**DIGGIBYTE TECHNOLOGIES**

****

*GIT ASSIGNMENT REPORT*

**Prepared By:**

**Name: Divakar C  
Designation: Trainee Engineer**

**GIT ASSIGNMENT**

**Question 1**

Step 1: I created a new Git repository.

A screenshot of a computer

AI-generated content may be incorrect.

Step 2: I created a file and committed the initial changes.

A computer screen with white text

AI-generated content may be incorrect.

Step 3: I viewed the commit history using git log to confirm the commit.

**A computer screen with white text and numbers

AI-generated content may be incorrect.**

Step 4: I reopened the file and made some changes to it.

Step 5: I executed git status and verified that the file was marked as "modified" and unstaged.

A screen shot of a computer

AI-generated content may be incorrect.

Step 6: I staged the modified file using git add and committed the changes with git commit.

A black screen with white text

AI-generated content may be incorrect.

Step 7: I successfully cloned the repository from GitHub using git clone.

A black screen with white text

AI-generated content may be incorrect.

Step 8: I navigated into the cloned repository using the command line and successfully used git fetch to retrieve any changes made to the original repository since it was cloned.

A black background with white text

AI-generated content may be incorrect.

Step 9: I pulled the latest changes from the remote repository into my local copy using the git pull command, successfully merged them, and then pushed my own local changes to the remote repository using git push.

A computer screen shot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

**Summary:**

1. I created a Git repository and added a file with initial commit.
2. Then I made some changes, checked its modified status using git status, and committed the changes.
3. I cloned the GitHub repo and used git fetch and git pull to sync the latest changes.
4. All operations were successfully performed using the command line.

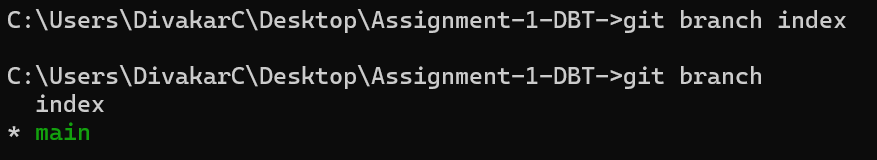
**Question 2**

Step 1: I cloned the repository from GitHub using the git clone command.

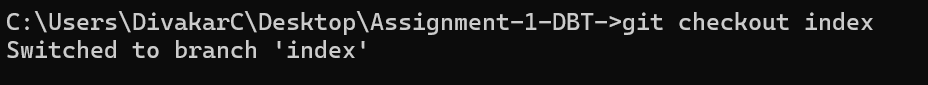
A black screen with white text

AI-generated content may be incorrect.

Step 2: Created a new branch using git branch.



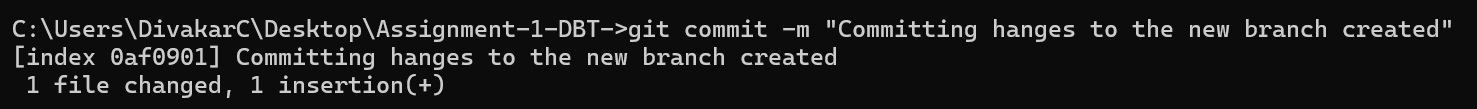
Step 3: Switched to the new branch using git checkout index.



Step 4: Made changes to the file locally.



Step 5: Committed the changes to the new branch with git commit.



Step 6: Switched back to the original branch (main) using git switch main.

A black background with white text

AI-generated content may be incorrect.

Step 7: Merged the new branch into the main branch using git merge index.

A black screen with white text

AI-generated content may be incorrect.

Step 8: Pushed the merged changes to the remote main branch using git push.

A black background with white text

AI-generated content may be incorrect.

**Summary:**

1. I cloned the repository and created a new branch from it.
2. After switching to the branch, I made file changes and committed them.
3. Then I switched back to the main branch, merged the changes, and pushed them.
4. The process verified successful collaboration between branches.

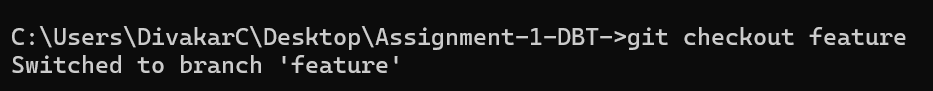
**Question 3**

Step 1: I created a feature branch using git branch feature.

