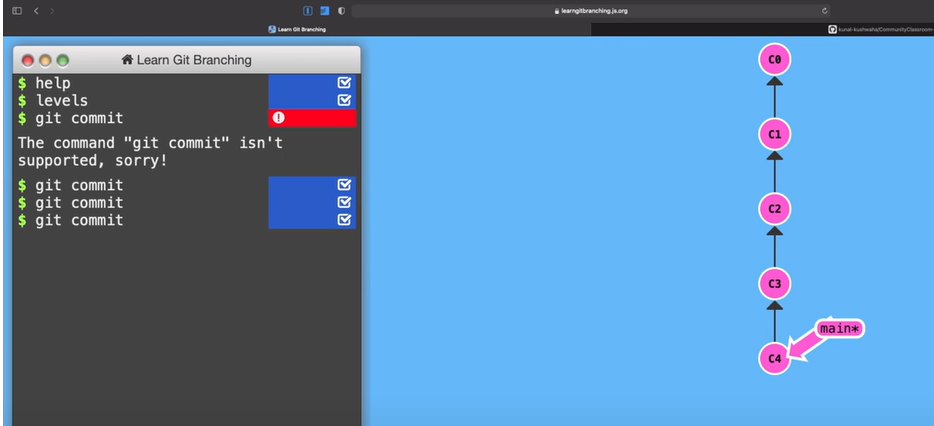
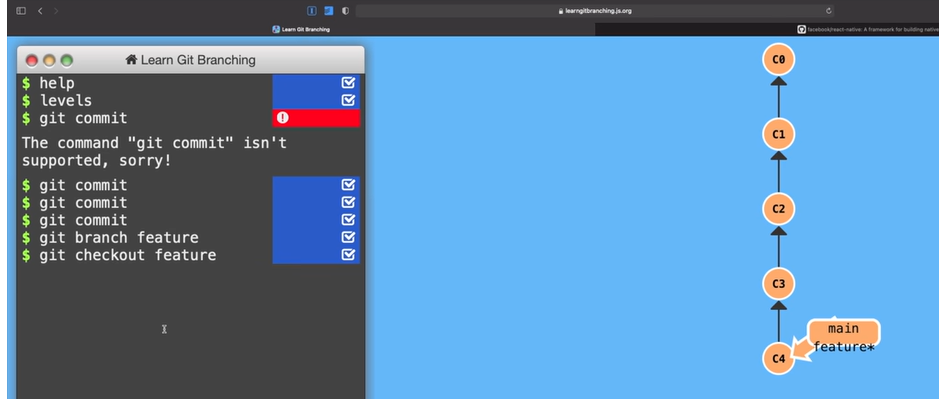
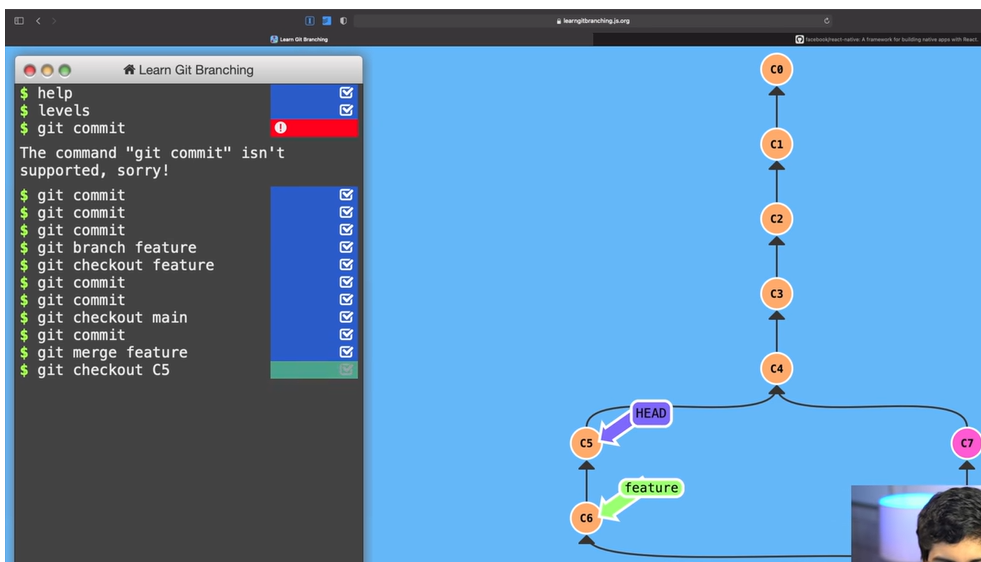
Branching : whenever you want to add any thing to repository always make branch





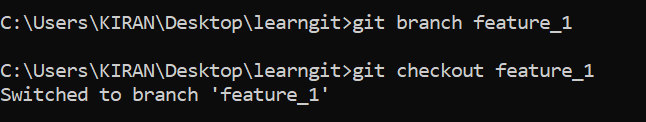


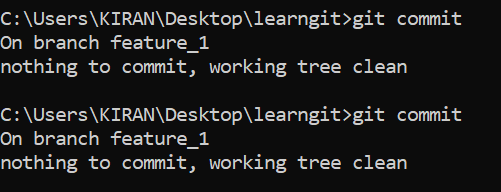


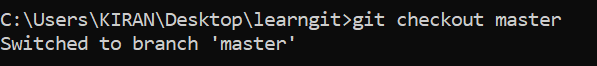
# Topic 13 : Create a new Branch

C:\Users\KIRAN\Desktop\learngit>git branch feature\_1

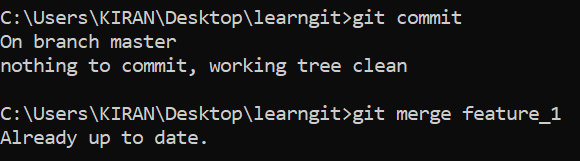
C:\Users\KIRAN\Desktop\learngit>git checkout feature\_1

