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: Mary Edlyn A

Department: CSE

Introduction

Git is a version control system that allows developers to track changes, collaborate efficiently, and manage code versions. One of Git's key features is branching, which enables developers to create separate branches for different tasks without affecting the main codebase. In this task, we will create a new branch, add a new feature, and merge it back into the main branch.

Objectives

The main objectives of this task are:

- To understand and implement Git branching.
- To create a separate branch for testing new features.
- To modify code and commit changes efficiently.
- To merge the new feature back into the main branch without conflicts.
- To enhance collaboration by managing multiple versions of a project.

Importance of Setting Up a Local Git Repository

1. **Isolates New Features**: Branching allows developers to work on new features or fixes without disturbing the stable code.

2. **Facilitates Team Collaboration**: Multiple developers can work on different branches, making teamwork smoother.

3. **Prevents Code Conflicts**: Merging allows for controlled integration of changes, minimizing errors.

4. **Enhances Project Organization**: Keeping features separate before merging ensures a clean and well-structured codebase.

5. **Supports Continuous Development**: Developers can work on multiple features in parallel without affecting the main branch.

Step-by-Step Overview

Step 1: **Navigate to Your Repository**

Open a terminal/command prompt and navigate to your Git repository.

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

C:\Users\nandh\OneDrive\Desktop\website>git init
Reinitialized existing Git repository in C:/Users/nandh/OneDrive/Desktop/website/.git/

C:\Users\nandh\OneDrive\Desktop\website>
```

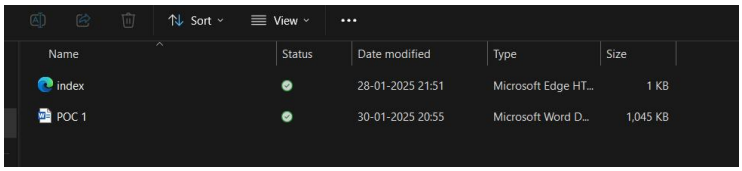
Step 2: Create a New Branch

Create a new branch for testing. This command creates and switches to a new branch named feature-branch.

```
C:\Users\nandh\OneDrive\Desktop\website>git checkout -b feature-branch
Switched to a new branch 'feature-branch'
```

Step 3: Make Changes and Add a New Feature

In your Desktop folder named website which you already created for your static website add new files and stage them.




```
C:\Users\nandh\OneDrive\Desktop\website>git add .
```

```
C:\Users\nandh\OneDrive\Desktop\website>git commit -m "Added a new feature"
[feature-branch 58141fb] Added a new feature
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 POC 1.docx
```

Step 4

Switch back to main branch and merge the new branch back to main branch.

```
C:\Users\nandh\OneDrive\Desktop\website>git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

C:\Users\nandh\OneDrive\Desktop\website>git merge feature-branch
Updating d93965b..58141fb
Fast-forward
 POC 1.docx | Bin 0 -> 1069453 bytes
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 POC 1.docx
```

Step 5

Push it to the Git Repository using " **git push origin main** " command.

```
C:\Users\nandh\OneDrive\Desktop\website>git push origin main
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 989.13 KiB | 63.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/vijayanandana25/my-website.git
   d93965b..58141fb  main -> main
```

Step 6

Once merged, you can delete the feature branch.

```
C:\Users\nandh\OneDrive\Desktop\website>git branch -d feature-branch
Deleted branch feature-branch (was 58141fb).
```

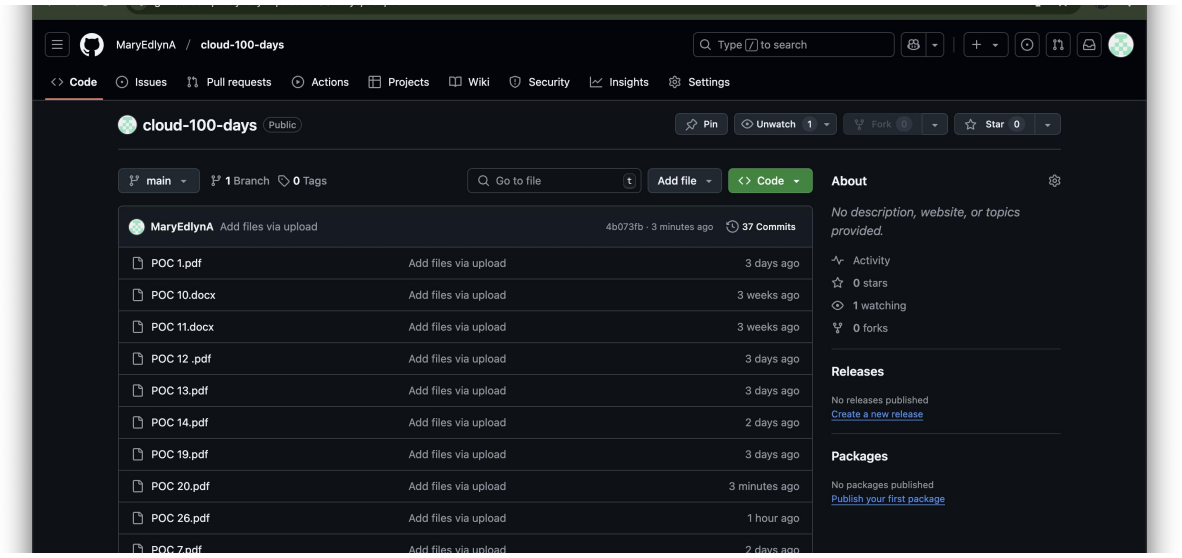
Step 7

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 new files there!



Outcome

By completing this PoC, you will:

1. **Successful Creation of a New Branch**
   - You will have a new branch (feature-branch) in your Git repository for development and testing.
2. **Addition of a New Feature**
   - You will modify or add code to implement a new feature within the newly created branch.
3. **Commit and Version Control Understanding**
   - You will gain hands-on experience with Git commands like git add, git commit, and git checkout, improving your version control skills.
4. **Smooth Integration via Merging**
   - You will successfully merge the new feature branch into the main branch without conflicts, ensuring seamless integration of changes.
5. **Improved Collaboration Workflow**

- If working in a team, you will understand how multiple developers can work on different branches and merge their work effectively.

6. **A Cleaner, More Organized Codebase**
    - Your repository will remain structured, with isolated feature development, reducing bugs and errors in the main branch.