A black screen with white text

AI-generated content may be incorrect.

Step 2: Switched to the new branch.



Step 3: Opened the file and made necessary changes.

Step 4: Added and committed the changes using git add . and git commit -m "Greet file updated to new feature branch".

A black screen with white text

AI-generated content may be incorrect.

Step 5: Pushed the changes to the remote feature using git push origin feature.

A computer screen shot of a black screen

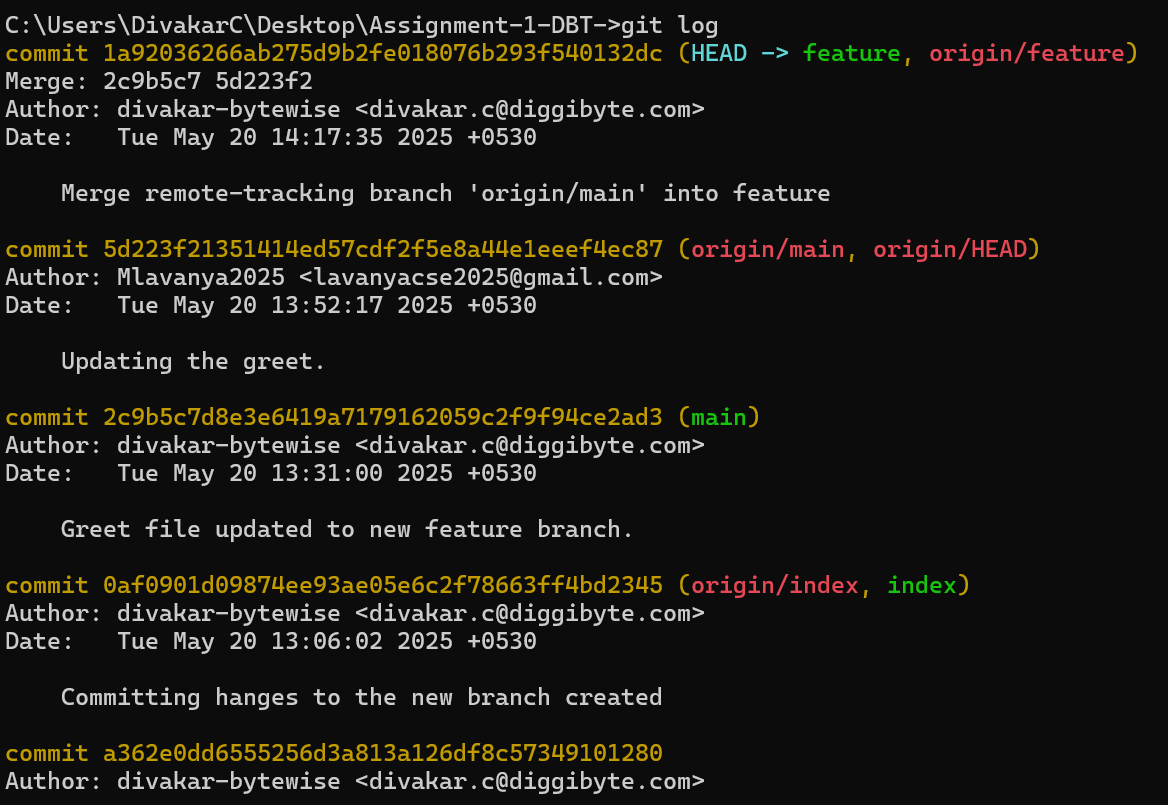
AI-generated content may be incorrect.

Step 6: Created a pull request from feature-branch to main on GitHub.

