CS 673 Software Engineering (F22)

Department of Computer Science
Metropolitan College
Boston University

Instructor Information

Name: Yuting Zhang
Office: 1010 Commonwealth Ave., Room 322
Email: danazh at bu dot edu
URL: http://people.bu.edu/danazh

Office Hours:

Monday, Wednesday 4:00-5:00pm or by appointment

Feel free to ask me any questions before or after class. You can always contact me by email. **Please always add "CS673 (or cs673)" in the subject of your email**.

Course Information

Lecture Time & Place
Wednesday 6:00-8:45 PM PSYB51

Prerequisites
At least two 500 level or above programming intensive courses. Or the instructor's consent. (This course is not about programming. However, programming skill is the prerequisite. Students should be **familiar** with **OO concepts** and **proficient** in at least **one high-level programming language** before taking this course. This course is better taken as a capstone course towards the end of your program study.)

Reference Books:

Preferred SE Textbook:

- Eric Braude, Michael E. Bernstein. ***Software Engineering: Modern Approaches (2rd Edition)***. Waveland Press, Inc. *(ISDN:9781478632306)*

Other Reference SE Textbooks:

- Robert C. Martin. Agile Software Development, Principles, Patterns, and Practices.

- Bernd Bruegge and Allen H. Dutoit. Object-Oriented Software Engineering: Using UML, Patterns and Java.
- Shari Lawrence Pfleeger, Joanne M. Atlee. Software Engineering: Theory and Practice
- Roger S. Pressman. Software Engineering: A Practitioner's Approach.
- Hans Van Vliet. Software Engineering: Principles and Practice.
- Ian Sommerville. Software Engineering
- Ian Sommerville. Engineering Software Products: An Introduction to Modern Software Engineering

Other Classical Books for Software Engineers

- Frederick P. Brooks, Jr. The Mythical Man Month.
- Elisabeth Freeman, Eric Freeman, Bert Bates, and Kathy Sierra. Head First Design Patterns.
- Martin Fowler, Kent Beck, Don Roberts. Refactoring: Improving the Design of Existing Code.
- Steve McConnell. Code Complete: A Practical Handbook of Software Construction.
- Robert C. Martin. Clean Code: A Handbook of Agile Software Craftsmanship.

Other Reading Materials

- Microsoft Security Development Lifecycle: https://www.microsoft.com/en-us/sdl/
- OWASP
    - SAMM project: https://www.owasp.org/index.php/OWASP_SAMM_Project
    - TOP 10: https://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project
    - Developer Guide: https://www.owasp.org/index.php/Category:OWASP_Guide_Project
    - Testing Guide: https://www.owasp.org/index.php/Category:OWASP_Testing_Project

Description (from Catalog)

Overview of techniques and tools to develop high quality software. Topics include software development life cycle such as Agile and DevOps, requirements analysis, software design, programming techniques, refactoring, testing, as well as software management issues. An overview of secure software development processes and techniques will also be introduced. This course also features a semester-long group project where students will design and develop a real world software system in groups using Agile methodology and various SE tools, including UML tools, project management tools, programming frameworks, unit and system testing tools , integration tools and version control tools.

Learning Outcomes

At the end of the semester, students are expected to

1. Describe and compare major software process models and activities in the software process.
2. Explain the agile methodology and techniques, and apply these in a real-world team-based project to develop a high-quality software system on time.
3. Be aware of some security risks in the software project and techniques to enhance the software security.
4. Use various SE tools proficiently including the UML tool, the project management tool, programming tools, testing tools, the version control tool, etc.
5. Communicate effectively with team members and customers.
6. Present clearly the software project in both the oral and written form.

Learning Outcomes Assessment

- Class participation
- Reading and study
- 3 Labs
    - Lab1: Set up Git (LO2, LO4)
    - Lab2: Requirement Analysis (LO2, LO4)
    - Lab3: Refactoring and TDD (LO2, LO4)
- Semester-long project: (All LOs)
- 3 Quizzes (LO1, LO2, LO3, LO4)
- The Final Exam (LO1, LO2, LO3, LO4)

This course is featured with a semester-long team-based project. Each team should have about 4-6 students. Every member of the team is expected to contribute a roughly equal share to the project.

