CS 519: Spark! Software Engineering X-Lab Practicum

General Information

Instructors

Name	Role	Contact	Office	Hours
Langdon White	Prof	langd0n@bu.edu	MCS/CDS 410k, Suite 410	W: 11-12 & 4:30-5:30, F: 1-3pm
Matthew Miller	Prof	mattm@bu.edu	TBD	6-8 T TH
Harsh Mutha	TA	hmutha31@bu.edu	TBD	6-8 M, 1-230 W, 4-530 F
Zhengqi (Drago) Dong	CA	dong760@bu.edu	TBD	10-12 Th F
Kamila Uranayeva	CA	ukamila@bu.edu	TBD	12-2pm Th, 11:30-1:30pm F
Aira Cosino	PM	ascosino@bu.edu	N/A	N/A
Apoorv Upadhye	PM	aupadhye@bu.edu	N/A	N/A

Course

- Class Location: EOP 262
- Class Time: 5-6:15pm (17h-18:15) T/Th
- Course Dates: Spring 2022
- Course Credits: 4

Course Description

Summary

A sizable portion of computer science graduates go on to become software engineers. The Spark! Software Engineering X- Lab Practicum affords students

opportunities to work on real-world projects, thus highlighting some of the challenges unique to working on a substantive software project. Ultimately, we hope to bridge the gap between computer science and software engineering, through project-based learning.

We will divide our discussion into 3 parts:

1. Software definition, teamwork, and management
2. Design and development
3. Delivery and maintenance

We begin by exploring the various phases of software development methodologies. In this part of the course, we will study software development models and learn to use scrum, a flexible, iteration-driven approach to project management. We will also learn the delicate art of managing client expectations—a task that often involves understanding how software engineers and clients view time estimates differently.

In the second part of the course, we will focus on design, architecture, and development of software. We will practice writing maintainable code and working with legacy code. Central to this part of the course are the concepts of test-driven development and defensive programming, both of which we will examine and integrate into our own projects.

Finally, we will wrap up our discussion by devising a software delivery and maintenance plan. Here we will study continuous delivery and deployment, two distinct but compatible approaches to release reliable software in short cycles. We will chart and showcase our progress through a sufficiently complex semester-long project.

Prerequisites:

To ensure that students get the most out of this class, we require students to have taken CS411 (Software Engineering) or have equivalent experience. You must have a strong programming background. Familiarity with web and/or mobile application development is helpful, though not required. Please consult with course staff during office hours if you have questions about the prerequisites. Assignment 1 (the pass/fail diagnostic test) will help you assess your readiness to take this class.

You must pass this assignment to receive a passing grade in the class.

Hub Learning Outcomes

While students cannot receive Hub credit(s) for this course, here are the learning outcomes we expect in the language that the Hub Program uses.

Teamwork/Collaboration Outcome #1: As a result of explicit training in teamwork and sustained experiences of collaborating with others, students will be able to identify the characteristics of a well-functioning team.

A sizable portion of computer science graduates go on to become software engineers, an inherently collaborative role. This course affords students opportunities to work on real-world projects for external partners in teams and teaches students to apply software development methodologies and models when designing, developing and presenting these projects. Students will explore and apply the various aspects of collaboration necessary for web/mobile application development, including developing a team agreement, assigning roles and responsibilities, making use of scrum, operating in sprints, and presenting a well-designed, functional final project to both their peers and to their clients from industry. Teams will use a team agreement and mid-term review as a framework to establish expectations and provide feedback on those agreements.

Teamwork/Collaboration Outcome #2: Students will demonstrate an ability to use the tools and strategies of working successfully with a diverse group, such as assigning roles and responsibilities, giving and receiving feedback, and engaging in meaningful group reflection that inspires collective ownership of results.

As a team, students will determine the parts of the application & project that they will be responsible for. While students will have specific roles within the group, course content will cover the design, architecture and development of software as a whole, including the connections between the front and back end, allowing students to appreciate the work of each member of the team. Throughout the course of the semester, students are expected to provide each other with continuous feedback through a technique called "code review." Code reviews will allow individual team members to work together and provide critical feedback on each others' work, to improve their contributions to the project. Students will practice Scrum, an agile method, a common industry approach to teamwork, collaboration, and project management. This will include weekly planning sessions, scrum meetings, and retrospectives aimed at continuous product delivery.

Oral/Signed Communication Outcome #1: Students will be able to craft and deliver responsible, considered, and well-structured oral and/or signed arguments using media and modes of expression appropriate to the situation.

This course emphasizes the importance of clearly and coherently delivering oral presentations to external project partners. Each team of students will craft presentations for their client, in which, through the use of Powerpoint, computer demonstrations, and a carefully presented talk, the students present the design and development of their web and/or mobile application development projects.

At least once during the semester (at mid-term), students will present their progress to the class as a whole. Students will learn how to translate technical tasks into product outcomes that meet the goals of the client, most of whom do not come from a technical background. This translational ability will be a key benefit of this course.

Oral/Signed Communication Outcome #2: Students will demonstrate an understanding that oral/signed communication is generally interactive, and they should be able to attend and respond thoughtfully to others.

