

Placement Empowerment Program

Cloud Computing and DevOps Centre

Set Up a Local Git Repository: Initialize a Git repository locally and version control your static website

Name: Shalini D

Department: IT

Introduction

Setting up a local Git repository is an essential step for version controlling your static website. By initializing a Git repository on your local machine, you can track changes, maintain version history, and collaborate more effectively. This process allows you to manage your website's source code efficiently, ensuring that you can revert to previous versions, resolve conflicts, and deploy updates smoothly. Whether you're working solo or as part of a team, using Git for version control streamlines your development workflow and enhances project organization.

Overview

Setting up a local Git repository for your static website involves a few key steps. First, you'll initialize a Git repository on your local machine, allowing you to track changes and maintain a history of your project's development. This is done using the `git init` command.

Once your repository is initialized, you'll add your website files to the repository using `git add`. This stages your files for the first commit, which you will create with the `git commit` command.

By version controlling your static website, you can:

Track Changes: Easily see what changes were made and who made them.

Collaborate: Work with others more efficiently by sharing changes through a central repository.

Revert to Previous Versions: If something goes wrong, you can roll back to a previous state.

Maintain History: Keep a record of your project's progress over time.

This overview highlights the importance of version control in managing your static website's development and ensuring a smooth workflow.

Objectives

Objectives of Setting Up a Local Git Repository for Your Static Website:

Version Control: Track changes and maintain a history of your website's development.

Collaboration: Facilitate efficient teamwork by sharing changes and working on different parts of the project simultaneously.

Change Management: Easily manage updates, enhancements, and bug fixes.

Backup: Ensure that your website's source code is safely stored and can be recovered if needed.

Deployment: Streamline the deployment process by integrating with deployment tools and services.

Revertibility: Quickly roll back to previous versions if something goes wrong during development.

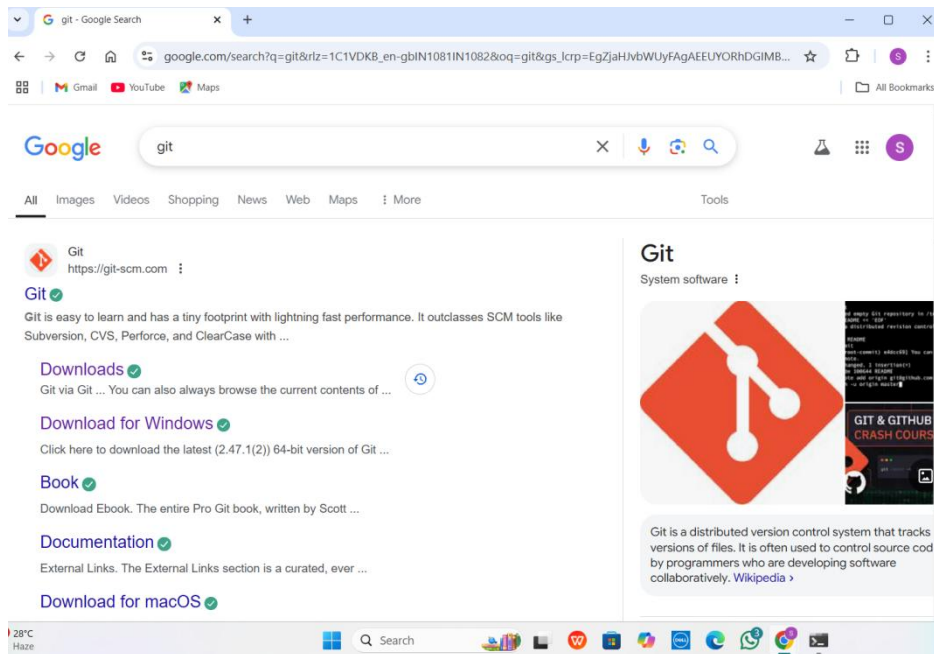
Documentation: Keep detailed records of changes, including who made them and why.

These objectives highlight the benefits of using Git for managing your static website's source code and development workflow.

