# **MILESTONE 5** -- SFT221 SCRUM Report and Reflection

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

**GROUP**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_E\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Kemal Batu Turgut | 4. Tu Yin Hnit Aung |
| 2. Kusum Acharya | 5. Roy Bryan D. Franck |
| 3. Krish Sanjaybhai Patel | 6. Shovana Shrestha |

## 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 4 days after your lab day:**

* Integration tests document stored in repository with at least 4 sets of distinct test cases (each case must have at least 4 distinct test data).
* Integration tests coded (store in repo), executed (results in Jira and in test documents) and debugged.
* Finish implementing/coding whitebox tests. Store in repo, executed, results in Jira (and on corresponding test documents, and debugged.
* Acceptance tests written and stored in repository.
* Updated requirements traceability matrix stored to the repository.
* Completed scrum report including reflection questions answered.

**Rubric:**

|  |  |  |
| --- | --- | --- |
| **Individual** | Group participation (includes GitHub commits and Jira usage) | 80% |
| Teamwork | 20% |
| **Group** | Integration test case document (well written, complete, good test data) | 10% |
| Integration test code (well designed and documented) | 10% |
| Finish coding all functions and main (well-designed, written, and documented) | 10% |
| Finish coding blackbox and whitebox cases (well-designed, written, and documented) | 10% |
| Acceptance tests (well-designed, written and documented) | 5% |
| Requirements traceability matrix updated | 5% |
| Test execution (performed, results recorded, issues created) | 10% |
| Debugging (bugs fixed, documented, Jira updated) | 10% |
| Git usage (used properly with good structure) | 5% |
| Jira usage (creates issues, tracks progress) | 10% |
| Scrum report & reflections | 15% |
| **Deadline** | 20% deduction for each day you are late |  |

**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** |
| **Kemal Batu Turgut** | Integration test case documents |  |
| **Krish Sanjaybhai Patel** | Updated requirements traceability matrix & Bug fixing |  |
| **Roy Bryan D. Franck** | Integration test code |  |
| **Tu Yin Hnit Aung** | Reflections |  |
| **Kusum Acharya** | Acceptance testing |  |
| **Shovana Shrestha** | Scrum Tables |  |
|  |  |  |

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 discussed in the meeting and the outcomes of the discussions.

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Task Allocation | We equally divided the tasks so that each member can finish their work on time. | Clear communication between team members. |
| Quality Assurance | We discussed about the importance of quality assurance with integration and acceptance testing so that our software meets high standards, enhancing functionality and reliability. | Results in high quality product before release. |
| Deadlines | The team discussed the upcoming deadline to ensure timely delivery without compromising quality work, deciding collectively to complete all tasks by Thursday for mutual review and potential fixes, ensuring all documents are submission-ready by Friday. | Timely delivery with quality work. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**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 |
| Dividing the tasks | We divided each task among the members, allowing for effective resource use and ensuring that each team member has clear responsibilities, hence increasing productivity and accountability. |
| Changing the deadline | As a team, we decided that all the tasks should be completed by Thursday so our team members can review each other’s work and suggest if some documents need to be fixed. Also, it will ensure that all documents are ready for submission by Friday. |
| Prioritizing integration and acceptance testing | We need to focus mainly on Quality Assurance for this week's milestone, so these tasks must be completed thoroughly and with great attention to detail. |
|  |  |
|  |  |
|  |  |
|  |  |