Students will engage with the client weekly to provide progress updates which will offer an opportunity to learn how to engage in a dynamic discussion around project goals, progress, and requested adjustments. Students will learn to practice deep listening to understand the priorities of the project partner and because of the need to deliver a final output by end of semester, they will learn to manage and communicate expectations in real-time. Students will also learn to communicate with team members, particularly around difficult topics like constructive criticism and coming from different fields and levels of expertise.

During and after project presentations, students will be prepared to take questions from project partners; these questions may be specific to the code base, the functionality of the application, etc. The instructor will provide feedback on how the team interacted with the audience during project presentations, the clarity of their presentations, etc.

Oral/Signed Communication Outcome #3:Students will be able to speak/sign effectively in situations ranging from the formal to the extemporaneous and interact comfortably with diverse audiences

At every stage in the project development process, students will be actively presenting their work and process to diverse audiences from project partners, to peers, to project managers, and the general public attending the end-of-semester showcase. These varied experiences will provide opportunities to share similar information in different formats and to audiences of varied levels of technical expertise deepening the students skills in translating technical concepts to layperson language. Teammates will present their progress and talk through any roadblocks with their teams during Scrum and peer evaluations, engage with their instructor and classmates during mid-term presentations, and formally present to an audience of industry professionals during final presentations.

Other Outcomes (e.g., School, Department, and/or Program Outcomes)

As a result of completing this course, students will be able to:

1. Plan, execute, and manage software engineering projects

2. Assess and write robust code
3. Appraise and extend software
4. Improve teamwork and communication

Instructional Format, Course Pedagogy, and Approach to Learning

In addition to regular lectures, we will also have labs. During labs, we will work on assimilating material covered in lecture into our projects. Labs are meant to be hands-on work sessions. See the Course Schedule (tentative) at the end of this document for details.

Books and Other Course Materials

There is no required textbook for this course. However, there are required readings. Usually articles found on the Internet. See the Course Schedule (tentative) for reading schedule and where to find them. For the readings, if the item is preceded by an "R" then this is a required reading. The expectation is that the reading has been completed **before** the lecture for which it is listed.

Courseware

- Piazza: https://piazza.com/bu/spring2022/dscs519/home (code cs-519-2022-01)
- Gradescope: https://www.gradescope.com/courses/353570
- Blackboard: https://learn.bu.edu/ultra/courses/_83792_1/cl/outline

Assignments and Grading

Assignments serve 2 purposes:

- Cement material learned in class
- Track team and project progress

Assignments are due the Monday before class at 11:59 pm and may be submitted up to 24 hours late with a 5% late penalty. No late submissions will be accepted after 24 hours. Assignments must be submitted on Gradescope. To account for emergencies, we will drop the assignment with the lowest score from your final grade calculations.

Grading: Final Grade

Your final grade will be a weighted sum of grades received in the following categories:

% of Grade	Category	Notes
15%	Assignments	Grading rubrics are available on individual assignment pages. You must pass the diagnostic assignment to receive a passing grade in the class.
5%	Attendance and Participation	N/A
65%	Project	See here.
5%	Peer evaluations	N/A
5%	Midterm presentation	Student teams will present their projects to the rest of the class. All students on a team will receive the same grade.
5%	Final presentations	Student teams will present their projects to the rest of the class. All students on a team will receive the same grade, barring special circumstances.

Given that this is a practicum, the project is central to this course and is worth 65% of your final grade. We will be partnering with BU Spark! to work on a semester-long, application (web and/or mobile) development project. Projects are sourced from external partners and are complex enough to provide students with real-world software engineering experience.

Grading: Project

% of Project Grade	Category	Description/Notes
20%	Stability	• Where applicable, are the backend and frontend connected? • Does the application respond to requests consistently? • Does the application have undocumented security vulnerabilities?
20%	Security	• Is the app vulnerable to known attacks (e.g. CSRF, XSS)? • Does the application function as intended? Are there any logical errors?
20%	Functionality	• Is the functionality of the application reflective of client needs?
30%	Code quality and organization	• Is the source code for the application well-structured? • Is there a reasonable separation of concerns? • Have abstractions been meaningfully utilized? • Is the code fully documented, files labeled, and project reproducible?
10%	Workflow	• Did the team meet periodically and operate in sprints? • Consistent forward progress over the semester?

Resources/Support/How to Succeed in This Course:

1. Office hours are available weekly throughout the semester.
2. Communicate with your Project Managers, **early and often**

Community of Learning: Class and University Policies

1. Course members' responsibility for ensuring a positive learning environment (e.g., participation/ discussion guidelines).
2. **Attendance & Absences.**
   Due to the sequential nature of the product creation experience and the goal of completing a product demo by the end of the semester,attendance is required. Missing more than 3 classes may affect your final grade. If you must miss class for any reason, please email ahead of time. Makeup exams are not offered, except in instances of emergencies (prior travel plans do not count).

