# SFT221 SCRUM Report and Reflections

**Member Present**:

1. Humaira Shaikh

## 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** |
| Humaira Shaikh | **Acceptance Testing , Integration Testing, Test Excution, SCRUM and Reflection** | **Completed.** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

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** | **N/A** |
| **Impact on Project** | **N/A** |
| **Solution or work-around** | **N/A** |
|  |  |
| **Delayed or Blocked Task** | **N/A** |
| **Reason for delay or block** | **N/A** |
| **Impact on Project** | **N/A** |
| **Solution or work-around** | **N/A** |

**Summary of Meeting:**

A summary of the main points discusses 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 then executed** | **Executed** |
| SCRUM | **SCRUM Done** | **Completed** |
| Reflection | **Discussing Reflection question** | **Completed** |
| Git | **Git update to each branch (Debug tickets on Git project)** | **Completed** |
| Jira | **Task Schedule setup in Jira (Debug ticket)** | **Completed** |

**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?  
     
   I've taken steps to enhance the development workflow by introducing informative prompt messages. These prompts guide developers through the commit process in the Bash terminal, ensuring careful consideration before finalizing changes.

Currently, I use the GitHub Desktop to monitor test outcomes, identifying any failures reflected in testing files. I make sure to commit the changes to the project, after resolving issues and implementing fixes. I also acknowledge the significance of the documenting progress, using commits to log significant advancements in files or the overall project, promoting transparency and collaboration.

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

Automating the testing process brings multiple advantages to software development:

Efficiency: Automated tests swiftly detect defects and regressions, accelerating the feedback loop.

Consistency: Precise test script execution eliminates human errors, ensuring reliable results.

Coverage: Automated tests comprehensively cover scenarios, including edge cases and configurations.

Frequency: Integration into agile and CI/CD pipelines allows frequent testing, catching issues early.

Scalability: Handling growing test cases efficiently suits evolving codebases.

Collaboration: Objective tests foster effective communication among stakeholders.

Cost Savings: Despite initial setup, long-term reduction in manual efforts yields cost benefits.

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?

Integration testing posed a greater challenge than white box testing due to the need for intricate connections between units and the creation of new testing code. Acceptance testing, particularly Alpha and Beta phases, brought further complexity.

Facing complexities, I intensified my focus on integration testing codes. Dissatisfied with initial black and white box testing outcomes, I invested more effort into refining our testing processes, particularly in MS05.

This dedication aimed to comprehensively understand the software's behavior, ensure compatibility, and proactively address potential issues, aligning with the commitment to a robust product. The testing journey demonstrated my goal to enhance strategies and deliver top-quality software.

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.

Black box and white box tests are vital for checking individual software parts, but integration and acceptance tests offer unique advantages that complement these stages.

Integration tests ensure different components, which passed earlier tests, work together without issues. They unveil hidden glitches, mismatches, or dependencies that emerge when parts interact. This guarantees seamless operation in real-world scenarios.

Acceptance tests (like Alpha and Beta testing) involve real users, revealing insights beyond technical correctness. They identify usability issues, user interface glitches, or performance bottlenecks that earlier tests might miss. Including real users ensures the software meets their needs, enhancing quality and market readiness.

In summary, integration and acceptance tests extend beyond black box and white box validations. Integration tests ensure smooth component interaction, while acceptance tests confirm user satisfaction and usability, minimizing defects and boosting software reliability.