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

**Members Present**:

|  |  |  |
| --- | --- | --- |
| 1.Fahad Bin Abdullah | 4. Sofiia Parkhomenko | 4. |
| 2. Sidharth Variyath | 5. Mihir Bakulbhai Patel | 5. |
| 3. Fahad Bin Adbullah | 6. Jaskaran 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) | 10% |
| Blackbox test code (in the C++ project) well-designed and documented | 10% |
| Functions implementation (coded in the C project & well documented) | 10% |
| 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) | 15% |
| Scrum report & reflections | 15% |
| **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** |
| **Anurag K C** | Milestone 3 planning, function Implementation, review functions and test plan | **N/A** |
| **Jaskaran Singh** | Function Designing and specs documentation in word file. | **N/A** |
| **Mihir Patel and sofiia parkhomenko** | Function Designing and add function description to header file | **N/A** |
| **Sidharth Variyath** | Black box test | **N/A** |
| **Fahad Bin Abdullah** | Scrum Report | **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 discusses in the meeting and the outcomes of the discussions.

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Milestone 3 | Quick overview of the milestone objectives, which includes the scrum report, Functions, and Black-box Test. | All members understood the milestone objective |
| Scrum report | topics to document in scrum report and appropriate answers to reflections | Scrum report completed with help of all members |
| Black-Box Testing | how to write black box test code, how to run it, and what test data to use. The repository was updated with the black box test code. | The team understood test plan and task has been assigned |
| Implementing functions | Who will implement the functions into implementation file and update the repository with newly added content. | Task assigned to Meet |
| Building a function-test matrix | The group created a matrix of functions and tests that displayed the Blackbox tests for each function. As additional tests were added or altered, the matrix was updated. The repository housed the matrix. | The team had a thorough overview of the testing coverage and outcomes for each function. |
|  |  |  |
|  |  |  |

**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 |
| Function Prioritization | The team listed the functions in order of importance and dependencies to make sure that the most crucial ones were implemented first. |
| Testing Strategy | To ensure thorough test coverage and find any potential issues early in the development process, it was decided to combine unit testing, integration testing, and black-box testing. |
| Task Assignment | Each team member was given a specific task to carry out, and a deadline was established for finishing it. Team members were urged to work together and support one another to maintain progress. |
| Issue Tracking | The team established a procedure for using Jira to track issues and bugs. To ensure a smooth development workflow, issues will be logged, assigned to team members, and tracked until resolution. |
|  |  |
|  |  |
|  |  |

**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? |
| Fahad Bin Abdullah | Scrum report | **1 hr** | **yes** |
| Sofiia Parkhomenko and Mihir Patel | Function implementation | **3 hr** | **yes** |
| Sidhart Variyath | Black box testing  - | **2 hr** | **yes** |
|  |  |  |  |
| Jaskaran Singh | Writing function specifications | **2 hr** | **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 |
| Anurag K C | Implement function definitions |
| Sidhart Variyath | Writing Black box and white box test for functions |
| Sidhart Variyath | Execute black box test and document result |
| Mihir patel and sofiia parkhomenko | Execute white box test and document result |
| Everyone | Update repository and complete preparation for test automation |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

-

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Function design | Add value to software by including functionalities |
| Understand algorithm for shortest path | Help to define function and provide effective delivery services. |
| Functions understanding | All team members understand functions so they can work with it and test it accordingly. |
| Potential Issues in projects | Issues such as vehicles becoming caught at the map's edge or in building corners have been identified. Recognizing and fixing these difficulties strengthens the algorithm, lowering the likelihood of delivery failures owing to vehicles being unable to reach locations. |
|  |  |
| Refine Plan | Improve overall plan for more stability and timely deliverables |
|  |  |

**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 |
| Clear understanding | The team had a clear understanding of milestone objectives |
| Effective task division | Tasks were successfully divided among team members |
| Comprehensive function design | Productive discussion on designing required functions |
| Well-defined test strategy | Defined strategy for writing black-box test code |
|  |  |
|  |  |
|  |  |

**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 |
| Updated Source Code | The Main Reason was pc of one of the team member was not functioning well and we have experience abnormal behavior in the machine by at the end we restated that task from beginning and resolved with the help of all the team member. |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**Reflections**:

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

1. What is the difference between blackbox tests cases and blackbox test code? Explain how we use assertion in Visual Studio to execute tests.

Answer:

Blackbox test cases and Blackbox test code are different in how they are used. Blackbox test cases are like written plans that outline what input a function should take and what output it should give, without looking at the actual code. They focus on what each function should do based on requirements, such as handling certain values or errors.

Blackbox test code, however, is the actual coding of these cases in a testing tool like Visual Studio. Here, each test case is programmed to check if the output matches the expected result automatically.

In Visual Studio, we use assertions to check if a function’s output is correct. For example, `ASSERT\_EQ` checks if the actual output equals the expected output. If it doesn’t, it shows there’s a bug in the code. Assertions help us quickly find and fix errors, making sure the functions work as intended.

1. How can a traceability matrix help in the testing process?

Answer: A traceability matrix is a helpful tool in testing because it connects each project requirement to specific test cases. This makes sure that every requirement has a test to check if it works correctly, reducing the chance of missing any important tests.

The matrix also helps track testing progress. As tests are run, the matrix shows which tests pass or fail, giving a clear view of the project’s status. This way, we can easily see if any areas need more attention. Overall, a traceability matrix makes the testing process more organized, thorough, and ensures all project requirements are met before finishing.

1. 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?

Answer:

Two function prototypes we submitted are:

**int findClosestTruck (Map &map, std::vector<Truck> &trucks, Shipment &shipment);**

* + **Purpose**: This function finds the truck closest to the delivery destination that also has enough space for the shipment.
  + **Importance**: It helps to assign shipments efficiently by considering both distance and capacity, making deliveries faster and using resources effectively.

**void updateTruckCapacity (Truck &truck, int weight, double volume);**

* + **Purpose**: This function updates a truck’s weight and volume after a shipment is loaded.
  + **Importance**: It ensures trucks don’t go over their weight or volume limits, which is important for safety and planning future shipments.

Both functions support the project’s goals by managing shipments efficiently and ensuring safe operations, which are essential for the delivery system’s success.