Overview of Git

Git is a distributed version control system (VCS) used for tracking changes in files and coordinating work among multiple developers. It helps manage source code efficiently, allowing collaboration, version tracking, and rollback capabilities.

Key Features of Git

Version Control: Tracks changes in code over time.
Branching & Merging: Work on different features independently and merge them later.
Collaboration: Enables multiple developers to work on the same project.
Distributed System: Every developer has a complete copy of the repository.
Speed & Efficiency: Performs operations quickly compared to centralized systems.

Advantages of Git

Version Control: Tracks changes and allows reverting to previous versions.

Distributed System: Enables offline work and reduces reliance on a central server.

Branching & Merging: Allows parallel development without affecting the main codebase.

Fast Performance: Optimized for speed, making commits, branching, and merging quick.

Collaboration: Enables multiple developers to work on the same project efficiently.

Disadvantages of Git

Steep Learning Curve: New users find Git commands and workflows challenging.

Merge Conflicts: Resolving conflicts can be complex, especially in large teams.

Large Repository Issues: Performance slows down with very large repositories.

Command-Line Dependency: Many advanced Git features require command-line usage.

Difficult Undo Process: Undoing changes requires knowledge of multiple commands (`reset`, `revert`, `checkout`).

Installing Git

Download and install Git from the official website: https://git-scm.com/

Verify Installation

```
git --version
```

Basic Git Workflow

1 .Initialize a Repository – `git init`
2 .Clone a Repository – `git clone <repo-url>`
3. Make Changes – Modify files in the project
4 .Stage Changes – `git add <file>` (Prepares files for commit)
5 .Commit Changes – `git commit -m "message"` (Saves the changes)
6 .Push Changes – `git push origin <branch>` (Uploads changes to a remote repository)
7 .Pull Updates – `git pull origin <branch>` (Gets the latest changes)
8 .Merge Branches – `git merge <branch>` (Combines code from different branches)

Common Git Commands

(1)Configuring Git

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

git config --global user.email "your.email@example.com"
```

(2)Initializing a Repository

```
git init
```

(3)Cloning a Repository

```
git clone <repository-url>
```

(4)Staging and Committing Changes

```
git add <file-name>  # Stage a file

git add .  # Stage all changes

git commit -m "Your commit message"
```

(5)Viewing Status and Logs

```
git status

git log
```

(6)Pushing and Pulling Changes

```
git push origin main

git pull origin main
```

(7)Creating and Switching Branches

```
git branch <branch-name>

git checkout <branch-name>

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

(8)Merging Branches

```
git checkout main
```

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

(9)Resolving Merge Conflicts

Open the conflicted file.

Edit and resolve conflicts.

Stage and commit the changes.

(10)Working with Remote Repositories

```
git remote add origin <repository-url>
```

```
git remote -v
```

```
git push -u origin main
```

(11)Undoing Changes

```
git reset --soft HEAD~1
```

```
git reset --hard HEAD~1
```

```
git checkout -- <file-name>
```

Why Use Git?

Better collaboration – Multiple people can work without conflicts.

Backup & recovery – Previous versions of code are always available.

Parallel development – Work on multiple features simultaneously.

Efficiency – Handles large projects quickly and effectively.