Step-by-Step Overview

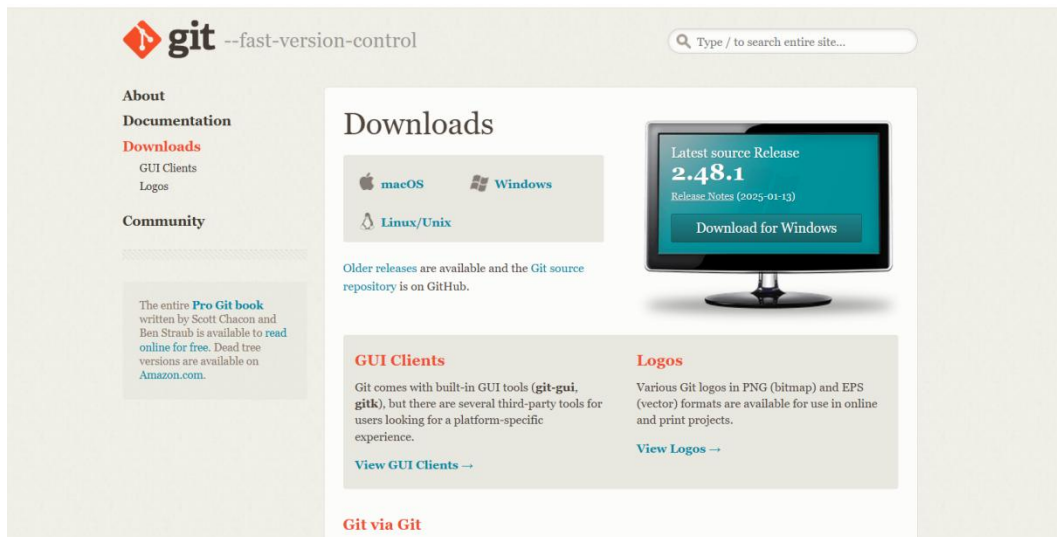
Step 1:

Search for "Git" in Chrome, download it.



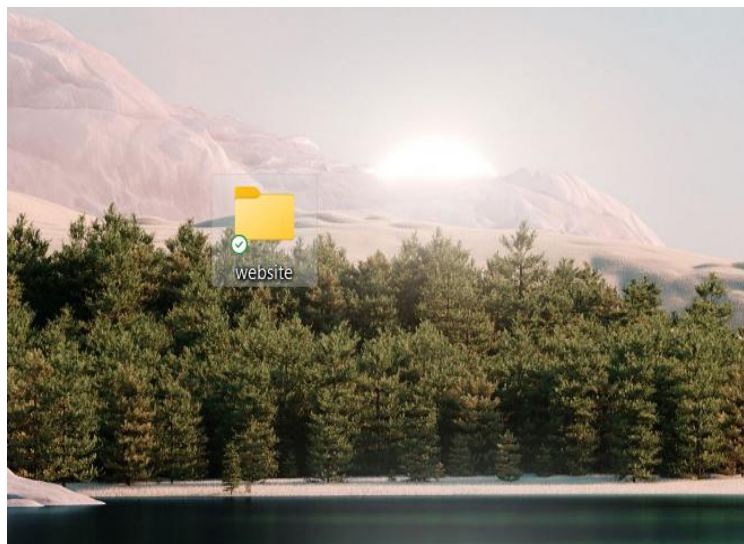
Step 2

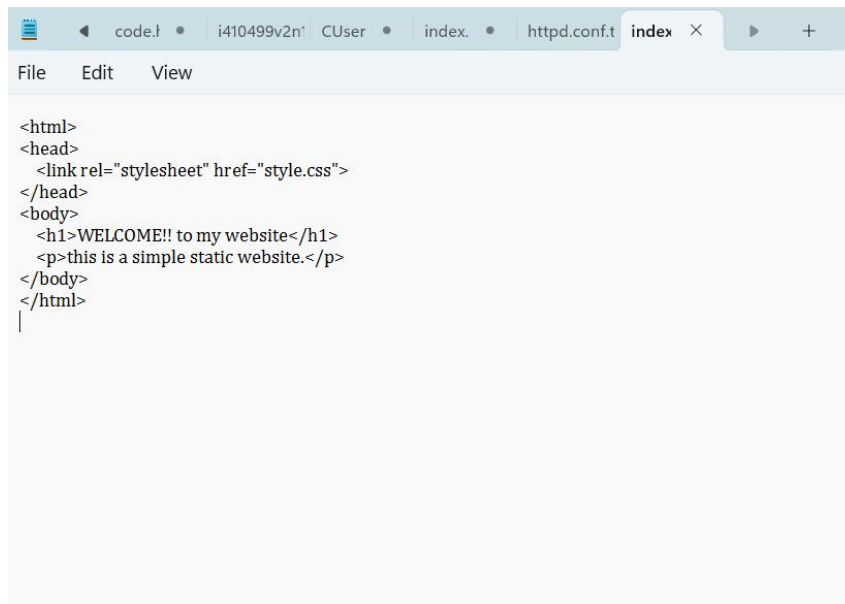
Click the **Windows** option on the download page and follow the installation wizard.



Step 3

- On your Desktop, create a folder named `website` for your static website.
- Inside that folder, create a simple HTML file named `index.html` and write some basic HTML.



