Fohlio Tech Course

Lesson 1: Introduction to Collaborative Development

Learning Goals

By the end of this lesson you will:

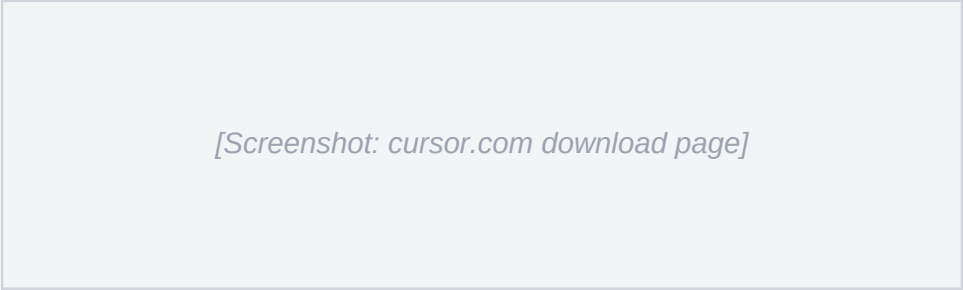
- Understand why Git is essential for teamwork
- Have Cursor and Git installed on your computer
- Know basic Git concepts: repository, commit, branch
- Be ready to complete your first homework assignment

Part 1: Installing Your Tools

Cursor

Cursor is an AI-powered code editor that will be your main tool for writing and editing code. It looks and works like VS Code but has built-in AI assistance.

1. Go to **cursor.com**
2. Click Download and install like any regular application
3. The free version is sufficient for this course



Git

Git is a version control system that tracks changes to your files. It's essential for collaborative work.

For Mac:

1. Open Terminal (search for it in Spotlight)
2. Type the following command and press Enter:

```
git --version
```

3. If Git is not installed, macOS will prompt you to install Command Line Tools. Click Install.

For Windows:

1. Go to **git-scm.com**
2. Download the installer
3. Run it and keep all default settings
4. After installation, open Command Prompt or PowerShell and verify:

```
git --version
```

Tip: If you see a version number like `git version 2.39.0`, Git is installed correctly!

Part 2: What is Git and Why Do We Need It?

The Problem Without Git

Imagine working on a document with a colleague. Sound familiar?

```
contract_final.docx

contract_final_v2.docx

contract_final_LAST.docx

contract_final_LAST_mary_edits.docx

contract_USE_THIS_ONE.docx
```

Now imagine hundreds of files and a team of 10 people working simultaneously. Chaos!

What Git Does

Git is a system that remembers the **complete history** of changes to your files. Each save (called a commit) is like a snapshot of your entire project at a specific moment in time.

Key Benefits:

- You can go back to any point in history
- Multiple people can work on the same project simultaneously
- You can see who changed what, when, and why
- Experiments don't break the main project

Three Core Concepts

1. Repository (repo)

A project folder that Git tracks. There are two types: **local** (on your computer) and **remote** (on GitHub, accessible by the whole team).

2. Commit

A saved snapshot of your project with a description of what changed. Think of it like a checkpoint in a video game - you can always return to it.

3. Branch

A parallel version of the project. You can experiment without affecting the main code. When your work is ready, you merge it back.

Part 3: Branches and Working Together

Why Branches Matter

The main branch is called `main` (or sometimes `master`). This is your "clean copy" - code that works and is ready to use.

When you need to add a feature or fix a bug, you create a separate branch. Work there, test it. When everything is ready - merge changes back to main.

> **Analogy:** `main` is like a published book. A branch is a draft of a new chapter. While the chapter isn't finished, readers see the old version. Once complete, you add the new chapter to the book.

The Basic Workflow

Step	Action
1	Create a branch from main
2	Make your changes
3	Commit (save changes with a description)
4	Push (send to GitHub)
5	Create a Pull Request
6	Someone reviews your code
7	Merge into main

Pull Request (PR)

A Pull Request is a formal request saying "please merge my changes into the main branch." It's created on GitHub.

Why use Pull Requests?

- Others can see exactly what you changed
- Code can be discussed and improved before merging

- History is preserved - you can understand why code looks the way it does

Merge Conflicts

Sometimes two people change the same part of the same file. Git doesn't know which version to keep - this is called a conflict.

