# **MILESTONE 3** -- 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**: \_11\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Members Present**:

|  |  |
| --- | --- |
| 1. Richa Koirala | 4.Ammar Gangat |
| 2.Akansha | 5. |
| 3.Manjot Singh | 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.

**Rubric**

|  |  |  |
| --- | --- | --- |
| **Individual** | Group participation (includes GitHub commits and Jira usage) | 80% |
| Teamwork | 20% |
| **Group** | Function specifications (documented, complete, well-written, added to the project) | 10% |
| Blackbox test cases document (well-written, complete, good test data) | 15% |
| Blackbox test code (in the C++ project) well-designed and documented | 15% |
| Functions implementation (coded in the C project & well documented) | 10% |
| Requirements traceability matrix (complete and added to GitHub) | 10% |
| Git usage (used properly with good structure) | 10% |
| Jira usage (creates issues, tracks progress) | 10% |
| Scrum report & reflections | 20% |
| **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** |
| **Akansha** | **Git control, Scrum Report, Jira** | **N/A** |
|  |  |  |
| **Richa** | **Git control, Scrum Report, Jira** | **N/A** |
|  |  |  |
| **Manjot** | **Git control, Scrum Report, Jira** | **N/A** |
|  |  |  |
| **Amaar** | **Git control, Scrum Report, Jira** | **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** |  |
| **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 |
| GitHub | **Updating git branches** | **DONE** |
| Test plan | **Test plan done** | **DONE** |
| Jira | **Task schedule setup in Jira** | **DONE** |
| .h file | **Using header file done** | **DONE** |
| Scrum Report | **Discussing task completed, not completed and reflection questions** | **DONE** |
|  |  |  |
|  |  |  |

**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 |
| Testing decision | Using integration test cases, unit test cases, black box, traceability matrix |
|  |  |
| Prioritization of tasks | Equal work to be divided among all the group member |
|  |  |
| Using new header file | Using header file to get desired output |
|  |  |
|  |  |

**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? |
| Akansha | **Black box test, SCRUM report and reflection questions** | **15 mins** | **Yes** |
|  |  |  |  |
| Ammar | **Requirements traceability matrix**  **SCRUM report and reflection questions** | **15 mins** | **Yes** |
|  |  |  |  |
| Richa | **Unit test for black box test (with CPP), SCRUM report and reflection questions** | **15 mins** | **Yes** |
|  |  |  |  |
| Manjot | **Function declaration and description,**  **SCRUM report and reflection questions** | **15 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 |
|  |  |
| Akansha | Looking after GitHub |
|  |  |
| Richa | Manjot will conduct the meeting with tasks for milestone 4 |
|  |  |
| Manjot | Matching the availability and get a time schedule for the meeting in person and the location for the meeting. |
|  |  |
| Ammar | Looking after Jira |
|  |  |
|  |  |
|  | - |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Test plan | **Testing algorithm was set according to the group member** |
|  |  |
| .H file | **Created .h file will help to get the desired output and identify any potential bugs and errors** |
|  |  |
| Attendance | **Everyone attended the meeting** |
|  |  |
|  |  |

**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 |
| SCRUM REPORT | **Everyone contributed to scrum report** |
| GIT | **Discussed more about git** |
| Jira | **Discussed more about Jira** |
|  |  |
|  |  |
|  |  |
|  |  |

**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 |
| N/A | **N/A** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

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

1. In this milestone, we write the blackbox tests but not the whitebox tests. Explain why we can write the blackbox tests but not the whitebox tests.   
   We have two different testing approaches which are used to validate functionality of software systems. They are black box testing and white box testing. Black box testing is an approach of software testing where it focuses on testing the functionality of a software system without examining its internal structure or implementation details. White box testing on the other hand is an approach.

In this milestone, we use Blackbox testing because it does not require knowledge of the internal implementation details of the software system. Testers can design test cases which are according to the system’s requirements, making it possible to start testing. Also, with the use of black box testing it allows for early validation of system’s functionality from an end -user perspective. Another possible reason is that black box testing is often driven by system requirements and specifications. The black box testing can also be used to validate the contract between different components or modules of the system.

Also, the white box testing requires access to the codebase and a deeper understanding of the implementation. It may not be feasible to conduct white box testing without the code.

1. Explain why we need the function-test matrix and why it is important in a large project.  
   A function test matrix, also known as a requirements traceability matrix, is used in software testing to map the relationship between functional requirements and test cases. In function test matrix, each row represents a specific requirement, and each column represents a corresponding test case. Below are some of the reasons why we need function test matrix and its importance in a large project:
2. A traceability matrix is important for a large project which may include large functional requirements challenges to ensure that each requirement is tested. So, the traceability matrix is important to ensure comprehensive coverage of all functional aspects of the systems.
3. The matrix establishes a traceable link between test cases and requirements, also it is crucial for demonstrating compliance with project specifications and regulatory requirements.
4. Other life cycle models left team members idle while waiting for parts of the project to be completed. Describe how an agile model, like the one we are using, avoids this problem and keeps the whole team busy all the time. Does this make managing the project simpler or more complex and why?

Agile model is an iterative and incremental approach to software development that emphasizes flexibility collaboration and customer feedback. Agile breaks down the projects into small, manageable iterations called sprints, typically lasting one to four weeks. Other life cycle models left team members idle while waiting for parts of the project to be completed. Here are some reasons why agile model avoids this problem and keeps the whole team busy:

1. Agile projects use user stories. Which are broken down into smaller tasks during sprint planning sessions.
2. In agile teams are provided with a clear set of goals and tasks defined for each sprint, team members have specific assignments they can work on, keeping everyone busy throughout the sprint duration.
3. In agile daily scrums provide opportunity for team members to discuss progress, challenges, and plans for the day.
4. In agile, individuals with different skill set work together to accomplish goals.