MERN Stack Training

Contents

* Applications
* Fundamentals of Programming Languages
* Algorithms & Pseudocode
* Linux
* Git
* HTML
* CSS
* Javascript
* Babel, Webpack
* React.js
* Node.js & Express.js
* Testing
* MongoDB

Application: It is a computer program that can perform some task, there are two types mainly

1. Standalone or Desktop
2. Web or Distributed application

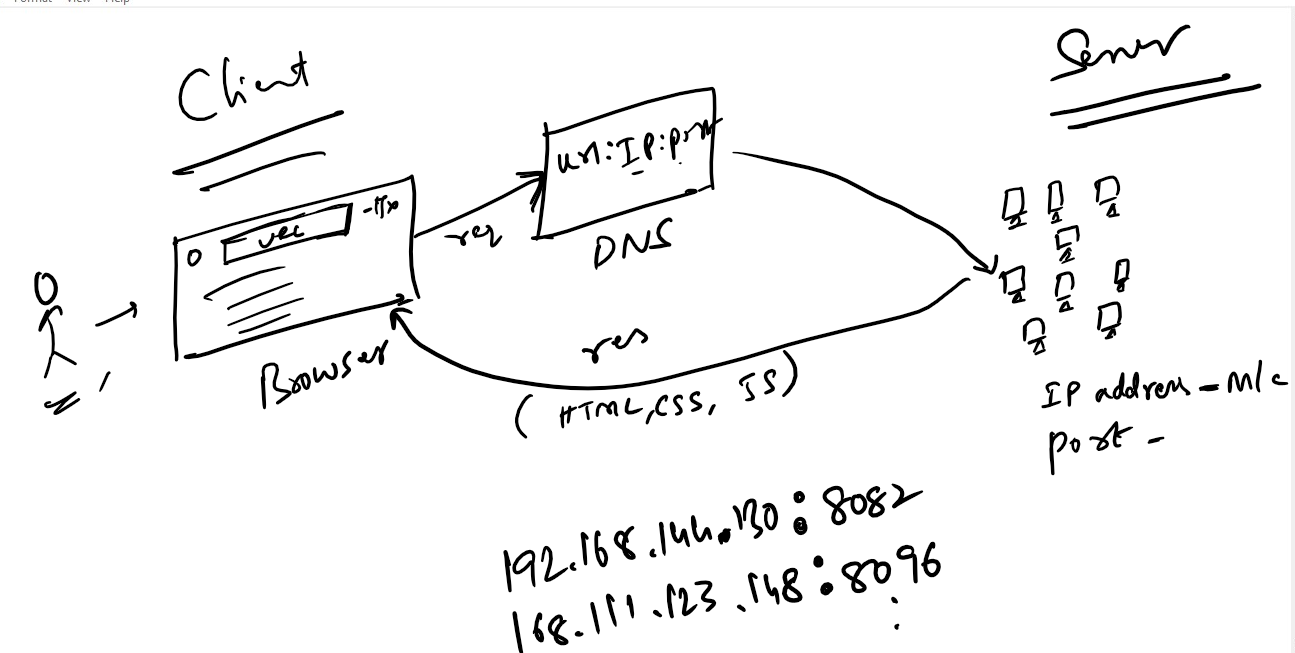
Standalone: You can use this application only after installing on your machine

ex: MS Word, MS Access, Browser, Mobile applications (G Pay, Phone pay, Uber)

Web / Distributed applications: These applications can be accessed over the internet without installing on your machine.

ex: Bank Applications, Gmail, Twitter, Ecommerce and son on

How you can access these web applications



Web applications are accessed through browser by entering the URL, each applications will have IP & Port, Since user wouldn’t remember IP & Port the DNS will have an unique URL for each applications, when user enters the URL the DNS will map the IP & Port of the applications & sends the request, the request is processed & response will be sent to the browser, because browser understands HTML, CSS & Javascript the response will be in those formats by default.

Web applications vs Web pages

Web applications can produce web pages, web applications can handle request & generate the response, these response are shown in web pages

Web applications are of 2 kinds which produce static pages & dynamic pages

1. Static pages: These contents doesn’t change, it is common to all the users

Ex: Wikipedia, Google search engine

1. Dynamic pages: These contents change at runtime, it is different for different users

Ex: Twitter home pages, Facebook news feed, Gmail, Ecommerce websites

Technologies used for static & dynamic pages:

Static pages can be created using HTML & CSS

Dynamic pages can be developed using various technologies like Javascript, Servlets, JSP, PHP, ASP.NET, Django

Various programming languages helps you to develop dynamic web applications

1. Java: Servlets & JSP
2. Javascript: Node.js & Express.js
3. C#: ASP.net
4. Python: Django

Client & Server

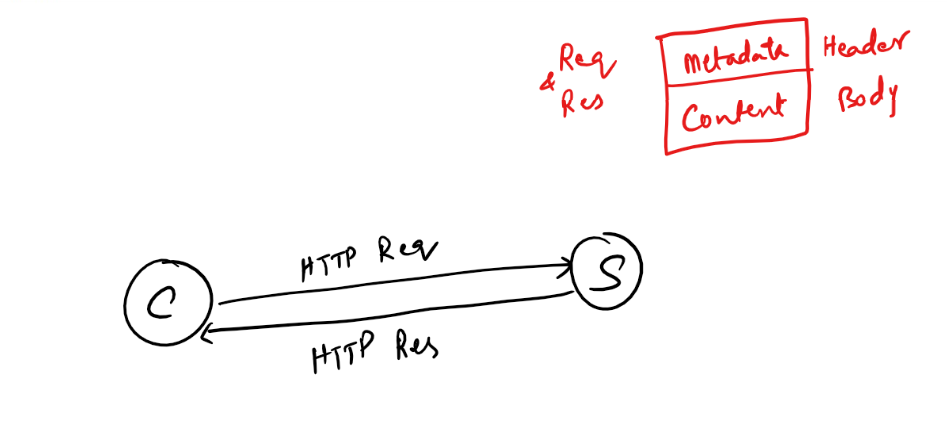
It is an architecture used to communicate from one program to another program

Client sends request through URL & Server sends response to the client

URL: Uniform Resource Locator, It is used to access any web application over the internet, when you make a request the client sends HTTP requests & receives HTTP response

HTTP: Hyper Text Transfer Protocol, Client & Server uses HTTP to communicate

HTTP Request & HTTP Response are messages which will some information

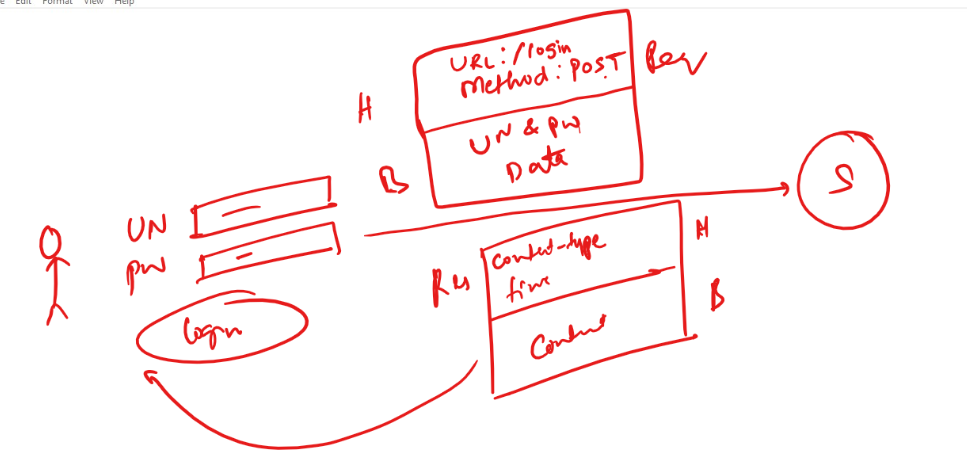


Request & Response will have header & body, header will have the metadata i.e., properties of the request & response, & body will have the content

Request Header: Will have properties like URL, type of HTTP method, accept type, content type and so on

Response Header: Will have properties like content type, time, content length

Body: It will have the content



History of Browsers

1. WorldWideWeb Browser
2. Mosaic
3. Internet Explorer - Replaced with Edge
4. Opera
5. Safari
6. Mozilla Firefox
7. Google Chrome

Distributed Applications: These are the applications that can be accessed over the internet, we have web applications & enterprise applications that can be accessed over the internet

Enterprise applications are accessed by various types of programs/applications like mobile applications, desktop applications, swiping machine applications, ATM machine applications and many more (these are all client applications that can access enterprise applications).

