SRIDHAR E
71772118145

Git Hands-on Documentation

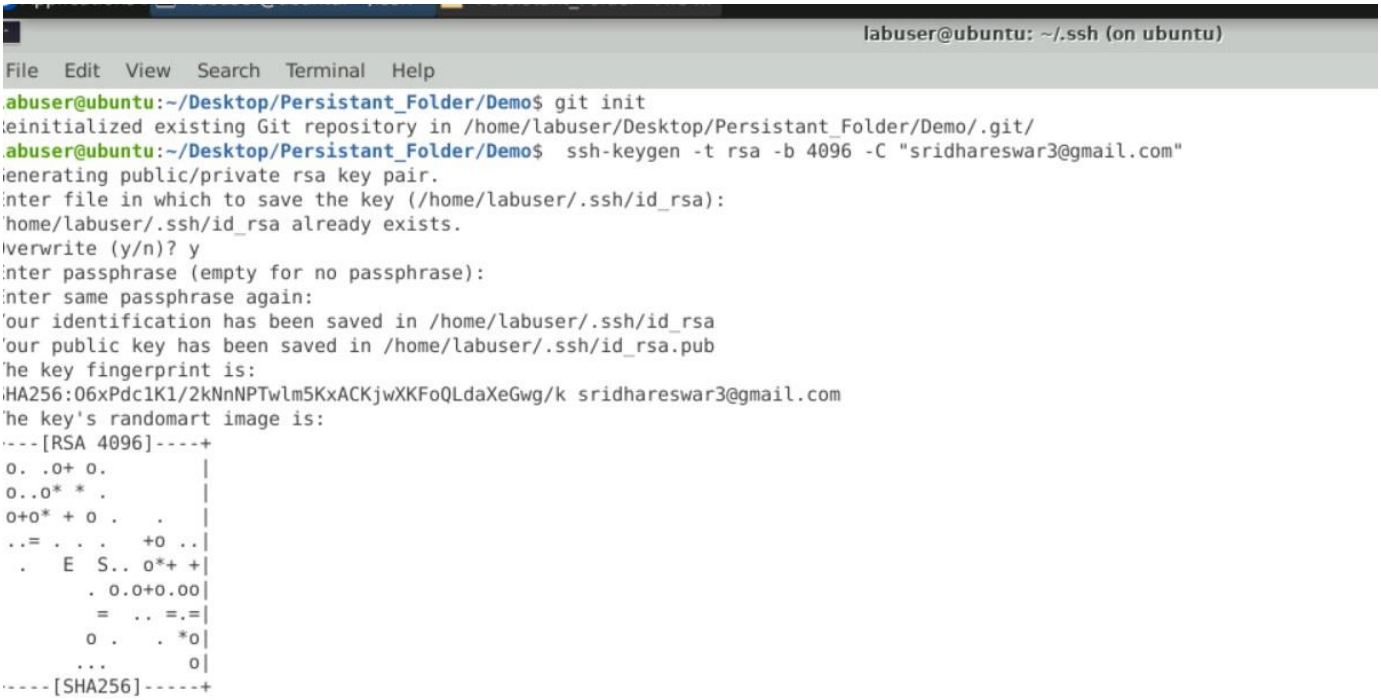
1. Creating and Initializing a Git Repository

- Initialize a Git repository locally with the '**git init**' command.
- Create a repository on GitHub through the GUI.
- Prior to pushing code, establish the remote repository on GitHub's website and add it as a remote for your local repository.
- Alternatively, clone a GitHub repository using the git clone **remote_repo_ssh_url** command.

```
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ git remote add origin git@github.com:sridhar8642/walmartrepo.git
```

```
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ cd /home/labuser/.ssh
labuser@ubuntu:~/.ssh$ ls
authorized_keys  id_ed25519  id_ed25519.pub  id_rsa  id_rsa.pub  known_hosts
labuser@ubuntu:~/.ssh$ cat id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQCjX0LLE3WfhWLG7mjzpedZHRm3Hj432GTJCt4hWmv+yAPX9CWhMrybBRlsT5HnYTMTIrW7VUqn69vt5SQGkew61GPCTTJIOmtVq+wbEuKSysxmt676zxqQJbJHg/2xPplpPWtXtAWrYRRPEDyXqYWhL
uSu02uwnblANmx+WGvk8cK+4u1boVeyHb0X7MeGNaW4XcDErRl9nXxVFc6oBmtrGjD6y/Kt58OQoFVPIlElojAESQtJmmMWFl+OO/jTOwCVMkRjdYR2+gD/RRCWlcCuCkffdfpK6gtnTXno5+VQsvwQ7/BQQ+w3P4S9gcyv82Kf92154vdldJGm0ABqBP
5Yhd9md9cMXeJ+RFTxOQvDMNyGPOvjG3HM3VX9csAgoRCwGk57GbmMa5M+TnIBp32oEkCyWIQbZbeScD8nfZ760/V5geCUL3bbnNXO1u9aHQCrqzySI5sXuEzfN3VyUOXnFzOmeUZpK2XISbrAG8zTT8Ql1Tj+Qtc9DgTqURKFyyrDca+hx+pJVN9Qj0c
Wfl8G+ZacuHn4CjIZooueED/sOuH2pEDn8h1d3pa0S+U5bPd5ky4qyNm+kWl00ULYQv7AYFhgeeY+nY9riJikFxsk0YaXEO+qj9djdg3NrjMfJTazlRnmpR/NMK345S+UR1Ta238k2+oNgV3GSt6ER+5rdw== sridhareswar3@gmail.com
labuser@ubuntu:~/.ssh$ ssh-add ~/.ssh/id_rsa
Identity added: /home/labuser/.ssh/id_rsa (sridhareswar3@gmail.com)
labuser@ubuntu:~/.ssh$ ssh -T git@github.com
git@github.com: Permission denied (publickey).
labuser@ubuntu:~/.ssh$ ssh -T git@github.com
Hi sridhar8642! You've successfully authenticated, but GitHub does not provide shell access.
labuser@ubuntu:~/.ssh$
```

