**SCM Sample Assignment - Task 1.1**

**Title: Implementing Git Basics**

**Problem Statement**

As a part of a small but dynamic development team, you are working on a project that requires efficient version control. To achieve this, you will be using Git and GitHub to manage code changes, collaborate with teammates, and maintain a structured workflow.

Your task is to perform and demonstrate key Git operations such as repository setup, branching, merging, and conflict resolution. This exercise will solidify your understanding of how teams work with Git in real-world projects.

**1. Setting Up the Repository and Making the Initial Commit**

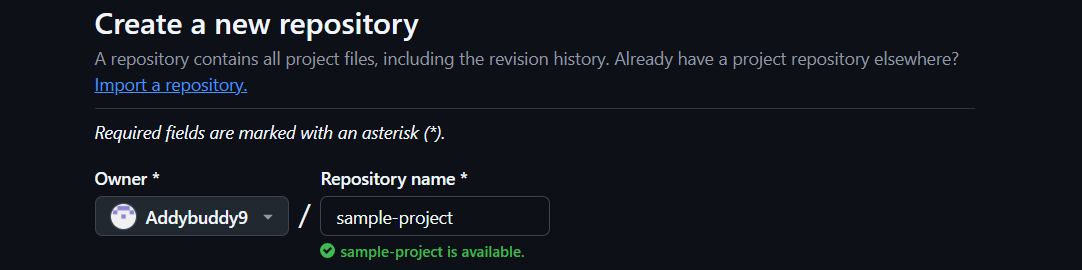
**Step 1: Creating a GitHub Repository**

A GitHub repository is essential for version control and collaboration. Follow these steps to create one:

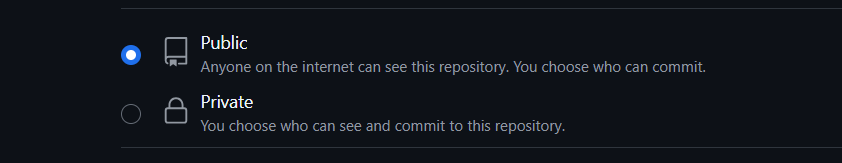
1. Log in to [GitHub](https://github.com/) and navigate to the repositories section.
2. Click the "New Repository" button.



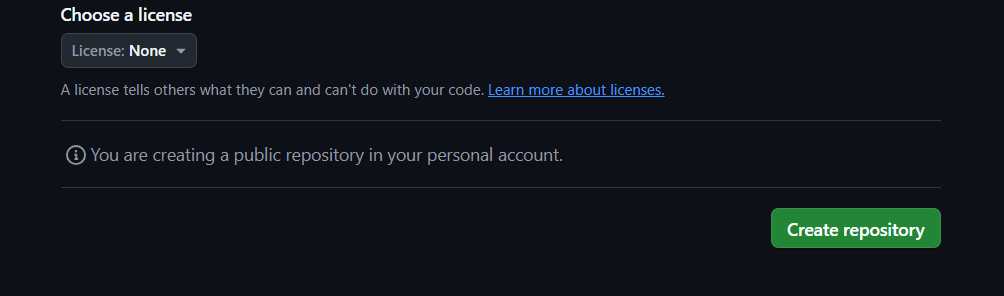
1. Provide a name for your repository (e.g., sample-project).



