# **Milestone 3 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. Abdiwali Warsame |
| 2. Judd Niemi | 5. |
| 3. Mustafa Siddiqui | 6. |

## Milestone 3 Tasks

In this milestone you will create issues to design the functions, design all of the functions you need to complete the project and store the specifications in the repository. As soon as the specifications start to be produced, you can start to design the blackbox tests (what they test, how to perform them and test data). Once tests are written, they can be implemented and added to the repository and any team members not otherwise busy can start to implement the functions. You will also build a function-test matrix that shows the blackbox tests for each function. This will be maintained through the testing cycle as new tests are added.

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

* A set of AT LEAST 4 function specifications added to a new header file and stored in the repository.
* A set of blackbox 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.
* **Create and add a C++ testing project to your solution.**
* Start writing blackbox test code (for the functions above) and store in repository (at least 1 is required for this milestone).
* Start implementing the functions and store them in repository (optional).
* A requirements traceability matrix added to the repository and shows the mapping between the requirements and test cases.
* Updated Jira project to show activities and progress.
* 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** | Function specifications (documented, complete, well-written, added to the project) | 15% |
| Blackbox test cases document (well-written, complete, good test data) | 10% |
| Blackbox test code (in the C++ project) well-designed and documented | 10% |
| Functions implementation (coded in the C project & well documented) | 5% |
| Visual Studio solution with 2 projects (complies and works) | 10% |
| Requirements traceability matrix (complete and added to GitHub) | 10% |
| Git usage (used properly with good structure) | 10% |
| 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** |
| **Riaz Hossain** | **Completed scrum report + review all source code** | **None** |
| **Mustafa Siddiqui** | **Completed Initial function execution** | **None** |
| **Abdiwali Warsame** | **Completed the blackbox test case document and wrote the function specs** | **None** |
| **Judd Niemi** | **Completed and setup unit test framework** | **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** |  |
| **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 |
| MS3 workload distribution | Talked about the workload and which member should do which task | **Members were assigned their tasks for MS2** |
| 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 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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? |
| Riaz Hossain | **Scrum report + reflections** | **60 mins** | **Yes** |
| Judd Niemi | **Unit test framework** | **60 mins** | **Yes** |
| Mustafa Siddiqui | **Initial Function Execution** | **60 mins** | **Yes** |
| Abdiwali Warsame | **Write Function Specs and Create blackbox test case** | **120 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 |
| Riaz Hossain | Scrum report + reflections + code review Updated requirements traceability matrix in the repository |
| Mustafa Siddiqui | Finish implementing/coding the functions. Completed hook file |
| Judd Niemi | Whitebox tests implemented |
| Abdiwali Warsame | A set of whitebox tests as test documents |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 ms3** |
| 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 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

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

1. How did your approach differ when creating test cases for blackbox testing versus whitebox testing? Reflect on the advantages and limitations of each method based on your experience with the assignment.  
     
   When creating test cases for blackbox testing, I focused on the functional requirements without considering how the code was written internally. My goal was to test inputs and expected outputs, covering normal cases, edge cases, and invalid inputs. For whitebox testing, my approach was very different. I studied the source code to identify all possible code paths, branches, and logic conditions to ensure complete coverage. Blackbox testing was quicker and easier because I didn’t need knowledge of the implementation, but it felt limited for complex logic. Whitebox testing was more thorough but required more effort and understanding of the program’s structure. Both methods complement each other; blackbox tests validate user-level functionality, while whitebox tests ensure internal code quality.
2. How did the traceability matrix help ensure that all functional specifications were adequately tested? Reflect on its role in maintaining comprehensive test coverage.  
     
   The traceability matrix was essential for keeping track of which test cases corresponded to which functional requirements. It gave me a clear view of whether each specification was addressed by at least one test. While developing test cases, I referred to the matrix to make sure I didn’t overlook any requirement. It also made it easier to identify gaps in our testing process and prioritize creating missing test cases. By mapping requirements directly to tests, the matrix provided confidence in the completeness of our testing and helped avoid redundant or irrelevant tests. It acted as both a planning tool and a verification tool to ensure nothing was missed.
3. Write down two of the function prototypes you submitted. Why did do you need each one of them and how will each one help you achieve the project needs?

One function prototype I submitted was int validatePostalCode(const char\* postalCode);. I needed this function to check whether a customer’s postal code matched the expected format for Canadian postal codes. It ensures that data entered into the system is clean and reliable, which is crucial for processing customer records accurately.

Another prototype was double calculateOrderTotal(struct Order\* order);. This function calculates the total price of all items in a customer’s order. It is necessary to support billing and reporting features in the system. Without it, we couldn’t automate billing, which is a key requirement for the project.