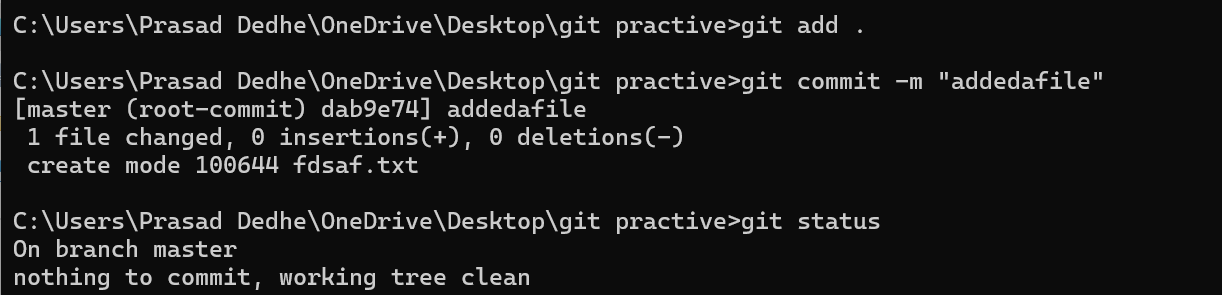
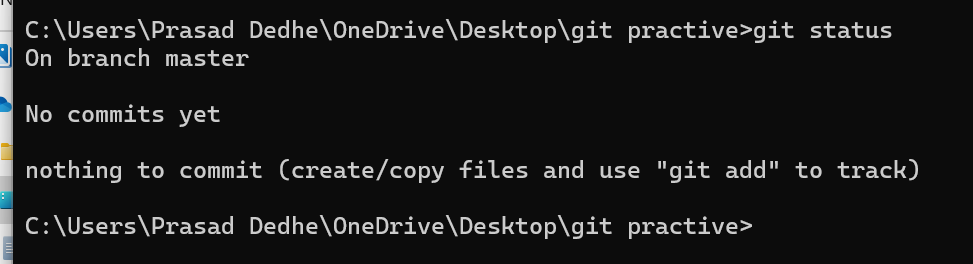
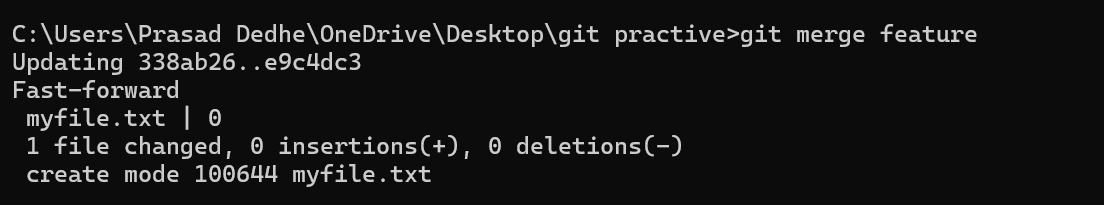
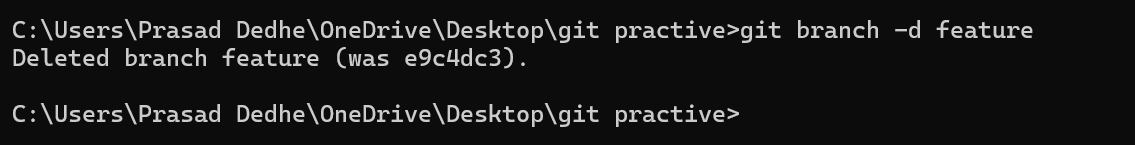
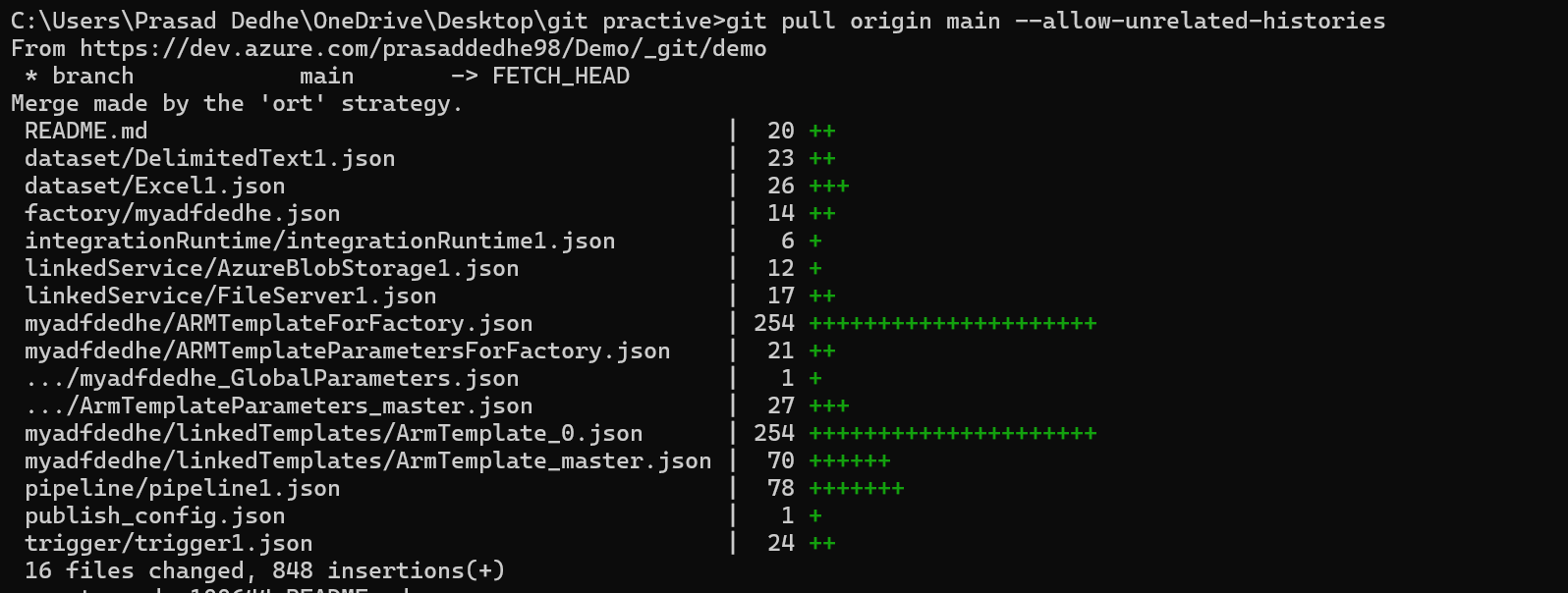
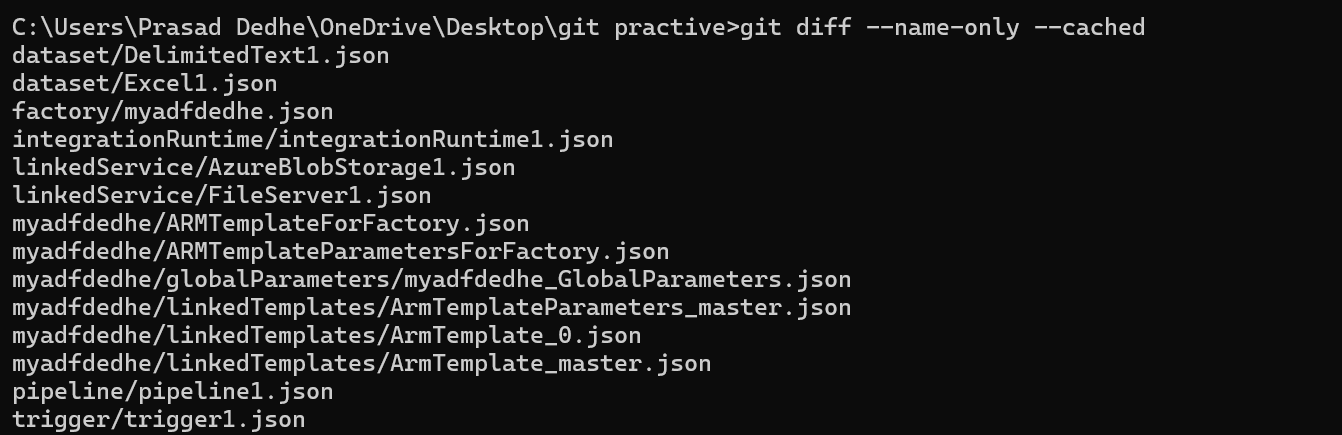
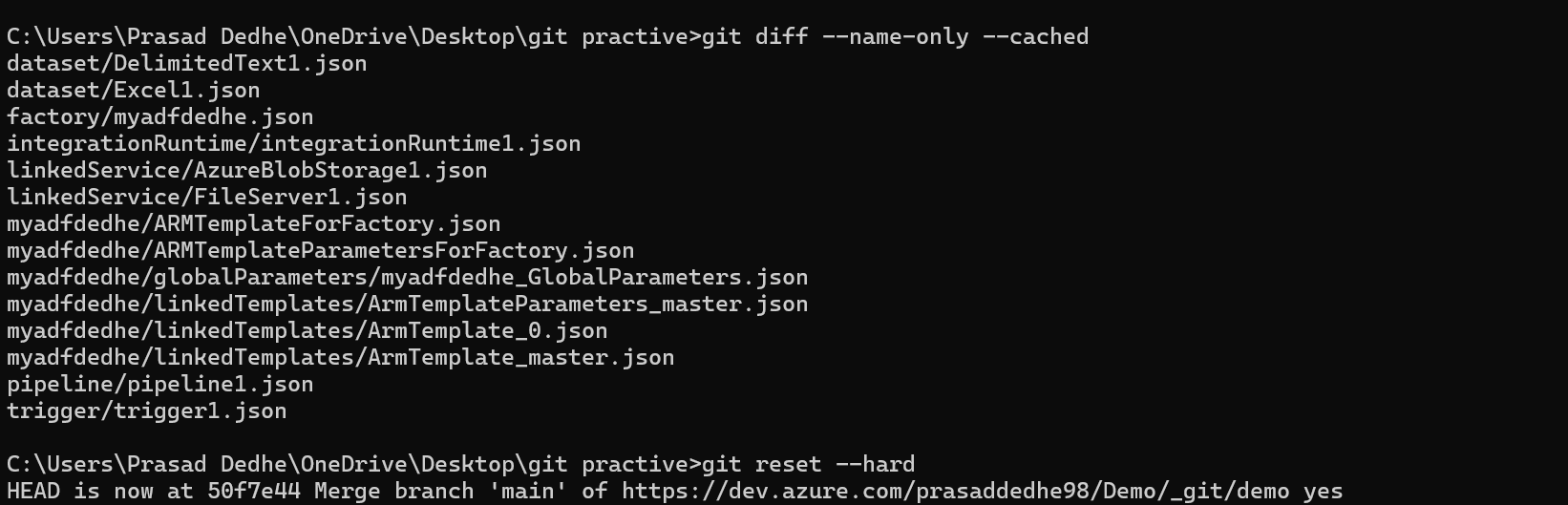
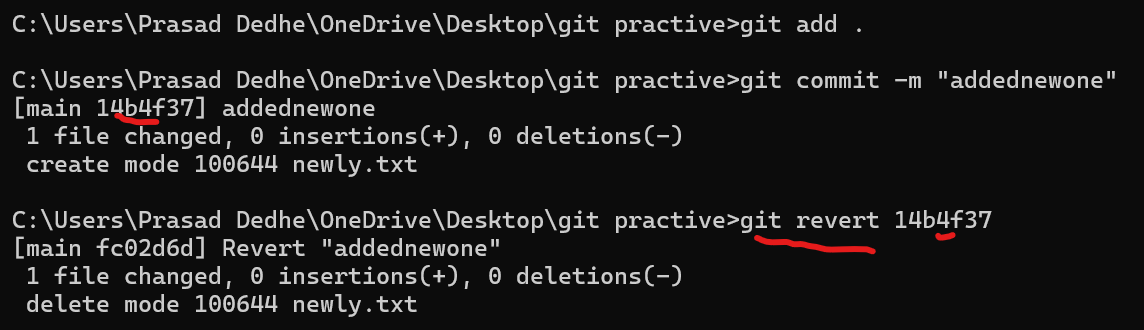
