**Git Challenges**

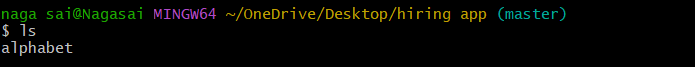
**Resolve Merge Conflicts**

Create a merge conflict intentionally (two users editing the same line).

Resolve the conflict and push the changes.

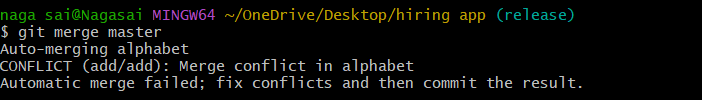
Create the file in your repo and create in another user with the same name of the file .

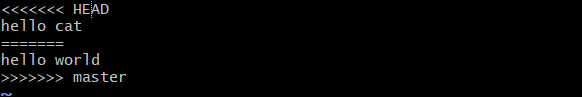
Then merge the repo from one to another repo user the merge conflict occurs because of the same file name and do vi to that file and keep the content you want and save it and add and commit and merge

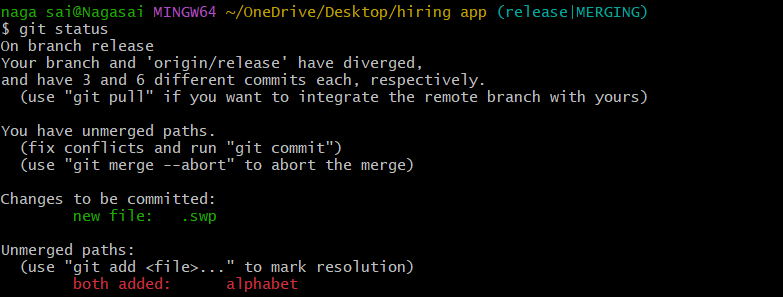


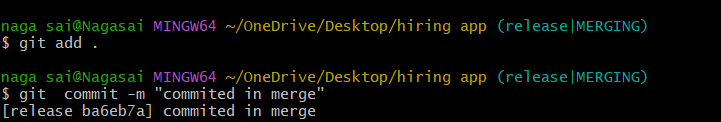


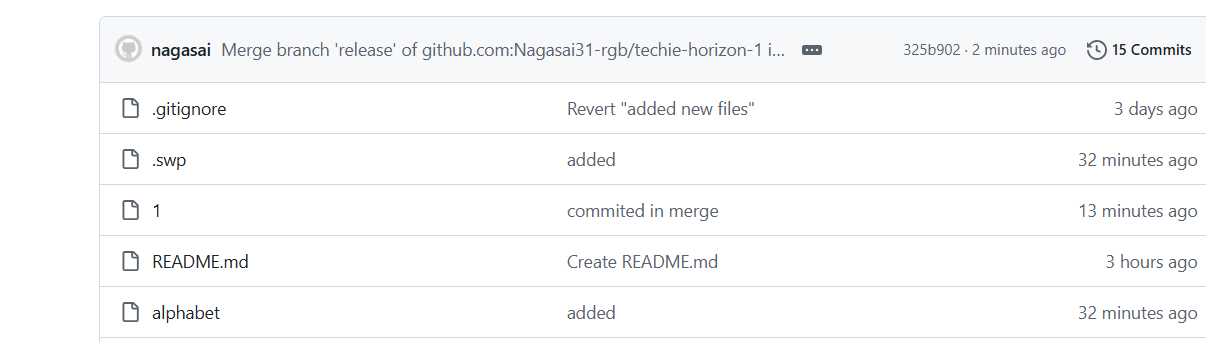
Resolve the conflict and push the changes











