on the "Compare & pull request" button that appears after pushing.
3. Write a clear **title** and **description** explaining what changes you made.
4. Choose the target branch (usually main or develop).
5. Click **"Create Pull Request"**.

**2. Reviewing a Pull Request**

If you're a project maintainer or team member:

Open the Pull Requests tab on GitHub.

Click on a PR to review the changes.

You can comment on specific lines of code, request changes, or approve.

Once satisfied, click Merge Pull Request to merge it into the main branch.

**3. Merging a Pull Request**

Once reviewed, merge the PR using one of the following methods:

* **Merge Commit**: Creates a new commit for the merge.
* **Squash and Merge**: Combines all commits into one before merging.
* **Rebase and Merge**: Merges changes without creating an extra commit.

|  |
| --- |
| git branch -d new-feature  git push origin --delete new-feature |

4. Updating a Pull Request

If changes are requested in a PR, update your branch:

|  |
| --- |
| git checkout new-feature  git add .  git commit -m "Implemented requested changes"  git push origin new-feature |

Note:-

Pull after every merge to escape from merge conflit.