# SFT221 SCRUM Report and Reflections

This report should be completed in the class and submitted at the end of class. Late submissions cannot be accepted without prior approval of the instructor.

**GROUP**: \_\_\_\_\_\_\_\_\_\_\_\_\_5\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1.Wilson Sum | 4.Sasawat Yimleang |
| 2. Lebna Noori | 5.Samin Sorayya |
| 3.Radmehr Behzadfar | 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 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** |
| **Wilson Sum** | **Made integration Unit tests for section B of coordinates, filled out scrum report individual parts, reflection question 1, summary of meetings, summary of decisions made, learned how to use githook. Prepped the acceptance unit test for group and made 9 acceptance tests. Updated data report.Debugged shortest path func, it now won’t cause a crash when entering a out of range coordinate.** |  |
| **Lebna Noori** | * **Worked with Radmehr to make integration Unit tests functions labelled with an A** * **filled out scrum report (individual part),** * **learned how to use git hook** * **Filled out reflection questions (Q1 not included).** |  |
| **Radmehr Behzadfar** | **Participating in group meeting.**  **Fulfilling the needs and requirements for doing test cases and unit testing.**  **Co operating with lebna in order to complete our testing parts.**  **Updating latest status on jira**  **Committing an pushing latest version of test cases and unit tests on github**  **Filling the scrum report**  **Learning about githook** |  |
| **Sasawat Yimleang** | * **Participating in group meeting.** * **Fill out scrum report (Major outcome, Things went well and Things did not go well parts).** * **Made integration Unit tests for getUserInput with validateWeight, validateBoxSize, ValidateVolumn and calculateLimitFactor functions.** * **Prepped the acceptance unit test for group and made 3 acceptance tests.** | **The getUserInput function has some bugs and I need to spend more time to fix it.** |
| **Samin Sorayya** | * **Created 15 integration tests for four functions, all of which passed successfully.** * **Improved the findAvailableTruck function for better performance and functionality.** * **Gained knowledge about GitHooks and their potential to enhance testing and development processes.** * **Contributed to the scrum report by filling out specific sections to detail progress and contributions.** * **Maintained up-to-date task status on Jira to ensure the team was informed about progress and updates.** * **Successfully committed and pushed the latest test cases and unit tests to our GitHub repository.** |  |
|  |  |  |
|  |  |  |

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 |
| How to split up tasks | **Discussed about how to split the amount of the functions to integrate and test.** | **Functions are divided into separate categories that they can integrate well and be tested with.** |
| Need to learn how to use Git hook | **Due to the fact that we are now moving on to using Git hook for our tests all members must now learn how to use git hook.** | **Everyone needs to learn how to use git hook.** |
| How to make tests | **Discussed about how to make the integration tests since there many combinations between the functions.** | **All team members may use any and all functions in their divided categories however they choose but their tests must not be the same.** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**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 |
| Everyone must learn git hook | It’s a required criteria so we must now all learn how to use git hook. |
| Functions are to be divided into categories to be integrates and tested | Matching the functions to a particular category will help integration and make things easier on us. |
| Tests are not required to be a fixed amount and not all combinations needed to be tested | There is a huge amount of combinations between the three different subdivided categories. So it would be hard for us to test all the functions combinations therefore each member may use whatever functions in their assigned categories but their tests may not be similar. |
| For next week after we test our own sections we are to start merging with eachothers categories | Since we have two weeks for MS5 it is best that we test out our own sections first and then start further integrating with eachothers functions. |
|  |  |
|  |  |
|  |  |