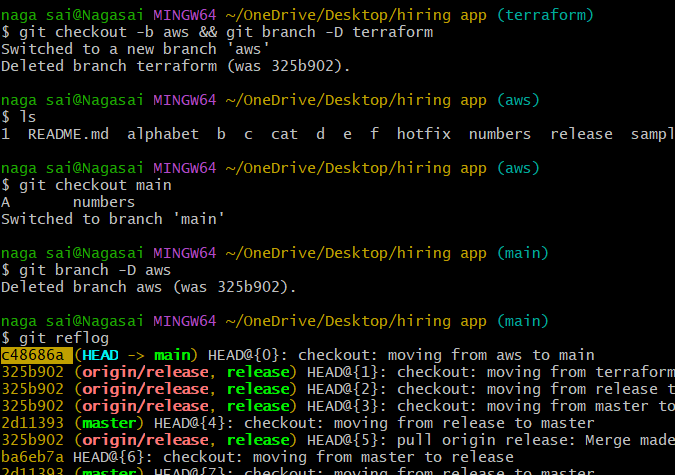
**Recover Deleted Branch**

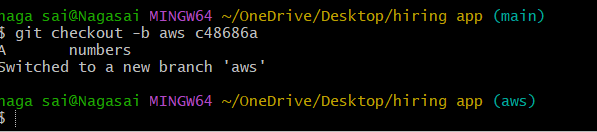
Delete a local branch and then recover it using the reflog.

Git branch -d <branch name> : to delete the branch

Git log : check the log

Git checkout -b release <commitid> : the commit id of deleted branch it will recover the deleted branch





**Undo Wrong Push**

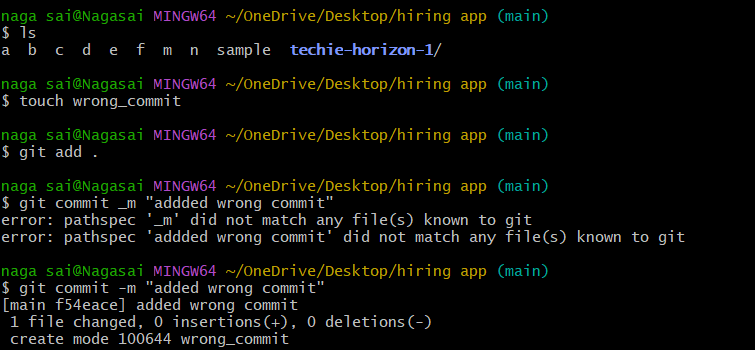
Push a wrong commit to GitHub, then undo it without losing history.

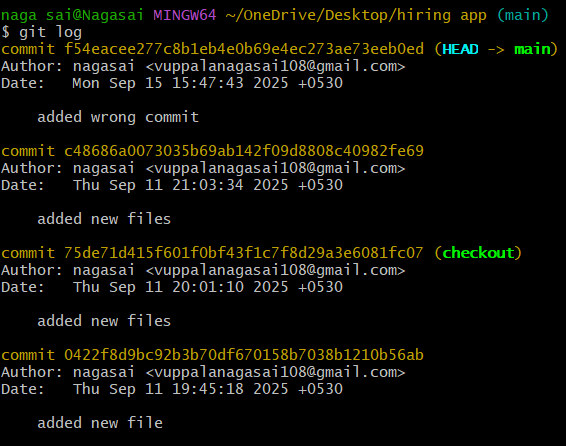
Identify the bad commit

Use to find the commit hash you want to undo.

Revert the commit

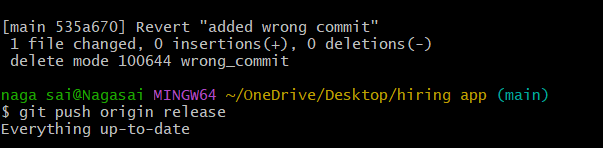
This creates a new commit that undoes the changes





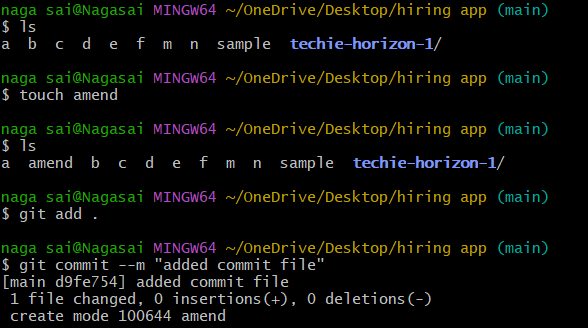
Git revert -commit id - This creates a new commit that undoes the changes.

