Path 2: Git & Version Control - Getting Started Guide

Month: 2
Duration: Self-Paced

Overview

This month focuses on learning Git through high-quality external resources. You'll work through 3 learning resources, then demonstrate your understanding through practical tasks.

No weekly schedule. Work at your own pace. Complete all tasks and submit documentation as evidence.

Learning Objectives

By the end of this month, you must:

✅ Complete all 3 Learning Resources
✅ Complete all 8 Practical Tasks
✅ Submit documentation with evidence for each task
✅ Have your Path 1 project on GitHub with proper Git workflow

Pass Criteria: Complete ALL learning resources + ALL tasks with documentation.

Setup: Install Git

Before starting, install Git on Windows.

Installation Steps

1. **Download:** https://git-scm.com/download/win
2. **Run installer** - Recommended settings:
    - Default editor: VS Code (if installed)

    - Initial branch name: Override → type `main`
    - Everything else: Default
3. **Verify installation:**
    - Press `Windows + R`, type `cmd`, press Enter
    - Type: `git --version`
    - Should show a version number
4. **Configure Git:**

```
git config --global user.name "Your Name"
git config --global user.email "your@email.com"
```

Use the same email you'll use for GitHub.

Learning Resources

Work through these 3 resources in order. Take notes and document your learning.

Resource 1: Git Handbook

Link: https://guides.github.com/introduction/git-handbook/

Time: 20-30 minutes

What to do:

- Read the entire handbook
- Take notes on key concepts
- Focus on understanding: repositories, commits, branches, remotes

Documentation Required:

Create a BookStack page: **"Git Concepts"**

Include:

1. **Definition:** What is Git? (in your own words, 3-5 sentences)
2. **Git vs GitHub:** Explain the difference (2-3 sentences)
3. **Key Terms:** Define these terms in simple language:
    - Repository
    - Commit
    - Branch
    - Remote
    - Clone

4. **Summary:** What are the 3 main benefits of using Git?

Evidence: Screenshot of your BookStack page.

Resource 2: Learn Git Branching (Interactive)

Link: https://learngitbranching.js.org/

Time: 2-4 hours

What to do:

Complete these levels:

Main Section:

1. Introduction to Git Commits
2. Branching in Git
3. Merging in Git
4. Rebase Introduction

Remote Section:

1. Push & Pull — Git Remotes!
2. Git Fetchin'
3. Faking Teamwork
4. Git Push

Total: 8 levels

Documentation Required:

Create a BookStack page: **"Git Commands Reference"**

Include:

1. **Commands List:** Every Git command you learned (minimum 10)
    - Format: `git command` - what it does
    - Example: `git commit` - saves a snapshot of your code
2. **Branching Diagram:** Draw or screenshot showing:
    - Main branch
    - Feature branch
    - Merge operation

3. **Challenges:** What was most difficult to understand? How did you figure it out?

4. **Key Takeaways:** 3 most important things you learned

Evidence:

- Screenshot showing completion of all 8 levels
- Screenshot of your BookStack page

Resource 3: Pro Git Book - Chapter 2

Link: https://git-scm.com/book/en/v2/Git-Basics-Getting-a-Git-Repository

Time: 1-2 hours

What to do:

Read **Chapter 2: Git Basics** - Focus on sections 2.1-2.5:

- 2.1 Getting a Git Repository
- 2.2 Recording Changes to the Repository
- 2.3 Viewing the Commit History
- 2.4 Undoing Things
- 2.5 Working with Remotes

Documentation Required:

Create a BookStack page: **"Git Workflow"**

Include:

1. **The Basic Workflow:** Describe the standard Git workflow:
    - Modify files
    - Stage changes (git add)
    - Commit changes (git commit)

2. **Command Reference Table:**

Command	Purpose	Example Usage
git init	Initialize repository	Starting a new project
git status	Check file status	See what's changed
git add	Stage changes	Before committing
git commit	Save snapshot	After staging

Command	Purpose	Example Usage
git push	Upload to remote	Share your work
git pull	Download updates	Get latest changes
git branch	Manage branches	Create/list branches
git checkout	Switch branches	Change working branch
git merge	Combine branches	Integrate features
git clone	Copy repository	Download existing project

3. **Undoing Mistakes:** Explain how to:
    - Undo changes before staging
    - Undo changes after staging
    - Undo last commit
4. **Key Concepts:** Summarize the 3 most important concepts from this chapter

Evidence: Screenshot of your BookStack page.