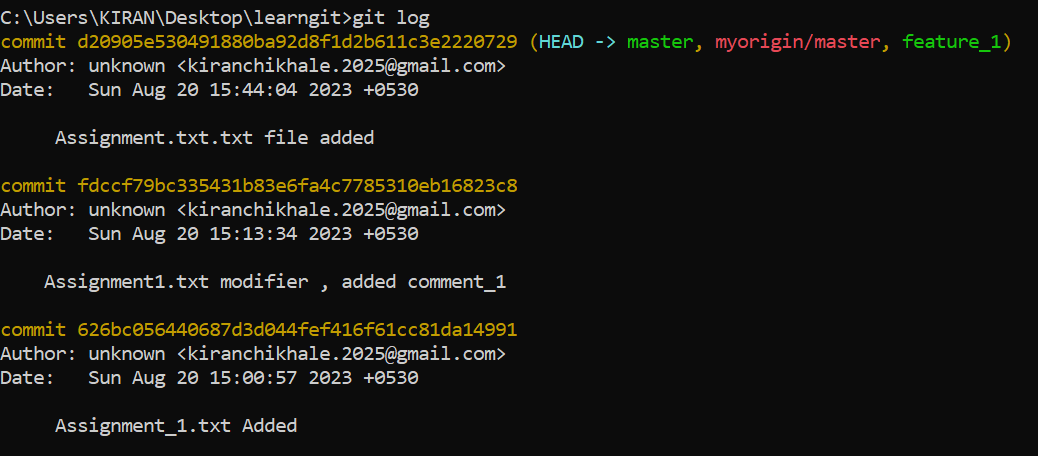




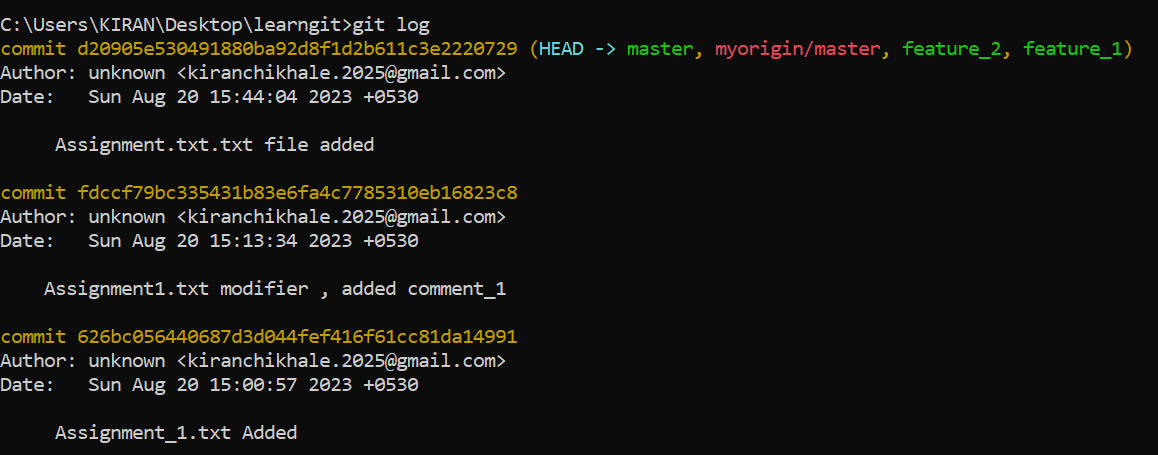


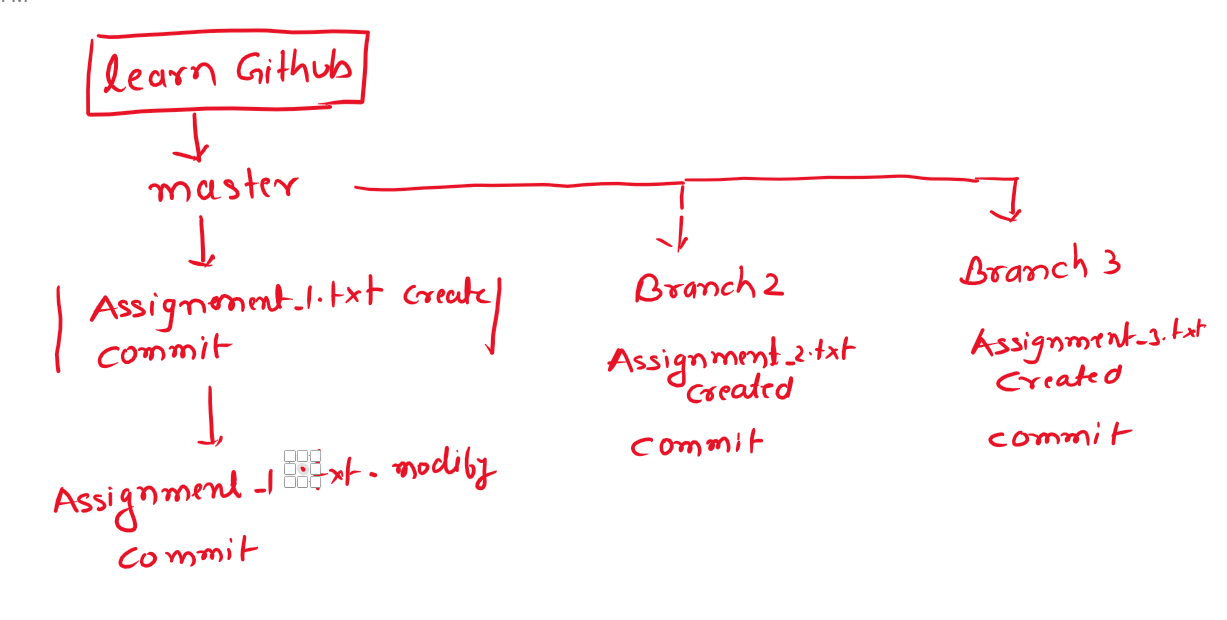
C:\Users\KIRAN\Desktop\learngit>git merge feature\_1