2. Creating SSH Key for Remote Repository.



- Add tGenerate a new SSH key pair using the command:

  Ssh-keygen -t rsa -b 4096 -C

  sridhareswar3@gmail.com

  This creates a public-private key pair.
- Add the generated key to the SSH agent using the command:
  Ssh-add key_name

  By default, the key is stored in the ~/.ssh/ directory.
- Add the public key to GitHub for authentication.he key to the SSH agent with ssh-add key_name.
- Add the public key to GitHub for authentication.

3. Adding Remote Repository to Local Repository

- Add a remote repository using git remote add origin git@github.com:sridhar8642/walmartrepo.git.

4. Configuring Username and Email

```
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ git config --global user.email "sridhareswar3@gmail.com"
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ git config --global user.name "sridhar8642"
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ git config user.name
sridhar8642
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ git config user.email
sridhareswar3@gmail.com
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$
```

Set username and email for remote repo with:

- Git –global user.name “sridhar8642”
- Git config –global user.email sridhareswar3@gmail.com

5. Creating Branches

Create a new branch with git branch dev and git checkout dev, or in one step with git checkout -b config dev.

```
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ git checkout -b dev
Switched to a new branch 'dev'
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ git status
On branch dev

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ git add .
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ git commit -m "first commit"
[dev (root-commit) 3add126] first commit
 1 file changed, 63 insertions(+)
 create mode 100644 index.html
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$ git push origin dev
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 833 bytes | 416.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To github.com:sridhar8642/walmartrepo.git
 * [new branch]      dev -> dev
labuser@ubuntu:~/Desktop/Persistant_Folder/Demo$
```

6. Tracking files

Track files using **git add filename**.

7. Committing Changes to Local Repository

```
labuser@ubuntu:~/Desktop/Persistant_Folder/demo_repo$ git add .
labuser@ubuntu:~/Desktop/Persistant_Folder/demo_repo$ git commit -m "third commit"
[dev 76c46af] third commit
 1 file changed, 1 insertion(+), 1 deletion(-)
labuser@ubuntu:~/Desktop/Persistant_Folder/demo_repo$ git push origin dev
Warning: Permanently added 'github.com,20.207.73.82' (ECDSA) to the list of known hosts.
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 339 bytes | 169.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:sridhar8642/demo_repo.git
   f68aa9c..76c46af  dev -> dev
labuser@ubuntu:~/Desktop/Persistant_Folder/demo_repo$
```

Commit changes with git commit -m “Commit message”.

8. Pushing Local Repository to Remote

Push changes to remote with git push origin dev.

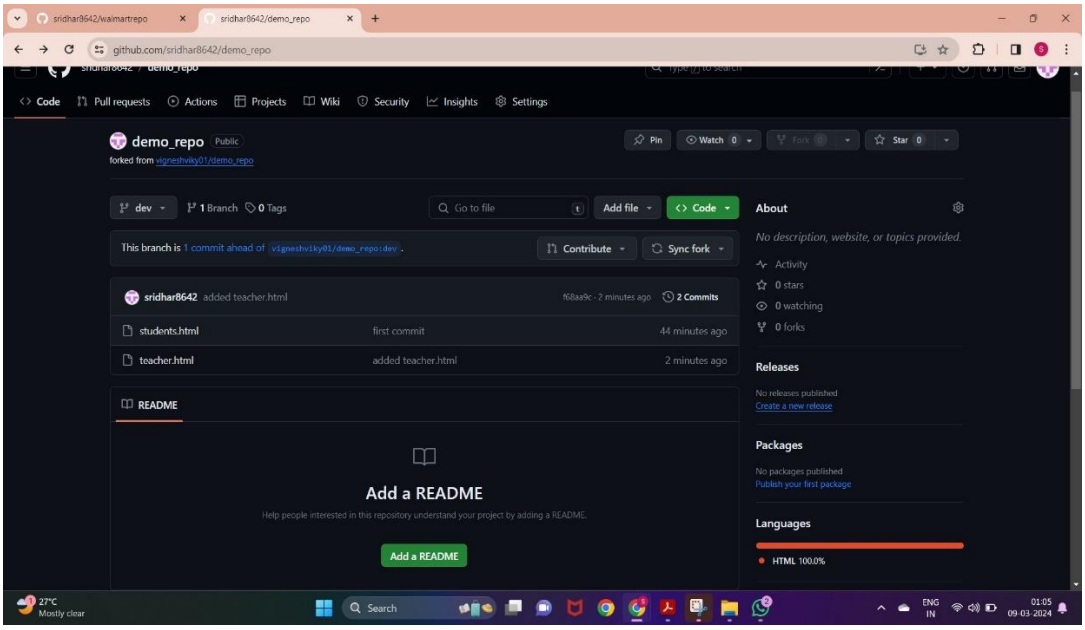
```
labuser@ubuntu:~/Desktop/Persistant_Folder/demo_repo$ git pull origin dev
From github.com:vigneshviky01/demo_repo
 * branch            dev        -> FETCH_HEAD
Already up to date.
```

```
labuser@ubuntu:~/Desktop/Persistant_Folder/demo_repo$ git pull origin dev
remote: Enumerating objects: 5, done.
remote: Counting objects: 100% (5/5), done.
remote: Compressing objects: 100% (4/4), done.
remote: Total 4 (delta 0), reused 3 (delta 0), pack-reused 0
Unpacking objects: 100% (4/4), 1.42 KiB | 18.00 KiB/s, done.
From github.com:vigneshviky01/demo_repo
 * branch            dev        -> FETCH_HEAD
   308ef48..2627db8  dev        -> origin/dev
Updating 308ef48..2627db8
Fast-forward
 teacher.html | 19 ++++++++++++++++++
 1 file changed, 19 insertions(+)
 create mode 100644 teacher.html
```

