Version control:

1. Introduction

[GitHub](https://www.softwaretestinghelp.com/best-source-code-management-tools/#1_GitHub)

[GitLab](https://www.softwaretestinghelp.com/best-source-code-management-tools/#3_GitLab)

[Bitbucket Server](https://www.softwaretestinghelp.com/best-source-code-management-tools/#8_Bitbucket_Server)

SVN

Azure Repo

1. Install git
2. Github account creation
3. What is repository?
4. Configure git

git config --global user.name "viswkum"

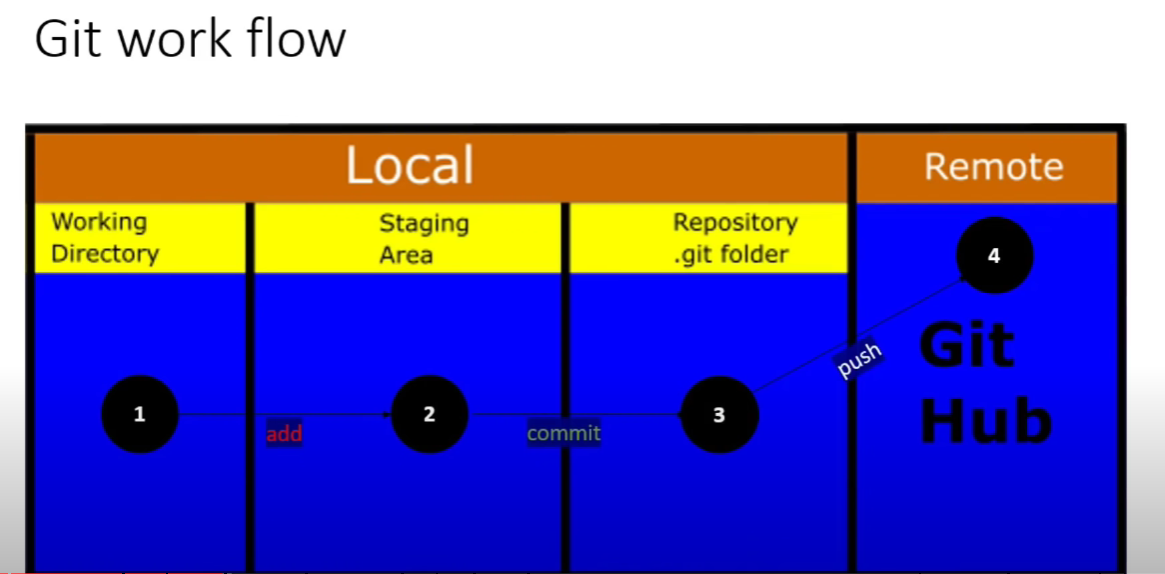
git config --global user.email "vishwanathamkishor@gmail.com"

--global

--system

--local

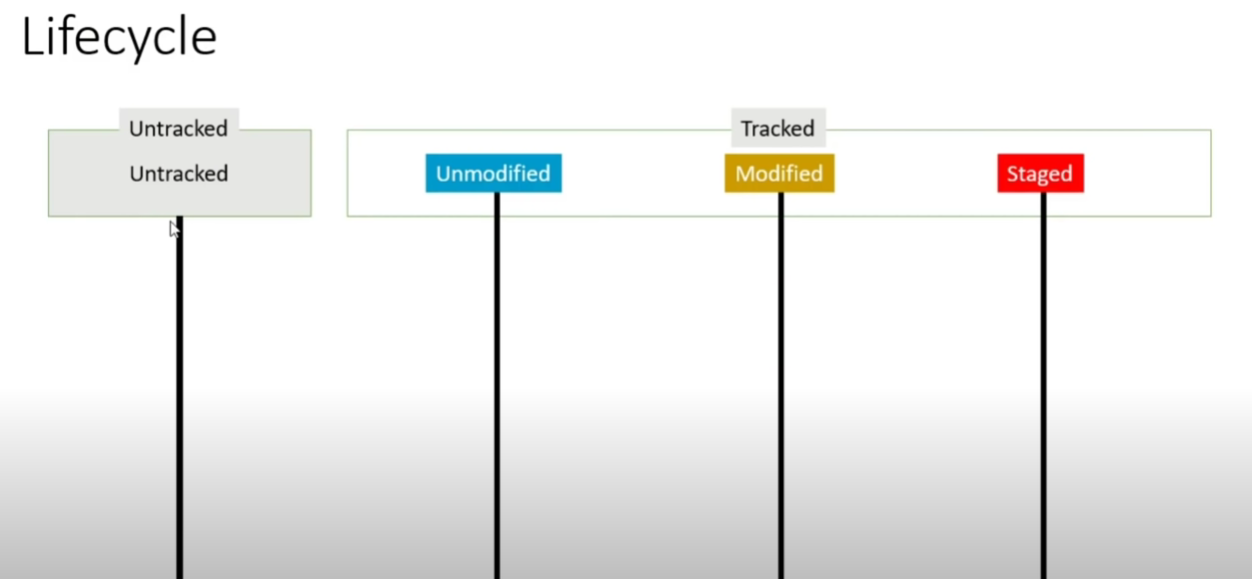
1. **For New Repository:**
2. git init
3. git add README.md or git add .
4. git commit "first commit"
5. git remote add origin https://viswkum....
6. git push origin master
7. Git workflow with examples

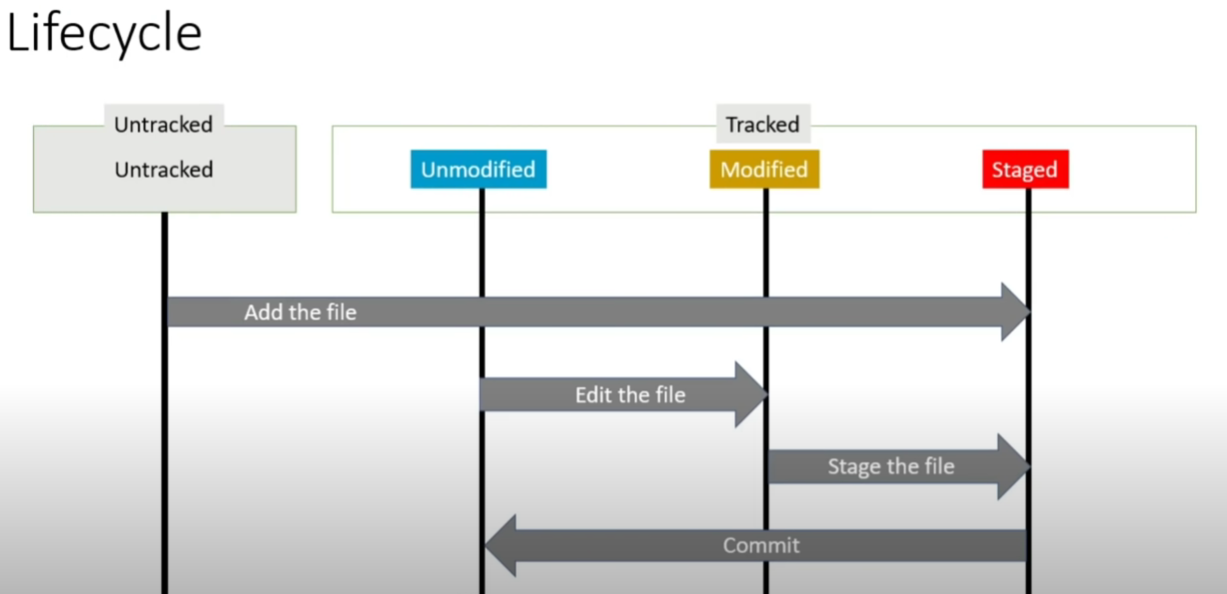


1. Cloning a repository

Ex: git clone https://github.com/BakeryProject/bakery-swap-dish.git

1. Initializing git in existing repository
2. rm -rf .git
3. git init
4. git remote add origin https://viswkum..
5. git push -u origin master
6. Status life cycle