1. Choose visibility: **Public** (visible to everyone) or **Private** (visible only to you and collaborators).

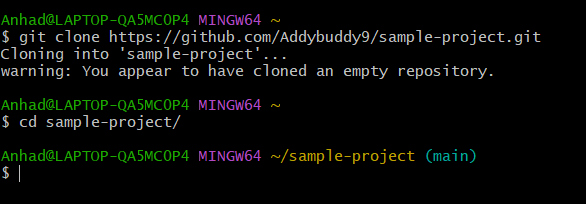


1. Click **Create Repository**.

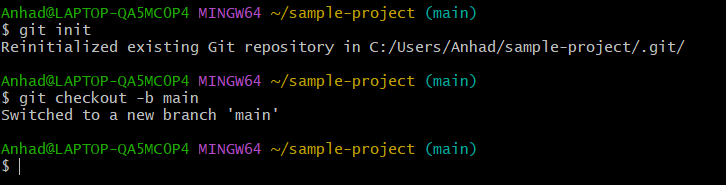


**Step 2: Cloning the Repository Locally**

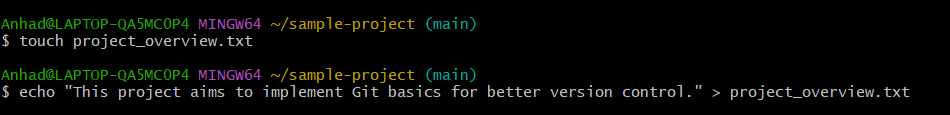
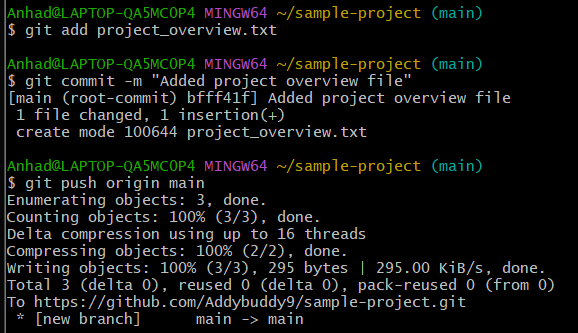
Once the repository is created, clone it to your local system:

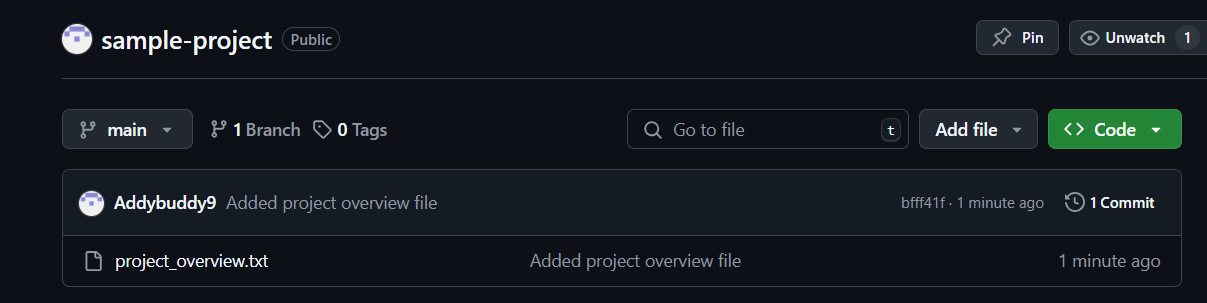
**Step 3: Initializing Git and Creating the Main Branch**

To ensure proper tracking of changes, initialize Git in your local repository and switch to the main branch:



**Step 4: Creating and Committing the Project Overview File**

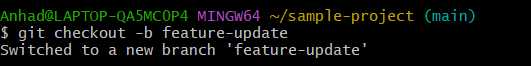
Create a file named project\_overview.txt. and add a brief description to the file2. 2.2. Stage and commit the file:   




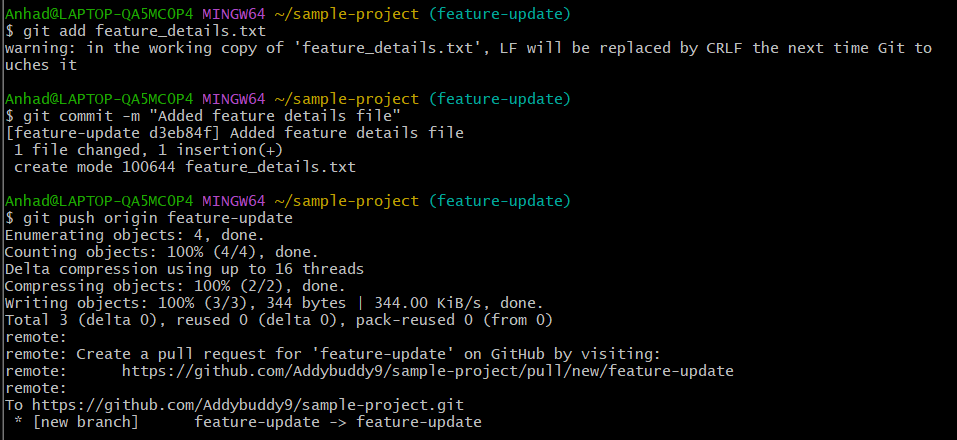
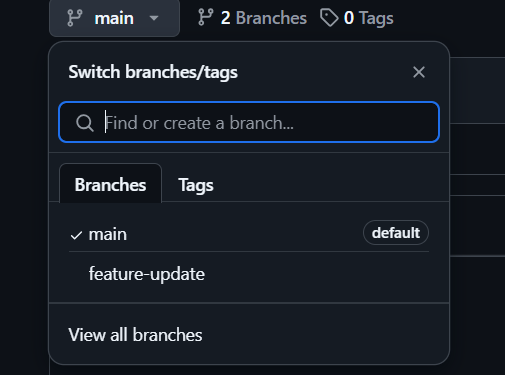
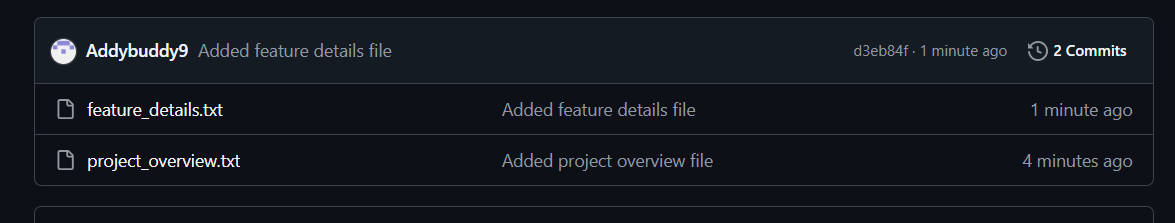
**2. Branch Creation and Feature Updates**

