OIM3640 - Problem Solving and Software Design

Version Control and GitHub*

*inspired by Ties de Kok

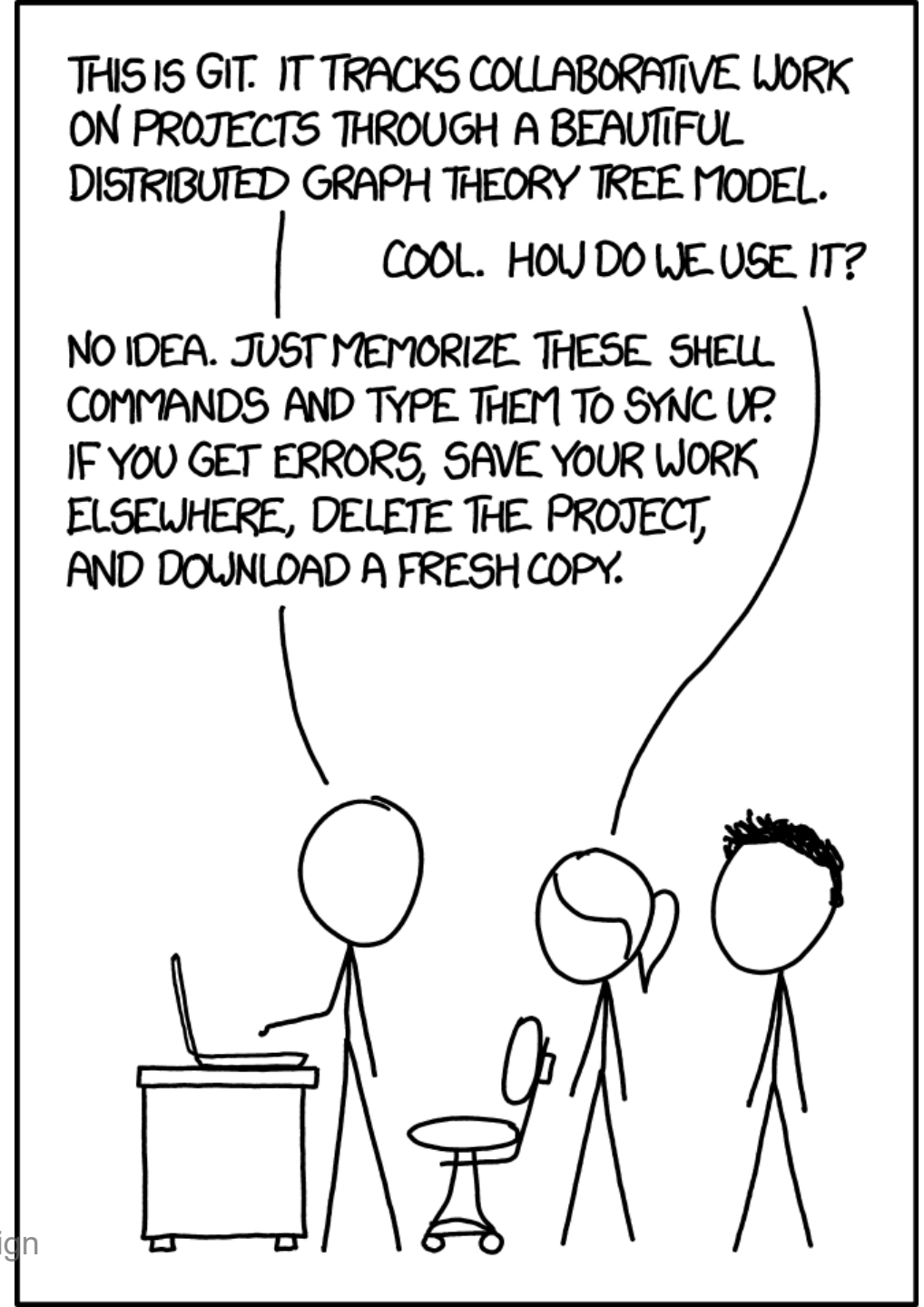
Version Control

- **Version control** is a method of keeping track of changes to a file or group of files over time, allowing users to access and revert to previous versions as needed.
- **Why** do you need it?
  - Do you have folders that look something like this?

```
essay_v1.2 [20230101]
essay_v1.2.1
essay_20230103_edited_Prof_Li
code_v1.2 (OLD!)
code_v1.1 (DO NOT DELETE!)
```

  - Version control solves this!