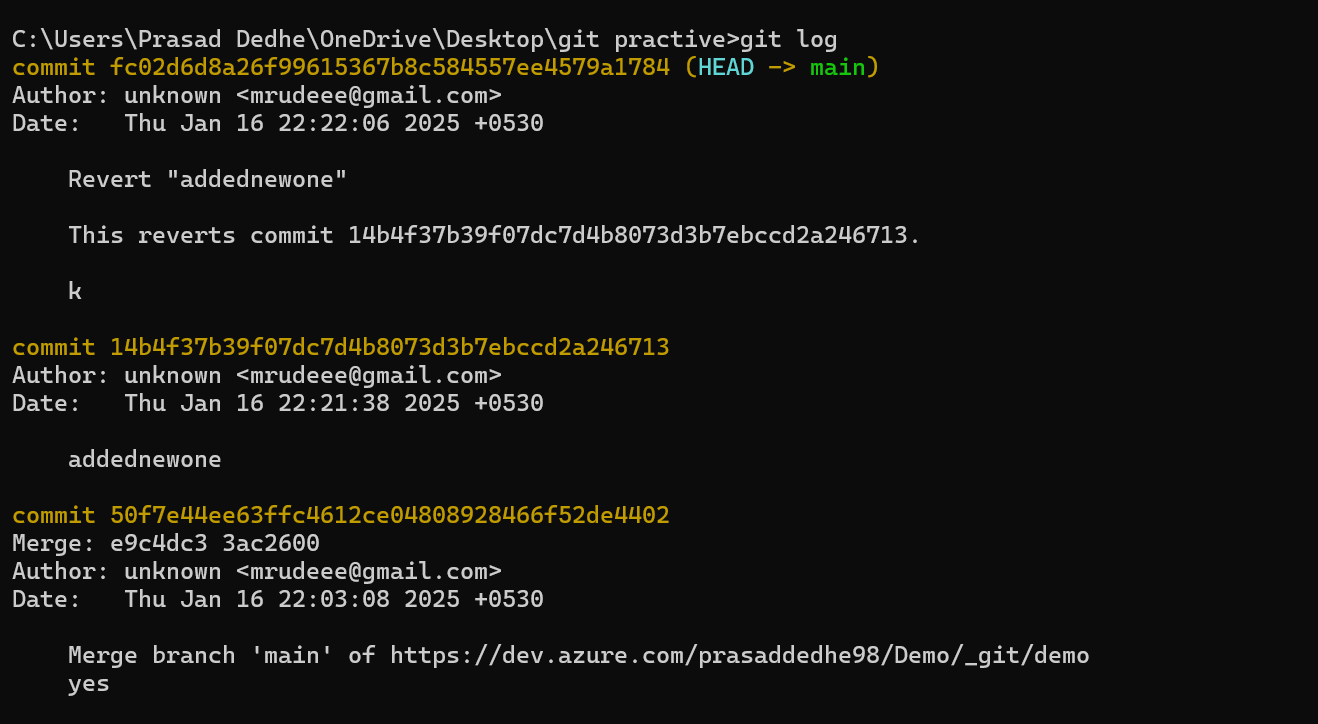
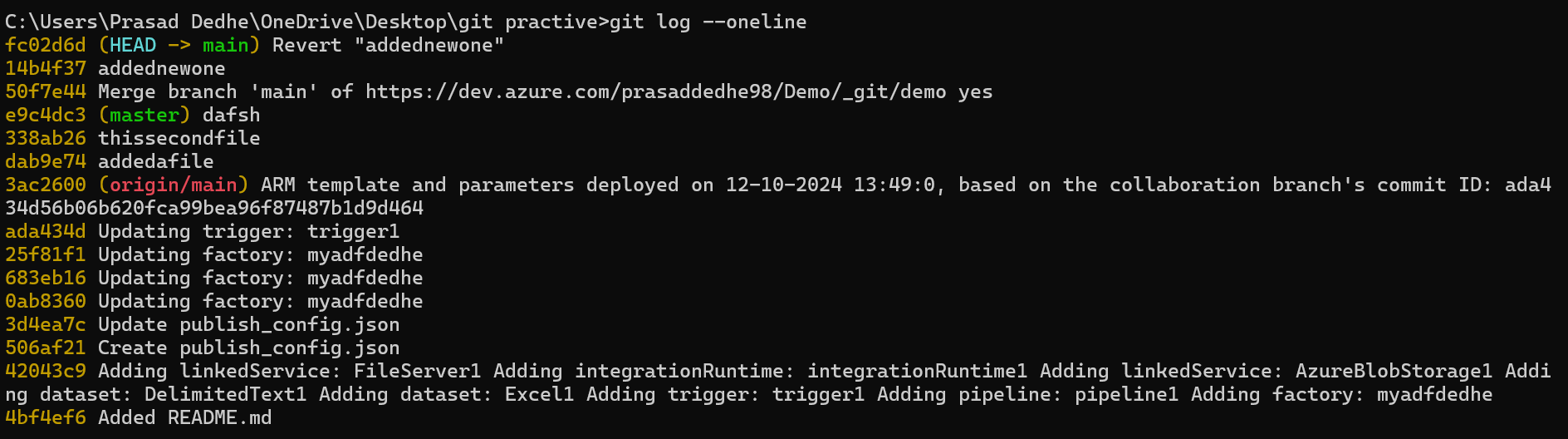
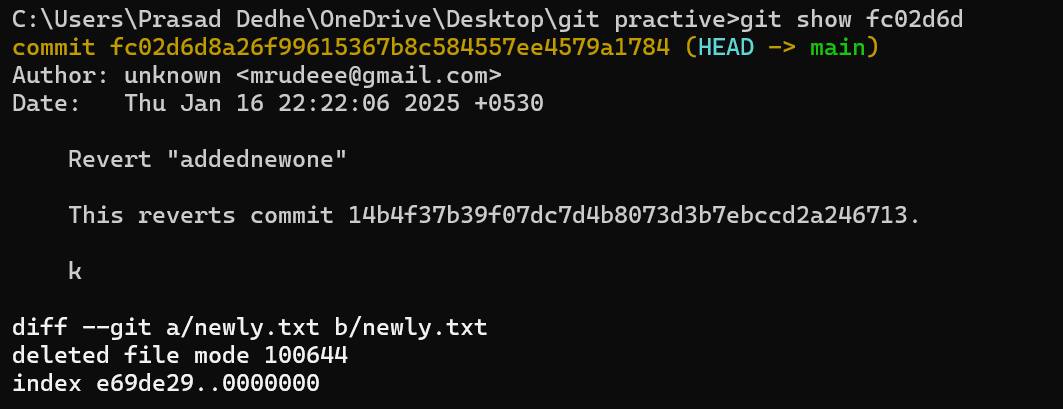
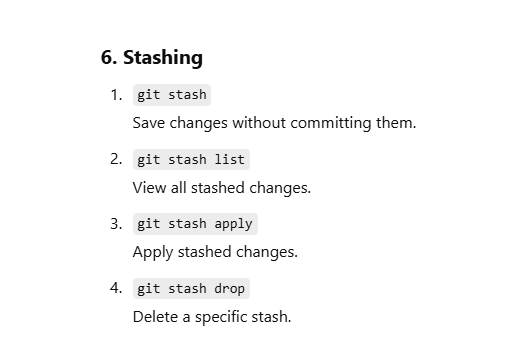
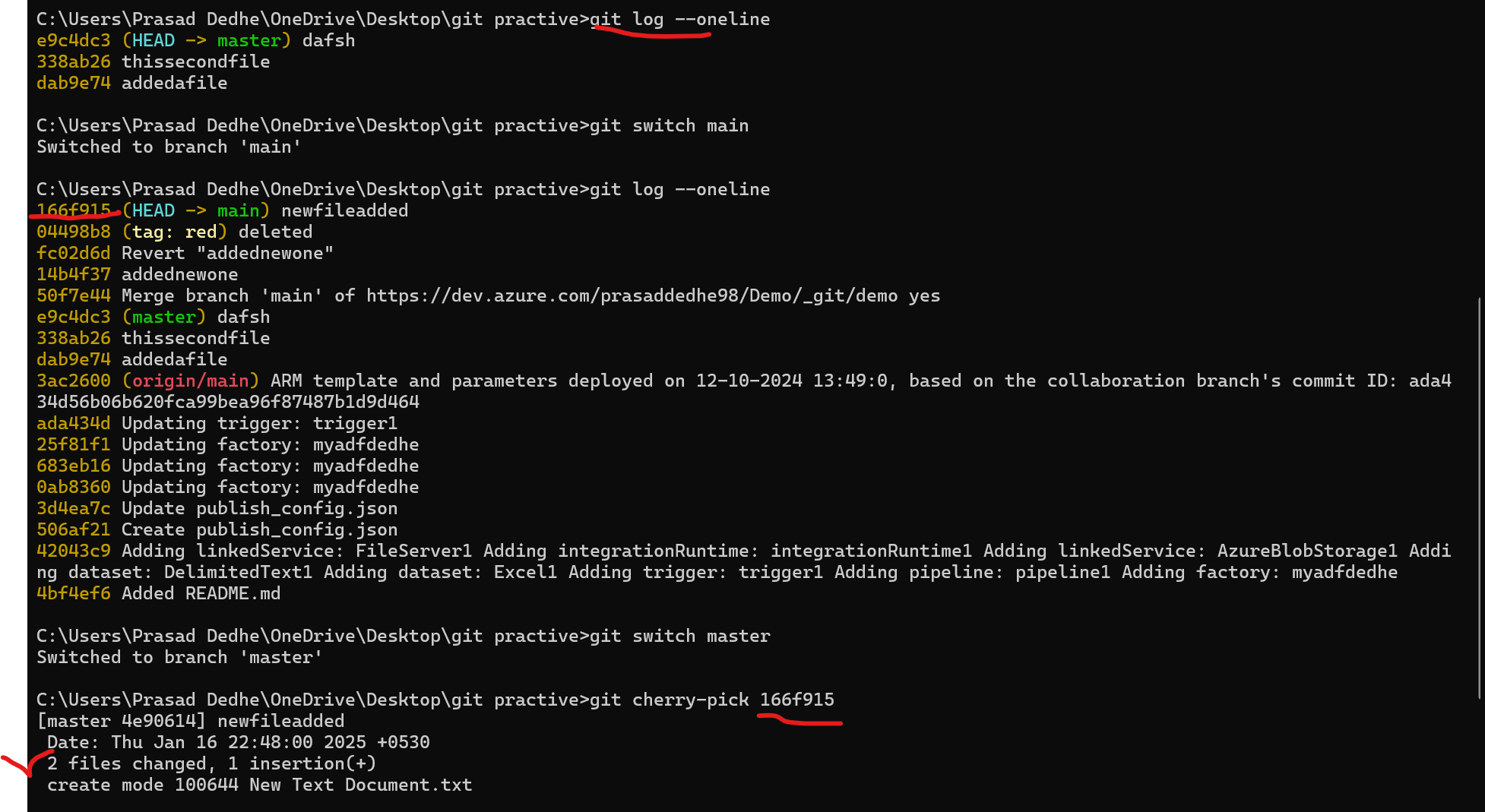
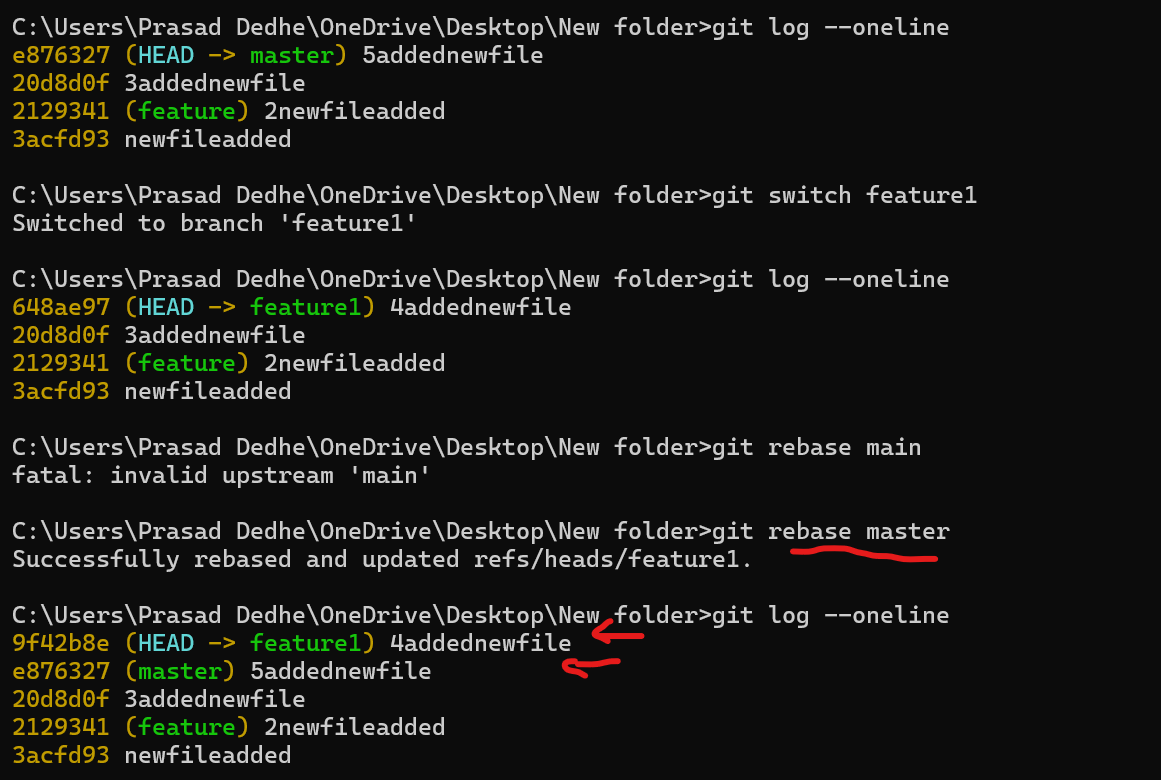
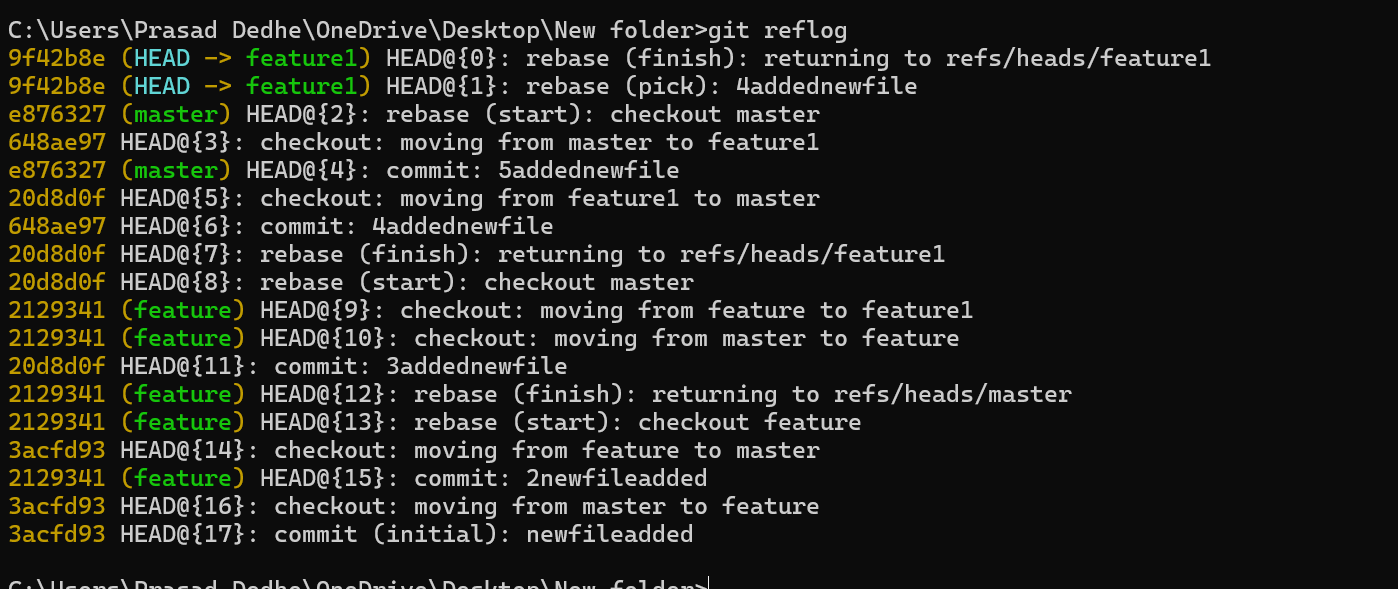
**Git commands**

1. Git status: 
