# **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**: 6-NCC

**Members Present**:

|  |  |
| --- | --- |
| 1. Shaheer | 4. Maharshi |
| 2. Pujan | 5. |
| 3. Jeelkumar | 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 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** |
| Shaheer | **SCRUM REPORT.DOCX** | **Delayed** |
| Maharshi | **TEST\_PLAN.MD/DOCX** | **Delayed** |
| Pujan | **DATA STRUCTURES** | **Delayed** |
| Jeelkumar | **GITHUB COMMITS** | **Delayed** |
| Shaheer | **JIRA ACTION PLAN** | **Delayed** |
|  |  |  |
|  |  |  |

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** | **SCRUM REPORT + DATA STRUCTURES** |
| **Reason for delay or block** | **The other group member deleted everything** |
| **Impact on Project** | **Missed due dates** |
| **Solution or work-around** | **Tried to bring everything together as a group** |
|  |  |
| **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 |
| Milestone 2 | **What is due and when** | **The team seemed to not lose hope and work it out.** |

**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 |
| NA | NA |

**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? |
| Shaheer | **Test Plan** | **6 hours** | **Yes** |
| Jeel | **Scrum Report** | **1.5 hours** | **Yes** |
| Maharshi | **Data Structures** | **2 hours** | **Yes** |
| Pujan | **Committed changes to GitHub** | **20 min** | **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 |
| Shaheer | Test Plan |
| Pujan | Commit changes |
| Maharshi | Coding problem |
| Jeel | Scrum |

**Major Outcomes of the Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| The team said that they will assist with the project | **Things looking to get streamlined.** |

**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 |
| Test Plan | **Took time to dive into understanding how to create the plan using information provided online and from SFT221 notes** |

**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 |
| Project assistance | **The management seems to be a bit of a problem cause of the team member leaving.** |

**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?

a. The difficulty of the process can depend on the complexity of the problem and the experience of team that works on it. However, in general, analyzing a problem and designing software solutions without writing the software can be seen a lot more challenging for several reasons:

* It requires a high level of abstraction: You need to understand the problem deeply and envision the solution without diving into the details of implementation. This can be very difficult, especially for complex problems.
* It requires good foresight and planning: You need to anticipate potential issues, edge cases, and performance considerations, and design your software to handle them.
* It requires clear and effective communication: If the design is to be implemented by someone else, or if it needs to be reviewed and approved, it must be clearly and thoroughly documented.  
    
  1. Describe two advantages of developing software in this manner rather than just moving on to writing the functions without writing specifications first.
  2. **Enhanced clarity and comprehension**: By focusing on design before implementation, you ensure a thorough understanding of the problem and the proposed solution. This can lead to better decision-making during the implementation/programming phase and can help dodge costly mistakes or rework.
  3. **Collaboration and division of work**: With a clear design and specifications, different parts of the software can be implemented by different developers or teams. This can help reach the deadlines and distribute the workload equally.
  4. **Easier maintenance and scalability**: A well-architected software is typically easier to maintain, debug, and scale. By thinking through the design upfront, you can plan for future needs and changes, making it easier to adapt the software later on.

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 beneficial for a variety of reasons:

1. **Clarity and Understanding**: A test plan helps in creating a clear understanding of the scope, approach, resources, and schedule of the testing activities. It helps in setting up the test objectives and strategies that will ensure the efficient use of resources and adherence to the project timeline.

2. **Risk Mitigation:** Test plans help identify potential risks and issues early in the process. By highlighting potential areas of concern, the team can work proactively to address these risks, thereby reducing the likelihood of unforeseen issues and roadblocks during the testing phase.

3. **Quality Assurance:** A well-structured test plan aids in ensuring the quality of the software product. It helps to verify that the product meets the specified requirements and helps uncover any defects or discrepancies before the product reaches the end-user.

1. Describe the process you used to analyze and understand the existing software.
2. **Review the Documentation**: I started by reading any available documentation, comments in the code, and specifications. This gave me a high-level understanding of what the software is supposed to do.
3. **Examine the Code Structure**: I then looked at the overall structure of the code, including the organization of files and directories, the naming conventions used, and the design patterns implemented. This helped me understand how the software is organized and how different components interact with each other.
4. **Deep Dive into Individual Components**: I went through the code in detail, focusing on one component or function at a time. I looked at the inputs and outputs of each function, the algorithms used, and how data is being manipulated. This helped me understand the logic of the software and how it achieves its goals.
5. **Run the Software**: I ran the software to see it in action. This allowed me to see the end result of the code and helped me understand the user experience. It also allowed me to identify any potential issues or bugs in the software.