Git


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

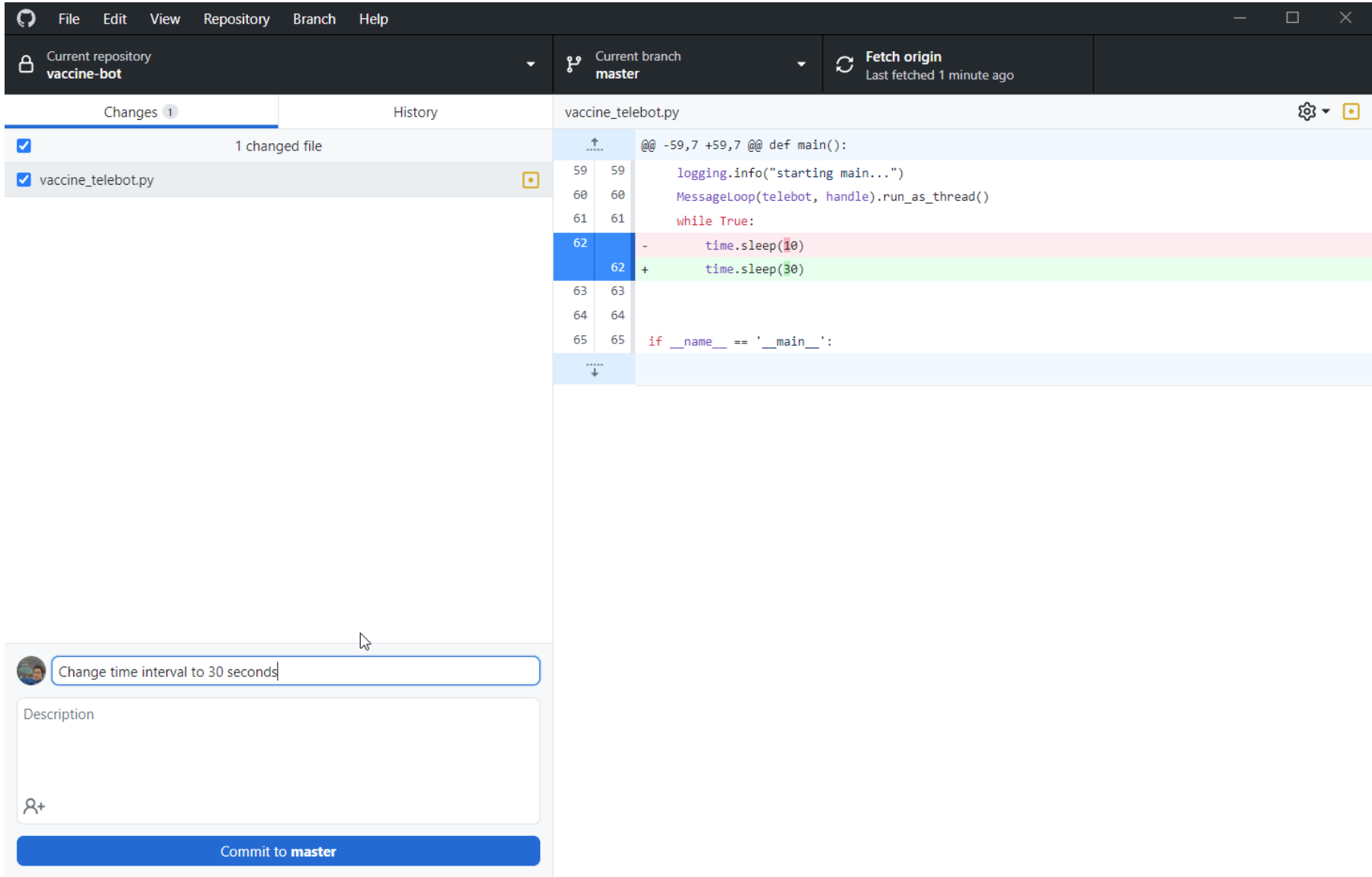
Git

- How to use version control?
  - The most widely used version control software is called **git** (Link)
  - You can run `git` locally, but it is better to use an online provider.
  - Storing your version control online makes it much less likely to lose it!
- Major providers for **git**:
  - **GitHub**
  - BitBucket
  - GitLab
- It is widely accepted that **GitHub** is the best choice.

GitHub

- **GitHub** is a web-based hosting service for version control using Git.
- **Why** do I use GitHub?
  - Clean and easy to use **interface**
  - Native support for **markdown** rendering
  - Convenient **GitHub Desktop** application
  - Free and unlimited **private** repositories for everyone
  - Integration with GitHub Actions for **CI/CD**
  - **Bonus feature**: ability to host webpages for free with GitHub Pages.

GitHub Desktop



Basic Workflow - First Time*

1. Open GitHub Desktop and select "New repository"
2. Choose a location for the repository on your computer, and give it a name. Make sure it is not inside any existing git folder
3. Select the option to include a "README.md" file when creating the repository.
4. Select the appropriate **Git ignore** type, depending on the type of project.
5. Click "Create repository" and make changes to files
6. Commit changes and then "Publish repository" to GitHub.com

**Note: You can also create a repository on GitHub.com website and then clone it to your computer and work on it.*

Basic Workflow - Working on Project*

1. Sync your local repository with the remote one by "pulling" the latest changes.
    - This ensures that the code is most up-to-date before making changes.
2. Edit the files in your preferred code editor, such as VSCode.
3. Before committing, sync your local repository with the remote repository on GitHub by "pulling" or "fetching" the latest changes.
    - This helps you avoid and resolve conflicts.
4. Create a "commit" to save the changes you have made locally.
5. Push the commit(s) to the remote repository on GitHub.com.
    - This "uploads" your changes to GitHub, making them available to other collaborators.

**Note: This is a very basic workflow. It is recommended to use Git branching strategy for managing different version of code in parallel and merge them later.*

Gitignore file

- There are a lot of things you don't want to sync with GitHub:
  - Data
  - Credentials
  - "Byproduct" files
- A ***.gitignore*** file allows you to select files that should be automatically ignored by git.
  - Ignoring files in the Git documentation
  - .gitignore in the Git documentation
  - A collection of useful .gitignore templates in the github/gitignore repository

Learn More about Git & GitHub

- Official documentation of Git
- Pro Git Book
- Learn Git Branching
- Lecture 6: Version Control (git), *The Missing Semester of Your CS Education*, MIT