**Placement Empowerment Program**

***Cloud Computing and DevOps Centre***

**Set Up Git Branching**Create a new branch in your Git repository for testing. Add a new feature and merge it.

Name: Yasvanth Rajan E Department: AML



**Introduction and Overview**

In this Proof of Concept (POC), we will walk through the steps required to upload an HTML file to a GitHub repository and perform a Git merge operation. This process will help us understand how to manage version control for a static website and effectively handle feature development and collaboration using Git.

**Objectives**

* To demonstrate how to initialize a local Git repository and upload files to GitHub.
* To show how to create, commit, and push changes to a remote repository.
* To explain how to manage multiple branches and merge them in Git.
* To emphasize the importance of version control and collaboration using Git.

**Importance**

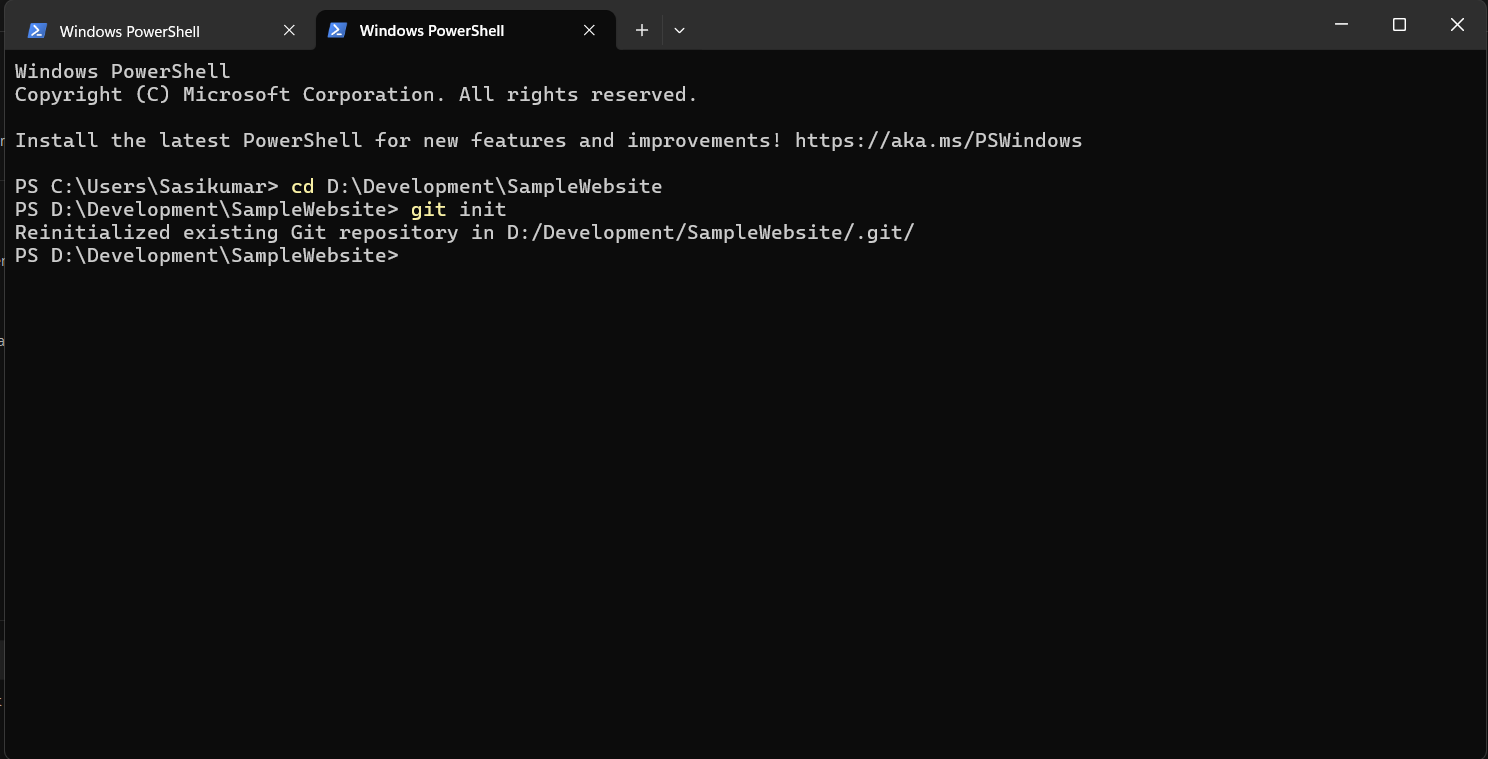
* **Version Control**: Git enables tracking and managing changes to code over time. It ensures that multiple versions of a file can be accessed, modified, and restored if necessary.
* **Collaboration**: Git and GitHub make it easier for multiple developers to collaborate on the same project, handle merges, and ensure that everyone is working on the latest version.

**STEPS:**

**STEP 1: Initialize a Git Repository Locally**

* Navigate to the folder where your static website is located.
* Run the following command to initialize a Git repository:

git init

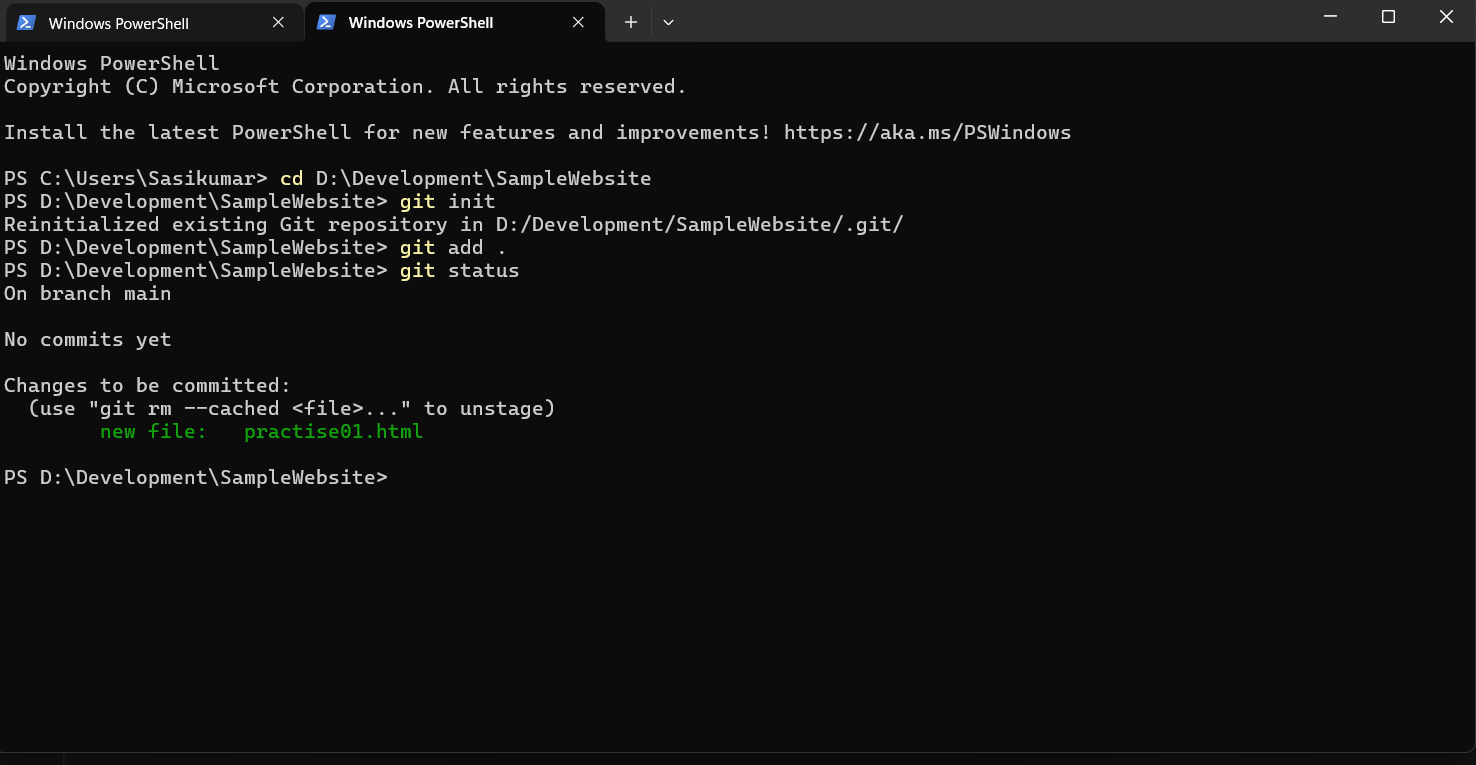


**STEP 2: Add HTML File to the Repository**

* Create or add your HTML file (e.g., index.html) in the local repository folder.
* Run the following command to add the file to staging and to check whether the file is uploaded successfully:

git add index.html (or) . [to add all files]

