**FSD Laboratory 01**

**Kanishk Singhania**

**Panel - H (H1-27)**

**PRN: 1032211295**

Aim: Version control with Git.

Objectives:

1. To introduce the concepts and software behind version control, using the example of Git.
2. To understand the use of 'version control' in the context of a coding project.
3. To learn Git version control with Clone, commit to, and push, pull from a git repository.

Theory:

1. What is Git? What is Version Control?

A Version control (or source control) is a system that manages changes to files and directories over time, providing a historical record of modifications and enabling collaborative development while ensuring data integrity.

Version control systems help developers work collaboratively on software projects, manage different versions or branches of code, and easily revert to previous states of a project if issues arise.

Git is a powerful version control system that helps developers manage code changes, collaborate on projects, and maintain a complete history of their work. It has become an essential tool in modern software development and is widely used by individuals and teams to track and control the evolution of their projects.

2. How to use Git for version controlling?

Essential steps to use GIT are:-

* Install Git: Download and install Git on your computer if not already installed.
* Configure Git: Set your name and email address with “git config”.
* Initialize Repository: Use “git init” to create a new Git repository in your project directory.
* Add Files: Add project files to the repository with “git add”.
* Make Commits: Create snapshots of your code with “git commit -m "Your message"”.
* View History: Check commit history with “git log”.
* Create Branches: Work on features in branches with “git branch” and “git checkout”.
* Merge Branches: Combine changes from branches with “git merge”.
* Collaborate: Connect to remote repositories using git remote and collaborate with others.
* Pull Changes: Get the latest updates from a remote repository with git pull.

These steps cover the basic Git workflow for version control.

FAQ:

1. What is branching in Git?

Branching in Git allows developers to create separate lines of development within a repository. Each branch represents a unique path of changes, enabling isolation and parallel development of features, bug fixes, and experiments. Developers can work on their specific branches, and when changes are complete, they can merge them into the main branch. Branching is essential for organized and collaborative software development, facilitating feature development, bug fixing, and release management.

2. How to create and merge branches in Git? Write the commands used.

To create and merge branches in git, we can use the following commands:

1) Creating a Branch:

* To create a new branch: “git branch branchname”.
* To create and switch to a new branch in one step: “ git checkout -b branchname”.

2) Merging a Branch:

* First, ensure you are on the branch where you want to merge the changes (e.g., the main branch): “git checkout main”
* Then, merge the changes from the other branch (e.g., feature-branch): git merge feature-branch

Output: Screenshots of the output to be attached.

**Problem Statement:**

Create a public git repository for your team and submit the repo URL as a solution to this assignment, Learn Git concept of Local and Remote Repository, Push, Pull, Merge and Branch.

URL - https://github.com/google/it-cert-automation-practice