**GIT - Global Information Tracker**

-->it is also know as version control system(VCS)

-->final version as source code

-->repository-->folders

-->security purpose

-->git is a central repository using which manage our project source code.

-->It is maintaining all the versions along with modifications done on specific application.

-->Because of maintaining versions we can troubleshoot & fix the bugs easily with previous version data.

-->It can record all the modifications done by timestamp(data,time,location).

--> Git is a distributed version control system

**Used For 2 Reasons:**

1.VCS(version control system)

2.Collaboration

**Why do we use get?**

-->In order to maintain multiple versions of the same file.

**Functionalities of VCS:**

1.It allows multiple developers to develop the code simultaneously.

2.Doesn't allow any type of overwriting.

3.Maintains the history of version.

4.Git is fast when compared with the other version controlling tools.

5. Multiple developers will collaborate and work on the same module

6.backup for project source code.

**Types of VCS:**

Their are 2 types

1.CVDS: Centralized version control system

-->SVN-sub version control system-->centralized server.

-->It consists of current version data only

-->V1.0---V2.0(Data)

-->Developer should connect to the server after then only they should doing their code.

**Disadvantage:**

1. No version history maintained
2. No security
3. If server destroyed then everything will get loss.

2.DVCS: Distributed version control system

-->github-It is a server.

-->Decentralized version control system.

-->Each & every developers need not want to connect to server everytime while they want to write the code

--->create a repository in gitbub-->remote repository.

--->we have the option of opting the source code by using the cloning method.

-->cloning method: Extracting the files or code from remote repository(github-server) to the local repository(our own laptops/systems)

command for cloning:

git clone URL

Git:

-->Git is a client-server architecture

-->Git bash-client

--->Github-server

Repository:

Group of project files to store in one single area.

-->Each project consists of one repository.

-->GitHub has many repositories.

Local Repository: our own laptops

Remote repository: Global repository

-->Repository which is present in GitHub(server).

Local Repository---->remote repository

push: We will send the files to the remote repository.

pull: remote repository to the local repository

------

task

**1.Differences b/w CVCS and DVCS**

| **Feature** | **CVCS (Centralized Version Control System)** | **DVCS (Distributed Version Control System)** |
| --- | --- | --- |
| **Structure** | Single central server stores all versions of the code. | Every developer has a full copy of the entire repository (including history). |
| **Dependency** | Requires constant connection to the central server. | Can work offline since the whole repo is on the local machine. |
| **Speed** | Slower because every action (like commit history check) requires server communication. | Faster because all actions are local. |
| **Backup** | Single point of failure (if central server crashes, data could be lost). | Every developer’s machine has a full backup of the repo. |
| **Collaboration** | Developers push and pull directly from the central server. | Developers can push/pull to/from each other or to a shared remote. |
| **Examples** | SVN (Subversion), CVS, Perforce | Git, Mercurial |

2.**VCS vs Collabartion**

**VCS (Version Control System)**

**What it is:**

A **VCS** is a **tool** that tracks changes made to files over time. It allows you to:

Save different versions of your code or documents.  
Revert to older versions if needed.  
See who made what changes and when.  
Work safely on new features without disturbing the main code.

**Types of VCS:**

* **CVCS** (Centralized VCS): All history stored in a central server.
* **DVCS** (Distributed VCS): Each developer has a full copy of history.

**Collaboration**

**What it is:**

**Collaboration** is the **process of working together** as a team to build something. This is about **communication**, **workflow**, and **team coordination**.

**Examples of Collaboration:**

* Assigning tasks.
* Writing documentation together.
* Discussing features in meetings.
* Reviewing each other's code.

Download git bash

create github accounts

download power shell

download vs code.