Course Policies

Grading Policy

The grade that a student receives in this class will be based on the class participation, in-class exercises or quizzes, the project and the final exam. The grade is broken down as below. All percentages are approximate and the instructor reserves the right to make necessary changes.

- 5% on class participation
- 9% on lab assignments (3 small labs)
- 6% on quizzes (3 quizzes)
- 50% on the semester-long project

- 30% on the final exam

Letter grade/numerical grade conversion is shown below:

A (95-100)	A- (90-94)	
B+ (85-89)	B (80-84)	B- (79-77)
C+ (74-76)	C (70-73)	C- (65-70)
D (60-65)	F (0 – 59)	

Assignment Submission

All labs should be submitted directly on blackboard. The project assignments are mostly done on the github and the google drive. A summary should be submitted on blackboard.

Assignment Late Policy

Each assignment has a deadline. For all individual assignments, the late assignments will be penalized within three days with a penalty. No assignments will be accepted three days after the deadline.

All project deadlines are firm. A deadline miss means zero for the grade of that phase.

It is the students' responsibility to keep secure backups of all assignments.

Academic Integrity

Academic conduct in general and MET College rule in particular require that all references and uses of the work of others must be clearly cited. All instances of plagiarism must be reported to the College for action. *For the full text of the academic conduct code, please check* http://www.bu.edu/met/for-students/met-policies-procedures-resources/academic-conduct-code/.

Course Outline

This course is organized into six modules of about 2 or 3 lectures each.

Module 1

Topics:

1. Introduction to Software Engineering & Software Process: SE is difficult, SE Concepts and Terminology, SE Ethics, Software Process including Waterfall, Spiral Model, Unified Process, and Agile.
2. Agile Methodology & Software Project Management: Manifesto, Agile Principles, eXtreme Programming, Scrum, AUP, DevOps, DevSecOps, Software Quality, Software Configuration Management, Risk Management

Reading: Online Lecture Notes, Braude (Part I, II and III), or related chapters in other textbooks. Additional papers provided

Assignments: Lab1, Project Assignments

Module 2

Topics:

1. Requirement Analysis and Management using User Stories: Gather requirements, User Stories, The INVEST Principle, Acceptance Tests, User Story Management, Product Backlog, Velocity and Story Points, Epics and themes, Scope Creep, Technical Debt, User Story Management Tool
2. From Requirements to Design - UML class diagrams and state transition diagrams: UI Mockups, State Transition Diagrams, Entity-Boundary-Control, Class Diagrams

Reading: Online Lecture Notes, Braude, Part IV, or related chapters in other textbooks

Assignments: Lab2, Project Assignments, Quiz 1

Module 3

Topics:

1. High Level Design: Design Goals, Software Architecture, MVC, Layered and Tiered Architecture, SOA, Microservices, REST
2. Design Principles and Design Patterns: Class diagrams, OO Reuse, Design Principles, MVC Compound Design Patterns

Reading: Online Lecture Notes, Braude, Part V, or related chapters in other textbooks

Assignments: Project Assignments

Module 4

Topics:

1. Implementation: Programming languages and Frameworks, Coding Standards, Code Smells, Refactoring Techniques
2. Testing: Regression Testing, Type of Testing, Testing Plan, Test Cases, Test Code, TDD and BDD

Reading: Online Lecture Notes, Braude, Part I, II and III, or related chapters in other textbooks

Assignments: Lab 3, Project Assignments, Quiz 2

Module 5

Topics:

1. More UML Tools in Requirement Analysis and Design: Use Case Model, Class Diagrams, State Machine Diagrams, State Pattern, Sequence Diagrams
2. Testing Techniques: Whitebox Testing, Testing Coverage, Blackbox Testing, Domain Testing, Integration Testing

Reading: Online Lecture Notes, Braude, Part I, II and III, or related chapters in other textbooks

Assignments: Project Assignments

Module 6

Topics:

1. Secure Software Development Process: CIA and IAAA, Software Security, Seven Touchpoints, Microsoft SDL, OWASP SAMM
2. Software Security Practices: Security requirements, Architecture Risk Analysis, Microsoft STRIDE, Code Review, Secure Coding Standards, Security Testing, SAST, DAST, IAST, RSP, Top Vulnerability Lists

Reading: Online Lecture Notes, Braude, Part I, II and III, or related chapters in other textbooks

Assignments: Project Assignments , Quiz 3

Project Assignments:

The project assignments are mostly done through google drive and github. To help both students and the instructor keep track of the assignments, we also create a checkpoint for each assignment on the blackboard. Make sure to submit checkpoints on the blackboard.