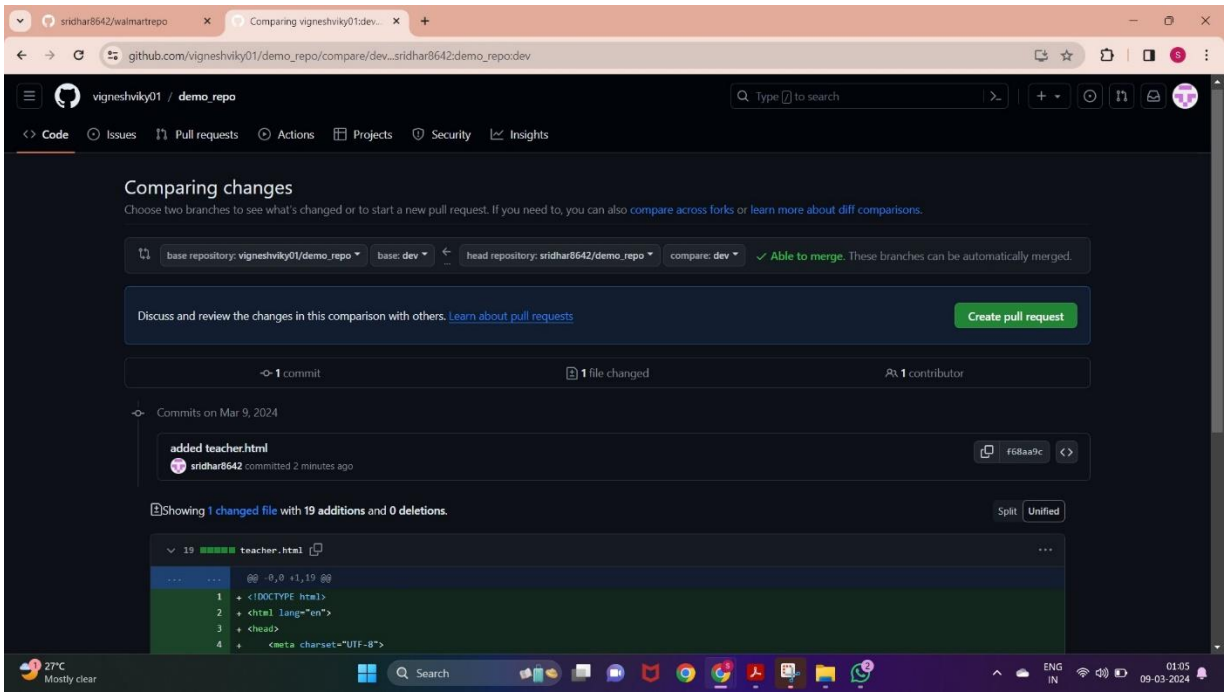
```
labuser@ubuntu:~/Desktop/Persistant_Folder/demo_repo$ git checkout master
Already on 'master'
labuser@ubuntu:~/Desktop/Persistant_Folder/demo_repo$ git merge dev
Already up to date.
labuser@ubuntu:~/Desktop/Persistant_Folder/demo_repo$ git push origin master
Total 0 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/vigneshviky01/demo_repo/pull/new/master
remote:
To github.com:vigneshviky01/demo_repo.git
 * [new branch]      master -> master
```