Web applications are accessed only by Browser, however through browser web applications would access enterprise applications

Session Agenda

* Various Technologies in Full Stack Program
* Algorithms and Pseudocode

Main technologies

* HTML, CSS & Javascript
* React.js
* Node.js
* Express.js
* MongoDB

Application is divided mainly into two layers

1. Front End - User Interfaces which is an application or program that user uses to access the Back End Services
2. Back End - It will have programs that can talk to the Databases

Backend Services: It will have business logics & database logics that help any front ends to communicate, most widely used languages for Backend Services are:-

* Java, Python, Javascript, C#

Front End Applications: It will have the logics to show the User interfaces to the end user, these applications will communicate with the Backend services on behalf of the end user, most widely used languages for Front end are:-

* Javascript, Java (JSP), .NET (ASP)

Javascript:

It is used to create both front-end & back-end applications, to run javascript we need Runtime Environment, for Front end we have Browser to run Javascript, for Backend we need Node.js to run Javascript.

In Our Full stack development we have list of technologies for Front end and Backend

Front end

* HTML
* CSS
* Javascript
* React.js

Backend

* Javascript
* Node.js
* Express.js
* MongoDB

Algorithms, Flowchart & Pseudocode

Algorithm:

* It is a step by step English instruction written to solve a particular problem
* It is used at the beginning to understand how to write a program to solve the problem
* It doesn’t have any rule that you need to write algorithms in a particular fashion

Flowchart:

* It gives humans a graphical representation of the flow of the problem solution

Pseudocode:

* It mimics the programming language, but still it is not understood by any compilers
* It is independent from any programming language
* It specifies what all the programming constructs you can use in your programs like conditional blocks, loops and so on.

Adding two numbers

Algorithm: You can write algorithm in your own way

Step 1: Start

Step 2: Read two numbers and add them

Step 3: Store the result of Step 2 in one variable called Result

Step 4: Print Result of Step 3

Step 5: End

Another way of Algorithm for adding 2 numbers

Step 1: Start

Step 2: Read first number

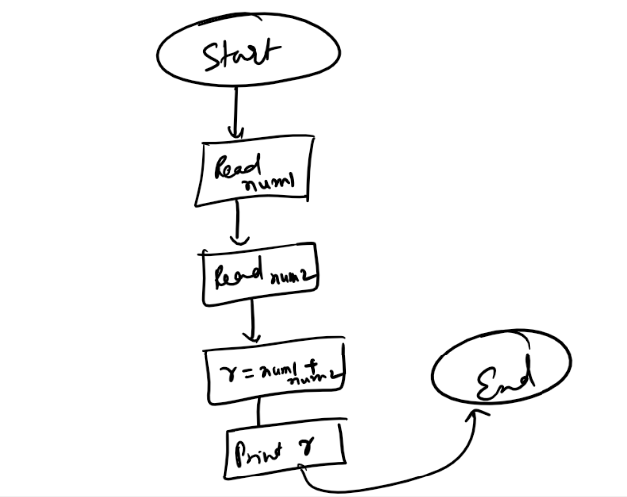
Step 3: Read second number

Step 4: Add Step2 and Step 3 numbers

Step 5: Print result of Step 4

Step 6: End

Flow Chart for Adding 2 numbers



Pseudocode

Since it mimics the programming language you can use below pseudocode to add 2 numbers

begin

read num1;

read num2;

result = num1 + num2

print result

end

Activity:

Write an algorithm and pseudocode to calculate area of a circle.

Note: Write in notepad

Linux:

It is one of the popular server OS, used to deploy many applications & install servers, almost all the applications we use are running on Linux type OS

Linux follows Unix feature, since Unix is not free we are using Linux which is Open Source.

Open Source: It enables users to customize or make changes as per the organization need

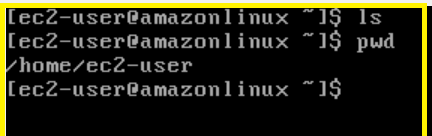
Linux comes in 2 flavours

1. Terminal based OS: Meant for hosting servers & applications
2. GUI based OS: Meant for users who needs UI, like Ubuntu, Fedora

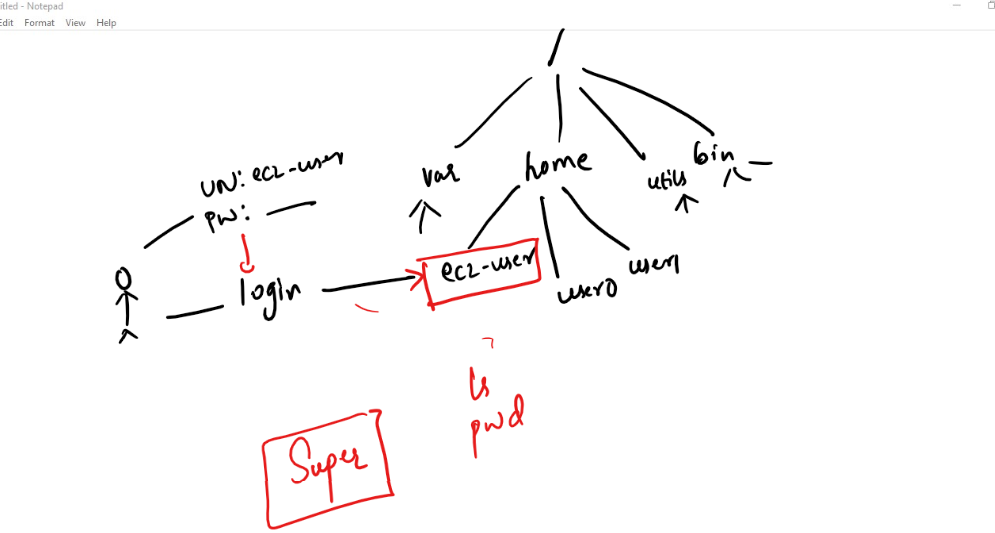
Development & Testing happens on GUI based OS’s like Ubuntu, Mac, Windows, however deployment of the application happens on the terminal based OS like Unix or Linux

In order to work with Linux you need to understand the commands it provide, you must able to do everything through commands provided by linux, these commands are predefined, which means you need to enter it as it is defined.

ex: date: if you enter this command its shows date, it is predefined, if you make a spelling mistake it doesn’t work, because it is probably not present in linux, ex: if you type dte instead of date it may not work



Once you login you will be inside user folder which is part of /home



List of commands in Linux

Online Linux terminal

<https://www.tutorialspoint.com/linux_terminal_online.php>

For Date & Calendar

$ date

$ cal

$ cal -3

$cal 2018

$ cal -m Jan

Creating file in terminal

$ touch file\_name.txt

$ ls

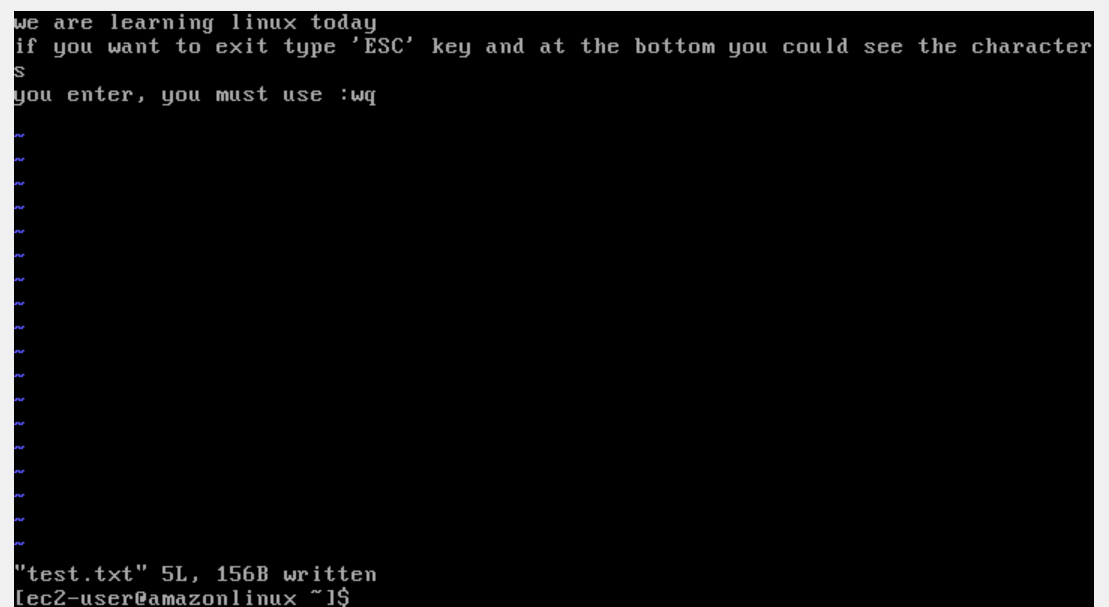
Will list the files & folders

Editing file

$ vi file\_txt

You must use INS key to insert data and if you want to exit you must use ESC key and type :wq

