Version control tools are a great way to enable collaboration, maintain versions, and track changes across the team. The greatest benefit of using version control tools is that you have the capacity to deal with an unlimited number of people, working on the same code base, without having to make sure that files are delivered back and forth.

Version Control Tools:

1. CVS (Concurrent Versions System)
   CVS was created in the UNIX operating system environment and is available in both Free Software Foundation and commercial versions. CVS works not by keeping track of multiple copies of source code files, but by maintaining a single copy and a record of all the changes. When a developer specifies a particular version, CVS can reconstruct that version from the recorded changes.CVS is typically used to keep track of each developer's work individually in a separate working directory. When desired, the work of a team of developers can be merged in a common repository. Changes from individual team members can be added to the repository through a "commit" command.

   CVS is used to keep track of collections of files in a shared directory called "The Repository". Each collection of files can be given a "module" name, which is used to "checkout" that collection.After checkout, files can be modified, "committed" back into the Repository and compared against earlier revisions. Collections of files can be "tagged" with a symbolic name for later retrieval. You can add new files, remove files you no longer want, ask for information about sets of files in three different ways, produce patch "diffs" from a base revision and merge the committed changes of other developers into your working files.

   CVS does not employ a 'lock' mechanism on 'files-in-use'. If a bunch of users have read/write access to the repository, **anyone can update their versions at any time.**

2. Mercurial:
   Mercurial is a distributed revision-control tool for software developers. It is supported on Microsoft Windows and Unix-like systems, such as FreeBSD, macOS and Linux.

   Mercurial's major design goals include high performance and scalability, decentralization, fully distributed collaborative development, robust handling of both plain text and binary files, and advanced branching and merging capabilities, while remaining conceptually simple. Mercurial uses SHA-1 hashes to identify revisions. For repository access via a network, Mercurial uses an HTTP-based protocol that seeks to reduce round-trip requests, new connections and data transferred.No partial checkouts are allowed. Rollback will only undo the last commit.you can find changesets you've destroyed through git rebase –interactive using git reflog, but if it has been more than a month since you made the change, those changesets will be lost

3. Git

Git is free and open-source software. every Git directory on every computer is a full-fledged repository with complete history and full version-tracking abilities, independent of network access or a central server. Git specifically works by taking “snapshots” of files; if files remain unchanged in a particular version, it simply links to the previous files – this keeps everything fast and lean.

The benefits of a system like this is that multiple developers can make changes, and each change can then be attributed to a specific developer. On the downside, the fact that everything is stored on a remote database means no changes can be made when that server goes down; and if the central database is lost, each client only has the current version of whatever they were working on.

One of the biggest advantages of Git is its branching capabilities. Unlike centralized version **control** systems, Git branches are **cheap** and easy to merge. This facilitates the feature branch workflow popular with many Git users. Feature branches provide an isolated **environment** for every change to your codebase. A key design objective of Git is the kind of flexibility it offers to support several kinds of nonlinear development workflows and its efficiency in handling both small scale and large scale projects as well as protocols. It is uniquely designed to support tagging and branching operations and store each and every activity carried out by the user as an integral part of "change" history.

When compared to other VCS Git is most widely accepted system owing to its universally accepted usability and performance standards.