WM393 Software Development Life Cycle

Assignment 2

1. Introduction

The purpose of this assignment is to assess the students with knowledge of software engineering methodology and the skills to apply it. This assignment is an extension of the previous assignment 1 and is performed individually. Based on the function you selected in the assignment 1, there are a further report and implementation required for this assignment. You need to submit the report with the implementation code for the assignment before specified deadline.

2. Scenario

The COVID-19 has results in schools and universities shut down across national level and even the world. As a result, education has dramatically changed with the distinctive rise of e-learning, whereby online teaching is being undertaken remotely through digital platforms such as Teams, Zoom, Meet, etc. With such a shift from offline teaching to online teaching, many people believe that the adaption of online teaching will persist even after COVID-19 ends. However, these digital platforms are mainly focusing on video conferencing and business communication. Their additional features provided by the platforms such as sharing or noticing information continuously may be limited for online teaching. For this reason, it is necessary to seek other online services from another digital platforms and use several digital platforms together for online teaching unfortunately.

Suppose that, despite Moodle learning management system, WMG is planning to develop a tailor-made system called **WMG Teaching Support System (WMGTSS)** to provide useful functions to effectively support online teaching for each teaching module. The basic functions for the WMGTSS are designed but the full requirement specification has not been clearly defined yet. Your team is assigned to this development and is responsible for creating this system to serve the purpose of the WMGTSS. There is the description for each basic function designed in the WMGTSS below:

- **Feedback board**

  The feedback board is a board where students can make any feedback during a lecture. There are conditions for this board. A tutor only can create one or more feedback boards in accordance with request. For example, a tutor likes to have feedback on a daily basis or on a teaching block basis (one board per 5 days). A student can leave any feedback as many as possible. Comments on a feedback is optional.

- **Notice board**

  The notice board is a board where a tutor can make any notice to students. There are conditions for this board. A tutor only can create one or more notice boards in accordance with request. A tutor only can make a notice and also assign a priority to a notice (such as a pin functionality). If a notice has a high priority, it should appear on the top of the notice list. Comments on a notice is optional.

- **Q&A board**

  The Q&A board is a board where a student can ask a question to a tutor and a tutor can give an answer for the question. Sometimes, a tutor can make a Q&A on behalf of students. A tutor

only can create one more Q&A boards in accordance with request. There are conditions for this board. A tutor or student only can create a question and a tutor only can leave an answer for the question. Comments on a Q&A is optional.

- **Discussion board**

  The discussion board is a board where a tutor can make a discussion topic for students. There are conditions for this board. A tutor only can create one discussion board. A tutor only can create a discussion topic on the board. A tutor or student can leave an opinion on each topic. There is no comments option.

- **Resource board**

  The resource board is a board where a tutor can upload any useful information for students. There are conditions for this board. A tutor only can create one or more resource board in accordance with request. A tutor only can upload any resource such as image, video, text, link, etc. Depending on resources, some are downloadable via a link and some have to be displayed such as image and video. Comments on a resource is optional.

- **Lecture board**

  The lecture board is a board where a tutor can upload the slides of a lecture note and a student can navigate the slides. There are conditions for this board. A tutor only can create one lecture board. A tutor only can create a lecture on a daily basis such as Day 1, Day 2, etc. and upload lecture slide images on each day. A student can navigate the lecture slides using navigation buttons. Uploading and downloading the original lecture slide such as PPT is optional.

- **Data file board**

  The data file board is a board where a tutor can upload file(s) and a student can download some or all. There are conditions for this board. A tutor only can create one or more data file board in accordance with request. A tutor can upload one or many data files with a description on each item. A student can download the data file via a link. Showing download counter on each file is optional.

- **Quiz board**

  The quiz board is a board where a tutor can make a quiz to check students' progress and a student can answer the quiz. There are conditions for this board. A tutor only can create one quiz board. A tutor can make a quiz with a number of questions. A student only can make the answers for the questions. A tutor can see the overall scores with good representation such as a graph, but a student can see own score. Deciding multiple attempts is optional.

- **Calendar board**

  The calendar board is a board where a tutor can inform any schedule or notice to students. There are conditions for this board. A tutor only can create one calendar board. A tutor only can create any event information on a day or days. A student can see the event information on monthly view or weekly view.

3. Task

Based on the function you selected in the assignment 1, the assignment 2 extends on the assignment 1 and additionally performs the following two tasks. First, you have to produce an individual report for the function (board) that you selected in the assignment 1. You should include the contents from

your assignment 1 in the individual report but needs to improve the contents for further requirements and implementation. The detail instructions for the individual report are described below. Second, you have to implement partial or close to a workable version of software based on the requirements and architecture design. Remember that it is not only the design of user interface, but you should implement some required features for the function you selected. The language decision is up to you, but the implementation should be based on the requirements defined in your individual report you have created. The detail instructions for the implementation are given below.

[Individual Report]

Here are the instructions of how to produce your individual report:

- You should include the contents for your function defined in the assignment 1 such as the functional requirements, non-functional requirement, user interface, etc.
- If necessary, you are free to improve the contents for your function previously defined in the assignment 1.
- If necessary, you are free to include some other contents related to your function from the assignment 1 such as the management of functions control, the management of users, non-functional requirements, etc.
- To improve the structural view in the architecture design, you must include at least one class diagram for your function.
- To improve the behavioural view in the architecture design, you must include at least one sequence diagram for your function.
- To include implementation technical review, you must include some important parts of your code and technical description with some screen captures. Remember that the code is not included in the total word counts but the description is included. So, put the code inside a table with a caption in order to make it an easy to recognise.
- To verify your implementation code, you must include at least one test case for some functions with testing code and description. Remember that the testing code is not included in the total word counts but the description is included. So, put the testing code inside a table with a caption in order to make it an easy to recognise.
- You must include the progression of your work. It is about how effectively you evolve your work considering agility, milestones, etc.
- You must include your current status and future work. The current status and future work are about what you have completed based on the functional, non-functional and user interface requirements, and what you need to improve in future.
- You must include a short conclusion.
- You can add more appendixes if necessary. Remember it is not included in the total word counts.
- This is only a proposed report format, but you can change the format if necessary.
    - Cover page
    - Table of Contents
    - Introduction
    - Overall description
    - Requirements
    - Design
    - Technical Detail
    - Test Design
    - Progression of Work

