# **Milestone 4 Scrum Report**

All students are expected to attend the scrum meetings and to participate. Failure to do so will result in greatly reduced grades.

**GROUP**: \_\_\_\_4\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Riaz Hossain | 4. Judd Niemi |
| 2. Mustafa Siddiqui | 5. |
| 3. Abdiwali Warsame | 6. |

## Milestone 4 Tasks

* Finish implementing/coding the functions.
* Finish implementing/coding blackbox tests. Store in repo, executed, results in Jira (and on corresponding test documents, and debugged.
* A set of whitebox tests as test documents (in an Excel file) with test data for the functions you created. At least 4 sets of test data are required for each function. You must have test cases for at least 6 functions (including all your custom function). Stored in the repository.
* Whitebox tests implemented (in the C++ testing project), stored in repository, executed, results in Jira and on corresponding test documents, and debugged (at least 1 SET is required).
* Updated requirements traceability matrix in the repository, ensuring it shows both passed (green) and failed (red) tests.
* Completed hook file (for EACH team member) for test automation stored in the repository.
* Completed scrum report including reflection questions answered.

Note: Your professor will **only grade** the **master** or **main** branch, unless you indicate otherwise.

**Rubric:**

|  |  |  |
| --- | --- | --- |
| **Individual** | Group participation (includes GitHub commits and Jira usage) | 80% |
| Teamwork | 20% |
| **Group** | Implemented functions and main (well-designed, and documented) | 10% |
| Finish coding blackbox code (well-designed, written, and documented) | 5% |
| Whitebox test case document (well written, complete, good test data) | 10% |
| Whitebox test code (well designed and documented) | 5+% |
| Updated requirements traceability matrix | 10% |
| Test execution (performed, results recorded, issues created) | 5% |
| Debugging (bugs fixed, documented, Jira updated) | 5% |
| Hook files | 15% |
| Git usage (used properly with good structure) | 5% |
| Jira usage (creates issues, tracks progress) | 20% |
| Scrum report & reflections | 10% |
| **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** |
| 1. Riaz Hossain | **Wrote scrum report draft, started white-box test document, updated traceability matrix** | **None** |
| 2. Mustafa Siddiqui | **Finished implementing functions with full algorithms and error handling** | **None** |
| 3. Abdiwali Warsame | **Completed Blackbox test code** | **None** |
| 4. Judd Niemi | **Completed pre push hook script** | **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** | **None** |
| **Reason for delay or block** | **None** |
| **Impact on Project** | **none** |
| **Solution or work-around** | **None** |
|  |  |
| **Delayed or Blocked Task** | **None** |
| **Reason for delay or block** | **None** |
| **Impact on Project** | **none** |
| **Solution or work-around** | **None** |

**Summary of Meeting:**

A summary of the main points discusses in the meeting and the outcomes of the discussions.

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| MS4 workload distribution | Talked about the workload and which member should do which task | **Members were assigned their tasks for MS4** |
| Submission deadline | **Finalized the submission date and time to set a deadline for all tasks** | **Handed in on time** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**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 |
| Assign Riaz to complete scrum report and white-box tests | To ensure report quality and testing documents are delivered |
| Mustafa to finish implementing functions for MS4 | He worked on algorithms in MS3 and could complete them faster |
| Traceability matrix updates done after white-box test doc | Logical sequencing: tests first, then matrix |
|  |  |
|  |  |
|  |  |
|  |  |

**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? |
| 1. Riaz Hossain | **Drafted scrum report, started white-box test doc** | **60 mins** |  |
| 2. Mustafa Siddiqui | **Implemented remaining functions and error handling** | **1.5 hours** |  |
| 3. Abdiwali Warsame | **Tested and ran black box tests using debug tools.** | **1.5 hours** |  |
| 4. Judd Niemi | **Added the pre push hook script** | **60 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 |
| 1. Riaz Hossain | Scrum report + reflections + code review |
| 2. Mustafa Siddiqui | Implement the remaining functions |
| 3. Abdiwali Warsame | Write and run tests. Add jira tasks. |
| 4. Judd Niemi | White box testing and code review |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Task assignments finalized for MS4 | |  | | --- | |  |  |  | | --- | | **Allowed members to work independently and avoid duplication** | |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Meeting time | **Everyone came on time to talk about ms4** |
| Even work distribution | **Everyone was assigned tasks evenly** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| None | **none** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

Answer the following questions using your own words. Make sure that each answer comprises a minimum of 100 words.

1. How did analyzing the internal logic and structure of the code help you design effective white-box test cases?  
     
   Analyzing the internal logic and structure of the code was crucial for designing effective white-box test cases. By understanding how each function handled edge cases like NULL pointers, maximum capacities, and fallback logic, I was able to target specific code paths that might otherwise be missed. For example, knowing calculateRouteDistance falls back to Euclidean distance helped me create tests for both A\* success and failure scenarios. This approach allowed me to check not only the expected outputs but also ensure all branches and conditions were covered. It made testing more thorough because I could validate that the functions behave correctly under normal, boundary, and error conditions, reducing the risk of hidden bugs.
2. How did using automated unit testing tools simplify or enhance your testing process? Reflect on the advantages and potential limitations of automation compared to manual testing methods.  
     
   Using automated unit testing tools simplified and enhanced the testing process significantly. Automation allowed me to quickly run all test cases after each code change without manually verifying outputs. This saved time and made it easier to catch regressions early. The tools also provided clear pass/fail indicators and detailed feedback when a test failed, making debugging faster. However, automation has limitations. It relies on well-written test cases and cannot verify aspects like user experience or system-level interactions. In contrast, manual testing is slower but allows for exploratory testing. Overall, automated tools made the process more efficient and repeatable, especially for functions with multiple edge cases like those in MS4.
3. How did you document and communicate the bugs you identified? Reflect on the importance of clear and detailed bug reports in ensuring that issues are effectively resolved by the development team.  
     
   I documented and communicated identified bugs by creating detailed reports in Jira and updating the Requirements Traceability Matrix. Each bug report included a clear title, description, steps to reproduce, expected versus actual results, and screenshots when necessary. This helped the development team quickly understand the issue and prioritize fixes. Clear and detailed bug reports are essential because they reduce miscommunication and prevent redundant debugging efforts. They also provide a historical record of issues, which can be useful for future maintenance. In MS4, this process ensured that all team members could see progress and address problems efficiently, improving our overall workflow and code quality.