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: Mary Edlyn A

Department: CSE

Introduction:

Version control is a fundamental practice in software development that allows you to manage changes to your code over time. It provides a systematic way to track updates, collaborate with others, and revert to previous versions if needed. Git is one of the most widely used version control systems, known for its efficiency, flexibility, and distributed nature.

In this POC, we'll initialize a local Git repository to version control your static website. By doing so, you'll be able to track changes to your project files, experiment with new features in a controlled way, and easily share your project with others if needed. Setting up a Git repository is a critical step towards maintaining a structured and reliable workflow, especially for developers and teams working on collaborative projects.

Overview:

Here's what we will cover in this setup:

1. Installing Git: Ensure Git is installed on your system and properly configured.

2. Creating a Local Repository: Initialize a Git repository in the root folder of your static website

3. Staging and Committing Files: Add your project files to the staging area and commit them to the repository to save a snapshot of your work.

4. Reviewing the evolves. **Repository State**: Use Git commands to check the status of your repository and verify that everything is tracked properly.

Objectives:

By the end of this POC, you will:

1. Understand the Basics of Version Control: Gain insight into the importance of Git for managing and tracking changes in your projects.

2. Set Up a Git Repository: Learn how to initialize a Git repository to version control your static website locally.

3. Track Changes Effectively: Understand how to stage and commit files to ensure every change is logged.

4. **Organize Your Project**: Maintain a clean and structured workflow for your static website, with the ability to roll back changes when needed.

5. **Prepare for Collaboration**: Lay the groundwork to share your repository and collaborate with others using Git when required

Importance of Setting Up a Local Git Repository :

Track Changes: Git records all modifications, ensuring a clear history of your project.

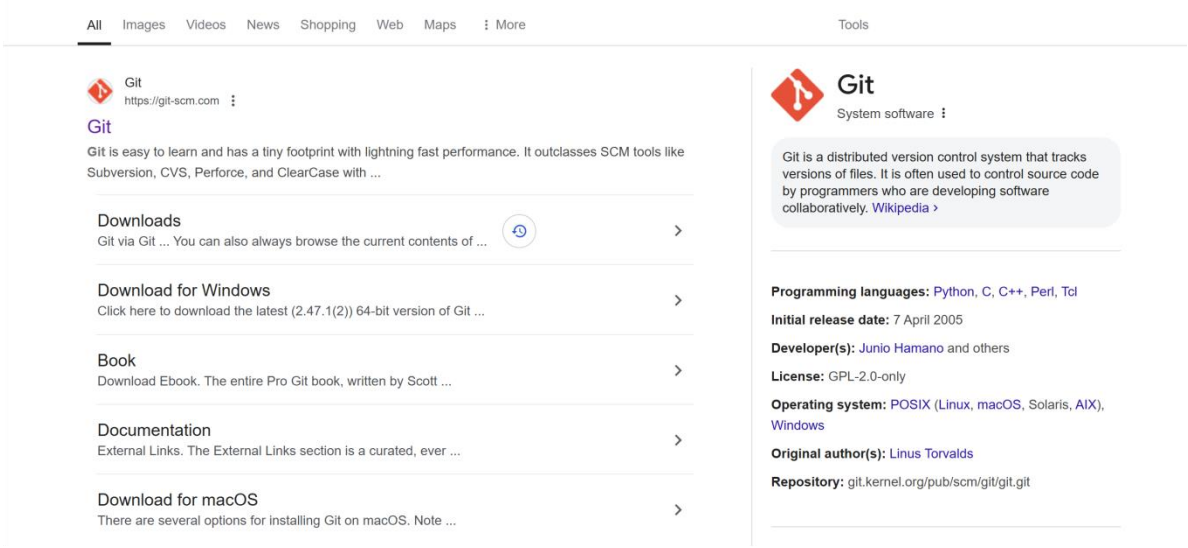
Rollback: Easily revert to previous versions to recover from mistakes.