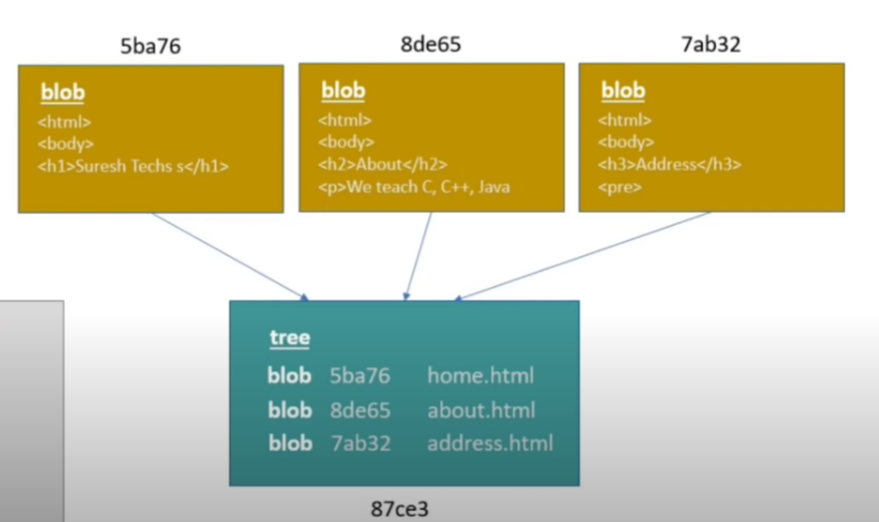


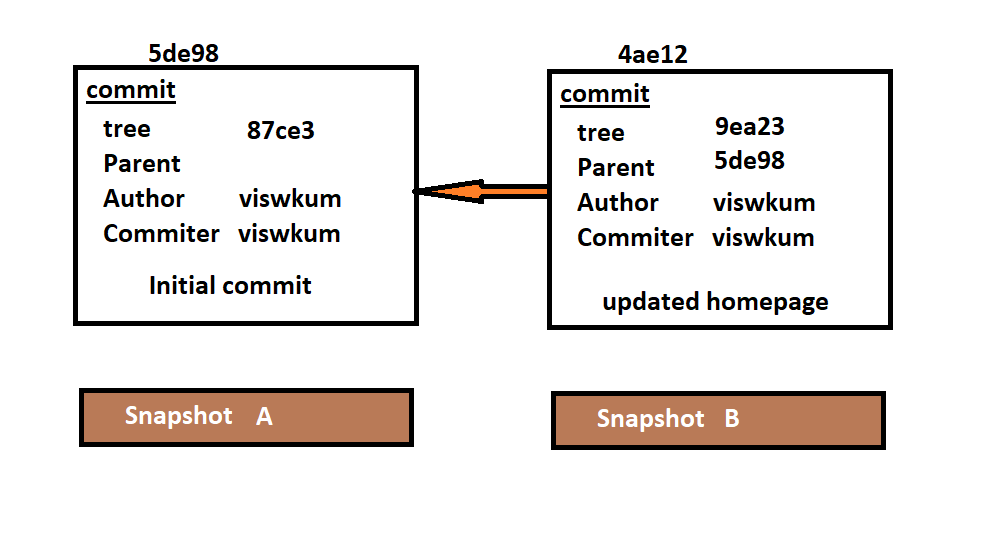


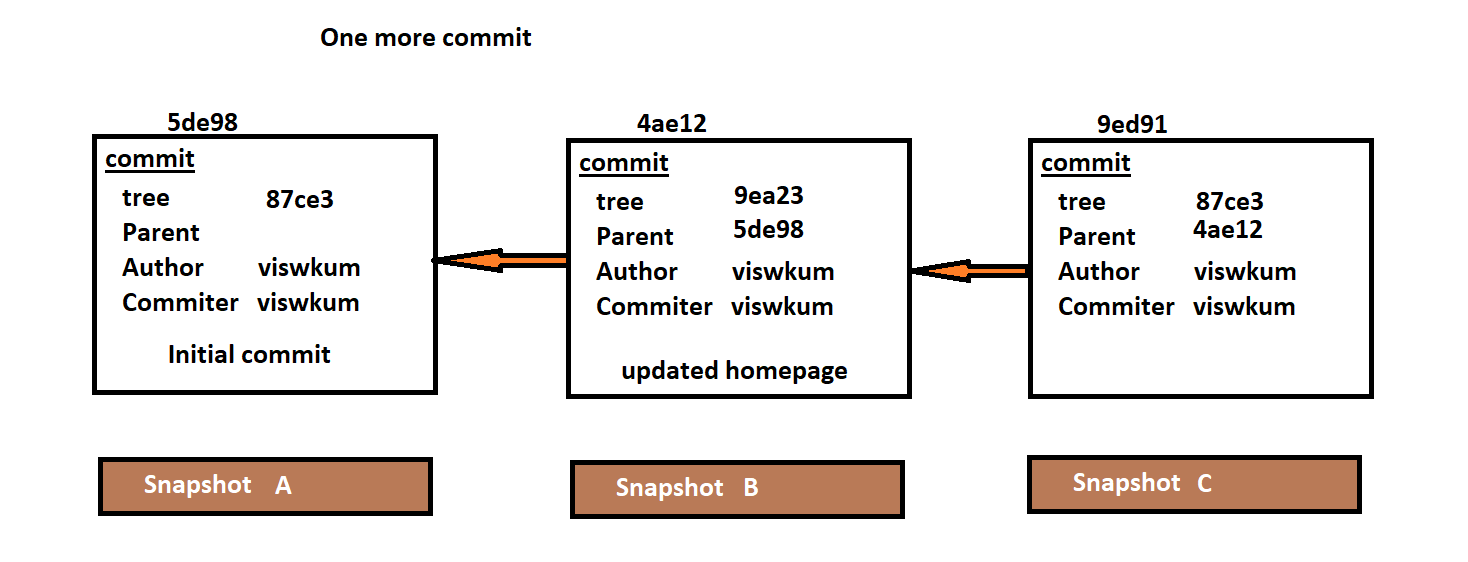
1. Ignoring files(.gitignore)
2. Git Diff
3. Git Commit
4. Git Log(History)
5. Git log ----------------this will give all commits happened in code it will give
6. Git log -p -----------------This will give last 2 changes between commits.
7. Git log -5 or git log -3 ----------This will give that many commits
8. Git log --pretty=oneline
9. Git log --pretty=short (for short information)
10. Git log --pretty=full (full information of all commits)
11. Git log --pretty=fuller