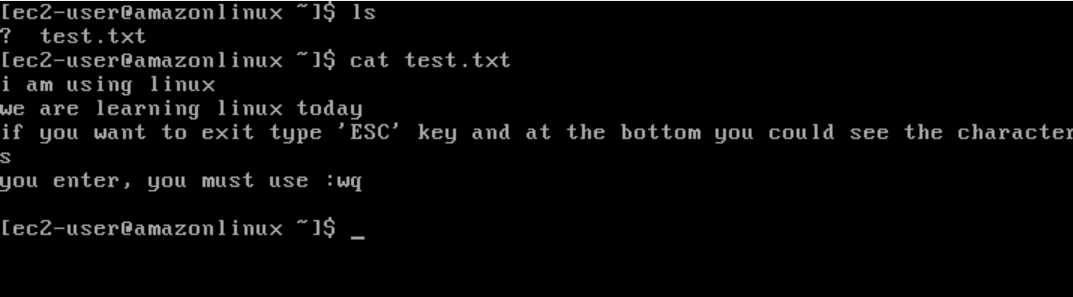
Note: :wq appears at the bottom of the terminal



To view the content of the file

cat filename.txt

$ cat test.txt



Navigating from one folder to another folder in Linux

cd: It is the command used to navigate from one directory to another

cd *path*

Some list of cd commands

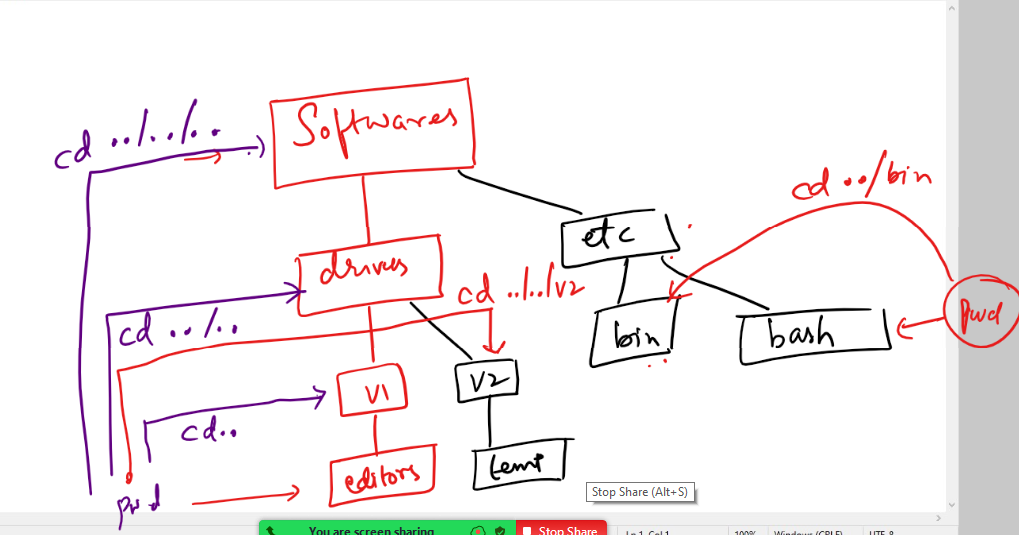
**cd ..** Navigate to the parent directory from the current directory

**cd ../..** Navigate to the parent directory of parent directory from the current directory

cd..

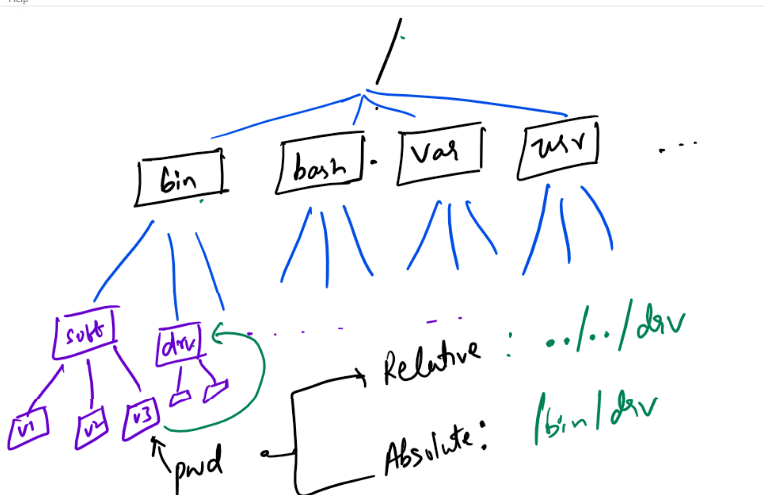
Suppose you are in the path **softwares/drivers/v1/editors**

Then you can use cd.. to get into **softwares/drivers/v1** folder



All the above commands are using relative path, because it is using cd to recognize the destination path from the current directory

Absolute path: This also navigates from one location to another, but it uses the path that starts from the root, doesn’t matter what is the pwd



Creating multiple files

touch 1.txt 2.txt 3.txt 4.txt

Rename the file

mv 1.txt one.txt

Copying the file

cp one.txt 2.txt

Deleting the file

rm 1.txt

rm -r folder\_name

You can use rm -r \* that removes all the files & folders of the specific director

grep (Global Regular Expression Print) command

It is used to search a pattern in a file

grep hello 1.txt

grep -c hello 1.txt

grep hello 1.txt 2.txt 3.txt 4.txt

Shell Script

It is a script file with linux commands and some expressions to simplify writing complex commands, you can create shell script to run linux commands in a shell script file with an extension .sh, the script file can have any linux command along with that you can also have some expressions like reading inputs, operations on those inputs like add, sub, and so on.

abc.sh

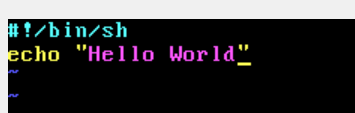
*#!/bin/sh  
echo “Hello World”*

The abc.sh has 2 lines of code, the first line is mandatory for script to know from where it needs to be run, the second line is just a print statement that prints in the terminal

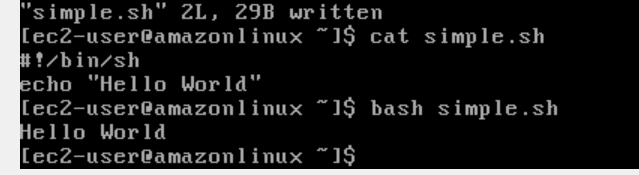
How to run the script

*bash abc.sh*

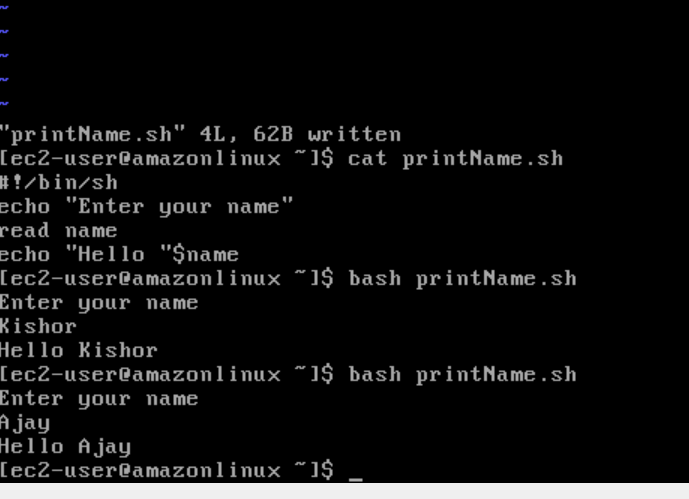
simple.sh



Execution: bash simple.sh



Reading the input and printing the input



Git

Git is a Centralized Version Controlling System / Distributed Version Controlling System which helps to collaborate everyone’s work within the project, it tracks every changes done by every author/user that helps GIT to identify who did what changes at any given point of time and also to go back to any time when the work done by any user is not right

Branch: It is a pointer that will have all the histories of work user has done, you can navigate to any history to see the changes happened at that point of time, by default GIT provides a branch called master/main branch which will have atleast one commit id.

