**Version Control System**

1. Centralized control version system
2. Distributed control verstion

**GIT**

Sourch code managment will take care of version control myfilever1 is today and myfilever2 will be the code of tomorrow of same persons’s work.

**Centralized control version system (cvcs):**

Repositry is like a folder or storage where i can put my code that how every person will commit there code to the repositry which is centerlized placed for everyone. Everyone can access and modified.

**Drawback:**

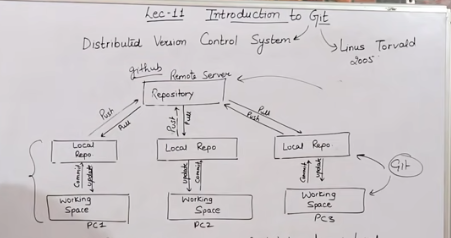
Possible to loss of remote server also need internet to use it, slow due to our the internet while update which is get back code from remote server.

**e.g** SVN tool

**Distributed control verstion:(GIT)**

linus torvold creater of git as it was linux was also developed by that linus.linux based and git is a software we can insatll in our laptop. **to keep differet version of data**

local machine have git and github is somthing about repositry storage,



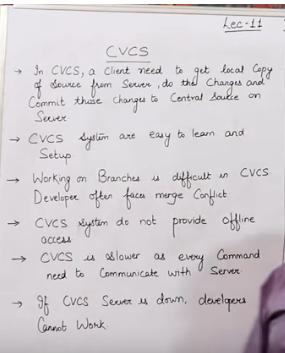
In case of dvcs we have a local Repo in our hard disk no need to connect to the internet now we have a harddisk repo in our own system now work is more fast dependency of internet is over no need of it. if repo down still no issue we also have a copy of local in our harddisk that why its a distributed version due to have multiple place of data repo in local and git hub repositry.

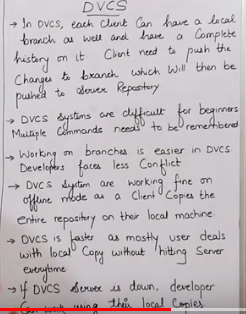
**e.g** git,mecurial

after data written on daily basis we will save like commit in local repo but if its not accessable for others due to its in your local repo of your system first it should to be in our centralized repo which is github first push it and there will be accessable to others. thats how a distribution repo is work main repo data is also available to every system of user.

very easy to move bac and fourth of code version of that day easy to change or modified anywhere of code.

**Difference between centerlized and distribution:**





Github and gitlab are the service for central repositry

**Three stages of Git and its terminology**

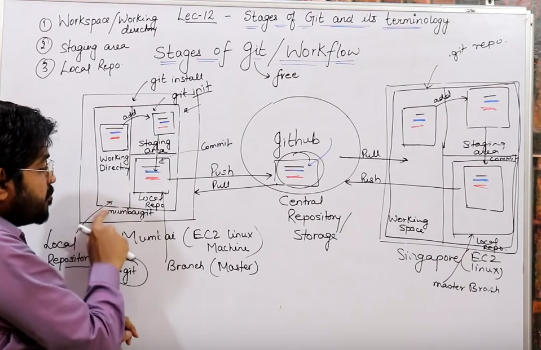
Git in ec2 linux machine ,git is free but github some services are free after git create a directory in our machine in directory run a command git\_int hidden file will be created and that dir will convert into the Repositry three things will be made logical local repositry will be made in your system git will convert into repositry and we have three stages workspace/working,staging area,local repo. code are in working dir jha hm kam kr rhy hoty hai also called workspace. Staging area adding into stagging area after written in working dir which is called add now we canot change instead of direct commit first staging. lets we have three code index.html temp and index css and we wish to add only first two in commit we have all in staging area now we wish to add which thing is to finalize and which is to commit or not all the things are placed in staging are before commit its like we have a snapshot. commit snapshot we wish to add into the local repo ,commit id for retrive in future use also have tag with it.

working in workspce area what ever the code is or modification required will be in that area after we can keep it staging area like all the req gridents collection then we will save or commit in our local repo.

**Github:**

Centeral Repo/Storage by Microsoft have public and private

(Gitlab is open source) after all the completion we will push all the data into the Github after creation of account. Now data is save in our **local system repo which is (master branch)** and also in Github which is also accesable to others.  
all other will **pull** from the central repo (Github) and after update there data of that day they will push back and update the repo. Everyone have there own commit id with different color to check every individuals work.



**Repositry:**

Repositry is a place where you have all your codes or kind of folder on server.