**Git Branching**

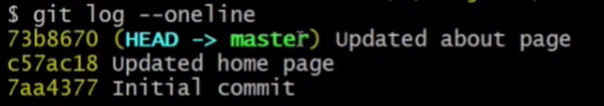
**Home.html, about.html and address.html**



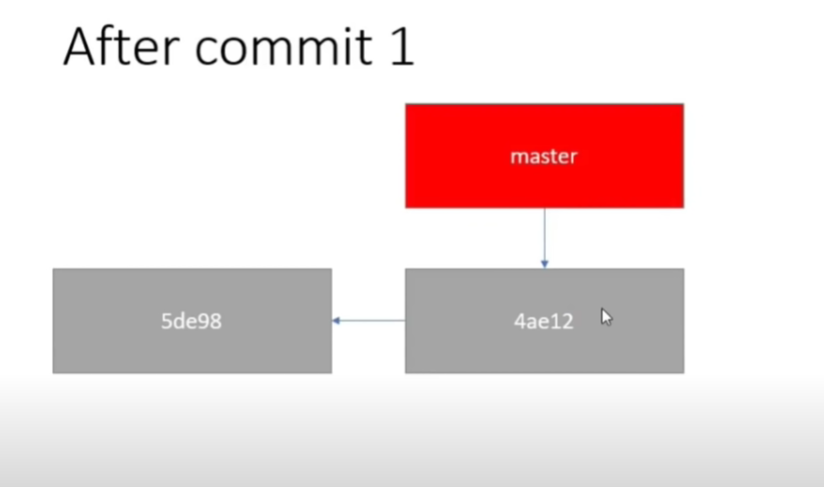


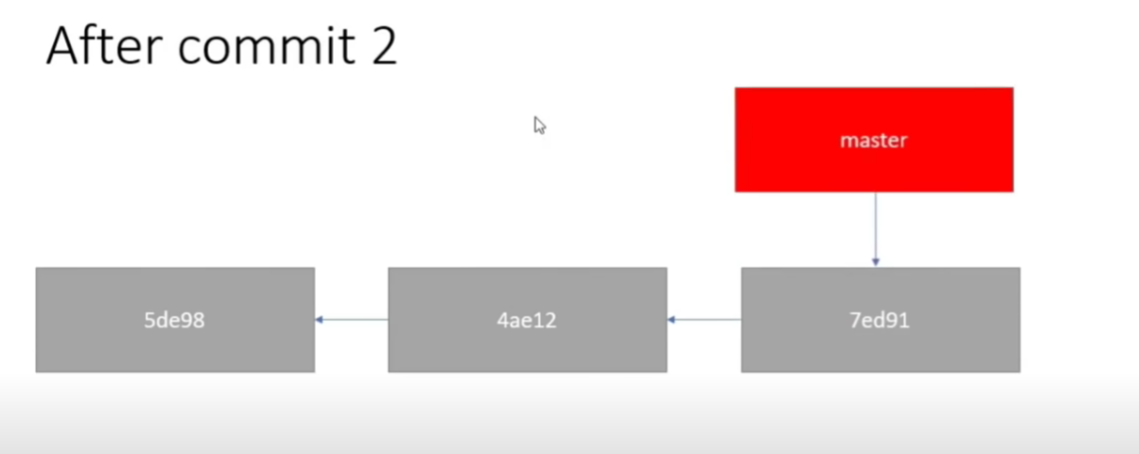


**What is branch?**









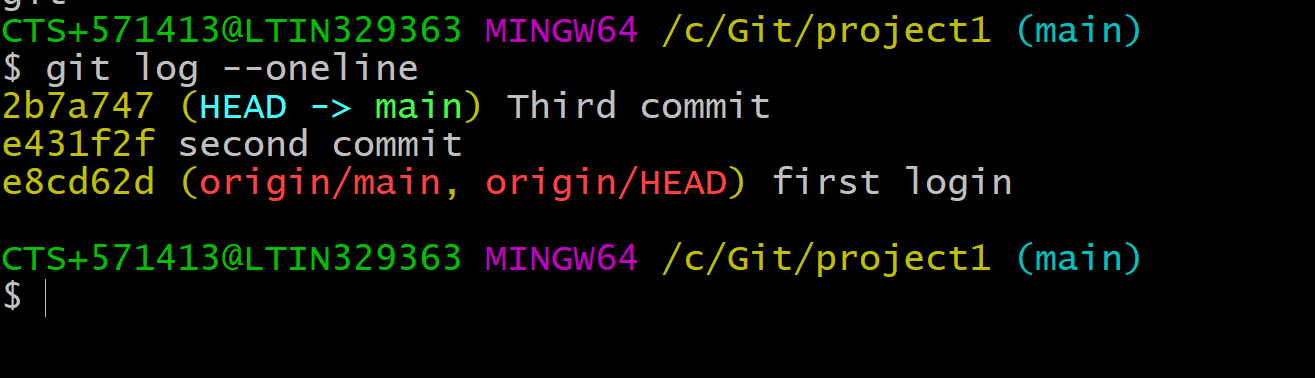
**Creating branch:**

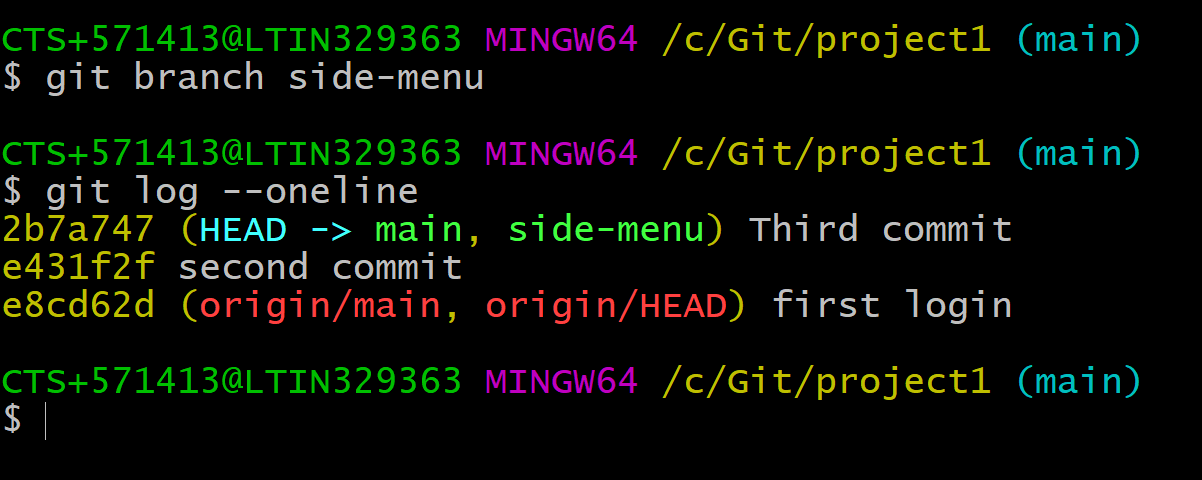
**Git branch branch-name**

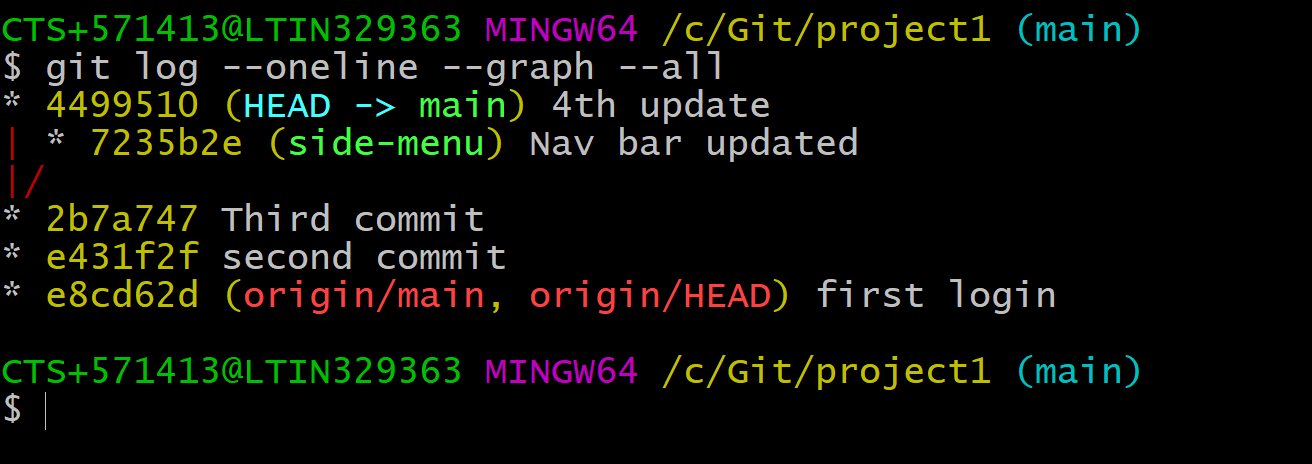
