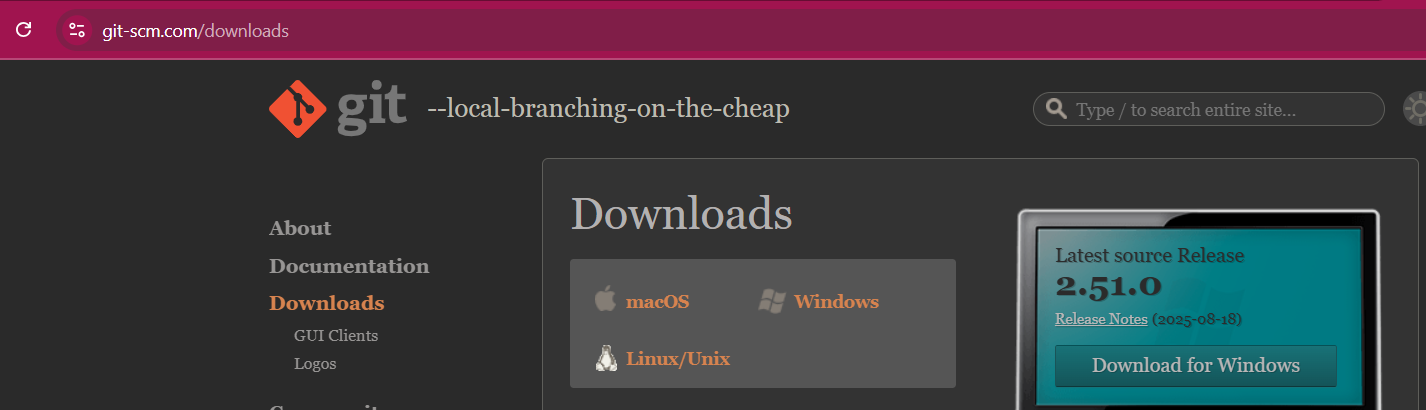
1. Install git.

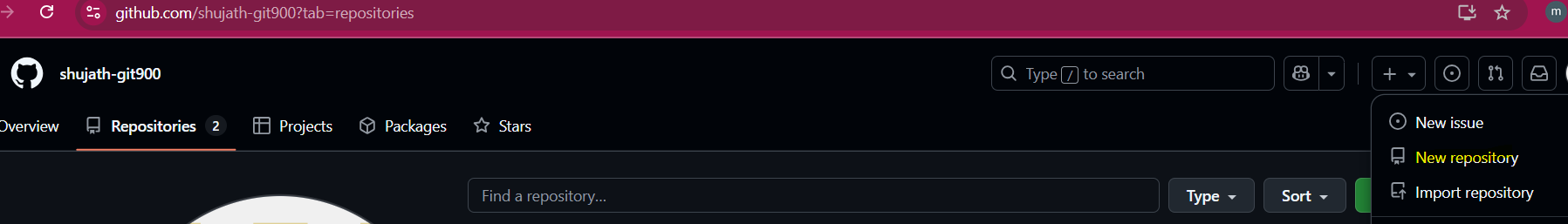
Install git :

<https://git-scm.com/downloads>

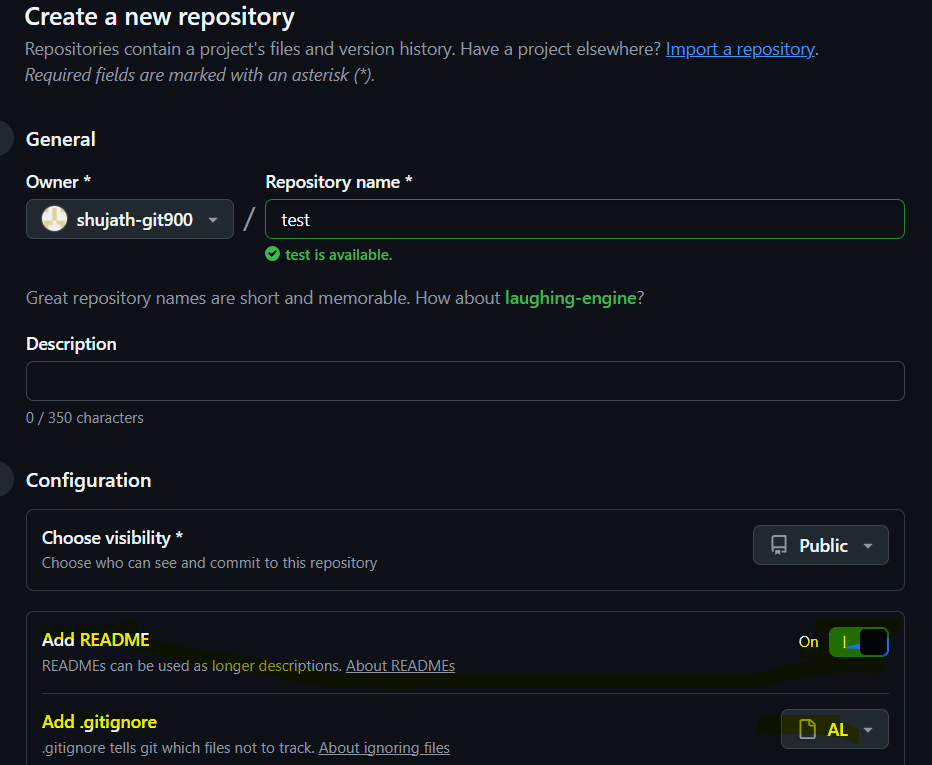


as per the os select option and install.

1. Create a repo in github with README.md and .ignore file.
   * Create a repo in github: open your github account and click on ‘New Repository’

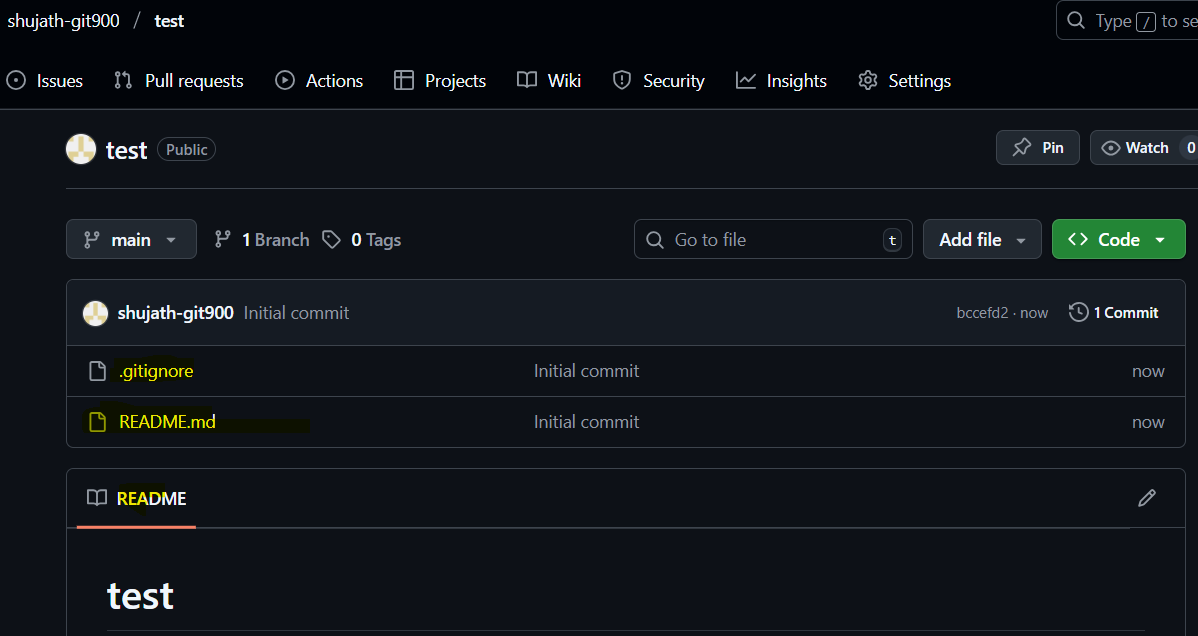


* + Give the repo name and enable the README file and Add .gitignore file:

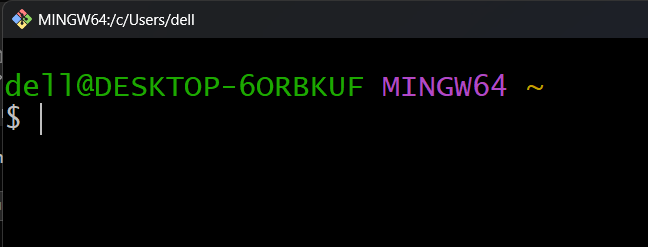


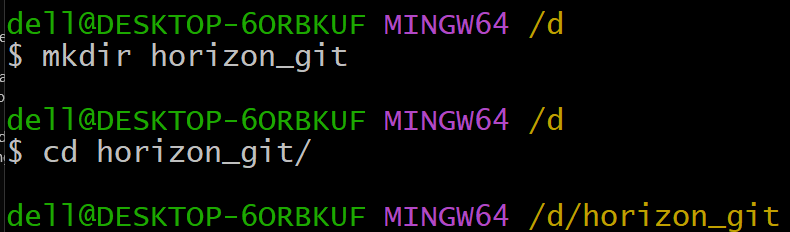
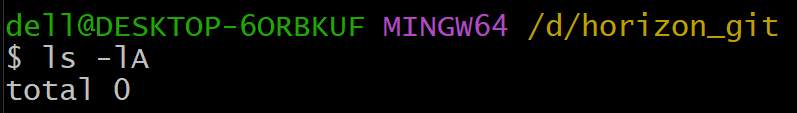
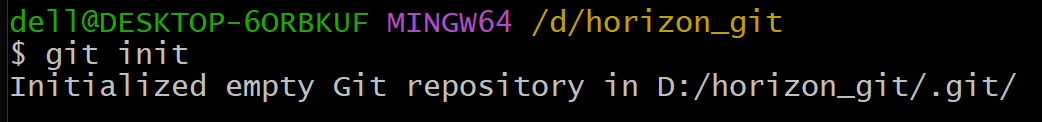
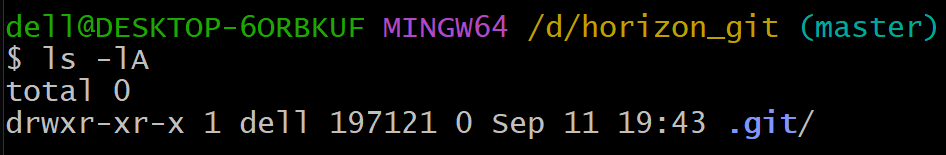
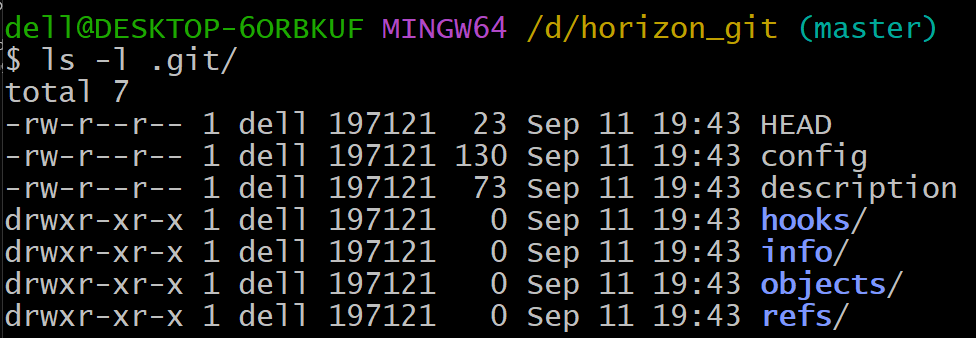
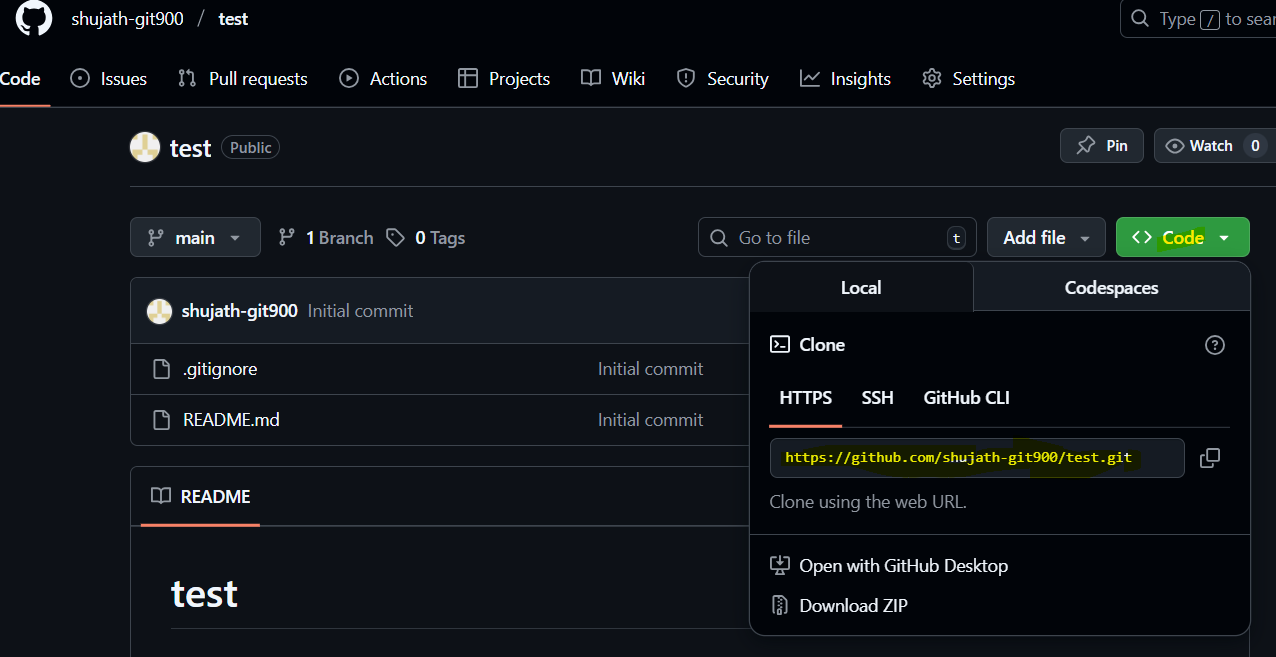
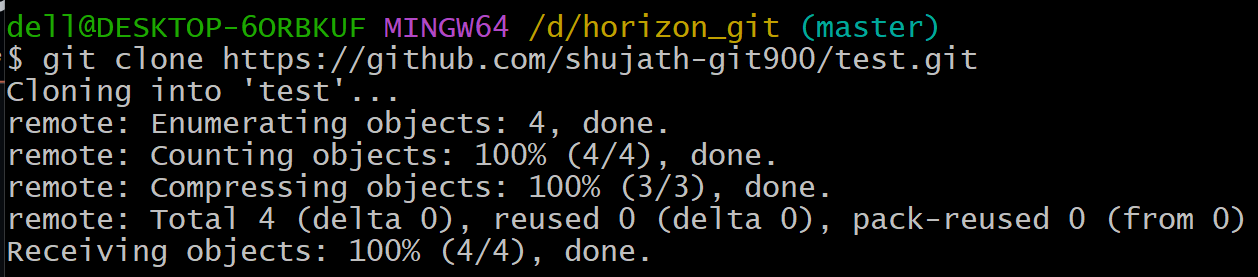
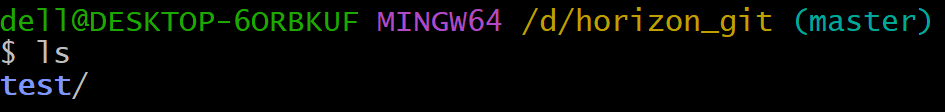
* + Click on Create Repository