**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? |
| Wilson Sum | **Filled in scrum report, update jira.** | **1 hr** |  |
| Lebna Noori | * **Filled in scrum report** * **Reflection questions (1 not included)** * **Updated jira** | **1 hr n 30 mins** |  |
| Sasawat Yimleang | **Filled scum report, analyze the task and design the test cases.** | **2 hr** |  |
| Samin Sorayya | **Filled in scrum report, update jira.** | **30 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 |
| Wilson Sum | Fill out scrum report, integrate tests to other members tests, make up at least 4 acceptance tests. |
| Lebna Noori | Fill out scrum report, reflections questions, and integrate tests to other members tests, make acceptance tests. |
| Sasawat Yimleang | Fill out scrum report, integrate tests to other members tests, make at least 3 acceptance tests. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Analyzed the assignment of MS5. | **A clear understanding of the requirement and goal of MS5 will make members understand their responsibility.** |
| Assigned responsibilities on testing part to each member. | **Each member is responsible for the integration test cases and debugging the function below:**   * **Radmehr, Lebna: getClosestPoint and distance functions.** * **Samin, Sasawat: getUserInput, validateWeight, validateBoxSize, ValidateVolumn and calculateLimitFactor functions.** * **Wilson:** **addPointToRoute, addPointToRouteIfNot, addRoute, eqPt functions** |
| Declared the due date of each part. | **The due date for all tasks in this Milestone is on August 5, 2023 in case that has the unexpected happens and combine all test cases together. This will ensure the project will go properly.** |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Task assignment | **We discussed and made agreements about the person responsible for the task of each section of the functions.**  **The meeting ensured that each task was allocated to a responsible team member.**  **Clear roles and responsibilities were defined, avoiding confusion or duplication of efforts.** |
| Participation | **All Team members presented actively participated and contributed to the discussions.**  **Active participation fostered collaboration and helped generate valuable input and solutions.** |
| Decision-Making | **Decisions were made in a timely and collaborative manner.** |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Work load balance | **Some of the members have to work, when we assign a task to two members, one of them has to do more than the other. To solve this solution we need to share the load with other teams.** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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?

So far we have not found any tests that have failed yet so we are in the clear. However though

If we were to find any what we would do is look at the test that had failed and investigate why it failed. How we would approach this is on a separate visual studios, we would recreate the test but we would also look into the blackbox cases so we can see what goes on for the output. Afterwards, once we find out the problem what we would do is fix it recreate the unit test and then update the project file with the .dll that had the failed test and run it again. We would continue to do this till all the tests have been run successfully and we can finally push the updates to our github repository.

1. Explain why we are automating the testing process and what the advantages of this automation are.
   1. Using software tools, scripts, or structures to run tests and compare the results with what was expected is known as automating the testing process. This is done to make sure that software programs are reliable over time, work as planned, and meet all criteria. Automating the testing process is extremely helpful for several reasons:
      1. Tests can be performed at any time and are quick.
      2. Testing is consistently executed in the same way.
      3. More situations and cases are tested in terms of coverage.
      4. Catch Problems Early: Locate issues before they spread.
      5. Reduces the cost of manual testing.
      6. Continuous Integration and Continuous Delivery (CI/CD): Works nicely with networks for many upgrades.
      7. Flexibility: Useful for large-scale, complex structures.
      8. Reports: Provides detailed logs for analysis.
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?
   1. Depending on the particulars of the project, the testing methods used, and the team's experience with the testing technique, the comparison of the difficulty of creating white box, black box, integration, and acceptance tests can change, and is based on the project's complexity. But ideally for several reasons, writing integration and acceptability tests can be more difficult than writing black box and white box tests. For example:
      1. Integrations test how parts work together, and acceptance tests check if the whole app meets needs. This complexity can lead to unpredictable results.
      2. These tests need realistic setups with databases, services, etc. This takes time and can be error prone.
      3. Tests often deal with external systems, like APIs. Handling these right and making mock environments can be tricky.
      4. Integration tests need to check many connections, and acceptance tests need to cover various user situations. More coverage means more work.
      5. As apps change, these tests can break due to new needs or dependencies. Keeping them accurate gets tough.
      6. These tests might take longer to run, so feedback on issues is slower compared to quick-running unit tests.
      7. They might need more computer power and time, which isn't ideal for fast, repeated testing during development.
3. 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.

Integration and acceptance tests are still important even if code passes black box and white box tests because they:

* + 1. Ensure parts work together correctly.
    2. Confirm the software meets user needs or the requirements.
    3. Uncover issues when different parts interact.
    4. Check full user interactions.
    5. New code changes might affect existing parts.
    6. Test in a setting that is more like reality.
    7. Confirm that the program offers real value.
    8. Get a full view of software quality.
    9. Catch issues from different angles.
    10. Detect problems not visible in other tests.