Git uses many commands to save someones’ work, upload their work or download someone work, they are:-

* git commit: It is to save the work
* git push: It is to upload the work to remote repository
* git pull: It is to download the work from remote repository
* git clone: It is to download the remote repository, its done for the first time only in the local machine

Repository: It is a folder that will have some work, it will have history of commits so that you can see the changes done in any time

How to create Repository

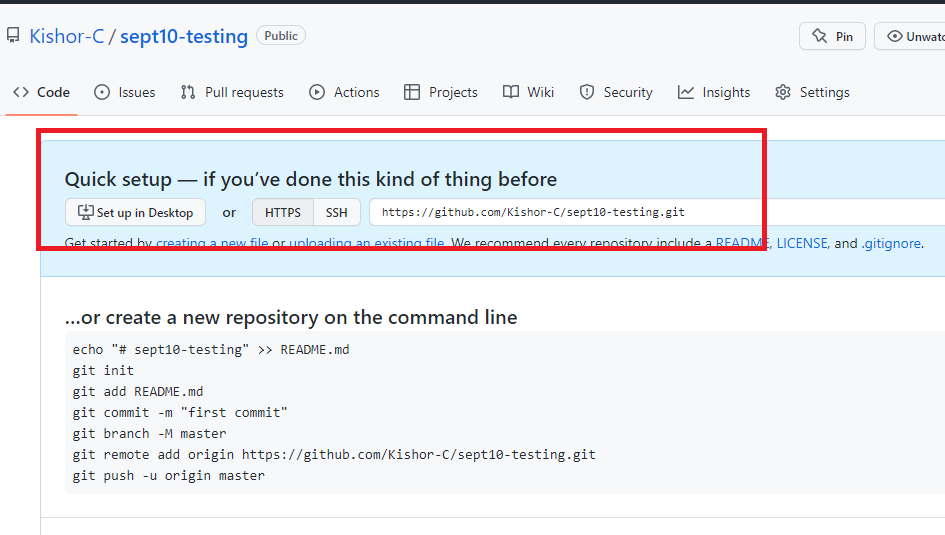
There are two ways

1. You can directly create in remote server (Remote Repository) i.e., Git hub & let all others download in their local machine (Local Repository)
2. You can create in local machine and push to the remote server

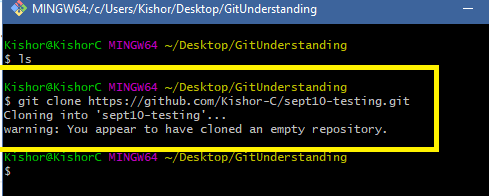
Creating the repository in remote server i.e., Git hub

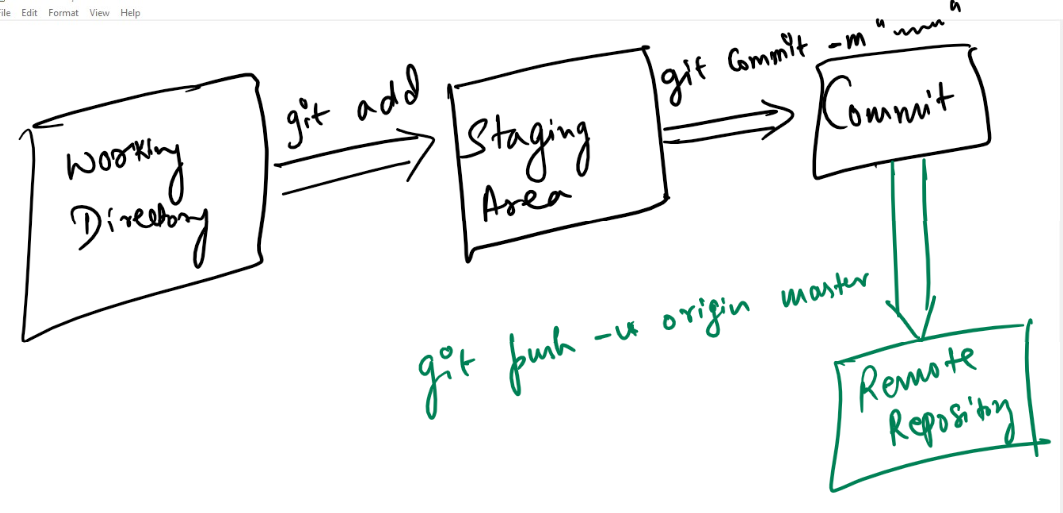
1. Login to Git-hub
2. Identify the button called New, that helps to create a new empty repository
3. Give a name to the new repository and click on finish
4. Observe the list of commands the page shows
5. use git clone <<url>> that will download your empty repository in your machine

You would notice some list of commands after repository is crated like this



Open Git bash in your machine in some folder and type git clone <<url>> this clones the empty repository in your local machine, which is called as local repository.





git clone <<url>>

git add filename

git commit -m ‘some description’

