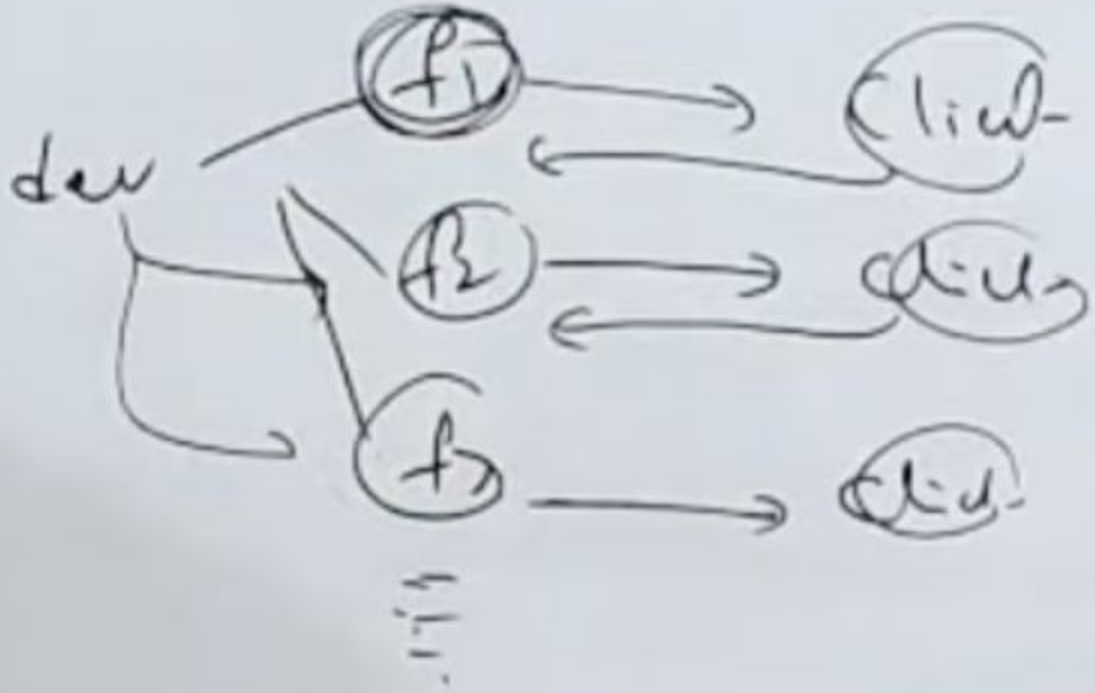
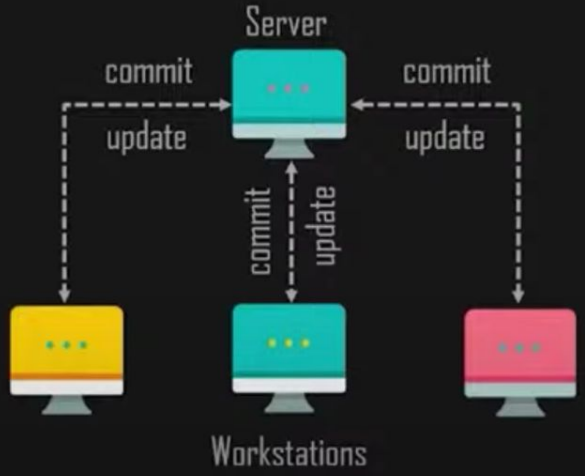
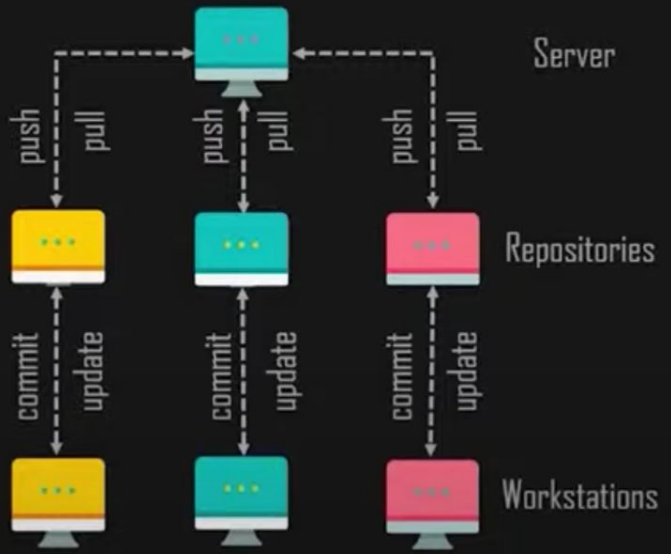


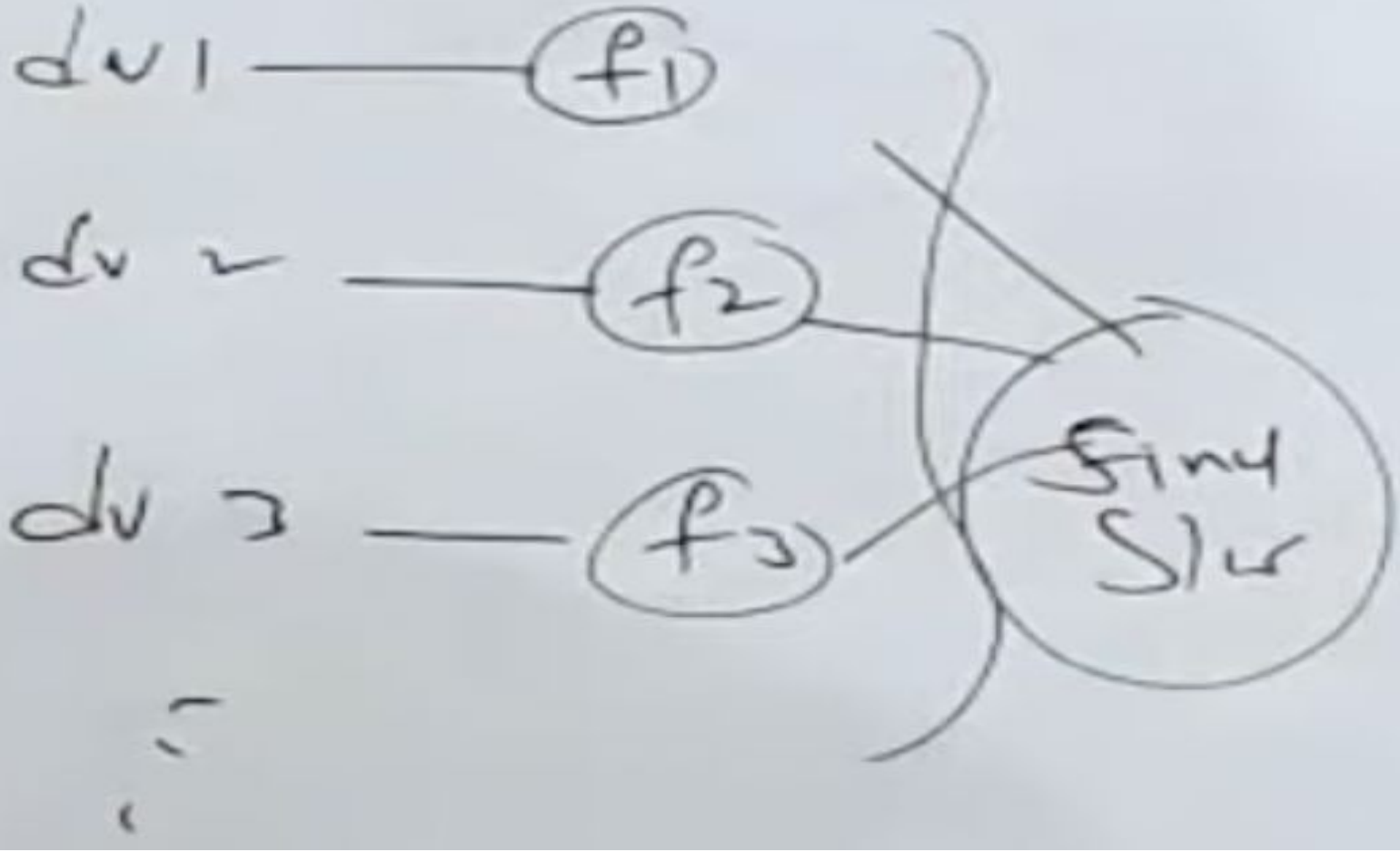
# CENTRALISED vs DISTRIBUTED VCS

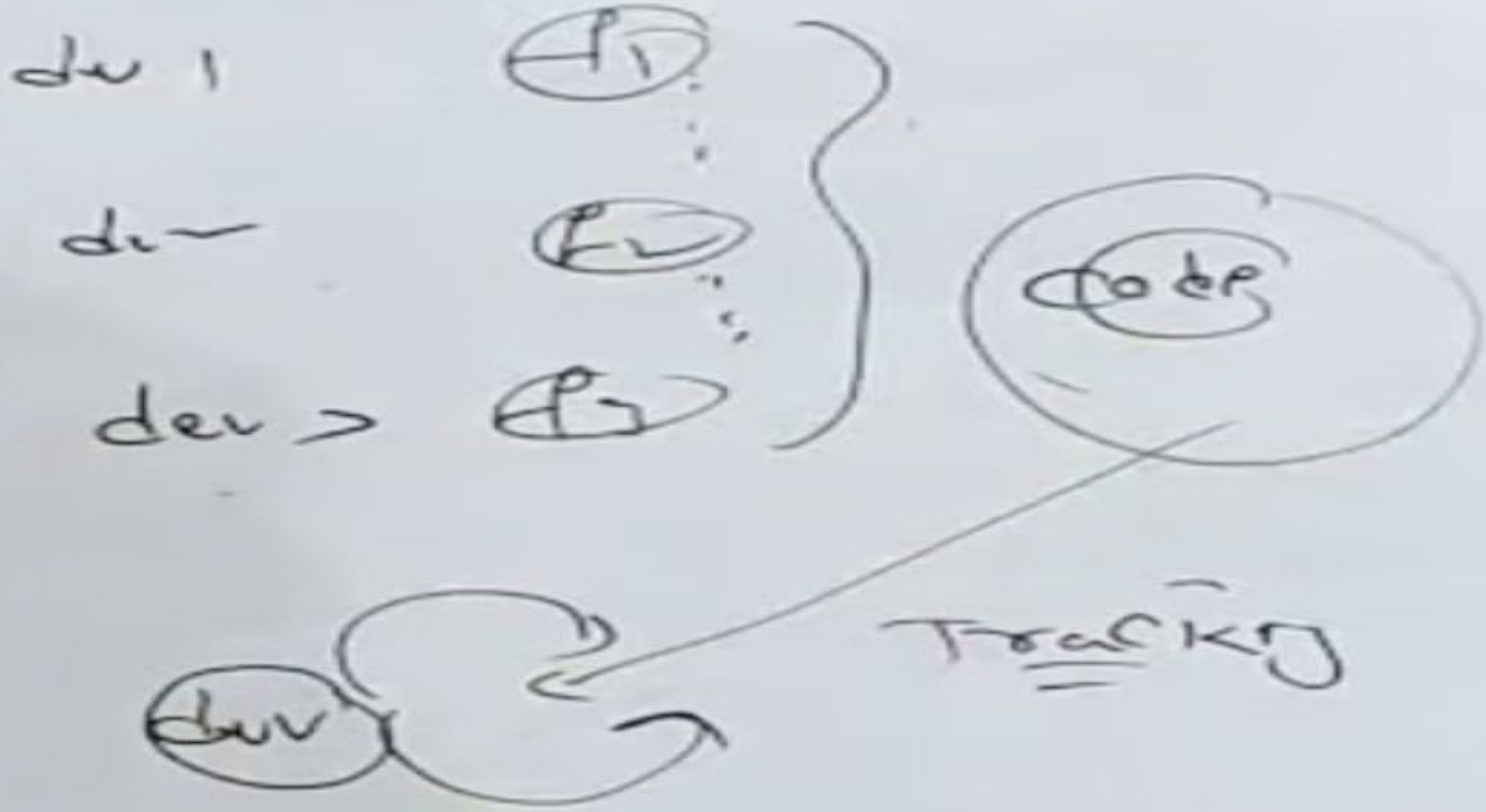



Centralized VCS

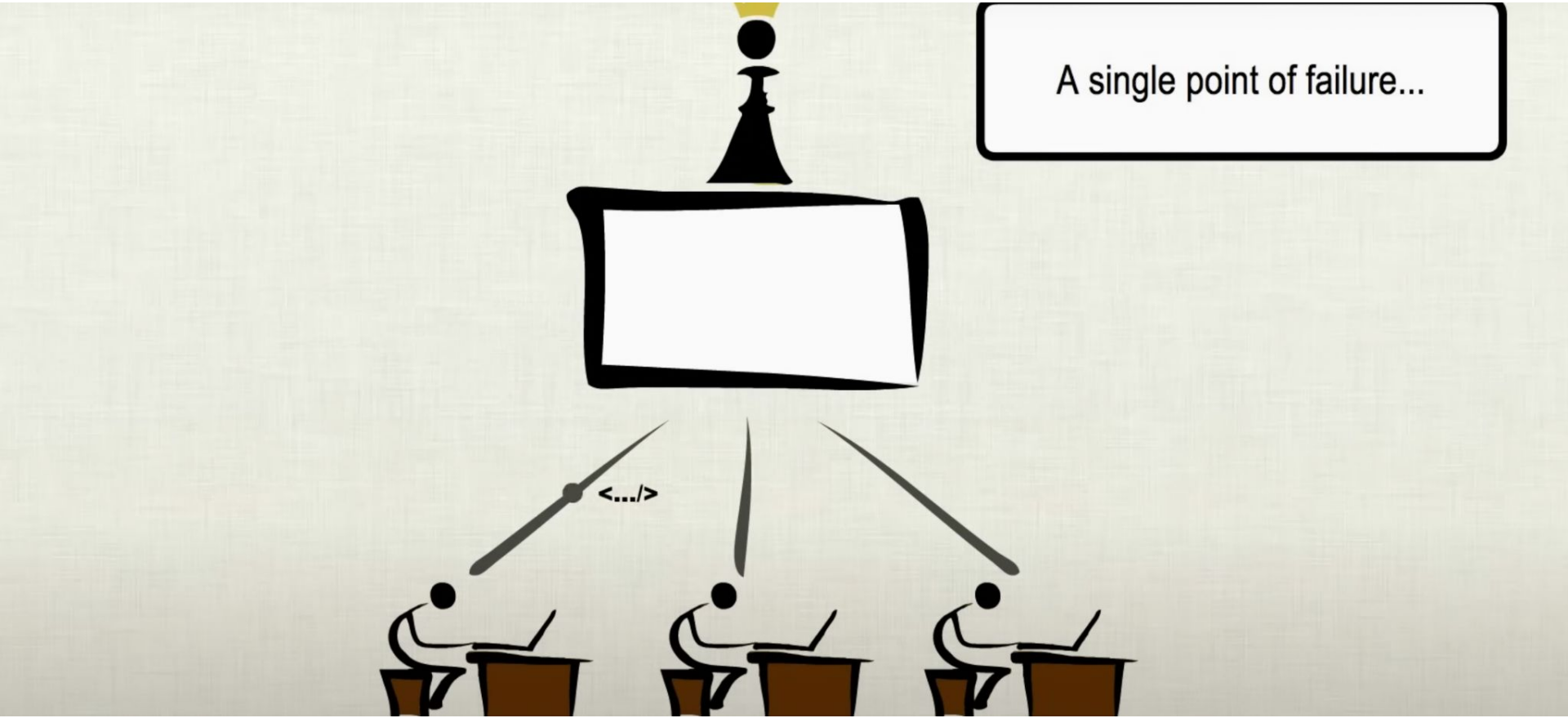


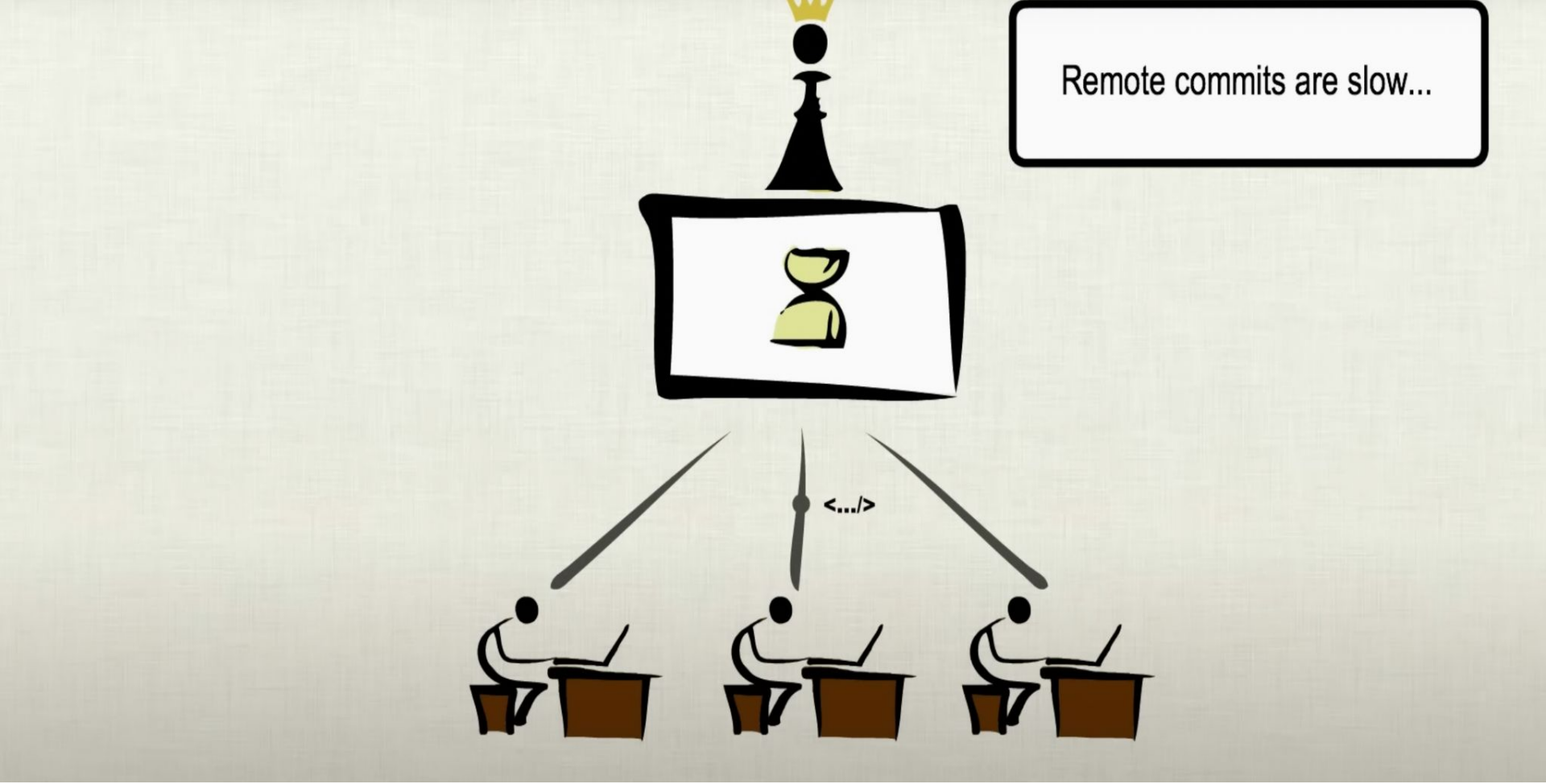

Distributed VCS


dv1
f1
dv2
f2
dv3
f3
Final
Slice

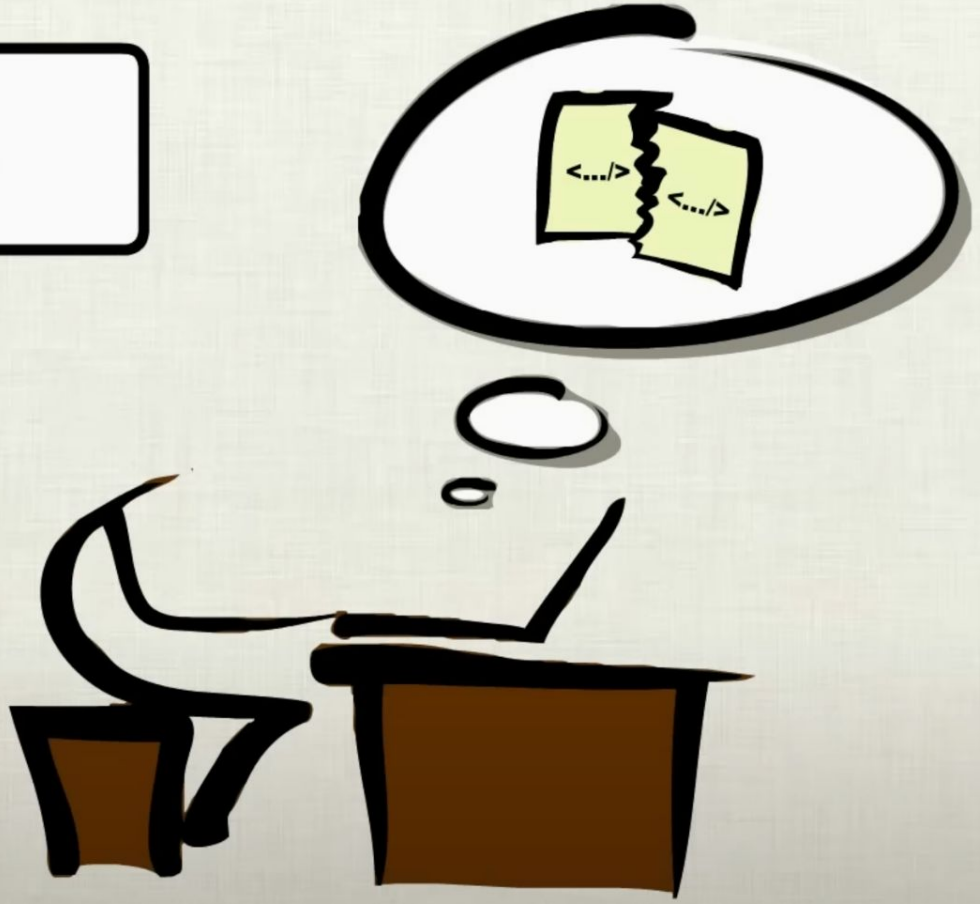

dev 1
dev
dev 3
Code
Tracking
dev

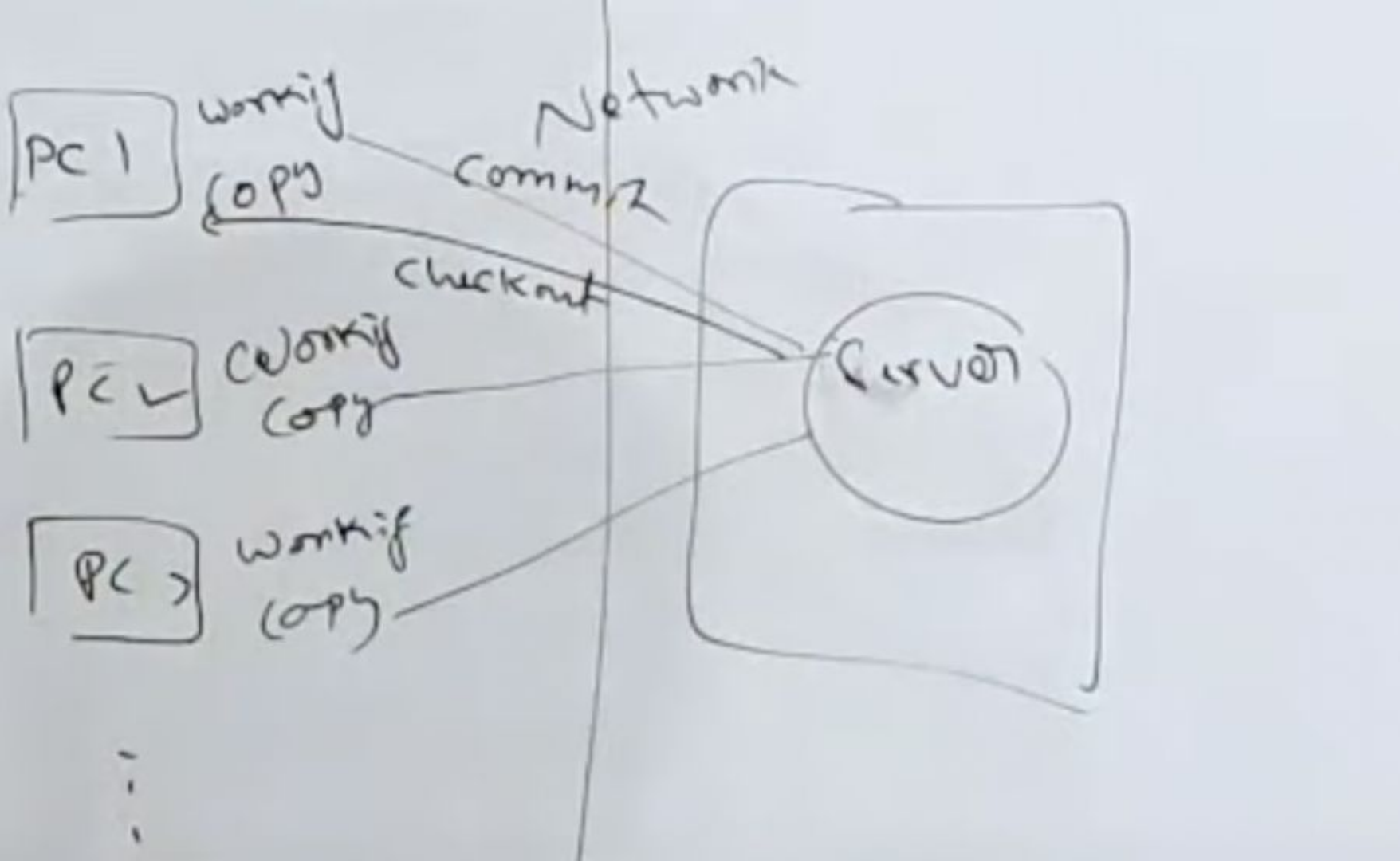
1. Keep versions
2. Tracking [modifications]
3. Sharing code between developers is easy
4. New developers can access code is easy way