**Server:**

It store all repositry. It contains metadata(file related data commit id,)

**Working Direcory/workspcae:**

Its like a rough space where all modification has to be done.

Working Directory-->Staging area-->local Repositry

only staging area files are considered to be ready for commit and not all the modified file.

**Commit:**

Store changes in repositry you will get and commit =ID, named as SHA1, checksum value (whenever the changes has to be done in an file then the checksum value will be changed) integrity is maintained.

It actually help to verify which modification has been made.

**Tag:**

I have changes in that particular thing in that code which is helpful through tag. assign a meaningful name for the specific version for a particular commit.

**Snapshots:**

Represent some data of particular time . it is always incremental

whenever we commit it will creat a snapshot and after update again we will commit then again snapshoot willl be created. jis time mai picture li gai hai bs wahi data hai. new data add/ append adding something new.

It store only append data or something new changes have been made.

**PUSH:**

On remote server push operations copy changes from local repositry on your system to remote server central server.

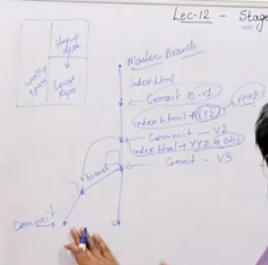
**PULL:**

To get back data into your local machine is called pull all the data of that repositry will be copied into your system and make a exact copy of remote server and local server on your machine and both of them will be in syncronized form like your Indian friend and your pakistani machine both systems have same data code file in syncronized form.

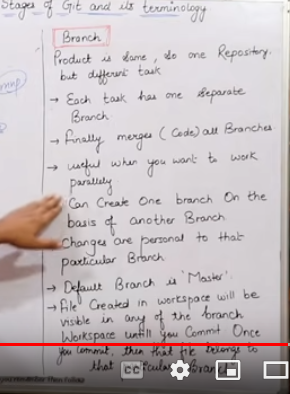
**Branch:**

Master branch after creation of creat is a by default if we change in our workspace and staging area commit to matster branch which is commit and have commit ID in future if we update in that same file then it will be index.html then +xyz now commit version is 2 again add something and commit we changing in our master branch now its a version 3 file index.html +abc +xyz if we change of plan in past version. We will create a new branch as a copy of that file made any modification possible and commit but still we have an orignal master branch file. Also possible to merge that branch and share with other free members to share that branch and have some practice.

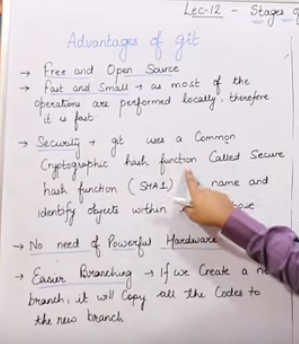
In one time multipe task at a time for multiple persons we will use branch



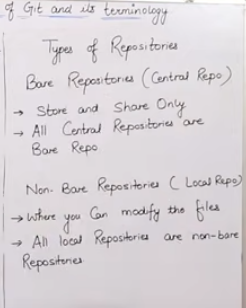
In FB application we have to made feed page status handling notification etc all of the individual task will be assign to individual person by assiging them a branch then we will merge together to make a Fb application.



**Advantages of git:**



**Types of Repositry:**



**Git installation+GitHub:**

AWS account create two EC2 Instance (Linux), allow port ssh and HTTP

and access through putty and login

#sudo su

#yum update -y

#yum install git -y

#git --version

#git config -- global user.name “ansar/shahbaz”

#git config --global user.email [“ansarshahbaz5881@gmail.com”](mailto:\“ansarshahbaz5881@gmail.com\”)

#git config --list

#which git

bumbai and singapur two machine in aws

lunch instance-->amazon linux -->2 AMI t2micro -->security group create-->http ssh-->anywhare -->keypair new -->lounch new instance,IP4 public ip copy--> puttygen Download all files will show it-->key file we will save in putt public ip -->ssh -->auth browse select downloaded file

#ec2-user

Run all the configuration on both machine of AWS

**Github account creation:**

[www.github.com](http://www.github.com)

name tech4u

[ansarshahbaz5881@gmail.com](mailto:ansarshahbaz5881@gmail.com)

passwd

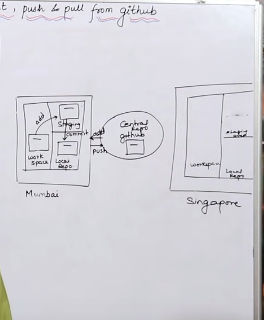
create your account

join a free plan

Student none learn to code complete setup

**How to commit, Push and pull from Git:**

we have to EC2 instance in mumbai and singapure. workspace,stageing area and local repositry in both of them add is from workspace to staging area and commit is from staging area to local repo. then attach it with central gitbub to central repo we will push from local to central repo.