git push -u origin master

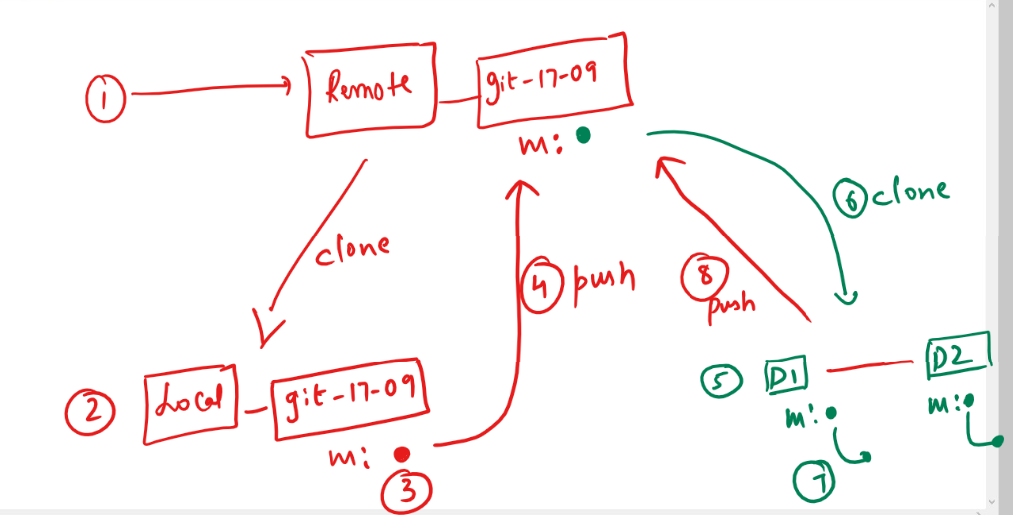
Agenda

* Git commands
* branch
* pull
* merge
* Resolving conflicts
* Git Organization

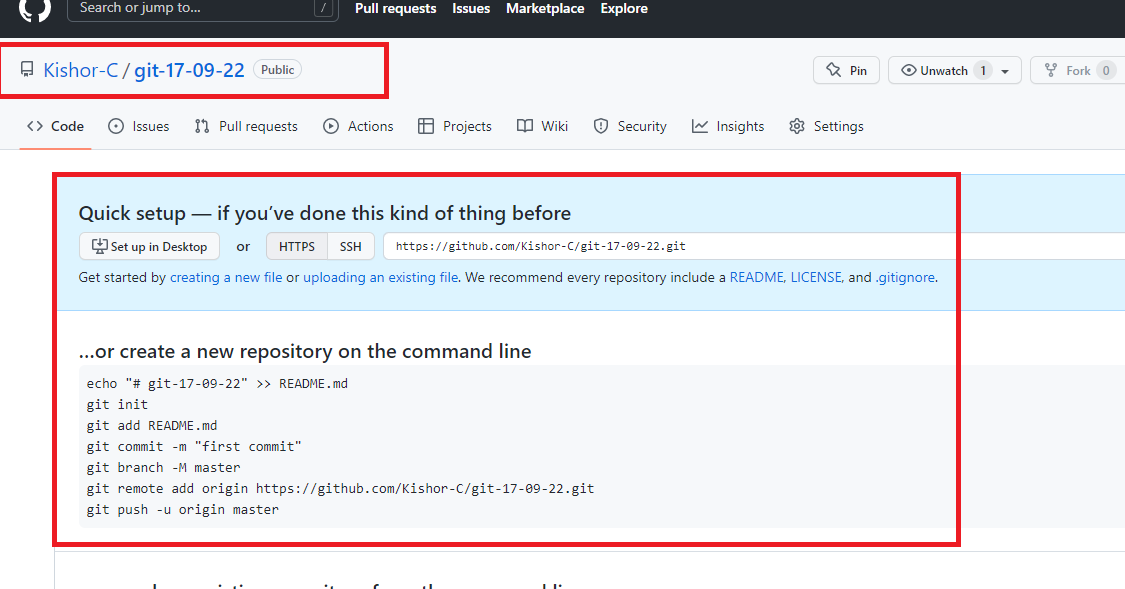
Some important points over main/master branch

1. It is like a original copy of your software with the latest changes
2. Nobody should update master branch directly either in the local repository or in the remote repository
3. When a new user wants to use the project he always clones master branch
4. When a project needs to be released the company uses the latest changes done in the master branch
5. When a project needs to be tested, then testing team also uses master branch

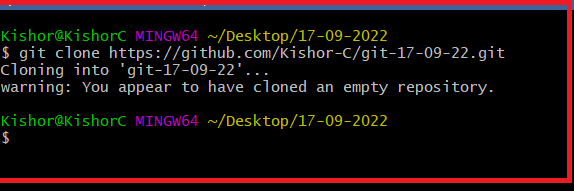
Working with all the important git commands starting from the new repository



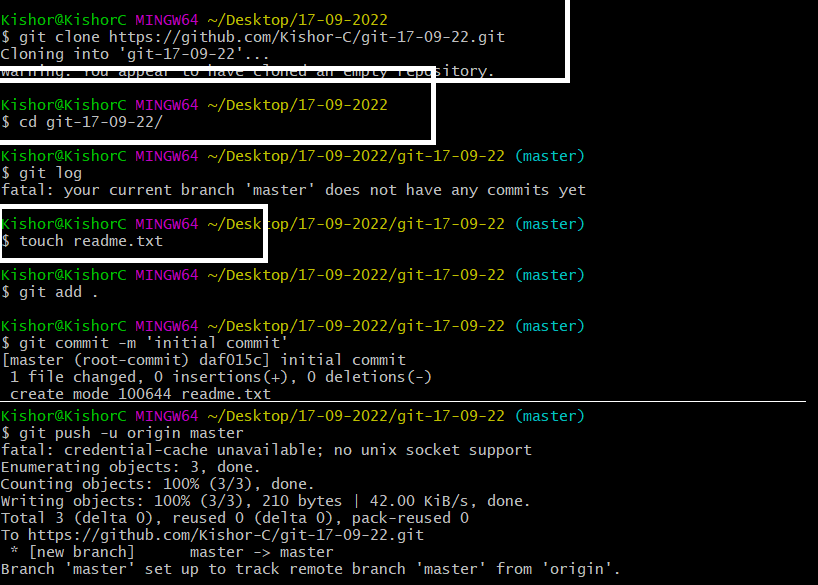
Create a new repository in your git hub account with a meaningful-name which you can easily recognize.



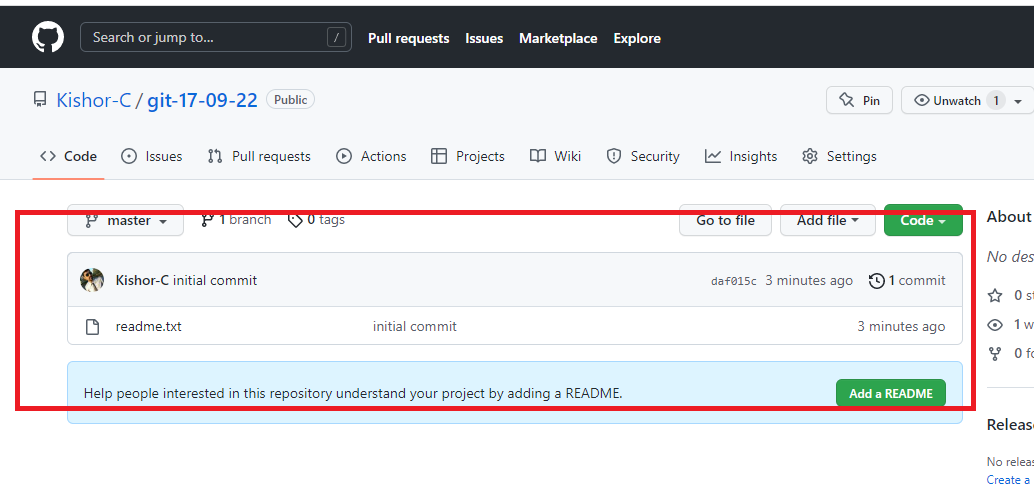
Step2: Clone the repository into your local machine and navigate to the repository to make your first commit to the master/main branch



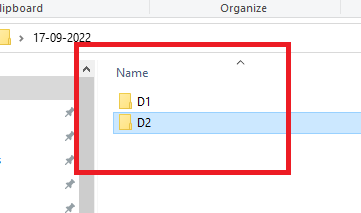
Note: Use cd and navigate



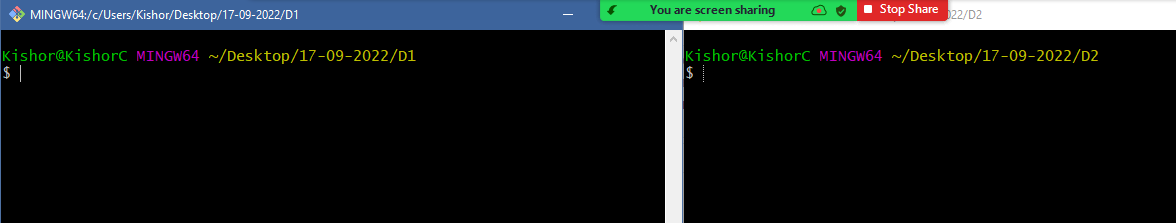
Now you must see the first commit in the remote repository on refresh



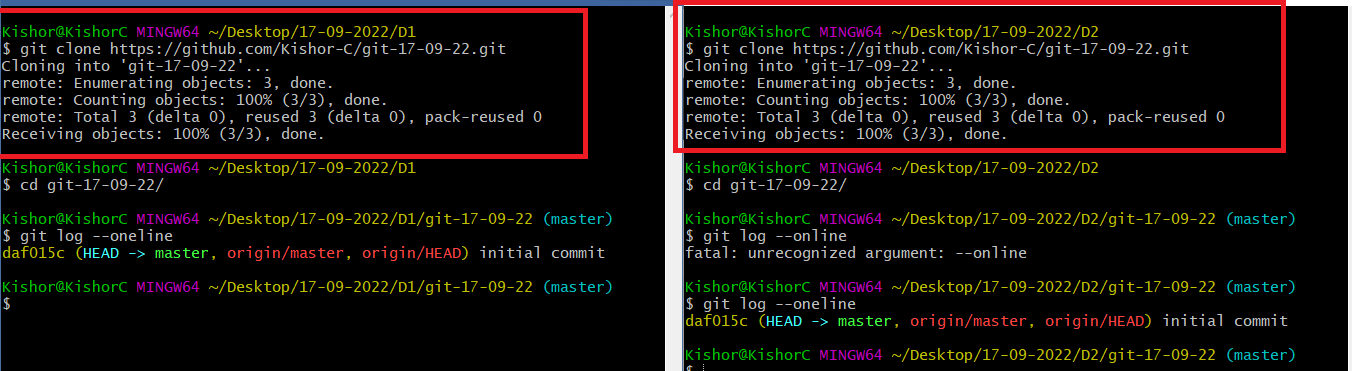
You can exit from that old terminal and delete that repository so that you can work with 2 folders like a 2 users i.e., D1 & D2, the reason to delete the old folder is to avoid confusion.