Now push the new "undo" commit to your remote branch.: git push origin <branch-name



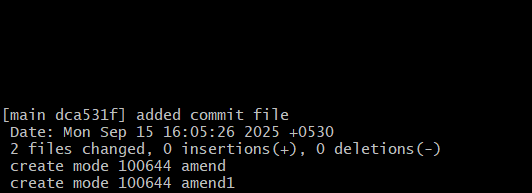
**Amend a Commit**

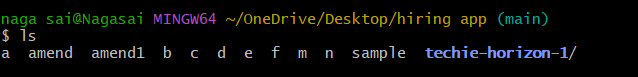
Make a commit, then add a missing file to it using git commit --amend.



Also add a second file name amend1

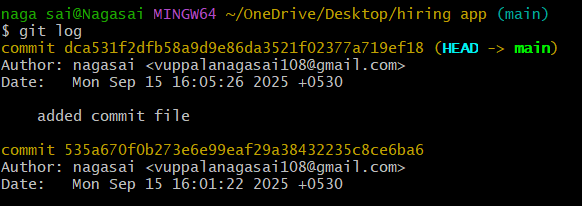
git commit –amend: Modifies the last commit (can change the commit message or add changes)

This will open the editor and edit the message. 



**Cherry-pick a Commit**

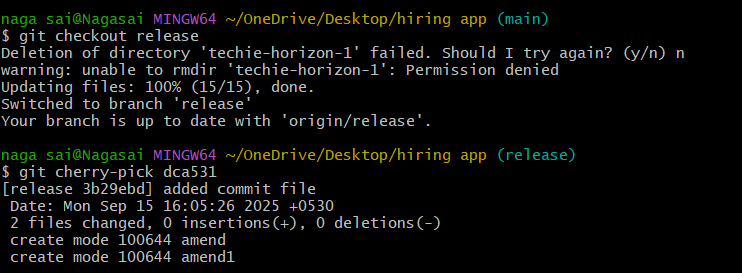
Take a specific commit from one branch and apply it to another branch.

git log --oneline (to check the commit id and copy commit id) 

switch to other branch and enter

git cherry-pick (commit \_id)

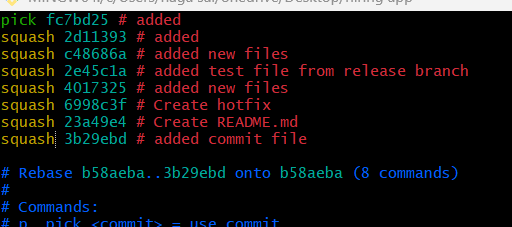
git log --oneline (hence you can get the spicifc id in other branch)