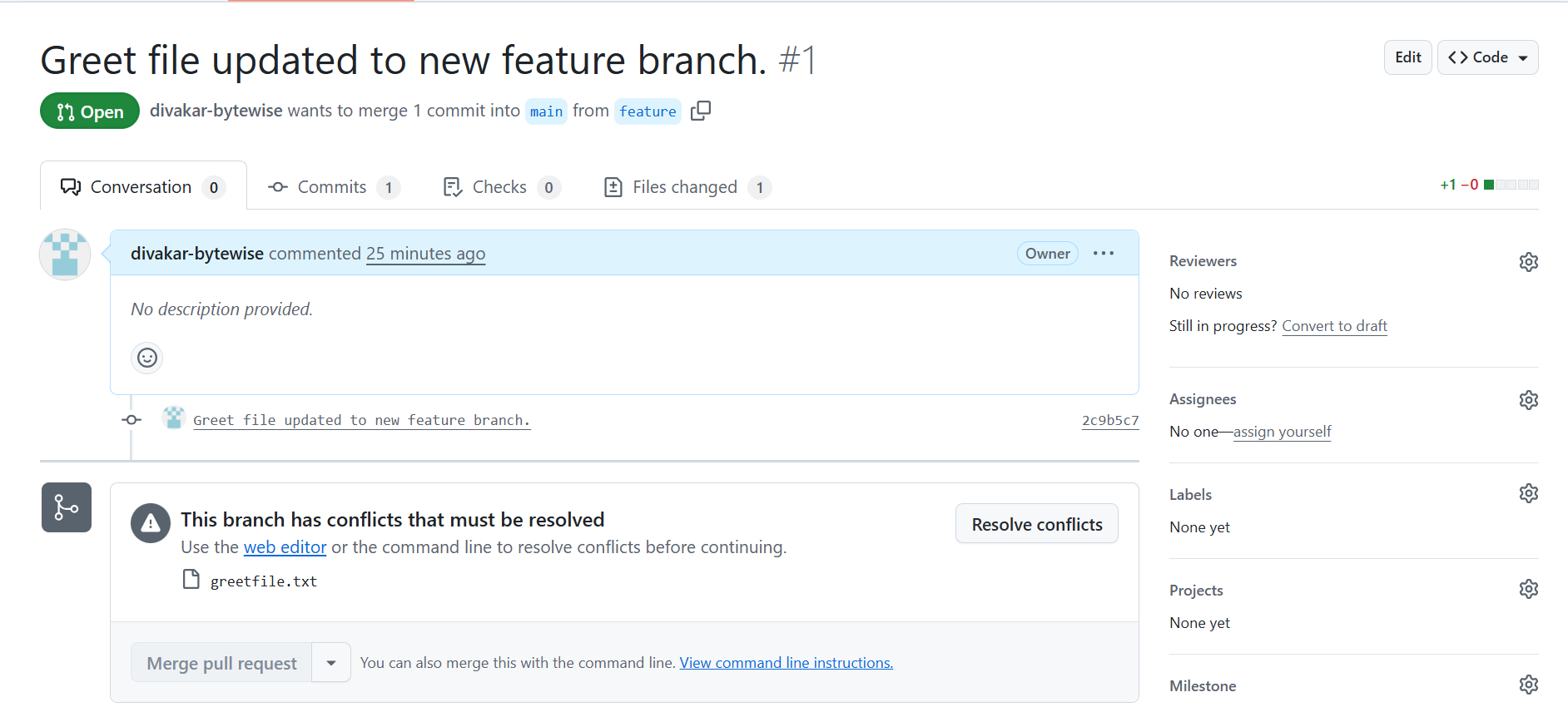
A screenshot of a computer

AI-generated content may be incorrect.