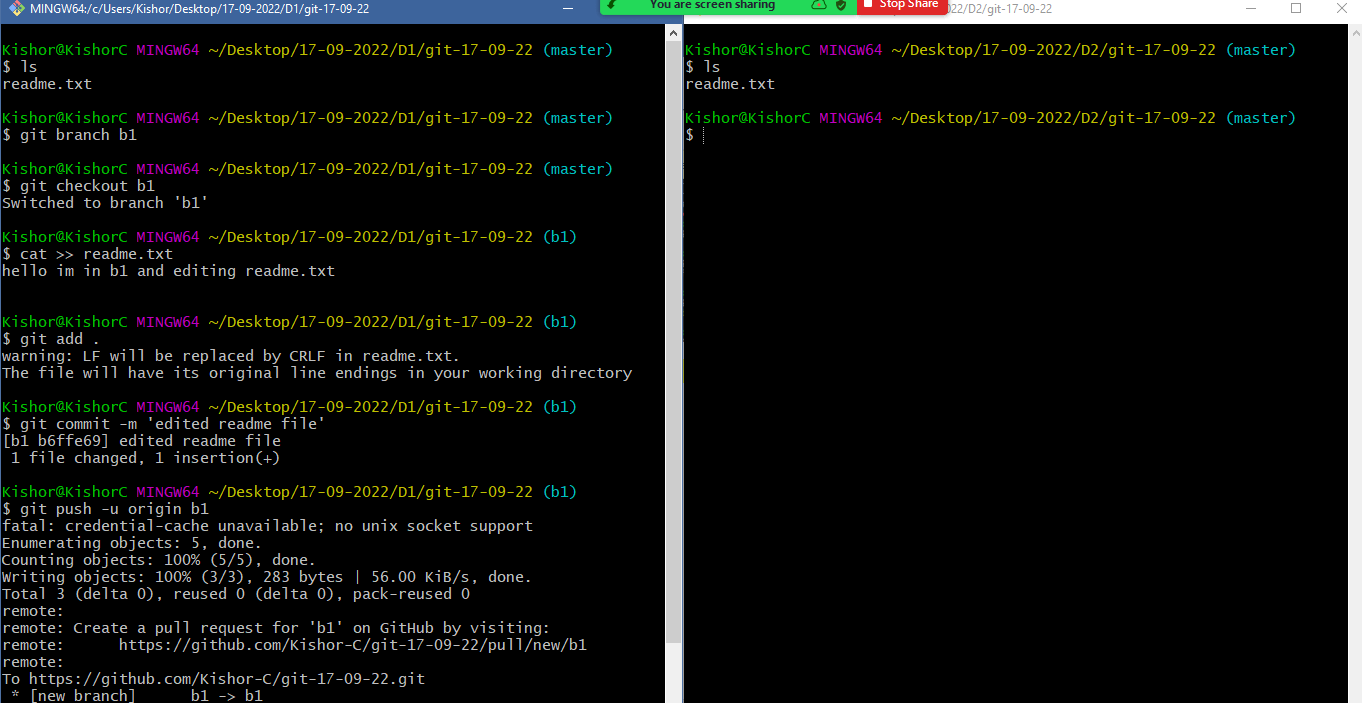
Open terminals in D1 & D2 so that you will have 2 terminals one inside D1 & other inside D2



Clone the same repository in both the folders

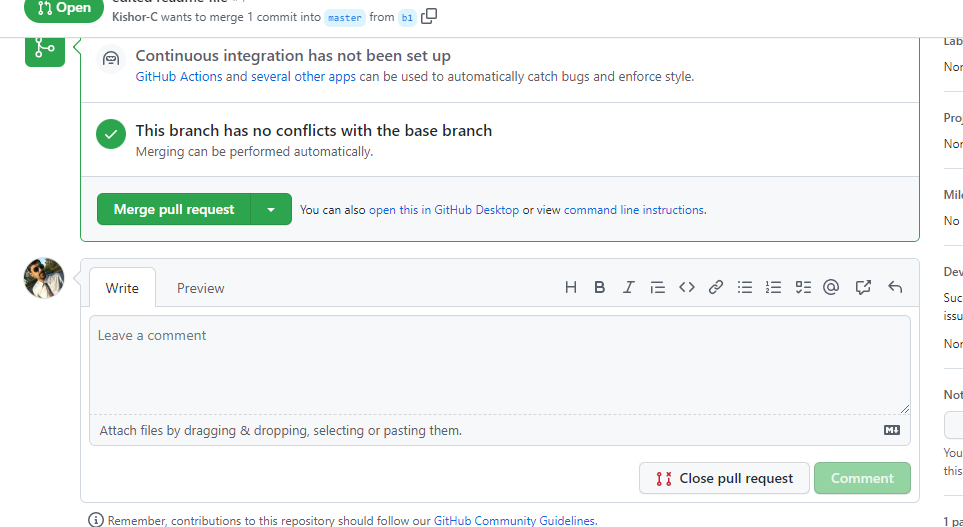


Create feature branch in both the folders and edit the readme.txt in D1 & push to the remote first



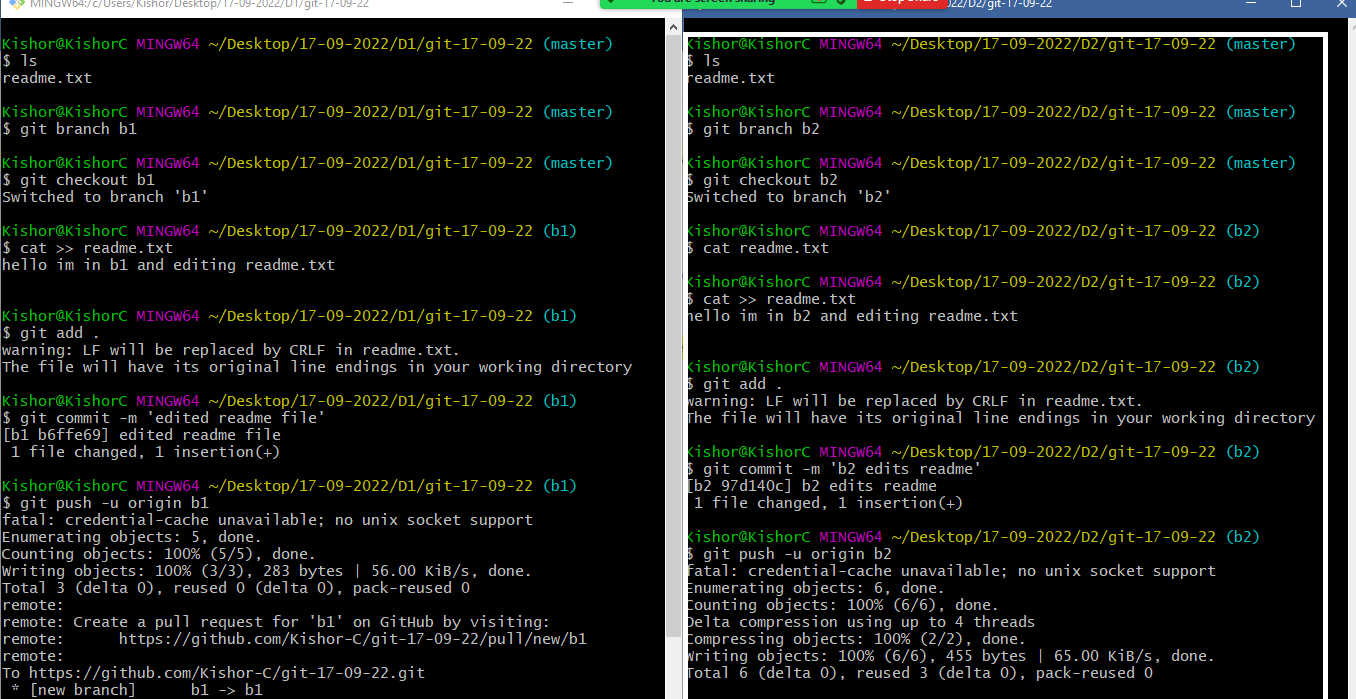
Notice that you will see a pull request in the Git hub

Just click on that pull request and merge so that you will see the remote master merging the feature branch without any problems.