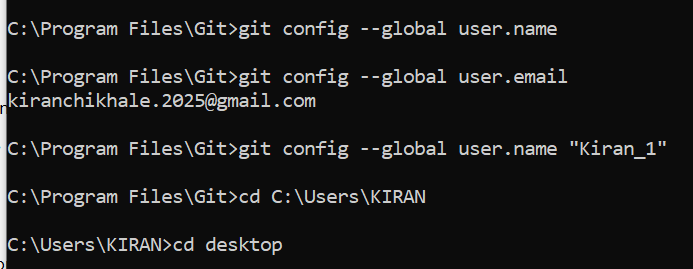




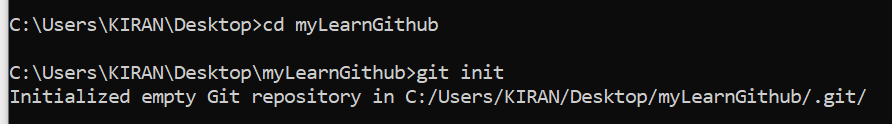
\

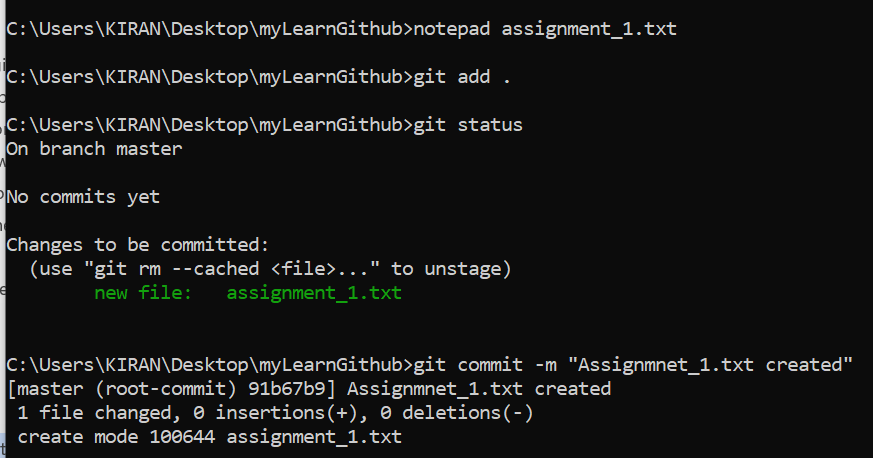


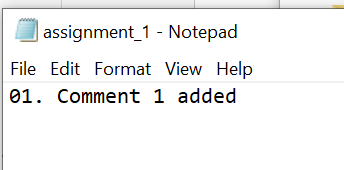


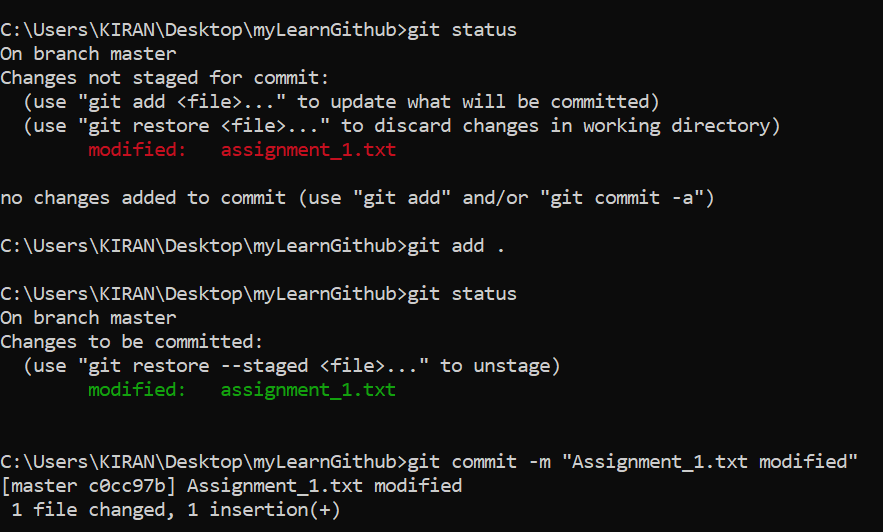


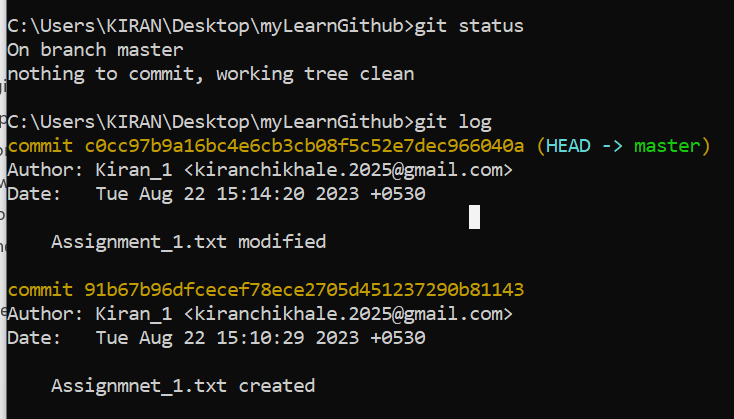
|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  | C:\Users\KIRAN\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image002.png |  |  |
|  |  | C:\Users\KIRAN\AppData\Local\Packages\Microsoft.Office.OneNote_8wekyb3d8bbwe\TempState\msohtmlclip\clip_image003.png |
|  |  |  |

