CS673 Software Engineering

Department of Computer Science

Boston University Metropolitan College

# Project Description

In this course, we will work collaboratively on a real-world semester-long software development project using microservice and Restful APIs. We will together develop an online project-based learning platform that may contain the following services:

* User management service: user registration, login, profile, roles, permissions, team assignments etc.
* Project management service: project creation, edit, management, search, rating etc.
* Content/Resource management service: learning materials management, blog management, etc.
* Collaboration service: real-time chat, discussion forums, document sharing, etc.
* Assessment and analytics service: progress tracking, feedback, reports management, etc.
* Notification services: sending notifications about deadlines, feedbacks, announcements, etc

Using microservices can help improve scalability, maintainability and flexibility. Each microservice will be containerized in a docker container to simplify deployment and management. We will also have a separate UI team(s) to develop UI. We may also have a separate team to focus on integration and API gateway.

Each team will consist of 4-6 students to work on one or two services. To coordinate the teamwork, each team will have a team leader. Every member of the team is expected to **contribute a roughly equal share** to the project throughout the whole process. The single biggest determinant of success or failure of a project is the people on your team. Respecting your fellow team members and committing to teamwork is the key.

The instructor (and/or facilitators) may act as the customers of your project. The teams will interact with the customer in order to figure out **what the customer wants** and deliver the product. The payment will not be dollars, but your grade : ).

# Project Process Description

While several different software development models will be introduced in the lectures, this project will mainly adopt **an iterative and incremental development process**. We will focus on some ***agile software development methodologies and DevOps practices***, but also introduce non-agile approaches and incorporate some into our project as needed. We will have an initial planning phase (iteration 0), and then 3 development iterations. *In each development iteration a working and quality software increment will be produced, and thus each iteration is a mini-project, involving planning, requirement analysis, design, coding and testing. A short presentation with a demo will be required at the end of each development iteration in order to get feedback from the customers.* Every student should participate in at least one intermediate iteration presentation as well as the final presentation. Each intermediate iteration presentation can be just 5-10 minutes. The final presentation is usually about 30-40 mins.

Each team needs to schedule at least one or two hour(s) of weekly meetings. **Meeting minutes** are required for each team meeting. Students should also communicate with each other through other asynchronous methods such as discord, email, and/or phone calls for informal meetings to solve any questions and issues. Each student should also submit a **progress report** that records the progress in each week, plan for the next week and any issues encountered. Each group should also submit a progress report that records the progress in each iteration. Meeting minutes and progress reports are shared through Google drive.

## Initial Planning Phase

1. Form teams, designate team leaders, other leader roles and responsibilities for each role. Set up a communication plan.
2. Brainstorm the requirements of the project with the customer. Define the project title and a project vision, including purpose of the application, major functionalities and requirements etc. Research related work, describe the challenges, explore feasibility and learn tools. We will use pivotaltracker (<http://www.pivotaltracker.com/>) as the requirement analysis tool. If you prefer to use a different tool, please let me know.
3. Set up a project environment, which includes management tools and development environment. We will use GIT as the version control tool and github ([https://github.com](https://github.com/)) to host your project. All documents and code should be configuration items and stored in your version control system.
4. Identify risks and risk retirement plans. Estimate the time, make an initial schedule of iterations. Define quality metrics and the techniques to assure quality.
5. Draft your SPPP (Software Project Proposal and Planning) document. The template of SPPP will be given through google doc. The SPPP will include project proposal, management plan, QA plan, and configuration plan.
6. Submission: SPPP (including the Risk Management spreadsheet), presentation, meeting minutes, and weekly report. All should be archived in the GitHub repository.

## In Each Iteration

1. Planning and Requirement analysis: work with the customer to prioritize the requirements from the backlog and choose the most important ones to implement in the current iteration. It is possible to get new requirements from the customer too. Analyze each requirement, write user stories and acceptance tests for each requirement. Break down into tasks, and estimate the time for each task and/or requirement based on the experience or the previous iteration. Do the detailed schedule for the current iteration and coarse granularity schedule for later iterations.
2. Design, implementation and testing: identify design goals, choose the proper architecture. Identify objects/classes and their attributes and operations. Apply design patterns to enhance re-usability if applicable. Refactor code on a regular basis. Apply coding standards across the team, and document source code. Testing should be in parallel with implementation. This may involve unit testing, integration testing, system testing and acceptance testing. Each iteration should result in a working software that passed all tests. Draft SDD (Software Design Document) to include software architecture, subsystem decomposition, design patterns etc. Draft STD (Software Testing Document) to include test plans, test cases and test reports.
3. All code and documentation should be subject to peer review at the end of each iteration.
4. Take security into consideration in all activities.
5. Containerize the microservice.
6. Update SPPP, SDD, STD in each iteration.
7. Submission: presentation (with demo), deployment if applicable, requirement analysis on pivotal tracker or other tools, updates on any previous documents (SPPP, SDD, SDT, weekly reports, meeting minutes), as well as the source code (production and test code) on GitHub. Post iteration self and peer reviews.