A single point of failure...
<.../>


Remote commits are slow...
<.../>

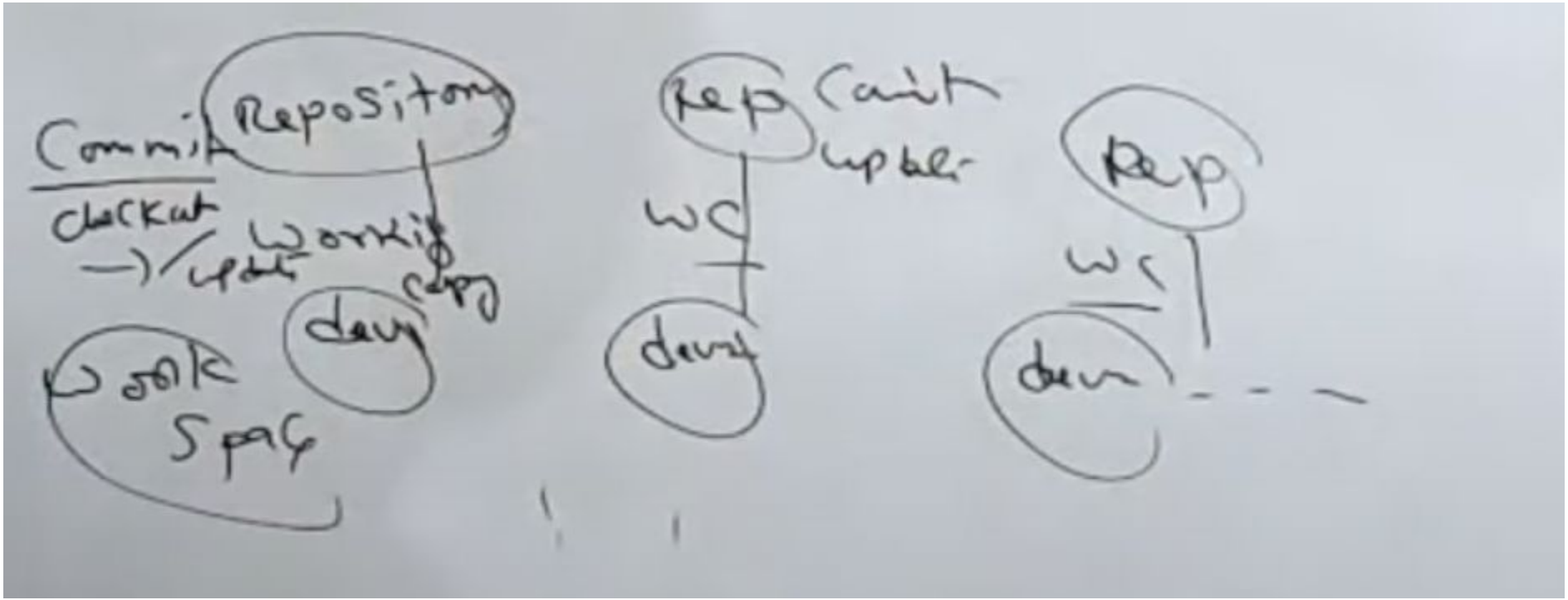
Merging is painful...

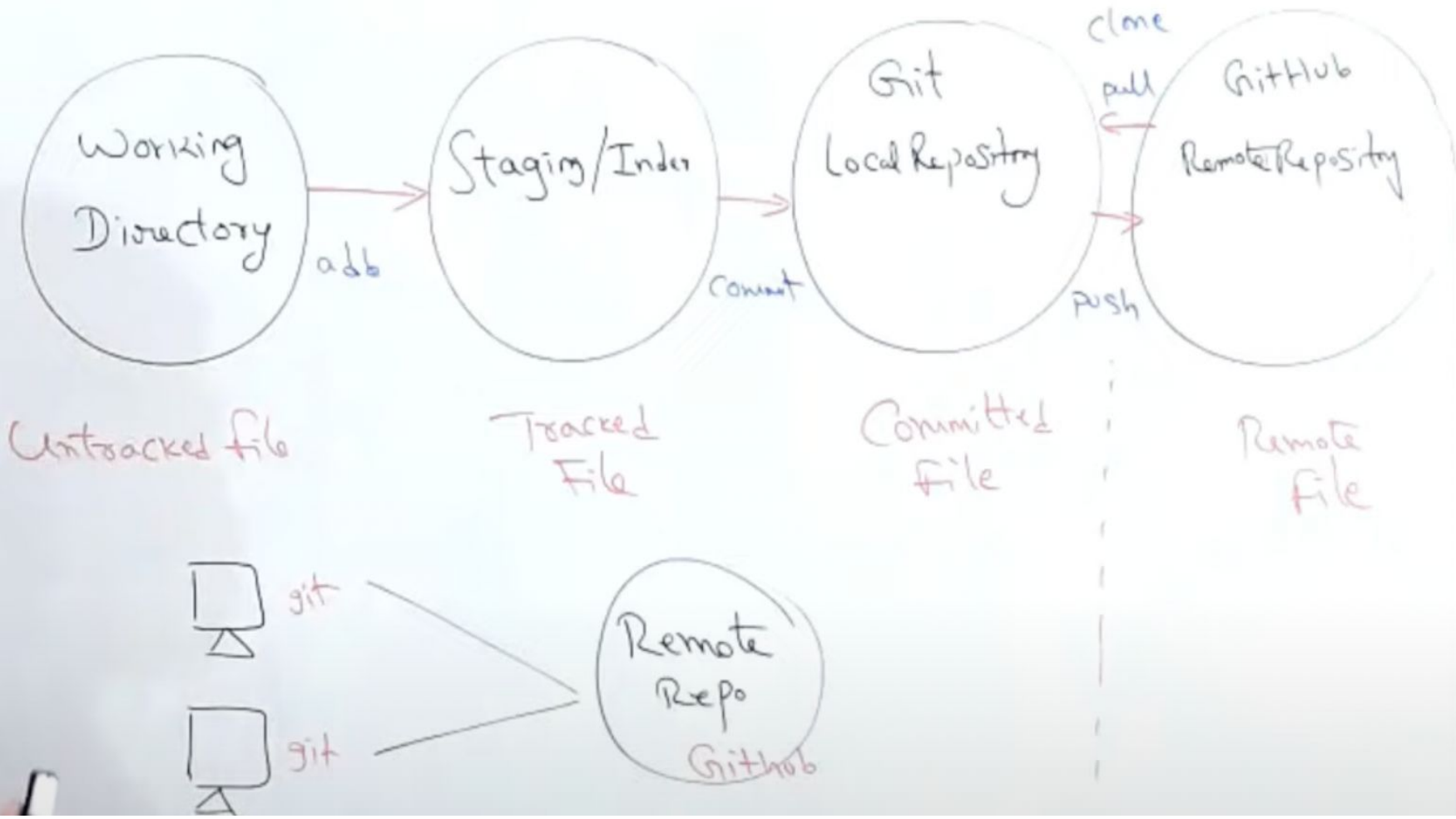



PC 1
working copy
Network
commit
checkout
PC 2
working copy
PC 3
working copy
Server

SVN is example of centralized version control system

# Distributed version control system




Working Directory
add
Staging/Index
commit
Git Local Repository
clone
pull
push
GitHub Remote Repository
Untracked file
Tracked File
Committed file
Remote file
git
git
Remote Repo
Github

# GIT vs. SVN

## Decentralized

- GIT is decentralized. You have a local copy that is a repository in which you can commit. In SVN you have to always connect to a central repository for check-in.

git --local-branching-on-the-cheap

Search entire site...

About
Documentation
Downloads
GUI Clients
Logos
Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

# Downloading Git

## Your download is starting...

You are downloading the latest (**2.31.1**) **64-bit** version of **Git for Windows**. This is the most recent maintained build. It was released **about 2 months ago**, on 2021-03-27.

**Click here to download manually, if your download hasn't started.**

## Other Git for Windows downloads

**Git for Windows Setup**
**32-bit Git for Windows Setup.**

**64-bit Git for Windows Setup.**

**Git for Windows Portable ("thumbdrive edition")**
**32-bit Git for Windows Portable.**

**64-bit Git for Windows Portable.**

The current source code release is version **2.31.1**. If you want the newer version, you can build it from the source code.

## Now What?

Now that you have downloaded Git, it's time to start using it.

### Read the Book

Dive into the Pro Git book and

### Download a GUI

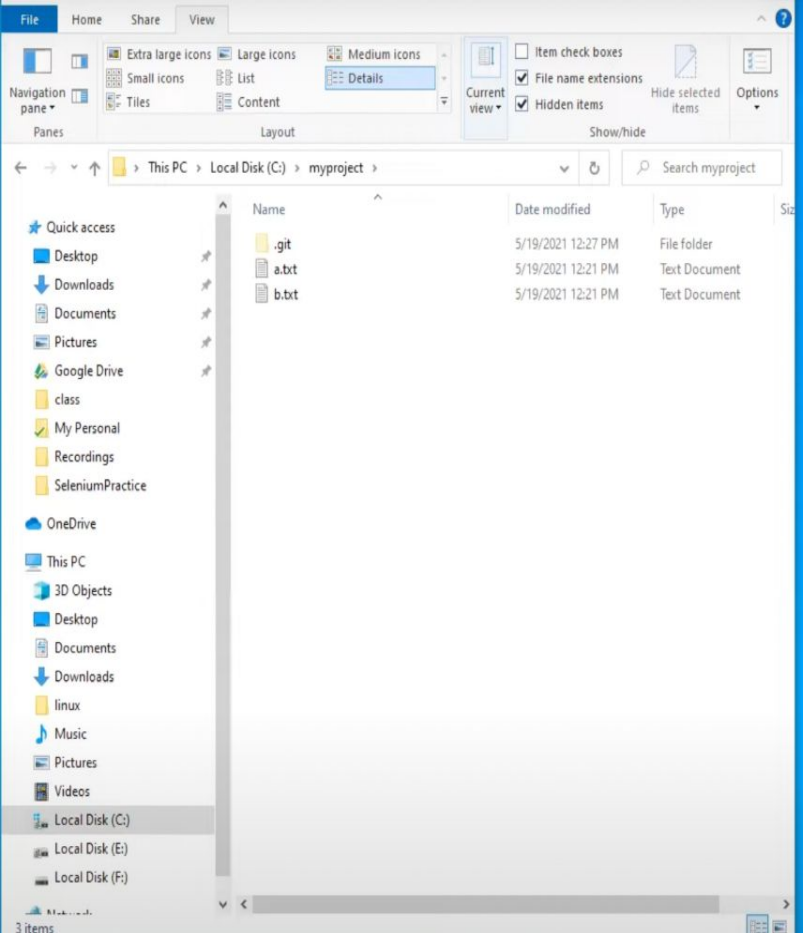
Several free and commercial GUI

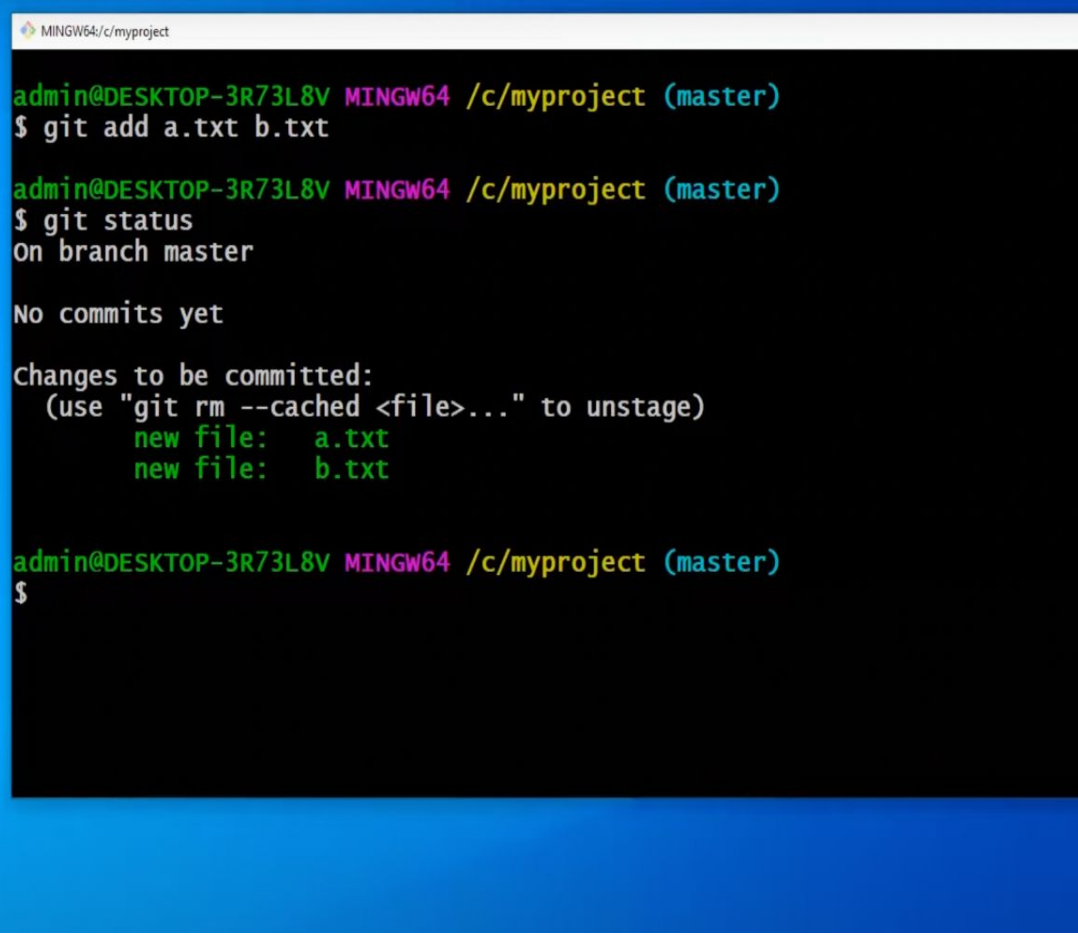
### Get Involved

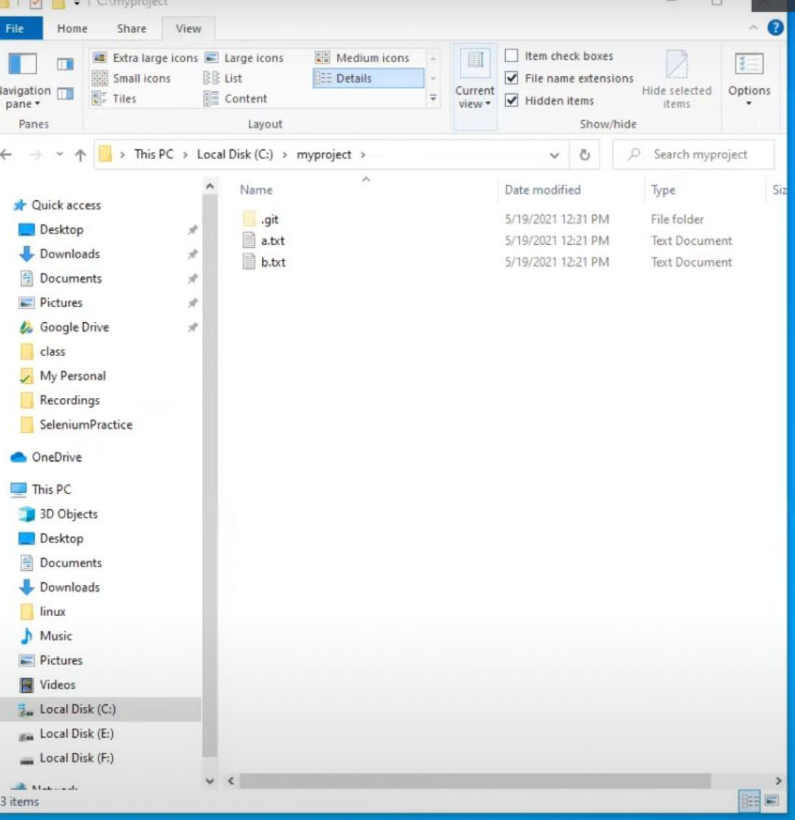
A knowledgeable Git community is

# Git Commands

1) init
2) status
3) add
4) commit
5) log
6) config

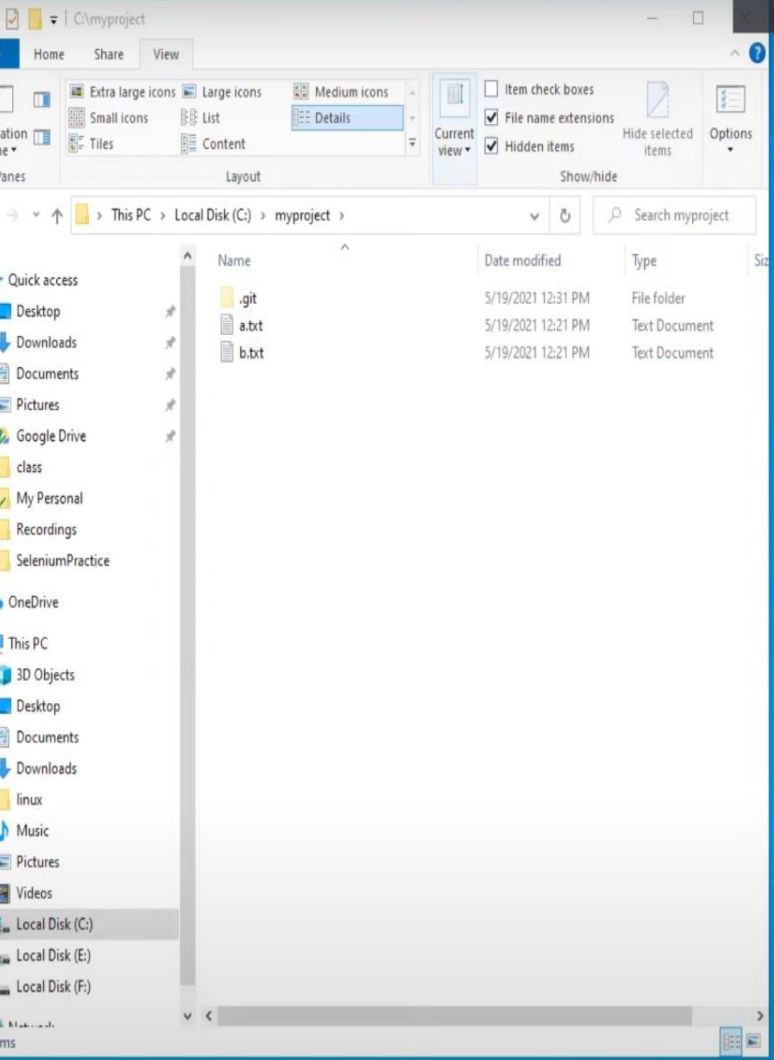

File
Home
Share
View
Navigation pane
Panes
Extra large icons
Large icons
Medium icons
Small icons
List
Details
Tiles
Content
Layout
Current view
Item check boxes
File name extensions
Hidden items
Hide selected items
Show/hide
Options
This PC
Local Disk (C:)
myproject
Search myproject
Name
Date modified
Type
.git
5/19/2021 12:27 PM
File folder
a.txt
5/19/2021 12:21 PM
Text Document
b.txt
5/19/2021 12:21 PM
Text Document
Quick access
Desktop
Downloads
Documents
Pictures
Google Drive
class
My Personal
Recordings
SeleniumPractice
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
linux
Music
Pictures
Videos
Local Disk (C:)
Local Disk (E:)
Local Disk (F:)
3 items