**Step 1: Creating a New Branch**

A separate branch allows development without affecting the main branch. Create a new branch named feature-update:

**Step 2:**

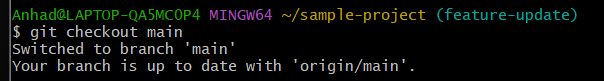
**Adding a Feature Details File**

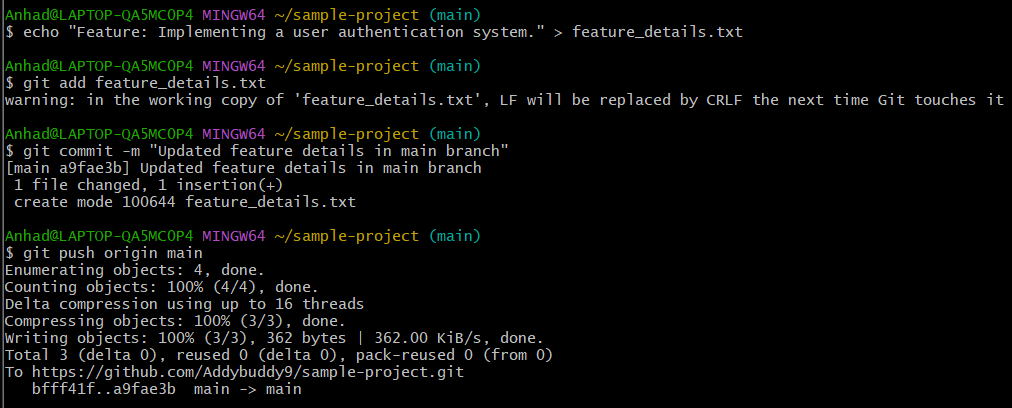
1. Create a file called feature\_details.txt and write details about the feature you plan to implement:   
   
2. Stage and commit the changes:  
   
3.    
     
   

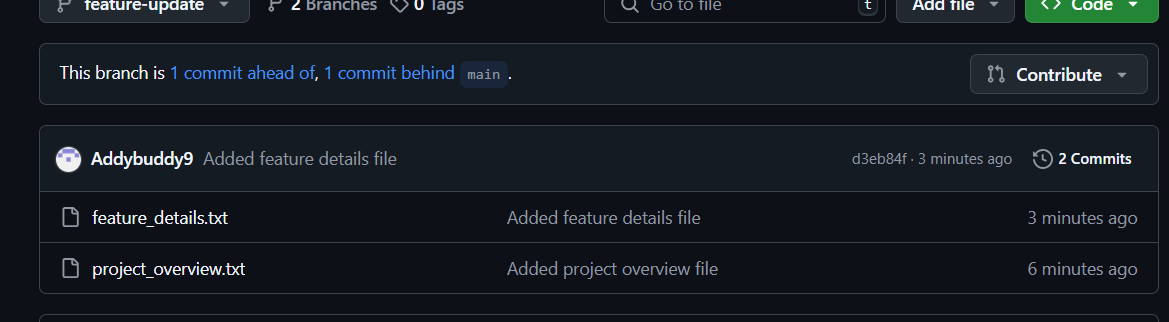
**3. Merging and Handling Merge Conflicts**