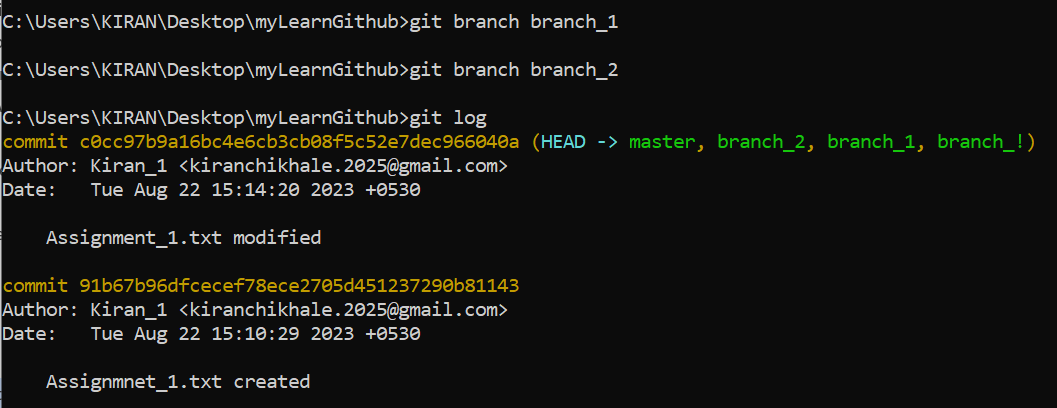


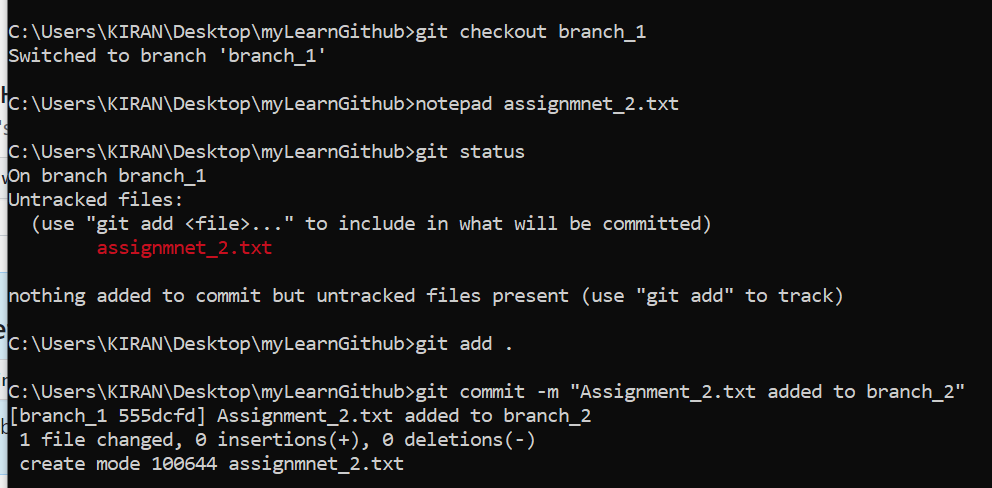


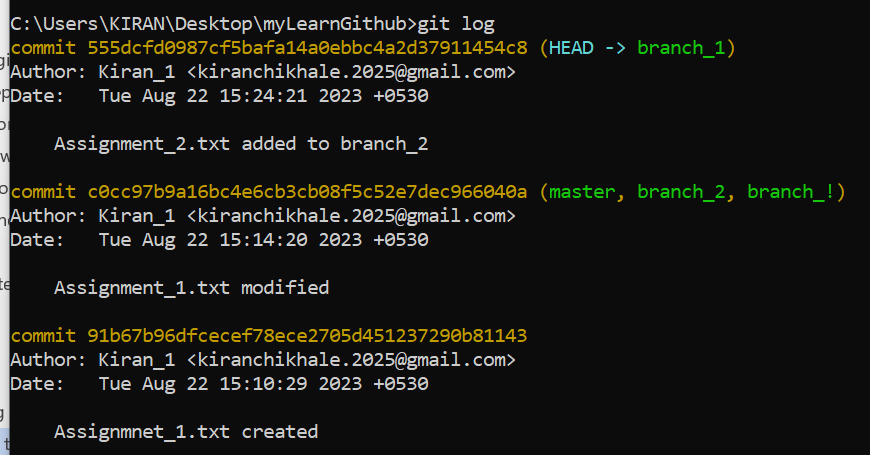


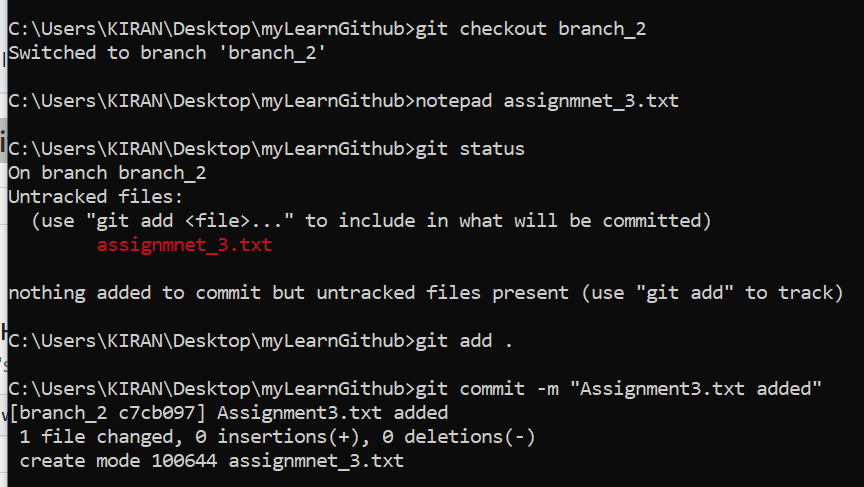


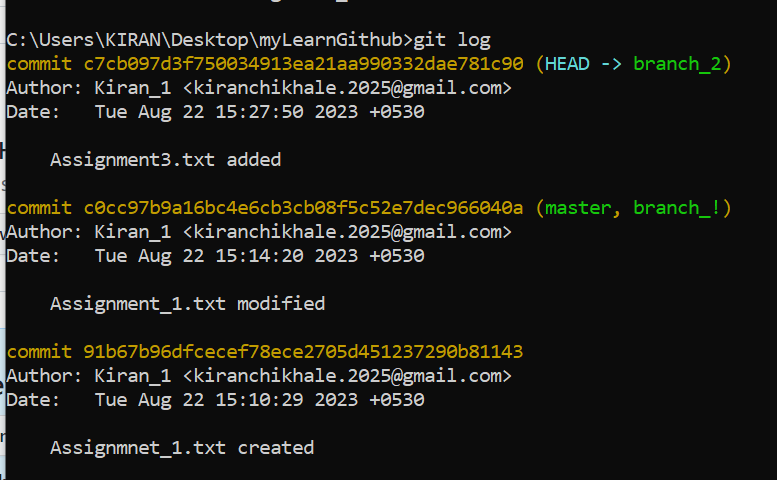


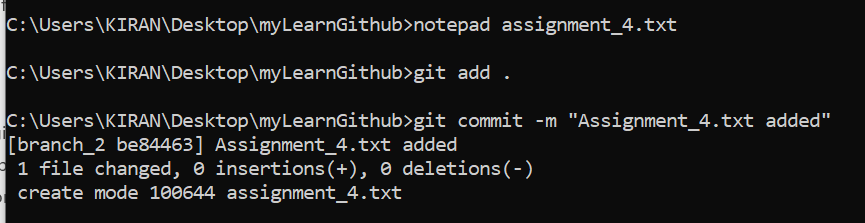


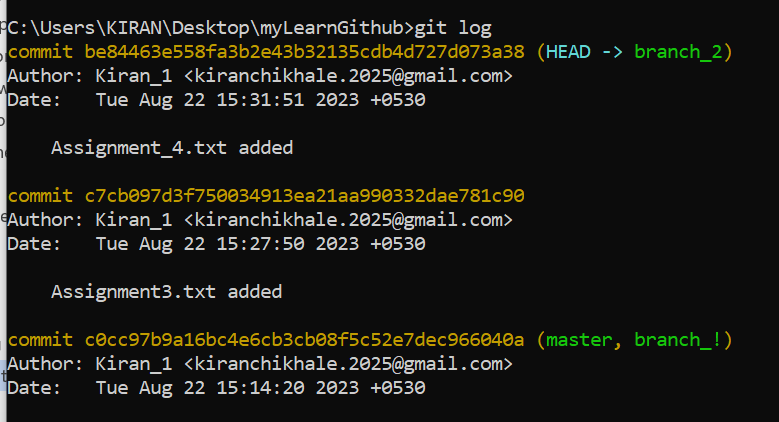


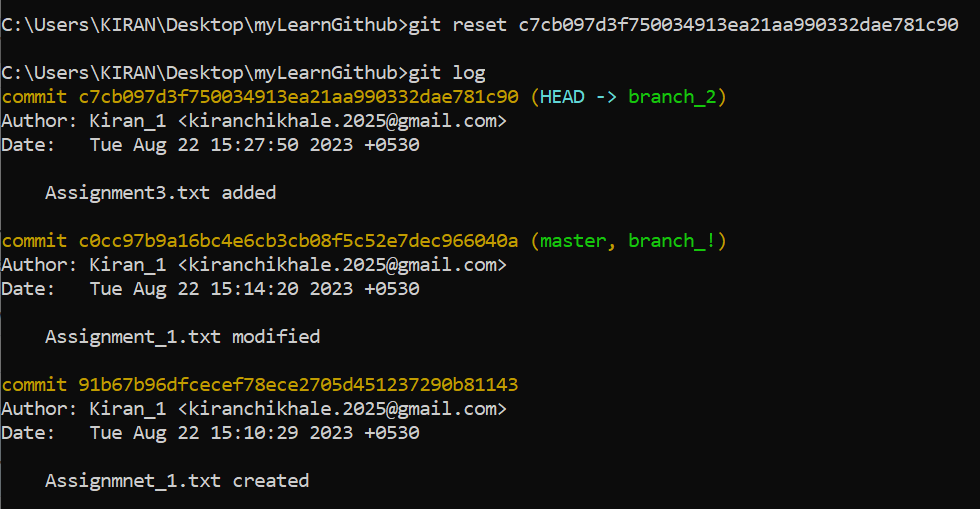


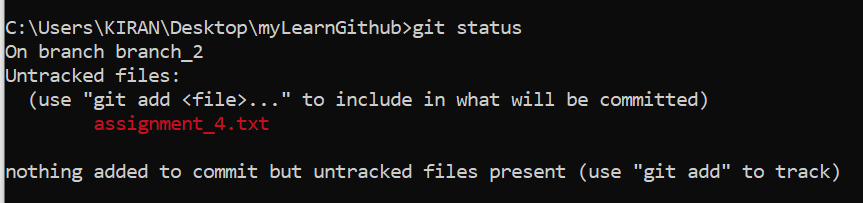


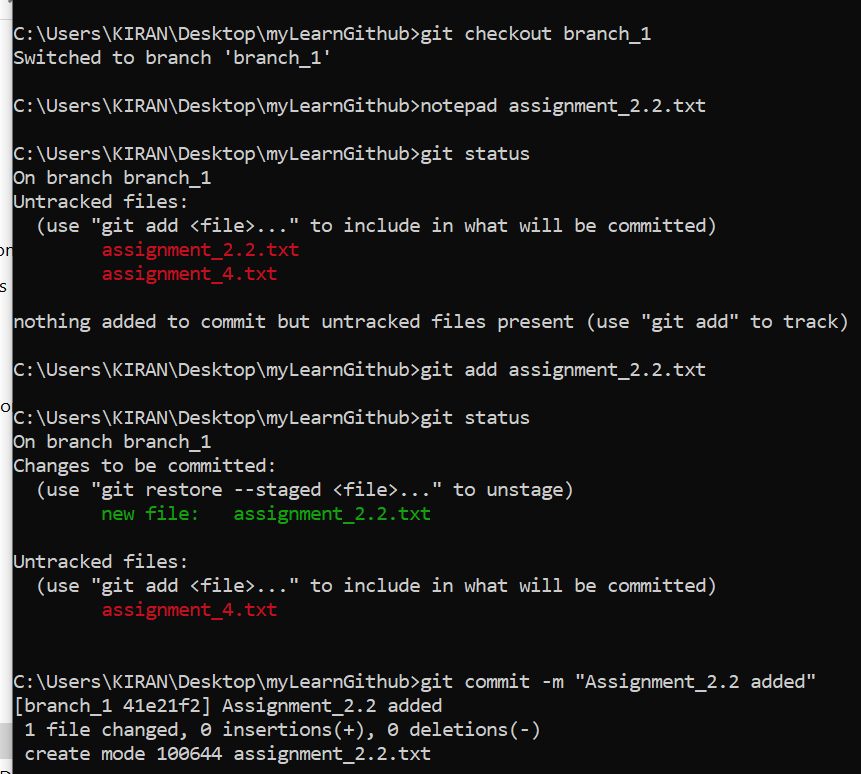


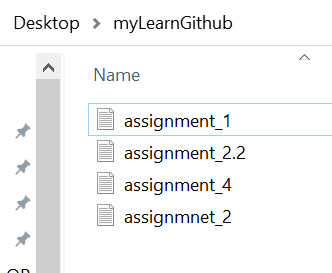


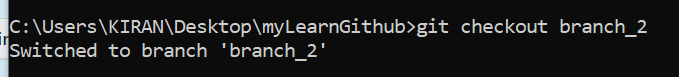


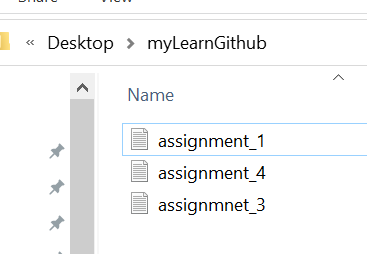


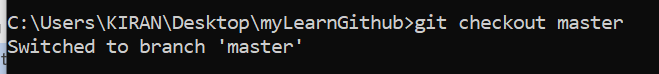


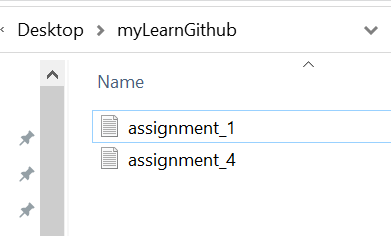


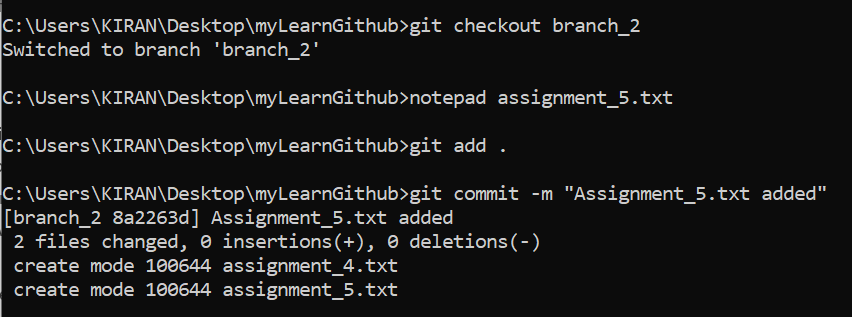


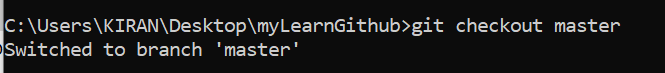


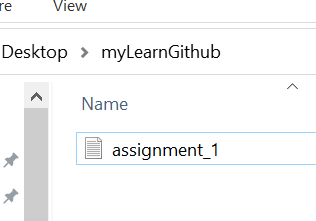


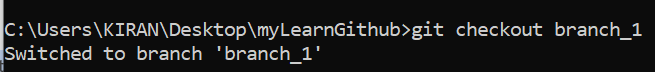


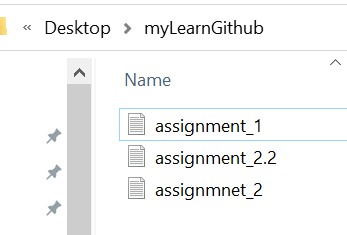


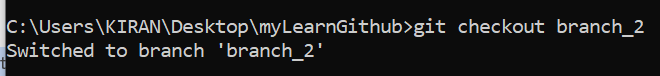


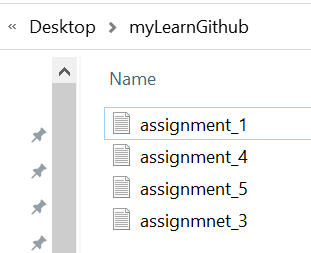


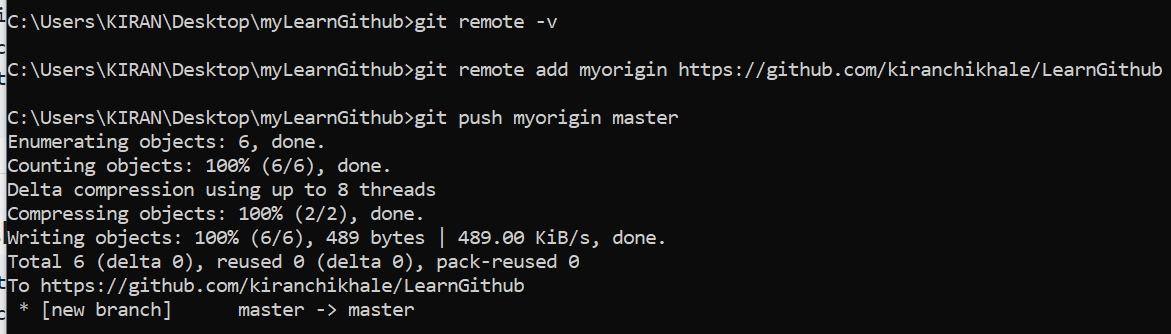


