# **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**: **\_\_\_\_\_1\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Members Present**:

|  |  |
| --- | --- |
| 1. Mahmadsahil Mahmadrafik Shah | 4. Ying Wang |
| 2. Pouya Rad | 5. Dil Humyra Sultana Borna |
| 3. Ryaan Farrukh | 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 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) | 10% |
| Updated requirements traceability matrix | 10% |
| 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) | 15% |
| 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** |
| Mahmadsahil Mahmadrafik Shah | Whitebox test case document (well written, complete, good test data),  Whitebox test code (well designed and documented). Git and Jira usage. | **n/a** |
| Ryaan Farrukh | Finish coding Blackbox code (well-designed, written, and documented),  Debugging (bugs fixed, documented, Jira updated), Git and Jira usage. | **n/a** |
| Pouya Rad | Implemented functions and main (well-designed, and documented),  Test execution (performed, results recorded, issues created). | **n/a** |
| Ying Wang | Requirements traceability matrix (complete and added to GitHub),  Blackbox test cases document (well-written, complete, good test data). | **n/a** |
| Dil Humyra Sultana Borna | Scrum report & reflections, Whitebox test code (well designed and documented), Hook files. | **n/a** |
|  |  |  |
|  |  |  |

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** | **n/a** |
| **Reason for delay or block** | **n/a** |
| **Impact on Project** | **n/a** |
| **Solution or work-around** | **n/a** |
|  |  |
| **Delayed or Blocked Task** | **n/a** |
| **Reason for delay or block** | **n/a** |
| **Impact on Project** | **n/a** |
| **Solution or work-around** | **n/a** |

**Summary of Meeting:**

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

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Distribution of the work | According to the MS requirements, the tasks were divided among the group members. Assigning creating functions, test cases, and SCRUM report fairly between all members. Everyone was given the choice to select their tasks and work on them as discussed to make it easy for their schedules. | Members were given the flexibility to select tasks based on their preferences, allowing them to maintain a sense of a fair amount of work. |
| Test plan | The black box and white box tests were divided among the group members, and they were asked to do the test cases. Discussing this ensured how functions and their test cases would work so they can be done accurately. | Each member worked on different functions and test cases. |
| Reflect Questions | To answer the reflection questions effectively, a comprehensive examination and research effort into both the project and relevant theoretical subjects are essential. Collaboratively, members engaged in discussions, sharing their perspectives and offering constructive suggestions to enhance the overall understanding. | 1 member were responsible for the scrum report and the reflections, but each member helps with the reflection 3 question, creating shell script. |
| Filling up the Scrum Report | This milestone presents a more extensive set of components compared to the previous one, emphasizing the crucial need for accurate completion of the SCRUM report. In the collaborative discussion, members strategized on the distribution of tables among the team. It was decided that each member would independently check the tables related to their individual work. | Members are responsible for individually rechecking all tables associated with their specific tasks. For the remaining tables, a collaborative approach was adopted. This ensures that the SCRUM report is accurately and efficiently compiled. |
| Blackbox testing & Whitebox testing | As this included coding, every member was responsible for this part, so that we can help each other and completed the task on time, this also increase our knowledge of coding and testing by group working. | It was done with everyone’s participation and whenever we faced any problem, solved it immediately without wasting any time. |
| Deadlines | To prevent last-minute panic and rushed work, the team made decisions regarding individual deadlines, emphasizing the importance of thoughtful planning and timely completion of tasks. | The team collectively agreed that each member is required to finalize their individual tasks at least 6 hours before the deadline, specifically by 6 PM on Tuesday. This decision was made to ensure ample time for review, adjustments, and the overall avoidance of last-minute rush. |
| JIRA Collaboration | Recognizing the integral role of each member's utilization of JIRA in determining their grade, the team engaged in discussions to establish measures that would ensure active team involvement in this aspect. | It is mandatory for each team member to provide comments under their assigned tasks and consistently update the status of their tasks. This practice ensures clear communication, accountability, and transparency within the team. |

**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 |
| The deadline for each individual task was set to 9 PM Tuesday | This made sure everyone got enough time to discuss and edit the group submission and gave enough time to review and consider the suggestions of each group member |
| SCRUM report must be checked by individuals after the completion of each task | This would help create an accurate log of all the activities carried out by each member. it is important because it makes sure every member is informed about the progress made. |
| Collaborating and communicating on JIRA is important and comments must be made under every completed or in-progress task. | To utilize the project management software JIRA, it is important to use it as a means of communication by using it to comment, add descriptions and tag other members. |
|  |  |
|  |  |
|  |  |
|  |  |

**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? |
| Mahmadsahil Mahmadrafik Shah | Worked on the Whitebox test and checked the individual SCRUM report. | 11 | completed |
| Ryaan Farrukh | Worked on the Blackbox test and checked the individual SCRUM report. | 9 | completed |
| Pouya Rad | Implemented functions and main and checked the individual SCRUM report. | 7 | completed |
| Ying Wang | Updated requirements traceability matrix and checked the individual SCRUM report. | 8 | completed |
| Dil Humyra Sultana Borna | Reflection (Shell script), testing, scrum report. | 6 | completed |
|  |  |  |  |
|  |  |  |  |