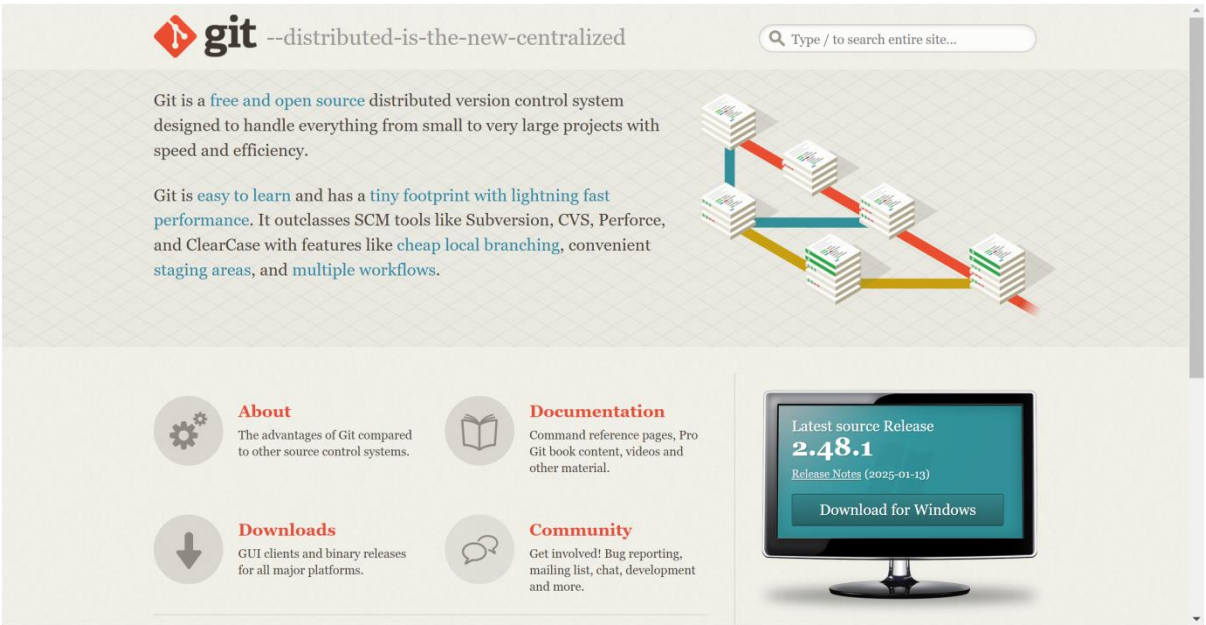
Collaboration: Prepares your project for team work, enabling smooth integration of changes.

Step-by-Step Overview

Step 1:

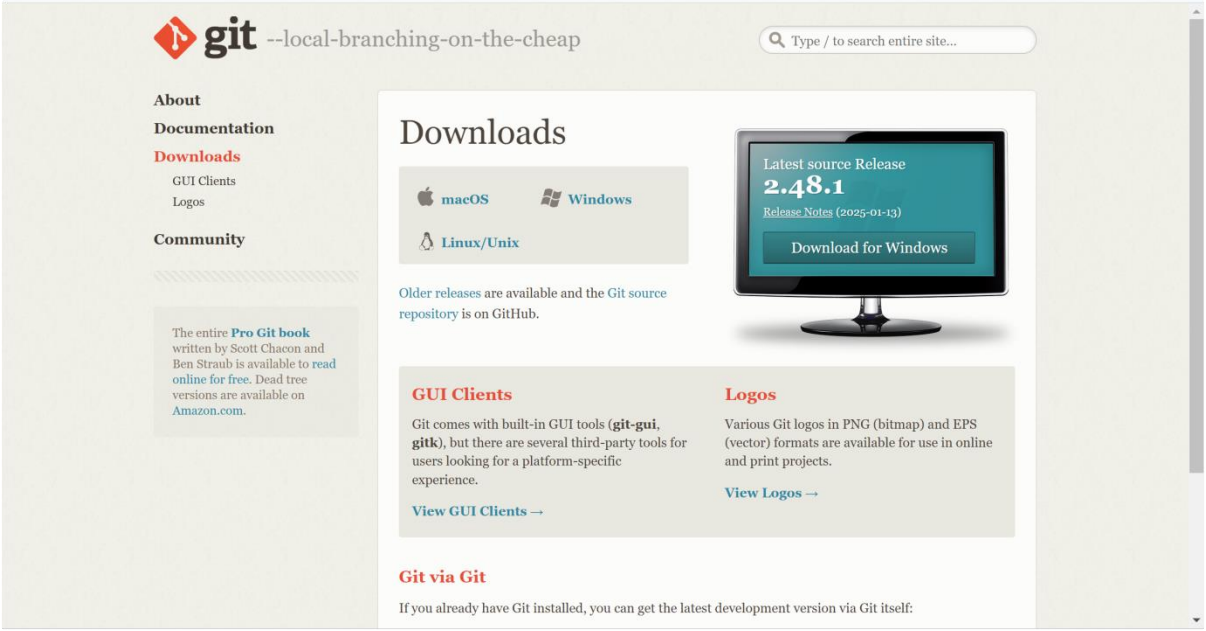
Search for "Git" in Chrome, download it, and click the "Downloads" option on the website.