3. **Assignment Completion & Late Work**.
   Assignments are due the Monday before class at 11:59 PM and may be submitted up to 24 hours late with a 5% late penalty. No late submissions will be accepted after 24 hours. Assignments must be submitted on Gradescope. To account for emergencies, we will drop the assignment with the lowest score from your final grade calculations.

4. **Academic Conduct Statement**
   Software engineering is an inherently collaborative endeavor. In most cases, you will find open source projects or code snippets on the internet that you might want to use in your own projects. While this is permitted, you **must** cite your sources appropriately. You are also responsible for ensuring that you have the original author's permission to use their work. The Open Source Initiative maintains an excellent page on the different types of software licenses and what you can and cannot do with them. Remember, source code with no mentioned license is, by default, not available for reuse.
   Using code you have borrowed from the internet without permission and/or attribution is an instance of plagiarism, which is a violation of the Academic Code of Conduct. If you are in doubt about whether something might be construed as plagiarism, please check with course staff and in general err on the side of caution.

5. **Collaboration on Assignments and Projects:**

   Unless explicitly stated, collaboration on assignments and projects among teammates is both allowed and encouraged.

6. **Disability Accommodations:**

   If you are a student with a disability or believe you might have a disability that requires accommodations, please contact the Office forDisability Services (ODS) at 617-353-3658 to coordinate any reasonable accommodation requests. For more information, please see http://www.bu.edu/disability.

Course Schedule (tentative)

For the readings, if the item is preceded by an "R" then this is a required reading. The expectation is that the reading has been completed **before** the next lecture. All assignments may be found in Piazza.

Date	Topics	Reading (R = Required)	Notes
			Assignment 1 (distributed). You must pass this assignment to receive a passing grade in this class.
Jan. 20	• Introduction, Logistics, Expectations, and Software processes (Waterfall, Agile, etc.) • Assignment 1 (diagnostic test) out. • Anatomy of an application. • Client-server and n-tiered architectures.	R: Cloud Native Application Architecture • R: Agile Manifesto • R: Agile Manifesto Principles	
Jan. 25	• HIgh-level containers. Demo/Lab • Collaborative software construction and requirements gathering. • High performance teams and team agreements.		
Jan. 27	• Ethics overview. • Ethics Assignment released.	R: Project Descriptions	
	• Partner pitches. • Assignment 1 due. • "Project Preference" Assignment out.	2 perspectives on User Stories • R: Atlassian User Stories • R: Your Best Agile User Story	For the reading, you only need to read up to the "Examples" in "Your Best Agile User Story".
Feb. 1	• Assignment 2a: Team agreements via PM. • Ethics followup. • Ethics Assignment due.		Assignment 1 due. Ethics Assignment due.
Feb. 3	• Assignment 3: Requirements Gathering released.	R: Git Book (up to chapter 4)	
Feb. 8	• Problem Statement & Backlog • Bug reporting • Communication	R: Git Book (chapters 5 and 6)	
Feb. 10	• Software licensing • Deployment		
Feb. 15	• Version control (Git).	R: OWASP Secure Coding Practices	Requirements Gathering Assignment due
Feb. 17	• Version control continued • Security considerations		
~~Feb. 22~~	Monday schedule. No class.	Docker labs	Git Project due
Feb. 24	Midterm Presentations discussed. Lab/work session		
Mar. 1	Deploying applications: An Introduction to CI/CD.		

Date	Topics	Reading (R = Required)	Notes
Jan. 20	• Introduction, Logistics, Expectations, and Software processes (Waterfall, Agile, etc.) • Assignment 1 (diagnostic test) out. • Anatomy of an application. • Client-server and n-tiered architectures.	R: Cloud Native Application Architecture • R: Agile Manifesto • R: Agile Manifesto Principles	Assignment 1 (distributed). You must pass this assignment to receive a passing grade in this class.
Jan. 25	• HIgh-level containers. Demo/Lab • Collaborative software construction and requirements gathering. • High performance teams and team agreements.		
Jan. 27	• Ethics overview. • Ethics Assignment released.	R: Project Descriptions	
Feb. 1	• Partner pitches. • Assignment 1 due. • "Project Preference" Assignment out. • Assignment 2a: Team agreements via PM. • Ethics followup. • Ethics Assignment due.	2 perspectives on User Stories • R: Atlassian User Stories • R: Your Best Agile User Story	For the reading, you only need to read up to the "Examples" in "Your Best Agile User Story". Assignment 1 due. Ethics Assignment due.
Feb. 3	• Assignment 3: Requirements Gathering released.	R: Git Book (up to chapter 4)	
Feb. 8	• Problem Statement & Backlog • Bug reporting • Communication	R: Git Book (chapters 5 and 6)	
Feb. 10	• Software licensing • Deployment		
Feb. 15	• Version control (Git).	R: OWASP Secure Coding Practices	Requirements Gathering Assignment due
Feb. 17	• Version control continued • Security considerations		
~~Feb. 22~~	Monday schedule. No class.	Docker labs	Git Project due
Feb. 24	Midterm Presentations discussed. Lab/work session		
Mar. 1	Deploying applications: An Introduction to CI/CD.		