**Git branch side-menu**

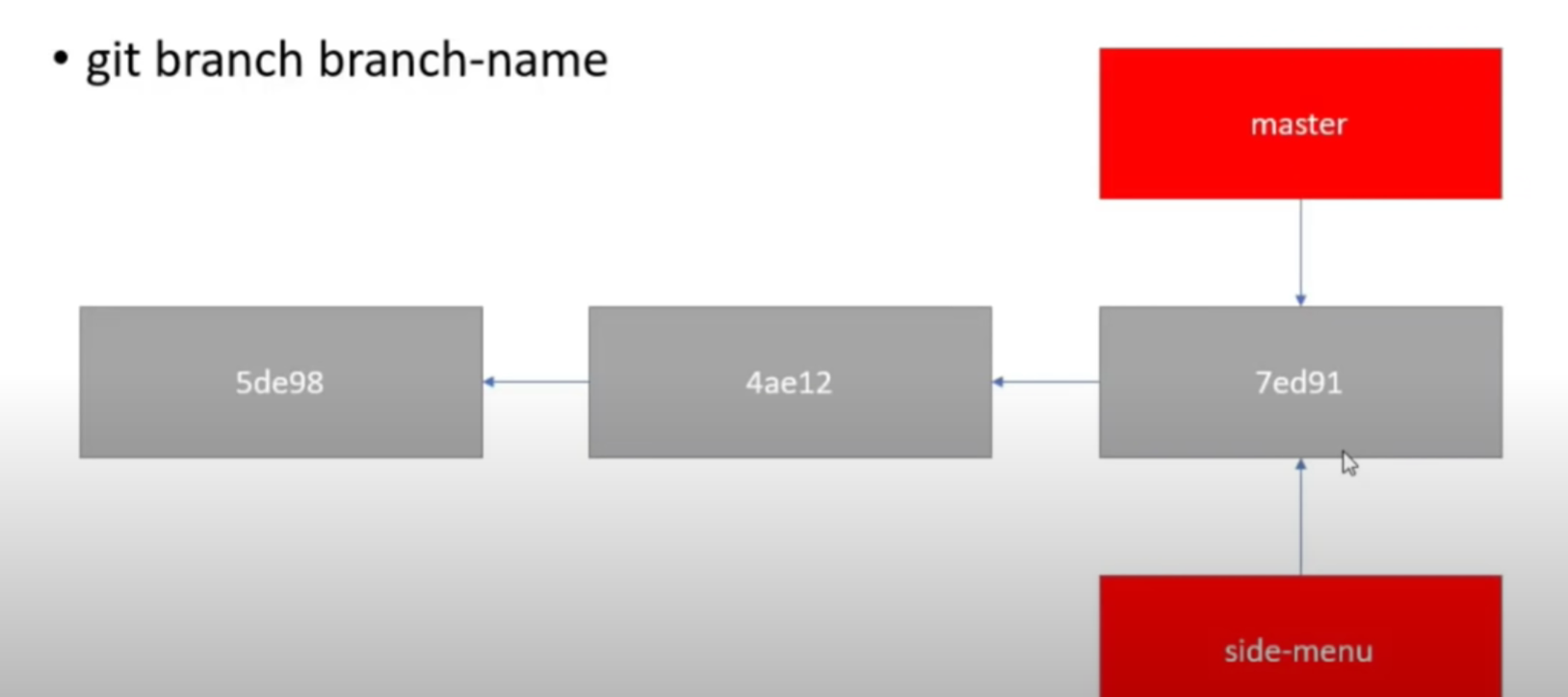
**Git log –oneline**

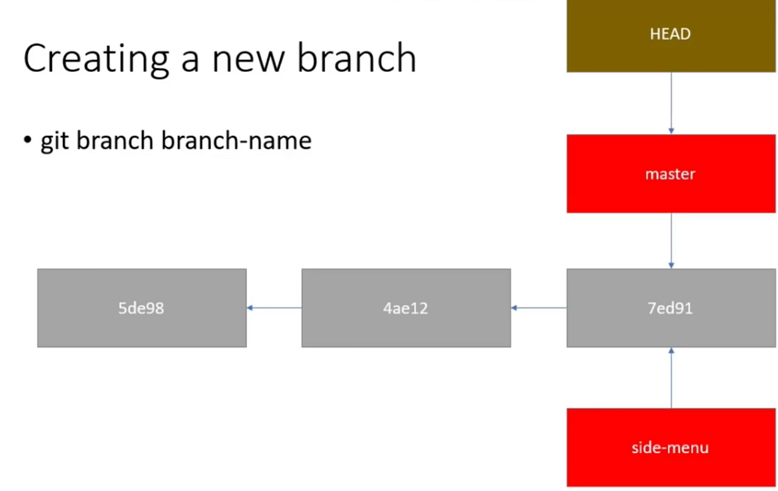
**Git log –oneline --graph**





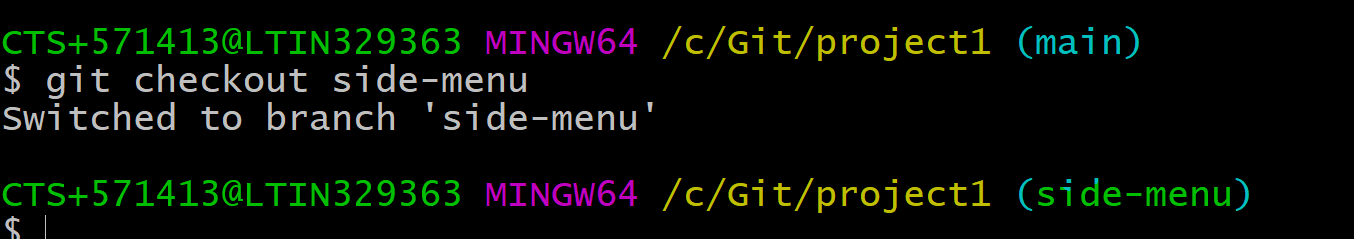






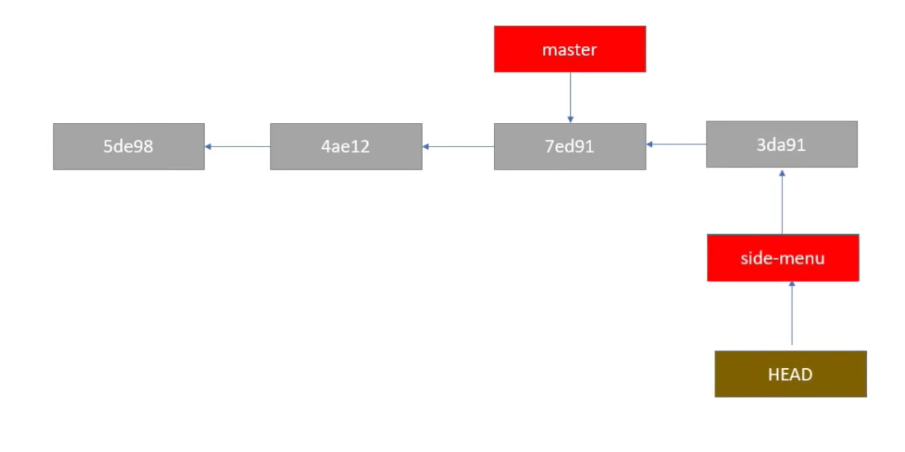
**How to enter to branch now:**

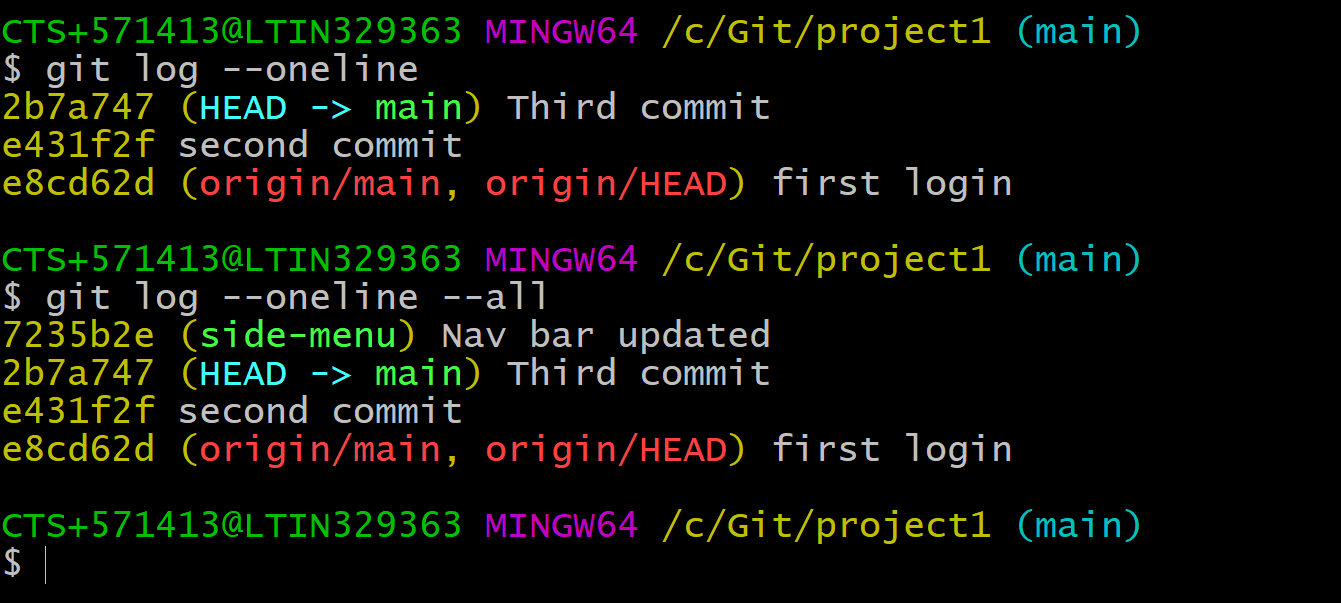