Step 2:

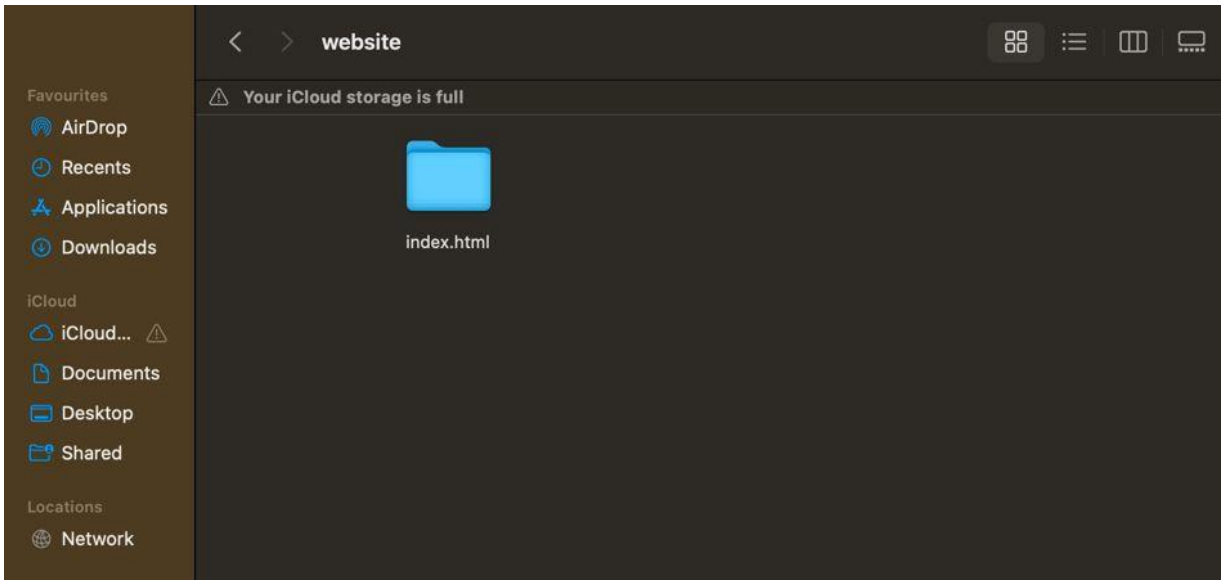
Click the **Windows** option on the download page and follow the installation wizard.

