

Git and GitHub for Test Automation:

A Practical Self-Explanatory Guide

What Are Git and GitHub?

- **Git:** A tool that keeps track of changes in your code and test scripts. It allows you to save different versions of your work and go back to older ones if needed.
 - **GitHub:** A website where you can store and share your Git projects with others. It's great for collaboration and working with a team.
-

Step 1: Installing Git

To use Git:

1. **Download Git:**

I went to git-scm.com and downloaded the latest version for my operating system.

2. **Install Git:**

I ran the installer and kept the default options.

3. **Check Installation:**

After installation, I opened the terminal (or Command Prompt) and typed:

```
git --version
```

```
C:\Users\Yogi>git --version
git version 2.39.1.windows.1
```

If a version number appeared, it meant Git was installed successfully.

Step 2: Setting Up Git

Before using Git, I configured my name and email. This information is needed to identify who made changes.

1. **Set Your Name:** `git config --global user.name "Your Name"`

2. **Set Your Email:**

```
git config --global user.email your.email@example.com
```

```
C:\Users\Yogi>git config --global user.name "Yogi"
C:\Users\Yogi>git config --global user.email "yogeshpandian97@gmail.com"
```

3. **Check Your Settings:**

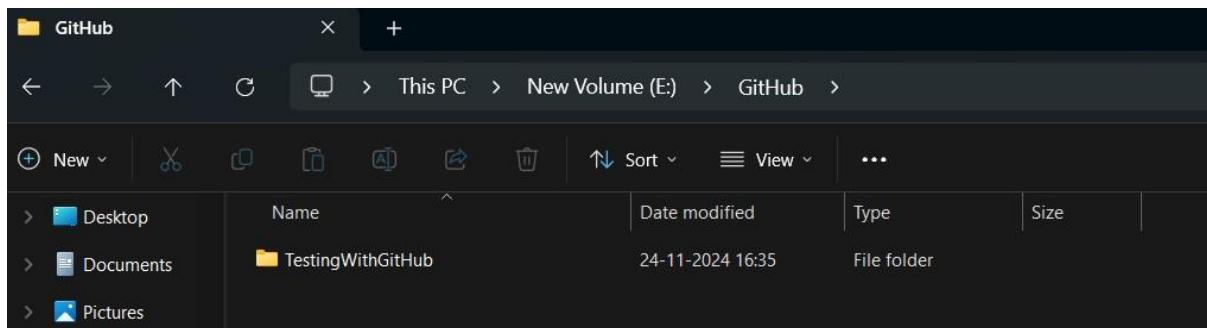
```
git config --list
```

```
user.name=Yogi  
user.email=yogeshpandian97@gmail.com  
core.editor=idea --wait
```

This shows the name and email I just set.

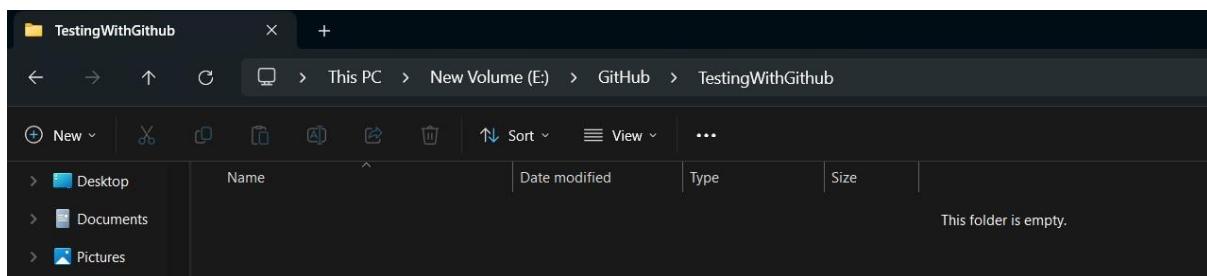
Step 3: Creating a Local Project

I created a new folder on my system called **TestingWithGitHub** to store my test scripts. Then, I opened this folder in IntelliJ IDEA so I could work on it directly.

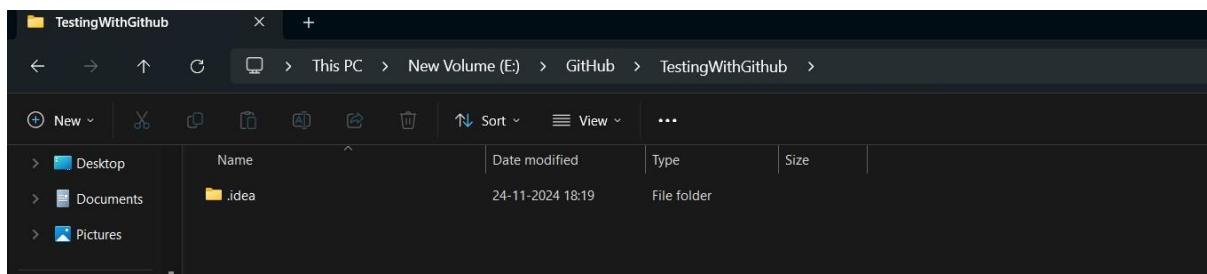


Next, I opened **IntelliJ IDEA**, navigated to **File > Open**, and selected the **TestingWithGitHub** folder. This made it easy to work within the folder directly from IntelliJ.

Before



After



To start using Git commands, I opened the built-in terminal in IntelliJ. I navigated to the path where the **TestingWithGitHub** folder was located using the cd command:

```
PS E:\GitHub> cd TestingWithGitHub  
PS E:\GitHub\TestingWithGitHub>
```

Step 4: Creating a GitHub Repository

Next, I logged into my GitHub account and created a new repository named TestingGitRepo. In this repository, I added a file called **TestInfo.txt** with the following content:

This file contains information about test automation:

- Tools: Selenium, Postman
- Frameworks: TestNG, Cucumber

The image consists of three vertically stacked screenshots of the GitHub web interface, all showing the same repository: AutomatedMind / TestingGitRepo.

- Screenshot 1 (Top):** Shows the repository's landing page. It features a "Get started with GitHub Copilot" button and a "Add collaborators to this repository" section.
- Screenshot 2 (Middle):** Shows the repository's code view. A message at the top states "AutomatedMind created TestInfo text file from GITHUB repo". The code editor shows the content of the TestInfo.txt file:

```
1 This file contains information about testing automation.
2 - Tools: Selenium, Postman
3 - Frameworks: TestNG, Cucumber
```
- Screenshot 3 (Bottom):** Shows the repository's main page again. It displays the file structure (TestInfo.txt, README), a "Clone" button with options for HTTPS, SSH, or GitHub CLI, and links for opening with GitHub Desktop or downloading as a ZIP. The "About" section states: "This repository is created to demonstrate and practice GitHub workflows."