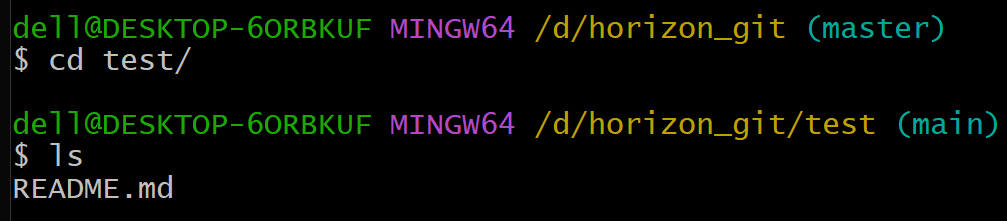
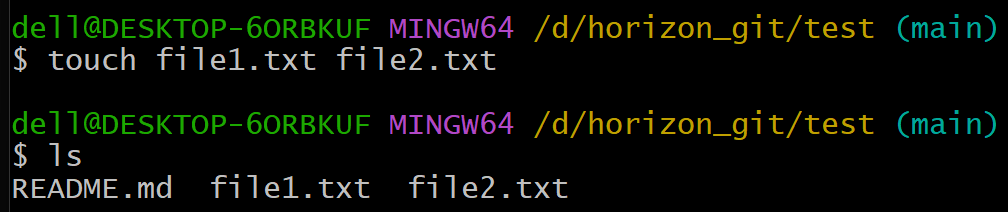
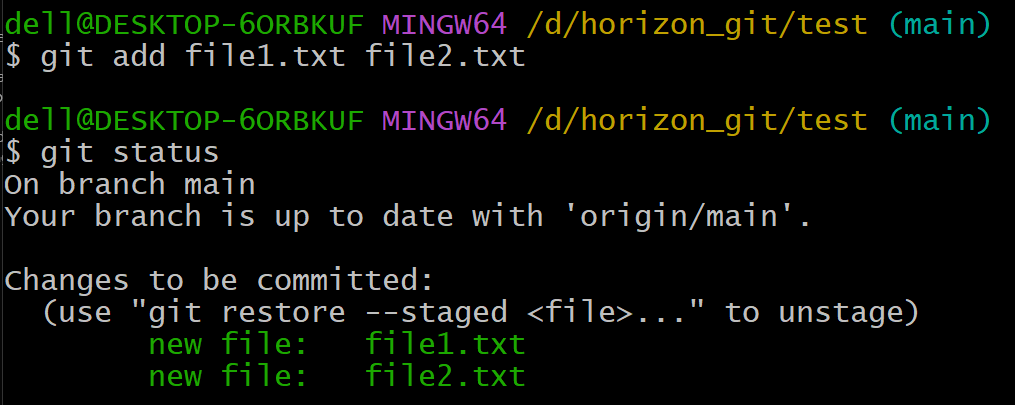
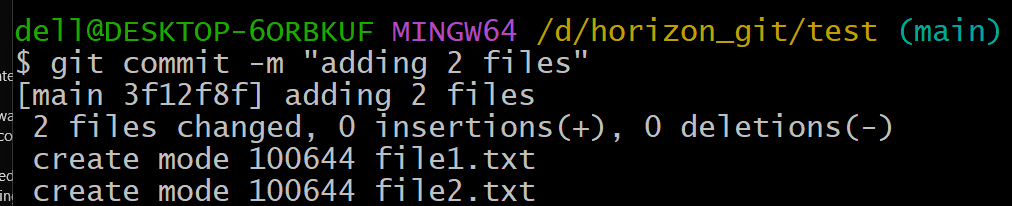
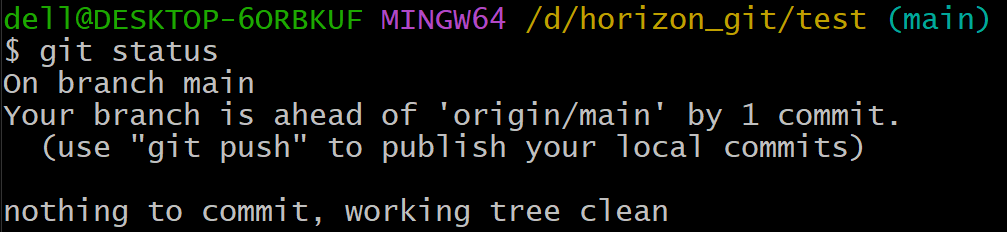
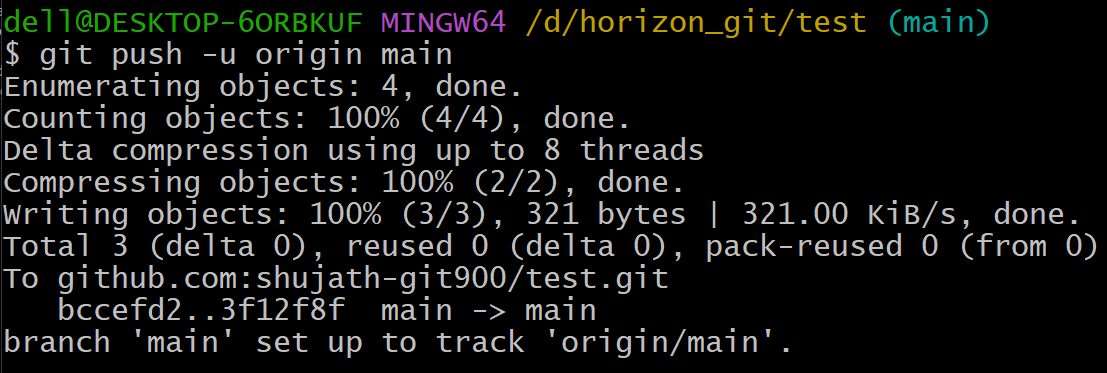
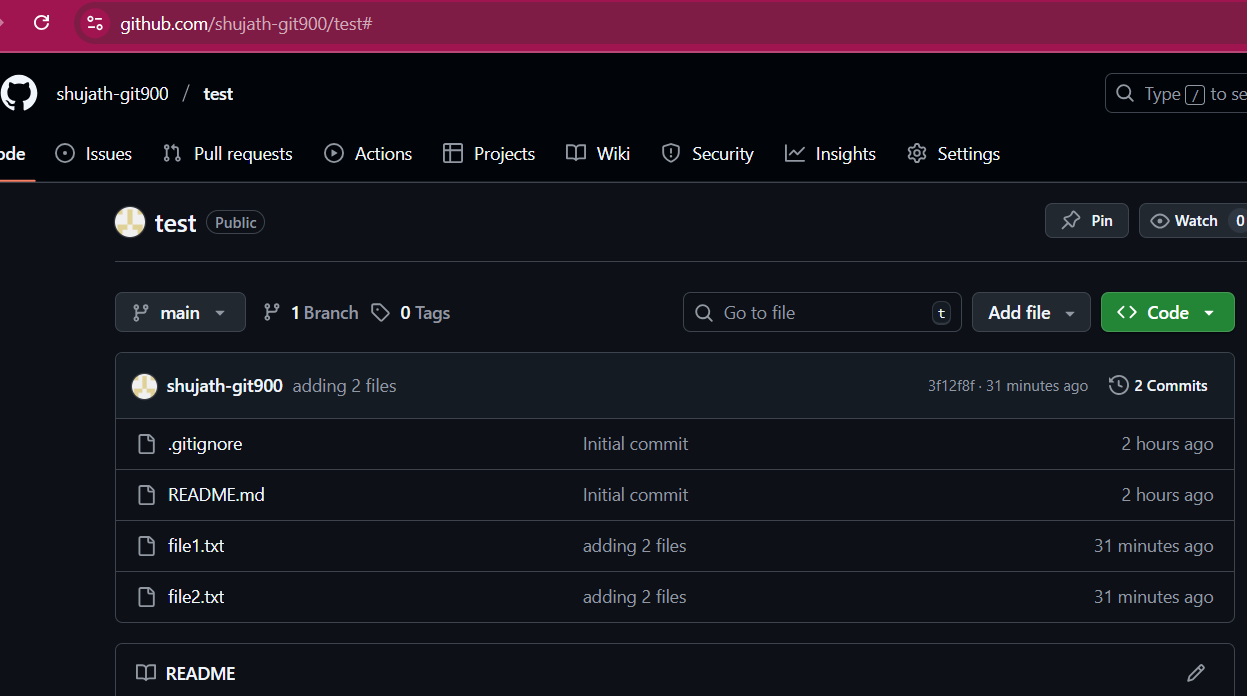
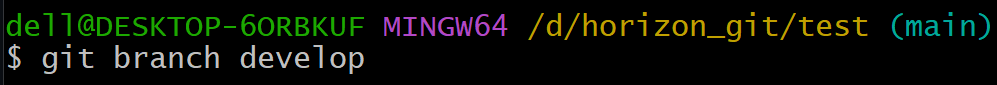
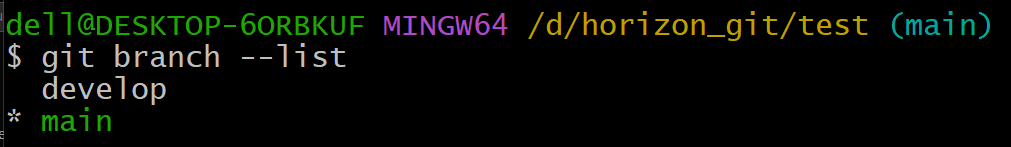
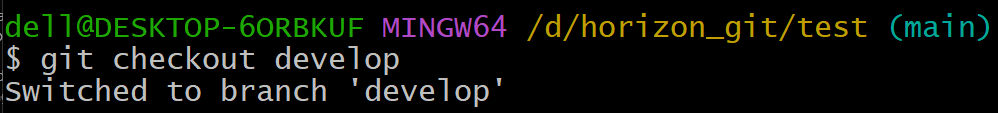
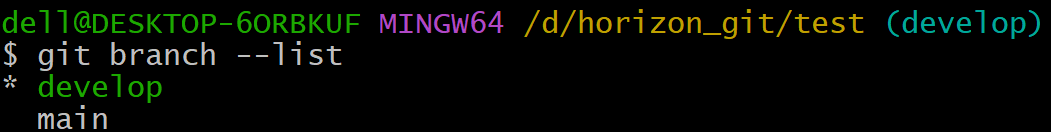
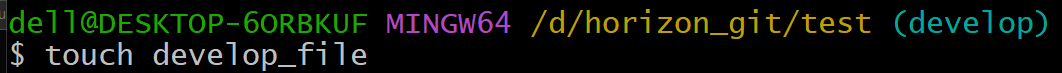
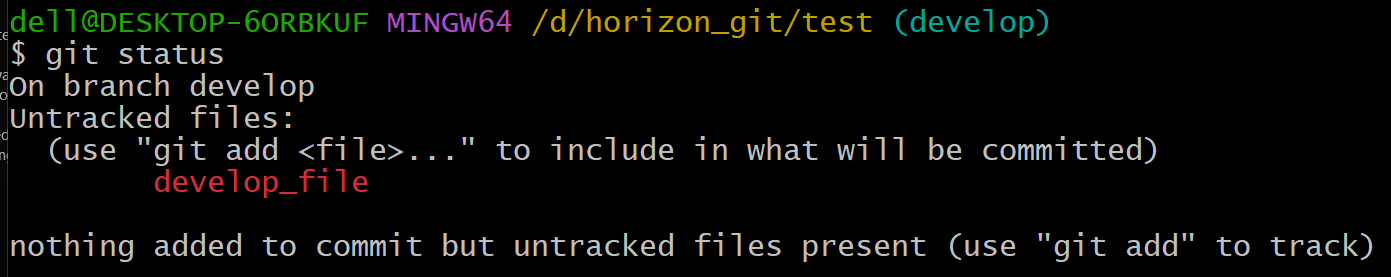
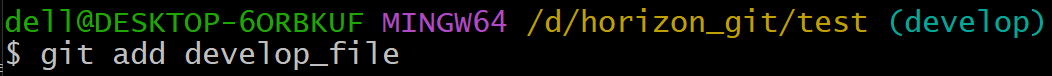
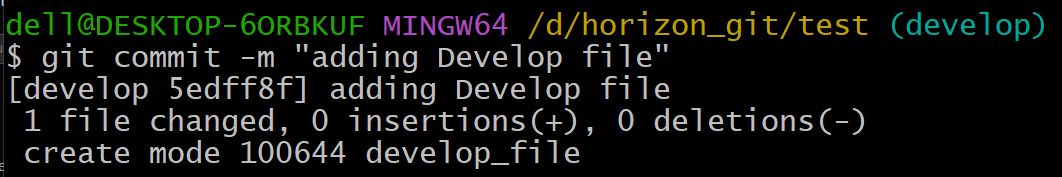
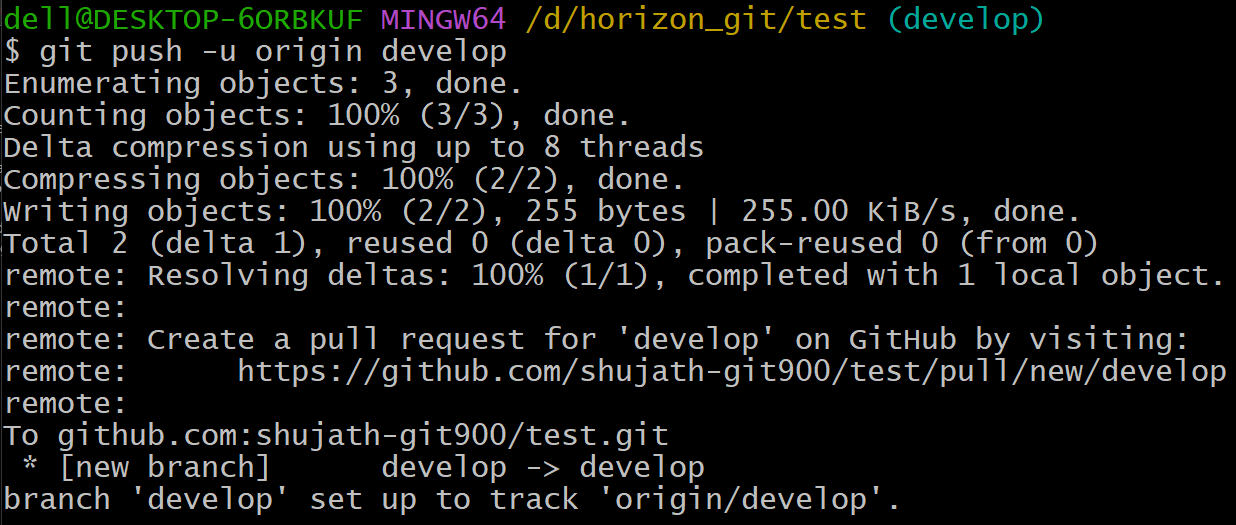
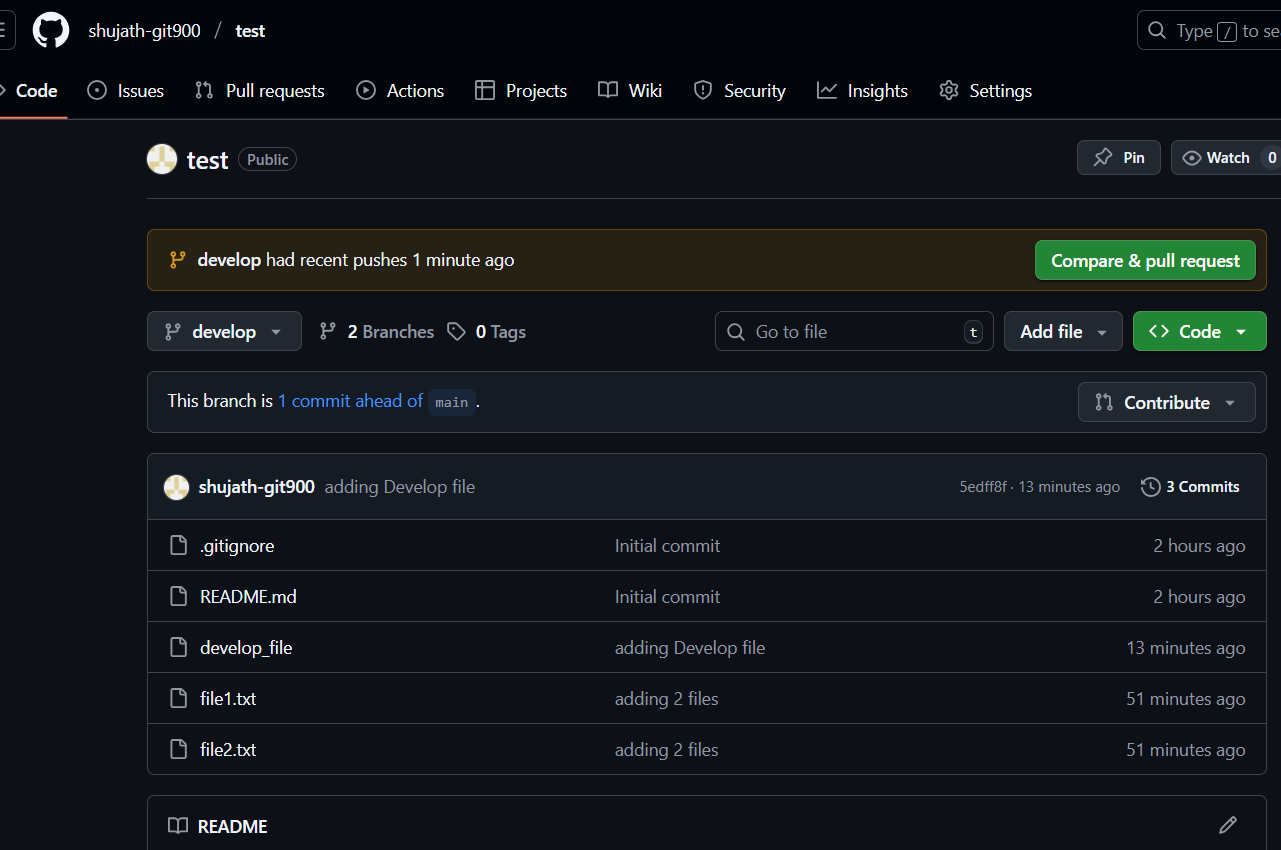
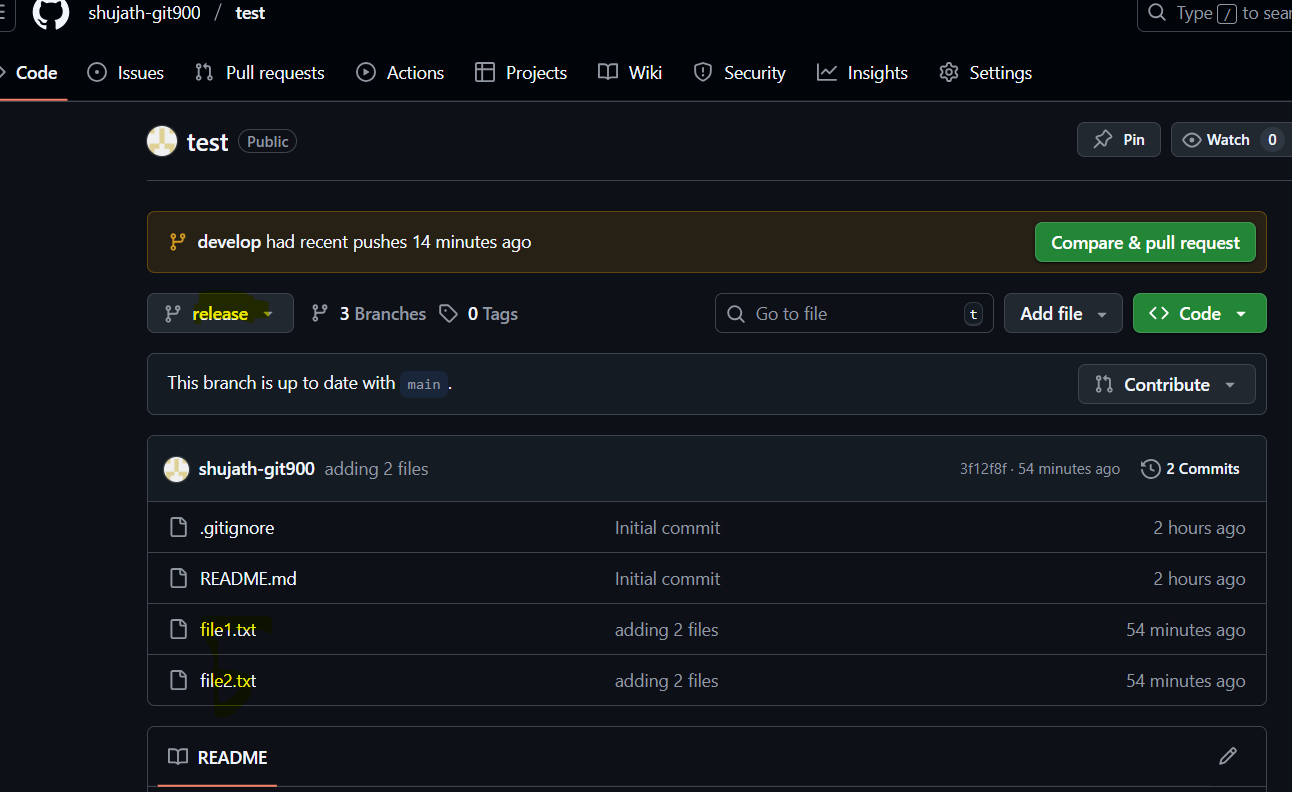
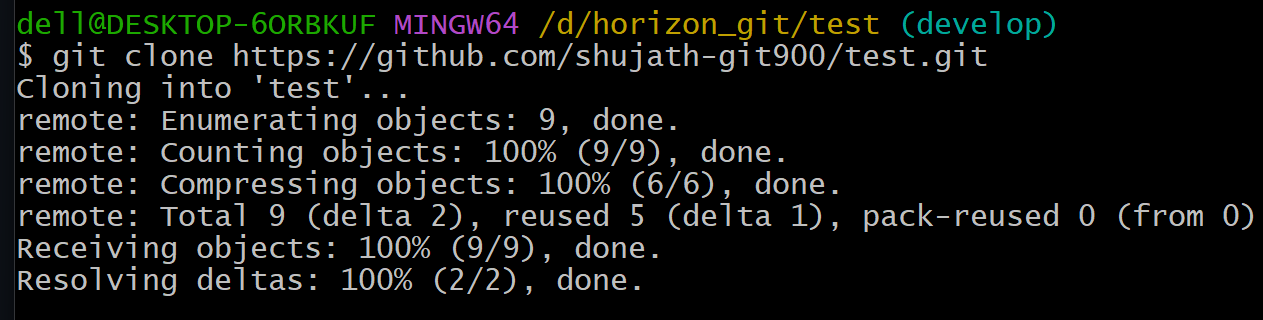
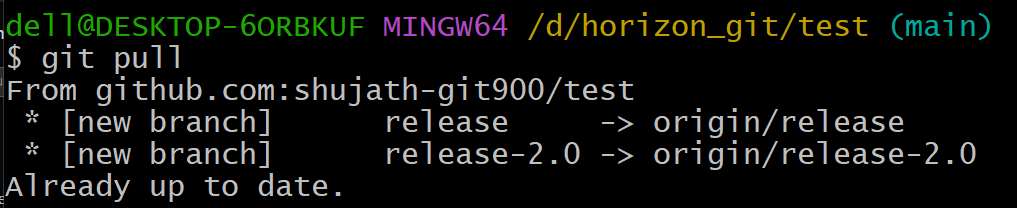
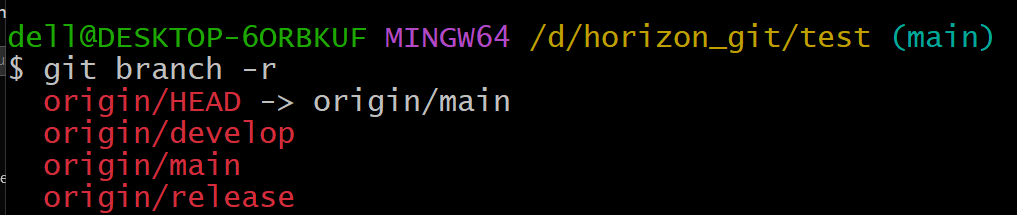
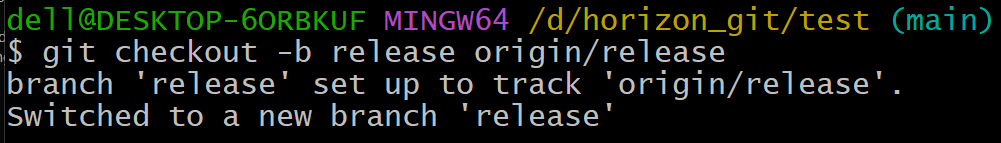
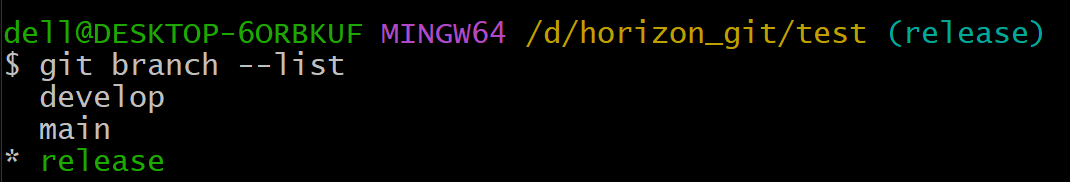
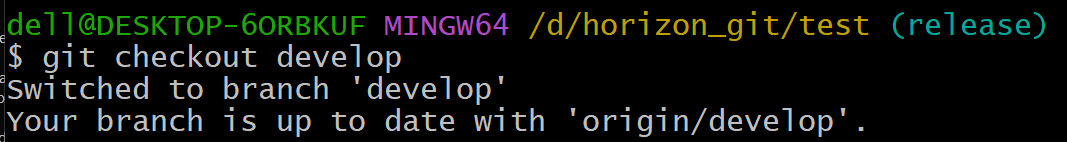
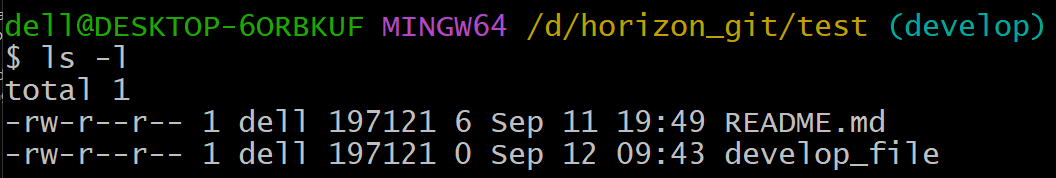
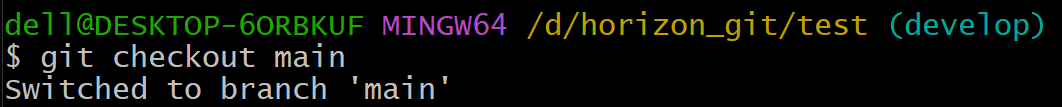
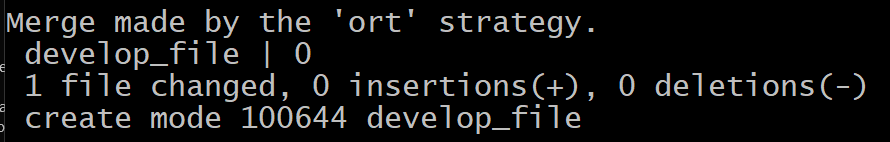
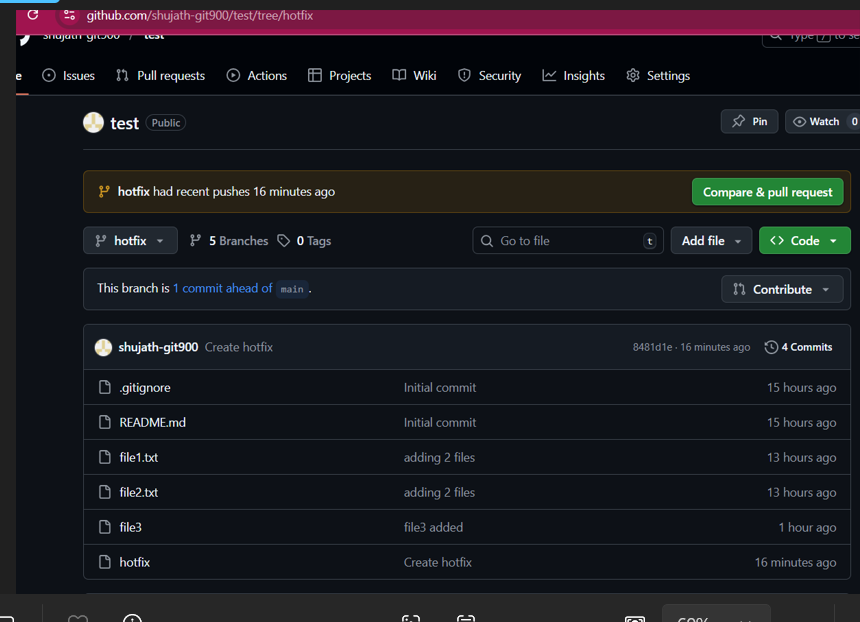
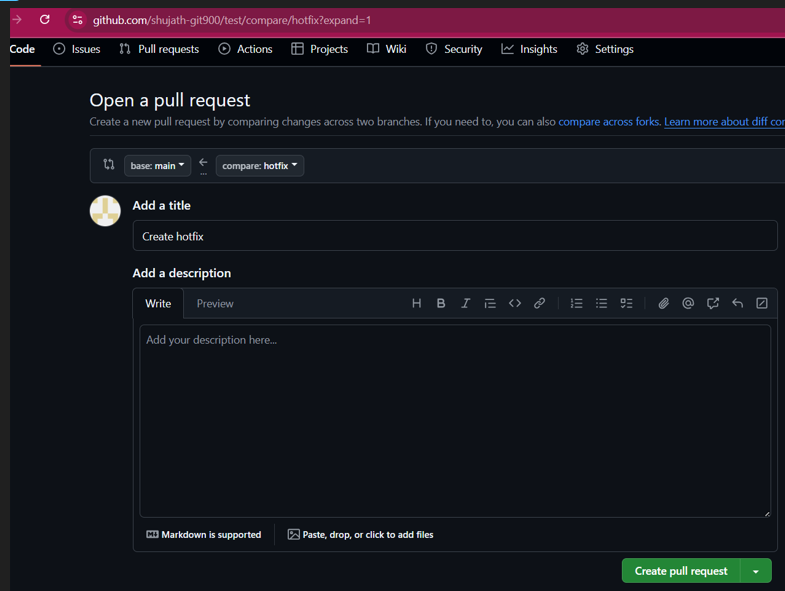
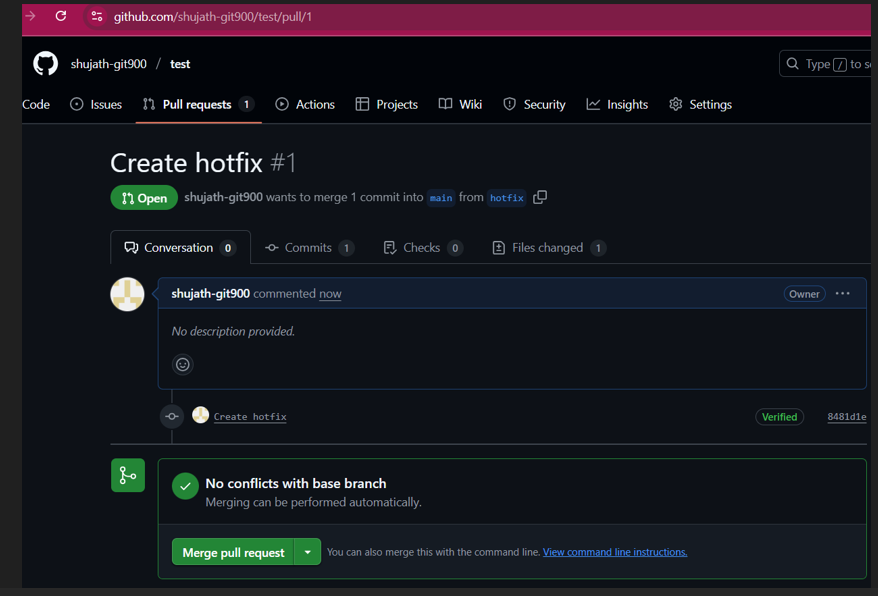
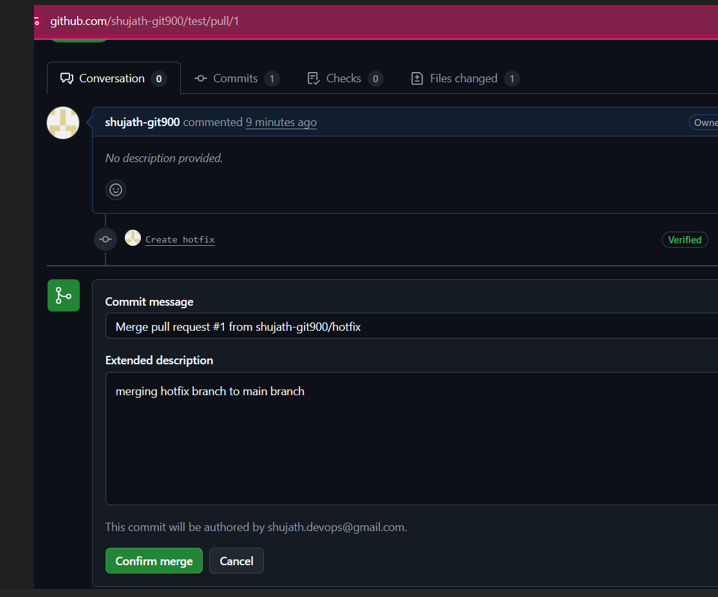
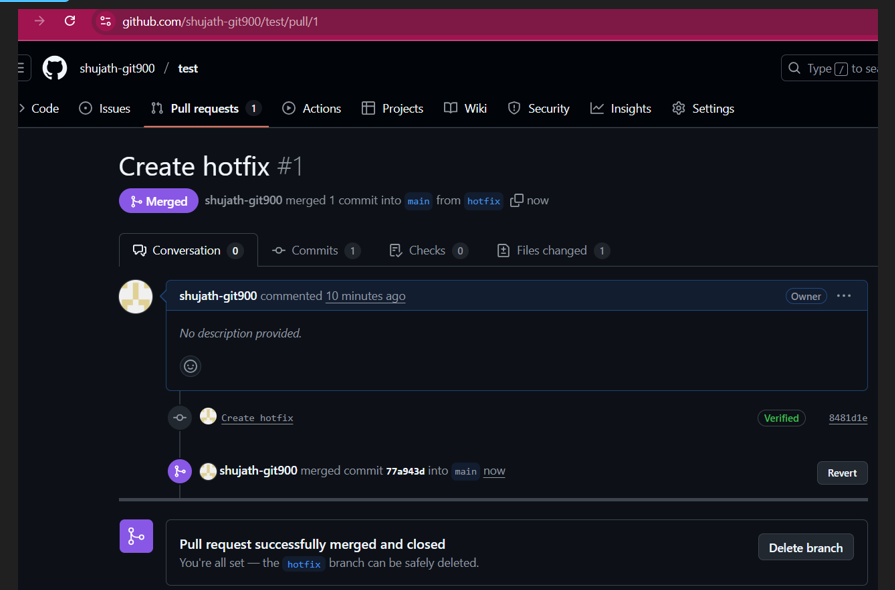
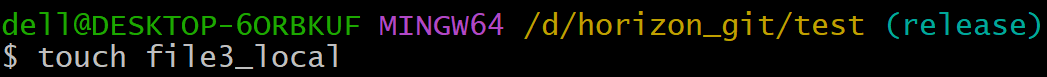
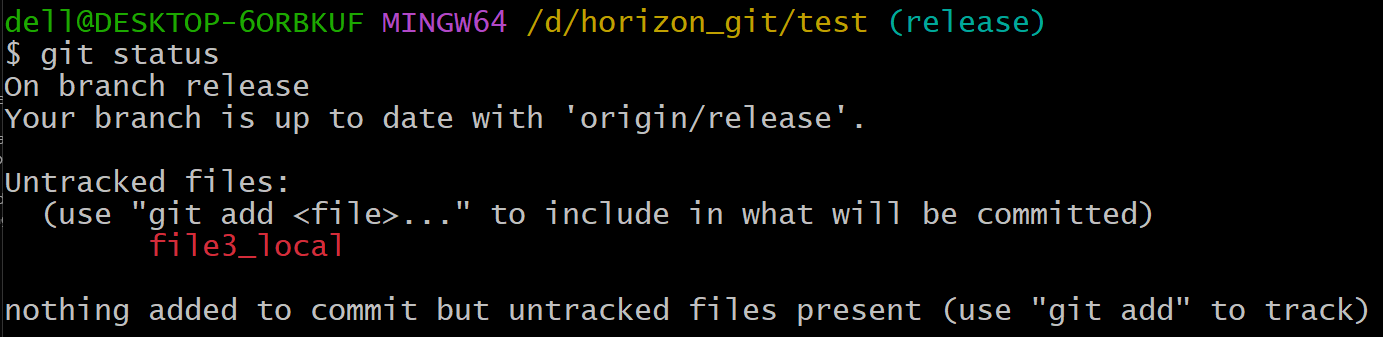
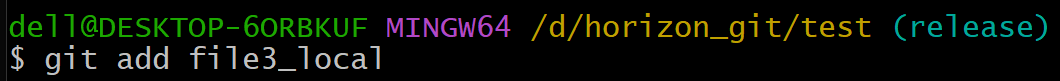
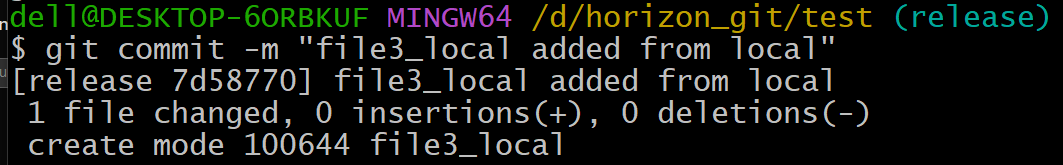
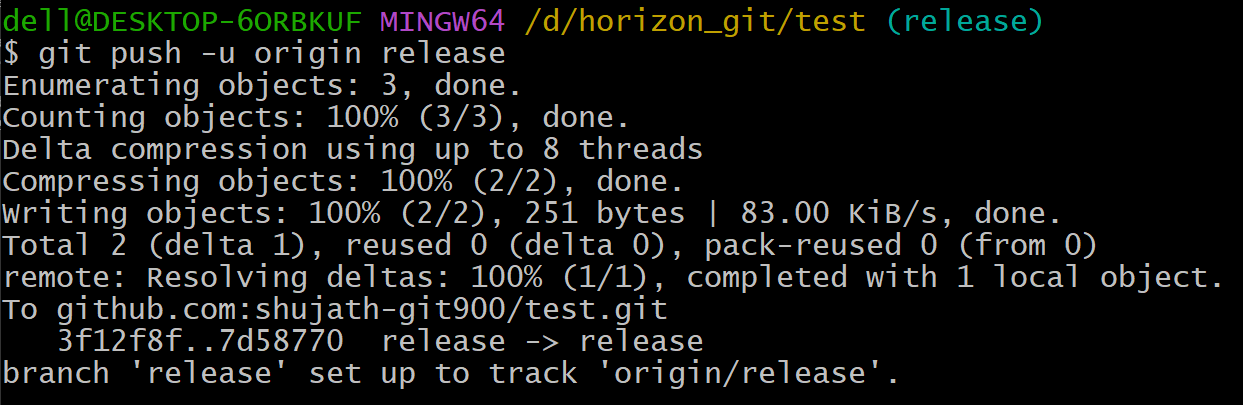
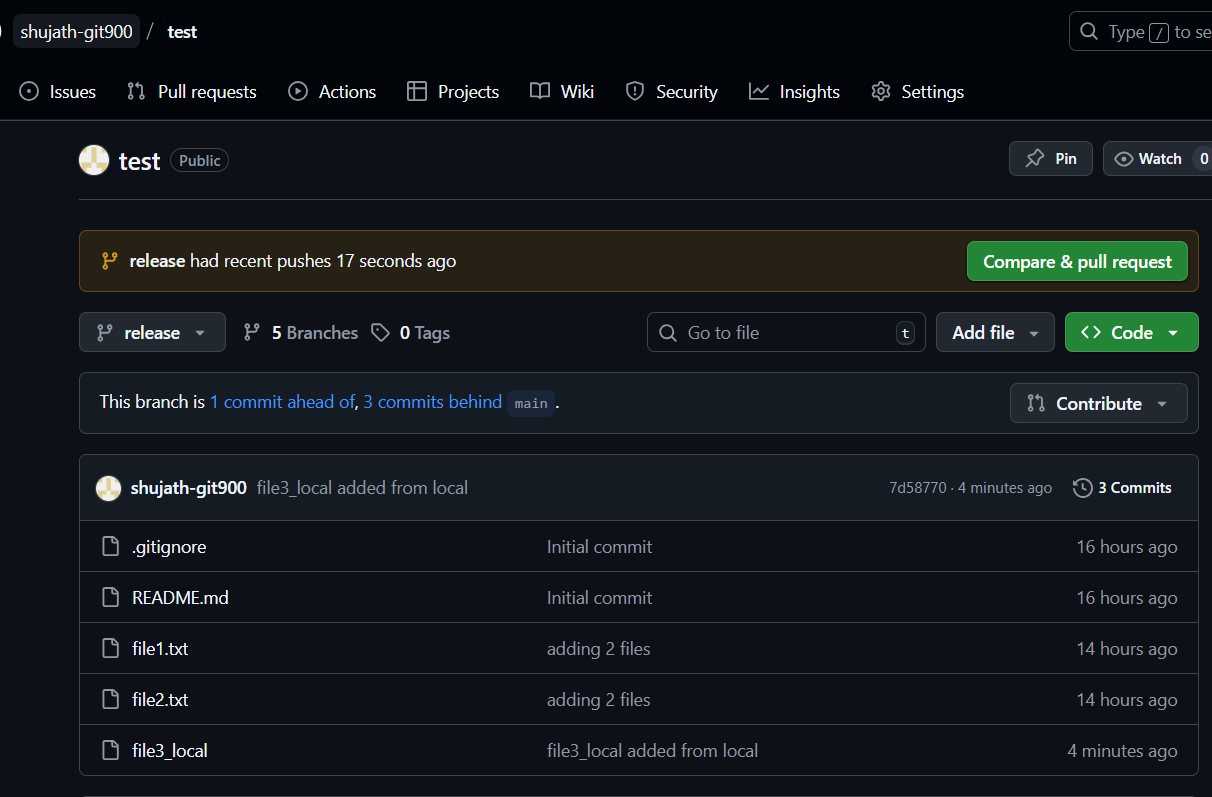
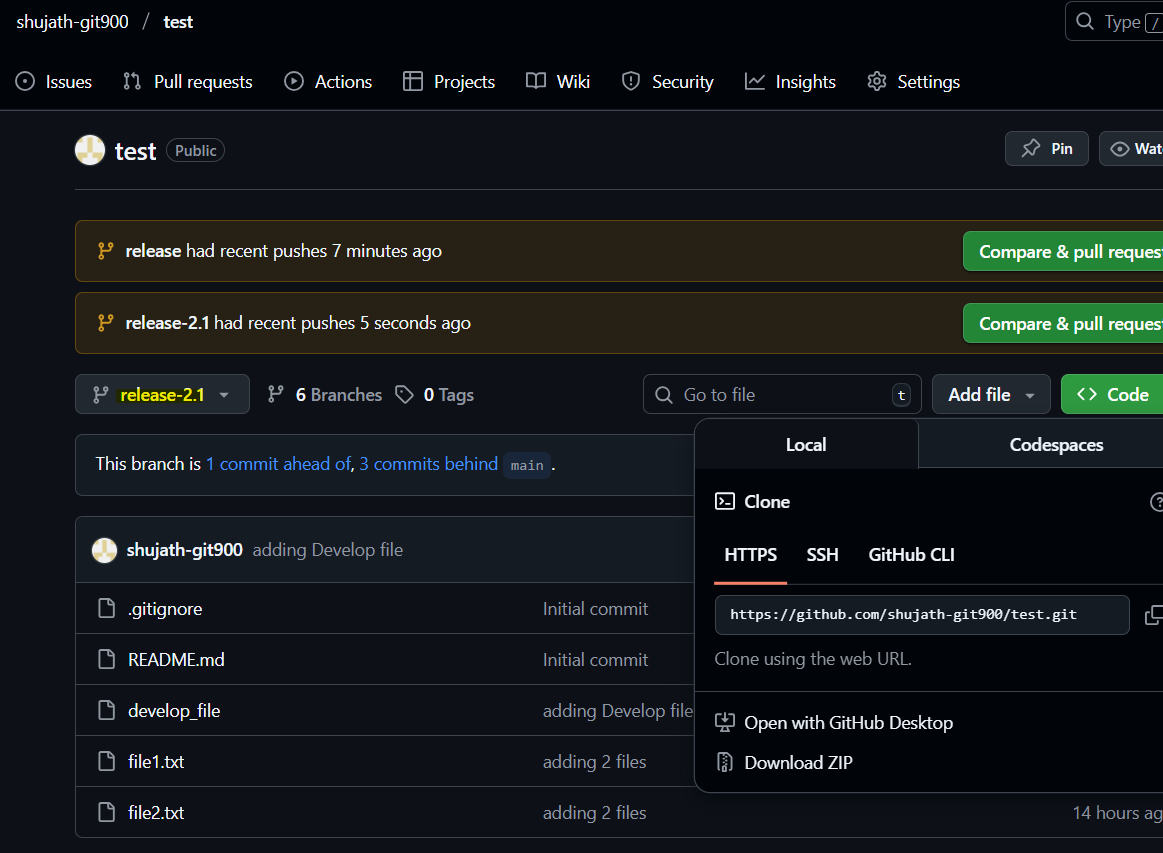
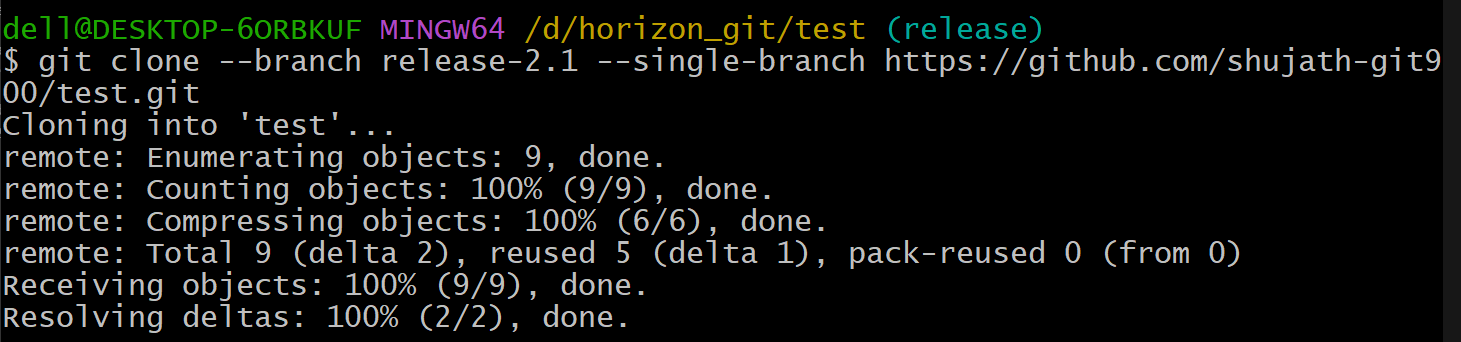




