Git & GitHub for Job-Ready Developers

Subtitle: Learn version control and collaboration tools essential for real-world development and interviews.

Website Name: haas.dev
Website Link: https://dev-roast-app.vercel.app

Introduction

Version control is not optional—it's how developers track changes, collaborate, and showcase work. This guide teaches Git and GitHub step-by-step, so even beginners can manage projects professionally and impress recruiters.

Step 1: Understand Git Basics

Git is a **version control system** that tracks changes in your code.

Core Concepts:

- **Repository (repo):** Your project folder tracked by Git
- **Commit:** A snapshot of your project at a point in time
- **Branch:** Separate line of development for features
- **Merge:** Combine branches
- **Remote:** Online copy of your repo (e.g., GitHub)

Mini Exercise:
Create a local repository and make your first commit with a simple "Hello World" file.

Step 2: Git Workflow

1. **Initialize repo:** `git init`
2. **Check status:** `git status`
3. **Add changes:** `git add <filename>` or `git add .`
4. **Commit changes:** `git commit -m "Your message"`
5. **View history:** `git log`

Tip: Write clear commit messages: short + descriptive.

Step 3: Working With Branches

Branches allow multiple features or fixes without breaking main code.

- **Create branch:** `git branch feature-1`
- **Switch branch:** `git checkout feature-1`
- **Merge branch into main:** `git checkout main` → `git merge feature-1`
- **Delete branch:** `git branch -d feature-1`

Mini Exercise:
Create a branch, make a small change, merge it back into main.

Step 4: GitHub Basics

GitHub hosts your repositories online, making them shareable and trackable.

1. **Create account** on GitHub
2. **Push local repo to GitHub:**
    - `git remote add origin <repo_url>`
    - `git push -u origin main`
3. **Clone a repo:** `git clone <repo_url>`
4. **Pull latest changes:** `git pull origin main`

Tip: Every portfolio project should have a GitHub repo with README and live link if possible.

Step 5: Collaboration & Pull Requests

In real-world projects, multiple developers work together:

- **Fork:** Copy someone else's repo
- **Branch → Commit → Push** your changes
- **Pull Request (PR):** Request to merge your branch into the main project
- **Code Review:** Team reviews your PR for quality and best practices

Mini Exercise:
Fork a beginner-friendly open-source repo, add a small change, submit a PR.

Step 6: Best Practices for Job-Ready Git Use

- Keep commits **small & meaningful**
- Use **branches for features**
- Write a **README.md** with project overview and setup instructions
- Tag releases for portfolio projects
- Avoid committing **sensitive information** like passwords

Step 7: Track Your Progress & Portfolio Integration

- Keep GitHub profile **active**: at least 1–2 meaningful commits per week
- Showcase projects with **README, screenshots, live demos**
- Share GitHub link in resumes, portfolio, LinkedIn

Key Takeaways

- Git & GitHub are essential for professional development and collaboration
- Mastering branches, commits, and PRs makes you job-ready
- Your GitHub profile is **proof of ability**, not just a backup
- Continuous practice and clean commit history impress recruiters

---

Visit **haas.dev https://dev-roast-app.vercel.app** for step-by-step Git & GitHub tutorials, project workflows, and beginner-friendly exercises.

Website Name: haas.dev
Website Link: https://dev-roast-app.vercel.app