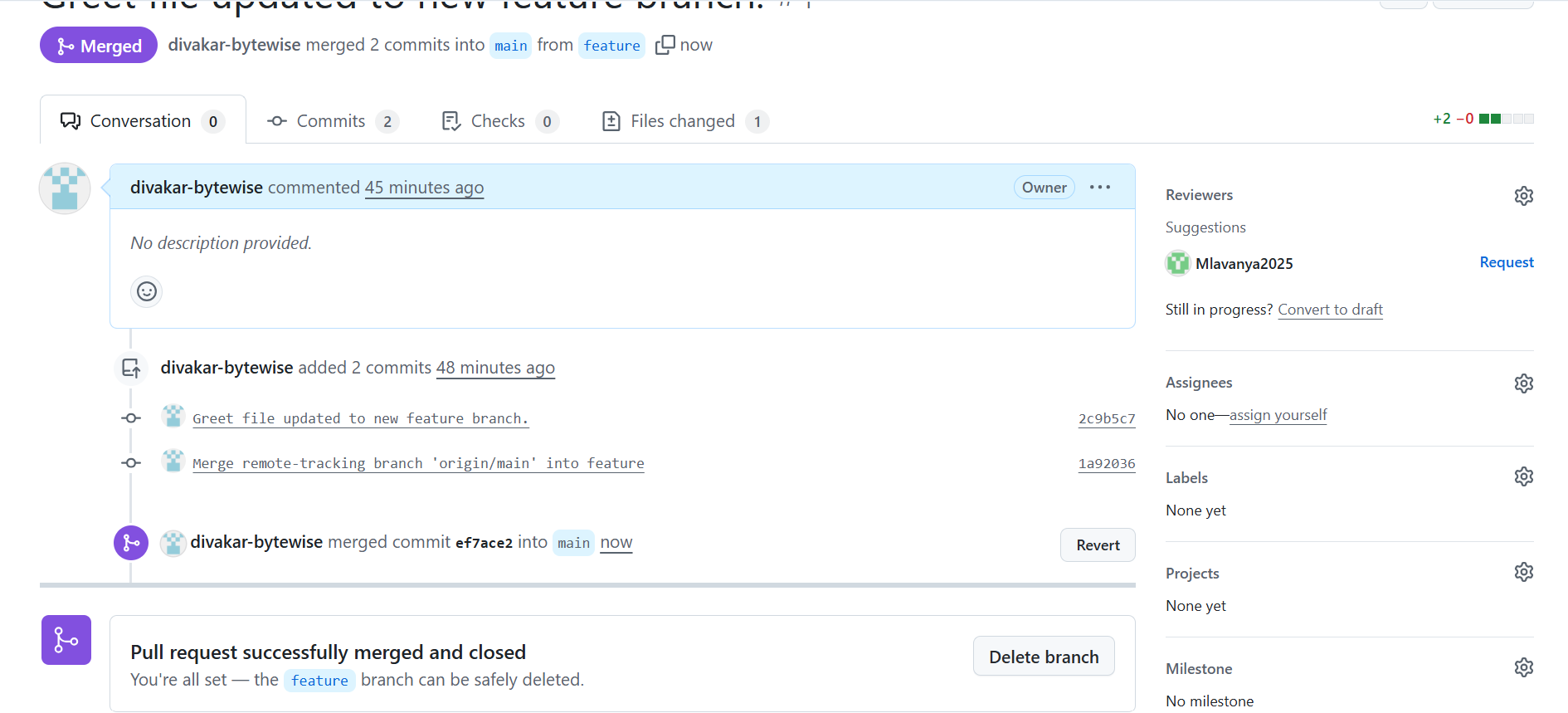
Step 7: As another user (or from another branch), made changes to the same file in the main branch.



Step 8: Added, committed, and pushed the changes to the main branch.



Step 9: Successfully resolved the conflict manually by editing the conflicted file, staging the changes using git add . , and continuing the rebase with git rebase. The updated branch was then pushed using git push origin feature.



**Summary:**

1. I created a feature branch and committed changes to a file.
2. Pushed those changes and raised a pull request.
3. Simultaneously, as another user, I made changes in the master branch causing a conflict.
4. Resolved the conflict using git rebase successfully and finalized the merge.

