# **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**: \_\_\_\_\_\_\_\_\_7\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Janipan Sivaguru | 4. |
| 2. | 5. |
| 3. | 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) | 20% |
| Test plan (complete, well-written) | 20% |
| Git usage (used properly with good structure) | 10% |
| Jira usage (creates issues, tracks progress) | 15% |
| Scrum report & reflections | 25% |
| Meets deadlines | 10% |

**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** |
| **Janipan** | **Github Contribution** |  |
| **Janipan** | **Scrum Report** |  |
| **Janipan** | **Reflection** |  |
| **Janipan** | **Jira** |  |
| **Janipan** | **Creating test plans** |  |
|  |  |  |
|  |  |  |

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

**Summary of Meeting:**

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

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Create new structs | **Create data structs to project** | **Created** |
| Testing, updates github, jira | **Updating and correcting schedules** | **completed** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**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 |
| Breaking down tasks | Splitting the amount of work needed to be completed to each member of the team. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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? |
| Janipan | **Every task.** | **2hrs** | **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 |
| Janipan | Github updating |
| Janipan | Jira updates |
| Janipan | Scrum report |
| Janipan | Reflection |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| N/A no meeting |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| N/A no meeting |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| N/A |  |
| N/A |  |
| N/A |  |
| N/A |  |
|  |  |
|  |  |
|  |  |

**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?  
        
      I thought it was more difficult to analyze an issue and design the software without writing the code than to write the software itself. The complexities of the problem, potential edge cases, and effective algorithmic solutions are all carefully considered during this process. It requires a thorough comprehension of the problem domain to guarantee that the functions that are designed fit in perfectly with the goals of the project. The challenge stems from the requirement for thorough planning and foresight to avert possible problems during the implementation stage. This process may be more difficult than quick coding since it necessitates a thorough understanding of the issue and the capacity to anticipate difficulties.
   2. Describe two advantages of developing software in this manner rather than just moving on to writing the functions without writing specifications first.  
      * One of the major advantages of developing software in this way is analyzing the problem and designing solutions beforehand, errors and bugs can be solved prior to beta testing, fewer bugs, fewer clean ups to make. This proactive approach assists with creating more robust and reliable software saving developers time and effort that would be spent on debugging and wasting an immense amount of time in the testing phase.
      * Another advantage following this method is the analysis, design and reviewing phases that allow developing members to better communicate, have a clear understanding of the problems and tasks while also aligning their efforts more effectively with other team members.
2. Why is it a good idea to create a test plan? Describe at least 3 advantages of test plans.  
   * + A test plan aids in the early detection of problems and defects in the software during development. Developers can save time and resources by thoroughly testing every component and functionality to identify and fix problems before they become more serious.
     + A well-defined test plan establishes explicit success criteria, thereby guaranteeing the software's quality. It contributes to ensuring software satisfies requirements and operates as intended, thereby boosting confidence in its dependability.
     + Test plans are documents that specify how the software is supposed to operate. When it comes to future maintenance and updates, this documentation is invaluable as it helps developers understand expected behavior and makes sure that changes don't cause new problems.

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

Code Review: Examine the current software's logic, structure, and coding standards in detail by conducting a thorough code review. This entails dissecting every class, module, and function to learn more about the overall structure.

Documentation Examination: Go over any documentation that is available, including external documentation files or comments found in the code. This can give more background information about the functions and relationships between different parts.

Testing and Execution: Run the current software under various conditions to see how it behaves. This aids in comprehending how the code behaves in various situations and in response to different inputs. Testing also helps to find possible problems or areas that need work.