COSC 340: Software Engineering, Fall 2018

University of Tennessee, Knoxville
Dr. Austin Henley

Room: Min Kao 404
Time: Tue/Thu 3:40pm-4:55pm
Textbook: No textbook
Website: https://tiny.utk.edu/cosc340

Instructor: Dr. Austin Henley
Office: 353 Min Kao
Office Hours: Tue/Thu 1:30pm-3:30pm or by appointment
Email: azh@utk.edu

Teaching Assistants: Rayhan Hossain, Brandon Kammerdiener
Office Hours: By appointment
Email: rhossai2@vols.utk.edu, bkammerd@vols.utk.edu

Course Goals

The course aims to prepare students for real-world software engineering tasks. This will include learning widely-used tools (e.g., git and VS Code) and working in a group to build a large software application that students can include in their portfolio/resume. Additionally, students will learn and apply processes that software engineering teams utilize to manage software projects (e.g., agile development and code reviewing).

Topics

- Software Requirements
- Software Analysis & Design
- Software Testing
- Software Configuration Management
- Software Project Management
- Software Engineering Process
- Software Engineering Research

Skills & Tools

- Version control (git and github)
- Integrated development environments (VS Code)
- Debugging
- Program analyzer (e.g., Pylint)
- Unit testing

- Documentation generator (e.g., Pydoc)
- Programming language (either Go, JavaScript, or Python)
- Code reviewing
- Usability testing
- If time permits: interview problem solving and contributing to an open-source project

Lectures and In-Class Activities

Most class time will be used for lectures. The slides will be made available on the course website. There will occasionally be in-class activities (e.g., writing a user story, meet with your group, short quiz, etc.). Although attendance will often not be explicitly recorded, missing a class with an activity, whether it is announced or unannounced, will result in a zero unless a documented excuse is provided or prior approval from the instructor.

Group Project

The focus of this course is a group project that consists of designing, building, and evaluating a software application. Students will form groups of 3-5, propose a project that is of suitable topic and scope for the course, and use specific development tools and processes to meet milestones. Over a series of 2-3 week iterations, the group will iteratively implement the software while applying a variety of software engineering techniques (e.g., version control management, user stories, unit tests, code reviewing, etc.). The group will be periodically given portions of class time to meet. Groups are expected to communicate regularly outside of class and setup their own system and schedule for doing so (e.g., every Tuesday evening on Slack). Grades will be given for each iteration to the individual based on their contributions (planning, coding, etc.), tracked via GitHub's Issues feature. Group grades will be given based on two demos of the software's functionality throughout the semester (attendance is required to receive credit). Additionally, individual contributions will be verified during the group demos to ensure features are implemented as claimed. (I may re-grade individual contributions as necessary.)

Note about using someone else's code: It is common while programming to use source code found from other resources (e.g., examples, tutorials, Stack Overflow). This is completely acceptable **if** you provide credit. For example, if you use a function from an online resource, add a comment with the URL of the resource. Even if you make considerable changes to it, it is best to add the original URL and a comment "Based on this SO answer that does …". However, someone else's code should make up only a small portion of any student's contribution.

Exams

There will be two exams that aim to evaluate students' ability to apply topics from the lectures. The specific topics will be disclosed in a class prior to the exam day. The exams will likely not contain any questions based solely on definitions, dates, names, or anything requiring only superficial memorization, nor will they be designed to be "tricky." The exam dates will be scheduled in advance, so students will not be allowed to take the exam on another date unless approved by the instructor.

Extra Credit

Throughout the semester there may be opportunities to go above-and-beyond that will result in extra credit applied to the student's overall grade. These will be judged on a case-by-case basis, in a manner that is fair to the entire class. Examples of opportunities for extra credit include: exemplary effort in a particular iteration of the group project, presenting a research topic to the class, contributing to an in-class activity, and contributing to an open-source project.

Grading

- 50% group project
    - 30% individual contributions (5 iterations worth 6% each)
    - 20% group milestones (2 demos worth 10% each)*
- 30% exams
    - 15% exam 1
    - 15% exam 2
- 20% in-class activities
    - Minimum of 4 graded activities

*If a student's average individual contribution grade leading up to a group milestone is below 60%, then that student's group milestone grade can be no higher than their average individual contribution grade up to that point. For example, individual contribution grades of 20% and 60% would mean the student cannot receive a group milestone grade higher than 40%.

I will use the standard grading scale to convert to letter grades (e.g., 90-100 is an A).

The instructor reserves the right to revise, alter or amend this syllabus as necessary. Students will be notified regarding any changes.