**About GIT and GITHUB**

**Table of Contents:**

1. Introduction
2. What is Git?
3. What is GitHub?
4. Installing Git
5. Basic Git Commands
6. Creating and Managing Repositories
7. Branching and Merging
8. Collaborating on GitHub
9. Common GitHub Workflows
10. Connecting Git with GitHub
11. GitHub for Deployment
12. Example: Full Stack Python Project with Git and GitHub
13. Best Practices
14. Conclusion

**1. Introduction**

Git and GitHub are essential tools for any full stack developer. Whether you're working on a solo project or collaborating in a team, version control ensures code quality, traceability, and collaboration efficiency.

**2. What is Git?**

Git is a distributed version control system that allows you to track changes in your codebase, revert to previous stages, and collaborate with others.

**3. What is GitHub?**

GitHub is a web-based platform built around Git. It provides cloud storage for your repositories and tools for collaboration, issue tracking, CI/CD, and more.

**4. Installing Git**

**Windows:** Download from https://git-scm.com/download/win  
**Mac:** Use Homebrew: brew install git

**Linux:** Use package manager: sudo apt install git

Verify installation:

git --version

**5. Basic Git Commands**

git init # Initialize a new Git repository

git status # Check the current state of the repository

git add . # Stage all changes

git commit -m "Initial commit" # Commit changes with message

git log # View commit history

git diff # View changes made

git reset # Unstage files or revert commits

**6. Creating and Managing Repositories**

1. Create a directory: mkdir myproject && cd myproject
2. Initialize Git: git init
3. Add files: git add filename
4. Commit: git commit -m "message"

**7. Branching and Merging**

git branch new-feature # Create new branch

git checkout new-feature # Switch to new branch

git merge new-feature # Merge with main branch

git branch -d new-feature # Delete branch

**8. Collaborating on GitHub**

1. Create a GitHub account
2. Create a new repository
3. Connect local to remote:

git remote add origin https://github.com/username/repo.git

git push -u origin main

1. Clone a repo:

git clone https://github.com/username/repo.git

**9. Common GitHub Workflows**

* Fork and Pull Request
* Feature Branch Workflow
* Gitflow Workflow

**10. Connecting Git with GitHub**

git remote add origin <repo-url>

git push -u origin main

Set username/email:

git config --global user.name "Your Name"

git config --global user.email "you@example.com"

**11. GitHub for Deployment**

Use GitHub with platforms like:

* **Heroku**
* **Vercel**
* **Netlify**
* **Render**

**Steps (Heroku Example):**

1. heroku login
2. Create app: heroku create
3. Push code: git push heroku main

**12. Example: Full Stack Python Project**

**Backend:** Flask/Django  
**Frontend:** React.js  
**Database:** MongoDB/PostgreSQL

1. Initialize project folder
2. Set up Git: git init
3. Add all files: git add .
4. Commit: git commit -m "Initial full stack app"
5. Create GitHub repo
6. Push to GitHub
7. Deploy using Render/Heroku

**13. Best Practices**

* Commit often
* Write clear commit messages
* Use .gitignore file to exclude unnecessary files
* Pull changes before pushing
* Always test before merging

**14. Conclusion**

Git and GitHub form the backbone of modern software development. As a full stack developer, mastering them will enhance your productivity, collaboration, and deployment capabilities.