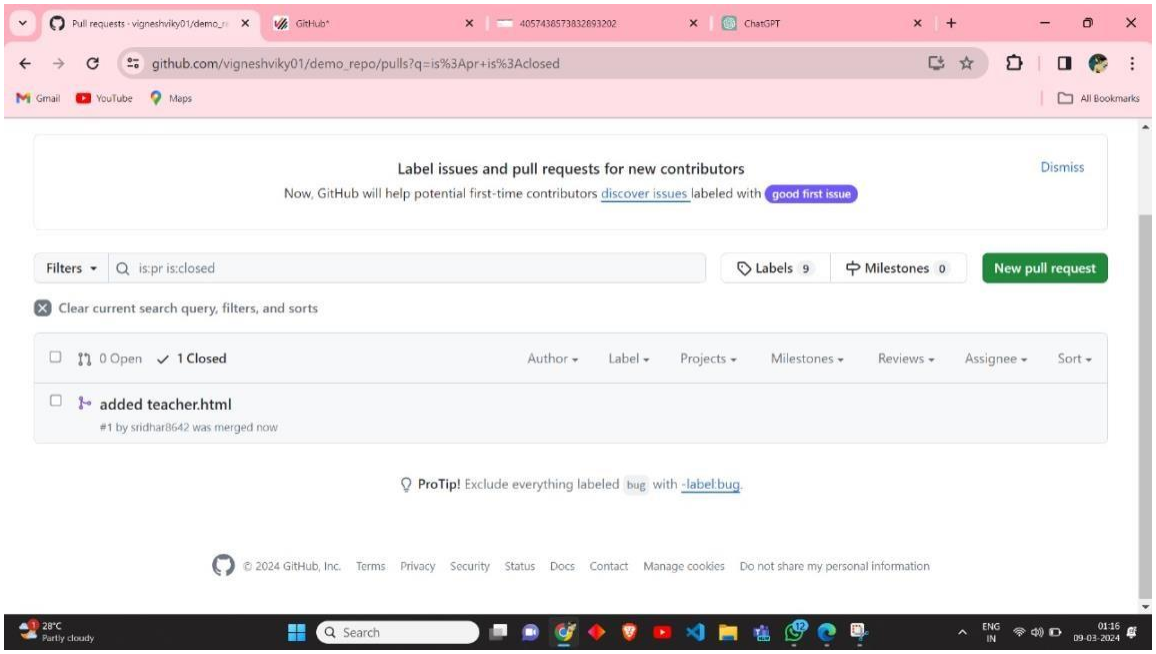
9. Managing Updates from Remote

Use git fetch and git pull to get the latest version of code from remote.