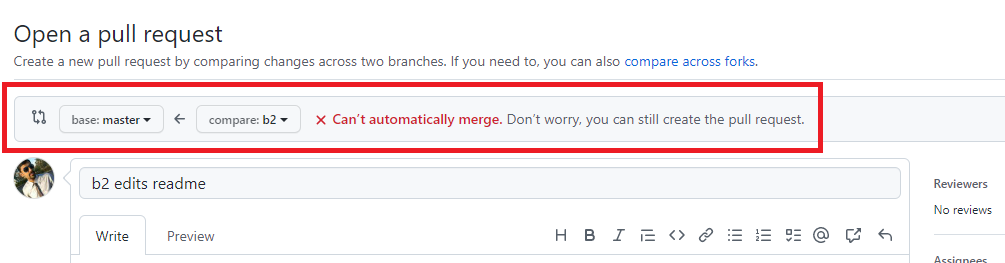
Note: You can delete the feature branch once merged

In D2 terminal edit the same file in another feature branch



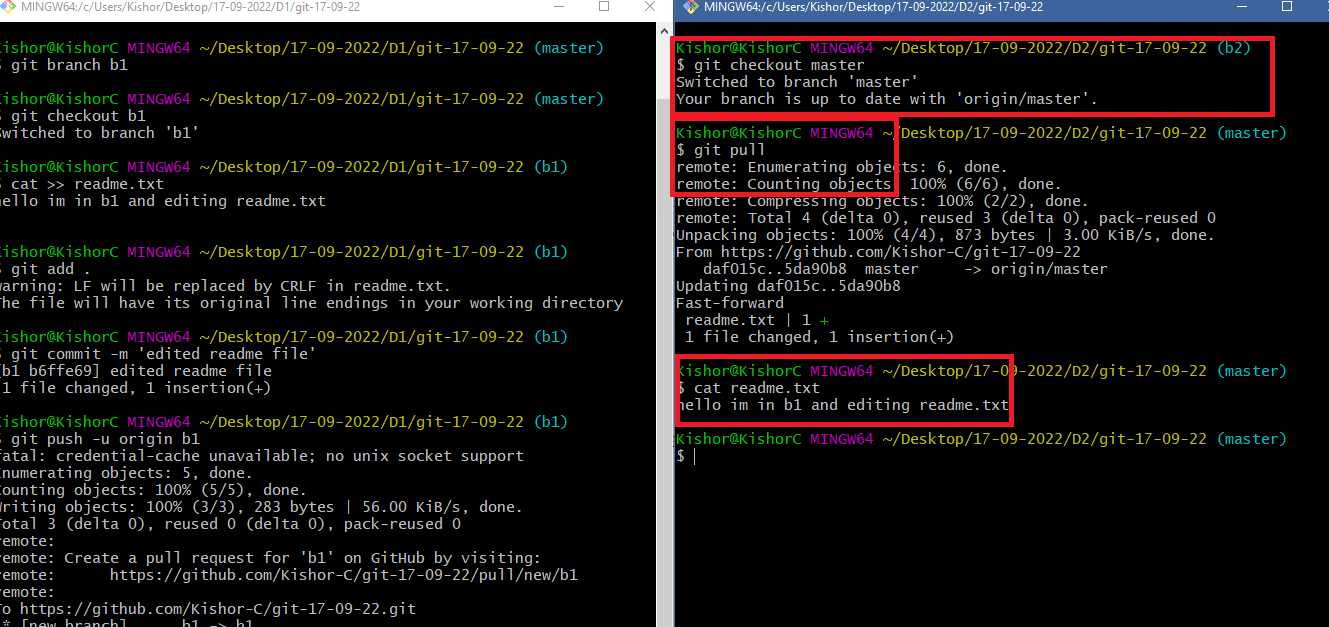
You will again see compare & pull request

When you click on it you will the error can’t merge



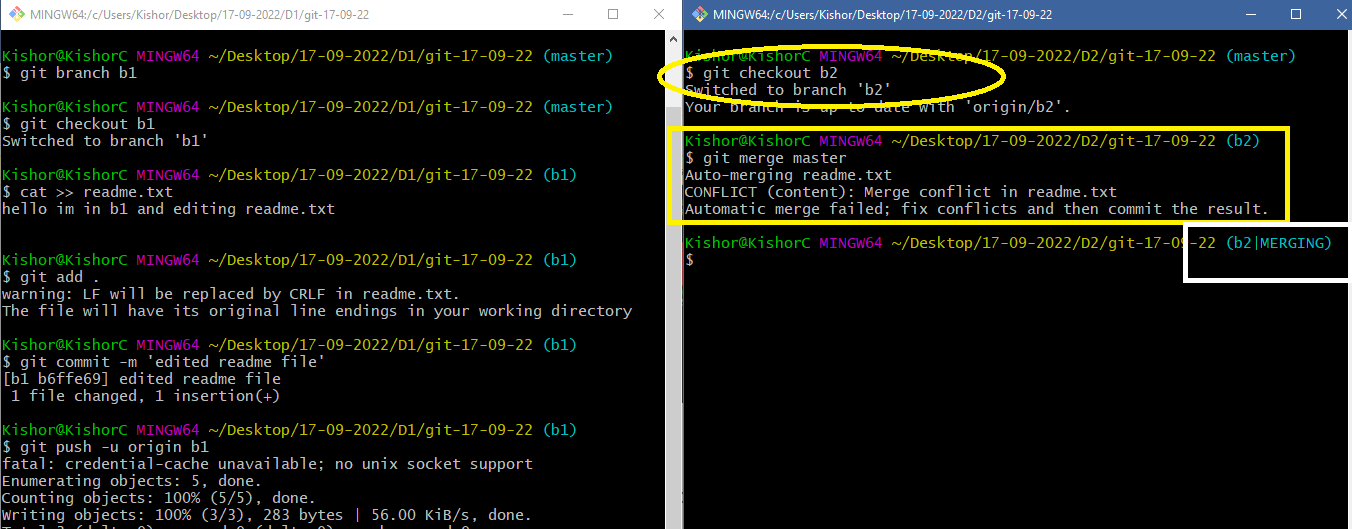
This error is because remote master branch has some changes which this new feature branch doesn’t have

Now you need to close this pull request in the Git Hub and delete the feature branch and then in the D2 folder we need to merge and resolve the conflict manually



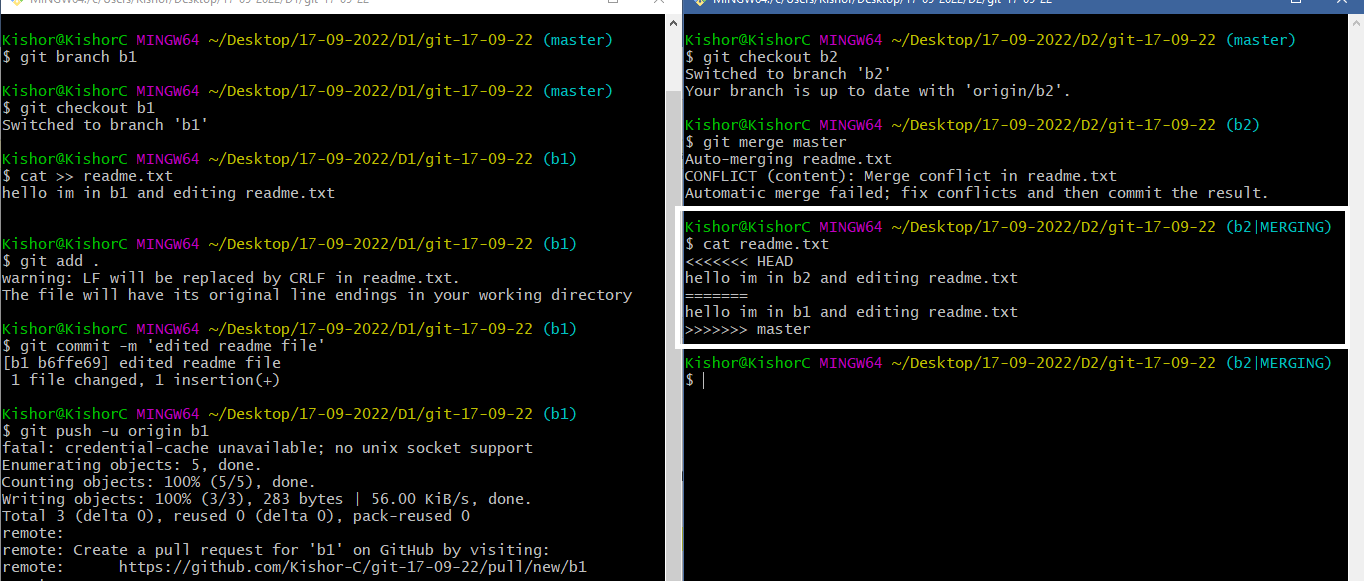
Note: Now the local master is updated with the remote master changes

Here the D2 folder has switched to master to view the changes in readme.txt, but you need to switch to b2 and merge the b2 with local master.

