Advantages of Git

- **Distributed Version Control System (DVCS):-**
  Every developer has a full copy of the repository.
  Work is not dependent on a central server; code history is always available locally.
- **Faster Performance:-**
  Local commits, diffs, and logs make operations extremely fast compared to centralized systems.
- **Strong Branching and Merging:-**
  Git allows easy creation, merging, and deletion of branches.
  Supports multiple workflows like feature branching, Git Flow, etc.
- **Data Integrity:-**
  Uses SHA-1 hashing for every commit and file; ensures integrity and prevents corruption.
- **Efficient Collaboration:-**
  Enables parallel development through branches and pull requests.
  Merges changes from multiple developers effectively.
- **Open Source and Free:-**
  No cost involved; large community support and frequent updates.
- **Cross-Platform Support:-**
  Available on Windows, macOS, Linux, and can be used with many tools like GitHub, GitLab, Bitbucket.
- **Staging Area (Index):-**
  Allows developers to review and selectively commit changes before making them permanent.
- **Lightweight and Scalable:-**
  Git repositories are smaller and more efficient than centralized systems like SVN or CVS.

Disadvantages of Git

- **Complexity for Beginners**
  Steep learning curve, especially when working with commands like rebase, merge conflicts, cherry-pick, etc.

- **Large Binary File Handling**
  Git is inefficient with large binary files (e.g., images, videos, compiled code).
  Workarounds like Git LFS are needed.
- **Not Ideal for All Projects**
  For simple projects or non-developer teams, Git might feel overkill.
- **Merge Conflicts**
  Frequent merges in large teams can lead to complex merge conflicts that require manual resolution.
- **Requires Discipline**
  Improper branching or commit practices can lead to messy history or lost work.
- **Initial Setup Can Be Tedious**
  Setting up remote repositories and configuring SSH keys may be confusing for new users