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

**GROUP**: 5

**Members Present**:

|  |  |
| --- | --- |
| 1. Audrey Duzon | 4. Ka Ying Chan |
| 2. Tae Yong Eom | 5.Julia Alekseev |
| 3.Azad Zeynalov | 6. |

## Milestone 4 Tasks

**Deliverables Due at end of Lab:**

* Completed SCRUM report and reflections

**Deliverables Due at 23:59 6 Days after Lab:**

* Implemented Functions
* Implemented blackbox tests (store in repo), executed (results in Jira and on corresponding test documents) and debugged,
* whitebox tests written and stored in repository.
* whitebox tests implemented (store in repo), executed (results in Jira and on corresponding test documents) and debugged.
* Updated function-test matrix stored in the repository.
* Completed hook for test automation

**Rubric**

|  |  |  |
| --- | --- | --- |
| Individual | Group Participation | 75% |
| Teamwork | 5% |
| SCRUM Report | 10% |
| Automation Hook | 10% |
| Group | Implemented Functions (well-designed, written and documented) | 20% |
| Whitebox tests (well-designed, written and documented) | 20% |
| Test Execution (performed, results recorded, issues created) | 20% |
| Debugging (Bugs fixed, documented, Jira updated) | 5% |
| Git Usage (used properly with good structure) | 5% |
| Jira Usage (creates issues, tracks progress) | 5% |
| Meets Deadlines | 5% |
| SCRUM Report and Reflections | 20% |

**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** |
| **Julia** | **Scrum report and reflection** |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

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 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**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 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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? |
| Julia | **Scrum report and reflection** | **1.5h** | **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 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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. After you run your blackbox and whitebox tests you are asked to record the results in both the original test document as well as in Jira. Explain why it is a good idea to record the results in both places.  
     
   Recording the test results in both the original test document and Jira is crucial for comprehensive documentation and traceability. The test document captures detailed information about test scenarios and outcomes, while Jira provides a structured platform for linking tests to specific tickets, facilitating better communication between team members, developers, and other relevant parties. This dual recording approach enhances transparency, collaboration, and ensures a clear record of the testing process for future reference.
2. Why did we wait until the fourth milestone to write the whitebox tests?

We waited until the fourth milestone to write the whitebox tests because whitebox testing involves analyzing the internal structure and logic of the software, which requires access to the source code. By waiting until the programming team completes the code implementation for the specific features or modules, the testing team can then develop potential black box test cases and perform them in the future thorough whitebox testing. This sequential approach ensures that the code is ready for examination, simulates a real-life development process, and allows for effective verification of the software's internal behavior and functionality.

1. For a given function did you produce more blackbox or whitebox tests? Explain why your answer (more blackbox or more whitebox) happens for most functions.  
     
   We would produce more blackbox tests because blackbox testing focuses on evaluating the function's external behavior without considering its internal code implementation. Testers design these tests based on specifications and requirements, ensuring the function behaves correctly from an end-user perspective. Since blackbox tests do not require knowledge of the internal structure, they are generally easier to create and can cover a wide range of scenarios. On the other hand, whitebox testing, which involves examining the internal code and logic, can be more time-consuming and requires a deep understanding of the code to create relevant test cases. While whitebox tests are crucial for achieving thorough code coverage and identifying specific implementation issues, they might be fewer in number due to the complexities involved in their creation. As a result, for most functions, we prioritize producing more blackbox tests for comprehensive and diverse test coverage.
2. Explain the purpose of the automation hook for GIT and explain how it can improve the quality of the software in the project.

The GIT automation hook plays a crucial role in automating various tasks within the version control workflow. Its primary goal is to enforce coding standards, design guidelines, and best practices consistently across the entire project. It achieves this by automating code quality checks, including tools like "fluff," as well as unit testing. These automated checks help detect potential issues early in the development process, thereby minimizing the chances of bugs and errors. As a result, the overall software quality is significantly enhanced, thanks to the promotion of uniform code styles, improved maintainability, and the integration of only high-quality code into the project's code base.