Objectives
To use GitHub and the GitHub flow model to collaboratively engineer, deliver, and evaluate a software product. Along with using GitHub features like the issue tracker and reviewing pull requests and using a project board to manage the project, in this assignment you will use Markdown to complete technical writing tasks and an appropriate programming language to implement a production quality software. As a side effect of working in a team, you will also experience challenges, such as the creation of merge conflicts in a version control repository, that force you to develop practical solutions. You will also gain experience in talking with team members and leaders, technical leaders, and the course instructor. Students will work together in a development team while mastering the technical and professional skills in the field of software engineering, completing an intermediate and final assessments of their progress. Students will also adhere to a code of conduct that governs how all team members will interact during the completion of software products. Students will receive both percentage and mastery grades on this project.

Reading Assignment
If you have not done so already, please read all of the relevant “GitHub Guides”, available at https://guides.github.com/, that explain how to use many of the features that GitHub provides. In particular, please make sure that you have read guides such as “Mastering Markdown” and “Documenting Your Projects on GitHub”; each of them will help you to understand how to use both GitHub and GitHub Classroom. To do well on this assignment you should also read all of the assigned chapters in the following textbooks: Cooperative Software Design, Head First Software Development, Engineering Software as a Service, Think Python, Exercises in Programming Style, and Python Testing with Pytest. You are also expected to find and read all of the online resources that you need to complete this software project. Please see the course instructor if you have questions on these reading assignments.

Project 2 Options
For the second software engineering assignment, you are invited to select a project that corresponds to one of the three options below.

1. Developing and Releasing a Suite of Pytest Plugins

For this option of a software project 2, your task is to collaborate with some members of your class to design, implement, test, deploy, and maintain one or more Pytest plugins that is implemented in the Python programming language. You are to form a small team to implement your own Pytest plugin. Everyone in the team should be responsible for working together to handle all technical and
management issues during the completion of this long-term software project that will result in one or a suite of Pytest plugins hosted in a GitHub organization. At the outset, you should collaboratively decide what plugin you will implement, leveraging your prior experiences with Pytest to inform your decisions. For instance, it would be useful to have a Pytest plugin that only runs the tests that focus on the modified parts of a Python program. Your team could also consider implementing a plugin that reorders a test suite so that those tests most likely to achieve high coverage or find a bug are run first. Another reordering plugin might run the test suite so that the fastest tests are run first. You could also consider implementing a plugin that uses statistics to localize the defect in the program given the history of the test runs and the test cases that fail. The student software engineers should work with the technical leads and the course instructor to identify one or more unique Pytest plugins that they can feasibly implement by this project’s due date.

2. Developing a New Feature in GatorMiner

For this option of a software project, your task is to collaborate with some members of your class to design, implement, test, deploy, and maintain a novel feature in the GatorMiner tool (https://github.com/Allegheny-Ethical-CS/GatorMiner). Everyone in the team should be responsible for working together to handle all technical and management issues during the completion of this long-term software project that will result in a merged new feature of GatorMiner. At the outset, you should collaboratively decide what feature you will implement, leveraging your prior experiences with GatorMiner to inform your decisions. Since project 1 solely focused on GatorMiner tool, for the second project the feature you decide to implement must be approved by the instructor or a TL before you begin working on the project to ensure your proposed feature is both feasible to complete in the timeframe given for software project 2 and that it also represents an adequate amount of work. As before, all new feature implementation must be done in a feature branch, and the team must get their pull request reviewed by TLs before merging it into the main branch.

3. New Software Project

For the third option of a software project, you are invited to propose a software project of your choosing. This can be a proposal to extend an existing open-source tool (other than GatorMiner) by designing, implementing, testing, deploying, and maintaining a new feature into the existing software. Alternatively, you can propose to design, implement, test, deploy, and maintain a brand new software. In the latter case, you will need to work with TLs and the instructor to ensure that your proposed project is feasible in the timeframe given for software project 2 and that it provides value. In both cases, your proposed project must be approved by the instructor or a TL before you begin working on it. Given the flexibility of this option, the choice of the programming language is not restricted to Python, instead it should be mandated by the language used in an existing tool you propose to enhance or a language that is most appropriate for the new software you write from scratch. If you decide to create a brand new software, the repository for this software must be created in the course organization. Additionally, you must still use appropriate GitHub flow practices as you work within your team to collaboratively implement the software. If you are enhancing an existing project, the instructor and the TLs must be able to get access to your work and your new implementation must be approved (through pull request) by the original author(s).
of the tool.