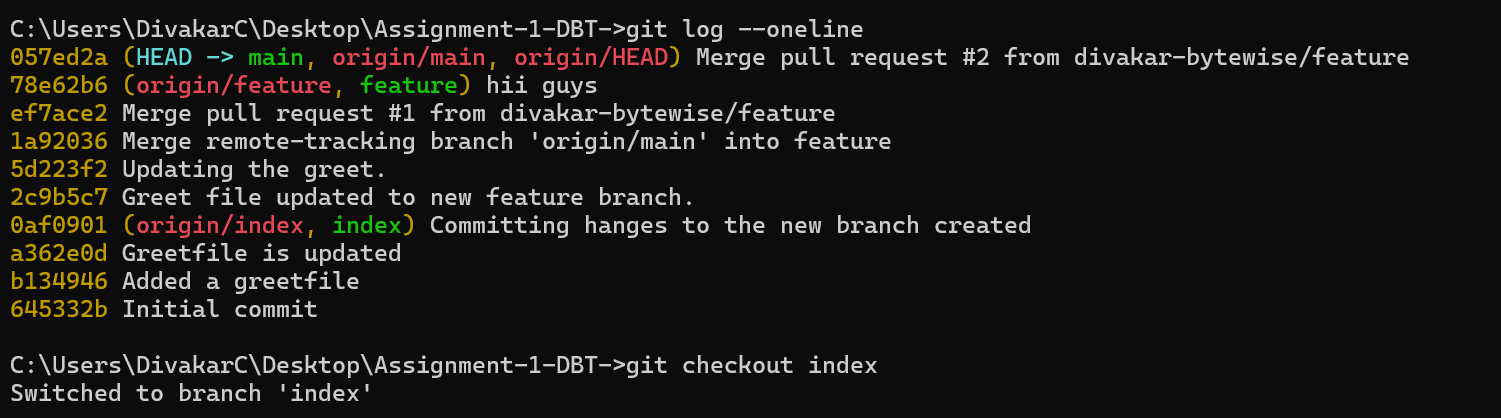
**Question 4**

Step 1: I created a new branch named index.

A black background with white text

AI-generated content may be incorrect.

Step 2: I switched to the index branch.  
 – I opened the file and made some changes.  
 – I added and committed the changes to the index branch.



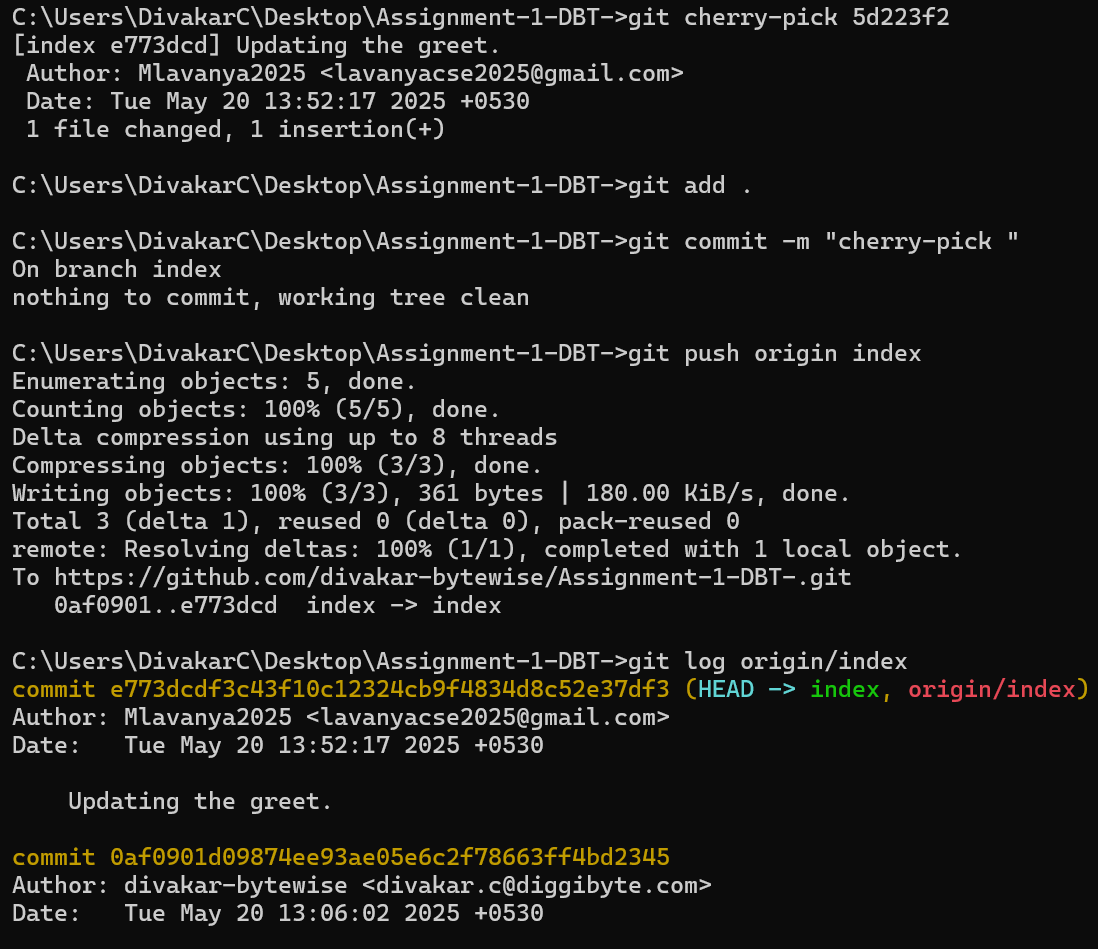
Step 3: I identified the specific commit(s) that I wanted to cherry-pick and noted down their commit hashes.