**Step 1: Modify feature\_details.txt in the Main Branch**

To simulate a conflict, make changes to the feature\_details.txt file in the main branch:

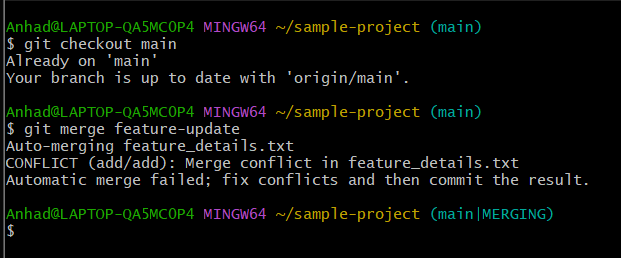




****

**Step 2: Attempting to Merge feature-update into main**

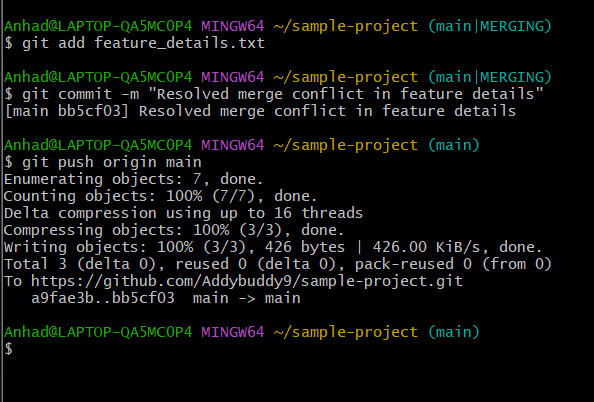
Switch back to the main branch and attempt to merge:

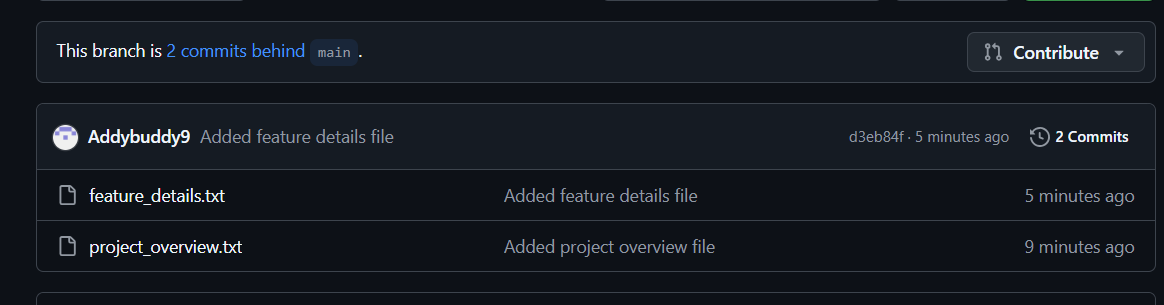


If there is a conflict, Git will indicate which files have conflicts. Open feature\_details.txt and manually resolve the conflict by keeping the most appropriate changes.

**Step 3: Committing the Resolved File**

Once resolved, commit the changes:

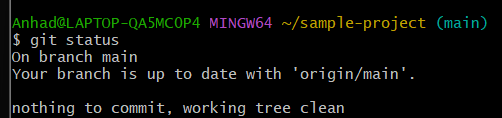




**4. Auditing the Repository**

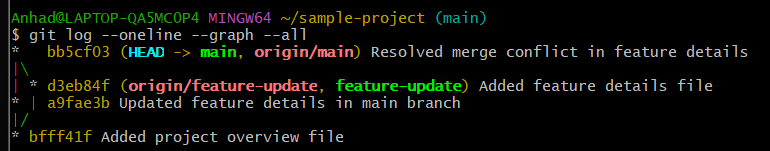
**Step 1: Checking the Repository Status**

To get an overview of the current state of the repository, use:

 **Step 2: Viewing**

**Commit History**

To see a log of all commits and branch activities:

 This will display a clear visual representation of the commit history, helping track changes over time.

**Link: https://github.com/Addybuddy9/sample-project**