**Git checkout side-menu**

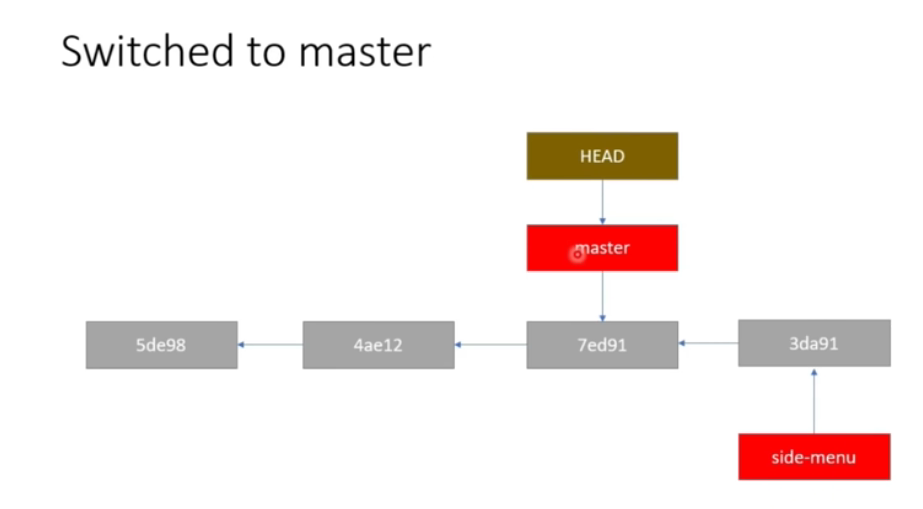












**Git log --oneline --all**

**This command will give total commits information.**

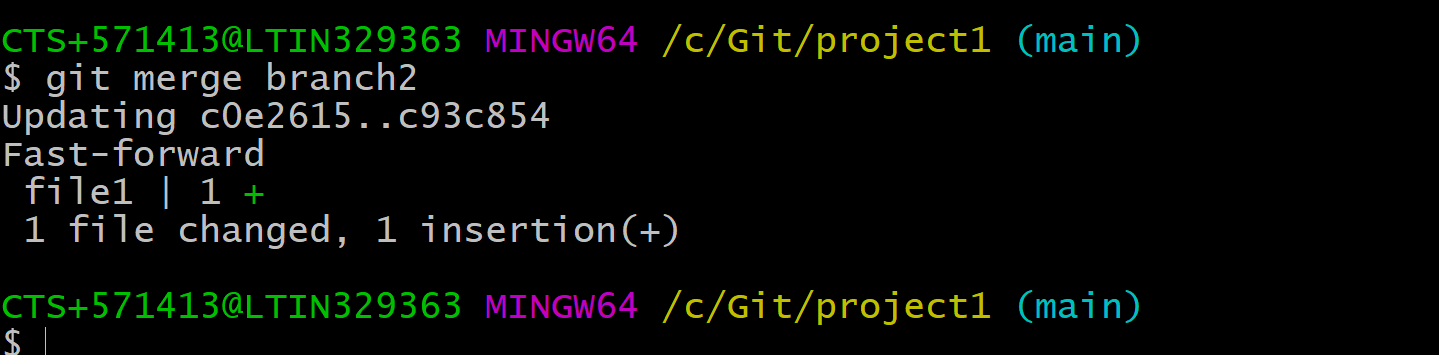
**Git checkout -b newbranch**

**(This will create newbranch and it will switch)**

Merging:

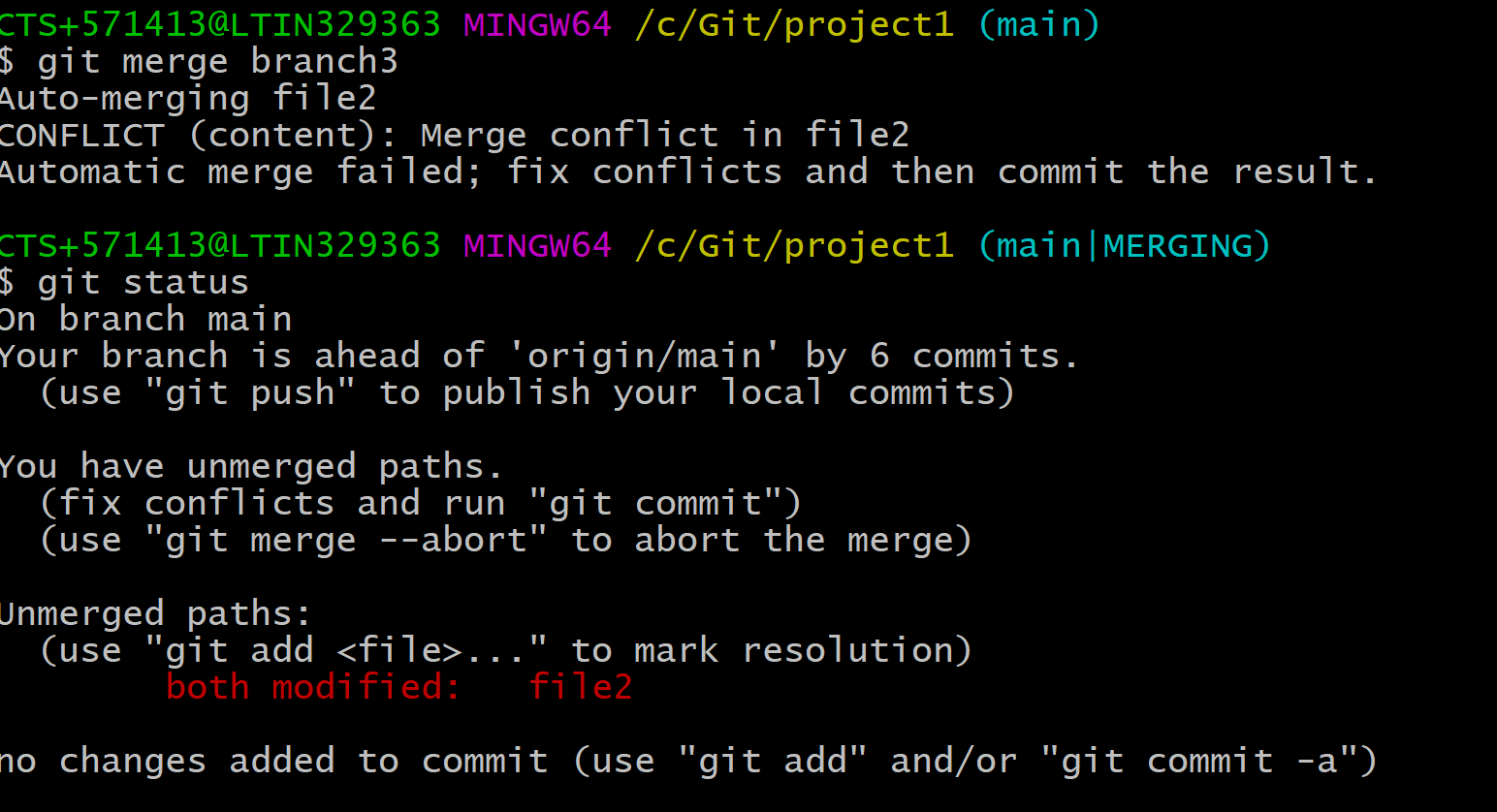