Now anyone can pull from central and can move to further move on. add other person to the github central repo he will pull a file and sync with it.

create a directory

git int (directory turn in to repo)

creat a myfile for testing

git status (kya para hwa hai us mai)

git add.(dot will add workspace to staging area)/ git add filename

commit into the local repo

git commit -m “1”commit from mumbai

git status

git log(commit kis bndy ny kiya hai with its name and email)

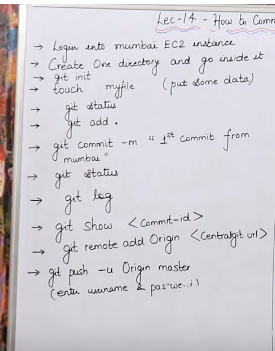
git show(kya code likha hai is bndy ny)

git show(commit-id)

git remote add Origin(centralgit.url)

git push -u orign master(by default branch is a master branch )

enter username and passwd



#Which git

#mkdir mumbai git

#ls

#cd mumbaigit

#git init

#cat >filemumbai1

#i am writing this code for testing

#ctrl d

#git status

#ctrl l

#git add .

#git status

#git commit -m “my first first commit from mumbai”

#git status

#git log (commit id of 40 character)

author with email id

commit date and time also in it

#git show some character from 40

send code to central repo

github open use user name and passwd

create a central repo in github

creat new repo name it describtion public create repositry

copy its url and in your machine

git remote add origin url

your local is attached with central

git push -u origin master

ask username password

cat>mumbai

ksjdfhask

ctrl d

git status

git add .

ctrl l

git status

git commit -m “my 2nd code from mumbai”

git status

git log

git show commit id

git push -u origin master

username passwd

refresh in github and verify the update parts

history will alsoshow in github

**2nd machine:**

Singapure EC2 instance

mkdir singapuregit

ls

cd singapuregit

git int

ls -a

git remote add origin url(central repo) at code click we have url

#git pull origin master

#git log

#git status

#git show commit id

#ctrl l

#cat >>filemumbai 1

#hume un sy hai wafa ki umeed

#ctrl d

#cat mumbai 1

#git status

#git add .

#git status

#git commit -m “commit from singapure”

#git status

#git log

#git show commit id

#ctrl l

wish to push data to central

# git push origin master

username plus passwd

verify into central github repo

git pull origin master every morning before start your work

#git show commit id

**Conclusion:**

git add / git rm

git restore (to make ampty staged area)

git commit -m “massege”

git push -u origin master

**Git ignore:**

for different file formats we wish to ignore some specific differen format files we will ignore that one. Specific files ignore nott wish to commit in local repositry. create a hidden file first .gitignore e.g \*.css , and add that folder to commit. git add commit that file and

#git add .

#git status

**LAB**

vi .gitignore

press i to insert \*.css file and .\*java esc and :wq

git add .gitignore

git commit -m “ignore files”

git status

touch file1.txt fille2. file3.java file4.css

ls

git status

git add .

git status

in green color means added

git commit -m “ignore files”

touch file ansar.java

git status

touch ansar.txt

git status

ctrl l

git log -1 latest commit

git log -2 latest last 2

git log --oneline

if you wish to specific commit with word

git log --grep “ignore”

this is just for testing.

**Git revert,remove:**

**Git revert:**

help you undo an existing commit.

delet from staging area or before commiting we use git reset or i have commited now how its possible. galat code ko commit kr dya to only solution is revert. commit ko delete nhi kr skty ho gya ho gya but we can revert every commit have a commit id. It does not delete but return to previous state. Revert will add new commit id.

#git revert <commit-id>

**How to remove untracked file:**

#git clean -n(dry run)

#git clean -f(forcefully)

clean is used to remove unused file in git

**Tag:**

Tag operation allow giving meaningful names to a specific version the repositry. through tag we can track specific commit id but unable to remember in the bulk of commit IDs.

#git tag

to see all the tag

#git show tag name

**Git Clone:**

add,clone different in nature. clone means central repo have a file and clone file same copy into your system. cloning of any file from central repo to into your own system.

#git clone url <central repo url> automatically create a local repo into your own machine with same name.

-->Master Branch , branch A, branch B, we will merge branches to the master branches

pulling request for to overcome the issue of conflict.

**LAB:**