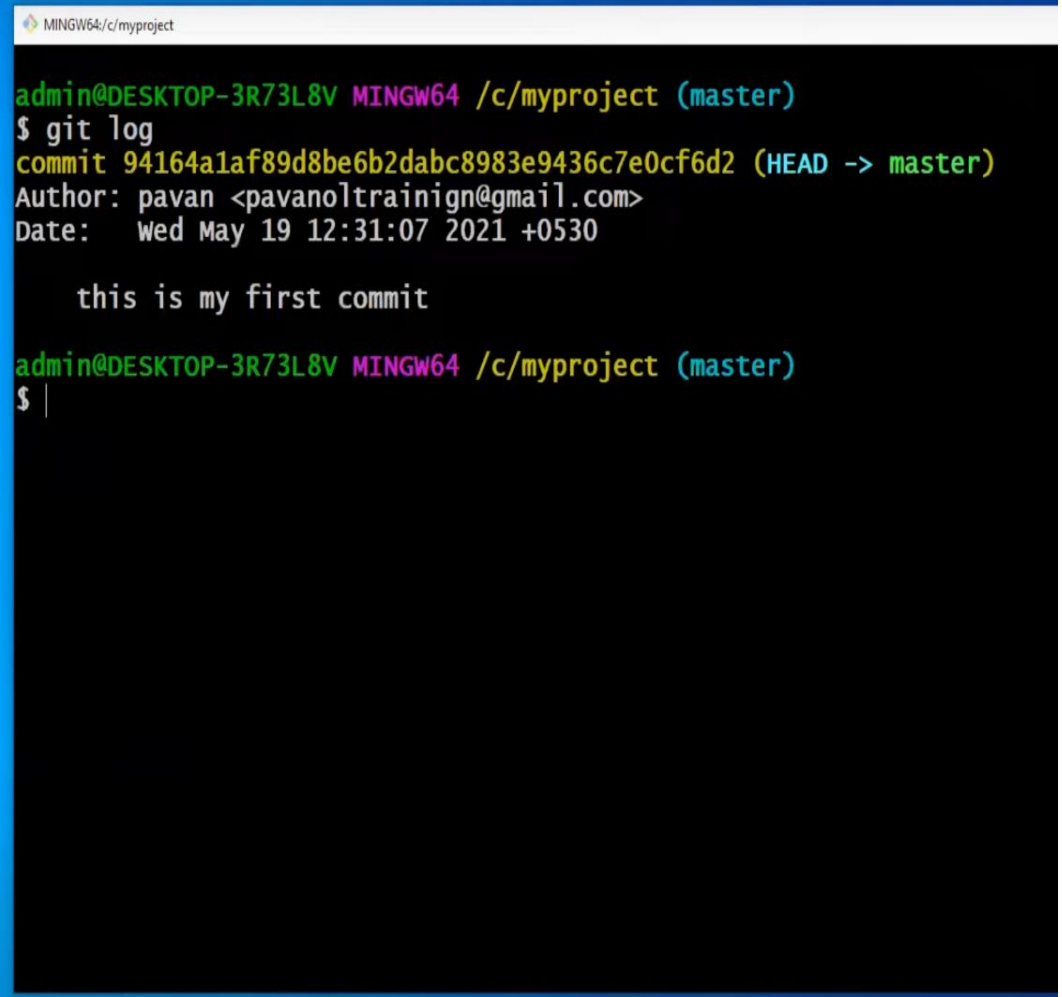

MINGW64:/c/myproject
admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git add a.txt b.txt
admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git status
On branch master
No commits yet
Changes to be committed:
(use "git rm --cached <file>..." to unstage)
new file: a.txt
new file: b.txt
admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$




MINGW64:/c/myproject

```
admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   a.txt
        new file:   b.txt


admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git config --global user.email "pavanoltrainign@gmail.com"

admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git config --global user.name "pavan"

admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git commit -m "this is my first commit"
[master (root-commit) 94164a1] this is my first commit
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 a.txt
 create mode 100644 b.txt

admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$
```


C:\myproject
Home
Share
View
Extra large icons
Large icons
Medium icons
Small icons
List
Details
Tiles
Content
Layout
Current view
Item check boxes
File name extensions
Hidden items
Hide selected items
Show/hide
Options
This PC › Local Disk (C:) › myproject ›
Search myproject
Quick access
Desktop
Downloads
Documents
Pictures
Google Drive
class
My Personal
Recordings
SeleniumPractice
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
linux
Music
Pictures
Videos
Local Disk (C:)
Local Disk (E:)
Local Disk (F:)
Name
Date modified
Type
.git
5/19/2021 12:31 PM
File folder
a.txt
5/19/2021 12:21 PM
Text Document
b.txt
5/19/2021 12:21 PM
Text Document


MINGW64:/c/myproject
admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git log
commit 94164a1af89d8be6b2dabc8983e9436c7e0cf6d2 (HEAD -> master)
Author: pavan <pavanoltrainign@gmail.com>
Date: Wed May 19 12:31:07 2021 +0530
this is my first commit
admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$

# What's in it for you?

- Download and install Git
- Git bash interface
- Basic Git commands
- Create a local repository
- **Connect to remote repository**
- Push the file to GitHub

git --fast-version-control

Search entire site...

About

Documentation

Downloads

GUI Clients

Logos

Community

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

# Downloading Git

## Your download is starting...

You are downloading the latest (**2.19.1**) **64-bit** version of **Git for Windows**. This is the most recent maintained build. It was released **about 1 month ago**, on 2018-10-05.

**If your download hasn't started, click here to download manually.**

## Other Git for Windows downloads

**Git for Windows Setup**

**32-bit Git for Windows Setup.**

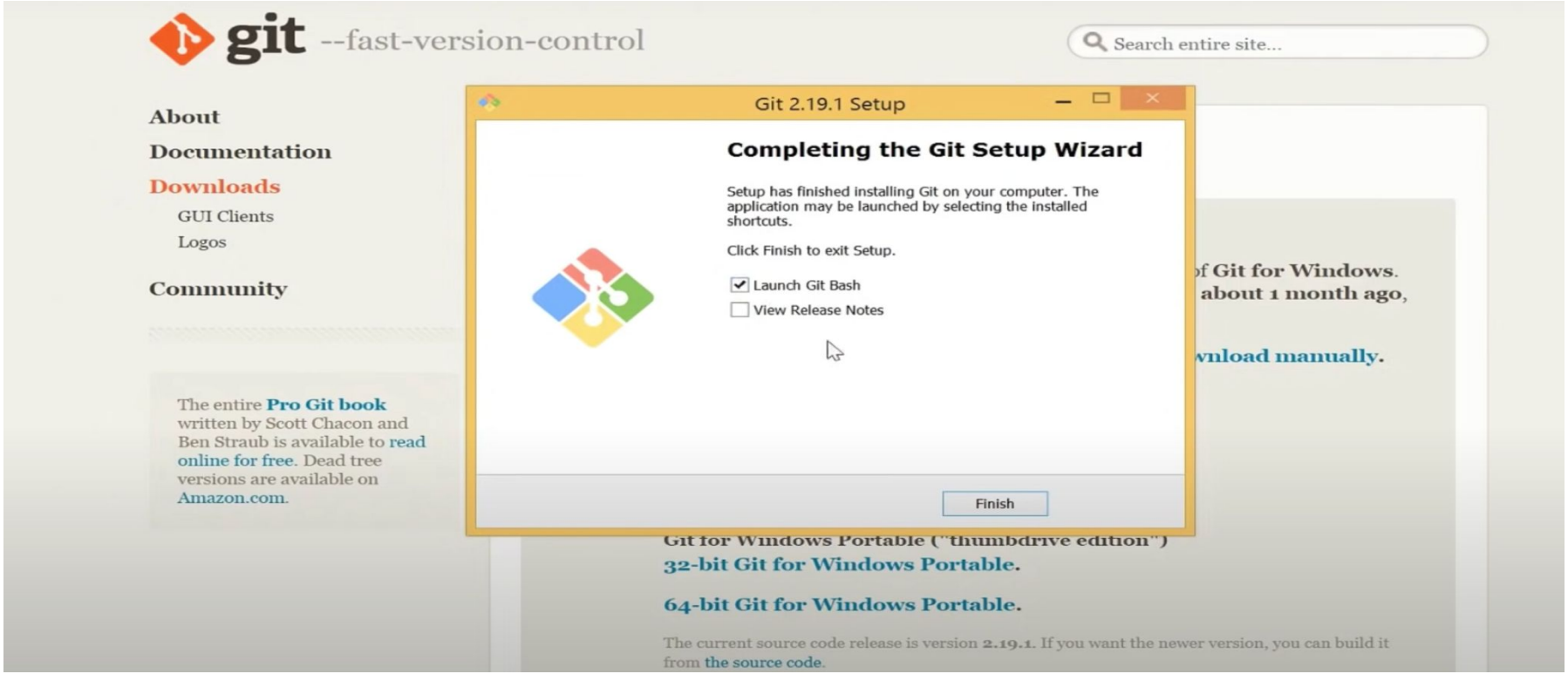
**64-bit Git for Windows Setup.**

**Git for Windows Portable ("thumbdrive edition")**

**32-bit Git for Windows Portable.**

**64-bit Git for Windows Portable.**

**Just click on next until you get below window**

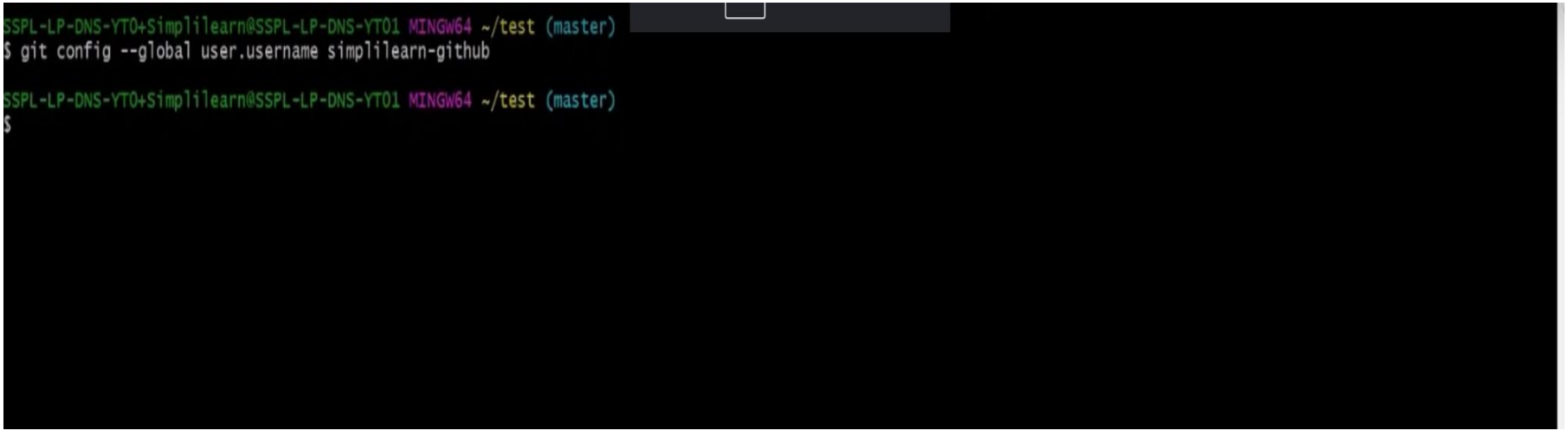


```
SSPL-LP-DNS-YTO+Simplilearn@SSPL-LP-DNS-YTO1 MINGW64 ~ (master)
$ git --version
git version 2.19.1.windows.1

SSPL-LP-DNS-YTO+Simplilearn@SSPL-LP-DNS-YTO1 MINGW64 ~ (master)
$ git help config

SSPL-LP-DNS-YTO+Simplilearn@SSPL-LP-DNS-YTO1 MINGW64 ~ (master)
$ git config --help

SSPL-LP-DNS-YTO+Simplilearn@SSPL-LP-DNS-YTO1 MINGW64 ~ (master)
$
```

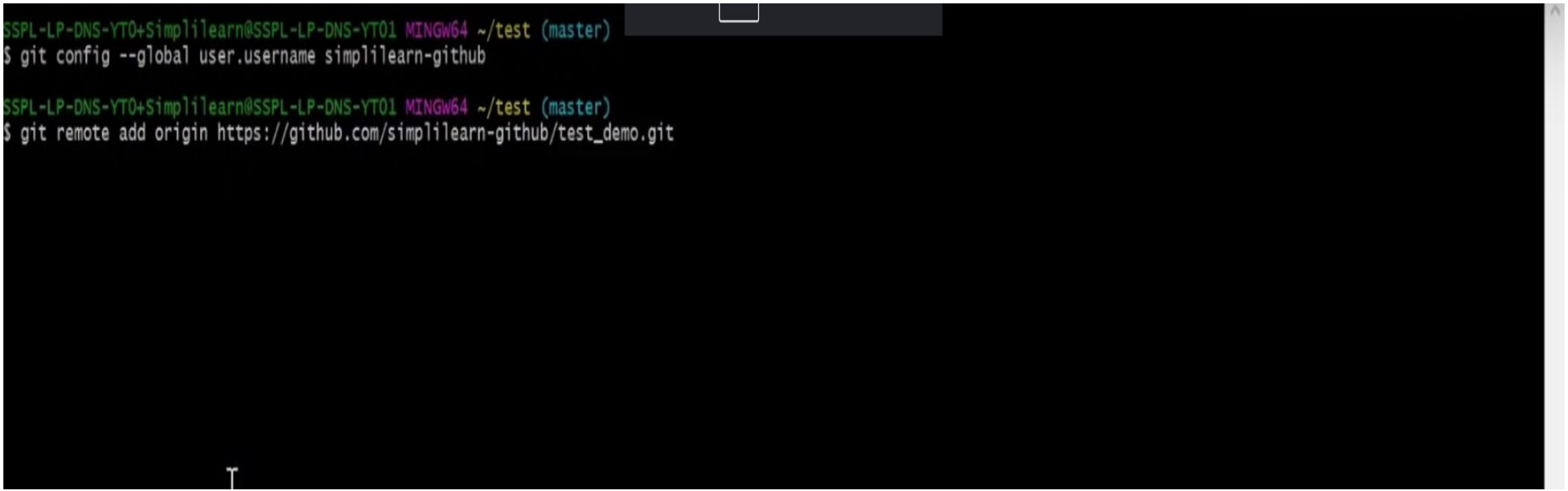
```
SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~ (master)
$ git --version
git version 2.19.1.windows.1

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~ (master)
$ git help config

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~ (master)
$ git config --help

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~ (master)
$ mkdir test

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~ (master)
$ cd test

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~/test (master)
$ git init
Initialized empty Git repository in C:/Users/Simplilearn/test/.git/

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~/test (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        demo.txt

nothing added to commit but untracked files present (use "git add" to track)

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~/test (master)
$ git add demo.txt

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~/test (master)
$
```

```
SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~ (master)
$ git --version
git version 2.19.1.windows.1

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~ (master)
$ git help config

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~ (master)
$ git config --help

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~ (master)
$ mkdir test

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~ (master)
$ cd test

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~/test (master)
$ git init
Initialized empty Git repository in C:/Users/Simplilearn/test/.git/

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~/test (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

        demo.txt

nothing added to commit but untracked files present (use "git add" to track)

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~/test (master)
$ git add demo.txt

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~/test (master)
$ git commit -m "comitting a text file"
[master (root-commit) ba65497] comitting a text file
 1 file changed, 1 insertion(+)
 create mode 100644 demo.txt

SSPL-LP-DNS-YT01+Simplilearn@SSPL-LP-DNS-YT01 MINGW64 ~/test (master)
```

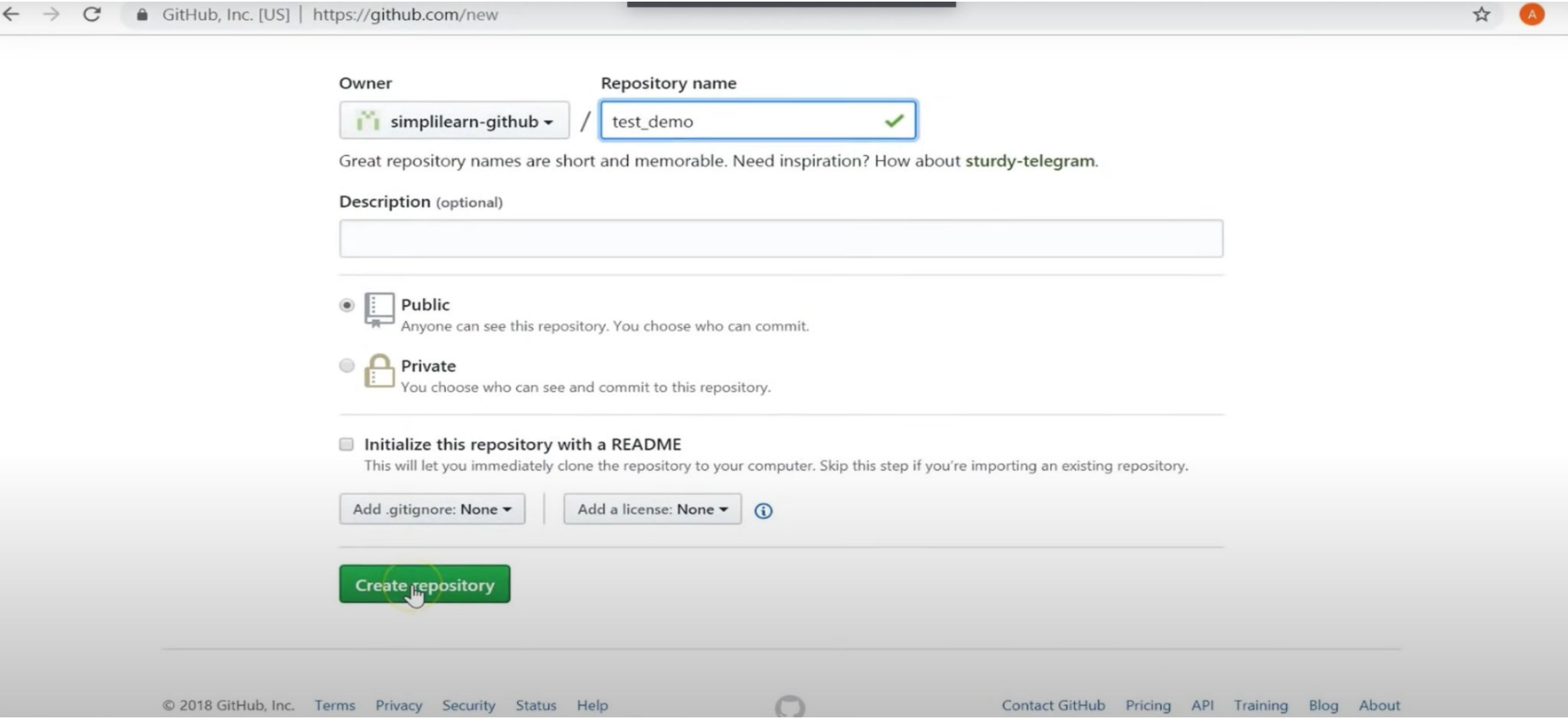
# Link git to github account



# Link git to github account [local repo is linked with remote repo]



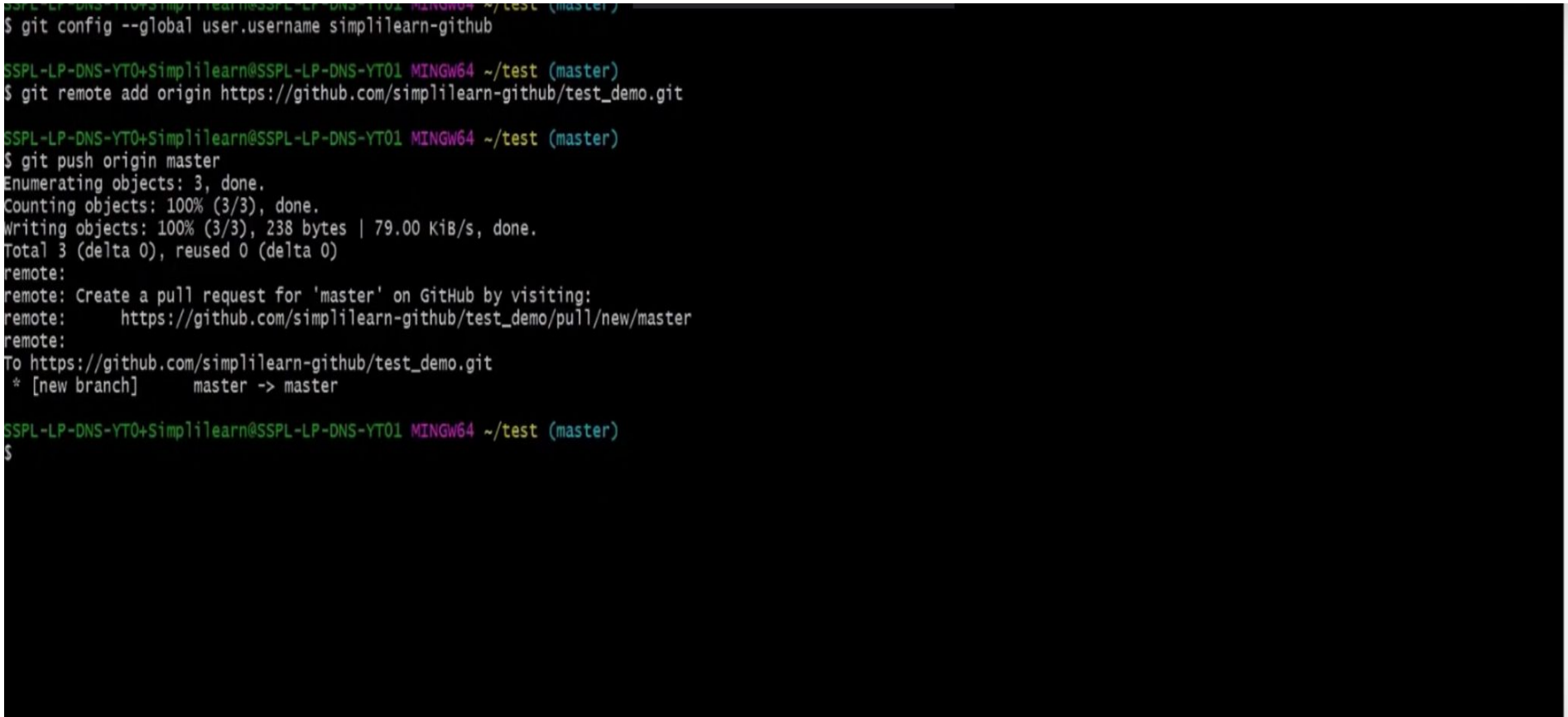
# Create github repository

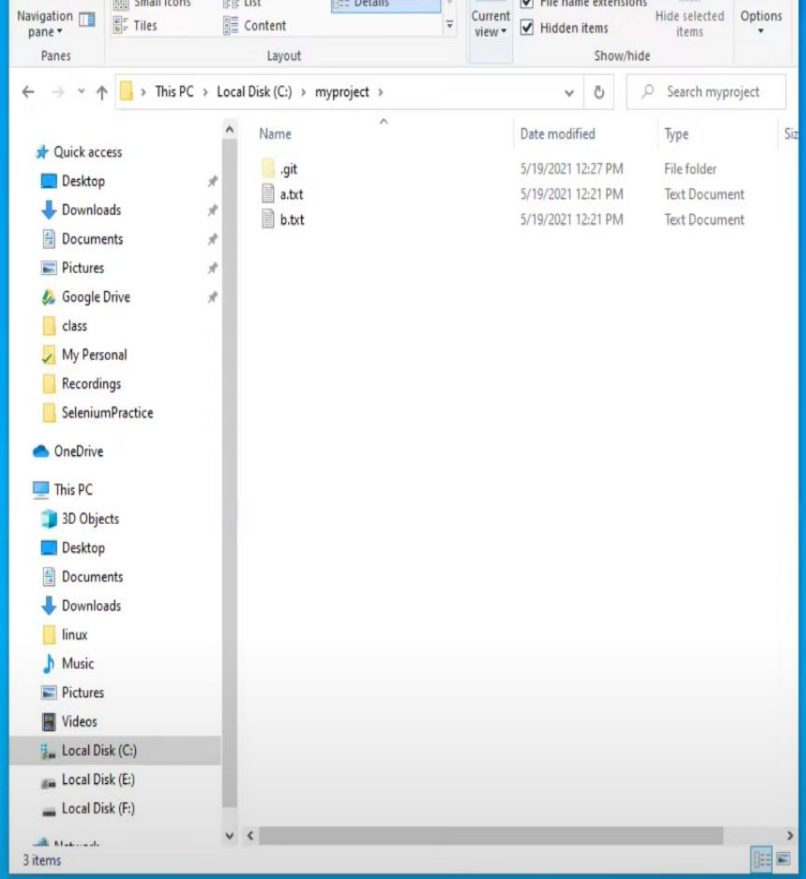


Git branch - list all the branches in your local repository by default only master branch is existing