**Git merge branch2**

**(Here branch2 is merging with main session)**





**Merge conflicts:**

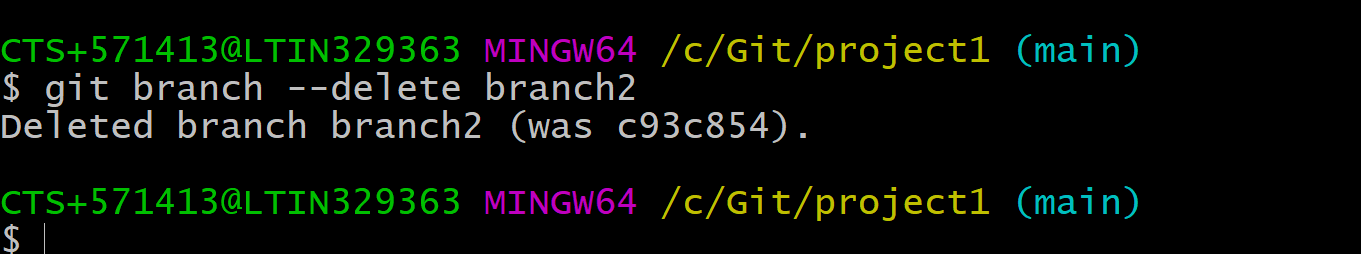


**Deleting branch:**

Git branch -d branch2

Git branch --delete branch2

(Now branch2 will be deleted)



**Fetch /pull:**

**Fetch:**

In fetch first we have to get local and merge

Git fetch origin master

Git merge origin/master

Pull:

Git pull origin master

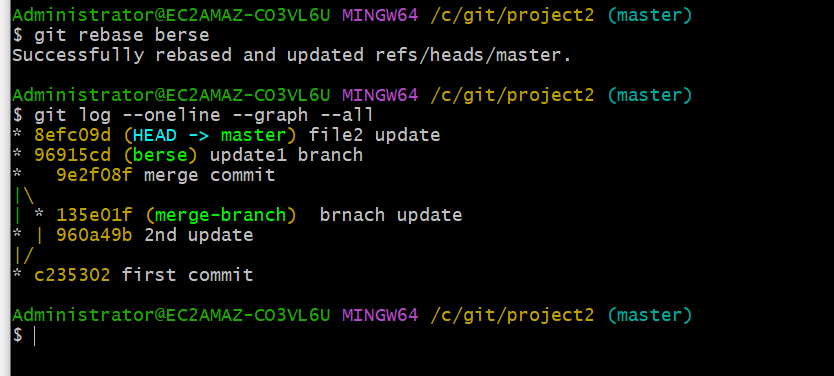
Push:

Git push origin master

**Alias:**

* git config --global alias.allcommits "log --oneline --all --graph"
* Git allcommits

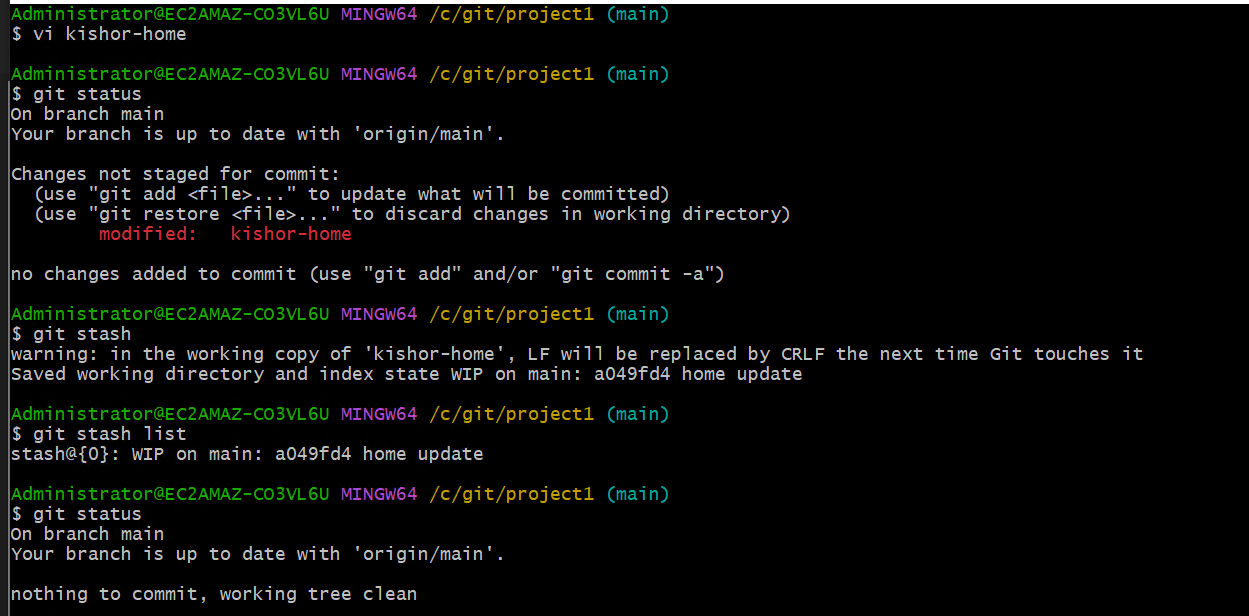
**Rebasing:**



* Git show c7aa43
* Git show commitid
* Git rebase branch2
* Git rebase –continue

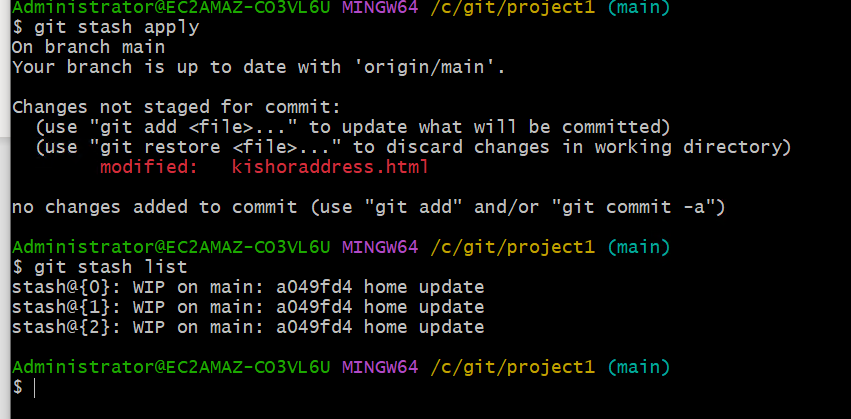
Stashing:

(Recyclebin)