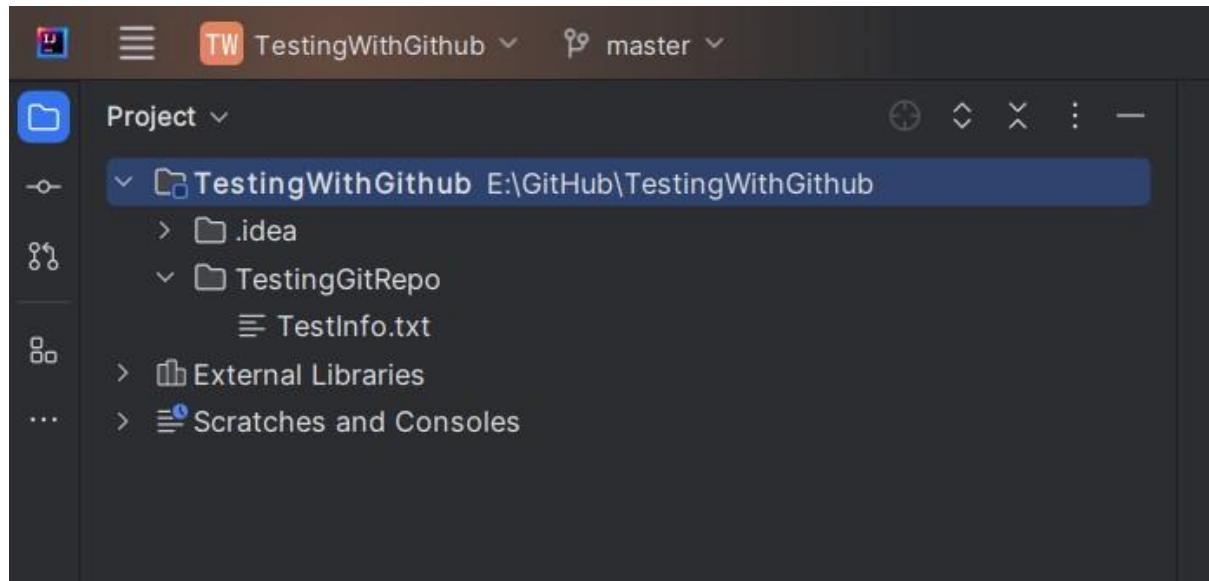
GitHub provided a URL for this repository, which I copied.

Step 5: Cloning the Repository

To connect my local project with the GitHub repository, I used the **clone** command. In the terminal, I typed: `git clone <GitHub-Repo-URL>`

```
PS E:\GitHub\TestingWithGitHub> git clone https://github.com/AutomatedMind/TestingGitRepo.git
Cloning into 'TestingGitRepo'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (3/3), done.
PS E:\GitHub\TestingWithGitHub>
```

This command downloaded the GitHub repository into my local system. Now, I could see the **TestingGitRepo** folder and the **TestInfo.txt** file inside IntelliJ.



This copied all the files from the GitHub repository into my local folder.

Step 6: Adding and Committing Files

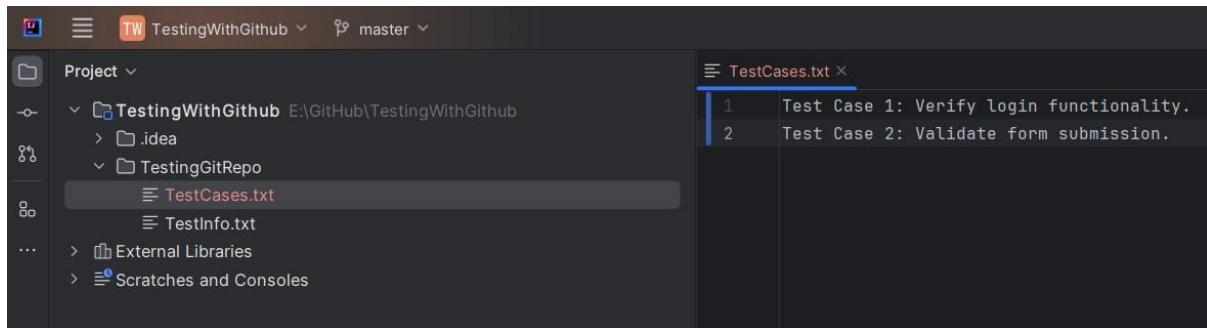
I navigated into the newly cloned repository: `cd TestingGitRepo`

```
PS E:\GitHub\TestingWithGitHub> cd TestingGitRepo
PS E:\GitHub\TestingWithGitHub\TestingGitRepo>
```

In IntelliJ, I created a new file named **TestCases.txt** with this content:

Test Case 1: Verify login functionality.

Test Case 2: Validate form submission.



The red line of the file indicates the file is untracked which means it is not in git

I saved the file, then checked its status with:

```
git status
```

```
PS E:\GitHub\TestingWithGitHub\TestingGitRepo> git status
On branch master
Your branch is up to date with 'origin/master'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    TestCases.txt

nothing added to commit but untracked files present (use "git add" to track)
PS E:\GitHub\TestingWithGitHub\TestingGitRepo>
```

The file showed as "untracked," meaning Git was not tracking it yet.

Staging the File

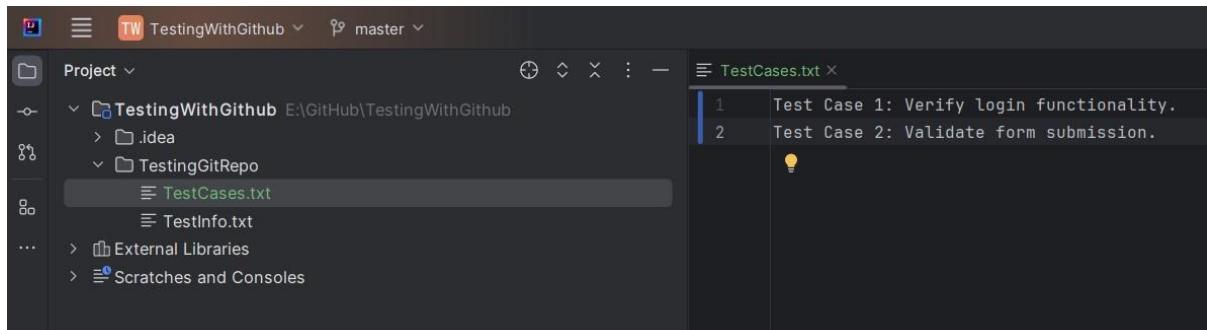
To tell Git to start tracking the file, I staged it using the following command: git

```
add TestCases.txt
```

```
PS E:\GitHub\TestingWithGitHub\TestingGitRepo> git add TestCases.txt
PS E:\GitHub\TestingWithGitHub\TestingGitRepo>
```

This moved the file to the **staging area**, which is where changes are prepared before being committed.

