# 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**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Group 4\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Jo Eric - 137057188 | 4. Peralta Joe - 114751209 |
| 2. Nguyen Huu Linh - 118197227 | 5. Dominguez Daniel - 110835188 |
| 3. Jang Hyeri - 115328221 | 6. Sadat Morsal - 102693215 |

## 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** |
| **Eric Jo** | * Tasks Assignment * Made more unit test code * Test Implementation * Helping executed (results in Jira) and debugged. * Helping Updated function-integration-requirements-test. | **There are no tasks delayed or blocked** |
| **Joe Peralta** | * Updated function-integration-requirements-test. * Debugged. * Helping integration tests written (store in repo), executed (results in Jira) * Created and implemented functions. * Helped in answering questions in scrum report. | **There are no tasks delayed or blocked** |
| **Nguyen Huu Linh** | * Filling scrum Report. * Hepling the test data for blackbox testing and white box documented them * Helping Integration tests written. * Helping to implement functions. | **There are no tasks delayed or blocked** |
| **Daniel Dominguez** | * Filling scrum Report. * Acceptance tests written and stored in repository. * Helped in answering questions in scrum report. * Updated function-integration-requirements-test. | **There are no tasks delayed or blocked** |
| **Hyeri Jang** | * Filling scrum report * Created the test data for blackbox testing and white box documented them * integration tests written and stored in repository * Helping unti test code | **There are no tasks delayed or blocked** |
|  |  |  |
|  |  |  |

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** | **There are no tasks delayed or blocked** |
| **Reason for delay or block** | **There are no tasks delayed or blocked** |
| **Impact on Project** | **There are no tasks delayed or blocked** |
| **Solution or work-around** | **There are no tasks delayed or blocked** |
|  |  |
| **Delayed or Blocked Task** | **There are no tasks delayed or blocked** |
| **Reason for delay or block** | **There are no tasks delayed or blocked** |
| **Impact on Project** | **There are no tasks delayed or blocked** |
| **Solution or work-around** | **There are no tasks delayed or blocked** |

**Summary of Meeting:**

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

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Jira bug reports | **Members who work on tests implementation will write the bug reports on Jira** | **We made the task more organized.** |
| Understanding of assigned task | **Each member must understand the requirements needed to successfully complete their assigned tasks.** | **Each member has reviewed the necessary requirements to complete their assigned task.** |
| Allocation of tasks | **Each member has been asked to meet and discuss the deliverables or tasks needed to complete the milestone 5** | **Each member participated and collaborated in allocating and determining the tasks needed to complete milestone 5.** |
| GitHub and Jira | **discuss and work more on github and jira. Improve working on github as well as jira. The work is clearly recorded in more detail** | **We work more clearly shows better efficiency** |
|  |  |  |
|  |  |  |
|  |  |  |

**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 |
| Integration Tests written, and executed | Integration tests ensure that different modules or services work together as expected. These tests help identify issues related to the interactions between different parts of the system, improving overall system reliability |
| Acceptance Tests written and stored in repository | Acceptance tests validate that the system meets business requirements. Storing them in a repository ensures versioning and collaboration, allowing team members to access and update the tests as needed. |
| Updated function-integration-requirements-test matrix stored in the repository | It helps in understanding how different components are connected and ensures that changes in one area can be tracked to their impact on others. |
| Debugging, implementing function | Concentrating on these tasks ensures that the code is functional and error-free, leading to a more robust product. |
|  |  |
|  |  |
|  |  |

**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? |
| Eric Jo | **Assigned Tasks to each member** **Decision making**  **Summary of meeting** | **12m** | **yes** |
| Nguyen Huu Linh | **Filling Reflection 3 and 4**  **Filling “Things That Went Well in This Meeting:”**  **“Things That Did NOT go Well in This Meeting”** | **20m** | **yes** |
| Hyeri Jang | **Completed reflection question #1**  **“Things That Did NOT go Well in This Meeting”**  **Filling Reflection 3** | **15m** | **yes** |
| Daniel Dominguez | **Completed reflection question #2, partially #4, overall scrum report refinement**  **“Things That Did NOT go Well in This Meeting”** | **18m** | **yes** |
| Joe Peralta | **Filling the scrum report questions**  **“Things That Went Well in This Meeting:”**  **Filling Reflection 3** | **20m** | **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 |
| Eric Jo | Execute acceptance tests (results in Jira),  Helping bugs fixed.  Checking JIRA, GitHub for last milestone. |
| Hyeri Jang | Final testing report listing tests conducted  Helping bugs fixed, and the final test passed. |
| Joe Peralta | bugs fixed, and the final test passed.  Helping Final testing report listing tests conducte. |
| Daniel Dominguez | Updated function-test matrix stored in the repository  Helping Final testing report listing tests conducted |
| Nguyen Huu Linh | Helping Checking JIRA, GitHub.  Helping Final testing report listing tests conducted. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| GitHub and Jira | **Through the results of MS3 we have tried to fix as well as improve the working environment on jira and github more. Everything becomes clearer after this improvement.** |
| All members have participated well | **Reduces the amount of time planning the requirements that needed to be done to successfully complete milestone 5.** |
|  |  |
| Integration test | **focus on debugging as well as integration tests as well as unit tests** |
|  |  |
|  |  |
|  |  |