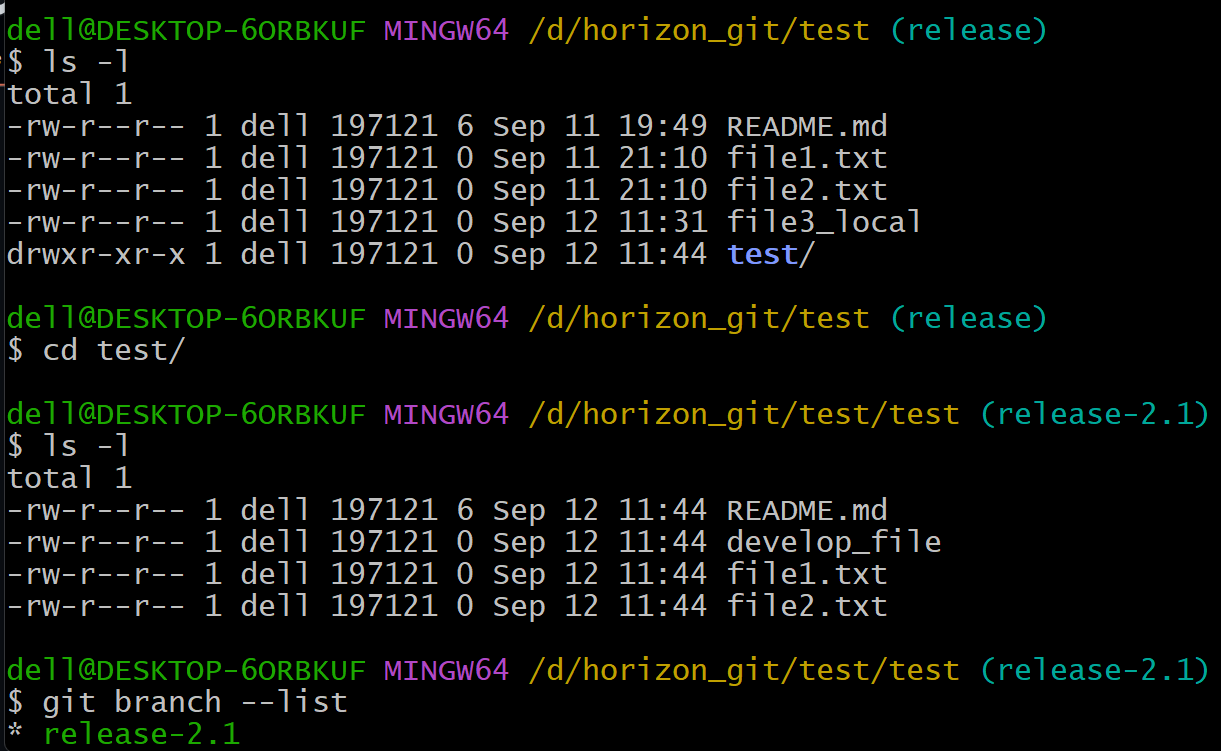
1. Clone the created repo to local.
   * Open the gitbash :

