Git -> Flavours - >

* Github
* Gitlab
* Bitbucket

Git is a popular version control system.

* **Control System**: This basically means that Git is a content tracker. So Git can be used to store content — it is mostly used to store code due to the other features it provides.
* **Version Control System**: The code which is stored in Git keeps changing as more code is added. Also, many developers can add code in parallel. So Version Control System helps in handling this by maintaining a history of what changes have happened. Also, Git provides features like branches and merges, which I will be covering later.
* **Distributed Version Control System**: Git has a remote repository which is stored in a server and a local repository which is stored in the computer of each developer. This means that the code is not just stored in a central server, but the full copy of the code is present in all the developers’ computers. Git is a Distributed Version Control System since the code is present in every developer’s computer.

Diagram

Description automatically generated

Download and install from [Git (git-scm.com)](http://git-scm.com/)

Create new repo

Graphical user interface, application, Teams

Description automatically generated

Access new repo

Graphical user interface, text, email

Description automatically generated

Clone repo URL

Graphical user interface, text, application, email

Description automatically generated

Readme.md for brief documentation about the project.

Basic Commands

* git branch => current branch will print on terminal
* git fetch --all => get all branches on your local
* git status
* git checkout branchName => switch/jump to other branch(already created)
* git checkout -b branch-name => create new branch (not existed)
* git pull => fetch updates from remote branch

Updating files and push to remote

* git status => check what are the added files in local
* git add filename
* git commit -m "some message about the change"
* git push

Create new branch and switch to it

* git status
* git checkout -b branchName
* try git push which will throw error message and will ask to use upstream
  + git push --set-upstream origin branch-name
  + alias command -> git push -u origin new-branch-name

Merge changes /resolve conflicts

*checkout the base branch (eg* ***develop*** *or main or master)*

* git checkout develop
* git pull
* git checkout your-branch
* (You must resolve conflicts)
* git merge develop
* git status
* git add .

rollback temproray changes and redo

* git stash
* git stash pop

Scrum, Confluence, JIRA