Practical Tasks

After completing all learning resources, demonstrate your skills through these tasks.

Task 1: Initialize Your Repository

Objective: Set up Git tracking for your Path 1 project.

Steps:

1. Open your Path 1 project folder
2. Right-click → "Git Bash Here"
3. Run: `git init`
4. Run: `git add .`
5. Run: `git commit -m "Initial commit"`
6. Run: `git log` to verify

Documentation Required:

Create BookStack page: **"Task 1: Repository Setup"**

Include:

- Screenshot of Git Bash showing successful commit
- Screenshot of `git log` output
- Any issues encountered and how you resolved them

Task 2: Create GitHub Account and Repository

Objective: Set up GitHub and create your first remote repository.

Steps:

1. Go to https://github.com and sign up
    - Use same email as Git configuration
    - Choose professional username
2. Verify your email
3. Create new repository:
    - Name: `path-1-program-manager`
    - Choose Public or Private
    - Do NOT initialize with README or .gitignore

Documentation Required:

Create BookStack page: **"Task 2: GitHub Setup"**

Include:

- Screenshot of your GitHub profile
- Screenshot of empty repository page
- Your GitHub username

Task 3: Push to GitHub

Objective: Connect local repository to GitHub and push your code.

Steps:

1. On GitHub repo page, copy the commands under "push an existing repository"
2. In Git Bash, run those commands:

```
git remote add origin https://github.com/YOUR-USERNAME/path-1-program-manager.git
git branch -M main
git push -u origin main
```

3. Refresh GitHub page - your files should appear

Note on Authentication:

- GitHub requires Personal Access Token, not password
- If prompted: GitHub → Settings → Developer Settings → Personal Access Tokens → Generate new token (classic)
- Give it "repo" access
- Use token as password when Git asks

Documentation Required:

Create BookStack page: **"Task 3: First Push"**

Include:

- Screenshot of successful push in Git Bash
- Screenshot of your code visible on GitHub
- Any authentication issues and how you resolved them

Task 4: Create and Merge a Branch

Objective: Demonstrate branch workflow.

Steps:

1. Create branch: `git checkout -b feature-test`
2. Make a change to any file
3. Stage and commit: `git add .` then `git commit -m "Test feature"`
4. Switch to main: `git checkout main`
5. Merge: `git merge feature-test`
6. Delete branch: `git branch -d feature-test`
7. Push: `git push`

Documentation Required:

Create BookStack page: **"Task 4: Branch Workflow"**

Include:

- Screenshot of branch creation
- Screenshot of successful merge
- Screenshot of `git log --oneline --graph` showing branch history
- Explanation of what you did in each step

Task 5: Resolve a Merge Conflict

Objective: Create and resolve a merge conflict intentionally.

Steps:

1. On main, edit line 10 of index.html to "Version A", commit
2. Create branch: `git checkout -b conflict-branch`
3. Edit same line 10 to "Version B", commit
4. Switch to main: `git checkout main`
5. Edit same line 10 to "Version C", commit
6. Try merge: `git merge conflict-branch` - This will conflict
7. Open file, you'll see markers: `<<<<<<<` , `=======` , `>>>>>>>`
8. Edit file to keep what you want, remove markers
9. Save file
10. Stage: `git add index.html`
11. Commit: `git commit -m "Resolve merge conflict"`
12. Push: `git push`

Documentation Required:

Create BookStack page: **"Task 5: Conflict Resolution"**

Include:

- Screenshot of conflict message in Git Bash
- Screenshot of conflict markers in file
- Screenshot of resolved file
- Screenshot of successful merge commit
- Step-by-step explanation of resolution process

Task 6: Create Version Tags

Objective: Tag versions of your project.