git status

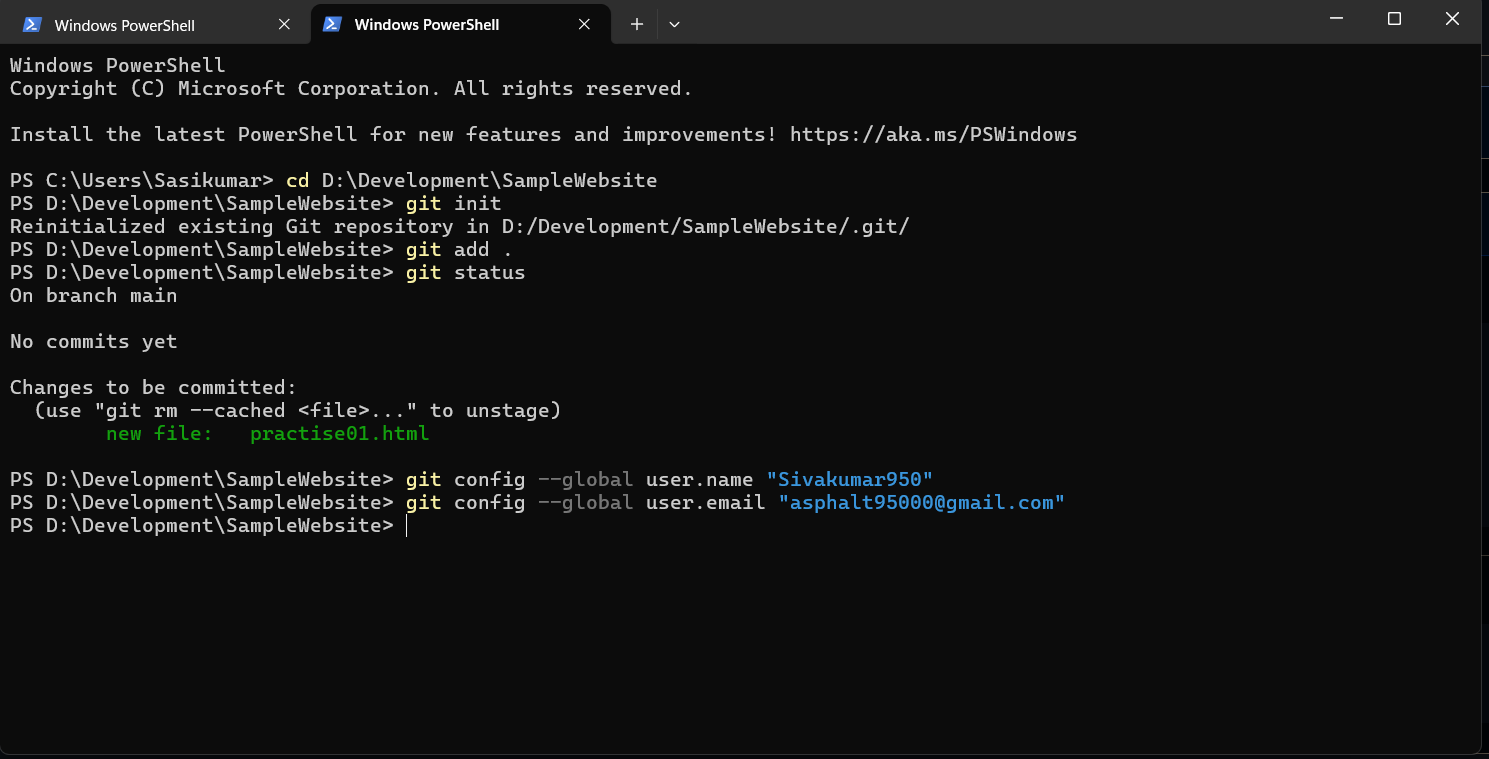


**STEP 3: Configure Git Settings:**

* Set up your Git username and email (if not configured already):

git config --global user.name "Your Name"

git config --global user.email [your.email@example.com](mailto:your.email@example.com)



**STEP 4: Create a Remote GitHub Repository**

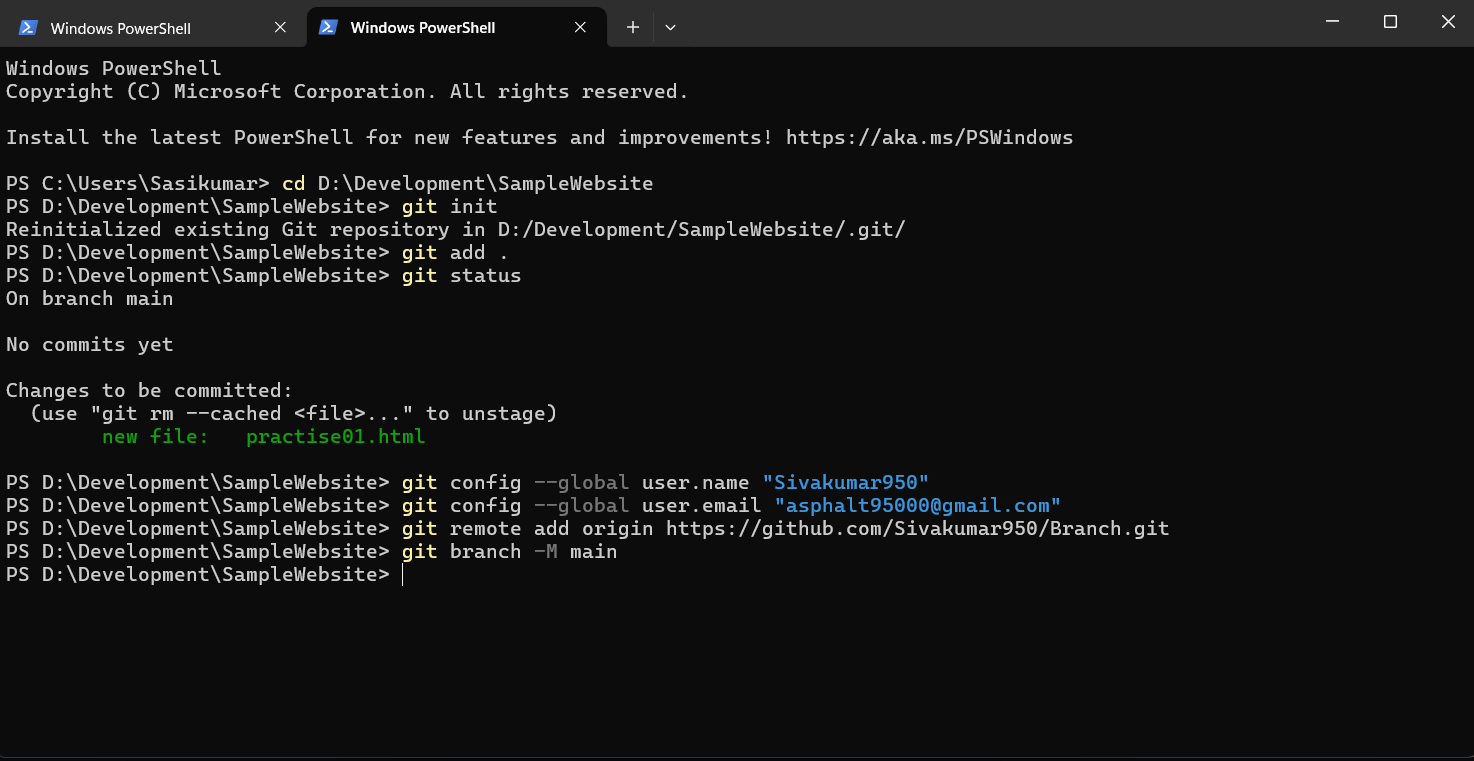
* Go to GitHub and create a new repository.