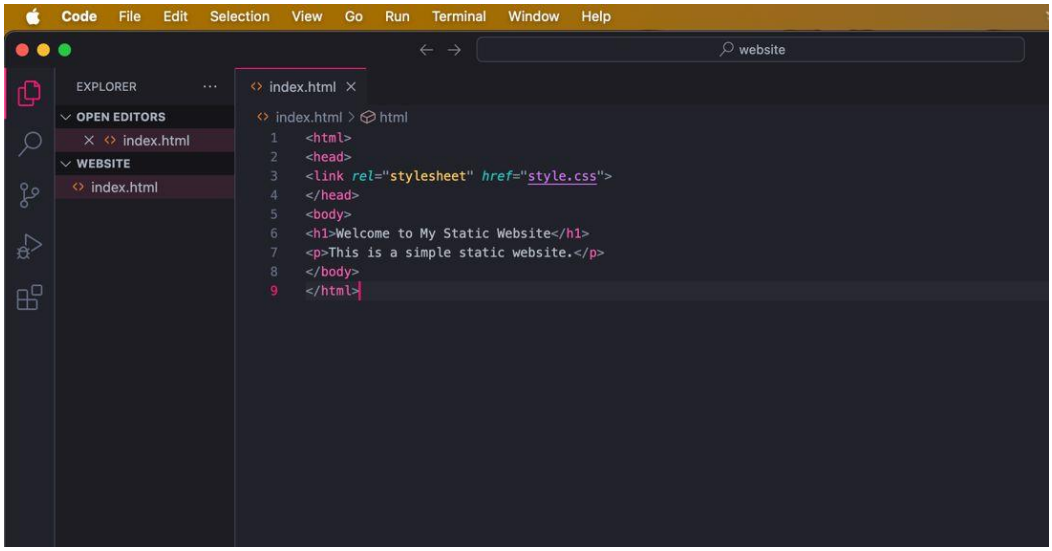


Step 3:

In your Desktop Create a folder named website for your static website

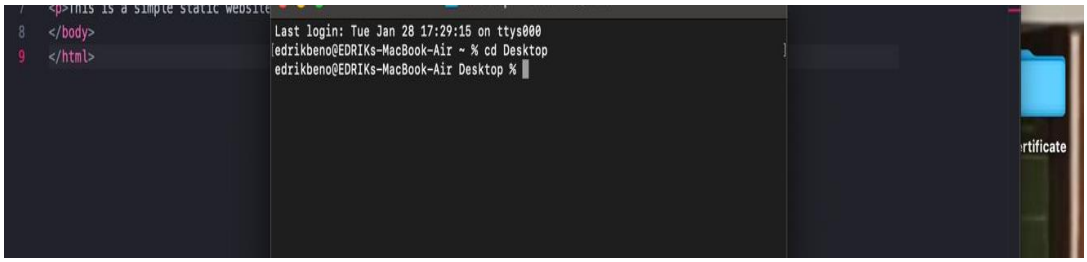
Inside that folder, create a simple HTML file named index.html. You can write some basic HTML





Step 4 :

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

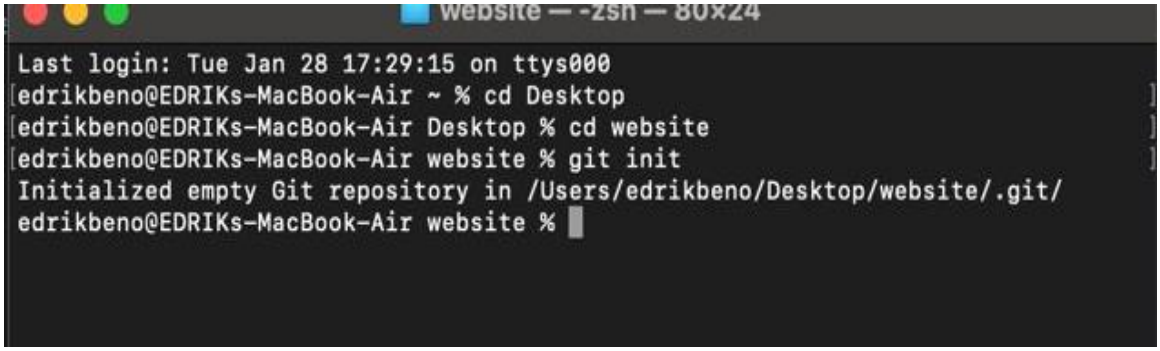


Step 5:

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.



