# **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**: \_\_\_\_\_\_\_\_\_\_\_\_6\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Shwe Yee Lin Aung | 4. Cynthia Fotso |
| 2. Jackey Zhou | 5. Tsz Wa Wong (Locus) |
| 3. Nevan Sargeant | 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** |
| **Nevan** | **Reflection Q2, Acceptance Test** | **None** |
| **Shwe** | **Integration Test Cases, Scrum Report** | **None** |
| **Cynthia** | **Traceability Matrix, Reflection Q4** | **None** |
| **Jackey** | **Reflection Q1, Q3, Integration Test Code** | **None** |
| **Locus** | **Integration Test Code, Traceability Matrix** | **None** |
|  |  |  |
|  |  |  |

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 |
| Task Allocation | Tasks required to be completed are identified and allocated to each member according to availability. The tasks were allocated based on urgency and importance. Time control was also a great factor here because we had a lot do in this milestone as we first had to fix our previous errors before proceeding with milestone 5 requirements | A general allocation of tasks for each member was completed by collaboration and discussion among the group about which person was allocated to which task. On some tasks, we collaborated. |
| Documentation of tasks | To keep a thorough documentation of task assignments, progress updates and any decisions made during the task allocation process. | It serves as a reference point and help maintain accountability. |
| Feedback Mechanisms | Establish mechanisms for providing feedback on task progress and addressing any challenges or obstacles that arise. | Regular check-ins or status updates helps keep tasks on track and identify issues early. It was mostly done on teams call and in Jira. |
| Task Completion | Discussion is done to have a general summary and confirmation of completed tasks by each group member. | Acknowledge the contributions of each team member to the project or initiative. |
| Collaboration | Parties involved: The collaboration involved team members working together on the milestone and communicating through teams and Jira | As a result of the discussion, the team agreed on clear project milestones, assigned responsibilities to each team member, established regular progress review meetings, and designated communication channels. Additionally, potential risks and mitigation strategies were identified to ensure smoother project execution. |
|  |  |  |
|  |  |  |

**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 |
| Cynthia does the Traceability Matrix and Reflection Q4 | With a better understanding of herself, she can make choices that are more aligned with her goals and values. Reflecting on past decisions can also help her identify patterns and learn from her experiences. A Traceability Matric acts as a map, tracing requirements all the way through to the final product. This helps her verify that everything we were building aligns with the initial needs and specifications. |
| Jackey does Reflection Q1, Q3, Integration Test Code | He did so to encourage himself to think critically which helped with analysis, synthesis, and evaluation. Taking time to reflect on a problem can help you see it from different angles and come up with creative solutions. Detecting Integration Issues Early:  It uncovers problems that arise when multiple modules or components interact, addressing compatibility issues and interface errors. Catching these flaws early saves time and resources as they're often more costly to fix in later stages. |
| Nevan does Reflection Q2, Acceptance Test | Reflective practices encourage him to articulate their thoughts and experiences effectively. This improved communication skill can facilitate better collaboration, conflict resolution, and relationship-building in future collaboration. |
| Shwe does the integration test cases and the scrum report | Modularity and Loose Coupling:  Creating integration tests encourages developers to design modular, loosely coupled components. This improves maintainability, reusability, and testability, leading to more adaptable and scalable systems.  Simplified Change Handling:  Integration tests serve as safety nets when modifications are made.  They help ensure changes in one module don't negatively impact other parts of the system, reducing the risk of introducing new errors.  Enhanced Team Communication and Collaboration: The process of creating the report itself fosters communication and collaboration within the Scrum team. Team members discuss their work, identify dependencies, and ensure everyone is on the same page.  Better Progress Tracking and Measurement: Scrum reports help track the team's velocity (rate of completion) over time. This allows you to measure progress against goals, identify trends, and forecast future performance more accurately. |
| Locus does integration test code, Traceability Matrix | Enhancing Debugging and Troubleshooting:  When issues do arise, integration tests can pinpoint the root cause more efficiently. By isolating problematic interactions, they streamline the debugging process, leading to faster problem resolution.  Enabling Continuous Integration and Delivery:  Automated integration tests are integral to CI/CD pipelines, ensuring code quality and functionality with every new change.  This fosters a culture of continuous improvement and faster delivery of reliable software. |
|  |  |
|  |  |