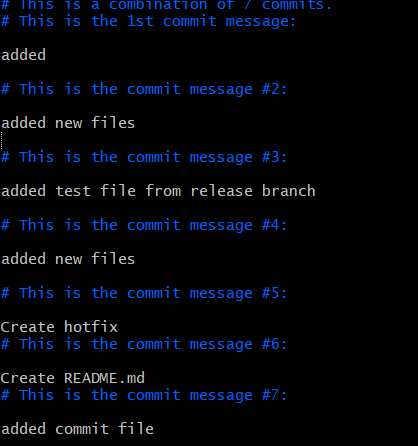


**Interactive Rebase**

Reorder and squash multiple commits into a single clean commit

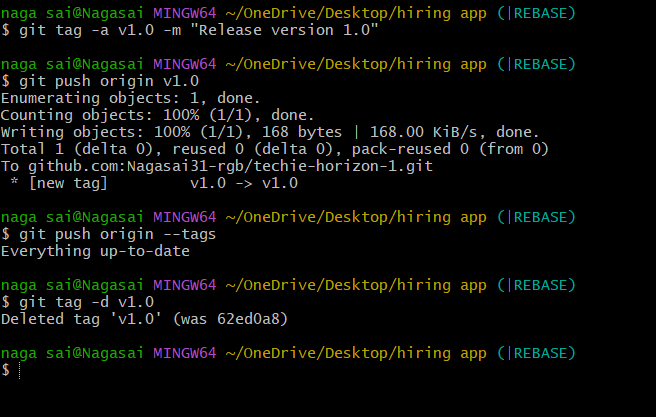


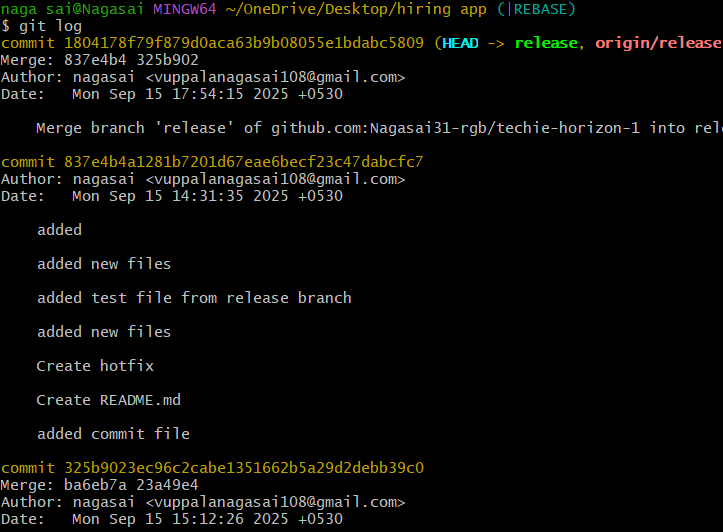




**Tagging & Release**

Create a version tag (v1.0), push it to GitHub, then delete and restore it.



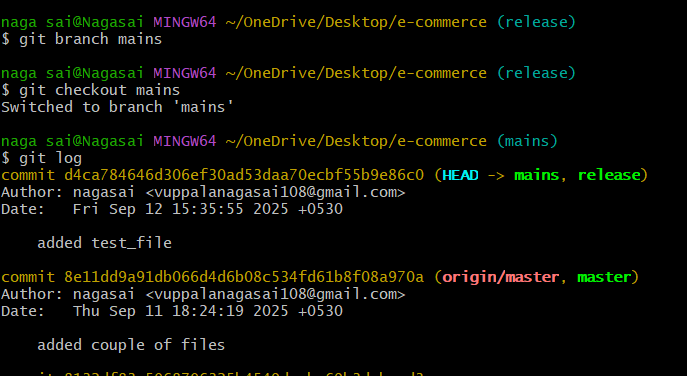




**Detached HEAD Challenge**

Checkout a specific commit (detached HEAD state) and create a new branch from it.

git log --oneline

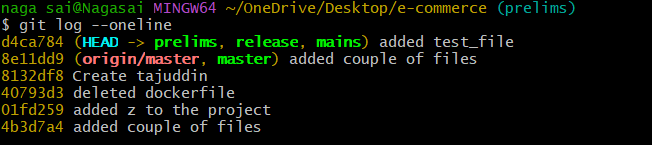


git checkout commit \_id ( it will show the head at commit\_id other are detached as wll it will show a commit id in place of branch)



