🔧 Git & GitHub - Full Developer Guide

🔁 What is Git?

Git is a **distributed version control system (VCS)** used to track changes in source code during software development.

🧠 Why Use Git?

Benefit	Explanation
🕰️ Version Control	Track changes to your code
🧪 Rollback	Revert to older code anytime
👥 Collaboration	Multiple developers can work together
🌐 Distributed	Everyone has the full copy of code (local repo)
🧩 Branching	Isolate features, fixes, experiments

☁️ What is GitHub?

GitHub is a **cloud-based platform** that hosts your Git repositories online, making collaboration easy.

Feature	Description
☁️ Remote Repo Hosting	Host your code online
🧑‍🤝‍🧑 Team Collaboration	Manage teams, pull requests
✅ CI/CD Integration	Easily connect with Jenkins, Docker
🔒 Security	Access control, private repos
📦 Project Management	Issues, Wiki, Discussions

📁 Git vs GitHub

Git	GitHub
Local tool	Remote platform
Manages your code	Hosts your code
CLI-based	Web-based
Works without internet	Needs internet
Created by Linus Torvalds	Created by Microsoft (owns GitHub)

🧱 Basic Git Architecture

mathematica

CopyEdit

Working Directory → Staging Area → Local Repository → Remote Repository

---

🧪 Essential Git Commands (with examples)

Action	Command	Example
Initialize Git	git init	Create new repo
Check Status	git status	See changes
Add files	git add file.txt	Stage file
Add all	git add .	Stage everything
Commit	git commit -m "msg"	Save snapshot
See commits	git log	History
Connect remote	git remote add origin <url>	Link GitHub
Push to GitHub	git push -u origin main	Upload code
Pull latest	git pull	Get latest code
Clone repo	git clone <url>	Download repo

🌿 Git Branching

Command	Purpose
git branch	View branches
git branch dev	Create branch
git checkout dev	Switch branch
git merge dev	Merge into main
git branch -d dev	Delete branch

🌐 Real Example:

bash

CopyEdit

```
git checkout -b feature-login
# work...
git add .
git commit -m "added login"
git checkout main
git merge feature-login
```

🔁 GitHub Workflow (Typical)

1. Create repo on GitHub
2. Clone it to your PC
3. Make changes
4. add → commit → push
5. Others pull, give feedback
6. Merge via Pull Request (PR)

🔒 Common GitHub Terms

Term	Meaning
Repo	Code project
Pull Request (PR)	Ask to merge your changes
Fork	Copy someone's repo
Clone	Download repo locally
Issue	Bug or task
README.md	Project intro
.gitignore	Files to ignore (e.g., .class, target/)

📦 Real-World Scenario

You're working on a team. Each developer creates a **branch** for a feature, pushes to GitHub, and then makes a **Pull Request**. After code review, it's **merged** into main.

🧪 Common Git Interview Questions

1. What is Git and how is it different from GitHub?
2. What are the stages in Git?
3. What does git add do?
4. What is the difference between git pull and git fetch?
5. What is a merge conflict and how do you resolve it?
6. How does branching work in Git?
7. What is a fork in GitHub?
8. What is .gitignore?
9. How do you undo a commit?
10. Explain Git workflow in your team.

🔥 Tips to Master Git + GitHub

- Practice using Git on your projects.

- Always create separate branches.
- Write meaningful commit messages.
- Use .gitignore properly.
- Try **GitHub Projects, Issues**, and **Actions** for CI/CD.

🧾 Summary

Topic	**Summary**
Git	Local tool for tracking code
GitHub	Remote code hosting + collaboration
Key Commands	init, add, commit, push, pull, clone, branch
Branching	For safe feature development
Real Use	Team pushes code → PR → Review → Merge