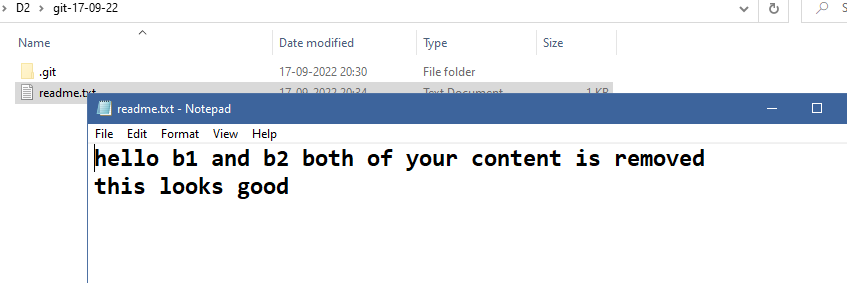


Since there’s a conflict in merge it creates a new branch till you enter the commit i.e., b2|MERGING

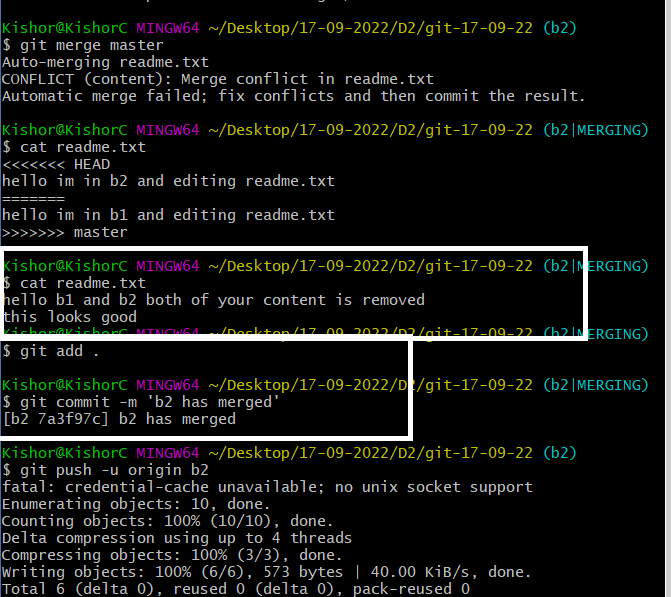
You can observe through the cat command the content of readme.txt



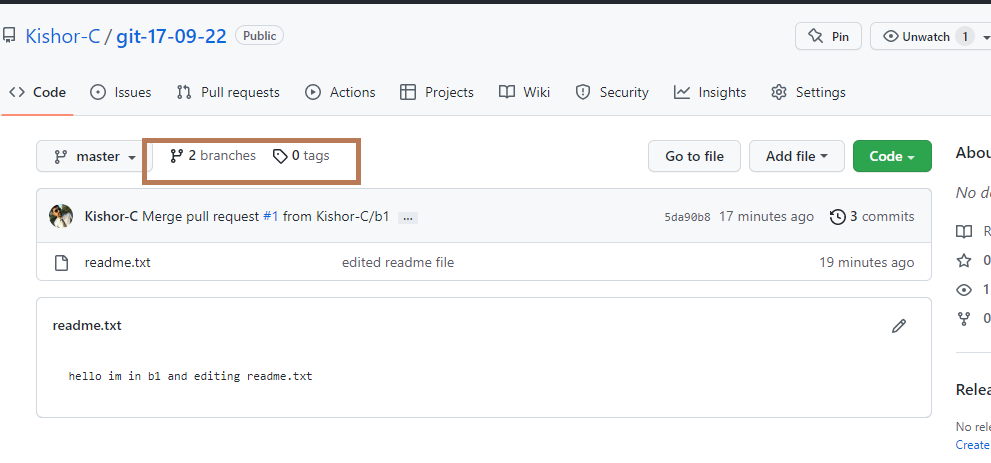
You can open the file in any editor and edit the content



Now you can use git add, git commit & git push -u origin feature-branch



Now you can see in Git hub the pull request, if not click on the 2 branches there you can open pull request

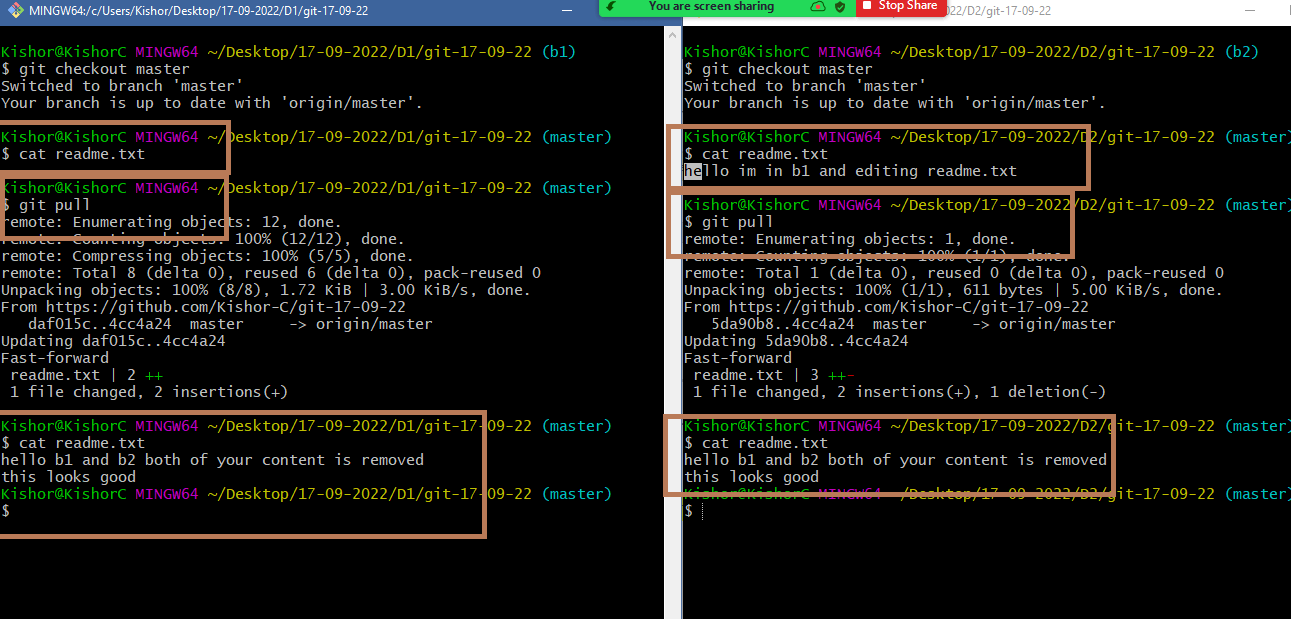


You can open pull request & try to merge, since the D2 has merged the remote changes with the new changes, there wouldn’t be any conflict in the merge in the remote server

Note: You can merge and delete the feature branch

Note: The local master will still be not up-to-date, you should never update master locally, you must update by pulling the changes from remote master to auto-merge the local master.

If you check in the local branch you don’t see the updated master branch



You can use git branch to list all the branches

Use git branch -D branch-name to delete the branch, but don’t delete the master branch