**STEP 5: Link Local Repository to Remote GitHub Repository**

* Run the following command to add the remote repository and create a main branch:

git remote add origin <https://github.com/username/repository-name.git>

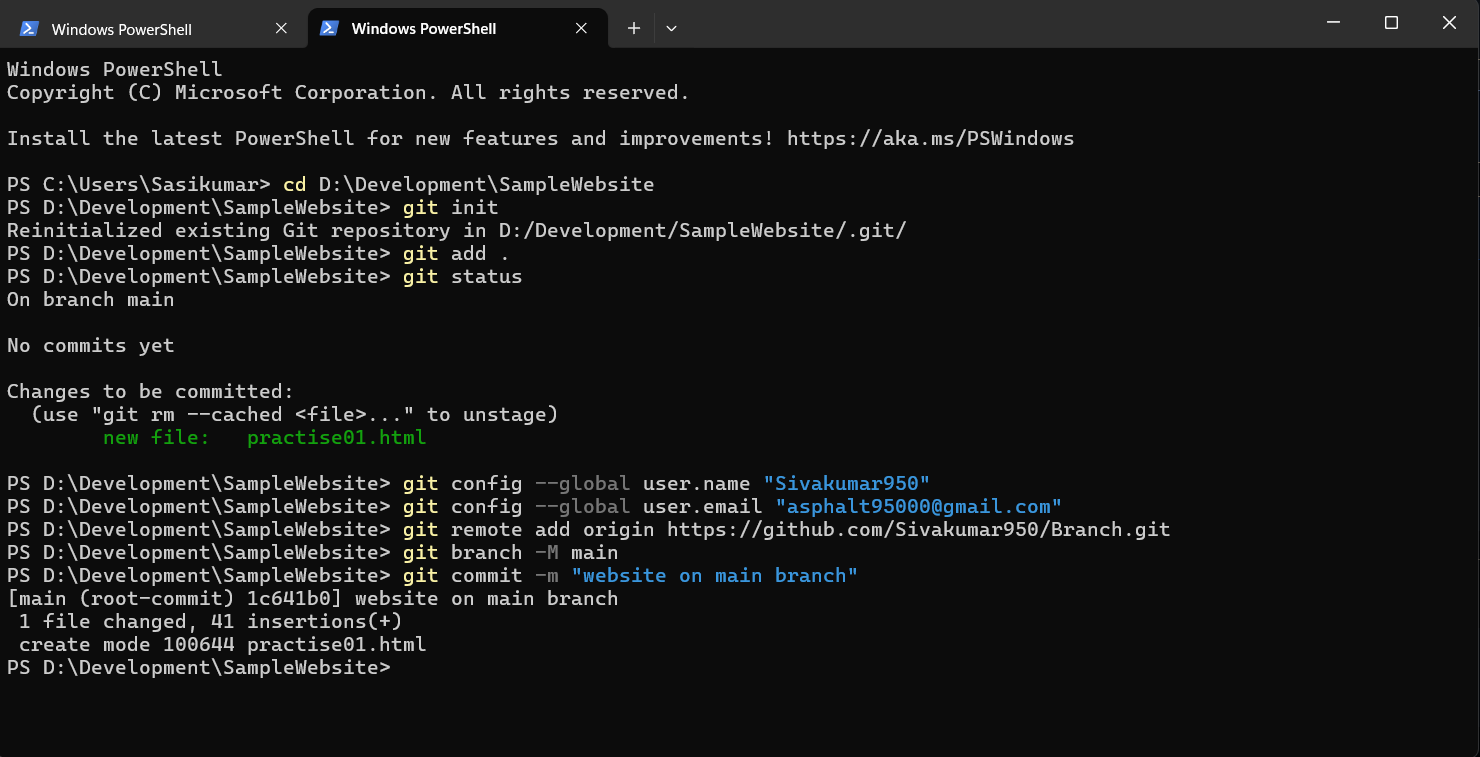
git branch -M main



**STEP 6: Commit the Changes**

* Commit the changes with a meaningful commit message:

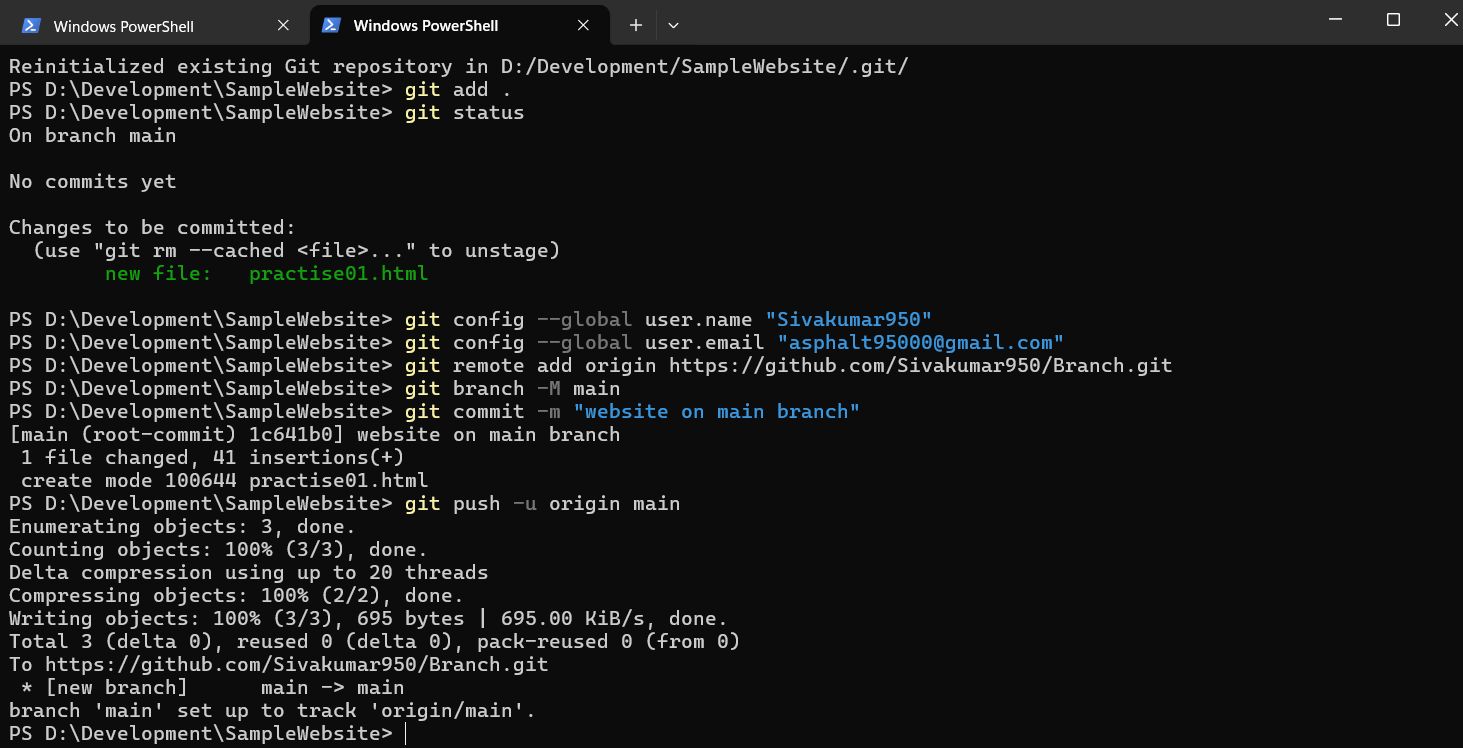
git commit -m "Add index.html"