There are two types of project assignments:

1. Individual assignments: each student should complete his/her own assignment through google drive and blackboard.
    a. Weekly report: fill a row in your own sheet each week in the group weekly report on google doc. After you are done, submit the check question assignment on the blackboard.
    b. Midterm and Final self and peer review: fill the review survey form on google form.

2. Group assignments: each group only needs to submit **one** copy of the whole group work on github and blackboard. Students will work on the group documents collaboratively on google doc. At each iteration release, the group leader or the configuration leader or some designated member will archive the documents on the github, together with the source code to create a release. After it is done, the group leader (or the designated member) should submit the checkpoints on the blackboard.
    a. Iteration 0 Release including
        i. Readme.md
        ii. Doc/ProjX_SPPP
        iii. Doc/ProjX_meetingminutes
        iv. Doc/ProjX_progressreport
        v. Doc/ProjX_presentation_iter0
    b. Iteration 1 Release including

i. README.md (updated)
ii. Doc/ProjX_SPPP (updated)
iii. Doc/ProjX_meetingminutes (updated)
iv. Doc/ProjX_progressreport (updated)
v. Doc/ProjX_SDD
vi. Doc/ProjX_Presentation_iter1
vii. Code/… : runnable source

c. Iteration 2 Release including
   i. README.md (updated)
   ii. Doc/ProjX_SPPP (updated)
   iii. Doc/ProjX_meetingminutes (updated)
   iv. Doc/ProjX_progressreport (updated)
   v. Doc/Proj1_SDD (updated)
   vi. Doc/ProjX_STD
   vii. Doc/ProjX_Presentation_iter2
   viii. Code/… : runnable source

d. Iteration 3 Release (from the master branch) including
   i. README.md (updated)
   ii. Doc/ProjX_SPPP (updated)
   iii. Doc/ProjX_meetingminutes (updated)
   iv. Doc/ProjX_progressreport (updated)
   v. Doc/ProjX_userstories (updated)
   vi. Doc/Proj1_SDD (updated)
   vii. Doc/ProjX_STD (updated)
   viii. Doc/ProjX_Deployment (if applied)
   ix. Doc/Proj1_Presentation_final
   x. Code/… : runnable source code

Course Schedule

(This is a tentative schedule. It is subject to change based on the class progress and students' feedback)

Both the lecture and the project use iterative approaches. The lecture includes two iterations. The project includes an initial planning and then three mini-project iterations.

Week # Date	Module #	Topics	Course Assignments	Project Assignments

Week 1 09/07	Module 1	Topic 1	Lab1	Iteration 0 Starts
Week 2 09/14	Module1	Topic 2		
Week 3 09/21	Module 2	Topic 1	Lab 2	Iteration 0 Presentation Iteration 1 Starts
Week 4 09/28	Module 2	Topic 2	Quiz 1	
Week 5 10/05	Module 3	Topic 1		
Week 6 10/12	Module 3	Topic 2		
Week 7 10/19	Module 4	Topic 1		Iteration 1 Presentation Iteration 2 Starts
Week 8 10/26	Module 4	Topic 2	Lab3	
Week 9 11/02	Module 5	Topic 1	Quiz 2	
Week 10 11/09	Module 5	Topic 2		Iteration 2 Presentation Iteration 3 Starts
Week 11 11/16	Module 6	Topic 1		
Week 12 11/23	Thanksgiving Break (No Class)			
Week 13 11/30	Module 6	Topic 2	Quiz 3	
Week 14 12/07	Final Project Representation			Final Presentation
Week 15 12/14	Study Period (No Class)			
Week 16 12/21	Final Exam			