Now the colour of the file changed to green which indicates it is staged now



To confirm check the git status

```
git status
```

```
PS E:\GitHub\TestingWithGitHub\TestingGitRepo> git status
On branch master
Your branch is up to date with 'origin/master'.

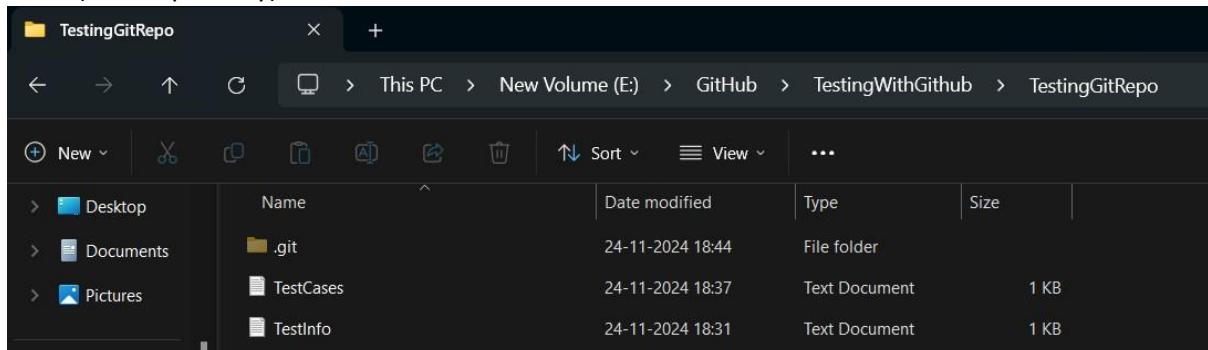
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   TestCases.txt

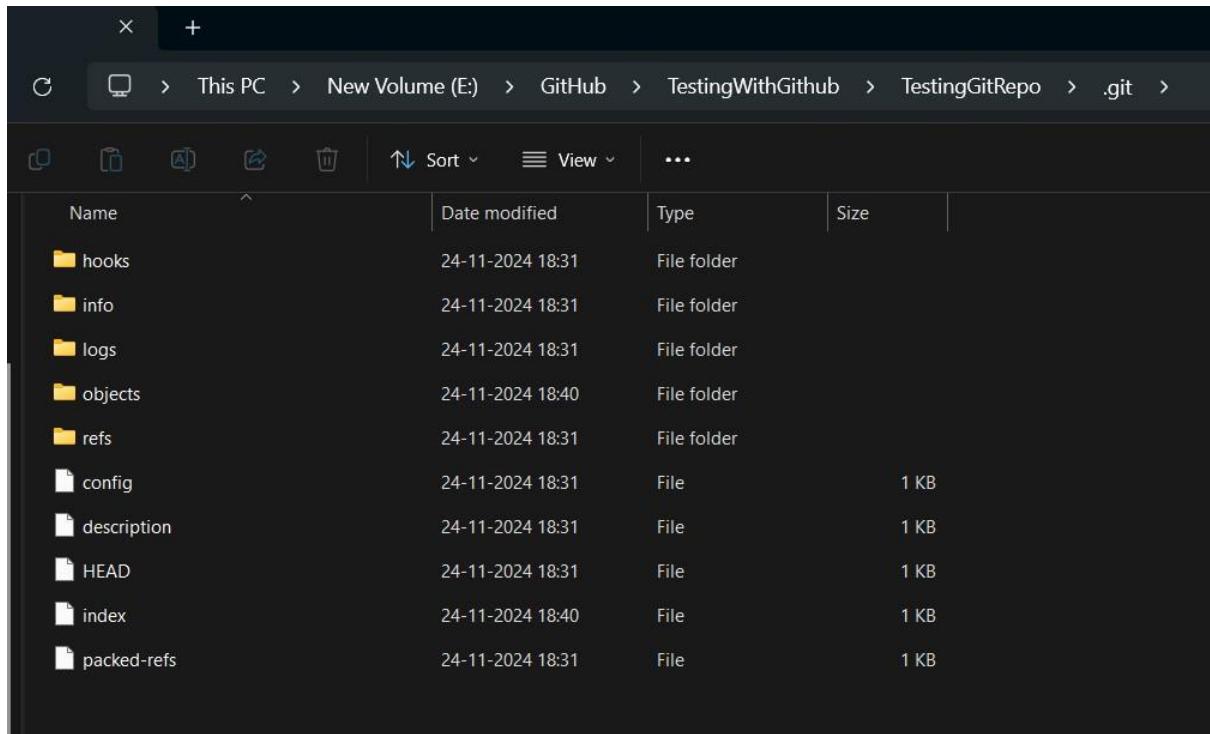
PS E:\GitHub\TestingWithGitHub\TestingGitRepo>
```

The file **TestCases.txt** is now in the staging area, which means it is ready to be committed to the local repository.

Additional Point: When we refer to the local repository, it is stored in the .git folder, which is created when we clone or initialize a repository. This folder is usually hidden in the project directory.

Once the **TestCases.txt** file is committed, it will be saved as part of the version history in the .git folder (local repository).





A screenshot of a file explorer window showing the contents of a .git folder. The path is C:\This PC\New Volume (E):\GitHub\TestingWithGithub\TestingGitRepo\.git. The table lists various files and folders within the .git directory, including hooks, info, logs, objects, refs, config, description, HEAD, index, and packed-refs. All files were modified on 24-11-2024 at 18:31 or 18:40, and most are 1 KB in size.

Name	Date modified	Type	Size
hooks	24-11-2024 18:31	File folder	
info	24-11-2024 18:31	File folder	
logs	24-11-2024 18:31	File folder	
objects	24-11-2024 18:40	File folder	
refs	24-11-2024 18:31	File folder	
config	24-11-2024 18:31	File	1 KB
description	24-11-2024 18:31	File	1 KB
HEAD	24-11-2024 18:31	File	1 KB
index	24-11-2024 18:40	File	1 KB
packed-refs	24-11-2024 18:31	File	1 KB

Then, I committed the changes with:

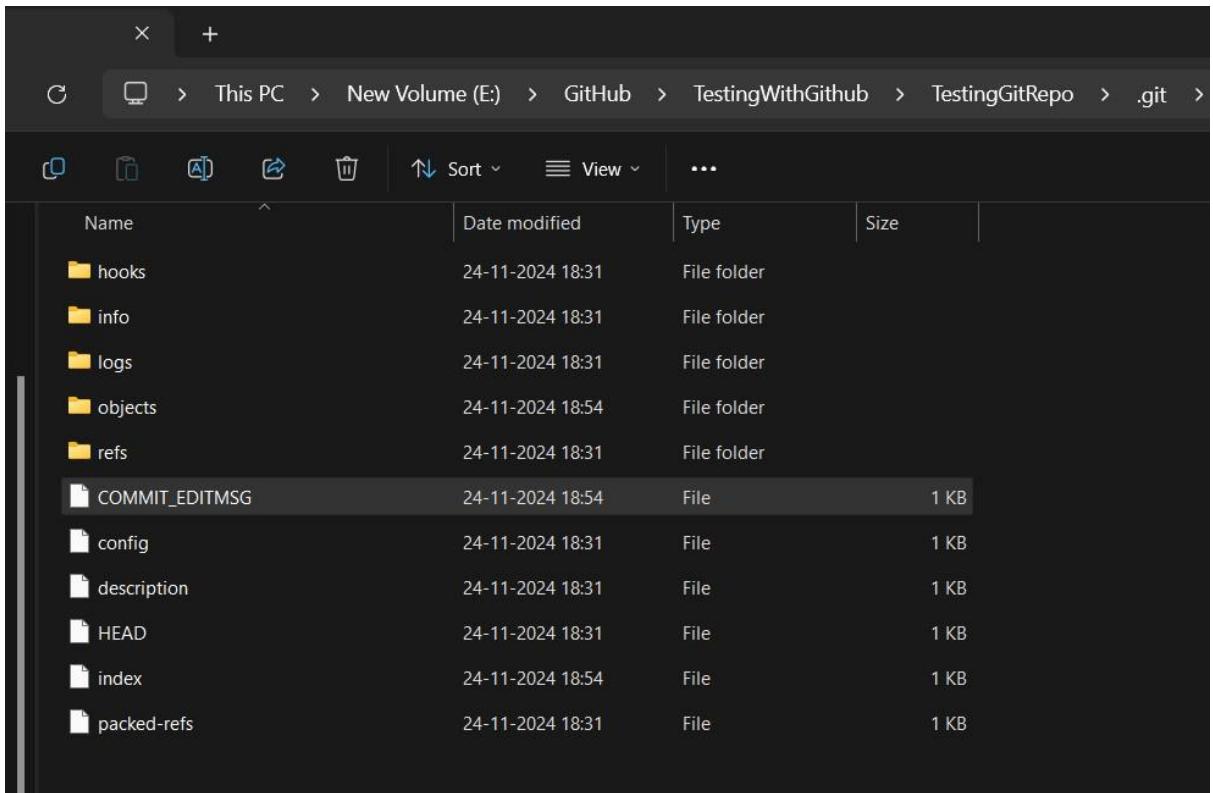
```
git commit -m "Added TestCases.txt"
```

```
PS E:\GitHub\TestingWithGitHub\TestingGitRepo> git commit -m "Added TestCases.txt file from editor"
[master 1ee7b92] Added TestCases.txt file from editor
 1 file changed, 2 insertions(+)
  create mode 100644 TestCases.txt
PS E:\GitHub\TestingWithGitHub\TestingGitRepo>
```

This saved the changes in my local Git repository.

Here's what happened:

- The commit saved the changes (adding the new file) in my local repository.



- It created a new version of my project's history, but the changes were still local and not yet visible on GitHub.