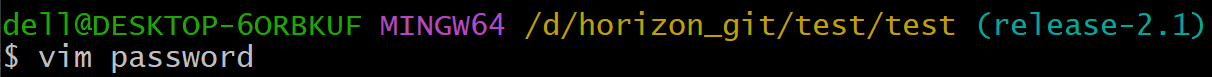
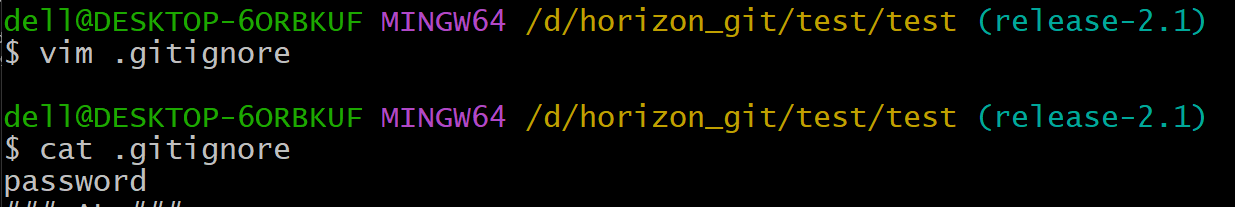
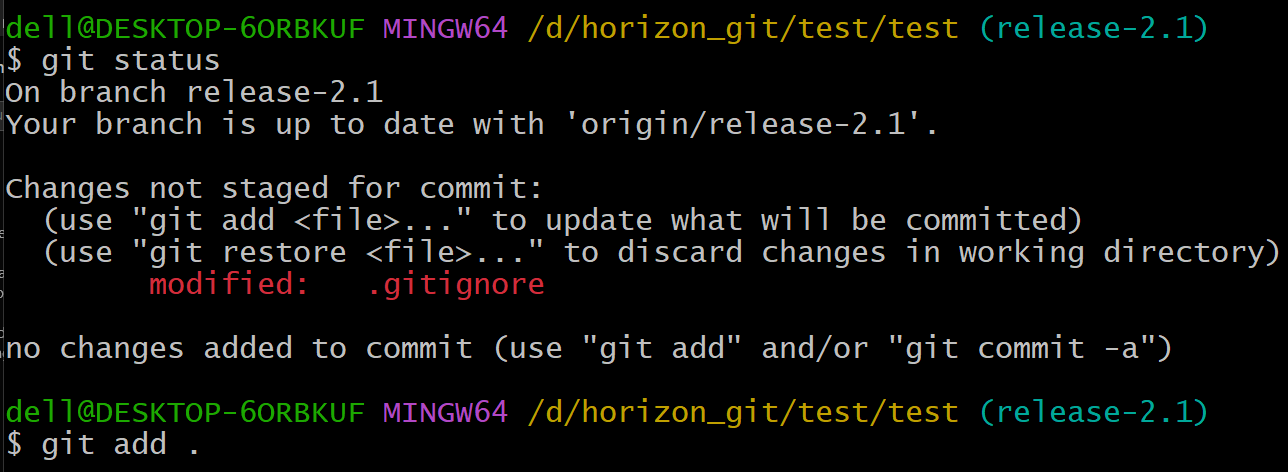
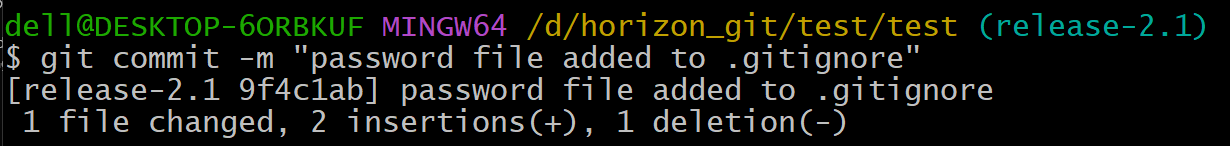
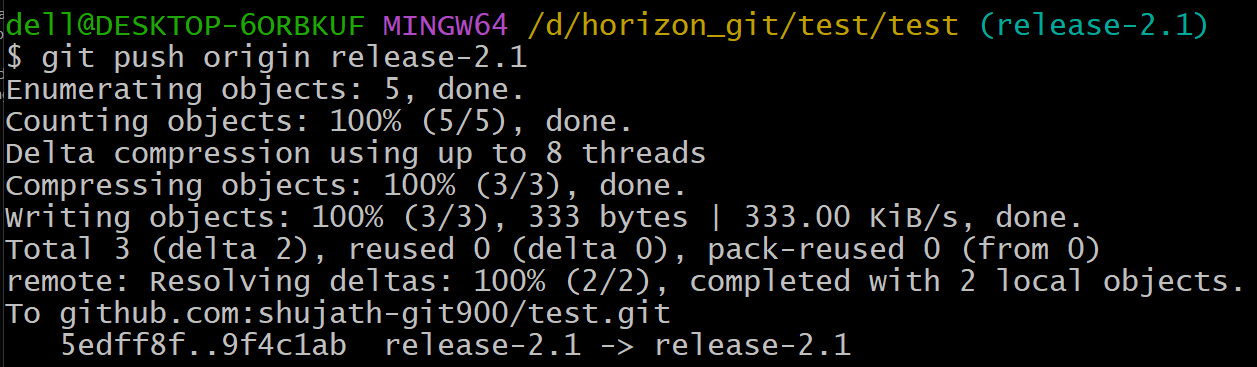
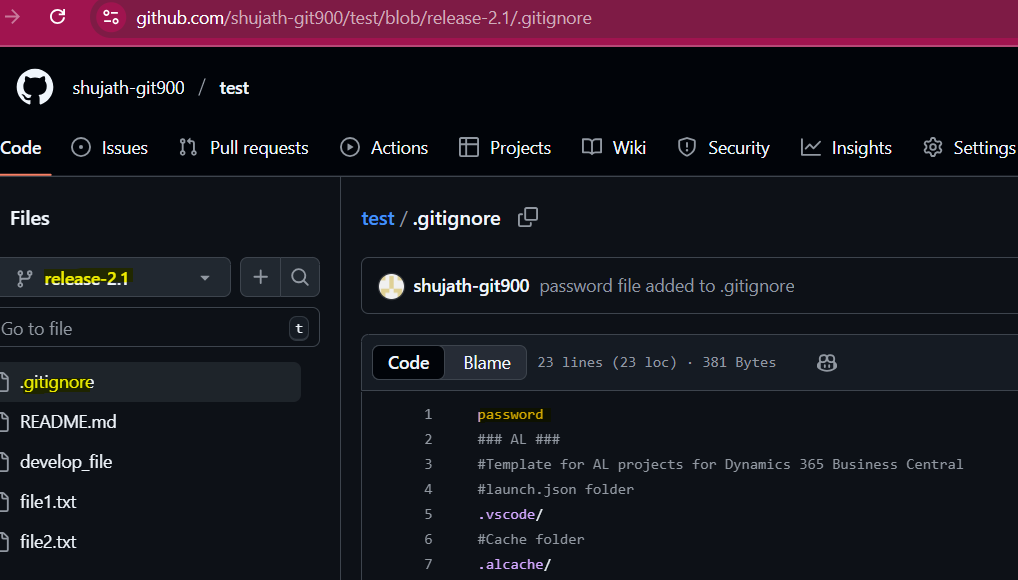
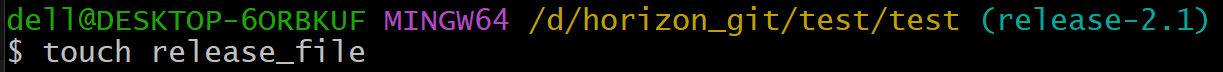
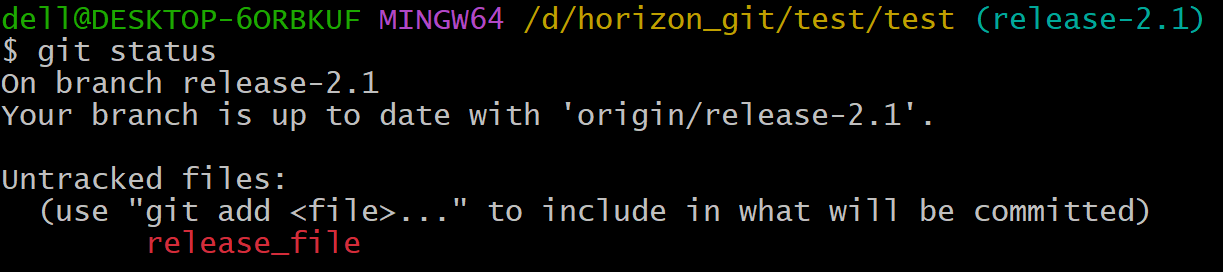
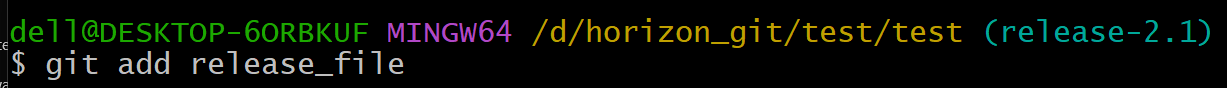
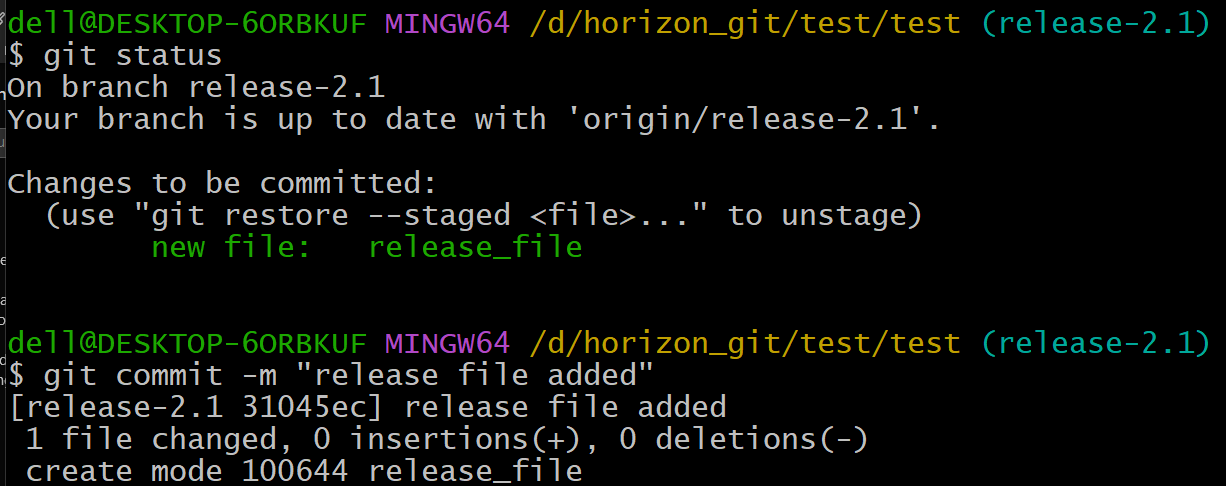
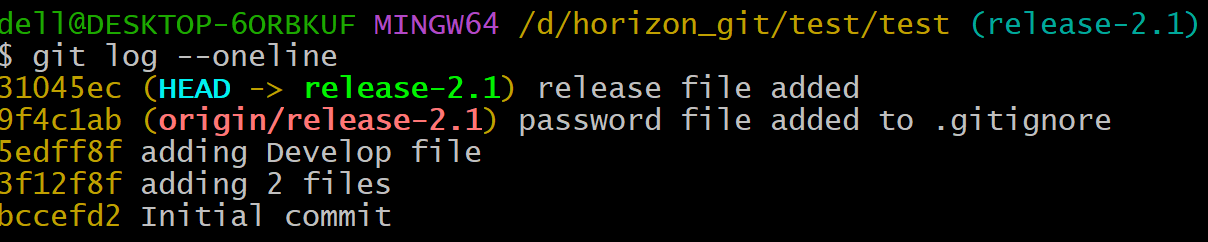
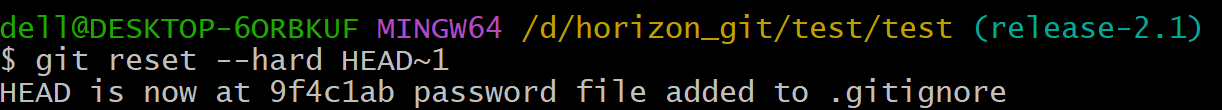
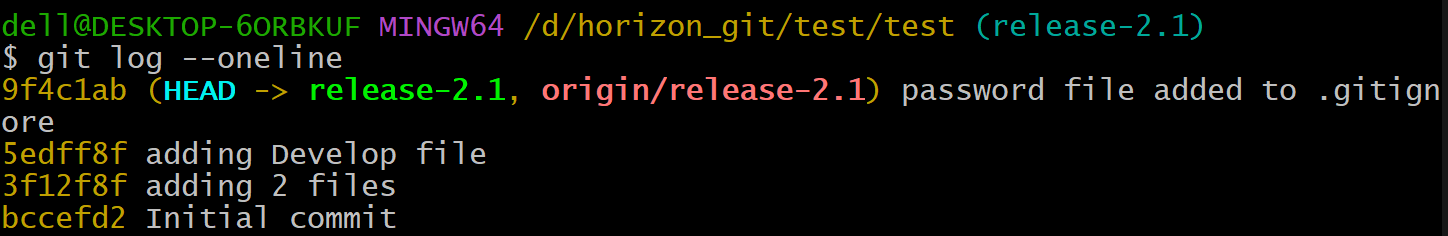
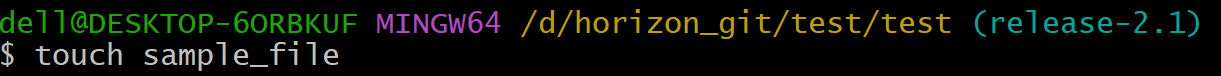
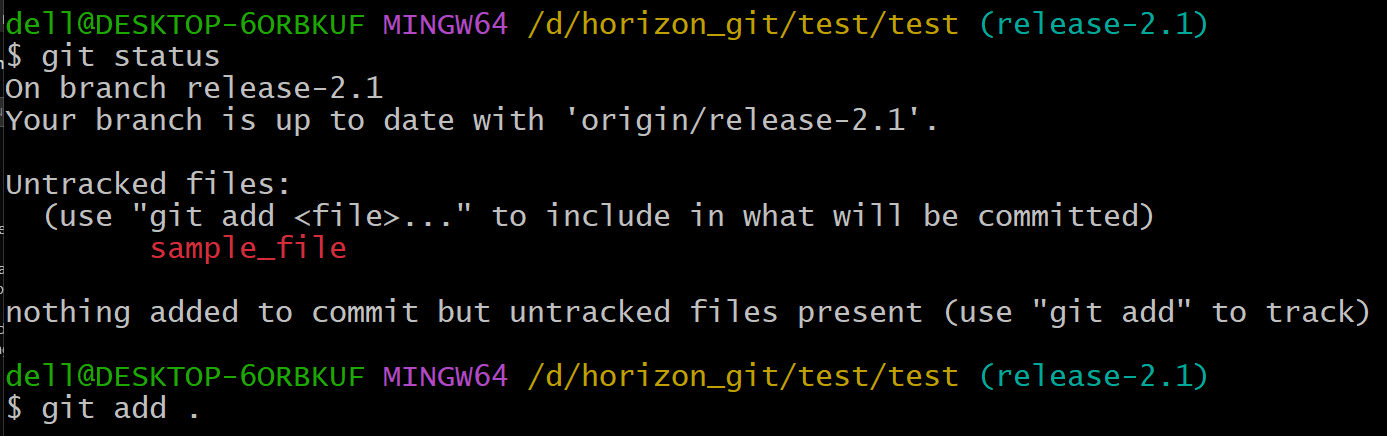


