# **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**: 2

**Members Present**:

|  |  |
| --- | --- |
| 1. Joon Dong | 4. Heqing Xu |
| 2. Doris Chai | 5. |
| 3. Xiaopeng Liu | 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** |
| **All** | **Functions implemented** |  |
| **All** | **Black box and white box test completion** |  |
| **All** | **Ensured all test cases pass and meet traceability requirements** |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

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 |
| Integration tests | **Discussed how integration tests are implemented** | **Understand that we need to make a function which encapsulates the functions we made** |
| Acceptance tests | **Discussed how acceptance tests are implemented** | **Acceptance tests are not implemented but rather documented ensuring they meet the project requirements** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**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 |
| Main should be calling a function which we use for integration testing | Cannot code in main as there is no way to test it effectively |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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? |
| Joon | **Reflection questions** | **15m** | **n** |
| Doris | **Implementing calculation of shortest route** | **15m** | **n** |
| Xiaopeng | **Implementing user input and user interface** | **15m** | **n** |
| Heqing | **Implementing user input and user interface** | **15m** | **n** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**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 |
| Joon | Implement a function to determine the path of the truck to be used in main |
| Doris | Jira management, implement a function to determine which truck to be used in main |
| Xiaopeng | Implement user interface that validates user input for package |
| Heqing | Implement user interface that validates user input for destination |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Learned what integration tests are | **Can see the full scope of the project** |
| Learned what acceptance tests are | **Ensure that everything we implement meets business requirements** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Integration/acceptance testing | **Clarified with professor Fagun** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 set up the git hook to make sure that when we push code, all unit tests are conducted, and the push can only be successful if all the tests are passed. So far we have not found any tests that are failed, mostly because we usually ran all the unit test before pushing to make sure there are no known bugs in our code. It is very important especially in a group work environment where we rely on each other’s code. If there are test failing, it is our responsibility and respect to team members that we run unit test before push and if any bugs are found, we fix the issues before pushing to the central repo. This prevents error spread to other team member’s code and logic.

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

We automate the test process to enforce that our test cases must pass before being able to push changes to our centralized repository. The advantages allow us to streamline the process while also minimizing bugs that get through the testing phase. Automation generally requires us to do less testing ourselves manually while making testing more frequent and efficient. As I said, it catches bugs earlier in the development process and consistently as it must adhere to the pre-push hook file. In summary, it is only advantageous as the tester has to do less manual testing and is given the ability to test more efficiently, frequently and in a robust manner.

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?  
     
   I thought Integration and acceptance tests can be more challenging than black box and white box.Integration tests involve testing the interactions between different components or modules of a system. This complexity can make it more difficult to set up test environments and scenarios. Integration tests often require coordinating dependencies between different parts of the system, which can be challenging to manage. And, Acceptance tests typically cover end-to-end scenarios from a user's perspective, which may involve multiple layers of the system. Ensuring comprehensive coverage of all possible scenarios can be challenging. The number of tests written for each category depends on various factors such as project requirements, time constraints, and the nature of the system being tested. In general, a balanced approach that includes a mix of all types of tests is often considered ideal for achieving comprehensive test coverage.
2. 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 testing and white box testing mainly focus on the functionality and internal logic of a single module or component. However, when these modules or components are integrated with other parts, new problems may arise. Integration testing ensures that the integration of different components, modules or systems works properly and exposes defects in interfaces and interactions. Acceptance testing typically involves end users or people acting on behalf of end users to ensure that the software meets their expectations and needs. So integration and acceptance testing are also necessary because they provide the opportunity to evaluate the software from different levels and perspectives to ensure that the software as a whole can meet technical specifications, business needs and user expectations.