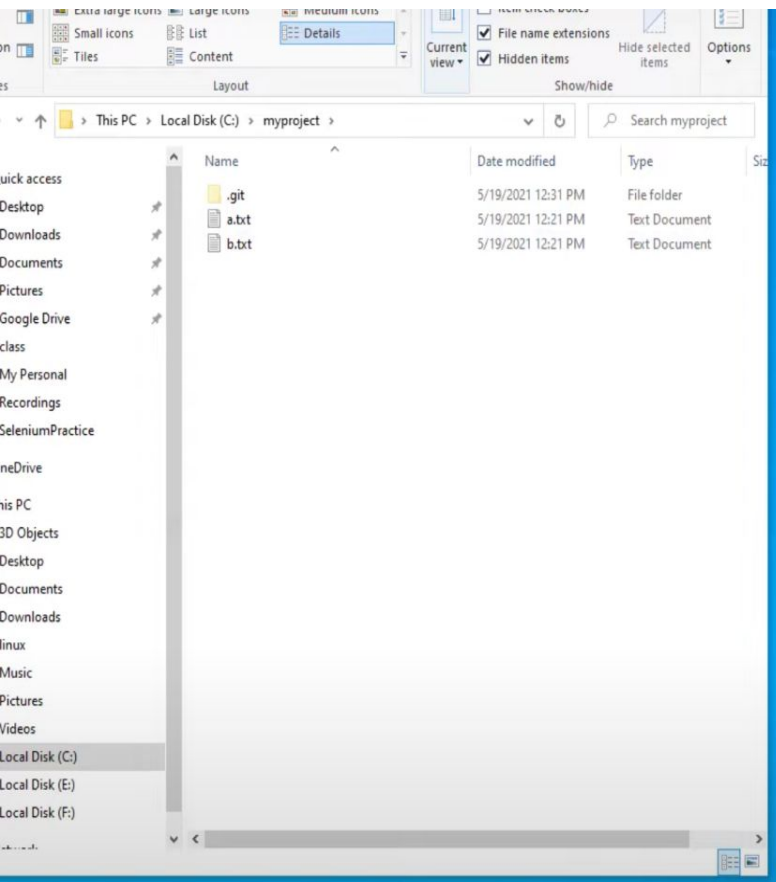
Git checkout -b branchname

# How to push local file to remote repository






```
admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git add a.txt b.txt

admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   a.txt
        new file:   b.txt


admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git config --global user.email "pavanoltrainign@gmail.com"

admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git config --global user.name "pavan"

admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$
```

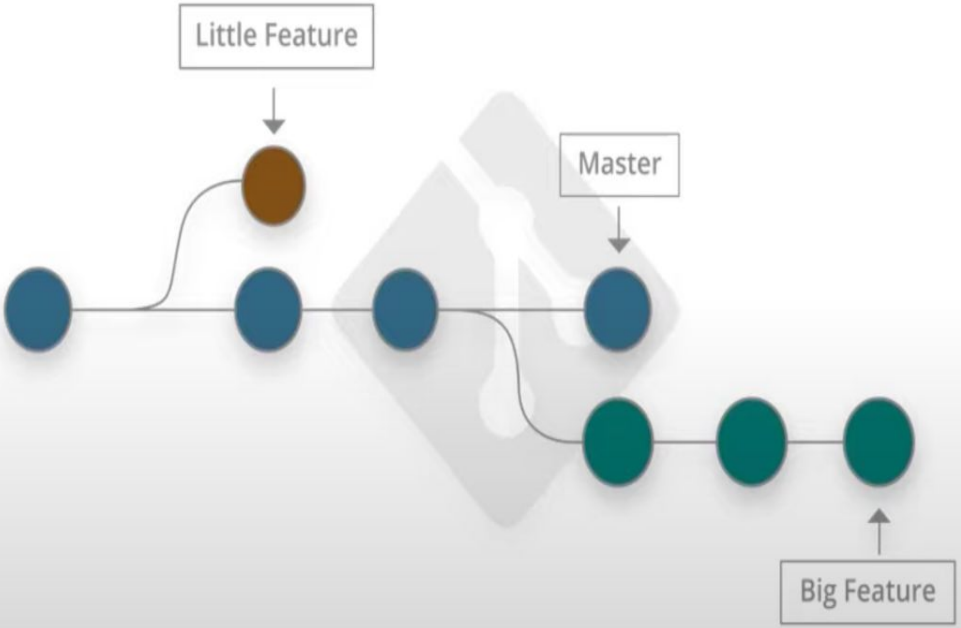



```
admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$ git log
commit 94164a1af89d8be6b2dabc8983e9436c7e0cf6d2 (HEAD -> master)
Author: pavan <pavanoltrainign@gmail.com>
Date:   Wed May 19 12:31:07 2021 +0530

    this is my first commit

admin@DESKTOP-3R73L8V MINGW64 /c/myproject (master)
$
```

# Git basic commands

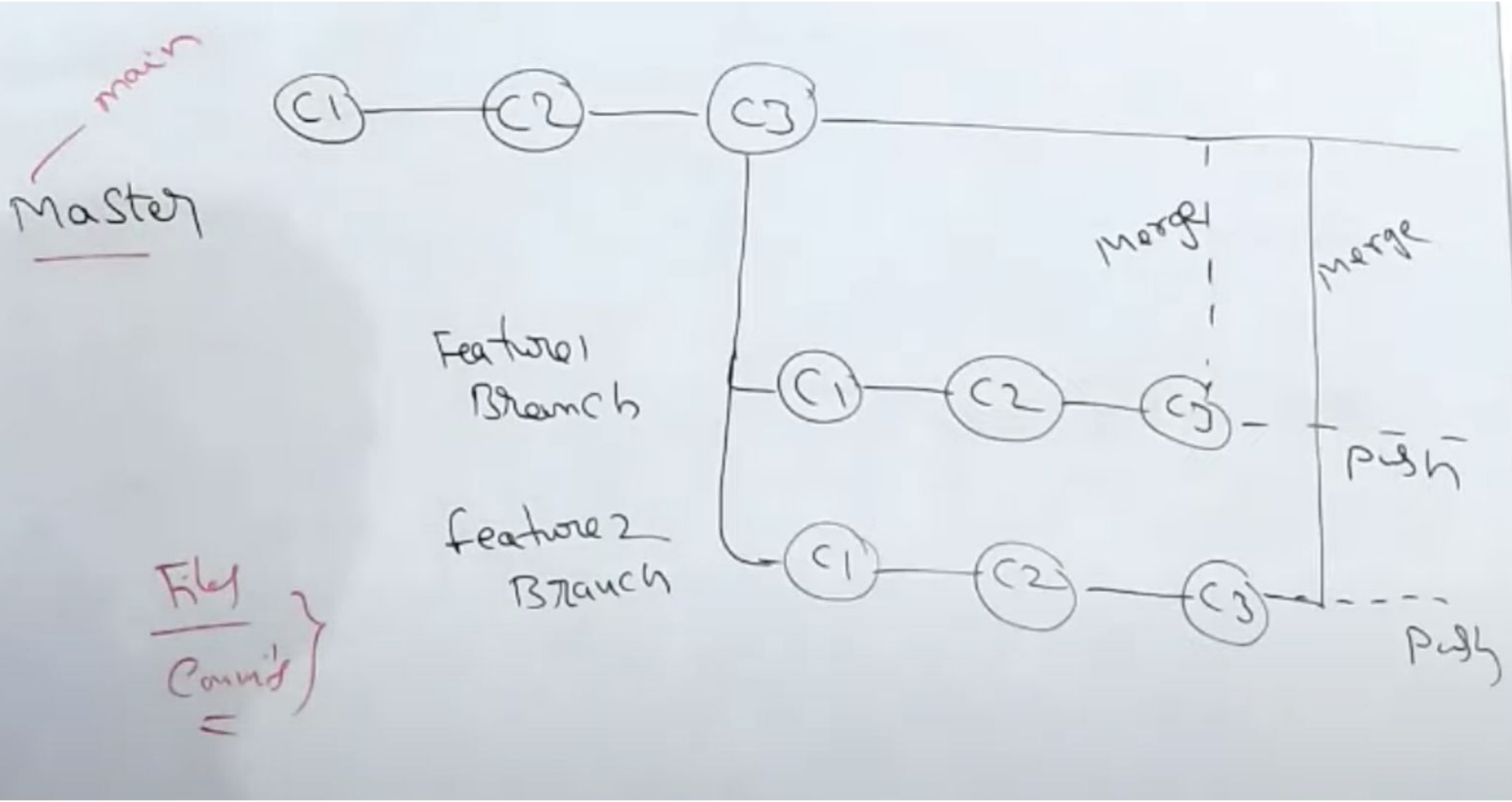
1. git init - new empty local repository will be created
2. git status-show the status of files like tracked and untracked
3. git add - add the files to your staging area[local directory]
4. git commit-commit all your changes to your local repository
5. git log - show who committed ,date,time
6. git config-you have to execute only once to configure your username and email id
7. `#set a new remote`

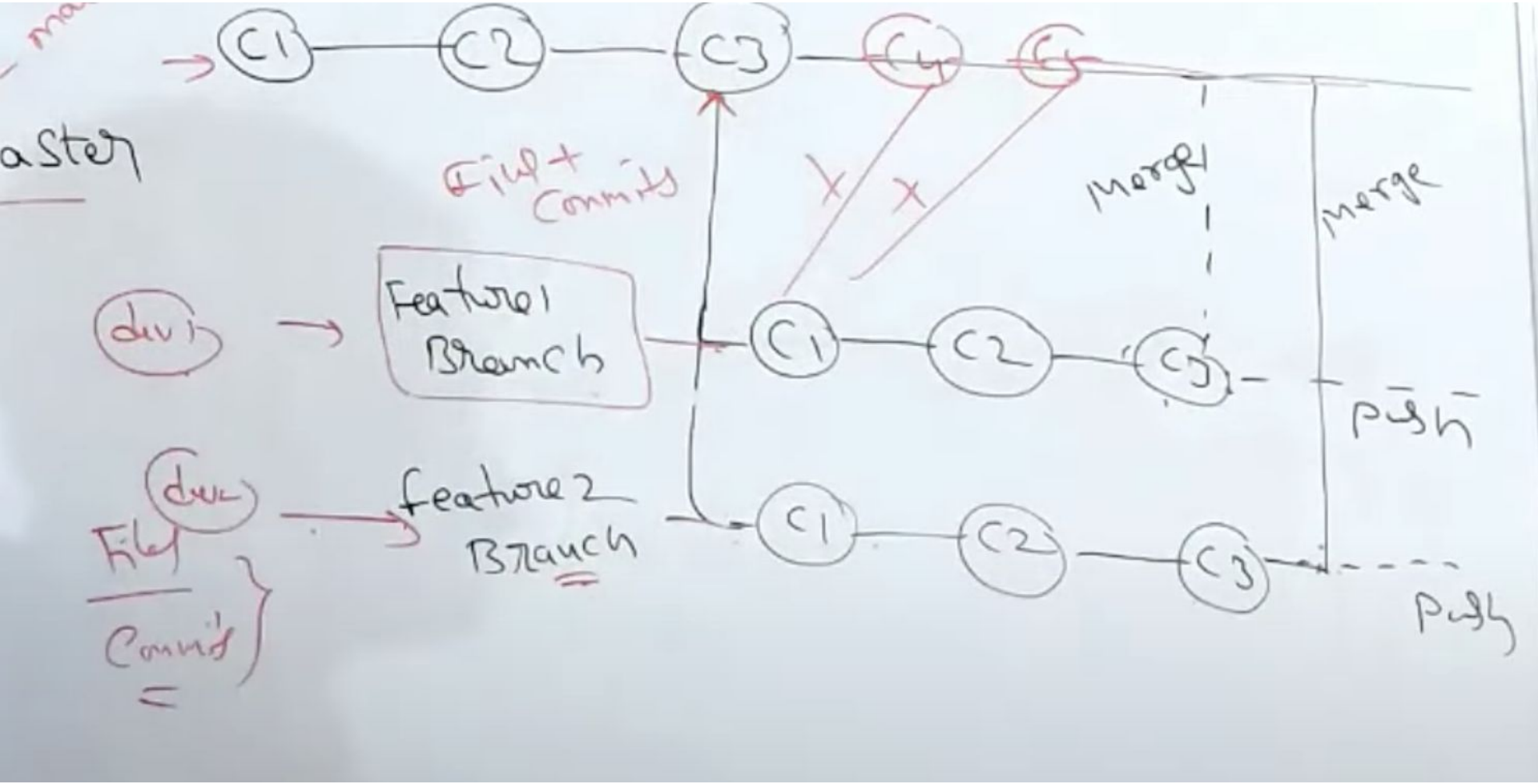
   `git remote add my_awesome_new_remote_repo git@git.assembla.com:portfolio/space.space_name.git`
8. `#Verify new remote`
9. `git remote -v`
10.




Two reasons:

- New features could break code
- Collaboration purposes


main
Master
C1
C2
C3
Merge
Merge
Feature1 Branch
C1
C2
C3
Push
feature2 Branch
C1
C2
C3
Push
File
Commit
=


C1
C2
C3
C4
C5
aster
File + Commit
X
X
Merge
Merge
dev1
Feature1 Branch
C1
C2
C3
push
dev2
Feature2 Branch
C1
C2
C3
push
File
Commit
=

C4 and c5 will not be part of commit we need pull changes from master and then merge code

Git Branching Commands

---

1) To view available branches

```
git branch
git status
```

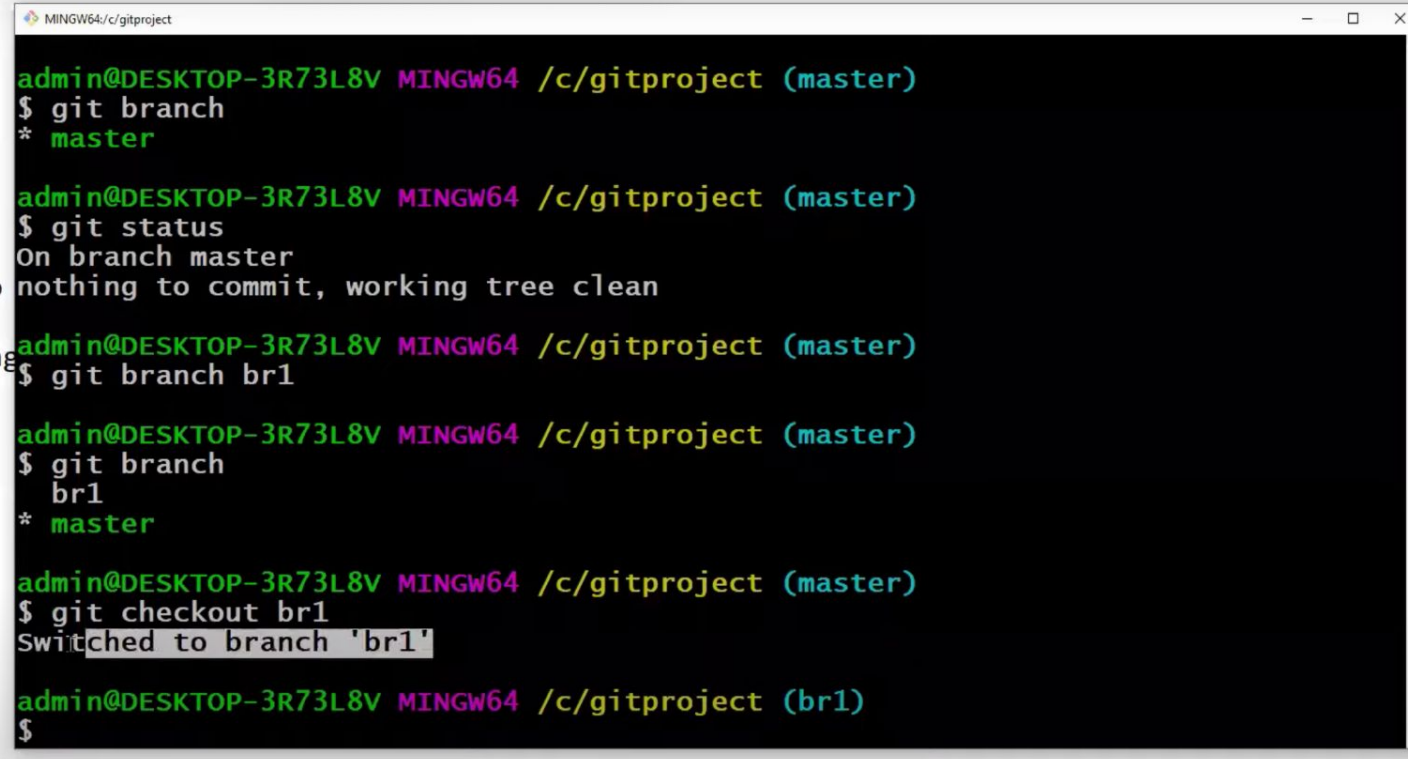
2) Create new branch

```
git branch branchname
```

3) Switch to from one branch to

git checkout  ==> discard unstag

git checkout branch_name

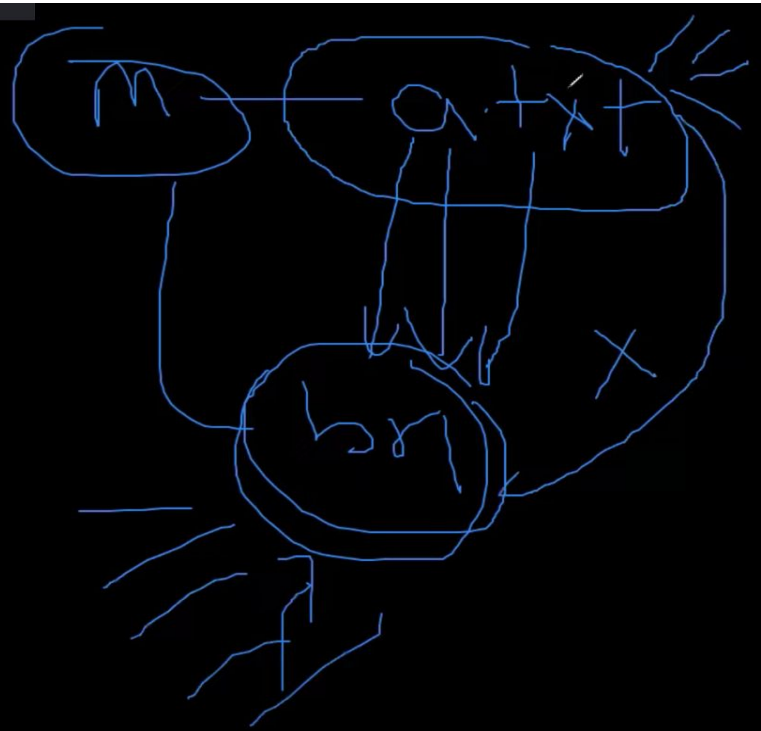


```
  br2
* master

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ ls
a.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ touch b.txt c.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ ls
a.txt  b.txt  c.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git add b.txt;git commit -m "b.txt:
> "
[master c379f96] b.txt:
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 b.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git add c.txt;git commit -m "c.txt"
[master 903f84e] c.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 c.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git ls-files
a.txt
b.txt
c.txt
```

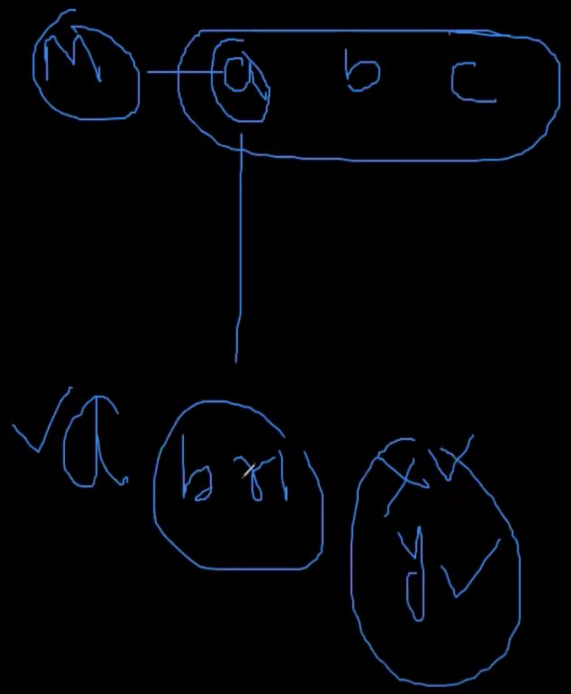
# Git log will give history of all commits

# Master branch commits are not affected in child branch

```
admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git status
On branch master
nothing to commit, working tree clean

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ ls
a.txt  b.txt  c.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git checkout br1
Switched to branch 'br1'

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (br1)
$ ls
a.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (br1)
$ git log --oneline
b5e48a3 (HEAD -> br1, br2) a.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (br1)
$
```



```
$ git branch
* br1
  br2
  master

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (br1)
$ touch x.txt y.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (br1)
$ git add x.txt;git commit -m "x.txt"
[br1 54ae755] x.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 x.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (br1)
$ git add y.txt;git commit -m "y.txt"
[br1 f3d327d] y.txt
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 y.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (br1)
$ ls
a.txt  x.txt  y.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (br1)
$ git checkout master
Switched to branch 'master'

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ ls
a.txt  b.txt  c.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$
```