10. Creating Pull Requests

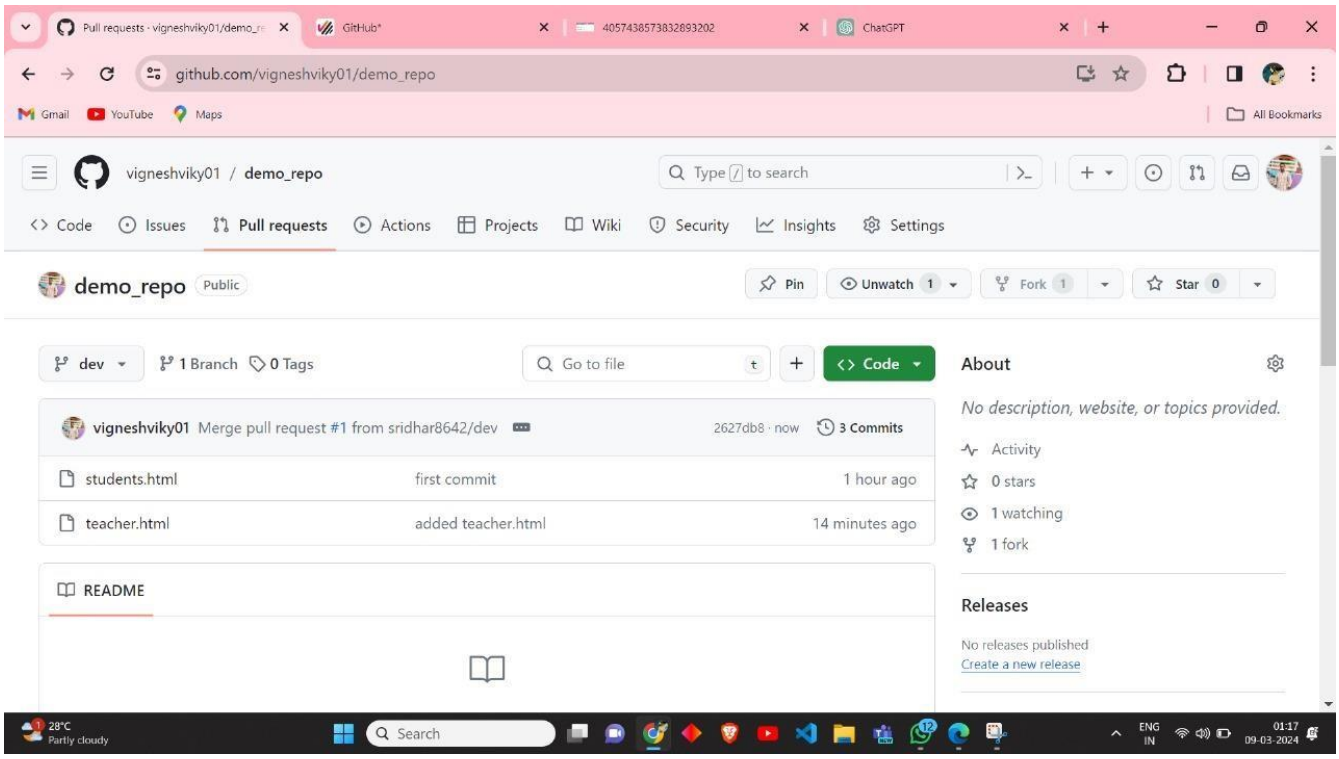


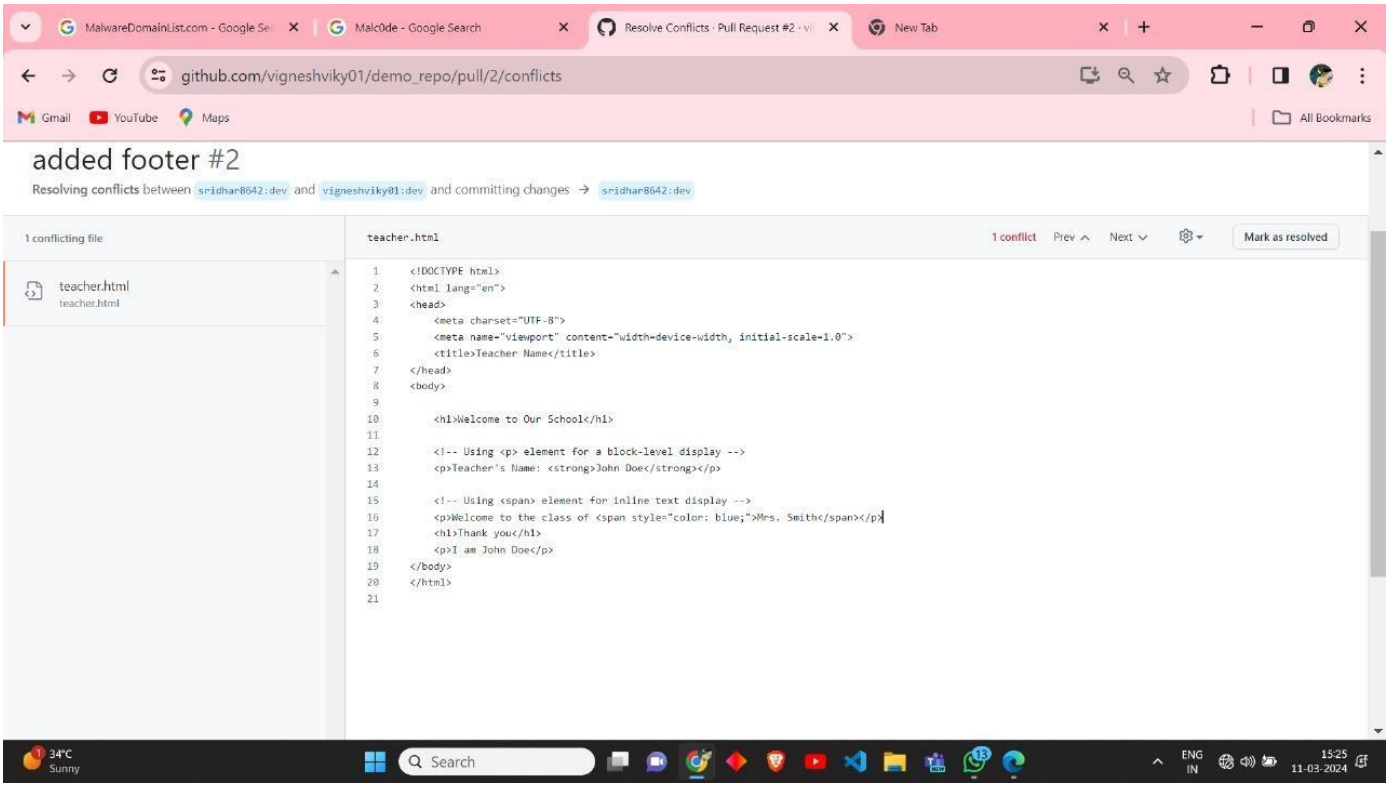


Create pull requests on GitHub to merge branches or forked repos. Choose base and compare branches.