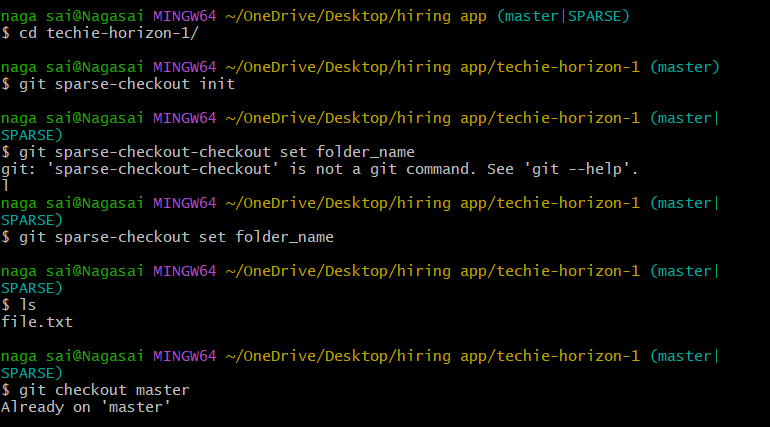
git log --oneline (it will show logs only before our given commit\_id)

git checkout commit\_id command enables us to create a branch till the specific commit without cloning the whole branch



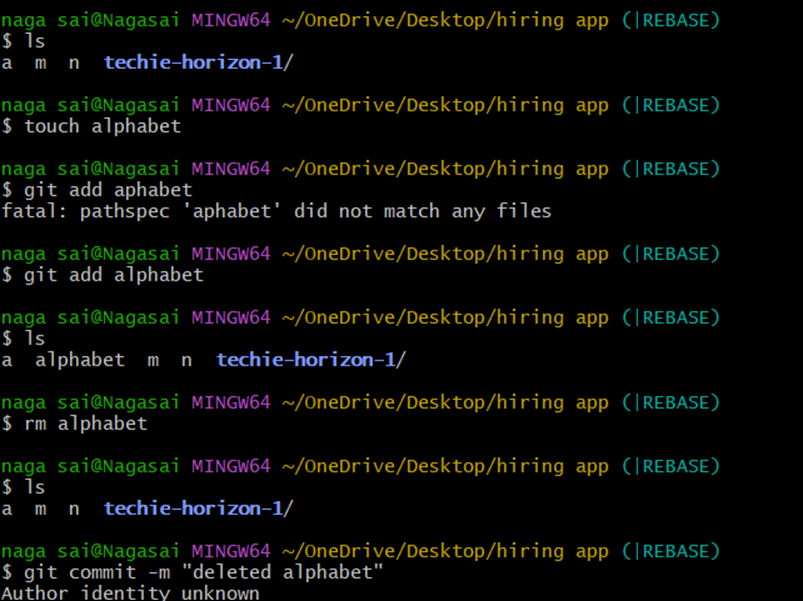
**Clone with Sparse Checkout**

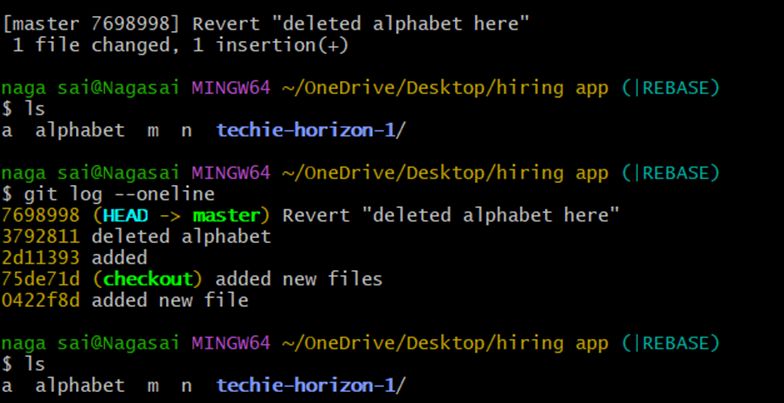
Clone only a subdirectory of a repo using sparse checkout



