# **MILESTONE 2** -- SFT221 Scrum Report and Reflection

All students are expected to attend the SCRUM meetings and to participate. Failure to do so will result in greatly reduced grades.

**GROUP**: 2

**Members Present**:

|  |  |
| --- | --- |
| 1. Joon Dong | 4. Heqing Xu |
| 2. Doris Chai | 5. |
| 3. Xiaopeng Liu | 6. |

## Milestone 2 Tasks

Some of the software for the project has already been written for you and is available on Blackboard. You must use this in your project and every team should add it to the source code for their repository. Anything in the main function is simply for demonstration purposes and can be replaced. The software you are being given has not been tested and you will need to test it.

You need to study the problem and the code provided for you and then:

* Add any new data structures you will require This will require a thorough analysis of the problem and the existing software. This should be done by creating a new header file in the directory where the rest of the source code has been placed. You do not want to go back and modify it later if you can avoid it as it will slow the project.
* Create a test plan for the project by replacing the text in the supplied test plan template with your test plan.

**Deliverables due 4 days after your lab day:**

* An analysis of the problem (no written artifacts produced).
* A series of data structures created as header files and stored in the repository.
* A test plan stored in the repository.
* Completed scrum report including reflection questions answered.

**Rubric**

|  |  |  |
| --- | --- | --- |
| **Individual** | Group participation (includes GitHub commits and Jira usage) | 80% |
| Teamwork | 20% |
| **Group** | Data structures (complete, correct, and well-designed, & project updated) | 25% |
| Test plan (complete, well-written) | 25% |
| Git usage (used properly with good structure) | 10% |
| Jira usage (creates issues, tracks progress) | 20% |
| Scrum report & reflections | 20% |
| **Deadline** | 20% deduction for each day you are late |  |

**Scrum Report**

**Summary of Tasks Completed or Delayed in the last week:**

Here you can list all the tasks completed in the last week along with any tasks which could not be completed with a reason why they could not be completed.

|  |  |  |
| --- | --- | --- |
| **Member** | **Tasks Completed** | **Tasks Delayed/Blocked** |
| **Joon** | **Github repository initialized** |  |
| **Doris** | **Jira initialized** |  |
| **Xiaopeng** | **MS1 Reflection Q1** |  |
| **Heqing** | **MS1 Reflection Q2** |  |
| **Doris** | **MS1 Reflection Q3** |  |
| **All** | **MS1 Scrum Report** |  |
|  |  |  |

For every task delayed or blocked, describe the reason for the delay or block, how it impacts the project and the proposed solution or workaround.

|  |  |
| --- | --- |
| **Delayed or Blocked Task** |  |
| **Reason for delay or block** |  |
| **Impact on Project** |  |
| **Solution or work-around** |  |
|  |  |
| **Delayed or Blocked Task** |  |
| **Reason for delay or block** |  |
| **Impact on Project** |  |
| **Solution or work-around** |  |

**Summary of Meeting:**

A summary of the main points discussed in the meeting and the outcomes of the discussions.

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Tasking | **How to evenly split tasks** | **Tasks delegated fairly** |
| Meeting up | **If another day to meet is required** | **Will meet virtually** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Summary of Decisions Made:**

This will include major architecture and design decisions, testing decisions, prioritization of tasks, dealing with problems encountered and other major outcomes from the meeting.

|  |  |
| --- | --- |
| Decision | Rationale |
| Members will work individually on test plan parts | It would be counterintuitive to huddle together and allocate time to only the test plan. It needs to be split to increase efficiency and increase time for other tasks required. |
| Members will take home source code, analyze, and design own functions | Members can focus and analyze the code at their own pace rather than potentially being lost and unable to contribute in-person. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Tasks Attempted During Meeting:**

Each member is assumed to participate in the SCRUM meeting and contribute to the completion of the SCRUM report and reflections. Since the SCRUM meeting will not take more than 20-30 minutes, there is lots of time left to undertake some of the actual work tasks. In the table below, each member should list what they did to complete the SCRUM report, the reflections, and 1-4 other tasks they completed during the class period. If a task cannot be completed, the student should indicate why this was not possible.

|  |  |  |  |
| --- | --- | --- | --- |
| Member | Task Attempted | Time Spent | Complete? |
| Joon | **Test plan question 5** | **30 m** | **No** |
| Doris | **Source code analysis** | **30 m** | **No** |
| All | **Scrum report** | **30m** | **Yes** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**SCRUM Tasks Selected for Next Week**:

The tasks each member has selected to pursue for this class or the next week.

|  |  |
| --- | --- |
| Group Member | Task Description |
| Joon | Complete test plan question 5 |
| Joon | Contribute function designs |
| Joon | Reflection question 1a |
| Doris | Complete test plan question 1,2,4,6,7,14,15,16,17 |
| Doris | Contribute function designs |
| Doris | Reflection question 3 |
| Xiaopeng | Complete test plan question 3 |
| Xiaopeng | Contribute function designs |
| Xiapeng | Reflection question 1b |
| Heqing | Complete test plan question 8-13 |
| Heqing | Contribute function designs |
| Heqing | Reflection question 2 |

**Major Outcomes of Meeting:**

This is where you should highlight the major accomplishments of the class.

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Successfully delegated tasks | **Able to hold each other accountable to assigned tasks** |
| Clarity on task requirements | **Able to effectively implement what is required of milestone 2** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Things That Went Well in This Meeting:**

Here you can highlight things which worked well. This indicates that the way you worked on these items is working and should be continued.

|  |  |
| --- | --- |
| Topic/Work Item | Reason for Success |
|  |  |
| Task delegation | **Good communication and willingness to take on tasks** |
| Clarity on task requirements | **Professor answered questions clearly, reiterating what is required of milestone 2** |
|  |  |
|  |  |
|  |  |
|  |  |

**Things That Did NOT Go Well in This Meeting:**

This is where you can list things which did not go well in the class. You should analyze why this happened and suggest how you can improve it next time. This will lead to the goal of *continuous process improvement*.

|  |  |
| --- | --- |
| Topic/Work Item | Reason for Problem and How to do Better |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflection Questions:**

Answer the following questions using your own words. Make sure that each answer comprises a minimum of 100 words.

1. **In this milestone you have been asked to analyze a problem and design software (functions) to complete the solution without writing the software.**
   1. **Is this process more difficult than just writing the software to complete the project? If so, why is it more difficult? If not, why is it easier than just writing the software?**  
        
      The process of designing functions before writing software is easier. This is what we call the planning stage. By effectively analyzing what the project requirements are, you can get a clearer idea about what needs to be implemented. Rather than just writing the software right away, you can effectively analyze the function itself. This would minimize the need to backtrack, for example, if you were to implement a function but find that it doesn’t follow the project requirements. By designing software before coding, you can be more confident that you are following project specifications because it is your main focus rather than focusing on syntactical writing processes.
   2. **Describe two advantages of developing software in this manner rather than just moving on to writing the functions without writing specifications first.**

First, this approach can improve the quality and completion of the project because by writing the specification first, the developer can clarify how the software should behave. This helps development team members reduce biases and errors in understanding since the development work is centered around behaviors that have been made clear and agreed upon. Secondly, this method can enhance efficiency and increase the project success rate. By gradually improving the program modules, timely feedback can be obtained. This includes the modification and improvement of requirements, allowing for the construction of the project's final form.

1. **Why is it a good idea to create a test plan? Describe at least 3 advantages of test plans.**  
     
   Creating a test plan is essential for ensuring the successful execution of testing activities. Test plans are essentially guidebooks that outline the approach, objective, scope, resources, and scheduling for testing efforts. Here are three advantages of having a test plan:

1. Clear direction:

It can let us know exactly the specific test objective and test scope. After making a test plan, it also provides clarity and instruction for testing teams, ensuring everyone can focus on testing activities.

1. More efficient testing:

A well-defined test plan helps improve the efficiency of testing. The test plan helps testers focus their efforts on critical areas of the software and testing tasks. This ensures that bugs are identified early, and potential risks are mitigated proactively ensuring no need for backtracking.

1. Communication and Collaboration:

We can arrange the work content of each member more reasonably and use more effective tools to communicate. It also gives us more time to communicate and make some decisions.

1. **Describe the process you used to analyze and understand the existing software.**

First, I carefully read through the specification and requirements for the software again to understand the problem we need to address and the expected behavior of the software. Following that, I went through the header file line by line to understand the purpose of the functions and structs, together with the implementation file to have a rough idea of the functionality. The next step is to run the code to try to analyze and understand the output. I also used the VS debugger to step through the code to help me trace the execution step-by-step and establish a better understanding of the relationships between functions. This process also helped us to discover missing logic and implementation of the final software.