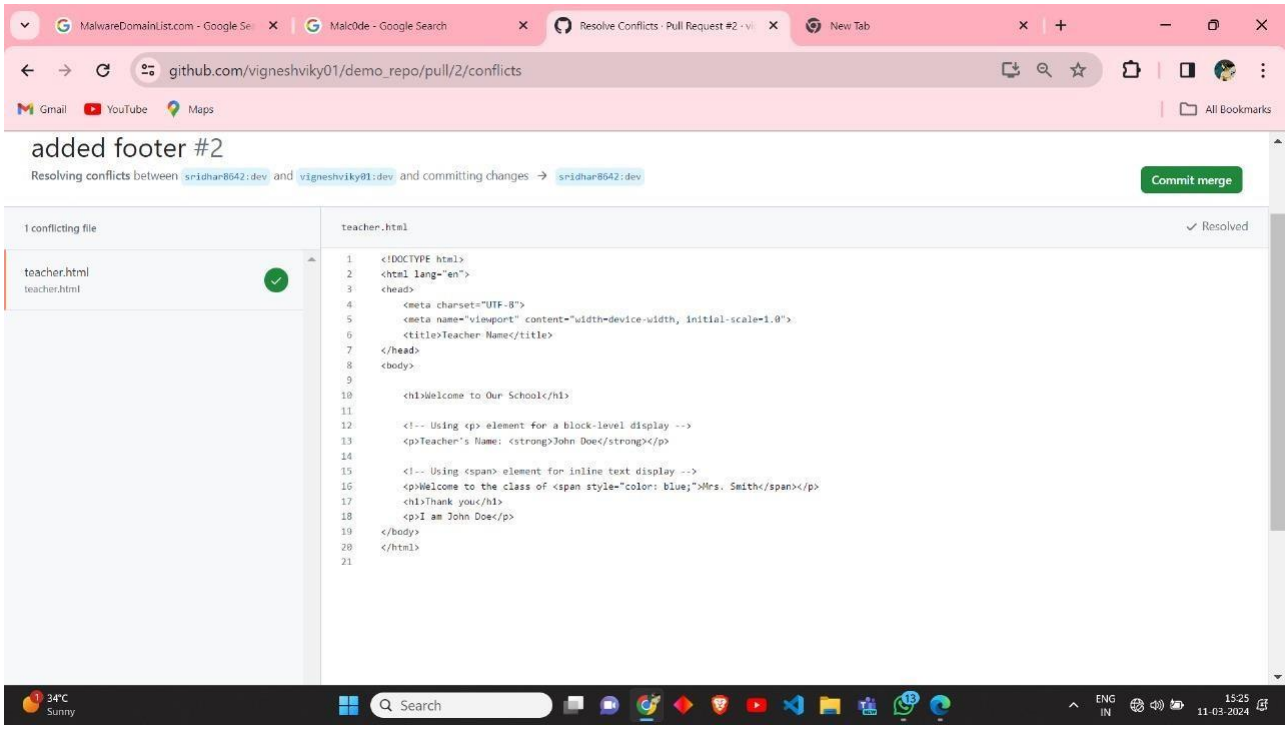
- Navigate to Repository: Go to the GitHub repository where you want to create the pull request.
- Click on Pull Requests Tab: Find and click on the “Pull Requests” tab on the repository’s page.
- Initiate New Pull Request: Click on the “New pull request” button.

- Choose Base and Compare Branches: Select the base branch (where you want to merge changes) and the compare branch (the branch containing the changes you want to merge).
- Review Changes: GitHub will display the changes between the two branches, allowing you to review them before merging.
- Title and Description: Provide a clear title and description for the pull request, explaining the purpose of the changes.
- Optional Settings: Optionally, you can assign reviewers, assignees, and labels to the pull request.
- Create Pull Request: Click on the “Create pull request” button to submit the pull request.
- Once the pull request is created, collaborators can review the changes, provide feedback, and discuss any necessary adjustments before merging the changes into the base branches..