**Reset vs Revert Challenge**

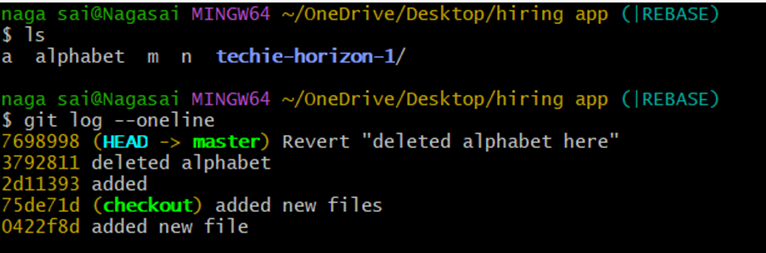
Demonstrate the difference between git reset --hard and git revert in a repo.

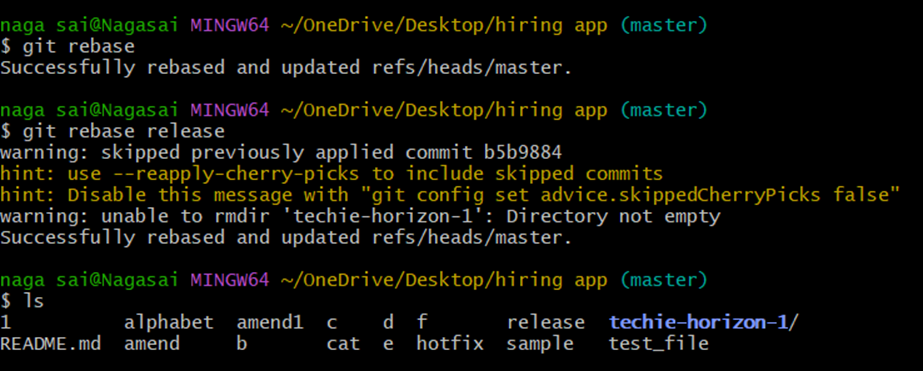




**Squash Merge vs Rebase Merge**

Show the difference between squash merge and rebase merge with evidence





**Fork & Pull Request Workflow**

Fork a repo, make a change, and submit a pull request to the original repo.

to make a fork request you need to add the collaborators

then click on fork on right side of your github account and add any repo of your collaborator acoount which is public.If it's private then we need to make a request and he needs to accept the request for the repo under fork requests.

fork request helps in merging one repo from one account to another account .

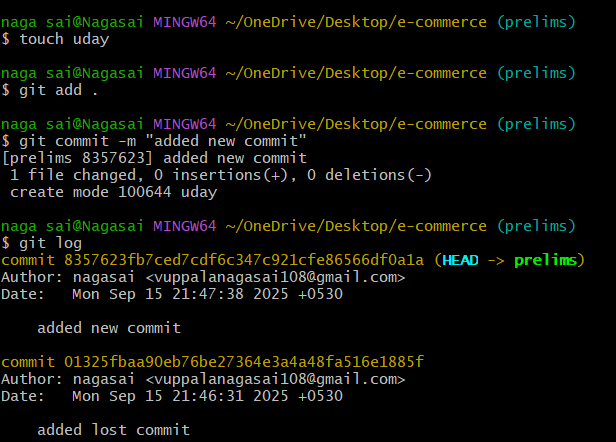
pull request helps in merging the branches on git hub

**Recover Lost Commit**

Commit something, reset hard, and then recover it using git reflog.

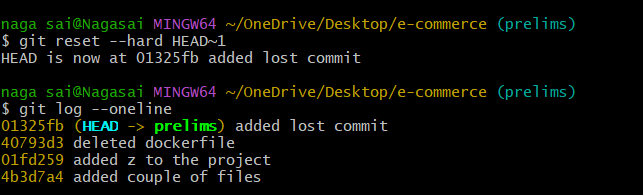


git log --oneline (check all logs and copy the commit id till where you want to reset the account



git reset --hard commit\_id (paste the commit id )

git log --oneline (now you will find all commits is recovered)



Git reflog-recover by using this command.