A black screen with white text

AI-generated content may be incorrect.

Step 4: I switched to the target branch using the git checkout command.

Step 5: I applied the changes using the git cherry-pick command with the respective commit hash(es).



**Summary:**

1. I created and switched to the index branch, made multiple commits to a file.
2. Then I identified specific commits to cherry-pick using git log.
3. I switched to the target branch and applied selected commits using git cherry-pick.
4. The cherry-picked changes were successfully applied without conflicts.

**Question 5**

Step 1: I created a new branch named index.

A black background with white text

AI-generated content may be incorrect.

Step 2: I switched to the index branch.  
 – I opened a file and made changes to it.  
 – I added and committed the changes to the index branch.

A computer screen with white text

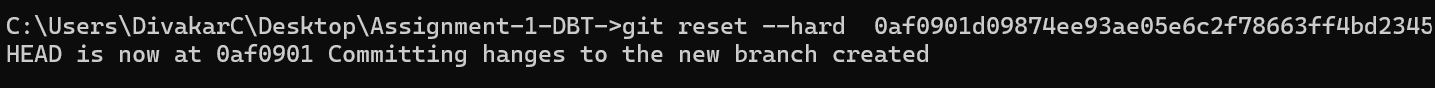
AI-generated content may be incorrect.

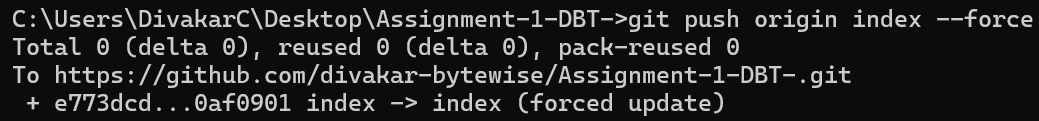
Step 3: I used the git log command to view the commit history and identified the commit to which I wanted to reset.

A screenshot of a computer program

AI-generated content may be incorrect.

Step 4: I used the git reset command with the desired reset type and the target commit hash to reset my branch.





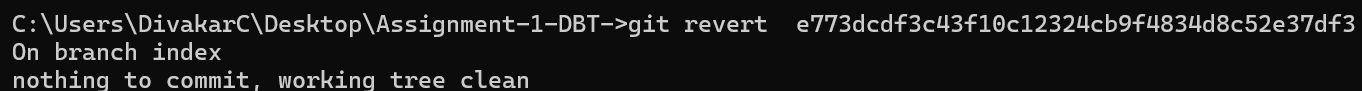
Step 5: I verified that the reset was successful using the git log command again.

Step 6: I again used git log to find the commit I wanted to revert.

A screenshot of a computer program

AI-generated content may be incorrect.

Step 7: I used the git revert <commit-hash> command to reverse the selected commit.



Step 8: I confirmed the successful revert by running git log once more.

A screen shot of a computer program

AI-generated content may be incorrect.

**Summary:**

1. On the index branch, I made and committed multiple changes to a file.
2. Then I identified an old commit and used git reset to revert the history to that point.
3. I also used git revert on a specific commit and verified changes using git log.
4. Observed that git reset changes history, while git revert adds a new commit to reverse.