**STEP 6: Push the HTML File to GitHub**

* Push the changes to your GitHub repository:

git push -u origin main

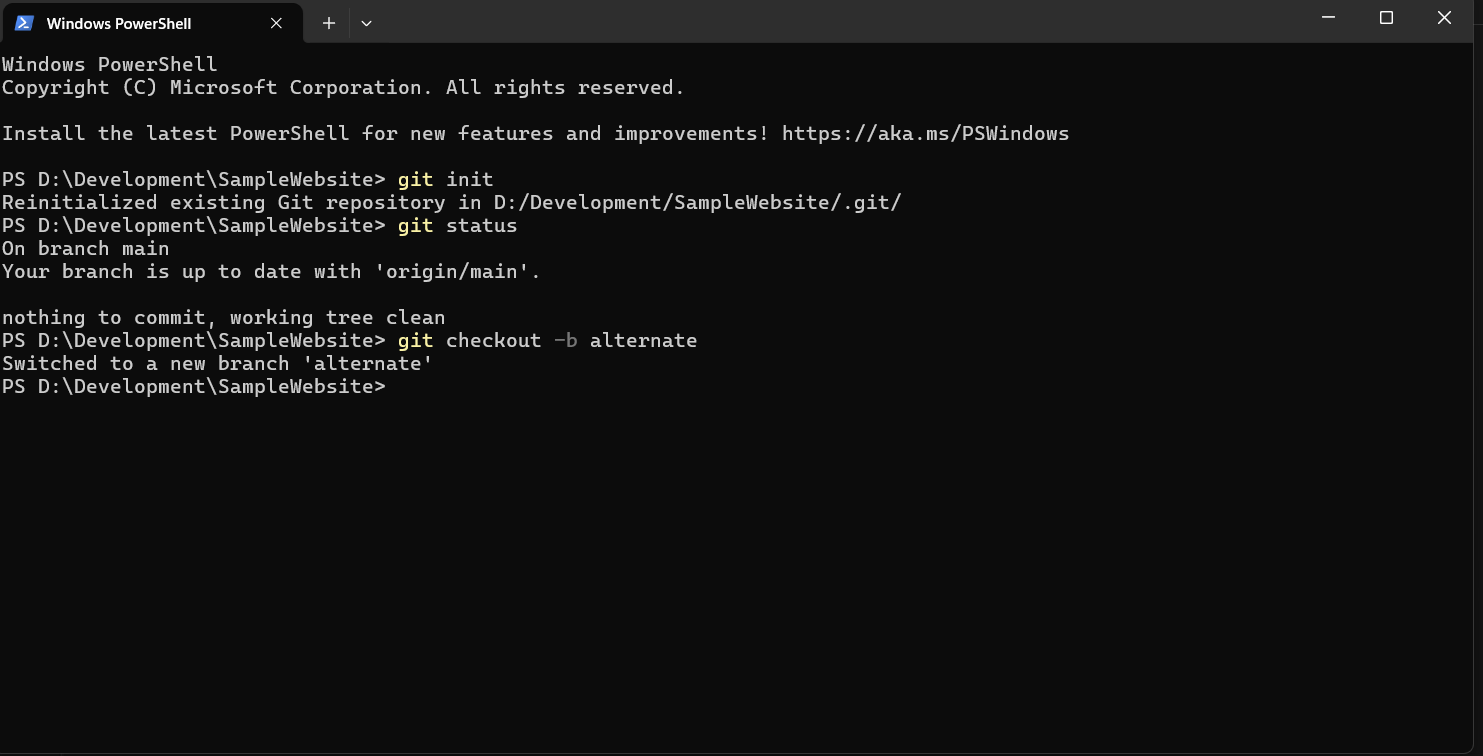


**MERGE TASK STEPS**

**STEP 1: Create a New Branch for Feature Development**

* To start working on a new feature or fix, create a new branch:

