📌 GitHub Training Plan

Objective: Build a strong foundation in Git and GitHub, ensuring hands-on experience in version control and collaboration.

🔹 Section 1: Understanding Version Control & Git

✅ What is Version Control? Why do we need it?
✅ Types of Version Control Systems (Centralized vs. Distributed)
✅ Why Git? How it works behind the scenes
✅ Installing Git on Windows/Linux/Mac

Hands-on:
🛠 Install Git & verify installation (`git --version`)
🛠 Configure Git with your name & email (`git config`)

🔹 Section 2: Git Basics – Local Repository Setup

✅ Initializing a Git Repository (`git init`)
✅ Cloning an Existing Repository (`git clone`)
✅ Understanding the Git Working Tree:

- **Working Directory → Staging Area → Repository**
  ✅ Adding & Committing Files (`git add`, `git commit`)

Hands-on:
🛠 Create a local Git repository
🛠 Add files, commit changes, and check commit history (`git log`)

🔹 Section 3: Working with GitHub

✅ Difference Between Git & GitHub
✅ Setting up a GitHub account
✅ Creating a GitHub repository
✅ Connecting local Git repo to GitHub (`git remote add origin`)
✅ Pushing and Pulling Changes (`git push`, `git pull`)

Hands-on:
🛠️ Create a GitHub repository & push a local project to GitHub
🛠️ Pull changes from a remote repository

🔹 Section 4: Exploring Git Logs & Undoing Changes

☑️ Understanding `git status`, `git log`, and `git diff`
☑️ Undoing changes:

- `git reset` (soft, mixed, hard)
- `git revert`
- `git checkout` for previous versions

Hands-on:
🛠️ Practice undoing commits and changes safely

🔹 Section 5: Working with .gitignore & GitHub README

☑️ What is `.gitignore` and why do we need it?
☑️ Creating a `.gitignore` file for best practices
☑️ Writing a **good README** for a repository
☑️ GitHub Markdown basics

Hands-on:
🛠️ Create a `.gitignore` file for a sample project
🛠️ Write a structured README.md file

🔹 Section 6: Introduction to Branching & Switching

☑️ What are branches in Git?
☑️ Creating and switching branches (`git branch`, `git checkout`, `git switch`)
☑️ Understanding HEAD and branch pointers

Hands-on:
🛠️ Create and switch between branches
🛠️ Explore `git log --oneline --graph` to visualize branches

🔹 Section 7: Merging & Handling Conflicts

☑ Fast-forward vs. 3-way merge
☑ Merge conflicts – why they happen & how to resolve them

Hands-on:
🛠 Merge two branches and resolve conflicts

🔹 Section 8: Forking & Pull Requests

☑ What is forking?
☑ How to fork a repository and contribute to open source
☑ Creating & managing pull requests (PRs)
☑ Code reviews & collaboration on GitHub

Hands-on:
🛠 Fork a repository and submit a PR
🛠 Review PRs and merge changes

🔹 Section 9: Working with Remote Repositories

☑ Remote repository basics (`git remote`, `git fetch`, `git pull`)
☑ Setting upstream branches (`git push --set-upstream origin <branch>`)
☑ Tracking remote changes

Hands-on:
🛠 Connect local branches to remote and sync changes

🔹 Section 10: Best Practices & Real-World Workflow

☑ Branching strategies (Feature branching, GitFlow, Trunk-based development)
☑ Handling large projects with multiple contributors
☑ Squashing commits (`git rebase -i`)

Hands-on:
🛠 Work on a mini team project using feature branches and pull requests

---