SOFTWARE ENGINEERING CONCEPTS – SUMMER ‘18
PROJECT DESCRIPTIONS & BASIC REQUIREMENTS

Given the greatly shortened nature of the course, the instructor ran initial requirements gathering meetings with the clients. I am attaching my minutes from those initial brainstorming sessions. While each project’s clients may be able to attend end-of-sprint reviews, the course instructor will act as Product Owner.

Like nearly all customers working with Agile developers, they will provide feedback on the executable released at each Sprint. The initial sets of requirements are incomplete and preliminary; you should expect changes.

Teams must accept a strict budget of $0 for this project. This should not prevent students from using software they already own or any free or open source software and libraries. Any use of outside resources MUST be clearly documented and be in keeping with the University at Buffalo’s Intellectual Integrity Policy.

CLIENT #0: DEVELOP PROJECT ON YOUR OWN

If your team has a project that you all would like to develop, this might be allowed. You must get the instructors permission before using this option. As you describe your proposed project, the instructor will be looking for:

- **The project is of an appropriate size and scale.** While we are only meeting for 6 weeks, this is still the equivalent of a semester-long class. A project that is too broad or large is fine (that is an ideal situation for Agile development), but permission will not be granted for projects that would not allow you to really learn software engineering concepts.
- **There is real demand for this project.** You do not need to have clients/customers at the start, but you should have a plan to attract users after the term is over. The goal is to develop meaningful software and that means it should serve a purpose beyond this course.
- **The project involves significant “new” development.** Extending an existing project is fine, but the proposed work must be entirely new. Since the focus is on software engineering (and not just coding), the new “development” needs to include requirements gathering, planning, AND coding.
CLIENT #1: ONLINE TA FEEDBACK SYSTEM

Introductory courses have always made use of graduate and undergraduate teaching assistants (TAs), but increased enrollments mean that even upper-level courses are now relying on TAs, too. The increased enrollments also limit the time that instructors can spend with any single student. With students’ interactions with the instructor limited, TAs also become the main points of contact in the course. But TAs have little practice teaching and, because they often direct small groups of students, will have an outsized impact on students’ experiences. This makes it important that students be able to provide feedback on their TAs to the instructor of that course.

Having a system that enables TA feedback is useful only if it is used. Students (like any user of a system) will only use a system when every interaction is natural, comfortable, and does not require more time/clicks than necessary. While positive feedback is very helpful, this system’s most important contribution is enabling complaints reach the instructor so they can fix issues before they grow and take over students’ course experiences. Making students confident that they can report complaints without fear of repercussions or reprisals will also be critical. The easiest way to do this is to allow students to provide feedback anonymously so that there is no record of which student(s) reported these issues. Depending on the nature and seriousness of the issue, faculty may want to follow-up to better understand what happened and make certain they act on this properly. Therefore, users should be offered the option of including a name and means of contact should the faculty member want to do more. Faculty need a way to log into your system and review the feedback for their course(s), but no other accounts should be possible.

This system was initially proposed by UB’s chapter of Scientista and was subsequently supported by the department's faculty. Maintaining this system will end up the responsibility of the IT staff and they require that the system use as few additional resources as possible. Development work should be done on departmental development machine (cheshire.cse.buffalo.edu) and will be limited to the current webserver and software. You can find these libraries listed at http://cheshire.cse.buffalo.edu/index.shtml. Teams will be provided access to cheshire starting the second week of the term.

- The system needs a panel which allows instructors to manage their courses. This includes making an offering of a course available/unavailable for feedback and adding and removing TA names from the course offering. This panel must be protected so that it can only be accessed by an instructor and that the instructor can use the data associated with their course.

- The system also needs a panel which allows students to provide feedback. Each feedback item MUST be associated with the offering of a course (and the instructor of that course must have made the offering available in the system). Because students may not know the TA’s name, the feedback SHOULD be associated with a TA but there also needs to be a method to describe the TA when they do not know the name. The feedback should include a description of the behavior being reported upon and how important the student thinks the feedback is. As a last step, the student should be asked if they want the instructor to contact them. When they want to be contacted, they will need to provide their name and a means of being contacted.

- Instructors will also need a way to review any feedback for their courses. When reviewing the feedback, it should be broken out by course offering and only the data from available courses should be reviewed. The feedback will need to be able to be organized so that the instructor can review and work with the items in the order they feel is appropriate.

- To make the barrier to entry as low as possible, the feedback submission form must be usable on both desktop and mobile devices.

- Students should receive some acknowledgement upon the system saving their feedback correctly. Any error messages need to be informative and helpful.

- Changes to this system are inevitable, so it is important that the system be open for future additions and that the code be well commented and make improvements easy.

- Systems must ultimately be hardened against common web attacks such as: SQL injections, Cross Site Scripting (XSS), and Cross Site Request Forgery (CSRF)