git checkout -b new-branch



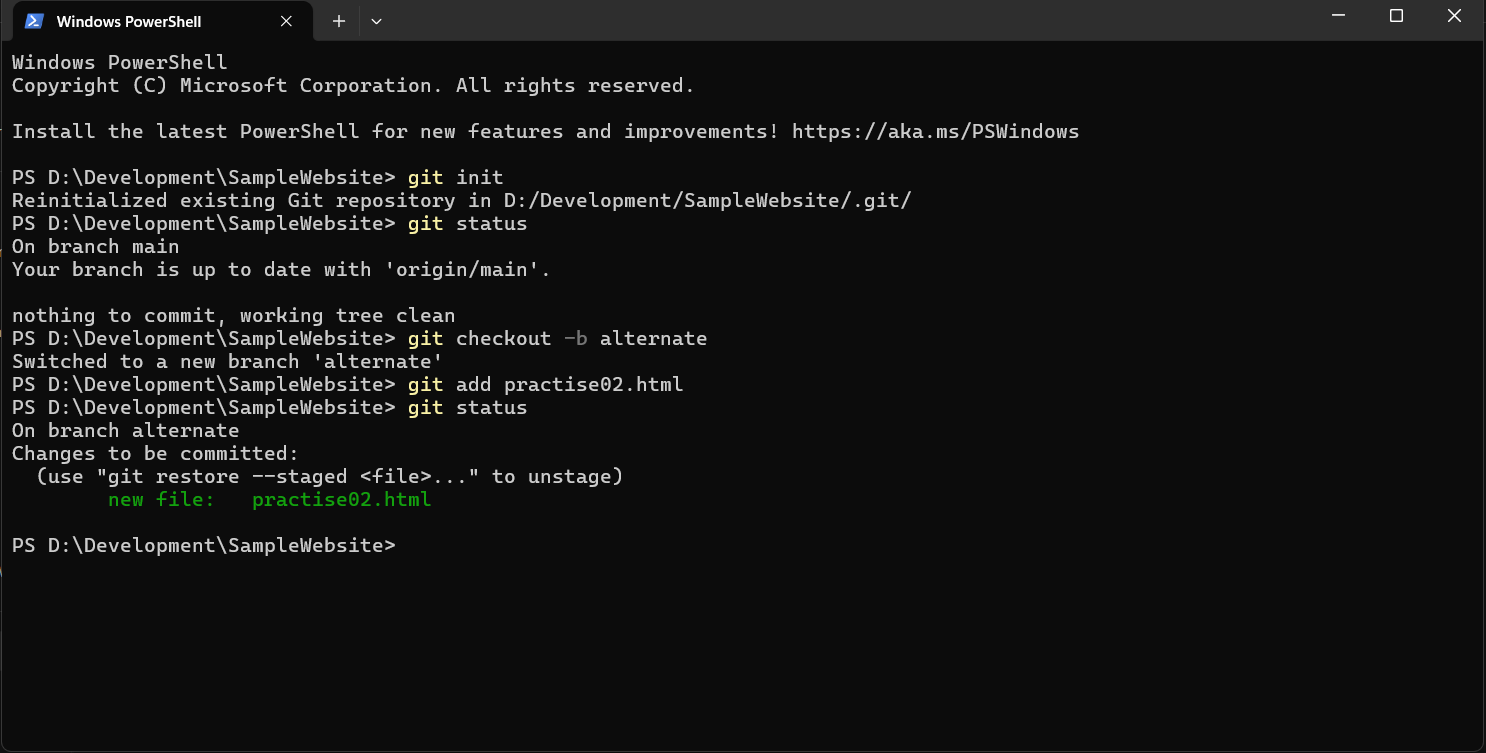
**STEP 2: Make Changes in the New Branch**

* Edit your files or create new files as part of your feature development.

**STEP 3: Add and Commit the Changes**

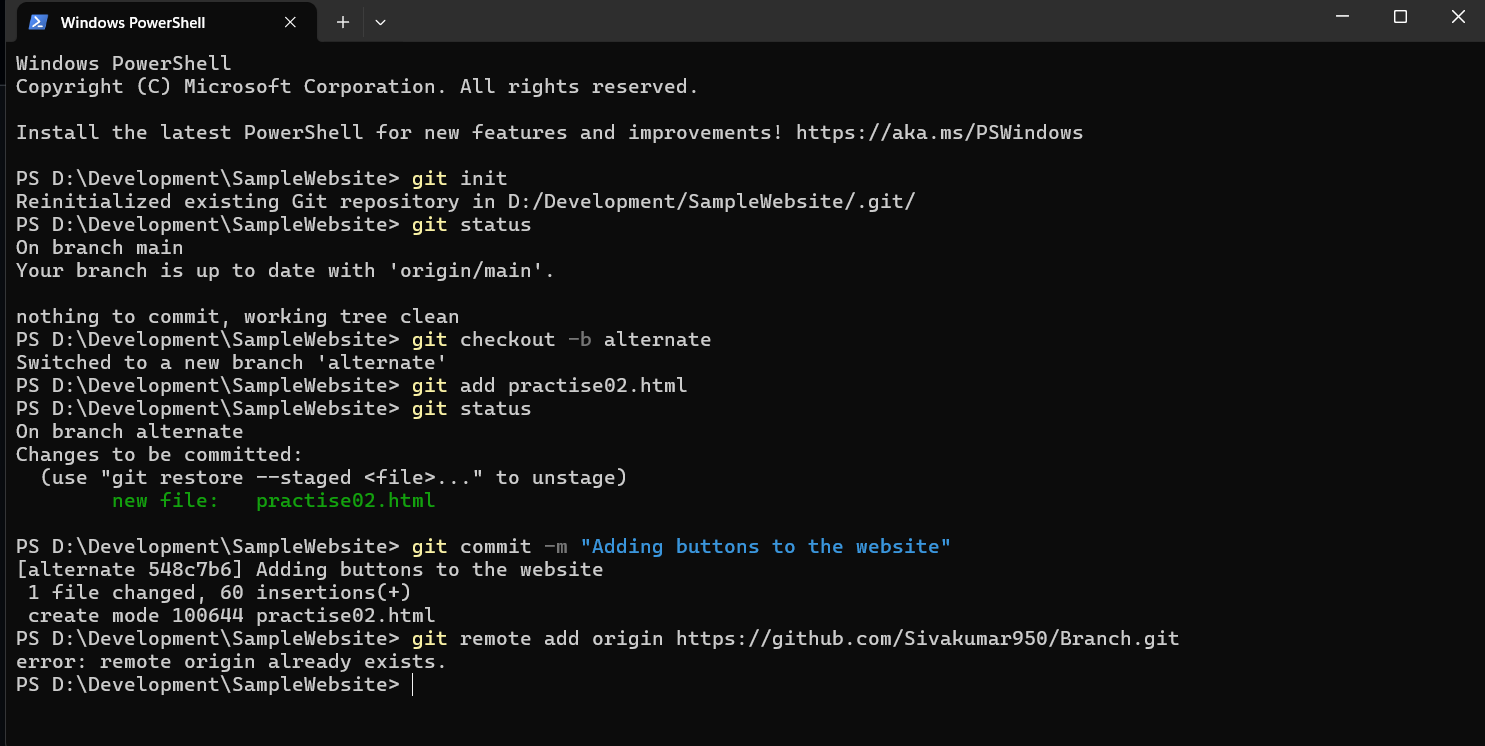
* Add the modified files to staging:

git add file\_name.html



* Commit the changes:

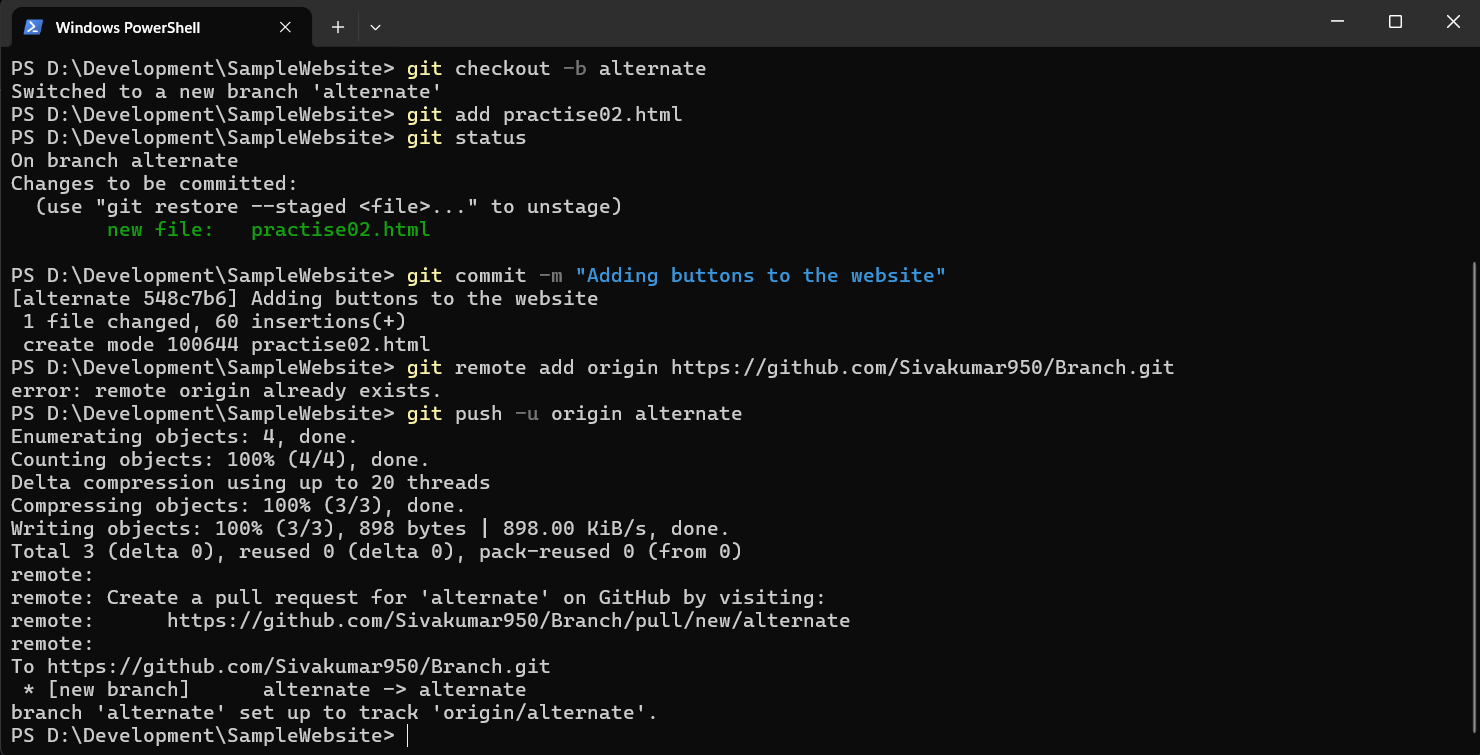
git commit -m "Add new feature"