**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 |
| Team Collaboration | **As always the atmosphere worked very well. Everyone is responsible for their assigned tasks** |
|  |  |
| Communication | **Everyone is responsible for the work they are doing even though the final exam is approaching. as well as help each other in work** |
| Understand the meaning of teamwork | **Debating and questioning makes work meaningful not only in a school setting but also helps to better understand future group work** |
|  |  |
|  |  |
|  |  |

**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 |
| Debate at work | **It can be said that sometimes we demand perfection in our work. So sometimes it feels like there are unnecessary worries. make work more than usual** |
| Meeting time | **Sometimes there are sudden meetings. There were times when we had to meet in the middle of the night to come up with a reasonable solution** |
| Testing and GitHub | **there are still actual tests that don't work as expected. as well as getting an error when pushing to github** |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

1. 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 use Git hooks to automate testing, we might encounter various challenges that prevent us from pushing our code to the repository. Some common scenarios we've faced include:

1. Code Conflicts: Collaborating with multiple developers on the same codebase can lead to conflicts. When we try to push our code, Git detects these conflicts and prevents the push until they are resolved. To resolve conflicts, we merge or rebase our changes with the latest code in the repository.

2. Code Style Issues: Our automated tests often include checks for code style and formatting. If the code doesn't adhere to the specified style guidelines, the tests can fail. To address this, we adjust the code to meet the required style standards.

3. Integration Issues: In projects with multiple components or services, integration tests ensure that different parts work together correctly. If an integration test fails, it might indicate problems in the communication between components or services. We investigate and address these issues before pushing the code.

In each of these situations, we diligently investigate the problem, fix the code accordingly, and rerun the tests locally. Once all tests pass, we proceed to push the code to the repository for review and integration into the main codebase. The use of automated testing through Git hooks is crucial for maintaining code quality and preventing the introduction of issues into our shared code repository.

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

1. Time Savings: Automated testing significantly reduces the time needed to execute test cases compared to manual testing. We can run automated tests repeatedly and consistently, swiftly identifying and resolving defects without repetitive manual work.

2. Higher Test Coverage: Automated testing empowers us to cover a larger number of test cases and scenarios that might be impractical with manual testing. We create extensive test suites covering various functionalities, edge cases, and regression testing to ensure comprehensive coverage.

3. Consistency and Reliability: Automated tests ensure consistent test execution, eliminating. This consistency enhances the reliability of our testing process.

4. Early Bug Detection: Continuous test execution during development enables us to catch and rectify bugs early.

5. Enhanced Collaboration: Automated tests provide a standardized way to communicate and share test cases among team members, promoting better collaboration and knowledge sharing.

6. Improved Software Quality: Overall, automation elevates software quality by ensuring consistent testing, faster feedback, and broader code coverage. This, in turn, results in a more robust and reliable software product.

In summary, automating the testing process grants us significant advantages, including time savings, faster feedback loops, higher test coverage, consistency, reliability.

1. 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?

We write more white box and black box tests compared to integration and acceptance tests. Writing integration and acceptance tests can be more challenging due to the need to test interactions between various components, set up an appropriate environment, and consider different user scenarios for comprehensive acceptance testing.

Integration tests focus on testing how components or services interact and exchange data, which can be complex. Setting up the environment and managing dependencies adds to the difficulty.

Acceptance tests evaluate whether the software meets stakeholder requirements. Testing the entire application requires careful consideration of various user scenarios and expected behavior.

In contrast, black box and white box tests focus on individual units or functionalities, making them more straightforward to write and maintain. Black box tests assess the system externally based on specifications. White box tests delve into internal logic for code correctness.

We prioritize writing more white box and black box tests to verify the correctness of individual components. However, we recognize the importance of integration and acceptance tests for validating the overall system behavior and aim to maintain a balance between all testing types for comprehensive coverage.

1. 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.

We find it necessary to write integration and acceptance tests even after passing black box and white box tests because each testing type serves a unique purpose:

1.Validation of System Behavior: Integration tests allow us to verify interactions between different components, ensuring the entire system functions as expected when all parts work together.

2.End-to-End Testing: Acceptance tests play a critical role in validating the software from an end-user perspective, covering real-world scenarios and ensuring it meets all desired requirements.

3.Real-World Scenarios: With integration and acceptance tests, we replicate real-world interactions, enabling us to validate the application's behavior under real conditions, including interactions with external dependencies and databases.

4.Risk Mitigation: By including integration and acceptance tests in our testing process, we add an extra layer of risk mitigation, ensuring that the system operates as expected as a whole, even if individual units pass black box and white box tests.

In summary, we prioritize writing integration and acceptance tests alongside black box and white box tests to ensure comprehensive testing and to deliver a robust, reliable, and high-quality software product. Each type of testing contributes to a better understanding of the software's behavior, allowing us to address potential issues and meet stakeholder expectations effectively.