# **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**: 5

**Members Present**:

|  |  |
| --- | --- |
| 1.Jubin Verma | 4.Photswat Boonme |
| 2.Sandhya Timsina | 5.Udhav Tamyal |
| 3.Susinta Bastola | 6. Parshv Nileshbhai Gandhi |

## 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** |
| **Jubin verma** | **Scrum report, Jira management, overall helping the group to complete their tasks, Making unit test project and linking them.** | **No delays or blocks** |
| **Sandhya Timsina** | **Black Box test cases (in unit tests)** | **No delays or blocks** |
| **Susinta Bastola** | **Traceability matrix** | **No delays or blocks** |
| **Photswat Boonme** | **Reflection questions, Function specifications.** | **No delays or blocks** |
| **Udhav Tamyal** | **Code (function implementation)** | **No delays or blocks** |
| **Parshv Nileshbhai Gandhi** | **Black box testing (in excel file)** | **No delays or blocks** |

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 |
| Task assignment and distribution | **Who wants to and is assigned which tasked** | **Assignment of tasks effectively with regards to each members strengths and weaknesses** |
| Traceability matrix | **What is traceability matrix and how to do it** | **Well, understanding of traceability matrix concept and further assignment of the task to a member** |
| Unit test for ms3 | **Unit test for ms3 with reference to videos of workshop 3** | **Assignment of unit testing part to a member** |
| Black box test cases for ms3 | **Who is to do ms3 black box test cases and the excel file format of it** | **Well understanding of the excel file format and the number of tasks to be done with it** |
| Test plan ms3 | **Understanding the given test plan template and further the requirements** | **Well, understanding of test plan and who is to do it** |
| Scrum report for ms 3 | **Understanding the given scrum report template and further the requirements** | **Assignment of whom to do the scrum report** |
| Ms 4 | **Discussing what is required in ms4** | **Better understanding of ms4** |
| Code of ms 3 | **Understanding and discussing what is needed to be made in ms 3 according to the requirements and further according with future milestones** | **Better understanding of milestones and its code** |
| Reflection questions | **What the reflections questions are and who is to do it** | **Better understanding of project entirely** |
| Task assignment and distribution | **Who wants to and is assigned which tasked** | **Assignment of tasks effectively with regards to each members strengths and weaknesses** |
| Function Specifications | **What are function specifications and what to include in them** | **Completion of Function specification task** |

**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 |
| Code implementation and need of additional functions and prototypes. | There are some concepts that come to mind only when u encounter a problem while coding and to solve that problem we might need extra function to which we decided to add more functions than the predefined ones made in ms2 |
| Task assignment and decision | Tasks were assigned according to their strengths and weaknesses |
| Code to be done first | Code Is the building block of every other task to be done so we assigned the coding task to be done first |
| Reflection to be done last | As reflection requires a general knowledge of the code, the test, unit testing, traceability matrix and the overall gist and knowledge of the project it should done last . |
|  |  |
|  |  |
|  |  |

**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? |
| Jubin verma | **Scrum report, Jira management, Overall help of the entire group in finishing their work, Unit Test Project creation and linking** | **4hr** | **Completed** |
| Sandhya Timsina | **Unit test function implementation** | **2hr** | **Completed** |
| Susinta Bastola | **Traceability Matrix** | **2hr** | **Completed** |
| Photswat Boonme | **Reflection questions, Function Specifications.** | **1hr** | **Completed** |
| Udhav Tamyal | **Code (function implementation)** | **6hr** | **Completed** |
| Parshv Nileshbhai Gandhi | **Black Box test cases** | **1hr** | **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 |
| Jubin verma | Checking everyone’s work, scrum report creation, Debugging, Hookfiles |
| Sandhya Timsina | White box test code bringing snacks for next meeting. |
| Susinta Bastola | Updated traceability matrix |
| Photswat Boonme | Reflection questions |
| Udhav Tamyal | Coding (function implementation) |
| Parshv Nileshbhai Gandhi | White box test cases |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Better understanding of MS\_3 Coding | **Faster and more efficient Completion of project(Milestone3)** |
| Better understanding of project | **Faster and more efficient Completion of project(Milestone3)** |
| Better understanding of Tortoise Git | **Faster and more efficient Completion of project(Milestone3)** |
| Better understanding of MS\_3 Testing (unit testing) | **Faster and more efficient Completion of project(Milestone3)** |
| Traceability Matrix | **Faster and more efficient Completion of project(Milestone3)** |
|  |  |
|  |  |

