Version Control & Git

AI109

The Problem We’re Solving

- You edit files over time
- You make mistakes
- You want to undo changes
- You want proof of progress
- You want to try ideas without breaking things

Life Without Version Control

- Files named:
  - final.docx
  - final_v2.docx
  - final_REAL_final.docx
- No clear history
- Hard to go back
- Easy to lose work

What Is Version Control?

- A system for tracking changes to files
- Records *what* changed and *when*
- Lets you move backward and forward in time
- Works for code, text, and data

Why Version Control Matters

- Protects your work
- Encourages experimentation
- Makes collaboration possible
- Creates a visible history of progress

Version Control as a Time Machine

- Every saved version is remembered
- You can compare any two points in time
- You can recover deleted or broken work
- Nothing is truly “lost”

Individual vs Shared Version Control

- Personal history (your own work)
- Shared history (working with others)
- Multiple people, one project
- Changes are coordinated, not overwritten

Tools That Do Version Control

- Many systems exist
- Some are simple, some very powerful
- Today’s focus:
- Git as the tool
- VS Code as the interface

What You'll Learn Today

- The idea behind version control
- How Git implements it
- How to use Git safely through a GUI
- How this fits into your class workflow

What Git Is (and Is Not)

- A system for tracking changes to files over time
- Lets you save *versions* of your work
- Helps you recover from mistakes
- **Not** a backup service by itself
- **Not** the same thing as GitHub

Git vs GitHub

- **Git**: runs on your computer
- **GitHub**: a website that stores Git repositories
- You can use Git without GitHub
- VS Code connects Git and GitHub for you

Repositories

- A repository = a project with history
- Contains files *and* their change history
- Usually just a normal folder with Git enabled
- VS Code detects repositories automatically

The Git Mental Model

- Working directory: your files right now
- Staging area: changes you are about to save
- Repository: saved history
- Changes move step-by-step through these stages

Using Git in VS Code

- Source Control panel
- File status indicators in the editor
- Visual cues instead of commands
- Most actions are button-based

Tracking Changes

- Files start as untracked
- Edited files become modified
- You choose which files to stage
- Staging prepares changes for a commit

Commits

- A commit is a snapshot of changes
- Includes:
- Files you staged
- A commit message
- Commit messages explain *why*, not just *what*

Good Commit Habits

- Commit small, logical changes
- Write clear messages
- Commit often
- Commits help you and your instructor understand progress

Branches (Big Idea)

- A branch is a separate line of work
- Lets you experiment safely
- Main branch holds stable code
- VS Code shows your current branch

Working with Branches in the GUI

- Create a branch with a click
- Switch branches visually
- Merge branches when ready
- No typing required

Remotes

- A remote is a copy stored elsewhere
- Usually on GitHub
- Keeps work safe and shareable
- VS Code shows sync status

Cloning and Syncing

- Clone: download a repository
- Push: send your commits
- Pull: get new commits
- Sync buttons combine steps

GitHub in VS Code

- Sign in once
- Publish repositories easily
- View commits online
- Used for submissions and collaboration

Reading History

- View commit list in VS Code
- See who changed what and when
- Compare versions of a file
- Useful for debugging and grading

Understanding Diffs

- Green lines: additions
- Red lines: deletions
- Side-by-side comparisons
- Shows exactly what changed

Making Mistakes Safely

- Discard uncommitted changes
- Revert a commit
- Git remembers your past work
- Most mistakes are recoverable

When to Ask for Help

- Merge conflicts you don’t understand
- Changes disappeared unexpectedly
- Unsure what a button will do
- Asking early prevents bigger problems

Git Etiquette for Class

- Do your own commits
- Don’t share private repositories unless told to
- Never commit passwords or keys
- Git is part of your workflow, not just submission

What the GUI Is Doing for You

- Buttons map to Git commands
- VS Code hides complexity at first
- Understanding this later gives you more control
- Terminal Git is optional, not required now

Big Picture

- Git helps you work confidently
- You can experiment without fear
- Your project has a visible history
- Git is a professional skill you are learning early