# ***Introduction to Version Control System with Git & GitHub***

# Version Control System

# Version control, also known as source control or revision control, is a software development practice that tracks and manages changes to files over time.

# Version control systems (VCS) 's particularly important for things like code, documents, or any project where multiple people might be working on the same files.

**How VCS works?**

1. **Keeps a history:** Every time you make a change to a file, the VCS takes a snapshot of that version and stores it. This allows you to see how the file has changed over time and revert to a previous version if needed.
2. **Collaboration:** Multiple people can work on the same files at the same time. The VCS helps to avoid conflicts by keeping track of who made which changes.
3. **Branching:** You can create different branches of your project, which allows you to experiment with changes without affecting the main version.

**Benefits of Using VCS** :

1. **Prevents data loss:** If you accidentally delete something or make a mistake, you can easily revert to a previous version.
2. **Collaboration:** Makes it easier for multiple people to work on the same project without conflicts.
3. **Organization:** Helps you keep track of changes and see how your project has evolved over time.

**Types of VCS:**

1. **Centralized Version Control System (CVCS):**

Here, all file versions are stored on a central server. This makes collaboration easy, but it can be a single point of failure if the server goes down.

1. **Distributed Version Control System (DVCS):**

Every user has a complete copy of the entire project history on their local machine. This allows for offline work and more flexibility, with popular options like ***Git*** being DVCS.

**Examples of VCS**

# Git

# GitHub

# Subversion (SVN)

# Mercurial

Differences between Git & GitHub

|  |  |  |
| --- | --- | --- |
| Features | Git | GitHub |
| Type | Version Control System (VCS) | Web-based Git Repository Hosting Service |
| Functionality | Tracks changes in code over time | Stores and manages Git repositories |
| Location | Installed locally on your computer | Hosted online on GitHub's servers |
| Collaboration | Requires additional tools for sharing | Built-in collaboration features like pull requests and issue tracking |
| Cost | Free and open-source software | Offers free and paid plans |

# **Different Features of GitHub**

1. **Git Comits :**

* A commit in a git repository records a snapshot of all the (tracked) files in your directory. It's like a giant copy and paste, but even better!
* Git also maintains a history of which commits were made when.

1. **Git Pull :**

* pull is a combination of 2 different commands: Fetch & merge.
* It is used to pull all changes from a remote repository into the branch you are working on. That is how you keep your local Git up to date from a remote repository.

1. **Git Push:**

* Git push is a command used in Git version control to transfer local commits to a remote repository, typically hosted on a platform like GitHub.
* It's like publishing your local changes and updates to a central location where others can access them.

1. Git Branch :

They allow you to create independent lines of development within your project without affecting the main codebase

* By default, you start with a single branch called ***master***. This branch represents the main line of development for your project.
* When you use the git branch ***<branch\_name>*** command, you create a new branch that diverges from the current commit on the timeline. This new branch acts like a separate working directory with its own history of commits.

## **Git Contribute**

At the heart of Git is collaboration. However, Git does not allow you to add code to someone else's repository without access rights.

In order to copy a repository, make changes to it, and suggest those changes be implemented to the original repository, follow the steps given below:

1. **Fork the Repository**

* A fork is a copy of a repository.
* This is useful when you want to contribute to someone else's project or start your own project based on theirs.
* fork is not a command in Git.

1. **Clone a Fork from GitHub**

* A clone is a full copy of a repository, including all logging and versions of files.
* Move back to the **original** repository, and click the green "Code" button to get the URL to clone

1. **Send Pull Request**

* A pull request is a formal way of proposing your changes to be merged into the main codebase of the project.
* You'll use the GitHub interface to create a pull request, which essentially sends a notification to the maintainers about your contributions and allows them to review your code before merging it.

MY LEARNING MATERIAL

* W3 School Tutorials
* MicroSoft following courses:
* Using GitHub Copilot with Python
* Challenge project - Build a minigame with GitHub Copilot and Python 
* Introduction to GitHub
* Manage an InnerSource program by using GitHub
* Create an open-source program by using GitHub best practices
* Upload your project by using GitHub best practices
* Migrate your repository by using GitHub best practices
* Maintain a secure repository by using GitHub best practices
* Guided project - Build auto suggest engine with Copilot
* Choose the DevOps tools
* Work with Azure Repos and GitHub
* Introduction to GitHub Actions
* Learn continuous integration with GitHub Actions
* Explore Azure Pipelines
* Create a build pipeline with Azure Pipelines
* Contribute to an open-source project on GitHub