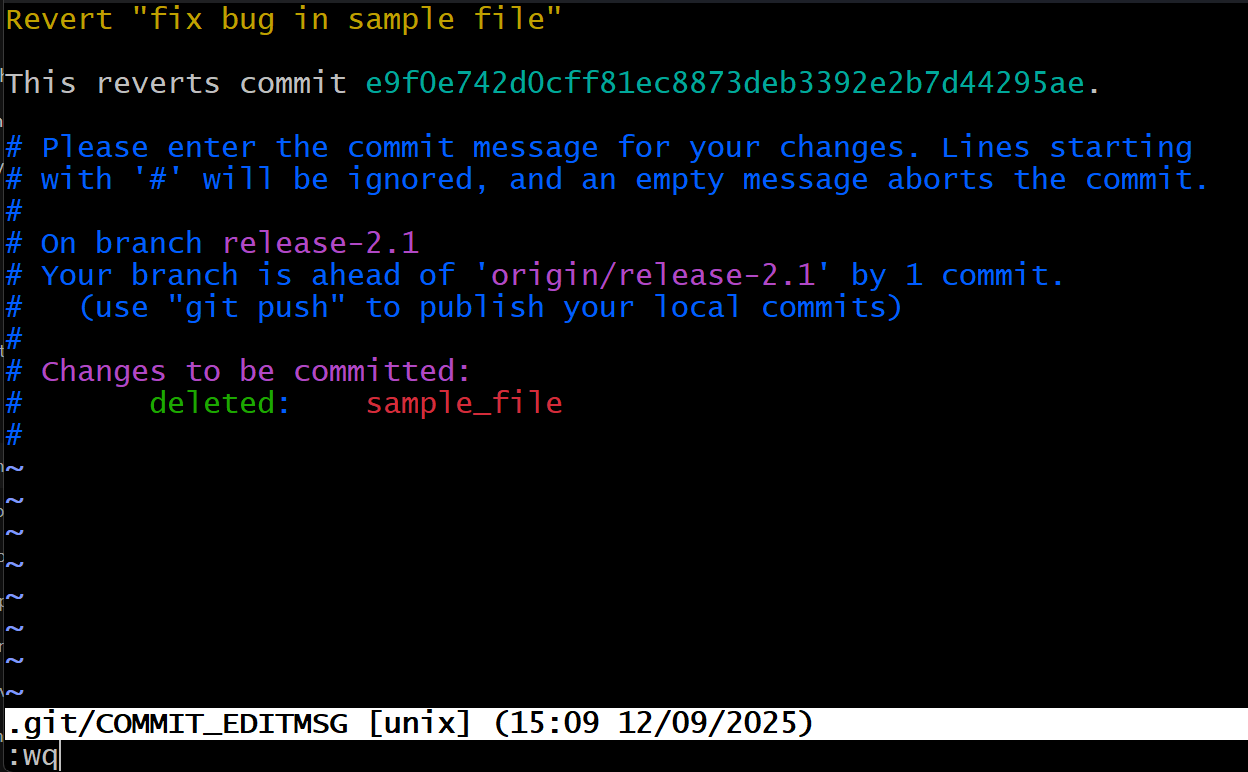
* + Create a directory ‘horizon\_git’ and change directory:
  + 
  + Here we can see we don’t have any files or directories:
  + 
  + Next initialize the git using ‘git init’:
  + 
  + Then it will create a hidden .git directory which will contain the configuration
  + 
  + 
  + Copy the repo url:
  + 
  + Next goto gitbash and clone the repo:
  + 
  + Check the cloned repo in gitbash:
  + 