A screenshot of a GitHub repository page for 'TestingGitRepo'. The repository is public and has 1 branch and 0 tags. The master branch is selected. The commit history shows one commit from 'AutomatedMind' created 'TestInfo.txt' 26 minutes ago. The repository details show 0 stars, 1 watching, and 0 forks. The 'About' section states: 'This repository is created to demonstrate and practice GitHub workflows.'

Step 7: Pushing Changes to GitHub

To upload the changes to GitHub, I used:

```
git push origin master
```

```

PS E:\GitHub\TestingWithGitHub\TestingGitRepo> git push origin master
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 368 bytes | 368.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/AutomatedMind/TestingGitRepo.git
  2486601..1ee7b92  master -> master

```

Now, the **TestCases.txt** file appeared in my GitHub repository.

Step 8: Creating a New Branch

To organize my work, I created a new branch named **feature-test-cases**:

```
git checkout -b feature-test-cases
```

```

PS E:\GitHub\TestingWithGitHub\TestingGitRepo> git checkout -b feature-test-cases
Switched to a new branch 'feature-test-cases'
PS E:\GitHub\TestingWithGitHub\TestingGitRepo>

```

This command created and switched to the new branch.

To verify which branch it points

```
git branch -a
```

```

PS E:\GitHub\TestingWithGitHub\TestingGitRepo> git branch -a
* feature-test-cases
  master
  remotes/origin/HEAD -> origin/master
  remotes/origin/feature-test-cases
  remotes/origin/master

```

Push the branch to the remote repository:

```
git push origin feature-test-cases
```

```
PS E:\GitHub\TestingWithGitHub\TestingGitRepo> git push origin feature-test-cases
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 12 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 1.31 KiB | 1.31 MiB/s, done.
Total 6 (delta 0), reused 3 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'feature-test-cases' on GitHub by visiting:
remote:     https://github.com/AutomatedMind/TestingGitRepo/pull/new/feature-test-cases
remote:
To https://github.com/AutomatedMind/TestingGitRepo.git
 * [new branch]      feature-test-cases -> feature-test-cases
PS E:\GitHub\TestingWithGitHub\TestingGitRepo>
```

Note:

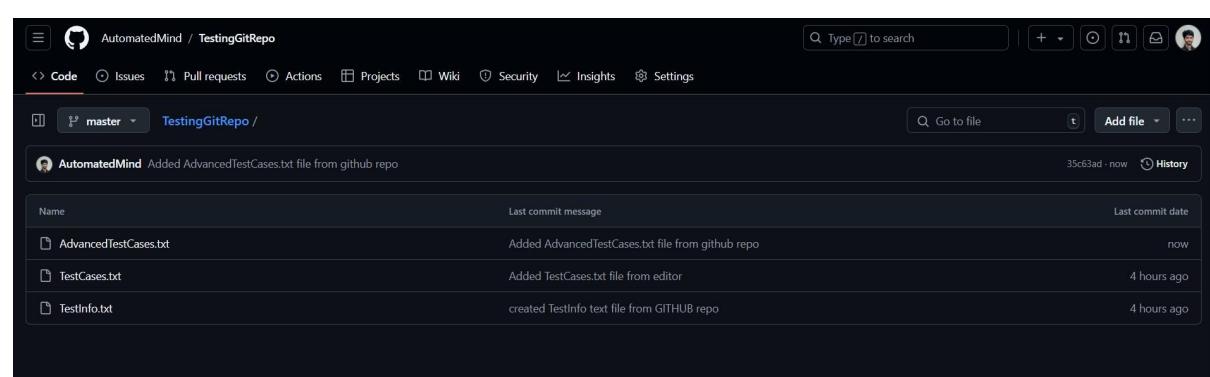
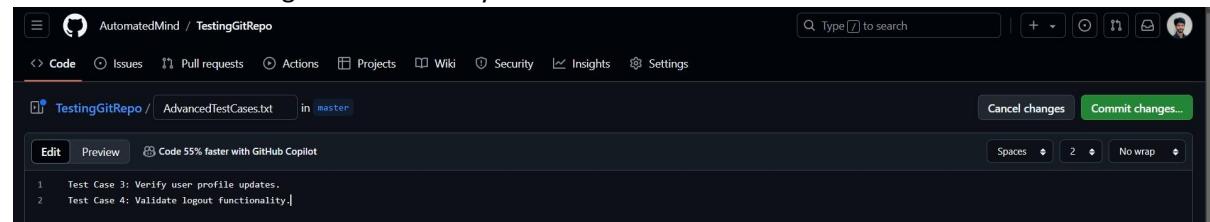
Creating a branch only? No need to stage or commit before pushing.

Made changes in the branch? You must stage and commit before pushing.

I then added a file called **AdvancedTestCases.txt** on GitHub with the content:

Test Case 3: Verify user profile updates.

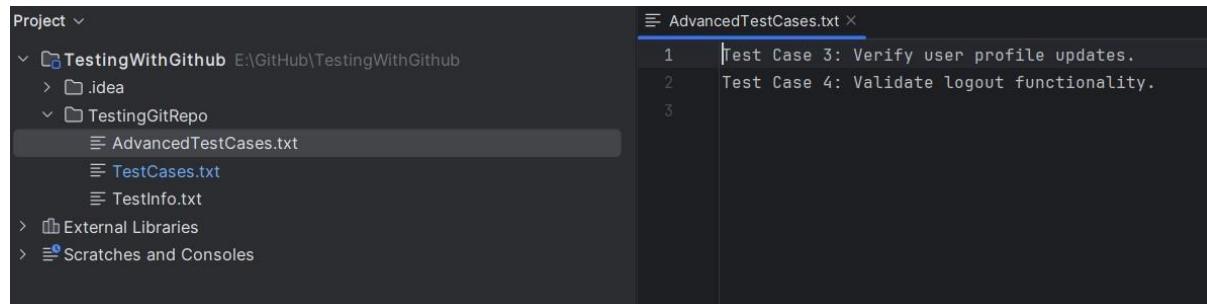
Test Case 4: Validate logout functionality.



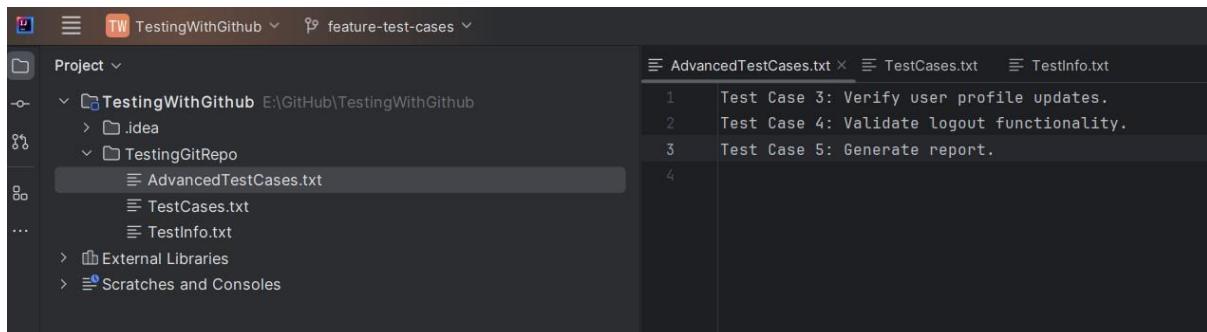
To pull this file into my local repository, I ran:

```
git pull origin feature-test-cases
```

```
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git pull origin feature-test-cases
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 1.02 KiB | 6.00 KiB/s, done.
From https://github.com/AutomatedMind/TestingGitRepo
 * branch            feature-test-cases -> FETCH_HEAD
   1ee7b92..1bff1cb  feature-test-cases -> origin/feature-test-cases
Updating 1ee7b92..1bff1cb
Fast-forward
 AdvancedTestCases.txt | 2 ++
 1 file changed, 2 insertions(+)
 create mode 100644 AdvancedTestCases.txt
```



Again, I made changes in AdvancedTestCases and TestCases text file.



