# **MILESTONE 2** -- SFT221 SCRUM Report and Reflections

This report should be completed in the class and submitted at the end of class. Late submissions cannot be accepted without prior approval of the instructor.

**GROUP**: \_\_\_\_\_\_\_5\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Siripa Purinruk | 5. Kishan Dewasi |
| 2. Bussarin Apichitchon | 6. Dhrumit Ketan Parekh |
| 3. Seyed Iman Modarres Sadeghi | 7. Jaskaran Singh |
| 4. Farbod Maoyari | 8. Varshilkumar Ileshkumar Parikh (added member) |

## 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 at End of Lab**

* Completed SCRUM report & reflections

**Deliverables Due within 48 hours of lab**

* 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.

**Rubric**

|  |  |  |
| --- | --- | --- |
| Individual | Group Participation | 75% |
| Teamwork | 10% |
| SCRUM Report | 15% |
| Group | Data structures (complete, correct and well-designed) | 20% |
| Test Plan (complete, well-written) | 20% |
| Git Usage (used properly with good structure) | 10% |
| Jira Usage (creates issues, tracks progress) | 10% |
| Meets Deadlines | 15% |
| SCRUM Report and Reflections | 25% |

**SCRUM Report**

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

Here you can list all of 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** |
| Siripa Purinruk | Create GIT | - |
| Bussarin Apichitchon | Create Jira project | - |
| Seyed Iman Modarres Sadeghi | Reflection Question 1 | - |
| Kishan Dewasi | Reflection Question 2 | - |
| Dhrumit Ketan Parekh | Reflection Question 3 | - |
| Jaskaran Singh | Group Contract | - |
| Farbod Maoyari | - | - |
| Parikh Varshilkumar Ileshkumar | - | - |

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 discusses in the meeting and the outcomes of the discussions.

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Assigning the tasks | Siripa Purinruk: responsible for the scrum report + test plan |  |
|  | Bussarin Apichitchon: responsible for the scrum report + test plan |  |
|  | Seyed Iman Modarres Sadeghi– responsible for the software designing |  |
|  | Dewasi Kishan – responsible for analyzing the code |  |
|  | Jaskaran Singh – responsible for analyzing the code |  |
|  | Dhrumit Ketan Parekh – responsible for analyzing the code |  |
|  | Farbod Maoyari – responsible for analyzing the code |  |
|  | Parikh Varshilkumar Ileshkumar - responsible for analyzing the code + summarizing the instructions |  |

**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 |
| Prioritization of the task:  -Scrum reports  -Analyzing the program  -Software Designing  -Test Plans | - to thoroughly understand the program and its requirements.  - to plan carefully in order to complete the project without ambiguity. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 could not be completed, the student should indicate why this was not possible.

|  |  |  |  |
| --- | --- | --- | --- |
| Member | Task Attempted | Time Spent | Complete? |
| Siripa Purinruk | Scrum Reports (Taking the summary of the discussion + Reflection question 1) | 30 mins | **yes** |
| Bussarin Apichitchon | Scrum Reports (Reflection question 2,3) | 30 mins | **yes** |
| Seyed Iman Modarres Sadeghi | Participate in the meetings + work assigned | 30 mins | **yes** |
| Dewasi Kishan | Participate in the meetings + work assigned | 30 mins | **yes** |
| Jaskaran Singh | Participate in the meetings + work assigned | 30 mins | **yes** |
| Dhrumit Ketan Parekh | Participate in the meetings + work assigned | 30 mins | **yes** |
| Farbod Maoyari | Participate in the meetings + work assigned | 30 mins | **yes** |
| Parikh Varshilkumar Ileshkumar | work assigned | 30 mins | **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 |
| Group Manager | Will designate with the task in the next week. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Work assigned | Group members can prepare according to their roles. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Good Collaboration | **-** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| - | **-** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

1. In this milestone you have been asked to analyze a problem and design software(functions) to complete the solution without actually 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?

It is more challenging because it requires a deep understanding of the program and the ability to conceptualize it into a software solution. This process involves careful planning and visualizing the software structure, including the interactions between functions, which can take time and effort. Despite not focusing on implementation details, the task is still difficult due to the need for critical thinking and effectively managing the complexity of the software.

* 1. Describe two advantages of developing software in this manner rather than just moving on to writing the functions without writing specifications first.  
       
      The advantages of developing software by analyzing the problem and writing function specifications are:
     1. Clear understanding: It ensures a clearer understanding of the program and its requirements, reducing the risk of misinterpretation or divergence during implementation.
     2. Minimized ambiguity: Writing function specifications minimizes ambiguity during development, leading to more accurate and efficient implementation and fostering better collaboration among team members.

1. Why is it a good idea to create a test plan? Describe at least 3 advantages of test plans.  
     
    That is because a test plan is the foundation of every testing effort. It helps determine how the software will be checked, what will be tested, and who will perform it. Everyone can work together effectively by creating a clear test plan all team members can follow.

The advantage of a test plan

1. Road Map to Testing Process: A test plan serves as a comprehensive guidebook for the testing process, providing a clear roadmap and direction for the testing activities. It helps define the scope, objectives, and practices to be followed, ensuring that the testing efforts are aligned with project goals and customer requirements.
2. Means of Communication: The test plan serves as a means of communication and collaboration among team members, testers, and stakeholders. It allows for feedback, review, and necessary changes. The test plan can capture comments and responses, serving as a historical record of relevant conversations and decisions.
3. Requirements for Test Environment: The test plan outlines the hardware and software requirements to set up the test environment. It ensures the tools, equipment, and resources are available before testing starts which helps avoid delays or obstacles in setting up the test environment.
4. Describe the process you used to analyze and understand the existing software.

"If you fail to plan, you are planning to fail." – Benjamin Franklin.

Analyzing and understanding existing software is a crucial process for software engineers to ensure they have a clear grasp of the software's functionality, structure, and potential areas for improvement. This process involves several key steps that help unravel the intricacies of the software and lay the foundation for effective maintenance and enhancement.

The first step is to obtain access to the software and gather any available documentation. Document review provides insights into the software's intended purpose, features, and usage instructions. It acts as a guide, helping engineers understand the software's context and objectives.

Next, engineers evaluate the software architecture. They analyze the structure, identifying the modules, layers, and components. Understanding the architecture helps comprehend how the different parts of the software interact with each other and contribute to its overall functionality.

Engineers examine the source code to delve deeper into the software's implementation. They review the codebase, focusing on coding conventions, patterns, and organization. This analysis allows them to identify areas requiring refactoring, complexity reduction, or performance optimization.

Dependencies and external integrations are also evaluated. Engineers identify the external libraries, frameworks, and APIs that the software relies on. They assess compatibility, stability, and security considerations associated with these dependencies, ensuring a thorough understanding of the software's ecosystem.