**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? |
| Kemal Batu Turgut | Integration test case documents | **1 hour** | **YES** |
| Krish Sanjaybhai Patel | Updated requirements traceability matrix & Bug fixing | **1 hour** | **YES** |
| Roy Bryan D. Franck | Integration test code | **1 hour** | **YES** |
| Tu Yin Hnit Aung | Reflections | **1 hour** | **YES** |
| Kusum Acharya | Acceptance testing | **1 hour** | **YES** |
| Shovana Shrestha | Scrum Tables | **1 hour** | **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 |
| Kemal Batu Turgut | Integration test case documents |
| Krish Sanjaybhai Patel | Updated requirements traceability matrix & Bug fixing |
| Roy Bryan D. Franck | Integration test code |
| Tu Yin Hnit Aung | Reflections |
| Kusum Acharya | Acceptance testing |
| Shovana Shrestha | Scrum Tables |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Decision Making | The decision-making was more transparent, and the tasks that were most crucial to achieving this week's objective were given top priority. This enhanced flexibility and adaptability, with a primary focus on quality assurance. |
| Task Division | We successfully divided this week's tasks among all of the members. Prior to concluding the meeting, all members were given the opportunity to voice their opinions and choose the task they wanted to attempt. |
| Deadlines | We decided to finish all the tasks by Thursday ensuring timely review and submission of the 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 |
| Organized schedule | The meeting was held on time with all the members present, allowing for timely discussions and decisions. |
| Division of tasks among members | The division of tasks among team members guaranteed that responsibilities were clear and manageable, leading to productive growth and teamwork. |
| Clear communication & Collaboration | Open and transparent communication between team members created effective collaboration and idea sharing. |
| Problem-solving | Team members identified issues and helped each other find a solution. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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**:

Answer the following questions using your own words. Make sure that each answer comprises a minimum of 100 words.

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?  
     
   Our professor has told us that we do not need to use GIT hook for this project as he has not taught it yet. Hence, we are not utilizing this feature to carry out tests and push code to the repository.
2. Explain why we are automating the testing process and what the advantages of this automation are.

Since we are not using automated testing process for this project, I will just list the advantages of this automation type of testing. It offers a range of advantages when we are trying to develop software. Firstly, it helps us as developers find problems early by running tests whenever code changes happen so that the issues can be fixed as soon as possible. Another thing is that it keeps things consistent as the same tests are run every time and the code is checked for the same bugs over and over again. This minimizes the risks of human error and ensures a reliable testing process that does not have to be looked over. There is also a faster feedback loop, where we as programmers have immediate insights into the quality of the code, which enables us to be better and more efficient the next time. This type of testing also supports regression testing and stops old problems from coming back. Last but certainly not the least, software like git hook enhances collaboration in a team and makes it easier to create reliable and high-quality products. As for why programmers would choose for the software testing process to be automated, it is because it helps catch bugs and errors early in the development process which saves time and effort in the long run. Human errors are minimized and it is more efficient as all the necessary tests are run after every change to the code. It also speeds up the testing process, leaving more time for cr4eating high-quality code.

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 found that the integration and acceptance tests were more difficult to write than the black box and white box tests. It is more challenging because of these reasons that I will list. Integration tests require a deep understanding of how the different functions are supposed to interact with one another on a large scale and on different types of occasions. This can be very time consuming as it require intricate knowledge of detail of the situations that will come up. Acceptance tests on the other hand involve simulating real-world interactions with the user and the software. This also require a deep understanding of user requirements and behaviour to come up with the tests. Ensuring that these test cases cover all of the possible user behaviour is hard and can also be really time-consuming. Generally, developers can write more black box and white box tests cases than integration and acceptance tests since they focus on individual components and functionalities.
2. 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.

Even if the code has already passed black box and white box tests, it is still essential that we carry out integration and acceptance tests. This is because black box and white box tests focus on testing the individual components and functionalities of the software when it is isolated. However, integration and acceptance tests test the entire behavior of the entire system as a whole and its interaction with each other and its users. Integration tests are used to ensure that the different and individual components of the software work together correctly when integrated. This is needed because even though each function may pass their individual tests, it does not mean that they will function together in an accurate way. Acceptance tests validate whether the software meets the specific requirements of what the program needs to do and for the users. It ensures that the software behaves as desired from a user’s perspective. Hence, it differs from black box and white box tests as it is mainly used to evaluate software’s usability and user experience. In conclusion, all of these tests are required to ensure that the program meets the requirements and expectations of the users.