 

**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: MIRDULA C Department: ADS



**Introduction**

Version control is an essential practice in software development, enabling efficient management of code changes over time. It provides a structured approach to tracking modifications, facilitating collaboration, and reverting to previous versions when necessary. Among the various version control systems, Git stands out for its efficiency, flexibility, and distributed architecture.

In this proof of concept (POC), we will initialize a local Git repository to implement version control for your static website. This setup will allow you to monitor changes in project files, experiment with new features in a controlled environment, and seamlessly share your project when needed. Establishing a Git repository is a crucial step toward maintaining an organized and reliable workflow, particularly for developers and teams engaged in collaborative projects.

**Overview**

**Overview of the Setup Process**

In this setup, we will cover the following key steps:

Installing Git – Verify that Git is installed on your system and properly configured.

Creating a Local Repository – Initialize a Git repository in the root directory of your static website.

Staging and Committing Files – Add project files to the staging area and commit them to the repository, creating a snapshot of your work.

Reviewing the Repository State – Use Git commands to check the status of your repository and ensure that all changes are tracked correctly.

**Objectives**

**Objectives of This Proof of Concept (POC)**

By the end of this POC, you will:

1. Understand the Fundamentals of Version Control – Gain a clear understanding of Git’s role in managing and tracking changes in software projects.
2. Set Up a Git Repository – Learn how to initialize a local Git repository to implement version control for your static website.
3. Track Changes Effectively – Master the process of staging and committing files to ensure every modification is properly recorded.
4. Maintain a Structured Workflow – Organize your project efficiently, enabling seamless version management and the ability to roll back changes when necessary.
5. Prepare for Collaboration – Establish a strong foundation for sharing your repository and collaborating with others using Git.

**Importance of Setting Up a Local Git Repository**

**Key Benefits of Using Git**

* Track Changes – Git maintains a comprehensive history of all modifications, ensuring transparency and version control.
* Rollback – Easily revert to previous versions to recover from errors or unintended changes.
* Seamless Collaboration – Facilitate teamwork by enabling smooth integration of updates and modifications from multiple contributors.

**Step-by-Step Overview**

* Setting Up Git and Pushing Your Static Website to GitHub
* Follow these steps to install Git, initialize a repository, and push your static website to GitHub.
* Step 1: Download and Install Git
* Open Chrome and search for "Git".
* Visit the official Git website and click the "Downloads" option.
* Select the Windows option and follow the installation wizard.
* Step 2: Create a Project Folder and HTML File
* On your desktop, create a new folder named website for your static website.
* Inside the website folder, create a simple HTML file named index.html with basic HTML content.
* Step 3: Open Command Prompt and Navigate to Your Folder
* Open the Command Prompt.
* Set the path to the website folder by using the cd command:
* sh
* CopyEdit
* cd path/to/website
* Step 4: Initialize a Git Repository
* Run the following command to initialize Git in your project folder:
* sh
* CopyEdit
* git init
* This creates a .git folder inside your project directory, allowing Git to start tracking changes.
* Step 5: Track Your Website Files
* To tell Git which files to track, add all files in the project folder:
* sh
* CopyEdit
* git add .
* This command stages all files, preparing them for the first commit.
* Step 6: Configure Git with Your Name and Email
* Before making commits, configure Git to recognize your identity:
* sh
* CopyEdit
* git config --global user.name "Your Name"
* git config --global user.email "your.email@example.com"
* This ensures that your commits are associated with your GitHub account.
* Step 7: Commit Your Changes
* Save the staged changes by committing them:
* sh
* CopyEdit
* git commit -m "Initial commit of my static website"
* The -m flag allows you to add a commit message describing your changes.
* Step 8: Create a New Repository on GitHub
* Log in to your GitHub account.
* Click the green "New" button on the top-right of your GitHub homepage.
* Name your repository, e.g., my-website.
* Leave the other settings as default and click "Create repository".
* Step 9: Link Your Local Repository to GitHub
* In the Command Prompt, add the GitHub repository as a remote origin:
* sh
* CopyEdit
* git remote add origin https://github.com/yourusername/my-website.git
* Replace yourusername with your GitHub username and my-website with your repository name.
* Step 10: Rename the Default Branch to Main
* Run the following command to rename the current branch to main:
* sh
* CopyEdit
* git branch -M main
* The -M flag forces the renaming, even if a branch named main already exists.
* Step 11: Push Your Code to GitHub
* Upload your local repository to GitHub using:
* sh
* CopyEdit
* git push -u origin main
* This pushes your code to the remote repository and sets main as the upstream branch.
* Step 12: Verify Your Files on GitHub
* Open your web browser and navigate to your GitHub repository:
* perl
* CopyEdit
* https://github.com/yourusername/my-website
* You should see your index.html file and other project files listed.

**Expected Outcome**

* By completing this PoC on setting up a local Git repository, you will:
* Initialize a Git Repository – Successfully set up Git in your local static website folder.
* Track File Changes – Monitor modifications to your website files (HTML, CSS, etc.) using Git version control.
* Master Essential Git Commands – Learn and apply fundamental Git commands such as git init, git add, and git commit for effective version control.
* Commit Changes with Clarity – Save project updates locally using meaningful commit messages.
* Gain Practical Git Experience – Develop hands-on knowledge of Git’s role in managing and tracking website file changes efficiently.