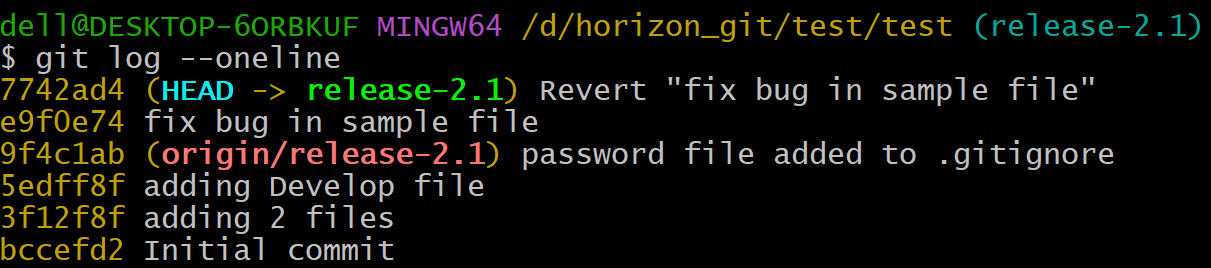
1. Create two files in local repo.
   * Goto local repo with cd:
   * 
   * Create 2 files:
   * 
2. Commit two files and push to central Repository.
   * Git add file1.txt file2.txt
   * 
   * Git commit -m “adding 2 files”:
   * 
   * Check the status with git status:
   * 
   * Push to central repo using git push:
   * 
   * Check the central repo, it will be pushed to github repo test the 2 files :
   * 
3. Create a branch in local and create a sample file and push to central.
   * Create a branch with name ‘develop’:
   * 
   * Check in which branch you are:
   * 
   * Switch to develop branch:
   * 
   * 
   * Create a file :
   * 
   * Check the status:
   * 
   * Do the ‘git add develop\_file’
   * 
   * Commit it:
   * 
   * Push to central repo:
   * 
   * Check in the github repo the ‘Develop’ branch with files’:
   * 
4. Create a branch in github and clone that to local.
   * Create a branch in github with name ‘release’ from main branch:
   * 
   * Next, do the git clone with url:
   * 
   * Do git pull:
   * 
   * By using git branch -r we can see our branch from github:
   * 
   * Then do git checkout -b <branch name> origin/<branch name>:
   * 
   * Check your branch in git branch –list :
   * 
5. Merge the created branch with master in git local.
   * Want to merge the created branch ‘Develop’ to main so checkout to develop and see all the files in that:
   * 
   * 
   * And switch to the main branch because we want to merge develop branch to main branch:
   * 
   * Merge the branch using ‘git merge develop’
   * Then the develop branch will get merge into the main branch means all the files of develop branch will get merge into the main branch, in this case develop branch is having develop\_file which will get merge into main branch:
   * 
6. Merge the created branch with master in github by sending a pull request.
   * Then click on compare & pull:
   * 
   * Click on Create Pull Request
   * 
   * 
   * Then click on confirm merge
   * 
   * 
7. Create a file in local and send that to branch in github.
   * Create a file in ‘release’ branch in the local repo:
   * 
   * Check the git status:
   * 
   * git add file3\_local:
   * 
   * Do the git commit:
   * 
   * Then push the file:
   * 
   * We can check in the github repo the pushed file which is ‘file3\_local’:
   * 
8. Clone only a branch from github to local.
   * Create a branch in github.
   * We have created release-2.1 branch in github:
   * 
   * Clone the created branch release-2.1 from github to local:
   * 