Since I Made changes in the branch I did stage and commit before pushing

```

PS E:\GitHub\TestingWithGithub\TestingGitRepo> git add .
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git commit -m "Updated AdvanceTestcases and TestCases file"
[feature-test-cases 3fe207f] Updated AdvanceTestcases and TestCases file
 2 files changed, 2 insertions(+), 2 deletions(-)
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git push origin feature-test-cases
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 440 bytes | 440.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/AutomatedMind/TestingGitRepo.git
 1bff1cb..3fe207f  feature-test-cases -> feature-test-cases

```

The screenshot shows the GitHub interface for the 'TestingGitRepo' repository. The 'feature-test-cases' branch is selected. A file named 'AdvancedTestCases.txt' is open, showing three test cases. The code is annotated with GitHub Copilot's performance metric: 'Code 55% faster with GitHub Copilot'.

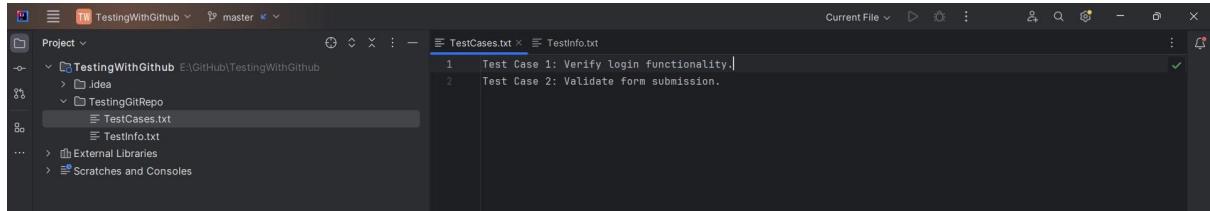
I could see the changes in feature-test-case branch in GITHUB repo after push. Verify master branch **from GITHUB**

The screenshot shows the GitHub interface for the 'TestingGitRepo' repository. The 'master' branch is selected. It displays recent activity from the 'feature-test-cases' branch, including the creation of 'AdvancedTestCases.txt' and 'TestInfo.txt' files. The repository description states: 'This repository is created to demonstrate and practice GitHub workflows.'

From local git

checkout master

```
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git checkout master
Switched to branch 'master'
Your branch is behind 'origin/master' by 1 commit, and can be fast-forwarded.
  (use "git pull" to update your local branch)
```



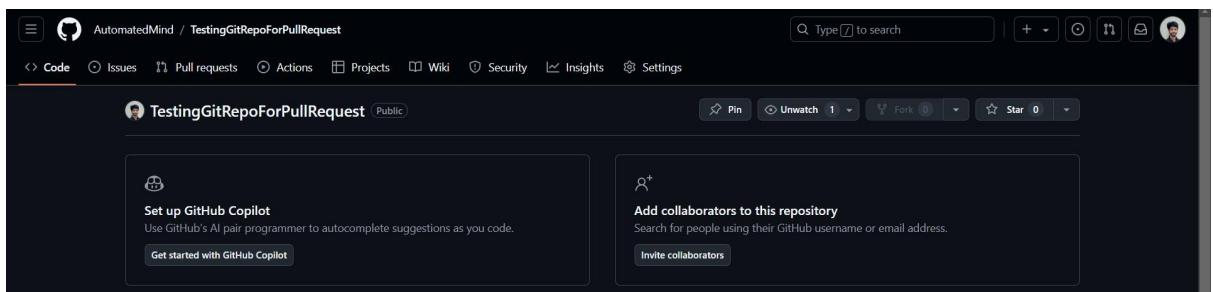
The newly added file is present only in new branch which is feature-test-case and not in master.

Step 9: Managing Multiple Repositories

I wanted to practice managing more than one repository.

1. Created a New Repository:

On GitHub, I created another repository called **TestingGitRepoForPullRequest**.

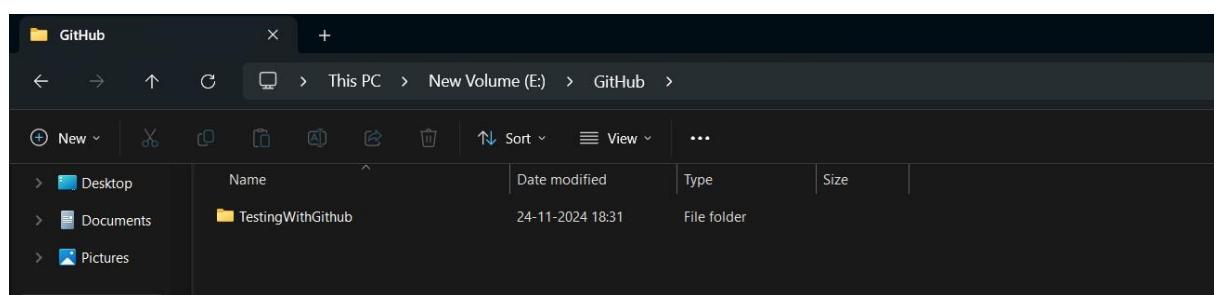


2. Initialized Git:

Before initialize it have the proper folder structure

In my local folder, I navigated to Base project folder **TestingWithGithub**

```
PS E:\GitHub\TestingWithGithub\TestingGitRepo> cd..
PS E:\GitHub\TestingWithGithub>
```



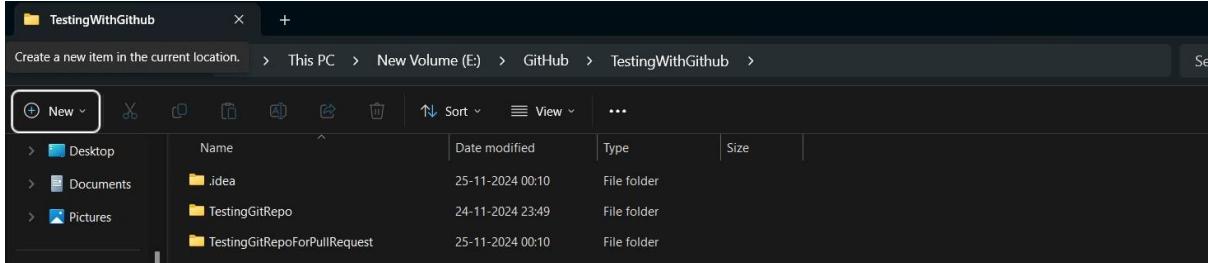
Now I created a new directory mkdir

TestingGitRepoForPullRequest

```
PS E:\GitHub\TestingWithGithub> mkdir TestingGitRepoForPullRequest
```

```
Directory: E:\GitHub\TestingWithGithub
```

Mode	LastWriteTime	Length	Name
d----	25-11-2024 00:09		TestingGitRepoForPullRequest

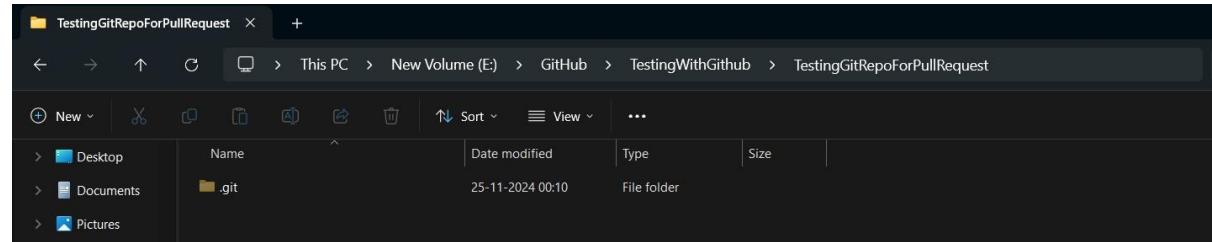


Then navigate to new repo and initialize git

```
git init
```

```
PS E:\GitHub\TestingWithGithub> cd TestingGitRepoForPullRequest
PS E:\GitHub\TestingWithGithub\TestingGitRepoForPullRequest> git init
Initialized empty Git repository in E:/GitHub/TestingWithGithub/TestingGitRepoForPullRequest/.git/
```

This created a .git folder to track the project.