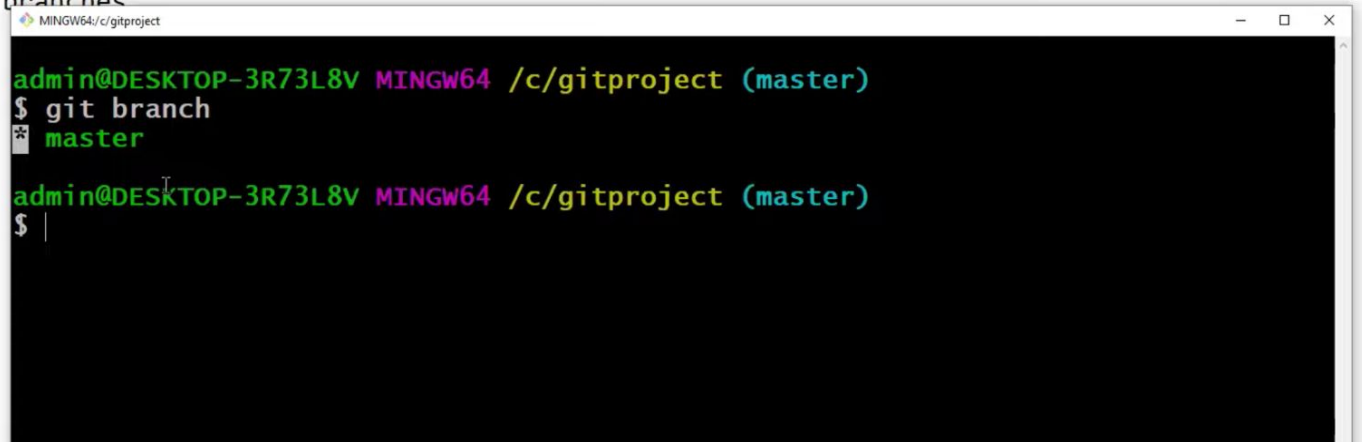


Git Branching Commands

---

1) To view available branches

```
git branch
```



Git Branching Commands

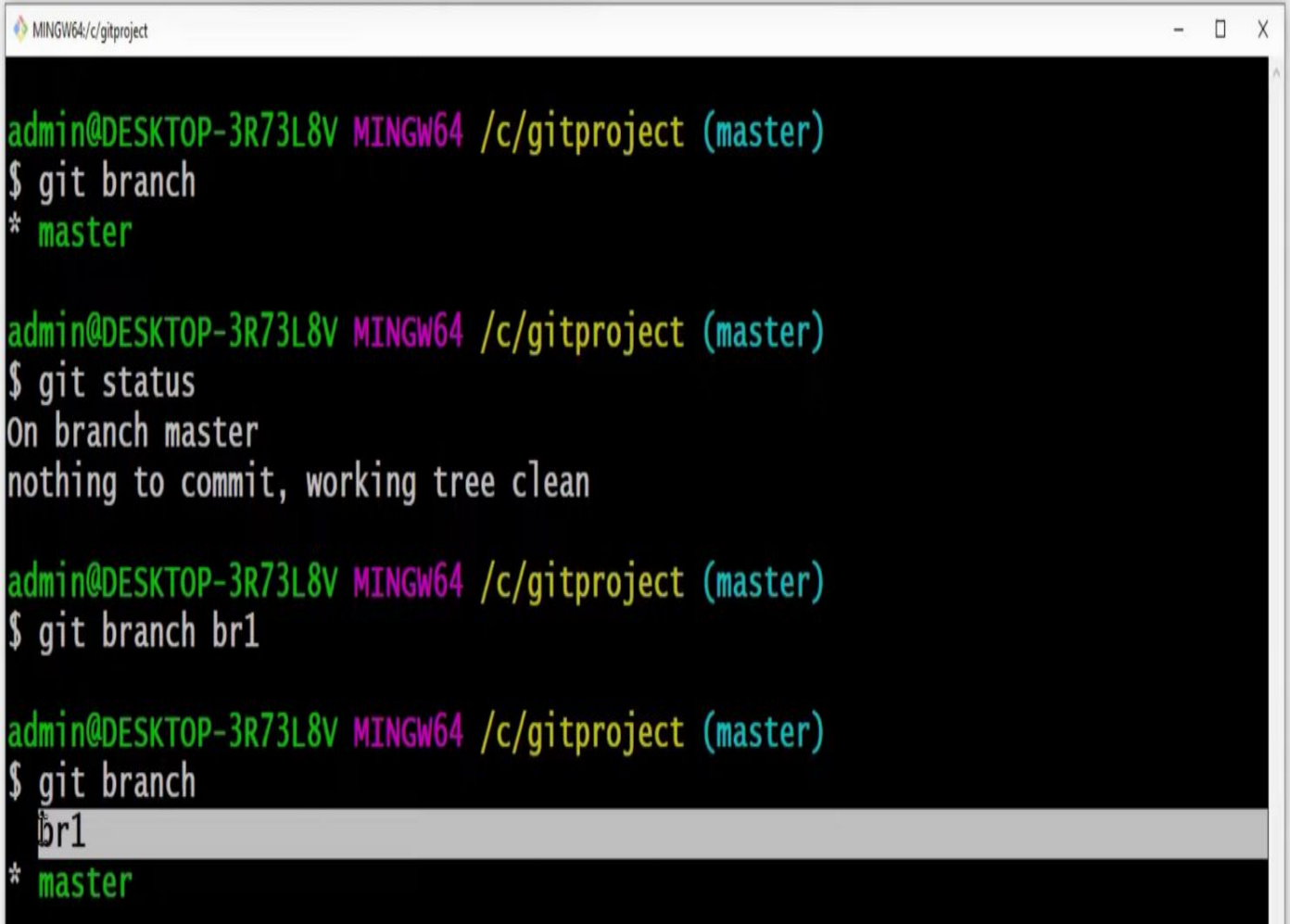
---

1) To view available branches

```
git branch
git status
```

2) Create new branch

```
git branch branchname
```



- Create a branch using the following command:

```
$ git branch new-local-repo
```

- The git branch command also works on remote branches

- A remote repo must first be configured and added to the local repo config

```
$ git remote add new-remote-repo https://github.com/user/repo.git  # Add remote repo to local repo config
$ git push new-local-repo # Pushes the new branch to new-remote-repo
```

- Now, if you want to create a new branch called "subbranch" under the "new-local-repo" follow the steps:

1. Checkout or change to branch "new-local-repo"

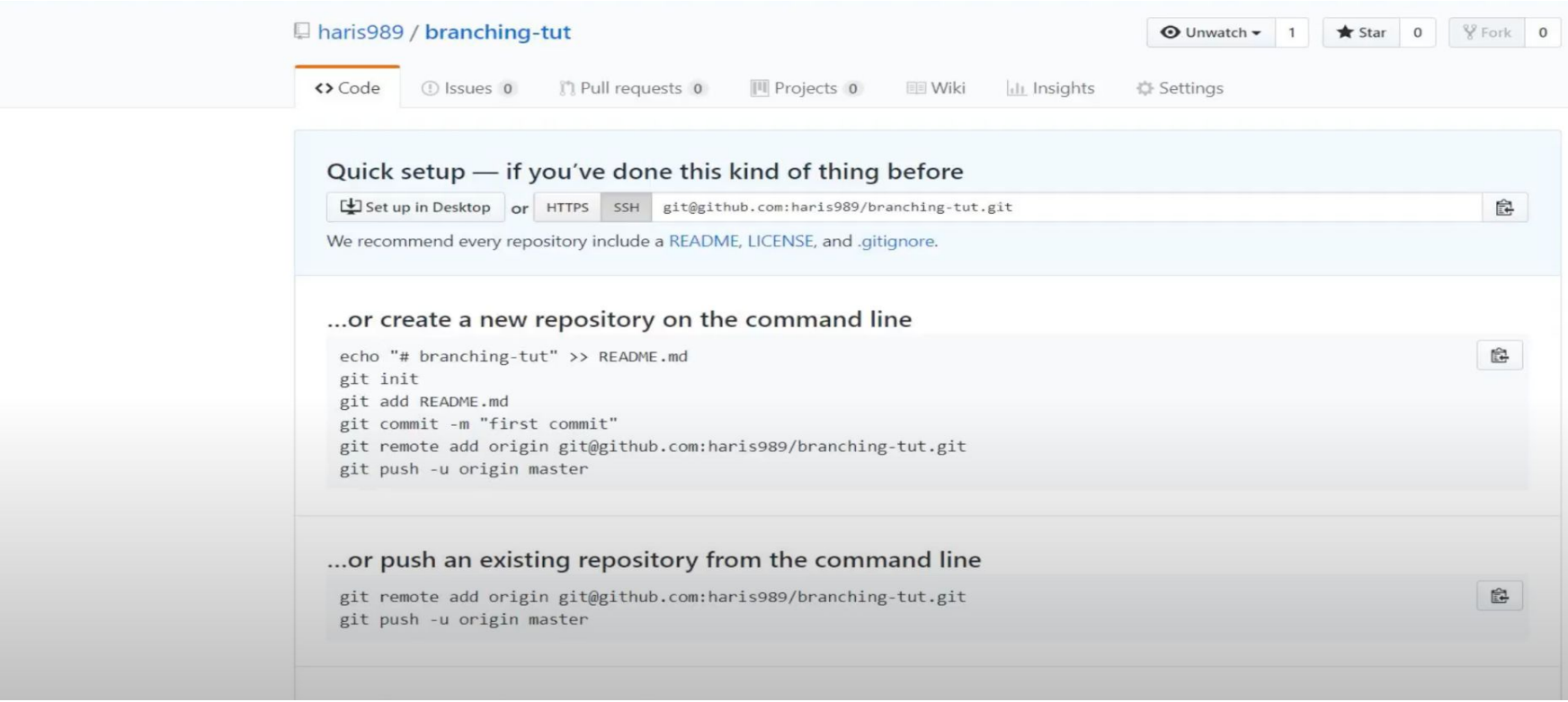
```
$ git checkout new-local-repo
```

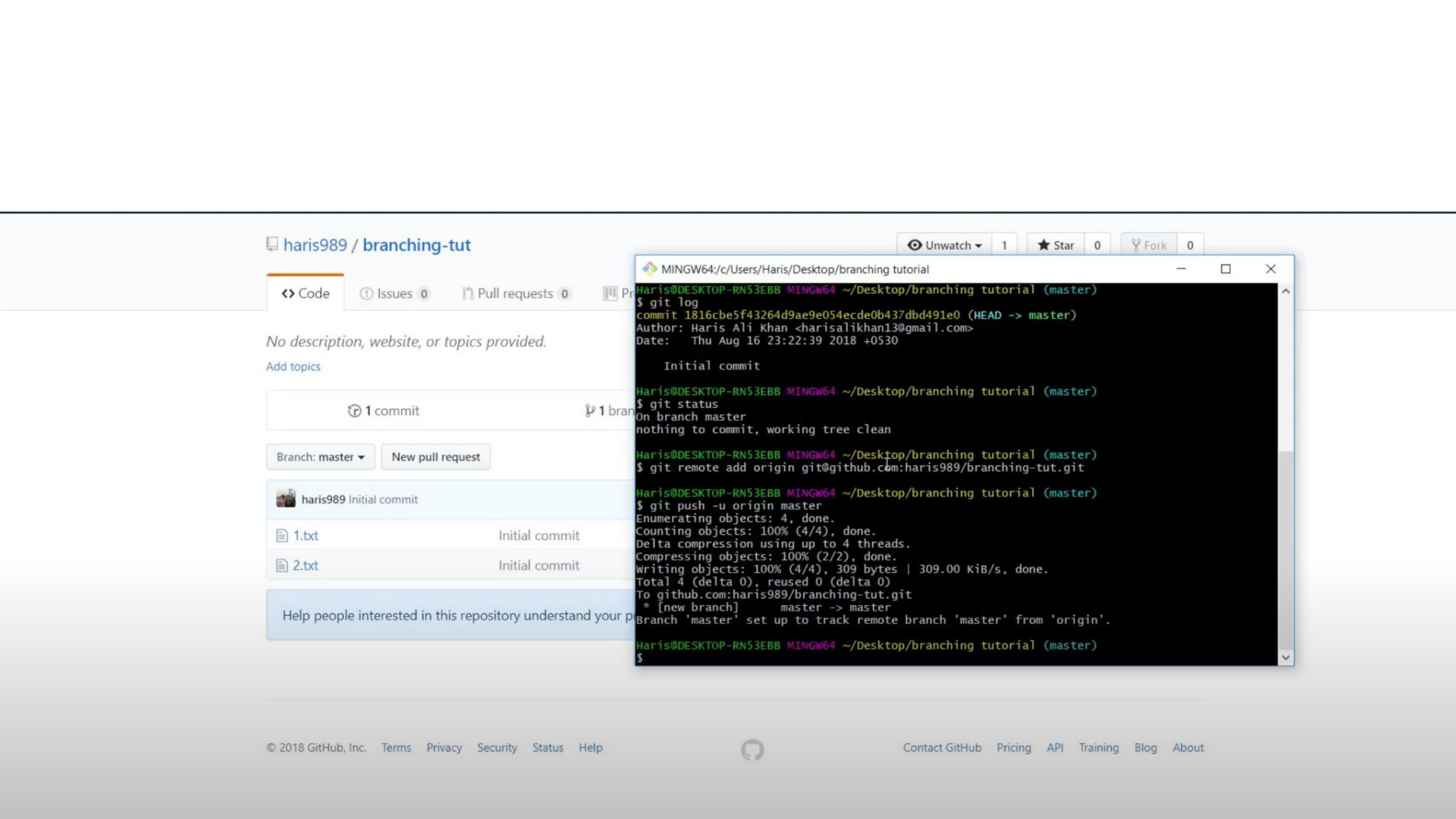
2. Now, create your new branch called "subbranch"

```
$ git checkout -b subbranch new-local-repo
```

3. You can commit and push this branch locally or remotely

# Branching in Git




haris989 / branching-tut
Unwatch 1
Star 0
Fork 0
Code
Issues 0
Pull requests 0
No description, website, or topics provided.
Add topics
1 commit
Branch: master
New pull request
haris989 Initial commit
1.txt Initial commit
2.txt Initial commit
Help people interested in this repository understand your pr
MINGW64:/c/Users/Haris/Desktop/branching tutorial
Haris@DESKTOP-RN53EBB MINGW64 ~/Desktop/branching tutorial (master)
$ git log
commit 1816cbe5f43264d9ae9e054ecde0b437dbd491e0 (HEAD -> master)
Author: Haris Ali Khan <harisalikhan13@gmail.com>
Date: Thu Aug 16 23:22:39 2018 +0530
Initial commit
Haris@DESKTOP-RN53EBB MINGW64 ~/Desktop/branching tutorial (master)
$ git status
On branch master
nothing to commit, working tree clean
Haris@DESKTOP-RN53EBB MINGW64 ~/Desktop/branching tutorial (master)
$ git remote add origin git@github.com:haris989/branching-tut.git
Haris@DESKTOP-RN53EBB MINGW64 ~/Desktop/branching tutorial (master)
$ git push -u origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (4/4), 309 bytes | 309.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0)
To github.com:haris989/branching-tut.git
* [new branch] master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
Haris@DESKTOP-RN53EBB MINGW64 ~/Desktop/branching tutorial (master)
$
© 2018 GitHub, Inc. Terms Privacy Security Status Help
Contact GitHub Pricing API Training Blog About

teluskotraining / tusk

Watch 0 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Settings | Insights

Branch: master | tusk / Recipes.txt | Find file | Copy path

teluskotraining Regards — f47eac7 2 hours from now

1 contributor

12 lines (5 sloc) | 70 Bytes | Raw | Blame | History

```
1
2
3  Welcome to Telusko Kitchen
4
5  Kheer
6  Pasta
7  Chicken Fry
8
9
10
11 thank you :)
```


GitHub
Merge Conflicts

ravdy / merge-conflict

Unwatch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Actions | Projects 0 | Wiki | Security 0 | Insights | Settings

Branch: master ▾ merge-conflict / myinfo

Find file | Copy path

ravdy Create myinfo e7b0bed 2 minutes ago

1 contributor

4 lines (4 sloc) | 60 Bytes

Raw | Blame | History

```
1 Hello,
2 My name is testuser
3 I am from CityName
4 I like movies
```

ravdy / merge-conflict

Unwatch 1 | Star 0 | Fork 0

Code | Issues 0 | Pull requests 0 | Actions | Projects 0 | Wiki | Security 0 | Insights | Settings

Branch: master ▾ merge-conflict / myinfo

Find file | Copy path

artti2 updated myinfo file with developer2 — 04ddaf0 — 1 minute ago

2 contributors

4 lines (4 sloc) | 59 Bytes

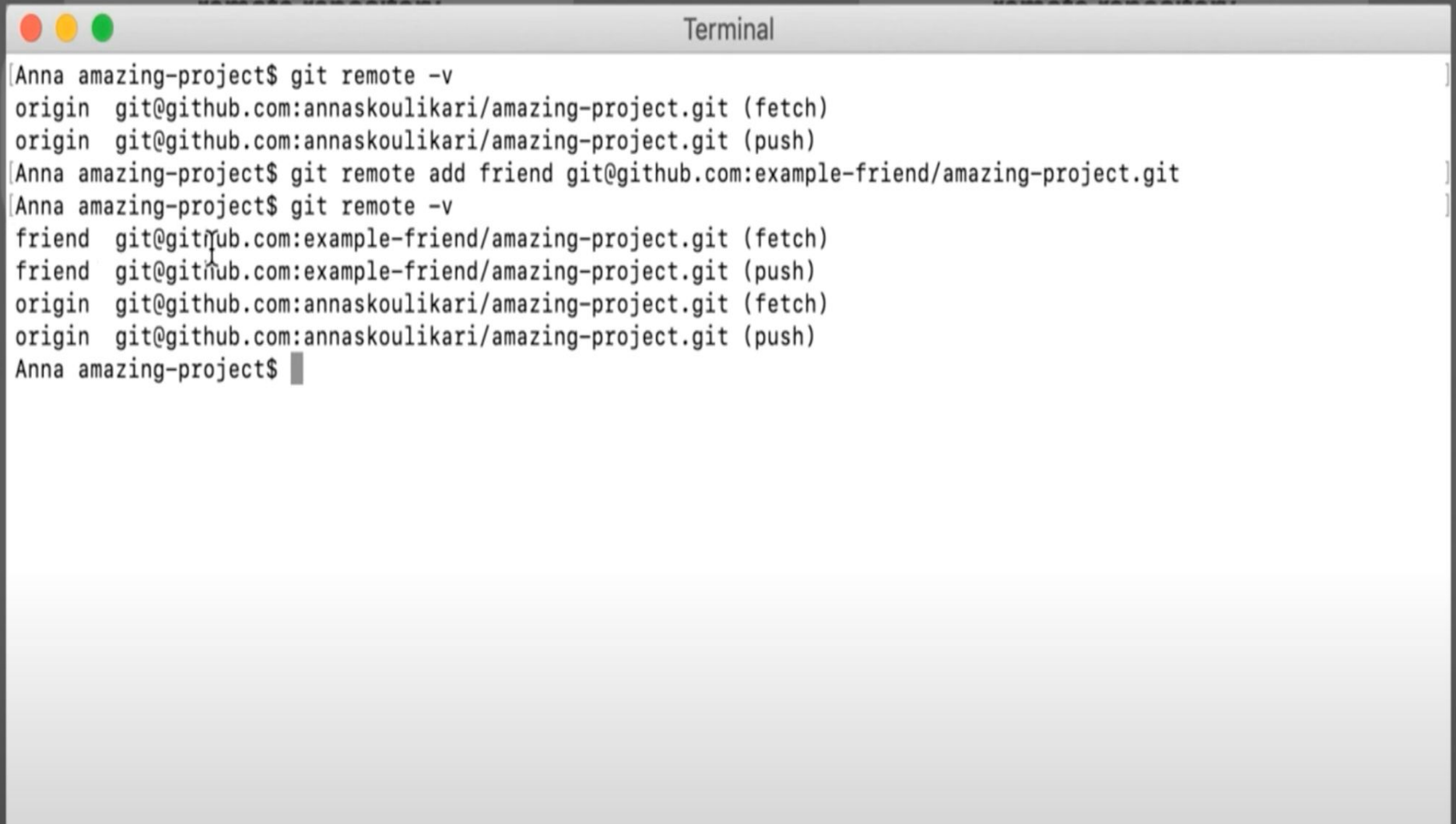
Raw | Blame | History

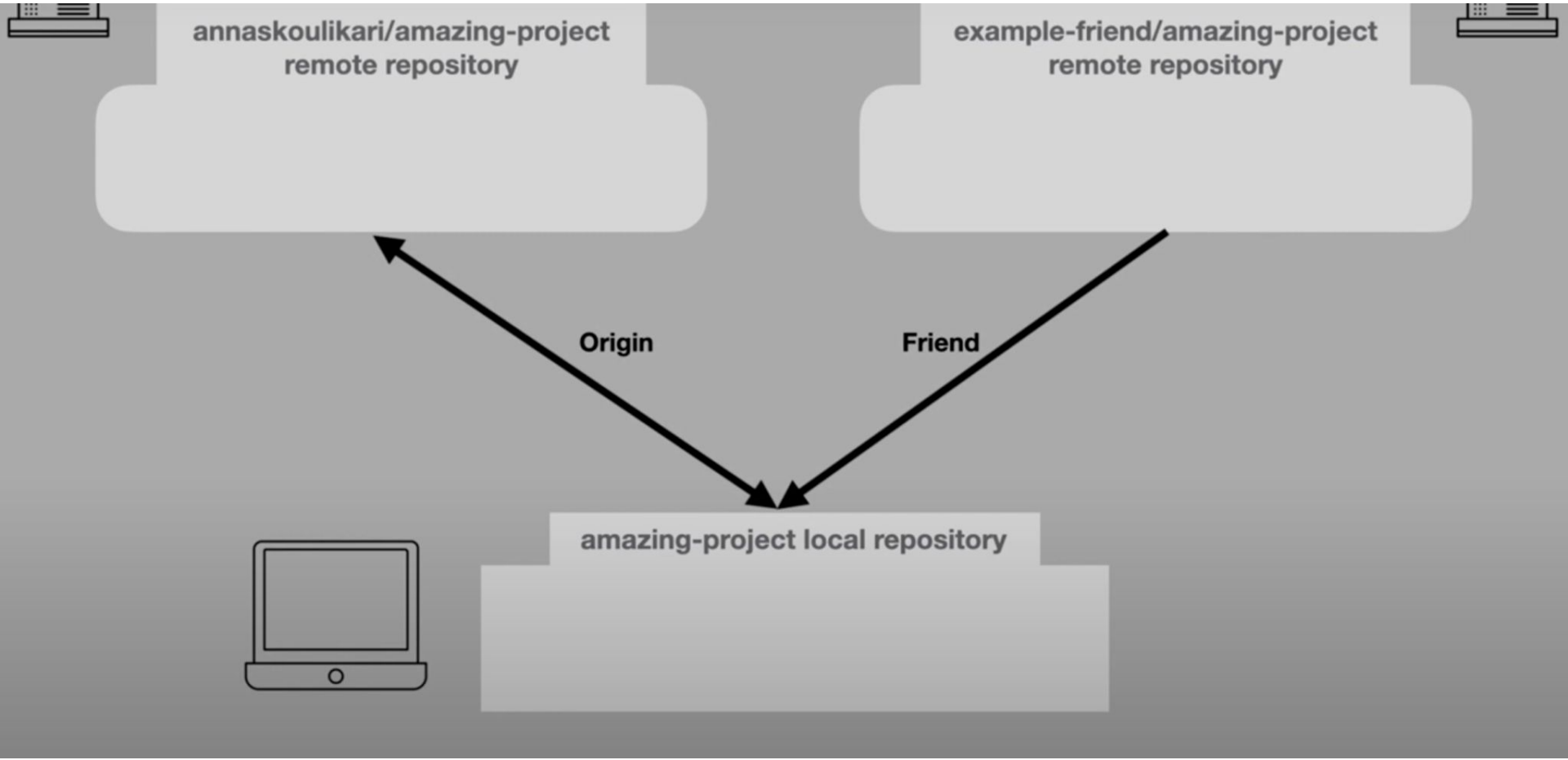
```
1 Hello,
2 My name is developer2
3 I am from Delhi
4 I like movies
```

```
developer1 (dev1branch) merge-conflict
$
gdeveloper1 (dev1branch) merge-conflict
$ git status
On branch dev1branch
nothing to commit, working tree clean
developer1 (dev1branch) merge-conflict
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
developer1 (master) merge-conflict
$ git pull
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
From https://github.com/ravdy/merge-conflict
   e7b0bed..04ddaf0  master     -> origin/master
Updating e7b0bed..04ddaf0
Fast-forward
 myinfo | 4 ++--
 1 file changed, 2 insertions(+), 2 deletions(-)
developer1 (master) merge-conflict
$
```

```
My name is developer2
I am from Delhi
I like movies
developer1 (master) merge-conflict
$ git checkout dev1branch
Switched to branch 'dev1branch'
developer1 (dev1branch) merge-conflict
$ cat myinfo
Hello,
My name is developer1
I am from Bangalore
I like movies
developer1 (dev1branch) merge-conflict
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
developer1 (master) merge-conflict
$ git merge dev1branch
Auto-merging myinfo
CONFLICT (content): Merge conflict in myinfo
Automatic merge failed; fix conflicts and then commit the result.
developer1 (master *+|MERGING) merge-conflict
$ ls
myinfo  README.md
developer1 (master *+|MERGING) merge-conflict
$ cat myinfo
Hello,
<<<<<<< HEAD
My name is developer2
I am from Delhi
=======
My name is developer1
I am from Bangalore
>>>>>>> dev1branch
I like movies
```

```
My name is developer2
I am from Delhi
I like movies
developer1 (master) merge-conflict
$ git checkout dev1branch
Switched to branch 'dev1branch'
developer1 (dev1branch) merge-conflict
$ cat myinfo
Hello,
My name is developer1
I am from Bangalore
I like movies
developer1 (dev1branch) merge-conflict
$ git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
developer1 (master) merge-conflict
$ git merge dev1branch
Auto-merging myinfo
CONFLICT (content): Merge conflict in myinfo
Automatic merge failed; fix conflicts and then commit the result.
developer1 (master *+|MERGING) merge-conflict
$ ls
myinfo  README.md
developer1 (master *+|MERGING) merge-conflict
$ cat myinfo
Hello,
<<<<<<< HEAD
My name is developer2
I am from Delhi
=======
My name is developer1
I am from Bangalore
>>>>>>> dev1branch
I like movies
```