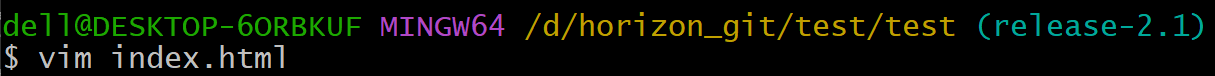
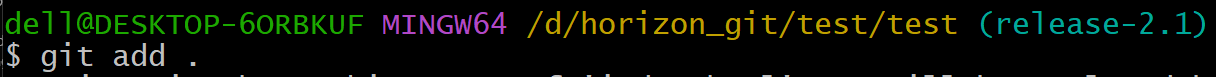
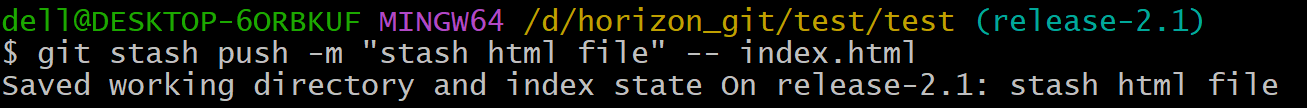
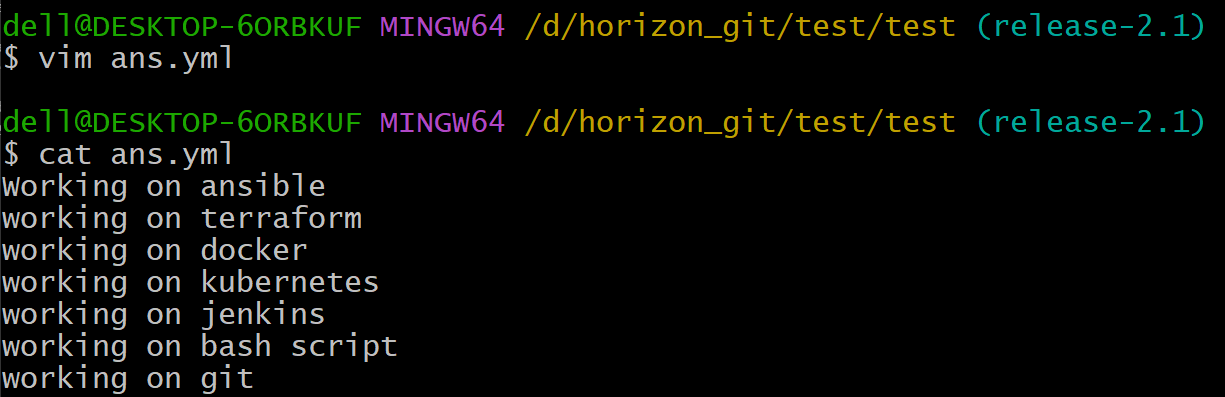
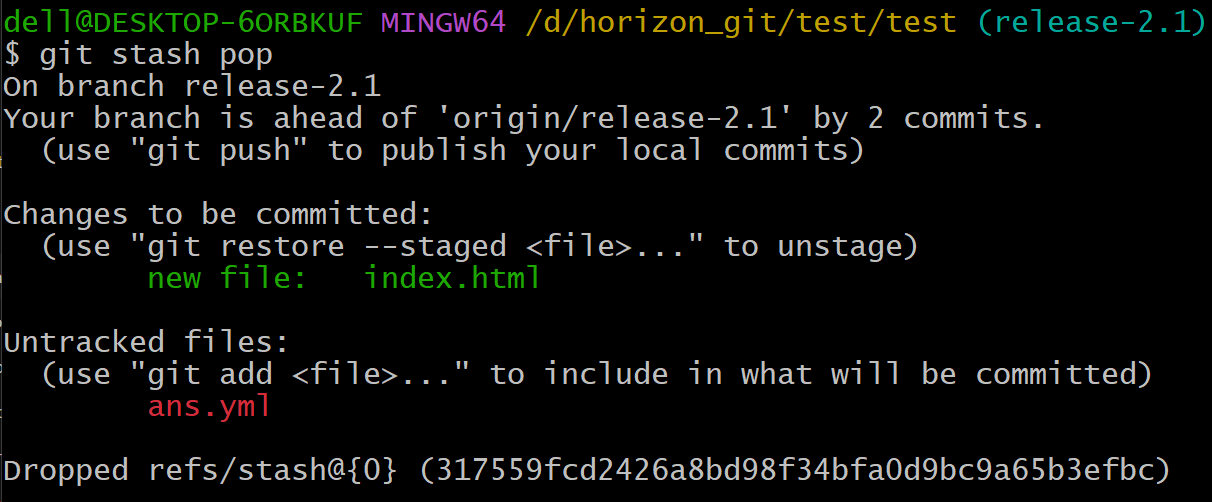
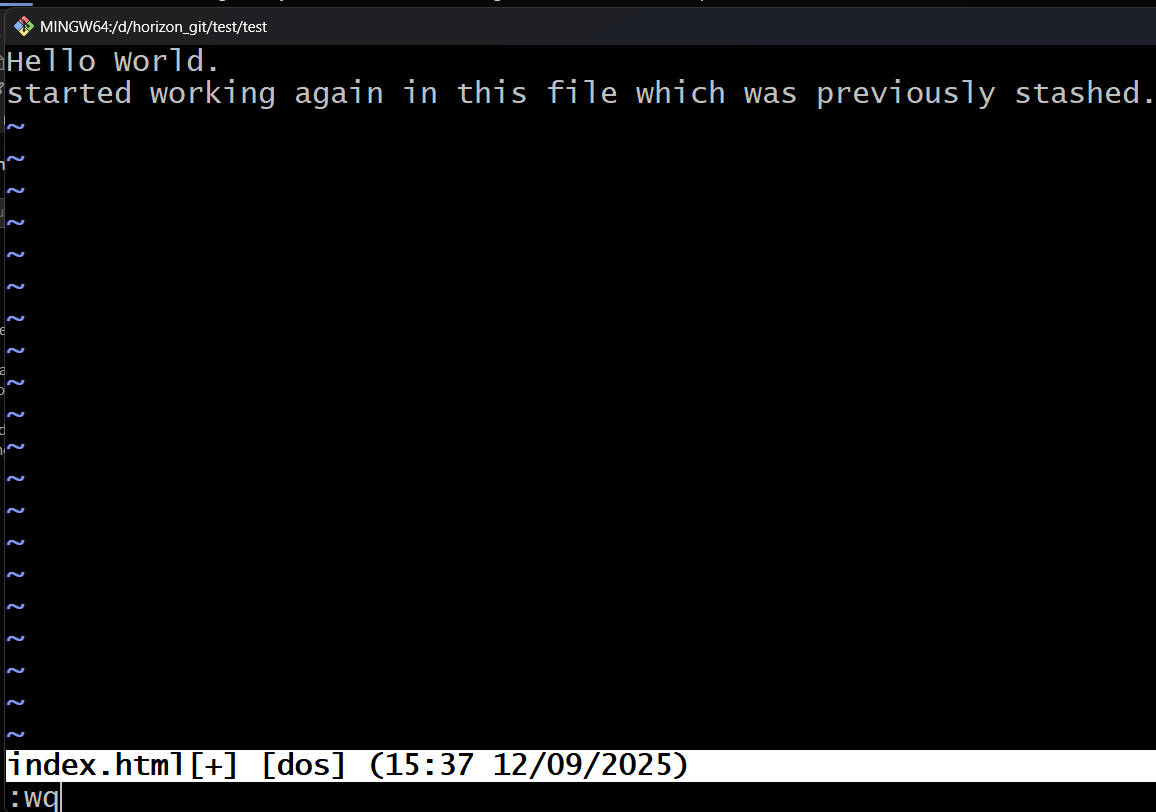
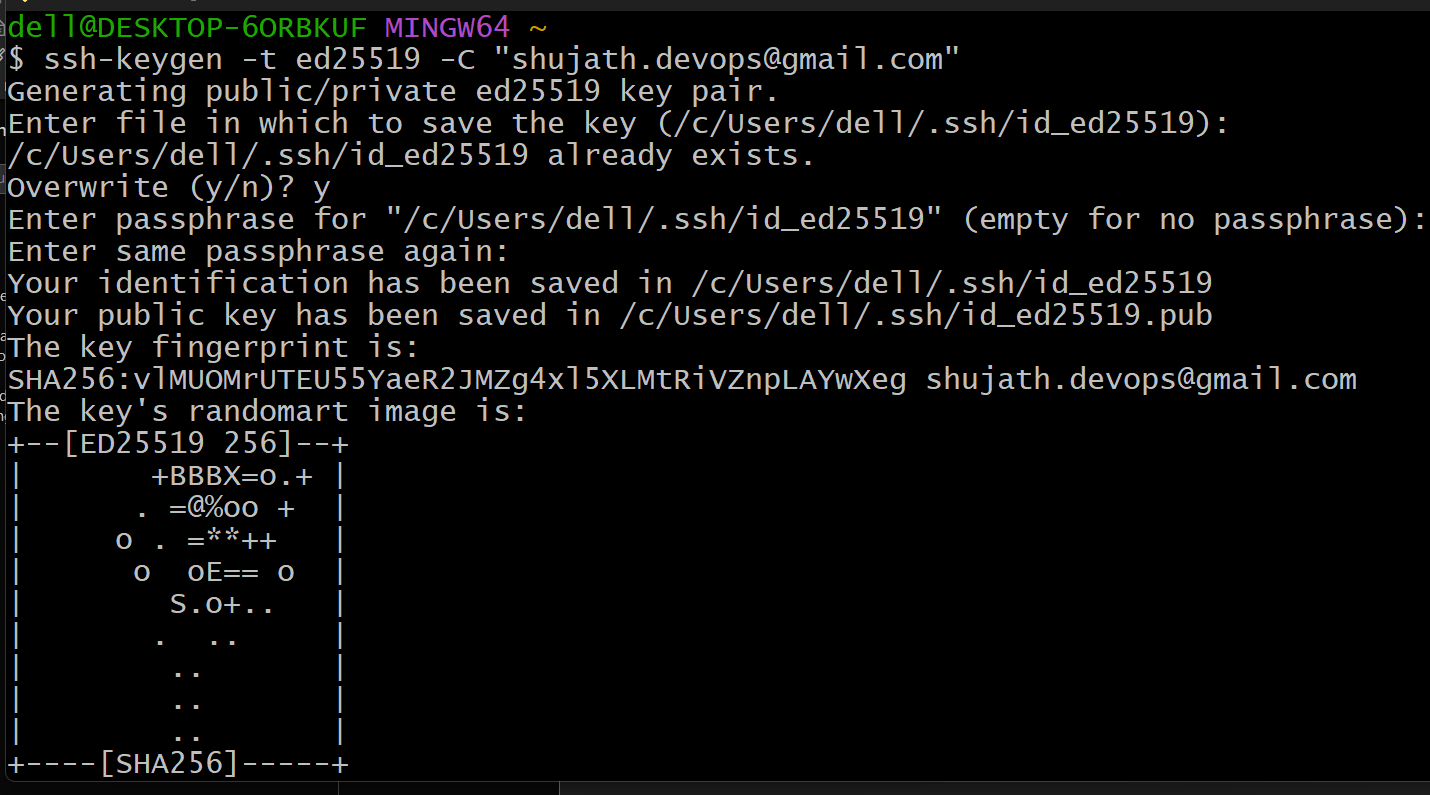
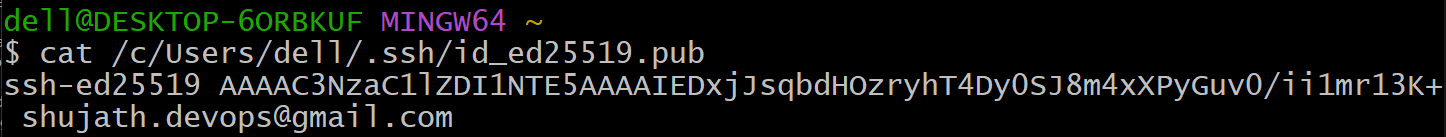
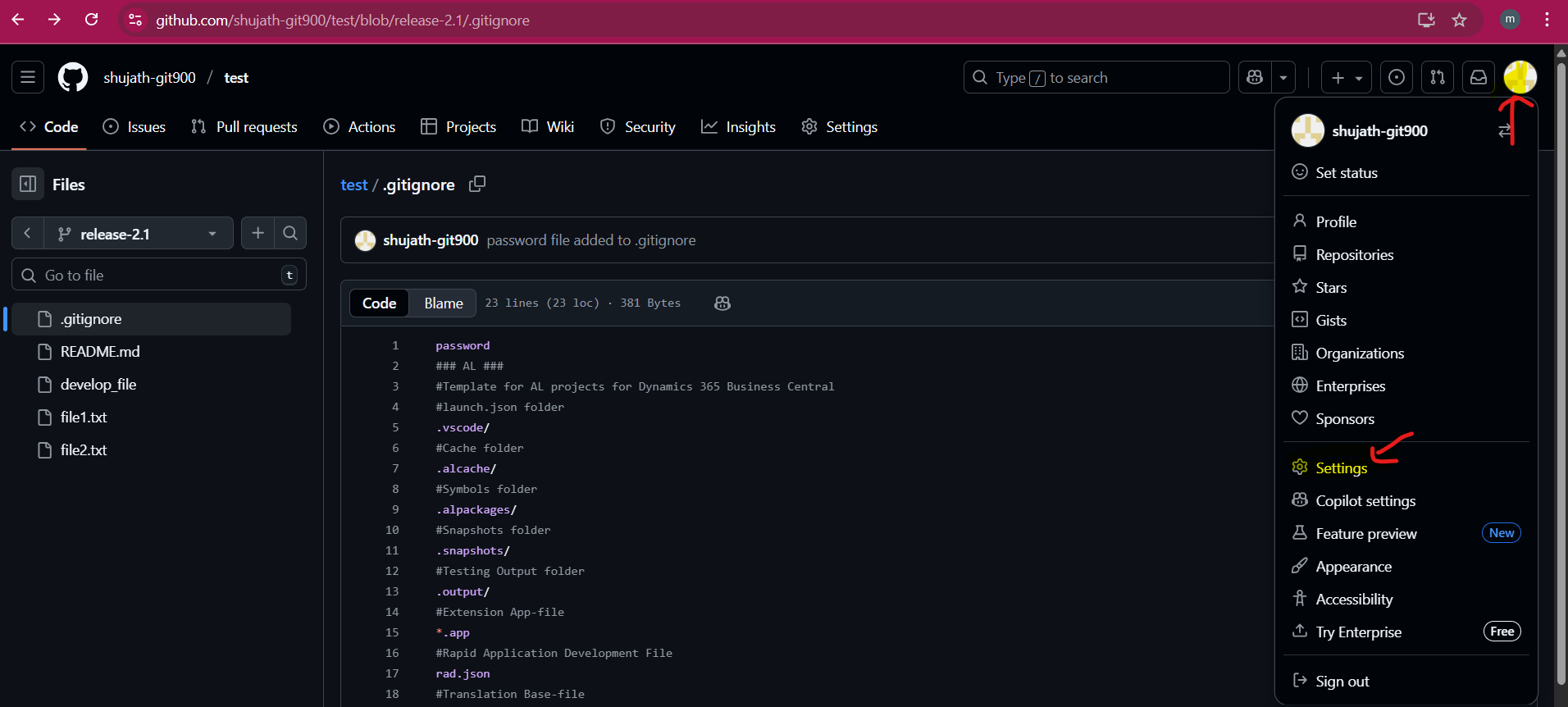
Verify the cloned branch:

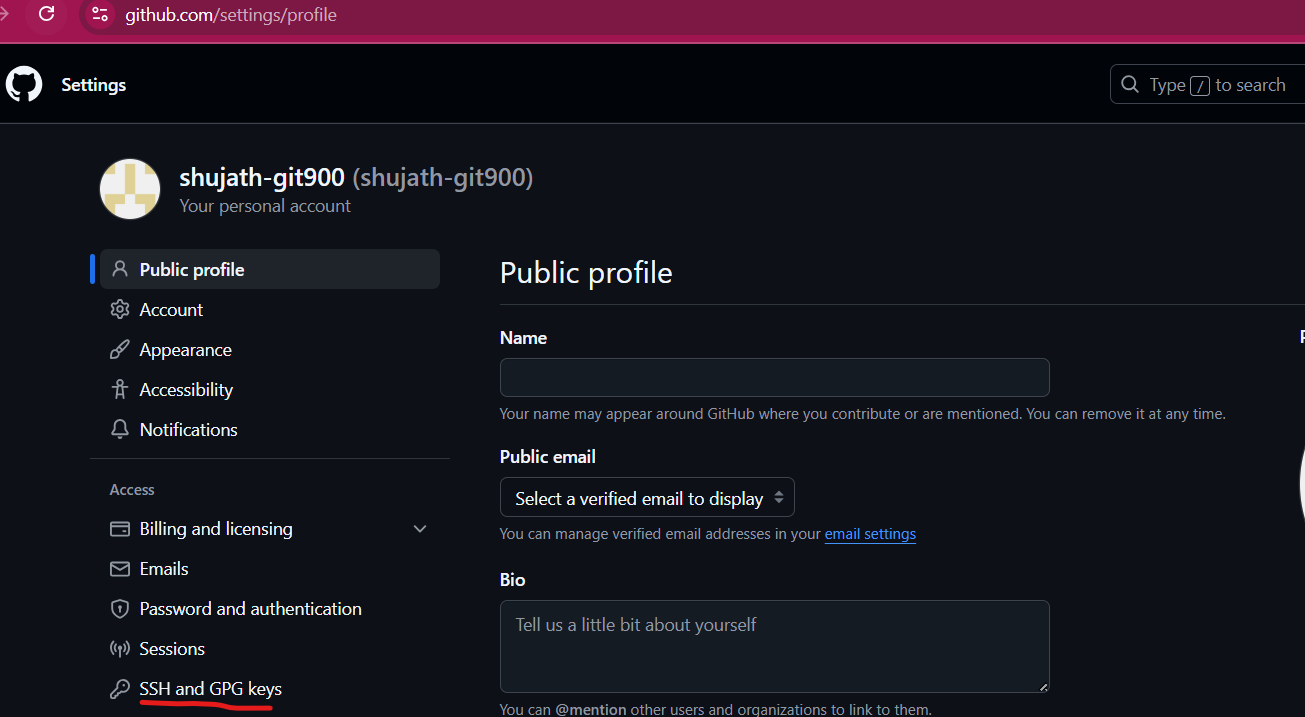


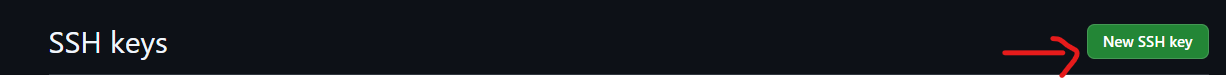
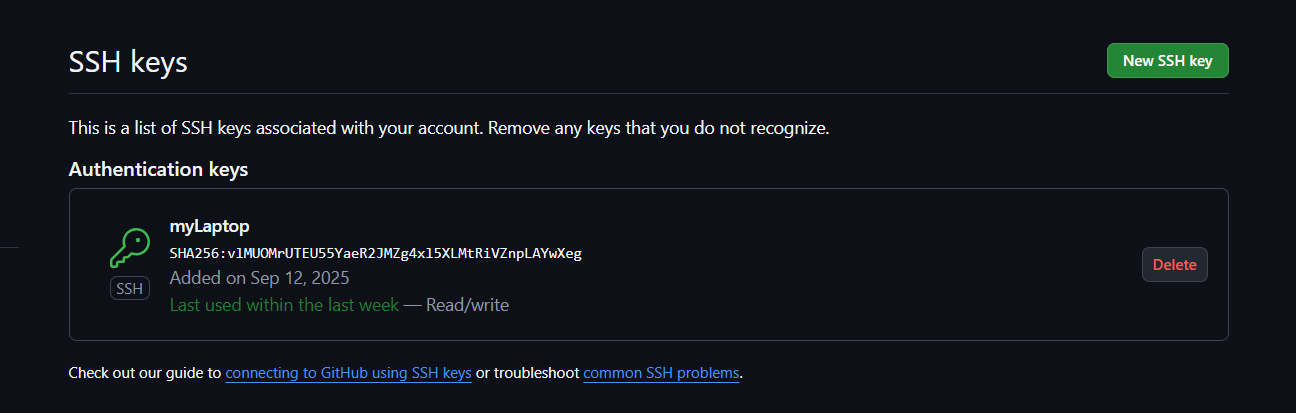
1. Create a file with all passwords and make that untrackable with git.
   * Create a file with name “password” with credentials:
   * 
   * Add the file ‘password’ into the .gitignore:
   * 
   * Do the git add:
   * 
   * Do the commit:
   * 
   * And push to the release-2.1 branch:
   * 
   * Verify the .gitignore file in the github:
   * 
2. Make a commit and make that commit reset without savings changes.
   * Create a file in the local repo:
   * 
   * 
   * Then do git add . :
   * 
   * Then commit :
   * 
   * Then we can see in the git log the commit id , now we want to reset the commit which we have done now : 
   * Then reset the commit:
   * 
   * Then we can see the commit has been deleted:
   * 
3. Revert a committed commit to the older version.
   * Create a file in local repo:
   * 
   * Do git add:
   * 
   * Then commit it.
   * Then see the commit id from git log
   * Then do the ‘git revert <commit id>’: This creates a **new commit** that undoes the changes introduced by commit e9f0e74



* + 

1. Push a file to stash without savings the changes and work on another file.
   * Create a file in local repo with name ‘index.htm’ and create content in it “Hello World”:
   * 
   * Then do the ‘git add . ‘:
   * 
   * Then send the file to stash:
   * 
   * Started Working on another file ‘ans.yml’:
   * 
2. Undo the stash file and start working on that again.
   * Undo the stash file means get back the file which was stashed i.e., index.html using ‘git stash pop’:
   * 
   * Start working on the index.html file again which was stashed previously:
   * 
3. Generate a ssh-keygen and configure into github.
   * Generate ssh-keygen:
   * 
   * Cat the public key:
   * 
   * Goto github and click on Settings:
   * 
   * Then select the ‘SSH and GPG keys’



