# **Introduction to Version Control**

## What is version control?

Version control is a system that records all changes and modifications to files for tracking purposes. Developers also use the term source control or source code management. The primary goal of any version control system is to keep track of changes. It achieves this by allowing developers access to the entire change history with the ability to revert or roll back to a previous state or point in time.

*Your development team is tasked to develop an app. Your project manager leads you to work in an Agile way of working where tasks are planned and executed in a two-week cycle. This is done to ensure which of the following benefits?*

* *Efficiency*

## Systems of version control and tools

Version control systems can be split into two types or categories. Centralized version control systems and distributed version control systems. Both types are quite similar, but they also have some key differences which set them apart from each other.

|  |  |
| --- | --- |
| **Centralized version control systems** | CVCS for short, contain a server and a client. The server contains the main repository that houses the full history of versions of the code base. Developers working on projects using a centralized system need to pull down the code from the server to their local machine. This gives the user their own working copy of the code base. The server holds the full history of changes. The client has the latest code, but every operation needs to have a connection to the server itself. In a centralized version control system, the server is the central copy of the project. After making changes to the code, the developer needs to push the changes to the central server so that other developers can see them. This essentially means that viewing the history of changes requires that you are connected to the server in order to retrieve and view them.  Advantages:   * Easier to learn * More access controls   Disadvantages:   * Slower |
| **distributed version control systems** | DVCS for short, are similar to the centralized model. You still need to pull code down from the server to view the latest changes. The key difference is that every user is essentially a server and not a client. This means that every time you pull down code from the distributed model, you have the entire history of changes on your local system.  Advantages:   * Speed * Performance |

|  |  |
| --- | --- |
| **Subversion** |  |
| **Perforce** |  |
| **AWS Code Commit** |  |
| **Mercurial** |  |
| **Git** |  |

*You are part of a development team and one of your colleagues ask you if a Centralized Version Control system (CVS) contains a server and a client. What would your answer be?*

* *Yes*

## A history of revisions

When working with a team of developers, it's essential for the codebase to have a source of truth that has all historical changes. Version control systems play an integral part in aiding this process by providing a full history of changes of every single file added to its repository. It makes collaboration across a team easier. Also aids in working toward a common goal. Whether it is adding new features and following the flow of how they were implemented, or discovering where a potential issue may have been introduced. Being able to accurately pinpoint the who, the when, and the what of those changes is paramount.

The history will show who made the change, for what reason, the code itself and its changes, and the date and time of when they occurred.

*You are working with a team of developers on an app to help people work more efficiently. While you are developing this app the codebase you are working on goes through many revisions as each potential change gets peer-reviewed and given the ok. What is this system of checks that provides a detailed record called?*

* *Version control*