# **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**: \_\_\_\_\_\_\_\_\_\_\_B\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Jeet Patel | 4. Arthav Patel |
| 2. Sahil Khatri | 5. Jeetkumar Patel |
| 3. Samarth Shah | 6. Yash Shah |

## 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** |
| **Arthav Patel** | **Debug and documented code** |  |
| **Sahil Khatri** | **Code integration testing** |  |
| **Jeet Patel** | **Test execution and document integration test cases** |  |
| **Yash shah** | **Finish coding remaining blackbox and whitebox cases** |  |
| **Samarth Shah** | **Finish coding all functions and add utils module** |  |
| **Jeetkumar Patel** | **Acceptance tests** |  |
|  |  |  |

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 |
| Split of tasks into each group member | **Each team member was given a distinct integration test job covering important situations.** | **Organized and great team work towards the completion of the milestone** |
| Integration Test Planning for Milestone 5 | **discussed how to do integration tests to make sure that various functions work together seamlessly.** | **Established a comprehensive plan for integration testing with clear responsibilities.** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**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? |
| Jeet Patel | **Assign tasks on Jira** | **10 mins** | **yes** |
| Samarth Shah | **Add remaining code files on github** | **15 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 |
| Jeetkumar Patel | Update Tracability matrix final one |
| Others | To be decided next week |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:+**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Task Allocation | Each member of the team was successfully given a specific task, ensuring that all crucial areas of the testing process were addressed. |
| Communication and Collaboration | The team actively participated in discussions, shared their knowledge and insights, and supported one another |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Planned meetings | **we planned the meeting so each member has proper time to attend the meeting** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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?  
     
   During the testing phase, we encountered some integration tests that didn't pass, preventing us from uploading code to the repository. Nevertheless, we promptly addressed these issues by identifying the root causes of the errors, refining the code, and repeatedly running the tests until they passed successfully. The GIT hook played a crucial role in ensuring that only reliable and thoroughly tested code was committed to the repository.
2. Explain why we are automating the testing process and what the advantages of this automation are.

Automating the testing procedure proved highly advantageous, offering several benefits. By automating tests, we gained faster feedback on code alterations, facilitating early detection and resolution of issues during development. This approach minimized the likelihood of human testing errors, thereby enhancing the overall reliability of the codebase. Additionally, streamlining the continuous integration process enabled us to deploy features with greater confidence and agility. The scalability of automated tests allowed us to effectively handle a large volume of test cases, supporting our endeavors to uphold code quality standards.

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?  
     
   Creating integration and acceptance tests presented greater challenges compared to black box and white box testing. It required a deeper understanding of how different functions operated and careful preparation of suitable test data, demanding significant consideration. Incorporating real-world user scenarios into acceptance test cases added further complexity. Despite these challenges, our team collaborated to craft comprehensive tests that accurately assessed system behavior. We prioritized black box and white box tests as they meticulously scrutinized individual components, whereas integration tests, focusing on function interactions, were less frequent due to their additional setup requirements. Acceptance tests were primarily aimed at ensuring key client requirements were fulfilled and the solution met user expectations.

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.

Although our code successfully passed black box and white box tests, conducting integration and acceptance tests remained necessary. Integration tests were crucial to validate the seamless interaction of diverse components, preempting any unforeseen issues arising from their collaboration. On the other hand, acceptance tests were indispensable in guaranteeing that the product aligned with consumer expectations and functioned as intended in real-world scenarios. These tests played a vital role in confirming that the product met customer demands and delivered the anticipated outcomes.