# **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**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_A\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Heungbin Oh | 5. Hiu Fung Chan |
| 2. Tien Vu To | 6. Trung Kien Phan |
| 3. Arshia Keshavarz Motamedi | 7. Ying Cheung Ellis Fung |
| 4. Aditya Tambe |  |

## 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** |
| Arshia Keshavarz Motamedi | Main function implementation and documents review | **-** |
| Heungbin Oh | checkInput() and checkShipmentVolume() functions implementation | **-** |
| Tien Vu To | checkShipmentWeight() and displayPercentageUsage() functions implementation | **-** |
| Aditya Tambe | Checkavailablespaces function coding and scrum report | **-** |
| Hiu Fung Chan | addDelivery() function implementation.  Blackbox, Whitebox test coding for addDelivery() and checkAvailableSpaces() and checkInput() functions | **-** |
| Trung Kien Phan | Blackbox, Whitebox test coding for checkShipmentWeight() and checkShipmentVolume() and displayPercentageUsage() functions | **-** |
| Ying Cheung Ellis Fung | Assigntruck() and findbestroute() function implementation and Blackbox, Whitebox test coding | **-** |

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 |
| White Box testing | 8 functions white box test coding arrangement | Everyone wanted to handle the Whitebox test coding |
| Integration Test cases | Integration Test cases design and coding arrangement | Everyone raised the issue according to task pick by their expertise |
| Function Implementation | 8 functions implementation arrangement | Everyone wanted to handle the Whitebox test coding |
| findBestRoute() function | Apply the algorithms for the function to find out the best route. | Ying Cheung Ellis Fung wanted to handle the function implementation |
| assignTruck() function | Handle special cases such as how to assign another truck if full or ships tomorrow if no truck’s available. | Ying Cheung Ellis Fung wanted to handle the function implementation |
| Documents update | Adding integration test cases and updating the test results. | Aditya Tambe and Arshia Keshavarz Motamedi wanted to work with teammates to complete the documentation. |

**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 |
| Function implementation | All team members will handle function implementation |
| Whitebox test coding | Who handle the function implementation also required to complete the Whitebox test coding |
| Integration test coding | Tien Vu To, Hiu Fung Chan and Trung Kien Phan will handle Integration test design and coding |
| Test case documentation and scrum report | Aditya Tambe and Arshia Keshavarz Motamedi will handle the Test case documents and scrum report |
|  |  |
|  |  |
|  |  |

**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? |
| Heungbin Oh | Checkinput() function implementation and Whitebox test coding | 13hrs | Y |
| Tien Vu To | displayPercentageUsage() function implementation and Whitebox test coding  Integration test design | 19hrs | Y |
| Arshia Keshavarz Motamedi | Updating Blackbox, Whitebox and Integration test cases and requirement traceability matrix files. Checkshipmentvolume() function Whitebox test coding | 18hrs | Y |
| Aditya Tambe | checkavailablespaces() function implementation and scrum report | 15hrs | Y |
| Hiu Fung Chan | addDelivery() function implementation and Whitebox test coding.  Integration Test coding. | 16hrs | Y |
| Trung Kien Phan | checkShipmentWeight() and checkShipmentVolume() functions implementation and checkAvailableSpaces() Whitebox test coding.  Integration Test design. | 16hrs | Y |
| Ying Cheung Ellis Fung | assignTruck() and findBestRoute() functions implementation and Whitebox test coding | 19hrs | Y |

**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 |
| Heungbin Oh | Integration test coding update and review.  Test execution. |
| Tien Vu To | Integration test coding update and review.  Test execution. |
| Arshia Keshavarz Motamedi | Integration test case document update and review.  Acceptance test document. |
| Aditya Tambe | Scrum report, documents review. |
| Hiu Fung Chan | Integration test coding update and review.  Test execution. |
| Trung Kien Phan | Integration test coding update and review.  Test execution. |
| Ying Cheung Ellis Fung | Functions review |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| As we mutually decided that we will raise the issue and assign to ourselves accordingly | This made it easier for all of us to focus on the task which was been issued by us and no need to of other to remind what is been assigned to each other |
| Holding and ensuring to fulfill the commitment | Everyone can select the tasks that they want to do and complete it on time without waiting for assignment. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Creating issues | We selected the topic we have expertise on. |
| Meeting the deadlines | As per agreement and topic chose by us, it became easy to deliver the topics on time by providing and submitting it on the time discussed and said in 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?  
     
   Yes, when we use the GIT hook file to automate testing, Git will stop to push the source code to GitHub if there has any test in the unit test project failed. We need to fix all the bugs of the source code that are stored in our computer until all the tests are passed, then we can push the bug-free source code to the GitHub repository that ensure the source code are workable and ensure the code quality is maintained. Tests are run automatically when you push the source code to the repository. It saves time and improves productivity.

1. Explain why we are automating the testing process and what the advantages of this automation are.  
     
   Automating the testing process can improve execution speed in software testing and allow testers to focus more on more strategic activities. Automated testing is a software or tool that automates the process of validating the functionality of software and ensures it meets all the requirements before release into production. Automated tests can run repeatedly and frequently at any time of day which is more cost-effective and providing higher efficiency to the project. Automated tests can perform steps very accurately and reduce human errors, eliminate the chances of human testers making errors like forgetting certain steps for test execution, therefore it can improve accuracy and provide full test results. Moreover, Automated tests can be re-used through different parts of the development lifecycle and on other similar projects.
2. 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 felt integration and acceptance tests can be more difficult to write than the black box and white box tests. Because we already used black box and white box tests to ensure the functions meet all the requirements and validated the functionality. Integration and acceptance tests ensure each individual component or function can work together and the outcome is the correct and expected result that finally meets the user requirements. So, we need to have deeper understand the relationship between the components or functions, how they process the data and pass the data to different functions for further processing and collect the results of the function and then provide the correct output in the program.

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.

The writing of integration and acceptance tests is still required even though the black box and white box tests have already passed. It is because integration and acceptance tests ensure each individual component or function is working well and together as a whole to complete the tasks. Moreover, the system fulfills the requirement of the end user and their perspective. It also helps to catch the issues that may or may not be found in black box and white box tests. For example, integration and acceptance tests can validate the data integrity, software performance, software reliability, interface layouts and user experience.