**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 |
| Mahmadsahil Mahmadrafik Shah | Integration test code |
| Ryaan Farrukh | Finish coding all functions and main |
| Pouya Rad | Finish coding Blackbox and Whitebox case |
| Ying Wang | Requirements traceability matrix updated |
| Dil Humyra Sultana Borna | Acceptance tests (well-designed, documented, and implemented) |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Functions | Understanding how the code works, and coming up with functions that benefit the code. |
| Testing cases | Making sure we have an organized test case plan (Blackbox & Whitebox) for each function so that it benefits the code. |
| Tracking purpose | Making sure we also fill out a traceability matrix sheet to keep track of functions purpose for the program. |
| Keeping in contact | Using JIRA to keep in touch, as well as committing to Github to collaborate. |
|  |  |
|  |  |
|  |  |

**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 |
| Participation | Everyone participated in the meeting and did their part. |
| Collaboration | We collaborated to understand the whole milestone and discussed all the requirements. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| Assigning Tasks | **As we have started working together for each task so that everyone can participate in the coding and testing part from previous milestones there was a bit of confusion in the beginning but then we figured it out by discussing it with each other. But we have started working late this time. That why it was a stressful for us to complete it on time.** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

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

1. Why did we wait until the fourth milestone to write the Whitebox tests?

In our project, we waited until the fourth milestone to write Whitebox tests due to several strategic reasons. Initially, we prioritized Blackbox tests to validate the software’s core functionalities and ensure it met the required criteria. Blackbox testing focuses on the system's behavior and user interactions, which are crucial in the early stages to ensure that the software delivers the intended value. By waiting until the fourth milestone, we allowed the codebase to stabilize, minimizing the risk of frequent changes that could lead to rework in our tests. Additionally, we adopted a risk-based testing approach, focusing first on high-risk areas with Blackbox tests to address critical issues early on. After verifying the essential parts, we introduced Whitebox testing to gain deeper insights into the software's internal workings and improve overall code quality. This sequencing ensured that both functional and structural aspects of the software were thoroughly tested. Prioritizing Blackbox testing initially allowed us to achieve broader test coverage and address high-level issues arising from user interactions and external systems. As the project progressed, the iterative approach gradually increased testing complexity, ensuring a comprehensive evaluation of the software. This strategic decision optimized our testing process by ensuring that our tests were effective and timely, enhancing the reliability and robustness of the final product.

1. How does the Agile methodology ensure that all team members are consistently engaged throughout the software development process, avoiding downtime due to dependencies on others? Provide an example to illustrate your point.

Agile methodology ensures that all team members are consistently engaged throughout the software development process by promoting collaboration, flexibility, and iterative progress. One of the core principles of Agile is to break down work into small, manageable tasks, which are then tackled in short iterations or sprints. This approach minimizes downtime due to dependencies and keeps everyone actively involved.

Here's how Agile achieves this:

1. Daily Standups (Scrum Meetings)
2. Cross-Functional Teams
3. Backlog Prioritization
4. Iteration Planning
5. Continuous Integration

**Example:**

In our Milestone 4, we applied Agile principles. Scrum meetings ensured clear communication between team members. Dividing tasks into smaller units and maintaining constant communication minimized downtime. Our cross-functional team, with members participating in both black-box and white-box testing, exemplifies Agile's emphasis on diverse skillsets. We also practiced continuous integration, frequently integrating and testing code changes, allowing for smoother development flow. This group work followed the Agile methodology and by following this we have achieved our group target which helped us to complete all the tasks.

3.What is a shell script and how are we going to utilize a hook script in this project?

**Shell Script**:

A shell script is a computer program written in a scripting language that can be executed by the operating system's command-line interpreter (often called a shell). We typically perform shell scripts manually using terminal commands.

**Hook Scripts in Projects:**

Hook scripts are special scripts used in various software development tools like Git (version control system). These scripts are automatically triggered when specific events occur within the project. Here's how hook scripts can be utilized:

1. **Pre-commit hooks:** Run before a commit is made. They can be used for various purposes, such as:
   1. Linting code to check for stylistic or formatting errors.
   2. Running unit tests to ensure new code doesn't break existing functionality.
   3. Performing code formatting automatically.
2. **Post-commit hooks:** Triggered after a successful commit. They can be used for actions like:
   1. Sending notifications to the team about the commit.
   2. Automatically deploying the code to a testing environment.
   3. Updating documentation based on code changes.

**Example:**

This project, involves developing a delivery app. We could utilize a pre-commit hook script to automatically run a linter that checks our code for style and formatting issues before allowing a commit. This ensures everyone follows consistent coding conventions and improves code readability.

By effectively using shell scripts and hook scripts, we can automate tasks, improve project consistency, and streamline our development workflow.