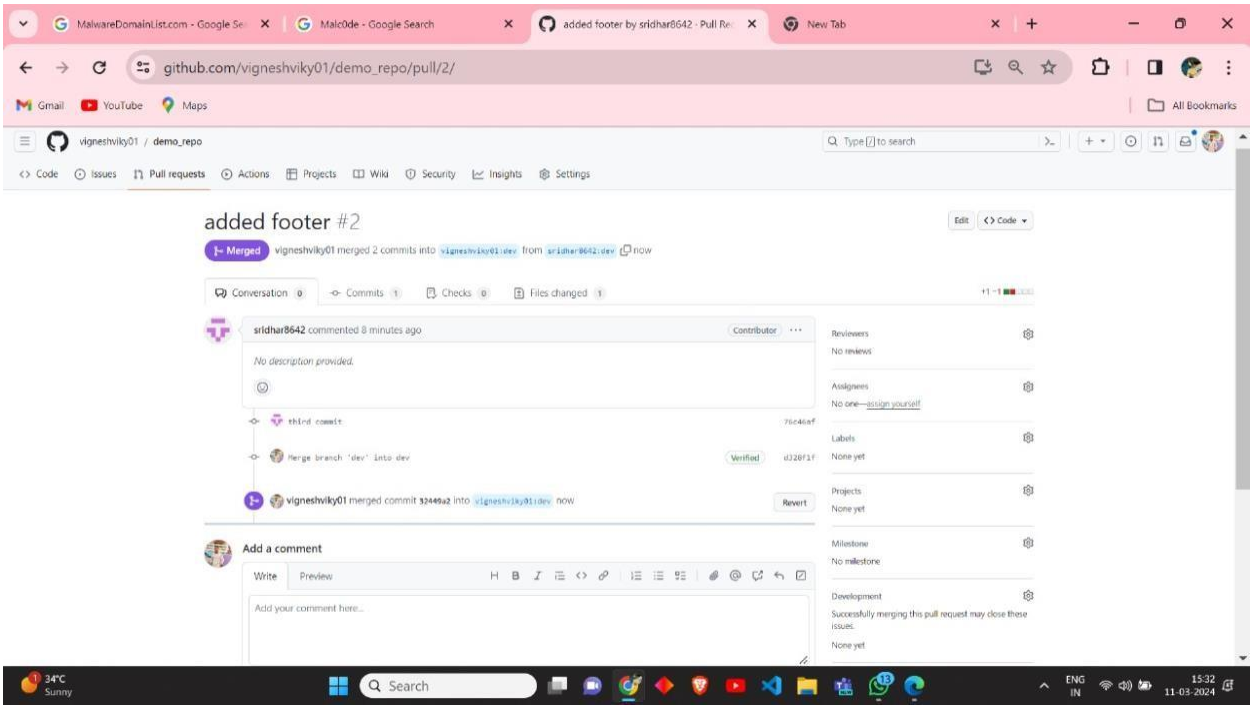

MalwareDomainList.com - Google Se
Malc0de - Google Search
Resolve Conflicts · Pull Request #2 · vi
New Tab
github.com/vigneshviky01/demo_repo/pull/2/conflicts
Gmail
YouTube
Maps
All Bookmarks
added footer #2
Resolving conflicts between sridhar8642:dev and vigneshviky01:dev and committing changes → sridhar8642:dev
1 conflicting file
teacher.html
teacher.html
1 conflict
Prev
Next
Mark as resolved
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Teacher Name</title>
</head>
<body>
<h1>Welcome to Our School</h1>
<!-- Using <p> element for a block-level display -->
Teacher's Name: John Doe
<!-- Using <span> element for inline text display -->
Welcome to the class of Mrs. Smith
<h1>Thank you</h1>
I am John Doe
</body>
</html>
34°C
Sunny
Search
ENG
IN
15:25
11-03-2024

11. Merging


MalwareDomainList.com - Google Se
Malc0de - Google Search
Resolve Conflicts · Pull Request #2 · vi
New Tab
github.com/vigneshviky01/demo_repo/pull/2/conflicts
Gmail
YouTube
Maps
All Bookmarks
added footer #2
Resolving conflicts between sridhar8642:dev and vigneshviky01:dev and committing changes → sridhar8642:dev
Commit merge
1 conflicting file
teacher.html
teacher.html
Resolved
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Teacher Name</title>
</head>
<body>
<h1>Welcome to Our School</h1>
<!-- Using <p> element for a block-level display -->
Teacher's Name: John Doe
<!-- Using <span> element for inline text display -->
Welcome to the class of Mrs. Smith
<h1>Thank you</h1>
I am John Doe
</body>
</html>
34°C
Sunny
Search
ENG
IN
15:25
11-03-2024

Merge branches using GUI or CLI, typically found under the pull request section on GitHub.

Collaboration Methods



Review Changes: Take the time to thoroughly review the code changes proposed in the pull request. Ensure they align with project goals, coding standards, and any established guidelines.

Resolve Conflicts (if any): In cases where there are conflicting changes between the pull request and the target branch, address these conflicts by manually resolving them. GitHub provides tools to assist in this process.

Choose Merge Method: Depending on your project's preferences and requirements, select the appropriate merge method:

Merge Commit: Retains the commit history of both branches by creating a new merge commit.

Squash and Merge: Condenses all commits from the pull request into a single commit before merging. This helps maintain a cleaner commit history.

Rebase and Merge: Incorporates the changes from the pull request onto the target branch by rewriting the commit history. This can result in a more linear and streamlined history.

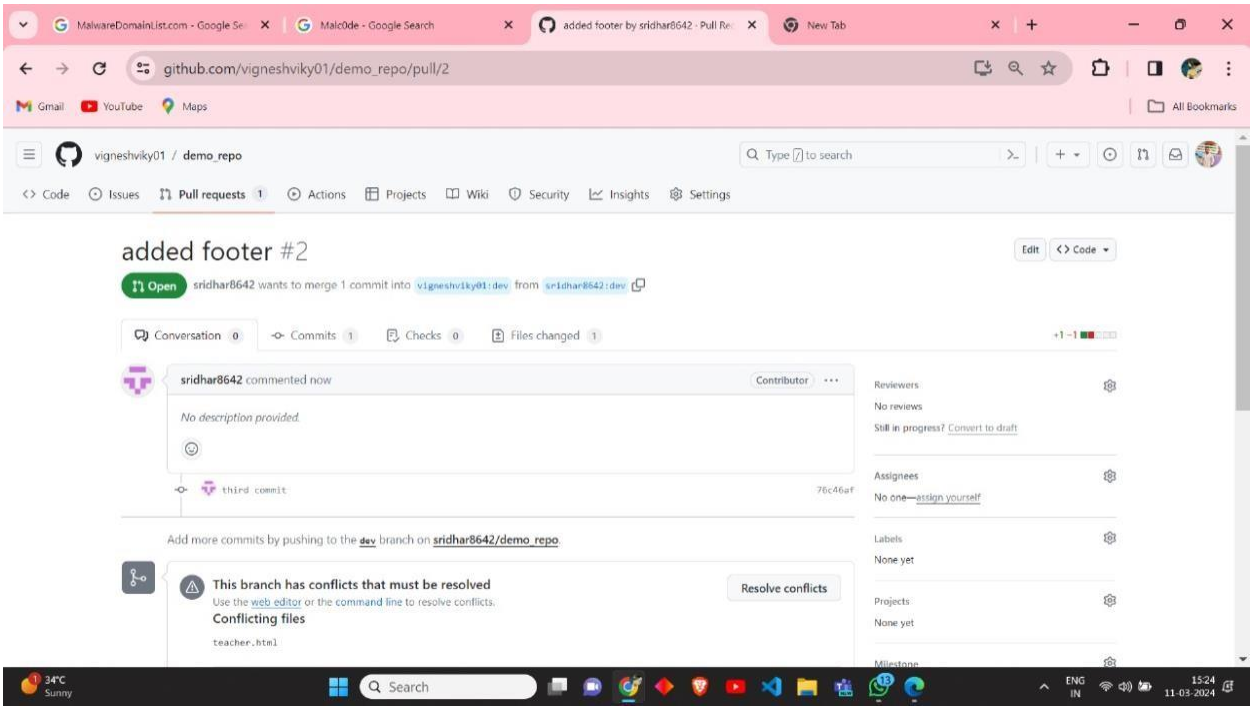
Click “Merge” Button: Once you’ve resolved any conflicts and selected the desired merge method, click the “Merge” button on the GitHub interface.