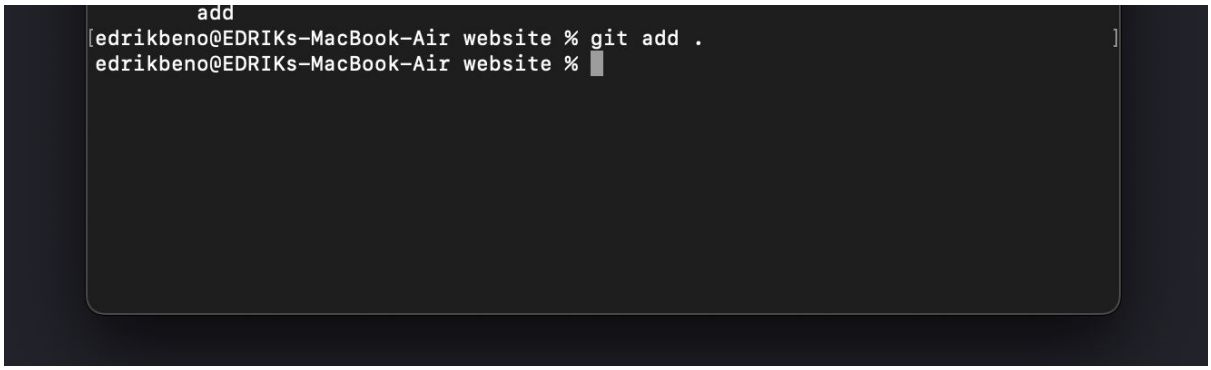
Step 6:

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.



Step 7:

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.

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.

```
[edrikbeno@EDRIKs-MacBook-Air website % git add.
git: 'add.' is not a git command. See 'git --help'.

The most similar command is
[edrikbeno@EDRIKs-MacBook-Air website % git config --global user.email "maryedlyn007@gmail.com"
[edrikbeno@EDRIKs-MacBook-Air website % git config --global user.name "Mary Edlyn A"
[edrikbeno@EDRIKs-MacBook-Air website % git config --global user.email "maryedlyn007@gmail.com"
[edrikbeno@EDRIKs-MacBook-Air website % git commit -m "Initial commit of my static website"
```

Step 8:

Create a New Repository & **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

Replace yourusername with your GitHub username and my-website with the name of your GitHub repository.

```
[edrikbeno@EDRIKs-MacBook-Air website % git remote add origin https://github.com/MaryEdlynA19/my-website.git
[edrikbeno@EDRIKs-MacBook-Air website % git branch -M main
[edrikbeno@EDRIKs-MacBook-Air website % git push -u origin main
```

Step 9:

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

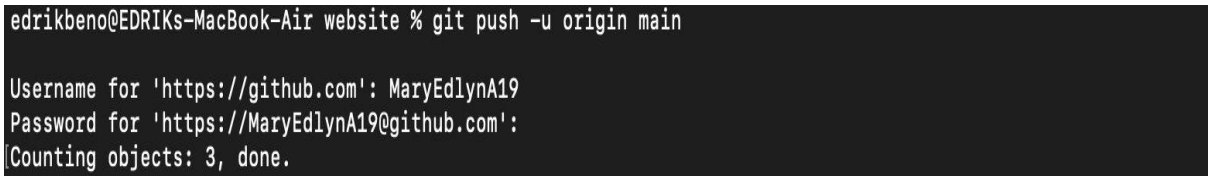
-M: This flag forces the renaming, even if a branch named main already exists. It will overwrite the existing main branch.