1. Adding and Committing Files:

I created a file named **TestPlan.txt** with this content:

Test Plan: Automation scripts for E2E testing.

Tools: Selenium, Postman, TestNG. Then,

I staged and committed the file: `git add .`

```
git commit -m "Added TestPlan.txt"
```

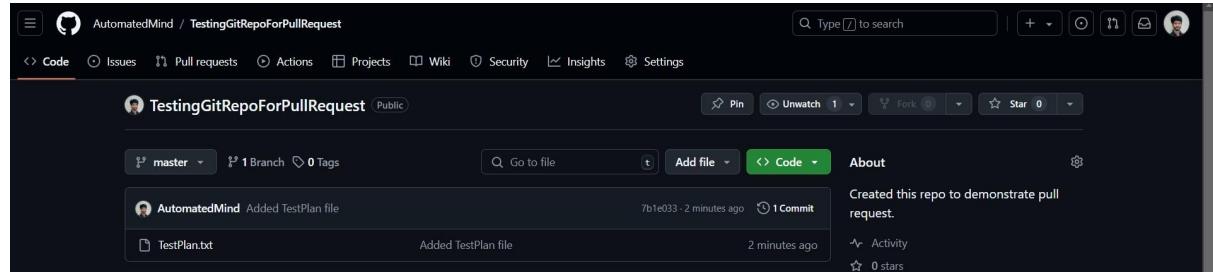
```
PS E:\GitHub\TestingWithGithub\TestingGitRepoForPullRequest> git init
Initialized empty Git repository in E:/GitHub/TestingWithGithub/TestingGitRepoForPullRequest/.git/
PS E:\GitHub\TestingWithGithub\TestingGitRepoForPullRequest> git add TestPlan.txt
PS E:\GitHub\TestingWithGithub\TestingGitRepoForPullRequest> git commit -m "Added TestPlan file"
[master (root-commit) 7b1e033] Added TestPlan file
 1 file changed, 2 insertions(+)
 create mode 100644 TestPlan.txt
```

2. Connected to GitHub:

I linked my local folder to the GitHub repository using:

```
git remote add origin <GitHub-Repo-URL>
```

```
PS E:\GitHub\TestingWithGithub\TestingGitRepoForPullRequest> git remote add origin https://github.com/AutomatedMind/TestingGitRepoForPullRequest.git
PS E:\GitHub\TestingWithGithub\TestingGitRepoForPullRequest> git push origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 302 bytes | 302.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/AutomatedMind/TestingGitRepoForPullRequest.git
 * [new branch]      master -> master
PS E:\GitHub\TestingWithGithub\TestingGitRepoForPullRequest>
```



Step 10: Merging Changes

To merge changes from one branch to another:

1. Switched to the **master** branch:

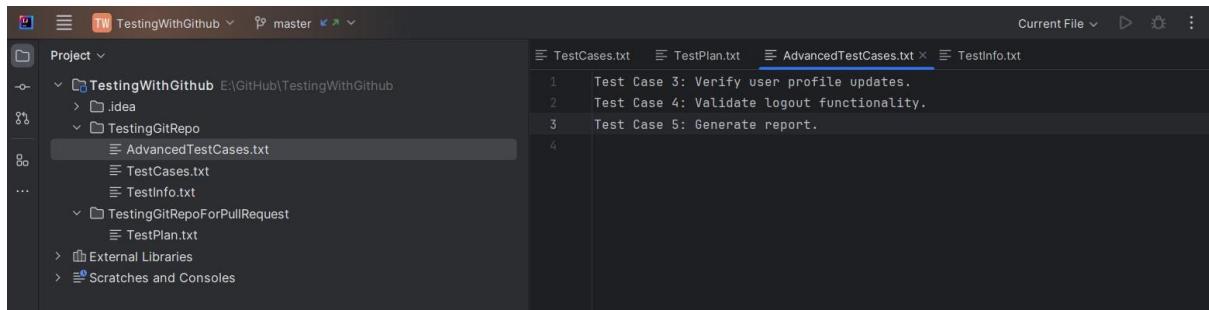
```
git checkout master
```

```
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git checkout master
Switched to branch 'master'
Your branch is behind 'origin/master' by 1 commit, and can be fast-forwarded.
(use "git pull" to update your local branch)
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git branch -a
  feature-test-cases
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/feature-test-cases
  remotes/origin/master
```

2. Merged the changes:

```
git merge feature-test-cases
```

```
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git merge feature-test-cases
Updating 1ee7b92..bf933b5
Fast-forward
 AdvancedTestCases.txt | 3 +++
 TestCases.txt          | 3 +--
 2 files changed, 4 insertions(+), 2 deletions(-)
 create mode 100644 AdvancedTestCases.txt
```



After merge now the AdvancedTestCases.txt file is merged to master from feature-test-cases branch

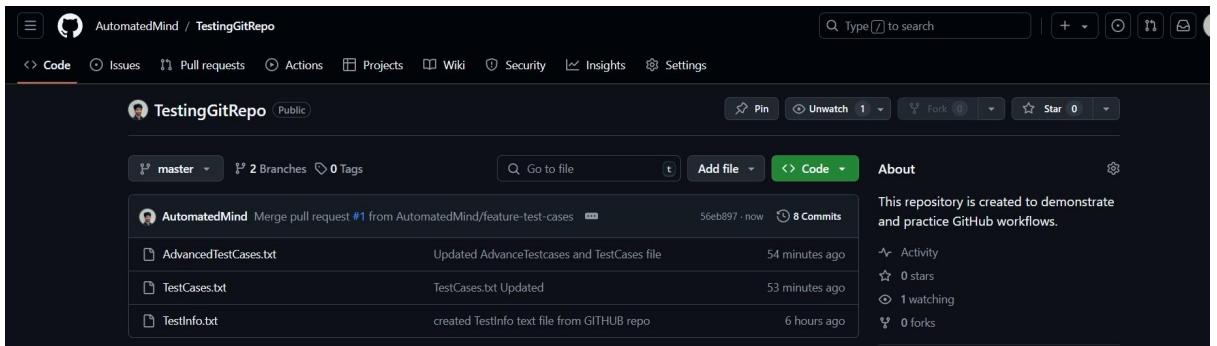
3. Created a pull request on GitHub for review and merged it.

Still in GITHUB repo the AdvancedTestCases.txt file is missing even after merge, because we merged in local repo but to merge in remote repo master or main branch we need to raise a pull request.

The screenshot shows the GitHub repository page for 'TestingGitRepo'. At the top, there's a banner indicating 'feature-test-cases had recent pushes 49 minutes ago'. Below the banner, the repository details show it's a public repository created by 'AutomatedMind'. It has 2 branches (master and feature-test-cases), 0 tags, and 4 commits. The latest commit, 'ca27c9a · 1 hour ago', is a file deletion: 'Delete AdvancedTestCases.txt'. The commit before it, 'ca27c9a · 5 hours ago', is 'Added TestCases.txt file from editor'. The commit before that, 'ca27c9a · 6 hours ago', is 'created TestInfo.txt file from GITHUB repo'. On the right side, there's an 'About' section stating the repository is created to demonstrate and practice GitHub workflows, with 0 stars, 1 watching, and 0 forks.