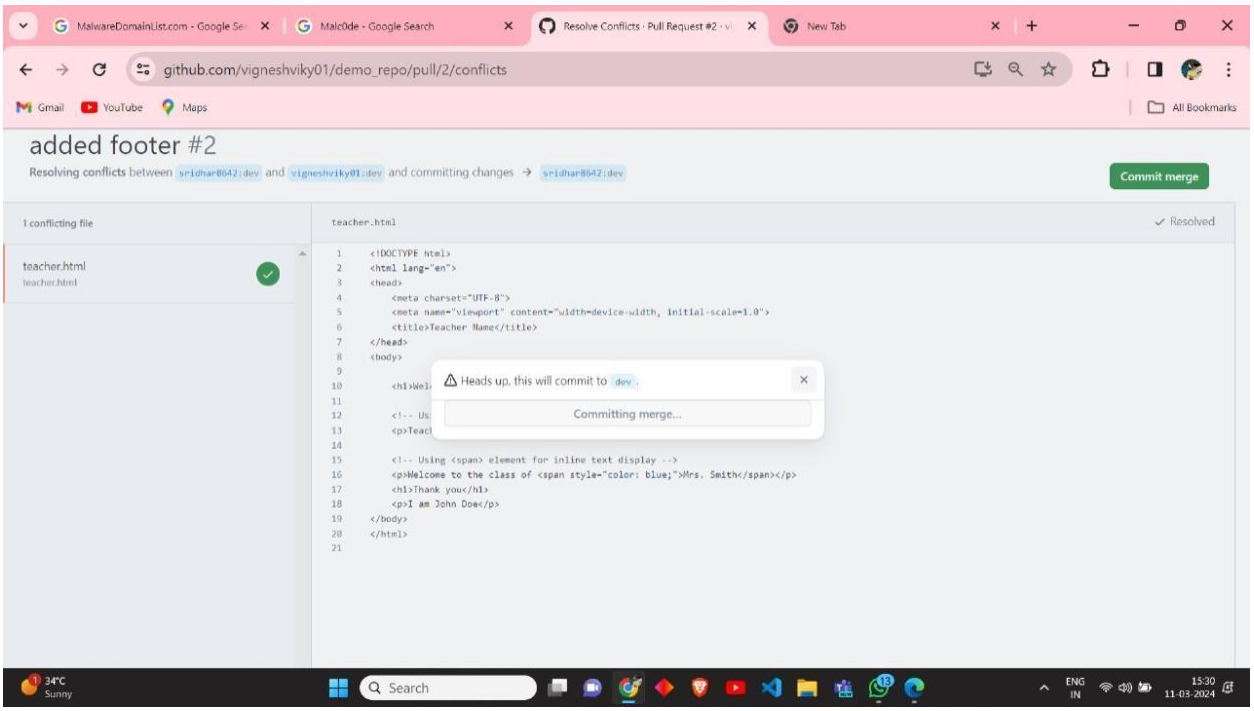
Confirm Merge: Confirm the merge action to proceed. This ensures that the merge is intentional and prevents accidental merges.

Close Associated Issues (Optional): If the pull request addresses specific issues or feature requests, consider closing them once the changes are successfully merged. This helps maintain an organized and up-to-date issue tracker.

Verify Changes: After the merge is complete, double-check the target branch to ensure that the changes have been successfully applied. Review the commit history and verify that the code functions as expected.

By following these steps, you can effectively manage the merging process on GitHub, ensuring that code changes are seamlessly integrated while maintaining project quality and integrity.





11. Collaboration

There are some ways to collaborate on GitHub.

Branching and Merging:

In this way, the collaborators have access to the source repository and have their Own branches. They work on their own branch and send pull requests to merge it to the Main branch. The owner of the source branch invites collaborators to work with the Source repo by sending an invitation.

Forking and Pull request:

In this way, the source repository is forked by the collaborator (a copy is created In collaborators' GitHub account). They work on the forked repository and send pull Request to the source branch. In this method, the owner of the source branch only can

Do the merging. This method is widely used in OpenSource Development.

12. Handling Merge Conflicts

Resolve merge conflicts manually when collaborators make conflicting changes to the same file.