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

```
edrikbeno@EDRIKs-MacBook-Air website % git branch -M main
```

Step 10:

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



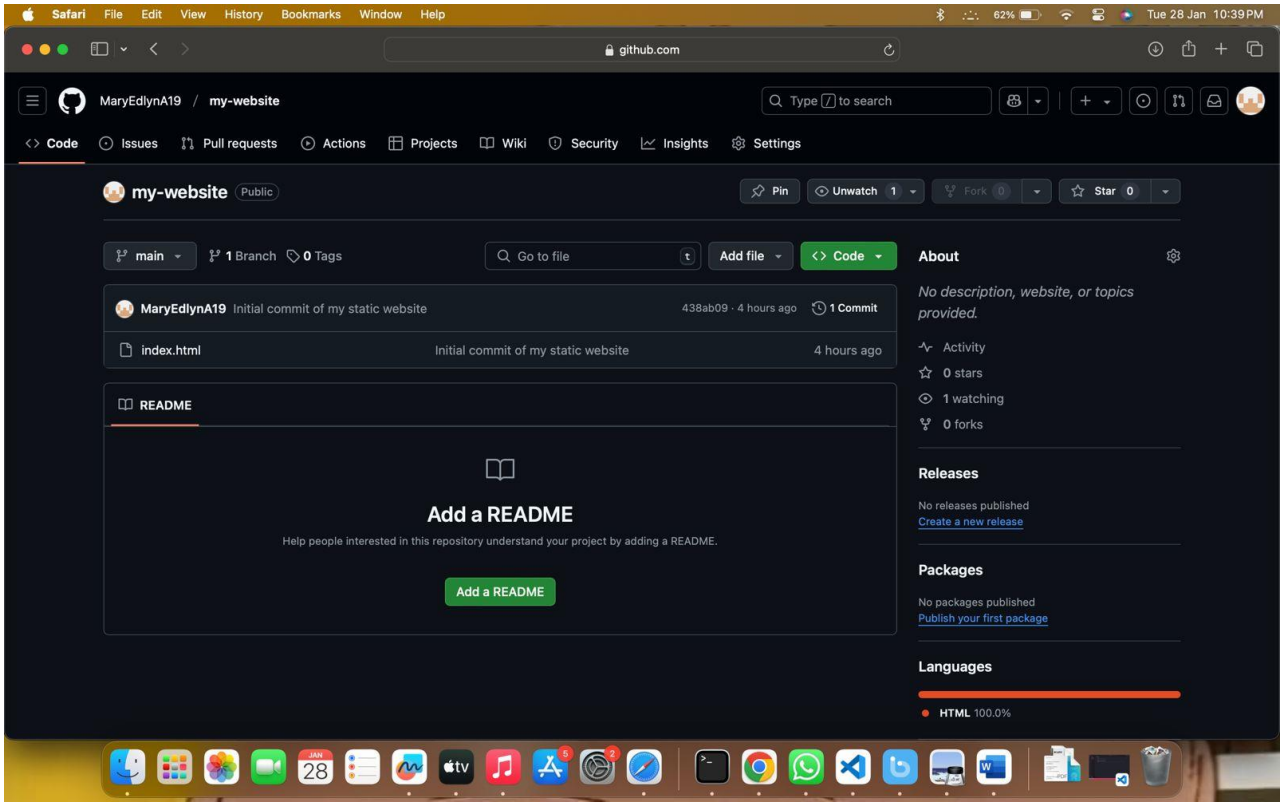
Step 11:

Verify Your Files on GitHub

Go to your GitHub Repository:

Open your web browser and navigate to your GitHub repository (e.g., https://github.com/yourusername/my-website).

You should see your website files there!



Outcome:

By completing this PoC of setting up a local Git repository, you will:

1. Successfully initialize a Git repository in your local static website folder.

2. Track changes made to your website files (HTML, CSS, etc.) using Git version control.

3. Understand the basic Git commands (git init, git add, git commit) for version control.

4. Commit your changes locally with a descriptive commit message.

5. Gain hands-on experience with Git and how it helps manage and track website file changes.