* + Then click on New SSH key:
  + 
  + And give the Name and paste the key:
  + 

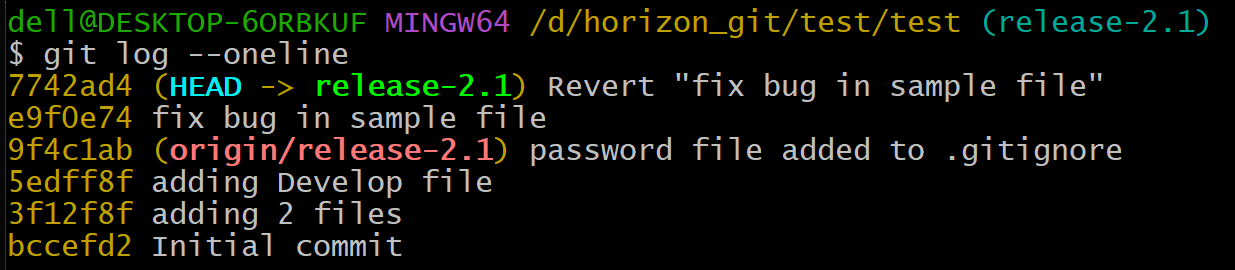
1. Configure webhooks to github.
2. Basic understanding of .git file.
   * + When you run git init in a project, Git creates a hidden folder called **.git** at the root.
     + This folder is the **local database of Git**.
     + It stores **everything Git needs**: commit history, branches, tags, configuration, and objects.
     + That’s why deleting .git makes your project “forget” it’s a Git repo.
3. Check all the logs of git.
   * 1. Commit history log

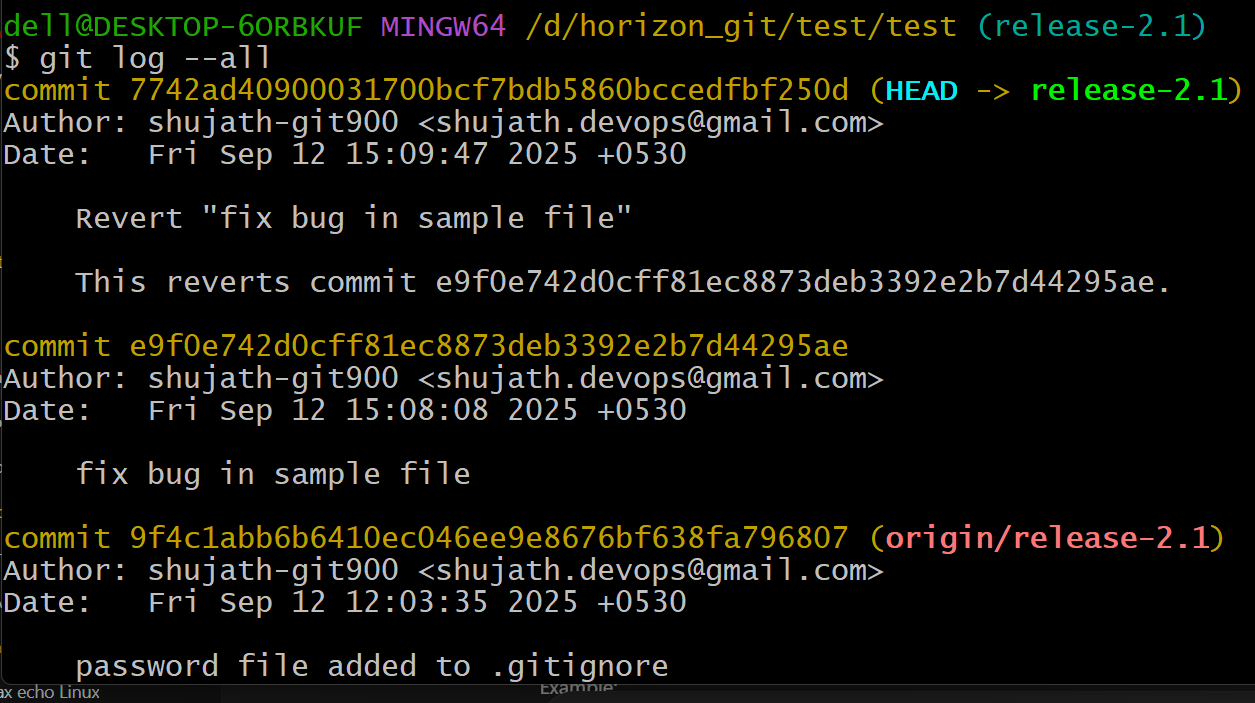
git log

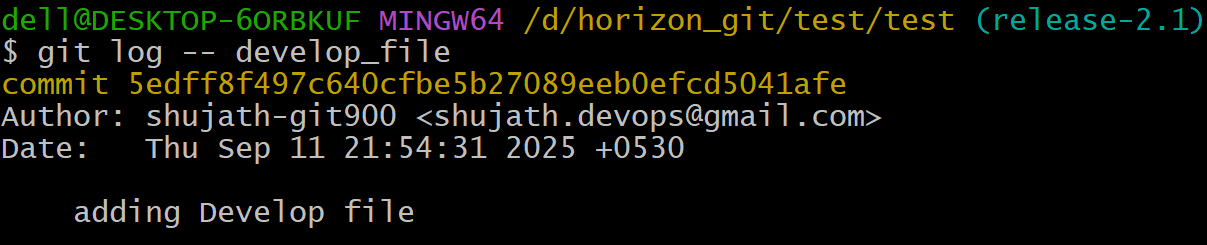
Shows all commits in the repo (by default, from your current branch):

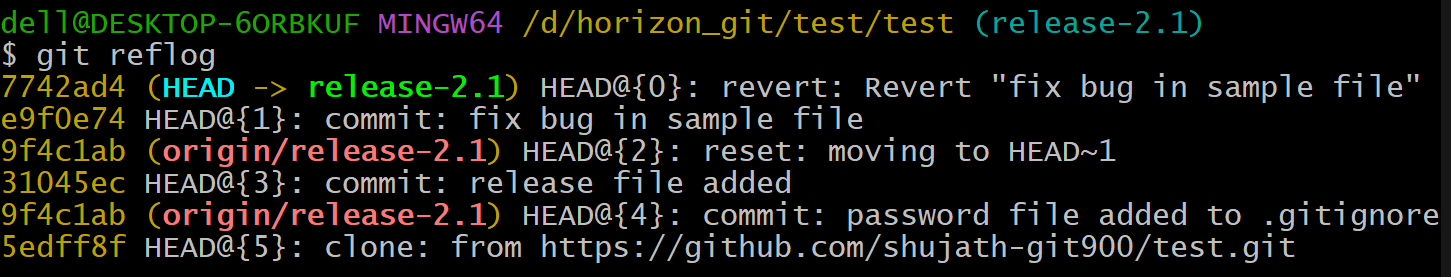
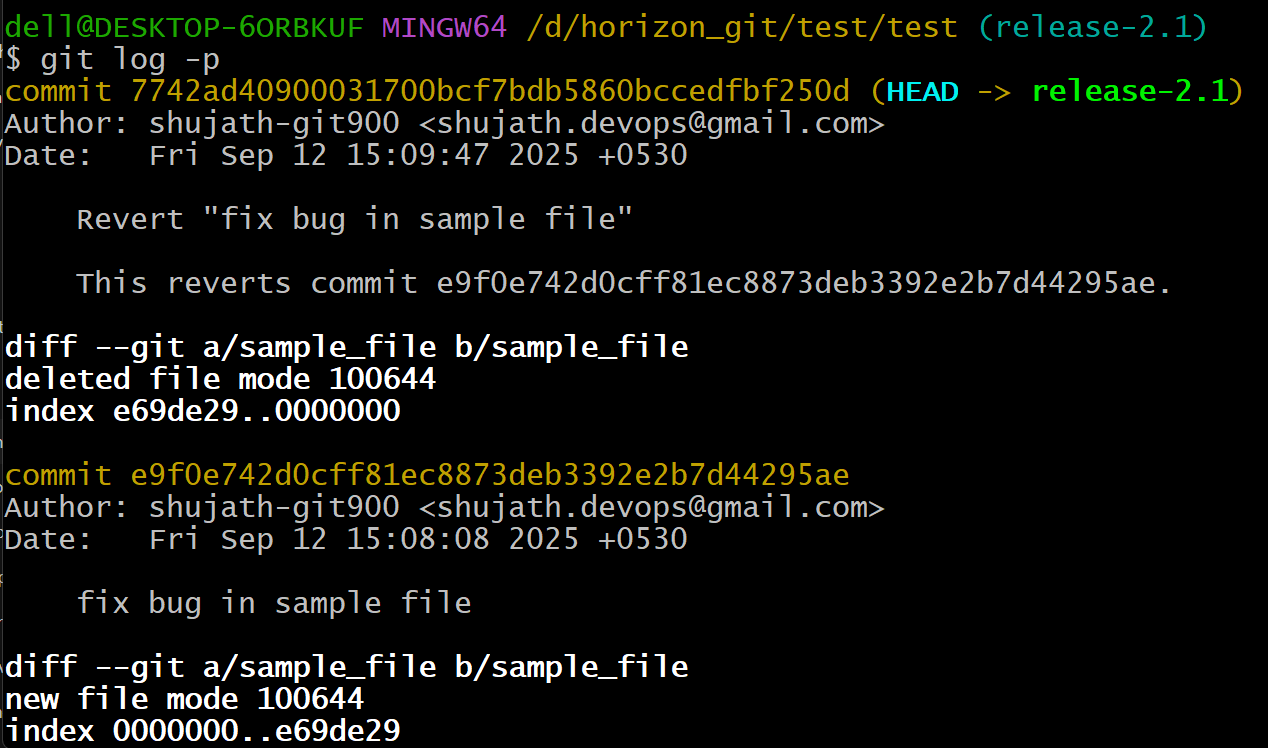
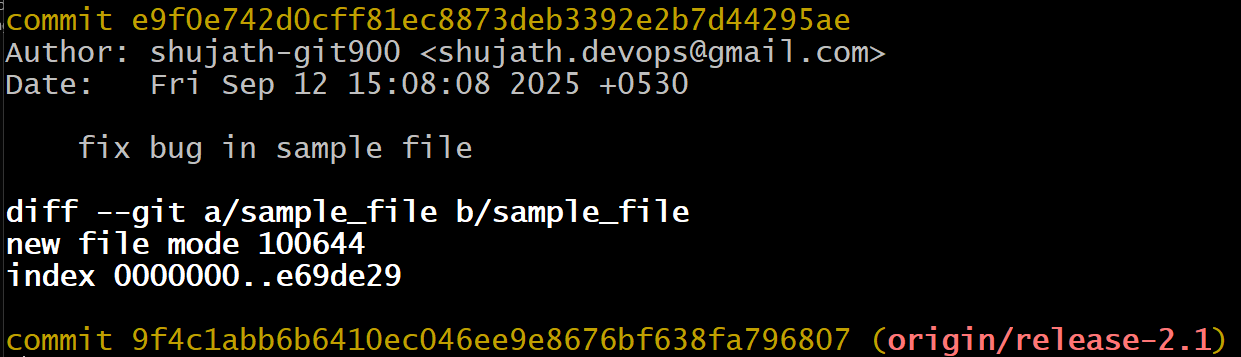
Compact view:

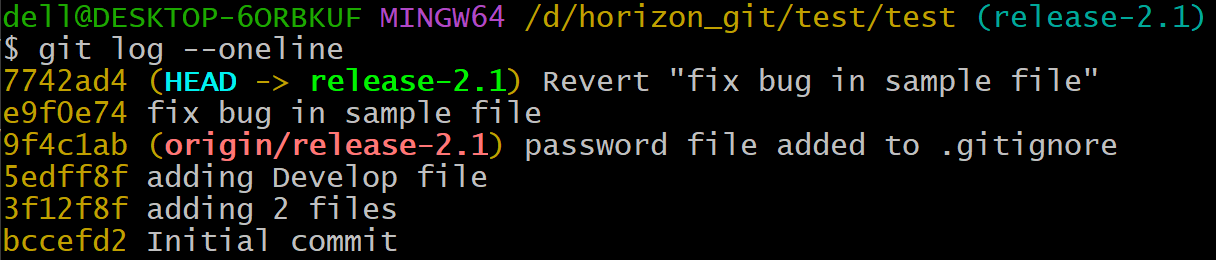
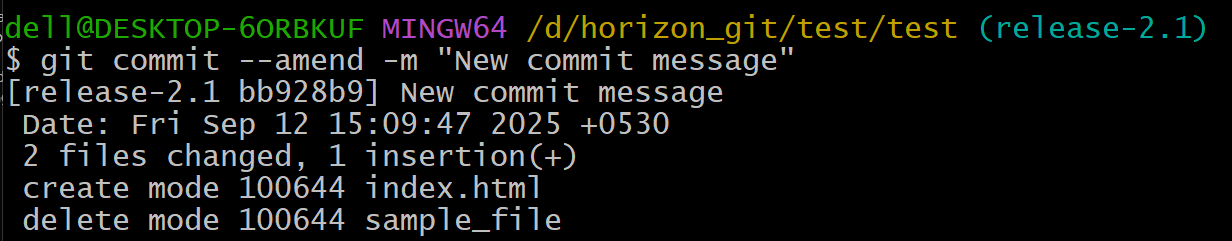
git log - -oneline

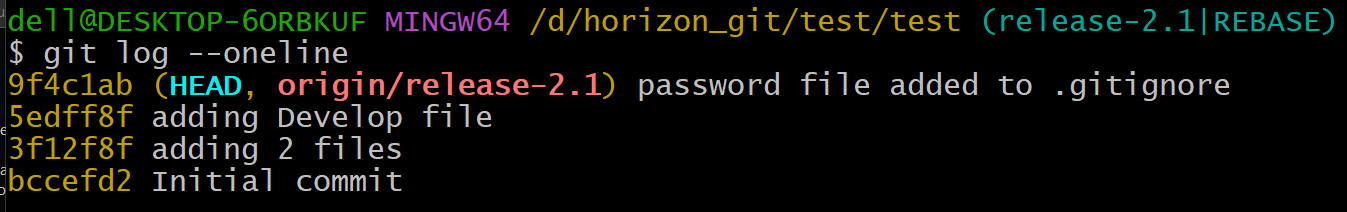


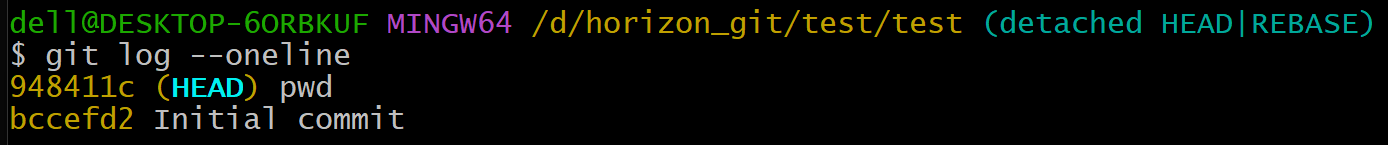
* + 2. Show commits across **all branches**
  + 
  + 3. Show who changed what in a file (history of a file): git log - -<filename>



* + 4. Reflog (local history of HEAD): This shows every moment of HEAD (even resets, checkouts etc.,)
  + 
  + 5. Stash Logs: you can see git stash logs with ‘ git stash list’:
  + 6. Shows patch/diffs in log:
  + 
  + 

1. Rename the commit message.
   * Case 1: Rename the **last commit** message
   * Check the commit message with git log:
   * 
   * Change the commit message with git commit - -ammend
   * 
   * Rename an older commit (not the last one):
2. Merge multiple commits into single commit.
   * git log and I want to merge the 2 commits into a single commit
   * git rebase -i HEAD~n
   * (where n = number of commits back you want to edit)





--------------------completed-------------------------------------------------------------