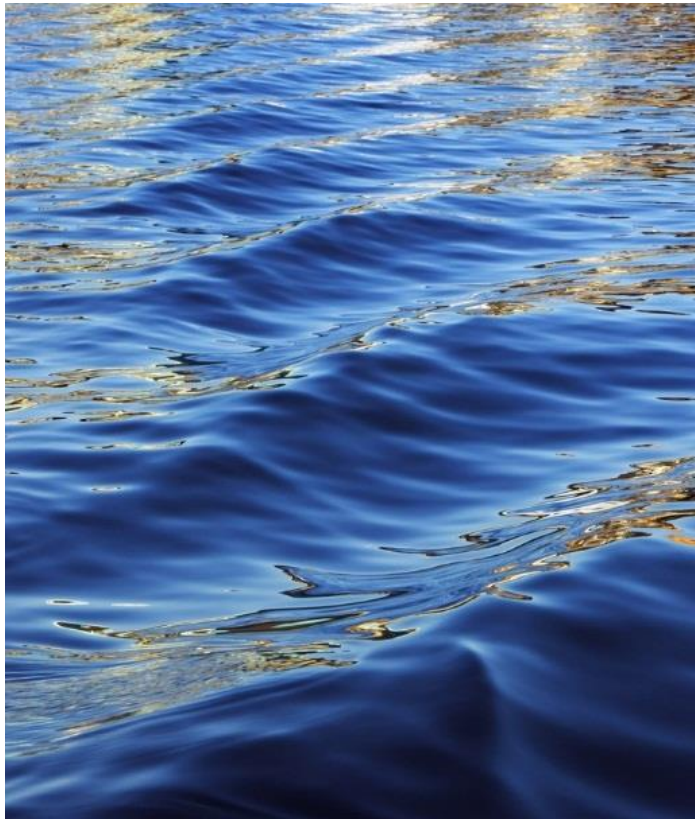
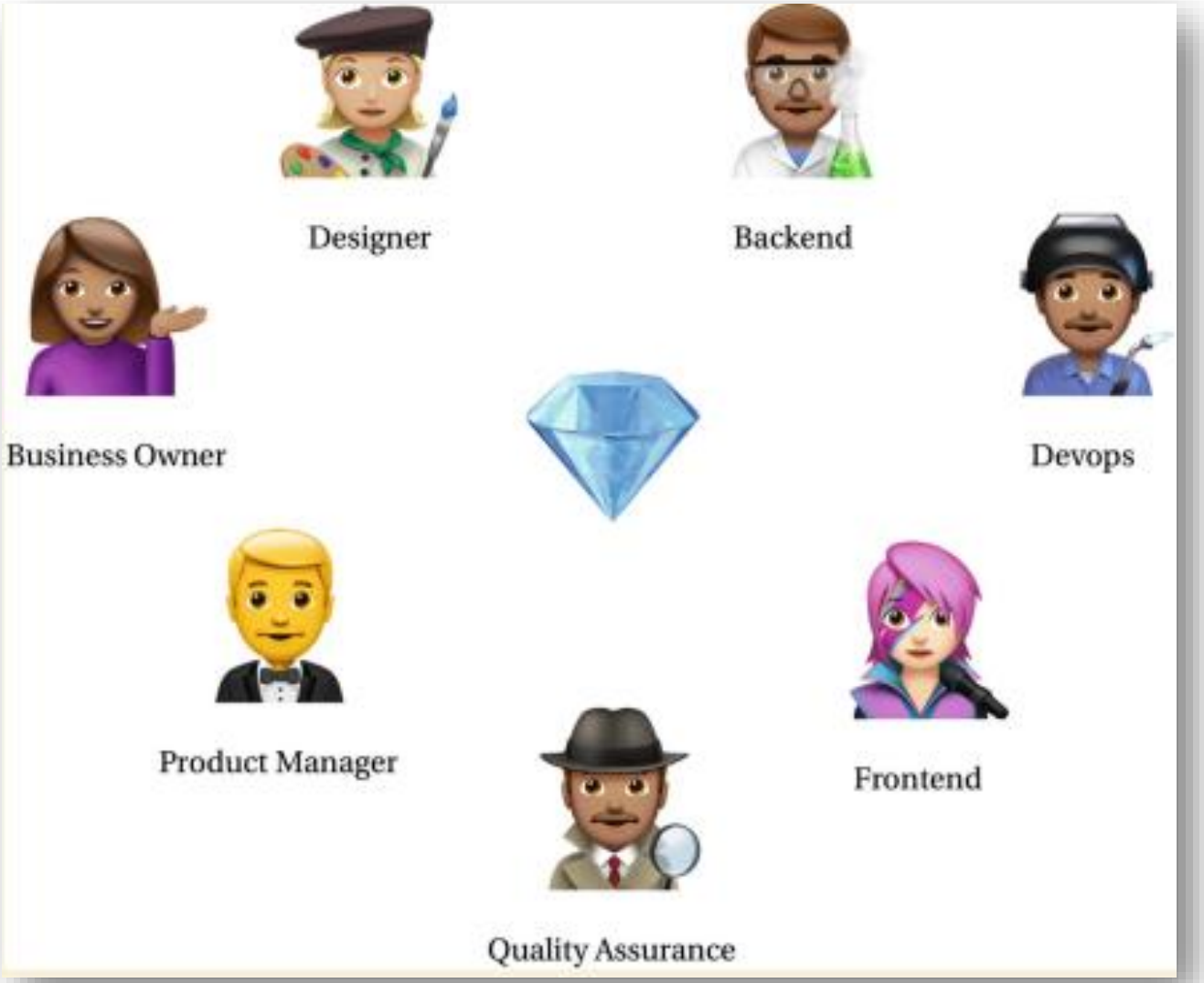
Version Control System (VCS)

Data Structures and Algorithms (310-2101)




REQUIREMENTS
ANALYSIS
DESIGN
CODING
TESTING
DEPLOYMENT
PROJECT PLAN
SOFTWARE DEVELOPMENT LIFECYCLE

Roles and Responsibilities



Designer: Design interaction of User with Product. Make a great User Interface (UI) & User Experience (UX)

Backend: Challenge on software performance.

Frontend: Implement interface & logic of application interaction with User

Quality Assurance (QA): make sure functional working properly. Test on heavy load / make sure software answer request within accept time frame.

Devops (Development + Operations): responsible for all operational aspects of development, make sure current infrastructure can handle expected load.

roadmap.sh

Roadmaps Guides Videos Thanks Subscribe

Frontend
Step by step guide to becoming a frontend developer in 2022

Backend
Step by step guide to becoming a backend developer in 2022

DevOps
Step by step guide for DevOps or operations role in 2022

React
Step by step guide to become a React Developer in 2022

Angular
Step by step guide to become a Angular Developer in 2022

Android
Step by step guide to becoming an Android Developer in 2022

Python
Step by step guide to becoming a Python Developer in 2022

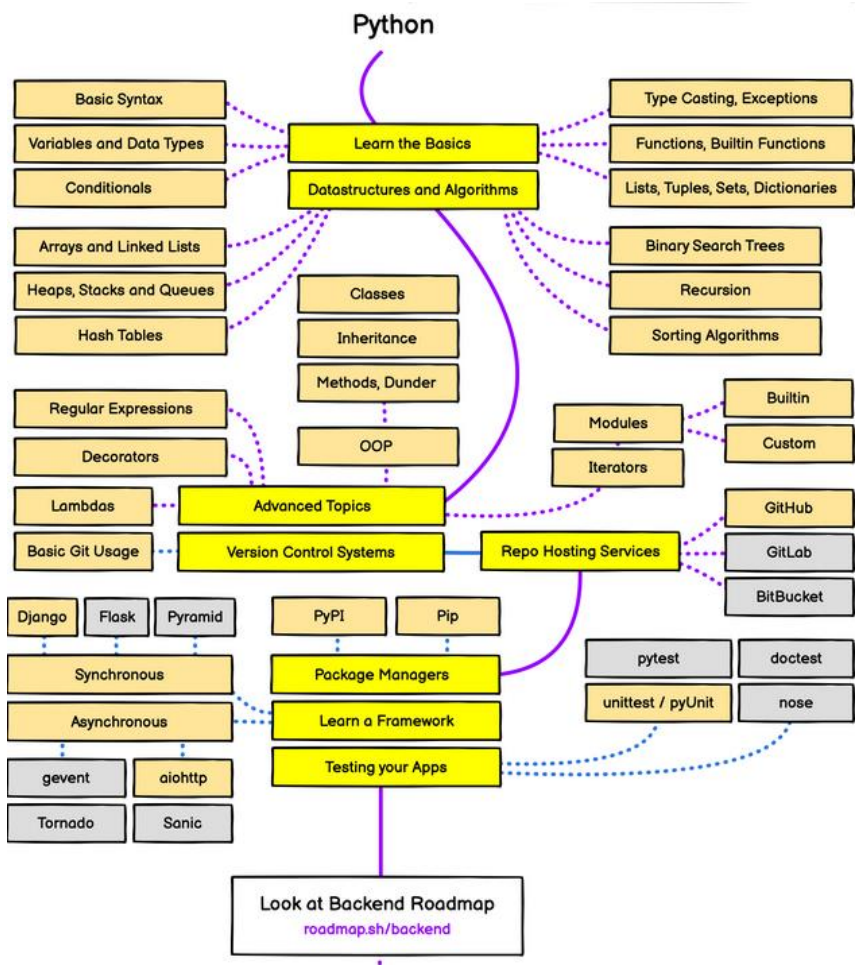
Go
Step by step guide to becoming a Go developer in 2022

Java
Step by step guide to becoming a Java Developer in 2022

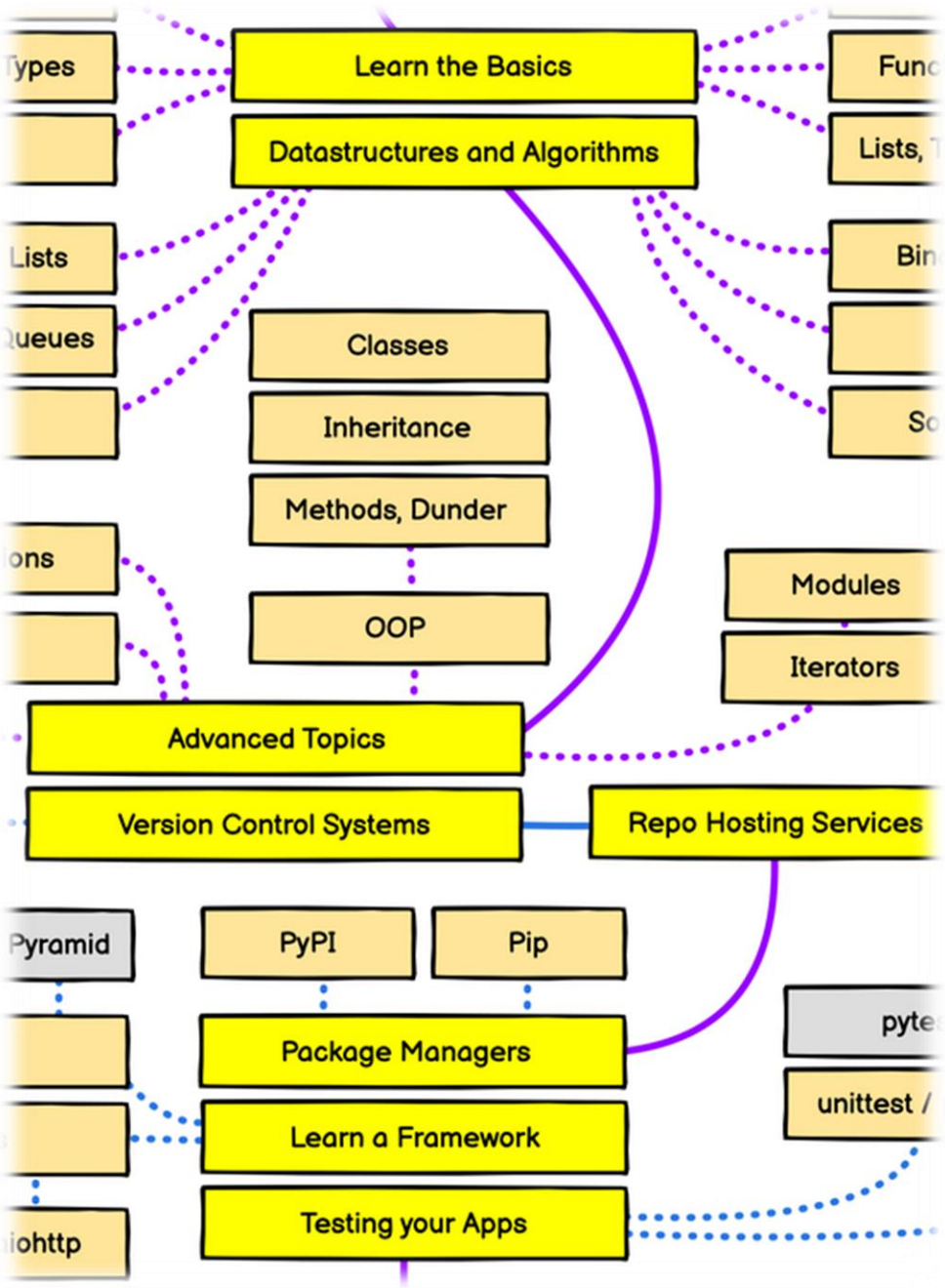
DBA
Step by step guide to become a PostgreSQL DBA in 2022

AWS Upcoming
Step by st n AWS in 2022

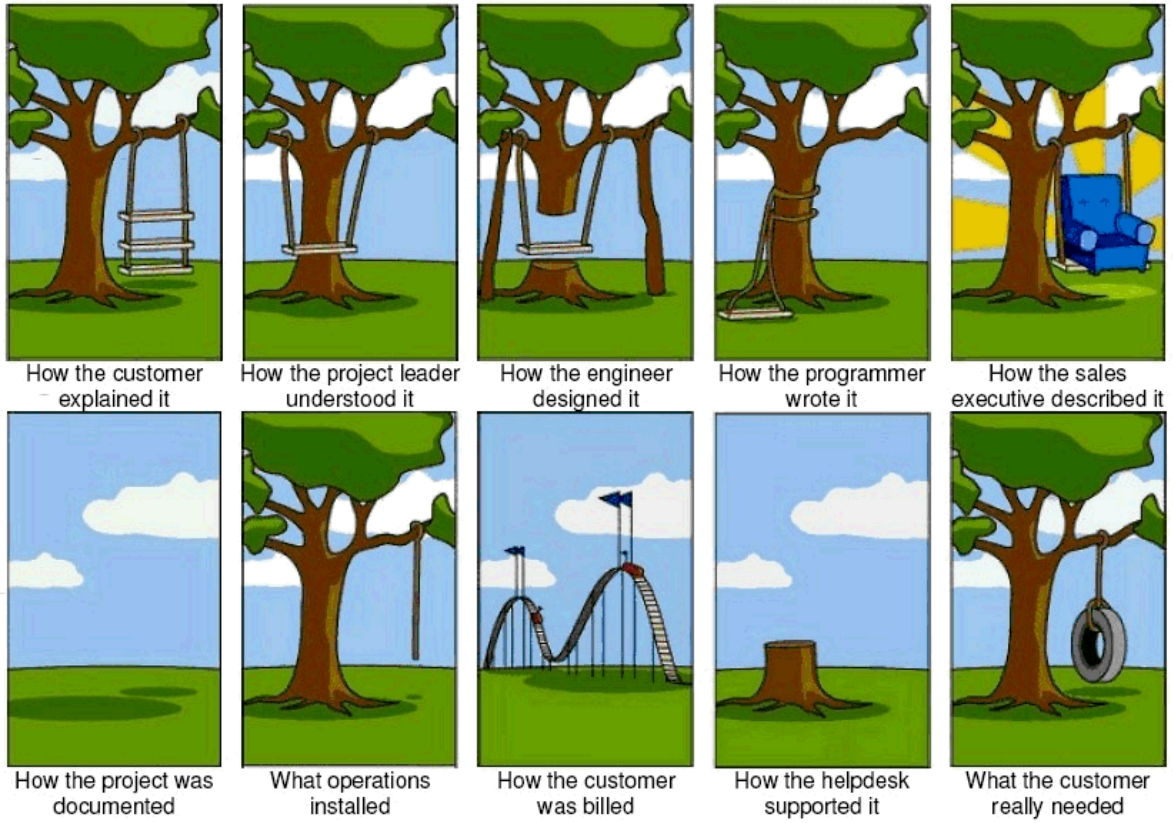
QA Upcoming
Step by st oming a modern QA Engineer in 2022

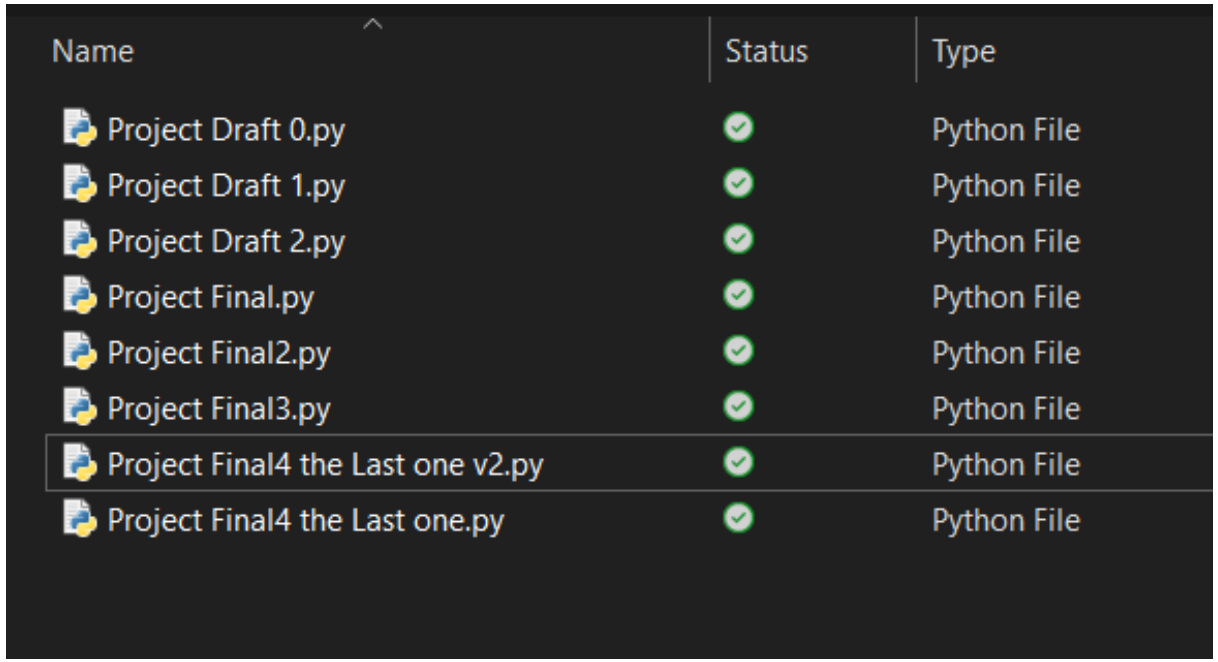


Python
Basic Syntax
Variables and Data Types
Conditionals
Learn the Basics
Datastructures and Algorithms
Type Casting, Exceptions
Functions, Builtin Functions
Lists, Tuples, Sets, Dictionaries
Arrays and Linked Lists
Heaps, Stacks and Queues
Hash Tables
Binary Search Trees
Recursion
Sorting Algorithms
Classes
Inheritance
Methods, Dunder
OOP
Regular Expressions
Decorators
Lambdas
Advanced Topics
Modules
Iterators
Builtin
Custom
Basic Git Usage
Version Control Systems
Repo Hosting Services
GitHub
GitLab
BitBucket
Django
Flask
Pyramid
PyPI
Pip
Synchronous
Asynchronous
Package Managers
Learn a Framework
Testing your Apps
pytest
doctest
unittest / pyUnit
nose
gevent
aiohttp
Tornado
Sanic
Look at Backend Roadmap
roadmap.sh/backend


Learn the Basics
Datastructures and Algorithms
Classes
Inheritance
Methods, Dunder
OOP
Modules
Iterators
Advanced Topics
Version Control Systems
Repo Hosting Services
PyPI
Pip
Package Managers
Learn a Framework
Testing your Apps

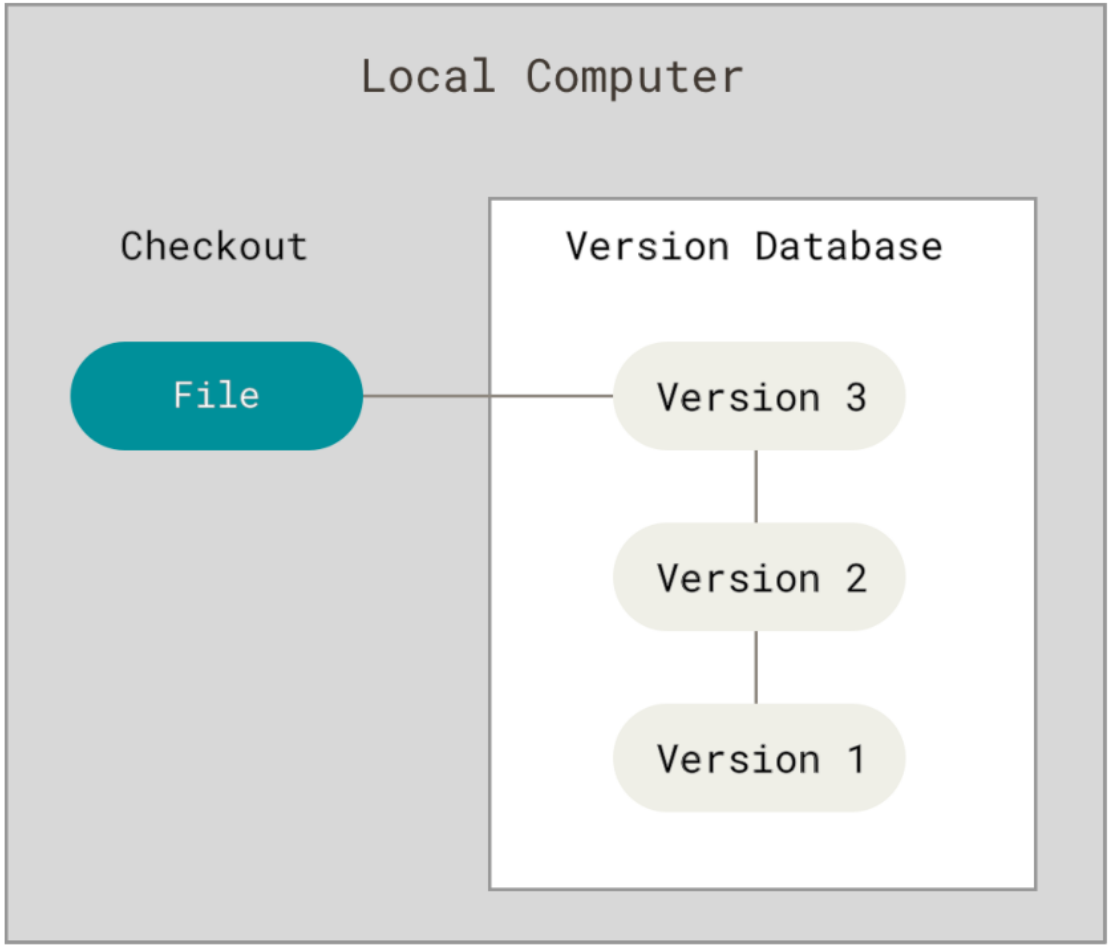
GitHub & Software Version Control

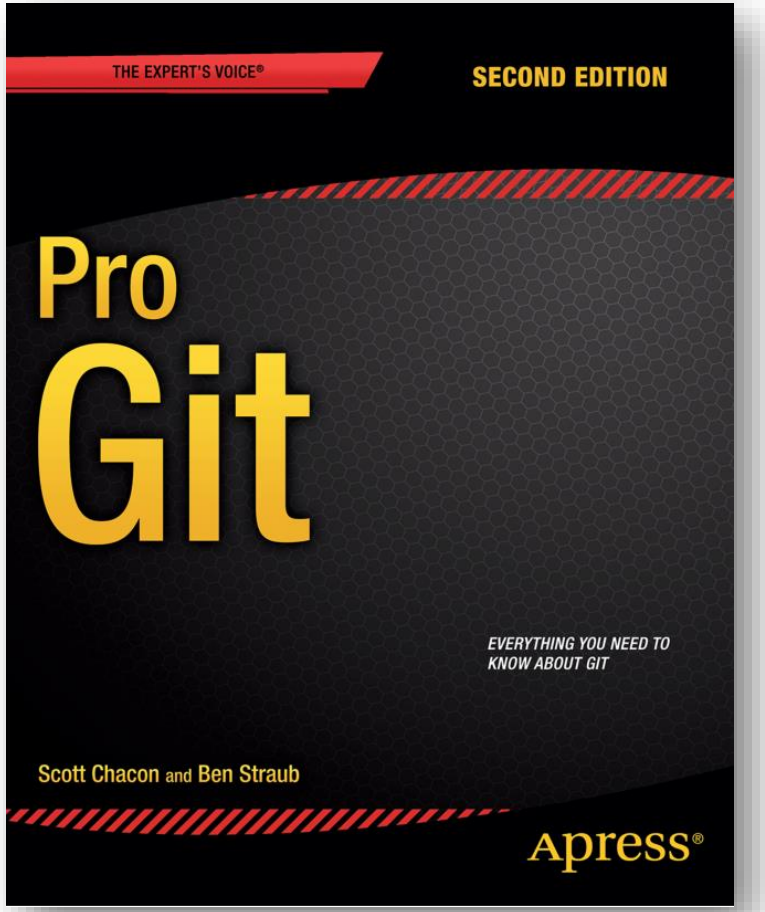




Software Version #1 #2 #3 #4

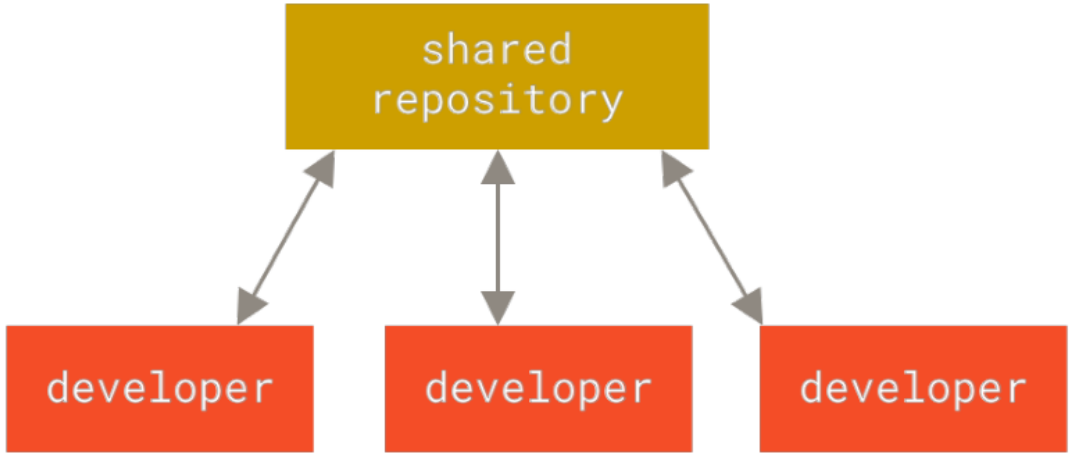
GitHub & Software Version Control



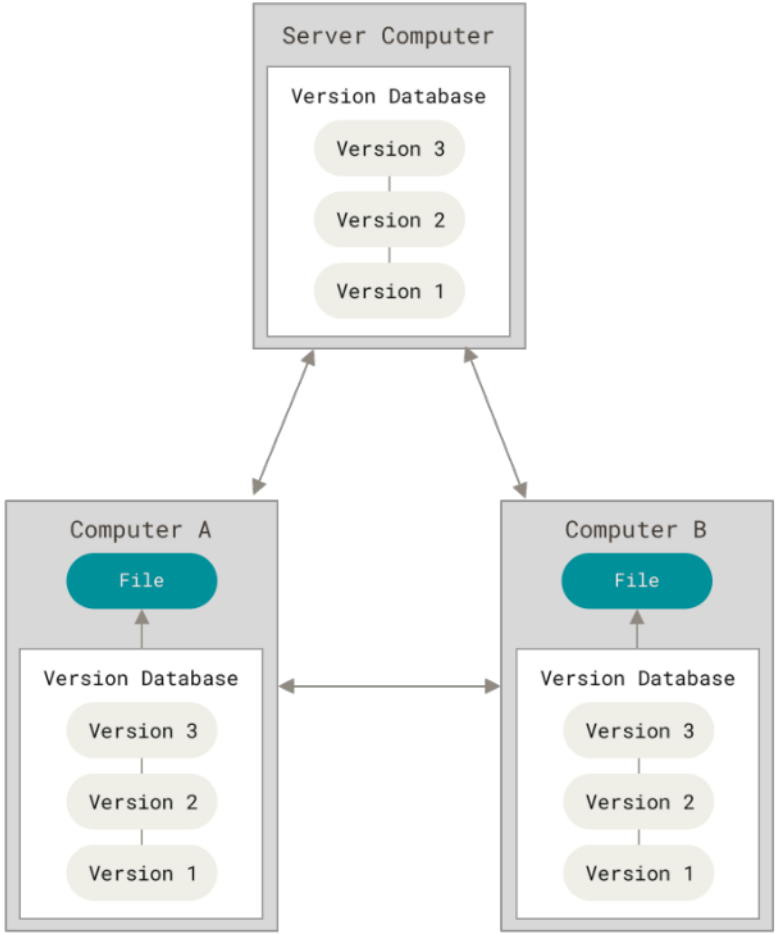


http://git-scm.com/book/en/v2

GitHub & Software Version Control



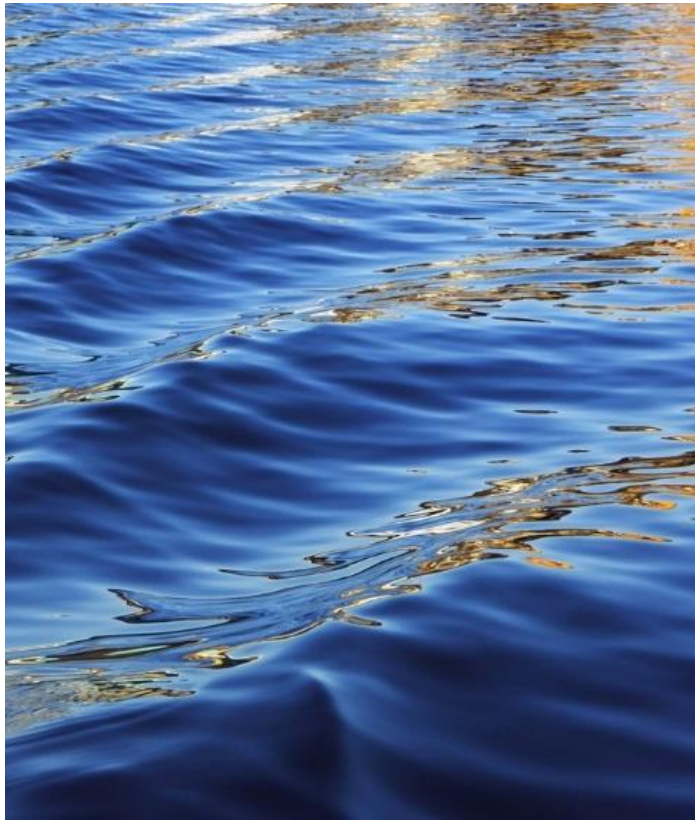
Centralized Version Control Systems



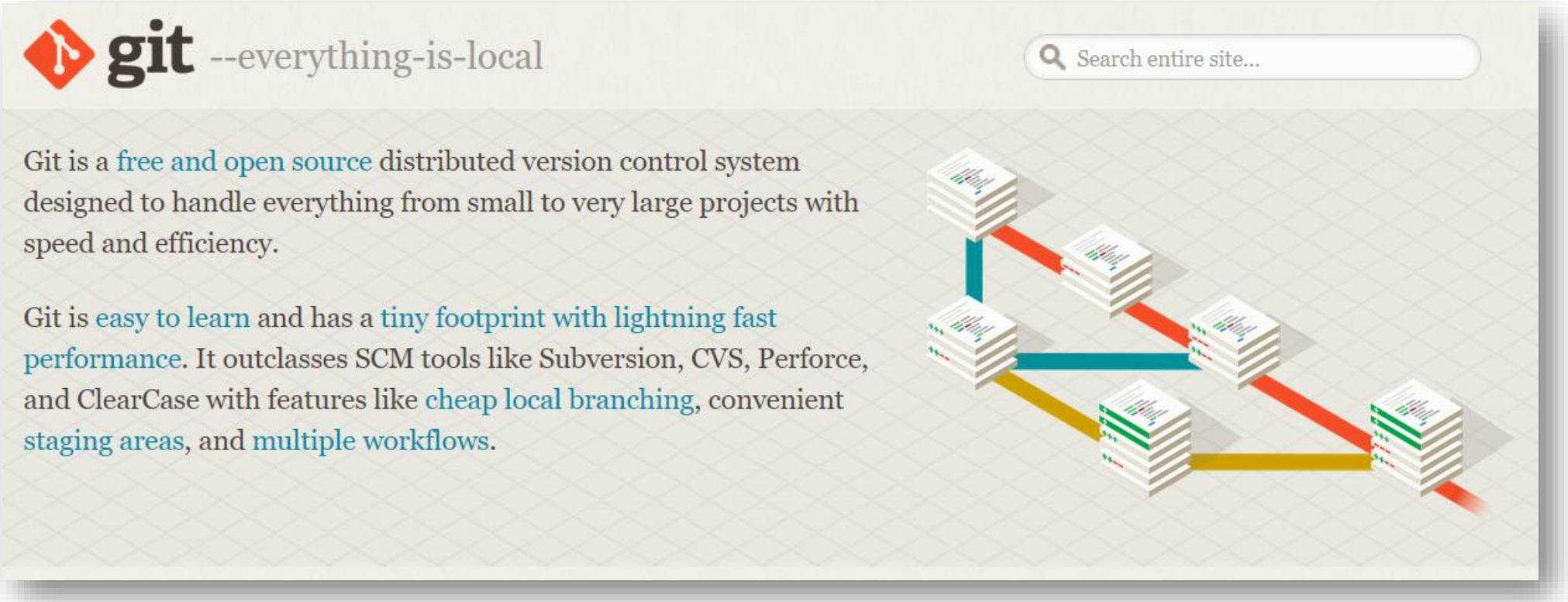
Distributed Version Control Systems

Now, we are going to install **Git** to our PC

>>>>

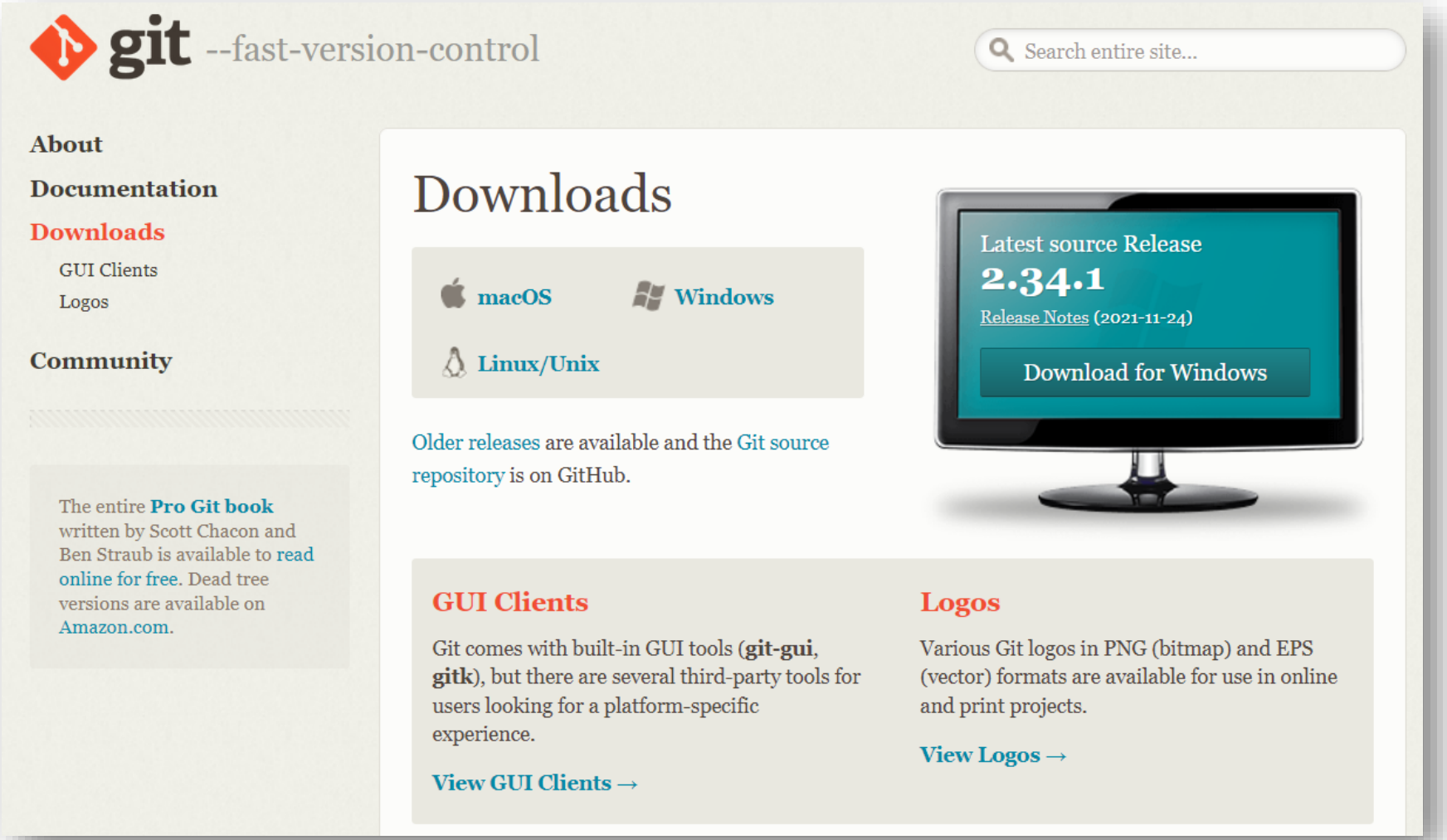


GitHub & Software Version Control

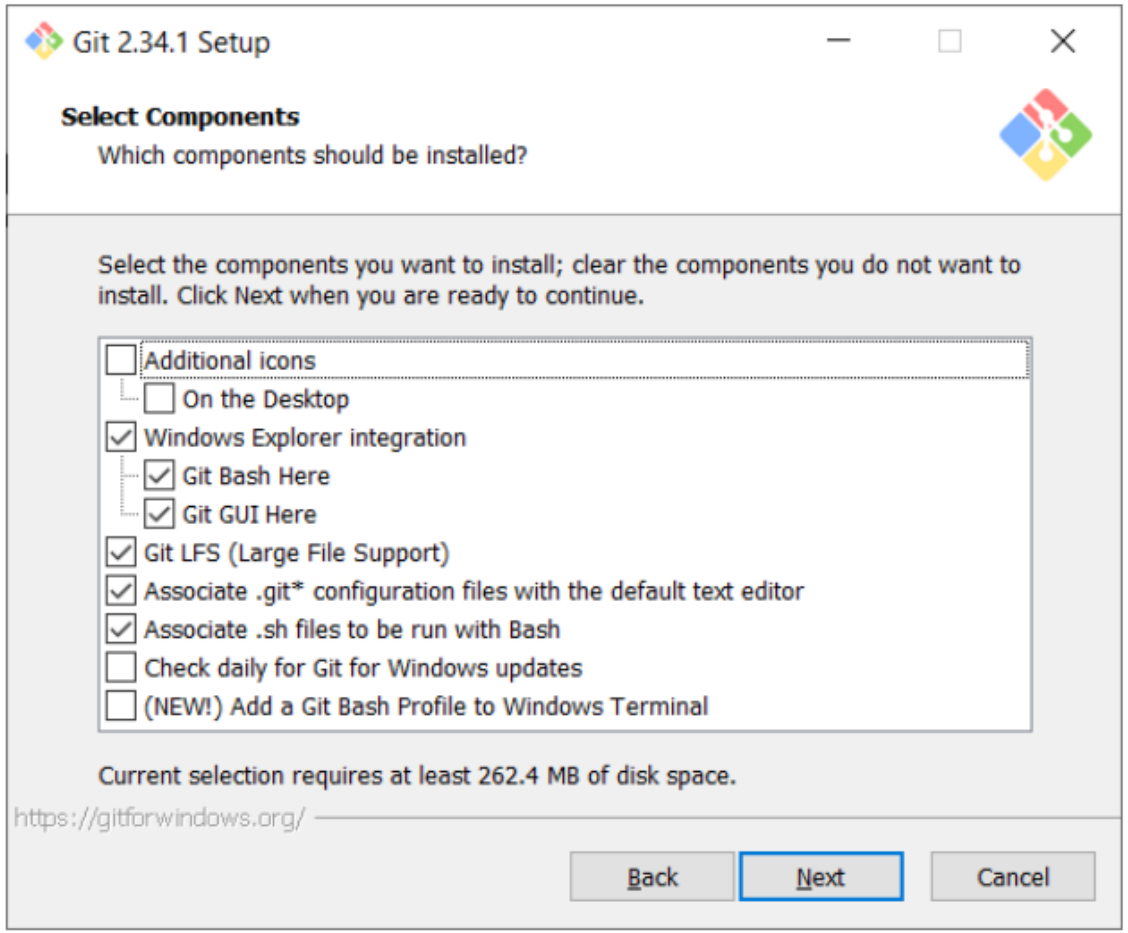


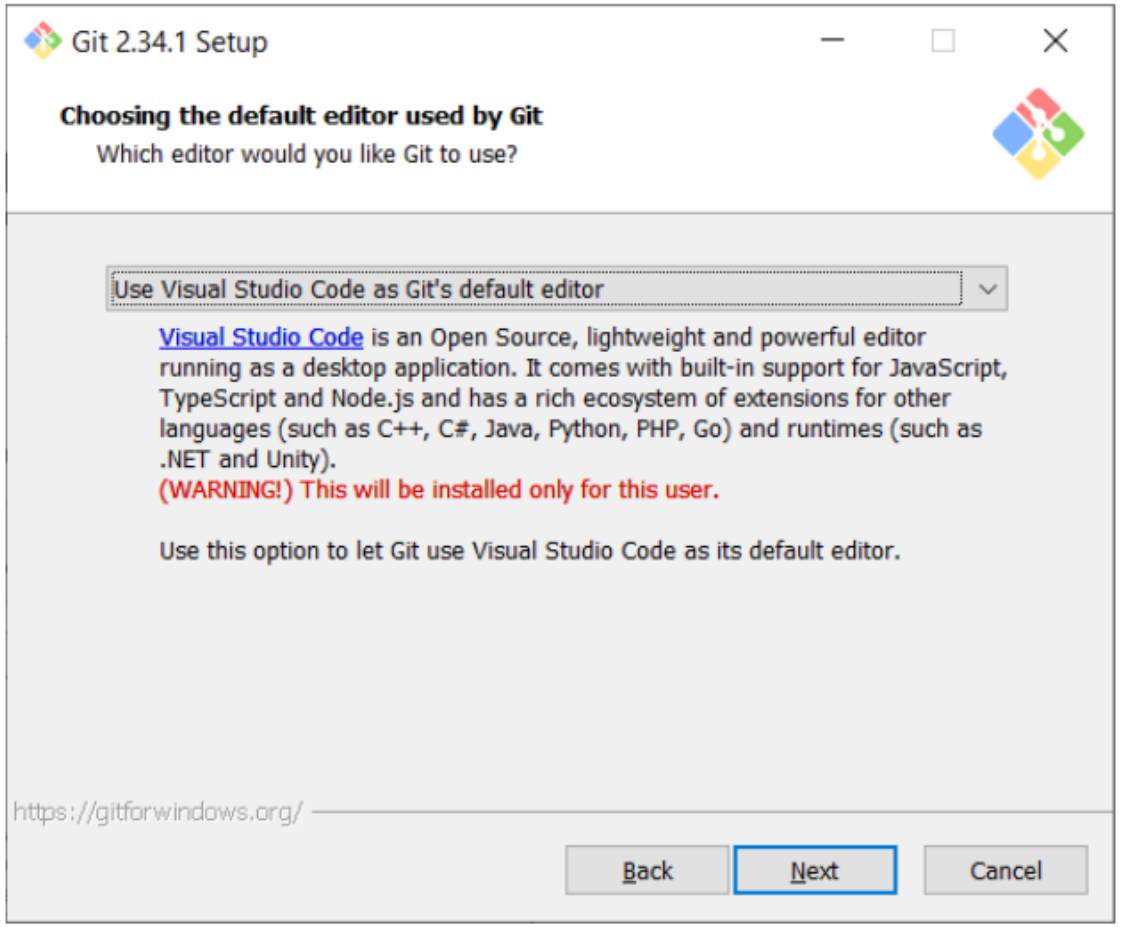
https://git-scm.com/

GitHub & Software Version Control

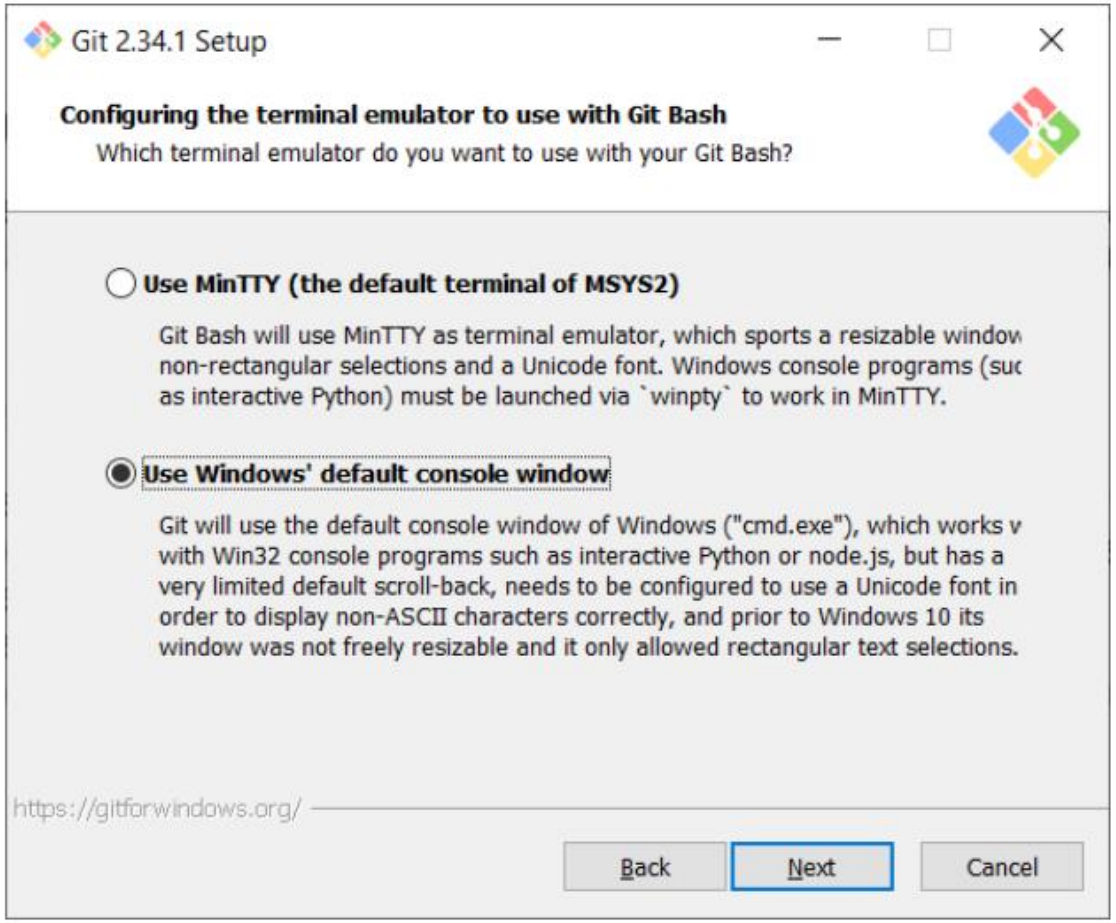


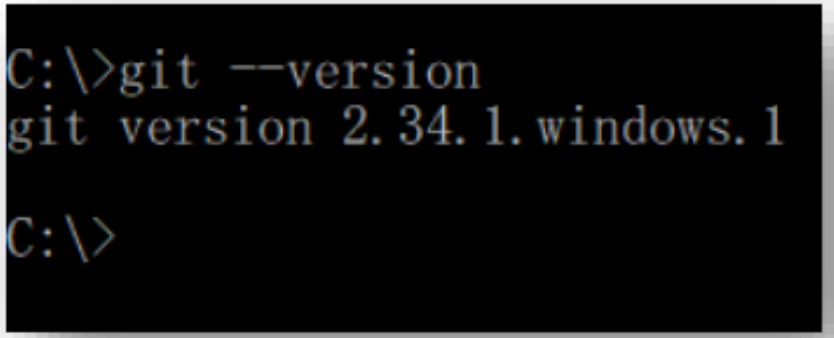
Git & Software Version Control





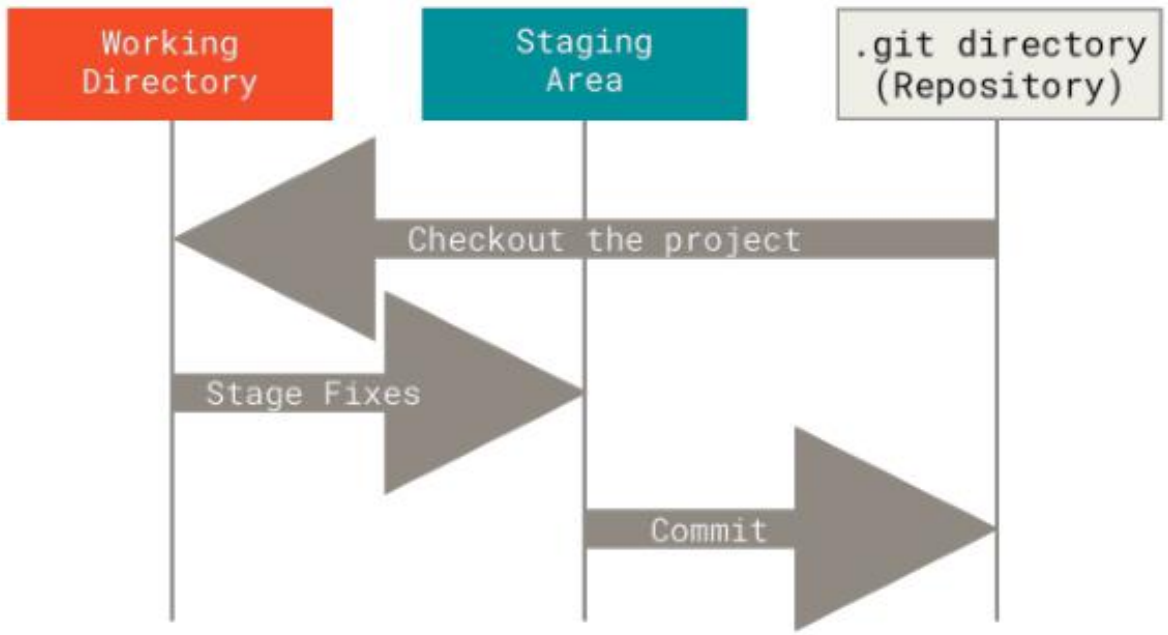
Git & Software Version Control





Git & Software Version Control

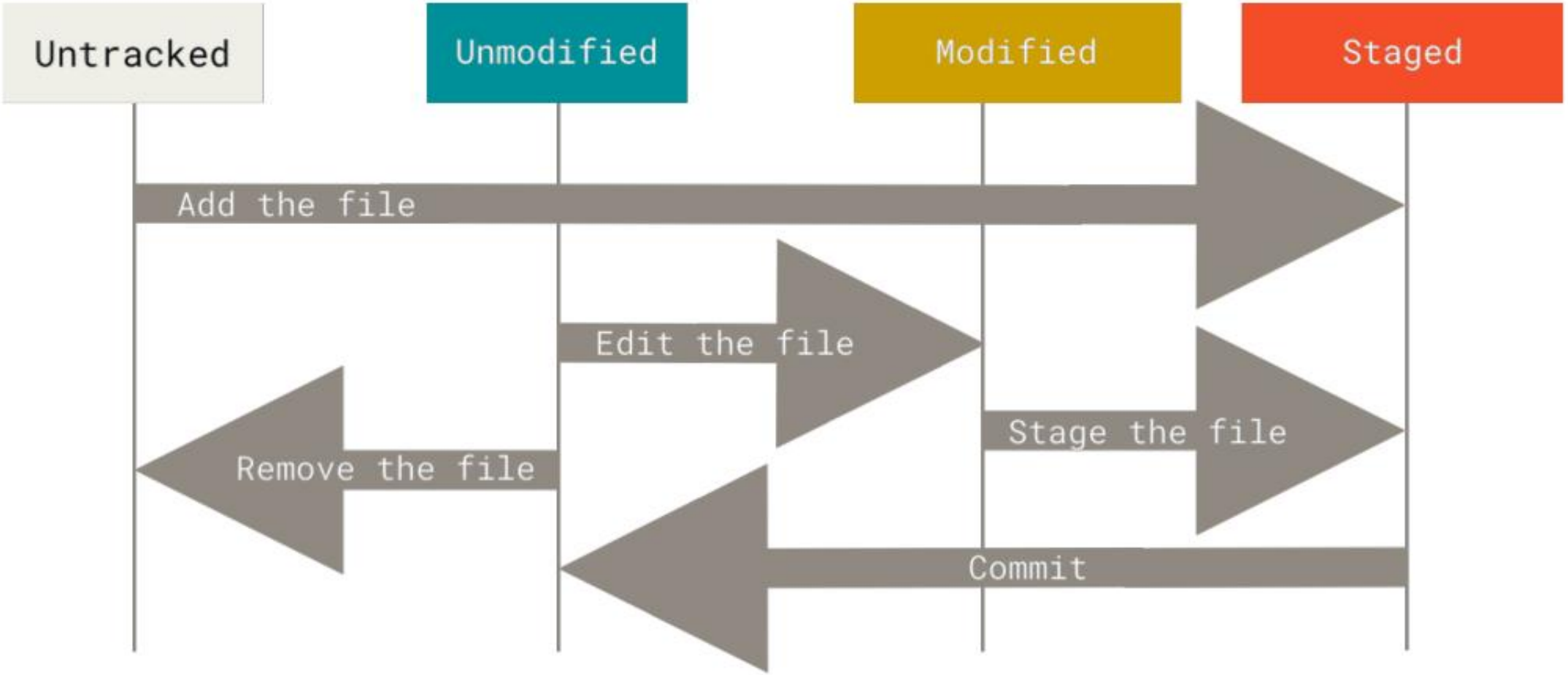
The 3 Stages of Git



The basic Git workflow

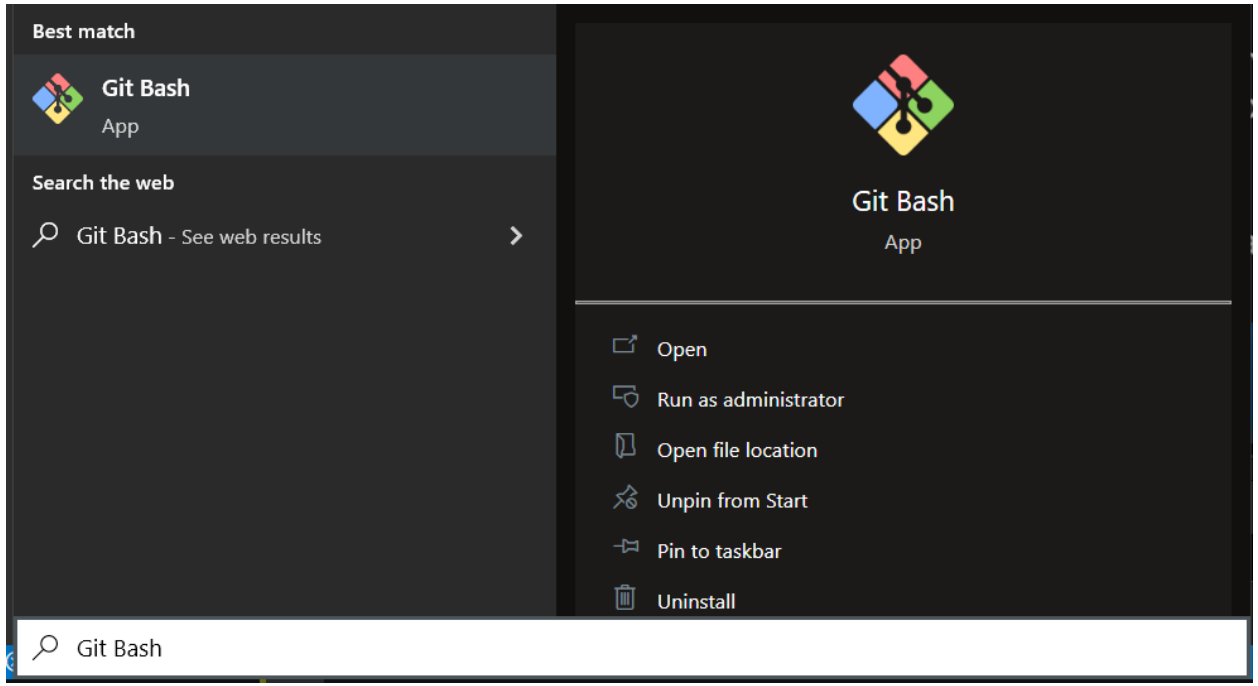
1) You modify files in your working directory
2) You selectively stage just those file(s) you want to commit
3) You do a commit, which takes the file to stores in Git directory

Git & Software Version Control

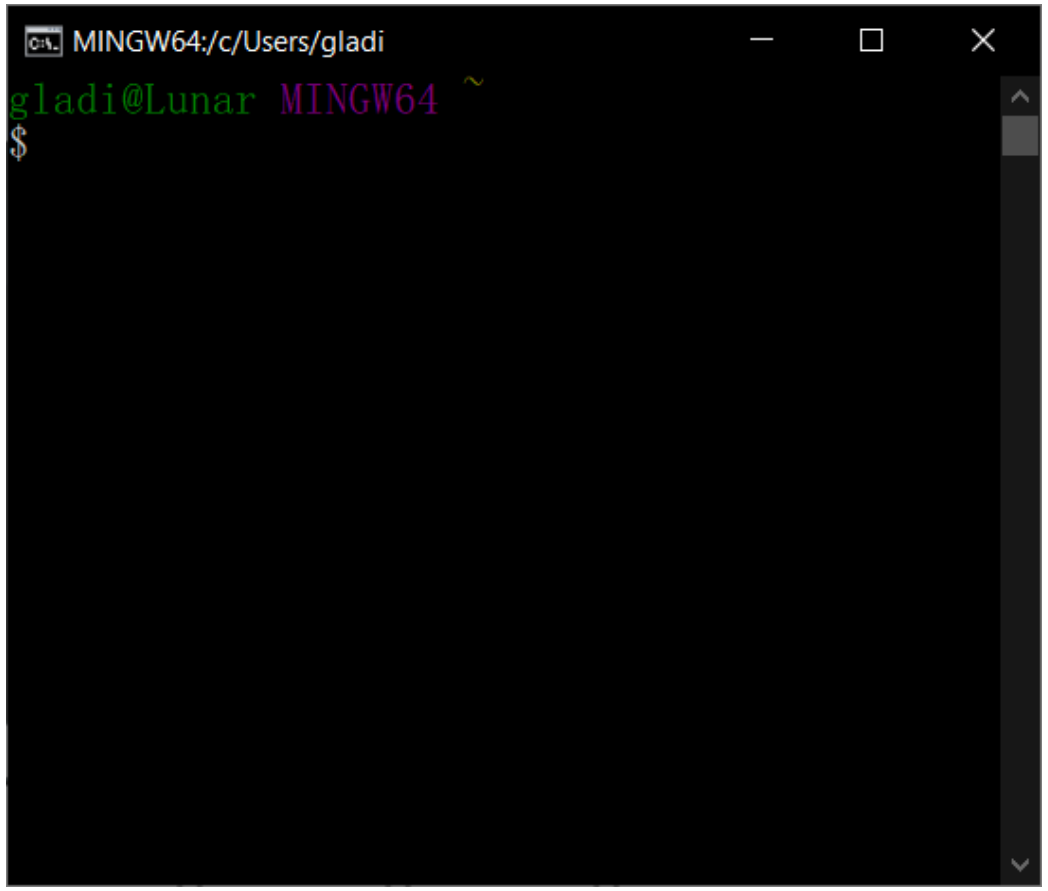


Lifecycle of the status of files

Git Command



[1] Open Git Bash



Git Bash window

Git Command

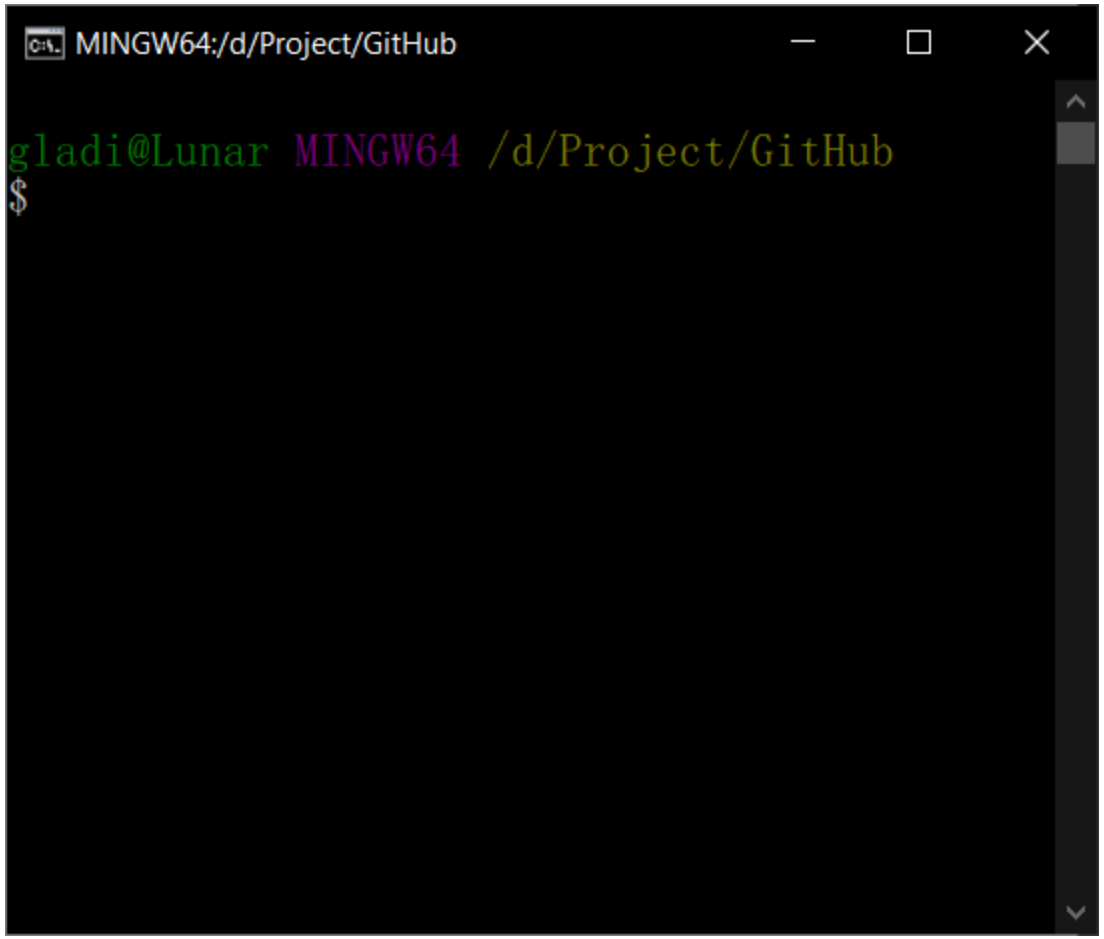
Note: for me, D:\Project\GitHub\

Linux Quick Reference

SOME USEFUL COMMANDS

Command	Task
File/Directory Basics	
ls	List files
cp	Copy files
mv	Rename files
rm	Delete files
ln	Link files
cd	Change directory
pwd	Print current directory name
mkdir	Create directory
rmdir	Delete directory
File Viewing	

Command	Task
File Location	
find	Locate files
slocate	Locate files via index
which	Locate commands
whereis	Locate standard files
File Text Manipulation	
grep	Search text for matching lines
cut	Extract columns
paste	Append columns
tr	Translate characters
sort	Sort lines



- ❑ Use “ls” command to list file/folder
- ❑ Use “cd” to change directory

Git Command

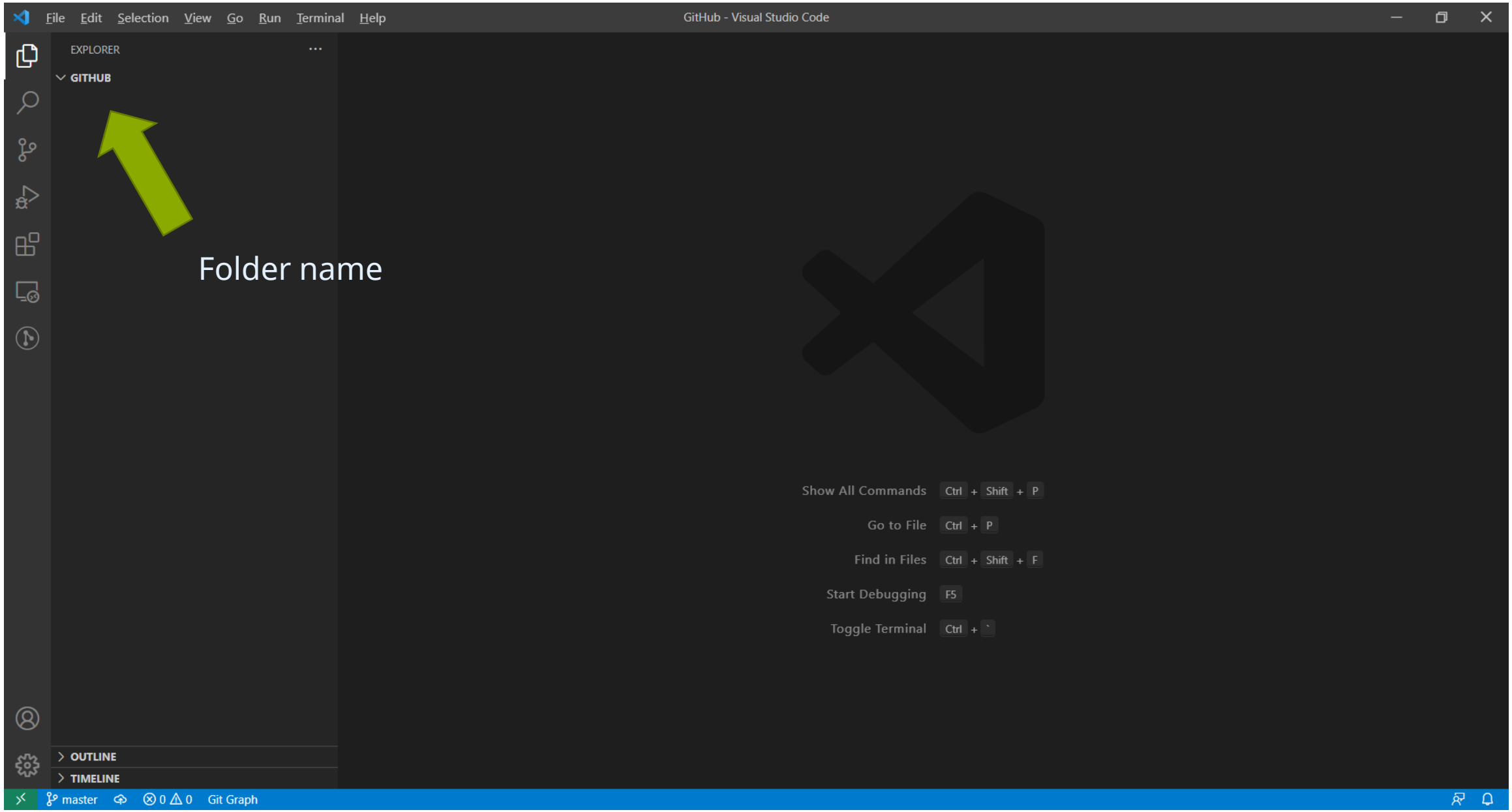
```
MINGW64:/d/Project/GitHub

gladi@Lunar MINGW64 /d/Project/GitHub
$ git init
Initialized empty Git repository in D:/Project/GitHub/.git/

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$
```

Initial Git by “git init” command

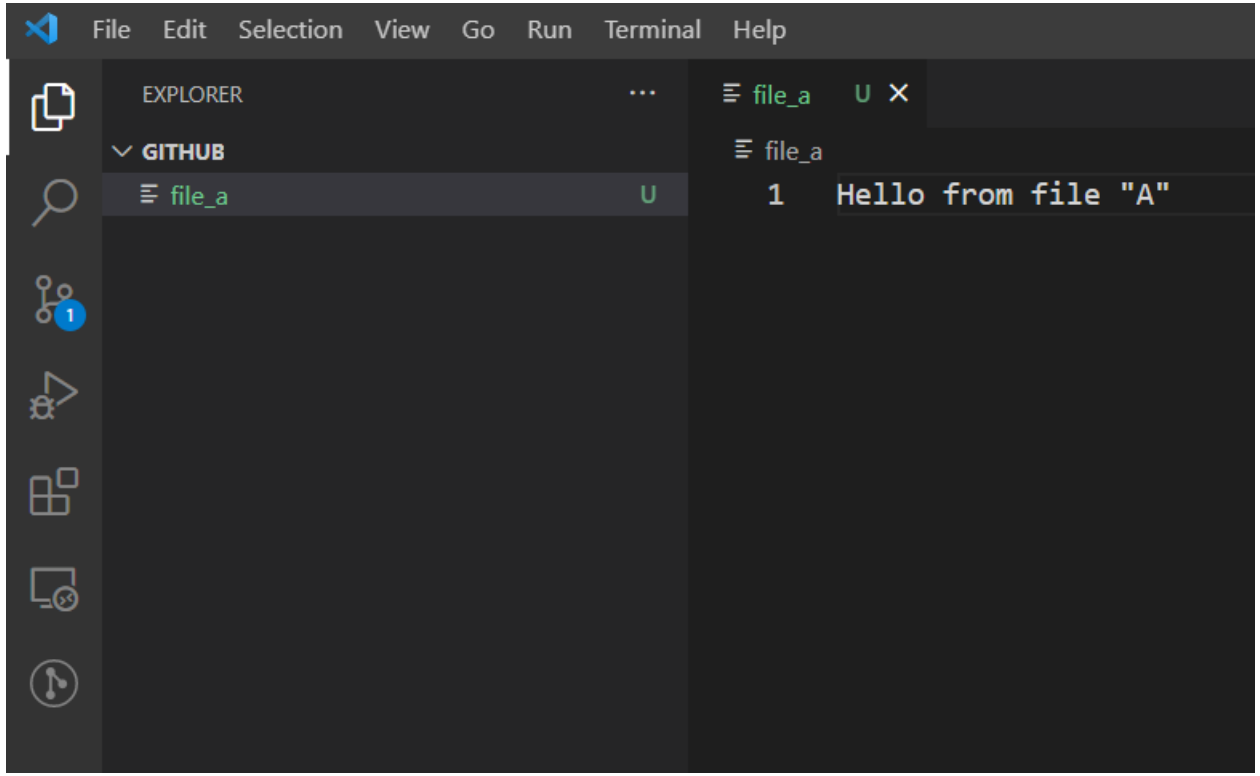
```
MINGW64:/d/Project/GitHub

gladi@Lunar MINGW64 /d/Project/GitHub
$ git init
Initialized empty Git repository in D:/Project/Gi

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ code.
bash: code.: command not found

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ code .
```

Use “code .” command

To open VS Code or do it by typical step

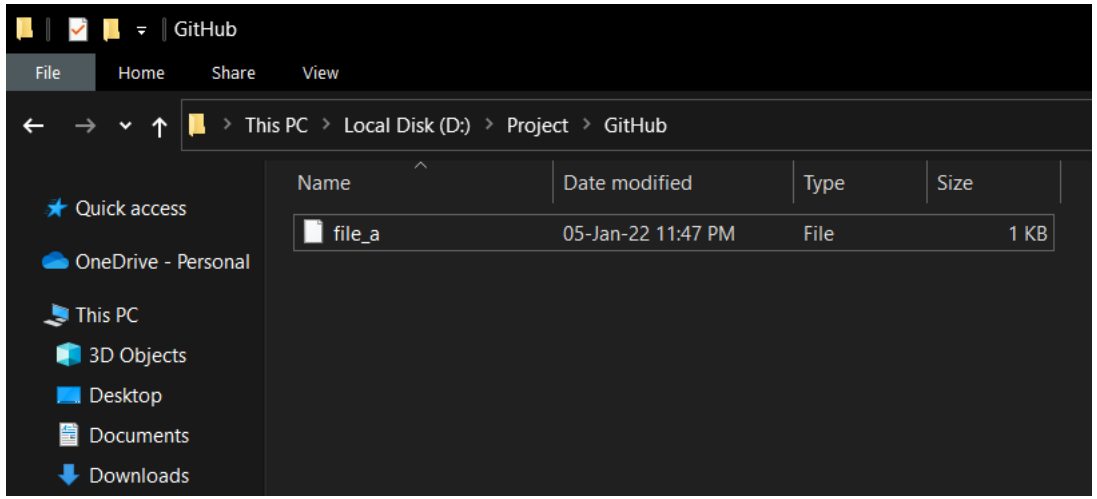


Initial Git by “git init” command



Create “file_a”

(type something you like on this file)



Now you will see “file_a” in directory

(There will be .git folder hidden in this folder)

```
MINGW64:/d/Project/GitHub

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git status
On branch master

No commits yet

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

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

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$
```

New file(s) to commit

Type “git status” to show current status of Git

```
MINGW64:/d/Project/GitHub

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git add file_a

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git status
On branch master

No commits yet

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


gladi@Lunar MINGW64 /d/Project/GitHub (master)
$
```

New file(s) prompt to commit

Type “git add”with untrack file (“file_a”)
and re-check the Git status

```
MINGW64:/d/Project/GitHub
On branch master

No commits yet

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


gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git commit
Author identity unknown

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'gladi@Lunar.(none)')

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$
```

Type "git commit"

(the 1st time of commit, you need to config a detail of **who** commit)

```
MINGW64:/d/Project/GitHub

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git commit
Author identity unknown

*** Please tell me who you are.

Run

  git config --global user.email "you@example.com"
  git config --global user.name "Your Name"

to set your account's default identity.
Omit --global to set the identity only in this repository.

fatal: unable to auto-detect email address (got 'gladi@Lunar.(none)')

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git config --global user.email [redacted]

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git config --global user.name [redacted]

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git commit
hint: Waiting for your editor to close the file...
```

(the 1st time of commit, you need to config a detail of **who** commit)

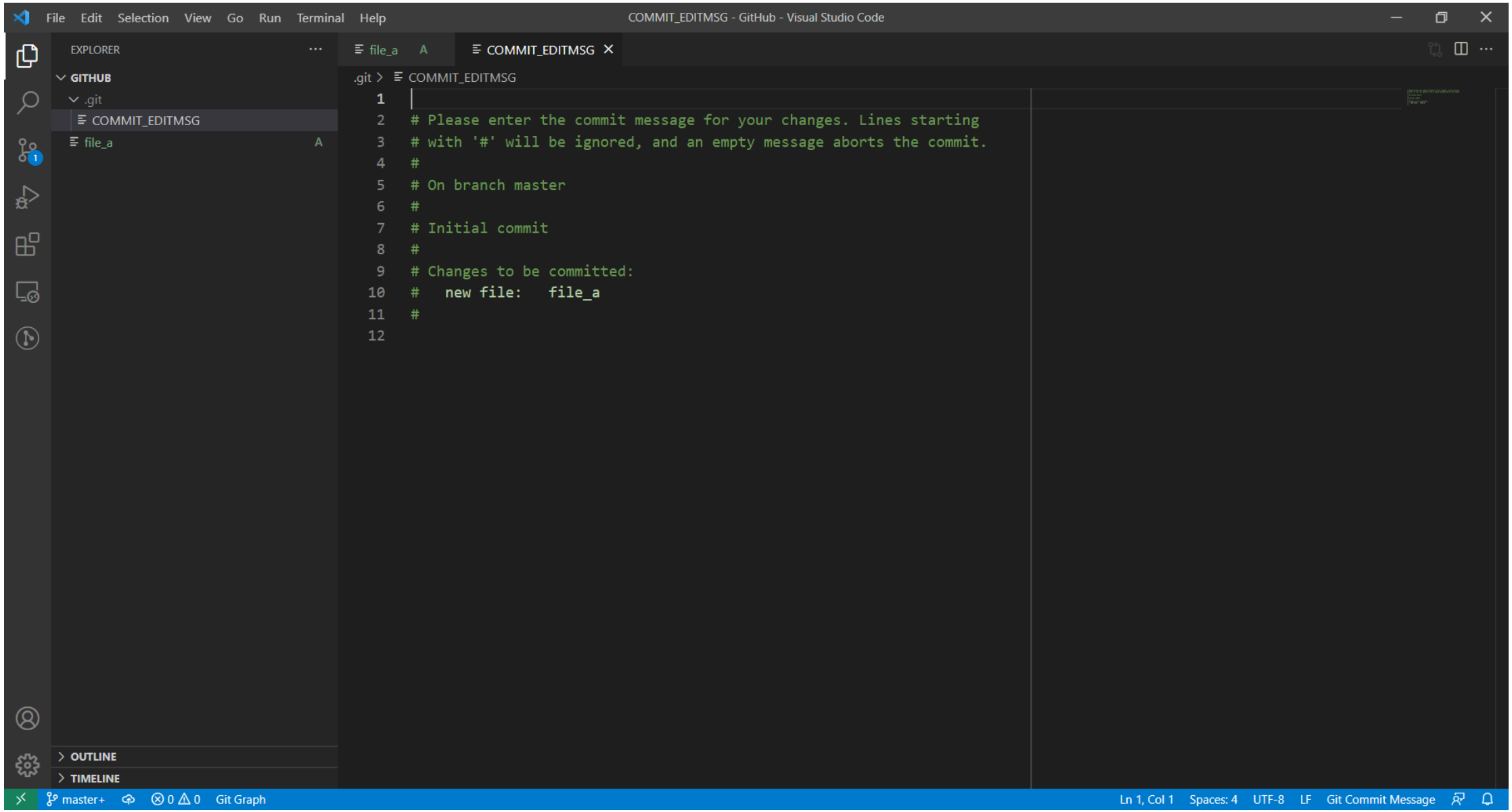
Insert your email & username, then re-commit

How can I edit the .git / config file from the git terminal?

```
git config --global --edit
```

This should open a text editor, make your changes ,save and exit the editor.

You can also edit your e-mail, name in each project instead of global.

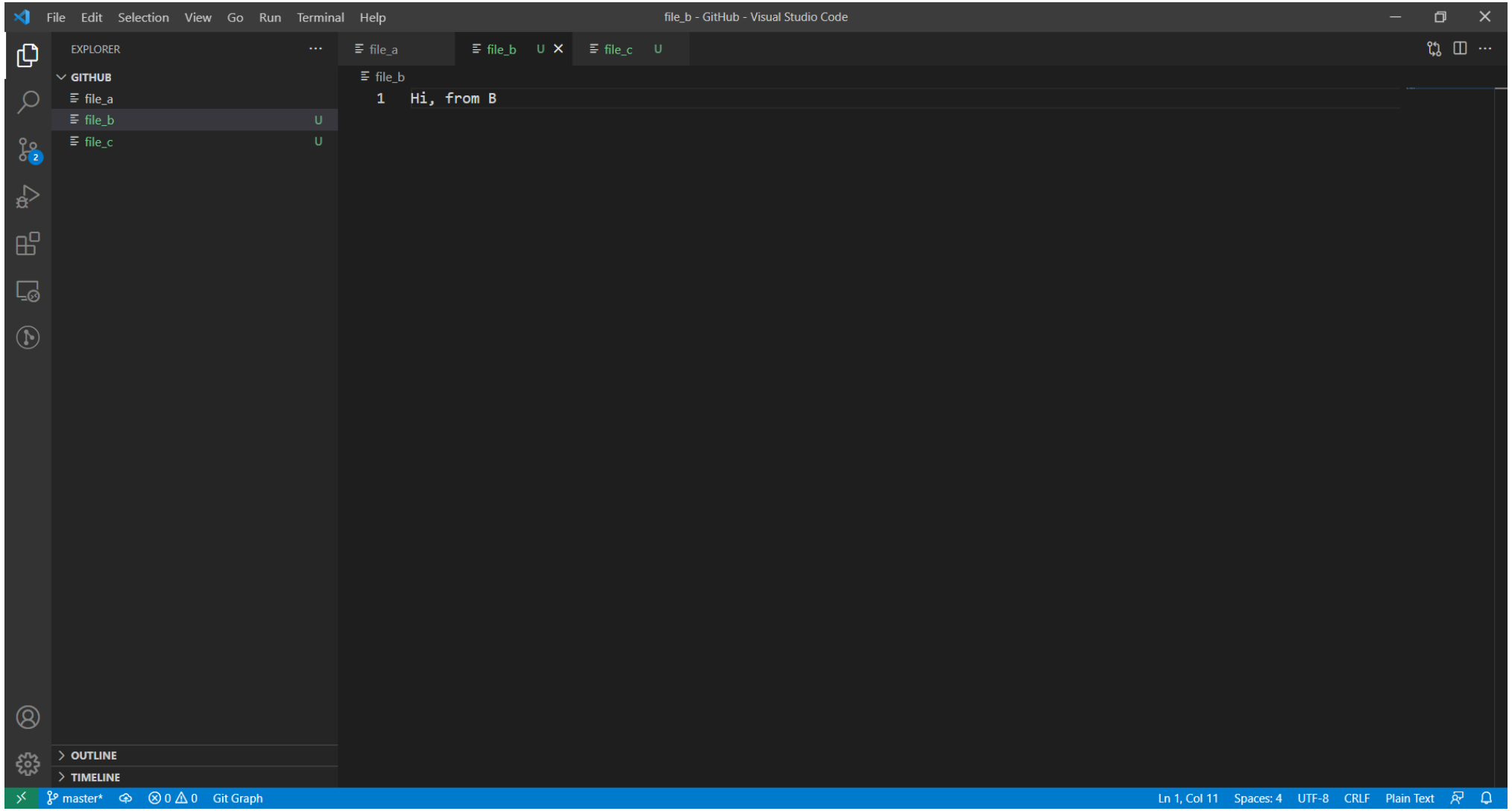


Git will open [TextEditor] to add message for Commit

MINGW64:/d/Project/GitHub

```
gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git commit -m "add file a"
[master (root-commit) 9cd3516] add file a
 1 file changed, 1 insertion(+)
 create mode 100644 file_a

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$
```

Type “git commit –m “add file a” “

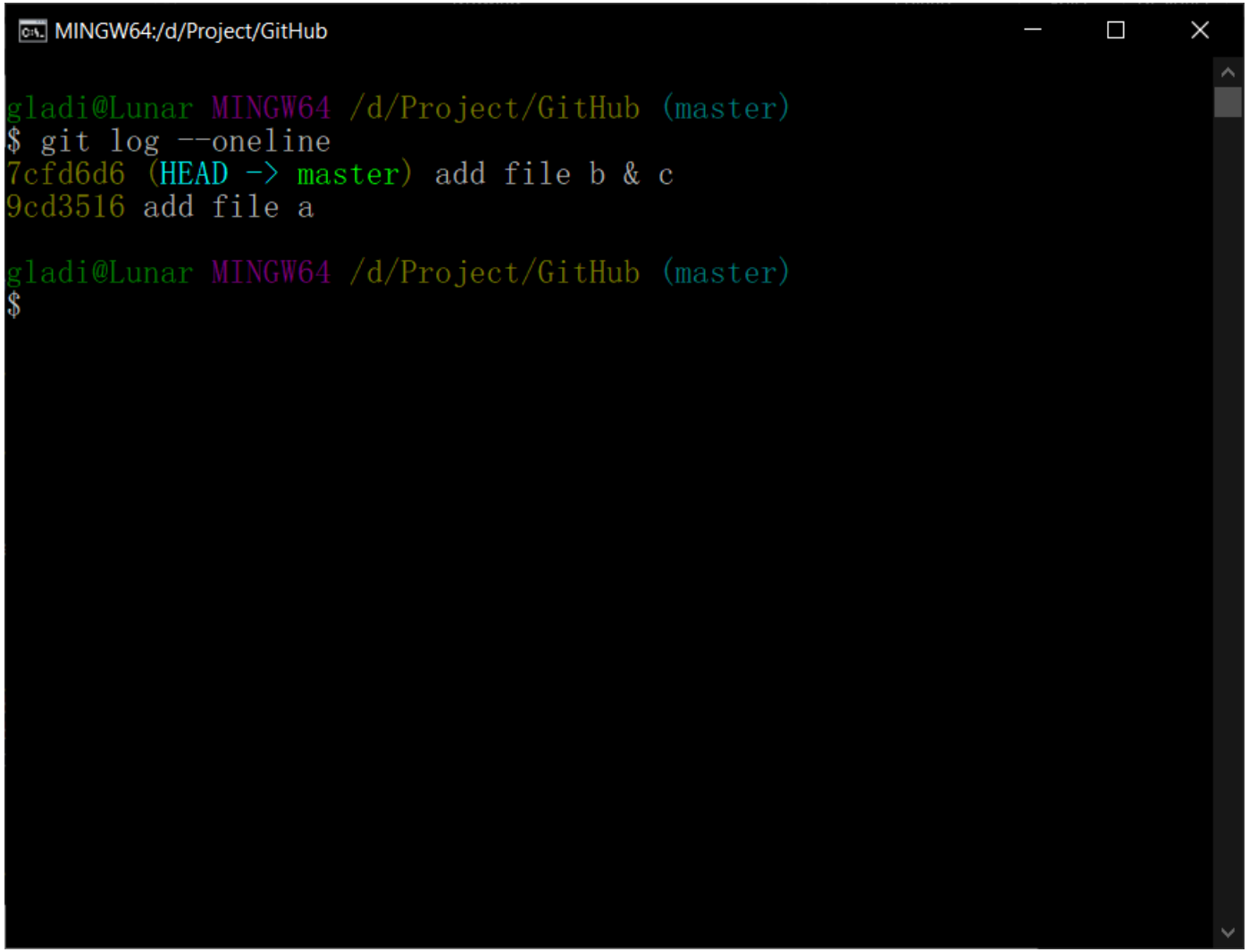


OK, now we going to add file_b, file_c with the same manner

(If you add the wrong file type “git restore --staged <file>”)

```
MINGW64:/d/Project/GitHub
        file_b
        file_c

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

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git add file_*

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   file_b
        new file:   file_c


gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git commit -m "add file b & c"
[master 7cfd6d6] add file b & c
 2 files changed, 2 insertions(+)
 create mode 100644 file_b
 create mode 100644 file_c

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$
```

Then, commit its like this.



We can check the log of git by type “git log” or “git log --oneline”

Next

We are going to do something on file_b

Then

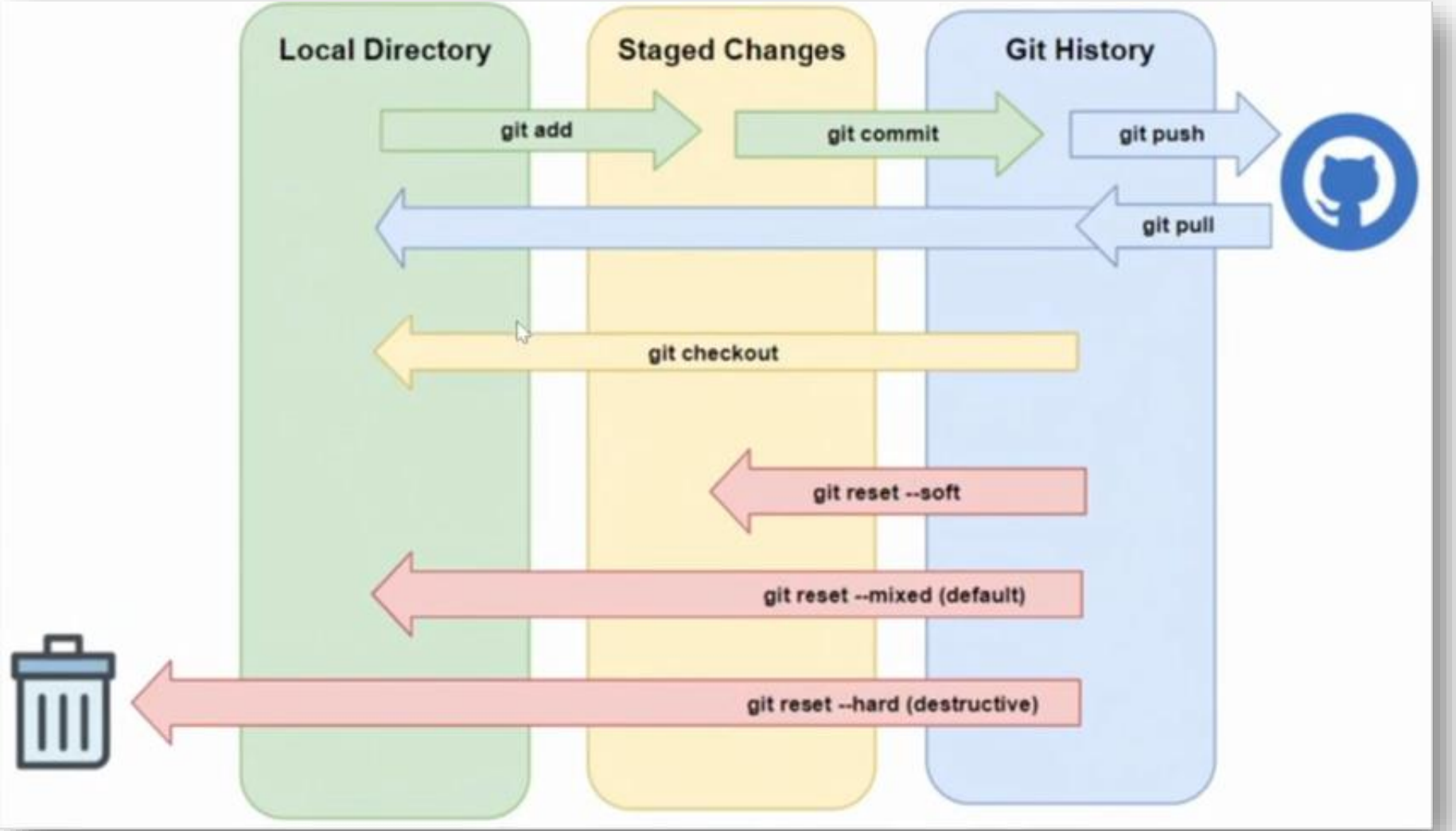
Roll back it to the stage we commit

Do this in Class

Edit & commit file_b

```
MINGW64:/d/Project/GitHub

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git log --oneline
e7bab84 (HEAD -> master) edit file b
7cfd6d6 add file b & c
9cd3516 add file a

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$
```

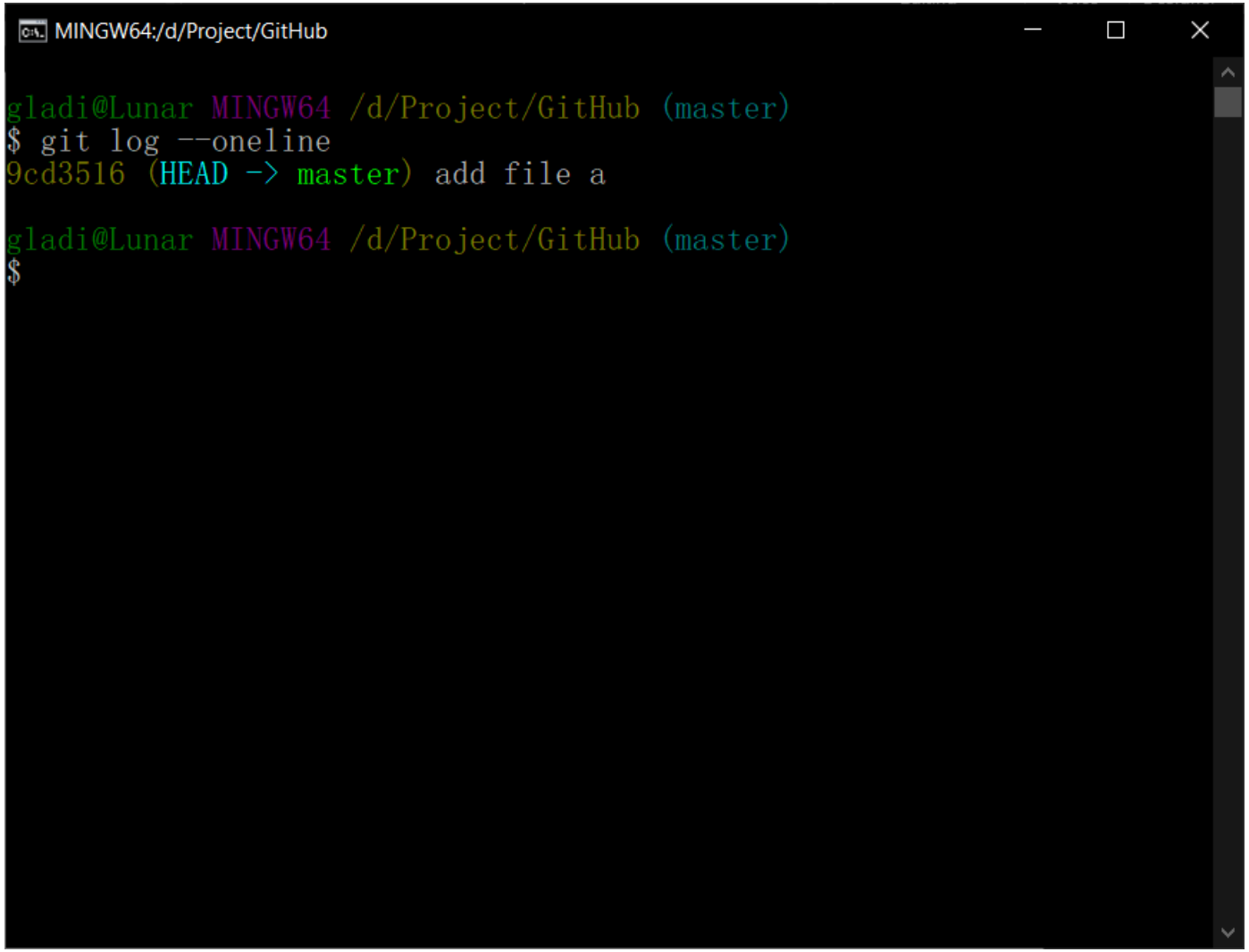
We should see 3 stage of commit like this



We can reset stage 3-ways

```
MINGW64:/d/Project/GitHub

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git log --oneline
e7bab84 (HEAD -> master) edit file b
7cfd6d6 add file b & c
9cd3516 add file a

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git reset 9cd3516
```

Let’s go back to the first stage, type “git reset [commit number]”



Re-check, we will see that <Head> back to 1st stage

```
MINGW64:/d/Project/GitHub

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git log --oneline
9cd3516 (HEAD -> master) add file a

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git reflog --oneline
9cd3516 (HEAD -> master) HEAD@{0}: reset: moving to 9cd3516
e7bab84 HEAD@{1}: commit: edit file b
7cfd6d6 HEAD@{2}: commit: add file b & c
9cd3516 (HEAD -> master) HEAD@{3}: commit (initial): add file a

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$
```

Next, we are going to move forward to 3rd stage

Type "git reflog --oneline" to see [commit number] ➔ or use HEAD@{1} instead

```
MINGW64:/d/Project/GitHub

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git log --oneline
e7bab84 (HEAD -> master) edit file b
7cfd6d6 add file b & c
9cd3516 add file a

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$
```

Jump back with the same command

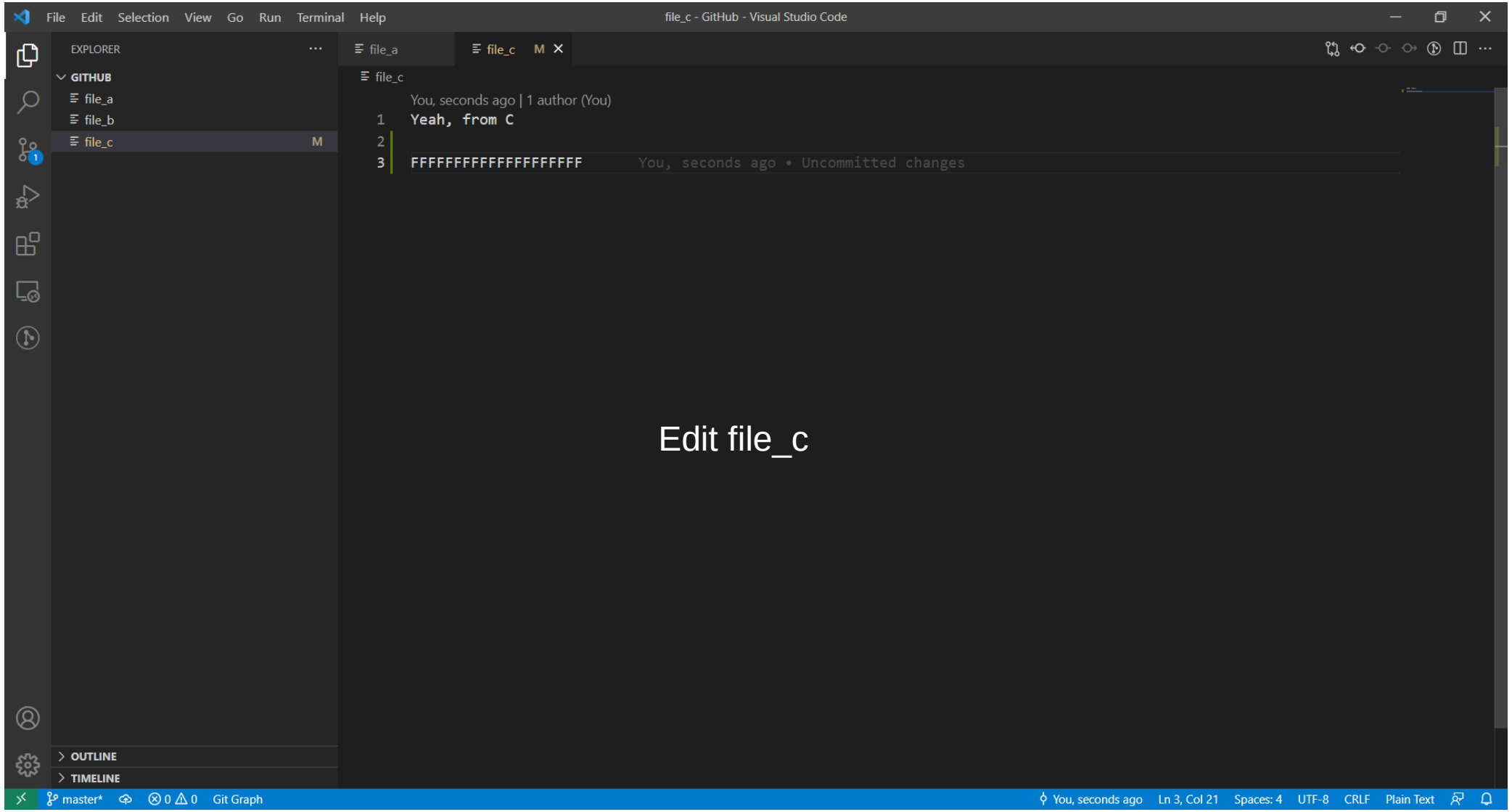
(re-check with git log)

```
MINGW64:/d/Project/GitHub

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory
)
        deleted:    file_b

no changes added to commit (use "git add" and/or "git commit -a")

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git restore file_b

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ ■
```

Try to Remove file_b, then type “git status” to see what happened

Restore file with Git command

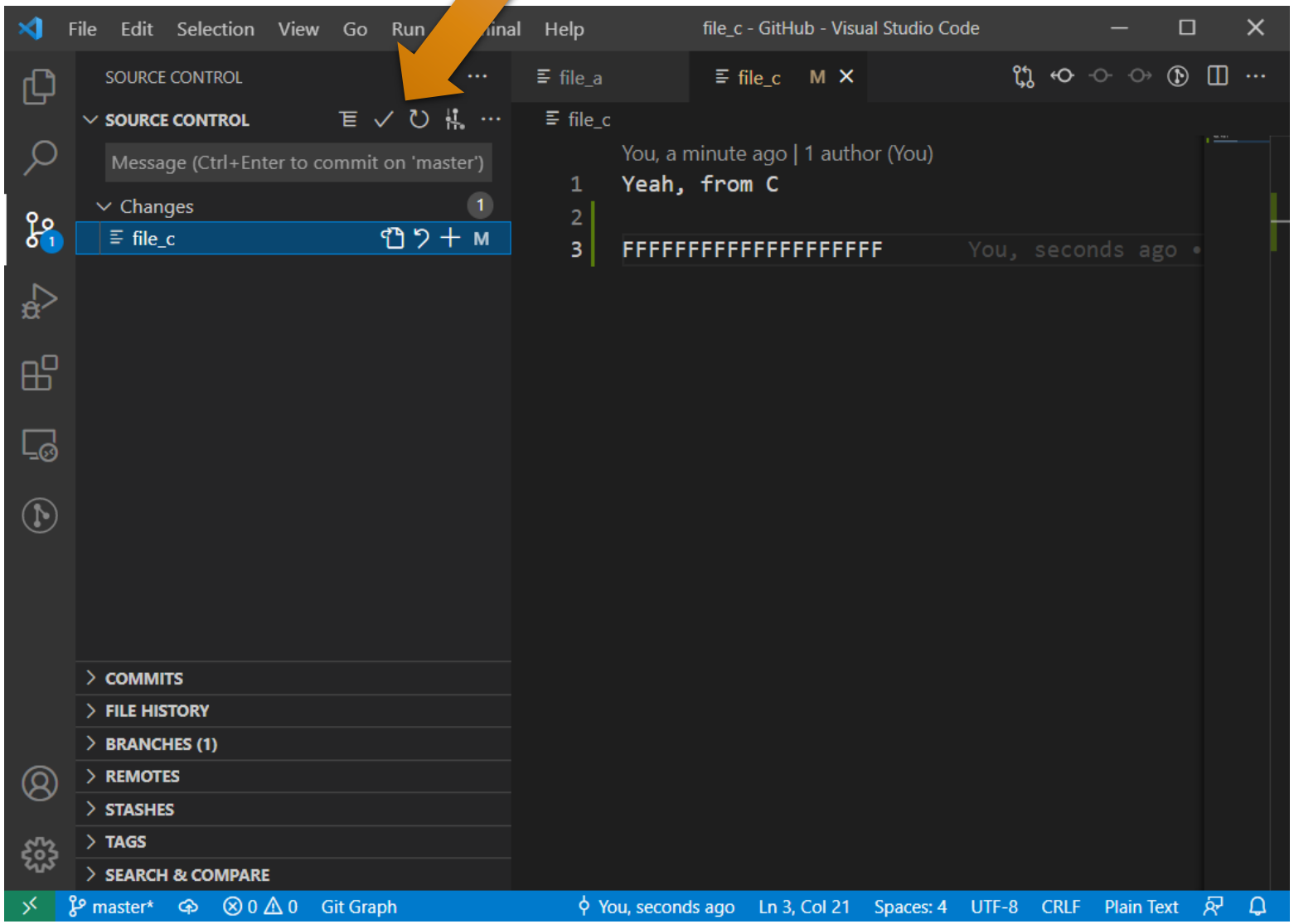
How to commit with VS Code



Edit file_c

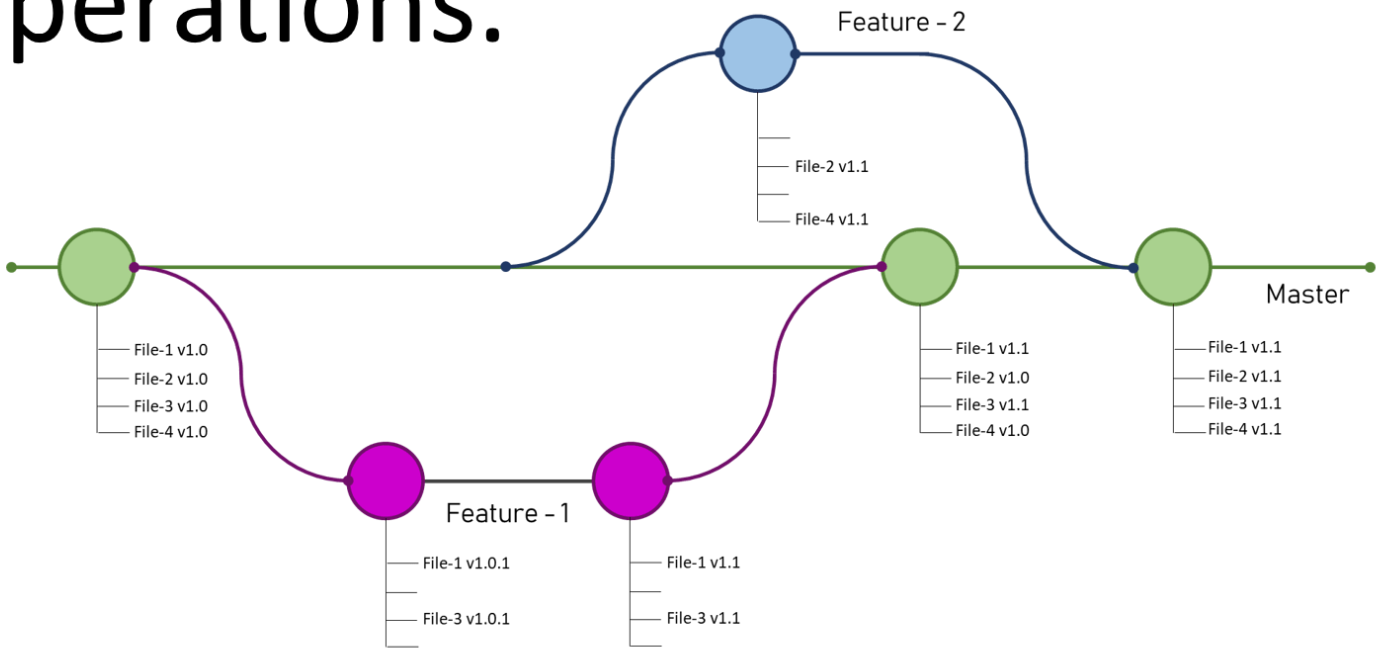
How to commit with VS Code

Type message and commit



Git Branch

GIT Branch and its Operations.



Git Branch

Command

View Branch

```
git branch
```

Create Branch

```
git branch <branch_name>
```

Remove Branch

```
git branch -d <branch_name>
```

Switch Branch

```
git checkout <branch_name>
```

Create and Switch Branch

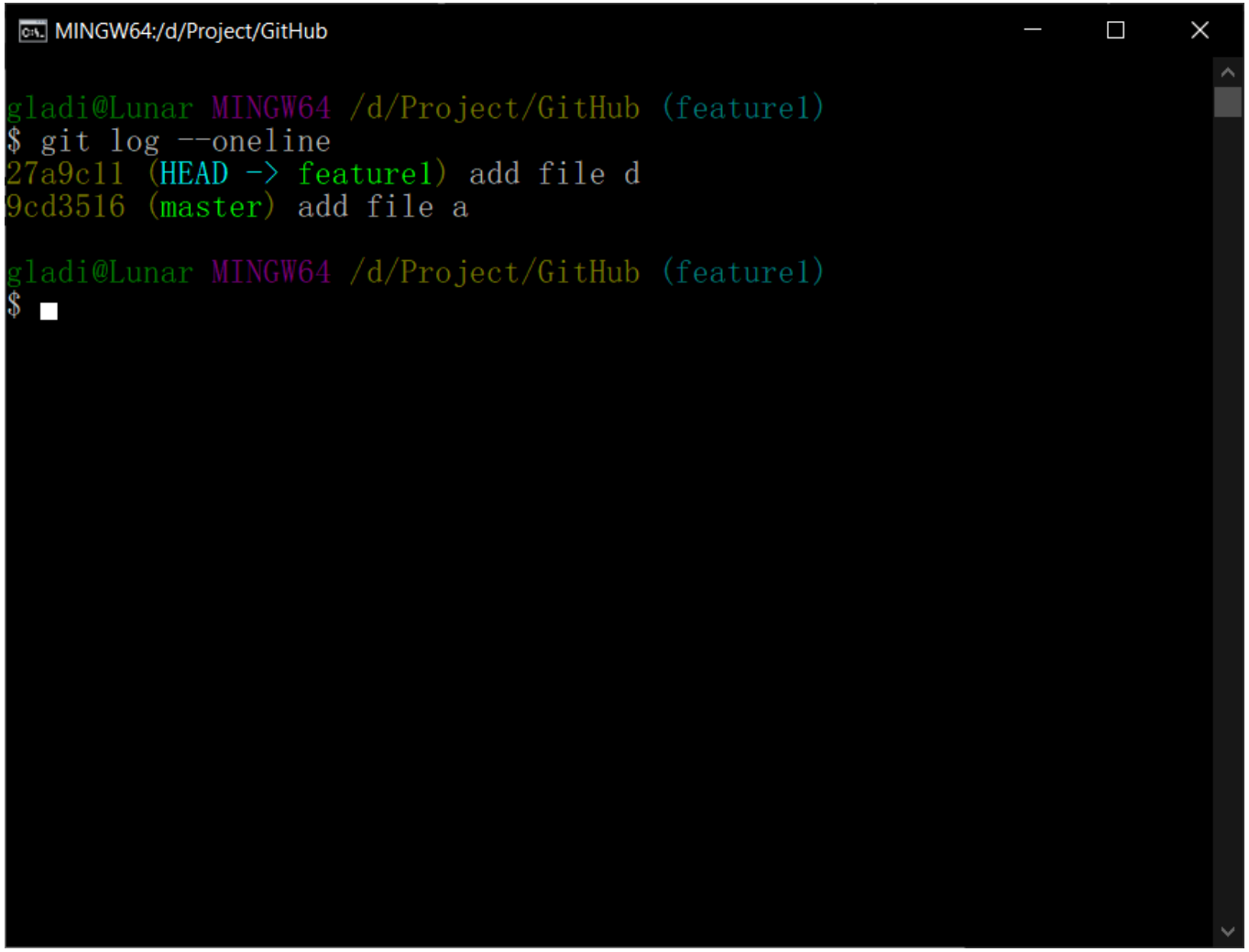
```
git checkout -b <branch_name> < commit_id (optional)>
```

Git Branch

Do this in Class

[1] Create branch “feature1” at 1st stage

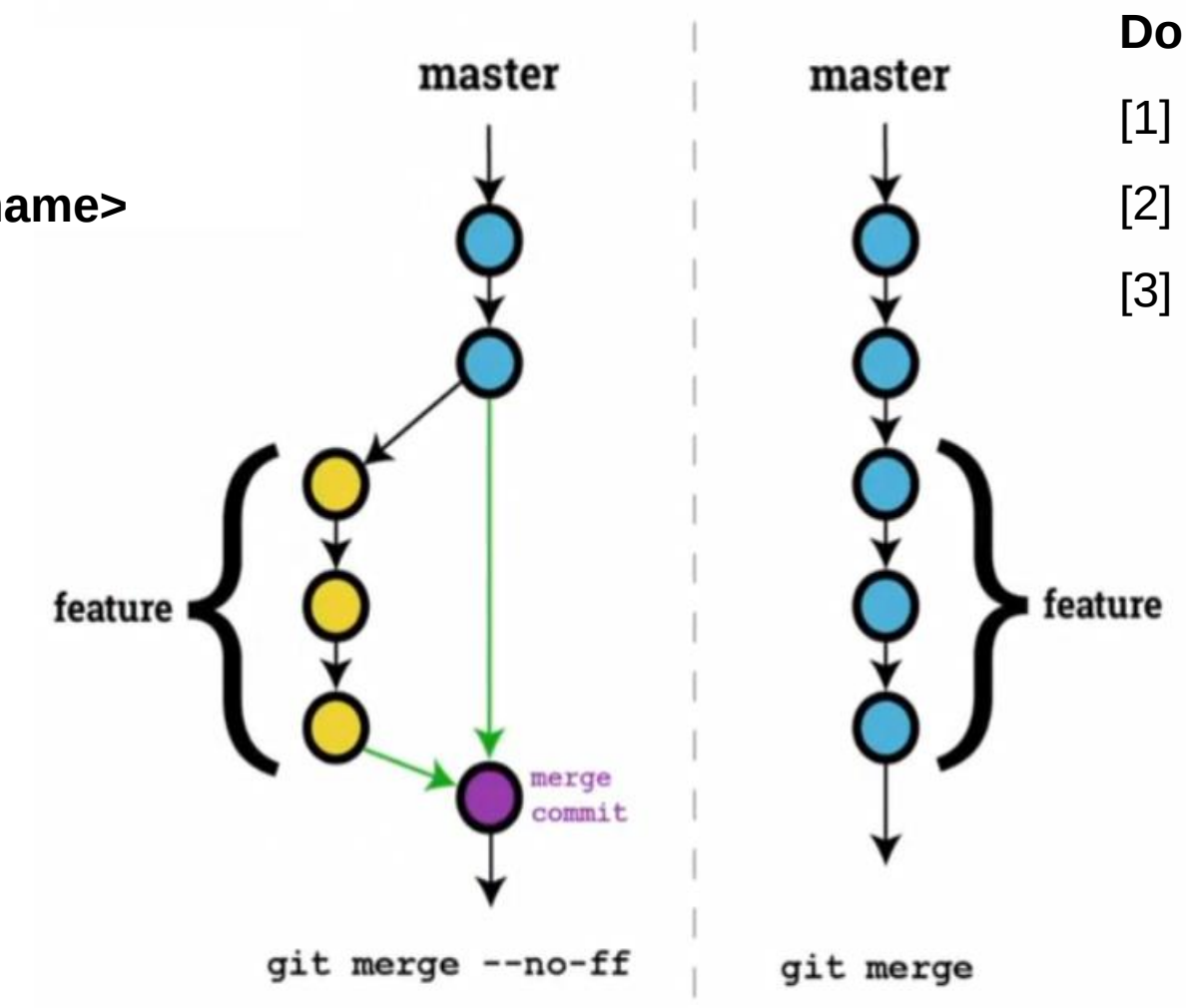
[2] Add file_d to branch “feature1”



Git Merge

Command

git merge <branch_name>



No Fast Forward **Fast Forward**

Do this in Class

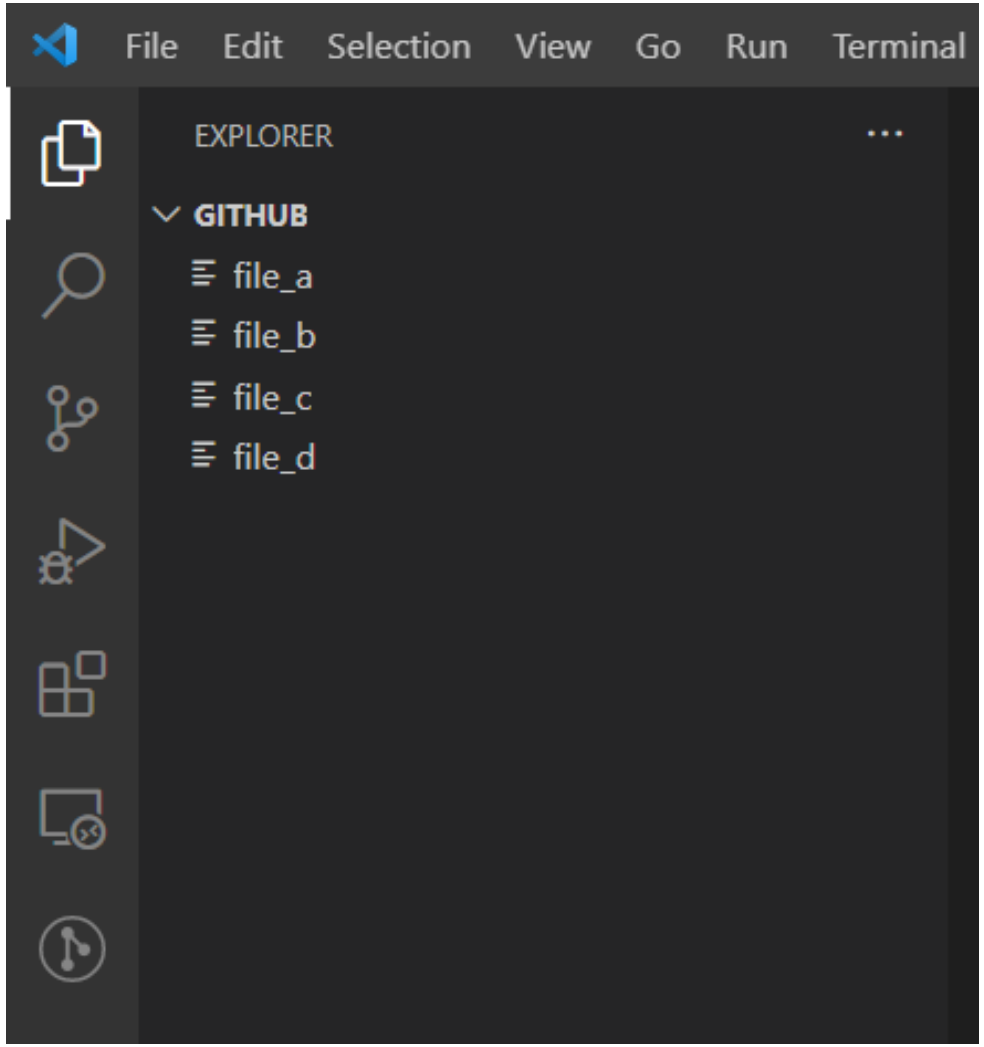
[1] Back to master 3rd stage

[2] Merge with branch “Feature1”

[3] Remove branch “Feature1”

Git Merge

```
gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git status
On branch master
nothing to commit, working tree clean

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$ git merge feature1
Merge made by the 'ort' strategy.
 file_d | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 file_d

gladi@Lunar MINGW64 /d/Project/GitHub (master)
$
```



After merge we will have

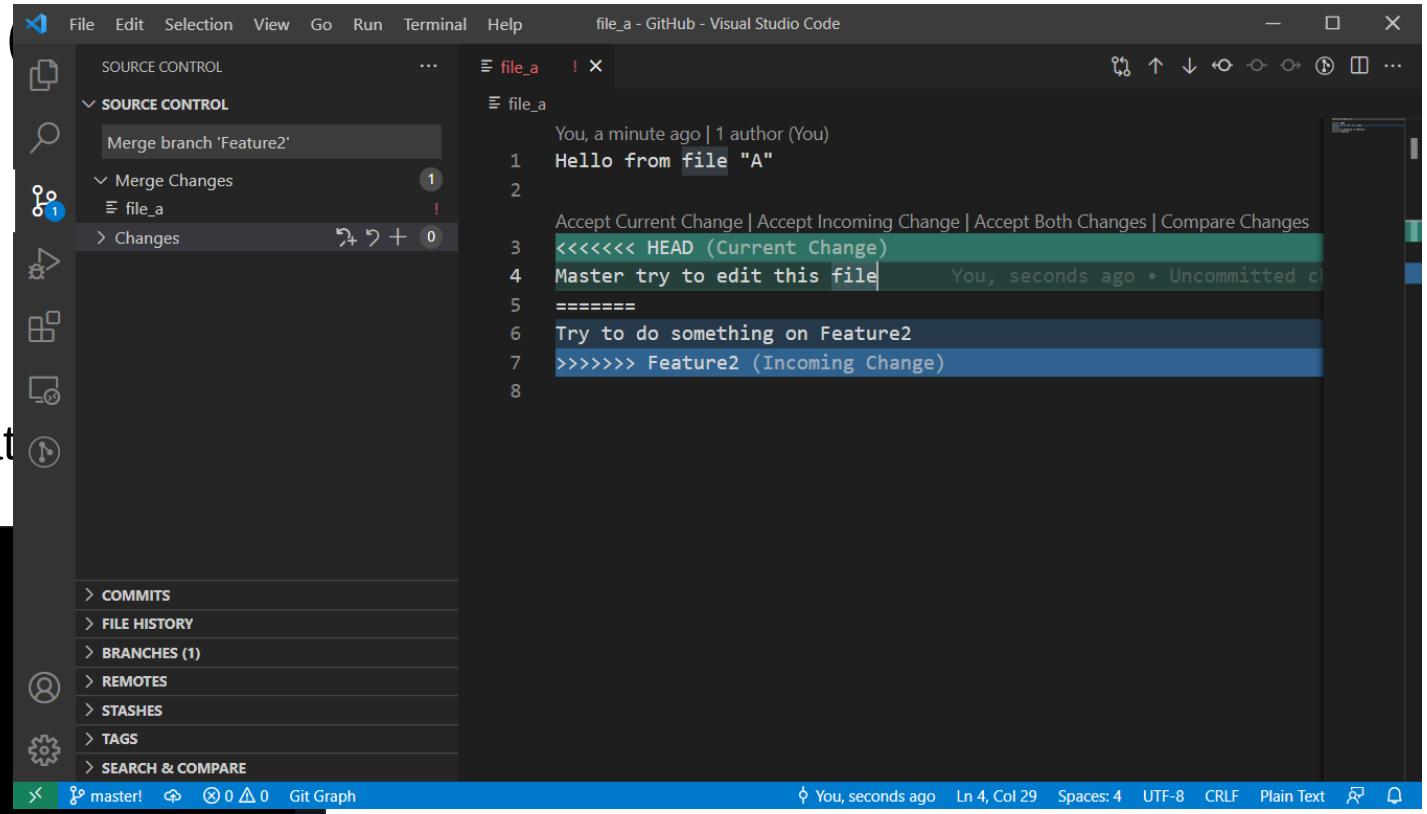
file_a, file_b, file_c, file_d

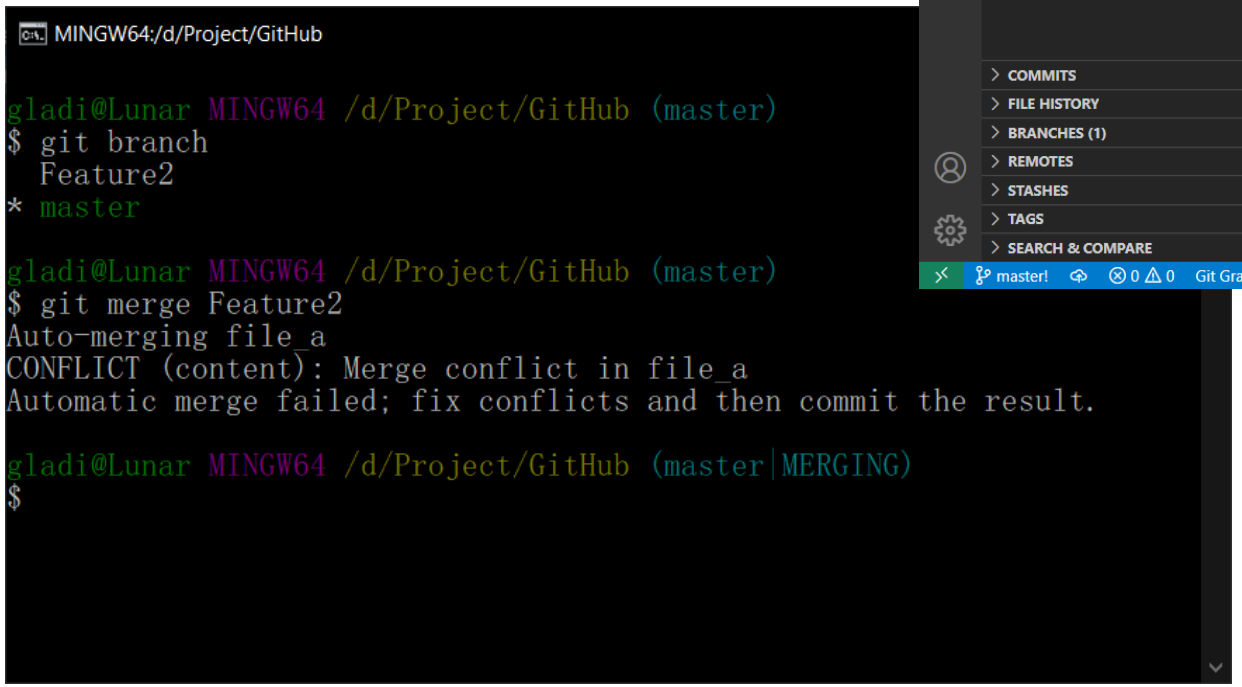
Do this in Class

[1] Create new branch “Feature2”

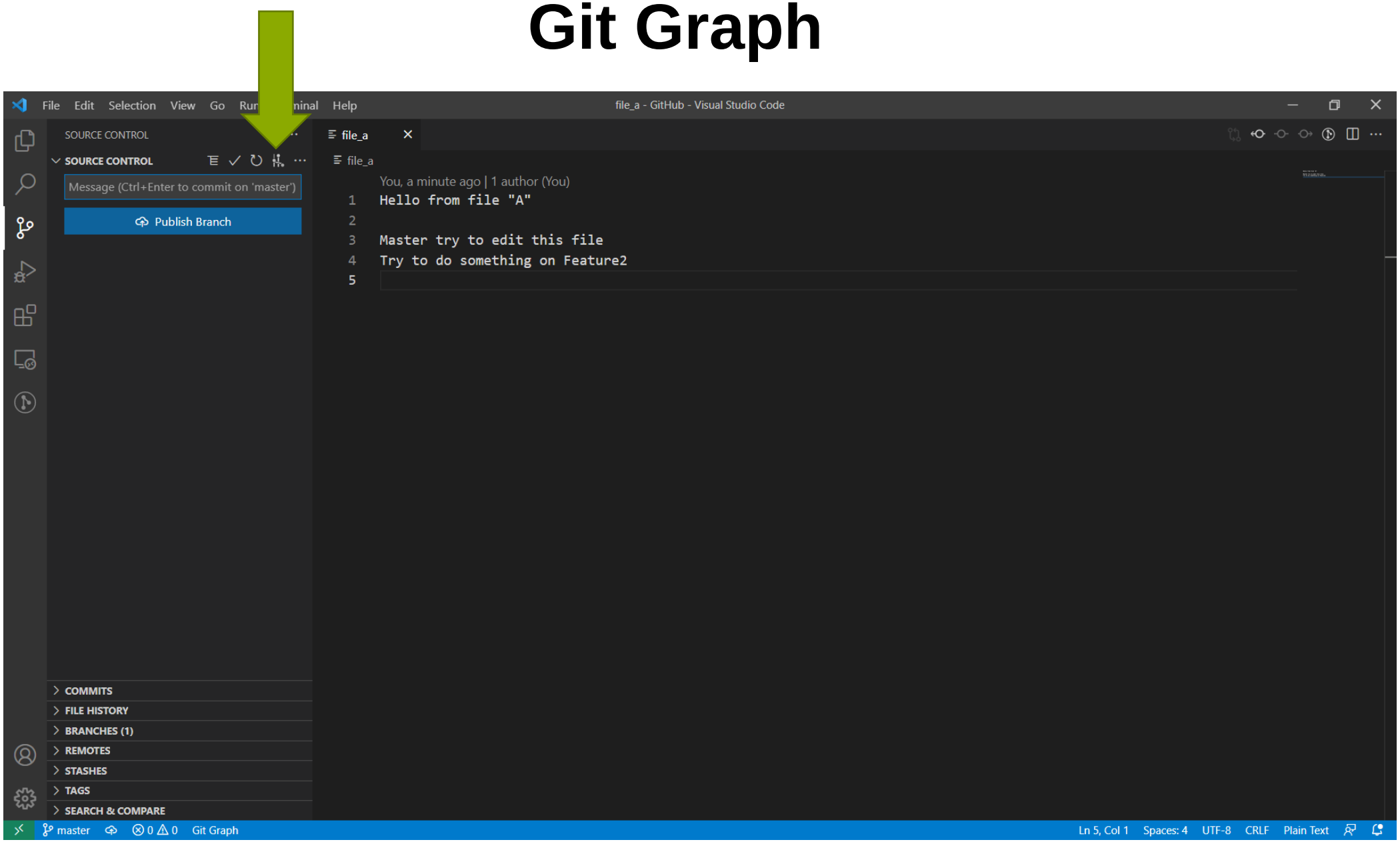
[2] Edit “file_a” (replace text in this file)

[3] Try to merge branch “master” with “Feat

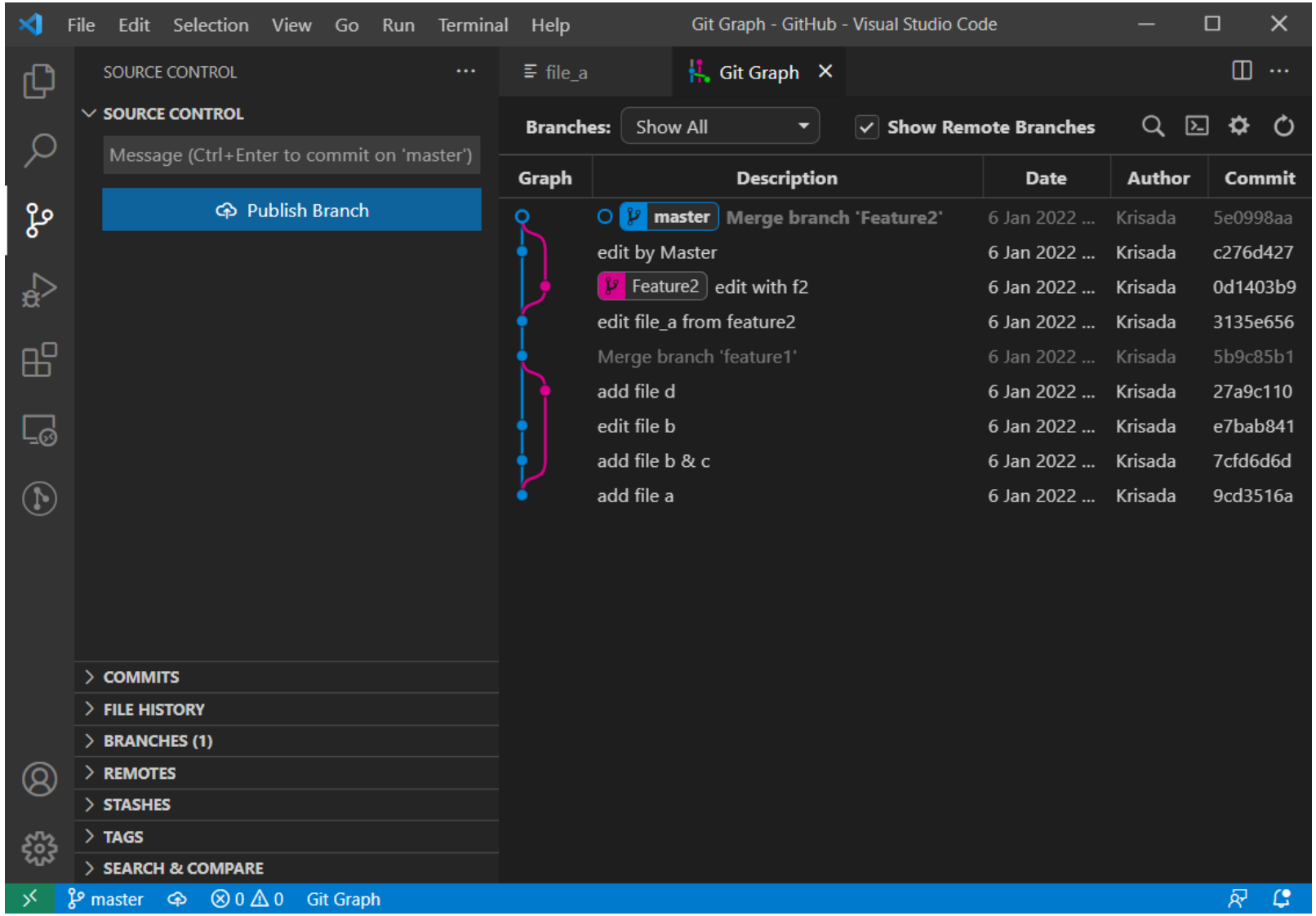


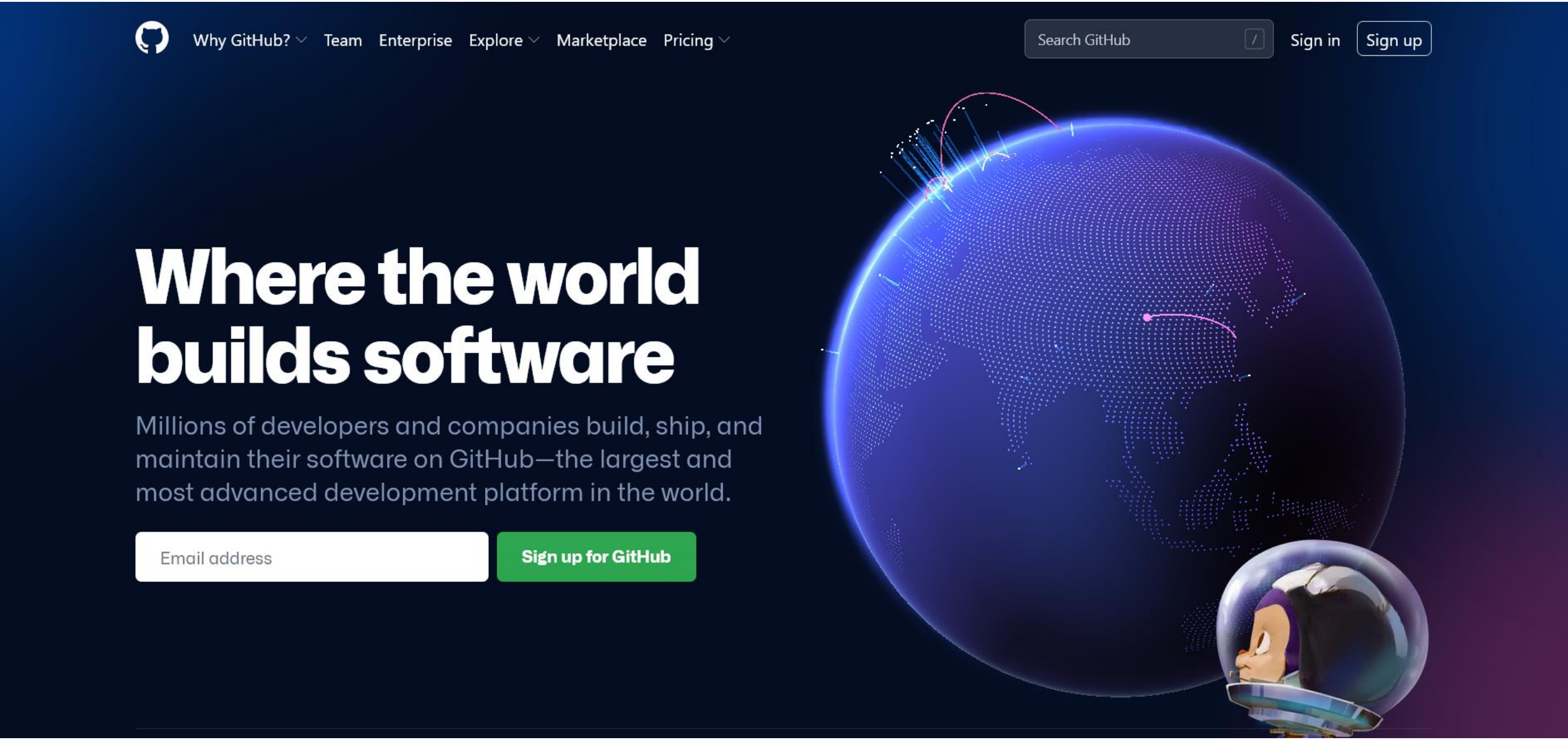


Git Graph


File Edit Selection View Go Run Terminal Help
file_a - GitHub - Visual Studio Code
SOURCE CONTROL
SOURCE CONTROL
Message (Ctrl+Enter to commit on 'master')
Publish Branch
file_a
file_a
You, a minute ago | 1 author (You)
Hello from file "A"
Master try to edit this file
Try to do something on Feature2
COMMITS
FILE HISTORY
BRANCHES (1)
REMOTES
STASHES
TAGS
SEARCH & COMPARE
master
0 0
Git Graph
Ln 5, Col 1
Spaces: 4
UTF-8
CRLF
Plain Text

Git Graph


File Edit Selection View Go Run Terminal Help
Git Graph - GitHub - Visual Studio Code
SOURCE CONTROL
SOURCE CONTROL
Message (Ctrl+Enter to commit on 'master')
Publish Branch
COMMITS
FILE HISTORY
BRANCHES (1)
REMOTES
STASHES
TAGS
SEARCH & COMPARE
file_a
Git Graph
Branches: Show All
Show Remote Branches
Graph Description Date Author Commit
master Merge branch 'Feature2' 6 Jan 2022 ... Krisada 5e0998aa
edit by Master 6 Jan 2022 ... Krisada c276d427
Feature2 edit with f2 6 Jan 2022 ... Krisada 0d1403b9
edit file_a from feature2 6 Jan 2022 ... Krisada 3135e656
Merge branch 'feature1' 6 Jan 2022 ... Krisada 5b9c85b1
add file d 6 Jan 2022 ... Krisada 27a9c110
edit file b 6 Jan 2022 ... Krisada e7bab841
add file b & c 6 Jan 2022 ... Krisada 7cfd6d6d
add file a 6 Jan 2022 ... Krisada 9cd3516a
master
0 0
Git Graph


Why GitHub?
Team
Enterprise
Explore
Marketplace
Pricing
Search GitHub
/
Sign in
Sign up
Where the world builds software
Millions of developers and companies build, ship, and maintain their software on GitHub—the largest and most advanced development platform in the world.
Email address
Sign up for GitHub

Product Team Enterprise Explore Marketplace Pricing Search / Sign in Sign up

adminho / Thai-IT-community Public

Notifications Fork 31 Star 87

Code Issues Pull requests 1 Actions Projects Wiki Security Insights

master 1 branch 0 tags Go to file Code

adminho Update README.md c02f098 on Apr 9, 2021 61 commits

README.md Update README.md 15 months ago

README.md

Thai-IT-community

ลิสต์รายชื่อกลุ่ม Facebook ของวงการไอทีแห่งประเทศไทย

เนื่องจากที่ผ่านมาได้เกิดกลุ่มบน Facebook ด้านไอทีของประเทศไทยมากมาย เป็นแหล่งเหล่าคนในวงการไอที เอาไว้ใช้ปรึกษา พร้อมทั้งแลกเปลี่ยนข้อมูลซึ่งกันและกัน ผมจึงได้รวบรวมกลุ่มไอทีพร้อมทั้งไว้ที่นี่นะจุดเดียว ใครมีแนะนำอะไรก็ทักมาได้ที่แฟนเพจ หรือจะ submit ผ่าน github ก็ยินดี ผมหวังว่ามันจะเป็นประโยชน์ต่อสังคมไทยครับ (จะพยายามให้เนื้อหาอัพเดตเวลาเกิดกลุ่มต่างๆ ขึ้นมา)

1. ขอให้มีสมาชิกสัก 2,000 คนขึ้นไปนะครับ
2. สำหรับคนไทยเท่านั้นนะครับ

About

No description, website, or topics provided.

Readme

87 stars

9 watching

31 forks

Releases

No releases published

Packages

No packages published

Contributors 2

adminho

GitHub - adminho/Thai-IT-community