Software Engineering Requirements

To allow for some flexibility, the second software project does not dictate a specific software engineering framework for the team to adapt. Additionally, since multiple approaches may support the effective completion of the required software, this assignment does not dictate team organization or communication strategies. The students in the course should instead work with each other, the technical leaders, and the course instructor to identify team roles and strategies for effective organization and communication. Course members are free to select their own teams based on interest in various options and their skills as identified in class. Each team is restricted to no more than five members. The team members are expected to decide on the project management strategy, team member roles, tasks each team member is to complete throughout the project, and the nature of collaboration the team will employ. In rare cases, and only if the chosen project includes enhancement to an existing software, a single person team will be allowed. The following are the baseline requirements for a software project 2, irrespective of the option chosen.

- Team decides on a software engineering model and follows a specific framework within that model. The choice of a framework needs to be based on the literature and/or existing software engineering practices. Teams are required to justify why the chosen framework is appropriate for their project and their team, which should include references to literature or documented practices in the industry. Teams are then required to demonstrate that their team actually followed the chosen framework, as documented in the progress and final reports and discussed during lab session updates and the presentation. Specifically, the selected framework should dictate how each team collaborates to ensure timely completion of the project.

- No matter the framework chosen, each team needs to manage their project in a project board (either ZenHub or GitHub Board), with pipelines that make sense for the project and the framework within which the team will operate.

- Each team uses GitHub and its features (e.g., issue tracker, pull requests, commit log, and code review request) to complete all of the tasks in the chosen project. Additionally, teams use either forks or branches of a GitHub repository to organize their work. While it is acceptable to have in-person discussions with team members or to talk about the project through Slack, please remember that all important discussions and decisions must be documented through GitHub and the project board.

- Each project must contain extensive testing with a high code coverage, ideally 100% or close to it. Teams are encouraged to practice test-driven approach, where you write a test case and then writing the code for that test case.

- Each project repository must include Continuous Integration using GitHub Actions that at the minimum runs the software, runs the test suite, and any linters as appropriate. Additionally, README should include badges that show status of GitHub Actions CI builds and project characteristics (e.g., coverage and language).
Each project repository must contain appropriate documentation, including information about the tool, its features, steps to be taken to install, run and test the tool, contributing guideline, dependencies, references, and contact information of the developers.

The written programs must be correctly formatted, documented, use correct naming conventions, use exception handling and refactoring as appropriate.

Reports and Presentation

For this project you are invited to submit two individual reports, one report to document your midway progress and a final project report. As you are working with your team, you should carefully document your experiences and contributions so that you can share them through writing in these two reports. For both report assignments, you are also required to include signed Code of Conduct to indicate that you adhered to it throughout the completion of the software project. Additionally, each class member is required to assess their efforts at midway and end points of the project and use the assessment document to record their progression of technical and professional skill development. You should thoughtfully reflect on your current areas of expertise and opportunities for improvement. In your final report assignment, you must complete self-assessment on all skills included in the assessment.md file.

During the lab session on May 12th, each team will give a short presentation/demonstration of the current status of their project.

Suggested Schedule for the Software Project 2

The course instructor invites the students in this class to work together to devise a schedule by which they can complete the software product by the stated deadline. Overall, you will work on this assignment for seven weeks. Here is a suggestion for a schedule to complete this project.

- **Week One: April 28 - May 5**: After deciding on your project, organizing your team and identifying the software engineering framework and process your team will follow, you should identify specific tasks to work on, organize your project board, and begin implementation.

- **Week Two: May 5 - May 12**: Continue implementation, while also ensuring project repository includes the required components such as CI, documentation, etc. Submit the progress report by the assigned deadline of May 7th.

- **Week Three: May 12 - May 18**: Plan to finish implementation, ensuring it is bug-free and fully tested, ultimately leading to a release of a production quality tool suitable for use by others. The final individual report and the software/feature is due by midnight of May 18th.