The screenshot shows the 'Open a pull request' interface. It displays a comparison between the 'base: master' branch and the 'compare: feature-test-cases' branch. A green checkmark indicates they are 'Able to merge'. The pull request title is 'Feature test cases' and the description is 'Merge file AdvanceTestCases from feature-test-cases branch to master'. The description area includes a note that Markdown is supported. To the right, there are sections for 'Reviewers' (No reviews), 'Assignees' (No one—assign yourself), 'Labels' (None yet), 'Projects' (None yet), and 'Milestone' (No milestone). A note at the bottom right says 'Use Closing keywords in the description to automatically close issues'. At the bottom right of the pull request form is a 'Create pull request' button.

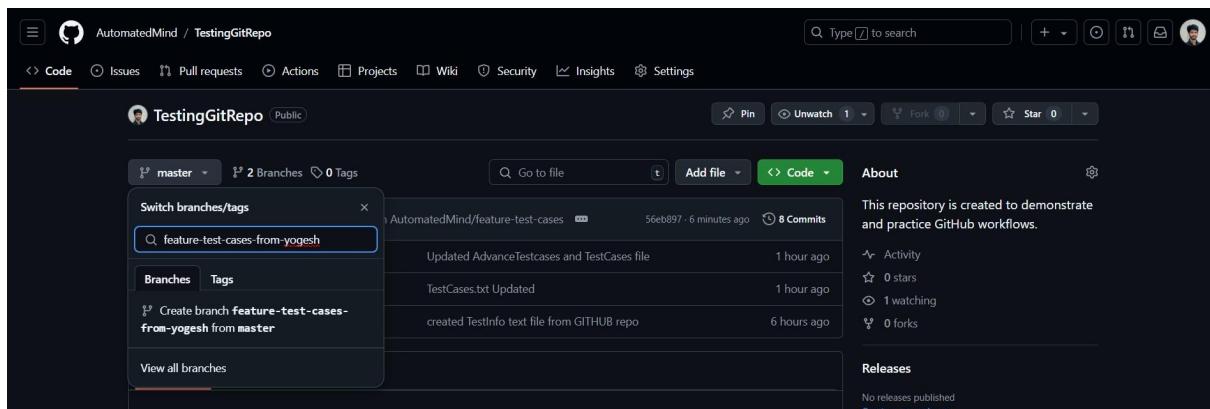
The screenshot shows the pull request review interface. At the top, it lists 'Conversation 0', 'Commits 3', 'Checks 0', and 'Files changed 2'. The first commit by 'AutomatedMind' is shown with the message 'Merge file AdvanceTestCases from feature-test-cases branch to master'. Below this, another commit by 'AutomatedMind' is listed with three sub-commits: 'Create AdvancedTestCases.txt', 'Updated AdvanceTestcases and TestCases file', and 'TestCases.txt Updated'. The commit messages include SHA-1 hash codes: '1bff1cb', '3fe207f', and 'bf933b5'. At the bottom of the screen, there's a summary of review requirements: 'Require approval from specific reviewers before merging' (with a note about Rulesets), 'Continuous integration has not been set up' (with a note about GitHub Actions and other apps), and 'This branch has no conflicts with the base branch' (with a note that merging can be performed automatically). There's also a 'Merge pull request' button and a note about opening the pull request in GitHub Desktop or viewing command line instructions.



Now I could see the merged file in master branch.

Step 11: Conflict

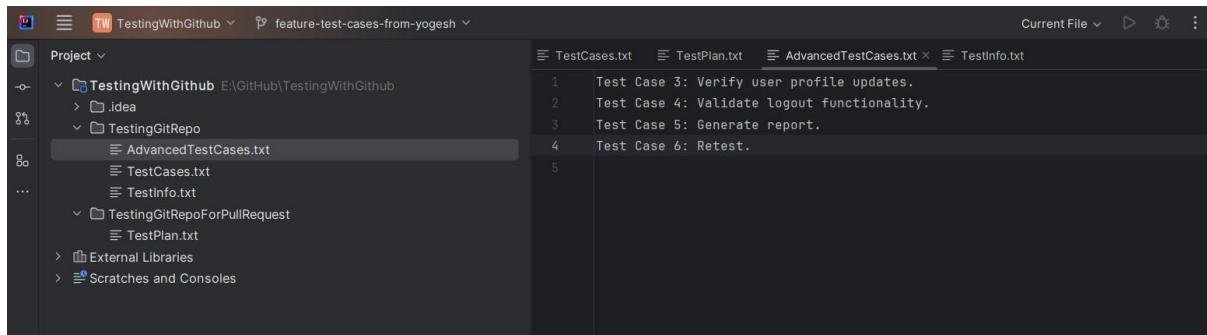
To demonstrate conflict I have created one more branch from master This time from GITHUB



Switched to new branch

```
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git branch -a
  feature-test-cases
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/feature-test-cases
  remotes/origin/feature-test-cases-from-yogesh
  remotes/origin/master
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git checkout feature-test-cases-from-yogesh
Switched to a new branch 'feature-test-cases-from-yogesh'
branch 'feature-test-cases-from-yogesh' set up to track 'origin/feature-test-cases-from-yogesh'.
PS E:\GitHub\TestingWithGithub\TestingGitRepo>
```

Updating the file AdvancedTestCases.txt



Now merged this file to master from feature-test-cases-from-yogesh, but still the changes are reflected in local to make the changes in remote repo. Raise a pull request and merge.

```

PS E:\GitHub\TestingWithGithub\TestingGitRepo> git add .
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git commit -m "Added Test Case 6: Retest"
[feature-test-cases-from-yogesh 9782dfb] Added Test Case 6: Retest
 1 file changed, 1 insertion(+)
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git push origin feature-test-cases-from-yogesh
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 318 bytes | 318.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/AutomatedMind/TestingGitRepo.git
 56eb897..9782dfb  feature-test-cases-from-yogesh -> feature-test-cases-from-yogesh
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git checkout master
Switched to branch 'master'
Your branch is up to date with 'origin/master'.
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git merge feature-test-cases-from-yogesh
Updating 56eb897..9782dfb
Fast-forward
 AdvancedTestCases.txt | 1 +
 1 file changed, 1 insertion(+)

```

The screenshot shows the GitHub 'Open a pull request' interface. At the top, it says 'base: master' and 'compare: feature-test-cases-from-yogesh'. A green checkmark indicates 'Able to merge'. The main area has fields for 'Add a title' (containing 'Added Test Case 6: Retest') and 'Add a description' (containing 'Approving code from feature-test-cases-from-yogesh'). To the right, there are sections for 'Reviewers' (no reviews), 'Assignees' (no one assigned), 'Labels' (None yet), 'Projects' (None yet), and 'Milestone' (No milestone). At the bottom, there's a note about using closing keywords and a 'Create pull request' button.

Now the changes merged in master remote repo

Again, make the changes in same file AdvancedTestCases.txt but from different branch feature-testcases.

```
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git checkout feature-test-cases
Switched to branch 'feature-test-cases'
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git branch -a
* feature-test-cases
  feature-test-cases-from-yogesh
    master
      remotes/origin/HEAD -> origin/master
      remotes/origin/feature-test-cases
      remotes/origin/feature-test-cases-from-yogesh
      remotes/origin/master
PS E:\GitHub\TestingWithGithub\TestingGitRepo>
```

Updating the file AdvancedTestCases.txt

Now merged this file to master from feature-test-cases, but still the changes are reflected in local to make the changes in remote repo. Raise a pull request and merge.

```
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git add .
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git commit -m "Test Case 6: is updated from feature-test-cases branch"
[feature-test-cases 5e15c31] Test Case 6: is updated from feature-test-cases branch
1 file changed, 1 insertion(+)
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git push origin feature-test-cases
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 342 bytes | 342.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/AutomatedMind/TestingGitRepo.git
 bf933b5..5e15c31  feature-test-cases -> feature-test-cases
```

Merge to master and make a pull request

```

PS E:\GitHub\TestingWithGithub\TestingGitRepo> git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)
PS E:\GitHub\TestingWithGithub\TestingGitRepo> git merge feature-test-cases
Auto-merging AdvancedTestCases.txt
CONFLICT (content): Merge conflict in AdvancedTestCases.txt
Automatic merge failed; fix conflicts and then commit the result.
PS E:\GitHub\TestingWithGithub\TestingGitRepo>

```

You will instantly see the Conflict message because I tried updating the same file from different branch and I pushed both of them to master branch.