Example:

Alice in her branch wrote:

```
button color: red
```

Bob in his branch wrote:

```
button color: blue
```

Git will show:

```
<<<<<<< HEAD
button color: red
=======
button color: blue
>>>>>>> bob-branch
```

You manually choose the correct version and delete the extra lines. Don't worry - Cursor and AI help resolve conflicts!

Part 4: Essential Git Commands

You'll use these commands in Cursor's terminal (open it with `Ctrl+`` or `Cmd+`` on Mac).

What You Want	Command
Clone a repository from GitHub	`git clone <url>`
Create a new branch and switch to it	`git checkout -b branch-name`
See what files changed	`git status`
Add files to be saved	`git add .`
Save with a message	`git commit -m "description"`
Send to GitHub	`git push`
Get latest changes	`git pull`
Switch to another branch	`git checkout branch-name`

A Typical Work Session

```
# 1. Get the latest code from main
git checkout main
git pull

# 2. Create your branch
git checkout -b fix-login-button

# 3. Make your changes in Cursor...

# 4. Save your work
git add .
git commit -m "Fix login button alignment"

# 5. Send to GitHub
git push

# 6. Go to GitHub and create a Pull Request
```

Homework Assignment

Required (for everyone):

1. Install Cursor

Download from cursor.com and install

2. Create a GitHub Account

Go to github.com and sign up

Send me your username

3. Your First Commit

a) Create a new repository on GitHub

b) Clone it to your computer using Cursor

c) Create a file called `totem.txt`

d) Write the name of your totem animal inside

e) Commit and push to GitHub

Advanced (optional, for those who want more):

1. Learn the Basics of Web Development

Read brief introductions to HTML, CSS, and JavaScript (links below)

2. Build a Simple Web Page

Create an `index.html` file with:

• A button

• When clicked, an image of your totem animal appears

Push this to your GitHub repository

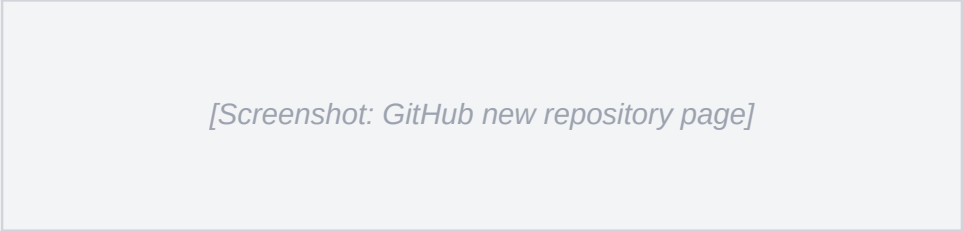
Helpful Resources:

- HTML Basics: developer.mozilla.org/en-US/docs/Learn/HTML
- CSS Basics: developer.mozilla.org/en-US/docs/Learn/CSS
- JavaScript Basics: developer.mozilla.org/en-US/docs/Learn/JavaScript
- Git Cheat Sheet: education.github.com/git-cheat-sheet-education.pdf

Step-by-Step: Completing the Homework

Step 1: Create a Repository on GitHub

1. Go to github.com and log in
2. Click the **+** icon in the top right, select **New repository**
3. Name it `my-first-repo`
4. Check **Add a README file**
5. Click **Create repository**



Step 2: Clone to Your Computer

1. On your repository page, click the green **Code** button
2. Copy the HTTPS URL
3. Open Cursor
4. Open the terminal (`Ctrl+`` or `Cmd+``)
5. Type:

```
git clone https://github.com/YOUR-USERNAME/my-first-repo.git
```

6. Open the folder: **File > Open Folder > my-first-repo**

Step 3: Create Your File

1. In Cursor, right-click in the file explorer on the left
2. Select **New File**
3. Name it `totem.txt`
4. Write your totem animal name inside and save (`Ctrl+S`)

Step 4: Commit and Push

In the terminal, type these commands one by one:

```
git add .
git commit -m "Add my totem animal"
git push
```

Step 5: Verify on GitHub

Go back to your repository on GitHub and refresh the page. You should see your `totem.txt` file!

> **Congratulations!** You've just completed your first Git workflow: clone, edit, commit, push. This is the foundation of everything we'll do in this course.