A screenshot of a web browser window. The address bar shows the URL 'http://localhost:410499v2n1/CUser/index'. The browser has tabs for 'code.l', 'i410499v2n1', 'CUser', 'index', 'httpd.conf.t', and 'index'. The page content is a simple HTML document with a white background and black text. It includes a head section with a link to 'style.css' and a body section with a welcome message and a paragraph.

```
<html>
<head>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <h1>WELCOME!! to my website</h1>
  <p>this is a simple static website.</p>
</body>
</html>
```

Step 5

Open the Command prompt and set the path to the folder named website we created

```
C:\Users\shalni>cd C:\Users\shalni\OneDrive\Desktop\website
C:\Users\shalni\OneDrive\Desktop\website>cd website
```

Step 6

Now, initialize Git by typing this command:

git init

This command will create a .git folder inside your project folder, which tells Git to start tracking your files.

```
C:\Users\shalni\OneDrive\Desktop\website>git init
Initialized empty Git repository in C:/Users/shalni/OneDrive/Desktop/website/.git/
```

Step 7

Next, we need to tell Git to start tracking your website files.

To tell Git which files to track, use the `git add` command. If you want to track all the files in your folder, type

`git add .`

This command adds all the files to Git's tracking system.

```
C:\Users\shalni\OneDrive\Desktop\website>git add
```

Step 8

Set Up Your Name and Email Globally Git doesn't know who is making the commit because you haven't configured your name and email yet. Git uses this information to track who made the changes.

```
C:\Users\shalni\OneDrive\Desktop\website>git config --global user.name "shalini"  
C:\Users\shalni\OneDrive\Desktop\website>git config --global user.email "shaludk13@gmail.com"
```

Step 9

Now, we need to save these changes in Git. When you "commit" changes, Git takes a snapshot of your files.

Type the following command to commit your changes:

`git commit -m "Initial commit of my static website"`

The `-m` flag allows you to add a message about your changes. In this case, we're saying this is the "initial commit," meaning the first time we're saving our work.

```
C:\Users\shalni\OneDrive\Desktop\website>git commit -m "Initial commit of my static website"
[master (root-commit) b12871a] Initial commit of my static website
1 file changed, 9 insertions(+)
create mode 100644 index.html
```

Step 10

Create a New Repository:

Once you're logged in, click the green **"New"** button on the top right of your GitHub homepage to create a new repository.

Give your repository a name, for example, my-website.

Leave the other settings as default, and click **"Create repository"**.

Step 11

Add the Remote Repository URL to Your Local Repository:

Go back to your Command Line and type the following:

```
git remote add origin https://github.com/yourusername/my-website.git
```

```
C:\Users\shalni\OneDrive\Desktop\website>git remote add origin https://github.com/Shaliniaa/my-website.git
```

Step 12

The **git branch -M** main command is used to **rename the current branch** to main. Here's what it does:

-M: This flag forces the renaming, even if a branch named main already exists. It will overwrite the existing main branch.

main: This is the new name for the current branch.

```
C:\Users\shalni\OneDrive\Desktop\website>git branch -M main
```

Step 13

The command **git push -u origin main** is used to push your local **main** branch to the remote repository (**origin**) and set it as the upstream branch

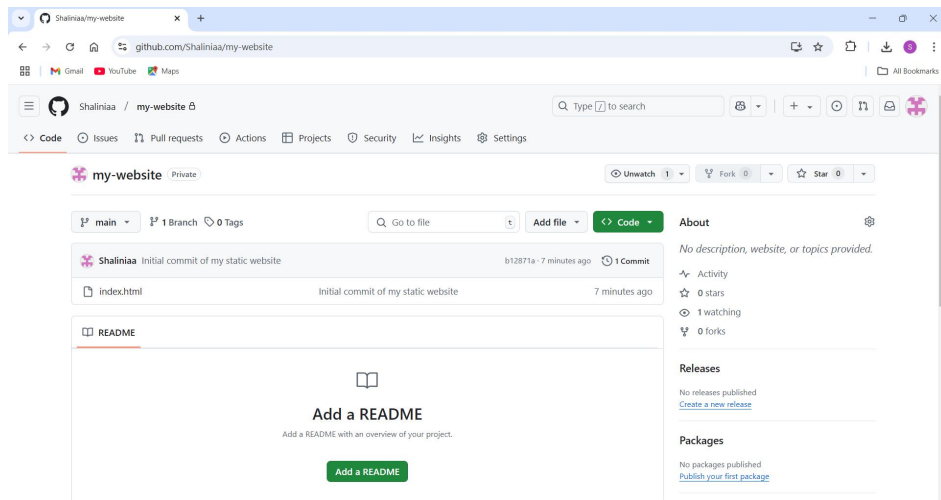
```
C:\Users\shalni\OneDrive\Desktop\website>git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 6 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 350 bytes | 175.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Shaliniaa/my-website.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

Step 14

Verify Your Files on GitHub:

Open your web browser and go to your GitHub repository .You should see your website files there.

This confirms that your files have been successfully uploaded to your GitHub repository.



Expected Outcome

By following these steps, you will have a basic structure for your static website stored in a folder named website on your Desktop. The folder will contain an index.html file with basic HTML content. This setup prepares you to initialize a Git repository and start version-controlling your static website, ensuring efficient management and tracking of changes.