CMPT 373 - D100 Software Development Methods

Instructor(s): Brian Fraser

Calendar Objective/Description:
Survey of modern software development methodology. Several software development process models will be examined, as will the general principles behind such models. Provides experience with different programming paradigms and their advantages and disadvantages during software development.

Instructor’s Objectives:
In this group project course, you will learn how to be an effective team member in an agile software development process.

You will work in a group with about 7 other students to complete a large semester long project. You will be assigned to a group by the instructor; however, you will be marked based on your individual contribution to the project, and how effectively you follow the required development process. Your group's project will be assigned by the instructor and may have a real “customer” who wants to put your project into active real-world use!

Your group project will be structured into iterations and developed using an agile methodology. The project may require you to program in a specific language (such as Java), or use specific tools or frameworks. You will make extensive use of Git and GitLab for contributing code and reviewing the work of your teammates. The course will include exercises to help you learn the basics of some required technologies, and you will work with your team to extend your skills and become proficient.

During lecture, you will learn several software development best practices and then apply these in your group work. Your group's code will be code reviewed in lecture by the instructor and fellow students to find ways to improve its design, clarity, and overall quality. Your active participation in lecture will help you become comfortable discussing many aspects of being a professional software developer.

Prerequisites:
CMPT 276 or 275.

Topics:
- Best practices: design patterns, refactoring, language-specific issues
- Agile software development: such as Scrum, extreme programming, or test-driven development
- Managing complexity and designing maintainable software
- Software-engineering tools and environments
- Software development process models: component-based development, iterative processes
- Requirements gathering and teamwork

Grading:
Project: 50% Reading Responses: 20% Exercises and in-class activities/quizzes: 30% To be confirmed in the first week of classes.

Required Books:

**Reference Books:**

Clean Code, Robert C Martin, Prentice Hall, 2009, 9780132350884

**Academic Honesty Statement:**

Academic honesty plays a key role in our efforts to maintain a high standard of academic excellence and integrity. Students are advised that ALL acts of intellectual dishonesty will be handled in accordance with the SFU Academic Honesty and Student Conduct Policies (http://www.sfu.ca/policies/gazette/student.html).