# Final Project Presentation

Every student should participate in the final presentation. The final presentation should cover the whole project process and product, with more emphasis on the last iteration. All activities such as requirement analysis, design, implementation, testing, management should be included. In particular, the following items should be addressed:

1. Requirement analysis: overview of functional requirements using use case diagram, user stories (examples, total # (planned vs completed), total points), how your project tracks requirements and handles requirement changes, nonfunctional requirements etc.
2. Design: software architecture, database, class diagram, patterns used etc.
3. Implementation: any special tools, experience, or techniques, refactoring examples,
4. Testing: types of testing, test case examples, testing metrics, such as testing coverage, such as number of tests passed, number of defects etc.
5. Deployment if applicable.
6. Security: any security issues, requirements, design, tools, implementation, testing used or considered should also be discussed.
7. Project management: role assignment as well as each individual contribution, risk management, quality management, particularly quality metrics, achievements of the project, challenges in the project, lessons learned.
8. Demo

There will be a final project evaluation form to fill out. Each student will evaluate the other group's presentation in the class.

# Final Project Submission

The following items should be included:

1. Github main branch: include all documentation (final version of SPPP, SDD, test case documentation, meeting minutes, weekly report, each iteration presentation), and all source code with comments, a description of the source code folder structures.
2. Pivotal Tracker: updated requirements with user stories
3. Final software: fully deployed website or an apk file ready for installation.
4. Final presentation evaluation form, completed by each student, submitted in class.
5. Final group/self/peer review form, completed by each student, submitted on BlackBoard.

# Project Grading

Project will be graded based on the performance in each phase and the final product. Since we use an iterative and incremental process, the later iteration will be affected by the cumulative results in the previous iterations. The deliverable for the final iterations will be the whole project. Each type of activity will be evaluated in term of 0-5 scale (0: Nothing done (F), 1: Very Poor (D), 2: Poor (C), 3: Fair (B), 4: Good (A-), 5: Excellent (A))

|  | Presentation | Management/Team work/ environment | Requirement analysis | Design | Implementation | Testing and Deployment |
| --- | --- | --- | --- | --- | --- | --- |
| Planning (10%) | 20% | 40% | 30% | 10% | N/A | N/A |
| Iter 1-2 (20%) | 10% | 15% | 25% | 25% | 15% | 10% |
| Iter 3 (whole project 50%) | 15% | 15% | 15% | 15% | 20% | 20% |

Presentation: At the end of each iteration, a presentation by each group is required. Except for the first planning phase, a demo is required in each iteration representation. Every student is required to participate in the final presentation, and any one of the previous presentations.

Management/Team work: Since this is a teamwork project. Management, communication and collaboration between members are very important. This activity is evaluated based on SPPP (and how SPPP is conducted in the project), meeting minutes, weekly report, group and peer reviews, and other informal feedback.

Requirement analysis: we will mainly use pivotal tracker to manage requirement analysis. It will also affect your final software product.

Design: SDD is the main document for this activity. Again, it will also affect your final product.

Implementation: the correctness, complexity and quality of the source code source will be evaluated for this activity. Refactor is an important practice to improve the source code quality.

Test: both unit testing and system testing should be performed. STD and the test code (e.g. unit test code and selenium test scripts) are the main deliverables for this activity.

Deployment: The final software should be easily deployed in the customer environment. If it is a desktop/mobile application, an executable file should be generated in order for the customer to easily install it. If it is a web-based application, it is better to be deployed on a web server in order for the customer to easily access it through common browsers such as chrome, firefox, IE etc.

The grade for each individual member will depend on your group project grade and his/her own contribution to the project, as well as type of activities based on your role in the group. Your self- review will be a very important way to promote your own contribution in the project.