# **Teaching plan document: Using Git and GitHub in Your Project**

Background of the students and Intention:This lesson is designed for undergraduate or graduate STEM students with either zero or some/limited experience with Git and/or GitHub. This lesson starts from the beginning and instructs learners how to understand, get, and begin to use Git and GitHub individually and as a group.

# Learning Objectives

1. What is Git, GitHub, GitLab, and CLI? (We will focus on Git and GitHub)
2. Why Use Git/GitHub?
3. Getting Git and Using the CLI.
4. Signing up for GitHub.
5. Connecting Git and GitHub and Creating a new Repo – Adding a New File – Committing Changes.
6. Introduction to Git/GitHub Common Terminology: Staging, Merging, Trunks, Branches, Pushing, Pulling, Cloning.
7. What are Merge Conflicts and using PR (pull request)?
8. Student experiences, ideas, discussion, and debate.

# Outline

1. **What is Git, GitHub, GitLab, and CLI? (We will focus on Git and GitHub)**
2. **Class Thinking Exercise:** Think about a time when you wished you could go back to an older version of your code. Think about what you might do if you and a team of 7 other developers were working on the same block of code of development operation. What would be the challenges.
3. **Lecture-Style Overview with Visual Support**:
   1. What is Git and what is “Version Control”.
      1. Using Git on a command line (brief introduction)
   2. What is GitHub and how does it interact with Git (part 1)?
   3. What is GitLab (vs GitHub)?
4. **Why Use Git/GitHub?** 
   1. **Class Discussion:** Question posed to the class: *Suppose you are working together with a team of other developers on a large project. You can assume that you are building a website or a Python application (or similar). Everyone on your team will be working on the same “code”. What issues or types of challenges do you foresee?*
   2. **Lecture-Style Overview**: Once the class talks about potential challenges, we will discuss some ways that Git/GitHub can address these types of issues. Ideas including track changes, historical back-ups, merging, individual benefits, and team benefits.
5. **Getting Git and Using the CLI** 
   1. **Example-Style Overview:** Explanation of the initial goal/plan to download Git and live example. Explanation of the differences for those with PCs vs. MACs and those who already have Git.
   2. **Group Activity: Downloading Git**: Students will get into groups of ~5 and will download Git to their own machines. Students who have Git will update to the latest version if they wish. Students will be encouraged to offer each other support as they do this activity. Each group will go to <https://git-scm.com/downloads> to download Git.
   3. **Instructor-Lead Class Activity**: Once all students have Git, as a class (together and with visual support) we will all open the command line (CLI) to use Git. We (as a class actively) will try a few command line examples. [This will not be a deep or comprehensive tutorial on Git Bash, but rather a review of concepts and quick CLI practice.]
6. **Getting and Logging into GitHub**:
   1. **Overview:** Explanation of the initial plan to create a login for GitHub
   2. **Instructor-Lead Class Activity**: As a class and with a visual instructor-lead example, students will register for and log in to GitHub. <https://github.com/login> [Students who already have an account will not need to register]
7. **Connecting Git and GitHub - Creating a new Repo – Adding a New File – Committing**
   1. **Lecture-Style Overview:** Explanation/review about the relationship between Git and GitHub as well as ideas behind storing files on one’s computer vs the Cloud including concepts of file sharing, project/code sharing, and DevOps.
   2. **Instructor-Lead Class Activity**:As a class we will connect Git to GitHub, we will add a new repository, and will place a file into it.
8. **Introduction to Git/GitHub Common Terminology: Staging, Merging, Trunks, Branches, Pushing, Pulling, Cloning** 
   1. **Lecture-Style Overview:** Explanation of common terminology and definitions with visual support and example.
   2. **Group Activity**:Get into Groups of ~ 5. As a group, create a shared GitHub repo and place a Word Document into it that contains a 1 paragraph story (about anything). Each member of the group will then pull the file, update it by adding a new paragraph and their name under it (so everyone can see the adds), and then merge the results into GitHub. Key notes – everyone will create their own paragraph 😊
9. **What are Merge Conflicts and using PR (pull request)**
   1. **Class Discussion**: Question to the class: What might happen if two or more developers, working on the same code, make changes?
   2. **Lecture-Style Overview:** Explanation of “Merge conflicts” and “PR”.
10. **Student experiences, ideas, and discussion**
    1. **Class Discussion**: Based on past experiences and/or today’s new knowledge, what would be the most important thing you would tell a new user about Git and GitHub? Discuss and debate.
    2. **Take-Home Assignment (Group):** As a group, use Git and GitHub to build a 5-page website that has a menu and a main page. Students will present and discuss next week.