Terminal
Anna amazing-project$ git remote -v
origin  git@github.com:annaskoulikari/amazing-project.git (fetch)
origin  git@github.com:annaskoulikari/amazing-project.git (push)
Anna amazing-project$ git remote add friend git@github.com:example-friend/amazing-project.git
Anna amazing-project$ git remote -v
friend  git@github.com:example-friend/amazing-project.git (fetch)
friend  git@github.com:example-friend/amazing-project.git (push)
origin  git@github.com:annaskoulikari/amazing-project.git (fetch)
origin  git@github.com:annaskoulikari/amazing-project.git (push)
Anna amazing-project$


annaskoulikari/amazing-project
remote repository
example-friend/amazing-project
remote repository
Origin
Friend
amazing-project local repository

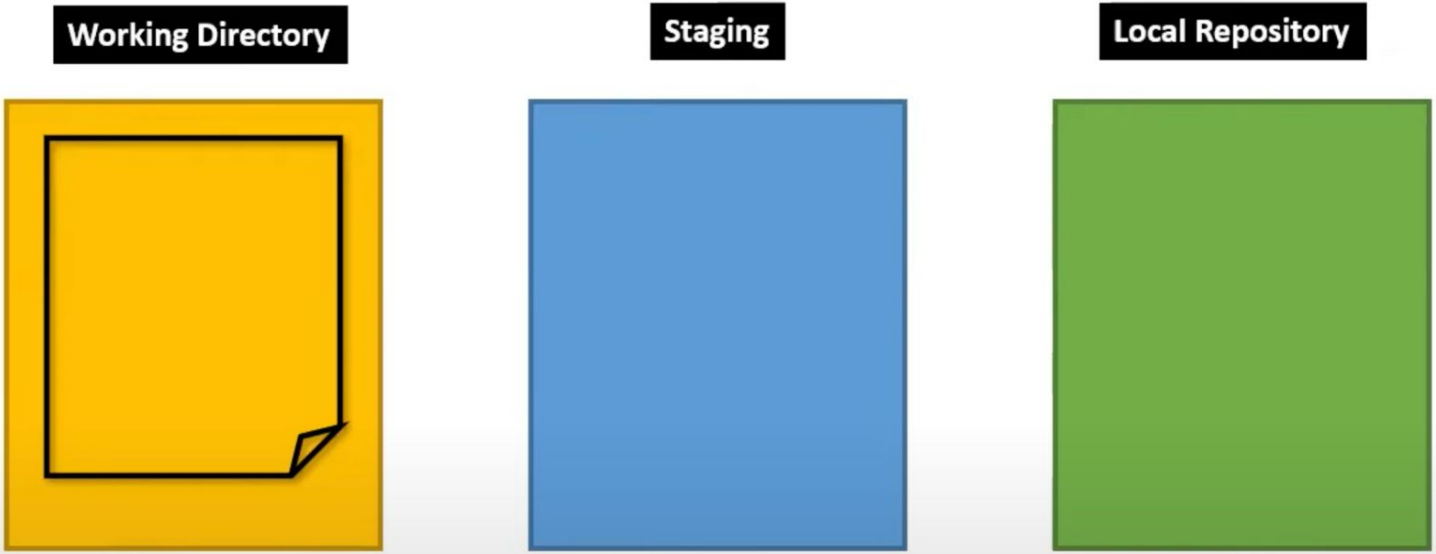
# Git diff is used to compare files in the version control system

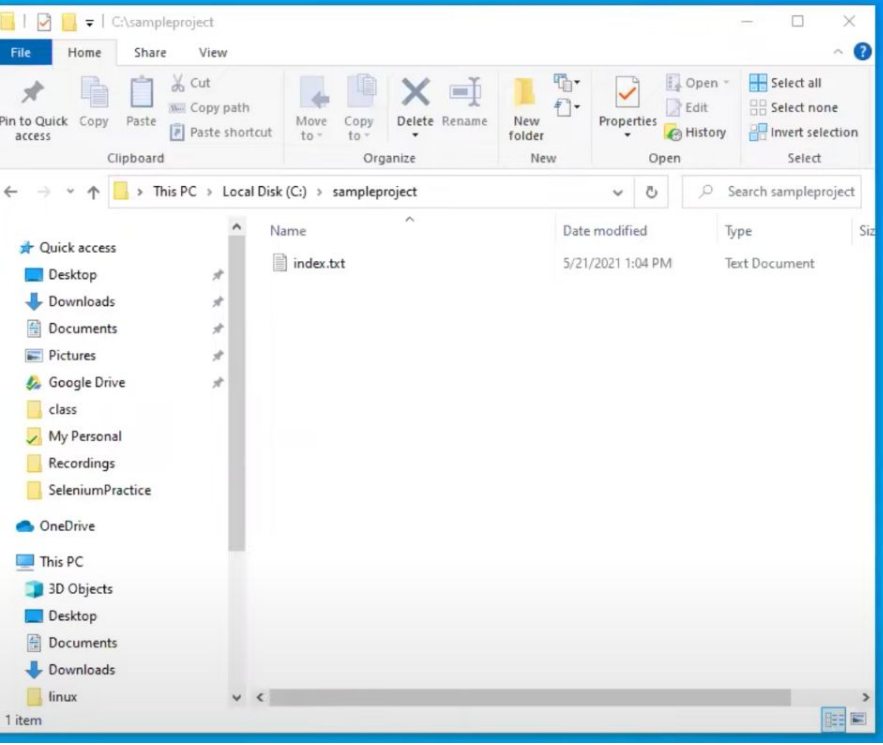
GIT Commands

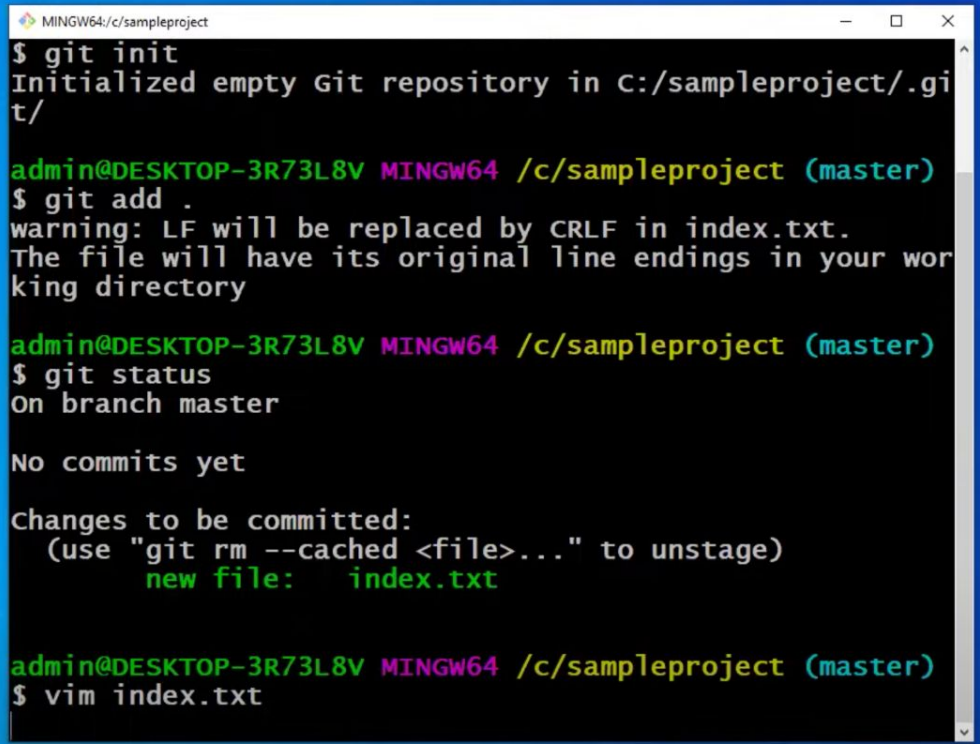
---

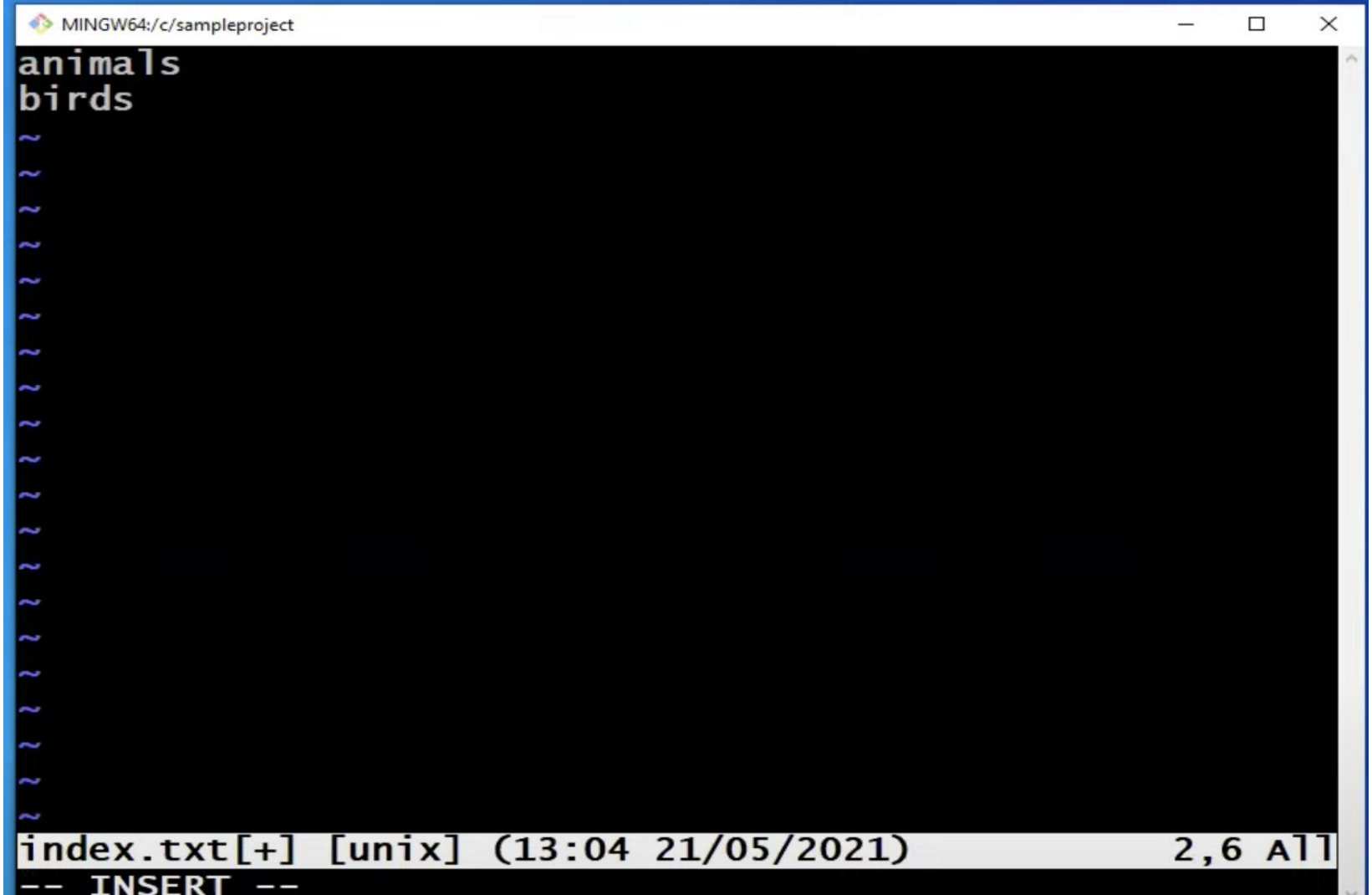
git diff

Working dir --->staging---->local repo ---> remote rep


Working Directory
Staging
Local Repository


C:\sampleproject
File
Home
Share
View
Pin to Quick access
Copy
Paste
Cut
Copy path
Paste shortcut
Clipboard
Move to
Copy to
Delete
Rename
Organize
New folder
New
Properties
Open
Edit
History
Open
Select all
Select none
Invert selection
Select
This PC › Local Disk (C:) › sampleproject
Search sampleproject
Quick access
Desktop
Downloads
Documents
Pictures
Google Drive
class
My Personal
Recordings
SeleniumPractice
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
linux
Name
Date modified
Type
index.txt
5/21/2021 1:04 PM
Text Document
1 item


MINGW64:/c/sampleproject
$ git init
Initialized empty Git repository in C:/sampleproject/.git/
admin@DESKTOP-3R73L8V MINGW64 /c/sampleproject (master)
$ git add .
warning: LF will be replaced by CRLF in index.txt.
The file will have its original line endings in your working directory
admin@DESKTOP-3R73L8V MINGW64 /c/sampleproject (master)
$ git status
On branch master
No commits yet
Changes to be committed:
(use "git rm --cached <file>..." to unstage)
new file: index.txt
admin@DESKTOP-3R73L8V MINGW64 /c/sampleproject (master)
$ vim index.txt


MINGW64:/c/sampleproject
animals
birds
index.txt[+] [unix] (13:04 21/05/2021) 2,6 All
-- INSERT --

GIT Commands

---

git diff

Working dir --->staging---->local repo ---> remote rep

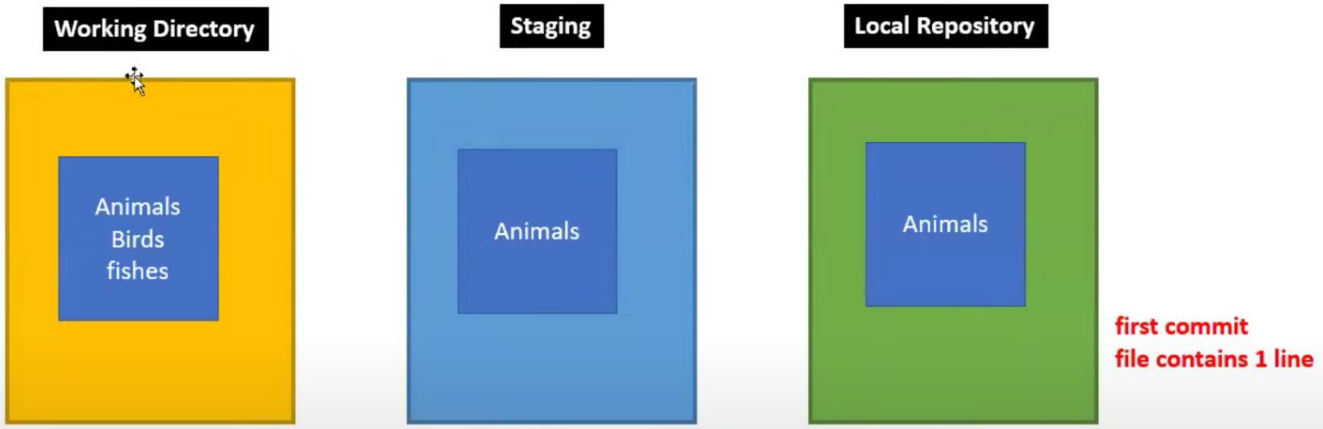
Req1: To see the difference in File Content between Working Directory and staging Area

git diff index.txt

Req1: To see the difference in File Content between Working Directory and staging Area

```
git diff index.txt

diff --git a/index.txt b/index.txt
index fcb5845..b5bf6bc 100644
--- a/index.txt
+++ b/index.txt
@@ -1 +1,2 @@
 animals
+birds

a/index.txt    --> represents source( staging area)
b/index.txt    --> represents destination (working dir)

fcb5845  --> Hash of file content from source/staging
b5bf6bc   -->  Hash of file content from  destination/workspce

100644  -- git file mode
        100  -- represents type of the file
        644 - File permissions      rw-r-r
                4 -r
                2 - w
                1 - e
--- a/index.txt   Source file missing some lines (staging)
+++ b/index.txt  New lines added in destination file (working dir)
```

1


Working Directory
Staging
Local Repository
Animals
Birds
fishes
Animals
Animals
first commit
file contains 1 line

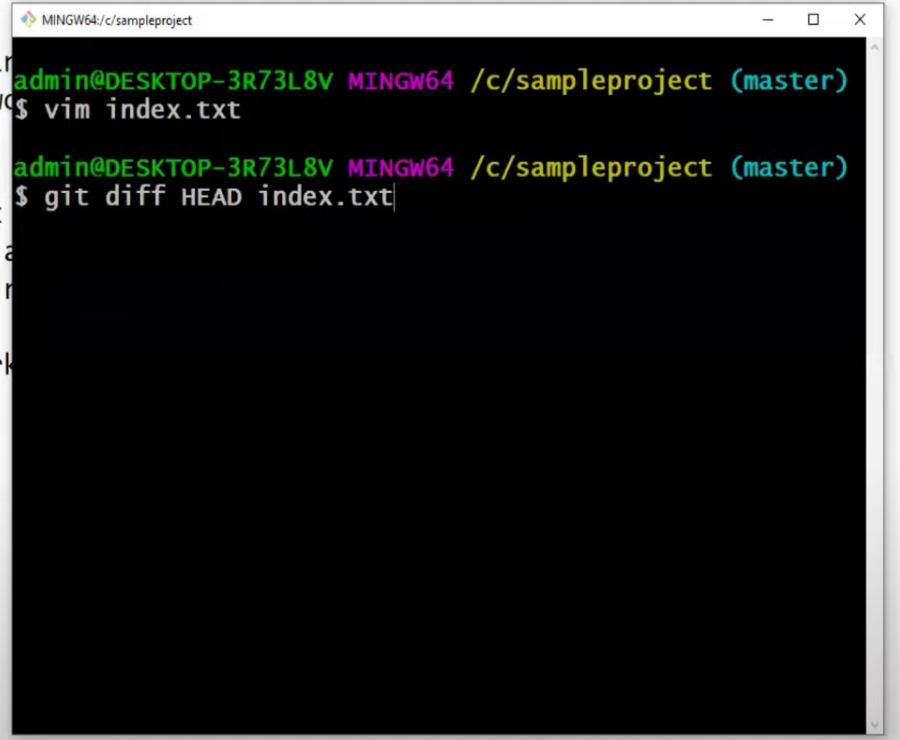
100644  -- git file mode
        100  -- represents type of the file
        644 - File permissions      rw-r-r
                4 -r
                2 - w
                1 - e
--- a/index.txt   Source file missing some lines (stagi
+++ b/index.txt  New lines added in destination file (w

                if anyline prefixed with space means it
                if anyline prefixed with + means it is a
                if anyline prefixed with - means it is r

