# **Milestone 5 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**: **Group C**

**Members Present**:

|  |  |
| --- | --- |
| 1. Chia-Ming Cheng | 4. Peter Bryson |
| 2. Md Arafat Koyes | 5. |
| 3. Md Asif Karim | 6. |

## Milestone 5 Tasks

In this milestone, you should write, implement, and execute integration tests. Integration tests test how multiple functions work together to complete a task. Depending on what is being tested, you might be able to write unit tests to do the testing and automatically compare the results. In other cases, you might need to manually check the output to check it. This will all be stated in the tests where it discusses how they should be run.

As you update the function-test matrix, you will need to add a very brief description for each integration test so the matrix will clearly show what the tests are testing. Acceptance tests will be tested against actual user requirements and will list all the tests for each requirement.

Acceptance tests are the final tests and are largely aimed at showing the customer that the correct output is produced for different inputs. This will largely require manual testing.

**Deliverables due 11 days after your lab day:**

* Integration tests document (for the new functions you added) stored in repository with at least 4 sets of distinct test cases (each case must have at least 4 distinct test data).
* Integration tests coded (store in repo), executed (results in Jira and in test documents) and debugged.
* Finish implementing/coding whitebox tests. Store in repo, executed, results in Jira (and on corresponding test documents, and debugged.
* One acceptance test case for each requirement added to the test cases excel sheet.
* All acceptance tests implemented and added to the testing C++ project.
* Updated requirements traceability matrix 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** | Integration test case document (well written, complete, good test data) | 10% |
| Integration test code (well designed and documented) | 10% |
| Finish coding all functions and main (well-designed, written, and documented) | 10% |
| Finish coding blackbox and whitebox cases (well-designed, written, and documented) | 5% |
| Acceptance tests (well-designed, documented, and implemented) | 15% |
| Requirements traceability matrix updated | 5% |
| Test execution (performed, results recorded, issues created) | 5% |
| Debugging (bugs fixed, documented, Jira updated) | 5% |
| Git usage (used properly with good structure) | 5% |
| 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** |
| Chia-Ming Cheng | Group meeting invitation and assign tasks, updated GitHub and Jira, wrote white box test cases, wrote scrum report, proofread, and submission |  |
| Md Arafat Koyes | Wrote white box test code |  |
| Md Asif Karim | Wrote black box test code |  |
| Peter Bryson | Wrote function code |  |

**Summary of Meeting:**

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

|  |  |  |
| --- | --- | --- |
| Topic | Discussion Summary | Outcome |
| Task arrangement | Every member knows their job | 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 |
| Assign appropriate tasks to each member | To ensure all team members can understand the project |
|  |  |
|  |  |

**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? |
| Chia-Ming Cheng | MS 3 Group meeting invitation and assign tasks | 30 mins | Yes |
| Chia-Ming Cheng | MS 3 Jira task and GitHub update | 30 mins | Yes |
| Chia-Ming Cheng | Update requirements traceability matrix | 30 mins | Yes |
| Chia-Ming Cheng | Writing integration test document | 1 hour | Yes |
| Chia-Ming Cheng | Writing scrum report | 30 mins | Yes |
| Chia-Ming Cheng | MS 3 Proofread, review and submission | 1 hour | Yes |
| Md Arafat Koyes | Writing white box test code | 2 hours | Yes |
| Md Arafat Koyes | Executed white box test code, write results in corresponding test documents, and debugged | 1 hour | Yes |
| Md Asif Karim | Completed hook file | 30 mins | Yes |
| Md Asif Karim | Executed black box test code, write results in corresponding test documents, and debugged. | 1 hour | Yes |
| Peter Bryson | Integration tests code, executed and debugged | 3 hours | Yes |
| Peter Bryson | All acceptance tests implemented and added to the testing C++ project | 1 hours | 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 |
| Chia-Ming Cheng | Group meeting invitation and assign tasks, update Jira project and GitHub, write scrum report, proofread and review everything, submission |
| Md Arafat Koyes | Execute acceptance tests (results in Jira), and debug |
| Md Asif Karim | Final testing report listing tests conducted, bugs fixed, and the final tests passed |
| Peter Bryson | Updated requirements traceability matrix stored in the repository |
|  |  |
|  |  |

**Major Outcomes of Meeting:**

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

|  |  |
| --- | --- |
| Outcome | Impact on Project |
| Every member knows their job | The project is progressing perfectly |
|  |  |
|  |  |

**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 |
| Task arrangement | Every member is good at communicate |
|  |  |
|  |  |

**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 |  |
|  |  |
|  |  |

**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 manual and automated testing? Why are we automating the testing process and what benefits does automation offer?

A:

Manual Testing:

Pros:

1. Useful for exploratory, ad-hoc, and usability testing.

2. Can provide subjective feedback on user experience.

3. Flexible and can easily adapt to changes in the application.

Cons:

1.Time-consuming and labor-intensive.

2. Prone to human error and inconsistencies.

3. Not ideal for repetitive tasks or large test suites.

Automated Testing:

Pros:

1. Increases test efficiency and coverage.

2. Reduces manual effort and repetitive testing.

3. Allows for continuous testing and integration (e.g., in CI/CD pipelines).

4. Provides consistent and reliable results.

Cons:

1.Initial setup can be time-consuming and costly.

2. Requires maintenance of test scripts and tools.

3. Less effective for exploratory testing and subjective assessment.

Automate Testing and Benefits:

1. Efficiency: Automated tests run faster than manual tests, especially for large test suites.

2. Reusability: Automated test scripts can be reused across different versions of the application.

3. Consistency: Automation ensures that tests are executed the same way every time, reducing variability in results.

4. Scalability: Automated testing can easily handle large volumes of tests and complex scenarios.

5. Continuous Integration/Continuous Deployment (CI/CD): Automation supports frequent testing and integration, enabling rapid feedback and faster development cycles.

1. Why it is necessary to write integration tests given that the code has already passed blackbox and whitebox tests?

A:

1. Component Interaction: Blackbox and white box tests often focus on individual components or functionalities. Integration tests are necessary to verify that these components work together correctly, handling data flow and interactions as expected.

2. End-to-End Scenarios: Integration tests simulate real-world scenarios where multiple components interact. This can reveal issues related to data exchange, interface compatibility, and overall system behavior.

3. Complex Dependencies: Modern applications often rely on third-party services, databases, and other external systems. Integration testing helps ensure that these dependencies are correctly integrated and function properly in the context of the overall system.

4. Uncovered Edge Cases: Integration tests can expose issues related to how components handle unexpected inputs or error conditions that might not be covered in unit or functional tests.

5. System-Level Assurance: While black box and white box tests provide assurance about individual aspects of the system, integration tests offer a broader view, confirming that the system as a whole operates seamlessly.

1. List and describe one of the integration tests you created. Provide a thorough explanation of how the integration operates, detailing the flow of parameters from one function to another. Use one of your integration tests to support your answer.

A:

FindTruckForShipmentWithAddedTruckAndDelivery

This test integrates several functions to verify that the system correctly assigns a truck to a delivery based on truck capacities and delivery requirements.

Objective: To verify that the findTruckForShipment function correctly identifies the best truck for a given delivery after adding a truck and a delivery to the system.

In this integration test, the flow of parameters involves adding a truck and a delivery to their respective lists, then invoking findTruckForShipment to determine if the system can correctly assign the truck to the delivery. The test validates that the truck with sufficient capacity and matching ID is selected for the delivery, ensuring that the integration between the addTruck, addDelivery, and findTruckForShipment functions works as intended.