**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? |
| Locus | **Test Code , Traceability Matrix** | **180 mins** |  |
| Shwe | **Integration test cases, Scrum report** | **120 mins** |  |
| Cynthia | **Traceability matrix, Reflection Q4** | **60 mins** |  |
| Jackey | **Test Code #4, #5 Reflection Q1, Q3** | **80 mins** |  |
| Nevan | **Reflection Q2, Acceptance Tests** | **120 mins** |  |
|  |  |  |  |
|  |  |  |  |

**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 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Integration Test Cases were done | Uncovers compatibility issues, interface errors, and problems that arise when components interact, preventing costly defects later in development or production. |
| Github | The files completed by each group members are uploaded for easier access by the professors and easier collaboration of the team going forward |
| Jira Updates | There is a documentation of who does what for each week tracked which makes it easier for the professor to see who made each contribution |
| Scrum Report | Help establish accountability within the team by documenting commitments and progress towards those commitments. Team members are accountable for completing their assigned tasks within the agreed-upon time frame |
| Reflections | Reflection encourages the group members to enhance their self-awareness, communication and collaboration which leads to overall improved performance and productivity within a team |
| Integration Test Code were done | Complements unit tests by focusing on interactions between modules, testing broader scenarios and ensuring more comprehensive coverage. |
|  |  |

**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 |
| Allocation of tasks | Group members identified what they were comfortable with doing and set off to do what was required for each task. |
| Completion of tasks | Since group members did what they were comfortable with and asked for clarifications during the group call, the progress went smoothly for each deliverable. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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?  
     
   We have not encountered any obstacles from using GIT hook to automate our testing. We have made effort of fixing the test case issues before pushing the code. It has been successful as we are able to push the code to out repository. In the event that a test were to fail, we would address it by going through the code line by line to see if we can figure out the issue and then attempt to push the code again ensuring that all tests pass successfully.
2. Explain why we are automating the testing process and what the advantages of this automation are.  
     
   Automating the testing process has its advantages when it comes to the testing process. This process allows developers to have a good strategic approach to software development that improves the efficiency, accuracy of the testing phases. This method allows software tools to execute predefined tests on applications still being developed. Things that gives the automating test process because with this, it can allow the test to create more coverage on the tests provided, tests are consistent and reliable as you can rely on the tests due to it being consistent so it can not be different on every new attempt.
3. 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?  
     
   In my opinion and experience, I find that writing black box and white box tests are more difficult to write than integration and acceptance tests because with black box and white box tests i have to test all potential user inputs, ensure that the software I am using is able to handle unexpected behavior effectively and efficiently. There typically might be more black box and white box tests compared to integration and acceptance tests. This is due to black box and white box tests focus on verifying individual componenets and functionalities wheareas integration and acceptance tests involve a system behaviour which may require more effort to define and execute. We created more whitebox and black box tests than integration and acceptance tests.
4. 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.

-COmprehensive testing scope: Black box and white box tests primarily focus on individual units of the software, meanwhile integration tests ensure that these units function correctly when combined together. On the other hand, acceptance tests verify that the software meets the specified requirements and is acceptable to the end user. Hence performing them allows us to cover a braoder spectrum of testing scenarios that might not have been addressed in white or black box tests.

-Detection of integration issues: WHen multiple components are integrated, if there are any issues with those, integration tests can help detect them early in the development process thereby allowing for timely resolution. Even though the components may have passed their respective tests individually, problems can still occur when they interact with each other.

- Acceptance tests ensure that the software meets user expectations and requirements. Even if individoual components pass black box and white box tests, acceptance tests help validate whether the software as a whole satisfies user needs and is user friendly.

- Integration and acceptance tests provide a more comprehensive assessment of the software's functionality, reliability and usability.