# **MILESTONE 4** -- 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**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_B\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Jeet Patel | 4. Arthav Patel |
| 2. Samarth Shah | 5. Jeetkumar Patel |
| 3. Sahil Khatri | 6. Yash Shah |

## Milestone 4 Tasks

**Deliverables due 4 days after your lab day:**

* 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 stored in the repository.
* Completed hook file (for EACH team member) for test automation stored in the repository.
* Completed scrum report including reflection questions answered.

**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) | 20% |
| Updated requirements traceability matrix | 5% |
| Test execution (performed, results recorded, issues created) | 10% |
| Debugging (bugs fixed, documented, Jira updated) | 5% |
| Hook files | 10% |
| Git usage (used properly with good structure) | 5% |
| Jira usage (creates issues, tracks progress) | 10% |
| 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** |
| **Samarth Shah** | **Functions implementation & main** |  |
| **Yash shah** | **Whitebox test coding** |  |
| **Sahil Khatri** | **Document Whitebox tests** |  |
| **Jeet Patel** | **Hook Files & test execution** |  |
| **Arthav Patel** | **Fixed bugs and documenting them** |  |
| **Jeetkumar Patel** | **Update tracability matrix** |  |
|  |  |  |

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 |
| Functions implementation | **as Samarth and sahil did blackbox coding last week,one of them can do coding this week.** | **The functions are to be coded by samarth** |
| Tracability matrix | **Jeetkumar created it last week, so he would know better to update it.** | **To be done by Jeetkumar** |
| Whitebox test cases | **Sahil did a good job doing blackbox testing last week. Should he do whitebox test cases?** | **Sahil will complete whitebox testing.** |
| Whitebox test code | **Between Arthav and Yash , who will code whitebox** | **Yash will do whitebox and Arthav will debug the functions implemented by Samarth.** |
|  |  |  |
|  |  |  |
|  |  |  |

**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? |
| Samarth Shah | **Start implementing functions** | **30 mins** | **Little bit** |
| Sahil | **Start writing whitebox tests** | **15 mins** | **no** |
| Jeet Patel | **Hook files** | **20 mins** | **yes** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**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 |
| Samarth | Finish coding all functions and main |
| Sahil | Traceability matrix |
| Others | To be decided next week |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Quick assignment of tasks | **As compared to previous meetings, members are now used to everything, and work has become easier as people understand** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Work distribution | **Members confirm equal distribution of workload as no single member should singlehandedly do more work than others. This ensures that no conflict arises and work becomes convenient for everybody.** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

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

1. After you run your blackbox and whitebox tests you are asked to record the results in both the original test document as well as in Jira. Explain why it is a good idea to record the results in both places.

Recording test results in both the original test document and Jira is advantageous for several reasons:

**Comprehensive Documentation**: The original test document provides detailed information about test cases, objectives, and expected outcomes. By recording results in this document, the testing process and its outcomes are comprehensively documented, aiding in historical analysis and reporting.

**Efficient Issue Tracking**: Jira is a widely used tool for issue and project tracking in software development teams. Logging test results in Jira enables quick documentation of any issues or bugs identified during testing. This facilitates efficient collaboration among team members and ensures that identified issues are promptly addressed.

**Centralized Reporting**: Recording results in both places centralizes test data, making it easily accessible for stakeholders and management to monitor testing progress and outcomes. Centralized reporting supports data-driven decision-making and provides insight into the overall health of the project.

**Audit and Compliance Support**: Some projects require adherence to specific testing standards or regulatory guidelines. Maintaining records in multiple locations provides better support for audit purposes, ensuring that testing activities align with required standards and compliance regulations.

1. Why did we wait until the fourth milestone to write the whitebox tests?  
     
     
   Writing whitebox tests at later stages of development, typically around the fourth milestone, offers several advantages. Firstly, by this point, a substantial portion of the codebase is likely to have been implemented and stabilized, ensuring that the code is mature enough for in-depth analysis and testing based on its internal logic. Secondly, waiting until later milestones ensures that the code is more stable and accessible, as in the early stages, the code may not be fully available or could undergo frequent changes. Additionally, in the initial stages, the primary focus often lies in validating functionality and user experience through blackbox testing, whereas later stages allow for prioritizing testing of internal logic and edge cases using whitebox testing techniques once core features are confirmed to be working as expected.
2. Pick one of the functions you created and list its name. For this function did you produce more blackbox or whitebox tests? Explain why your answer (more blackbox or more whitebox) happens for most functions.

**assigntruck** function:

For this function , whitebox tests were more produced as more details about the function were known because the function was implemented in this milestone , in the previous one , function code was not known and assumotions were to be made during blackbox testing. This was not the case in this workhop, hence more number of whitebox tests were possible. Whitebox tests are typically focused on testing the internal logic, paths, and code structure of a function. They are particularly useful for coverage of specific code branches, boundary conditions, and code optimization. However, writing comprehensive whitebox tests may require more effort, and they may not be necessary for every function, especially if blackbox tests provide sufficient coverage of the function's behavior.

1. Explain the purpose of the automation hook for GIT and explain how it can improve the quality of the software in the project.

Automation hooks within GIT are scripts or programs triggered automatically by specific actions in a repository, such as code commits or pushes. They aim to automate processes, tests, or actions, thereby improving software quality.

One notable benefit of automation hooks in GIT lies in their integration within Continuous Integration (CI) systems. Here, these hooks play a pivotal role in triggering a sequence of build, test, and deployment procedures whenever fresh code alterations are pushed to the repository. Through CI, potential integration issues are promptly identified, bugs are unearthed, and newly introduced code changes undergo thorough validation against existing codebase and tests, thus fortifying the project's stability.

Moreover, automation hooks facilitate the initiation of automated test suites encompassing various levels, including unit tests, integration tests, and end-to-end tests. Such automated testing procedures serve to safeguard against regression, ensuring that code modifications uphold the desired software behavior while minimizing the likelihood of unintended side effects.

In addition to testing, automation hooks can enforce predefined quality gates, such as executing static code analysis tools, prior to accepting code alterations. By adhering to established coding standards and identifying potential code issues, these hooks contribute to bolstering the overall quality of the codebase, thereby fostering a robust development environment.