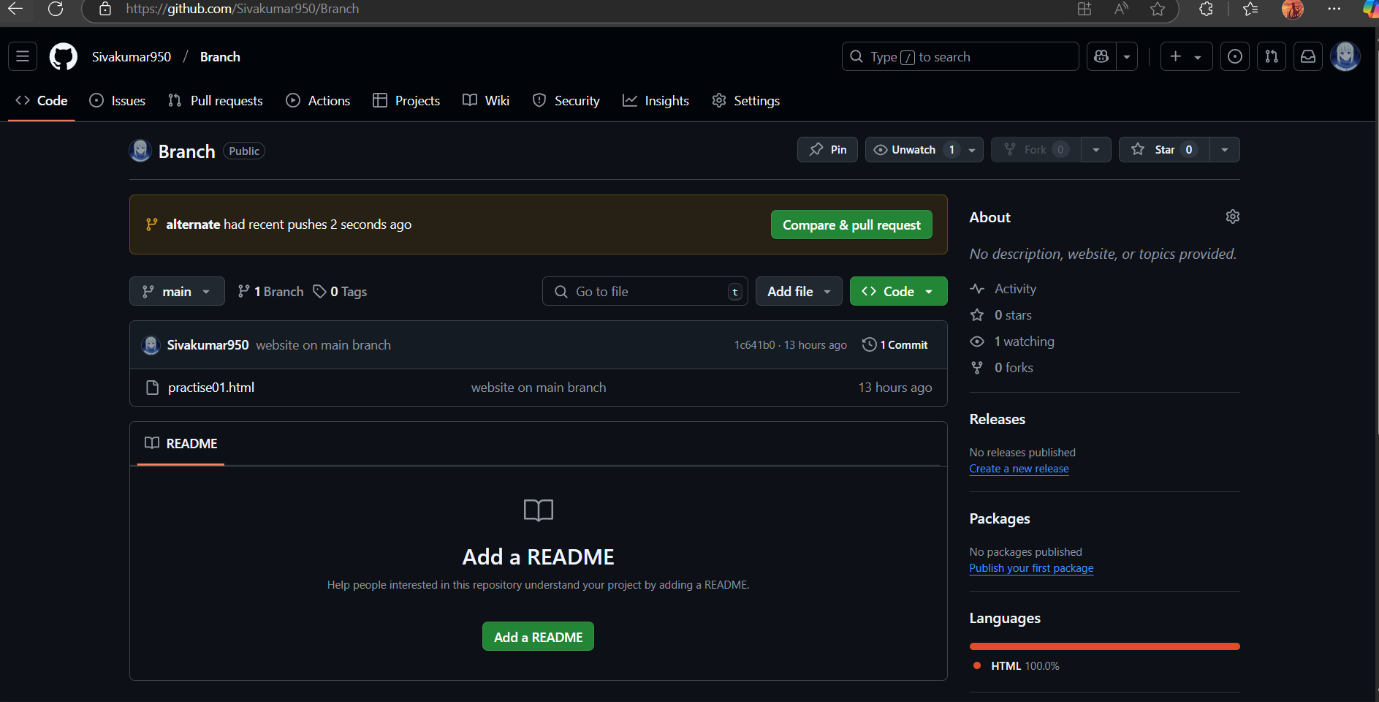


**STEP 4: Push the New Branch to GitHub**

* Push the new branch to your GitHub repository:

git push origin new-branch



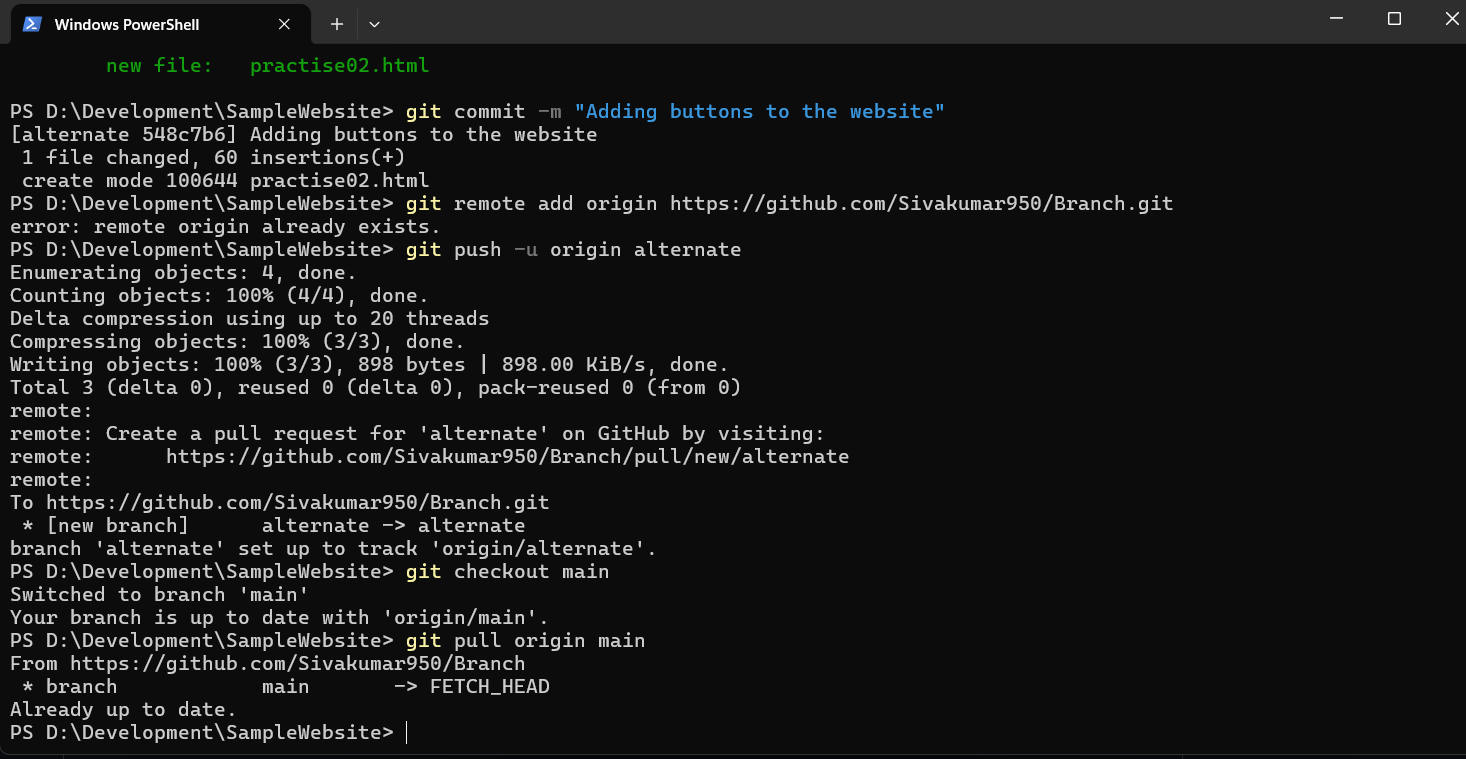


**STEP 5: Open a Pull Request (PR)**

* Go to your GitHub repository and open a Pull Request (PR) from new-feature to master (or the main branch).

git checkout main

git pull origin main



**STEP 6: Review and Merge the Pull Request**

* Once the PR is approved, merge the changes into the master (or main) branch.

git merge branch\_name

