# **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**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_11\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Richa Koirala | 4.Akanksha |
| 2.Manjot Singh | 5. |
| 3.Ammar Gangat | 6. |

## 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 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** |
| **Akansha** | **Acceptance Testing** | **N/A** |
|  |  |  |
| **Ammar** | **Integration Testing** | **N/A** |
|  |  |  |
| **Richa, Manjot** | **Test Execution** | **N/A** |
|  |  |  |
| **ALL** | **SCRUM and Reflection** | **N/A** |

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** | **N/A** |
| **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 |
| Integration Testing | **Integration testing written and discussed** | **Completed** |
|  |  |  |
| Acceptance Testing | **Acceptance testing written and discussed** | **Completed** |
|  |  |  |
| Test Execution | **Discussed and executed test (black box and white box)** | **Completed** |
|  |  |  |
| Reflection | **Discussed reflection questions** | **Completed** |

**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 |
| Correction of function implementation | Functions made were corrected in order to make it work correctly |
| Whitebox testing | Discussed problems in whitebox testing |
| Blackbox test | Linked unit test properly as it was not working correctly previously |
| Integration testing | Discussed abou integration test |
|  |  |
|  |  |
|  |  |

**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? |
|  |  |  |  |
| Manjot | **Finish and correct Whitebox test cases and test document** | **1hr** | **Yes** |
| Richa | **Integration test plan document, finish and correct Blackbox test** | **1hr** | **Yes** |
|  |  |  |  |
| Akansha | **Acceptance testing** | **1hr** | **Yes** |
|  |  |  |  |
| Ammar | **update traceability matrix and store it to repository , integration testing coded** | **1hr** | **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 |
| Ammar | Jira control |
| ALL | Meeting |
| ALL | SCRUM, Reflection |
| ALL | Final test report |
| ALL | Test Execution () |
| ALL | Debugging |
| ALL | Test Matrix |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| N/A |  |
| Whitebox testing | **It was having problem earlier but resolved it by changing some function implementation** |
| Time clashes | **Everyone was having different schedule so was having time clashes** |
|  |  |
|  |  |
|  |  |
|  |  |

**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?  
   I've certainly come across scenarios where tests have failed by automating a Git hook, preventing smooth code push. When I find myself in these situations, my approach is to quickly identify the root causes of failures. This requires a careful review of test results and a thorough review of recent changes to the code base. After I have identified the problems, I work on fixing the problems in the code itself. This may require debugging the code, adjusting dependencies, or experimenting with improving the intended functionality. I then rigorously redo the tests locally to confirm the effectiveness of the fixes and ensure the stability of the codebase. If the verification is successful, I accept the changes and try to enter the code again. Importantly, I make sure the tests pass during the pre-push hook before I finish pushing to the repository. Additionally, I take the opportunity to take a deep look at failed tests and try to identify recurring patterns or areas for improvement in our testing processes. This ongoing analysis helps improve our practices and increase the reliability and robustness of our code base for future iterations.
2. Explain why we are automating the testing process and what the advantages of this automation are.

There are several benefits to automating the testing process.  
**Efficiency:** Automated tests can run swiftly and repeatedly without the need for manual input, providing rapid feedback on code modifications. This efficiency speeds up the development process by spotting problems early and allowing for quicker debugging.

**Precision:** Automated tests ensure consistency in carrying out predetermined steps, minimizing the chances of human mistakes in comparison to manual testing. They offer accurate and dependable outcomes, guaranteeing consistency in the execution of tests every time.

**Coverage:** Automated testing can test a variety of scenarios, including edge cases and error conditions, that may not be feasible to test through manual methods. This thorough coverage assists in recognizing possible problems throughout various sections of the codebase.

**Time saved:** Through automation of repetitive and time-consuming testing tasks, developers and QA teams can use their time more efficiently to concentrate on more intricate and innovative aspects of testing, like exploratory testing and refining test strategy.

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 white box and black box tests or more integration and acceptance tests?

Yes, I did find the integration and acceptance tests more difficult to write than the black box and white box tests for several reasons:

1. Integration tests often involves testing interactions between different components and this complexity makes the environment for testing harder.
2. Integration testing has a fully functional environment with all dependencies in place which are hard to manage and should be checked to see if they are set up correctly.
3. With a huge scope of integration tests there are more scenarios to consider and test.
4. Integration tests may involve external systems or services such as databases, APIS, or third-party services. Coordinating the dependencies and ensuring they are available is complex.
5. Also, it is time consuming and requires additional resources to prepare the environment for integration tests.
6. Explain why it is necessary to write integration and acceptance tests given that all the code has already passed black box and white box tests.

Blackbox and white box tests are important for checking individual software parts, but integration and acceptance test gives advantages that complement these stages.

Integration testing makes sure that all the pieces of the software work well together when they are put together. Sometimes it can be hard when different parts of the software interact with one another, so it is necessary to check to make sure they are talking properly.

Integration tests may involve external systems or services such as databases, APIS, or third-party services. Coordinating the dependencies and ensuring they are available is complex.

Acceptance tests help to validate if the code is executing properly or not.

Acceptance tests are carried out to verify that a software system satisfies the defined criteria and meets user needs. These evaluations confirm that the system meets predetermined acceptance standards in terms of functionality, user-friendliness, and performance, assuring stakeholders that the software meets the desired business objectives.