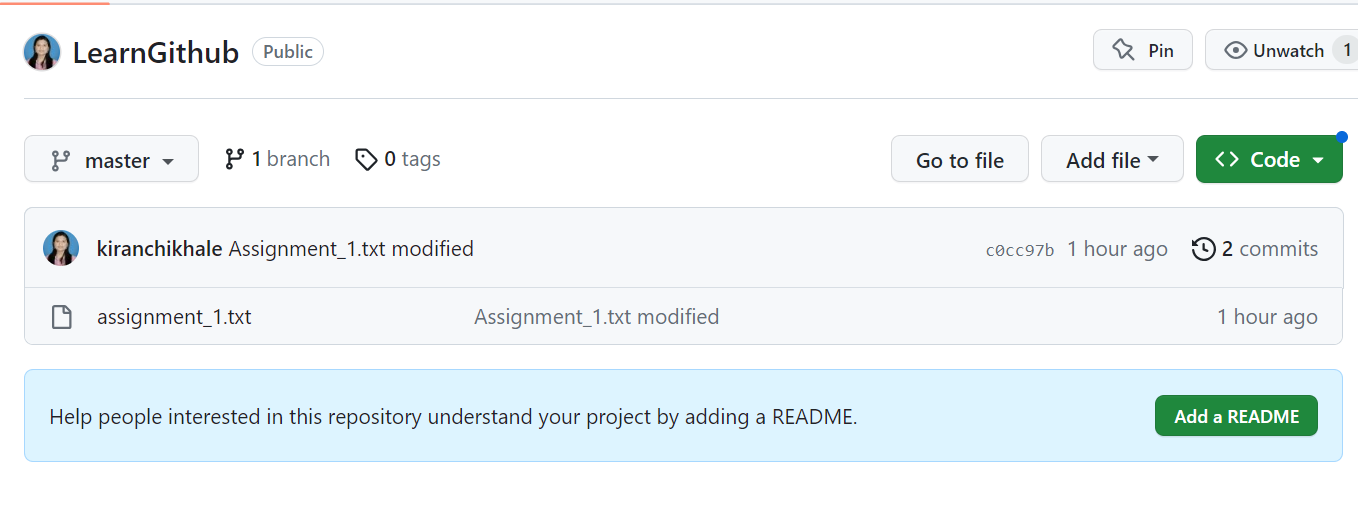


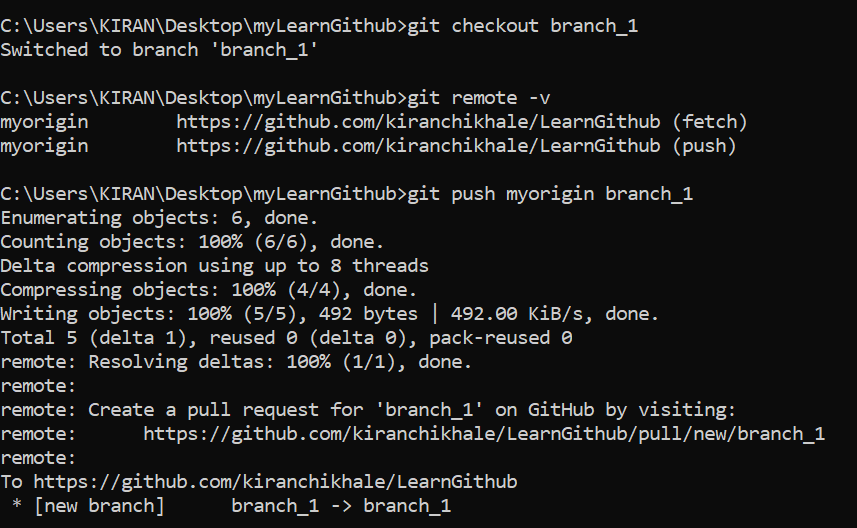


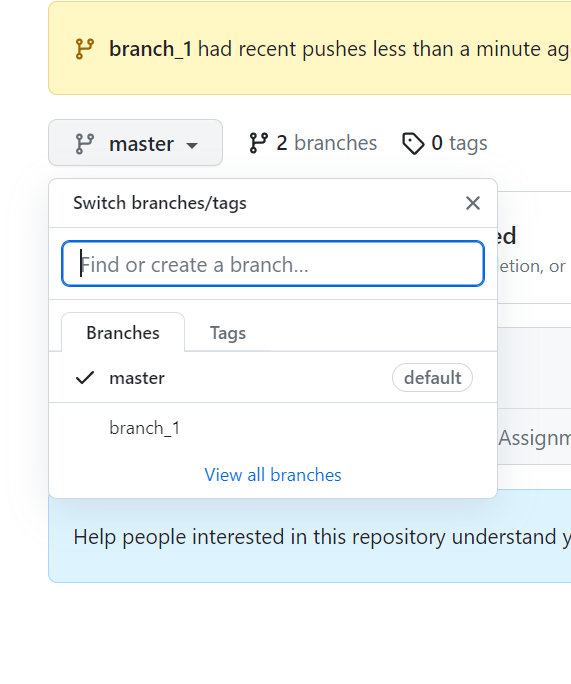


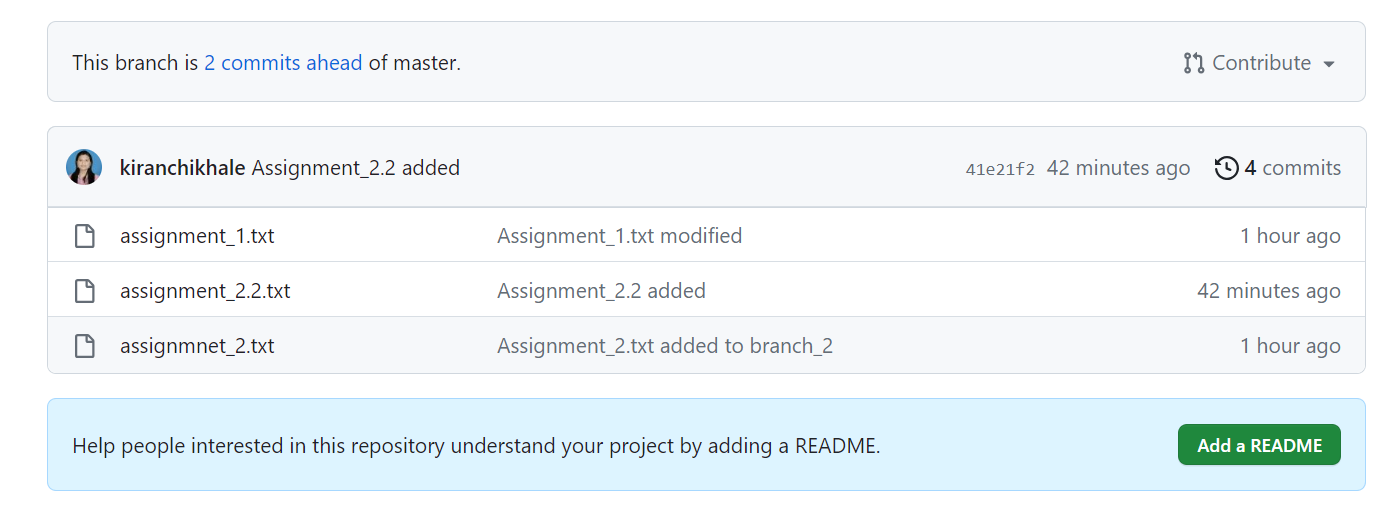


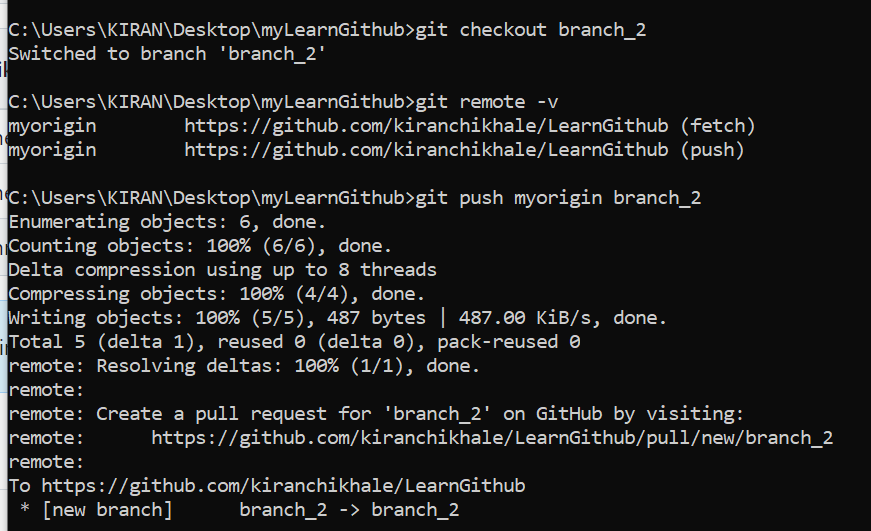


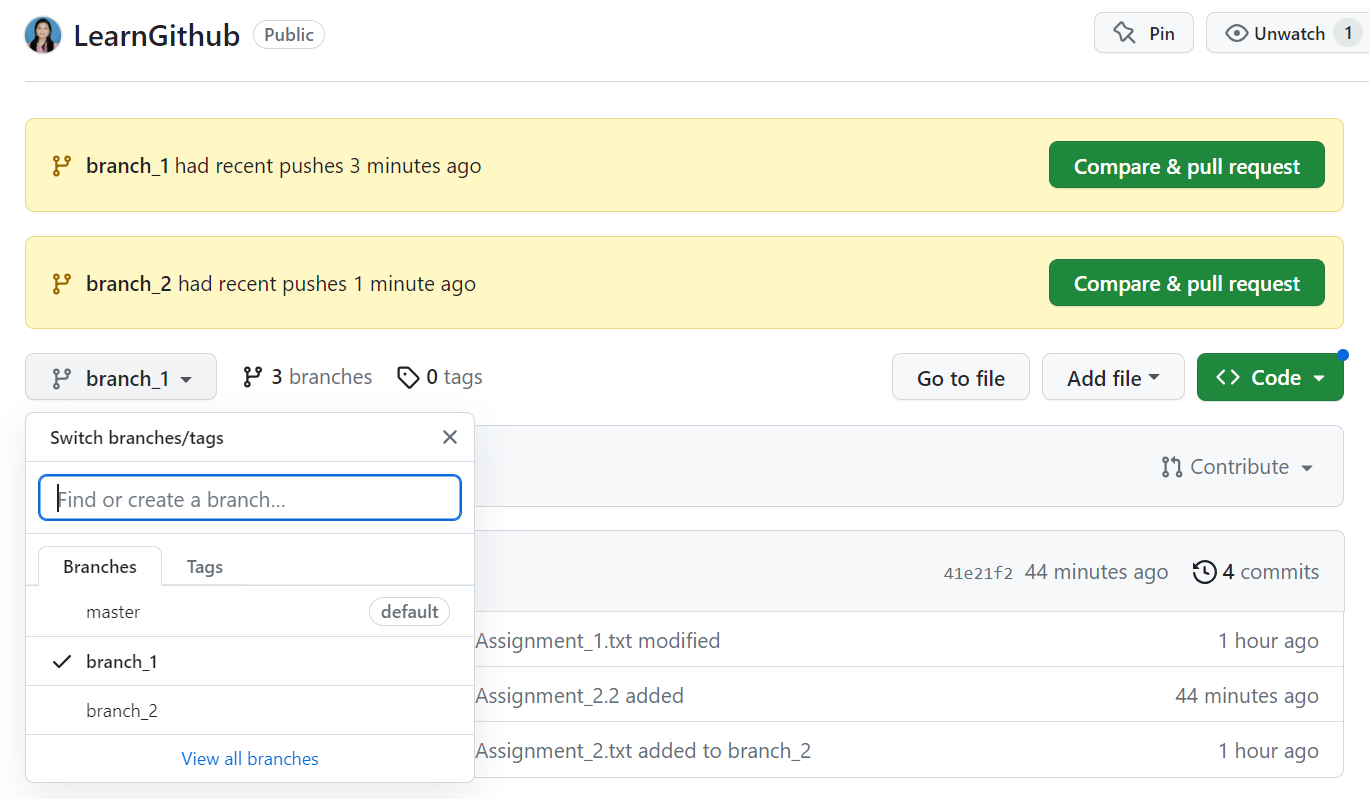


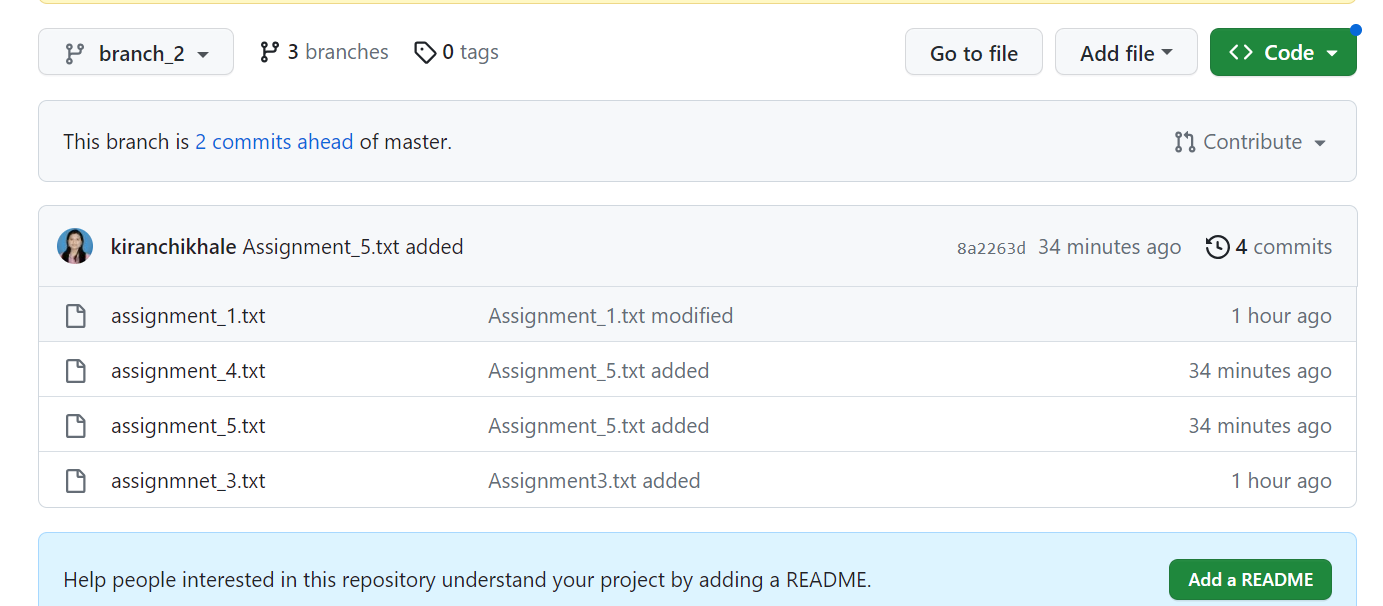






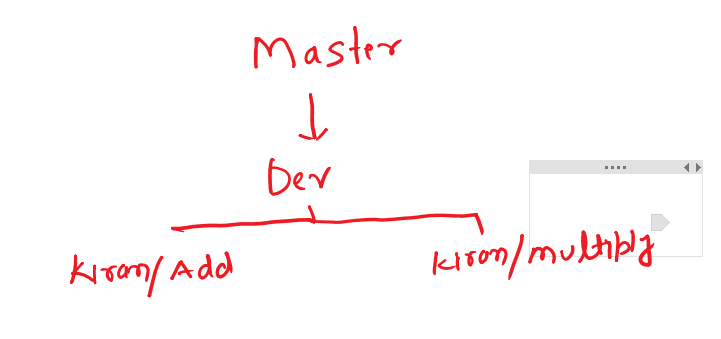






This is how it happen in production





Creation of the branch and checkout happen in one command

Git checkout –b anuj/multiply

Choosing a consistent and meaningful naming convention for branches in your GitHub repository is crucial for maintaining clarity and organization within your development process. Here's a recommended naming convention:

1. **Main/Branch:**
   * **main**: The primary branch that reflects the production-ready code.
2. **Feature Branches:**
   * **feature/descriptive-name**: Used for developing new features. Replace **descriptive-name** with a short, clear description of the feature.
3. **Bug Fix Branches:**