2. Git merge (execute on master branch with the branch name you want to merge with): 
3. Delete a branch: 
4. Git fetch: Finds the difference between the local and remote repo.
5. Git Pull: Fetches & merges (basically clones) the remote repo into your local branch. 
6. To check which files are currently in staging area: 
7. Git reset <file>: Removes the added files from staging area to dev area, but keeps the changes in dev directory.
8. Git reset –hard: Removes the added files from staging area to dev area, and also reverts the changes in dev directory. 
9. git revert <commit-hash>: Undo a commit by creating a new commit 
10. git log: View commit history 
11. git log –oneline 🡪 compact commit history 
12. git show <commit-hash> 🡪View detailed information about a specific commit. 
13. git stash:



1. git cherry-pick <commit-hash> 🡪 Apply a specific commit to the current branch. 
2. Git rebase 🡪 basically lets say, if you have main branch and you created a new branch feature1 branch. After that you added newfile4.txt in feature1 branch. Post which you added, newfile5.txt in main brach. Now you switch back to feature1 branch. But now you want to add newfile5 in feature1 branch, and that should happen before the newfile4 should have happened. In that case you can use rebase. (note the same thing can be achieved via merge as well. The difference between the two is, merge creates a merge commit history while, git rebase creates a linear history) (I know its difficult to understand, just remember if you want to make sure every while in main is in feature then use rebase, simple)
3. Git reflog 🡪 nothing but advanced, git log (which tells HEAD postion and all) 
4. The other most common commands that I use is
   1. git remote add origin <repository-url>
   2. git push -u origin <branch-name>
   3. git checkout -b <branch-name> 🡪 creates new branch
   4. git branch 🡪 list all branches
   5. git switch <branch-name> 🡪 switches to the branch
   6. git add .
   7. git commit -m "message"
   8. git commit –amend 🡪 modify the last commit message
   9. git init
   10. git clone <repository-url>