# VERSION CONTROL SYSTEM

WHAT IS VERSION CONTROL ?

Version control is a system that records the changes of a file or a set of files over time so that you can recall specific versions later.
These versions are recorded in a repository and can be recalled from the same.
There are three types of version control systems :

1. LOCAL VERSION CONTROL SYSTEM (LVCS)
2. CENTRALIZED VERSION CONTROL SYSTEM (CVCS)
3. DISTRIBUTED VERSION CONTROL SYSTEM (LVCS)

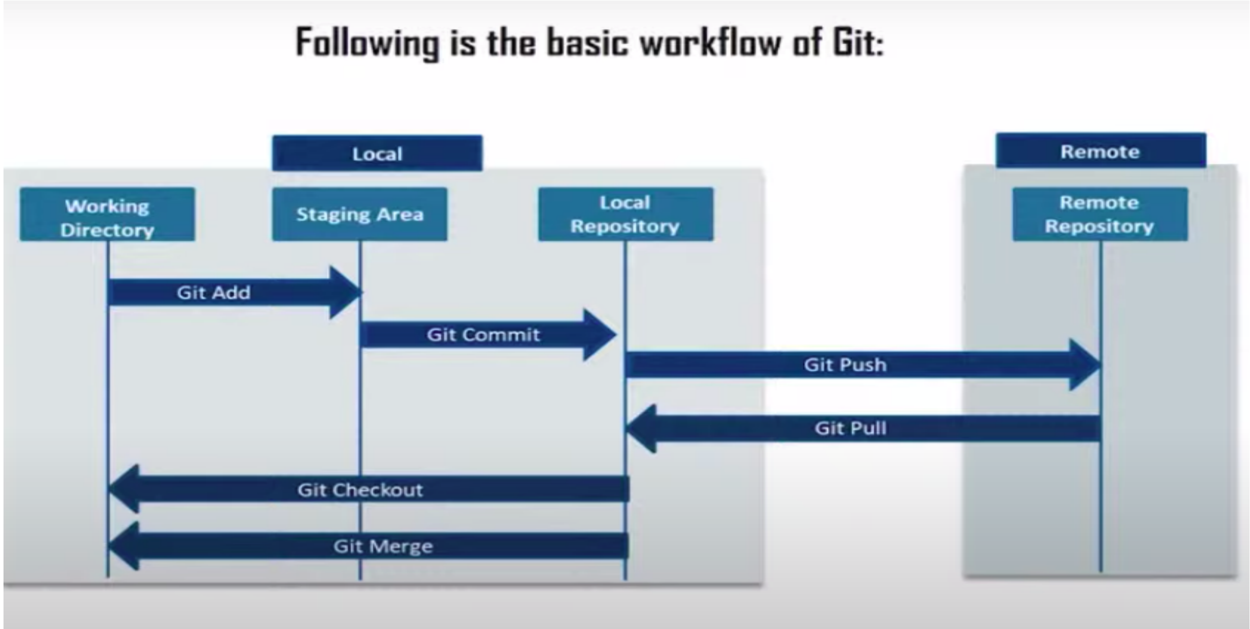
**REPOSITORY (Repo)**

A repository or a Repo is a storage space or a directory where your project can live. It can be local to a folder in your computer or it can be a storage space on another online host (like GitHub). you can keep code files, text files, image files. You name it inside a repository.

**GIT**

Git is a Distributed Version Control tool that supports distributed non-linear workflows by providing data assurance for developing quality software. It lets your team of developers work together on the same project from anywhere. Team members can work on files and easily merge their changes into one source.

**Basic workflow of the Git**



# VERSION CONTROL SYSTEM

GIT INSTALLATION AND SETUP

Installation of Git in linux operating system the following commands are used:

1. Start by updating the package index:

```
$sudo apt update
```

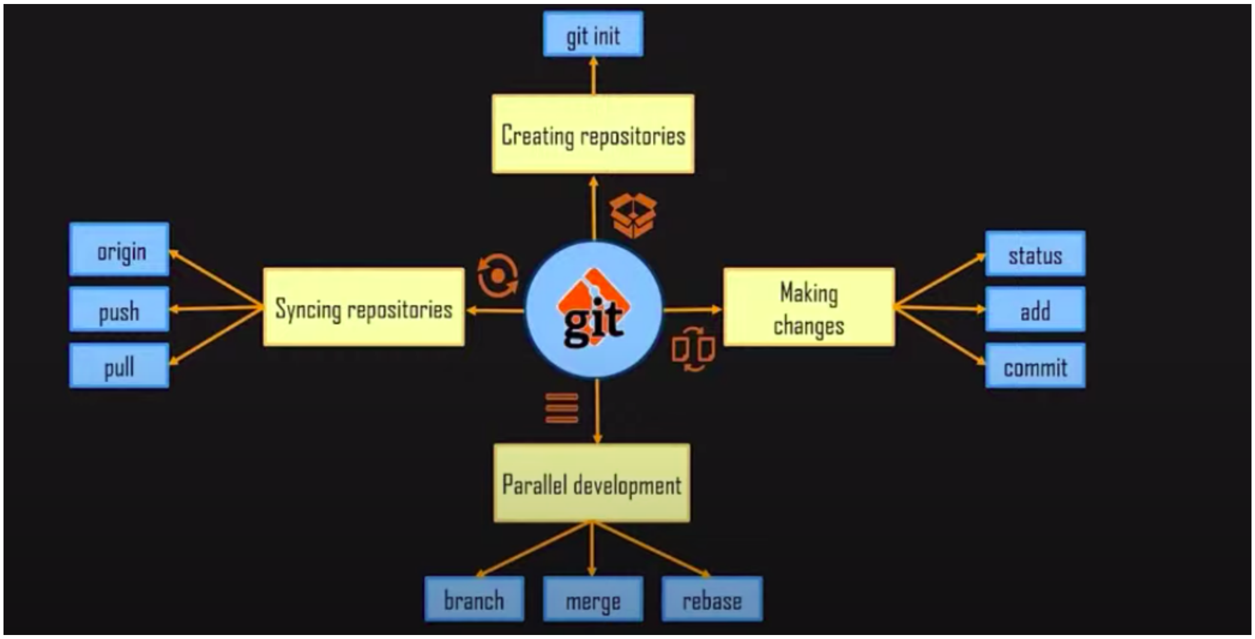
2. Install git

```
$sudo apt install git
```

3. Verify the installation by printing the version of git

```
$git --version
```

## Git operations and commands:



## Creating Repository:

Commands to perform this task are given below:

```
$git init
```

This command creates a new local git repository in the current working directory.

# VERSION CONTROL SYSTEM

**`$git clone`**

This command helps you to clone a repository.
You create a copy of the original repository on your Local Machine.

**`$git fork`**

When you fork a repository, you create a copy of the original repository on your GitHub Account.

**SYNCING REPOSITORY:**

**Task:1**

First of all you have to link a remote repository with your local repository, and to do the same the command **`git origin`** is used.

Syntax: **`git remote add origin <repo_link>`**

**Task:2**

To copy all the files from the master branch of the remote repository to your local repository the command **`git pull`** is used.

Syntax : **`git pull origin master`**

**TASK:3**

To push your local changes into the central repository the **`git push`** command is used.

Syntax : **`git push origin master`**

**Configuring your repository with global parameter:**

After Creating an empty local repo using `git init` command,

You should configure your user details with your local repo to track the changes against the user details.

And for configuring the user details the commands used are listed below,

## VERSION CONTROL SYSTEM

1. Configuring user name

```
git config --global user.name “user_name”
```

2. Configuring user email

```
git config --global user.email “user_email”
```

**Git commands:**

```
$git status
```

This command prints out the current status of your local repo.

```
$git add <file_name>
```

This command lets you to stage the untracked files from the working directory to the staging area.

```
$git add .
```

This command adds all the untracked files present in the working area to the staged area by a single command.

```
$git commit -m “commit_message”
```

Lets you commit all the staged files from the staging area to the local repo with a commit message.

# VERSION CONTROL SYSTEM

```
$git log
```

This command prints all the changes made to your local repository with all commit messages. And also shows the commit ids for all commits along with the user details.

```
$git show <commit_id>
```

This command lets you show the information of a specific commit possessing that commit id.

```
$git ls-files
```

This command lets you list out all the files that have been tracked by our local Repository.

This command does not list those files which are not tracked by our local repo.

```
$git remote -v
```

## Parallel development: