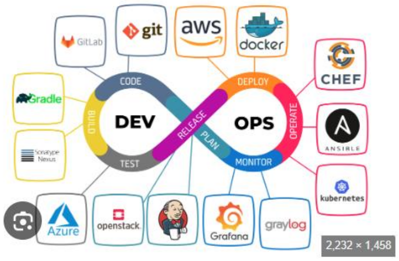
GIT Notes for batch 51

DEVOPS

Fronted & Backend & Scripting



APPlication

v1 --> code

v2 --> code

v3 --> code

Def:
Git is DISTRIBUTED VERSION CONTROL system to develop and manage Projects (code) with high speed and efficient.

Uses :

OPEN SOURCE # public

SCALABLE # number of users

Distributed # one repo can access by many people at time

Secure # lock ,private

Purpose:

SAVE the TIME

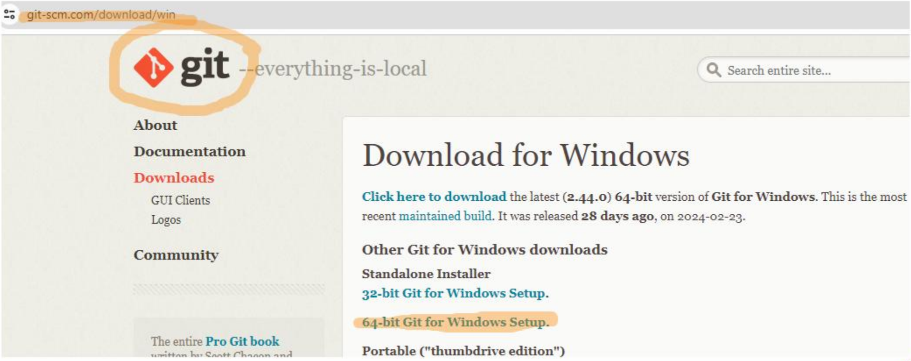
OFFLINE WORKING

TRACK CHANGES

1. GIT
2. GIT HUB

Software tool we org code as different versions

LOCAL SYSTEM



Download for Windows

Click here to download the latest (2.44.0) 64-bit version of Git for Windows. This is the most recent maintained build. It was released 28 days ago, on 2024-02-23.

Other Git for Windows downloads

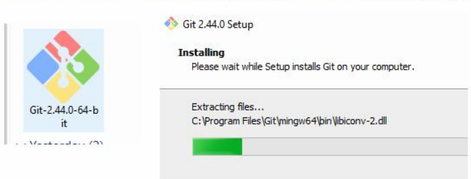
Standalone Installer

32-bit Git for Windows Setup.

64-bit Git for Windows Setup.

Portable ("thumbdrive edition")

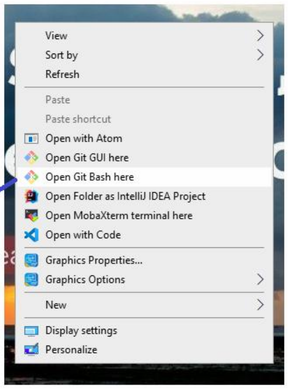
The entire Pro Git book written by Scott Chacon and

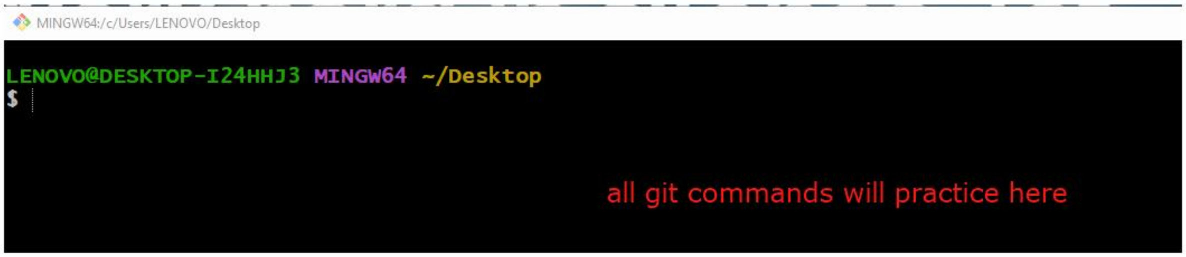


Installing
Please wait while Setup installs Git on your computer.

Extracting files...
C:\Program Files\Git\mingw64\bin\libiconv-2.dll

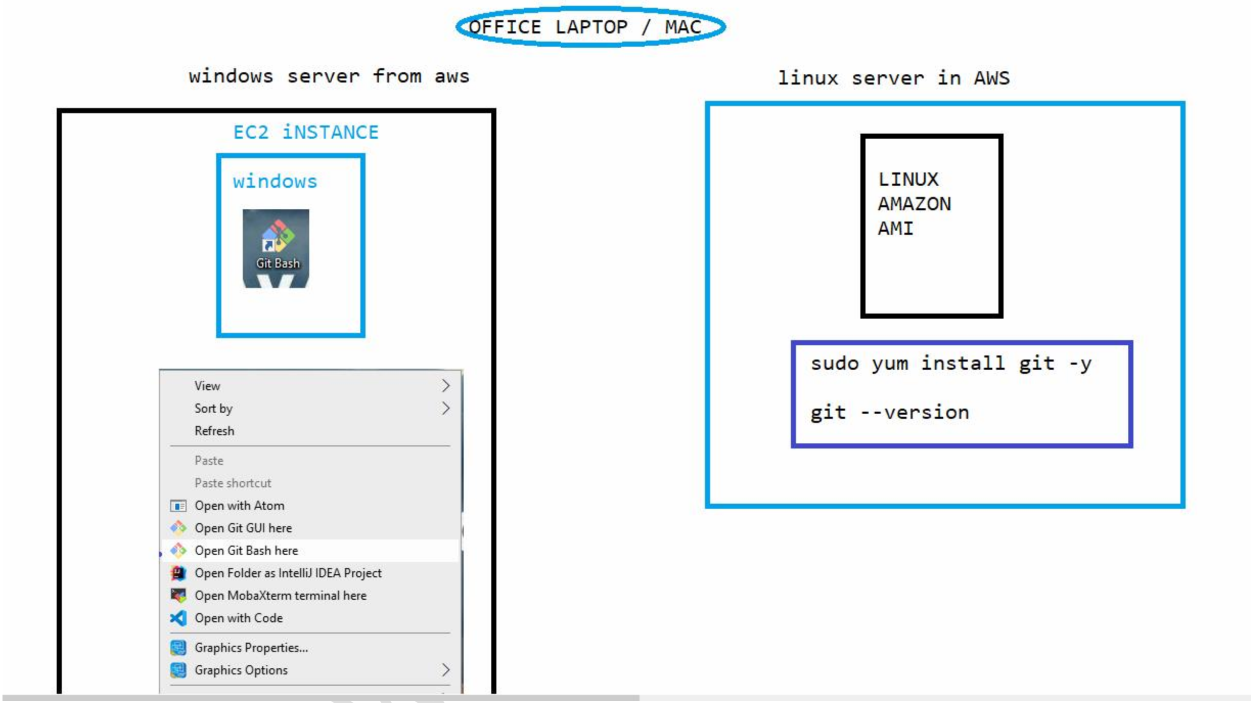
RGT CLICK ON DESKTOP





```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop
$
```

all git commands will practice here



OFFICE LAPTOP / MAC

windows server from aws

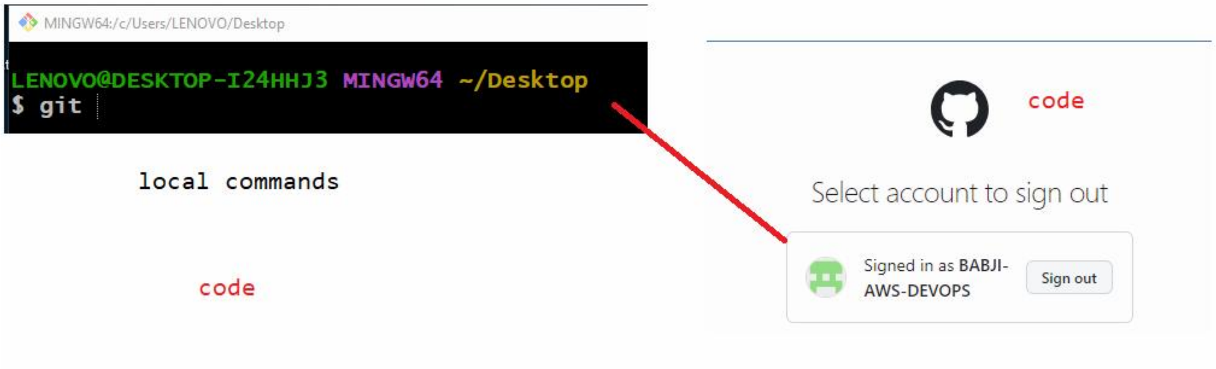
EC2 iNSTANCE

windows

linux server in AWS

LINUX
AMAZON
AMI

```
sudo yum install git -y

git --version
```



```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop
$ git
```

local commands

code

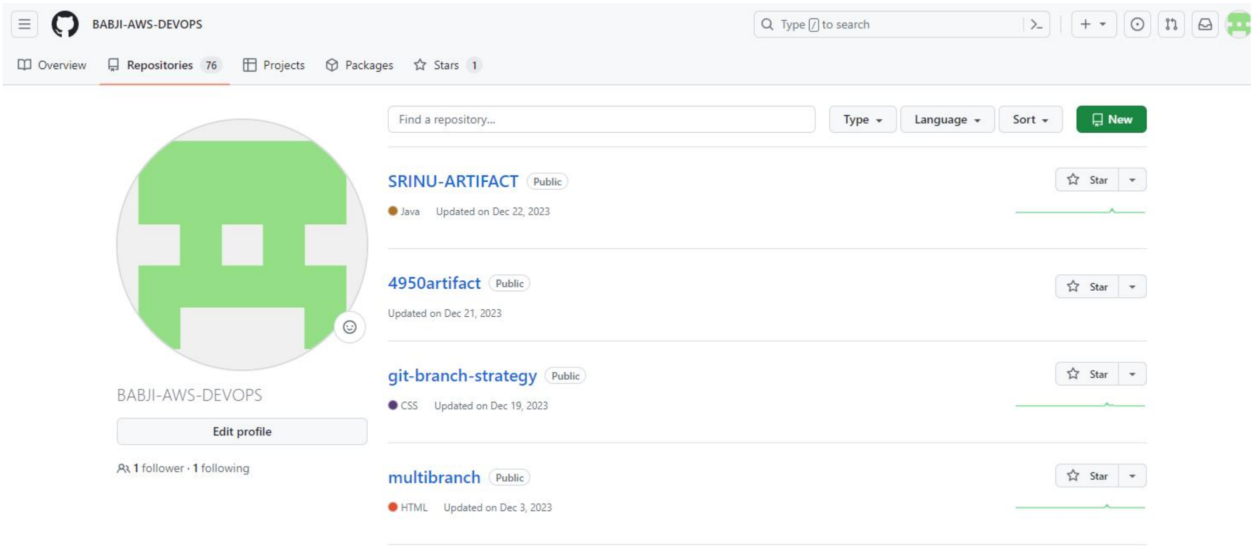
code

Select account to sign out

Signed in as BABJI-AWS-DEVOPS

Create git hub signup & sign-in :

NAME & PASSWORD from git hub



BABJI-AWS-DEVOPS

Overview · Repositories 76 · Projects · Packages · Stars 1

Find a repository... | Type | Language | Sort | New

BABJI-AWS-DEVOPS

Edit profile

1 follower · 1 following

SRINU-ARTIFACT Public — Java — Updated on Dec 22, 2023 — Star

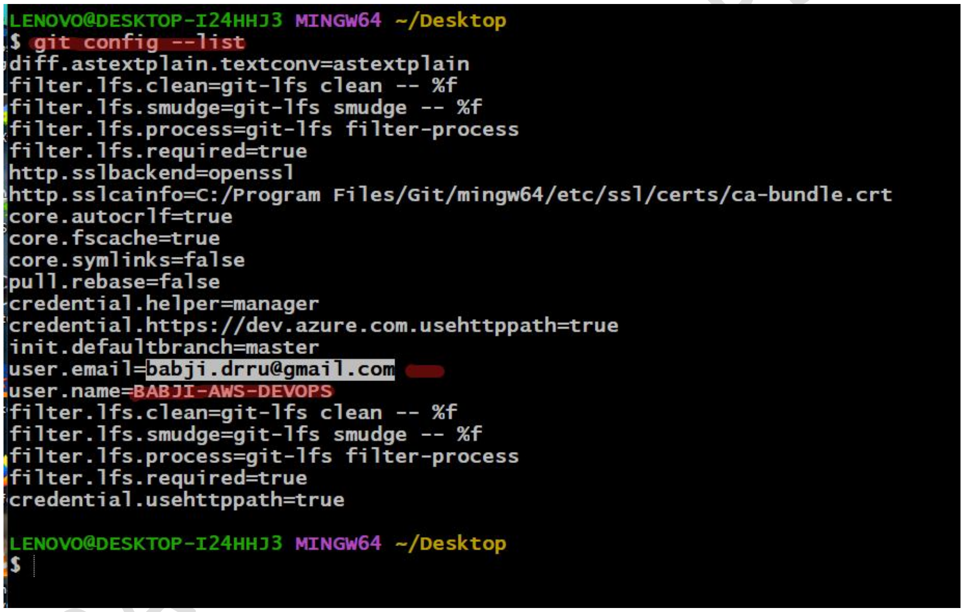
4950artifact Public — Updated on Dec 21, 2023 — Star

git-branch-strategy Public — CSS — Updated on Dec 19, 2023 — Star

multibranch Public — HTML — Updated on Dec 3, 2023 — Star

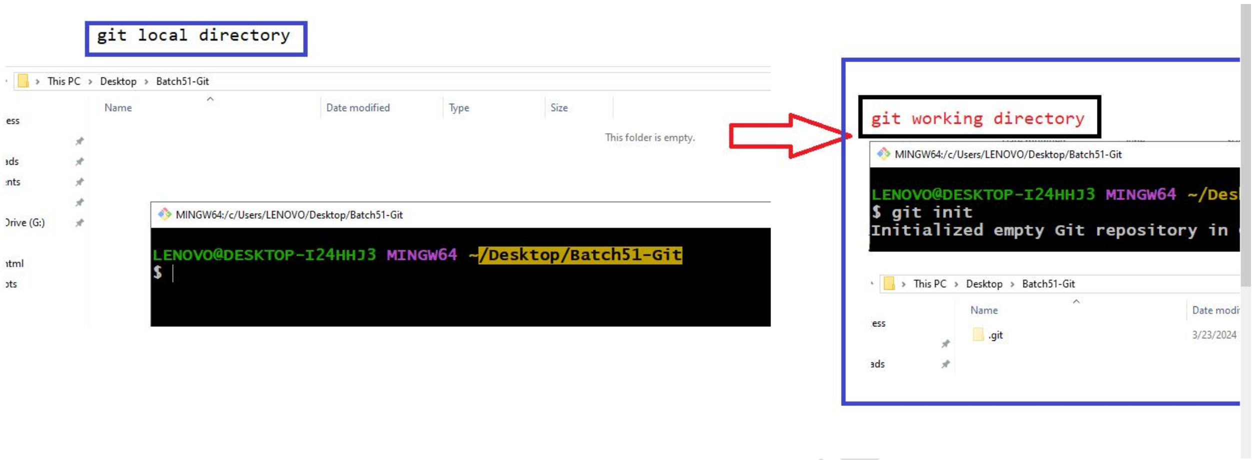
+++

Before perform git commands try to config with your git terminal need to config with hub account



```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.email=babji.drru@gmail.com [redacted]
user.name=BABJI-AWS-DEVOPS
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
credential.usehttppath=true

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop
$
```

Local Directory created as working directory.



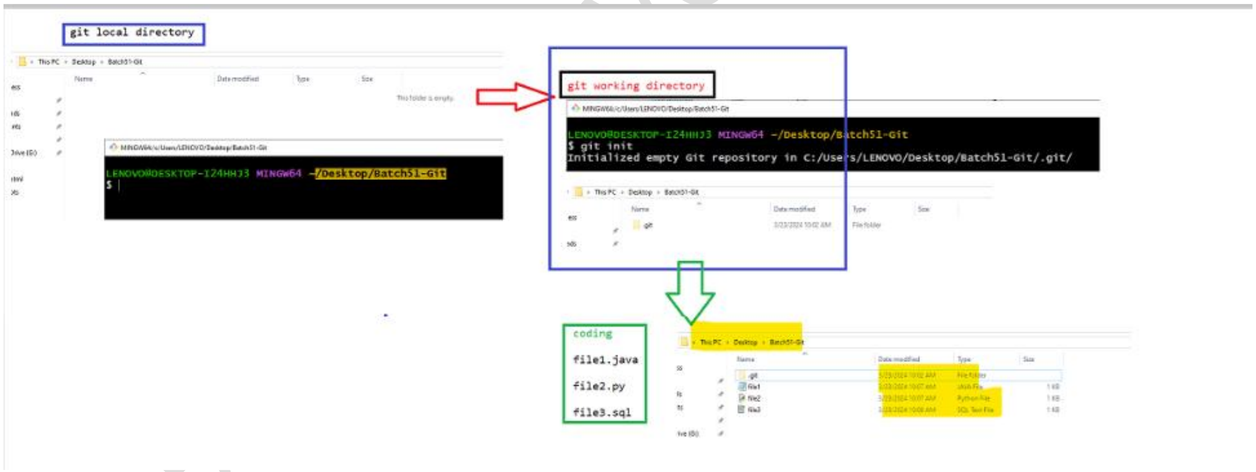
MINGW64:/c/Users/LENOVO/Desktop/Batch51-Git

```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/Batch51-Git
$ |
```

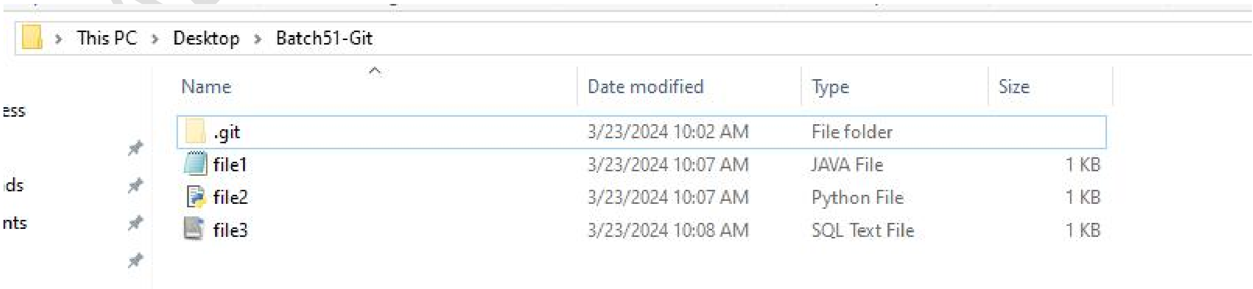
MINGW64:/c/Users/LENOVO/Desktop/Batch51-Git

```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Des
$ git init
Initialized empty Git repository in
```

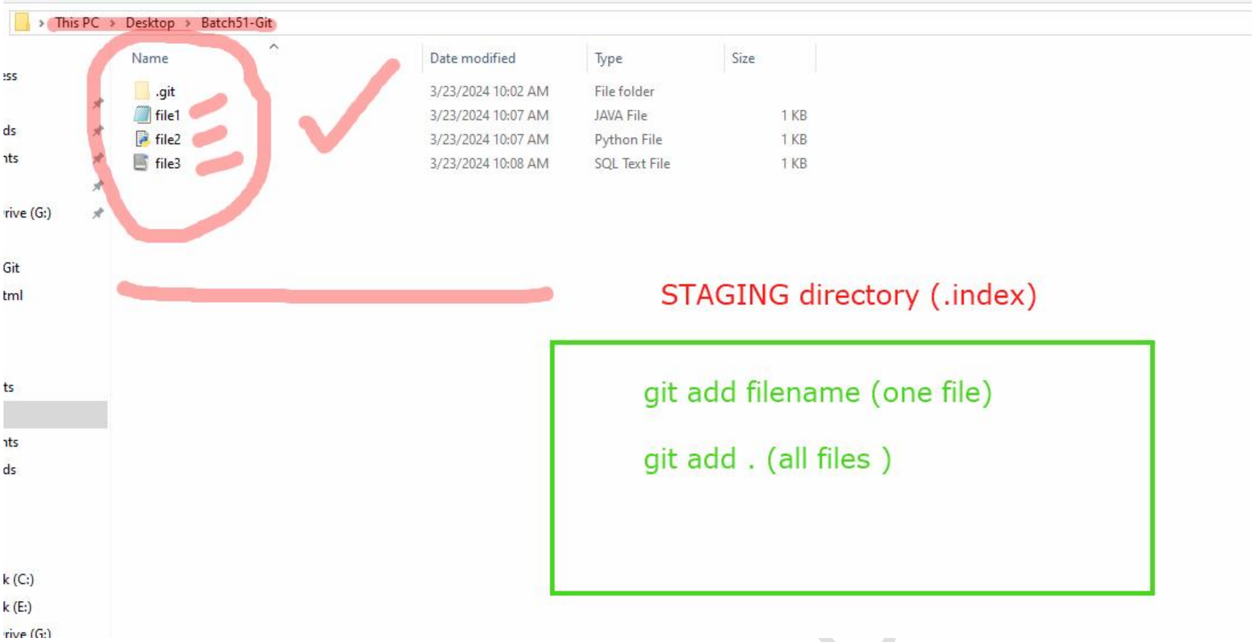
You can store your program files



```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/Batch51-Git
$ git init
Initialized empty Git repository in C:/Users/LENOVO/Desktop/Batch51-Git/.git/
```



Name	Date modified	Type	Size
.git	3/23/2024 10:02 AM	File folder	
file1	3/23/2024 10:07 AM	JAVA File	1 KB
file2	3/23/2024 10:07 AM	Python File	1 KB
file3	3/23/2024 10:08 AM	SQL Text File	1 KB



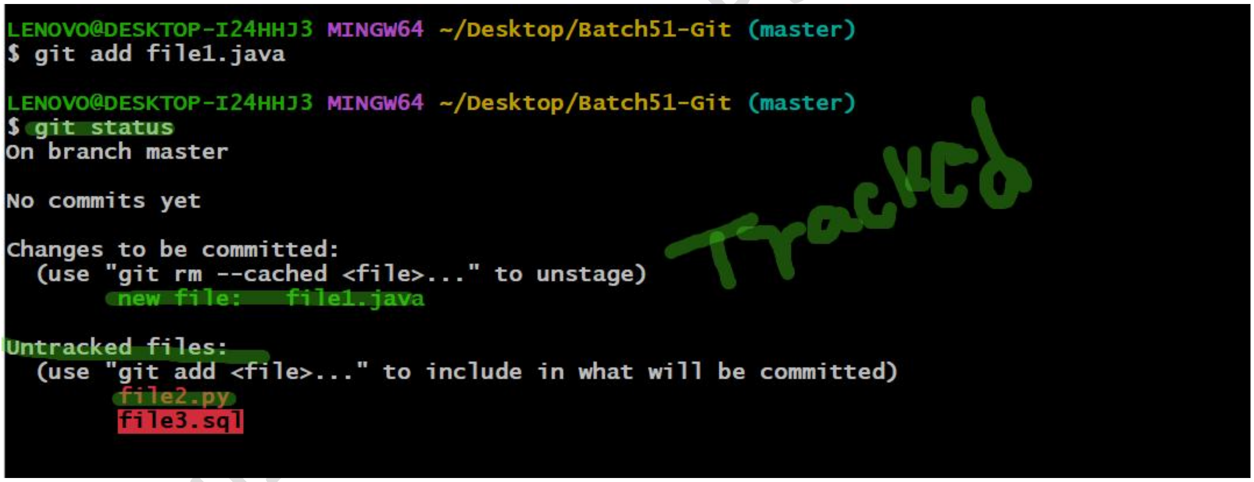
This PC > Desktop > Batch51-Git

Name	Date modified	Type	Size
.git	3/23/2024 10:02 AM	File folder	
file1	3/23/2024 10:07 AM	JAVA File	1 KB
file2	3/23/2024 10:07 AM	Python File	1 KB
file3	3/23/2024 10:08 AM	SQL Text File	1 KB

STAGING directory (.index)

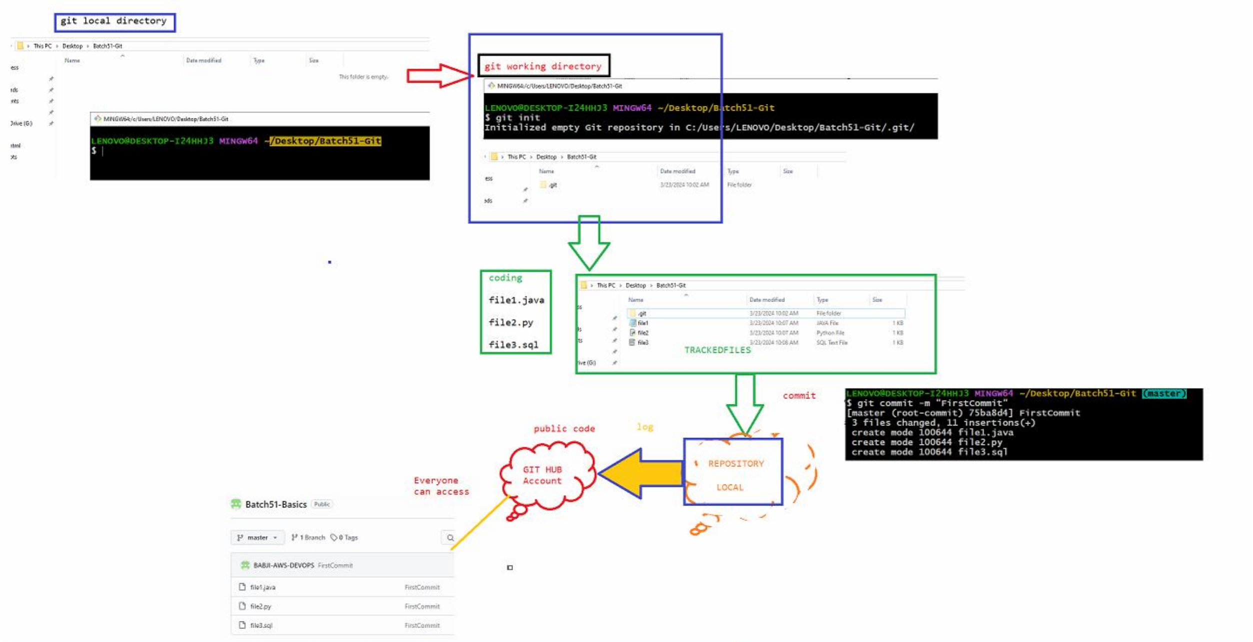
git add filename (one file)

git add . (all files)



```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/Batch51-Git (master)
$ git add file1.java

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/Batch51-Git (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   file1.java

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        file2.py
        file3.sql
```

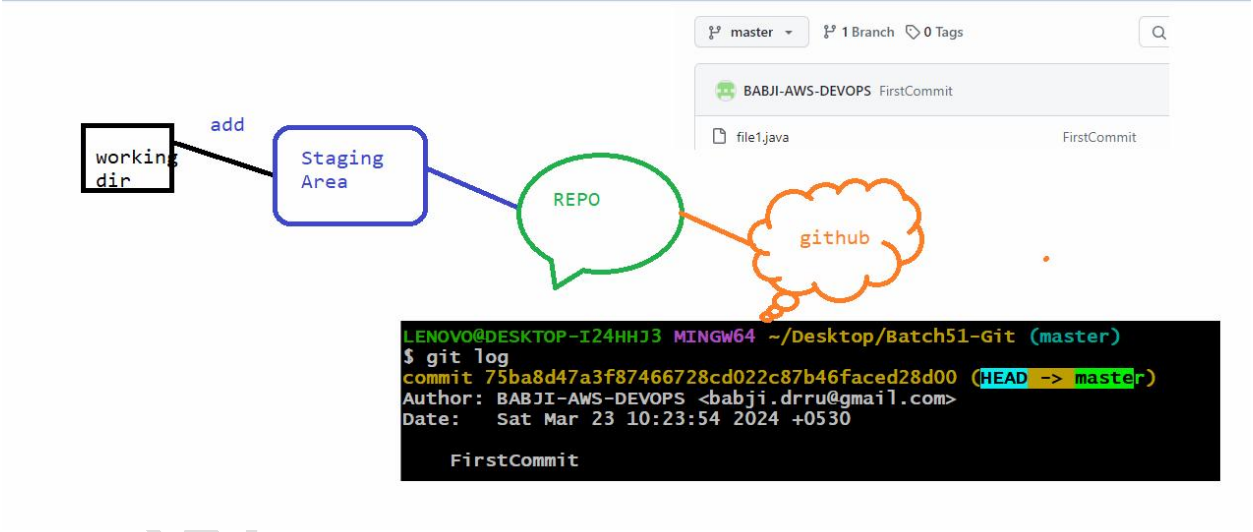
Tarcked file and untracked file



```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/Batch51-Git
$ |
```

```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/Batch51-Git
$ git init
Initialized empty Git repository in C:/Users/LENOVO/Desktop/Batch51-Git/.git/
```

```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/Batch51-Git (master)
$ git commit -m "FirstCommit"
[master (root-commit) 75ba8d4] FirstCommit
 3 files changed, 11 insertions(+)
 create mode 100644 file1.java
 create mode 100644 file2.py
 create mode 100644 file3.sql
```



```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/Batch51-Git (master)
$ git log
commit 75ba8d47a3f87466728cd022c87b46faced28d00 (HEAD -> master)
Author: BABJI-AWS-DEVOPS <babji.drru@gmail.com>
Date:   Sat Mar 23 10:23:54 2024 +0530

    FirstCommit
```

git commands

configuration

git config --list --> all details you see after configuration

USERNAME & PASSWORD of your git hub account need to add

git config --global user.name "USERNAME"

git config --global user.email "EMAIL"

+++++++++++++++++++++++++++++++++++++

git init --> intiliase as working dir

git add filename --> one perticular file move to staging area

git status --> what is status of my file. (tracked file / untrackedfile)

git add . --> all the files move to stating area

git commit --> all tracked files need move to git repo locally

git log --> behaviour of your git commit

+++
add repo in to git hub account

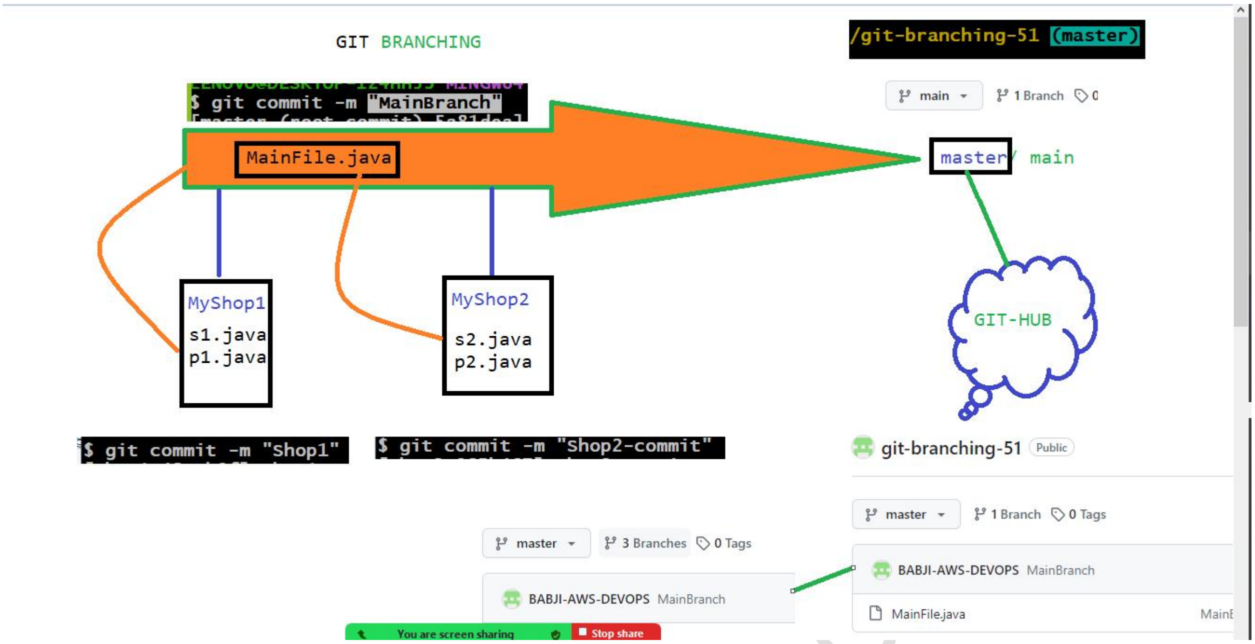
git remote add origin https://github.com/BABJI-AWS-DEVOPS/Batch51-Basics.git (URL) --> added

git push -u origin master (add your local to global)

---

public :

https://github.com/BABJI-AWS-DEVOPS/Batch51-Basics.git

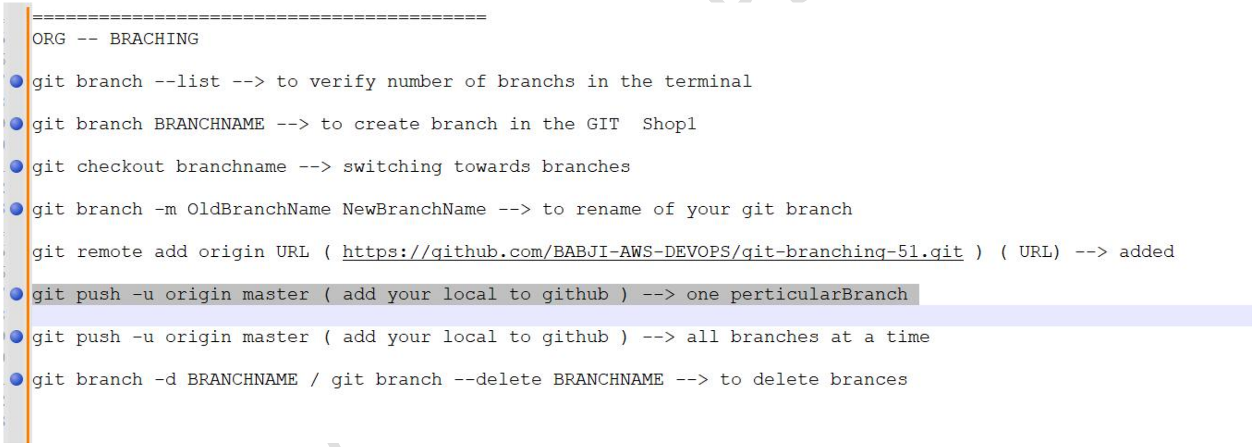


```
$ git commit -m "MainBranch"
```

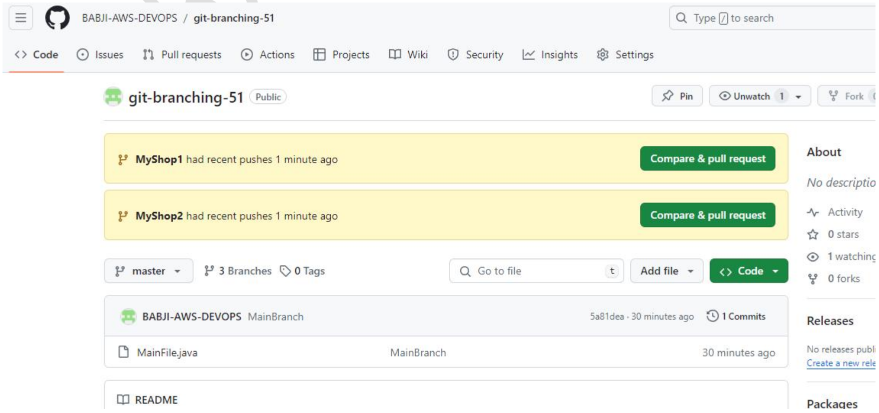
```
$ git commit -m "Shop1"
```

```
$ git commit -m "Shop2-commit"
```

```
/git-branching-51 (master)
```



```
==========================================
ORG -- BRACHING

git branch --list --> to verify number of branchs in the terminal

git branch BRANCHNAME --> to create branch in the GIT  Shop1

git checkout branchname --> switching towards branches

git branch -m OldBranchName NewBranchName --> to rename of your git branch

git remote add origin URL ( https://github.com/BABJI-AWS-DEVOPS/git-branching-51.git ) ( URL) --> added

git push -u origin master ( add your local to github ) --> one perticularBranch

git push -u origin master ( add your local to github ) --> all branches at a time

git branch -d BRANCHNAME / git branch --delete BRANCHNAME --> to delete brances
```

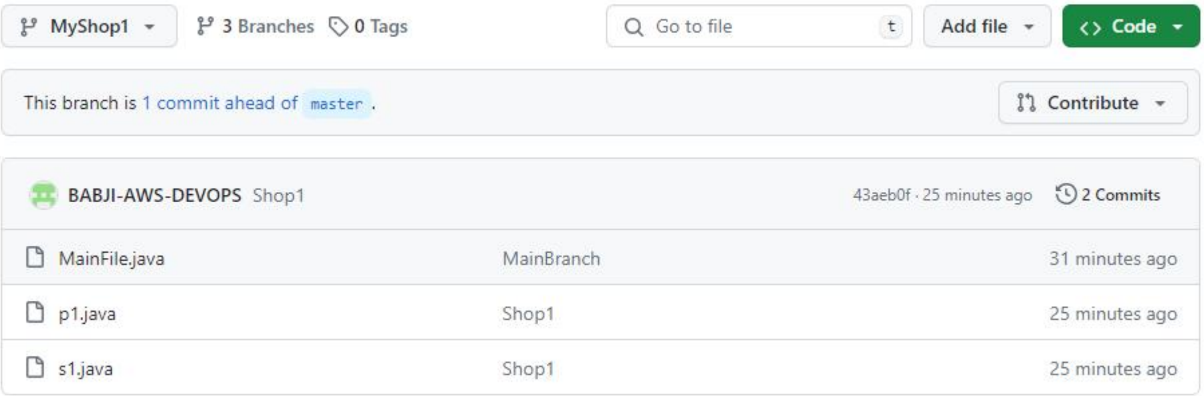


MyShop1 had recent pushes 1 minute ago

MyShop2 had recent pushes 1 minute ago

BABJI-AWS-DEVOPS MainBranch 5a81dea · 30 minutes ago 1 Commits

MainFile.java MainBranch 30 minutes ago



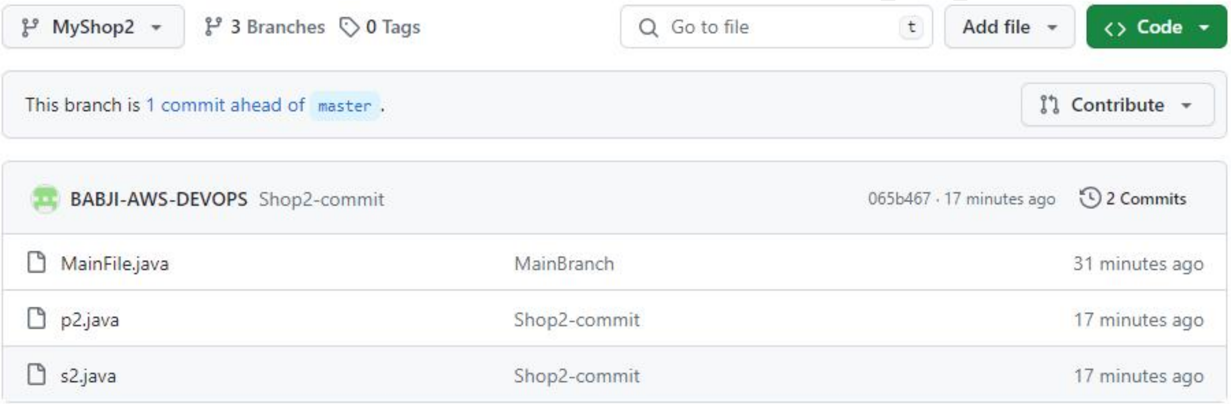
MyShop1 ▾ | 3 Branches | 0 Tags | Go to file t | Add file ▾ | <> Code ▾

This branch is 1 commit ahead of master .

Contribute ▾

BABJI-AWS-DEVOPS Shop1 — 43aeb0f · 25 minutes ago — 2 Commits

MainFile.java	MainBranch	31 minutes ago
p1.java	Shop1	25 minutes ago
s1.java	Shop1	25 minutes ago



MyShop2 ▾ | 3 Branches | 0 Tags | Go to file t | Add file ▾ | <> Code ▾

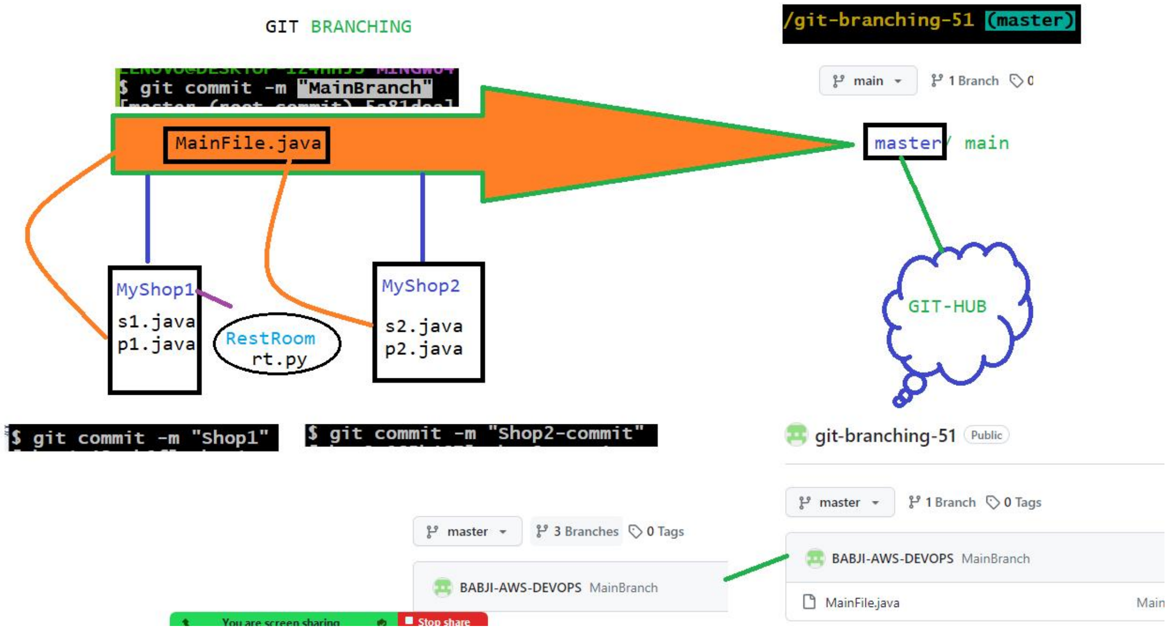
This branch is 1 commit ahead of master .

Contribute ▾

BABJI-AWS-DEVOPS Shop2-commit — 065b467 · 17 minutes ago — 2 Commits

MainFile.java	MainBranch	31 minutes ago
p2.java	Shop2-commit	17 minutes ago
s2.java	Shop2-commit	17 minutes ago

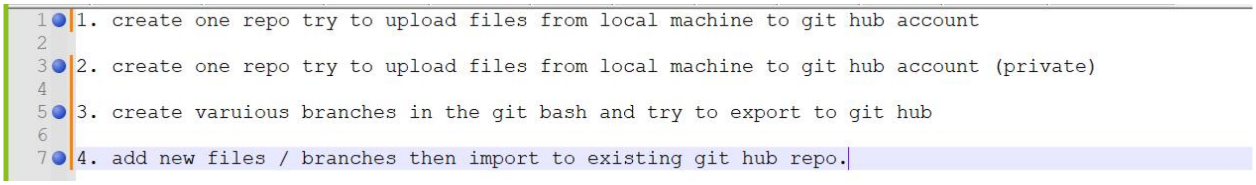
GIT –St



```
$ git commit -m "MainBranch"
```

```
$ git commit -m "shop1"
```

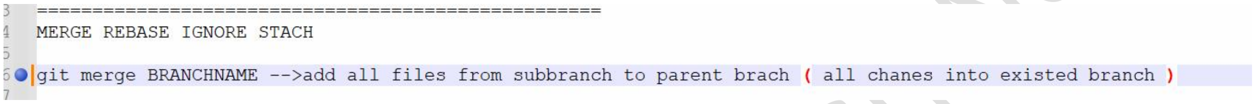
```
$ git commit -m "Shop2-commit"
```



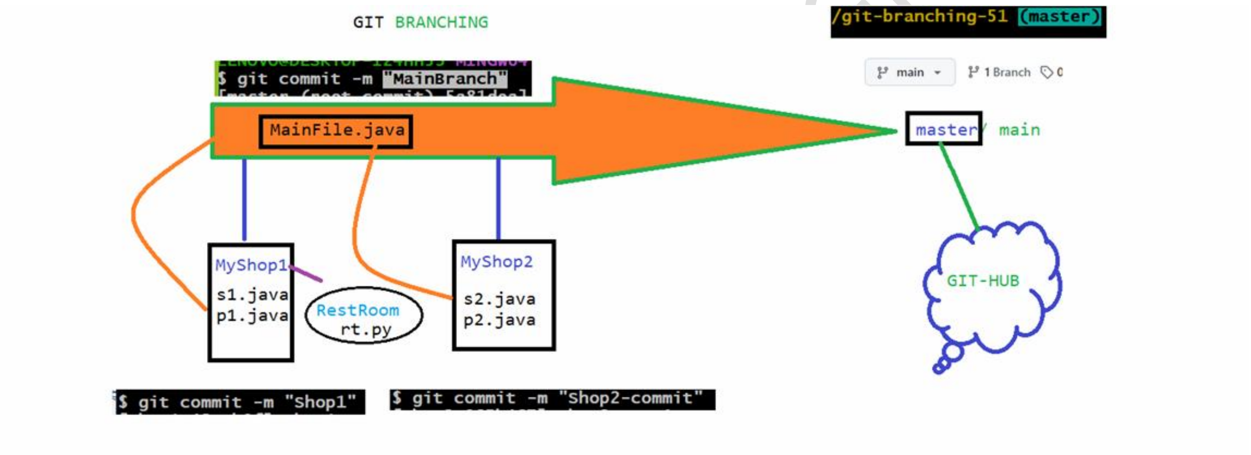
```
1. create one repo try to upload files from local machine to git hub account

2. create one repo try to upload files from local machine to git hub account (private)

3. create varuious branches in the git bash and try to export to git hub

4. add new files / branches then import to existing git hub repo.
```

++

MERGE :



```
===================================================
MERGE REBASE IGNORE STACH

git merge BRANCHNAME -->add all files from subbranch to parent brach ( all chanes into existed branch )
```



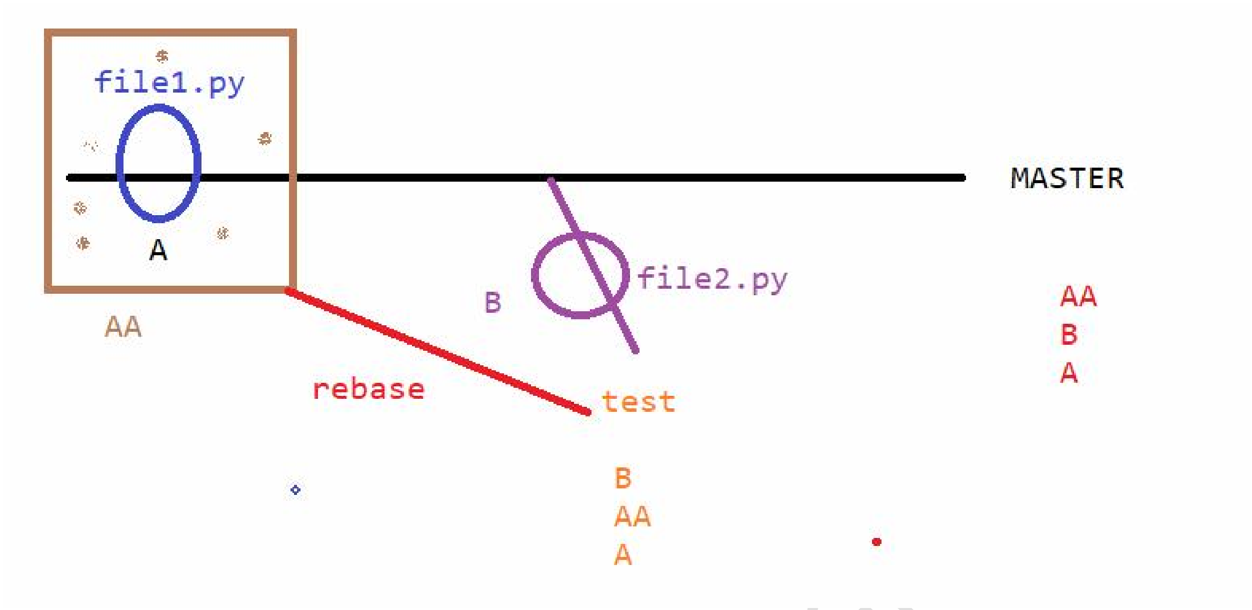
```
$ git commit -m "MainBranch"
```

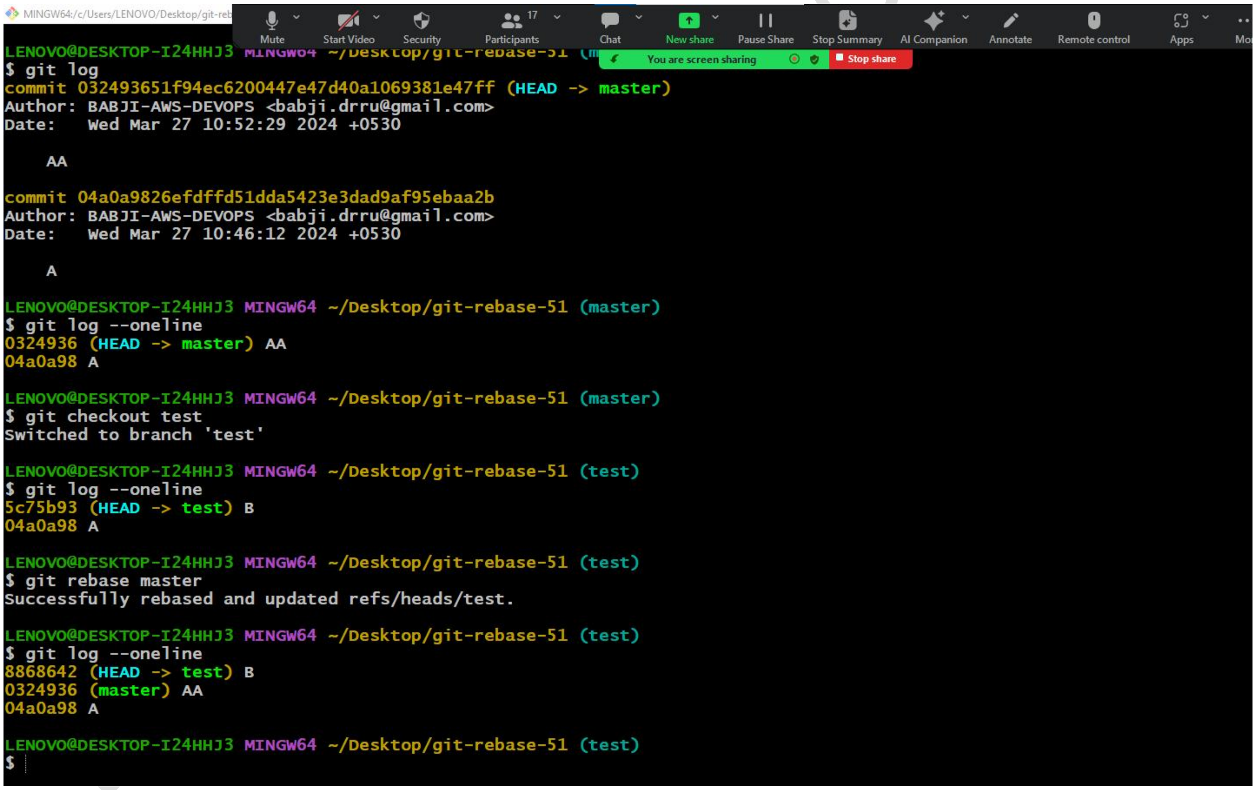
```
$ git commit -m "shop1"
```

```
$ git commit -m "Shop2-commit"
```

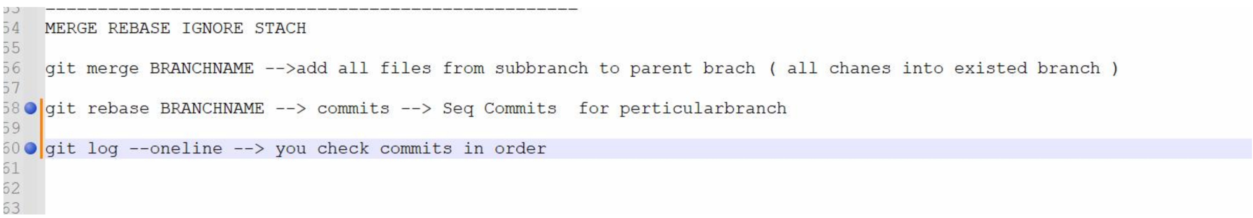
```
/git-branching-51 (master)
```

REBASE

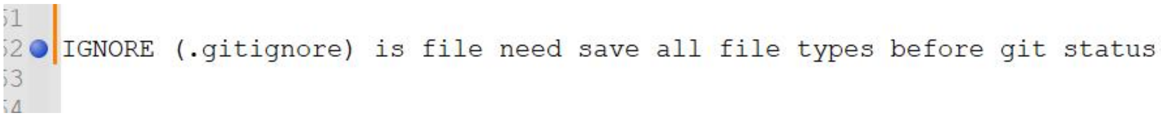




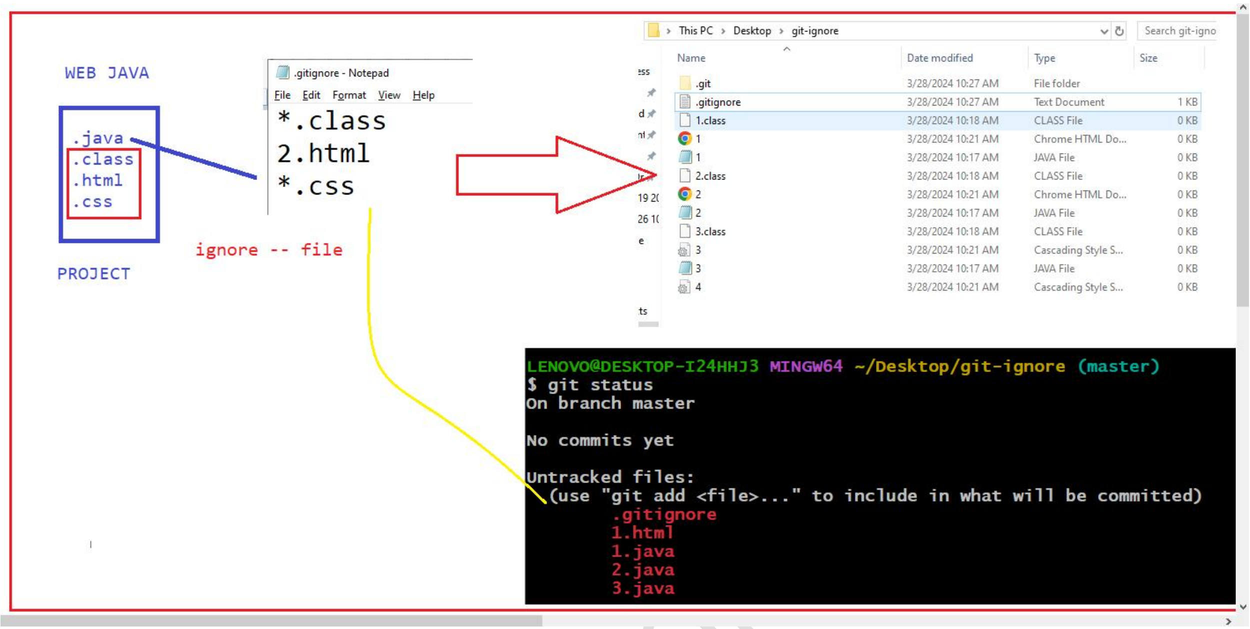
```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-rebase-51 (m
$ git log
commit 032493651f94ec6200447e47d40a1069381e47ff (HEAD -> master)
Author: BABJI-AWS-DEVOPS <babji.drru@gmail.com>
Date:   Wed Mar 27 10:52:29 2024 +0530

    AA

commit 04a0a9826efdffd51dda5423e3dad9af95ebaa2b
Author: BABJI-AWS-DEVOPS <babji.drru@gmail.com>
Date:   Wed Mar 27 10:46:12 2024 +0530

    A

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-rebase-51 (master)
$ git log --oneline
0324936 (HEAD -> master) AA
04a0a98 A

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-rebase-51 (master)
$ git checkout test
Switched to branch 'test'

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-rebase-51 (test)
$ git log --oneline
5c75b93 (HEAD -> test) B
04a0a98 A

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-rebase-51 (test)
$ git rebase master
Successfully rebased and updated refs/heads/test.

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-rebase-51 (test)
$ git log --oneline
8868642 (HEAD -> test) B
0324936 (master) AA
04a0a98 A

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-rebase-51 (test)
$
```



```
-----------------------------------------------------
MERGE REBASE IGNORE STACH

git merge BRANCHNAME -->add all files from subbranch to parent brach ( all chanes into existed branch )

git rebase BRANCHNAME --> commits --> Seq Commits  for perticularbranch

git log --oneline --> you check commits in order
```



```
IGNORE (.gitignore) is file need save all file types before git status
```



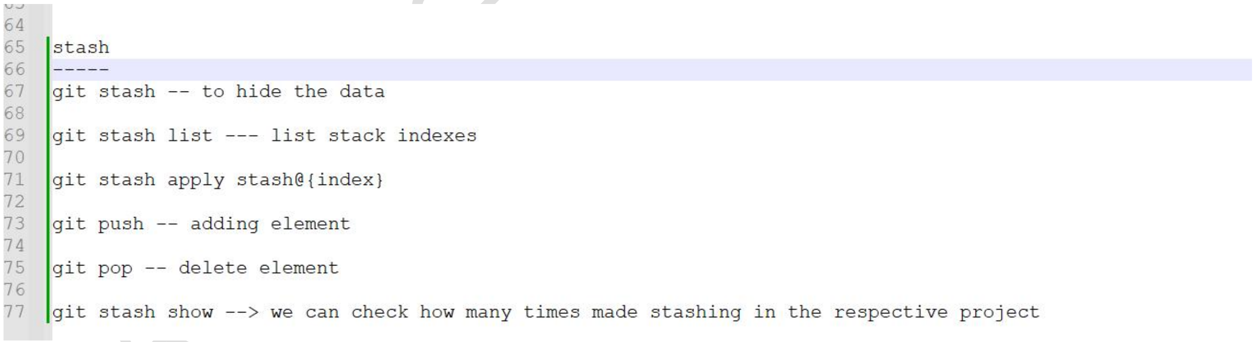
```
*.class
2.html
*.css
```

Name	Date modified	Type	Size
.git	3/28/2024 10:27 AM	File folder	
.gitignore	3/28/2024 10:27 AM	Text Document	1 KB
1.class	3/28/2024 10:18 AM	CLASS File	0 KB
1	3/28/2024 10:21 AM	Chrome HTML Do...	0 KB
1	3/28/2024 10:17 AM	JAVA File	0 KB
2.class	3/28/2024 10:18 AM	CLASS File	0 KB
2	3/28/2024 10:21 AM	Chrome HTML Do...	0 KB
2	3/28/2024 10:17 AM	JAVA File	0 KB
3.class	3/28/2024 10:18 AM	CLASS File	0 KB
3	3/28/2024 10:21 AM	Cascading Style S...	0 KB
3	3/28/2024 10:17 AM	JAVA File	0 KB
4	3/28/2024 10:21 AM	Cascading Style S...	0 KB

```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-ignore (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        1.html
        1.java
        2.java
        3.java
```

+++

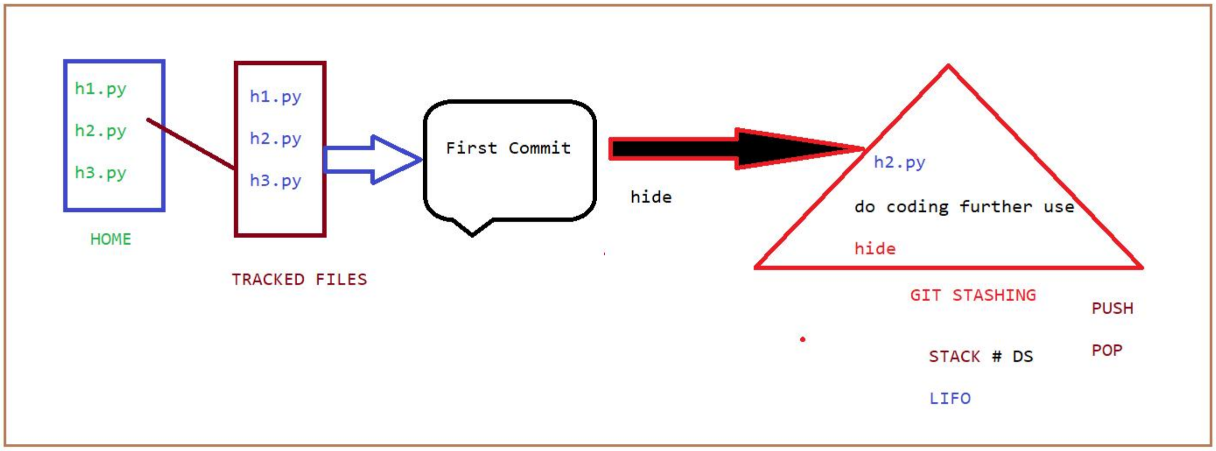
Git STASH

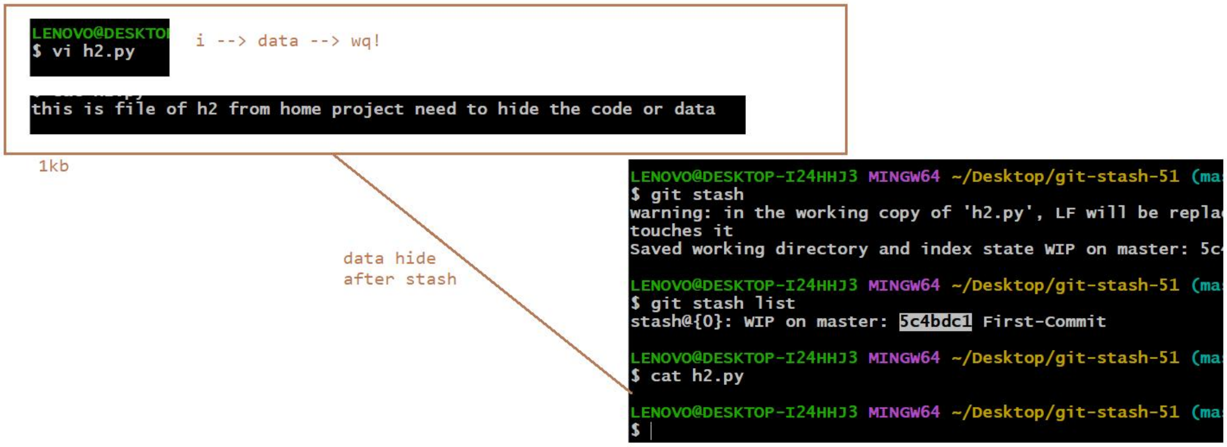


```

stash
-----
git stash -- to hide the data

git stash list --- list stack indexes

git stash apply stash@{index}

git push -- adding element

git pop -- delete element

git stash show --> we can check how many times made stashing in the respective project
```

GIT STASHING

We can hide the data / code for perticular file in the project





```
LENOVO@DESKTOI
$ vi h2.py

this is file of h2 from home project need to hide the code or data
```

```
LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-stash-51 (ma
$ git stash
warning: in the working copy of 'h2.py', LF will be repla
touches it
Saved working directory and index state WIP on master: 5c

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-stash-51 (ma
$ git stash list
stash@{0}: WIP on master: 5c4bdc1 First-Commit

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-stash-51 (ma
$ cat h2.py

LENOVO@DESKTOP-I24HHJ3 MINGW64 ~/Desktop/git-stash-51 (ma
$ |
```

```
git init
501  touch h1.py h2.py h3.py
502  git add .
503  git status
504  git commit -m "First-Commit"
505  cat h1.py
506  cat h2.py
507  cat h3.py

510  vi h2.py
511  cat h2.py
512  git stash
513  git stash list
```

```
  514  cat h2.py
  515  git stash apply stash@{0}
  516  git stash show
  517  history
```

List of git commands :

```
314  git branch --list
  315  cd branching
  316  git branch --list
  317  rd branching
  318  rd -r branching
  319  rmdir branching
  320  cd ..
  321  rmdir branching
  322  clear
  323  rm -rf branching
  324  git branch --list
  325  git checkout shop
  326  ls
  327  rm -rf file1.java  file2.py  file3.sql
  328  git branch s1
  329  git --list
  330  git --list clear
  331  clear
  332  git branch --list
  333  git checkout s1
  334  touch f1 f2
  335  git checkout shop
  336  git branch s1
  337  git checkout s1
  338  ls -la
  339  clear
  340  git checkout shop
  341  git branch s2
  342  git branch --list
  343  git checkout s2
  344  touch f3 f4
  345  ls
  346  git checkoout shop
  347  clear
  348  git checkout shop
  349  ls -la
  350  git checkoout s1
  351  git checkout s1
  352  ls -la
  353  git branch --list
  354  git init
  355  git branch --list
  356  git branch shop
  357  touch 1.java
  358  git branch A
  359  git branch a1
  360  git add .
  361  git commit a s1
  362  git commit a -m "shop1"
  363  git commit  -m "shop1"
  364  git branch s1
  365  git branch --list
  366  git checkout s1
  367  ls
  368  touch 1s.java
  369  git add .
  370  git commit -m "shop2"
  371  ls
  372  git checkout master
  373  ls
  374  git branch s2
  375  git branch --list
  376  git checkout s2
  377  ls
  378  touch s3.java
  379  git add .
  380  git commit -m "shop3"
  381  ls
  382  git checkout master
  383  ls
  384  git checkout s2
  385  git branch toilet
  386  ls
  387  git checkout toilet
  388  touch t.py
  389  git add .
  390  git commit -m "T"
  391  ls
  392  git checkout s2
  393  ls
  394  git checkout toilet
  395  git checkout s2
  396  git merge toilet
  397  git branch --list
  398  git checkout toilet
  399  ls
  400  git checkout s2
  401  ls
  402  git branch -d toilet
  403  git branch --list
  404  git branch -m s2 myshop
  405  git branch --list
  406  git rebase
  407  git checkout s1
  408  git rebase
  409  git rebase s2
  410  git rebase myshop
  411  git branch
  412  git checkout rebase
  413  git checkout myshop
  414  ls
  415  git checkout s1
  416  git .gitignore
  417  touch .gitignore
  418  git status
  419  git add .
  420  git commit -m "ignorecommit"
  421  exit
  422  git init
  423  touch 1.java 1.class .gitignore'
  424  ;
  425  touch 1.java 1.class .gitignore
  426  git status
  427  git add .
  428  git commit -m "l"
  429  git status
  430  git 1.class
  431  git add 1.class
  432  git commit -m "k"
  433  git add .
  434  git commit -m "k"
  435  git init
  436  touch MainFile.java
  437  git add .
  438  git commit -m "MainBranch"
  439  git branch --list
  440  git branch MyShop1
  441  git branch --list
  442  git checkout MyShop1
  443  touch s1.java p1.java
  444  git add .
  445  git commit -m "MyShop1"
  446  clear
  447  git branch --list
  448  git checkout Myshop1
  449  git checkout master
  450  git branch MyShop2
  451  git branch --list
  452  git checkout MyShop2
  453  clear
  454  touch s2.java p2.java
  455  ls
  456  git add .
  457  git commit -m "MyShop2"
  458  ls
  459  git checkout master
  460  cclea
  461  clear
  462  git checkout MyShop1
  463  git branch --list
  464  git branch RestRoom
  465  git branch --list
  466  git checkout RestRoom
  467  touch rt.py
  468  git add .
  469  git commit -m "restbranch"
  470  clear
  471  git branch --list
  472  ls
  473  git checkout MyShop1
  474  git merge RestRoom
  475  ls
  476  git branch --list
  477  git branch -d RestRoom
  478  git branch --list
  479  clear
  480  git checkout master
  481  git checkout MyShop2
  482  git branch --list
  483  git branch food
  484  git branch --list
  485  git branch food
  486  git checkout food
  487  git branch --list
  488  clear
  489  git branch --list
  490  touch veg.py nveg.py
  491  git add .
  492  git commit -m "eating"
  493  ls
  494  git checkout MyShop2
  495  ls
  496  git merge food
  497  clear
  498  git branch -d food
  499  git branch --list
```