# 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\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Shaheer | 4. Maharshi |
| 2. Pujan |  |
| 3. Jeelkumar |  |

## Milestone 5 Tasks

In this milestone, you should write, implement, and execute integration tests. Integration tests test how multiple functions work together to complete a task. Depending on what is being tested, you might be able to write unit tests to do the testing and automatically compare the results. In other cases, you might need to manually check the output to check it. This will all be stated in the tests where it discusses how they should be run.

As you update the function-test matrix, you will need to add a very brief description for each integration test so the matrix will clearly show what the tests are testing. Acceptance tests will be tested against actual user requirements and will list all the tests for each requirement.

Acceptance tests are the final tests and are largely aimed at showing the customer that the correct output is produced for different inputs. This will largely require manual testing.

**Deliverables Due at end of Lab:**

* Completed SCRUM report and reflections

**Deliverables Due at 23:59 12 Days after Lab:**

* integration tests written and stored in repository,
* integration tests written (store in repo), executed (results in Jira and in test documents) and debugged.
* acceptance tests written and stored in repository.
* Updated function-integration-requirements-test matrix stored to the repository.

**Rubric**

|  |  |  |
| --- | --- | --- |
| Individual | Group Participation | 75% |
| Teamwork | 10% |
| SCRUM Report and reflections | 15% |
| Group | integration tests (well-designed, written and documented) | 20% |
| acceptance tests (well-designed, written and documented) | 20% |
| Test Execution (performed, results recorded, issues created) | 15% |
| Debugging (Bugs fixed, documented, Jira updated) | 5% |
| Function-test matrix 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** |
| Shaheer Ali Khan | Scrum report + test automation + white box testing + function implementation | **-** |
| Jeel Kumar | Whitebox testing + reflection question 1 | **-** |
| Pujan Shah | Function implementations | **-** |
| Maharshi Patel | Reflection question | **-** |

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** | **Whitebox testing** |
| **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 | Shaheer Ali Khan: responsible for scrum report+ traceability matrix. | **-** |
|  | Maharshi Patel: responsible for traceability matrix. | **-** |
|  | Jeel Kumar: responsible for integration testing. | **-** |
|  | Pujan Shah: responsible for adding some code. | **-** |

**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 |
| Assigning work | * Integration tests * Acceptance tests * Updated function-integration-requirements-test matrix |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 Ali Khan | Discussion + assigning work | 30 mins | **yes** |
| Pujan Shah | work assigned | 30 mins | **yes** |
| Maharshi Patel | work assigned | 30 mins | **yes** |
| Jeel Kumar | 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 |
| Project 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 |
| - | **-** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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**:

**At this point, you are using the GIT hook to automate testing. Have you found that any of the tests failed and prevented you from pushing your code to the repository? If so, how did you handle the situation?**

As we integrated the GIT hook for automated testing, we have not encountered any test failures that prevented us from pushing code to the repository. Our team has been proactive in resolving test case issues before attempting to push the code. In the event of test failures during the automated testing process, we promptly address the identified bugs, retest the code, and ensure it meets the required standards before attempting to push it again.

**Explain why we are automating the testing process and what the advantages of this automation are.**

We are automating the testing process using GIT hooks to execute tests and verify the software's functionality automatically. The advantages of this automation are numerous. Firstly, it significantly saves time and increases overall efficiency by enabling faster test execution. Secondly, automation ensures consistency in test execution, reducing the likelihood of human errors that can occur during manual testing. Thirdly, automated testing allows for broad test coverage, including edge cases and negative scenarios that may be challenging to cover manually. Additionally, automation facilitates effective regression testing, enabling us to quickly identify and address any new issues that might arise after code changes. Overall, automating the testing process with GIT hooks is highly cost-effective in the long run, saving time and effort during the testing phase and contributing to a higher quality software product.

**Did you find the integration and acceptance tests more difficult to write than the black box and white box tests? If so, why were they harder to write? Did you write more white box and black box tests or more integration and acceptance tests?**

In my experience, I found writing black box and white box tests more challenging compared to integration and acceptance tests. The difficulty with black box and white box tests lies in the need to cover all potential user inputs and ensure the software can handle unexpected behaviors effectively. However, once we thoroughly executed unit tests (black box and white box), writing integration and acceptance tests became easier as they validate the interactions between different functions and ensure the software meets the specified requirements.

Regarding the number of tests written, we focused more on white box and black box tests to ensure that the individual functions work correctly. While we wrote fewer integration and acceptance tests, they are equally important as they validate the software's overall functionality and user experience.

**Explain why it is necessary to write integration and acceptance tests given that all of the code has already passed black box and white box tests.**

Integration tests are necessary to validate that different functions and components interact cohesively as an integrated system. Even though individual functions may pass black box and white box tests, issues can arise when they interact with each other. Integration tests provide essential confidence that the software functions correctly as a whole.

Similarly, acceptance tests are crucial as they assess the software from an end-user perspective. Despite passing black box and white box tests, acceptance tests ensure that the software meets the specified requirements and fulfills the needs of its intended users. They focus on validating whether the software's features and functionalities align with the overall project goals.

In conclusion, both integration and acceptance tests are necessary to ensure the software behaves as expected and meets the project's objectives, even after passing black box and white box tests.