MET CS 673/473: Software Engineering

Fall 2023, Section A1

Tuesday, 6:00 PM – 8:45 PM, CAS 213

Bora Eryilmaz, Ph.D.

Email: beryilma@bu.edu
LinkedIn Profile

Course website
Blackboard website

Office Hours: Before/after class or by prior appointment
(Last revised on August 20, 2023)

Course Description: This course is an overview of modern techniques and tools to develop high-quality software. Topics that you will learn about include software development lifecycle, requirements analysis, software design, programming techniques, code refactoring, testing, and software integration & delivery.
The course features a semester-long group project where you will design and develop a real-world software application using modern software development methodologies such as Agile and various software engineering tools, including project management & communication tools, programming frameworks, unit & system testing tools, integration & deployment tools, and version control tools.

Prerequisites: This is a capstone course to be taken after at least two programming intensive courses toward the end of a program of study. Familiarity with object-oriented design concepts and proficiency in at least one high-level programming language is required. Familiarity with web, desktop, or mobile application development preferred.

Textbook (recommended): Eric J. Braude, Michael E. Bernstein. *Software Engineering: Modern Approaches*. 2nd Edition. Waveland Press, 2011. (ISBN 13: 978-1-4786-3230-6).

Course Objectives: At the completion of this course, you will be able to:

1. Use major software development lifecycle steps as they apply to professional software development projects.

2. Apply modern software development techniques and best practices in real-world, collaborative projects to develop high-quality software systems on time and within budget.

3. Use various software engineering tools proficiently including project management and collaboration tools, software design tools, programming tools, testing tools, version control tools, etc.

4. Communicate more effectively with team members and customers.

5. Present clearly software projects in both the oral and written forms.

Credit Hours: 4

Grading and Assignments: Your grade in this class will be based on class participation (including your individual project contributions), lab assignments, quizzes, project presentations, and the success of your group project (including code, tests, design documents, documentation, and project status reports).

The grade breakdown is shown below. All percentages are approximate and the instructor reserves the right to make necessary changes. The group project success score will depend on the instructor's subjective assessment as well as the relative success of your project against all other projects in the course.

Class participation	5%
Lab assignments (3 labs)	10%
Quizzes (3 quizzes)	10%
Individual project contributions	10%
Project status reports (weekly)	15%
Presentations (mid-semester and final)	20%
Group project (success and all artifacts)	30%

The letter vs. numerical grade conversion is shown below.

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

- **Quizzes**
  - There will be 3 quizzes, which should all be submitted directly to Blackboard.
  - You are expected to work independently on quizzes.
  - **Late submissions will incur a 33% penalty per day**.
- **Labs Assignments**
  - There will be 3 lab assignments, which should all be submitted directly on Blackboard.
  - You are expected to work independently on lab assignments.
  - **Late submissions will incur a 33% penalty per day**.
- **Individual Project Contributions**
  - **The weekly project status report, which is described below, is required to contain a short section for each project team member, detailing his/her individual contributions to the project for the week in question.**
- **Group Project**
  - This course features a semester-long group project. Each team will have 4–6 members. Every member is expected to contribute an equal share to the project.

  - You are expected to work as a team on your group project, each of you performing various roles over the course of the semester.
  - You will be graded on the overall success of the group project as well as your individual contributions to it. The project success will be evaluated based on the completeness, correctness, complexity, and quality of your final product (including source code, tests, design documents, and product documentation).
  - **Each team is required to submit an initial Project Proposal (as a PDF or MS Word file) that should be approved by the instructor before proceeding with project development**.
  - **Each team is required to submit a weekly project status report (as a PDF or MS Word file) describing the overall status of the project, including all notes from your weekly team meetings. You will also need to include all relevant design documents along with the status report.**
  - **Each team will give a short mid-semester and a final presentation of their project.**
  - All project artifacts should be made available on GitHub (code files, test files, product documentation, etc.). **Your weekly project reports should contain a link to your GitHub project site**.

Course Policies:

- **General**
  - Note that this is a live document and is subject to change according to the progress of the class and your feedback. Remember to check the syllabus for updates occasionally.
  - **Late quiz, lab, and report submissions will incur a 33% penalty per day**.
- **Attendance and Absences**
  - Attendance and class participation is expected.
  - Students are responsible for all material covered in the lectures.
- **Academic Conduct Code**
  - Cheating and plagiarism will not be tolerated in any Metropolitan College course. They will result in no credit for the assignment or examination and may lead to disciplinary actions. Please take the time to review the Student Academic Conduct Code.
  - Use of AI tools such as ChatGPT is strictly prohibited for creating any **original content** for labs, project reports, etc. You are allowed to use AI tools to **improve** the content that you have first created yourself. However, you must explicitly indicate any such use in your labs, reports, etc. Any other use of AI tools in your submissions will be treated as plagiarism,

Course Outline: The weekly coverage might change as it depends on the progress of the class. However, you must keep up with the reading assignments and all labs, quizzes, reports, etc.

Week	Topics	Due
Week 1 (09/05)	• Course outline and syllabus • Introduction to software engineering, the goals and terminology of software engineering, software process, software development lifecycle • Reading assignments: Chapters 1 and 3	• Project teams assigned
Week 2 (09/12)	• Agile software processes, software configuration management, version control • Introduction to version control (GitHub lab and demo) • Reading assignments: Chapters 4 and 6	• Project proposals
Week 3 (09/19)	• Project management, software teams and roles, software organization and tools • Software estimation, scheduling, and planning • Reading assignments: Chapter 7 and 8	• Project status report • Lab 1 (GitHub)
Week 4 (09/26)	• Requirements analysis, analyzing high-level requirements, use cases • Reading assignments: Chapter 10 and 11	• Project status report
Week 5 (10/03)	• Software design, Unified Modeling Language, software design patterns • State transition diagrams and state machines (example and demo) • Reading assignments: Chapters 16 and 17	• Project status report • Lab 2 (Requirements Analysis)
Week 6 (10/10)	• No class - Substitute Monday schedule	• Project status report • Quiz 1 (all material)

Week	Topics	Due
Week 7 (10/17)	• Mid-semester project presentations • About 10-15 minutes per team to present the current state of their project	• Project status report • Presentation slides
Week 8 (10/24)	• User interface and API design • Reading assignments: TBD	• Project status report
Week 9 (10/31)	• Software architecture, detailed design • Classes, functions, algorithms, and reusability • Reading assignments: Chapters 18 and 19	• Project status report
Week 10 (11/07)	• Principles of implementation, coding standards, refactoring • Code reviews (examples and discussion) • Reading assignments: Chapters 22, 23, and 24	• Project status report • Quiz 2 (Software, UI, API design)
Week 11 (11/14)	• Introduction to software testing, unit testing, • Code coverage, edge cases, function-based tests • Reading assignments: Chapters 25 and 26	• Project status report
Week 12 (11/21)	• Integration testing, system testing • Manual and automated testing • Reading assignments: Chapters 27 and 28	• Project status report • Lab 3 (Unit Testing)

Week	Topics	Due
Week 13 (11/28)	• Special topic: software and digital accessibility • Web content accessibility guidelines (WCAG), ARIA Authoring Practices • Reading assignments: TBD	• Project status report
Week 14 (12/05)	• Special topic: software deployment, security, server management • Reading assignments: TBD	• Project status report • Quiz 3 (accessibility)
Week 15 (12/12)	• Final project presentations • About 10-15 minutes per team to present the final state of their project and demo their product.	• Presentation slides • Peer Reviews