Steps:

1. Create v1.0 tag: `git tag -a v1.0 -m "Version 1.0 - Initial release"`
2. Push tag: `git push origin v1.0`
3. Make small change, commit it
4. Create v1.1 tag: `git tag -a v1.1 -m "Version 1.1 - Minor update"`
5. Push tag: `git push origin v1.1`

6. View on GitHub: Repository → Releases or Tags

Documentation Required:

Create BookStack page: **"Task 6: Version Tags"**

Include:

- Screenshot of creating tags in Git Bash
- Screenshot of tags visible on GitHub
- Explanation of what version tags are used for

Task 7: Create Professional README

Objective: Document your project on GitHub.

Steps:

1. Create or edit `README.md` in your project
2. Include:
    - Project title
    - Description
    - Features list
    - How to run
    - Technologies used
    - Author name
3. Commit and push
4. View on GitHub (displays automatically)

Documentation Required:

Create BookStack page: **"Task 7: README Creation"**

Include:

- Screenshot of README rendered on GitHub
- Copy of your README text

Task 8: Complete Workflow Demonstration

Objective: Demonstrate full Git workflow from start to finish.

Steps:

1. Start on main branch, everything committed and pushed
2. Create feature branch: `git checkout -b feature-final`
3. Make 3 separate changes (edit 3 different files or features)
4. Commit each change separately with descriptive messages
5. Push branch: `git push origin feature-final`
6. Switch to main: `git checkout main`
7. Merge: `git merge feature-final`
8. Delete branch: `git branch -d feature-final`
9. Push: `git push`
10. Create tag: `git tag -a v2.0 -m "Complete workflow demonstration"`
11. Push tag: `git push origin v2.0`

Documentation Required:

Create BookStack page: **"Task 8: Workflow Demonstration"**

Include:

- Screenshot of `git log --oneline --graph --all` showing workflow
- Screenshot of v2.0 tag on GitHub
- Screenshot of commit history on GitHub (showing your 3 commits)
- List of all Git commands used in order
- Reflection: What was easiest? What was most challenging?

Documentation Checklist

Your BookStack must include:

Learning Resources Documentation

- ☐ Git Concepts
- ☐ Git Commands Reference
- ☐ Git Workflow

Task Documentation

- ☐ Task 1: Repository Setup
- ☐ Task 2: GitHub Setup
- ☐ Task 3: First Push
- ☐ Task 4: Branch Workflow

- [ ] Task 5: Conflict Resolution
- [ ] Task 6: Version Tags
- [ ] Task 7: README Creation
- [ ] Task 8: Workflow Demonstration

Total: 11 BookStack pages required.

Each page must include:

- Required screenshots as evidence
- Your explanations in your own words
- Problems encountered and solutions

Common Issues & Solutions

"fatal: not a git repository"

Problem: Not in a Git-tracked folder.
Solution: Navigate to project folder or run `git init`

Authentication Failed

Problem: GitHub doesn't accept passwords.
Solution: Use Personal Access Token instead of password.

"failed to push some refs"

Problem: Remote has changes you don't have.
Solution: Run `git pull` then try pushing again.

Merge Conflict

Problem: Git can't automatically merge.
Solution:

1. Open conflicted file
2. Find `<<<<<<<` , `=======` , `>>>>>>>` markers
3. Edit to keep what you want
4. Remove markers
5. `git add filename`

6. `git commit -m "Resolve conflict"`

Getting Help

When stuck:

1. Check error message - Git tells you what's wrong
2. Google the exact error message
3. Re-read the learning resources
4. Ask in Discord with:
    - What you're trying to do
    - Screenshot of error
    - What you already tried

Submission

When complete:

1. Verify all BookStack pages are created
2. Check all screenshots are included
3. Ensure GitHub repository is accessible
4. Notify mentor for review
5. Be prepared to demonstrate any Git command