Comparing changes

Choose two branches to see what's changed or to start a new pull request. If you need to, you can also compare across forks or learn more about diff comparisons.

base: master ⌂ ... compare: feature-test-cases ✘ Can't automatically merge. Don't worry, you can still create the pull request.

Discuss and review the changes in this comparison with others. [Learn about pull requests](#)

[Create pull request](#)

⌚ 1 commit

⌚ 1 file changed

⌚ 1 contributor

↳ Commits on Nov 25, 2024

Test Case 6: is updated from feature-test-cases branch
 AutomatedMind committed 6 minutes ago

5e15c31 ⌂

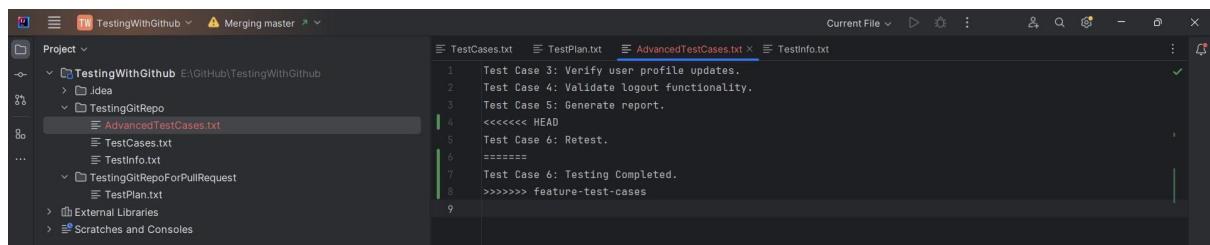
⌚ Showing 1 changed file with 1 addition and 0 deletions.

[Split](#) [Unified](#)

```

1 AdvancedTestCases.txt
...
@@ -1,3 +1,4 @@
1   1 Test Case 3: Verify user profile updates.
2   2 Test Case 4: Validate logout functionality.
3   3 Test Case 5: Generate report.
4 + Test Case 6: Testing Completed.

```



To resolve this approve the pull request and click on **Resolve conflicts** button

The screenshot shows a GitHub pull request page. At the top, there's a navigation bar with links for Code, Issues, Pull requests (which is the active tab), Actions, Projects, Wiki, Security, Insights, and Settings. A search bar is also at the top right. The main title of the pull request is "Test Case 6: is updated from feature-test-cases branch #3". Below the title, it says "AutomatedMind wants to merge 1 commit into master from feature-test-cases". The pull request has 0 conversations, 1 commit, 0 checks, and 1 file changed. The commit message is "Test Case 6: is updated from feature-test-cases branch". The commit hash is Se15c31. On the right side, there are sections for Reviewers (No reviews), Assignees (No one—assign yourself), Labels (None yet), Projects (None yet), Milestone (None), and Branch protection rules (None). A note at the bottom left says "This branch has conflicts that must be resolved" and provides links to the web editor or command line. A "Resolve conflicts" button is also present. At the bottom, there are buttons for "Merge pull request" and "You can also open this in GitHub Desktop or view command line instructions."

Resolve manually

AutomatedMind / TestingGitRepo

Type ⌘ to search | + - ⌂ ⌓ ⌔ ⌕

<> Code Issues Pull requests 1 Actions Projects Wiki Security Insights Settings

Test Case 6: is updated from feature-test-cases... #3

Resolving conflicts between feature-test-cases and master and committing changes → feature-test-cases

1 conflicting file	AdvancedTestCases.txt	1 conflict	Prev ⌂	Next ⌓	Mark as resolved
AdvancedTestCases.txt	AdvancedTestCases.txt	1	Test Case 3: Verify user profile updates.		
	2	Test Case 4: Validate logout functionality.			
	3	Test Case 5: Generate report.			
	4	<<<< feature-test-cases			
	5	Test Case 6: Testing Completed.			
	6	=====			
	7	Test Case 6: Retest.			
	8	>>>> master			
	9				

The screenshot shows a GitHub pull request interface. The top navigation bar includes links for Code, Issues, Pull requests (with 1 new item), Actions, Projects, Wiki, Security, Insights, and Settings. The main title of the pull request is "Test Case 6: is updated from feature-test-cases... #3". Below the title, it says "Resolving conflicts between feature-test-cases and master and committing changes → feature-test-cases". A table lists a single conflicting file, "AdvancedTestCases.txt", which contains the following content:

1 conflicting file	AdvancedTestCases.txt	1 conflict	Prev ↺	Next ↻	Mark as resolved
AdvancedTestCases.txt AdvancedTestCases.txt	1 Test Case 3: Verify user profile updates. 2 Test Case 4: Validate logout functionality. 3 Test Case 5: Generate report. 4 Test Case 6: Testing Completed.				

Quick Revision: Git Terminologies and Commands (With Examples)

1. Git

- **Purpose:** Checks if Git is installed on your system.
 - **Command:**
git --version
 - **Example Output:**
git version 2.42.0
-

2. GitHub

- **Purpose:** A cloud platform to store, manage, and collaborate on Git repositories.
-

3. Git Config

- **Purpose:** Set your identity for commit messages.
- **Commands & Examples:**

```
git config --global user.name "Yogesh Pandian"
git config --global user.email "yogesh.pandian@example.com"
git config --list
```

- **Output:**

```
user.name=Yogesh Pandian
user.email=yogesh.pandian@example.com
```

4. Git Repository

- **Purpose:** Initialize a Git project in your local folder.
- **Command:**
git init
- **Example:**

```
mkdir my-project
cd my-project
git init
```

- **Output:**

```
Initialized empty Git repository in /Users/yourname/my-project/.git/
```

5. Clone

- **Purpose:** Copy an existing GitHub repo to your machine.
- **Command:**
git clone https://github.com/username/repo.git

- **Example:**

```
git clone https://github.com/octocat>Hello-World.git
```

6. Add

- **Purpose:** Stage changes (new or modified files) for commit.
- **Command:**
git add filename
- **Example:**

```
git add index.html
```

7. Commit

- **Purpose:** Save staged changes with a message.
- **Command:**
git commit -m "Meaningful commit message"
- **Example:**

```
git commit -m "Added login functionality"
```

8. Push

- **Purpose:** Upload local commits to GitHub.
- **Command:**
git push origin branch-name
- **Example:**

```
git push origin main
```

9. Pull

- **Purpose:** Get the latest changes from GitHub.
- **Command:**
git pull origin branch-name
- **Example:**

```
git pull origin main
```

10. Branch

- **Purpose:** Create and switch between versions.
- **Commands & Examples:**

```
git branch feature-ui          # Create branch  
git checkout -b bug-fix       # Create and switch  
git checkout main             # Switch to main
```

11. Merge

- **Purpose:** Combine another branch into current one.
- **Commands & Example:**

```
git checkout main          # Switch to main
git merge feature-ui      # Merge feature-ui into main
```

12. Status

- **Purpose:** View current file status (tracked, staged, etc.)
- **Command:**
git status
- **Example Output:**

```
Changes to be committed:
  modified: index.html
```

13. Remote

- **Purpose:** Link your local project to a GitHub repo.
- **Command:**
git remote add origin <repo-url>
- **Example:**

```
git remote add origin https://github.com/johndoe/my-project.git
```