**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 |
| Code | **Good communication, Teamwork, overall great team synergy.** |
| Test cases and unit testing | **Good communication, Teamwork, overall great team synergy.** |
| Reflection questions | **Good communication, Teamwork, overall great team synergy.** |
| Assignment of tasks | **Good communication, Teamwork, overall great team synergy.** |
| Traceability matrix | **Good communication, Teamwork, overall great team synergy.** |
|  |  |
|  |  |

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

In software testing, blackbox test cases and blackbox test code serve different purposes. Plans referred to as blackbox test cases outline which program characteristics should be examined, with a focus on how the program should function without revealing its internal workings. By specifying exact inputs and expected outcomes, these examples ensure that the software meets its requirements. Blackbox test code is the actual code written to run these test scenarios automatically. It interacts with the program by providing input and verifying that the outcomes live up to the expectations. Assertions are used in Visual Studio testing to perform these checks. If an assertion fails, the test is stopped and an issue is signalled, allowing testers to find and fix errors more quickly. Together, blackbox test scenarios and test code ensure that the program functions correctly from the user's perspective without necessitating an analysis of its internal workings.

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

In C and C++ software testing, a traceability matrix helps keep track of how requirements link to specific test cases, making the testing process more organized and thorough. By using a matrix, we can see if every requirement has a corresponding test case, which reduces the chance of leaving out important features. When changes happen in the code, especially in complex C and C++ projects, the matrix makes it easier to spot which tests need updates, saving time by focusing only on affected areas. By enabling us to track problems back to the original requirement, it also aids in bug discovery by illuminating the point at which something went wrong. The matrix can be used as evidence that we evaluated all required characteristics, which is helpful for audits if we operate in sectors with high requirements. All things considered, a traceability matrix facilitates the management of C and C++ testing, enabling us to produce software that is dependable.

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?

int addShipmentToBestTruck(struct Truck\* trucks, const struct Map\* map, const struct Shipment\* shipment, const int numOfTrucks)

{

if (!trucks || !map || !shipment || numOfTrucks < 1)

return -1;

int bestTruckIndex = findTruckForShipment(trucks, map, shipment, numOfTrucks);

if (bestTruckIndex >= 0)

addShipmentToTruck(&trucks[bestTruckIndex], shipment);

return bestTruckIndex;

}

int findTruckForShipment(const struct Truck\* trucks, const struct Map\* map, const struct Shipment\* shipment, const int numOfTrucks)

{

if (!trucks || !map || !shipment || numOfTrucks < 1)

return -1;

struct Point orgin = { 0, 0 };

struct Route route = getPossibleMoves(map, shipment->destination, orgin);

if (!route.numPoints)

return -1;

if (shipment->destination.row > map->numRows || shipment->destination.col > map->numCols)

return -1;

int bestTruckIndex = -1;

for (int i = 0; i < numOfTrucks; i++) {

if (!hasTruckRoomForShipment(&trucks[i], shipment))

continue;

if (bestTruckIndex < 0)

bestTruckIndex = i;

int best = compareTrucksForShipment(&trucks[i], &trucks[bestTruckIndex], map, shipment);

if (best > 0)

bestTruckIndex = i;

}

return bestTruckIndex;

}

The function addShipmentToBestTruck finds and assigns the most suitable truck from a fleet to carry a given shipment. It first calls findTruckForShipment, which evaluates all available trucks based on criteria like capacity and proximity to the destination. If a truck is identified as suitable, addShipmentToTruck is used to load the shipment onto it. This function is essential to efficiently allocate shipments to trucks, preventing resource waste and ensuring timely deliveries.

findTruckForShipment is pivotal because it systematically evaluates each truck's ability to carry the shipment. It relies on helper functions such as hasTruckRoomForShipment, which confirms if a truck has enough space and weight capacity, and compareTrucksForShipment, which identifies the best candidate by comparing trucks based on their routes and limiting factors. By integrating functions like getPossibleMoves and shortestPath, findTruckForShipment considers both distance and truck limitations, selecting the best truck for the shipment. This approach optimizes resource allocation, supports efficient delivery, and helps the project meet its logistics requirements.