LAB 1

Title: Getting Started with Git for Managing Code

Objective: To learn the basic skills needed to manage source code using Git

- Start a new Git project (repository)
- Save changes using git add and git commit
- Check the history of changes
- Create and join branches
- Upload code to websites like GitHub (remote repositories)

Theory: Source Code Management (SCM) is a way to keep track of changes made to your code. It helps many people work on the same code, see what was changed and when, go back to earlier versions if needed, and keep a full record of the project's history.

Git is a popular SCM tool that developers use around the world. It is fast, works offline, and is great for teams.

Important Git Terms:

- **Repository:** A folder where Git keeps track of your code and its changes.
- **Commit:** A saved version of your code at a certain time, with a message about what changed.
- **Branch:** A separate line of work in your project. You can try new features without changing the main version.
- **Merge:** Bringing changes from one branch into another — often used to combine your work with the main code.
- **Remote Repository:** An online version of your Git project, saved on platforms like GitHub, GitLab, or Bitbucket. It allows teamwork and keeps your code safe.

Steps:

1. Create a new project folder
   ```
   mkdir myproject
   cd myproject
   ```

2. Initialize a Git repository
   ```
   git init
   ```

3. Create a new file (example: index.html)

4. Check the status of the repository
   git status

5. Add files to the staging area
   git add index.html

6. Commit the changes
   git commit -m "Initial commit"

7. Create a remote repository (on GitHub, GitLab, etc.)
   Go to GitHub and click on "New Repository"
   Copy the repository URL

8. Link the local repo with the remote
   git remote add origin

9. Push the code to the remote repository
   git push -u origin main

Conclusion: Understanding the basics of Git is essential for any developer working on software projects. With Git, you can easily track changes, work on different features using branches, and safely manage your code with version control. It also makes teamwork easier by allowing developers to share and update code through remote repositories like GitHub. Learning these fundamental Git commands and concepts will help you work more efficiently and collaborate better in real-world development environments.