- Current Status and Future Work
- Conclusion
- Appendixes

[Implementation]

Here are the instructions of how you implement your software and how you submit your code:

- The implementation must be based on the requirements defined in your report.
- You should include comments in your code to provide better understanding and analysis of your code for readers.
- You should include a README file which includes how to setup and how to execute your software and additional information required for your software. There is no format restriction in the README file such as text only, markdown, json, xml, etc. As long as it is understandable by readers, it will be acceptable.
- However, if any problems do occur due to not providing the detail and clear explanations of setup and execution of your software in a README file, it is your responsibility for the problems.
- In a case that your software is available online, the README file should include how to access your software.
- You should provide a zip file containing all source code but should not include dependencies. Instead, you should provide the instructions in the README file explaining how to install them if you need dependencies for your software.
- In a case that it is difficult to execute or access your software, you should include as many screen-captured images as possible in your report or provide the images separately with your code in a zip file.
- You must include some test codes in the zip file and also include some description regarding these tests in your report.
- It is important that your code has to be well-arranged. If it is necessary, please consider refactoring your code before submitting it.

4. Submission

The individual report should follow the constraints below:

Submission requirements

- You must submit a project report **indicating the Student ID number** in the title of the submission, i.e., **0000001_Report.pdf**.
- The report must be in **PDF** format.
- The report must be submitted via Tabula and must not be Zipped.
- You must zip all the codes used for your implementation and READMME file, and submit the zipped file **indicating the Student ID number** in the title of the submission, i.e., **0000001_Code.zip**.
- You **must** check if the report and the zipped file have been uploaded successfully.
- You must include the assessment front sheet in your report.

Report Requirements

- The report should be 2500 words ± 10%.
- The report should include a title page, table of contents in the report.

- The report should include your student ID.
- There is no page limit as long as it fits the total number of words for the report.
- The report should follow a logical and well-defined structure with headings and subheadings.
- Diagrams in the report should be clearly labelled and well-presented.
- Appendices will not normally be marked but they must not include material essential to the argument developed in the main body of the work.

Late submission policy

If work is submitted late, penalties will be applied at the rate of **5 marks per University working day** after the due date, up to a **maximum of 10 working days** late. After this period the mark for the work will be reduced to 0 (which is the maximum penalty). "Late" means **after the submission deadline time as well as the date** – work submitted after the given time even on the same day is counted as 1 day late.

Resubmission policy

If you fail this assignment or module, please be aware that the University allows students to remedy such failure (within certain limits). Decisions to authorise such resubmissions are made by Exam Boards. Normally these will be issued at specific times of the year, depending on your programme of study. More information can be found from your programme office if you are concerned.

5. Grading

The learning outcomes measured by this assignment are the following, which essentially covers the topics of the last two blocks of teaching:

- Apply the range of software tools used tools for configuration management, version control and software build.
- Discriminate the key concepts and techniques used in the Agile Manifesto and Scrum, carefully design and critically evaluate project plans using these techniques.
- Distinguish current and emerging XP, Lean, and Kanban values, principles, and practices, use these techniques to analyse and optimize process workflow.

The following rubric will be applied to your submission.

	0 – 40%	41 – 60%	61 – 80%	81 – 100%
Report (40%)	Little evidence of producing an individual report required in the assignment.	Evidence of producing an individual report required in the assignment. However, some contents in the report are not enough to provide information required in the assignment.	Good report covering the most of contents required in the assignment. Also, the contents in the report are well presented and provides the right information to understand the implementation.	Excellent report fully covering the contents required in the assignment. Also, the contents in the report are well presented, and provides not only the right and correct information to understand the implementation but also clear evidence to understand software development life cycle such as key concepts and principles.
Software Build (40%)	Little evidence of the implementation based on the functional and non-functional requirements. Also, the implementation is not based on the report.	Evidence of the implementation based on the functional and non-functional requirements. However, the implementation does not fully support some important features described in the report.	Good implementation based on the functional and non-functional requirements. Also, the user interface is well designed and implemented based on the report. The implementation supports some important features described in the report.	Excellent implementation based on the functional and non-functional requirements. Also, the user interface is well designed and implemented based on the report. The implementation fully supports the most important features and user interfaces described in the report.
Integration (10%)	Little evidence of specifying the integration with the main features such as the board management and privilege management.	Evidence of specifying the integration with the main features such as board management and privilege management. However, the integration is not well justified and does not cover the full integration with the main features.	Good specification of the integration with the main features such as board management and privilege management. The integration is working very well with the main features and design architecture.	Excellent specification of the integration with the main features such as board management and privilege management. The integration is working very well with the main features and design architecture. There is strong evidence for integration with other boards such as the integrated user interface.
Testing (10%)	Little evidence of providing testing codes.	Evidence of providing testing codes for some functions. However, it is not very well justified.	Good testing codes with the descriptions for some selected functions. Also, it provides the reasons of why these functions are selected to provide test cases in the report.	Excellent testing codes with the well-defined descriptions for some selected functions or many functions. It provides the reasons of why these functions are selected to provide test cases in the report. The testing codes fully covers the most testing cases for the codes.