Req2: To see the difference in File Content between Work

* Last commit refred using HEAD

git diff HEAD index.txt



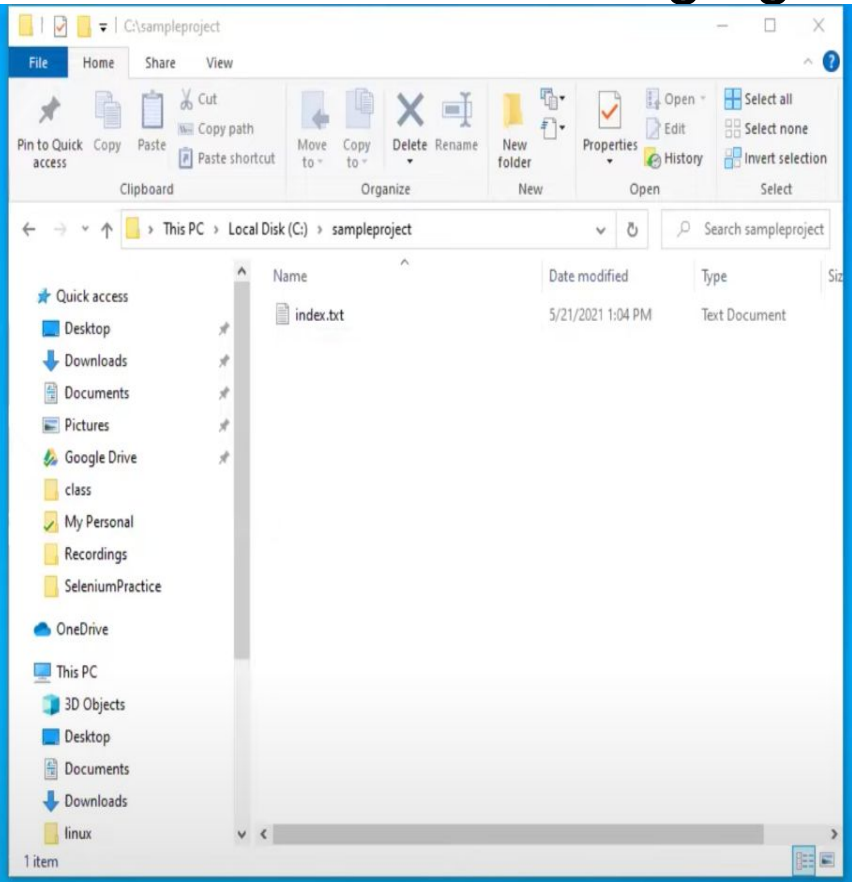
Working Directory

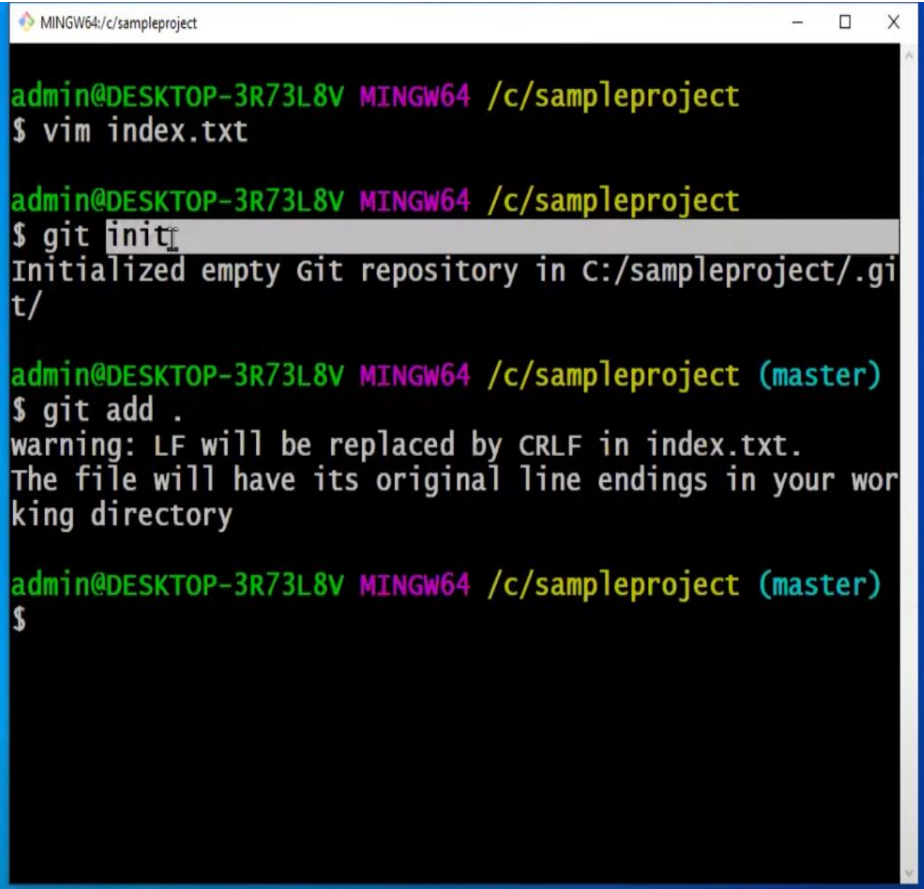
Animals

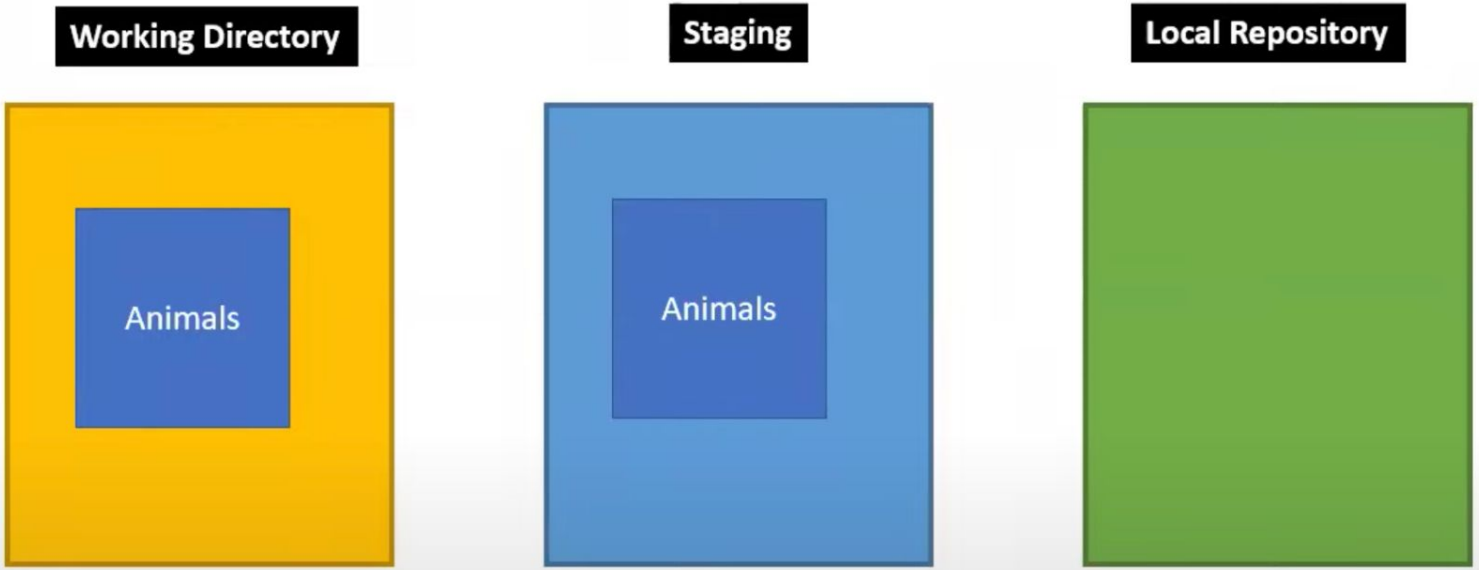
MINGW64:/c/sampleproject

```
admin@DESKTOP-3R73L8V MINGW64 /c/sampleproject
$ vim index.txt

admin@DESKTOP-3R73L8V MINGW64 /c/sampleproject
$
```

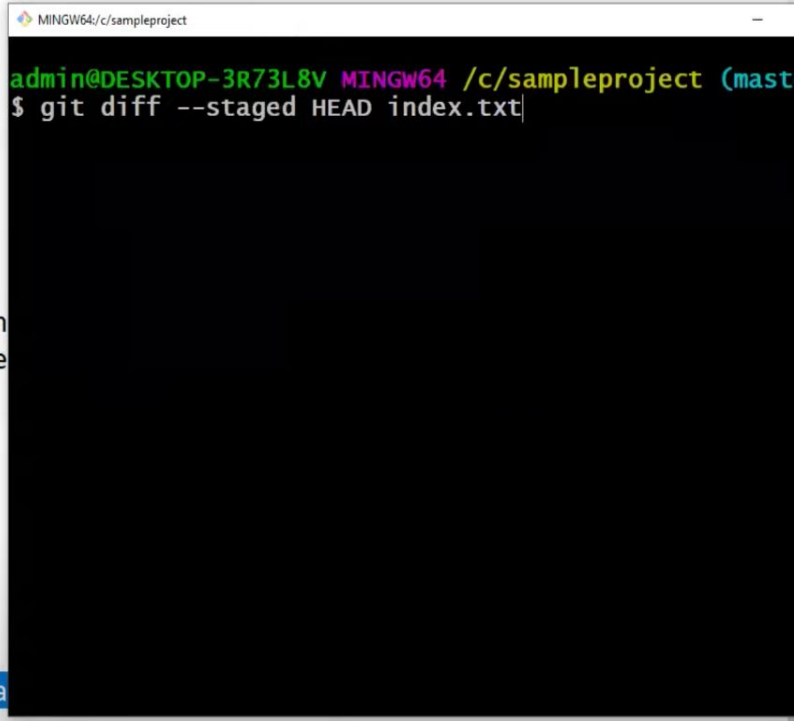
# Add the files into staging area



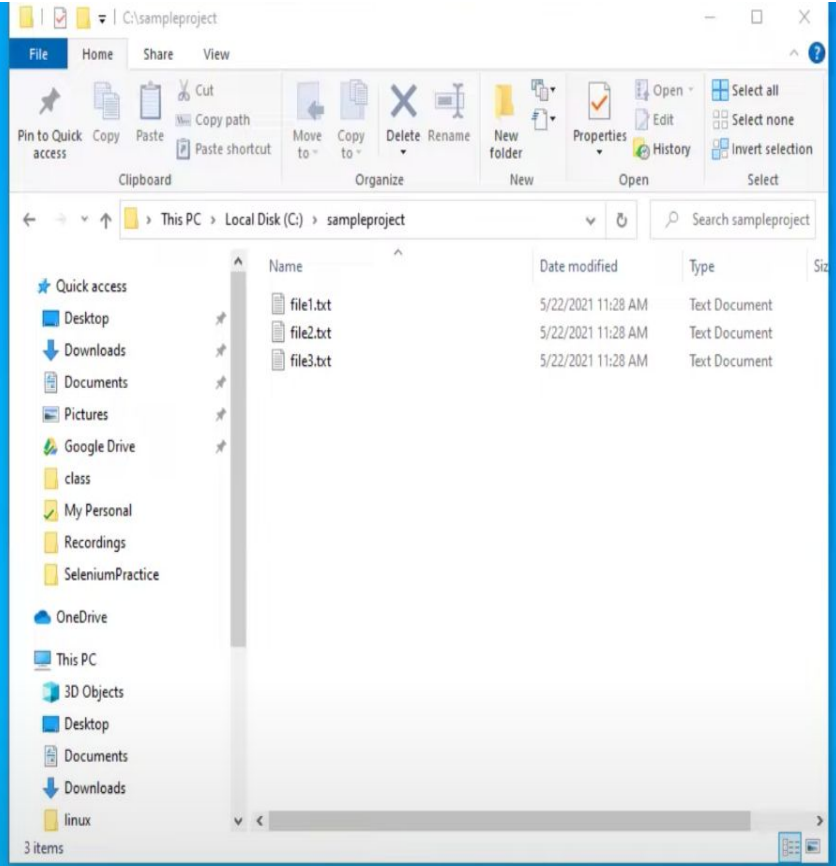



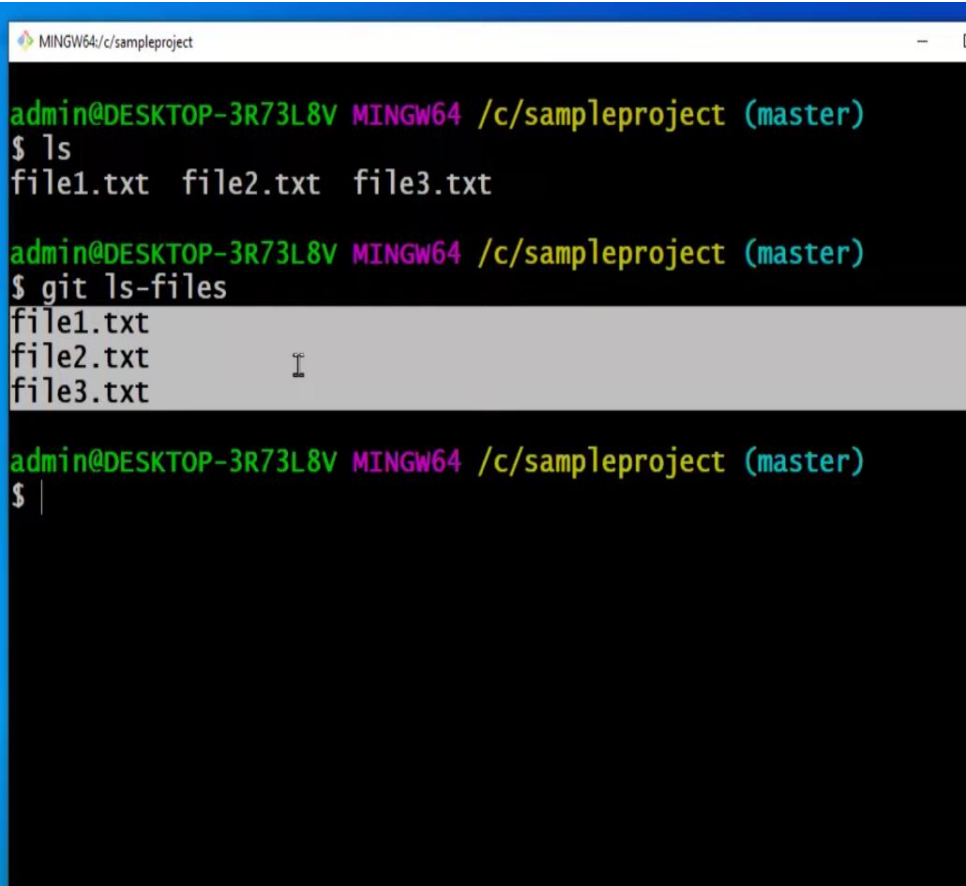

Working Directory
Animals
Staging
Animals
Local Repository

```
100644  -- git file mode
        100  -- represents type of the file
        644 - File permissions      rw-r-r
                4 -r
                2 - w
                1 - e
--- a/index.txt   Source file missing some lines (staging)
+++ b/index.txt  New lines added in destination file (working dir)


                if anyline prefixed with space means it is unchanged.
                if anyline prefixed with + means it is added in destin
                if anyline prefixed with - means it is removed from de

Req2: To see the difference in File Content between Working Directory

* Last commit refred using HEAD

git diff HEAD index.txt


Req3: To see the difference in File Content between staged Copy and La

git diff  --staged HEAD index.txt
```



# Removing Files by using git rm Command



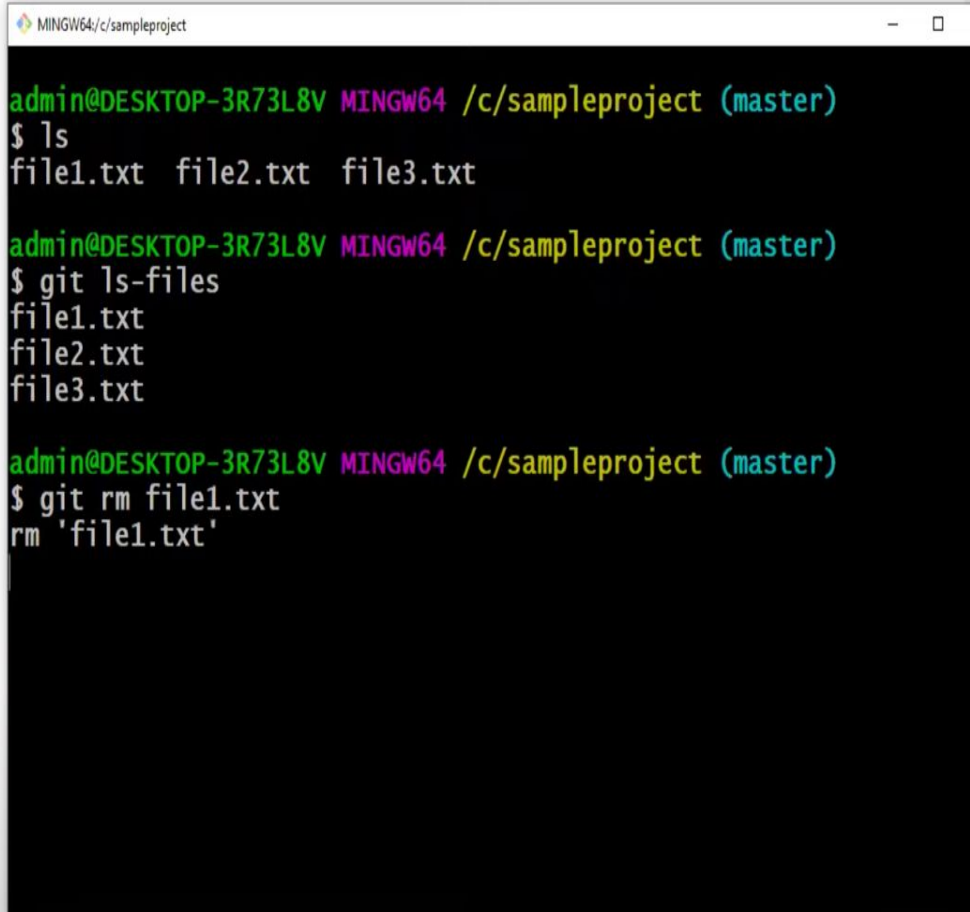
Git ls-files list all the files in the staging area

Git Commands

---

git rm

Req1: Remove files from both staging & working directory



Git Commands

---

git rm

Req1: Remove files from both staging & working directory

```
git rm file1.txt
git rm  -r .
```

Git Commands
---
git rm

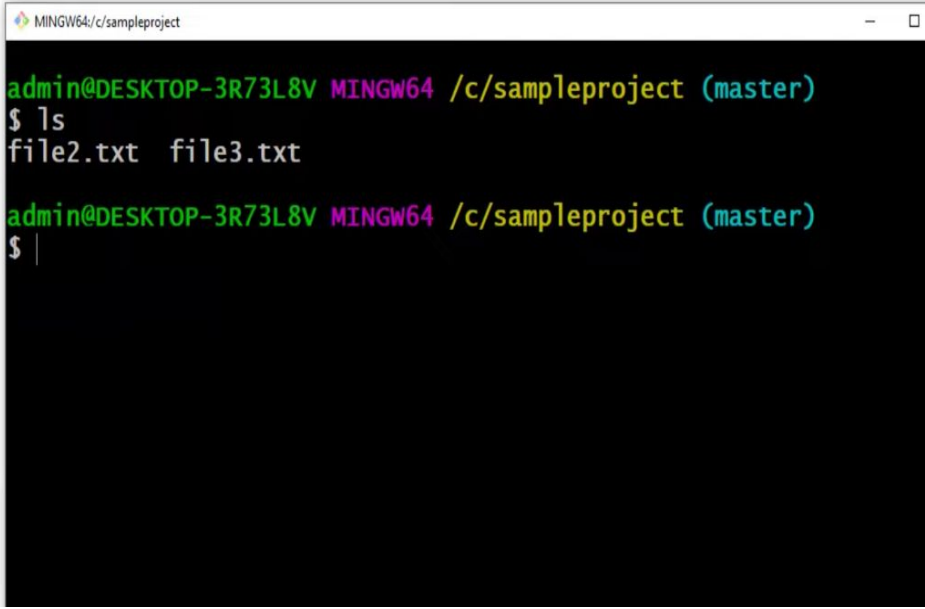
Req1: Remove files from both staging & working directory

```
git rm file1.txt
git rm  -r .
```

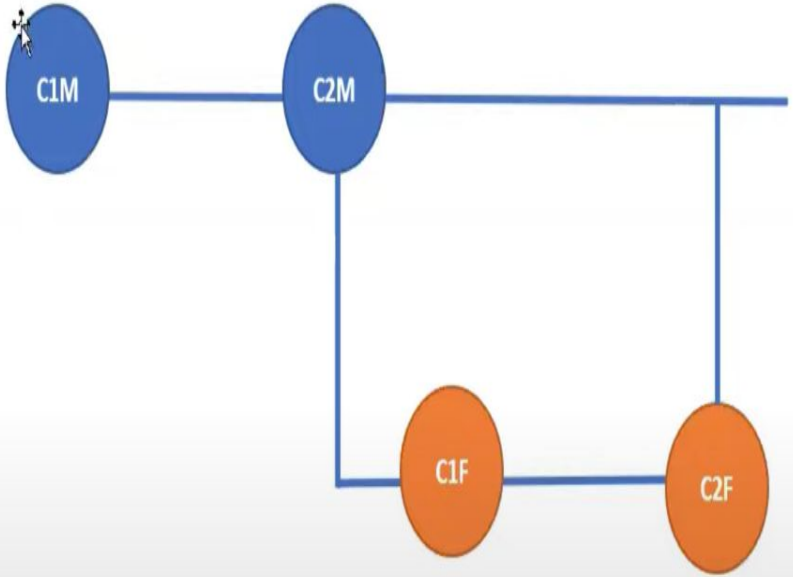
Req2: Remove files from only from staging

```
git rm --cached file2.txt
```

Req3: Remove files only working directory

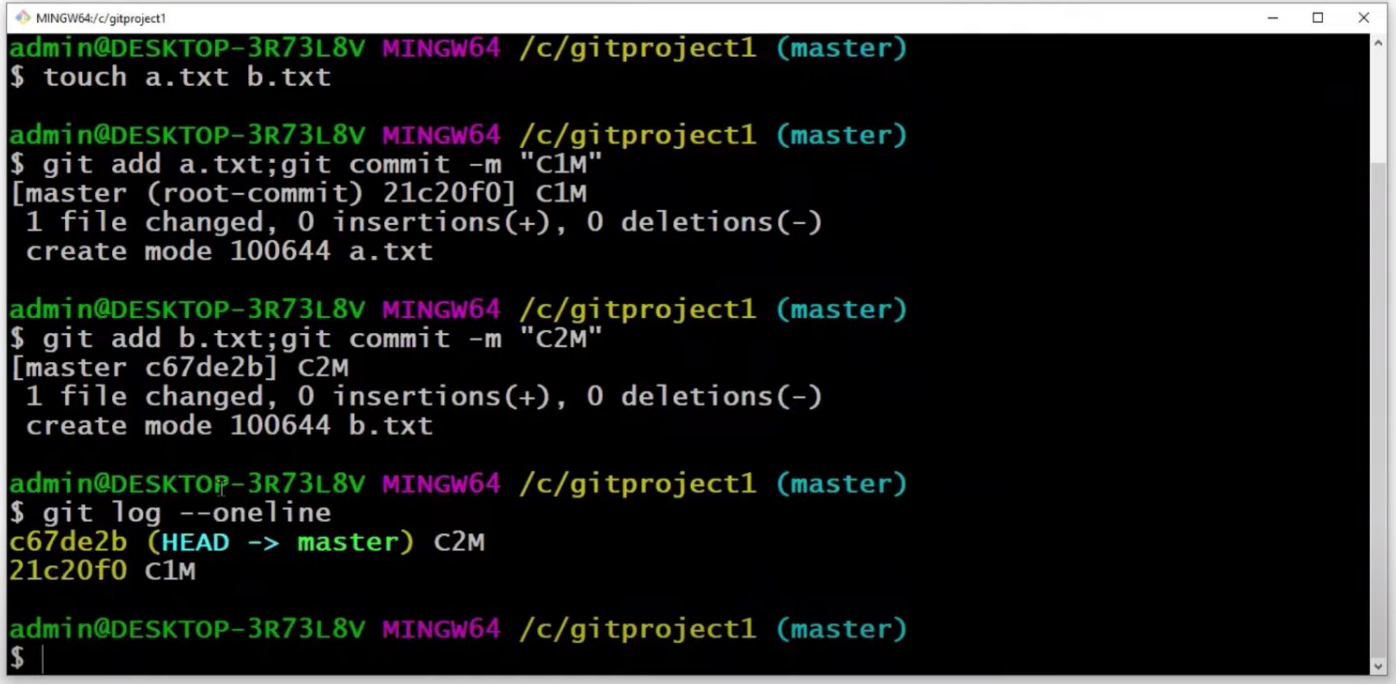


# Fast-forward Merge



# Merging

Merging
---



Merging
---

Master -----> a.txt

Feature ---> x.txt y

Merging

---

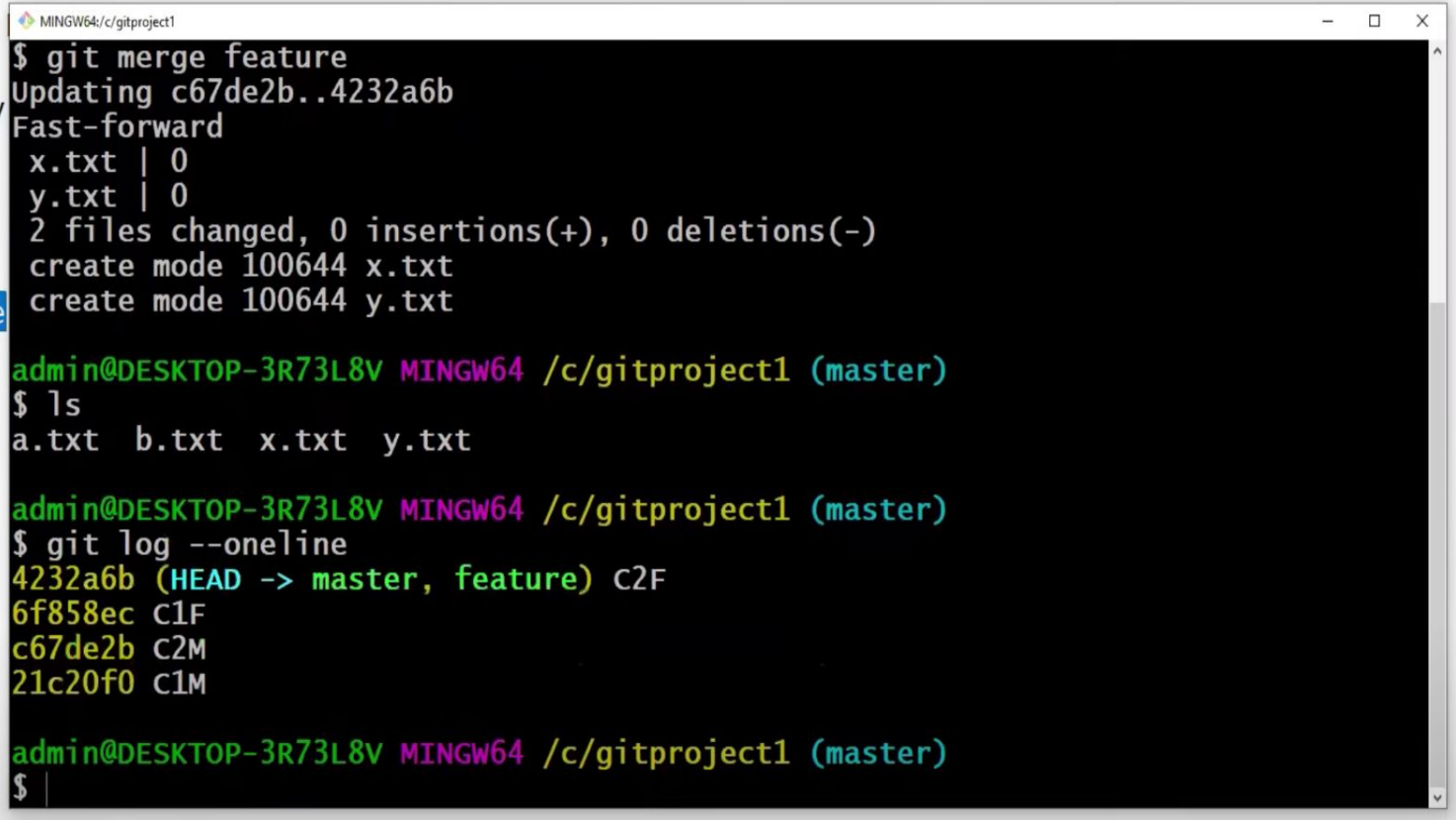
Master -----> a.txt

Feature ---> x.txt  y

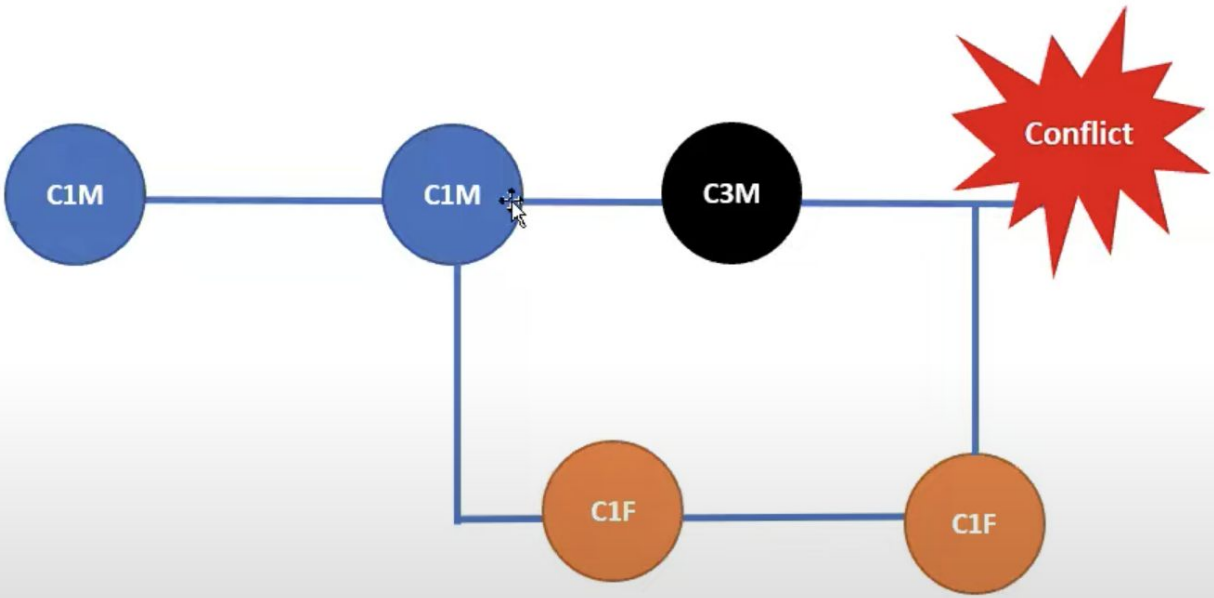
Master

git merge feature

1) Fast-forward merge

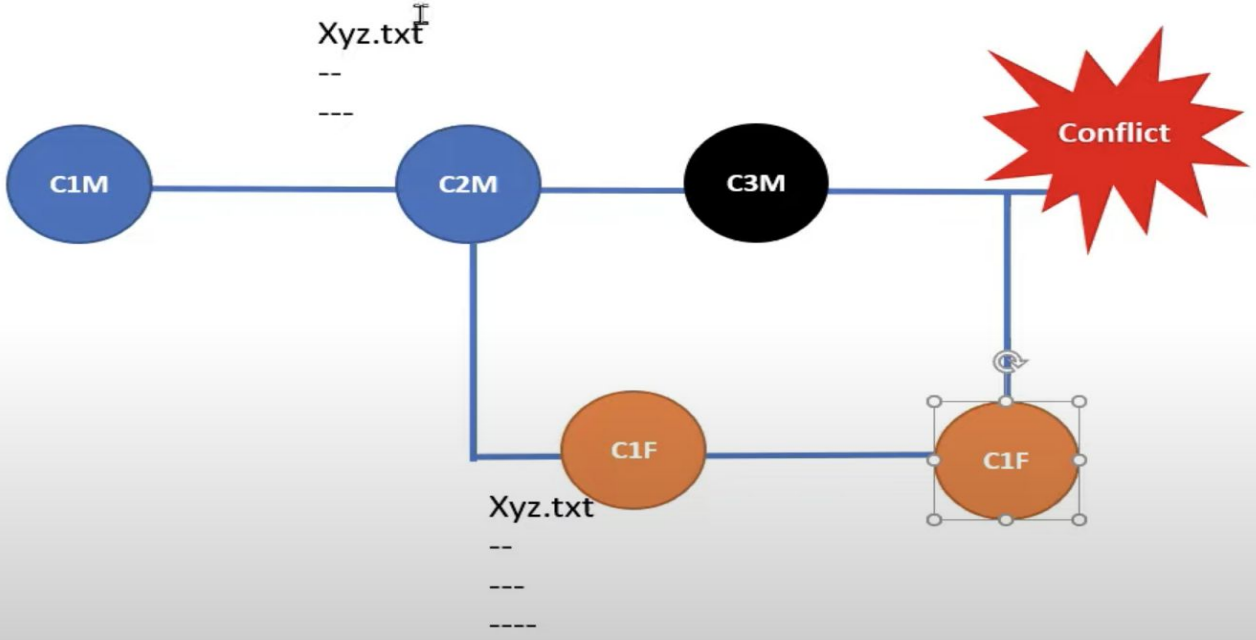


# Three-way Merge



If we do changes for the existing files there will be definitely conflict with master branch

## Three-way Merge



Merging
---

Master -----> a.txt  b.txt

Feature ---> x.txt  y.txt

Master
  git merge feature

1) Fast-forward merge
2) Three-way merge

Master -- 2 files, 2 commits
Feature -- 4 files,  4 commits

Merging
---

Master -----> a.txt  b.txt

Feature ---> x.txt  y.txt

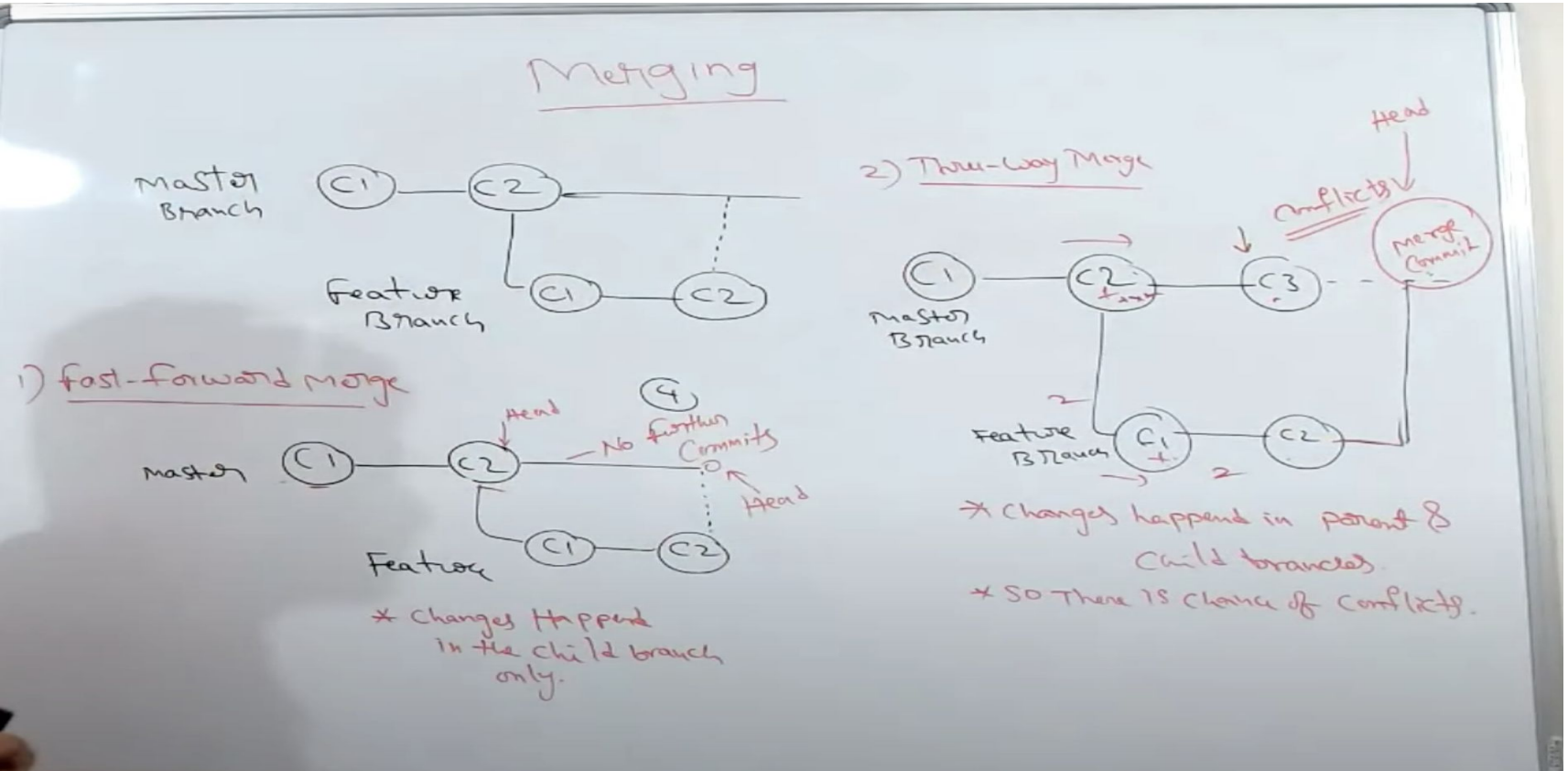
Master
  git merge feature

1) Fast-forward merge
2) Three-way merge

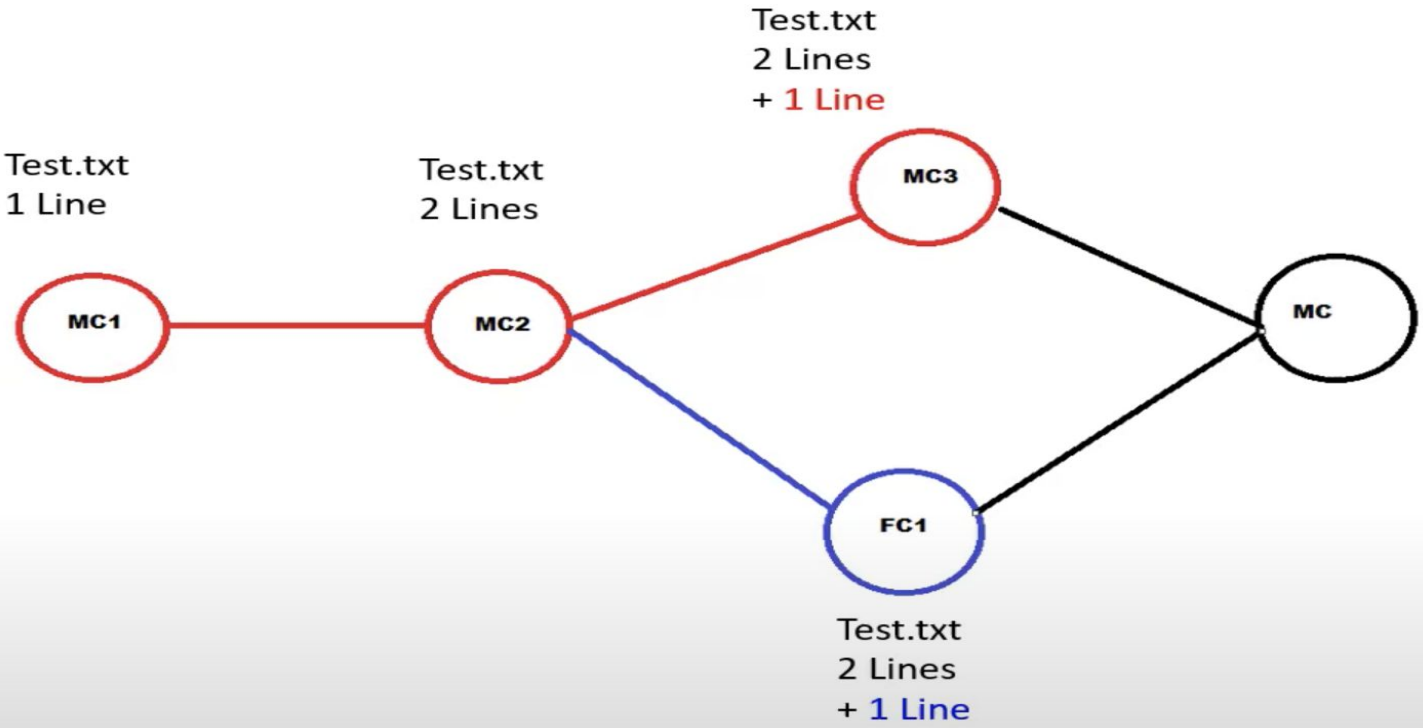
Master -- 2 files, 2 commits
Feature -- 4 files,  4 commits

```
admin@DESKTOP-3R73L8V MINGW64 /c/gitproject2 (master)
$ git merge feature
Merge made by the 'recursive' strategy.
 x.txt | 0
 y.txt | 0
 2 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 x.txt
 create mode 100644 y.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject2 (master)
$
```

# Git merge

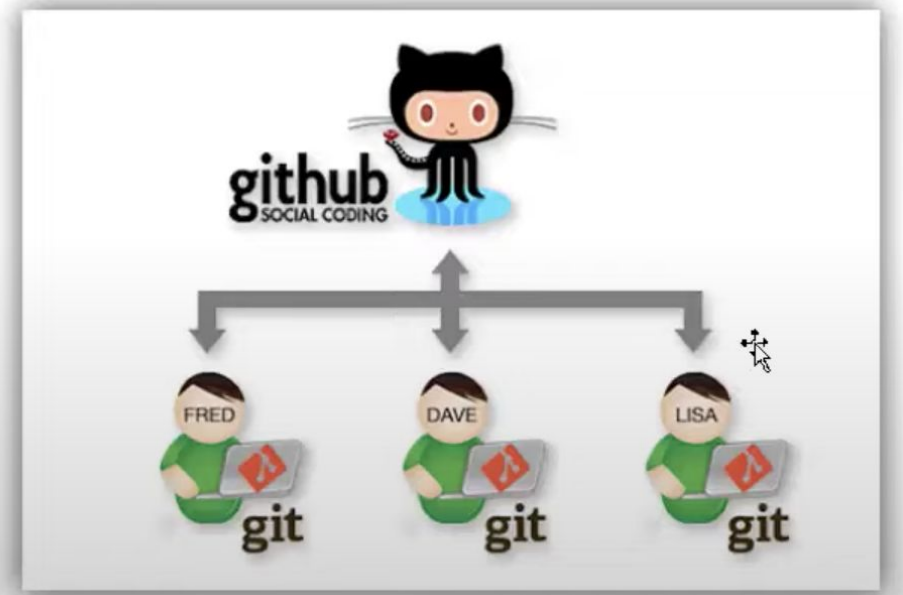

Merging
Master Branch
C1
C2
Feature Branch
C1
C2
1) fast-forward merge
Head
4
No further Commits
Master
C1
C2
Head
Feature
C1
C2
* Changes Happend in the child branch only.
2) Three-way Merge
Head
Conflicts
Merge Commit
C1
C2
C3
Master Branch
2
Feature Branch
C1
C2
2
* Changes happend in parent & child branches
* So There is chance of conflicts.

# Merge Conflict



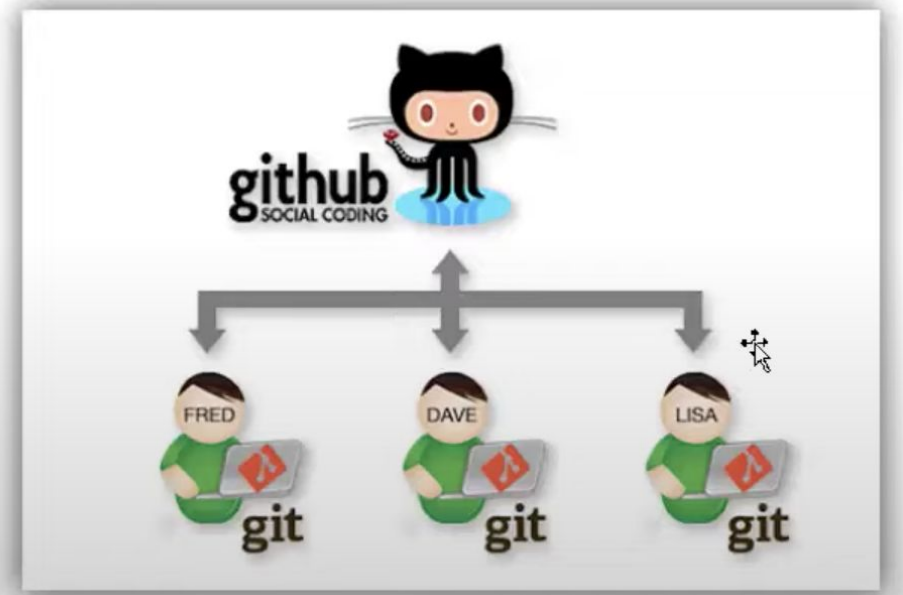
# Github

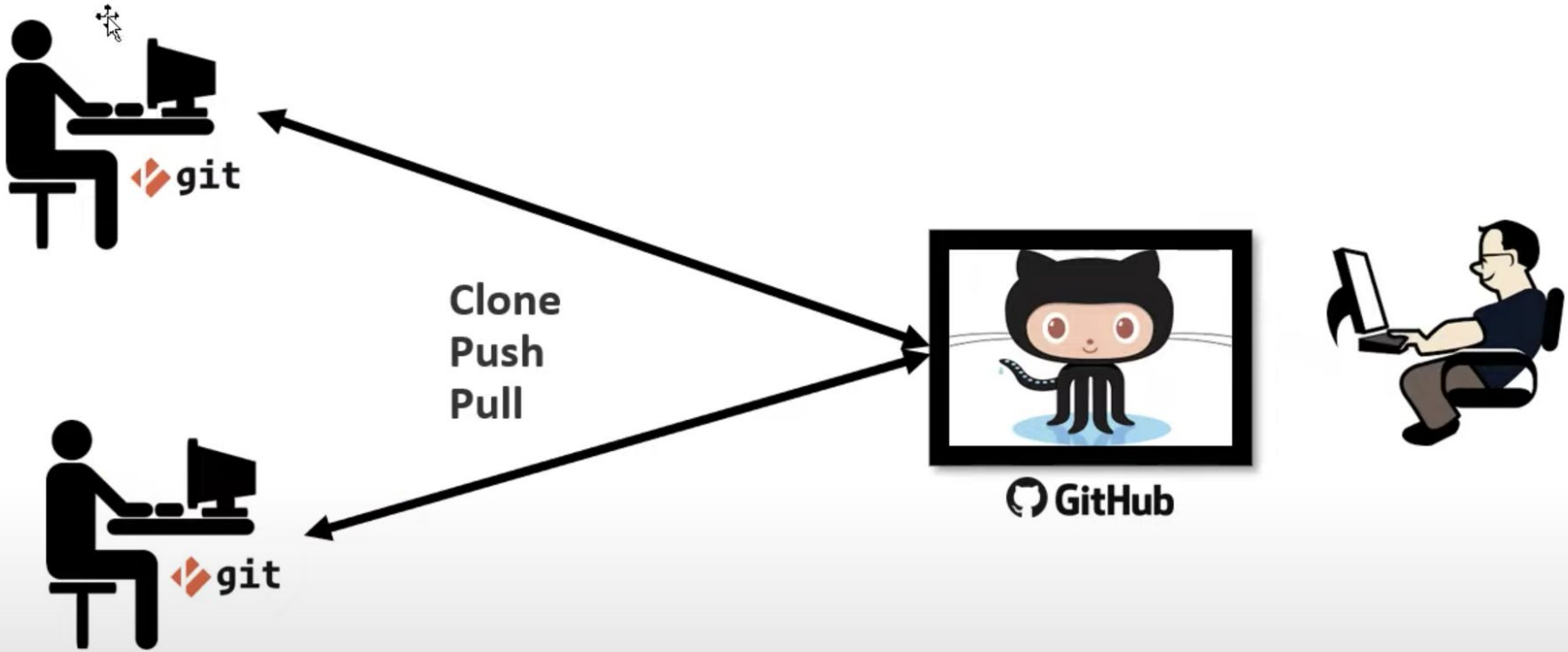
- GitHub is a hosting service for git repositories.
- Git is the tool, while GitHub is the service to use git.



# Github

- GitHub is a hosting service for git repositories.
- Git is the tool, while GitHub is the service to use git.



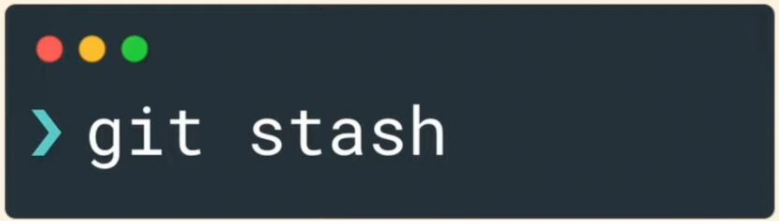

git
Clone
Push
Pull
GitHub
git

```
$ git checkout master
Switched to branch 'master'

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ cat text.txt
cat: text.txt: No such file or directory

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ cat test.txt
this is my first line
this is my second line

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ vim test.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git add test.txt; git commit -m "MC3"
[master f3bbeb9] MC3
 1 file changed, 1 insertion(+)

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git log --oneline
f3bbeb9 (HEAD -> master) MC3
259c46c MC2
2b62d05 MC1

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git merge feature
Auto-merging test.txt
CONFLICT (content): Merge conflict in test.txt
Automatic merge failed; fix conflicts and then commit the result.

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master|MERGING)
$
```

# Git Stash

**git stash** is super useful command that helps you save changes that you are not yet ready to commit. You can stash changes and then come back to them later.

Running git stash will take all uncommitted changes (staged and unstaged) and stash them, reverting the changes in your working copy.




You can also use **git stash save** instead

```
admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ vim test.txt

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git add test.txt; git commit -m "MC3"
[master f3bbeb9] MC3
 1 file changed, 1 insertion(+)

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git log --oneline
f3bbeb9 (HEAD -> master) MC3
259c46c MC2
2b62d05 MC1

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master)
$ git merge feature
Auto-merging test.txt
CONFLICT (content): Merge conflict in test.txt
Automatic merge failed; fix conflicts and then commit the result.

admin@DESKTOP-3R73L8V MINGW64 /c/gitproject (master|MERGING)
$ cat test.txt
this is my first line
this is my second line
<<<<<<< HEAD
added new line by master
=======
added new line by feature
>>>>>>> feature
```