   * **bugfix/issue-number**: Created to fix specific issues. Use the associated issue number for clear reference.
4. **Release Branches:**
   * **release/version-number**: Created to prepare for a new release. Use the version number of the release.
5. **Hotfix Branches:**
   * **hotfix/issue-number**: Created for urgent fixes in the production code. Use the associated issue number.
6. **Topic Branches:**
   * **topic/descriptive-name**: Used for broader topics or experiments. Replace **descriptive-name** with a clear description.

Remember, the naming convention should be consistent across the team to avoid confusion. It's also a good idea to avoid spaces and special characters in branch names, as they might cause issues with certain tools and systems.

Additionally, consider including a short prefix to categorize branches (like **feature/**, **bugfix/**, etc.), which can make it easier to differentiate between the types of branches when working with them.

git checkout -b main

git push -u origin main

git checkout -b feature/descriptive-name

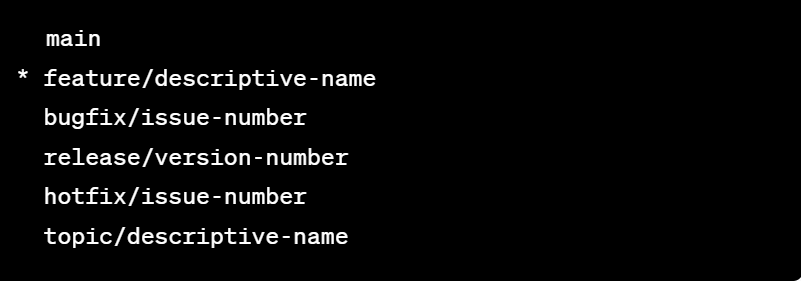
git checkout -b bugfix/issue-number

git checkout -b release/version-number

git checkout -b hotfix/issue-number

git checkout -b topic/descriptive-name

git branch



To see all the branches

Git branch

To delete the branch

To delete a branch in Git, you can use the **git branch** command with the **-d** or **-D** option followed by the branch name. Here are the two options:

1. **Delete a Merged Branch (Safe):** To delete a branch that has been merged into the current branch, you can use the **-d** option. This option ensures that the branch is fully merged before deletion.

bashCopy code

git branch -d branch-name

Replace **branch-name** with the name of the branch you want to delete.

1. **Force Delete an Unmerged Branch:** If you want to forcefully delete a branch, even if it's not fully merged, you can use the **-D** option. Be cautious with this option, as it will delete the branch without checking for merge status.

bashCopy code

git branch -D branch-name

Again, replace **branch-name** with the name of the branch you want to delete.

Remember to be careful when deleting branches, especially unmerged ones, as you might lose work that hasn't been merged into other branches.

Additionally, please note that the above commands only remove the branch locally. If you want to delete a branch on a remote repository (like GitHub), you will need to use the appropriate remote commands, such as **git push origin --delete branch-name**.

Always double-check your actions before deleting branches, as Git does not provide an automatic way to recover deleted branches once they're gone.

**Visualize Branches:** If you want to visualize the branches and their relationships, you can use graphical tools like Git GUI or GitKraken

To merge the branch: first checkout into the branch where you want to merge

Git merge branchname

