Detailed Syllabus for Git and GitHub

Module 1: Introduction to Git and Version Control

Learning Objectives:

- Understand the concept of Version Control Systems (VCS) and their importance.
- Learn the differences between centralized and distributed version control systems.
- Understand the role of Git as a distributed VCS and why it's widely used.

Topics:

1. What is Version Control?
2. Types of Version Control Systems: Centralized vs. Distributed.
3. What is Git? History and Benefits.
4. Git vs Other VCS (e.g., SVN, Mercurial).
5. Installing Git:
    - Installation on Windows, macOS, and Linux.
    - Configuring Git (git config for username and email).

Practical Tasks:

- Install Git on your system.
- Set up Git with your user name and email.

Resources:

- Official Git Documentation - Getting Started
- Git Installation Guide (YouTube)

Module 2: Core Git Commands and Local Repository

Learning Objectives:

- Learn the foundational Git commands for initializing and managing a local repository.
- Understand the staging area, working directory, and commit history.

Topics:

1. Initializing a Git Repository:
    - git init – Creating a repository.
    - .git folder structure.
2. Adding Files to the Staging Area:
    - git add – Adding files and folders.
    - Wildcards and selective staging.

3. Committing Changes:
    - git commit – Saving changes.
    - Writing meaningful commit messages.
4. Checking Status:
    - git status – Monitoring repository state.
5. Viewing Commit History:
    - git log – Reviewing commits with options like --oneline and --graph.

Practical Tasks:

- Create a local repository for a mini HTML project.
- Use git add, git commit, and git log commands to manage and track changes.
- View the differences between working, staging, and committed changes using git diff.

Resources:

- Git Basics Explained in Detail (YouTube)
- Learn Git from Scratch by Amigoscode

Module 3: GitHub – Remote Repositories and Collaboration

Learning Objectives:

- Understand GitHub and its role in collaborative development.
- Learn how to create remote repositories and push local repositories to GitHub.

Topics:

1. Introduction to GitHub:
    - What is GitHub and how it works.
    - Key features: repositories, pull requests, issues, forks, etc.
2. Setting Up GitHub:
    - Creating a GitHub account.
    - Configuring SSH keys or HTTPS for secure interaction.
3. Creating and Managing GitHub Repositories:
    - Creating a remote repository.
    - Cloning a repository using git clone.
4. Pushing Local Repositories to GitHub:
    - Adding a remote (git remote add origin).
    - Pushing changes using git push.

5. Pulling Changes from GitHub:
    - git pull – Fetching and merging updates from the remote.

Practical Tasks:

- Create a GitHub account.
- Create a GitHub repository for the mini HTML project and push your local project to it.
- Clone an existing repository from GitHub.

Resources:

- GitHub Full Beginner's Guide by freeCodeCamp
- Official GitHub Docs

Module 4: Git Branching and Workflow

Learning Objectives:

- Master Git branching for feature development.
- Understand the workflow of creating, merging, and deleting branches.
- Learn to resolve merge conflicts effectively.

Topics:

1. Understanding Branching in Git:
    - What are branches?
    - Use cases for branching in collaborative workflows.
2. Working with Branches:
    - Creating branches: git branch, git checkout -b.
    - Switching between branches.
3. Merging Changes:
    - git merge – Combining branch changes.
    - Fast-forward vs. three-way merge.
4. Resolving Merge Conflicts:
    - How merge conflicts occur.
    - Steps to resolve conflicts.
    - Using tools like VSCode for conflict resolution.
5. Deleting Branches:
    - Deleting merged branches (git branch -d).

Practical Tasks:

- Create a new branch for a feature in your project.
- Make changes to the branch and merge it back to the main branch.
- Simulate a merge conflict and resolve it.

Resources:

- Git Branching and Merging by Atlassian
- Git Branching Tutorial (YouTube)

Module 5: Advanced Git Techniques

Learning Objectives:

- Explore advanced Git concepts like stashing, rebasing, and cherry-picking.
- Learn how to rewrite commit history and handle large repositories.

Topics:

1. Stashing Changes:
    - git stash – Temporarily saving changes.
    - Applying and dropping stashes.
2. Rebasing:
    - git rebase vs. git merge.
    - Interactive rebasing for commit squashing.
3. Cherry-picking:
    - Selecting specific commits to apply to other branches.
4. Undoing Changes:
    - git reset (soft, mixed, hard) and git revert.
    - Restoring deleted commits.
5. Managing Large Repositories:
    - Shallow clones for large repositories (--depth flag).
    - Working with Git LFS (Large File Storage).

Practical Tasks:

- Use git stash to save uncommitted changes.
- Rebase a branch and squash commits.
- Undo a commit using git reset and git revert.

Resources:

- Advanced Git Tutorial by freeCodeCamp

- Git Pro Book by Scott Chacon

Module 6: Collaborating in Teams Using GitHub

Learning Objectives:

- Learn GitHub workflows for team collaboration.
- Understand forking, pull requests, and code reviews.

Topics:

1. Forking and Pull Requests:
    - Forking a repository.
    - Submitting a pull request for changes.
2. Code Reviews:
    - Reviewing pull requests and suggesting changes.
    - Best practices for collaborative coding.
3. Working with Issues:
    - Creating and managing issues on GitHub.
    - Linking issues to pull requests.
4. GitHub Actions:
    - Basics of automation with GitHub Actions.

Practical Tasks:

- Collaborate with a friend by forking their repository and submitting a pull request.
- Set up an issue tracker for your project.

Resources:

- GitHub Collaboration Workflow (YouTube)
- GitHub Docs on Pull Requests

Module 7: Real-World Git Workflows

Learning Objectives:

- Implement Git workflows used in professional software development environments.
- Learn branching models like Git Flow.

Topics:

1. Introduction to Git Workflows:
    - Centralized workflow.

    - Feature branch workflow.
    - Forking workflow.
2. Git Flow:
    - Setting up and using Git Flow.
    - Release branches, hotfixes, and long-term support (LTS) strategies.

Practical Tasks:

- Set up a Git Flow workflow for a project.

Resources:

- Git Flow Workflow Tutorial
- GitLab Git Flow Tutorial

Additional Learning Resources

1. **YouTube Channels:**
    - freeCodeCamp
    - Amigoscode
    - Traversy Media
2. **Books:**
    - *Pro Git* by Scott Chacon (free and detailed).
    - *Git Pocket Guide* by Richard E. Silverman.
3. **Practice Platforms:**
    - Git Kata – Interactive Git exercises.
    - Learngitbranching – Visual and interactive Git learning.

This syllabus ensures that you cover both the fundamentals and advanced aspects of Git and GitHub, empowering you to handle version control efficiently in real-world scenarios. Start by following the modules sequentially, using the resources provided for deep understanding.