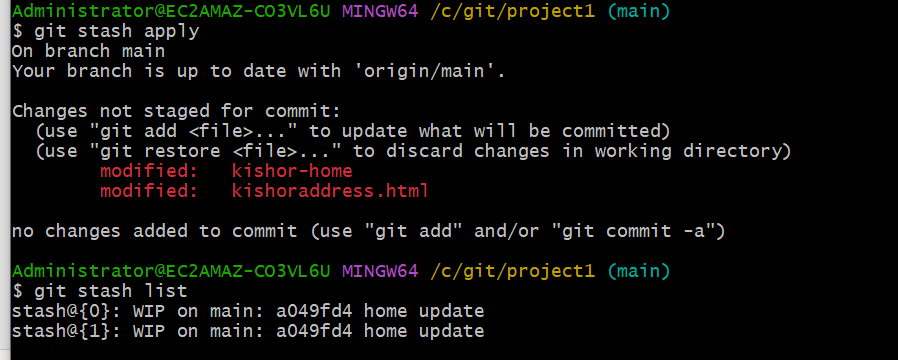
Git stash

Git stash list



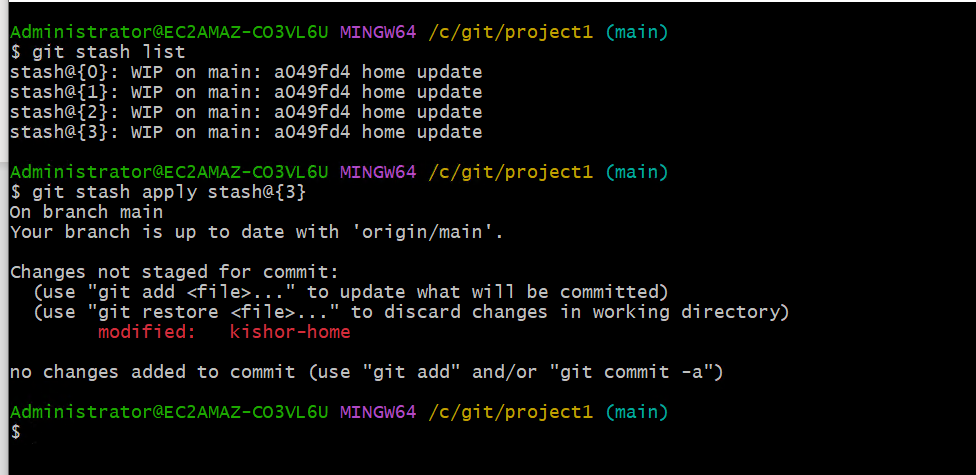
Git stash apply

If we want to delete top level stack should use: git stash drop

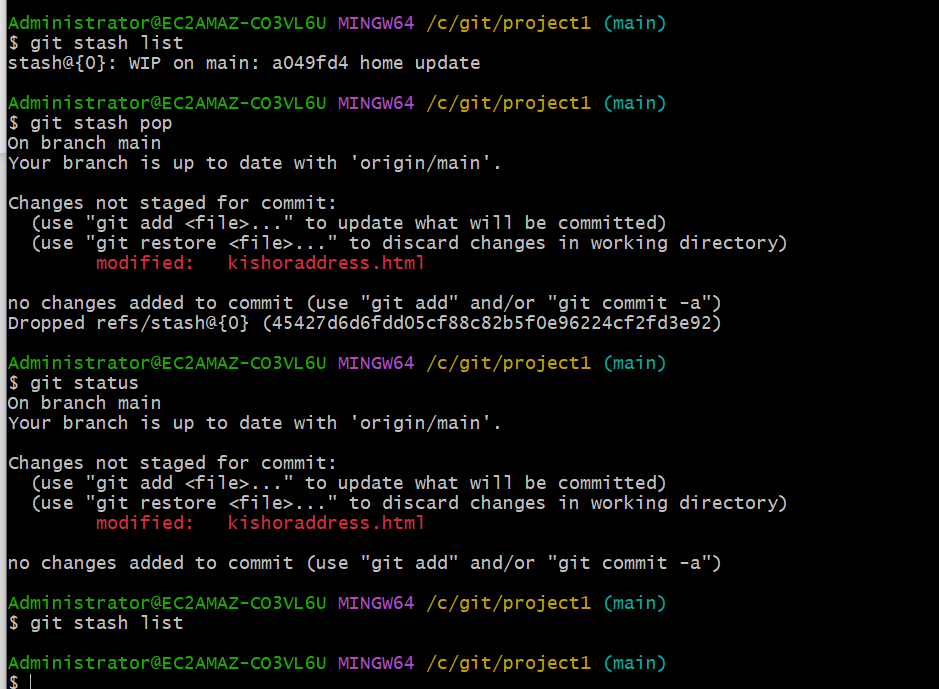


IP we want to apply any specific stash mean 1st one.

Git stash apply stash@{3}



Git stash pop: This command will stash and drop both tasks can perform at same time.

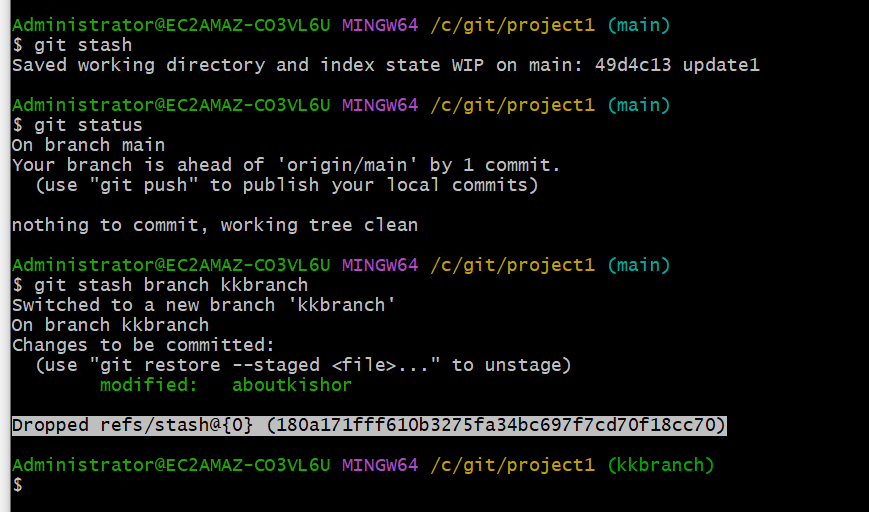


Untracked files; if we want untracked files also to be track git stash -u

**Commands:**

* Git stash or git stash push
* Git stash list
* Git stash apply
* Git stash apply stash@{1}
* Git stash drop
* Git stash pop
* Git stash -u
* Git stash show stash@{1}

Creating new branch with latest stash.



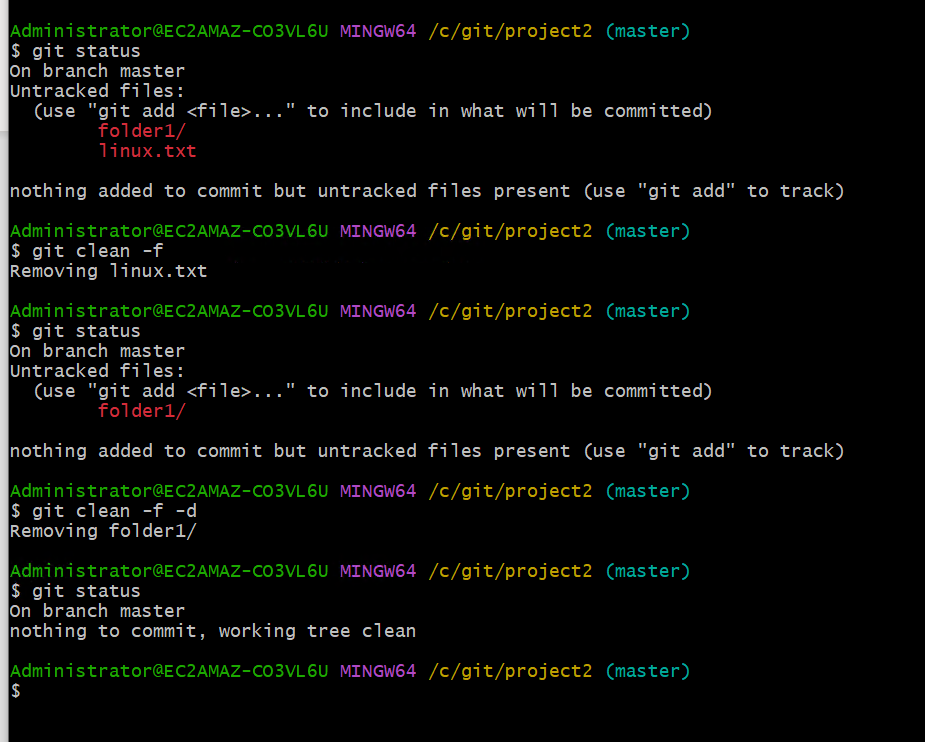
Cleaning:

Git clean -f

This command will delete only untracked files.

Git clean -f -d

This command will delete directory’s as well.



**Tagging:**

* **Git tag**

**(to check all tags)**

* **Git tag tagname**

**To create one specific tag ex: git tag v1.0**

* **Git tag -a tagname (a is** Annotated**)**

**Git tag -a v2.0**

* **Git diff tag1 tag2**

**Check difference between v1.0 and v2.0 chnges**

* **Git tag -a tagname commitid**

**Git tag -a v1.1 c53c66**

* **Git tag -a tagname -f newcommitid**

**Git push origin tagname**

**Only specific tag can be push**

**Git push origin master –tags**

**All tags can be push**